DISPOSITIVO LoRa DE COMUNICACIÓN A LARGO ALCANCE Y BAJO CONSUMO ENERGÉTICO PARA APLICACIONES DEL ÁMBITO DEL DESARROLLO

JOSÉ DANIEL RODRÍGUEZ MUNCA

Asignatura

Trabajo fin de Master (técnico)

Tutor Académico: **Manuel Sierra Castañer**

Tutor Profesional: **Álvaro Gutiérrez Martín**

GEDIRCI - GENERACIÓN DISTRIBUIDA RENOVABLE Y CONTROL INTELIGENTE - INSTITUTO DE ENERGÍA SOLAR

UNIVERSIDAD DE POLITÉCNICA DE MADRID
UNIVERSIDAD COMPLUTENSE DE MADRID
MASTER INTERUNIVERSITARIO EN ESTRATEGIAS Y TECNOLOGÍAS PARA EL DESARROLLO
MADRID - ESPAÑA
2016

AGRADECIMIENTOS

Agradezco a la *Fundación Carolina* por brindarme la valiosa oportunidad de haber sido beneficiario del programa de becas para el master interuniversitario en Estrategias y Tecnologías para el Desarrollo durante la versión 2015-2016, permitiéndome afianzar nuevos conocimientos y mejorar mi perfil profesional. Agradezco a los docentes Álvaro Gutiérrez Martín y Manuel Sierra Castañer por compartir sus valiosos conocimientos técnicos en su calidad de tutores profesional y académico, además del respaldo incondicional en el desarrollo de mis prácticas profesionales en la ETSIT-UPM; a los maestros de la Universidad Politécnica de Madrid y Universidad Complutense de Madrid quienes que me impartieron sus conocimientos y experiencias, y a todas las personas que de una u otra manera hicieron posible la realización y culminación de esta maestría.

Considero que soy el fruto de las buenas intenciones de las personas.
Muchas gracias.

José Daniel Rodríguez Munca

TABLA DE CONTENIDO

IDENTIFICACIÓN DEL TRABAJO FIN DE MASTER 9
Título del proyecto 9
Facultad y programa en los que está inscrito el proyecto 9
Grupo y línea de investigación 9
Temática de estudio 9
Tutores 9
Estudiante investigador 9
INTRODUCCIÓN 10
1.1 RESUMEN 12
1.2 ABSTRACT 13
2. OBJETIVOS 14
2.1 Objetivo General 14
2.2 Objetivos específicos 14
3. MÉTODO Y FASES DEL TRABAJO 15
3.1 Descripción de los dispositivos utilizados 15
3.1.1 Transceiver Semtech LoRa 15
3.1.2 Sistemas embebidos - importancia del hardware libre 21
3.2 Descripción del software utilizado- Importancia del software libre 30
3.2.1 IDE Arduino 31
3.2.2 IDE Eclipse – Sistema operativo FreeRTOS 32
3.3 Metodología de trabajo 34
3.3.1 Enfoque de la investigación 34
3.3.2 Tipo de investigación 34
3.3.3 Método de investigación 34
3.3.4 Población y muestra 35
3.3.5 Metodología 36
4. RESULTADOS 52
4.1 Pruebas de alcance y consumo energético 52
4.1.1 Pruebas de alcance para LoRa SX1272 53
4.1.2 Pruebas de consumo energético y alcance para LoRa SX1278 56

4.1.3 Pruebas de dispositivos LoRa con diseño de red tipo estrella 69
CONCLUSIONES 70
BIBLIOGRAFÍA 83

LISTADO DE FIGURAS

Figura 1. Disposición de elementos necesarios para la creación del driver con LoRa 15
Figura 2.Diagrama de bloques con principales características y diferencias entre SX1272 y SX1278. 16
Figura 3. Principales características de comunicación SPI para los dispositivos LoRa seleccionados 19
Figura 4. Tarjeta SX1272RF1 de Semtech (868Mhz) 21
Figura 5.Tarjeta LoRa1278 de NiceRF (433Mhz) 21
Figura 6. Placa Arduino UNO 23
Figura 7. Arquitectura simplificada de Arduino UNO R3 24
Figura 8. Placa Arduino DUE 25
Figura 9. Arquitectura simplificada de Arduino DUE. 25
Figura 10. Módulo SDIN Therm 26
Figura 11. Arquitectura de monitoreo para una vivienda autosostenible. 27
Figura 12. Arquitectura a nivel de hardware de los módulos SDIN. 28
Figura 13. Tarjeta principal del módulo SDIN. 29
Figura 14. Tarjeta secundaria del módulo SDIN tipo 0: Termopares. 29
Figura 15. Conexión mecánica de los módulos SDIN entre la tarjeta primaria y secundaria 29
Figura 16. Interfaz del IDE Arduino 31
Figura 17. Interfaz del IDE Eclipse 33
Figura 18. Árbol de decisión dicotómica de los métodos de investigación experimental. 35
Figura 19. Patrón lineal de la investigación cuantitativa. 36
Figura 20. Esquema propuesto para pruebas del driver con los dispositivos LoRa. 39
Figura 21. Esquema de conexión para SX1272 de Semtech y Arduino UNO 40
Figura 22. Esquema de conexión para LoRa1278 de NiceRF y Arduino UNO 41
Figura 23. Esquema de conexión para LoRa1278 de NiceRF y Arduino DUE 42
Figura 24. Esquema de conexión para LoRa1278 de NiceRF y módulo SDIN 43
Figura 25. Espacio de pruebas Indoor para los dispositivos LoRa 45
Figura 26. Espacio de pruebas Outdoor para los dispositivos LoRa 46
Figura 27. Tipología estrella para la prueba del driver generado 47
Figura 28. Principio de funcionamiento "ping-pong" para la realización de pruebas de alcance en dispositivos LoRa 52
Figura 29. Máximo alcance en trasmisión/recepción de datos para LoRa SX1272 (plano XY) 54
Figura 30. Esquema para determinar el alcance en trasmisión/recepción de datos para LoRa SX1272 (plano XY y YZ) 55
Figura 31. Dispositivo esclavo conformado por sistema embebido Arduino Uno y transceiver SX1272 56
Figura 32. Representación gráfica del consumo energético de los transceivers DRF1278F y LoRa1278 para modo recepción 59

Figura 33. Representación gráfica del consumo energético de los transceivers DRF1278F y LoRa1278 para modo transmisión 60
Figura 34. Esquema propuesto para pruebas de alcance Indoor 61
Figura 35. Máximo alcance en trasmisión/recepción de datos para LoRa SX1278 (plano XY). 62
Figura 36. Esquema para determinar el alcance en trasmisión/recepción de datos para LoRa SX1278 (plano XY y YZ) 63
Figura 37. Dispositivo maestro conformado por sistema embebido SDIN y transceiver SX1278 en el laboratorio B-301 de la ETSIT 64
Figura 38. Dispositivo esclavo conformado por sistema embebido Arduino Uno y transceiver SX1278 en sótano del edificio A de la ETSIT 64
Figura 39. Esquema propuesto para pruebas de alcance "Outdoor" 65
Figura 40. Dispositivo maestro conformado por sistema embebido Arduino Uno, transceiver SX1278 y batería de 5V en la terraza del edificio C de la ETSIT 66
Figura 41. Dispositivo esclavo conformado por sistema embebido Arduino Uno, transceiver SX1278 en el parque Dehesa de la Villa 66
Figura 42. Máximo alcance en espacio "outdoor" utilizando transceivers SX1278 67
Figura 43. Perfil de elevación entre la terraza del edificio C y el punto de máximo alcance en el parque Dehesa de la Villa 68
Figura 44. Esquema para determinar el alcance en trasmisión/recepción de datos para LoRa SX1278 (plano YZ) 68
Figura 45. Vínculo entre las tecnologías y el desarrollo humano 70
Figura 46. IoT: la red de redes. 71
Figura 47. Comparación del ancho de banda requerido (frecuencia) con respecto al rango de cobertura para los principales estándares inalámbricos. 72
Figura 48. Comparación de los niveles de voltaje entre Arduino Uno y los transceiver LoRa. 76
Figura 49. Esquema de conexión para visualizar el nivel de alimentación con respecto a los modos trasmisión y recepción en los dispositivos LoRa. 76
Figura 50. Diagrama de tiempos de la señal de alimentación con respecto a los modos trasmisión y recepción en los dispositivos LoRa utilizando Arduino Uno. 77
Figura 51. Diagrama de tiempos de la señal de alimentación con respecto a los modos trasmisión y recepción en los dispositivos LoRa utilizando Arduino DUE. 78

LISTADO DE TABLAS

Tabla 1. Relación de frecuencias de trabajo para el SX1272 y SX1278 17
Tabla 2. Banda de frecuencia ISM compatibles con los transceivers de radio propuestos 17
Tabla 3. Principales características de los dispositivos LoRa seleccionados 18
Tabla 4. Principales características eléctricas de los dispositivos LoRa seleccionados 18
Tabla 5. Comparación de pines de control según los fabricantes de tarjetas para los transceivers estudiados 20
Tabla 6. Descripción del uso de los pines RXEN y TXEN para la tarjeta LoRa1278 de NiceRF 20
Tabla 7. Nomenclatura utilizada para la conexión entre la tarjeta SX1272RF1 de Semtech y la tarjeta Arduino UNO 41
Tabla 8. Nomenclatura utilizada para la conexión entre la tarjeta LoRa1278 de NiceRF y la tarjeta Arduino UNO 42
Tabla 9. Nomenclatura utilizada para la conexión entre la tarjeta LoRa1278 de NiceRF y la tarjeta Arduino DUE 43
Tabla 10. Nomenclatura utilizada para la conexión entre la tarjeta LoRa1278 de NiceRF y el módulo SDIN 44
Tabla 11. Instrumento propuesto para la recopilación de datos en las pruebas con dispositivos LoRa para diferentes configuraciones 48
Tabla 12. Combinación de los principales parámetros configurables para los dispositivos LoRa 49
Tabla 13. Parámetros de configuración por software en los dispositivos SX1272 para obtener máximo alcance 53
Tabla 14. Instrumento con la recopilación de datos para las pruebas con los dispositivos LoRa para diferentes configuraciones en DRF1278F de Dorji 57
Tabla 15. Instrumento con la recopilación de datos para los dispositivos LoRa a diferentes configuraciones en LoRa1278 de NiceRF 58
Tabla 16. Parámetros de configuración por software en los dispositivos SX1278 para obtener máximo alcance 60
Tabla 17. Cuadro comparativo de tecnologías inalámbricas LoRa, ZigBee, Bluetooth y Wi-Fi. 74
Tabla 18. Tarjetas embebidas STMicroelectronics de ultra bajo consumo 78

ANEXOS

Anexo A. Desarrollo del driver para el uso de los dispositivos de radio LoRa con el uso de sistemas embebidos.

Anexo B. Instalación y uso del driver LoRa para sistemas embebidos.

Anexo C. DataSheet LoRa SX1278.

Anexo D. DataSheet LoRa SX1272.

IDENTIFICACIÓN DEL TRABAJO FIN DE MASTER

Título del proyecto

DISPOSITIVO LoRa DE COMUNICACIÓN A LARGO ALCANCE Y BAJO CONSUMO ENERGÉTICO PARA APLICACIONES DEL ÁMBITO DEL DESARROLLO

Facultad y programa en los que está inscrito el proyecto

Escuela Técnica Superior de Ingenieros de Telecomunicación ETSIT.
Departamento de Tecnología Fotónica y Bioingeniería.
Universidad Politécnica de Madrid.

Programa de Maestría Interuniversitaria en Estrategias y Tecnologías para el Desarrollo.
Universidad Complutense de Madrid.
Universidad de Politécnica de Madrid.

Grupo y línea de investigación

Grupo de investigación: Generación Distribuida Renovable y Control Inteligente (GEDIRCI).
Escuela Técnica Superior de Ingenieros de Telecomunicación ETSIT.

Línea de investigación: Sistemas de control inteligente y bioinspirado.

Temática de estudio

TIC para el Desarrollo.

Tutores

Tutor Académico: Manuel Sierra Castañer
Departamento de Señales, Sistemas y Radiocomunicaciones.
Escuela Técnica Superior de Ingenieros de Telecomunicación ETSIT.

Tutor Profesional: Álvaro Gutiérrez Martín
Departamento de Tecnología Fotónica y Bioingeniería.
Escuela Técnica Superior de Ingenieros de Telecomunicación ETSIT.

Estudiante investigador

José Daniel Rodríguez Munca.
Ingeniero Electrónico. Especialista en Docencia mediada por TIC. Candidato a magíster en Ciencias de la Educación.

INTRODUCCIÓN

Comúnmente se asocia la comunicación con los medios masivos de difusión de información. Este término también involucra el diálogo e intercambio de ideas y conocimientos entre individuos, ya sea de forma verbal o utilizando medios electrónicos que facilitan estas tareas, recortan distancias y brindan precisión a la información. Esto permite la toma asertiva de decisiones en favor de una comunidad, un negocio, familia o individuo.

Según la FAO (1994), la comunicación es decisiva en la promoción del desarrollo humano aumentando en las comunidades la sensibilización, participación, capacidad, conocimientos y tecnologías. Sin embargo, estas últimas suelen estar subutilizadas. Además, "*Los programas de desarrollo sólo podrán dar todos sus frutos si los conocimientos y tecnologías se comparten efectivamente*".
También enfatiza que el principal denominador en cuestiones relacionadas con el desarrollo es el factor humano y que se depende más de la población, del nivel de sensibilización, de la participación y las competencias. El segundo denominador es la comunicación que puede influir en cambios económicos, sociales y culturales ofreciendo mejores posibilidades y oportunidades al desarrollo.

A un nivel práctico, el termino *comunicación* con el tiempo fue absorbido y generalizado con lo que hoy se le conoce como TIC o Tecnologías de la Información y las Comunicación; encontrando similitudes en su uso y aplicación: *"…las TIC son un instrumento cada vez más poderoso para poder participar en los mercados mundiales; promover la responsabilidad política; mejorar la provisión de servicios básicos, y realizar las oportunidades de desarrollo local. Pero sin políticas innovadoras de TIC, muchas personas de los países en desarrollo –especialmente los pobres– se quedarán atrasados*" (PNUD, 2003).

Con el auge de las Tecnologías de la Información y las Comunicaciones en el siglo XXI, aparecen nuevas tendencias como lo es el Big Data y el Internet de las cosas (IoT o IdC). Internet no solo ha permitido conectar personas, sino dispositivos, sensores y complejos sistemas. Es así como aparece el término "Internet de las cosas" o "internet de los objetos" que corresponde a la interconexión de diversos dispositivos a internet, los cuales generan datos en tiempo real.

Lo importante del "internet de las cosas" no radica en la conexión de las "cosas" a internet, sino en la información y conocimiento que estas puedan aportar para mejorar la calidad de las personas.

Con la posibilidad de obtener información valiosa de diferentes elementos a través de internet, aparecen nuevas técnicas y protocolos de comunicación a bajo consumo; es el caso de *LoRaWAN*, un protocolo de comunicación inalámbrica de bajo consumo y largo alcance para IoT.

Sin embargo:
"La tecnología se crea en respuesta a las presiones del mercado y no de las necesidades de los pobres, que tienen escaso poder de compra… en la era de las redes, cada país necesita contar con capacidad para comprender las tecnologías mundiales y adaptarlas a las necesidades locales", PNUD (2001).

Los dispositivos LoRa son un claro ejemplo de lo planteado por el PNUD en 2001; por tal razón se propone adaptarlos a confiables sistemas de monitoreo y a su vez a posibles aplicaciones del ámbito del desarrollo.

1.1 RESUMEN

Teniendo en cuenta los avances en innovación tecnológica, se han desarrollado estándares inalámbricos abiertos de bajo consumo energético, alto radio de cobertura y bajo costo como respuesta a las tendencias y necesidades del mercado del *IoT* como lo es el estándar *LoRAWAN*. Sin embargo, es necesario adaptar su uso para aplicaciones del ámbito del desarrollo debido a que el uso de la tecnología apropiada puede ser la clave en el aseguramiento de los servicios básicos que sustentan redes de infraestructuras como agua potable, comunicaciones, electricidad, saneamiento, entre otros; mejorando la calidad de la vida de las personas y en especial las que se encuentran más aisladas de la población. Sin embargo, esta tecnología debe ser ajustada al contexto, características y población beneficiada:
"En la era de las redes, cada país necesita contar con capacidad para comprender las tecnologías mundiales y adaptarlas a las necesidades locales", PNUD (2001).

Es así que se planteó el desarrollo de un driver para los dispositivos LoRa SX127X junto con una serie de pruebas para determinar el consumo energético y distancia acorde con condiciones reales como *espacios Indoor* (obstáculos como infraestructuras y muros) y *espacio Outdoor* (obstáculos como vegetación) junto con características y particulares de los dispositivos LoRa plasmadas en este documento con ejemplos de sistemas de monitoreo con sensores remotos.

Durante el desarrollo de la librería o driver para el uso y control de los dispositivos LoRa propuestos, se utilizó *Arduino* UNO/DUE (tarjeta de adquisición de datos a 8/32 bits, catalogada como hardware libre) y *SDIN* (plataforma de adquisición de datos con microcontrolador ARM de 32 bits desarrollada por el grupo de investigación GEDIRCI) mediante el uso de herramientas de software libre como el IDE *eclipse*. En el proceso fueron generadas pruebas de alcance, consumo energético y se documentó la librería respecto al uso y control de los dispositivos.
De forma transversal, se plantearon posibles aplicaciones de la tecnología explorada en aplicaciones del contexto del desarrollo (agua, saneamiento, energía, salud, entre otros), con relación al Big Data, Internet de las cosas (*IoT*) y los principios del desarrollo digital.

1.2 ABSTRACT

Taking into account the advances in technological innovation, open wireless standards of low energy consumption, high radio coverage and low cost in response to the trends and needs of the market of the IoT have been developed as it's the standard LoRAWAN. However, it's necessary adapt its use for development applications since the use of appropriate technology can be the key in securing basic services that support infrastructure networks such as potable water, communications, electricity, sanitation, among others; improving the quality of life of people and especially those found most isolated population.. However, this technology must be adjusted to the context, characteristics and population benefited:
"In the networks age, every country needs the capacity to understand and adapt global technologies for local needs" UNDP (2001).

The development of a driver for the LoRa SX127X devices together with a series of tests to determine the energy consumption and distance according to real conditions like Indoor spaces (obstacles like infrastructures and walls) and Outdoor space (obstacles like vegetation) Together with features and particulars of the LoRa devices embodied herein with examples of monitoring systems with remote sensors.

During the development of the library or driver for the use and control of the proposed LoRa devices, it used Arduino UNO/DUE (8/32 bit data acquisition card, cataloged as open hardware) and SDIN (platform for data acquisition with 32-bit ARM microcontroller developed by the GEDIRCI research group) by using open software tools such as the eclipse IDE. In the process were generated scope tests, energy consumption and the library was documented regarding the use and control of the devices.
In a transversal way, possible applications of the technology explored in applications of the development context (water, sanitation, energy, health, among others), with respect to Big Data, Internet of Things (IoT) and the digital for digital development.

2. OBJETIVOS

2.1 Objetivo General

Evaluar y documentar las características de los transceiver LoRa como dispositivo de comunicación de largo alcance y bajo consumo energético para aplicaciones del ámbito del desarrollo.

2.2 Objetivos específicos

- Plantear posibles aplicaciones de la tecnología explorada en aplicaciones del contexto del desarrollo (agua, saneamiento, energía, entre otros).
- Evaluar las características de consumo energético y alcance máximo para espacios indoor/outdoor de los dispositivos de radio LoRa SX1272 y SX1278 a partir de diferentes configuraciones de firmware.
- Generar y documentar una librería o driver estándar para los dispositivos de radio LoRa SX1272 y SX1278 con el uso de hardware libre y microcontroladores ARM de 32 bits, al igual que utilizando herramientas de software libre.
- Evaluar la implementación de sistemas básicos de monitoreo con el uso de los dispositivos LoRa SX1272/SX1278 con sistemas embebidos Arduino y módulos SDIN.
- Comparar el estándar inalámbrico abierto LoRaWAN con respecto a tecnologías inalámbricas utilizadas para la trasmisión de datos.

3. MÉTODO Y FASES DEL TRABAJO

3.1 Descripción de los dispositivos utilizados

Se propone el uso de dispositivos transceivers con la tecnología LoRa (fabricante *Semtech*) con las versiones SX1272 y SX1278. Un transceiver es un dispositivo que cuenta con circuitos electrónicos capaces de procesar información que podrá ser transmitida o recibida sobre el mismo.

Sin embargo, el transceiver requiere de la lógica de programación para ser configurado. Es así como se debe incorporar un sistema embebido. Un sistema embebido se define como un circuito electrónico o hardware diseñado para realizar un conjunto limitado de funciones específicas (Herrera, 2011); en este caso debe controlar los dispositivos LoRa. Como sistemas embedidos se propone el uso de *SDIN* (plataforma desarrollada por GEDIRCI) y *Arduino* con la versión UNO y DUE (hardware libre).

La *Figura 1* presenta la disposición de los elementos necesarios para realizar y probar el driver para los dispositivos LoRa.

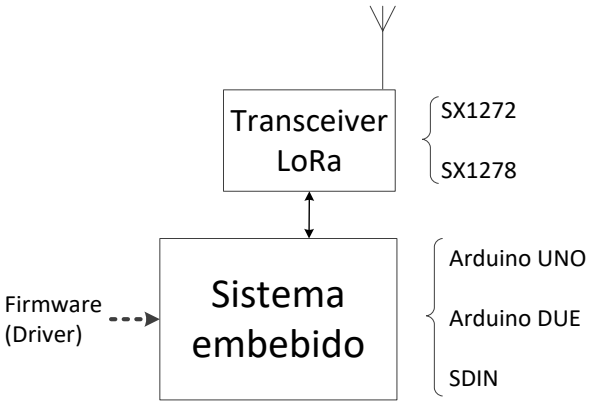


Figura 1. Disposición de elementos necesarios para la creación del driver con LoRa
Fuente: Esquema desarrollado por autor

3.1.1 Transceiver Semtech LoRa

Los módulos Semtech LoRa son transceivers RF (serie SX127'x') con la capacidad de comunicarse con la tipología M2M (maquina a máquina). Con estos dispositivos se pueden crear complejas redes de comunicación de largo alcance y con la posibilidad de conectar millones de dispositivos. Estos dispositivos optimizan el consumo energético, aumentando el ciclo de vida de las baterías de los sistemas embebidos utilizados, brindando aplicaciones con

redes ideales para Internet de las Cosas (IoT), automatización, meteorología, seguridad, tracking y aplicaciones M2M (Semtech, s.f.).

Los dispositivos de radio *Semtech* LoRa cuentan con un modem de largo alcance con alta inmunidad al ruido y mínimo consumo de corriente.
La serie SX1272/78 utilizan la técnica de modulación patentada LoRa alcanzando una sensibilidad de -137dBm (SX1272) y -148 dBm (SX1278). Esta alta sensibilidad combinada con amplificador LNA (amplificador de bajo ruido) de +20 dBm optimiza el rango de alcance y brindan la robustez necesaria para aplicaciones industriales (Semtech SX1272, 2015 & Semtech SX1278, 2015).

El consumo de corriente es nulo cuando se encuentra en modo sleep y tiene un mínimo consumo cuando se encuentra en modo recepción llegando hasta 12mA. Tiene como ventaja la incorporación de la Verificación por redundancia cíclica (CRC) de hasta 256 Bytes, característica muy útil para comprobar si los datos fueron recibidos correctamente en el transceiver.

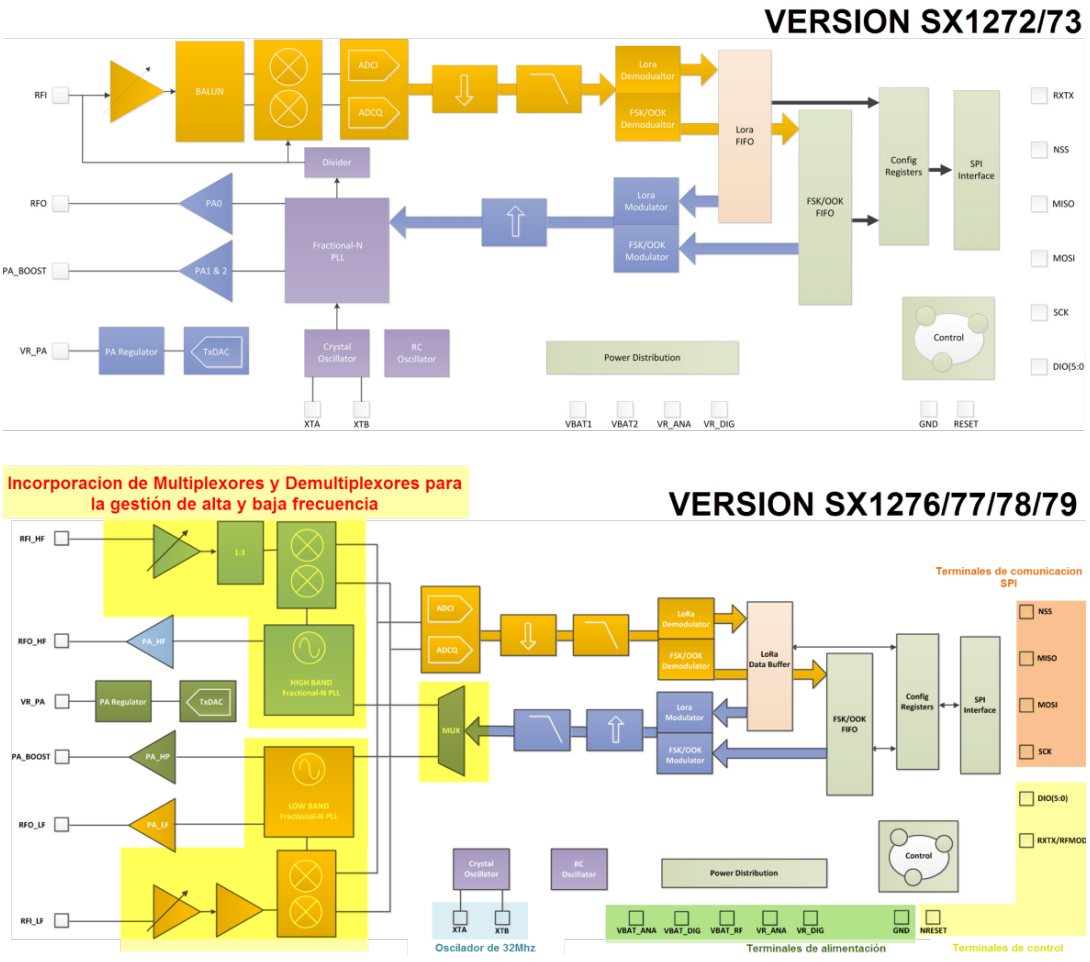


Figura 2.Diagrama de bloques con principales características y diferencias entre SX1272 y SX1278.
Fuente: Semtech SX1272 (2015) & Semtech SX1278 (2015). Editado por autor

La versión SX1278 presenta mejoras a nivel de la arquitectura respecto al SX1272. La principal mejora es la incorporación de multiplexores y demultiplexores para el manejo de bajas y altas frecuencias (ver Figura 11). Para el chip radio SX1278, bajas frecuencias (LF) corresponden a frecuencias por debajo de 525 MHz y altas frecuencias (HF) se encuentran por encima de 779 MHz (Semtech SX1272, 2015 & Semtech SX1278, 2015).

La *Tabla 1* presenta las frecuencias máximas y mínimas de trabajo de los transceivers de radio seleccionados.

Tabla 1. Relación de frecuencias de trabajo para el SX1272 y SX1278
Fuente: Semtech SX1272 (2015) & Semtech SX1278 (2015)

Dispositivo	Frecuencia mínima (MHz)	Frecuencia máxima (MHz)
SX1272	860	1020
SX1278	137	525

Existen frecuencias de libre uso conocidas como ISM (Industrial, Scientific and Medical) que son frecuencias intencionalmente libres para uso no comercial en aplicaciones de tipo industrial, científico y médico. La *Tabla 2* presenta las principales frecuencias ISM compatibles con los dispositivos LoRa propuestos.

Tabla 2. Banda de frecuencia ISM compatibles con los transceivers de radio propuestos
Fuente: Texas Instruments (2005)

Banda ISM (MHz)	Frecuencia mínima (MHz)	Frecuencia máxima (MHz)	Ancho de Banda (MHz)	Usos por continente
433	433.050	434.790	1.84	Europa, África y parte del norte de Asia.
869	868	870	2	Europa, África, Asia y Oceanía.
900	902	928	26	Continente americano

De acuerdo a la *Tabla 2*, serán seleccionadas las frecuencias de trabajo a 433Mhz para el chip SX1278 y 869 MHz para el chip radio SX1272 teniendo en cuenta el desarrollo de pruebas en el espacio Europeo.

Algunas características importantes de los dispositivos seleccionados se presentan en la *Tabla 3*.

Tabla 3. Principales características de los dispositivos LoRa seleccionados
Fuente: Semtech SX1272 (2015) & Semtech SX1278 (2015)

Radio SX	Frecuencia seleccionada MHz	Parámetros importantes del LoRa			
		Factor de alcance SF	Ancho de banda/ canal (kHz)	Tasa efectiva de bits (kbps)	Sensibilidad (dBm)
1272	869.525	6 a 12	125 a 500	0.24 a 37.5	-117 a -137
1278	433.800	6 a 12	7.8 a 500	0.018 a 37.5	-111 a -148

La sensibilidad del receptor identifica el valor mínimo de potencia que necesita un receptor para poder decodificar o extraer datos para alcanzar una determinada tasa de bits o de velocidad (Ramirez, 2015). Comúnmente se expresan en dBm e indican que entre menor sea su valor, el receptor tendrá mejor sensibilidad; por lo que se concluye que el dispositivo SX1278 tiene como ventaja el tener mejor sensibilidad al recibir datos.

Otra importante diferencia a favor del dispositivo SX1278, es contar con canales o ancho de banda más reducidos de hasta 7.8kHz como valor mínimo. Sin embargo, lo anterior implica trasmisión y recepción de datos con tasas efectivas de datos significativamente bajas que podrían dar lugar a pérdidas de datos o tiempos excesivos de transmisión o recepción de datos.

Tabla 4. Principales características eléctricas de los dispositivos LoRa seleccionados
Fuente: Semtech SX1272 (2015) & Semtech SX1278 (2015)

Voltaje (V)			Rango de temperatura (°C)			Corriente RX (mA)	Corriente TX (mA) **CONFIGURABLE**		
Min	Est	Max	Min	Est	Max		Min	Est	Max
1.8	3.3	3.9	-40	25	+85	<12	10	100	240

Min: Mínimo Max: máximo Est: Estándar

Existen parámetros eléctricos configurables como la máxima corriente cuando el dispositivo trasmite (ver la *Tabla 4*). Estos parámetros se configuran con registros internos del transceiver mediante el protocolo de comunicación síncrono SPI.

El SPI es un protocolo de comunicación serial sincrónico *full dúplex*, es decir permite trasmitir y recibir información al mismo tiempo. Para usar este protocolo se requiere configurar un dispositivo como maestro y los demás como esclavos (López, s.f.).

Este protocolo utiliza 4 líneas de comunicación que se describen como:
SCLK: corresponde a la señal de reloj, es generada por el maestro y sincroniza la transferencia de datos.
MOSI (Master Out Slave In): transfiere los datos del maestro hacia el esclavo.

MISO (Master In Slave Out): transfiere los datos del esclavo hacia el maestro.
NSS (Slave Select): esta línea es generada por el maestro y permite seleccionar el dispositivo esclavo para iniciar la comunicación. Con esta línea es posible hacer uso de varios dispositivos esclavos utilizando las mismas líneas de comunicación.

Los dispositivos de radio LoRa se comportan como esclavos y el microcontrolador del sistema embebido será el maestro en la comunicación SPI.

A nivel lógico de sincronización y trasmisión de datos, la comunicación SPI requiere la configuración de la polaridad del reloj (CPOL) y el bit fase del reloj (CPHA).

Los dispositivos de radio LoRa SX1272/78 utilizan CPOL=0 y CPHA=0; lo que se conoce como modo 0 SPI. El bit más significativo (MSB) de un byte enviado debe ser el primero y la velocidad del SCLK no debe superar los 10MHz (Semtech SX1278, 2015)

El esquema propuesto de conexión entre los transceiver de radio LoRa y el microcontrolador del sistema embebido se presenta en la *Figura 3*.

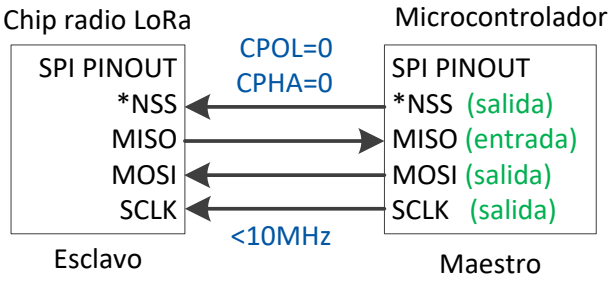


Figura 3. Principales características de comunicación SPI para los dispositivos LoRa seleccionados
Fuente: Semtech (2013). Esquema adaptado por autor

Algunos fabricantes realizan tarjetas de adecuación para facilitar la manipulación de los transceivers de radio LoRa. Al realizar estas tarjetas, se eliminan o incorporan pines físicos de control respecto a las características originales del chip. Los fabricantes mantienen los pines de comunicación SPI y eliminan algunos pines de control del chip radio. Las principales características de las tarjetas LoRa adquiridas para realizar las pruebas de funcionamiento se presentan en la *Tabla 5*.

Tabla 5. Comparación de pines de control según los fabricantes de tarjetas para los transceivers estudiados
Fuente: Semtech (2013), Dorji (2015) & NiceRF (2015)

<table>
<tr><th rowspan="2">Tarjeta</th><th rowspan="2">Chip radio</th><th rowspan="2">Marca</th><th colspan="4">Pines de control físicos</th></tr>
<tr><th>DIO0 – DIO5</th><th>RXTX</th><th>Pines SPI</th><th>RESET</th></tr>
<tr><td>SX1272RF1</td><td>SX1272</td><td>Semtech</td><td rowspan="2">Pines de entrada/ salida, configurables por software.</td><td>Pin de salida. Se coloca en “1” cuando el dispositivo está trasmitiendo datos.</td><td rowspan="3">NSS
MISO
MOSI
SCLK</td><td rowspan="3">Pin de entrada. Utilizado al iniciar o arrancar el dispositivo.</td></tr>
<tr><td>DRF1278F</td><td rowspan="2">SX1278</td><td>Dorji</td><td rowspan="2">NI*</td></tr>
<tr><td>LoRa1278</td><td>NiceRF</td><td>Pines de entrada/ salida, configurables por software NI* DIO3–DIO5</td></tr>
</table>

*NI: No implementado

La versión LoRa1278 de *NiceRF* realiza adecuaciones relacionadas con el hardware incorporando 2 pines adiciones de control que permiten habilitar el uso de la antena para trasmitir o recibir. La descripción de las funciones de los pines TXEN y RXEN se presentan en la *Tabla 6.*

Tabla 6. Descripción del uso de los pines RXEN y TXEN para la tarjeta LoRa1278 de NiceRF
Fuente: NiceRF (2015)

Estados lógicos de pines. Pines de entrada para LoRa.		Descripción de funcionamiento para la Antena
Pin TXEN	Pin RXEN	
0	0	Modo Sleep o bajo consumo
0	1	Configurada para Recibir
1	0	Configurada para Transmitir

Físicamente, la tarjeta SX1272RF1 de Semtech es de mayor tamaño y cuenta con un conector SMA para antenas a 868Mhz e impedancia de 50Ω. Las tarjetas DRF1278F y LoRa1278 tienen el mismo tamaño (significativamente más pequeño que la versión SX1272RF1) y permiten conectar antenas pasivas tipo primavera a 433Mhz con impedancia de 50Ω. Los tamaños de pueden observar en las *Figura 4* y *Figura 5*.

Figura 4. Tarjeta SX1272RF1 de Semtech (868Mhz)
Fuente: Fotografía tomada por autor

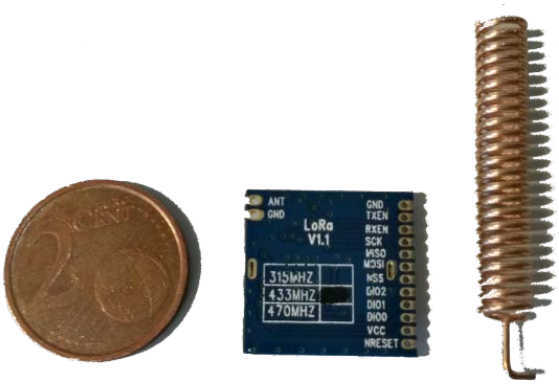


Figura 5.Tarjeta LoRa1278 de NiceRF (433Mhz)
Fuente: Fotografía tomada por autor

3.1.2 Sistemas embebidos - importancia del hardware libre

Un sistema embebido corresponde a un conjunto de componentes electrónicos diseñados para funciones específicas. Teniendo en cuenta el estudio de los dispositivos LoRa para aplicaciones del ámbito del desarrollo, los sistemas embebidos deben ser libremente replicados, por lo que se contempla el uso de *hardware libre*.

"El hardware libre es aquel hardware cuyo diseño se hace disponible públicamente para que cualquier persona lo pueda estudiar, modificar, distribuir, materializar y vender, tanto el original como otros objetos basados en ese diseño. Las fuentes del hardware (entendidas como los ficheros fuente) habrán de estar disponibles en un formato apropiado para poder realizar modificaciones sobre ellas. Idealmente, el hardware de fuentes abiertas utiliza componentes y materiales de alta disponibilidad, procesos estandarizados, infraestructuras abiertas, contenidos sin restricciones, y herramientas de fuentes abiertas de cara a maximizar la habilidad de los individuos para materializar y usar el hardware. El hardware de fuentes abiertas da libertad de controlar la tecnología y al mismo tiempo compartir conocimientos y estimular la comercialización por medio del intercambio abierto de diseños" (Lazalde, Torres & Vila-Viñas, 2015).

En términos prácticos, hardware libre es una placa que incorpora un microcontrolador programable conectado a circuitos discretos o integrados y a su vez a pines o terminales con los que usuario puede cómodamente conectar sensores, actuadores o dispositivos electrónicos para desarrollar aplicaciones específicas. Es así como se propone utilizar la placa Arduino, que cumple con los parámetros de hardware libre, se encuentra ampliamente documentada en la web y cuenta con un IDE propio para el desarrollo del firmware o aplicación lógica de control.

Utilizar Arduino para el desarrollo de aplicaciones específicas tiene las siguientes ventajas (Torrente, 2013):

- *Comunidad de desarrolladores:* muchos usuarios lo utilizan, enriquecen la documentación y comparten sus ideas y conocimientos.
- *Entorno de programación multiplataforma:* es posible instalar y ejecutar en sistemas operativos Windows, Mac OS X y Linux.
- *Sencillo entorno y lenguaje de programación:* resulta fácil de aprender y de utilizar para principiantes. Además, las librerías se encuentran documentadas con ejemplos detallados y gran cantidad de proyectos publicados en diferentes formatos.
- *Placas Arduino económicas:* la placa Arduino estándar (llamada Arduino UNO) ya preensamblada y lista para funcionar cuesta alrededor de 20 euros. También se puede construir la placa adquiriendo los componentes por separado, con lo que el precio total de la placa resultante sería incluso menor.
- *Reutilizables y versátiles*: reutilizables porque se puede aprovechar la misma placa para varios proyectos (es sencillo de desconectar reconectar y reprogramarla), y versátiles porque las placas proveen varios tipos de entradas y salidas de datos, los cuales permiten capturar información de sensores y enviar señales a actuadores de múltiples formas.

Sin embargo, existen diversas versiones de la placa Arduino entre las que se propone utilizar *Arduino UNO* que dispone de un procesamiento de 8 bits con un nivel bajo de complejidad y *Arduino DUE* de 32 bits con un nivel intermedio de complejidad. También se propone el uso

de la placa *SDIN* desarrollada por GEDIRCI que cuenta con 32 bits de procesamiento y un nivel alto de complejidad.

3.1.2.1 Arduino UNO

Arduino cuenta con un modelo estándar de placa denominada Arduino UNO siendo la más utilizada de las diferentes versiones. Apareció en 2010 y ha contado con tres revisiones, por lo que el modelo actual se suele llamar UNO Rev3 o simplemente UNO R3. Cuenta con un microcontrolador de 8 bits con referencia ATmega328P de la marca Atmel que funciona con un voltaje nominal de 5V y a una frecuencia de 16 MHz. Tiene 14 entradas/salidas digitales (6 de los cuales pueden ser usados como salidas PWM), 6 entradas analógicas, comunicación SPI, comunicación I^2C y comunicación UART (Torrente, 2013).

Arduino funciona con voltajes de 7 a 12 voltios por alimentación externa (conector Jack macho) o por USB (voltaje suministrado por el ordenador). Para programar la lógica de programación o firmware, Arduino utiliza un segundo microcontrolador que le permite ser configurado por USB y realizar una depuración básica del código.

La Figura 6 presenta una fotografía de la placa Arduino UNO y la Figura 7 presenta el esquema de arquitectura aproximada utilizada por la placa en donde se puede observan la incorporación del protocolo SPI necesario para los dispositivos LoRa.

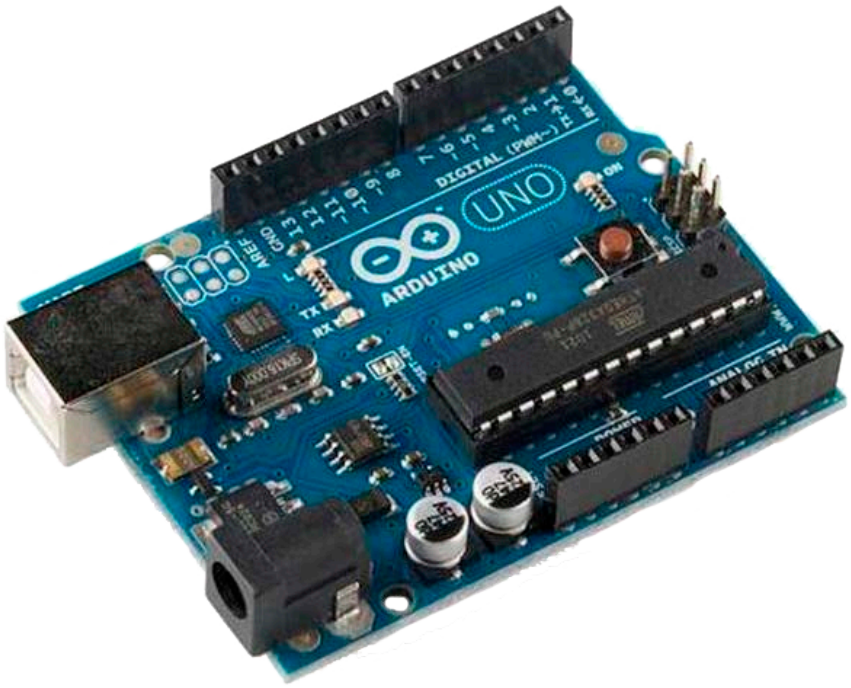


Figura 6. Placa Arduino UNO
Fuente: Arduino Board UNO (s.f.)

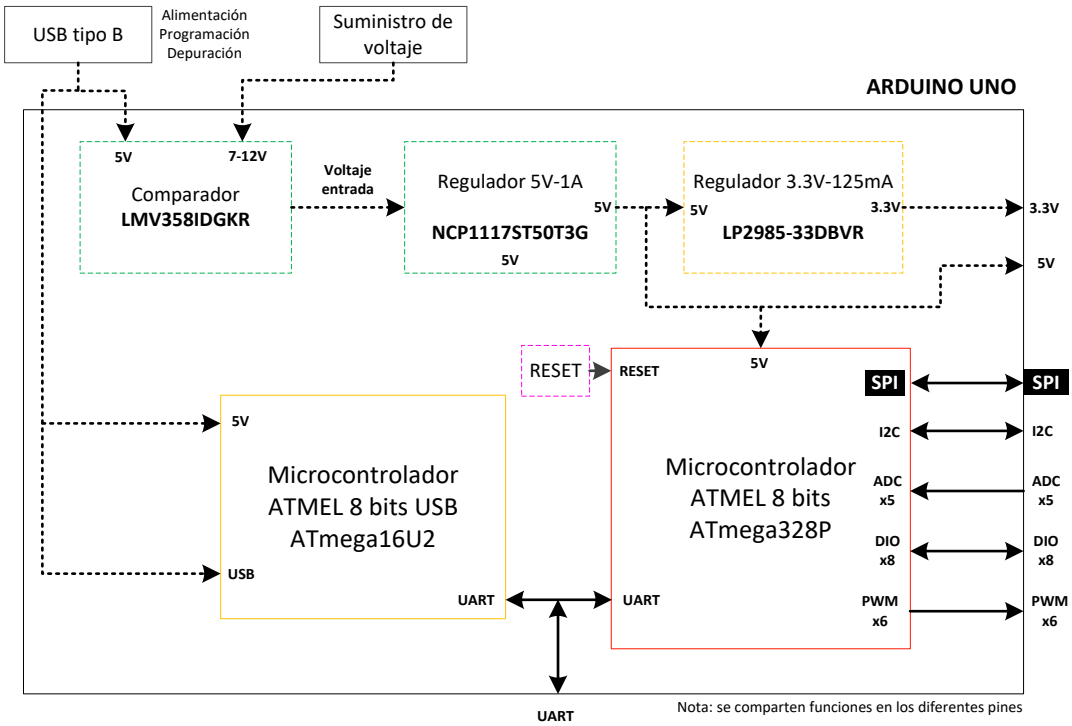


Figura 7. Arquitectura simplificada de Arduino UNO R3
Fuente: Arduino Board UNO (s.f.). Esquema desarrollado por el autor

3.1.2.2 Arduino DUE

Arduino DUE pertenece a una familia de microcontroladores de 32 bits siendo la primera tarjeta basada en ARM Cortex-M3. Dispone de un microcontrolador SAM3X8E fabricado también por Atmel que funciona con un voltaje nominal de 3.3V, a una frecuencia de 84 MHz.
Dispone de 54 pines de entrada/salida digital (12 de los cuales pueden ser usados como salidas PWM), 12 entradas analógicas, 4 módulos UART, 2 conversores digitales-analógicos, 2 puertos I2C independientes, 1 puerto SPI, 1 conector USB de tipo mini-B, y un conector USB de tipo mini-A.

Arduino DUE funciona con voltajes de 7 a 12 voltios por alimentación externa (conector Jack macho) o por USB (voltaje suministrado por el ordenador). Para programar la lógica de programación o firmware, Arduino utiliza un segundo microcontrolador que le permite ser configurado por USB y realizar una depuración básica del código. Sin embargo, existe un segundo puerto USB que puede ser utilizado para propósitos relacionados con la aplicación.

La *Figura 8* presenta una fotografía de la placa Arduino DUE y la *Figura 9* presenta el esquema de la arquitectura aproximada utilizada por la placa en donde se puede observan la incorporación del protocolo SPI necesario para el funcionamiento de los dispositivos LoRa.

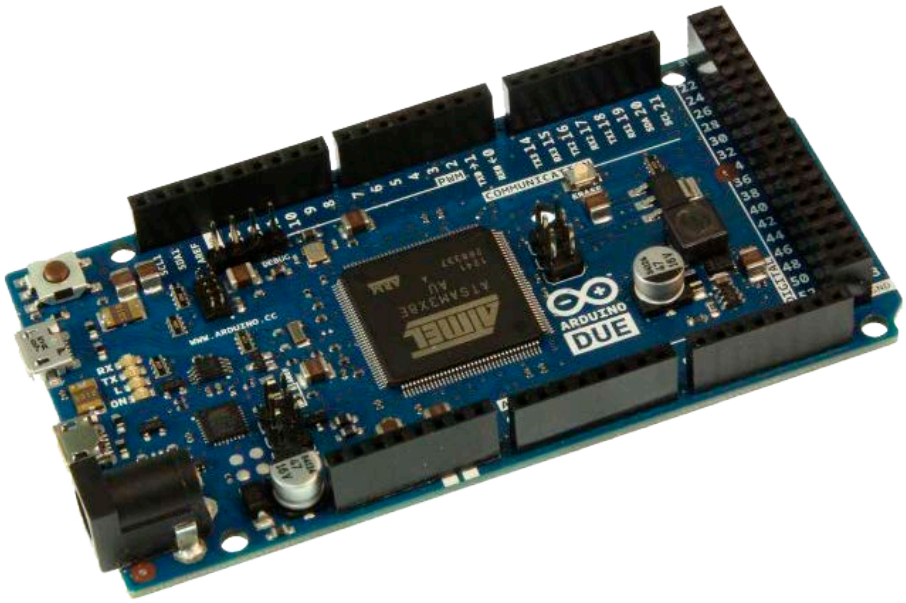


Figura 8. Placa Arduino DUE
Fuente: Arduino Board DUE (s.f.)

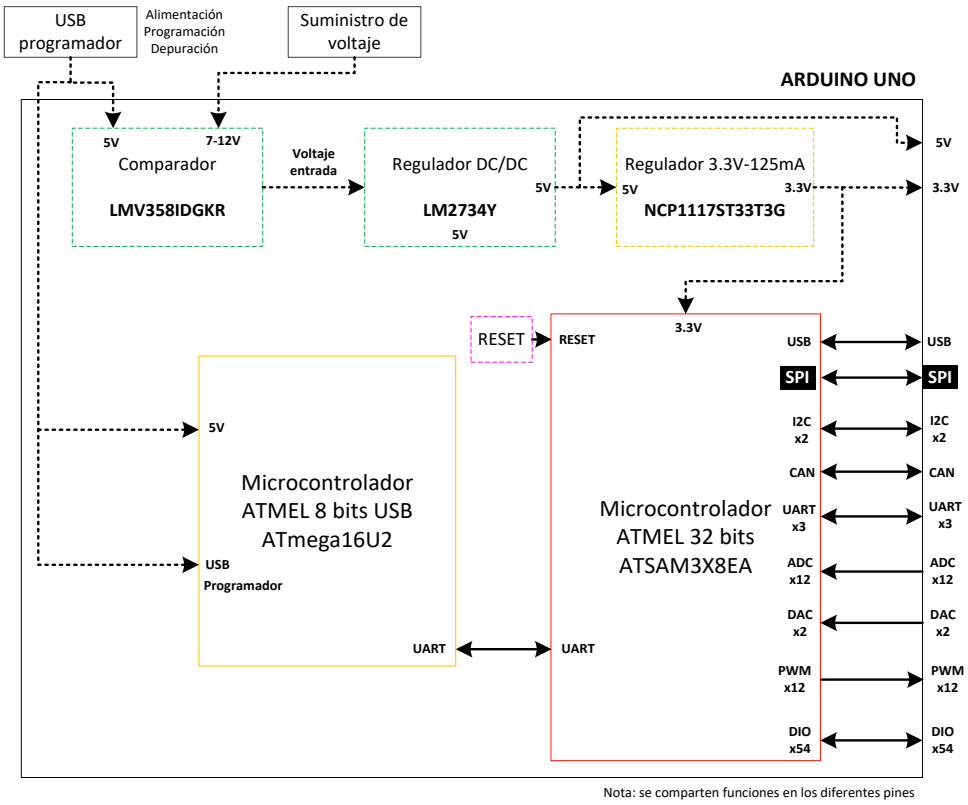


Figura 9. Arquitectura simplificada de Arduino DUE.
Fuente: Arduino Board DUE (s.f.). Esquema desarrollado por el autor

3.1.2.3 SDIN

Los módulos SDIN desarrollados por GEDIRCI, son tarjetas de adquisición de datos entre los que se encuentran, termopares, RTD, sensores analógicos (4-20mA y 0-10V) y módulos de alimentación. Estos módulos han sido ampliamente utilizados por diversos sistemas de monitoreo brindando la robustez y eficiencia de tipo industrial.

Figura 10. Módulo SDIN Therm
Fuente: Solar Decahtlon Europe (2014).

Algunos de estas aplicaciones se encuentran en el desarrollo de la casa solar ”MagicBox” presentada en la competición de viviendas autosuficientes “Solar Decathlon” 2005, utilizado en el Campus de Montegancedo de la Universidad Politécnica de Madrid y recientemente en las instalaciones del Centro de Innovación para el Desarrollo de la Universidad Politécnica de Madrid itdUPM.
Sin embargo, en las ediciones del “Solar Decathlon” del 2014, los módulos SDIN fueron utilizados de forma masiva como dispositivo de monitoreo para las viviendas autosuficientes de la competición.

El tipo de comunicación utilizada entre los módulos SDIN es RS-485 que es un protocolo de comunicación industrial serial, utiliza dos hilos y permite múltiples conexiones sobre el mismo bus o canal. Algunas de las principales ventajas de este tipo de comunicación son las siguientes (López, s.f.).

a) Bajo costo: los circuitos integrados para trasmitir y recibir son económicos y solo requieren una fuente de alimentación +5V para poder generar una diferencia mínima de 1.5v entre las salidas diferenciales.
b) Capacidad de interconexión: RS-485 es una interface multi-enlace con la capacidad de lograr contar con múltiples transmisores y receptores. Con una alta impedancia receptora, los enlaces con RS-485 pueden llegar a tener a lo máximo hasta 256 nodos sobre el mismo bus.

c) Longitud de enlace: En un enlace RS-485 puede tener hasta 4000 pies de longitud.
d) Rapidez: La razón de bits puede ser tan alta como 10 Mega bits/ segundo.

El circuito integrado utilizado para la comunicación en la placa de los módulos SDIN corresponde al MAX13487E que lo hace *half dúplex*, permite contar con velocidades de transmisión de hasta 16Mbps y una conexión de hasta 128 transceiver sobre el bus (Maxim Integrated Products, 2015).

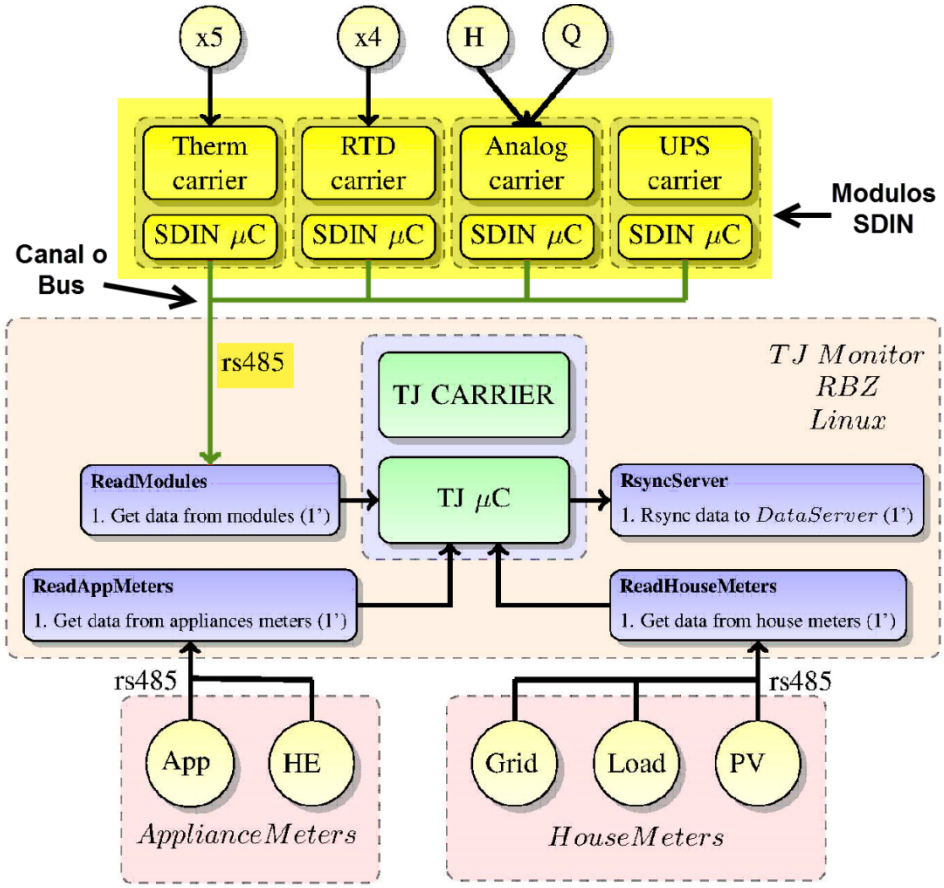


Figura 11. Arquitectura de monitoreo para una vivienda autosostenible.
Fuente: Solar Decahtlon Europe (2014). Editado por autor

El esquema de la *Figura 11* presenta la arquitectura de conexión de los sistemas de monitoreo para las viviendas sostenibles de la competencia Decahtlon Europe en Francia para el 2014. En este esquema se observa el TJ Monitor (equivale al dispositivo maestro del sistema), que se encarga de recibir la información de medidores de potencia y de los módulos SDIN. Además, el TJ Monitor utiliza LINUX, por lo que el sistema está desarrollado con software libre y de código abierto.
También se presentan los módulos SDIN conectados a un mismo bus RS-485 sobre el cual se transmiten los parámetros de termopares, RTD, dispositivos analógicos, dispositivos digitales y UPS (ver *Figura 12*).

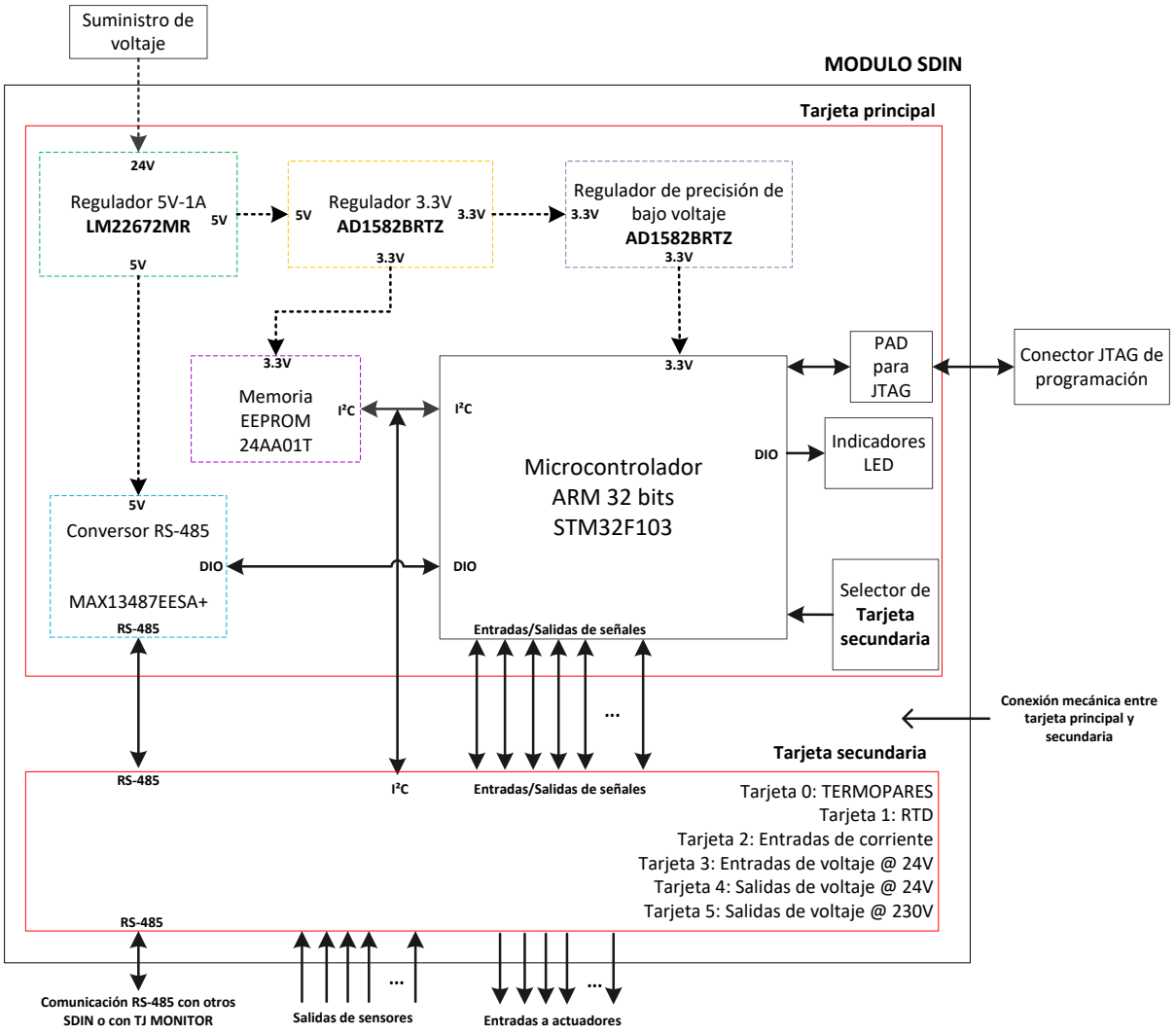


Figura 12. Arquitectura a nivel de hardware de los módulos SDIN.
Fuente: Planos compartidos por GEDIRCI. Esquema desarrollado por autor

A nivel de hardware, los módulos SDIN están compuestos por dos tarjetas de conexión modular, una tarjeta primaria donde se encuentra el microcontrolador ARM de 32 bits (ver *Figura 13*) y la posibilidad de conexión de una tarjeta secundaria acorde a un selector que determina la función y conexión de la tarjeta secundaria.

Las tarjetas secundarias pueden ser de tipo 0: termopares, 1: RTD, 2: entradas de corriente, 3: entradas de voltaje @ 24V, 4: salidas de voltaje @ 24V y 5: salidas de voltaje @ 230V (ver *Figura 14*). La conexión de estas tarjetas se realiza de forma mecánica donde el usuario ajusta la tarjeta secundaria debajo de la tarjeta principal (ver *Figura 15*).

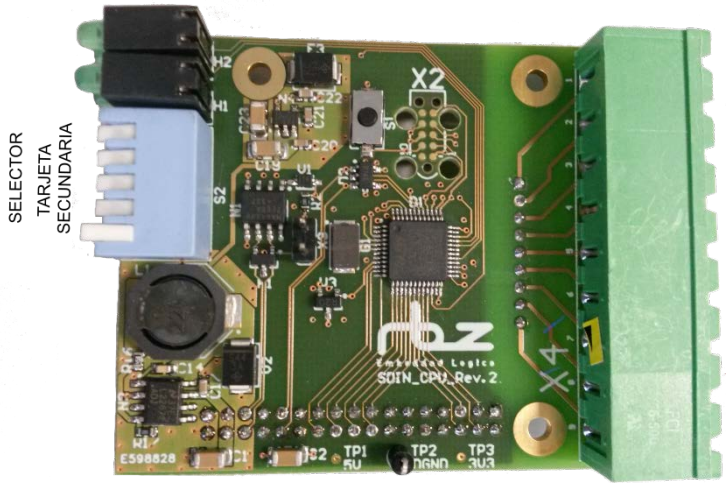


Figura 13. Tarjeta principal del módulo SDIN.
Fuente: Fotografía tomada por autor

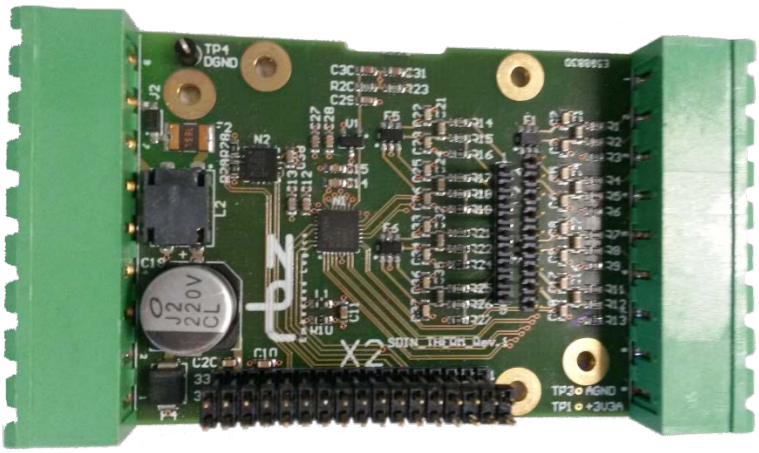


Figura 14. Tarjeta secundaria del módulo SDIN tipo 0: Termopares.
Fuente: Fotografía tomada por autor

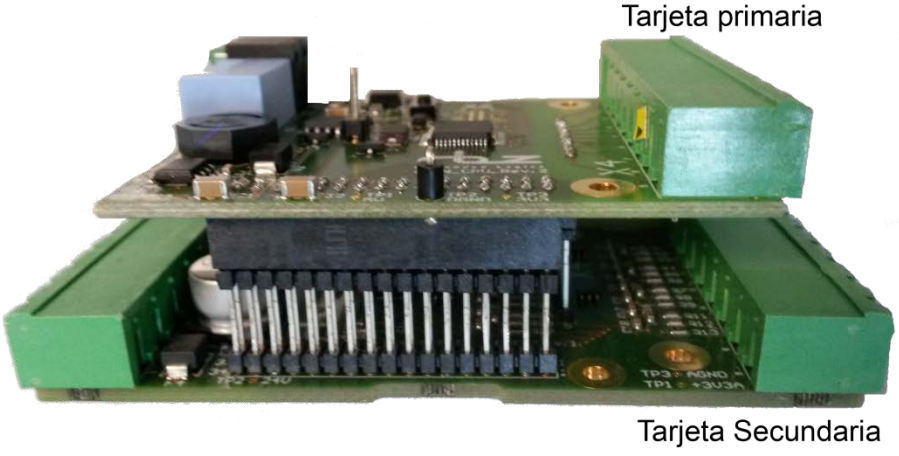


Figura 15. Conexión mecánica de los módulos SDIN entre la tarjeta primaria y secundaria
Fuente: Fotografía tomada por autor

En la tarjeta principal del SDIN se encuentra el microcontrolador STM32F103 que dispone del protocolo de comunicaciones SPI para dos módulos. La placa SDIN funciona a 24V.

3.2 Descripción del software utilizado - Importancia del software libre

El *software libre* cuenta con un papel relevante en el ámbito de la cooperación y el desarrollo por cuanto asegura la soberanía tecnológica, impulsa la innovación nacional, optimiza los gastos fortaleciendo el desarrollo local y facilita la inclusión digital. Algunas características del software libre en relación con el ámbito del desarrollo son las siguientes (Administración publica Ecuador, s.f):

Autonomía tecnológica: Adoptando el uso de software libre y con las posibilidades que éste ofrece de acceder al código fuente, muchos usuarios pasarán de ser consumidores a ser desarrolladores de software. Esto significa que se podrán adaptar el código o programas a necesidades específicas, y todas las modificaciones deberán realizarse siguiendo los requisitos exigidos por el modelo del software libre (un ejemplo claro se encuentra con el uso del S.O. FreeRTOS).

Estandarización e integración: El software libre es producido utilizando especificaciones y estándares tecnológicos libres y públicos, también denominados “estándares abiertos”. Esto beneficia la integración de sistemas y el intercambio de información, de forma que se garantiza la accesibilidad sin restricciones para los usuarios.

Seguridad: El hacer públicos los códigos de los programas o aplicaciones favorece la seguridad de los mismos. Utilizando software libre es posible conocer y analizar lo que está siendo ejecutado realmente por el programa, qué tipo de información maneja y cómo lo hace. Una buena seguridad debe basarse en la transparencia. El software privativo oculta estos aspectos y muchas veces no se conoce si la información se envía a servicios remotos.

Independencia de proveedores: Adquiriendo un software privativo se genera una relación de dependencia con respecto al fabricante. Una vez que se instala dicho software, se dependerá del fabricante para obtener actualizaciones. En muchos casos, el fabricante obligará a actualizar a nuevas versiones aunque no se desee.

Democratización de la información: Las tecnologías de la información han pasado a ocupar un lugar central en la sociedad. Si bien cada vez son más los usuarios que acceden a dichas tecnologías, la “brecha tecnológica” todavía es grande y es un factor más de exclusión social. El software libre favorece la democratización de la información permitiendo la utilización de protocolos, formatos y lenguajes.

Economía: se estima que la compra de un sistema operativo más un paquete de suite de oficina, ambos con una licencia comercial, cuestan entre 300 y 600 dólares por cada

computadora, y ese gasto debe renovarse cada dos o tres años debido a la dependencia hacia el fabricante. Los países en vías de desarrollo con las carencias de recursos que cuentan, pueden ahorrar una gran cantidad de recursos económicos.

El sistema operativo planteado para ser usado en el desarrollo de las pruebas de los dispositivos LoRa fue *Linux* con el uso de la versión *Ubuntu*, la cual es conocida por ser bastante intuitiva, contar con un aspecto similar a Windows y ser gratuita. Sobre Linux, se utilizaron IDE (Entorno de Desarrollo Integrado), que es un conjunto de herramientas de software para crear, depurar y compilar el código fuente.

Se utilizó el *IDE Arduino* para programar las versiones UNO y DUE de las placas Arduino.
También se utilizó sobre Linux el *IDE eclipse* para programar el driver compatible con FreeRTOS, sistema operativo utilizado por los módulos SDIN.

3.2.1 IDE Arduino

Arduino cuenta con su propio IDE capaz de crear, editar, depurar y transferir el código (también llamado *"sketch"*) a la placa Arduino. El IDE puede ser utilizado por cualquier versión de placa y es compatible con los sistemas operativos Windows, Mac OS X, y Linux.
El entorno de desarrollo está escrito en JAVA, por lo que previamente se debe instalar la última versión de la JDK (Java Development Kit).
Gran cantidad de desarrolladores comparten "sketch" y librerías para las placas Arduino en *GitHub*, la cual es una plataforma para alojar proyectos utilizando un sistema de control de versiones Git, almacenados de forma gratuita y compartidos al público.

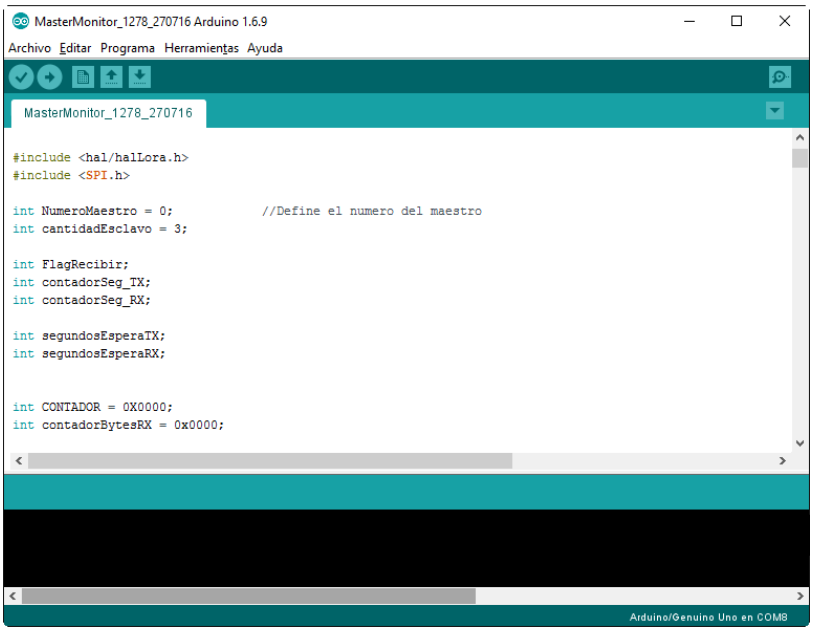


Figura 16. Interfaz del IDE Arduino
Fuente: Captura de pantalla realizada por el autor

El IDE Arduino utiliza el lenguaje de programación Arduino que en el sentido estricto de la palabra *no* es un lenguaje, sino un conjunto de instrucciones del lenguaje C y C++ "camufladas". Estas instrucciones están diseñadas para simplificar el desarrollo de programas para microcontroladores AVR. Es decir, al escribir un sketch en "lenguaje Arduino", realmente se está programando en una versión simplificada del lenguaje C/C++ ocultando gran cantidad de detalles técnicos para el desarrollador, lo que facilita el uso de la plataforma Arduino en conjunto hardware-software.
Sin embargo, al reducir el nivel de complejidad en la programación, se está utilizando un subconjunto de toda la funcionalidad que puede llegar a ofrecer el lenguaje C/C++ (Torrente, 2013).

3.2.2 IDE Eclipse – Sistema operativo FreeRTOS

Eclipse, es una plataforma IDE (Entorno de Desarrollo Integrado) de software de código abierto para el desarrollo de aplicaciones, diseñada para ser extendida (escalable) de forma indefinida a través de *plugins* o complementos. Está compuesta por un conjunto de herramientas de programación de código abierto multiplataforma para desarrollar aplicaciones sin la exclusividad de un lenguaje específico, sino que es un IDE genérico. Sin embargo, es popular entre las comunidades de desarrolladores para el desarrollo de aplicaciones JAVA.
Entre las principales características del IDE Eclipse se encuentran (Genbetadev, 2014):

Perspectivas, editores y vistas: en Eclipse el concepto de trabajo está basado en las perspectivas, que no es otra cosa que una pre-configuración de ventanas y editores, relacionadas entre sí, y que permiten trabajar en un determinado entorno de forma óptima.

Gestión de proyectos: El desarrollo sobre Eclipse se basa en los proyectos, que son el conjunto de recursos relacionados entre sí, como pueden ser el código fuente, librerías, documentación, ficheros configuración, árbol de directorios, archivos multimedia, entre otros. El IDE proporciona asistentes y ayudas para la creación de proyectos. Sin embargo, en la web se encuentran la documentación completa con tutoriales para principiantes.

Depurador de código: Incluye un potente depurador, de uso fácil e intuitivo, y que visualmente ayuda a optimizar el código (utilizando un botón de la interfaz). De nuevo, se tiene una perspectiva específica para la depuración de código, la perspectiva depuración, donde se muestra de forma ordenada toda la información necesaria para realizar dicha tarea.

Extensa colección de plugins (complementos): se encuentran disponibles en una gran cantidad de complementos, algunos publicados por Eclipse y otros compartidos por terceros. Algunos complementos son gratuitos, de pago, bajo distintas licencias, pero casi para cualquier necesidad existe un *plugin* adecuado.

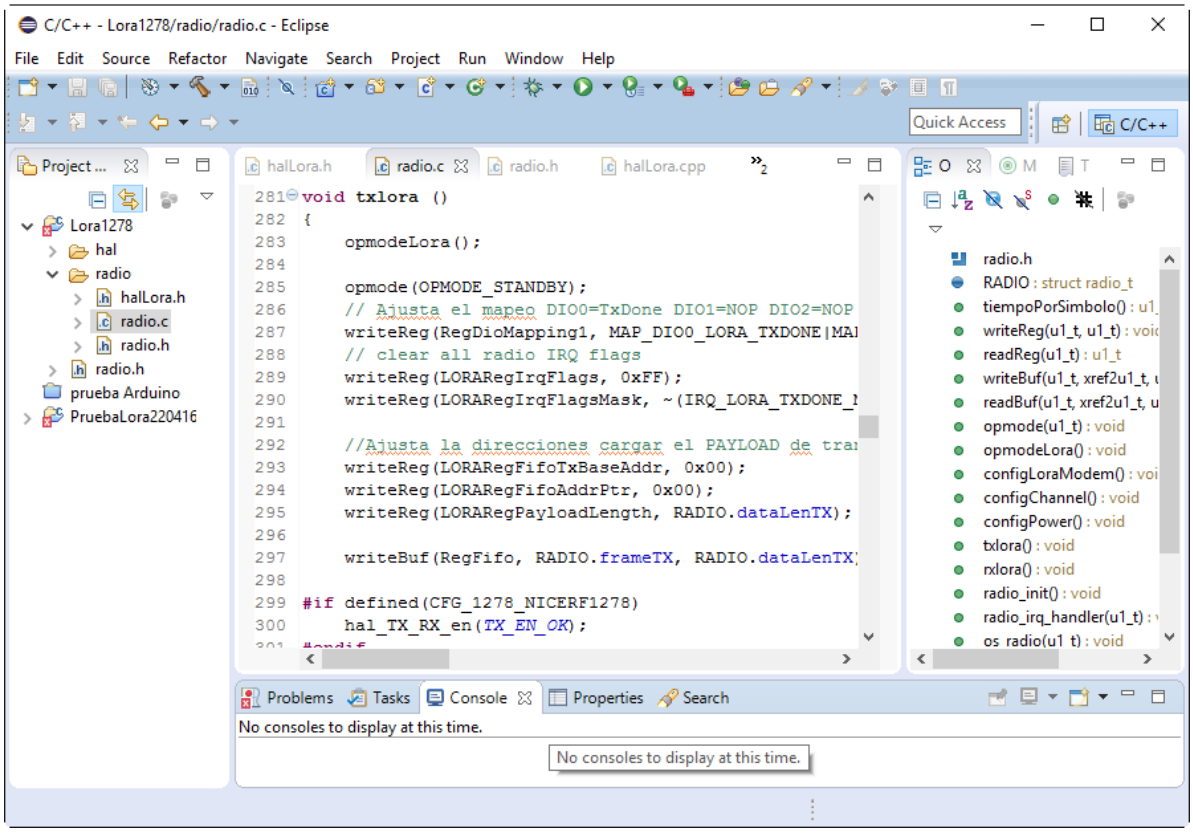


Figura 17. Interfaz del IDE Eclipse
Fuente: Captura de pantalla realizada por el autor

Sobre Linux, se instaló el IDE Eclipse con la versión MARS 2. Se realizó la programación y depuración de las librerías de los dispositivos LoRa utilizando el lenguaje de programación C y C++ para adaptar el driver al S.O. FreeRTOS, utilizado por el módulo SDIN.

Los módulos SDIN utilizan como software FreeRTOS, que es un sistema operativo en tiempo real y de código abierto capaz de funcionar sobre un microcontrolador.
Utilizar FreeRTOS como software de programación sobre los SDIN cuenta con las siguientes ventajas (Barry, 2009):

- FreeRTOS se encuentra ampliamente documentado.
- Cuenta con gran control en el tiempo de ejecución de cada tarea que se desarrolla.
- Tiene un “scheduler” (comúnmente utilizado en los sistemas operativos) diseñado de tal forma que permita ejecutar tareas en tiempos predecibles (normalmente descrito como determinístico). Esto resulta de gran utilidad en sistemas embebidos donde se busca que el sistema responda en tiempos estrictos.
- Permite establecer prioridades a las tareas a ejecutar.
- Es lo suficientemente ligero para funcionar sobre un microcontrolador.

- Es software libre y de código abierto.
- Permite aplicaciones escalables en el tiempo.
- Utiliza un estándar para las definiciones de tareas, lo que promueve la independencia de la aplicación al desarrollador del software.

3.3 Metodología de trabajo

3.3.1 Enfoque de la investigación

Es seleccionado el *enfoque cuantitativo de investigación*, por cuanto las medidas y la cuantificación de los datos constituyen el procedimiento empleado para alcanzar la objetividad en el proceso investigativo. Este enfoque promueve la búsqueda de la objetividad y la cuantificación orientadas a establecer tendencias a partir del estudio de las características de las pruebas realizadas (Monje, 2011).

El enfoque investigativo cuantitativo plantea la unidad de la ciencia, es decir, la utilización de una metodología única que es la misma de las ciencias exactas y naturales (Bonilla & Rodríguez, 1997). De acuerdo a lo anterior, los resultados se obtienen exclusivamente de la observación directa, de la comprobación y de la experimentación. Los conocimientos obtenidos se fundamentan en el análisis de hechos reales, sobre los cuales se realiza una validación neutra, objetiva y completa de los resultados.

3.3.2 Tipo de investigación

Teniendo en cuenta la aplicación de los dispositivos LoRa para el ámbito del desarrollo se propone utilizar el tipo *de investigación aplicada o tecnológica*, por cuanto se busca utilizar los conocimientos obtenidos de la investigación en aspectos prácticos y aplicados (servicios básicos), y con ello *traer beneficios a la sociedad o al sector productivo*. Además la *investigación aplicada* estructura procedimientos, innova estrategias, crea y prueba artefactos, y estima su valor pragmático (Tam, Vera & Oliveros, 2008).

3.3.3 Método de investigación

De acuerdo a las variables de la investigación (configuración de los dispositivos LoRa), se plantea el uso del método de investigación experimental. Básicamente se desarrolla mediante la manipulación de una o varias variables experimentales (configuración de los dispositivos LoRa) manipuladas por el investigador en condiciones rigurosamente controladas, con el fin de describir la evidencia causa-efecto produciendo una situación o acontecimiento (alcance máximo y consumo energético).

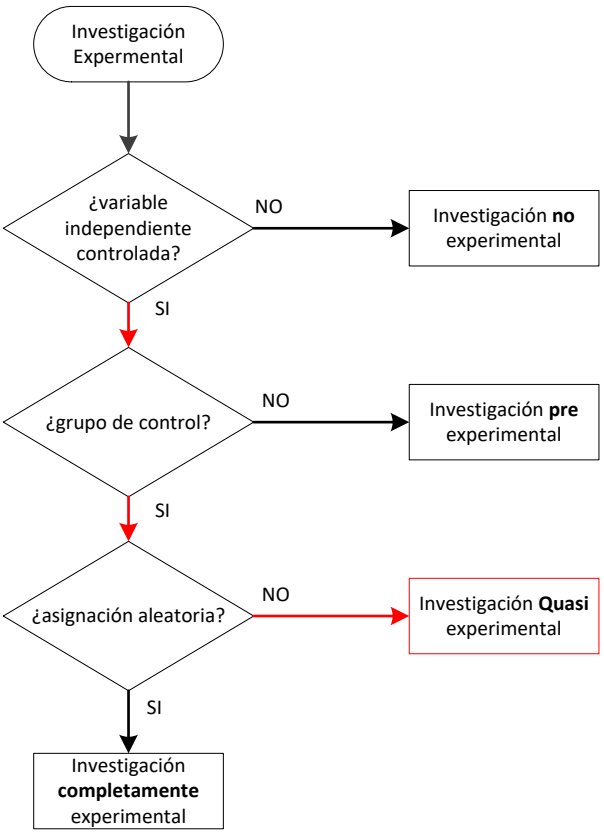


Figura 18. Árbol de decisión dicotómica de los métodos de investigación experimental.
Fuente: Tam, Vera & Oliveros (2008). Editada por autor

La *Figura 18* presenta un flujograma que permite determinar el subtipo de investigación experimental. Las variables independientes controladas en el proceso investigativo corresponde a los parámetros de configuración por software de los dispositivos LoRa (parámetros SF, BW y CR), al igual que el tipo de hardware utilizado (Arduino UNO, Arduino DUE y SDIN). Se establecen un subgrupo de dispositivos LoRa que corresponde a las versiones SX1272 y SX1278. Sin embargo, no se realizan pruebas aleatorias sino controladas sobre los dispositivos LoRa junto a sus respectivas configuraciones, por lo que la metodología de investigación es *Quasi-experimental*.

3.3.4 Población y muestra

La población corresponde a dispositivos transceivers de comunicación inalámbrica LoRa de Semtech junto con sistemas embebidos de hardware libre.
La muestra de la población propuesta corresponde a dispositivos transceivers de las referencias SX12782 (@868 MHz) y SX1278 (@434 MHz), al igual que placas de desarrollo Arduino UNO, Arduino DUE y la tarjeta de adquisición de datos SDIN de GEDIRCI.

3.3.5 Metodología

La metodología utilizada corresponde al modelo del patrón lineal de investigación cuantitativa propuesto por Spradley (1980). La *Figura 19* presenta el esquema propuesto de etapas para el desarrollo de la investigación propuesta.

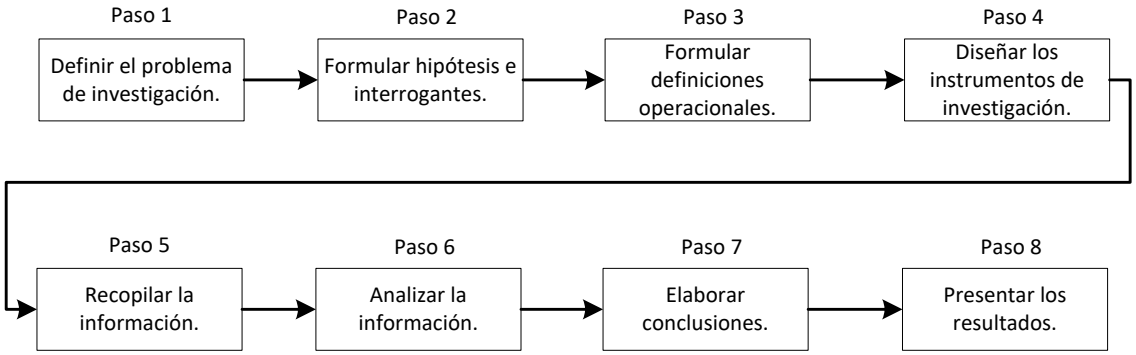


Figura 19. Patrón lineal de la investigación cuantitativa.
Fuente: Spradley (1980). Editada por autor

Paso 1: Definición del problema de investigación

El seguimiento o monitoreo se define como un ejercicio a identificar de manera sistemática la calidad del desempeño de un sistema, subsistema o proceso con el fin de introducir los ajustes o cambios pertinentes y oportunos para el logro de sus resultados y efectos en el entorno. De esta forma, el monitoreo permite analizar el estado actual de un sistema y propone acciones a tomar para lograr los objetivos, lo que se traduce en identificar estados reales o potenciales para hacer los ajustes oportunos a la ejecución (OIE - IDIE, 2010).

Respecto a infraestructuras, los sistemas de monitoreo permiten determinar el estado de sus variables a partir del uso de sensores. Los sensores son elementos físicos que pertenecen a un dispositivo denominado transductor y tienen la capacidad de transformar una variable física a otro tipo de variable (generalmente eléctrica). Los sensores captan las señales necesarias para conocer el estado de un proceso y de esta manera decidir un comportamiento a futuro (Velásquez, s.f.). Es así como en múltiples campos se hace uso ampliamente de los sensores como en la automatización, en la domótica y el ámbito del desarrollo, entre otros.

En el ámbito del desarrollo, los sensores se utilizan de acuerdo al contexto, por ejemplo, para agua y saneamiento se utilizan sensores en infraestructuras para los sistemas de recolección de aguas lluvia (cisternas) con el fin de medir el nivel de agua almacenamiento, en infraestructuras de sistemas fotovoltaicos se utilizan piranómetros para medir la irradiación sobre el panel y establecer la eficiencia en la producción energética, para aplicaciones de

cocinas mejoradas se utilizan sensores para medir la calidad del aire, entre cientos de aplicaciones.

Los sistemas de control o de almacenamiento de los datos para el monitoreo de variables generalmente se ubican en puntos centrales para el fácil acceso, es decir, no necesariamente se ubican cerca a los sensores por lo que se deben buscar estrategias para trasmitir la información de los sensores a los puntos de almacenamiento, recolección de datos o sistemas de monitoreo. Tradicionalmente se utilizan cables para la trasmisión de información de los sensores, solucionado un problema de comunicación siempre y cuando las distancias entre los sensores y el sistema de adquisición de datos sean cortas y si se utilizan los protocolos de comunicación adecuados para realizar esta tarea, como el RS485, CAN, RS232, entre otros. Sin embargo, algunos protocolos utilizan la técnica de aumentar los niveles de voltaje e incorporan circuitos adicionales que para aplicaciones del ámbito del desarrollo no resulta ser pertinente debido al aumento del consumo energético y aumento en los costos de implementación. Otro agravante corresponde en muchos casos a la limitante de expansión y modificaciones necesarias del hardware cuando se incorporan más sensores en los sistemas de monitoreo. Por lo que el uso de comunicación cableada entre los sensores y los sistemas de adquisición de datos se podría implementar en casos muy específicos y no siempre permite ser usa solución escalable.

Paso 2: Formular hipótesis e interrogantes.

Con el auge de las Tecnologías de la Información y las Comunicaciones en el siglo XXI, aparecen nuevas tendencias como lo es el Big Data y el Internet de las cosas (IoT o IdC). Internet no solo ha permitido conectar personas, sino dispositivos, sensores y complejos sistemas. Es así como aparece el término “Internet de las cosas” o “internet de los objetos” que corresponde a la interconexión de diversos dispositivos a internet, los cuales generan datos en tiempo real.

Lo importante del “internet de las cosas” no radica en la conexión de las “cosas” a internet, sino en la información y conocimiento que estas puedan aportar para mejorar la calidad de las personas.

Con la posibilidad de obtener información valiosa de diferentes elementos a través de internet, aparecen nuevas técnicas y protocolos de comunicación a bajo consumo; es el caso de *LoRaWAN*, un protocolo de comunicación inalámbrica de bajo consumo y largo alcance para IoT.

Sin embargo, *“La tecnología se crea en respuesta a las presiones del mercado y no de las necesidades de los pobres, que tienen escaso poder de compra… en la era de las redes, cada país necesita contar con capacidad para comprender las tecnologías mundiales y adaptarlas a las necesidades locales”*, PNUD (2001).

Los dispositivos LoRa (estándar LoRaWAN) son un claro ejemplo de lo expuesto por el PNUD en 2001; por tal razón se propone adaptarlos a confiables sistemas de monitoreo y a su vez a posibles aplicaciones del ámbito del desarrollo.

Por lo anteriormente presentado, es necesario plantear la preguntas: ¿Es posible adaptar los dispositivos LoRa como dispositivos de comunicación entre los sensores y los sistemas de adquisición de datos para aplicaciones del ámbito del desarrollo?, ¿qué condiciones y características deben tener los dispositivos LoRa para alcanzar su máximo rendimiento en términos consumo energético y alcance de transmisión?, ¿es posible crear un ruta guiada para el correcto uso de estos dispositivos?

Finalizado el proceso de investigación, se pretende demostrar o refutar la hipótesis: *"De acuerdo a las características de bajo consumo energético, bajo costo de adquisición, fácil configuración y alto alcance de transmisión, los dispositivos LoRa resultan ser idóneos para aplicaciones del ámbito del desarrollo como mecanismo de comunicación entre sensores y dispositivos de adquisición de datos"*

Paso 3: Formular definiciones operaciones.

Fase I: Búsqueda y documentación de los dispositivos propuestos.

En el apartado *Descripción de los dispositivos utilizados*, se presentan las principales características de los dispositivos propuestos entre los que se tiene:

- Características físicas y eléctricas de los transceiver LoRa con las versiones SX1272 y SX1278
- Definición de las bandas de frecuencia ISM (Industrial, Scientific and Medical).
- Descripción del protocolo SPI utilizado por los dispositivos LoRa.
- Definición y características de los sistemas embebidos.
- Importancia del hardware libre para el ámbito del desarrollo.
- Principales características de Arduino Uno y Arduino DUE.
- Principales características del módulo SDIN (desarrollado por GEDIRCI).
- Descripción de los IDE utilizados para la programación: Eclipse y Arduino IDE.
- Importancia del software libre para el ámbito del desarrollo.
- FreeRTOS: sistema operativo en tiempo real de código abierto.

Fase II: Desarrollo del Driver para el control de los dispositivos LoRa

El Anexo A presenta el *"Desarrollo del driver para el uso de los dispositivos de radio LoRa con sistemas embebidos"*. En el mencionado documento se presenta de forma detalla el método adecuado de configuración de los dispositivos LoRa (en el documento, ver apartado *"Configuración por software de los radio LoRa"*).

Fase III: Pruebas del driver para el control de los dispositivos LoRa

Para realizar las pruebas del driver en los dispositivos LoRa, fue utilizado el esquema presentado en la *Figura 20*. Se utilizaron transceivers compuestos por dispositivos LoRa (versiones SX1272 y SX1278) junto con sistemas embebidos como Arduino UNO, DUE y los módulos SDIN. Para determinar la pertinencia en el funcionamiento del driver desarrollado, se utilizaron instrumentos electrónicos, entre los que se tienen:

- *Osciloscopio:* útil para medir tiempos de transmisión y recepción de tramas.
- *Multímetro:* utilizado para medir la corriente consumida por el dispositivo cuando se encuentra trasmitiendo y recibiendo tramas de datos. Con la medición de la corriente, implícitamente es posible calcular el consumo energético para las diferentes configuraciones de SF, CR y BW en los dispositivos LoRa.
- *Equipo de cómputo:* se utilizó un equipos de cómputo con sistema operativo *Linux Ubuntu 14*, sobre los cuales se instalaron las versiones IDE de Arduino y eclipse para la programación y depuración del código fuente incorporado en los sistemas embebidos (configuración de parámetros lógicos SF, CR y BW).

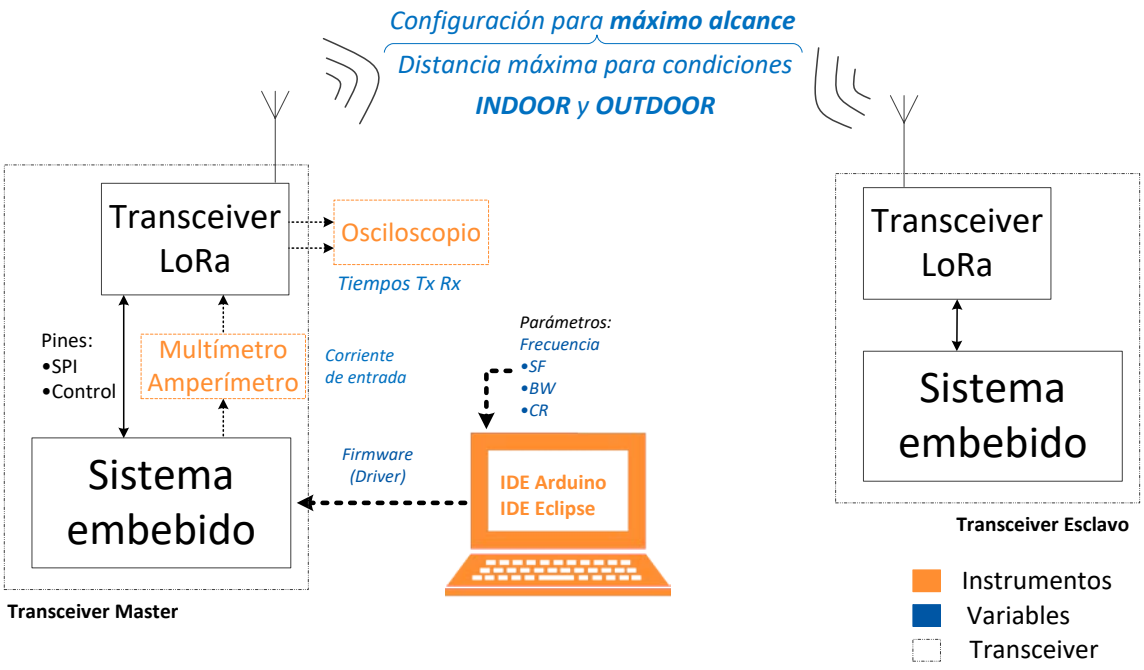


Figura 20. Esquema propuesto para pruebas del driver con los dispositivos LoRa.
Fuente: Esquema desarrollado por autor

Sin embargo, la *Figura 20* resulta ser muy general, por lo que se hace necesario presentar el esquema de conexión para los diferentes dispositivos LoRa y sistemas embebidos utilizados. La *Figura 21* presenta el esquema de conexión utilizado entre los dispositivos LoRa SX1272 y el sistema embebido Arduino Uno. La *Tabla 7* presenta la nomenclatura utilizada en el transceiver SX1272 y la tarjeta Arduino Uno.

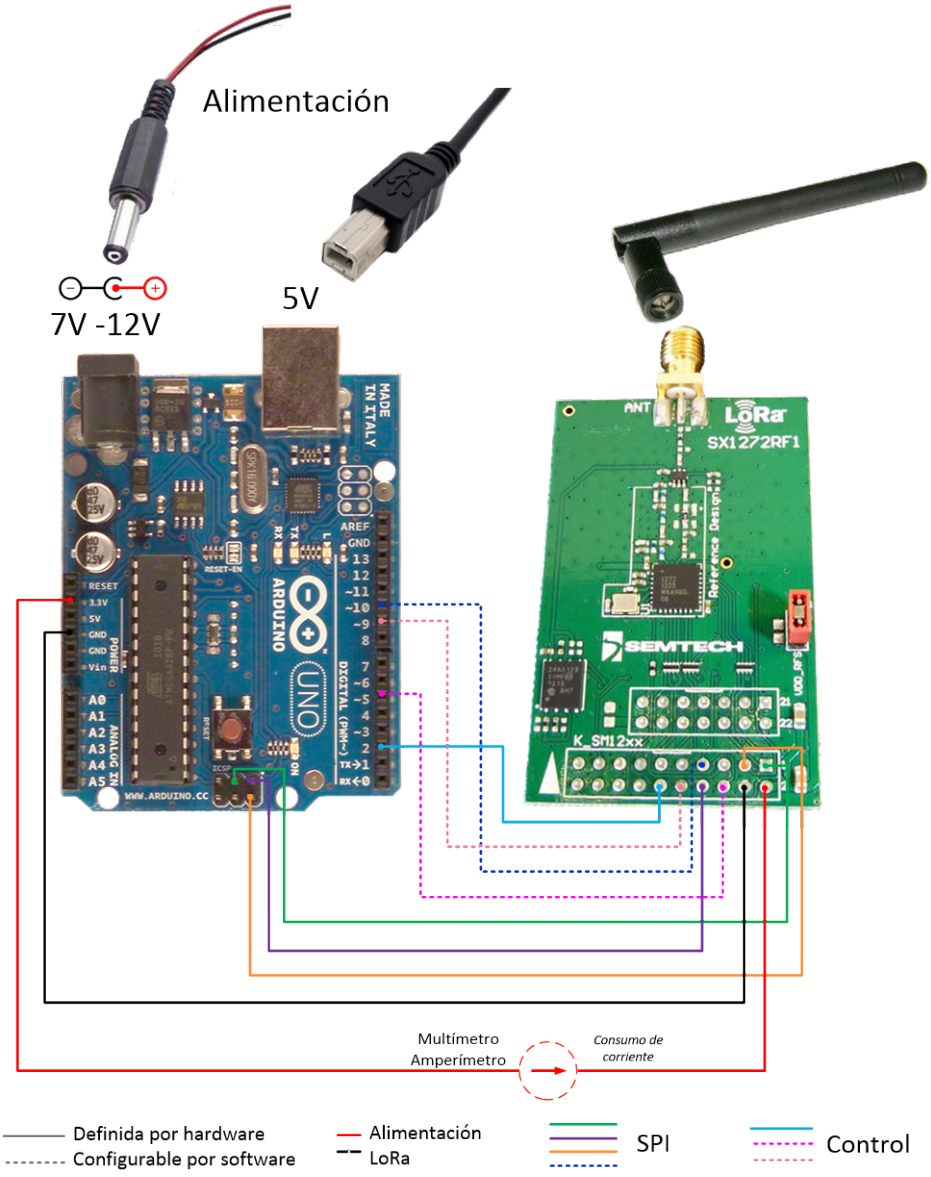


Figura 21. Esquema de conexión para SX1272 de Semtech y Arduino UNO
Fuente: Esquema propuesto por el autor

Tabla 7. Nomenclatura utilizada para la conexión entre la tarjeta SX1272RF1 de Semtech y la tarjeta Arduino UNO
Fuente: Semtech SX1272 (2015)

Tipo	Nomenclatura estándar	Notación en tarjetas	
		SX1272	Arduino UNO
Alimentación	VCC	2	3.3V
	GND	1	GND
SPI	NSS	7	10
	MOSI	3	11
	MISO	8	12
	SCK	1	13
CONTROL	RESET	10	9
	DIO0	12	2
	DIO3	6	5

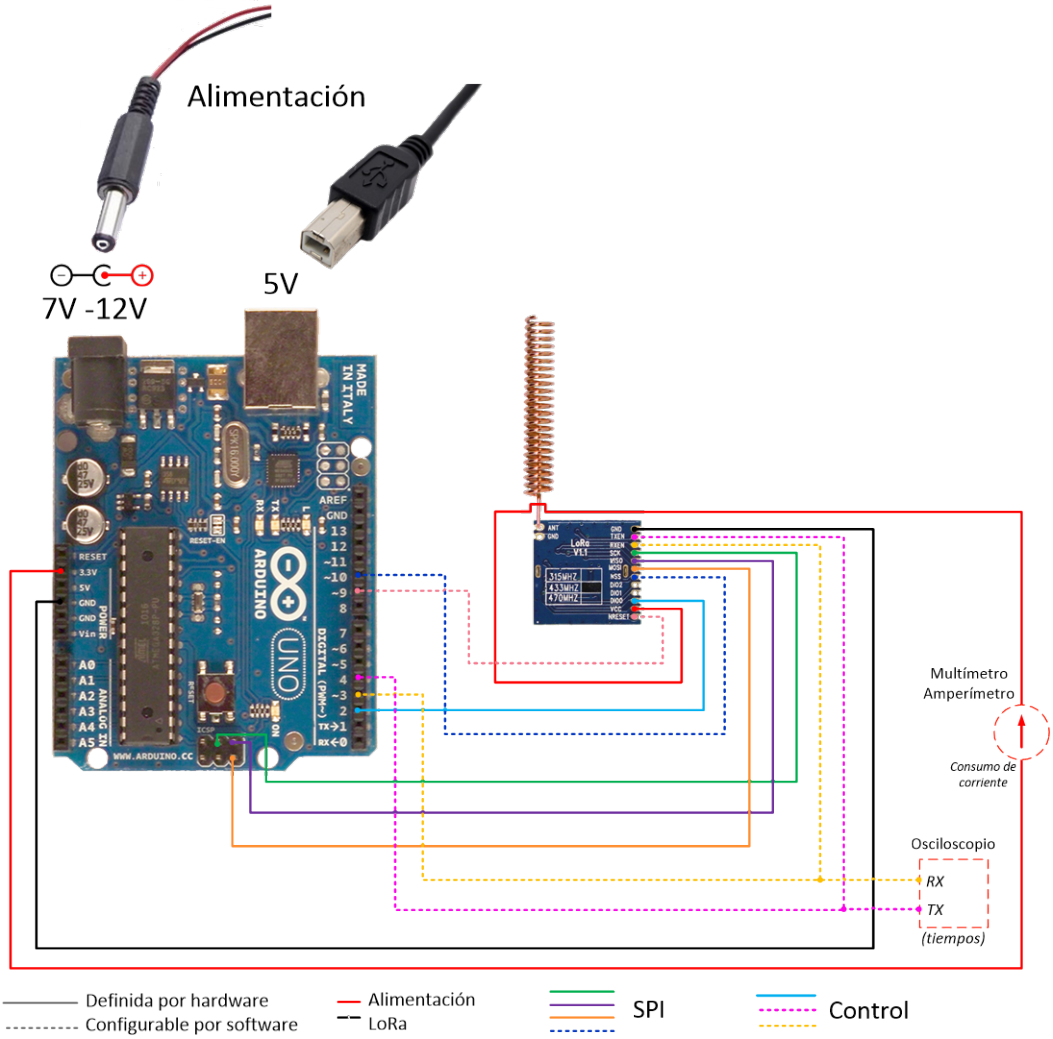


Figura 22. Esquema de conexión para LoRa1278 de NiceRF y Arduino UNO
Fuente: Esquema propuesto por el autor

Tabla 8. Nomenclatura utilizada para la conexión entre la tarjeta LoRa1278 de NiceRF y la tarjeta Arduino UNO
Fuente: NiceRF (2015)

Tipo	Nomenclatura estándar	Notación en tarjetas	
		LoRa1278	Arduino UNO
Alimentación	VCC	VCC	3.3V
	GND	GND	GND
SPI	NSS	NSS	10
	MOSI	MOSI	11
	MISO	MISO	12
	SCK	SCK	13
CONTROL	RESET	NRESET	9
	DIO0	DIO0	2
	TXEN	TXEN	3
	RXEN	RXEN	4

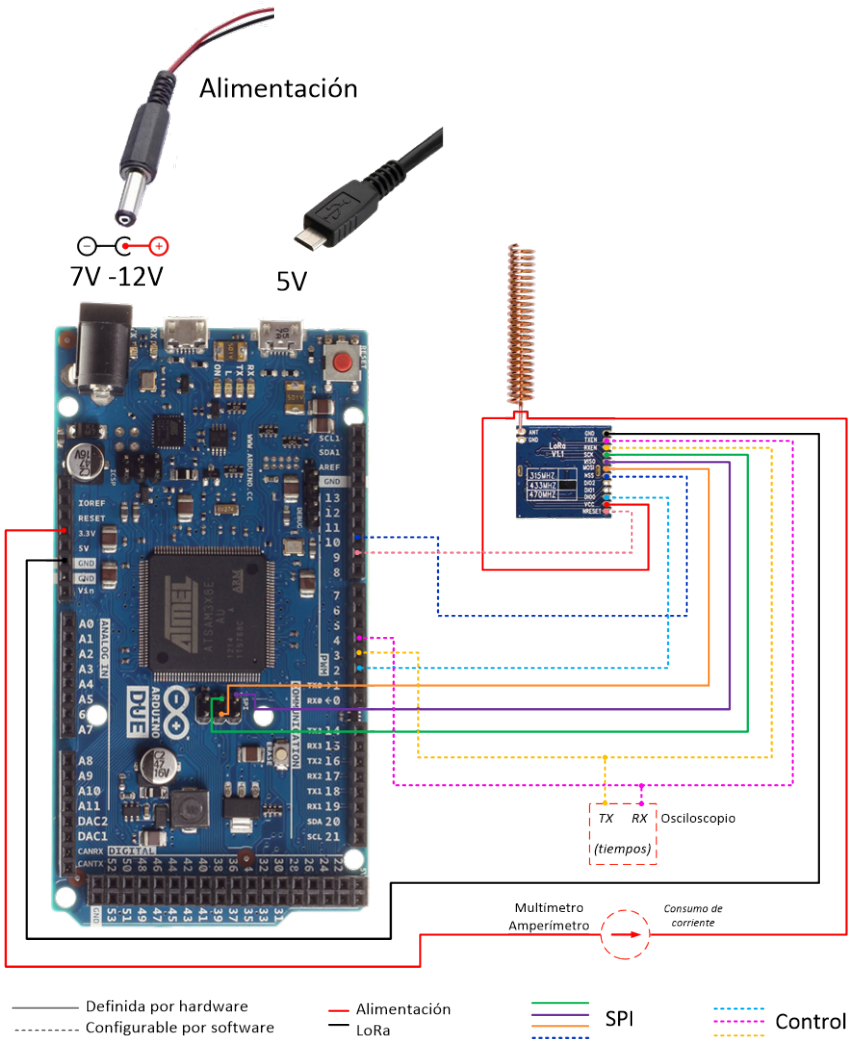


Figura 23. Esquema de conexión para LoRa1278 de NiceRF y Arduino DUE
Fuente: Esquema propuesto por el autor

Tabla 9. Nomenclatura utilizada para la conexión entre la tarjeta LoRa1278 de NiceRF y la tarjeta Arduino DUE
Fuente: NiceRF (2015)

Tipo	Nomenclatura estándar	Notación en tarjetas	
		LoRa1278	Arduino DUE
Alimentación	VCC	VCC	3.3V
	GND	GND	GND
SPI	NSS	NSS	10
	MOSI	MOSI	MOSI
	MISO	MISO	MISO
	SCK	SCK	SCK
CONTROL	RESET	NRESET	9
	DIO0	DIO0	2
	TXEN	TXEN	3
	RXEN	RXEN	4

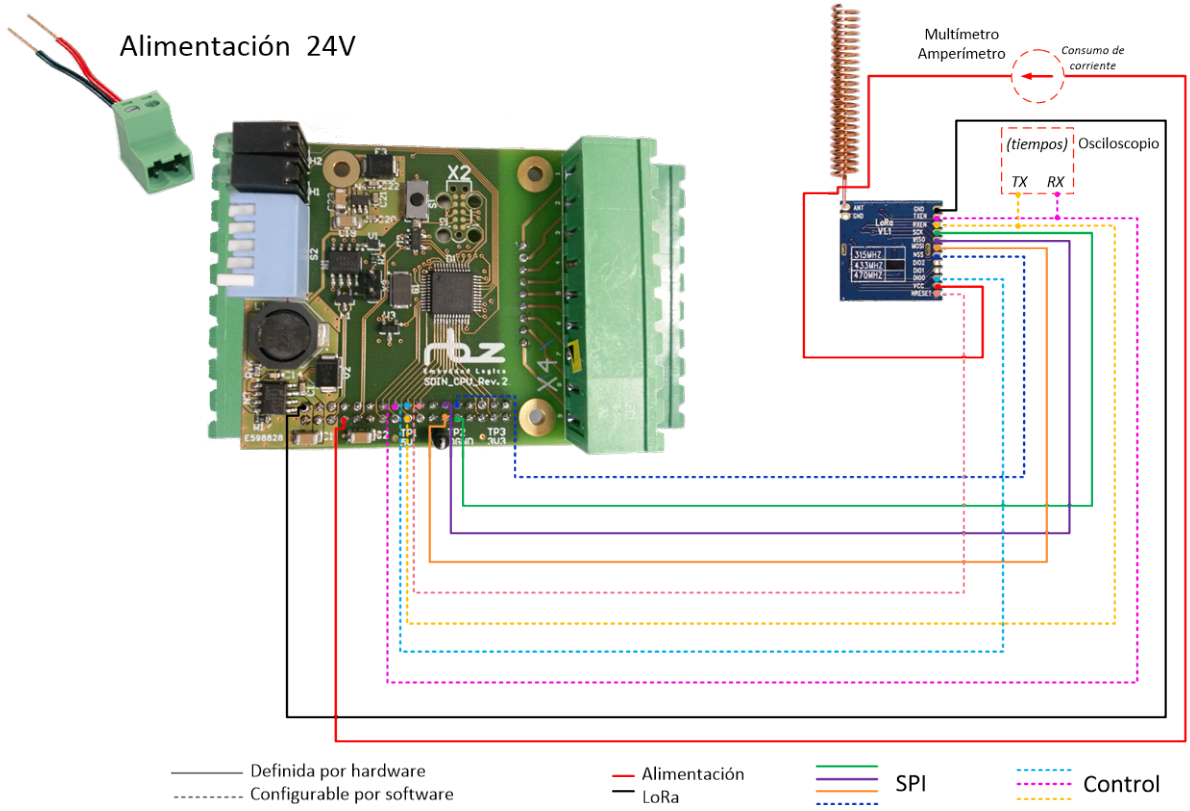


Figura 24. Esquema de conexión para LoRa1278 de NiceRF y módulo SDIN
Fuente: Esquema propuesto por el autor

Tabla 10. Nomenclatura utilizada para la conexión entre la tarjeta LoRa1278 de NiceRF y el módulo SDIN
Fuente: NiceRF (2015)

Tipo	Nomenclatura estándar	Notación en tarjetas	
		LoRa1278	SDIN
Alimentación	VCC	VCC	3.3V
	GND	GND	GND
SPI	NSS	NSS	PB12
	MOSI	MOSI	PB15
	MISO	MISO	PB14
	SCK	SCK	PB13
CONTROL	RESET	NRESET	PA7
	DIO0	DIO0	PA5
	TXEN	TXEN	PA3
	RXEN	RXEN	PA4

En las *Figura 21* (*Tabla 7*) a la *Figura 24* (*Tabla 10*) se presentan los esquemas de conexión en donde se puede apreciar los instrumentos utilizados para determinar las variables relacionadas con el consumo de corriente, utilizando un amperímetro conectado en serie entre el terminal de salida de voltaje de 3.3V del sistema embebido y el pin de alimentación del dispositivo LoRa; y los tiempos de trasmisión y recepción de tramas utilizando un osciloscopio. Con la corriente de consumo de corriente y los tiempos de trasmisión y recepción de datos es posible determinar el consumo energético en los modos trasmisión y recepción.

Las pruebas para los diferentes dispositivos se desarrollaron en las instalaciones de la Escuela Técnica Superior de Ingenieros de Telecomunicación ETSIT de la Universidad Politécnica de Madrid. Tanto los dispositivos como los instrumentos para el desarrollo del driver y para la ejecución de las pruebas fueron suministrados por GEDIRCI, explícitamente por el laboratorio B-301 del Departamento de Tecnología Fotónica y Bioingeniería.

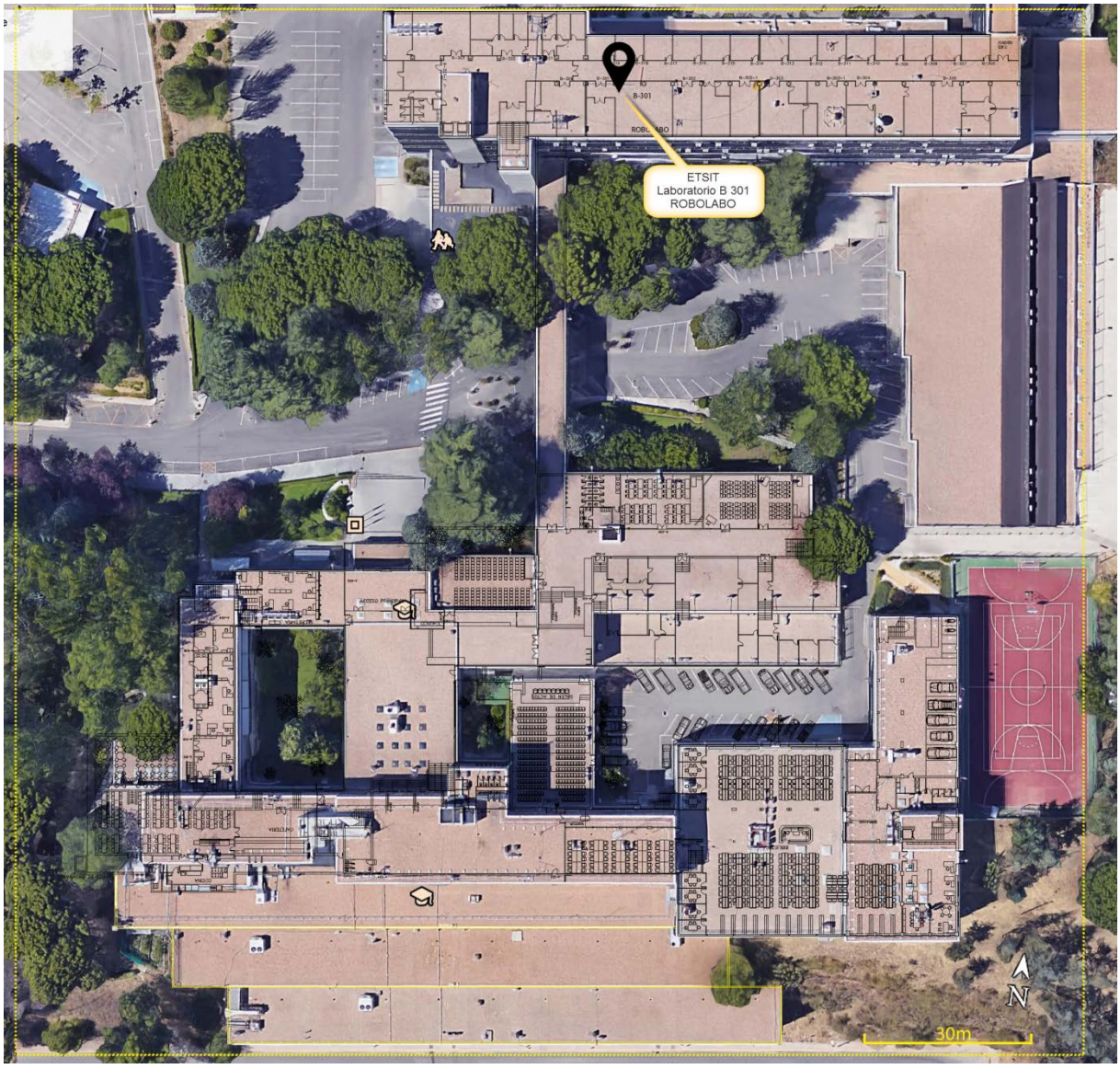


Figura 25. Espacio de pruebas Indoor para los dispositivos LoRa
Fuente: Google earth 7.1.5.1557 y planos compartidos por la ETSIT

Se desarrollaron pruebas de máximo alcance para espacio Indoor y Outdoor. Las pruebas de máximo alcance buscan encontrar la mejor configuración de los dispositivos para obtener la máxima distancia de trasmisión sin comprometer la veracidad de los datos, es decir que los datos trasmitidos y recibidos sean correctos y además que no se comprometan los consumos energéticos. Las pruebas de configuración a máximo alcance para espacios Indoor (con obstáculos como infraestructuras) se desarrollaron en las instalaciones de la ETSIT – UPM (ver *Figura 25*).

Las pruebas de configuración a máximo alcance para espacios Outdoor (con línea de vista u obstáculos como vegetación en espacios amplios) se desarrollaron en el *parque Dehesa de la Villa*, espacio muy próximo a las instalaciones de la ETSIT – UPM como se observa en la *Figura 26*.

Figura 26. Espacio de pruebas Outdoor para los dispositivos LoRa
Fuente: Google earth 7.1.5.1557

Fase IV: Pruebas del driver para una red tipo estrella

Finalmente se aumenta la complejidad del diseño creando una red estrella con un nodo central (circuito maestro) y 3 dispositivos esclavos. Como circuitos esclavos utilizan diversas versiones de sistemas embebidos. La
Figura 27 presenta el esquema propuesto donde se aprecia la comunicación simultánea de 3 dispositivos hacia un nodo central. Esta prueba demuestra la eficacia del driver desarrollado para aplicaciones complejas con diversos sensores ubicados en sitios remotos. Cabe resaltar que los dispositivos LoRa deben ser de la misma referencia, estar configurados a la misma frecuencia y con los parámetros SF, BW y CR idénticos.

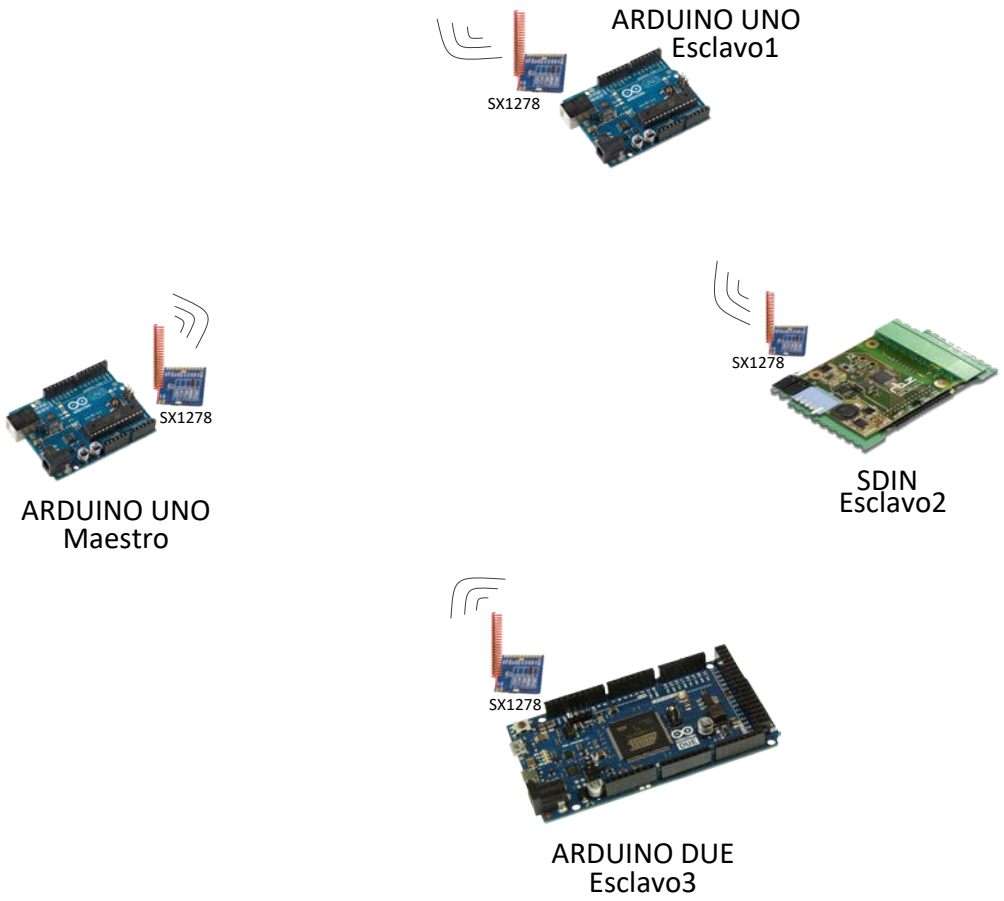


Figura 27. Tipología estrella para la prueba del driver generado
Fuente: Esquema propuesto por el autor

Paso 4: Diseñar los instrumentos de investigación

Una vez establecidas las definiciones operacionales, se proponen los instrumentos para recolectar información valida y confiable con el fin de corroborar o refutar la hipótesis planteada.

Según Rojas (1989), la información recolectada es clasificada en información primaria y secundaria: para la investigación, la información primaria corresponde a la información obtenida directamente de los instrumentos planteados por el investigador (ver *Tabla 11*); y la información secundaria se refiere a información extraída de fuentes documentales, que para el caso particular corresponde a hojas de especificaciones técnicas resumidas en el apartado *Transceiver Semtech LoRa*

.

La *Tabla 11* presenta el instrumento propuesto para determinar los consumos energéticos de acuerdo a diferentes configuraciones por software en lo transceivers LoRa. Este instrumento se utilizara para el dispositivo LoRa (SX1272 o SX1278) que presente el mayor alcance y estabilidad en la trasmisión y recepción de los datos.

Tabla 11. Instrumento propuesto para la recopilación de datos en las pruebas con dispositivos LoRa para diferentes configuraciones
Fuente: Propuesta por el autor

Parámetros configurables por software			Consumo de corriente mA			Tiempos segundos		Consumo energético mW•s	
SF	BW	CR	TX	RX	Sleep	TX	RX	TX	RX

Parámetros
- Configurables por software
- Medidos con multímetro
- Determinados por osciloscopio
- Calculados a partir de los datos

Paso 5: Recopilar la información.

La técnica utilizada para recopilar información corresponde a observaciones directas que corresponde al método más preciso para obtener información para todas las variables debido a obtención de datos cuantitativos por medio de la medición del fenómeno que se esté observando (Hernández, s.f.).

Acorde con la metodología de investigación, las variables utilizadas son de tipo cuantitativo, las cuales según las características o propiedades pueden presentarse en diversos grados o intensidad y tienen un carácter numérico o cuantitativo.
De acuerdo a las relaciones de causalidad, las variables se definen como independientes y dependientes. Las variables independientes son los parámetros configurables por software SF, BW y CR que según las combinaciones entre estos permiten determinar el alcance o distancia máxima de recepción de datos, la velocidad de los datos, sobrecarga por corrección o detección de errores en la información trasmitida y el consumo energético al trasmitir y recibir datos. Estos valores deben ser los mismos en todos los dispositivos radio LoRa de la red inalámbrica. Los parámetros independientes de configuración se definen como (Semtech SX1272, 2015 & Semtech SX1278, 2015):

BW: Ancho de banda (entre más bajo sea este valor, el tiempo de durante la trasmisión será mayor)

SF rate: Es el factor de alcance expresado como logaritmo de base 2. Entre mayor sea este valor, se tendrá un mejor rendimiento en la trasmisión de datos.

CR: Tasa de codificación de errores: Entre mayor sea este valor, mayor será la fiabilidad de los datos, pero con una sobrecarga en el tiempo de transmisión.

La *Tabla 12* presenta la combinación de los principales variables independientes para la investigación. Cabe aclarar que estos valores deben ser los mismos en todos los dispositivos radio LoRa de la red inalámbrica.

Tabla 12. Combinación de los principales parámetros configurables para los dispositivos LoRa
Fuente: Semtech SX1278 (2015) & Semtech SX1272 (2015)

Ancho de banda BW (kHz)		Factor de alcance SF		Tasa de codificación de errores. CR	
Hardware	Software **RADIO.BW**	Hardware	Software **RADIO.sf**	Hardware	Software **RADIO.cr**
500	BW500	6 → 64 chips / symbol (modo utilizado para FSK)	SF_6	4/5	CR_4_5
250	BW250	7 → 128 chips / symbol	SF_7	4/6	CR_4_6
125	BW125	8 → 256 chips / symbol	SF_8	4/7	CR_4_7
62.5*	BW62_5*	9 → 512 chips / symbol	SF_9	4/8	CR_4_8
41.7*	BW41_7*	10 → 1024 chips / symbol	SF_10		
31.25*	BW31_25*	11 → 2048 chips / symbol	SF_11		
20.8*	BW20_8*	12 → 4096 chips / symbol	SF_12		
15.6*	BW15_6*				
10.4*	BW10_4*				
7.8*	BW7_8*				

*: Aplica solamente para los transceiver SX1278

Los parámetros son configurados en el driver (Ver Anexo A: Desarrollo del driver para el uso de los dispositivos de radio LoRa con sistemas embebidos – apartado *"Registros de configuración SF, BW y CR del radio LoRa"*) mediante la definición de las variables *RADIO.sf*. *RADIO.bw* y *RADIO.cr*.
A partir de la combinación de los parámetros SF, BW y CR se realizan las mediciones de consumos de corriente cuando el dispositivo se encuentra trasmisión y recepción datos.

En las *Figura 21* (*Tabla 7*) a la *Figura 24* (*Tabla 10*) se presentan los esquemas de conexión en donde se puede apreciar los instrumentos utilizados para determinar las variables dependientes relacionadas con el consumo de corriente. El consumo de corriente se determina utilizando un amperímetro conectado en serie entre el terminal de salida de voltaje de 3.3V del sistema embebido y el pin de alimentación del dispositivo LoRa; y los tiempos de trasmisión y recepción de tramas son determinados utilizando un osciloscopio. Con la información de consumo de corriente y los tiempos de trasmisión y recepción de datos es posible determinar el consumo energético en los modos trasmisión y recepción. El modelo matemático para determinar la potencia necesaria para trasmitir y recibir datos está dada por *(1)*:

$$P_E = V_{CC} \cdot I_{CC} \tag{1}$$

Dónde:
P_E: Potencia eléctrica necesaria para trasmitir o recibir datos. Se presenta en vatios.

V_{CC}: Voltaje nominal de alimentación. Según las especificaciones técnicas de los dispositivos LoRa corresponde a 3.3V de voltaje constante. Esta variable será constante o invariante.

I_{CC}: Corriente consumida cuando el dispositivo trasmite o recibe datos. El valor máximo de corriente consumida puede ser configurado para 100mA o 240mA. Para el caso de las pruebas, se plantea un máximo de consumo de hasta 100mA. Esta variable será adquirida con el uso de un amperímetro según las *Figura 21* a la *Figura 24*.

Sin embargo, la potencia también se define como la razón de cambio del trabajo o energía por unidad de tiempo (Chapman, 2012), como se observa en la ecuación *(2)*:

$$P_E = \frac{W}{t} \tag{2}$$

Dónde:
W: corresponde a la energía.
P_E: Potencia eléctrica.
t: Tiempo en segundos.

De acuerdo a *(2)* se define la energía como:

$$W = P_E \cdot t \tag{3}$$

Por lo que se concluye que la energía está dada como la potencia por unidad de tiempo. Para determina la energía necesaria para trasmitir y recibir datos en los dispositivos LoRa, se utilizan los modelos *(1)* y *(3)*:

$$W = (V_{CC} \cdot I_{CC}) \cdot t \quad (4)$$

$$W_{Tx} = (3.3 \cdot I_{Tx}) \cdot t_{Tx} \quad (5)$$

$$W_{Tx} = 3.3 \cdot I_{Tx} \cdot t_{Tx} \quad (6)$$

$$W_{Rx} = 3.3 \cdot I_{Rx} \cdot t_{Rx} \quad (7)$$

Según las ecuaciones *(6)* y *(7)*, a partir de la corriente eléctrica necesaria para trasmitir y recibir datos (utilizando un amperímetro) y los tiempos necesarios para trasmitir y recibir información, se establecen los valores de energía necesaria para trasmitir y para recibir datos del instrumento de la *Tabla 11*.

Paso 6: Analizar la información.

El instrumento de la *Tabla 11* diligenciado será presentado y analizado en el apartado *RESULTADOS*, donde se determinara las características de consumo energético según la configuración de los parámetros SF, BW y CR. Además se presentan los resultados de las pruebas de máximo alcance para espacios *Indoor* como *Outdoor* (ver Figura 25 y Figura 26).

Paso 7: Elaborar conclusiones.

A partir del apartado *RESULTADOS*, se desarrollan las *CONCLUSIONES* junto con comentarios y recomendaciones a partir de la observación y uso de los dispositivos LoRa con los sistemas embebidos Arduino y el módulo SDIN.

Paso 8: Presentar los resultados.

Los resultados se presentan en el apartado *RESULTADOS* . Estos resultados se basan en los anexos creados: Anexo A: "*Desarrollo del driver para el uso de los dispositivos de radio LoRa con sistemas embebidos*" y Anexo B: "*Instalación y uso del driver LoRa para sistemas embebidos*". El driver y algunos ejemplos son presentados en el espacio GitHub del grupo de investigación GEDIRCI con el fin de compartir la información de forma gratuita y colaborar con diferentes desarrolladores (ver https://github.com/Robolabo). Finalmente, se realiza un video resumen de la investigación desarrollada.

4. RESULTADOS

4.1 Pruebas de alcance y consumo energético

De acuerdo con la *Metodología*, en el apartado "*Fase III: Pruebas del driver para el control de los dispositivos LoRa"* se plantea el uso de dos sistemas embebidos, uno utilizado como maestro y el otro utilizado como esclavo.

El dispositivo maestro se encarga de realizar la petición de datos a un dispositivo esclavo ubicado en un sitio remoto, el cual obtiene datos de sensores. El esclavo remite la información solicitada y continuamente se encuentra esperando peticiones del dispositivo maestro. El flujograma y código fuente empleado para la prueba se encuentra en el "*ANEXO A Desarrollo del driver para el uso de los dispositivos de radio LoRa con sistemas embebidos*".
En la *Figura* 28 se presenta un diagrama esquemático de la aplicación propuesta: maestro-esclavo (sensor) con la utilización de dispositivo LoRa para la realización de las pruebas de alcance (prueba "*ping-pong*").

Figura 28. Principio de funcionamiento "ping-pong" para la realización de pruebas de alcance en dispositivos LoRa
Fuente: Desarrollado por autor

Básicamente esta prueba consiste en él envió continuo de 3 Bytes codificados desde el dispositivo maestro para realizar la petición de datos. Una vez el esclavo recibe estos 3 Bytes, captura los datos de un sensor, los codifica en 5 Bytes y los tramite hacia el maestro. En caso que el maestro detecte información errada (utilizando el CDC o verificación por redundancia cíclica), vuelve a solicitar la información al dispositivo esclavo.

Para realizar las pruebas de máximo alcance, se realizaron desplazamientos entre los edificios A y B de la Escuela Técnica Superior de Ingenieros en Telecomunicación - ETSIT. El dispositivo maestro permaneció en el edificio B, laboratorio B-301 de GEDIRCI; mientras que con el dispositivo esclavo se realizaron los desplazamientos entre el edificio B y el edificio A de la ETSIT.

Una vez se logró establecer el enlace de comunicación entre ambos dispositivos, se realizó la comprobación en tiempo real de los datos trasmitidos/recibidos y la detección de errores utilizando el depurador (*debugger*) en el IDE Arduino (para sistemas embebidos Arduino) y en el IDE Eclipse (para el módulo SDIN). Cabe aclara que ambos dispositivos (maestro y esclavo) contaron con configuraciones idénticas para mantener la comunicación.

Durante el proceso se cambió la configuración de los principales parámetros para los dispositivos LoRa. Estos parámetros corresponden a SF, BW y CR cuyas opciones se presenta en la *Tabla 12*.

4.1.1 Pruebas de alcance para LoRa SX1272

La configuración más estable y de mayor alcance para los dispositivos SX1272 LoRa detectada a partir de pruebas de comunicación, se presenta en la *Tabla 13*.

Tabla 13. Parámetros de configuración por software en los dispositivos SX1272 para obtener máximo alcance
Fuente: Información recolectada por el autor

Frecuencia Freq (Mhz)		Ancho de banda BW (kHz)		Factor de alcance SF		Tasa de codificación de errores. CR	
Hardware	Software **RADIO.freq**	Hardware	Software **RADIO.BW**	Hardware	Software **RADIO.sf**	Hardware	Software **RADIO.cr**
869.525	869525000	125	BW125	12 → 4096 chips / symbol	SF_12	4/5	CR_4_5

De acuerdo a los parámetros de la *Tabla 13*, se encuentra lo siguiente:

- Entre menor sea el BW, mayor tiempo tomara la comunicación y esto se debe a que la frecuencia es inversamente al tiempo. Sin embargo entre menor sea la frecuencia, mayor será el alcance de trasmisión esperado. El valor de BW125 corresponde al mínimo valor posible para el SX1272.
- El valor de SF o factor de alcance determina el rendimiento en la trasmisión de los datos, es decir que entre mayor sea este valor, el dispositivo tendrá menor probabilidad de recibir datos errados y mayor radio de cobertura.
- El CR asegura la fiabilidad de los datos. Entre mayor sea este valor, mayor será la fiabilidad de los datos, pero con una sobrecarga en el tiempo de transmisión.

La configuración presentada en la *Tabla 13*, asegura el máximo alcance y una fiabilidad de la información trasmitida/recibida razonable. Sin embargo estas características favorables para la comunicación, implican un tiempo de trasmisión mayor y por ende un mayor consumo energético.

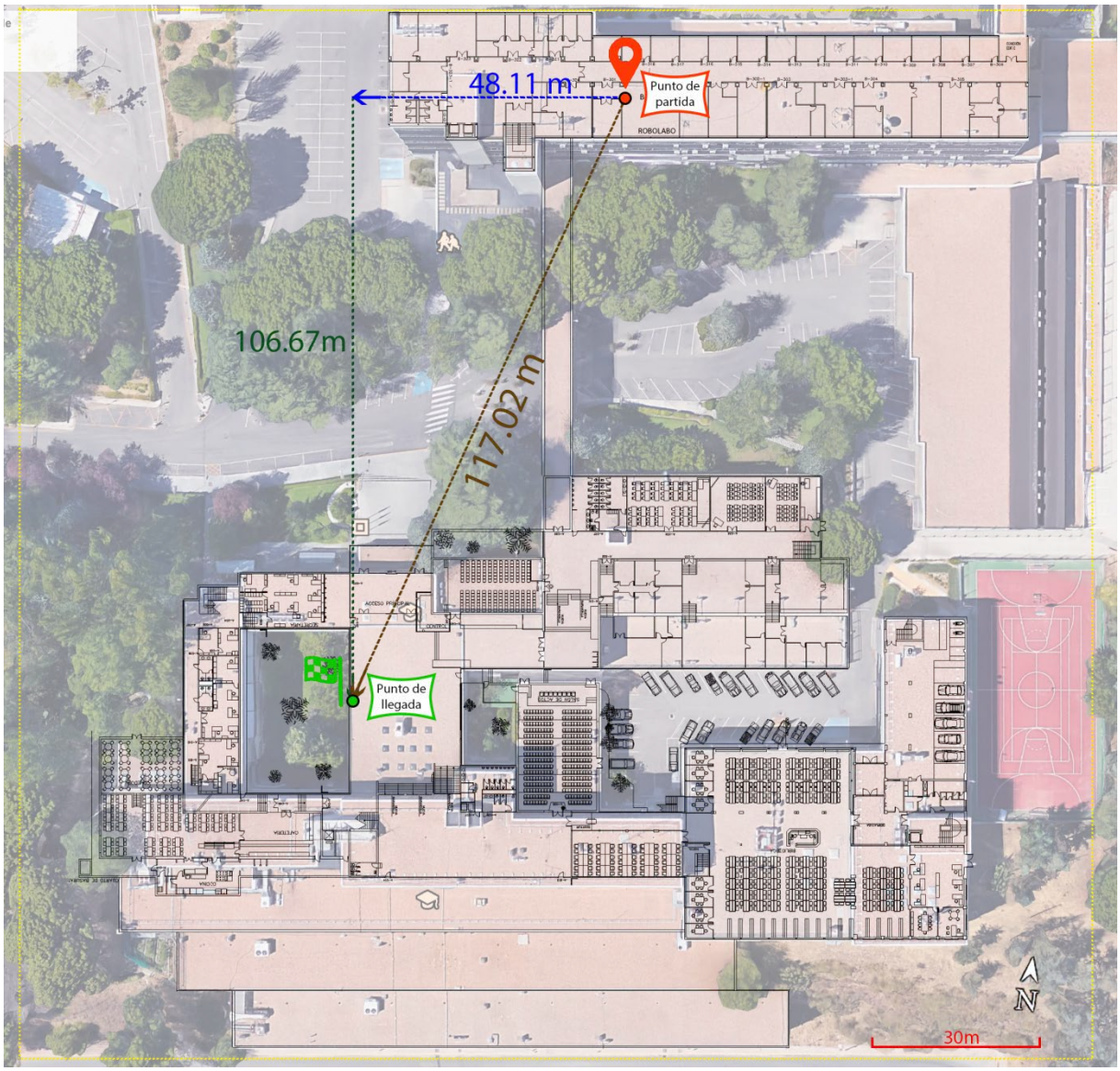


Figura 29. Máximo alcance en trasmisión/recepción de datos para LoRa SX1272 (plano XY)
Fuente: Google earth 7.1.5.1557 y planos compartidos por la ETSIT. Escala 1:30

En términos de distancia o máximo alcance para trasmisión y recepción *Indoor*, se realizaron los cálculos a partir de planos de las instalaciones de la Escuela técnica Superior de Ingenieros en Telecomunicación (compartidos por el tutor académico). El punto de partida fue el laboratorio B-301 y el punto de máximo alcance o final corresponde al intermedio del salón del acceso principal del edificio A de la ETSIT.

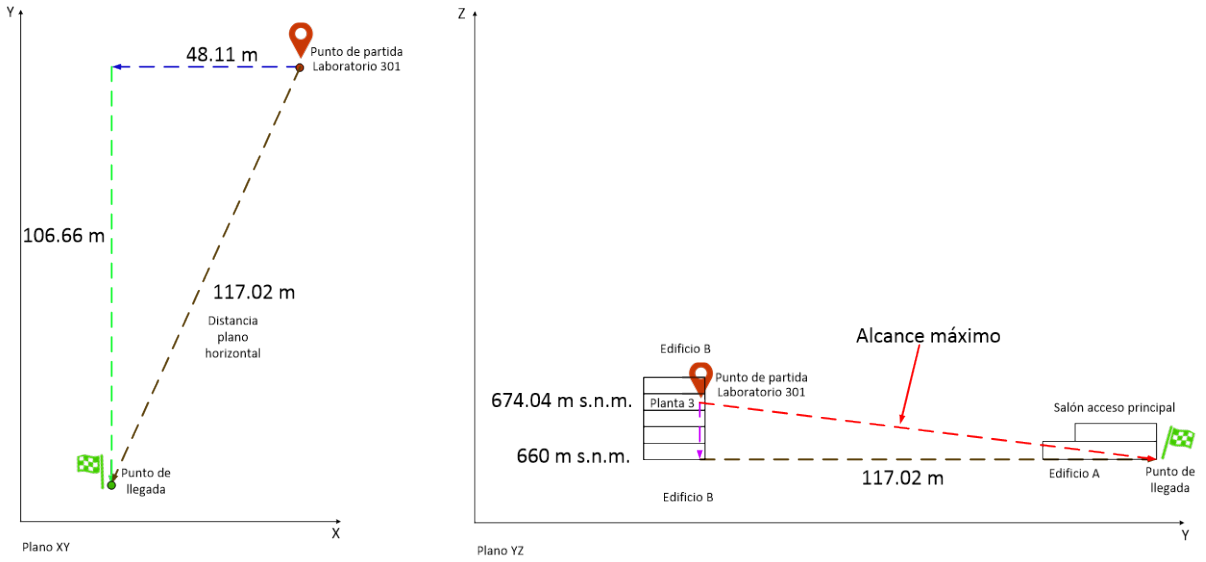


Figura 30. Esquema para determinar el alcance en trasmisión/recepción de datos para LoRa SX1272 (plano XY y YZ)
Fuente: Información del autor y planos compartidos por la ETSIT

La *Figura 30* presenta los plano XY y YZ referencia para realizar el cálculo del máximo alcance en las instalaciones de la ETSIT. Para determinar la distancia desde el punto de partida hasta el punto de llegada o de máximo alcance, se realizan cálculos a partir de teoremas de triangulo rectángulos. En la *Figura 30* (izquierda), se presenta la máxima distancia alcanzada con referencia del suelo o plano horizontal, esta distancia se calculó utilizando el teorema de Pitágoras con referencia a las distancias de los ejes X (48.11m) y Y (106.66m).
Para determinar el máximo alcance con referencia a los transceivers, se utiliza la *Figura 30* (derecha). La diferencia de altura entre los dispositivos se calcula utilizando el siguiente modelo:

$$h_{entre\ dispositivos} = h_{dispositivo\ maestro} - h_{dispositivo\ esclavo} \quad (8)$$

$$h_{entre\ dispositivos} = 674.04 - 660 \quad (9)$$

$$h_{entre\ dispositivos} = 14.04\ \text{m} \quad (10)$$

Finalmente, utilizando el teorema de Pitágoras se calcula la distancia de alcance máximo lograda por la configuración presentada en la *Tabla 13* por el dispositivo LoRa SX1272:

$$distancia_{máx\ alcance} = \sqrt{h_{entre\ dispositivos}{}^2 + distancia_{referencia\ suelo}{}^2} \quad (11)$$

$$distancia_{máx\ alcance} = \sqrt{14.04^2 + 117.02^2} \quad (12)$$

$$distancia_{máx\ alcance\ SX1272} = 117.86\ \text{m} \quad (13)$$

El máximo alcance logrado para los dispositivos SX1272 con la configuración que permite obtener el mejor rendimiento en términos de alcance (*Tabla 13*) es de 118 metros en aplicaciones *Indoor*.

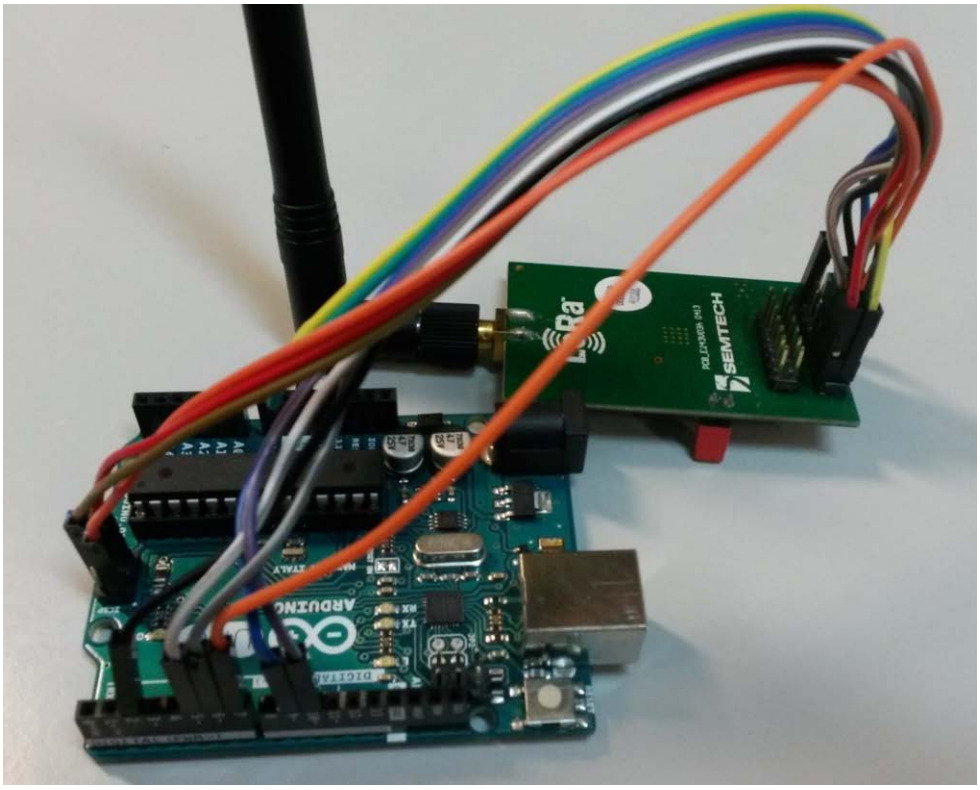


Figura 31. Dispositivo esclavo conformado por sistema embebido Arduino Uno y transceiver SX1272
Fuente: Fotografía tomada por autor

4.1.2 Pruebas de consumo energético y alcance para LoRa SX1278

En el caso del LoRa SX1278, se utilizaron dos fabricantes de tarjetas que realizan adecuaciones sobre el chip para facilitar la manipulación. Las referencias de las tarjetas utilizadas son las *DRF1278F de Dorji* y *LoRa1278 de NiceRF* (ver *Tabla 5*). Para cada tarjeta se realizó la comprobación de consumos energéticos a diferentes configuraciones con la frecuencia de 433.800Mhz.

De acuerdo con la *Metodología*, en el apartado "*Paso 4: Diseñar los instrumentos de investigación"* se presentaron los instrumentos planteados para determinar la relación entre los parámetros configurables por software de los dispositivos LoRa y el consumo energético.

4.1.2.1 Pruebas de consumo energético para DRF1278F de Dorji

La *Tabla 14* presenta la recopilación de las pruebas de consumo energético a diferentes configuraciones para los dispositivos *DRF1278F de Dorji*. El cálculo del consumo energético se presenta en el apartado "*Paso 5: Recopilar la información"*, con el modelo *(7)*.

Tabla 14. Instrumento con la recopilación de datos para las pruebas con los dispositivos LoRa para diferentes configuraciones en DRF1278F de Dorji
Fuente: Información recolectada por el autor

# Conf	Parámetros configurables por software			Consumo de corriente mA					Tiempos segundos		Consumo energético mili-Vatios·s		Comentarios
	SF	BW	CR	I_{TXmin}	I_{TX}	I_{TXmax}	I_{RX}	I_{Sleep}	t_{TX}	t_{RX}	W_{TX}	W_{RX}	
1	9	500	4_5	50	52,5	55	11,5	1,7	0,026	0,035	4,50	1,33	Estable
2	12	500	4_5	25	57,5	90	11,5	1,7	0,21	0,22	39,85	8,35	Estable
3	9	125	4_5	38	54	70	11,7	1,7	0,105	0,13	18,71	5,02	Estable
4	12	125	4_5	80	95	110	12	1,7	0,8	0,9	250,80	35,64	Estable
5	9	61,5	4_5	30	62,5	95	11,7	1,7	0,205	0,23	42,28	8,88	Estable
6	12	61,5	4_5	84	98,5	113	11,2	1,7	1,6	1,8	520,08	66,53	Estable
7	9	41,7	4_5	60	82	104	11	1,7	0,3	0,48	81,18	17,42	Estable
8	**12**	**41,7**	**4_5**	79	**96**	113	**11,1**	**1,7**	**2,5**	**2,6**	**792,00**	**95,24**	**Estable**
9	9	31,5	4_5	93	101	109	11,9	1,7	0,4	0,505	133,32	19,83	Estable
10	**12**	**31,5**	**4_5**	113	**113**	113	**11,9**	**1,7**	**3,5**	**4**	**1305,1**	**157,0**	**Estable**
11	9	20,8	4_5	99	106	113	11,6	1,7	0,75	0,8	262,35	30,62	Estable
12	7	20,8	4_5	28	60	92	11	1,7	0,5	0,6	99,00	21,78	Estable
13	9	7,8	4_5	113	113	113	11,7	1,7	X	X	X	X	Inestable

#Conf: Número de configuración
Parámetros
- Configurables
- Medidos con multímetro (multitester)
- Determinados por osciloscopio
- Calculados a partir de los datos

De acuerdo a los resultados presentados en la *Tabla 14*, la configuración SF=12, BW=32.5 y CR=4/5 (ver configuración # **10**), cuenta con un significativo consumo energético 1305 mW·s para trasmitir 3 Bytes y de 157 mW·s para recibir 5 Bytes. Si bien es la configuración que logra establecer el máximo alcance debido al que el parámetro BW es el más bajo, en la práctica se evidenciaron problemas por trasmitir/recibir gran cantidad de datos errados. Por las anteriores razones, se establece el máximo rendimiento en términos de alcance con la configuración # **8**: SF=12, BW=41.7 y CR=4/5.

4.1.2.2 Pruebas de consumo energético para LoRa1278 de NiceRF

La *Tabla 15* presenta la recopilación de las pruebas de consumo energético a diferentes configuraciones para los dispositivos *LoRa1278 de NiceRF*. El cálculo del consumo energético se presenta en el apartado "*Paso 5: Recopilar la información.*, con el modelo *(7)*.

Tabla 15. Instrumento con la recopilación de datos para los dispositivos LoRa a diferentes configuraciones en LoRa1278 de NiceRF
Fuente: Información recolectada por el autor

# Conf	Parámetros configurables por software			Consumo de corriente mA					Tiempos segundos		Consumo energético mili-Vatios·s		Comentarios
	SF	BW	CR	I_{TXmin}	I_{TX}	I_{TXmax}	I_{RX}	I_{Sleep}	t_{TX}	t_{RX}	W_{TX}	W_{RX}	
1	9	500	4_5	45	46	47	11,5	1,7	0,025	0,03	3,80	1,14	Estable
2	12	500	4_5	25	52,5	80	11,5	1,7	0,21	0,23	36,38	8,73	Estable
3	9	125	4_5	34	47	60	11,7	1,7	0,105	0,13	16,29	5,02	Estable
4	12	125	4_5	37	58,5	80	11,7	1,7	0,8	0,9	154,44	34,75	Estable
5	9	61,5	4_5	20	40	60	11,7	1,7	0,205	0,26	27,06	10,04	Estable
6	12	61,5	4_5	80	85	90	11,2	1,7	1,7	1,8	476,85	66,53	Estable
7	9	41,7	4_5	40	60	80	11	1,7	0,3	0,4	59,40	14,52	Estable
8	**12**	**41,7**	**4_5**	80	**85**	90	**11**	**1,7**	**2,4**	**2,6**	**673,20**	**94,38**	**Estable**
9	9	31,5	4_5	50	65	80	11,5	1,7	0,4	0,52	85,80	19,73	Estable
10	**12**	**32,5**	**4_5**	80	**80**	80	**11**	**1,7**	**3,5**	**4**	**924,00**	**145,2**	**Estable**
11	9	20,8	4_5	60	75	90	11	1,7	X	X	X	X	Inestable
12	7	20,8	4_5	30	45	60	11	1,7	X	X	X	X	Inestable

#Conf: Número de configuración
Parámetros
- Configurables
- Medidos con multímetro (multitester)
- Determinados por osciloscopio
- Calculados a partir de los datos

De acuerdo a los resultados presentados en la *Tabla 15*, la configuración SF=12, BW=32.5 y CR=4/5 (ver configuración # **10**), cuenta con un significativo consumo energético 924 mW·s para trasmitir 3 Bytes y de 145.2 mW·s para recibir 5 Bytes. Si bien es la configuración que logra establecer el máximo alcance debido al que el parámetro BW es el más bajo, en la práctica se evidenciaron problemas por trasmitir/recibir gran cantidad de datos errados. Por las anteriores razones, se establece el máximo rendimiento en términos de alcance con la configuración # **8**: SF=12, BW=41.7 y CR=4/5

4.1.2.3 Comparación de consumo energético entre DRF1278F de Dorji y LoRa1278 de NiceRF

De acuerdo a la *Tabla 14* y *Tabla 15*, se establece un bajo consumo en corriente de 1.7 mA cuando los transceivers se encuentran en modo SLEEP para las dos versiones de tarjetas SX1278. Se puede apreciar que el consumo de corriente cuando el transceiver se encuentra modo trasmisión I_{TX} es superior para la versión *DRF1278F de Dorji* e incluso supera los umbrales de los 100mA. Además, el consumo de corriente se encuentra entre los 11 a 12 mA cuando el transceiver se encuentra en modo recepción para ambas versiones. También se puede observar que no existen diferencias significativas en los tiempos de trasmisión y recepción entre las versiones *DRF1278F* y *LoRa1278*.

La *Figura 32* representa la comparación de configuraciones y consumos energéticos entre las versiones *DRF1278F* y *LoRa1278* cuando los transceivers se encuentran en modo recepción. Se puede observar que no existen diferencias significativas en los consumos energéticos entre las dos versiones para el modo recepción.

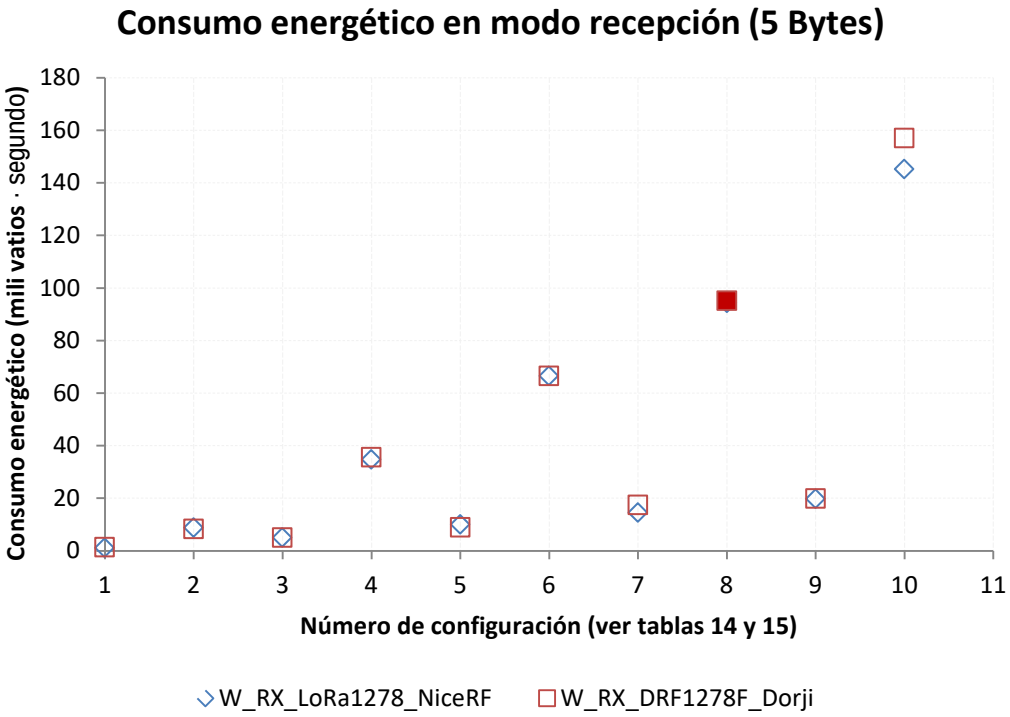


Figura 32. Representación gráfica del consumo energético de los transceivers DRF1278F y LoRa1278 para modo recepción
Fuente: Información recolectada por el autor

La *Figura 33* representa la comparación de configuraciones y consumos energéticos entre las versiones *DRF1278F* y *LoRa1278* cuando los transceivers se encuentran en modo transmisión. Se puede apreciar que el dispositivo *DRF1278F* cuenta con un mayor consumo de corriente con respecto al dispositivo *LoRa1278*. Esta característica se hace más evidente en la configuración # **8** y # **10** (ver *Tabla 14* y *Tabla 15*) que corresponde a los parámetros donde se obtiene el máximo alcance.

Por lo anterior, se establece un mejor rendimiento en términos del consumo energético por parte de la versión *LoRa1278* de NiceRF y se continúa con esta versión para realizar las pruebas de máximo alcance en condiciones *Indoor* y *Outdoor*.

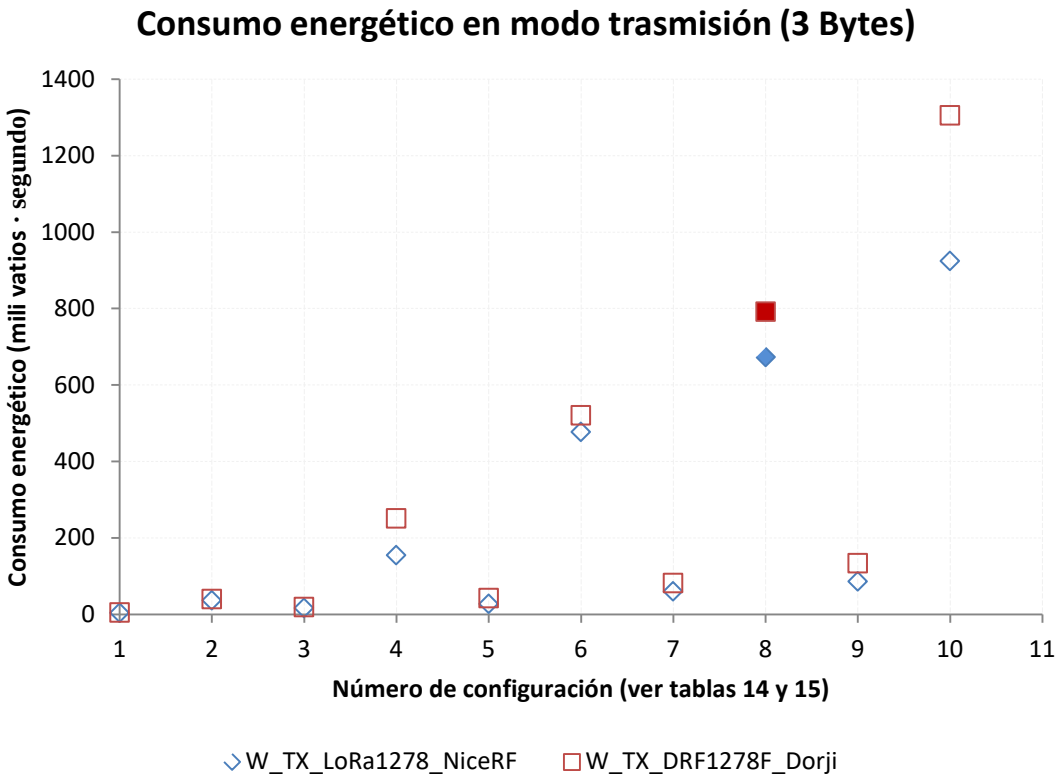


Figura 33. Representación gráfica del consumo energético de los transceivers DRF1278F y LoRa1278 para modo transmisión
Fuente: Información recolectada por el autor

4.1.2.4 Pruebas de alcance para LoRa SX128 (LoRa1278 de NiceRF)

La configuración más estable y de mayor alcance para los dispositivos LoRa1278 de NiceRF detectada a partir de "*Pruebas de consumo energético para LoRa1278 de NiceRF*", se presenta en la *Tabla 16* .

Tabla 16. Parámetros de configuración por software en los dispositivos SX1278 para obtener máximo alcance
Fuente: Información recolectada por el autor

Frecuencia Freq (Mhz)		Ancho de banda BW (kHz)		Factor de alcance SF		Tasa de codificación de errores. CR	
Hardware	Software **RADIO.freq**	Hardware	Software **RADIO.BW**	Hardware	Software **RADIO.sf**	Hardware	Software **RADIO.cr**
433.800	433800000	41.7	BW41_7	12 → 4096 chips / symbol	SF_12	4/5	CR_4_5

De acuerdo a los parámetros de la *Tabla 16*, se encuentra lo siguiente:

- Entre menor sea el BW, mayor tiempo tomara la comunicación y esto se debe a que la frecuencia es inversamente al tiempo. Sin embargo entre menor sea la frecuencia, mayor será el alcance de trasmisión esperado.
- El valor de SF o factor de alcance determina el rendimiento en la trasmisión de los datos, es decir que entre mayor sea este valor, el dispositivo tendrá menor probabilidad de recibir datos errados y mayor radio de cobertura.
- El CR asegura la fiabilidad de los datos. Entre mayor sea este valor, mayor será la fiabilidad de los datos, pero con una sobrecarga en el tiempo de transmisión.

La configuración presentada en la *Tabla 16*, asegura el máximo alcance y una fiabilidad de la información trasmitida/recibida razonable. Sin embargo estas características favorables para la comunicación, implican un tiempo de trasmisión mayor y por ende un mayor consumo energético.

4.1.2.4.1 Pruebas de alcance "Indoor" para LoRa SX128 (LoRa1278 de NiceRF)

En términos de distancia o máximo alcance para trasmisión y recepción de datos *Indoor*, se realizaron los cálculos a partir de planos de las instalaciones de la Escuela Técnica Superior de Ingenieros en Telecomunicación (compartidos por el tutor académico). El punto de partida fue el laboratorio B-301 y el punto de máximo alcance o final corresponde al sótano del almacén de mantenimiento del edificio A de la ETSIT. Esta representación se observa en la *Figura 35*.

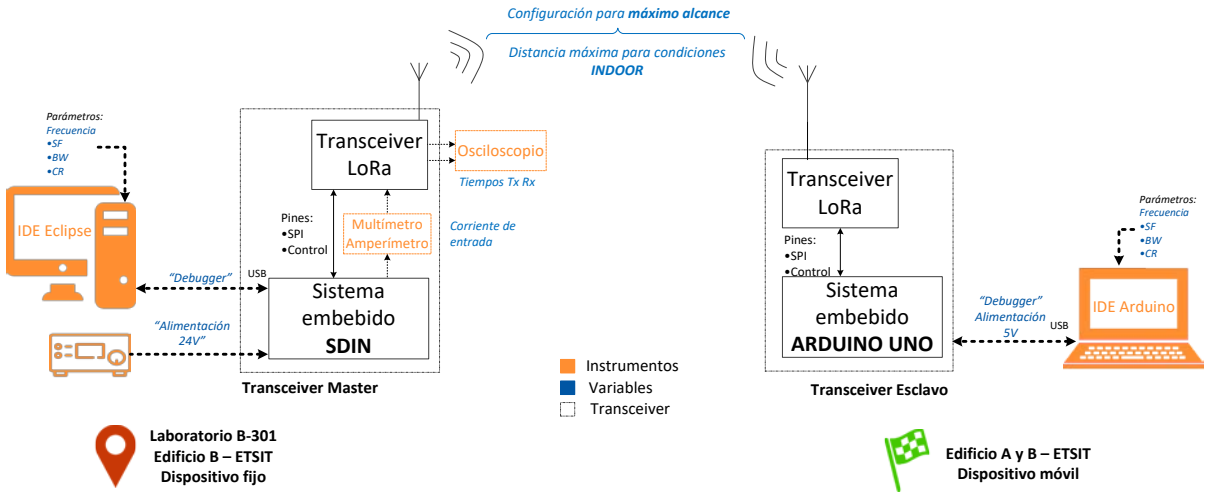


Figura 34. Esquema propuesto para pruebas de alcance Indoor
Fuente: Esquema desarrollado por Autor

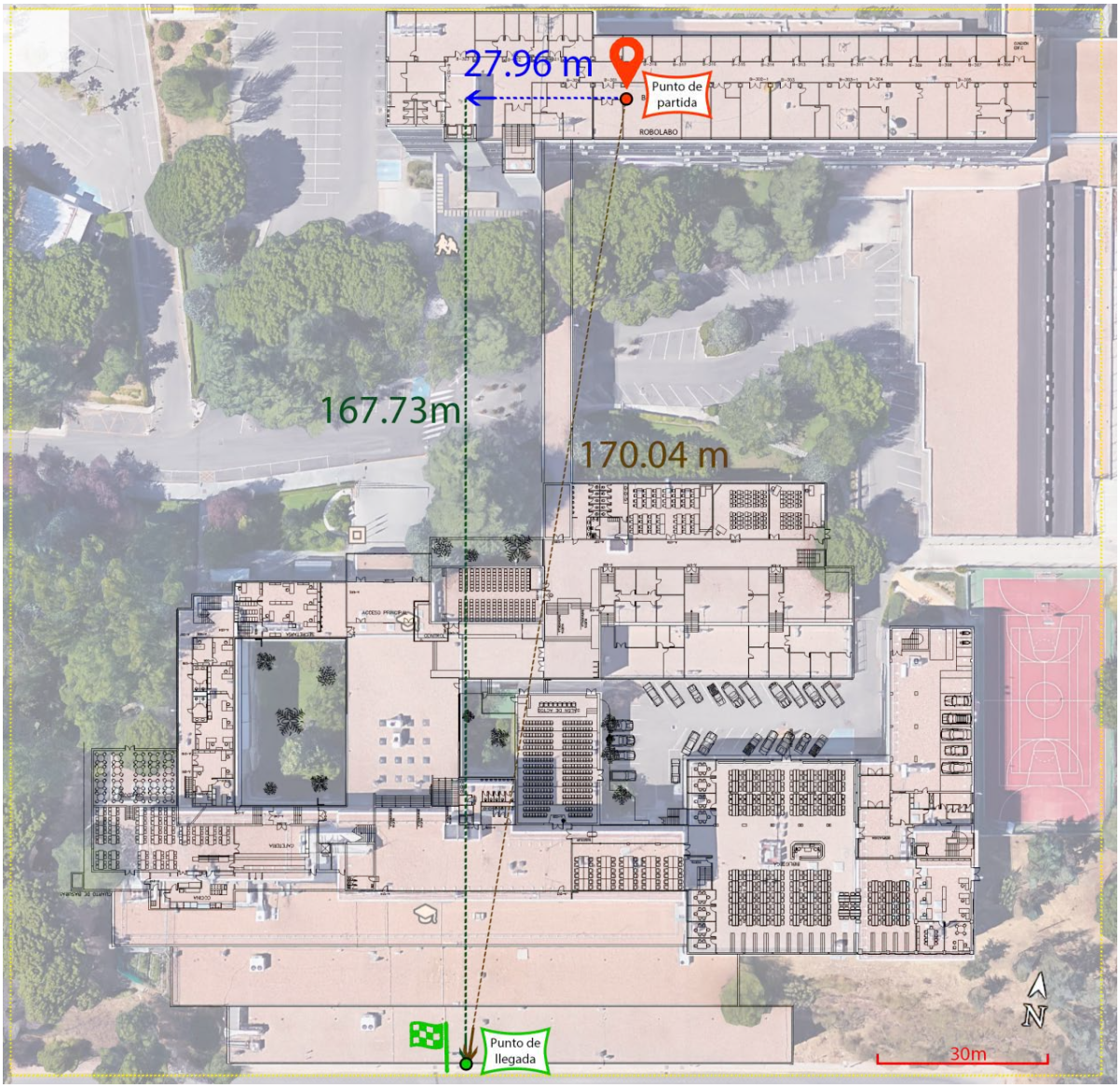


Figura 35. Máximo alcance en trasmisión/recepción de datos para LoRa SX1278 (plano XY).
Fuente: Google earth 7.1.5.1557 y planos compartidos por la ETSIT. Escala 1:30

La *Figura 36* presenta los plano XY y YZ referencia para realizar el cálculo del máximo alcance en las instalaciones de la ETSIT. Para determinar la distancia desde el punto de partida hasta el punto de llegada o de máximo alcance, se realizan cálculos a partir de teoremas de triangulo rectángulos. En la *Figura 36* (izquierda), se presenta la máxima distancia alcanzada con referencia del suelo, esta distancia se calculó utilizando el teorema de Pitágoras con referencia a las distancias de los ejes X (27.96m) y Y (167.73m).

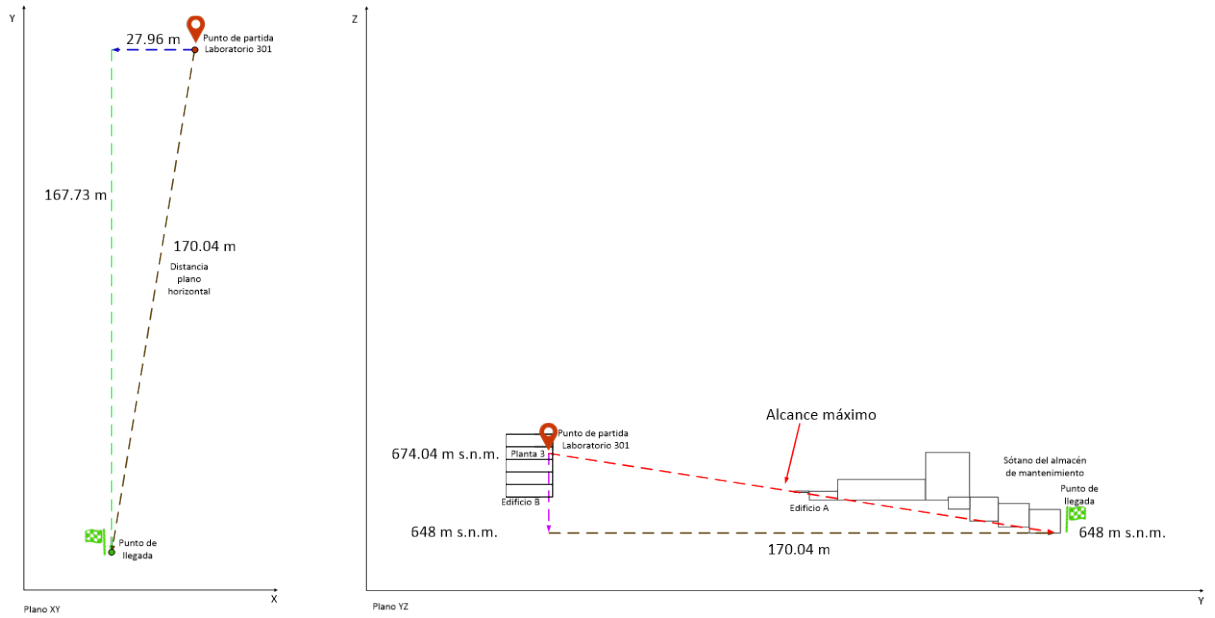


Figura 36. Esquema para determinar el alcance en trasmisión/recepción de datos para LoRa SX1278 (plano XY y YZ)
Fuente: Información del autor y planos compartidos por la ETSIT

Para determinar el máximo alcance con referencia a los transceivers, se utiliza la *Figura 36* (derecha). La diferencia de altura entre los dispositivos se calcula utilizando el modelo *(8)*:

$$h_{entre\ dispositivos} = h_{dispositivo\ maestro} - h_{dispositivo\ esclavo} \quad (8)$$

$$h_{entre\ dispositivos} = 674.04 - 648 \quad (9)$$

$$h_{entre\ dispositivos} = 26.04\ \text{m} \quad (10)$$

Finalmente, utilizando el teorema de Pitágoras se calcula la distancia de alcance máximo lograda por la configuración presentada en la *Tabla 16* por el dispositivo LoRa SX1278:

$$distancia_{máx\ alcance} = \sqrt{h_{entre\ dispositivos}{}^2 + distancia_{referencia\ suelo}{}^2} \quad (11)$$

$$distancia_{máx\ alcance} = \sqrt{26.04^2 + 170.04^2} \quad (12)$$

$$distancia_{máx\ alcance\ SX128} = 172.03\ \text{m} \quad (13)$$

El máximo alcance logrado para los dispositivos SX1278 con la configuración que permite obtener el mejor rendimiento en términos de alcance (*Tabla 16*) es de 172 metros en aplicaciones *Indoor*.

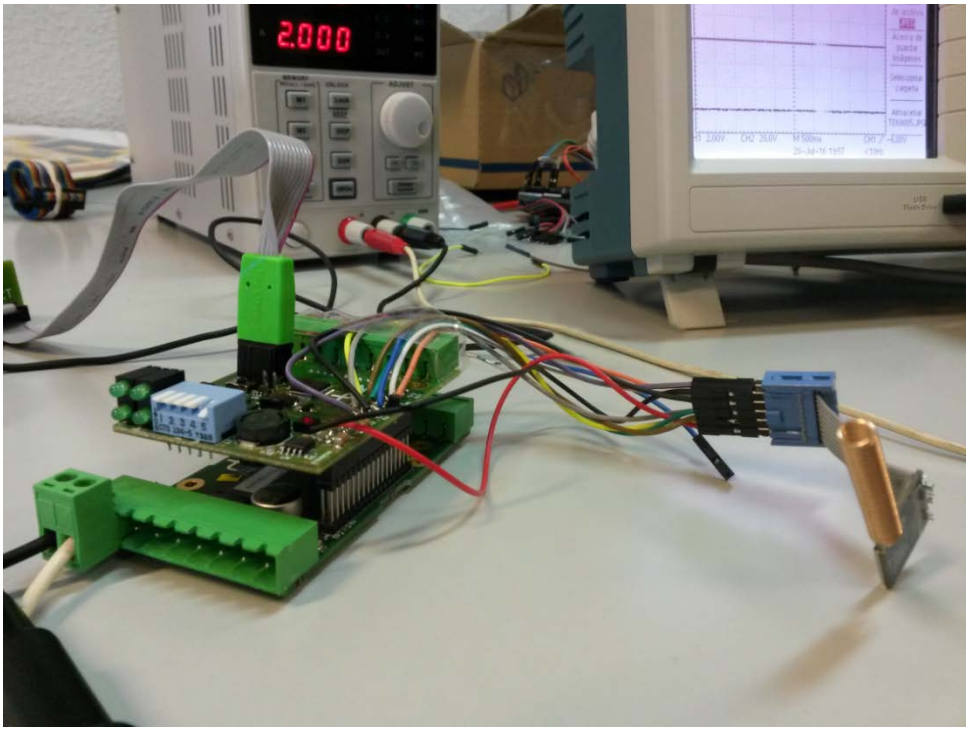


Figura 37. Dispositivo maestro conformado por sistema embebido SDIN y transceiver SX1278 en el laboratorio B-301 de la ETSIT
Fuente: Fotografía tomada por autor

Figura 38. Dispositivo esclavo conformado por sistema embebido Arduino Uno y transceiver SX1278 en sótano del edificio A de la ETSIT
Fuente: Fotografía tomada por autor

4.1.2.4.2 Pruebas de alcance "Outdoor" para LoRa SX128 (LoRa1278 de NiceRF)

Las pruebas de configuración a máximo alcance para espacios *Outdoor* (con línea de vista u obstáculos como vegetación en espacios amplios) se desarrollaron en el *parque Dehesa de la Villa*, espacio muy próximo a las instalaciones de la ETSIT – UPM como se observa en la *Figura 26*.

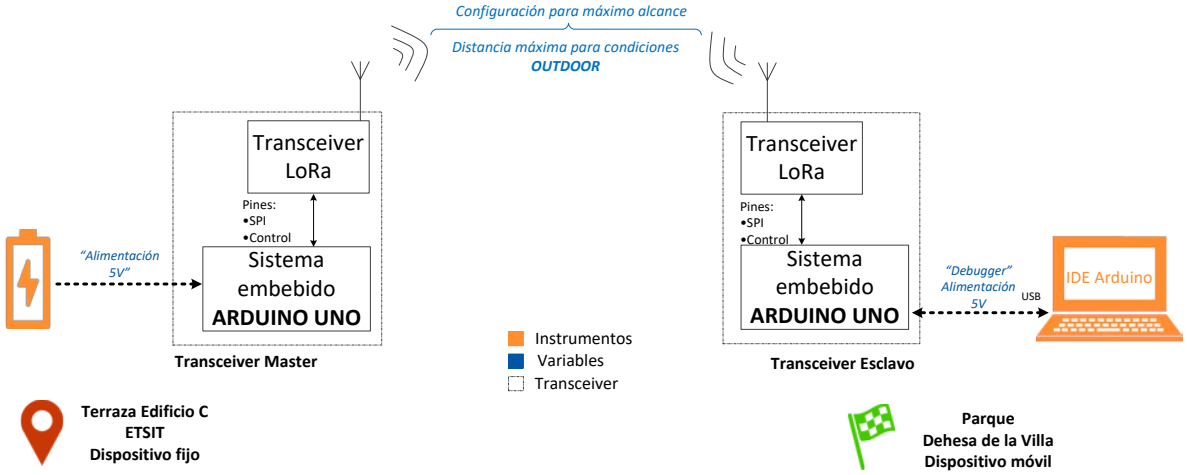


Figura 39. Esquema propuesto para pruebas de alcance "Outdoor"
Fuente: Esquema desarrollado por Autor

La *Figura 39* presenta el esquema de pruebas de máximo alcance para espacios Outdoor. El dispositivo maestro utilizado para el desarrollo de la prueba, estuvo compuesto por un sistema embebido Arduino UNO conectado al transceiver LoRa1278 de NiceRF y alimentado por una batería de 5V. Este dispositivo fue ubicado en la terraza del edificio C de la ETSIT como se observa en la *Figura 40* (dispuesto como dispositivo de ubicación fija).

Como dispositivo esclavo se utilizó la tarjeta Arduino UNO como sistema embebido, conectado al transceiver LoRa1278 de NiceRF y alimentado por un equipo de cómputo portátil utilizando la salida de voltaje del puerto USB (5V). La tarjeta Arduino Uno se encontraba continuamente conectada al equipo de cómputo para el suministro de voltaje y para realizar continuamente el "debugger", operación necesaria para comprobar la efectiva comunicación de los datos con el uso del IDE Arduino. Este dispositivo se desplazó en el *parque Dehesa de la Villa* debido a que es el espacio abierto más cercano a las instalaciones de la ETSIT y para establecer condiciones similares a las que se encuentran en aplicaciones del ámbito del desarrollo (ver *Figura 41*).

Figura 40. Dispositivo maestro conformado por sistema embebido Arduino Uno, transceiver SX1278 y batería de 5V en la terraza del edificio C de la ETSIT
Fuente: Fotografía tomada por autor

Figura 41. Dispositivo esclavo conformado por sistema embebido Arduino Uno, transceiver SX1278 en el parque Dehesa de la Villa
Fuente: Fotografía tomada por autor

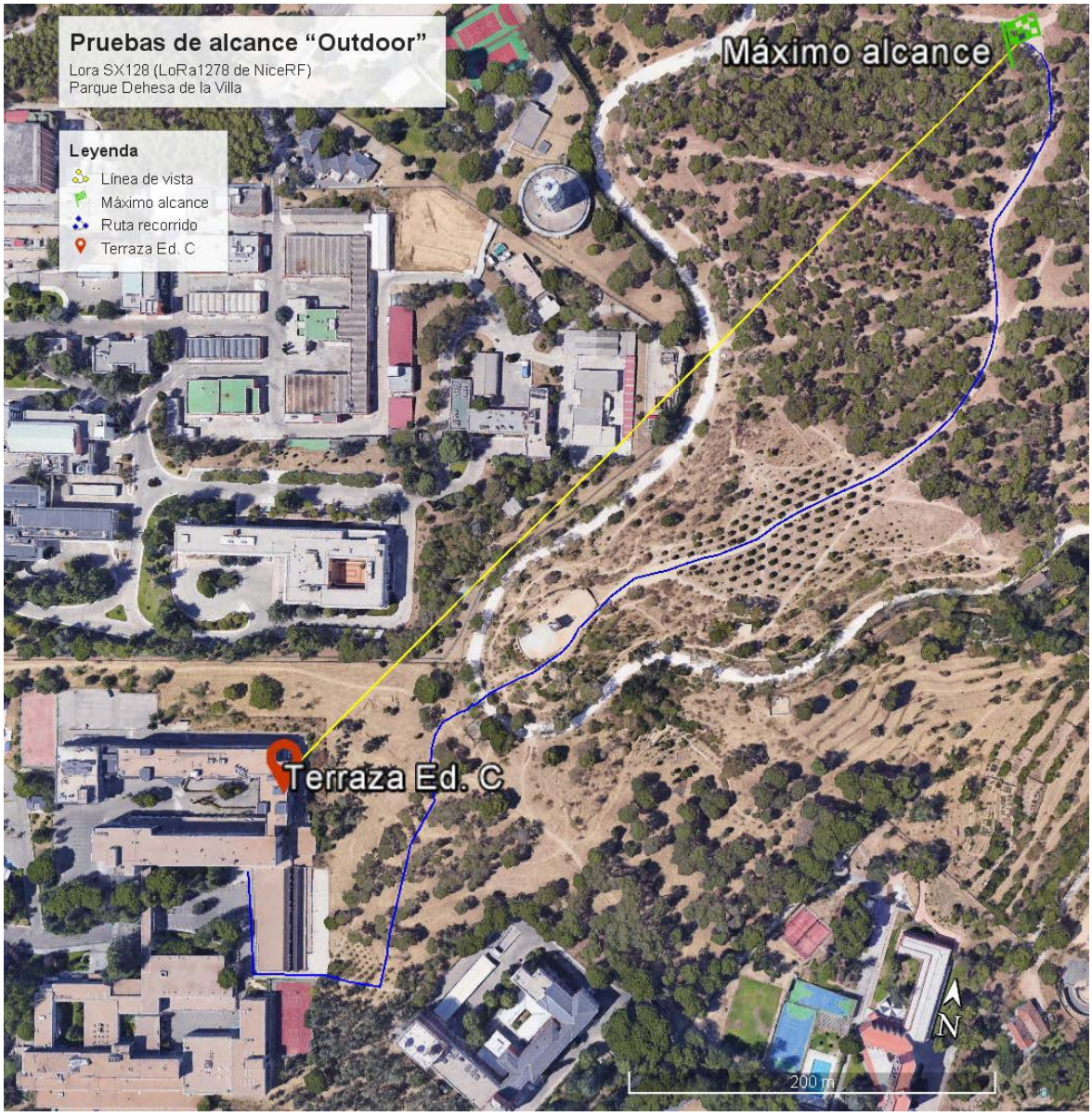


Figura 42. Máximo alcance en espacio "outdoor" utilizando transceivers SX1278
Fuente: Google earth 7.1.5.1557

La *Figura 42* presenta el máximo alcance para espacios *Outdoor* determinado con el uso de los transceivers SX1278. El esquema presenta el recorrido realizado para encontrar el punto de máximo alcance de trasmisión/recepción de datos y determinar la línea de vista generada entre los transceivers ubicados en la terraza del edificio C de la ETSIT y el parque Dehesa de la Villa.

Utilizando *google earth* con la herramienta "*perfil de elevación*", se estableció la posición y la diferencia de elevación para calcular la distancia en línea recta o línea de vista. El resultado se presenta en la *Figura 43*.

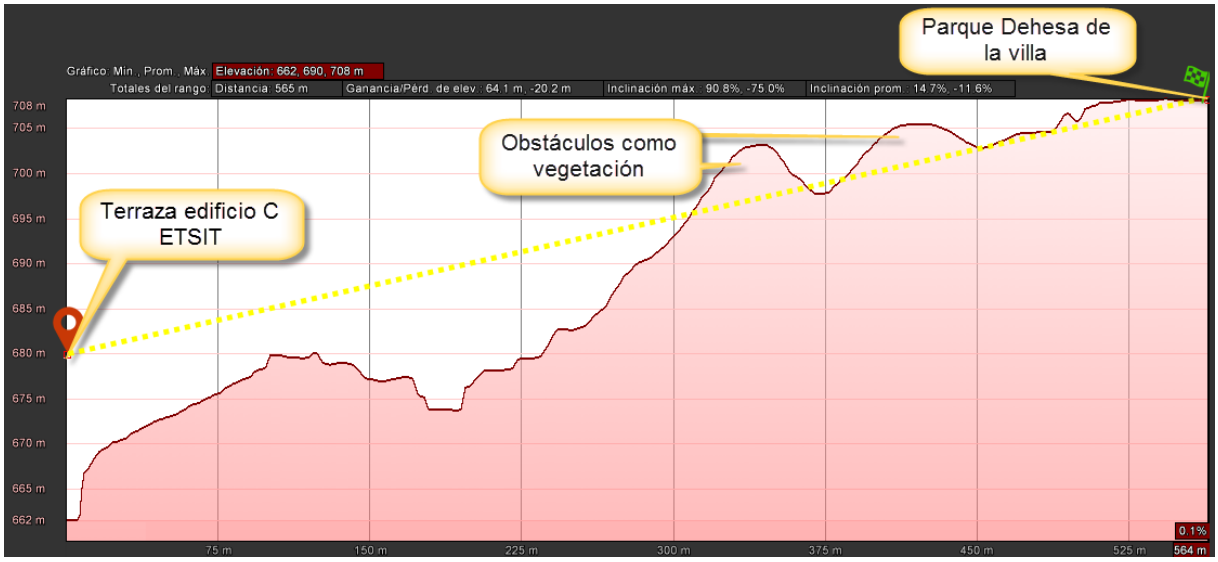


Figura 43. Perfil de elevación entre la terraza del edificio C y el punto de máximo alcance en el parque Dehesa de la Villa
Fuente: Google earth 7.1.5.1557

Utilizando los datos suministrados por el "*perfil de elevación*" de la *Figura 43*, se realizan los cálculos para establecer la distancia de máximo alcance. La distancia sobre el plano horizontal es de 564 metros y la diferencia de altura entre los transceivers es de 28 metros.

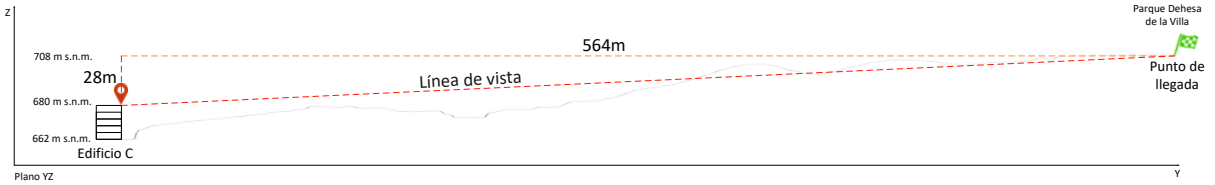


Figura 44. Esquema para determinar el alcance en trasmisión/recepción de datos para LoRa SX1278 (plano YZ)
Fuente: Información del autor y perfil de elevación de google earth 7.1.5.1557

Para determinar el máximo alcance con referencia a los transceivers se utiliza la *Figura 44*. La diferencia de altura entre los dispositivos se calcula utilizando el modelo *(14)*:

$$h_{entre\ dispositivos} = h_{dispositivo\ esclavo} - h_{dispositivo\ maestro} \quad (14)$$

$$h_{entre\ dispositivos} = 708 - 680 \quad (15)$$

$$h_{entre\ dispositivos} = 28\ \text{m} \quad (16)$$

Finalmente, utilizando el teorema de Pitágoras se calcula la distancia de alcance máximo lograda por la configuración presentada en la *Tabla 16* por el dispositivo LoRa SX1278:

$$distancia_{máx\ alcance} = \sqrt{h_{entre\ dispositivos}{}^2 + distancia_{referencia\ suelo}{}^2} \quad (17)$$

$$distancia_{máx\ alcance} = \sqrt{28^2 + 564^2} \quad (18)$$

$$distancia_{máx\ alcance\ SX1278} = 565\ \text{m} \quad (19)$$

El máximo alcance logrado para los dispositivos SX1278 con la configuración que permite obtener el mejor rendimiento en términos de alcance (*Tabla 16*) es de 565 metros en aplicaciones *Outdoor*.

4.1.3 Pruebas de dispositivos LoRa con diseño de red tipo estrella

Finalmente se aumentó la complejidad del diseño creando una red estrella con un nodo central (circuito maestro) y 3 dispositivos esclavos. Como circuitos esclavos, se utilizan diversas versiones de sistemas embebidos (Arduino Uno, Arduino DUE y SDIN). Cabe resaltar que los dispositivos LoRa deben ser de la misma referencia, estar configurados a la misma frecuencia y con los parámetros SF, BW y CR idénticos. Estos parámetros se presentan en la *Tabla 16*.
El modem LoRa es *half-duplex*, es decir que los LoRa no puede trasmitir o recibir datos en el mismo instante; esto significa que los transceiver LoRa deben ser configurados para trasmitir o recibir datos. Cuando se incorpora más de dos dispositivos en un protocolo de comunicación *half dúplex*, se deben establecer estrategias que mitiguen los efectos adversos por colisión indeseada de datos cuando se reciben datos.
Si los dispositivos no cuentan con estrategias para evitar la trasmisión de datos sobre el mismo canal de comunicación por parte de los dispositivos esclavos en un mismo instante, los datos llegaran errados al dispositivo maestro y la comunicación de datos será incoherente.
La solución para evitar la colisión de datos, corresponde a establecer tiempos de muestreos y tiempos aleatorios de "des-sincronización" cuando se trasmiten datos; es decir evitar la colisión de datos con los tiempos adicionales de trasmisión.
Una vez se establece el tiempo en el que no se presentan colisiones, se almacena dicho valor para ser utilizado consecutivamente en las próximas trasmisiones de datos y con lo anterior el sistema *half dúplex* se comporta sincronizado. En el documento Anexo A: "Desarrollo del driver para el uso de los dispositivos de radio LoRa con sistemas embebidos" – apartado *"Ejemplo de aplicación: recepción de datos de varios dispositivos remotos (red punto – multipunto)"*, se describe de manera detallada el proceso de desarrollo de la prueba desarrollada para crear la red punto-multipunto con dispositivos LoRa.

CONCLUSIONES

Las TIC o tecnologías de la información, corresponden al conjunto de tecnologías que permiten la adquisición, almacenamiento y manipulación de la información. Se encuentran presentes en diferentes áreas de la sociedad siendo un elemento transversal como su uso y aplicación en el gobierno, en la participación de la sociedad, salud, educacion, medio ambiente, entre otras áreas. Por ser considerado un elemento transversal en el desarrollo, se considera como apoyo, respaldo o una herramienta.

Para el año 2001, el PNUD dedico su informe anual haciendo referencias importantes respecto a los adelantos tecnológicos como herramienta al servicio del desarrollo humano. El informe presenta relaciones y vínculos ente la innovación tecnológica y la brecha digital con el desarrollo humano.

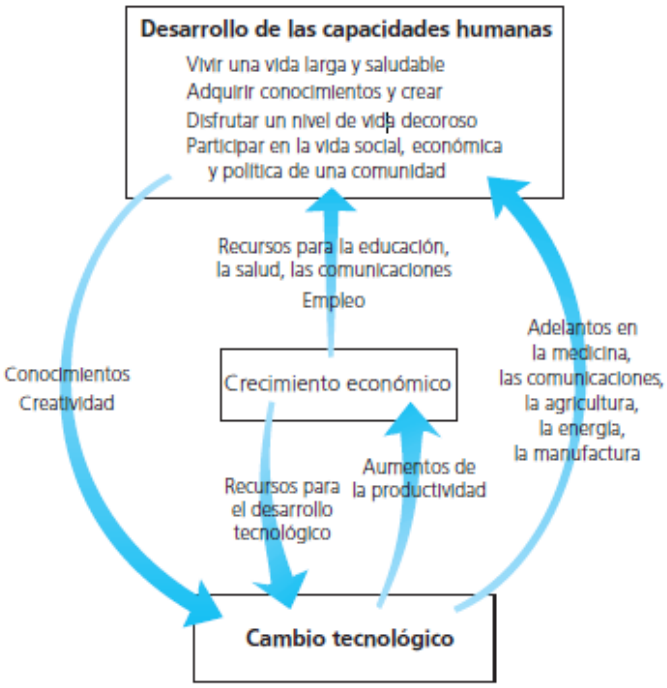


Figura 45. Vínculo entre las tecnologías y el desarrollo humano
Fuente: PNUD (2001)

De acuerdo a la *Figura 45*, las tecnologías influyen en el desarrollo humano debido a que innovaciones aumentan la capacidad de las personas gracias a sus aportaciones en sectores prioritarios como la salud, servicios básicos y la educacion; y se constituye en un medio para generar crecimiento económico gracias a la incidencia de los cambios tecnológicos.

El auge de las Tecnologías de la Información y las Comunicaciones en el siglo XXI ha permitido almacenar grandes volúmenes de información debió a la dinámica de creación

intencionada o no de información por parte de usuarios de diferentes tecnologías, llámense teléfonos móviles, rastros de posición GPS, equipos de cómputo, cámaras de seguridad, diferentes sensores, entre otras. Esta información proviene en su mayoría de datos generados en internet. Con la aparición de internet se revoluciono la vida de los seres humanos por ofrecer la posibilidad de compartir conocimientos, experiencias y facilitar la comunicación entren personas y organizaciones de diversas formas.
Sin embargo, internet no solo ha permitido conectar personas, sino dispositivos, sensores y complejos sistemas. Es así como aparece el término "Internet de las cosas" o "internet de los objetos" o "IoT" que corresponde a la interconexión de diversos dispositivos a internet que generan datos en tiempo real.
Lo importante del "internet de las cosas" no radica en la conexión de las "cosas" a internet, sino en la información y conocimiento que estas puedan aportar para mejorar la calidad de las personas.

El IoT se compone por una colección dispersa de redes diferentes y con distintos fines, por lo que se considera como la red de redes.

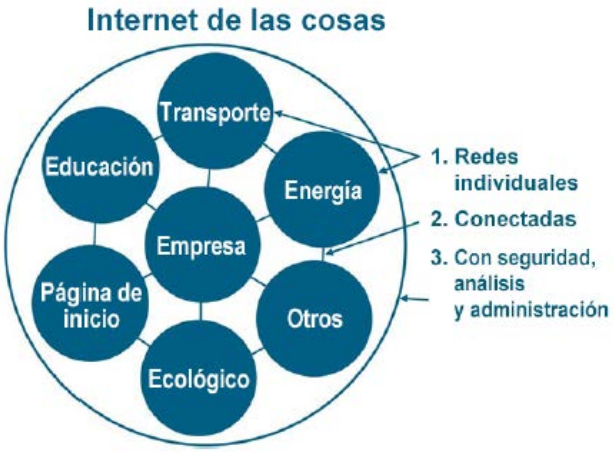


Figura 46. IoT: la red de redes.
Fuente: Cisco IBSG (2011)

Según Cisco IBSG (2011) , se encuentran en marcha proyectos IoT que prometen cerrar la brecha de desigualdad, mejorar la distribución de los recursos en el mundo para quienes más los necesitan y ayudar a comprender el planeta. Sin embargo, existen barreras que amenazan con retrasar el desarrollo de IoT, como la transición del protocolo IPv4 a IPv6, el establecimiento de un conjunto de normas en común, el desarrollo de fuentes propias de energías para sensores diminutos y la *optimización en el consumo energético en las redes de comunicación entre los objetos.*
Respecto a la optimización del consumo energético de las redes, es un importante campo de estudio del IoT debido a que se requieren considerables consumos energéticos para estar conectados a internet. Es así como aparecen proyectos relacionados con "Chirp Networks" o

redes alternativas que consumen significativamente menos energía que Wi-Fi o Bluetooth. La clave en la reducción energética consiste en lograr desarrollar protocolos de comunicación que varíen la modulación de la frecuencia para trasmitir información, reduciendo la potencia y por ende mejorando el consumo energético. Un ejemplo claro de la solución en reducción de consumo energético, en la variación del ancho de banda y el aumento en el radio de cobertura son las redes *LPWAN* (*Low power wide area network o red de área extendida de bajo consumo*). La comparación grafica del ancho banda requerido para la comunicación con respecto al radio de cobertura se presenta en la *Figura 47*.

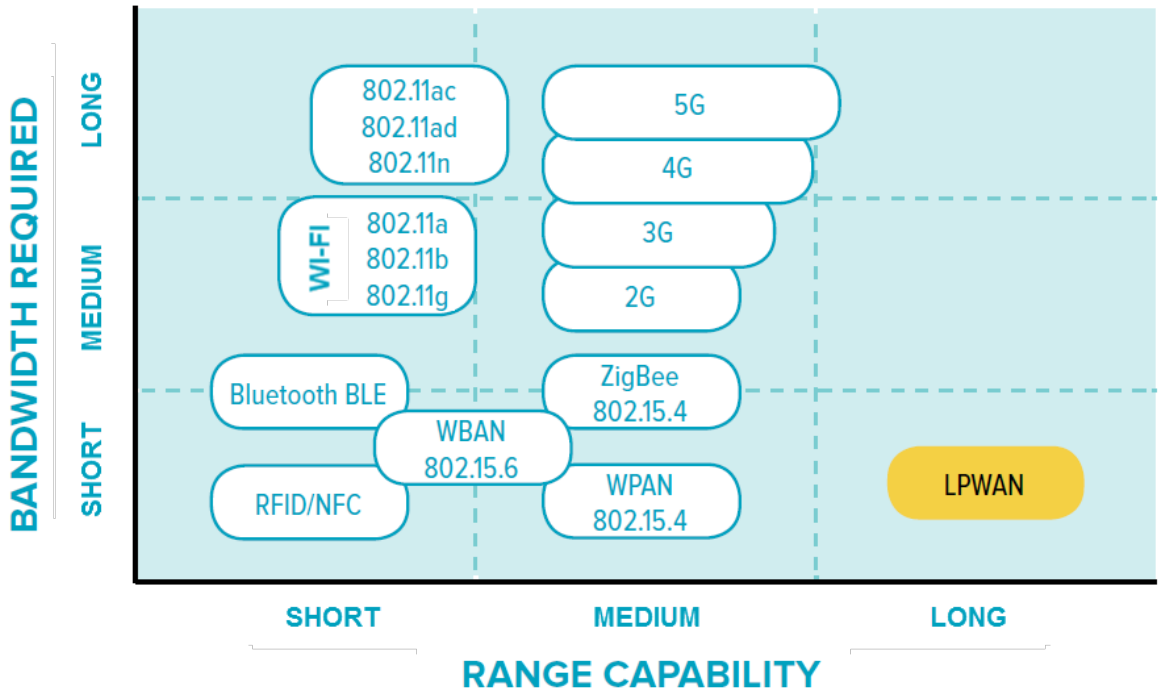


Figura 47. Comparación del ancho de banda requerido (frecuencia) con respecto al rango de cobertura para los principales estándares inalámbricos.
Fuente: Link Labs (2016)

Si bien este protocolo (LPWAN) aparece como respuesta a las tendencias y necesidades del mercado del IoT, es necesario adaptar su uso para aplicaciones del ámbito del desarrollo debido a que el uso de la tecnología apropiada puede ser la clave en el aseguramiento de los servicios básicos que sustentan redes de infraestructuras como agua potable, comunicaciones, electricidad, saneamiento, entre otros; mejorando la calidad de la vida de las personas y en especial las que se encuentran más aisladas de la población. Sin embargo, esta tecnología debe ser ajustada al contexto, características y población beneficiada:
"En la era de las redes, cada país necesita contar con capacidad para comprender las tecnologías mundiales y adaptarlas a las necesidades locales", PNUD (2001).

Es así que se planteó el desarrollo de un driver para los dispositivos LoRa SX127X junto con una serie de pruebas para determinar el consumo energético y distancia acorde con

condiciones reales como *espacios Indoor* (obstáculos como infraestructuras y muros) y *espacio Outdoor* (obstáculos como vegetación) junto con características y particulares de los dispositivos LoRa plasmadas en este documento.

Los dispositivos LoRa, hacen parte de la red LoRaWAN, la cual es una red de área extensa (LPWAN*: Low power wide area network*) que proporciona bajo consumo energético, bajo costo y largo alcance de cobertura.
Este estándar facilita la interoperabilidad entre las cosas o dispositivos inteligentes sin necesidad de instalaciones locales complejas y brinda la libertad al usuario, desarrollador o empresas en el despliegue para IoT.
Actualmente existe la "LoRa Aliance", la cual es una asociación abierta, sin fines de lucro e iniciada por la industria *Semtech* con la misión de estandarizar las redes de baja potencia de área extendida (LPWAN) desplegadas en todo el mundo para permitir la conexión con IoT, dispositivos máquina a máquina (M2M), Smart City y aplicaciones industriales. Los miembros de la Alianza colaborarán para impulsar el éxito global del protocolo de LoRa (LoRaWAN), mediante el intercambio de conocimiento y experiencia para garantizar la interoperabilidad entre los operadores en un estándar global abierto (LoRa Alliance, 2016).
Una de las principales razones por las cuales fue seleccionado *el estándar LoRaWAN, corresponde a que es un estándar de abierto y es ideal para el uso con sensores* (Link Labs, 2016)*, lo que implica que es adecuado para aplicaciones del ámbito del desarrollo.*

Sin embargo, fue necesario explorar las especificaciones técnicas de otros estándares inalámbricos debido a que existen en los mercados estándares muy bien documentados y conocidos.

La *Tabla 17* presenta el cuadro comparativo de las tecnológicas inalámbricas LoRa, ZigBee, Bluetooth y Wi-Fi. Esta información se ha complementado con las pruebas de alcance desarrolladas en el apartado "*Pruebas de consumo energético y alcance para LoRa SX1278"*, junto con el *Anexo C: DataSheet LoRa SX1278*.

Se puede observar la superioridad del alcance nominal de los dispositivos LoRa con respecto a las demás tecnologías (hasta 5 veces mayor alcance que Wi-Fi).
En cuanto a la máxima tasa de datos, la baja tasa de trasmisión de datos de hasta 37.5 Kbps en los dispositivos LoRa los hace poco eficientes para aplicaciones en la que sea necesaria la trasmisión de video o audio; es decir, el estándar LoRa es adecuado para la trasmisión de datos de sensores (sensórica) que no necesiten control estricto en tiempo real; es decir, aplicaciones en la que el tiempo de respuesta no sea crítico. Respecto a la frecuencia de trabajo, el estándar LoRa utiliza la más baja frecuencia y esto tiene sentido debido a que la frecuencia es inversamente proporcional al tiempo; lo que implica que si la frecuencia se reduce, el tiempo de trasmisión de datos aumenta (trasmisión lenta). Sin embargo, al trabajar a tan bajas frecuencias, es posible utilizar las frecuencias ISM (Industrial, Scientific and Medical), las cuales son intencionalmente libres para uso no comercial en aplicaciones de tipo

industrial, científico y médico (ver *Tabla 2*: Banda de frecuencia ISM compatibles con los transceivers LoRa).

Tabla 17. Cuadro comparativo de tecnologías inalámbricas LoRa, ZigBee, Bluetooth y Wi-Fi. Fuente: Jin-Shyan, Yu-Wei & Chung-Chou (2007). Información basada en las pruebas desarrolladas por el autor

Característica	LoRa	ZigBee	Bluetooth	Wi-Fi
Especificación IEEE	802.15.4g	802.15.4	802.15.1	802.11ª/b/c
Frecuencia de trabajo	137 – 1020 MHz	868/915 MHz; 2.4 GHz	2.4 GHz	2.4 GHz; 5 GHz
Ancho de banda de canales BW	7.8 – 500 kHz	0.3/0.6 MHz; 2 MHz	1 MHz	22 MHz
Máxima tasa de datos	0.018 – 37.5 Kb/s	250 Kb/s	1 Mb/s	54 Mb/s
Alcance nominal	**172 m (Indoor)* 565 m (Outdoor)***	10 -100 m	10 m	10 -100 m
Máxima potencia nominal de TX	Hasta 20 dBm	(-25) - 0 dBm	0 - 10 dBm	15 - 20 dBm
Protección de datos (Error de redundancia cíclica)	16-bit CRC	16-bit CRC	16-bit CRC	32-bit CRC
Tiempo de conexión	Menos de un segundo*	Menos de 30 ms	Entre 3 - 5 segundos	Máximo 10 segundos
Tipo de modulación	FSK/OOK/LoRa™	BPSK (+ ASK), O-QPSK	GFSK	BPSK, QPSK COFDM, CCK, M-QAM
Consumo energético	Muy bajo*	Bajo	Intermedio	Alto
Tipo de comunicación	Half dúplex	Half dúplex	Full dúplex	Full dúplex
Tipo de red	LPWAN **Red de área ampliada de bajo consumo**	WPAN Red de área personal	WPAN Red de área personal	WLAN Red de área local
Topologías de red	Punto a punto, punto a multipunto, estrella, malla	Punto a punto, punto a multipunto, estrella, malla, arbol	Scatternet (propio de Bluetooth)	Punto a hub ESS malla
Aplicaciones	Automatización, Domótica, sistemas de alarmas inalámbricas, monitoreo y control industrial, sistemas de irrigación, **sensórica***	Dispositivos portátiles, Domótica (control y monitoreo), automatización, juguetes, sistemas de irrigación, **sensórica**	Dispositivos portátiles (periféricos como auriculares, teclados, mouse, entre otros) y algunos dispositivos industriales de comunicación	Se utiliza para acceder a Internet y para interconexión de computadoras

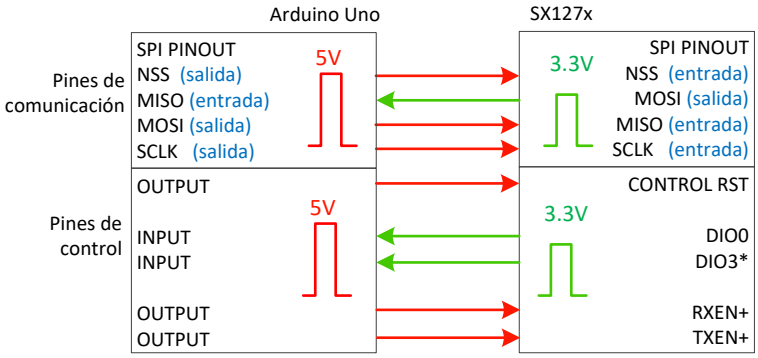
*Información obtenida a partir de las pruebas de alcance y consumo energético del presente documento.

La comparación evidencia que el estándar LoRa es el de menor consumo energético con respecto a los estándares ZigBee, Bluetooth y Wi-Fi. Sin embargo, el estándar LoRa presenta como principal desventaja el utilizar comunicación *half dúplex*; es decir, que los dispositivos pueden trasmitir o recibir datos en un instante. Cuando se crean complejas redes de comunicación, el utilizar comunicación *half dúplex* resulta ser una limitante si no se implementan estrategias adecuadas para efectuar la correcta comunicación. En el documento Anexo A: "Desarrollo del driver para el uso de los dispositivos de radio LoRa con sistemas embebidos" – apartado "*Ejemplo de aplicación: recepción de datos de varios dispositivos remotos (red punto - multipunto)*", se describe de manera detallada estrategias para utilizar el tipo de comunicación *half dúplex* en complejas redes punto-multipunto para sensores con dispositivos LoRa.

Con respecto a las redes cableadas para aplicaciones del ámbito del desarrollo, el uso de medios cableados aumenta los costos de implementación debido al uso de cables (si se requieren largas distancias) y por ende se aumenta el costo de mantenimiento en la red. En algunos casos se hace necesario utilizar cables blindados para evitar la interferencia del ruido electromagnético (mejorar la calidad del cableado). Además, para asegurar la correcta trasmisión de los datos se deben aumentar los niveles de voltaje, por lo que en algunos casos no estarían acordes como tecnologías para el desarrollo.

Estos dispositivos fueron probados con Arduino, los cuales tienen como especial característica el ser hardware libre, estar documentados en la web, contar con un costo de adquisición razonable, tener aceptación por parte de la comunidad de desarrolladores y posibilidad de implementar sistemas básicos de monitoreo y control con el uso de sensores. En el apartado "*Sistemas embebidos - importancia del hardware libre"* se presenta la importancia del uso de hardware libre para el ámbito del desarrollo. "*El hardware de fuentes abiertas da libertad de controlar la tecnología y al mismo tiempo compartir conocimientos y estimular la comercialización por medio del intercambio abierto de diseños*" (*Lazalde, Torres & Vila-Viñas, 2015*).

Si bien Arduino es un referente importante al indagar sobre hardware abierto, durante las pruebas desarrolladas con los dispositivos LoRa se evidenciaron problemas particulares.
El principal problema corresponde a los niveles incompatibles de voltaje del sistema embebido Arduino Uno y los transeivers. Arduino utiliza 5 voltios para pines de control y comunicación SPI (Torrente, 2013), mientras los transceiver LoRa utilizan 3.3 voltios para pines de control y comunicación (Semtech SX1272, 2015 & Semtech SX1278, 2015). Sin bien las pruebas no se vieron perturbadas por la incompatibilidad con niveles de voltaje, si continuamente permanecen conectados estos dispositivos, el transceiver LoRa podría verse afectado y hasta destruirse por recibir más voltaje que el nominal necesario. Este planteamiento se representa en la *Figura 48*.



*No implementado en LoRa1278 +Implementado en LoRa1278

Figura 48. Comparación de los niveles de voltaje entre Arduino Uno y los transceiver LoRa.
Fuente: Imagen planteada por autor a partir de Torrente (2013) & Semtech SX1278 (2015)

Sin embargo, este no fue el único problema evidenciado con el uso del dispositivo Arduino Uno como sistema embebido. Al realizar las mediciones de los niveles de voltaje de alimentación en el transceiver LoRa con respecto a los modos trasmisión y recepción, se evidencio un aumento de voltaje sobre el pin de alimentación del transceiver. El esquema de conexión utilizado se presenta en la *Figura 49*.

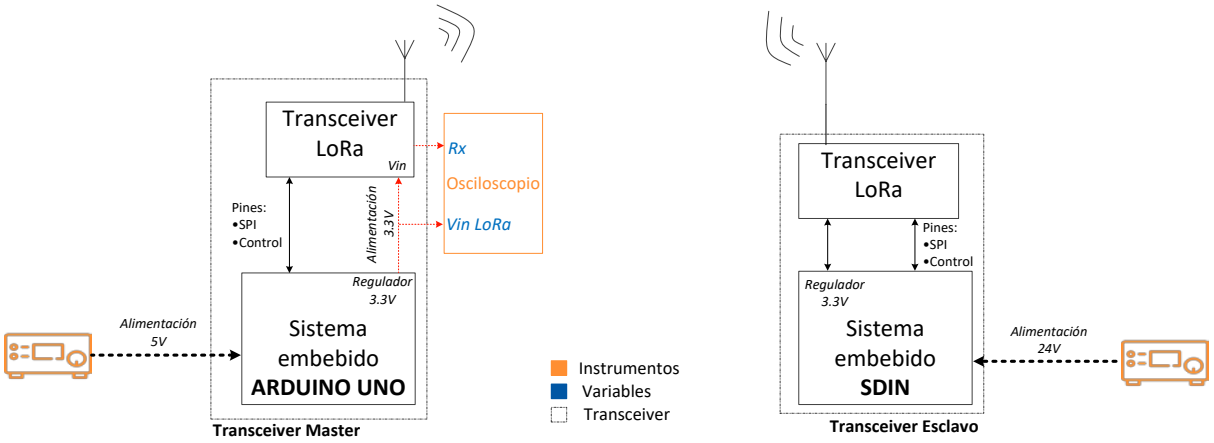


Figura 49. Esquema de conexión para visualizar el nivel de alimentación con respecto a los modos trasmisión y recepción en los dispositivos LoRa.
Fuente: Esquema planteado por autor

Cuando el dispositivo se encuentra en modo recepción de datos, existe un efecto de "sobre" alimentación que pasa de los 3.3 voltios nominales de alimentación, hasta los 4.1 voltios. Esto ocurrió para cualquiera de las configuraciones planteadas en la *Tabla 14* y *Tabla 15*.
Según las especificaciones del fabricante de tarjetas Arduino Uno, el regulador utilizado como alimentación de 3.3 voltios corresponde a la referencia *LP2985-33DBVR*, el cual es un regulador de bajo ruido con voltaje de salida nominal de 3.3V con una máxima corriente de

suministro hasta 150mA (Texas Instruments, 2004). Este regulador en teoría debería mantener el voltaje estable y no permitir el aumento o reducción en el voltaje. Sin embargo, como se observa en la *Figura 50*, existe un efecto nocivo de "sobre" alimentación en los dispositivos LoRa que con el uso continuo con Arduino Uno, podría llegar a destruir el transceiver.

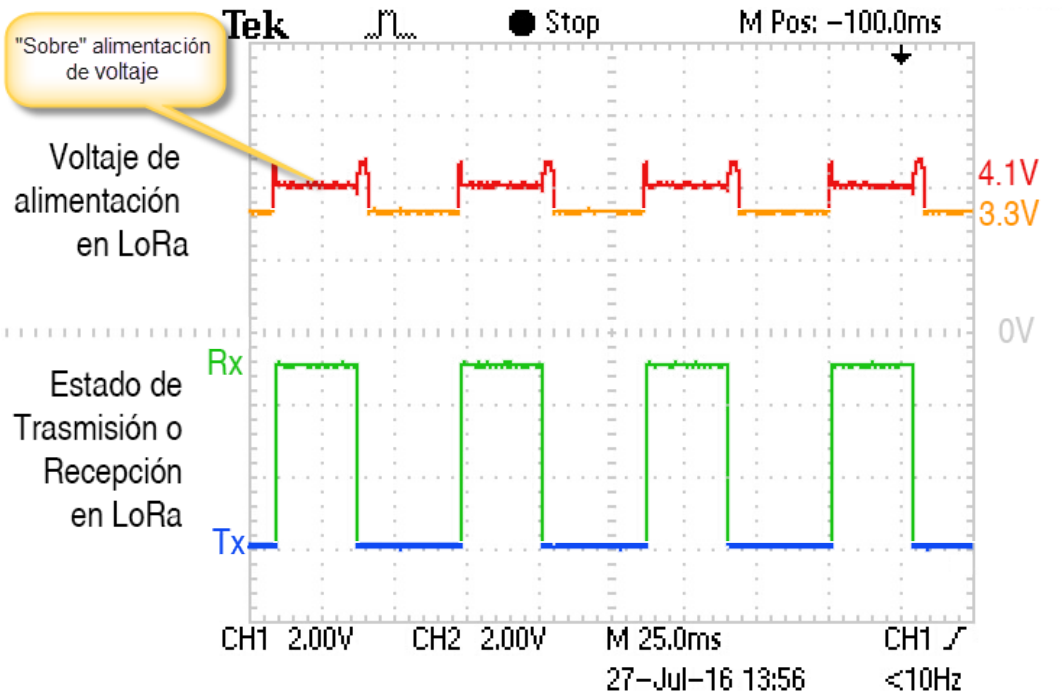


Figura 50. Diagrama de tiempos de la señal de alimentación con respecto a los modos trasmisión y recepción en los dispositivos LoRa utilizando Arduino Uno.
Fuente: Captura de pantalla de osciloscopio Tektronix

Sin embargo, un efecto nocivo para los dispositivos LoRa también ocurre cuando se utilizó Arduino DUE. Cuando el dispositivo se encuentra en modo trasmisión, el suministro de voltaje se reduce a 2.0 voltios. Según las especificaciones del fabricante de tarjetas Arduino DUE, el regulador utilizado como alimentación de 3.3 voltios corresponde a la referencia *NCP1117ST33T3G*, el cual cuenta con un voltaje de salida nominal de 3.3V con una máxima corriente de suministro de 1A (ON semiconductor, 2014). Este regulador en teoría debería mantener el voltaje estable y no permitir el aumento o reducción en el mismo. Sin embargo, como se observa en la *Figura 51*, existe un efecto nocivo de "reducción" en la alimentación de voltaje de los dispositivos LoRa, lo que podría llegar a dañar el transceiver.

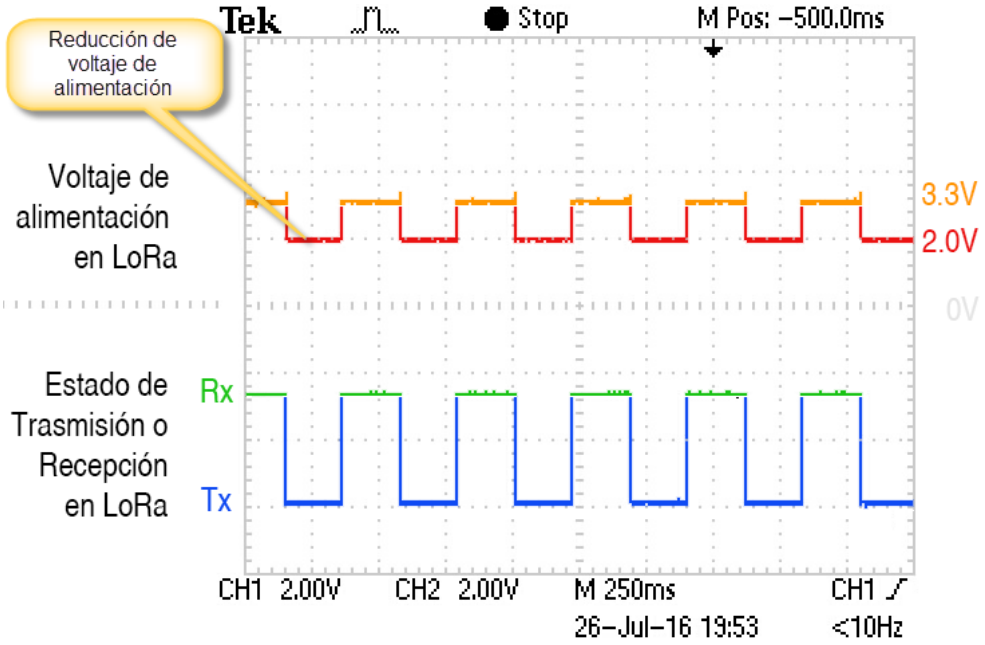


Figura 51. Diagrama de tiempos de la señal de alimentación con respecto a los modos trasmisión y recepción en los dispositivos LoRa utilizando Arduino DUE.
Fuente: Captura de pantalla de osciloscopio Tektronix

Por los motivos expuestos anteriormente, se recomienda el uso de Arduino con dispositivos LoRa siempre y cuando se utilice una fuente de alimentación externa de 3.3V como alimentación de los dispositivos LoRa, evitando utilizar los reguladores de las placas Arduino. Para realizar sistemas complejos y funcionales de monitoreo con los transceiver LoRa en aplicaciones del ámbito del desarrollo, se recomienda el uso el módulo SDIN (utilizados durante las pruebas) o tarjetas de embebidas STMicroelectronics, específicamente los módulos de la familia STM32 con microcontroladores CORTEX M0, M3 o M4 de ultra bajo consumo, series L0, L1 y L4 respectivamente. La *Tabla 18* presenta algunos sistemas embebidos propuestao para la familia de STM32 a ultra bajo consumo energético.

Tabla 18. Tarjetas embebidas STMicroelectronics de ultra bajo consumo
Fuente: ST (2016)

Tipo de Microcontrodor	STM32L0	STM32L1	STM32L4
Tarjeta de desarrollo ST	STM32L073RZ STM32L011K4 STM32L053R8 STM32L031K6 STM32L053C8 STM32L073VZ	STM32L152RE STM32L100RC STM32L152RC STM32L152VB STM32L152ZD	STM32L476RG STM32L432KC STM32L476VG STM32L476ZG

Teniendo en cuenta que las TIC son un medio y no el fin, se plantean algunas aplicaciones para el ámbito del desarrollo con el uso de los dispositivos LoRa:

Agua y saneamiento: Una de las principales funciones del agua es cubrir las necesidades básicas de las personas para su ingesta: agua potable apta para el consumo humano, saneamiento básico, higiene personal, alimentación, producción de alimentos a escala familiar y limpieza del entorno. (Pérez & Jiménez, 2011). Basados en la necesidad de dotar a las poblaciones de agua potable, existen diversos programas que han contado con gran acogida debido a los impactos producidos en los beneficiarios. Entre estos programas se encuentran las cisternas de placa ampliamente difundida en Brasil. Estos programas buscan a través de la captación de agua de los tejados, almacenar el agua en cisternas y sobrellevar las épocas más secas del año. Con el uso de sensores de nivel de agua, se podría realizar el monitoreo del estado actual de las cisternas en términos de capacidad de almacenamiento en tiempo real y buscar una redistribución de este recurso hídrico.

Energía: El acceso a la energía está directamente relacionado con el bienestar de las personas. La energía es necesaria para cocinar, para la iluminación, para la conservación de los alimentos, climatización y servicios básicos que cualquier hogar debería tener cubiertos para asegurar unas condiciones mínimas de confort.
La energía solar con el uso de paneles solares se ha convertido en una solución rentable para las zonas aisladas (alejadas de las redes de distribución eléctrica) como un mecanismo eficaz para suplir el déficit energético. Se propone el uso de redes LoRa con el uso de sensores piranómetros para monitorizar la producción energética respecto a la eficiencia de las instalaciones fotovoltaicas instaladas.

Producción alimenticia- agricultura: La agricultura se podría considerar como uno de los estrategias más importantes para desarrollar los pilares de los Objetivos de Desarrollo Sostenible. Entre estos pilares se tienen la erradicación de la pobreza extrema, combatir la desigualdad y la injusticia, y dar soluciones al cambio climático. Según la FAO, el crecimiento agrícola es en promedio dos veces más eficaz para reducir la pobreza y cinco veces más eficaz que el impulsado por otros sectores (FAO, 2012). Ahora bien, al promover estrategias y metodologías óptimas como la aplicación de la agricultura de conservación, la rotación de los cultivos, la modificación genética de las semillas, el monitoreo y seguimiento continuo a los cultivos; este impacto podría aumentar significativamente. Se propone utilizar las redes LoRa para monitorizar diversas variables de los cultivos como PH, humedad, temperatura y de esta manera optimizar el suministro de agua o hacer seguimiento a plagas u otros factores de interés.
La agricultura es fundamental para los sectores menos favorecidos, por cuanto “más del 75% de los pobres viven en zonas rurales y dependen en gran medida de la agricultura” (FAO - ODS, 2015).

Los anteriores ejemplos de aplicación con el uso de redes LoRa deben ajustarse a los "*Principios para el Desarrollo Digital*" que son un conjunto de criterios utilizados en la implementación de programas basados en tecnología. Tienen el propósito de informar, pero no dictar, el diseño de programas de desarrollo (Digital principles, s. f.).

1. Diseñe con el usuario
Desarrolle soluciones adecuadas al contexto conforme a las necesidades del usuario. Esto hace referencia a establecer la mejor configuración de los transceiver LoRa acorde a las necesidades en términos de alcance de trasmisión necesario y consumo energético deseado. Las *Tabla 14* y *Tabla 15* presentan esta relación para diferentes configuraciones.

2. Comprenda el ecosistema
Alinéese con políticas tecnológicas, legales y normativas vigentes. Debido a que el estándar LoRa es abierto, es posible utilizan frecuencias abiertas para la comunicación entre los dispositivos. Se recomienda utilizar bandas de frecuencias ISM que cumplan con el libre uso de acuerdo a la normativa de cada país. La relación de frecuencias de libre uso se presenta en la *Tabla 2*.

3. Diseñe para multiplicar la eficiencia
Diseñe con el fin de multiplicar la eficiencia desde el principio, evalúe y mitigue las dependencias que podrían limitar la capacidad de escalar. Una de las ventajas del estándar LoRa es ser inalámbrico. Esto implica la independencia de la infraestructura cableada y la escalabilidad del sistema en caso de ser necesaria. En caso de ser necesario el aumento en la cantidad de sensores y dispositivos transceivers, se realizarían cambios a nivel de configuración de software (el hardware podría mantenerse sin modificaciones), permitiendo la escalabilidad del sistema.

4. Siente las bases para la sustentabilidad
Planifique el desarrollo de sustentabilidad desde el principio, incluida la planificación de salud financiera a largo plazo, por ejemplo, al evaluar el costo total de propiedad. Una de las ventajas de los chip radio LoRa es su precio asequible (el par de transeivers tiene un costo de 17€ a corte de julio de 2016). El protocolo LoRaWAN está respaldado por la LoRa Alliance (creado en 2015), lo que repercute en el respaldo y seguridad en el soporte de esta tecnología.

5. Cuente con un servicio basado en datos
Utilice información en tiempo real para monitorear e informar decisiones de gestión en todos los niveles. De acuerdo a la configuración y tiempo de muestreo personalizado por el usuario en los LoRa, los datos trasmitidos pueden ser almacenados y estudiados para verificar tendencias que influyan en la mejora de toma de acciones de la sociedad beneficiaria.

6. Utilice datos abiertos, estándares abiertos, código abierto, innovación abierta
Desarrolle software de código abierto por defecto con el código disponible para repositorios públicos y comunidades respaldadas por desarrolladores. El código fuente e instrucciones

propuestas en el Anexo A: "Desarrollo del driver para el uso de los dispositivos de radio LoRa con sistemas embebidos", se encuentran documentadas y fue basadas en código abierto, probado con arduino (hardware abierto, ver apartado "*Sistemas embebidos - importancia del hardware libre*") y SDIN (sistema embebido desarrollado por GEDIRCI). Además los IDE utilizados para el implementación del driver, se desarrollaron utilizando software libre (ver apartado "*Descripción del software utilizado- Importancia del software libre*"). Finalmente, el protocolo LoRa es un estándar global abierto (LoRa Alliance, 2016) que utiliza bandas de frecuencias abiertas denominadas ISM (ver *Tabla 2*).

7. Reutilice y mejore.
Desarrolle software de código abierto por defecto con el código disponible para repositorios públicos y comunidades respaldadas por desarrolladores. El código e instrucciones propuestas en el Anexo A: "Desarrollo del driver para el uso de los dispositivos de radio LoRa con sistemas embebidos" se encuentra documentadas y fue basado en código abierto, reutilizado y optimizado para brindar mayor independencia al usuario en términos de configuración y funcionamiento.

8. Aborde la privacidad y la seguridad
Tenga en cuenta el contexto y las necesidades de privacidad de la información de identificación personal en el diseño de soluciones y mitigue en consecuencia. Debido a que es un estándar, el protocolo se encuentra protegido según se presenta en las especificaciones técnicas presentadas en la *Tabla 18*. Sin embargo, el desarrollador podrá generar métodos de encriptación de datos de acuerdo con la configuración de las tramas de datos trasmitidos.

9. Sea Cooperativo
Documente trabajos, resultados, procesos y mejores prácticas, y compártalos ampliamente. El desarrollo del driver para uso con microcontroladores de los LoRa fue documentado utilizando las especificaciones técnicas del fabricante con el uso de diagramas de flujo y comentarios explícitos del porque y para qué se realizan las instrucciones.
Se documentó el proceso de configuración, uso y aplicación de estos transceivers según el documento Anexo A: "*Desarrollo del driver para el uso de los dispositivos de radio LoRa con sistemas embebidos*". Este proceso busca que cualquier usuario pueda configurar dispositivos de radio sin la necesidad de contar con conocimientos avanzados en desarrollo de software o en la interpretación de hojas de especificaciones técnicas; y busca ser un guion de configuración con la posibilidad de ajustarse según la necesidad (alcance, consumo energético, frecuencia utilizada, entre otros).

Los transceiver LoRa debido a la adaptación según uso que desee darle el usuario, el bajo consumo energético y el alto alcance de cobertura, resulta ser un estándar idóneo para diferentes áreas enmarcadas en el desarrollo.

Sin embargo, *el uso de estos dispositivos no debe ser tomado como un fin sino como un medio y deben adaptarse a las necesidades de los países en desarrollo.*

De acuerdo con los argumentos y pruebas desarrolladas en el presente documento, se confirma la hipótesis planteada en el apartado *Paso 2: Formular hipótesis e interrogantes*.
con lo que se confirma la hipótesis: "*De acuerdo a las características de bajo consumo energético, bajo costo de adquisición, fácil configuración y alto alcance de transmisión, los dispositivos LoRa resultan ser idóneos para aplicaciones del ámbito del desarrollo como mecanismo de comunicación entre sensores y dispositivos de adquisición de datos*".

BIBLIOGRAFÍA

Administración publica Ecuador (s.f.). Importancia del Software Libre para un país. Secretaría Nacional de la Administración Pública de Ecuador. Recuperado de http://www.administracionpublica.gob.ec/importancia-del-software-libre-para-un-pais/. Ecuador.

Arduino Board Due (2016). STM32 MCU Eval Tools. Recuperado de http://www.st.com/content/st_com/en/products/evaluation-tools/product-evaluation-tools/mcu-eval-tools/stm32-mcu-eval-tools.html?querycriteria=productId=SS1532, Suiza. Consultado el 29 de julio de 2016.

Arduino Board DUE (s.f.). Arduino DUE. Recuperado https://www.arduino.cc/en/Main/ArduinoBoardDue, Italia. Consultado el 29 de julio de 2016.

Arduino Board UNO (s.f.). Arduino UNO & Genuino UNO. Recuperado de https://www.arduino.cc/en/Main/ArduinoBoardUno, Italia. Consultado el 29 de julio de 2016.

Barry, R. (2009). Using the FreeRTOS Real Time Kernel - A Practical Guide. FREERTOS. Estados Unidos

Bonilla, E. & Rodríguez, P. (1997). Más allá del dilema de los métodos. La investigación en ciencias sociales. 3ª Ed, Ediciones Uniandes, Colombia.

Chapman, S. (2012). Máquinas eléctricas. 5a edición. McGraw-Hill. México.

Cisco IBSG (2011). Internet de las cosas: cómo la próxima evolución de Internet lo cambia todo. Internet Business Solutions Group. Estados Unidos.

Digital principles (s. f.). Principios para el desarrollo digital. Recuperado de http://digitalprinciples.org/. Estados Unidos. Consultado el 30 de junio de 2016.

Dorji (2015). DRF1278F - 20dBm LoRa Long Range RF Front-end Module. Version 1.11. China.

FAO - ODS (2015). Cinco agentes de cambio para un mundo sostenible. Recuperado de http://www.fao.org/post-2015-mdg/news/detail-news/es/c/281558/, Italia. Consultado el 06 de diciembre de 2015.

FAO (1994). La comunicación clave para el desarrollo humano. Repositorio de documentos de la FAO. Recuperado de http://www.fao.org/docrep/t1815s/t1815s01.htm. Italia.

FAO (2012). EL ESTADO MUNDIAL DE LA AGRICULTURA Y LA ALIMENTACIÓN: invertir en la agricultura para construir un futuro mejor. Roma. Organización de las Naciones Unidas para la alimentación y la agricultura. Italia

Genbetadev (2014). Eclipse IDE. Recuperado de http://www.genbetadev.com/herramientas/eclipse-ide, Español. Consultado el 29 de julio de 2016.

Hernández, J. (s.f.). Recopilación de la información. Conceptos de estadística y su clasificación. Departamento de ciencias básicas. Instituto Tecnológico de Apizaco. México.

Herrera J. (2011). Programación en tiempo real y bases de datos: Un enfoque práctico. Universidad de Cataluña. España

Jin-Shyan, L., Yu-Wei, S. & Chung-Chou, S. (2007). A Comparative Study of Wireless Protocols: Bluetooth, UWB, ZigBee, and Wi-Fi, The 33rd Annual Conference of the IEEE Industrial Electronics Society (IECO5). Taiwán.

Lazalde, A., Torres, J. & Vila-Viñas, D. (2015). Hardware: ecosistemas de innovación y producción basados en hardware. Buen Conocer - FLOK Society, Ecuador.

Link Labs (2016). A COMPREHENSIVE LOOK AT Low Power, Wide Area Networks for 'Internet of Things' Engineers and Decision Makers, Estados Unidos

López, E. (s.f.). Protocolo RS-485. Ingeniería en microcontroladores. México.

López, E. (s.f.). Protocolo SPI (Serial Peripherical Interface). Ingeniería en microcontroladores. México

LoRa Alliance (2016). Wide Area Network for IoT. Recuperado de https://www.lora-alliance.org. Estados Unidos. Consultado el 30 de junio de 2016.

Maxim Integrated Products (2015). MAX13487E/MAX13488E Half-Duplex RS-485-/RS-422-Compatible Transceiver with AutoDirection Control. Estados Unidos.

Monje, C. (2011). Metodología de la investigación Cuantitativa y Cualitativa. Guía didáctica. Universidad Surcolombiana. Colombia.

NiceRF (2015). LoRa1278 100 mW 4 km larga distancia y alta sensibilidad (-139 dBm) 433 MHz módulo de transceptor inalámbrico. Recuperado de http://es.aliexpress.com/item/2pcs-lot-LoRa1278-100mW-4km-Long-Distance-and-High-Sensitivity-139-dBm-433MHz-Wireless-Transceiver-Module/32461365864.html, China. Consultado el 22 de junio de 2016.

OIE - IDIE (2010). Monitoreo e indicadores. Texto de apoyo al proceso deconstrucción de un Sistema Regional de Indicadores sobre Atención y Educación Inicial. Junta de Andalucía. Instituto para el Desarrollo y la Innovación Educativa. Nicaragua.

ON semiconductor (2014). NCP1117, NCV1117 1.0A Low-Dropout Positive Fixed and Adjustable Voltage Regulators. Datasheet. Estados Unidos.

Pérez-Foguet, A., Jiménez A., (2011). El agua como elemento clave para el desarrollo. CanalEduca. España

PNUD (2001). Informe de desarrollo humano 2001: poner el adelanto tecnológico al servicio del desarrollo humano. Programa de las Naciones unidas para el Desarrollo. Estados Unidos.

PNUD (2003). Informe sobre desarrollo humano 2003. Los Objetivos de Desarrollo del Milenio: un pacto entre las naciones para eliminar la pobreza. Programa de las Naciones unidas para el Desarrollo. Estados Unidos.

Ramirez, R. (2015). Sistemas de radiocomunicaciones. Paraninfo, ciclos formativos. Sistemas de Telecomunicación e Informáticos. España.

Rojas, R. (1989). Guía para realizar investigaciones sociales, Plaza y Valdés. México.

Semtech (2013). SX1272 Development kit. User guide. Semtech Corporation. Estados Unidos

Semtech (s.f.). LoRa® Product Family. Recuperado de http://www.semtech.com/wireless-rf/lora.html, Estados Unidos. Consultado el 29 de agosto de 2016.

Semtech SX1272 (2015). SX1272/73 - 860 MHz to 1020 MHz Low Power Long Range Transceiver. Datasheet. Semtech Corporation. Estados Unidos.

Semtech SX1278 (2015). SX1276/77/78/79 - 137 MHz to 1020 MHz Low Power Long Range Transceiver. Datasheet. Semtech Corporation. Estados Unidos.

Solar Decahtlon Europe (2014). Módulos SDIN – Manual de desarrollo. Equipo de monitoreo. Francia

Spradley, J. (1980). Observación Participante. Waveland Press Inc. Estados Unidos.

ST (2016). STM32 MCU Eval Tools. Recuperado de http://www.st.com/content/st_com/en/products/evaluation-tools/product-evaluation-tools/mcu-eval-tools/stm32-mcu-eval-tools.html?querycriteria=productId=SS1532, Suiza. Consultado el 29 de julio de 2016.

Tam, J., Vera, G. & Oliveros, R. (2008). Tipos, métodos y estrategias de investigación. Pensamiento y Acción 5, pp. 145-154. Perú.

Texas Instruments (2004). LP2985 150-mA Low-noise Low-dropout Regulator with Shutdown. Datasheet. Estados Unidos.

Texas Instruments (2005). SM-Band and Short Range Device Regulatory Compliance Overview. Estados Unidos.

Torrente, O. (2013). ARDUINO: Curso práctico de formación. RC libros. España.

Velásquez, J. (s.f.). Los sensores en la producción. Universidad Ricardo Palma. Perú.

ANEXO A

Desarrollo del driver para el uso de los dispositivos de radio LoRa con sistemas embebidos.

JOSÉ DANIEL RODRÍGUEZ MUNCA

Tutor Académico: **Manuel Sierra Castañer**

Tutor Profesional: **Álvaro Gutiérrez Martín**

UNIVERSIDAD DE POLITÉCNICA DE MADRID
UNIVERSIDAD COMPLUTENSE DE MADRID
MASTER INTERUNIVERSITARIO EN ESTRATEGIAS Y TECNOLOGÍAS PARA EL DESARROLLO
MADRID - ESPAÑA
2016

Tabla de Contenido

CONFIGURACIÓN A NIVEL DE HARDWARE 6
Lectura y escritura de registros SPI 12
Modos de operación de los LoRa 12
CONFIGURACIÓN POR SOFTWARE DE LOS RADIO LORA 15
Comprobación de comunicación entre LoRa y MC 16
Registros de configuración SF, BW y CR del radio LoRa 16
Registros de configuración de potencia del radio 24
Registros de configuración de frecuencia del radio 28
Configuración por software del radio lora para la trasmisión datos 29
Configuración por software del radio lora para la recepción de datos 34
Ejemplo de aplicación: red punto a punto (aplicación ping-pong) 38
Ejemplo de aplicación: recepción de datos de varios dispositivos remotos (red punto - multipunto) 40
BIBLIOGRAFÍA 47

LISTADO DE FIGURAS

Figura 1. Principales características de comunicación SPI para los dispositivos LoRa seleccionados 6
Figura 2. Tarjeta SX1272RF1 de Semtech (868Mhz) 8
Figura 3.Tarjeta LoRa1278 de NiceRF (433Mhz) 8
Figura 4. Configuración de pines de control para la tarjeta SX1272RF1 de Semtech 9
Figura 5. Configuración de pines de control para la tarjeta DRF1278F de Dorji 9
Figura 6. Configuración de pines de control para la tarjeta LoRa1278 de NiceRF 9
Figura 7. Esquema de configuración de pines de control y SPI para el microcontrolador 10
Figura 8. Diagrama de tiempos para iniciar chip SX1272 y SX1278 11
Figura 9. Flujograma propuesto para iniciar chip SX1272 y SX1278 11
Figura 10. Flujograma propuesto para la lectura y escritura de registros en los radio SX1272 y SX1278 12
Figura 11. Configuración de los modos de operación en registro RegOpMode (0x01) 14
Figura 12. Configuración de modo LoRa en registro RegOpMode (0x01) 14
Figura 13. Esquema propuesto para inicio en la configuración del LoRa 15
Figura 14. Comprobación de comunicación SPI entre el microcontrolador y el LoRa seleccionado. 16
Figura 15. Esquema para determinar el TS según BW y SF 21
Figura 16. Esquema lógico de configuración de los registros RegModemConfig1_0x1D y RegModemConfig2_0x1E para SX1272 22
Figura 17. Esquema lógico de configuración de los registros RegModemConfig1_0x1D, RegModemConfig2_0x1E y RegModemConfig3_0x26 para SX1278 23
Figura 18. Esquema lógico de configuración de los registros RegLna_0x0C, RegPaConfig_0x09, RegOcp_0x0B y RegPaDac para los dispositivos LoRa 27
Figura 19. Esquema lógico de configuración de frecuencia para los radio LoRa. 29
Figura 20. Esquema lógico de almacenamiento de datos sobre la FIFO del LoRa 32
Figura 21. Esquema lógico de configuración para transmisión de datos en el LoRa 33
Figura 22. Esquema lógico propuesto para gestión por interrupción cuando se efectua una trasmisión de datos en el LoRa 34
Figura 23. Esquema lógico de configuración en el LoRa para recepción de datos 35
Figura 24. Esquema lógico de atención a la interrupción por recepción o trasmisión de datos en el LoRa 36
Figura 25. Esquema lógico de almacenamiento de datos sobre la FIFO del LoRa. 37
Figura 26. Esquema lógico para dispositivo maestro utilizando radio LoRa. 38
Figura 27. Esquema lógico para dispositivo esclavo utilizando radio LoRa. 39
Figura 28. Principio de funcionamiento "ping-pong" para la realización de pruebas de alcance en dispositivos LoRa 40
Figura 29. Tipología estrella para comunicación con diversos dispositivos 41
Figura 30. Líneas de tiempos para dispositivos esclavos y dispositivo maestro sin estrategias adecuadas para la trasmisión de información 42

Figura 31. Líneas de tiempos para dispositivos esclavos y dispositivo maestro con tiempos aleatorios para la trasmisión adecuada de datos 43
Figura 32. Esquema lógico para dispositivo maestro en aplicación punto-multipunto 44
Figura 33. Esquema lógico para dispositivos esclavo en aplicación punto-multipunto..... 45

LISTADO DE TABLAS

Tabla 1. Comparación de pines de control según los fabricantes de tarjetas de los LoRa seleccionados 7
Tabla 2. Descripción del uso de los pines RXEN y TXEN para la tarjeta LoRa1278 de NiceRF 7
Tabla 3. Configuración de pines de control para los LoRa seleccionados 10
Tabla 4. Descripción del registro RegOpMode (0x01) 13
Tabla 5. Descripción del registro modo de operación del LoRa 13
Tabla 6. Descripción del registro RegModemConfig1_0x1D del LoRa SX1272 17
Tabla 7. Descripción del registro RegModemConfig2_0x1E del LoRa SX1272 18
Tabla 8. Descripción del registro RegModemConfig1_0x1D del LoRa SX1278 18
Tabla 9. Descripción del registro RegModemConfig2_0x1E del LoRa SX1278 19
Tabla 10. Descripción del registro RegModemConfig3_0x26 del LoRa SX1278 20
Tabla 11. Descripción del registro RegLna_0x0C del LoRa 24
Tabla 12. Descripción del registro RegPaConfig_0x09 del LoRa 25
Tabla 13. Descripción del registro RegPaConfig_0x09 del LoRa 25
Tabla 14. Descripción del registro RegOcp_0x0B del LoRa 26
Tabla 15. Descripción del registro RegPaDac del LoRa* 26
Tabla 16. Banda de frecuencia ISM compatibles con los transceivers de radio LoRa 28
Tabla 17. Descripción del registro de configuración de frecuencia en los radio LoRa 28
Tabla 18. Descripción del registro RegIrqFlagsMask_0x11 para habilitación de interrupciones 30
Tabla 19. Descripción del registro RegIrqFlags_0x12 para la detección de interrupciones por software 30
Tabla 20. Configuración de pines de control con el registro RegDioMapping1_0x40 para el uso con interrupciones en los pines DIO0-3 31
Tabla 21. Configuración de pines de control con el registro RegDioMapping2_0x41 para el uso con interrupciones en los pines DIO4-5 31

CONFIGURACIÓN A NIVEL DE HARDWARE

Los parámetros de configuración de los transceivers LoRa se establecen modificando registros internos del chip mediante el protocolo de comunicación síncrono SPI.
A nivel lógico de sincronización y trasmisión de datos, la comunicación SPI requiere la configuración de la polaridad del reloj (CPOL) y el bit fase del reloj (CPHA).

Los dispositivos de radio LoRa SX1272/78 utilizan CPOL=0 y CPHA=0; lo que se conoce como modo 0 SPI. El bit más significativo (MSB) de un byte enviado debe ser el primero y la velocidad del SCLK no debe superar los 10MHz (Semtech-SX1278, 2015).

El esquema propuesto de conexión entre los transceiver LoRa y el microcontrolador (sistema embebido) se presenta en la *Figura 1*. Los dispositivos de radio LoRa se comportan como esclavos y el microcontrolador será el maestro en la comunicación SPI.

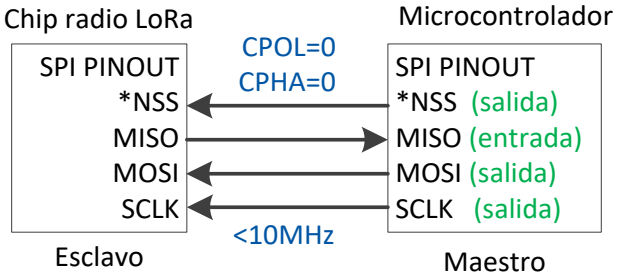


Figura 1. Principales características de comunicación SPI para los dispositivos LoRa seleccionados
Fuente: Semtech SX1272 (2015) & Semtech SX1278 (2015). Desarrollado por autor

Algunos fabricantes realizan tarjetas de adecuación para facilitar el manejo de los dispositivos de radio LoRa. Al realizar estas tarjetas, se eliminan o incorporan pines físicos de control respecto a las características originales del chip. La siguiente tabla presenta las tarjetas propuestas para realizar el driver de los radio SX1272/78. Los fabricantes mantienen los pines de comunicación SPI y eliminan algunos pines de control del chip. Las principales características de las tarjetas LoRa adquiridas para realizar las pruebas de funcionamiento y desarrollo del driver, se presentan en la *Tabla 5*.

Tabla 1. Comparación de pines de control según los fabricantes de tarjetas de los LoRa seleccionados
Fuente: Semtech (2013), Dorji (2015) & NiceRF (2015)

<table>
<tr><th rowspan="2">Tarjeta</th><th rowspan="2">Chip radio</th><th rowspan="2">Marca</th><th colspan="4">Pines de control físicos</th></tr>
<tr><th>DIO0 – DIO5</th><th>RXTX</th><th>Pines SPI</th><th>RESET</th></tr>
<tr><td>SX1272RF1</td><td>SX1272</td><td>Semtech</td><td rowspan="2">Pines de entrada/ salida, configurables por software</td><td>Pin de salida. Se coloca en “1” cuando el dispositivo está trasmitiendo datos</td><td rowspan="3">NSS
MISO
MOSI
SCLK</td><td rowspan="3">Pin de entrada. Utilizado al iniciar o arrancar el dispositivo.</td></tr>
<tr><td>DRF1278F</td><td rowspan="2">SX1278</td><td>Dorji</td><td rowspan="2">NI*</td></tr>
<tr><td>LoRa1278</td><td>NiceRF</td><td>Pines de entrada/ salida, configurables por software
NI* DIO3–DIO5</td></tr>
</table>

*NI: No implementado

La versión LoRa1278 de NiceRF realiza adecuaciones con la incorporación de 2 pines adiciones de control que permiten habilitar el uso de la antena para trasmitir o recibir. La descripción de las funciones de los pines TXEN y RXEN se presentan en la *Tabla 2*.

Tabla 2. Descripción del uso de los pines RXEN y TXEN para la tarjeta LoRa1278 de NiceRF
Fuente: NiceRF (2015)

Estados lógicos de pines		Descripción de funcionamiento para la Antena
Pin TXEN	Pin RXEN	
0	0	Modo Sleep o bajo consumo
0	1	Configurada para Recibir
1	0	Configurada para Transmitir

Físicamente, la tarjeta SX1272RF1 de *Semtech* es de mayor tamaño y cuenta con un conector SMA para antenas a 868Mhz e impedancia de 50Ω. Las tarjetas DRF1278F y LoRa1278 tienen el mismo tamaño (significativamente más pequeño que la versión SX1272RF1) y permiten conectar antenas pasivas tipo primavera a 433Mhz con impedancia de 50Ω. Los tamaños de pueden observar en las *Figura 2* y *Figura 3*.

Figura 2. Tarjeta SX1272RF1 de Semtech (868Mhz)
Fuente: Fotografía tomada por autor

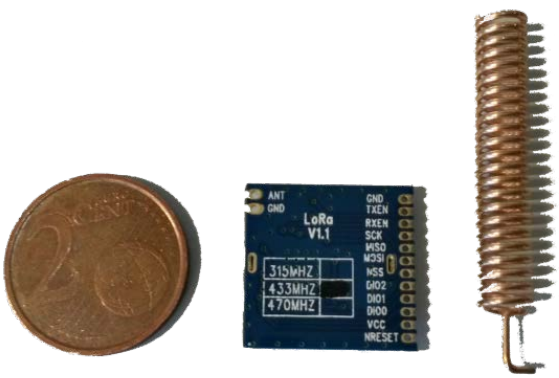


Figura 3.Tarjeta LoRa1278 de NiceRF (433Mhz)
Fuente: Fotografía tomada por autor

Las *Figura 4*, *Figura 5* y *Figura 6* presentan la conexión básica de los pines de control para las tres tarjetas LoRa propuestas y la configuración del microcontrolador en relación a pines

de entrada o salida. La configuración de los pines es una de las tareas prioritarias al iniciar cualquier aplicación con sistemas embebidos.

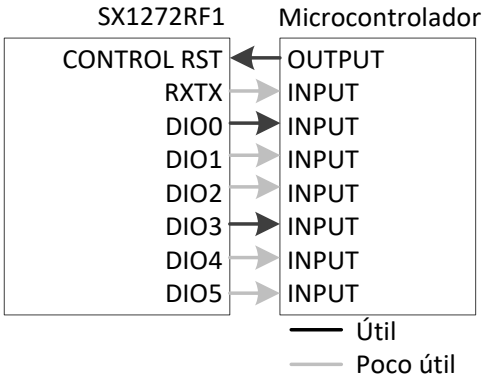


Figura 4. Configuración de pines de control para la tarjeta SX1272RF1 de Semtech
Fuente: Semtech SX1272 (2015). Desarrollada por autor

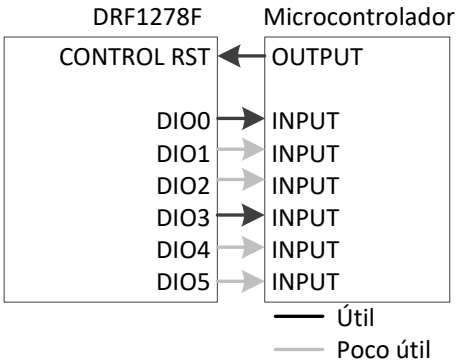


Figura 5. Configuración de pines de control para la tarjeta DRF1278F de Dorji
Fuente: Dorji (2015). Desarrollada por autor

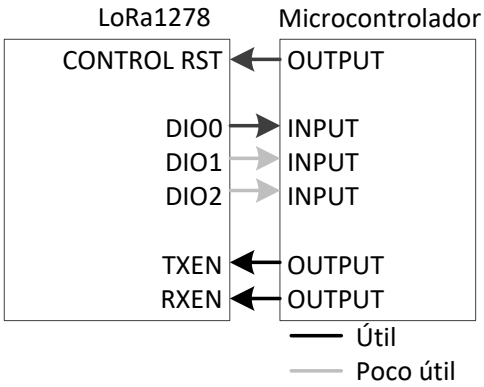


Figura 6. Configuración de pines de control para la tarjeta LoRa1278 de NiceRF
Fuente: NiceRF (2015). Desarrollada por autor

Los pines DIO de control más importantes de acuerdo la información suministrada por el fabricante, son los DIO0 y DIO3. La tabla 7 presenta las funciones de los pines de control de acuerdo a la configuración.

Tabla 3. Configuración de pines de control para los LoRa seleccionados
Fuente: Semtech SX1272 (2015) & Semtech SX1278 (2015)

Conf. de mapeo	DIO5	DIO4	DIO3	DIO2	DIO1	DIO0
00	ModeReady	CadDetected	CadDone	FhssChangeChannel	RxTimeout	RxDone
01	ClkOut	PllLock	ValidHeader	FhssChangeChannel	FhssChangeChannel	TxDone
10	ClkOut	PllLock	PayloadCrcError	FhssChangeChannel	CadDetected	CadDone
11	Sin uso					

De acuerdo a la configuración, el pin DIO0 permite conocer si el chip radio LoRa ha trasmitido correctamente los datos (TXDONE) o si el chip ha recibido datos y está a la espera de la gestión por parte del microcontrolador (RXDONE).
El DIO3 permite conocer si los datos llegaron correctamente de acuerdo con la Verificación por redundancia cíclica (CRC).

Teniendo en cuenta lo anterior, se plantea en la figura 8 el esquema de configuración de pines de control y pines SPI para el microcontrolador:

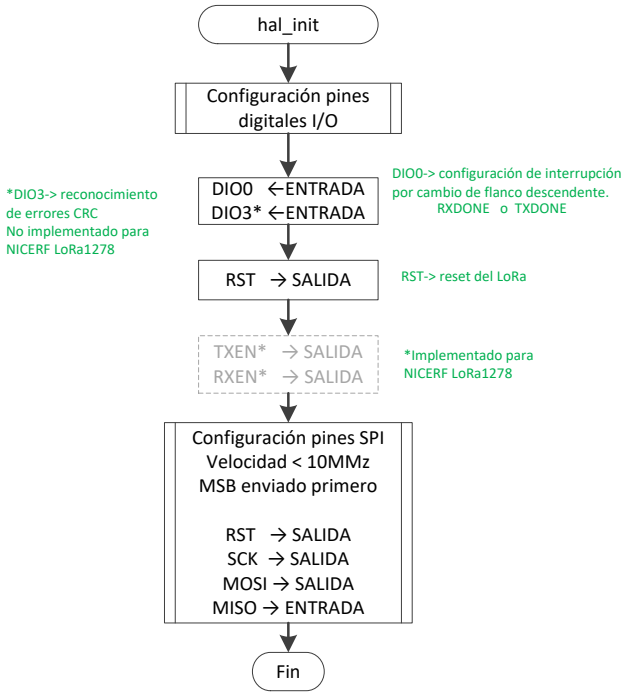


Figura 7. Esquema de configuración de pines de control y SPI para el microcontrolador
Fuente: Semtech SX1272 (2015) & Semtech SX1278 (2015). Desarrollado por autor

Una vez se tengan configurados los pines SPI y de control en el microcontrolador, el fabricante presenta el protocolo de inicio o arranque del chip LoRa para el SX1272 y para el SX1278 (ver *Figura 8*). La principal diferencia el estado del pulso para el reset del chip, para el SX1272 se realiza con un estado lógico "0", mientras que para el SX1278 se efectúa con un estado lógico "1" durante 100 microsegundos. A nivel lógico, la *Figura 9* presenta el protocolo de inicio a nivel de hardware del chip LoRa.

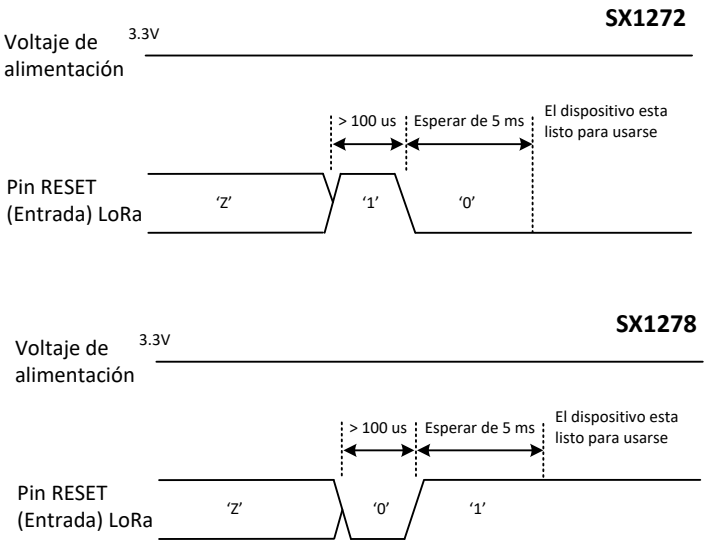


Figura 8. Diagrama de tiempos para iniciar chip SX1272 y SX1278
Fuente: Semtech SX1272 (2015) & Semtech SX1278 (2015). Desarrollado por autor

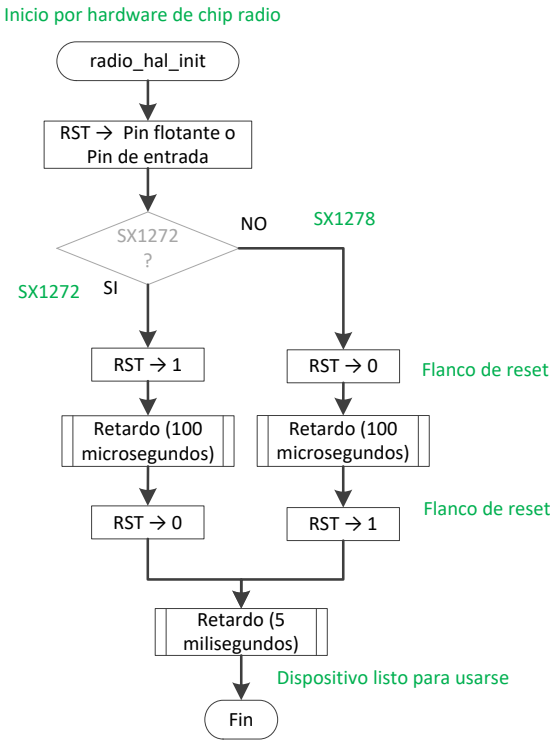


Figura 9. Flujograma propuesto para iniciar chip SX1272 y SX1278
Fuente: Semtech SX1272 (2015) & Semtech SX1278 (2015). Desarrollado por autor

Lectura y escritura de registros SPI

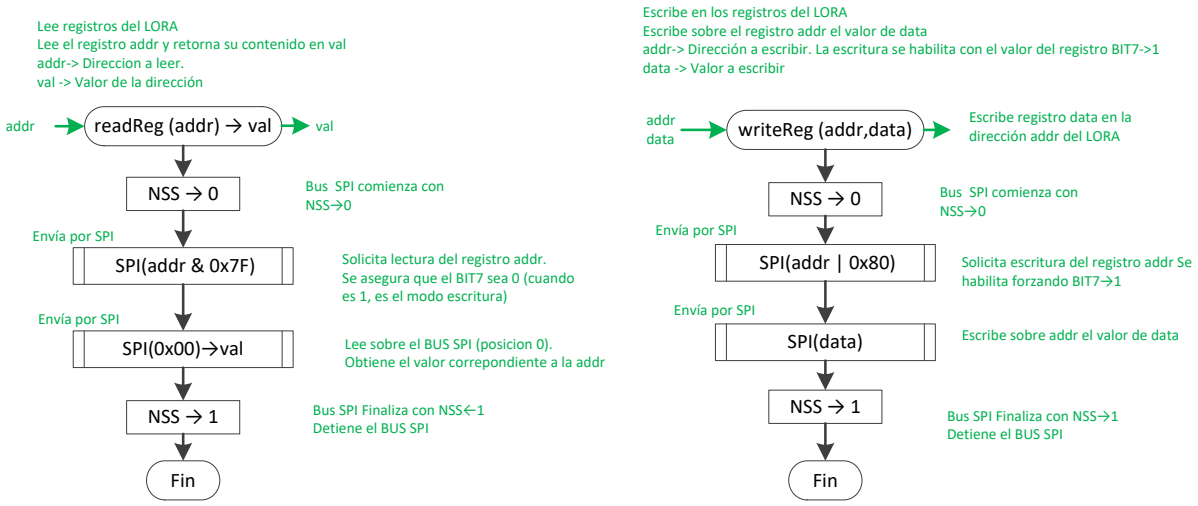


Figura 10. Flujograma propuesto para la lectura y escritura de registros en los radio SX1272 y SX1278
Fuente: Semtech SX1272 (2015) & Semtech SX1278 (2015). Desarrollado por autor

La lectura y escritura de registros en los radio LoRa se realiza utilizando el bus SPI (siempre desde los modos SLEEP o STANDBY). Para realizar la lectura, se escribe sobre el bus SPI la dirección del registro, forzando que el bit 7 sea “0” y luego se hace escritura con el valor “cero”, obteniendo el valor del registro.

Para realizar la escritura, se escribe sobre el bus SPI la dirección del registro forzando que el bit 7 (MSB) sea “1” y luego se hace la escritura del dato. Lo anterior se presenta en el flujograma de la *Figura 10*.

Modos de operación de los LoRa

El registro *RegOpMode_0x01* permite configurar el modo de operación del dispositivo LoRa. Este registro se podría considerar como el más usado en cualquier aplicación. La descripción del registro *RegOpMode* se presenta en la *Tabla 4*.

Tabla 4. Descripción del registro RegOpMode (0x01)
Fuente: Semtech SX1272 (2015) & Semtech SX1278 (2015)

Bit	Nombre de la variable	Uso	Defecto	Descripción
7	LongRangeMode	r w	0x00	0 → FSK/OOK Modo 1 → LoRa™ Modo
6	AccessSharedReg	r w	0x00	0 → Acceso registros LoRa 1 → Acceso registros FSK
5-4	reserved	r	0x00	Reservado
3	LowFrequencyModeOn	r w	0x01	0 → Modo High Frequency Mode (Acceso registros HF) 1 → Modo Low Frequency Mode (Acceso registros LF)
2-0	Mode	r w	0x01	**Modos del dispositivo** 000 → SLEEP 001 → STDBY 010 → Frequency synthesis TX (FSTX) 011 → Transmit (TX) 100 → Frequency synthesis RX (FSRX) 101 → Receive continuous (RXCONTINUOUS) 110 → receive single (RXSINGLE) 111 → Channel activity detection (CAD)

Nota r: lectura. w: escritura. La variable LowFrequencyModeOn aplica para el SX1278 (SX1272 no está implementado)

Los modos de operación del LoRa más utilizados se presentan en la *Tabla 5* y se habilitan siguiendo el diagrama de la *Figura 11*. Los modos de operación utilizados para la configuración de los dispositivos LoRa son SLEEP y STANDBY.

Tabla 5. Descripción del registro modo de operación del LoRa
Fuente: Semtech SX1272 (2015) & Semtech SX1278 (2015)

Modo	Descripción	Valor de 'Mode' en RegOpMode
SLEEP	Es el modo de bajo consume. Solo los registros SPI del LoRa tienen acceso en este modo. La FIFO no tiene acceso	0x00
STANDBY	El LoRa se coloca en este modo al transmitir o tener pendiente la gestión en la recepción de datos. Los registro SPI y de la FIFO del LoRa tienen acceso en este modo.	0x01
TX	Es el modo utilizado cuando se tienen configurada la interrupción por TX, datos en la FIFO y el LoRa está listo para trasmitir. Una vez transmite, el LoRa cambia automáticamente al modo STANDBY.	0x03
RXCONTINUOUS	Es el modo utilizado cuando se tienen configurada la interrupción por RX, el LoRa tiene datos en la FIFO y el chip está a espera de la gestión de los datos. Una vez el LoRa recibe los datos y valida los datos (CRC), el LoRa cambia automáticamente al modo STANDBY. Recibe varios bytes en una trama.	0x05

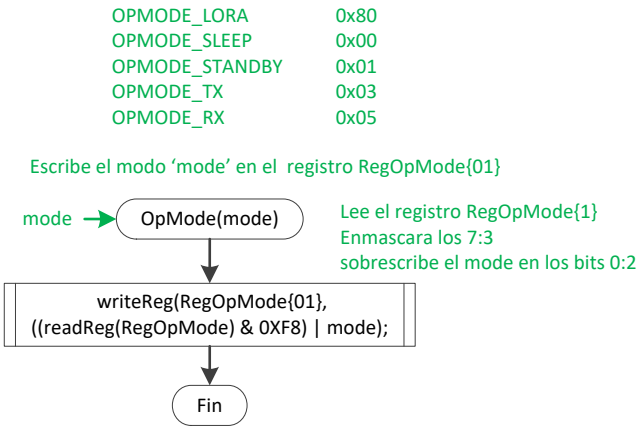


Figura 11. Configuración de los modos de operación en registro RegOpMode (0x01)
Fuente: Semtech SX1272 (2015) & Semtech SX1278 (2015). Desarrollado por autor

Para realizar cualquier configuración sobre los registros SPI del LoRa es necesario que el modo de operación pase de SLEEP a STANDBY.

Para usar el "*modo LoRa*", se debe escribir inicialmente sobre el registro RegOpMode_0X01 un "1" en el bit 7 que corresponde a la variable LongRangeMode (ver *Tabla 4*). Basta con habilitar dicho bit una sola vez al iniciar cualquier aplicación (ver *Figura 12*).

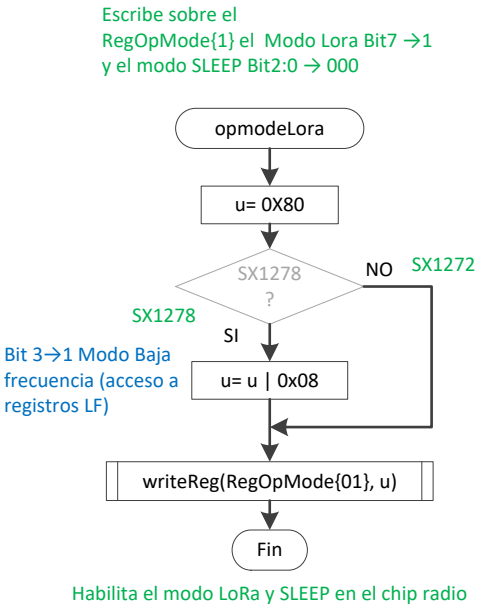


Figura 12. Configuración de modo LoRa en registro RegOpMode (0x01)
Fuente: Semtech SX1272 (2015) & Semtech SX1278 (2015). Desarrollado por autor

CONFIGURACIÓN POR SOFTWARE DE LOS RADIO LORA

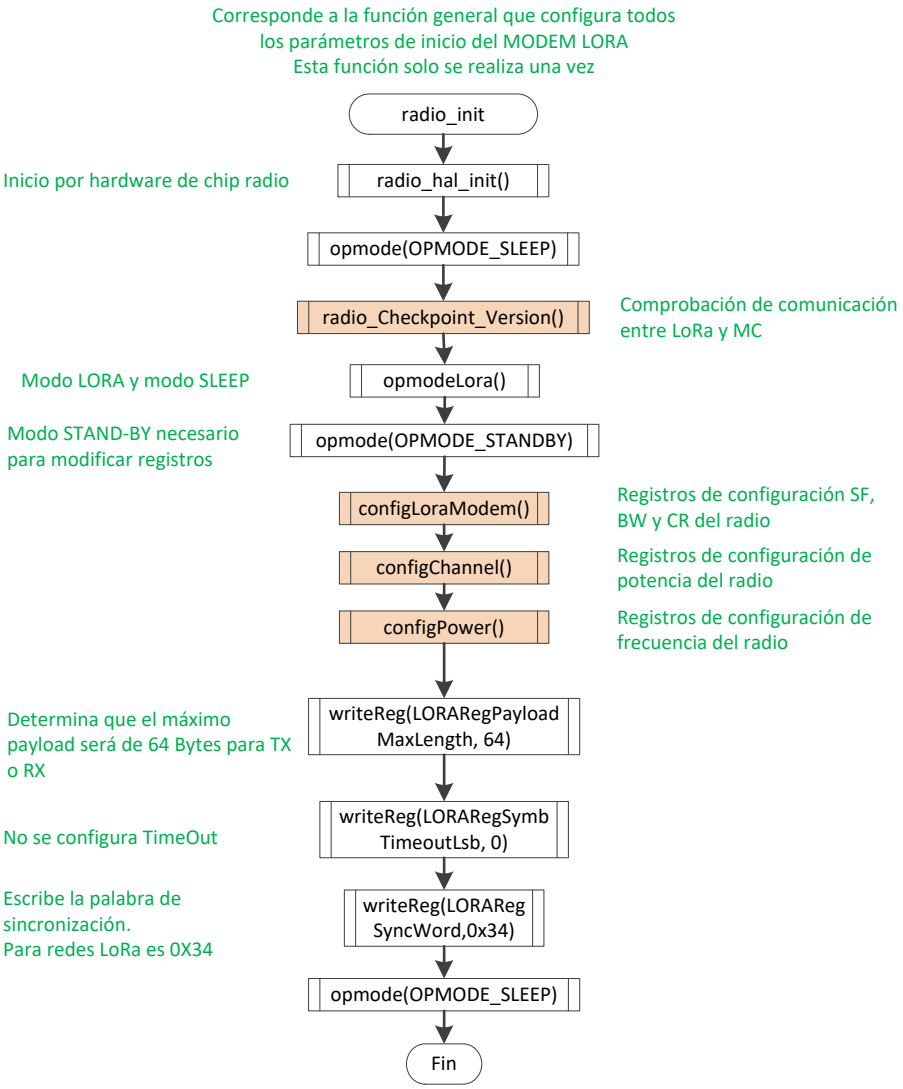


Figura 13. Esquema propuesto para inicio en la configuración del LoRa
Fuente: Semtech SX1272 (2015) & Semtech SX1278 (2015). Desarrollado por autor

La *Figura 13* presenta el esquema lógico propuesto para el inicio en la configuración de los dispositivos LoRa. El conjunto de instrucciones puede ser ejecutado al alimentar el sistema embebido (microcontrolador) y los LoRa. El esquema lógico de instrucciones basta con ser ejecutado una sola vez por el microcontrolador. Las funciones *radio_Checkpoint_Version()*, *configLoraModem()*, *configChannel()* y *configPower()* se presentan en los siguientes apartados.

Comprobación de comunicación entre LoRa y MC

Se recomienda realizar la comprobación de la versión de revisión del chip de radio LoRa. Esto permitirá reconocer si la comunicación entre el microcontrolador y el chip radio LoRa es correcta y si se está configurando el dispositivo correctamente. La *Figura 14* presenta el esquema lógico propuesto.

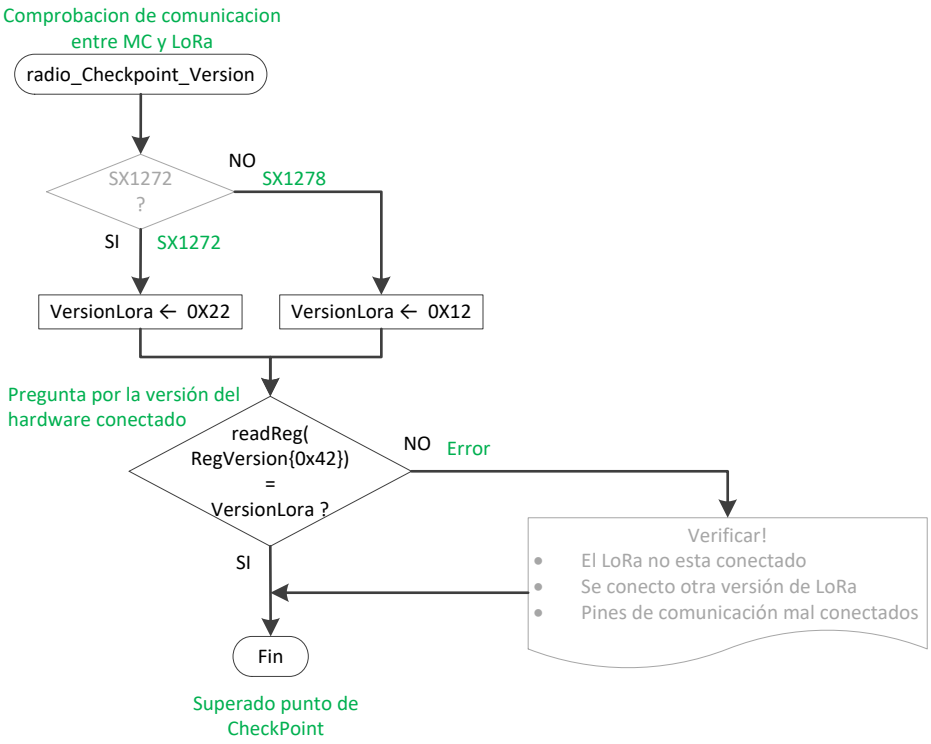


Figura 14. Comprobación de comunicación SPI entre el microcontrolador y el LoRa seleccionado.
Fuente: Semtech SX1272 (2015) & Semtech SX1278 (2015). Desarrollado por autor

Registros de configuración SF, BW y CR del radio LoRa

Los dispositivos LoRa cuentan con los parámetros *SF*, *BW* y *CR* que según las combinaciones entre estos, permite determinar el alcance o distancia máxima de recepción de datos, la velocidad de los datos y sobrecarga por corrección o detección de errores en los datos. Estos valores deben ser los mismos en todos los dispositivos radio LoRa de la red inalámbrica.

Los valores de SF, BW y CR se establecen en los registros *RegModemConfig1_0x1D* y *RegModemConfig2_0x1E* para los radio SX1272 y SX1278. Se incorpora un registro adicional denominado *RegModemConfig3_0x26* para el SX1278.

Existen otros parámetros de interés en estos registros de configuración, como las variables ImplicitHeaderModeOn (modo del encabezado), RxPayloadCrcOn (habilitación de errores por verificación CRC), AgcAutoOn (ajuste en el amplificador adicional LNA) y LowDataRateOptimize (bit de optimización de datos a baja velocidad).

La *Tabla 6* y *Tabla 7* presentan la descripción de los registros *RegModemConfig1_0x1D* y *RegModemConfig2_0x1E* respectivamente para el SX1272. La *Tabla 8*, *Tabla 9* y *Tabla 10* presenta la descripción de los registros *RegModemConfig1_0x1D*, *RegModemConfig2_0x1E* y *RegModemConfig3_0x26* para el SX1278.

Tabla 6. Descripción del registro RegModemConfig1_0x1D del LoRa SX1272
Fuente: Semtech SX1272 (2015)

Bit	Nombre de la variable	Uso	Defecto	Descripción
7-6	Bw	r w	0x00	Ancho de banda (entre más bajo sea este valor, el tiempo de durante la trasmisión será mayor)
				00 → 125 kHz
				01 → 250 kHz
				10 → 500 kHz
				11 → reservado
5-3	CodingRate	r w	‘001’	Tasa de codificación de errores: Entre mayor sea este valor, mayor será la fiabilidad de los datos, pero con una sobrecarga en el tiempo de transmisión.
				001 → 4/5
				010 → 4/6
				011 → 4/7
				100 → 4/8
				Otros valores→ reservado
2	ImplicitHeaderModeOn	r w	0x00	Tipo de Encabezado utilizado por el LoRa. Se recomiendo el *encabezado explicito* por contar con información adicional como la cantidad de bytes trasmitidos/recibidos y si contienen errores de CRC.
				0 → Modo del Header explicito
				1 → Modo del Header implícito
1	RxPayloadCrcOn	r w	0x0	Habilita la generación de CRC Verificación por redundancia cíclica en la recepción de los datos
				0 → CRC no habilitado
				1 → CRC habilitado
0	LowDataRateOptimize	r w	0x0	Habilita la optimización en la recepción de los datos cuanto esta configurado el LoRa en baja trasmisión de velocidad de los datos
				0 → no habilitado
				1 → habilitado. Solo debe ser habilitado cuando está configurado SF11 o SF12 a un BW=125kHz

Nota r: lectura. w: escritura.

Tabla 7. Descripción del registro RegModemConfig2_0x1E del LoRa SX1272
Fuente: Semtech SX1272 (2015)

Bit	Nombre de la variable	Uso	Defecto	Descripción
7-4	SpreadingFactor	r w	0x07	SF rate Es el factor de alcance expresado como logaritmo de base 2. Entre mayor sea este valor, se tendrá un mejor rendimiento en la trasmisión de datos.
				6 → 64 chips / symbol (modo utilizado para FSK)
				7 → 128 chips / symbol
				8 → 256 chips / symbol
				9 → 512 chips / symbol
				10 → 1024 chips / symbol
				11 → 2048 chips / symbol
				12 → 4096 chips / symbol
				Otros valores→ reservado
3	TxContinuousMode	r w	0x0	0 → modo normal de trasmisión simple de paquetes.
				1 → modo continúo de trasmisión. Múltiples paquetes enviados a través de la FIFO.
2	AgcAutoOn	r w	0x00	Habilita la generación de CRC Verificación por redundancia cíclica en la recepción de los datos.
				0 → Ajuste de ganancia en el LNA según el registro LNA LnaGain
				1 → Ajuste de ganancia en el LNA por loop interno en el AGC
1-0	SymbTimeout(9:8)	r w	0x00	Timeout para la espera de datos. Se recomienda 0.

Nota r: lectura. w: escritura.

Tabla 8. Descripción del registro RegModemConfig1_0x1D del LoRa SX1278
Fuente: Semtech SX1278 (2015)

Bit	Nombre de la variable	Uso	Defecto	Descripción
7-4	Bw	r w	0x07	Ancho de banda (entre más bajo sea este valor, el tiempo de durante la trasmisión será mayor)
				0000 → 7.8 kHz
				0001 → 10.4 kHz
				0010 → 15.6 kHz
				0011 → 20.8kHz
				0100 → 31.25 kHz
				0101 → 41.7 kHz
				0110 → 62.5 kHz
				0111 → 125 kHz
				1000 → 250 kHz
				1001 → 500 kHz
				Otros valores→ reservado

Bit	Nombre de la variable	Uso	Defecto	Descripción
3-1	CodingRate	r w	'001'	Tasa de codificación de errores: Entre mayor sea este valor, mayor será la fiabilidad de los datos, pero con una sobrecarga en el tiempo de transmisión.
				001 → 4/5
				010 → 4/6
				011 → 4/7
				100 → 4/8
				Otros valores→ reservado
0	ImplicitHeaderModeOn	r w	0x00	Tipo de Encabezado utilizado por el LoRa. Se recomiendo el *encabezado explicito* por contar con información adicional como la cantidad de bytes trasmitidos/recibidos y si contienen errores de CRC.
				0 → Modo del Header explicito
				1 → Modo del Header implícito

Nota r: lectura. w: escritura.

Tabla 9. Descripción del registro RegModemConfig2_0x1E del LoRa SX1278
Fuente: Semtech SX1278 (2015)

Bit	Nombre de la variable	Uso	Defecto	Descripción
7-4	SpreadingFactor	r w	0x07	SF rate Es el factor de alcance expresado como logaritmo de base 2. Entre mayor sea este valor, se tendrá un mejor rendimiento en la trasmisión de datos.
				6 → 64 chips / symbol (modo utilizado para FSK)
				7 → 128 chips / symbol
				8 → 256 chips / symbol
				9 → 512 chips / symbol
				10 → 1024 chips / symbol
				11 → 2048 chips / symbol
				12 → 4096 chips / symbol
				Otros valores→ reservado
3	TxContinuousMode	r w	0x0	0 → modo normal de trasmisión simple de paquetes.
				1 → modo continúo de trasmisión. Múltiples paquetes enviados a través de la FIFO.
2	RxPayloadCrcOn	r w	0x00	Habilita la generación de CRC Verificación por redundancia cíclica en la recepción de los datos.
				0 → CRC no habilitado
				1 → CRC habilitado
1-0	SymbTimeout(9:8)	r w	0x00	Timeout para la espera de datos. Se recomienda 0.

Nota r: lectura. w: escritura.

Tabla 10. Descripción del registro RegModemConfig3_0x26 del LoRa SX1278
Fuente: Semtech SX1278 (2015)

Bit	Nombre de la variable	Uso	Defecto	Descripción
7-4	Sin uso	r	0x00	reservado
3	LowDataRateOptimize	r w	0x0	Habilita la optimización en la recepción de los datos cuanto esta configurado el LoRa en baja trasmisión de velocidad de los datos
				0 → no habilitado
				1 → habilitado. Solo debe ser habilitado cuando excede los 16ms en el tiempo por símbolo T_S.
2	AgcAutoOn	r w	0x00	Habilita la generación de CRC Verificación por redundancia cíclica en la recepción de los datos.
				0 → Ajuste de ganancia en el LNA según el registro LNA LnaGain
				1 → Ajuste de ganancia en el LNA por loop interno en el AGC
1-0	Sin uso	r	0x00	reservado

Nota r: lectura. w: escritura.

El registro *RegModemConfig3_0x26* aplica únicamente para el chip radio SX1278. Una de las variables de este registro es el bit de *LowDataRateOptimize*. El criterio para habilitar este bit corresponde a que se exceda los 16ms en el tiempo por símbolo (T_S). La ecuación para el cálculo de este parámetro la comparte el fabricante y se presenta en (1).

$$T_S = \frac{2^{SF}}{BW} \quad (1)$$

El T_S se define como el periodo o tiempo en el aire de cada símbolo.

La *Figura 15* presenta es esquema lógico propuesto para determinar el TS según los valores de SF y BW.

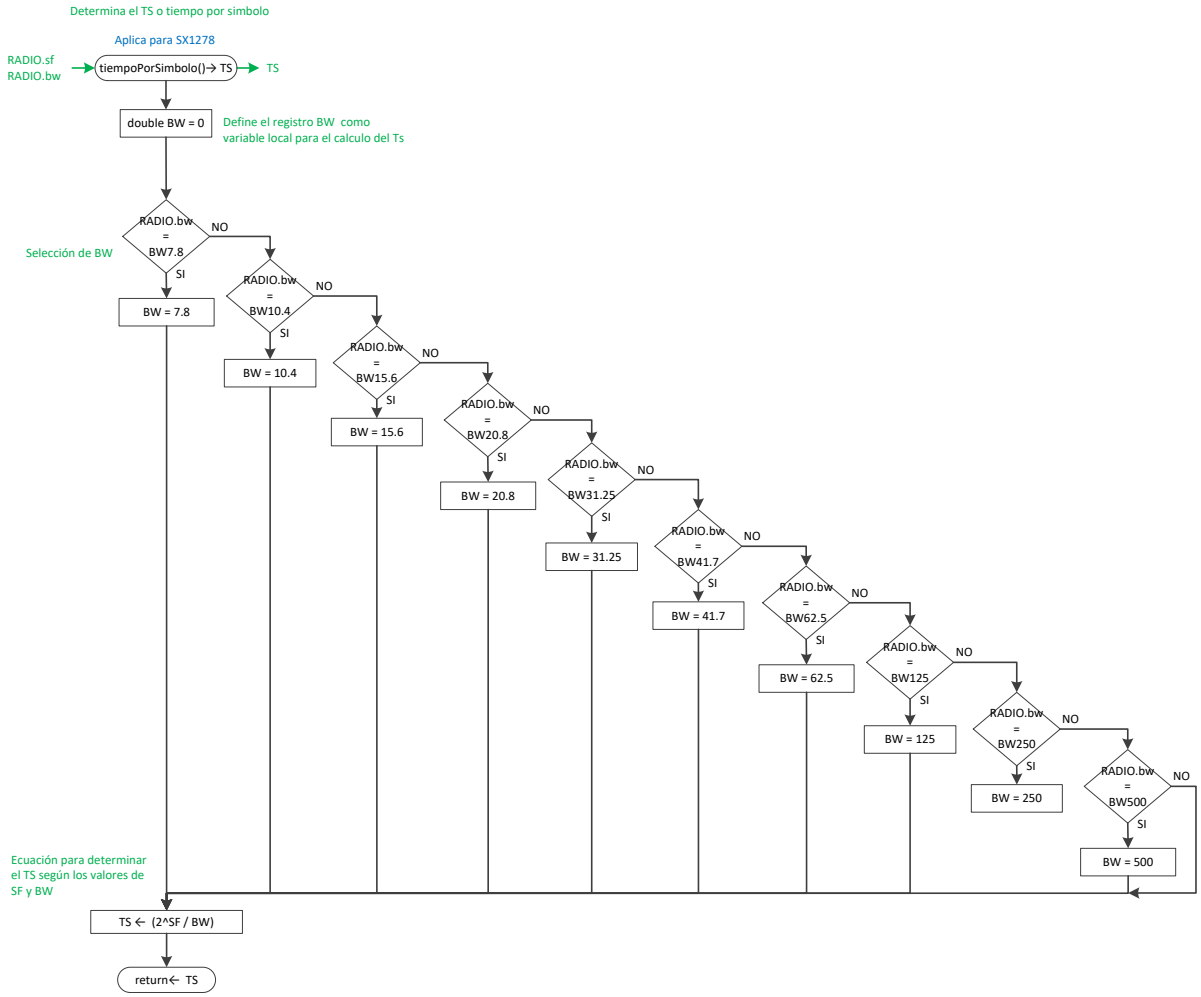


Figura 15. Esquema para determinar el TS según BW y SF
Fuente: Semtech SX1272 (2015) & Semtech SX1278 (2015). Desarrollado por autor

Para la configuración de los registros *RegModemConfig1_0x1D*, *RegModemConfig2_0x1E* y *RegModemConfig3_0x26* (aplica para el SX1278), se requiere de los valores RADIO.sf, RADIO.bw y RADIO.cr definidos por el usuario.

El flujograma presentado en la *Figura 16* representa el esquema propuesto de configuración para los registros *RegModemConfig1*_0x1D y *RegModemConfig2_0x1E* para el chip radio SX1272. El flujograma presentado en la *Figura 17* representa el esquema propuesto de configuración de registros *RegModemConfig1_0x1D*, *RegModemConfig2_0x1E* y *RegModemConfig3_0x26* para el chip radio SX1278.

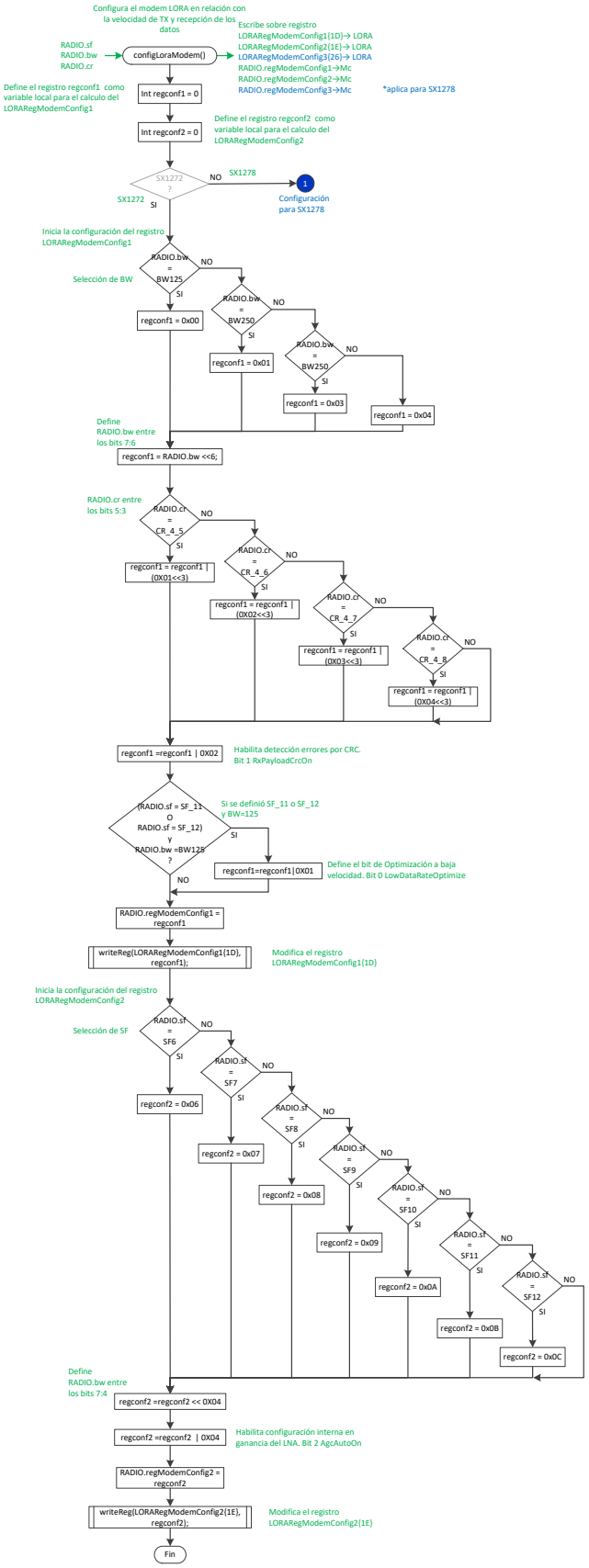


Figura 16. Esquema lógico de configuración de los registros RegModemConfig1_0x1D y RegModemConfig2_0x1E para SX1272

Fuente: Semtech SX1272 (2015) & Semtech SX1278 (2015). Desarrollado por autor

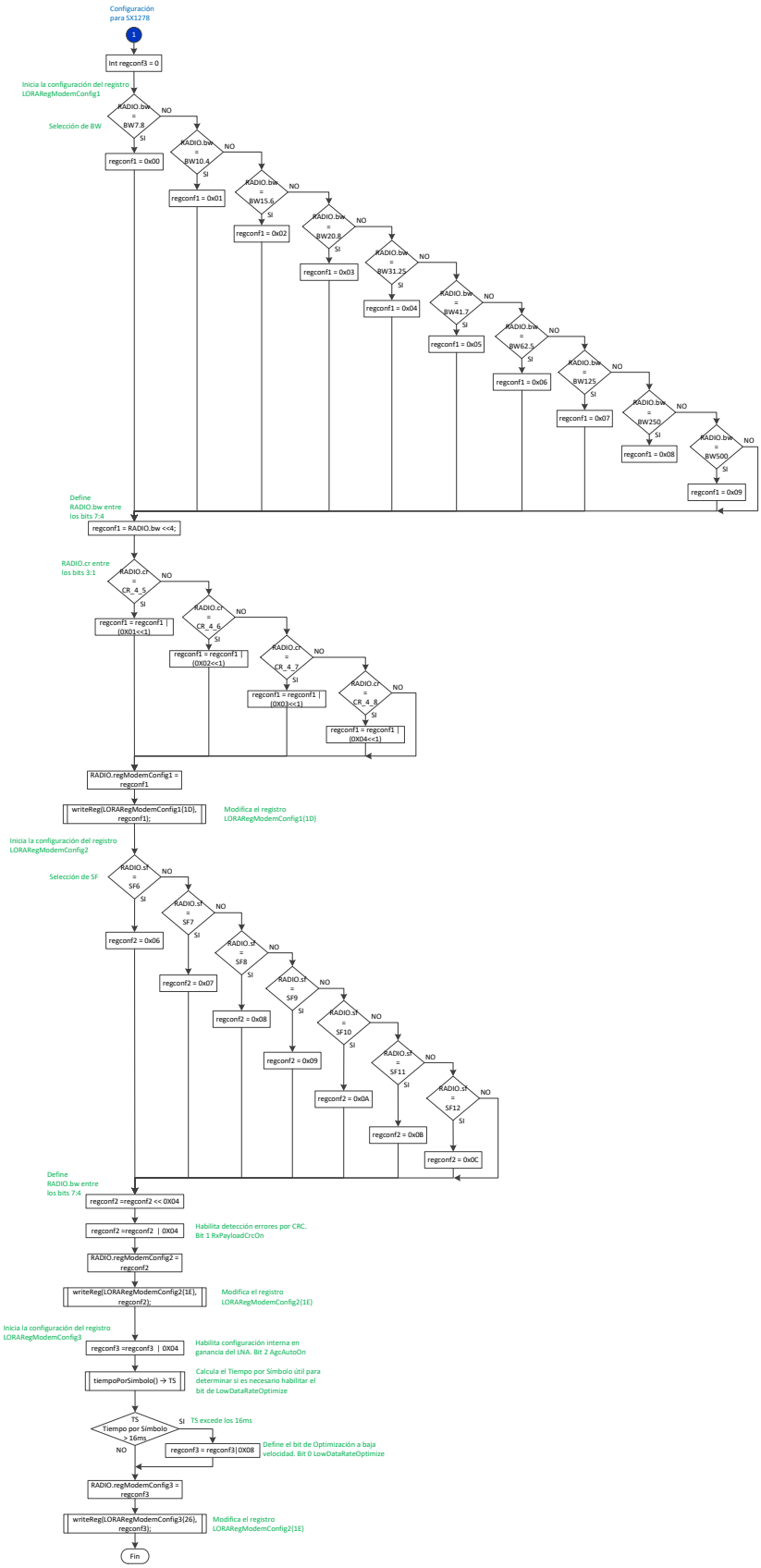


Figura 17. Esquema lógico de configuración de los registros RegModemConfig1_0x1D, RegModemConfig2_0x1E y RegModemConfig3_0x26 para SX1278
Fuente: Semtech SX1272 (2015) & Semtech SX1278 (2015). Desarrollado por autor

Registros de configuración de potencia del radio

Los chip de radio SX1272 y SX1278 permiten configurar la potencia máxima utilizada para trasmitir datos, máxima corriente permitida y ganancia del LNA (HF y LF para el SX1278).

El registro *RegLna_0x0C* permite configurar las características del LNA para HF y LF (LF aplica solo para el SX1278). La *Tabla 11* presenta las variables del registro *RegLna_0x0C* como la configuración de ganancia y los ajustes en corriente para alta frecuencia (HF) y baja frecuencia (LS).

Tabla 11. Descripción del registro RegLna_0x0C del LoRa
Fuente: Semtech SX1272 (2015) & Semtech SX1278 (2015)

Bit	Nombre de la variable	Uso	Defecto	Descripción
7-5	LnaGain	r w	0x01	Ajustes en la ganancia del LNA
				000 → no usado
				001 → G1 = máxima ganancia
				010 → G2
				011 → G3
				100 → G4
				101 → G5
				110 → G6 = mínima ganancia
				111 → no usado
4-3	LnaBoostLf*	r w	0x00	Ajuste en corriente a baja frecuencia del LNA (RFI_LF)
				00 → Corriente por defecto
				Otro → Reservado
2	Reservado	r	0x00	Reservado
1-0	LnaBoostHf	r w	0x00	Ajuste en corriente a alta frecuencia del LNA (RFI_HF)
				00 → Corriente por defecto
				11 → Boost on, 150% de corriente en el LNA

Nota r: lectura. w: escritura. *: Aplica para SX1278

El registro *RegPaConfig_0x09* permite configurar las características del amplificador de potencia (PA). La *Tabla 12* presenta las principales variables del registro *RegPaConfig_0x09* para el chip radio SX1278 en las que se recomienda habilitar el *PA_BOOST* para contar con una potencia máxima de salida de hasta +20dBm, un valor de 0x0F para *MaxPower* (Pmax≈+20dBm) y 0x0F para *OutputPower*.

Tabla 12. Descripción del registro RegPaConfig_0x09 del LoRa
Fuente: Semtech SX1278 (2015)

Bit	Nombre de la variable	Uso	Defecto	Descripción
7	PaSelect	r w	0x00	Selección de la arquitectura del Amplificador de Potencia PA (pin de salida del PA)
				0 → RFO pin. Salida de potencia (Output power) limitada hasta +14 dBm.
				1 → PA_BOOST pin. Salida de potencia (Output power) limitada hasta +20 dBm.
6-4	MaxPower	r w	0x04	Selección de potencia máxima de salida de potencia: Pmax=10.8+0.6*MaxPower [dBm] (0≤MaxPower≤15)
3-0	OutputPower	r w	0x0F	Pout=Pmax-(15-OutputPower) SI PaSelect ← 0 (RFO pin)
				Pout=17-(15-OutputPower) SI PaSelect ← 1 (PA_BOOST pin)

Nota r: lectura. w: escritura.

La *Tabla 13* presenta las principales variables del registro *RegPaConfig_0x09* para el chip radio SX1272 en las que se recomienda habilitar el PA_BOOST para contar con una potencia máxima de salida de hasta +20dBm y un valor de 0x0F para *OutputPower*.

Tabla 13. Descripción del registro RegPaConfig_0x09 del LoRa
Fuente: Semtech SX1272 (2015)

Bit	Nombre de la variable	Uso	Defecto	Descripción
7	PaSelect	r w	0x00	Selección de la arquitectura del Amplificador de Potencia PA (pin de salida del PA)
				0 → RFO pin. Salida de potencia (Output power) limitada hasta +13 dBm.
				1 → PA_BOOST pin. Salida de potencia (Output power) limitada hasta +20 dBm.
6-4	Sin uso	r	0x04	Sin uso
3-0	OutputPower	r w	0x0F	Pout = -1 + OutputPower(3:0) SI PaSelect ← 0 (RFO pin)
				Pout = 2 + OutputPower(3:0) SI PaSelect ← 1 (PA_BOOST pin)

Nota r: lectura. w: escritura.

Los radio LoRa cuentan con un circuito de protección para evitar el sobre-exceso de corriente al transmitir (OCP) en el Amplificador de Potencia (PA). Esta característica de los LoRa permite aumentar el ciclo de vida útil de las baterías, utilizar bajos consumos energéticos y evitar picos excesivos al trasmitir datos.

El registro *RegOcp_0x0B* permite regular los consumos de corrientes pico al trasmitir hasta de 240mA cuando el transceiver está en modo trasmisión. Para valores significativos de corrientes (mayores a 125mA), se recomienda verificar la corriente máxima de suministro soportada por la fuente o batería de alimentación. Algunos sistemas embebidos como Arduino cuentan con reguladores con corriente máxima de suministro hasta de 120mA, por lo que al realizar una inadecuada configuración en el LoRa, podría destruirse el regulador y dañar la placa del sistema embebido.

La *Tabla 14* presenta los valores para el registro *RegOcp_0x0B* de los radio SX1272 y SX1278. El registro *RegPaDac** establece la máxima potencia en el *PA_BOOST* de hasta +20dBm. La descripción es este registro representa en la *Tabla 15*.

Tabla 14. Descripción del registro RegOcp_0x0B del LoRa
Fuente: Semtech SX1272 (2015) & Semtech SX1278 (2015)

Bit	Nombre de la variable	Uso	Defecto	Descripción
7-6	Sin uso	r	0x00	Sin uso
5	OcpOn	r w	0x01	Habilita el circuito de protección por exceso de corriente al trasmitir (OCP) el amplificador de potencia (PA):
				0 → OCP no lo habilita
				1 → OCP lo habilita
4-0	OcpTrim	r w	0x0B	Corriente máxima soportada por el OCP:
				Imax = 45+5*OcpTrim [mA] SI OcpTrim ≤ 15 (Imax hasta 120mA)
				Imax = -30+10*OcpTrim [mA] SI 15 < OcpTrim≤27 (130 <Imax< 240 mA)
				Imax = 240mA SI OcpTrim >27
				Por defecto Imax = 100mA

Nota r: lectura. w: escritura.

Tabla 15. Descripción del registro RegPaDac del LoRa*
Fuente: Semtech SX1272 (2015) & Semtech SX1278 (2015)

Bit	Nombre de la variable	Uso	Defecto	Descripción
7-3	Reservado	r w	0x10	Reservado
2-0	PaDac	r w	0x04	Habilita la opción de +20dBm SI PA_BOOST pin-> 1
				0x04 → Valor por defecto
				0x07 → +20dBm en PA_BOOST cuando OutputPower=1111

Nota r: lectura. w: escritura. *La dirección para el SX1272 es 0x5A y para SX1278 es 0x4D

Se plantea en la *Figura 18* el esquema lógico de configuración de los registros relacionados con la potencia de los transceivers LoRa utilizando las variables RADIO.txpow y RADIO.imax configurables por el usuario.

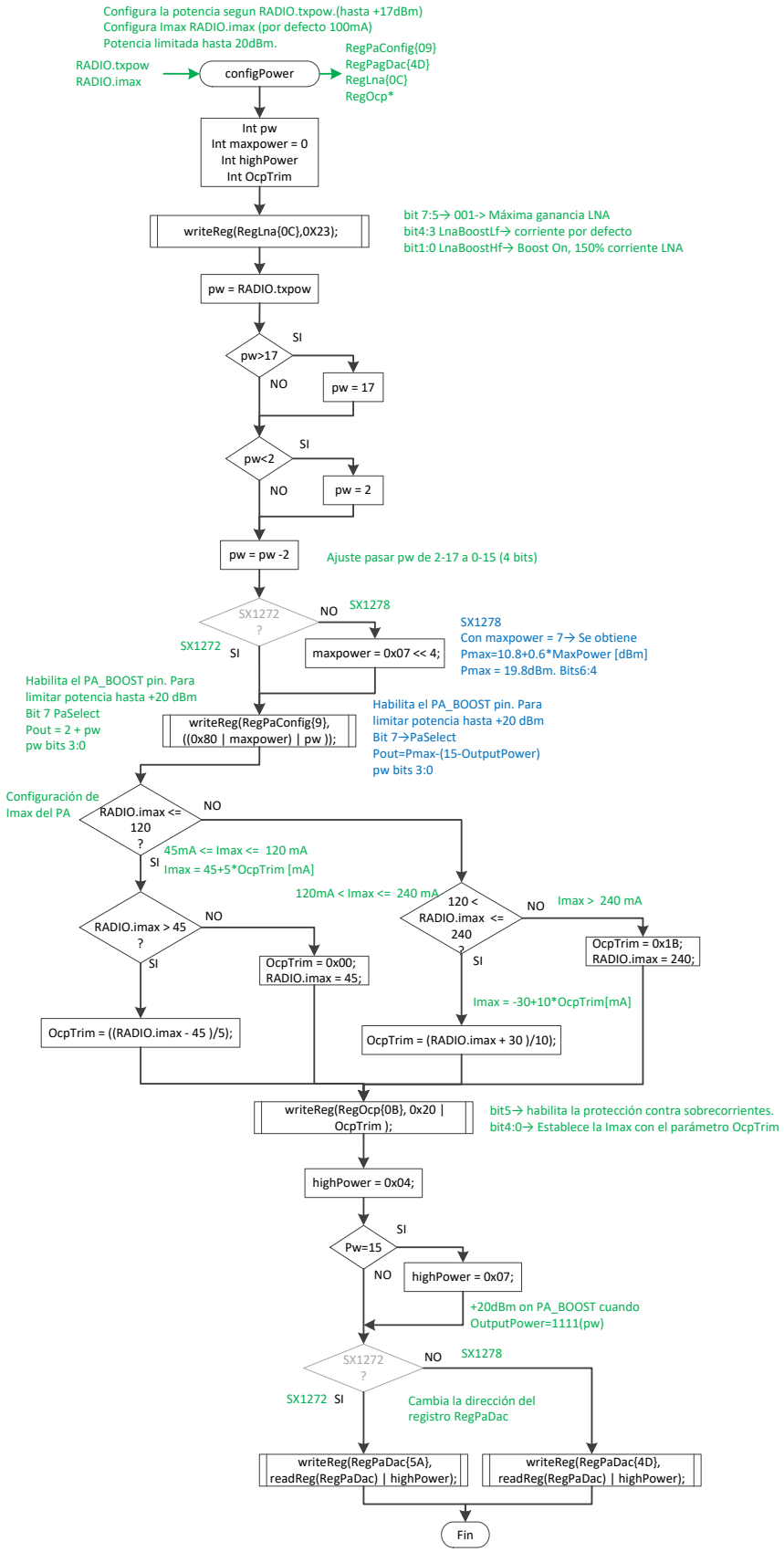


Figura 18. Esquema lógico de configuración de los registros RegLna_0x0C, RegPaConfig_0x09, RegOcp_0x0B y RegPaDac para los dispositivos LoRa
Fuente: Semtech SX1272 (2015) & Semtech SX1278 (2015). Desarrollado por autor

Registros de configuración de frecuencia del radio

Teniendo en cuenta que existen bandas de frecuencias de uso privativo (telefonía, internet, radio, entre otras), existen frecuencias de libre uso conocidas como ISM (Industrial, Scientific and Medical) que son frecuencias intencionalmente libres para uso no comercial en aplicaciones de tipo industrial, científico y médico. Se recomienda al usuario verificar la compatibilidad de la frecuencia de trabajo con la *Tabla 16* (bandas de uso ISM).

Tabla 16. Banda de frecuencia ISM compatibles con los transceivers de radio LoRa
Fuente: Texas Instruments (2005)

Banda ISM (MHz)	Frecuencia mínima (MHz)	Frecuencia máxima (MHz)	Ancho de Banda (MHz)	Usos por continente
433	433.050	434.790	1.84	Europa, África y parte del norte de Asia.
869	868	870	2	Europa, África, Asia y Oceanía.
900	902	928	26	Continente americano

Para fijar la frecuencia de trabajo en el los transceivers LoRa, el fabricante presenta la ecuación (2):

$$f_{RF} = \frac{2^{19} \cdot Frecuencia}{F_{OSC}} \quad (2)$$

Dónde:
f_{RF} : Frecuencia utilizada en los registros SPI *RegFrLsb*, *RegFrMid* y *RegFrMsb*
F_{OSC}=Frecuencia del oscilador del LoRa (32 MHz)
Frecuencia : Frecuencia de trabajo fijada por el usuario.

La *Tabla 17* presenta la descripción de los registros de configuración de frecuencia.

Tabla 17. Descripción del registro de configuración de frecuencia en los radio LoRa
Fuente: Semtech SX1272 (2015) & Semtech SX1278 (2015)

Nombre del registro	Dirección del registro	Uso	Defecto		Descripción
			SX1272	SX1278	
RegFrLsb	0x08	r w	0x00	0x00	Bits 7:0 del valor calculado de f_{RF}
RegFrMid	0x07	r w	0xC0	0x80	Bits 15:8 del valor calculado de f_{RF}
RegFrMsb	0x06	r w	0xE4	0x6C	Bits 15:23 del valor calculado de f_{RF}

Nota r: lectura. w: escritura.

La *Figura 19* presenta el esquema lógico propuesto para fijar la frecuencia de trabajo en el chip LoRa. La variable RADIO.Freq será definida por el usuario en Hz.

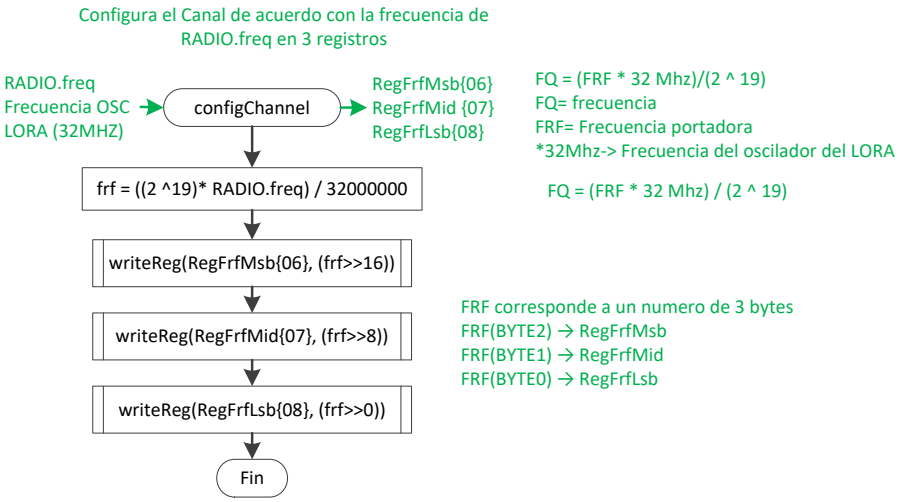


Figura 19. Esquema lógico de configuración de frecuencia para los radio LoRa.
Fuente: Semtech SX1272 (2015) & Semtech SX1278 (2015). Desarrollado por autor

Configuración por software del radio lora para la trasmisión datos

Para trasmitir los datos, es necesario utilizar los registros relacionados con la FIFO (memoria RAM de almacenamiento de los datos que serán transmitidos o que gestiona la recepción de datos). Los transceivers LoRa cuentan con una capacidad de 256 bytes en la FIFO.

Se recomienda el uso de interrupciones las cuales resultan ser útiles para la transmisión de datos por cuanto una vez se escriben los datos en la FIFO, de acuerdo a la configuración (BW, SF y CR), los transceivers LoRa tardan cierto tiempo en realizar esta tarea y la interrupción TXDONE permite preparar al LoRa ya sea para una nueva trasmisión, colocarse en modo SLEEP o prepararse para recibir datos.

Cada una de las interrupciones utilizadas por los dispositivos LoRa se relaciona con un bit o pin físico de control (los bits de control son los DIO y se presentan en la *Figura 4*, *Figura 5* y *Figura 6*).

La *Tabla 18* presenta la descripción del registro *RegIrqFlagsMask_0x11* que permite habilitar el uso de las interrupciones. La *tabla 19* presenta el registro de *RegIrqFlags_0x12* que permite determinar si se ha presentado una interrupción por software. Sin embargo, los pines DIO se

encuentran relacionados con estos flags y pueden ser utilizados para detectar interrupciones por cambio de flanco ascendente (Low-High), lo cual resulta ser muy útil en cuanto a la optimización de tareas con el uso de microcontroladores. La *Tabla 20* y *Tabla 21* presenta la descripción de los registros *RegDioMapping1_0x40* y *RegDioMapping2_0x41* para la configuración de pines DIO con el uso de interrupciones por hardware.

Tabla 18. Descripción del registro RegIrqFlagsMask_0x11 para habilitación de interrupciones
Fuente: Semtech SX1272 (2015) & Semtech SX1278 (2015)

Bit	Nombre de la variable	Uso	Defecto	Descripción
7	RxTimeoutMask	r w	0x00	Mascara de interrupción para Timeout. Con “0” se habilita la interrupción.
6	RxDoneMask	r w	0x00	Mascara de interrupción para RXDONE o recepción completa de paquetes. Con “0” se habilita la interrupción.
5	PayloadCrcErrorMask	r w	0x00	Mascara de interrupción para detección de error por CRC en recepción de datos. Con “0” se habilita la interrupción.
4	ValidHeaderMask	r w	0x00	Mascara de interrupción para detección Header o encabezado en recepción de datos. Con “0” se habilita la interrupción
3	TxDoneMask	r w	0x00	Mascara de interrupción para TXDONE o trasmisión completa de datos en la FIFO. Con “0” se habilita la interrupción.
2	CadDoneMask	r w	0x00	Mascara de interrupción para CADDONE o detección de preámbulo. Con “0” se habilita la interrupción.
1	FhssChangeChannelMask	r w	0x00	Mascara de interrupción para FhssChangeChannel o cambio de canal. Con “0” se habilita la interrupción.
0	CadDetectedMask	r w	0x00	Mascara de interrupción para detección del CAD. Con “0” se habilita la interrupción.

Nota r: lectura. w: escritura.

Tabla 19. Descripción del registro RegIrqFlags_0x12 para la detección de interrupciones por software
Fuente: Semtech SX1272 (2015) & Semtech SX1278 (2015)

Bit	Nombre de la variable	Uso	Defecto	Descripción
7	RxTimeout	r c	0x00	Flag de interrupción por Timeout. Escribiendo “1” se limpia a IRQ.
6	RxDone	r c	0x00	Flag de interrupción por RXDONE o recepción completa de paquetes. Escribiendo “1” se limpia a IRQ.
5	PayloadCrcError	r c	0x00	Flag de interrupción para detección de error por CRC en recepción de datos. Escribiendo “1” se limpia a IRQ.
4	ValidHeader	r c	0x00	Flag de interrupción para detección Header o encabezado en recepción de datos. Escribiendo “1” se limpia a IRQ.
3	TxDone	r c	0x00	Flag de interrupción para TXDONE o trasmisión completa de datos en la FIFO. Escribiendo “1” se limpia a IRQ.
2	CadDone	r c	0x00	Flag de interrupción para CADDONE o detección de preámbulo. Escribiendo “1” se limpia a IRQ.
1	FhssChangeChannel	r c	0x00	Flag de interrupción para FhssChangeChannel o cambio de canal. Escribiendo “1” se limpia a IRQ.
0	CadDetected	r c	0x00	Flag de interrupción para detección del CAD. Escribiendo “1” se limpia a IRQ.

Nota r: lectura. c: limpiar.

Tabla 20. Configuración de pines de control con el registro RegDioMapping1_0x40 para el uso con interrupciones en los pines DIO0-3
Fuente: Semtech SX1272 (2015) & Semtech SX1278 (2015)

Bit	Nombre de la variable	Uso	Defecto	Descripción
7-6	Dio0Mapping	rw	0x00	00 → RxDone. Se activa el pin cuando se ha recibido datos en la FIFO.
				01 → TxDone. Se activa el pin cuando se ha transmitido datos desde la FIFO.
				10 → CadDone
				11 →Sin uso del pin
5-4	Dio1Mapping	rw	0x00	00 → RxTimeout
				01 → FhssChangeChannel
				10 → CadDetected
				11 →Sin uso del pin
3-2	Dio2Mapping	rw	0x00	00 → FhssChangeChannel
				01 → FhssChangeChannel
				10 → FhssChangeChannel
				11 →Sin uso del pin
1-0	Dio3Mapping	rw	0x00	00 → CadDone
				01 → ValidHeader
				10 → PayloadCrcError. Se activa el pin cuando se detecta un error por verificación por redundancia cíclica (CRC)
				11 →Sin uso del pin

Nota r: lectura. w: escritura.

Tabla 21. Configuración de pines de control con el registro RegDioMapping2_0x41 para el uso con interrupciones en los pines DIO4-5
Fuente: Semtech SX1272 (2015) & Semtech SX1278 (2015)

Bit	Nombre de la variable	Uso	Defecto	Descripción
7-6	Dio4Mapping	rw	0x00	00 → CadDetected
				01 → PllLock
				10 → PllLock
				11 →Sin uso del pin
5-4	Dio5Mapping	rw	0x00	00 → ModeReady
				01 → ClkOut
				10 → ClkOut
				11 →Sin uso del pin

Para escribir datos sobre la FIFO de los transceivers LoRa, se utiliza el registro *RegFifo_0x00* para indicar la posición de escritura de los datos. Para tener acceso a la FIFO, el chip radio debe estar en *modo STAND-BY*. La *Figura 20* presenta el esquema lógico propuesto para almacenar datos sobre la FIFO antes de preparar el chip para transmitirlos. Para utilizar la propuesta de configuración se requiere que el usuario defina sobre RADIO.frameTX los valores que desea transmitir y en RADIO.dataLenTX la cantidad o longitud de datos a transmitir.

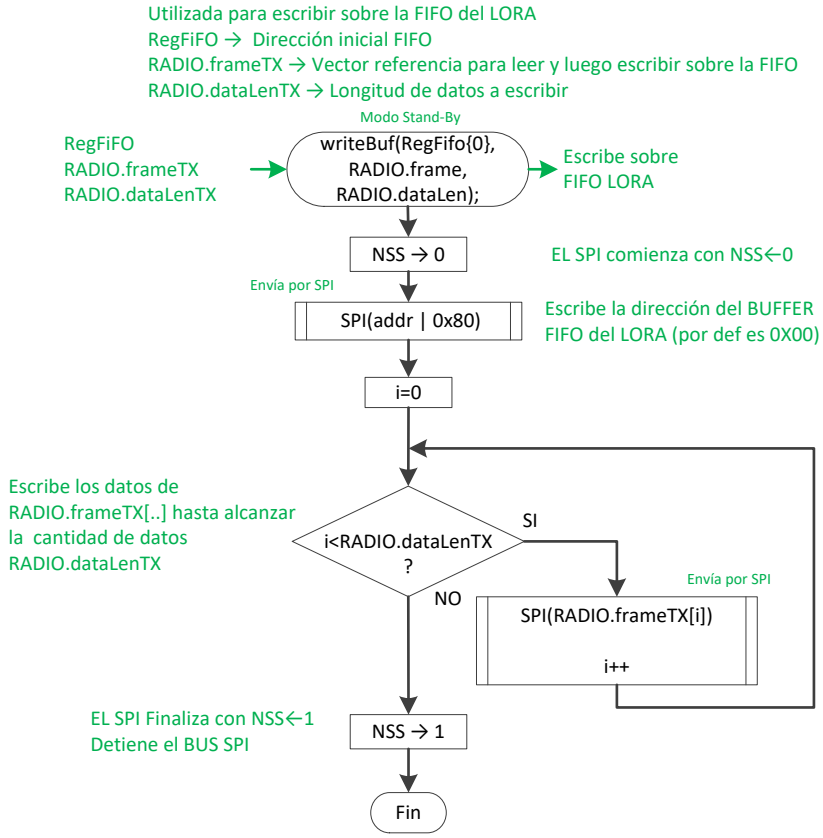


Figura 20. Esquema lógico de almacenamiento de datos sobre la FIFO del LoRa
Fuente: Semtech SX1272 (2015) & Semtech SX1278 (2015). Desarrollado por autor

La *Figura 21* presenta el esquema lógico propuesto para trasmitir datos previamente almacenados en la FIFO. Para utilizar el esquema, se requiere que el usuario defina sobre RADIO.frameTX los valores que desea transmitir y en RADIO.dataLenTX la cantidad o longitud de datos. Además, antes de finalizar el flujograma se recomienda que el usuario configure una interrupción por cambio de flanco ascendente en el pin conectado al terminal del control DIO0.

Una vez se establece el modo *OPMODE_TX* en los dispositivos LoRa, comienza formalmente la trasmisión de los datos escritos sobre la FIFO. Cuando finaliza la trasmisión, el flag *TxDone* del registro *RegIrqFlags_0x12* se habilitara al igual que el pin *DIO0*. Si no se habilita ninguno de los dos (flag *TxDone* o pin *DIO0*), el usuario deberá reenviar los datos. El esquema lógico para determinar si se deben retransmitir los datos se presenta en la *Figura 22*.

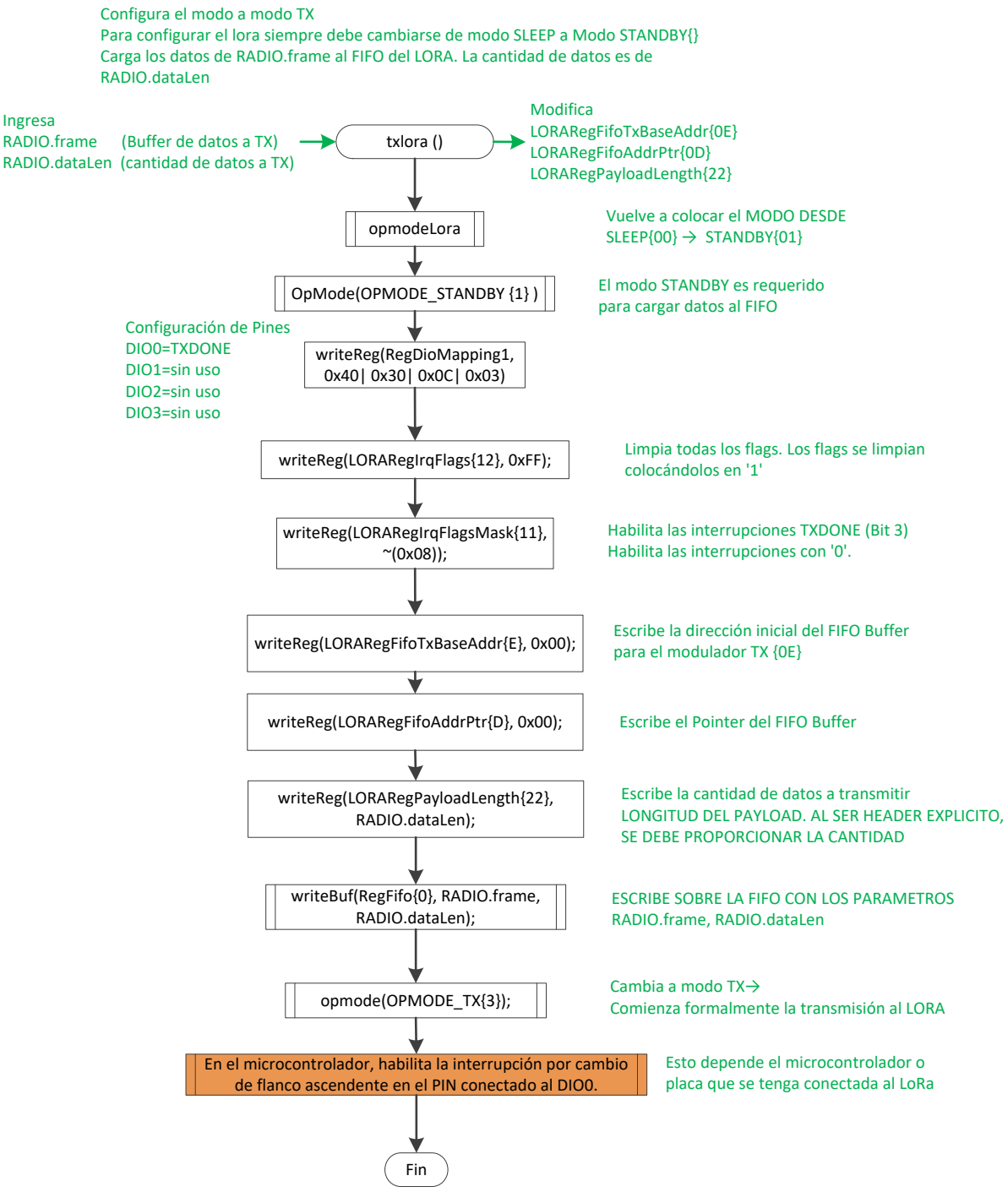


Figura 21. Esquema lógico de configuración para transmisión de datos en el LoRa
Fuente: Semtech SX1272 (2015) & Semtech SX1278 (2015). Desarrollado por autor

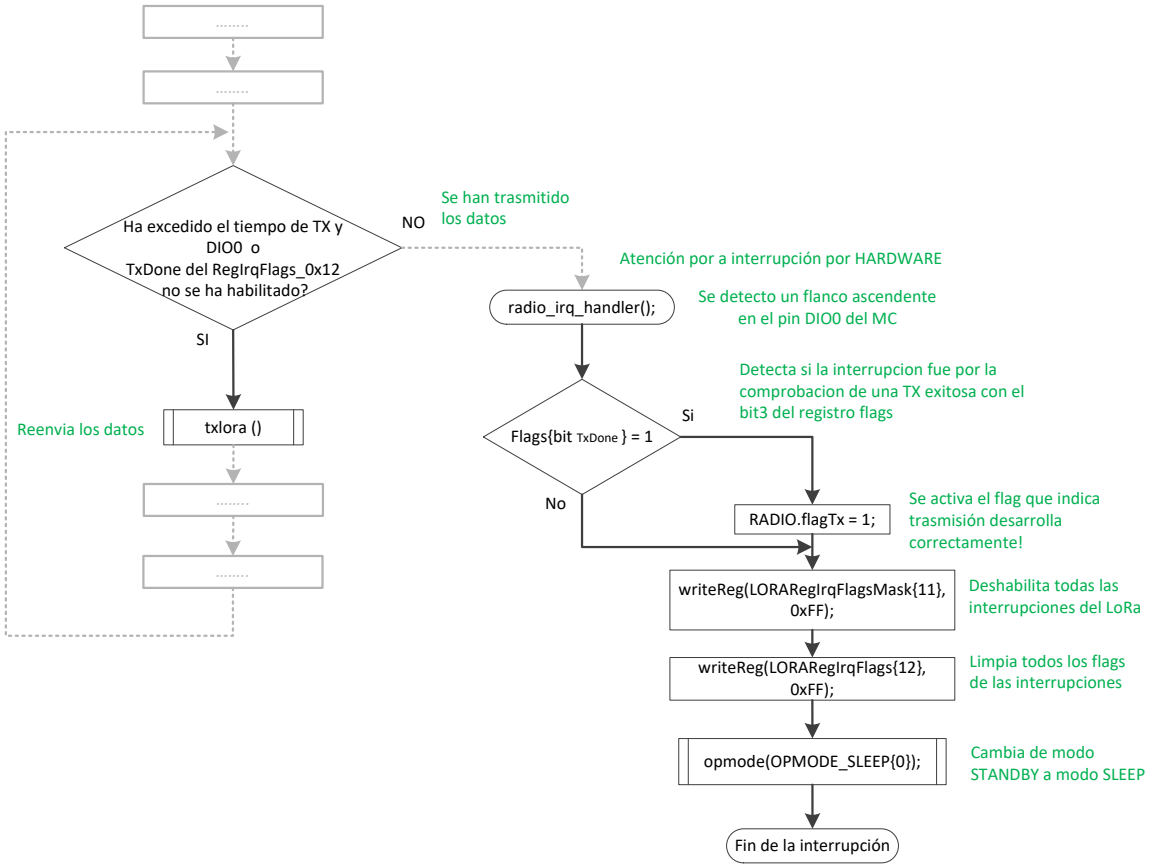


Figura 22. Esquema lógico propuesto para gestión por interrupción cuando se efectua una trasmisión de datos en el LoRa
Fuente: Semtech SX1272 (2015) & Semtech SX1278 (2015). Desarrollado por autor

Configuración por software del radio lora para la recepción de datos

El modem LoRa es half-duplex, es decir que no es posible trasmitir o recibir datos en el mismo instante; esto significa que se debe configurar el modem para trasmitir (txlora) o recibir datos (rxlora). El LoRa tiene la capacidad de recibir hasta 256 bytes de manera simultánea (capacidad de la FIFO).

Para recibir datos, se debe configurar el modem junto con el conjunto de interrupciones de configuración para dicha tarea. La interrupción para detectar la recepción y gestión necesaria de datos sobre la FIFO se habilita utilizando el bit 6 o *RxDoneMask* del registro *RegIrqFlagsMask_0x11* (ver *Tabla 18*). Esta interrupción está relacionada con el pin de control DIO0 (ver *Tabla 20*).

Una ventaja adicional del chip LoRa es la incorporación de la verificación por redundancia cíclica (CRC), es decir que en caso de detectar un error al recibir tramas de datos sobre la FIFO, el LoRa tiene la posibilidad de emitir o presentar un error. La detección de errores en la recepción de datos CRC se habilita utilizando el bit 5- *PayloadCrcErrorMask* del registro RegIrqFlagsMask_0x11 (ver *Tabla 18*). Esta interrupción está relacionada con el pin de control DIO3 (ver *Tabla 20*) y solo será activada cuando se reciban datos errados; por lo que se activara el bit 5 – *PayloadCrcError* del registro *RegIrqFlags_0x12* "o" el pin DIO3 del microcontrolador (el pin DIO3 se encuentra físicamente para las placas SX1272RF1 y DRF1278F, para la versión LoRa1278 de NiceRF se utiliza únicamente el bit 5 del registro *PayloadCrcError*).

La *Figura 23* presenta el esquema propuesto de configuración de registros en el LoRa para la recepción de datos utilizando la interrupción *RXDONE* y la detección de errores en la recepción de datos *PayloadCrcError*.

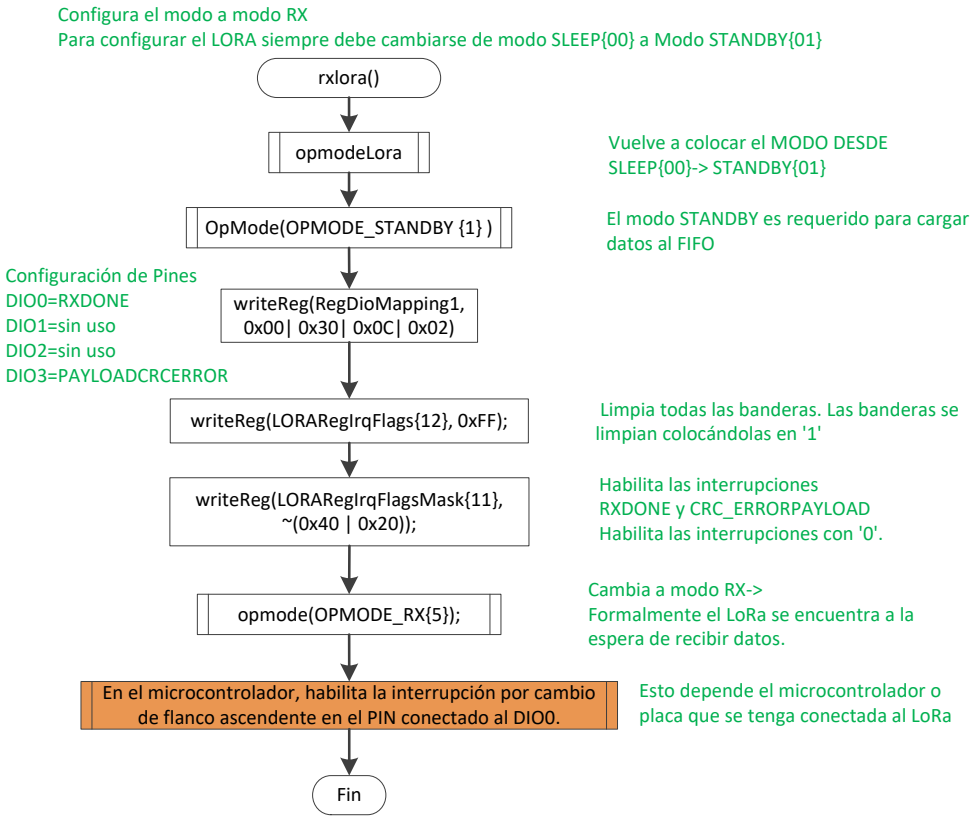


Figura 23. Esquema lógico de configuración en el LoRa para recepción de datos
Fuente: Semtech SX1272 (2015) & Semtech SX1278 (2015). Desarrollado por autor

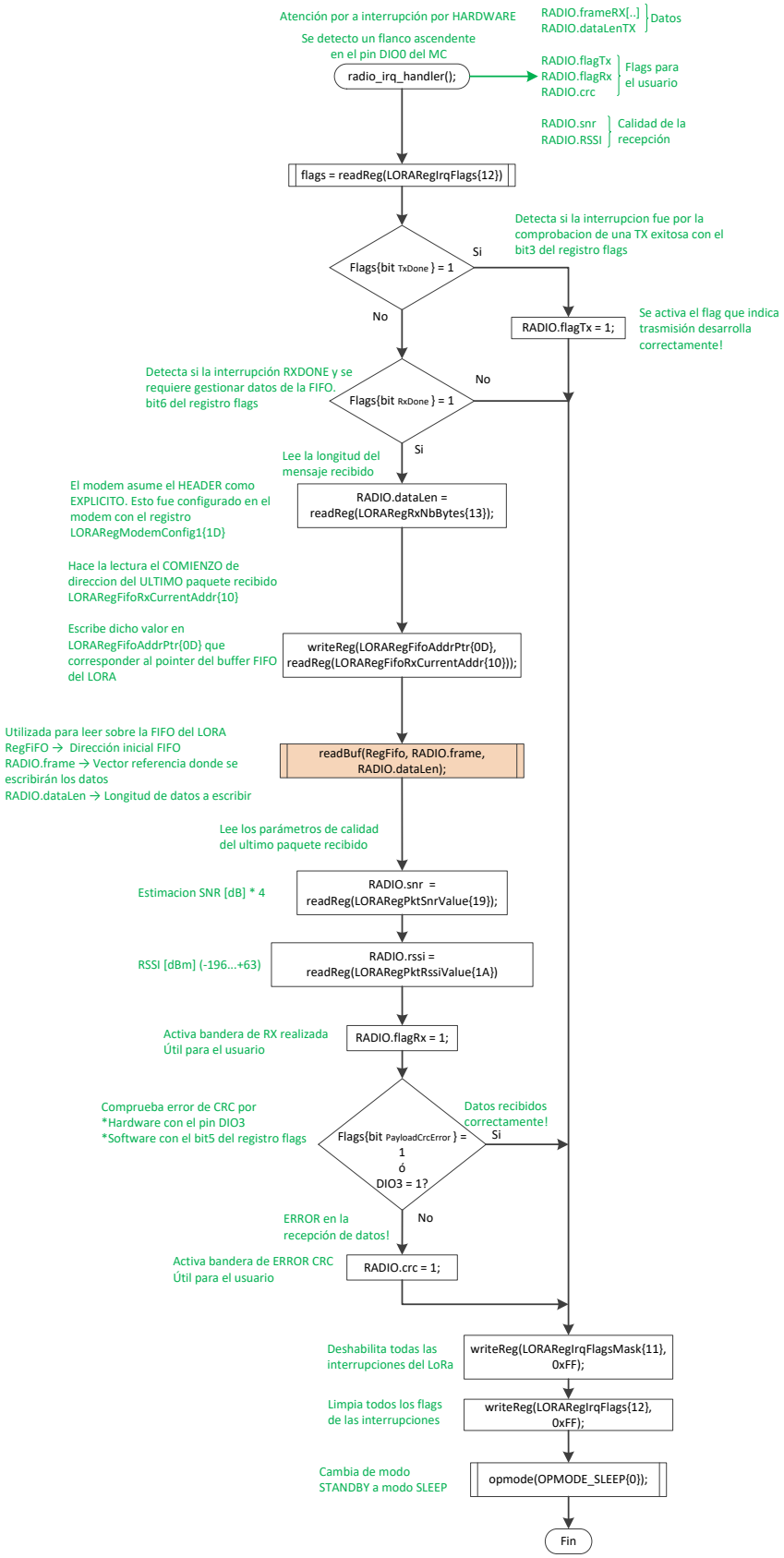


Figura 24. Esquema lógico de atención a la interrupción por recepción o trasmisión de datos en el LoRa

Fuente: Semtech SX1272 (2015) & Semtech SX1278 (2015). Desarrollado por autor

Es posible detectar la recepción de los datos comprobando el registro *RegIrqFlags_0x12* con el flag *RxDone* o por la detección de una interrupción de flanco ascendente por parte del pin DIO0 en el microcontrolador. La *Figura 24* presenta el esquema lógico propuesto para la detección de la interrupción generada por el pin DIO0 en el microcontrolador cuando se ha efectuado una trasmisión correctamente o cuando se disponen de datos en la FIFO que deben ser gestionados por el microcontrolador.

Los parámetros generados por la interrupción corresponden a las variables:

- **RADIO.frameRX[..]:** vector de datos recibidos por el LoRa
- **RADIO.dataLenTX:** cantidad de datos recibidos
- **RADIO.flagTx:** Flag que le indica al usuario si se desarrolló correctamente una trasmisión.
- **RADIO.flagRx:** Flag que le indica al usuario si se desarrolló la gestión de los datos recibidos.
- **RADIO.crc:** Flag que le indica al usuario si los datos recibidos están corruptos o contienen errores que debe gestión realizando nuevamente una petición.
- **RADIO.snr:** Calidad de la señal a ruido de los datos recibidos.
- **RADIO.rssi:** Indicador de fuerza de los datos recibidos.

Para leer datos de la FIFO del LoRa, se utiliza el registro *RegFifo_0x00* para indicar la posición de lectura de los datos. Para tener acceso a la FIFO, el chip radio debe estar en *modo STANDBY*. La *Figura 25* presenta el esquema lógico propuesto para extraer los datos de la FIFO luego de recibir datos. Para utilizar el esquema, se requiere que el usuario cree el vector *RADIO.frameRX* para el almacenamiento de los valores recibidos en la FIFO y cree la variable *RADIO.dataLenRX* que corresponderá a la cantidad o longitud de datos recibidos.

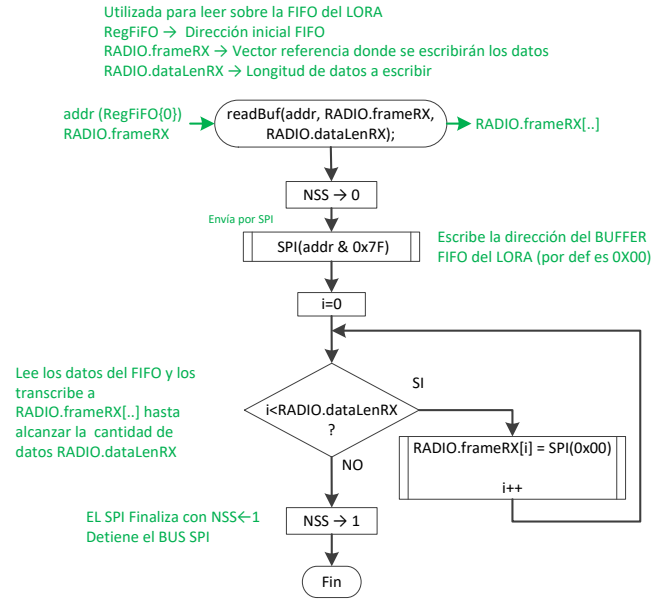


Figura 25. Esquema lógico de almacenamiento de datos sobre la FIFO del LoRa.
Fuente: Semtech SX1272 (2015) & Semtech SX1278 (2015). Desarrollado por autor

Ejemplo de aplicación: red punto a punto (aplicación ping-pong)

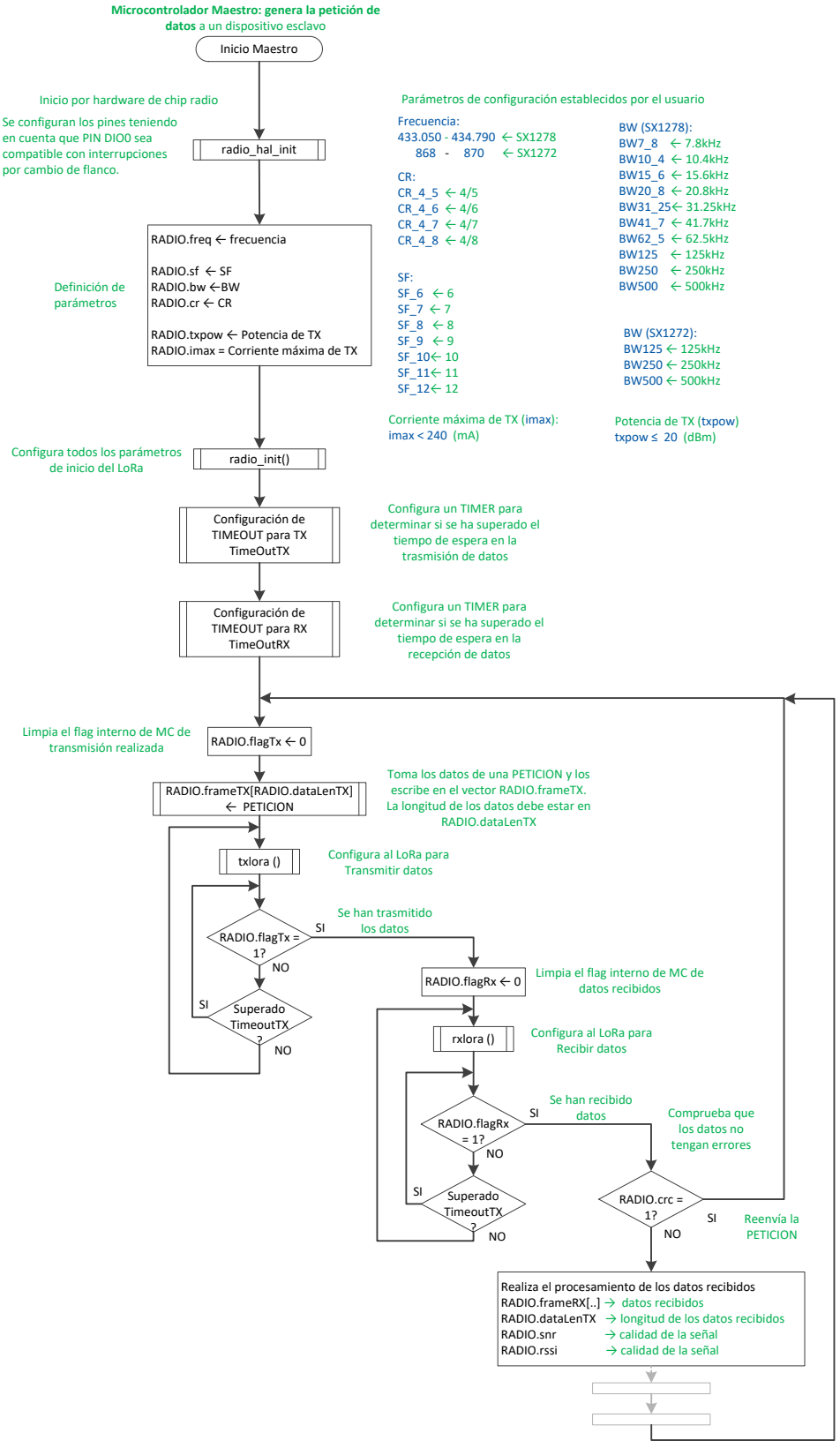


Figura 26. Esquema lógico para dispositivo maestro utilizando radio LoRa.
Fuente: Desarrollado por autor.

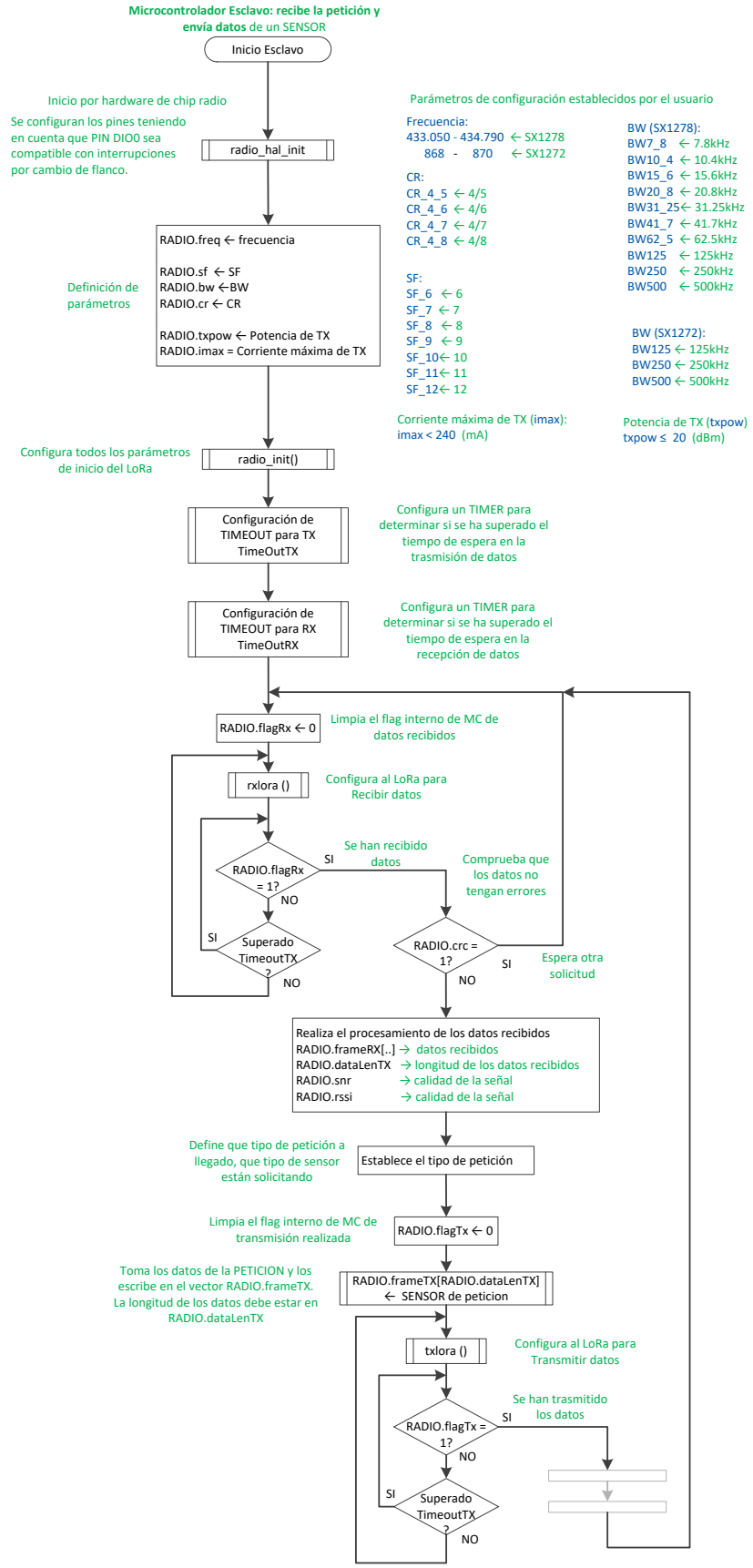


Figura 27. Esquema lógico para dispositivo esclavo utilizando radio LoRa.
Fuente: Desarrollado por autor.

Como ejemplo se plantea el uso de dos sistemas embebidos, uno utilizado como maestro y el otro utilizado como esclavo.

El dispositivo maestro se encarga de realizar la petición de datos a un dispositivo esclavo ubicado en un sitio remoto, el cual obtiene datos de sensores. El esclavo remite la información solicitada y continuamente se encuentra esperando peticiones del dispositivo esclavo. La *Figura 26* presenta el flujograma propuesto para la configuración del dispositivo maestro y la *Figura 27* presenta el esquema propuesto de configuración para el dispositivo esclavo (sensor). Finalmente en la *Figura 28* se presenta un diagrama pictórico de la aplicación propuesta: maestro-esclavo (sensor) con la utilización de dispositivo LoRa.

Figura 28. Principio de funcionamiento "ping-pong" para la realización de pruebas de alcance en dispositivos LoRa
Fuente: Desarrollado por autor

Ejemplo de aplicación: recepción de datos de varios dispositivos remotos (red punto - multipunto)

El ejemplo plantea un esquema de dispositivos conectados por una red punto-multipunto tipo estrella con un nodo central (circuito maestro) y 3 dispositivos esclavos. Como circuitos esclavos, se utilizan diversas versiones de sistemas embebidos.

La *Figura 29* presenta el esquema propuesto donde se aprecia la comunicación simultánea de 3 dispositivos hacia un nodo central. Se plantea el uso de esta aplicación con el uso del driver desarrollado para aplicaciones complejas con diversos sensores ubicados en sitios remotos. Cabe resaltar que los dispositivos LoRa deben ser de la misma referencia, estar configurados a la misma frecuencia y con los parámetros SF, BW y CR idénticos.

El dispositivo maestro se debe configurar como dispositivo “pasivo”, es decir continuamente recibir la información de los dispositivos esclavos. Una vez recibe la información de un dispositivo esclavo, remite la confirmación de recepción correcta de datos.

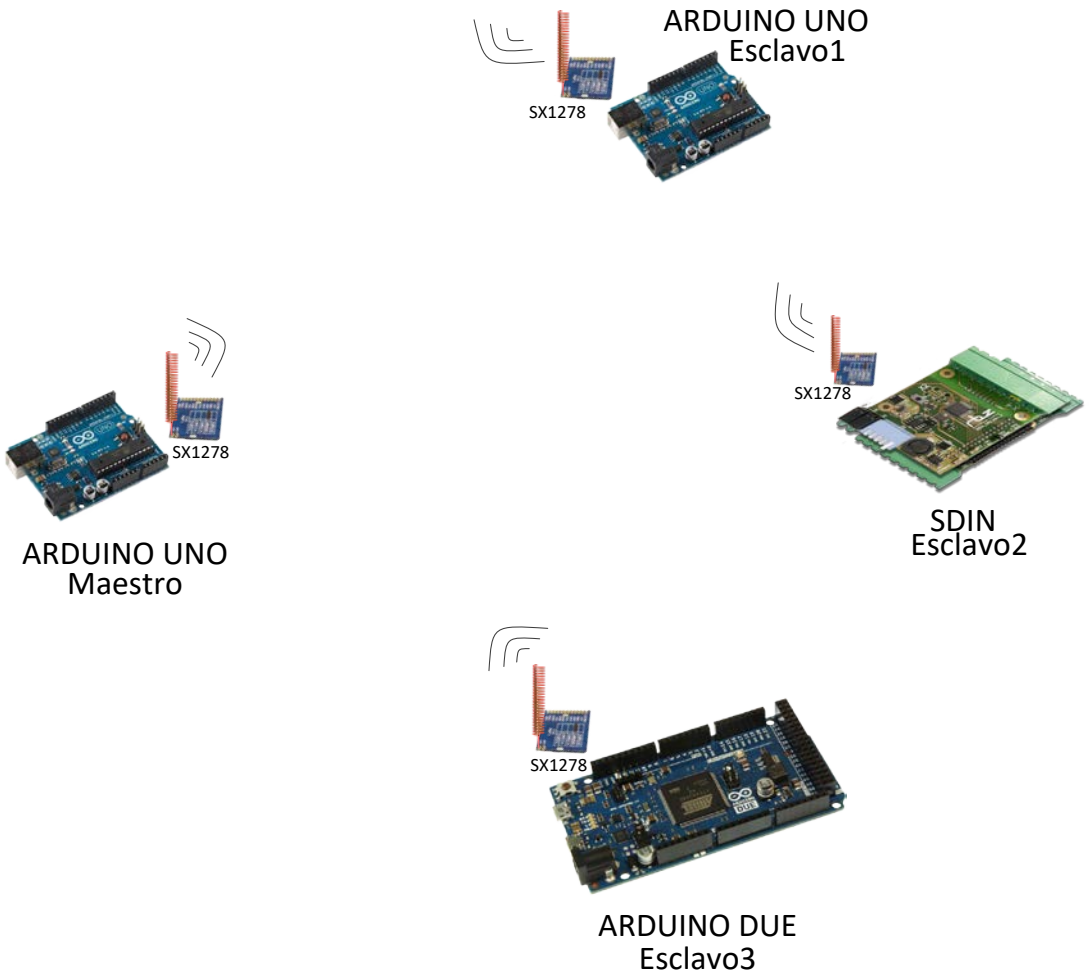


Figura 29. Tipología estrella para comunicación con diversos dispositivos
Fuente: Esquema propuesto por el autor

El modem LoRa es *half dúplex*, es decir que los LoRa no puede trasmitir o recibir datos en el mismo instante; esto significa que los transceiver LoRa deben ser configurados para trasmitir o recibir datos. Cuando se incorpora más de dos dispositivos en un protocolo de comunicación *half dúplex*, se deben establecer estrategias que mitiguen los efectos adversos por colisión indeseada de datos cuando se reciben datos.

Si los dispositivos no cuentan con estrategias para evitar la trasmisión de datos sobre el mismo canal de comunicación por parte de los dispositivos esclavos en un mismo instante, los datos llegaran errados al dispositivo maestro y la comunicación de datos será errada.

Este efecto se observa en las líneas de tiempo presentadas en la *Figura 30* donde los dispositivos esclavos tramiten información en un mismo instante y el dispositivo maestro no cuenta puede responder con la confirmación de comunicación efectuada, lo que ocasiona un

consumo energético innecesario de energía en la transmisión de información por parte de los esclavos.

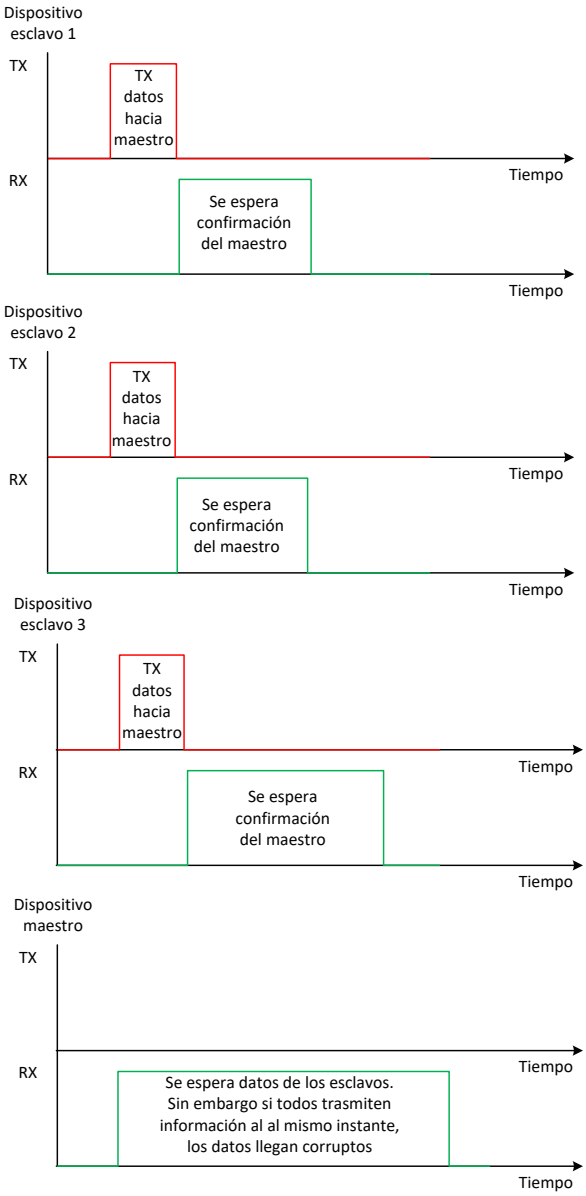


Figura 30. Líneas de tiempos para dispositivos esclavos y dispositivo maestro sin estrategias adecuadas para la trasmisión de información
Fuente: Esquema propuesto por el autor

La solución para evitar la colisión de datos, corresponde a establecer tiempos de muestreos y tiempos aleatorios de "des-sincronización" cuando se trasmiten datos; es decir evitar con los tiempos adiciones de trasmisión de datos la colisión de datos.
Una vez se establece el tiempo en el que no se presentan colisiones, se almacena dicho valor para ser utilizado consecutivamente en las próximas trasmisiones de datos y con lo anterior el sistema *half dúplex* se comporta sincronizado como se presenta en la *Figura 31*.

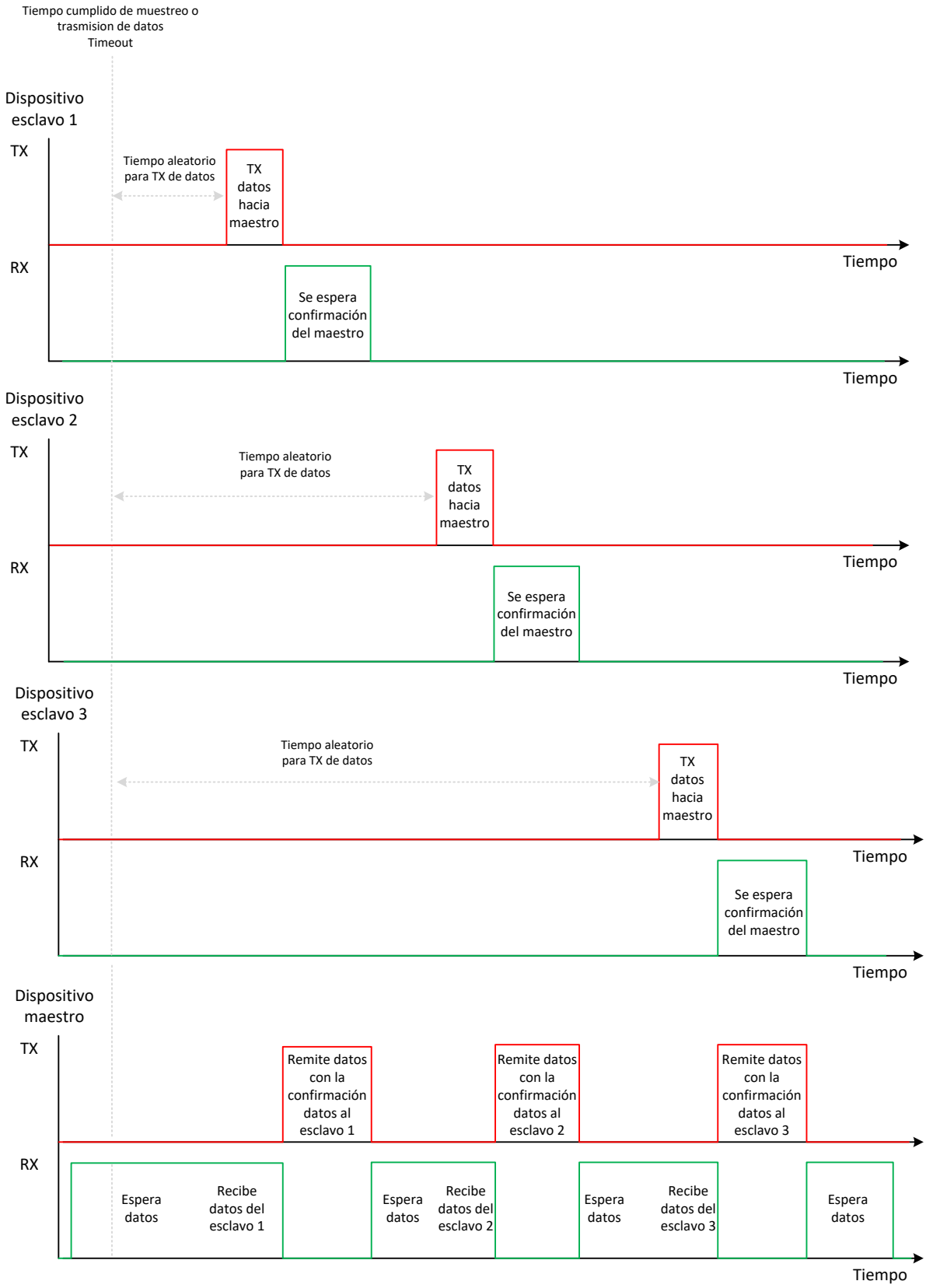


Figura 31. Líneas de tiempos para dispositivos esclavos y dispositivo maestro con tiempos aleatorios para la trasmisión adecuada de datos
Fuente: Esquema propuesto por el autor

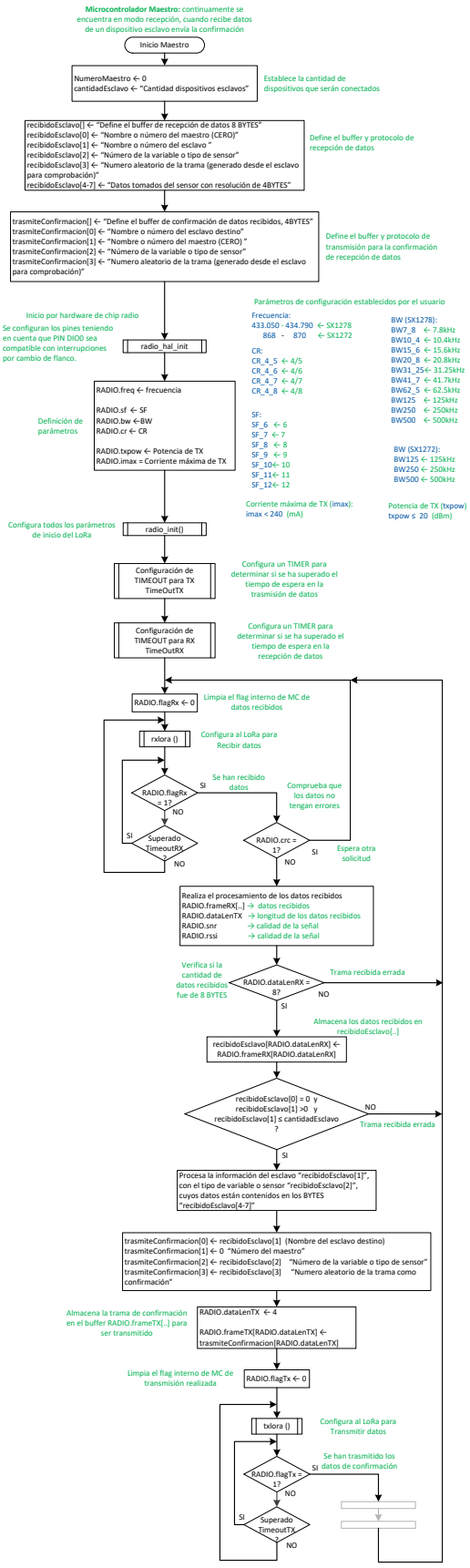


Figura 32. Esquema lógico para dispositivo maestro en aplicación punto-multipunto
Fuente: Desarrollado por autor.

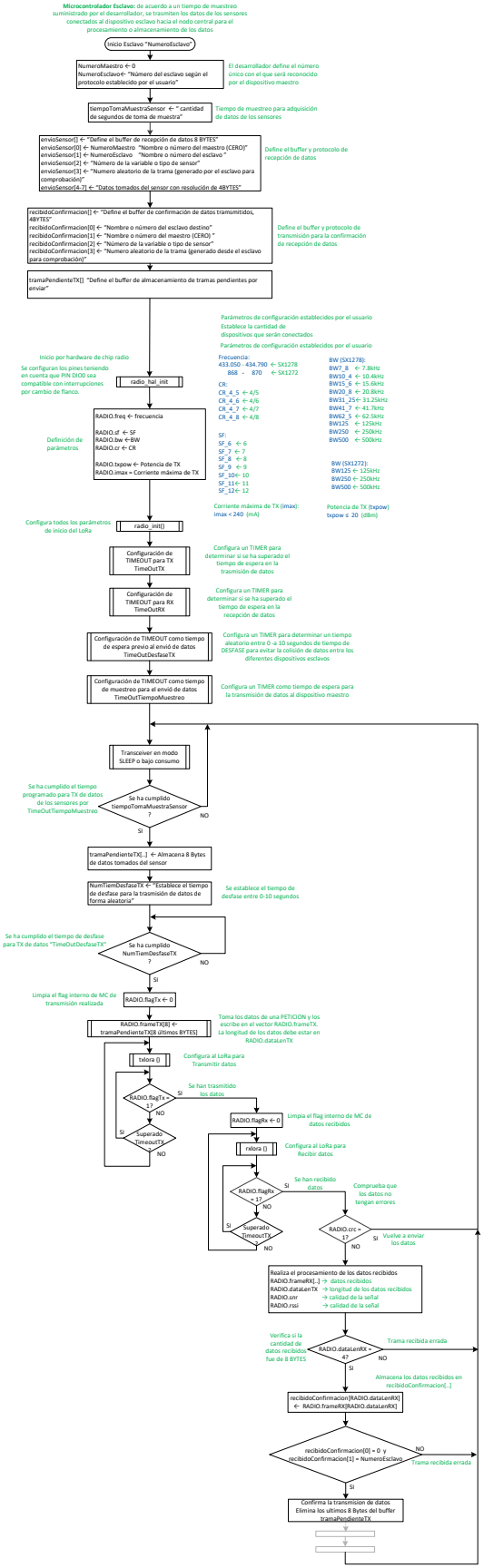


Figura 33. Esquema lógico para dispositivos esclavo en aplicación punto-multipunto
Fuente: Desarrollado por autor.

El esquema presentado en la *Figura 32* representa la lógica de programación planteada en el flujograma utilizado en sistema embebido maestro. Este dispositivo se encuentra continuamente en modo recepción de datos, cuando recibe la trasmisión de datos de dispositivos esclavos, procesa la información y retorna la confirmación. El dispositivo esclavo se encarga de determinar los valores de los sensores remotos, una vez se cumple con el tiempo de muestreo o captura de los valores de los sensores, se tramite al dispositivo maestro. En caso que dos o más dispositivos esclavos tramitan al mismo instante, el dispositivo maestro no responderá la confirmación y en el próximo tiempo cumplido de muestreo se trasmitirá los valores pendientes por trasmitir. El flujograma propuesto del dispositivo esclavo se presenta en la *Figura 33*.

BIBLIOGRAFÍA

Dorji (2015). DRF1278F - 20dBm LoRa Long Range RF Front-end Module. Version 1.11. China.

NiceRF (2015). LoRa1278 100 mW 4 km larga distancia y alta sensibilidad (-139 dBm) 433 MHz módulo de transceptor inalámbrico. Recuperado de http://es.aliexpress.com/item/2pcs-lot-LoRa1278-100mW-4km-Long-Distance-and-High-Sensitivity-139-dBm-433MHz-Wireless-Transceiver-Module/32461365864.html, China. Consultado el 22 de junio de 2016.

Semtech (2013). SX1272 Development kit. User guide. Semtech Corporation. Estados Unidos

Semtech SX1272 (2015). SX1272/73 - 860 MHz to 1020 MHz Low Power Long Range Transceiver. Datasheet. Semtech Corporation. Estados Unidos.

Semtech SX1278 (2015). SX1276/77/78/79 - 137 MHz to 1020 MHz Low Power Long Range Transceiver. Datasheet. Semtech Corporation. Estados Unidos.

Texas Instruments (2005). SM-Band and Short Range Device Regulatory Compliance Overview. Estados Unidos.

ANEXO B

Instalación y uso del driver LoRa para sistemas embebidos.

JOSÉ DANIEL RODRÍGUEZ MUNCA

Tutor Académico: **Manuel Sierra Castañer**

Tutor Profesional: **Álvaro Gutiérrez Martín**

UNIVERSIDAD DE POLITÉCNICA DE MADRID
UNIVERSIDAD COMPLUTENSE DE MADRID
MASTER INTERUNIVERSITARIO EN ESTRATEGIAS Y TECNOLOGÍAS PARA EL DESARROLLO
MADRID - ESPAÑA
2016

Tabla de Contenido

INSTALACIÓN Y USO DEL DRIVER LORA PARA SISTEMAS EMBEBIDOS ARDUINO 5

Estructura de archivos que componen el driver 5

Instalación de la librería para el uso con el IDE Arduino ..... **¡Error! Marcador no definido.**

Uso de librería para el control de dispositivos LoRa en aplicaciones o sketch de Arduino ..... **¡Error! Marcador no definido.**

Ejemplo de aplicación: red punto a punto (aplicación ping-pong) 12

Ejemplo de aplicación: recepción de datos de varios dispositivos remotos (red punto - multipunto) 14

INSTALACIÓN Y USO DEL DRIVER LORA PARA SISTEMAS EMBEBIDOS SDIN ... 17

BIBLIOGRAFÍA 27

LISTADO DE FIGURAS

Figura 1. Estructura de archivos y carpetas del driver propuesto para el manejo de dispositivos LoRa utilizando Arduino 5
Figura 2. Gestión entre archivos del driver propuesto para el manejo de dispositivos LoRa .. 6
Figura 3. Base de aplicación o "sketch" para utilizar transceiver LoRa con IDE Arduino ... 11
Figura 4. Principio de funcionamiento de aplicación "ping-pong" 13
Figura 5. Esquema de conexión para LoRa1278 de NiceRF y Arduino UNO 13
Figura 6. Tipología estrella para comunicación con diversos dispositivos Arduino 15
Figura 7. Estructura de archivos y carpetas del driver propuesto para el manejo de dispositivos LoRa utilizando FreeRTOS -SDIN 17
Figura 8. Gestión entre archivos del driver propuesto para el manejo de dispositivos LoRa para los módulos SDIN 19
Figura 9. Principio de funcionamiento de aplicación "ping-pong"con SDIN 23
Figura 10. Esquema de conexión para LoRa1278 de NiceRF y módulo SDIN 23
Figura 11. Tipología estrella para comunicación con diversos dispositivos 25

LISTADO DE TABLAS

Tabla 1. Instrucción para selección de transceiver LoRa en archivos radio.h.....7
Tabla 2. Banda de frecuencia ISM compatibles con los transceivers de radio LoRa.....9
Tabla 3. Combinación de los principales parámetros configurables para los LoRa 10
Tabla 4. Comportamiento de los registros tipo flag o banderas de los estados finales de trasmisión o recepción de datos para los LoRa 12
Tabla 5. Nomenclatura utilizada para la conexión entre la tarjeta LoRa1278 de NiceRF y la tarjeta Arduino UNO.....14
Tabla 6. Nomenclatura utilizada para la conexión entre la tarjeta LoRa1278 de NiceRF y el módulo SDIN 24

INSTALACIÓN Y USO DEL DRIVER LORA PARA SISTEMAS EMBEBIDOS ARDUINO

Estructura de archivos que componen el driver

El driver está compuesto por una serie de archivos con *código fuente* y archivos tipo *header* organizados en dos subcarpetas. La estructura organizada de archivos se presenta en la *Figura 1*.

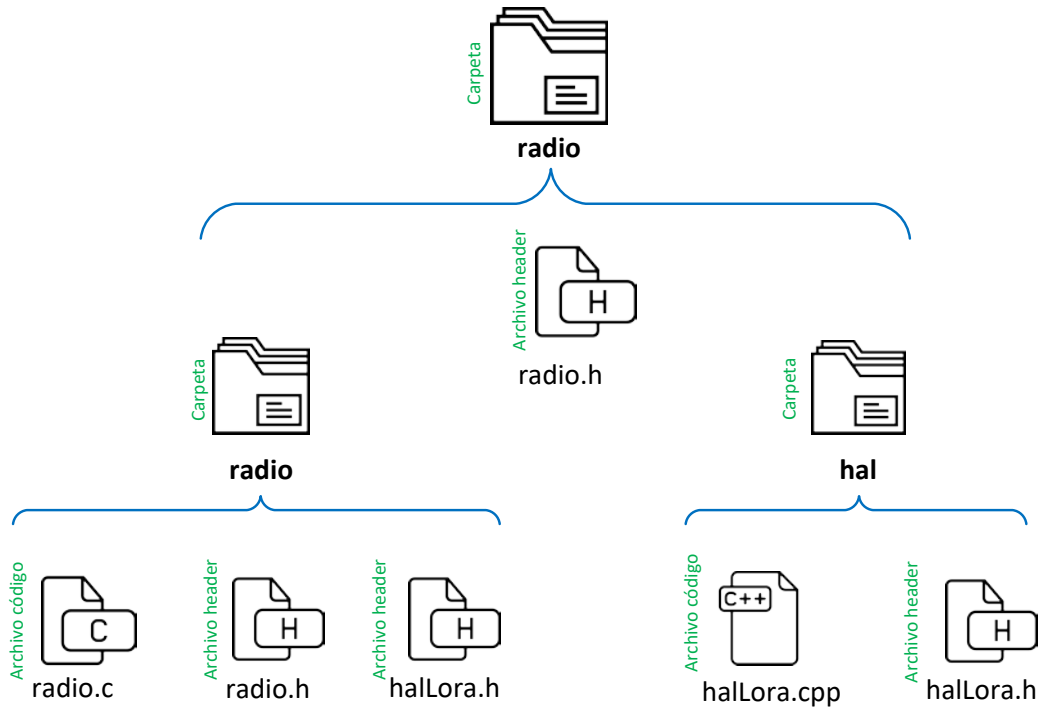


Figura 1. Estructura de archivos y carpetas del driver propuesto para el manejo de dispositivos LoRa utilizando Arduino
Fuente: Esquema propuesto por autor

La carpeta principal de denomina **radio** y está compuesta por los elementos:

- **Carpeta radio**: contiene los archivos necesarios para el funcionamiento del driver en aplicaciones o "*sketch*" para Arduino.
    - **radio.h**: el *header file* contiene la definición o selección del transceiver LoRa utilizado por la aplicación de Arduino. Es suficiente con definir:

Instrucción			Comentario
#define	CFG_1272_SEMTECH	1	Selecciona el CHIP **SX1272** de **SEMTECH**
#define	CFG_1278_DRF1278F	1	Selecciona el CHIP **SX1278** de **DORJI** versión de tarjeta **DRF1278F**
#define	CFG_1278_NICERF1278	1	Selecciona el CHIP **SX1278** de **NiceRF** versión de tarjeta **LORA1278**

- **Carpeta radio**: la carpeta contiene el código fuente, junto la definición de variables y funciones necesarias para la configuración de los dispositivos LoRa.
  - **radio.h**: contiene la definición de variables y funciones para la configuración de los LoRa a nivel de software relacionado con el código presentado en **radio.c**.
  - **radio.c**: contiene las funciones para la configuración de los LoRa a nivel de software.
  - **halLora.h:** cuenta con la definición de funciones del archivo **halLoRa.cpp** (*HAL o hardware abstraction layer*).

- **Carpeta hal**: la carpeta contiene el código fuente relacionado con la capa de más bajo nivel que configura los periféricos internos de Arduino.
  - **halLora.h**: cuenta con la definición de la nomenclatura asignada a los pines o terminales de control de los dispositivos LoRa.
  - **halLora.cpp**: contiene el código fuente de la capa de más bajo nivel que configura los periféricos internos de arduino (interrupciones por cambio de flanco, comunicación SPI, comunicación UART, cambios de estados en terminales). A esta capa se le conoce como *HAL* o capa de abstracción de hardware.

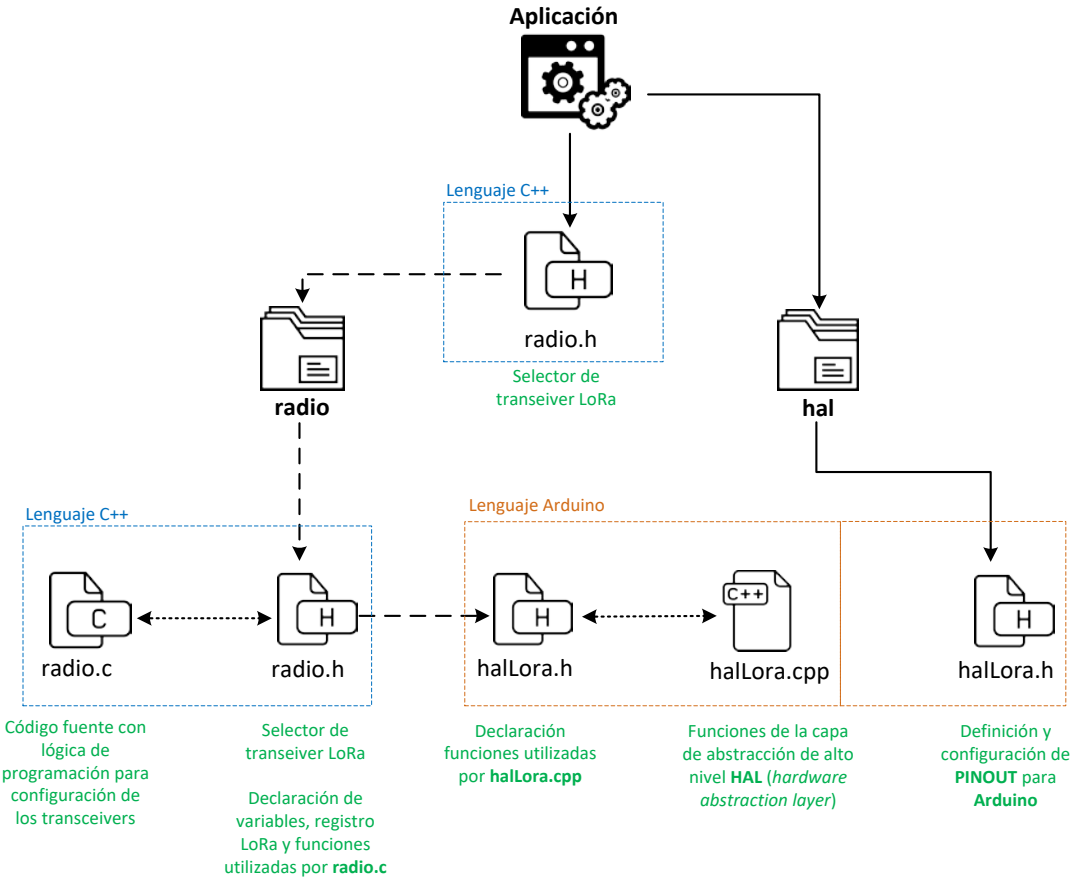


Figura 2. Gestión entre archivos del driver propuesto para el manejo de dispositivos LoRa
Fuente: Esquema propuesto por autor

La *Figura 2* presenta la gestión entre los diferentes archivos que componen el driver planteado.

Resulta estratégica la organización de los diferentes archivos debido a que el driver busca que el código presentado en ***radio.c*** se comporte como una capa de alto nivel, manteniendo el código invariante al uso de diferentes sistemas embebidos como Arduino, módulos SDIN o tarjetas de STMicroelectronics; mientras que el código presentado en **halLora.cpp** corresponde a una capa de alto bajo nivel de abstracción de hardware que de acuerdo al tipo de sistema embebido utilizado pueda ser ajustado para realizar la correcta homologación al microcontrolador utilizado.

Instalación de la librería para el uso con Arduino

La instalación del driver para el control de los dispositivos LoRa utilizando Arduino (se recomienda que el equipo de cómputo tenga instalado el IDE Arduino), se realiza copiando los archivos del fichero principal **radio** en la carpeta *libraries* de la instalación del *IDE arduino*:

Ruta por defecto para Linux (ubuntu)	*Ruta por defecto para Windows*
/usr/share/arduino/libraries	C:\Program Files (x86)\Arduino\libraries
	C:\Program Files\Arduino\libraries

Una vez copiado los archivos en la carpeta *libraries* de la *instalación del IDE arduino*, el usuario debe seleccionar el dispositivo LoRa que será utilizado por Arduino. Para seleccionar el transceiver, se debe habilitar una de las siguientes instrucciones en los archivos **radio/radio.h** y **radio/radio/radio.h**:

Tabla 1. Instrucción para selección de transceiver LoRa en archivos radio.h.
Fuente: Planteamiento del autor

Instrucción			Selección de transeiver LoRa
#define	CFG_1272_SEMTECH	1	**SX1272** de **SEMTECH**
#define	CFG_1278_DRF1278F	1	**SX1278** de **DORJI** versión de tarjeta **DRF1278F**
#define	CFG_1278_NICERF1278	1	**SX1278** de **NiceRF** versión de tarjeta **LORA1278**

Uso de librería para el control de dispositivos LoRa en aplicaciones o sketch de Arduino

Una vez copiado los archivos del driver en la carpeta donde se ha instalado el IDE Arduino, el usuario debe crear una aplicación o sketch y seguir los siguientes pasos:

- Incluir la librerías para control por software (**radio**) y hardware (**halLora**):

```
#include <radio.h>              //Librería de alto nivel para configuración por software de los radio LoRa
#include <hal/halLora.h>        //Librería de bajo nivel para configuración de hardware radio LoRa (HAL)
#include <SPI.h>                //Librería para manejo de comunicación SPI
```

- Definir los pines de control de la tarjeta Arduino con respecto a los transceiver LoRa:

```
//Definición de los pines de control de la tarjeta Arduino
const radio_pinmap radio_pins = {
   .nss =          [número de pin],
   .rxtx =RADIO_UNUSED_PIN,
   .rst =          [número de pin],
   .dio0 =         [número de pin],

#if (defined CFG_1278_DRF1278F) || (defined CFG_1272_SEMTECH)      //Selección la versión del SX1278
   .dio3 =         [número de pin],
#endif
#if defined(CFG_1278_NICERF1278)                                   //Selecciona el CHIP SX1278 de NICERF
   .tx_en =        [número de pin],
   .rx_en =        [número de pin],
#endif
};
```

- Configurar los dispositivos LoRa de acuerdo a las necesidades del usuario. Esto se debe incorporar en la función propia de Arduino "*void setup()*":
  - Iniciar las variables tipo *flag* o *banderas*:

```
//Inicialización de variables del driver
RADIO.flagTx = 0;
RADIO.flagRx =0;
RADIO.crc = 0;
```

  - Definir la frecuencia de trabajo del transceiver:

```
RADIO.freq = [frecuencia de trabajo];        //Frecuencia de la banda ISM compatible con LoRa
```

Teniendo en cuenta que existen bandas de frecuencias de uso privativo (telefonía, internet, radio, entre otras), existen frecuencias de libre uso conocidas como ISM (Industrial, Scientific and Medical) que son frecuencias intencionalmente libres para uso no comercial en aplicaciones de tipo industrial, científico y médico. Se recomienda al usuario verificar la compatibilidad de la frecuencia de trabajo con la *Tabla 2* (bandas de uso ISM).

Tabla 2. Banda de frecuencia ISM compatibles con los transceivers de radio LoRa
Fuente: Texas Instruments (2005)

Banda ISM (MHz)	Frecuencia mínima (MHz)	Frecuencia máxima (MHz)	Ancho de Banda (MHz)	Usos por continente
433	433.050	434.790	1.84	Europa, África y parte del norte de Asia.
869	868	870	2	Europa, África, Asia y Oceanía.
900	902	928	26	Continente americano

- Definir la potencia de trasmisión del transceiver:

```
//Configuración de potencia
RADIO.txpow = [Máxima potencia (dBm)]          //Máxima TX potencia
RADIO.imax =  [Corriente máxima (mA)]          //Por defecto 100mA   45mA <=Imax <= 240mA
```

La potencia máxima se expresa en dBm y el valor definido debe estar entre 2 a 17.
La corriente máxima está dada en mA y el valor definido se debe encontrar entre 45mA y 240mA. El valor típico de corriente es de 100mA. Para valores significativos de corrientes (mayores a 125mA), se recomienda verificar la corriente máxima de suministro soportada por la fuente o batería de alimentación. Algunos sistemas embebidos como Arduino cuentan con reguladores con corriente máxima de suministro hasta de 120mA, por lo que al realizar una inadecuada configuración en el LoRa, podría destruirse el regulador y dañar la placa del sistema embebido.

- Definir las variables de configuración de la comunicación entre los transceivers LoRa:

```
//Variables de configuración de la comunicación LoRa
RADIO.sf =  [Spread factor SF]          //Configuración del Spread factor
RADIO.bw =  [Ancho del canal BW]        //Configuración del ancho de canal
RADIO.cr =  [Coding rate CR]            //Configuración de Coding rate
```

Los dispositivos LoRa cuentan con los parámetros *SF*, *BW* y *CR* que según las combinaciones entre estos, permite determinar el alcance o distancia máxima de recepción de datos, la velocidad de los datos y sobrecarga por corrección o detección de errores en los datos. Estos valores deben ser los mismos en todos los dispositivos radio LoRa de la red inalámbrica.

Los parámetros independientes de configuración se definen como (Semtech SX1272, 2015 & Semtech SX1278, 2015):

BW: Ancho de banda (entre más bajo sea este valor, el tiempo de durante la trasmisión será mayor)

SF rate: Es el factor de alcance expresado como logaritmo de base 2. Entre mayor sea este valor, se tendrá un mejor rendimiento en la trasmisión de datos.

CR: Tasa de codificación de errores: Entre mayor sea este valor, mayor será la fiabilidad de los datos, pero con una sobrecarga en el tiempo de transmisión.

La *Tabla 3* presenta la combinación de los principales variables independientes para la configuración de la comunicación entre transceivers LoRa. Cabe aclarar que estos valores deben ser los mismos en todos los dispositivos radio LoRa de la red inalámbrica.

Tabla 3. Combinación de los principales parámetros configurables para los LoRa
Fuente: Semtech SX1278 (2015) & Semtech SX1272 (2015)

Ancho de banda BW (kHz)		Factor de alcance SF		Tasa de codificación de errores. CR	
Hardware	Software **RADIO.BW**	Hardware	Software **RADIO.sf**	Hardware	Software **RADIO.cr**
500	BW500	6 → 64 chips / symbol (modo utilizado para FSK)	SF_6	4/5	CR_4_5
250	BW250	7 → 128 chips / symbol	SF_7	4/6	CR_4_6
125	BW125	8 → 256 chips / symbol	SF_8	4/7	CR_4_7
62.5*	BW62_5*	9 → 512 chips / symbol	SF_9	4/8	CR_4_8
41.7*	BW41_7*	10 → 1024 chips / symbol	SF_10		
31.25*	BW31_25*	11 → 2048 chips / symbol	SF_11		
20.8*	BW20_8*	12 → 4096 chips / symbol	SF_12		
15.6*	BW15_6*				
10.4*	BW10_4*				
7.8*	BW7_8*				

*: Aplica solamente para los transceiver SX1278

- Configurar los pines I/O y la comunicación SPI:

```
hal_init()                    //Configuración de puertos de Arduino
```

- Configurar los pines I/O y la comunicación SPI:

```
radio_init();                 //Configuración de inicio para los LoRa
```

Corresponde a la función general que configura todos los parámetros de inicio del transceiver LoRa.

- Crear la aplicación con la lógica propuesta por el desarrollador:

```
void loop()
{
   [Lógica de control implementada por el usuario]
}
```

SKETCH

```
/*************************************************************************
 * Definiciones necesarias para utilizar driver generado
 *************************************************************************/

#include <radio.h>                  //Librería de alto nivel para configuración por software de los radio LoRa
#include <hal/halLora.h>            //Librería de bajo nivel para configuración de hardware radio LoRa (HAL)
#include <SPI.h>                    //Librería para manejo de comunicación SPI

//Definición de los pines de control de la tarjeta Arduino
const radio_pinmap radio_pins = {
   .nss =       [número de pin],
   .rxtx =RADIO_UNUSED_PIN,
   .rst =       [número de pin],
   .dio0 =      [número de pin],

#if (defined CFG_1278_DRF1278F)  ||  (defined CFG_1272_SEMTECH)          //Selección la versión/fabricante del SX1278
   .dio3 =      [número de pin],
#endif
#if defined(CFG_1278_NICERF1278)                                         //Selecciona el CHIP SX1278 de NICERF
   .tx_en =     [número de pin],
   .rx_en =     [número de pin],
#endif
};

//Declaración de variables
[variables]

//Configuración de los dispositivos LoRa
void setup()                                        //Función de configuración inicial
{
   //Configuración de comunicación para realizar debugger
   Serial.begin([velocidad de comunicación del puerto USB(COM) para debugger]);

   //Inicialización de variables de aplicación general
   [variables = 0]

   //Inicialización de variables del driver
   RADIO.flagTx = 0;
   RADIO.flagRx =0;
   RADIO.crc =    0;

   //Configuración de frecuencia
   RADIO.freq = [frecuencia de trabajo (Hz)]

   //Configuración de potencia
   RADIO.txpow = [Máxima potencia (dBm): 2 – 17]              //Máxima TX potencia
   RADIO.imax =   [Corriente máxima (mA):  45 <=Imax <= 240]  //Por defecto 100mA   45mA <=Imax <= 240mA

   //Variables de configuración de la comunicación LoRa
   RADIO.sf =   [Spread factor: SF_6, SF_7, SF_8, SF_9, SF_10, SF_11, SF_12 ]   //Configuración del Spread factor
   RADIO.bw = [Ancho del canal: BW7_8, BW10_4, BW15_6, BW20_8, BW31_25, BW41_7, BW62_5, BW125, BW250, BW500]
   RADIO.cr =  [Coding rate: CR_4_5, CR_4_6, CR_4_7, CR_4_8]                    //Configuración de Coding rate

   hal_init();             //Configuración de puertos de Arduino
   radio_init();           //Configuración de inicio para los LoRa

   Serial.flush();
}

void loop()
{
   [Lógica de control implementada por el usuario]
}
```

Figura 3. Base de aplicación o "sketch" para utilizar transceiver LoRa con IDE Arduino
Fuente: Código propuesto por autor

La *Figura 3* presenta el código propuesto para el uso del driver generado para aplicaciones con dispositivos LoRa y sistemas embebidos Arduino.

Cabe aclara que de acuerdo al modo de configuración de los dispositivos LoRa, los registros tipo *flags* o *banderas* se comportan de acuerdo a la lógica presentada en la *Tabla 4*.

Tabla 4. Comportamiento de los registros tipo flag o banderas de los estados finales de trasmisión o recepción de datos para los LoRa
Fuente: Propuesta del autor

Registro tipo flag o bandera	Estado	Comentario
RADIO.flagTx	0	En modo trasmisión, indica que aún **no** se ha trasmitido la información del buffer *RADIO.frameTX* de cantidad *RADIO.dataLenTX Bytes.*
	1	En modo trasmisión, indica que se ha trasmitido la información del buffer *RADIO.frameTX* de cantidad *RADIO.dataLenTX Bytes.*
RADIO.flagRx	0	En modo recepción, indica que aún no se ha recibido información.
	1	En modo recepción, indica que se ha recibido *RADIO.dataLenRX Bytes* en el buffer *RADIO.frameRX* y que el usuario debe gestionar la información.
RADIO.crc	0	En modo recepción, indica que si *RADIO.flagRx* = 1, la información recibida en el buffer *RADIO.frameRX* es **correcta**.
	1	En modo recepción, indica que si *RADIO.flagRx* = 1, la información recibida en el buffer *RADIO.frameRX* es **errada**.

El driver también generan las variables:

- ***RADIO.snr:*** Calidad de la señal a ruido de los datos recibidos.
- ***RADIO.rssi:*** Indicador de fuerza de los datos recibidos.

Ejemplo de aplicación: red punto a punto (aplicación ping-pong)

Como ejemplo se plantea el uso de dos sistemas embebidos, uno utilizado como maestro y el otro utilizado como esclavo.

El dispositivo maestro se encarga de realizar la petición de datos a un dispositivo esclavo ubicado en un sitio remoto, el cual obtiene datos de sensores. El esclavo remite la información solicitada y continuamente se encuentra esperando peticiones del dispositivo esclavo. La

Figura 4 se presenta un diagrama pictórico de la aplicación propuesta: maestro-esclavo (sensor) con la utilización de dispositivo LoRa.

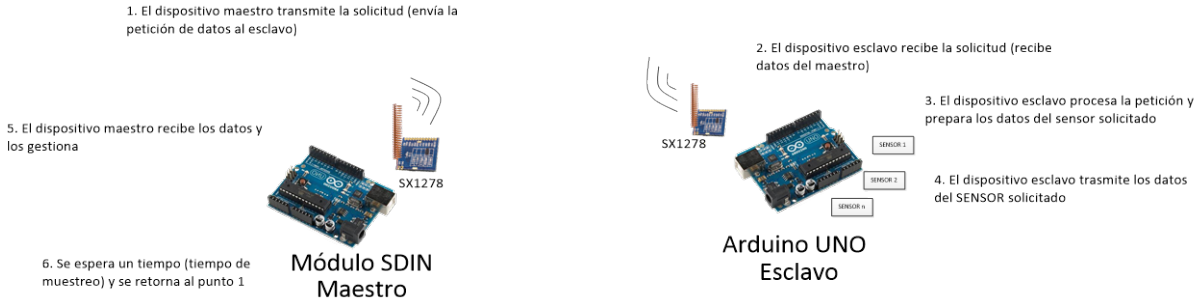


Figura 4. Principio de funcionamiento de aplicación "ping-pong"
Fuente: Desarrollado por autor

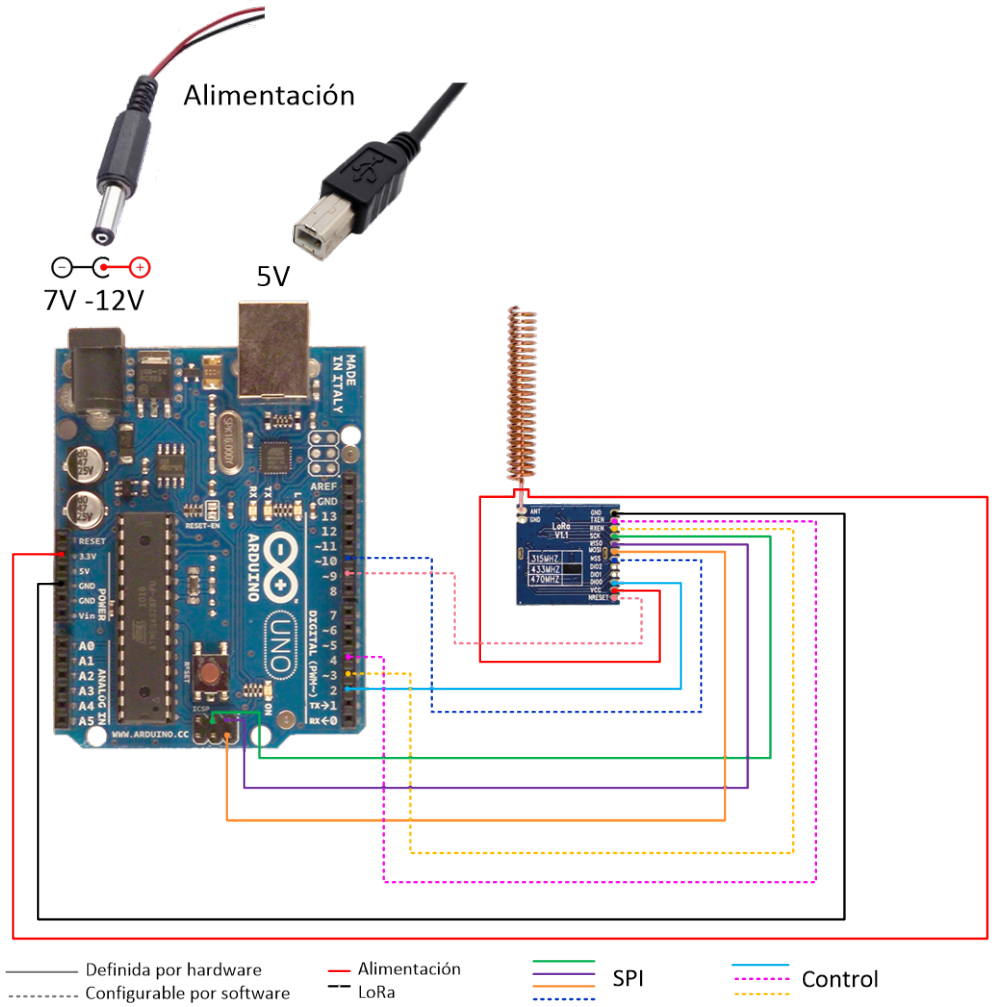


Figura 5. Esquema de conexión para LoRa1278 de NiceRF y Arduino UNO
Fuente: Esquema propuesto por el autor

La *Figura 5* presenta el esquema de conexión utilizado entre los dispositivos LoRa SX1278 y el sistema embebido Arduino Uno. La *Tabla 5* presenta la nomenclatura utilizada en el transceiver SX1278 y la tarjeta Arduino Uno.

Tabla 5. Nomenclatura utilizada para la conexión entre la tarjeta LoRa1278 de NiceRF y la tarjeta Arduino UNO
Fuente: NiceRF (2015)

Tipo	Nomenclatura estándar	Notación en tarjetas	
		LoRa1278	Arduino UNO
Alimentación	VCC	VCC	3.3V
	GND	GND	GND
SPI	NSS	NSS	10
	MOSI	MOSI	11
	MISO	MISO	12
	SCK	SCK	13
CONTROL	RESET	NRESET	9
	DIO0	DIO0	2
	TXEN	TXEN	3
	RXEN	RXEN	4

El flujograma del código fuente empleado para la prueba se encuentra en el "*ANEXO A Desarrollo del driver para el uso de los dispositivos de radio LoRa con sistemas embebidos*" y los códigos fuente se presenta en la carpeta *radio/ejemplos/ping_pong* que acompaña al driver.

El sketch *Master_Sx1278_PingPong.ino* corresponde al archivo maestro y el sketch *Esclavo_Sx1278_PingPong.ino* corresponde al código del sistema embebido esclavo.

Ejemplo de aplicación: recepción de datos de varios dispositivos remotos (red punto - multipunto)

El ejemplo plantea un esquema de dispositivos conectados por una red punto-multipunto tipo estrella con un nodo central (circuito maestro) y 3 dispositivos esclavos.
La *Figura 6* presenta el esquema propuesto donde se aprecia la comunicación simultánea de 3 dispositivos hacia un nodo central. Se plantea el uso de esta aplicación con el uso del driver desarrollado para aplicaciones complejas con diversos sensores ubicados en sitios remotos. Cabe resaltar que los dispositivos LoRa deben ser de la misma referencia, estar configurados a la misma frecuencia y con los parámetros SF, BW y CR idénticos.

El dispositivo maestro se debe configurar como dispositivo “pasivo”, es decir continuamente recibir la información de los dispositivos esclavos. Una vez recibe la información de un dispositivo esclavo, remite la confirmación de recepción correcta de datos.

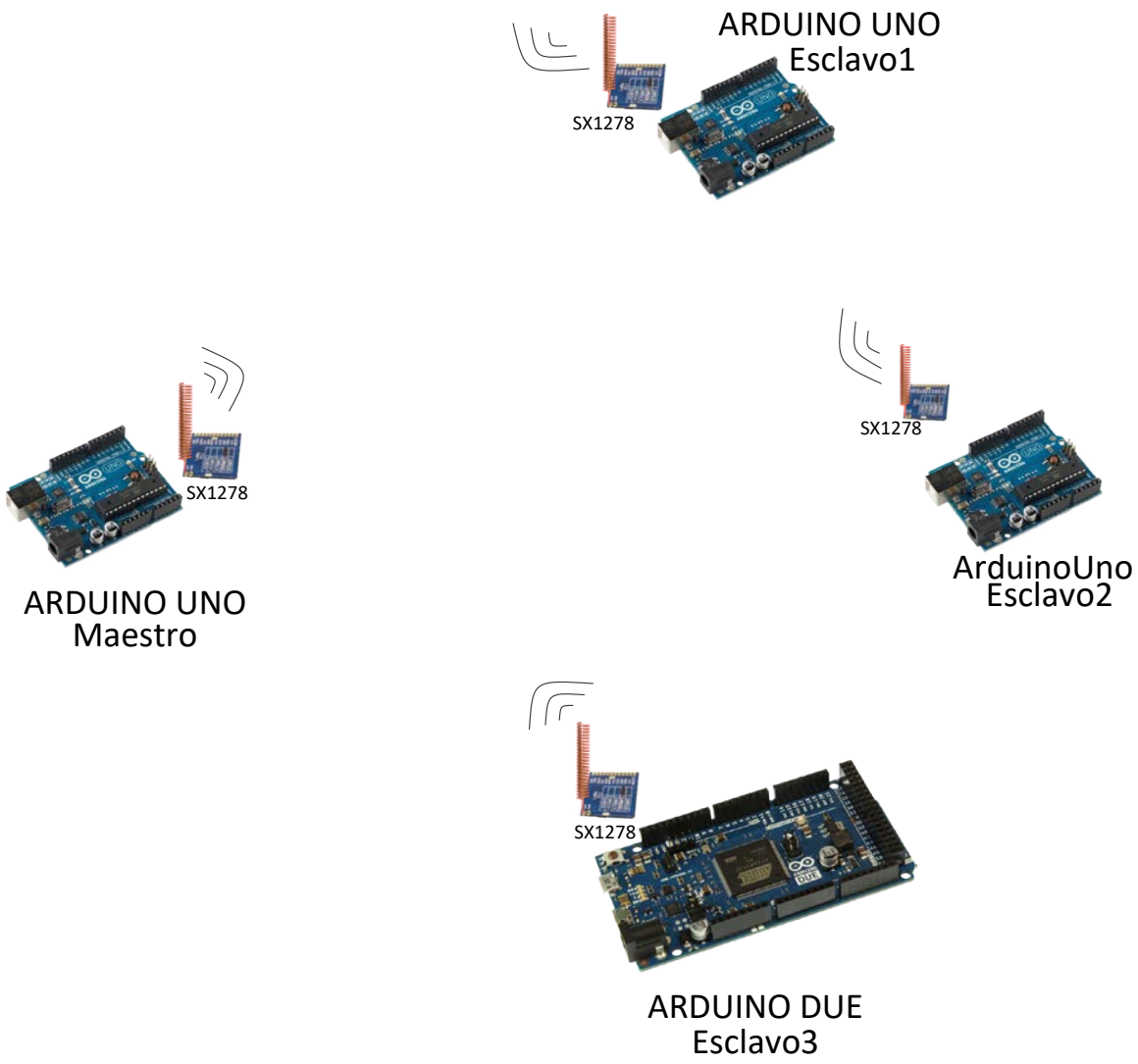


Figura 6. Tipología estrella para comunicación con diversos dispositivos Arduino
Fuente: Esquema propuesto por el autor

El modem LoRa es *half dúplex*, es decir que los LoRa no puede trasmitir o recibir datos en el mismo instante; esto significa que los transceiver LoRa deben ser configurados para trasmitir o recibir datos. Cuando se incorpora más de dos dispositivos en un protocolo de comunicación *half dúplex*, se deben establecer estrategias que mitiguen los efectos adversos por colisión indeseada de datos cuando se reciben datos.
Si los dispositivos no cuentan con estrategias para evitar la trasmisión de datos sobre el mismo canal de comunicación por parte de los dispositivos esclavos en un mismo instante, los datos llegaran errados al dispositivo maestro y la comunicación de datos será errada y

ocasiona un consumo energético innecesario de energía en la transmisión de información por parte de los esclavos.

La solución para evitar la colisión de datos, corresponde a establecer tiempos de muestreos y tiempos aleatorios de "des-sincronización" cuando se trasmiten datos; es decir evitar con los tiempos adiciones de trasmisión de datos la colisión de datos.

En el documento Anexo A: "Desarrollo del driver para el uso de los dispositivos de radio LoRa con sistemas embebidos" – apartado *"Ejemplo de aplicación: recepción de datos de varios dispositivos remotos (red punto – multipunto)"*, se describe de manera detallada el proceso de desarrollo de la prueba desarrollada para crear la red punto-multipunto con dispositivos LoRa.

La *Figura 5* presenta el esquema de conexión utilizado entre los dispositivos LoRa SX1278 y el sistema embebido Arduino Uno. La *Tabla 5* presenta la nomenclatura utilizada en el transceiver SX1278 y la tarjeta Arduino Uno.
El código fuente utilizado por los dispositivos maestro y esclavos se presenta en la carpeta *radio/ejemplos/multipunto* que acompaña al driver.

El sketch *Master_Sx1278_Multipunto.ino* corresponde al archivo maestro y el sketch *Esclavo_Sx1278_ Multipunto.ino* corresponde al código del sistema embebido esclavo.

INSTALACIÓN Y USO DEL DRIVER LORA PARA SISTEMAS EMBEBIDOS SDIN

Estructura de archivos que componen el driver

El driver está compuesto por una serie de archivos con *código fuente* y archivos tipo *header* organizados en dos subcarpetas. La estructura organizada de archivos se presenta en la *Figura 7*.

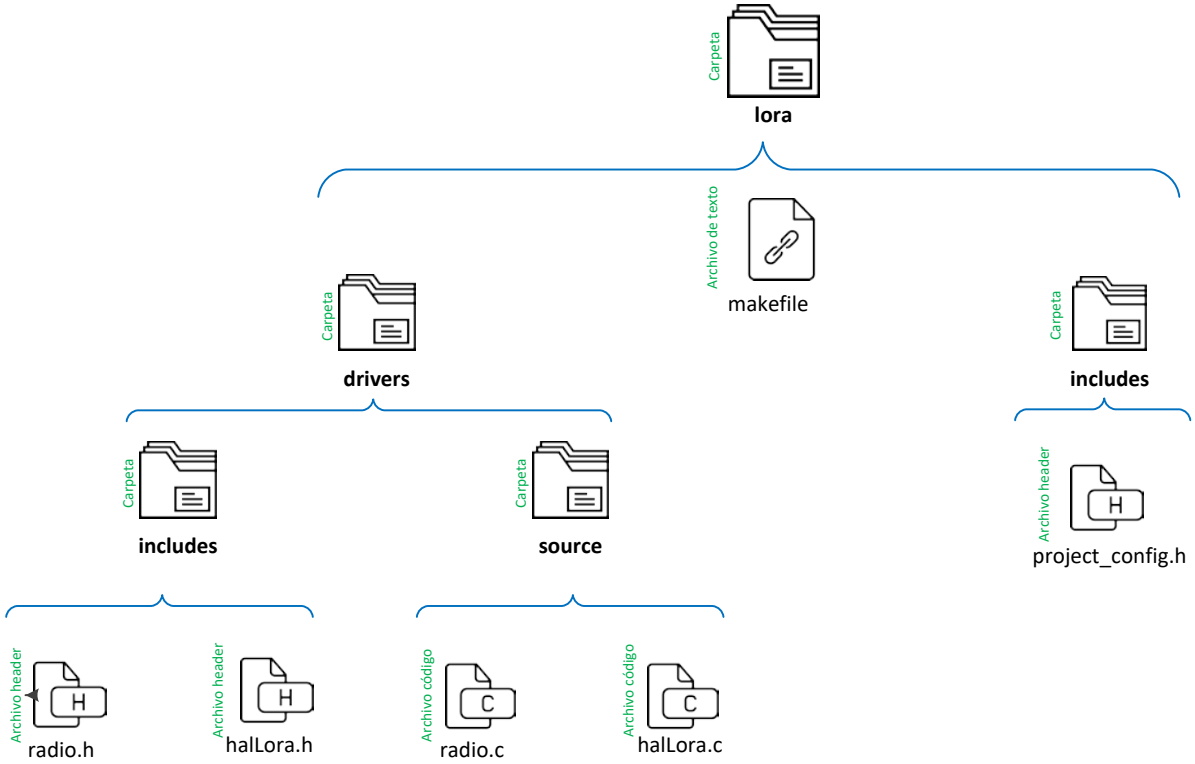


Figura 7. Estructura de archivos y carpetas del driver propuesto para el manejo de dispositivos LoRa utilizando FreeRTOS -SDIN
Fuente: Esquema propuesto por autor

La carpeta principal de denomina **lora** y está compuesta por los elementos:

- **Makefile**: es el archivo que cuenta con la información del proyecto para el uso de FreeRTOS. Con respecto al driver, el apartado "DRIVERS_SOURCE= \" debe contener la ruta de las librerías relacionadas con el control de los LoRa:

```
DRIVERS_SOURCE= \
        $(DRIVERS_SOURCE_DIR)/source/halLora.c      \
        $(DRIVERS_SOURCE_DIR)/source/radio.c        \
```

- **Carpeta drivers**:
  - Carpeta includes:
    - **radio.h**: contiene la definición de variables y funciones para la configuración de los LoRa a nivel de software relacionado con el código presentado en **radio.c**.

- **halLora.h:** cuenta con la definición de funciones del archivo **halLoRa.c** (*HAL o hardware abstraction layer*). Además contiene la definición de la nomenclatura asignada a los pines o terminales de control de los dispositivos LoRa.

o Carpeta source:

- **radio.c**: contiene las funciones para la configuración de los LoRa a nivel de software.
- **halLora.c**: contiene el código fuente de la capa de más bajo nivel que configura los periféricos internos de SDIN (interrupciones por cambio de flanco, comunicación SPI, comunicación UART, cambios de estados en terminales). A esta capa se le conoce como *HAL* o capa de abstracción de hardware.

- **Carpeta includes**:

o **project_config.h**: el *header file* contiene la definición o selección del transceiver LoRa utilizado por la aplicación de SDIN. Es suficiente con definir:

Instrucción			Comentario
#define	CFG_1272_SEMTECH	1	Selecciona el CHIP **SX1272** de **SEMTECH**
#define	CFG_1278_DRF1278F	1	Selecciona el CHIP **SX1278** de **DORJI** versión de tarjeta **DRF1278F**
#define	CFG_1278_NICERF1278	1	Selecciona el CHIP **SX1278** de **NiceRF** versión de tarjeta **LORA1278**

o

La *Figura 8* presenta la gestión entre los diferentes archivos que componen el driver planteado.

Resulta estratégica la organización de los diferentes archivos debido a que el driver busca que el código presentado en ***radio.c*** se comporte como una capa de alto nivel, manteniendo el código invariante al uso de diferentes sistemas embebidos como el módulo SDIN, módulos SDIN (microcontrolador de 32 bits STM32f10x) o tarjetas de STMicroelectronics; mientras que el código presentado en **halLora.c** corresponde a una capa de alto bajo nivel de abstracción de hardware que de acuerdo al tipo de sistema embebido utilizado pueda ser ajustado para realizar la correcta homologación al microcontrolador utilizado.

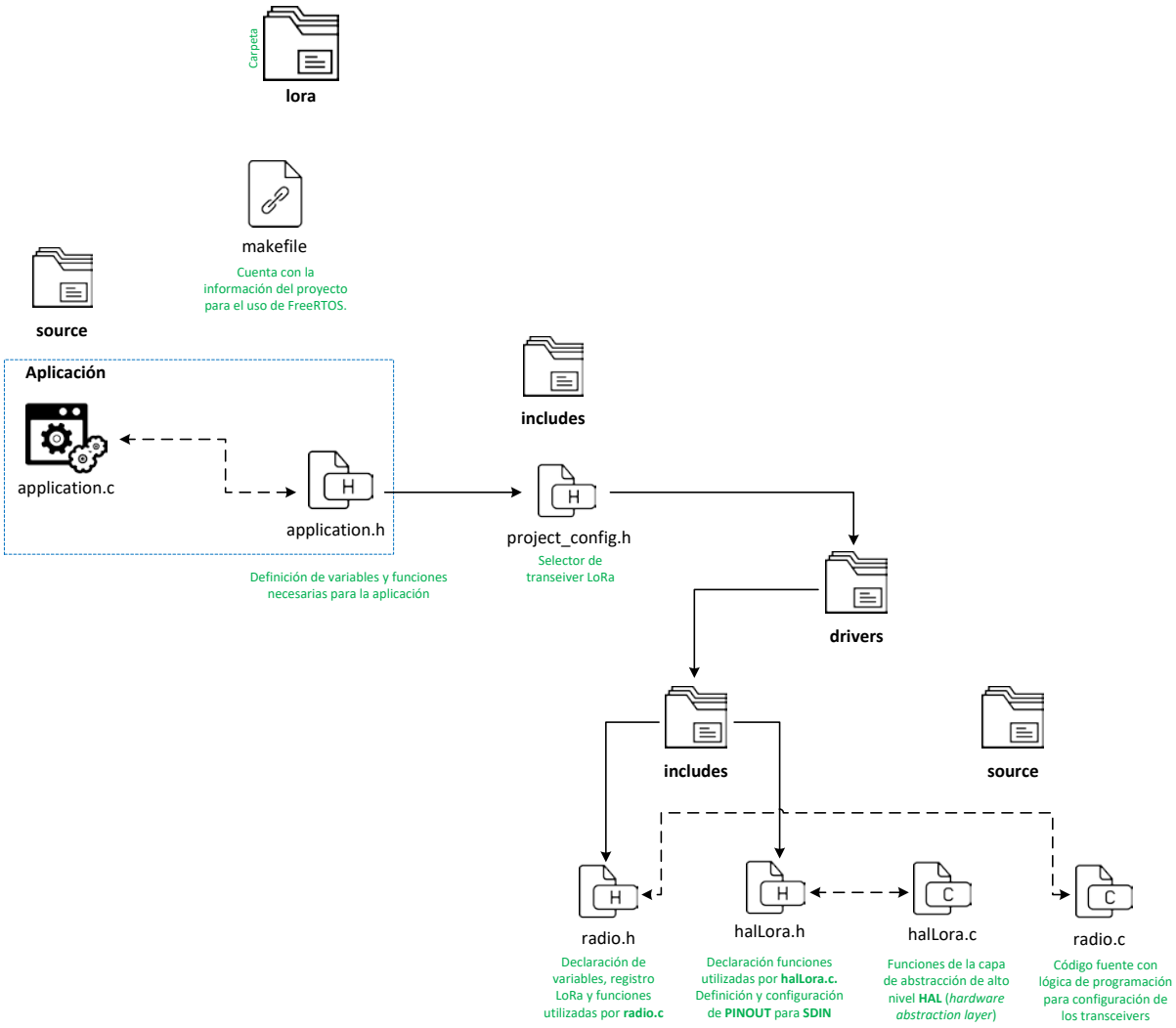


Figura 8. Gestión entre archivos del driver propuesto para el manejo de dispositivos LoRa para los módulos SDIN
Fuente: Esquema propuesto por autor

La *Figura 8* presenta la gestión entre los diferentes archivos que componen el driver planteado.

Resulta estratégica la organización de los diferentes archivos debido a que el driver busca que el código presentado en ***radio.c*** se comporte como una capa de alto nivel, manteniendo el código invariante al uso de diferentes sistemas embebidos como Arduino, módulos SDIN o tarjetas de STMicroelectronics; mientras que el código presentado en ***halLora.c*** corresponde a una capa de alto bajo nivel de abstracción de hardware que de acuerdo al tipo de sistema embebido utilizado pueda ser ajustado para realizar la correcta homologación al microcontrolador utilizado.

Uso de librería para el control de dispositivos LoRa en aplicaciones para SDIN

La instalación del driver para el control de los dispositivos LoRa utilizando SDIN (se recomienda que el equipo de cómputo tenga instalado el IDE Eclipse), se realiza copiando los archivos del fichero principal **lora** junto con la carpeta donde se encuentra instalado los archivos del S.O. FreeRTOS:

- Incluir la librerías para control por software (**radio**) y hardware (**halLora**) en el header file **application.h**:

```
#include "radio.h"          //Librería de alto nivel para configuración por software de los radio LoRa
#include "halLora.h"        //Librería de bajo nivel para configuración de hardware radio LoRa (HAL)
#include "project_config.h" //Librería para la selección del transceiver LoRa
```

- Definir los pines de control de la tarjeta SDIN con respecto a los transceiver LoRa:

```
//Definición de los pines de control de la tarjeta SDIN
#define LORA_SPI                    SPI2
#define LORA_SPI_CLK                RCC_APB1Periph_SPI2
#define LORA_SPI_IRQn               SPI2_IRQn
#define LORA_SPI_PORT               GPIO [letra del puerto]

// SPI CS/NSS
#define LORA_NSS_PIN                GPIO_Pin_[número del Pin]
#define LORA_NSS                    LORA_SPI_PORT, LORA_NSS_PIN

// SPI SCK
#define LORA_SCK_PIN                GPIO_Pin_[número del Pin]
#define LORA_SCK                    LORA_SPI_PORT, LORA_SCK_PIN

// SPI MISO
#define LORA_MISO_PIN               GPIO_Pin_[número del Pin]
#define LORA_MISO                   LORA_SPI_PORT, LORA_MISO_PIN

// SPI MOSI
#define LORA_MOSI_PIN               GPIO_Pin_[número del Pin]
#define LORA_MOSI                   LORA_SPI_PORT, LORA_MOSI_PIN


#define LORA_RST_PORT               GPIOA
#define LORA_RST_PIN                GPIO_Pin_[número del Pin]
#define LORA_RST                    LORA_RST_PORT, LORA_RST_PIN
#define  LORA_RST_MODE              GPIO_Mode_Out_PP
```

- Configurar los dispositivos LoRa de acuerdo a las necesidades del usuario. Esto se debe incorporar en la función propia de **application.c** (ejemplo "TestLora_Task()"):
    - Iniciar las variables tipo *flag* o *banderas*:

```
//Inicialización de variables del driver
RADIO.flagTx = 0;
RADIO.flagRx =0;
RADIO.crc = 0;
```

- Definir la frecuencia de trabajo del transceiver:

```
RADIO.freq = [frecuencia de trabajo];          //Frecuencia de la banda ISM compatible con LoRa
```

Teniendo en cuenta que existen bandas de frecuencias de uso privativo (telefonía, internet, radio, entre otras), existen frecuencias de libre uso conocidas como ISM (Industrial, Scientific and Medical) que son frecuencias intencionalmente libres para uso no comercial en aplicaciones de tipo industrial, científico y médico. Se recomienda al usuario verificar la compatibilidad de la frecuencia de trabajo con la *Tabla 2* (bandas de uso ISM).

- Definir la potencia de trasmisión del transceiver:

```
//Configuración de potencia
RADIO.txpow = [Máxima potencia (dBm)]          //Máxima TX potencia
RADIO.imax =  [Corriente máxima (mA)]          //Por defecto 100mA   45mA <=Imax <= 240mA
```

La potencia máxima se expresa en dBm y el valor definido debe estar entre 2 a 17.
La corriente máxima está dada en mA y el valor definido se debe encontrar entre 45mA y 240mA. El valor típico de corriente es de 100mA. Para valores significativos de corrientes (mayores a 125mA), se recomienda verificar la corriente máxima de suministro soportada por la fuente o batería de alimentación.

- Definir las variables de configuración de la comunicación entre los transceivers LoRa:

```
//Variables de configuración de la comunicación LoRa
RADIO.sf =   [Spread factor SF]          //Configuración del Spread factor
RADIO.bw =  [Ancho del canal BW]         //Configuración del ancho de canal
RADIO.cr =  [Coding rate CR]             //Configuración de Coding rate
```

Los dispositivos LoRa cuentan con los parámetros *SF*, *BW* y *CR* que según las combinaciones entre estos, permite determinar el alcance o distancia máxima de recepción de datos, la velocidad de los datos y sobrecarga por corrección o detección de errores en los datos. Estos valores deben ser los mismos en todos los dispositivos radio LoRa de la red inalámbrica.

Los parámetros independientes de configuración se definen como (Semtech SX1272, 2015 & Semtech SX1278, 2015):

BW: Ancho de banda (entre más bajo sea este valor, el tiempo de durante la trasmisión será mayor)

SF rate: Es el factor de alcance expresado como logaritmo de base 2. Entre mayor sea este valor, se tendrá un mejor rendimiento en la trasmisión de datos.

CR: Tasa de codificación de errores: Entre mayor sea este valor, mayor será la fiabilidad de los datos, pero con una sobrecarga en el tiempo de transmisión.

La *Tabla 3* presenta la combinación de los principales variables independientes para la configuración de la comunicación entre transceivers LoRa. Cabe aclarar que estos valores deben ser los mismos en todos los dispositivos radio LoRa de la red inalámbrica.

- Configurar los pines I/O y la comunicación SPI:

```
hal_init()                    //Configuración de puertos de SDIN
```

- Configurar los pines I/O y la comunicación SPI:

```
radio_init();                 //Configuración de inicio para los LoRa
```

Corresponde a la función general que configura todos los parámetros de inicio del transceiver LoRa.

- Crear la aplicación con la lógica propuesta por el desarrollador:

```
while(1)
{
   [Lógica de control implementada por el usuario]
}
```

- En la función **Application_Start()**, crear la tarea relacionada con la aplicación de uso de los LoRa (ejemplo: TestLora_Task):

```
xTaskCreate(TestLora_Task, "TestLora_Task", TEST_LORA_TASK_STACK, NULL,
TEST_LORA_TASK_PRIORITY, &TestLoraTask_Handle);          //TestLora_Task->
PRUEBA UNO A UNO - Esclavo
```

Cabe aclara que de acuerdo al modo de configuración de los dispositivos LoRa, los registros tipo *flags* o *banderas* se comportan de acuerdo a la lógica presentada en la *Tabla 4*.

El driver también generan las variables:

- ***RADIO.snr:*** Calidad de la señal a ruido de los datos recibidos.
- ***RADIO.rssi:*** Indicador de fuerza de los datos recibidos.

Ejemplo de aplicación: red punto a punto (aplicación ping-pong) con módulos SDIN

Como ejemplo se plantea el uso de dos sistemas embebidos, uno utilizado como maestro y el otro utilizado como esclavo.

El dispositivo maestro se encarga de realizar la petición de datos a un dispositivo esclavo ubicado en un sitio remoto, el cual obtiene datos de sensores. El esclavo remite la información solicitada y continuamente se encuentra esperando peticiones del dispositivo esclavo. La

Figura 9 se presenta un diagrama pictórico de la aplicación propuesta: maestro-esclavo (sensor) con la utilización de dispositivo LoRa.

Figura 9. Principio de funcionamiento de aplicación "ping-pong"con SDIN
Fuente: Desarrollado por autor

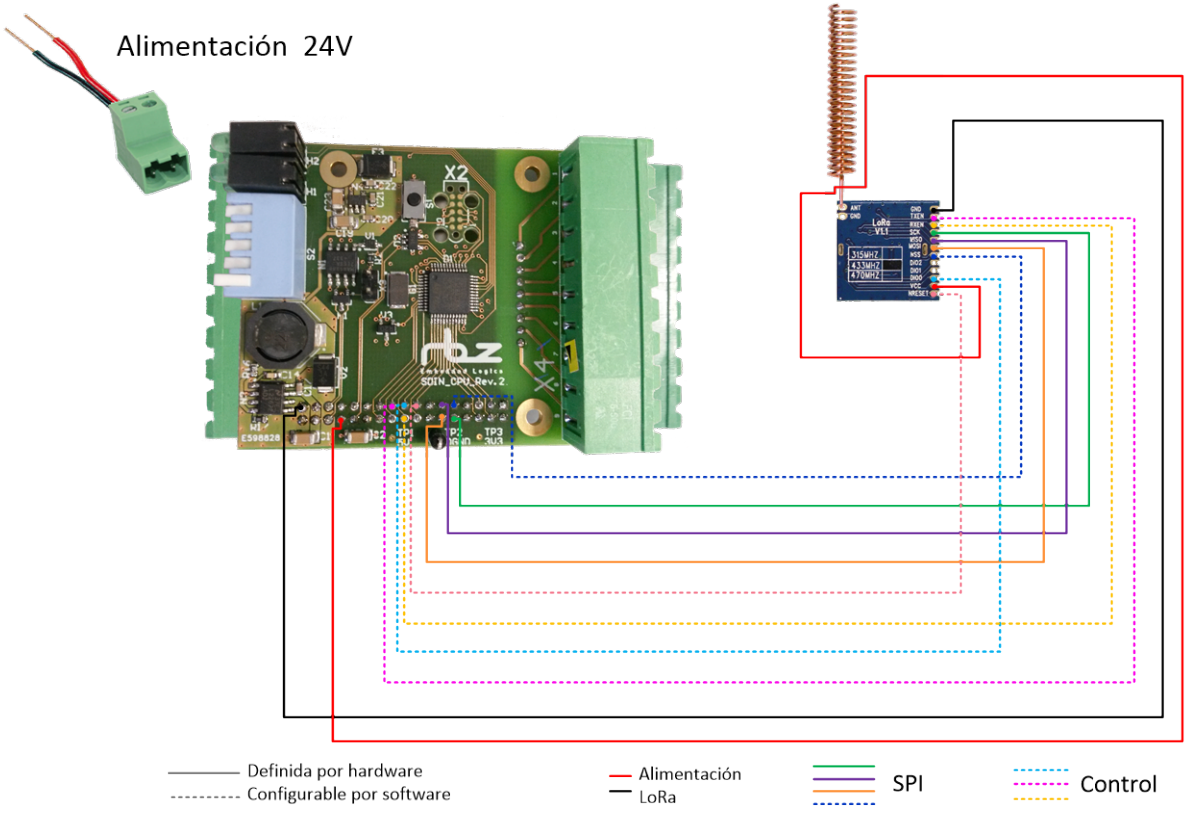


Figura 10. Esquema de conexión para LoRa1278 de NiceRF y módulo SDIN
Fuente: Esquema propuesto por el autor

La *Figura 10* presenta el esquema de conexión utilizado entre los dispositivos LoRa SX1278 y el sistema embebido SDIN. La *Tabla 6* presenta la nomenclatura utilizada en el transceiver SX1278 y la tarjeta SDIN.

Tabla 6. Nomenclatura utilizada para la conexión entre la tarjeta LoRa1278 de NiceRF y el módulo SDIN
Fuente: NiceRF (2015)

Tipo	Nomenclatura estándar	Notación en tarjetas	
		LoRa1278	SDIN
Alimentación	VCC	VCC	3.3V
	GND	GND	GND
SPI	NSS	NSS	PB12
	MOSI	MOSI	PB15
	MISO	MISO	PB14
	SCK	SCK	PB13
CONTROL	RESET	NRESET	PA7
	DIO0	DIO0	PA5
	TXEN	TXEN	PA3
	RXEN	RXEN	PA4

El flujograma del código fuente empleado para la prueba se encuentra en el "*ANEXO A Desarrollo del driver para el uso de los dispositivos de radio LoRa con sistemas embebidos*" y los códigos fuente se presenta en la carpeta *lora/source/application.c* que acompaña al driver.

La tarea se denomina *TestLora_Task()* corresponde al código fuente del dispositivo maestro del sistema embebido SDIN.

Ejemplo de aplicación: recepción de datos de varios dispositivos remotos (red punto - multipunto) con módulos SDIN

El ejemplo plantea un esquema de dispositivos conectados por una red punto-multipunto tipo estrella con un nodo central (circuito maestro) y 3 dispositivos esclavos.

La *Figura 11* presenta el esquema propuesto donde se aprecia la comunicación simultánea de 3 dispositivos hacia un nodo central. Se plantea el uso de esta aplicación con el uso del driver desarrollado para aplicaciones complejas con diversos sensores ubicados en sitios remotos. Cabe resaltar que los dispositivos LoRa deben ser de la misma referencia, estar configurados a la misma frecuencia y con los parámetros SF, BW y CR idénticos.
El dispositivo maestro se debe configurar como dispositivo "pasivo", es decir continuamente recibir la información de los dispositivos esclavos. Una vez recibe la información de un dispositivo esclavo, remite la confirmación de recepción correcta de datos.

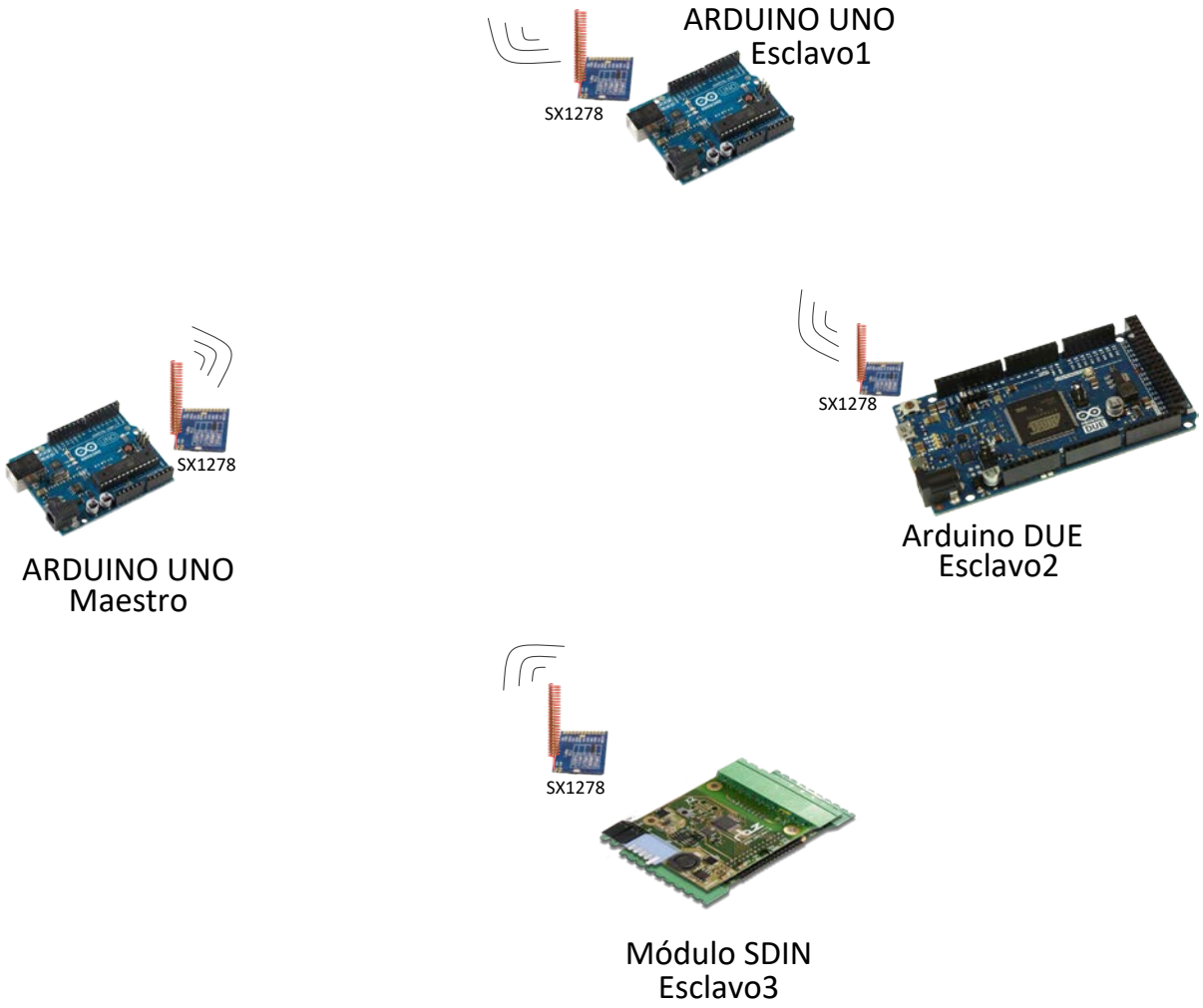


Figura 11. Tipología estrella para comunicación con diversos dispositivos
Fuente: Esquema propuesto por el autor

El modem LoRa es *half dúplex*, es decir que los LoRa no puede trasmitir o recibir datos en el mismo instante; esto significa que los transceiver LoRa deben ser configurados para trasmitir o recibir datos. Cuando se incorpora más de dos dispositivos en un protocolo de comunicación *half dúplex*, se deben establecer estrategias que mitiguen los efectos adversos por colisión indeseada de datos cuando se reciben datos.
Si los dispositivos no cuentan con estrategias para evitar la trasmisión de datos sobre el mismo canal de comunicación por parte de los dispositivos esclavos en un mismo instante, los datos llegaran errados al dispositivo maestro y la comunicación de datos será errada y ocasiona un consumo energético innecesario de energía en la transmisión de información por parte de los esclavos.

La solución para evitar la colisión de datos, corresponde a establecer tiempos de muestreos y tiempos aleatorios de “des-sincronización” cuando se trasmiten datos; es decir evitar con los tiempos adiciones de trasmisión de datos la colisión de datos.

En el documento Anexo A: "Desarrollo del driver para el uso de los dispositivos de radio LoRa con sistemas embebidos" – apartado "*Ejemplo de aplicación: recepción de datos de varios dispositivos remotos (red punto – multipunto)*", se describe de manera detallada el proceso de desarrollo de la prueba desarrollada para crear la red punto-multipunto con dispositivos LoRa.

La *Figura 10* presenta el esquema de conexión utilizado entre los dispositivos LoRa SX1278 y el sistema embebido Arduino Uno. La *Tabla 6* presenta la nomenclatura utilizada en el transceiver SX1278 y la tarjeta SDIN.
El código fuente utilizado por el dispositivo esclavo se presenta en la carpeta *lora/source/application.c* que acompaña al driver.

La tarea se denomina *TestLoraMultiple_Task()* corresponde al código fuente del dispositivo esclavo del sistema embebido SDIN.

BIBLIOGRAFÍA

NiceRF (2015). LoRa1278 100 mW 4 km larga distancia y alta sensibilidad (-139 dBm) 433 MHz módulo de transceptor inalámbrico. Recuperado de http://es.aliexpress.com/item/2pcs-lot-LoRa1278-100mW-4km-Long-Distance-and-High-Sensitivity-139-dBm-433MHz-Wireless-Transceiver-Module/32461365864.html, China. Consultado el 22 de junio de 2016

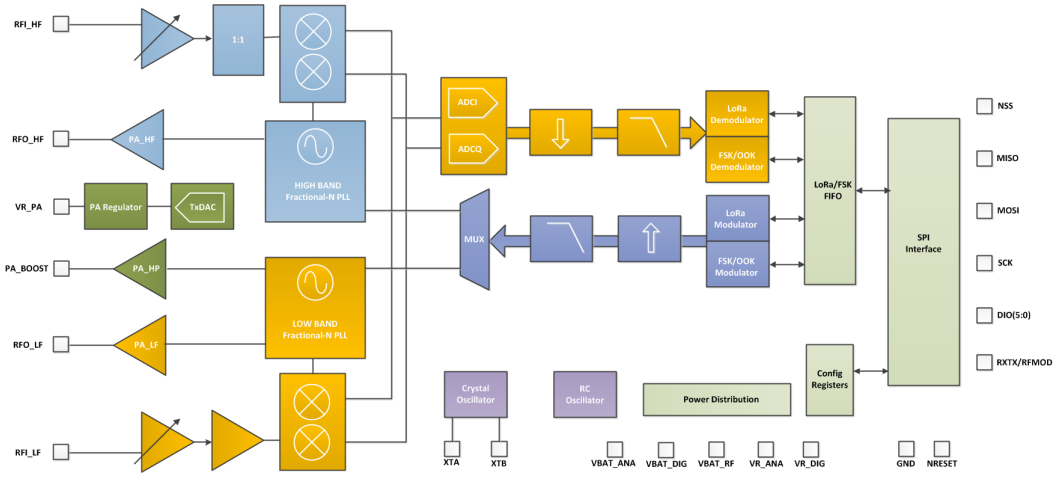
Semtech SX1272 (2015). SX1272/73 - 860 MHz to 1020 MHz Low Power Long Range Transceiver. Datasheet. Semtech Corporation. Estados Unidos.

Semtech SX1278 (2015). SX1276/77/78/79 - 137 MHz to 1020 MHz Low Power Long Range Transceiver. Datasheet. Semtech Corporation. Estados Unidos.

Texas Instruments (2005). SM-Band and Short Range Device Regulatory Compliance Overview. Estados Unidos.

WIRELESS, SENSING & TIMING DATASHEET

SX1276/77/78/79 - 137 MHz to 1020 MHz Low Power Long Range Transceiver



GENERAL DESCRIPTION

The SX1276/77/78/79 transceivers feature the LoRa™ long range modem that provides ultra-long range spread spectrum communication and high interference immunity whilst minimising current consumption.

Using Semtech's patented LoRa™ modulation technique SX1276/77/78/79 can achieve a sensitivity of over -148dBm using a low cost crystal and bill of materials. The high sensitivity combined with the integrated +20 dBm power amplifier yields industry leading link budget making it optimal for any application requiring range or robustness. LoRa™ also provides significant advantages in both blocking and selectivity over conventional modulation techniques, solving the traditional design compromise between range, interference immunity and energy consumption.

These devices also support high performance (G)FSK modes for systems including WMBus, IEEE802.15.4g. The SX1276/77/78/79 deliver exceptional phase noise, selectivity, receiver linearity and IIP3 for significantly lower current consumption than competing devices.

ORDERING INFORMATION

Part Number	Delivery	MOQ / Multiple
SX1276IMLTRT	T&R	3000 pieces
SX1277IMLTRT	T&R	3000 pieces
SX1278IMLTRT	T&R	3000 pieces
SX1279IMLTRT	T&R	3000 pieces

- QFN 28 Package - Operating Range [-40;+85°C]
- Pb-free, Halogen free, RoHS/WEEE compliant product

KEY PRODUCT FEATURES

- LoRa™ Modem
- 168 dB maximum link budget
- +20 dBm - 100 mW constant RF output vs. V supply
- +14 dBm high efficiency PA
- Programmable bit rate up to 300 kbps
- High sensitivity: down to -148 dBm
- Bullet-proof front end: IIP3 = -11 dBm
- Excellent blocking immunity
- Low RX current of 9.9 mA, 200 nA register retention
- Fully integrated synthesizer with a resolution of 61 Hz
- FSK, GFSK, MSK, GMSK, LoRa™and OOK modulation
- Built-in bit synchronizer for clock recovery
- Preamble detection
- 127 dB Dynamic Range RSSI
- Automatic RF Sense and CAD with ultra-fast AFC
- Packet engine up to 256 bytes with CRC
- Built-in temperature sensor and low battery indicator

APPLICATIONS

- Automated Meter Reading.
- Home and Building Automation.
- Wireless Alarm and Security Systems.
- Industrial Monitoring and Control
- Long range Irrigation Systems

Table of contents

Section **Page**

1. General Description 9
1.1. Simplified Block Diagram 9
1.2. Product Versions 10
1.3. Pin Diagram 10
1.4. Pin Description 11
1.5. Package Marking 12
2. Electrical Characteristics 13
2.1. ESD Notice 13
2.2. Absolute Maximum Ratings 13
2.3. Operating Range 13
2.4. Thermal Properties 13
2.5. Chip Specification 14
2.5.1. Power Consumption 14
2.5.2. Frequency Synthesis 14
2.5.3. FSK/OOK Mode Receiver 16
2.5.4. FSK/OOK Mode Transmitter 17
2.5.5. Electrical Specification for LoRaTM Modulation 19
2.5.6. Digital Specification 22
3. SX1276/77/78/79 Features 23
3.1. LoRaTM Modem 24
3.2. FSK/OOK Modem 24
4. SX1276/77/78/79 Digital Electronics 25
4.1. The LoRaTM Modem 25
4.1.1. Link Design Using the LoRaTM Modem 26
4.1.2. LoRaTM Digital Interface 34
4.1.3. Operation of the LoRaTM Modem 36
4.1.4. Frequency Settings 37
4.1.5. Frequency Error Indication 37
4.1.6. LoRaTM Modem State Machine Sequences 38
4.1.7. Modem Status Indicators 46
4.2. FSK/OOK Modem 46
4.2.1. Bit Rate Setting 46
4.2.2. FSK/OOK Transmission 47
4.2.3. FSK/OOK Reception 48
4.2.4. Operating Modes in FSK/OOK Mode 54
4.2.5. Startup Times 56
4.2.6. Receiver Startup Options 59
4.2.7. Receiver Restart Methods 60
4.2.8. Top Level Sequencer 61

Table of contents

Section **Page**

4.2.9. Data Processing in FSK/OOK Mode 65
4.2.10. FIFO 66
4.2.11. Digital IO Pins Mapping 69
4.2.12. Continuous Mode 70
4.2.13. Packet Mode 71
4.2.14. io-homecontrol® Compatibility Mode 79
4.3. SPI Interface 80
5. SX1276/77/78/79 Analog & RF Frontend Electronics 81
5.1. Power Supply Strategy 81
5.2. Low Battery Detector 81
5.3. Frequency Synthesis 81
5.3.1. Crystal Oscillator 81
5.3.2. CLKOUT Output 82
5.3.3. PLL 82
5.3.4. RC Oscillator 82
5.4. Transmitter Description 83
5.4.1. Architecture Description 83
5.4.2. RF Power Amplifiers 83
5.4.3. High Power +20 dBm Operation 84
5.4.4. Over Current Protection 85
5.5. Receiver Description 85
5.5.1. Overview 85
5.5.2. Receiver Enabled and Receiver Active States 85
5.5.3. Automatic Gain Control In FSK/OOK Mode 85
5.5.4. RSSI in FSK/OOK Mode 86
5.5.5. RSSI and SNR in LoRaTM Mode 87
5.5.6. Channel Filter 88
5.5.7. Temperature Measurement 89
6. Description of the Registers 90
6.1. Register Table Summary 90
6.2. FSK/OOK Mode Register Map 93
6.3. Band Specific Additional Registers 106
6.4. LoRaTM Mode Register Map 108
7. Application Information 115
7.1. Crystal Resonator Specification 115
7.2. Reset of the Chip 115
7.2.1. POR 115
7.2.2. Manual Reset 116
7.3. Top Sequencer: Listen Mode Examples 116

Table of contents

Section **Page**

7.3.1. Wake on Preamble Interrupt 116
7.3.2. Wake on SyncAddress Interrupt 119
7.4. Top Sequencer: Beacon Mode 122
7.4.1. Timing diagram 122
7.4.2. Sequencer Configuration 122
7.5. Example CRC Calculation 124
7.6. Example Temperature Reading 125
8. Packaging Information 127
8.1. Package Outline Drawing 127
8.2. Recommended Land Pattern 128
8.3. Tape & Reel Information 129
9. Revision History 130

Table of contents

Section **Page**

Table 1. SX1276/77/78/79 Device Variants and Key Parameters 10
Table 2. Pin Description 11
Table 3. Absolute Maximum Ratings 13
Table 4. Operating Range 13
Table 5. Thermal Properties 13
Table 6. Power Consumption Specification 14
Table 7. Frequency Synthesizer Specification 14
Table 8. FSK/OOK Receiver Specification 16
Table 9. Transmitter Specification 17
Table 10. LoRa Receiver Specification 19
Table 11. Digital Specification 22
Table 12. Example LoRaTM Modem Performances, 868MHz Band 25
Table 13. Range of Spreading Factors 27
Table 14. Cyclic Coding Overhead 27
Table 15. LoRa Bandwidth Options 28
Table 16. LoRaTM Operating Mode Functionality 36
Table 17. LoRa CAD Consumption Figures 45
Table 18. DIO Mapping LoRaTM Mode 46
Table 19. Bit Rate Examples 47
Table 20. Preamble Detector Settings 53
Table 21. RxTrigger Settings to Enable Timeout Interrupts 54
Table 22. Basic Transceiver Modes 54
Table 23. Receiver Startup Time Summary 57
Table 24. Receiver Startup Options 60
Table 25. Sequencer States 61
Table 26. Sequencer Transition Options 62
Table 27. Sequencer Timer Settings 63
Table 28. Status of FIFO when Switching Between Different Modes of the Chip 67
Table 29. DIO Mapping, Continuous Mode 69
Table 30. DIO Mapping, Packet Mode 69
Table 31. CRC Description 77
Table 32. Frequency Bands 82
Table 33. Power Amplifier Mode Selection Truth Table 83
Table 34. High Power Settings 84
Table 35. Operating Range, +20dBm Operation 84
Table 36. Operating Range, +20dBm Operation 84
Table 37. Trimming of the OCP Current 85
Table 38. LNA Gain Control and Performances 86
Table 39. RssiSmoothing Options 86

Table of contents

Section **Page**

Table 40. Available RxBw Settings 88
Table 41. Registers Summary 90
Table 42. Register Map 93
Table 43. Low Frequency Additional Registers 106
Table 44. High Frequency Additional Registers 107
Table 45. Crystal Specification 115
Table 46. Listen Mode with PreambleDetect Condition Settings 118
Table 47. Listen Mode with PreambleDetect Condition Recommended DIO Mapping 118
Table 48. Listen Mode with SyncAddress Condition Settings 121
Table 49. Listen Mode with PreambleDetect Condition Recommended DIO Mapping 121
Table 50. Beacon Mode Settings 123
Table 51. Revision History 130

Table of contents

Section **Page**

Figure 1. Block Diagram 9
Figure 2. Pin Diagrams 10
Figure 3. Marking Diagram 12
Figure 4. SX1276/77/78/79 Block Schematic Diagram 23
Figure 5. LoRaTM Modem Connectivity 26
Figure 6. LoRaTM Packet Structure 29
Figure 7. Interrupts Generated in the Case of Successful Frequency Hopping Communication. 33
Figure 8. LoRaTM Data Buffer 34
Figure 9. LoRaTM Modulation Transmission Sequence. 38
Figure 10. LoRaTM Receive Sequence. 39
Figure 11. LoRaTM CAD Flow 43
Figure 12. CAD Time as a Function of Spreading Factor 44
Figure 13. Consumption Profile of the LoRa CAD Process 45
Figure 14. OOK Peak Demodulator Description 49
Figure 15. Floor Threshold Optimization 50
Figure 16. Bit Synchronizer Description 51
Figure 17. Startup Process 56
Figure 18. Time to RSSI Sample 57
Figure 19. Tx to Rx Turnaround 58
Figure 20. Rx to Tx Turnaround 58
Figure 21. Receiver Hopping 59
Figure 22. Transmitter Hopping 59
Figure 23. Timer1 and Timer2 Mechanism 63
Figure 24. Sequencer State Machine 64
Figure 25. SX1276/77/78/79 Data Processing Conceptual View 65
Figure 26. FIFO and Shift Register (SR) 66
Figure 27. FifoLevel IRQ Source Behavior 67
Figure 28. Sync Word Recognition 68
Figure 29. Continuous Mode Conceptual View 70
Figure 30. Tx Processing in Continuous Mode 70
Figure 31. Rx Processing in Continuous Mode 71
Figure 32. Packet Mode Conceptual View 72
Figure 33. Fixed Length Packet Format 73
Figure 34. Variable Length Packet Format 74
Figure 35. Unlimited Length Packet Format 74
Figure 36. Manchester Encoding/Decoding 78
Figure 37. Data Whitening Polynomial 79
Figure 38. SPI Timing Diagram (single access) 80
Figure 39. TCXO Connection 81

Table of contents

Section **Page**

Figure 40. RF Front-end Architecture Shows the Internal PA Configuration. 83
Figure 41. Temperature Sensor Response 89
Figure 42. POR Timing Diagram 115
Figure 43. Manual Reset Timing Diagram 116
Figure 44. Listen Mode: Principle 116
Figure 45. Listen Mode with No Preamble Received 117
Figure 46. Listen Mode with Preamble Received 117
Figure 47. Wake On PreambleDetect State Machine 118
Figure 48. Listen Mode with no SyncAddress Detected 119
Figure 49. Listen Mode with Preamble Received and no SyncAddress 119
Figure 50. Listen Mode with Preamble Received & Valid SyncAddress 120
Figure 51. Wake On SyncAddress State Machine 120
Figure 52. Beacon Mode Timing Diagram 122
Figure 53. Beacon Mode State Machine 122
Figure 54. Example CRC Code 124
Figure 55. Example Temperature Reading 125
Figure 56. Example Temperature Reading (continued) 126
Figure 57. Package Outline Drawing 127
Figure 58. Recommended Land Pattern 128
Figure 59. Tape and Reel Information 129

1. General Description

The SX1276/77/78/79 incorporates the LoRa™ spread spectrum modem which is capable of achieving significantly longer range than existing systems based on FSK or OOK modulation. At maximum data rates of LoRa™ the sensitivity is 8dB better than FSK, but using a low cost bill of materials with a 20ppm XTAL LoRa™ can improve receiver sensitivity by more than 20dB compared to FSK. LoRa™ also provides significant advances in selectivity and blocking performance, further improving communication reliability. For maximum flexibility the user may decide on the spread spectrum modulation bandwidth (BW), spreading factor (SF) and error correction rate (CR). Another benefit of the spread modulation is that each spreading factor is orthogonal - thus multiple transmitted signals can occupy the same channel without interfering. This also permits simple coexistence with existing FSK based systems. Standard GFSK, FSK, OOK, and GMSK modulation is also provided to allow compatibility with existing systems or standards such as wireless MBUS and IEEE 802.15.4g.

The SX1276 and SX1279 offer bandwidth options ranging from 7.8 kHz to 500 kHz with spreading factors ranging from 6 to 12, and covering all available frequency bands. The SX1277 offers the same bandwidth and frequency band options with spreading factors from 6 to 9. The SX1278 offers bandwidths and spreading factor options, but only covers the lower UHF bands.

1.1. Simplified Block Diagram

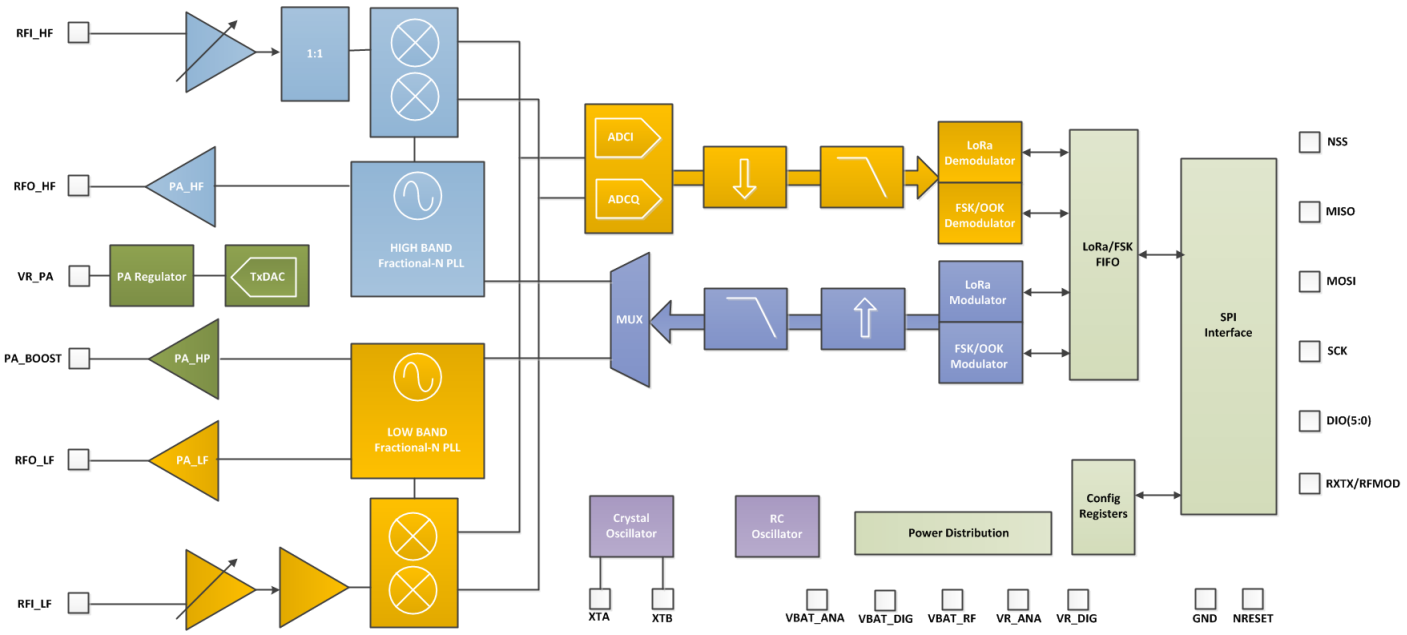


Figure 1. Block Diagram

1.2. Product Versions

The features of the four product variants are detailed in the following table.

Table 1 SX1276/77/78/79 Device Variants and Key Parameters

Part Number	Frequency Range	Spreading Factor	Bandwidth	Effective Bitrate	Est. Sensitivity
SX1276	137 - 1020 MHz	6 - 12	7.8 - 500 kHz	.018 - 37.5 kbps	-111 to -148 dBm
SX1277	137 - 1020 MHz	6 - 9	7.8 - 500 kHz	0.11 - 37.5 kbps	-111 to -139 dBm
SX1278	137 - 525 MHz	6- 12	7.8 - 500 kHz	.018 - 37.5 kbps	-111 to -148 dBm
SX1279	137 - 960MHz	6- 12	7.8 - 500 kHz	.018 - 37.5 kbps	-111 to -148 dBm

1.3. Pin Diagram

The following diagram shows the pin arrangement of the QFN package, top view.

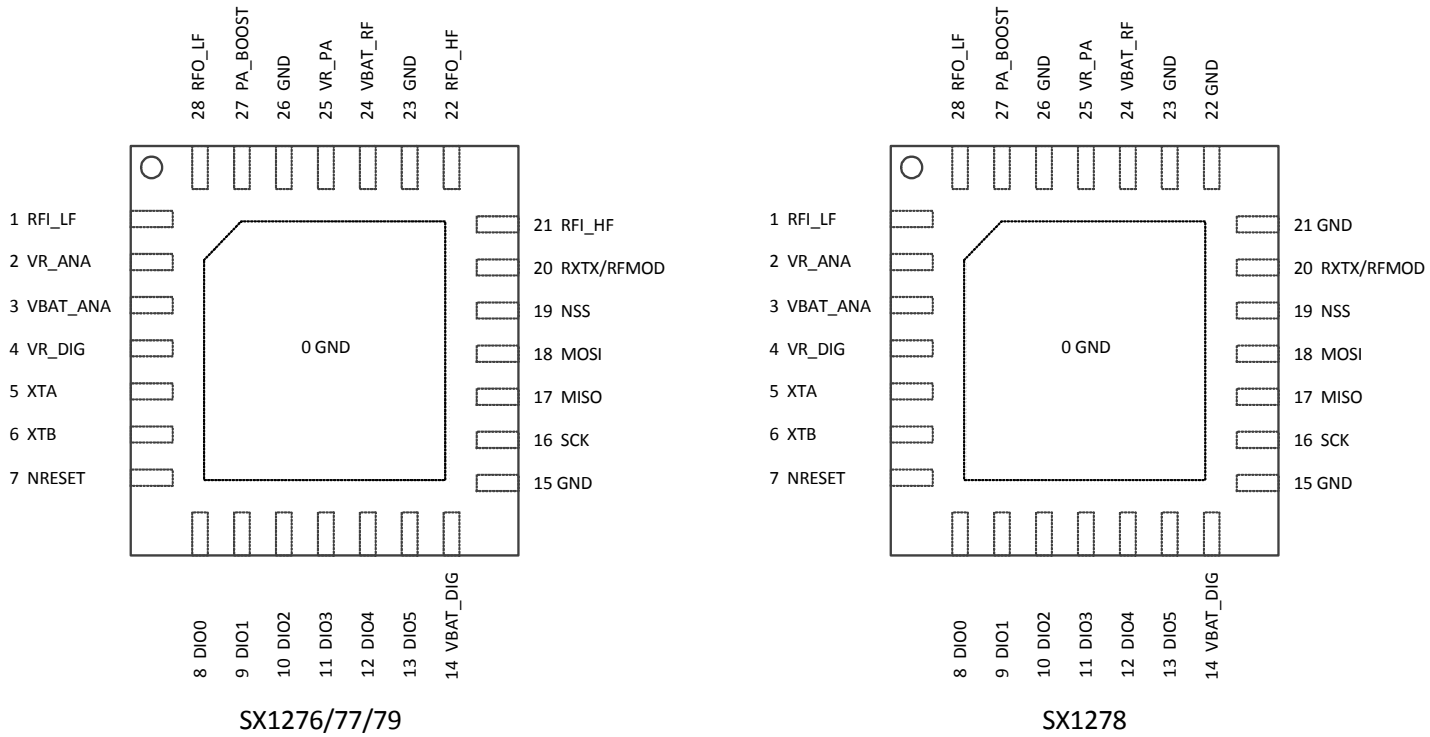


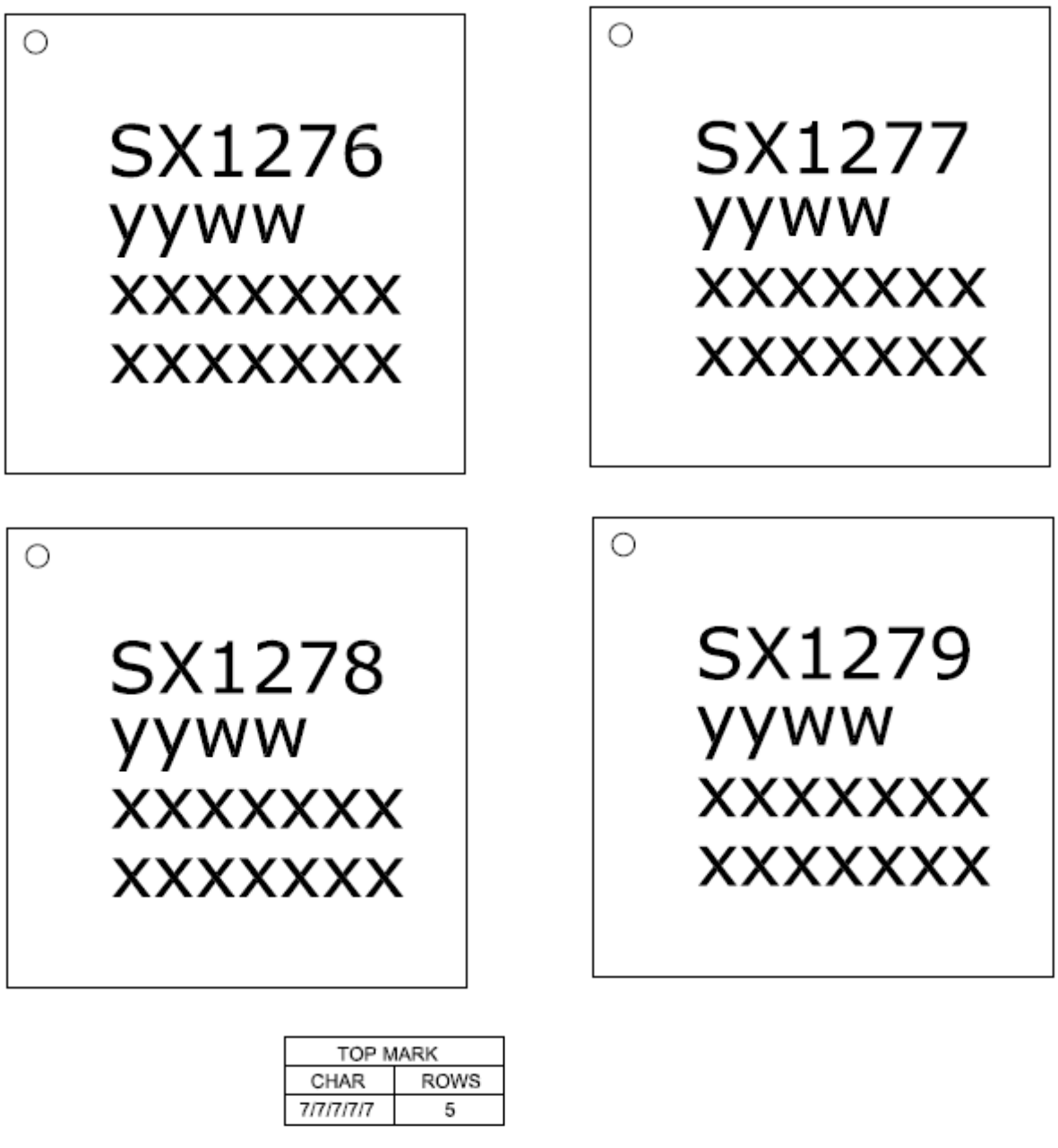
Figure 2. Pin Diagrams

1.4. Pin Description

Table 2 Pin Description

Number	Name	Type	Description
	SX1276/77/79/(78)	SX1276/77/79/(78)	SX1276/77/79/(78)
0	GROUND	-	Exposed ground pad
1	RFI_LF	I	RF input for bands 2&3
2	VR_ANA	-	Regulated supply voltage for analogue circuitry
3	VBAT_ANA	-	Supply voltage for analogue circuitry
4	VR_DIG	-	Regulated supply voltage for digital blocks
5	XTA	I/O	XTAL connection or TCXO input
6	XTB	I/O	XTAL connection
7	NRESET	I/O	Reset trigger input
8	DIO0	I/O	Digital I/O, software configured
9	DIO1/DCLK	I/O	Digital I/O, software configured
10	DIO2/DATA	I/O	Digital I/O, software configured
11	DIO3	I/O	Digital I/O, software configured
12	DIO4	I/O	Digital I/O, software configured
13	DIO5	I/O	Digital I/O, software configured
14	VBAT_DIG	-	Supply voltage for digital blocks
15	GND	-	Ground
16	SCK	I	SPI Clock input
17	MISO	O	SPI Data output
18	MOSI	I	SPI Data input
19	NSS	I	SPI Chip select input
20	RXTX/RF_MOD	O	Rx/Tx switch control: high in Tx
21	RFI_HF (GND)	I (-)	RF input for band 1 (Ground)
22	RFO_HF (GND)	O (-)	RF output for band 1 (Ground)
23	GND	-	Ground
24	VBAT_RF	-	Supply voltage for RF blocks
25	VR_PA	-	Regulated supply for the PA
26	GND	-	Ground
27	PA_BOOST	O	Optional high-power PA output, all frequency bands
28	RFO_LF	O	RF output for bands 2&3

1.5. Package Marking



TOP MARK	
CHAR	ROWS
7/7/7/7/7	5

Marking for the 6 x 6 mm MLPQ 28ld Lead package:

nnnnnn = Part Number (Example: SX1276)
yyww = Date Code (Example: 1352)
xxxxxxx = Semtech Lot No. (Example: EA90101)
xxxxxxx 0101-10)

Figure 3. Marking Diagram

2. Electrical Characteristics

2.1. ESD Notice

The SX1276/77/78/79 is a high performance radio frequency device. It satisfies:

- Class 2 of the JEDEC standard JESD22-A114 (Human Body Model) on all pins.
- Class III of the JEDEC standard JESD22-C101 (Charged Device Model) on all pins

It should thus be handled with all the necessary ESD precautions to avoid any permanent damage.

2.2. Absolute Maximum Ratings

Stresses above the values listed below may cause permanent device failure. Exposure to absolute maximum ratings for extended periods may affect device reliability.

Table 3 Absolute Maximum Ratings

Symbol	Description	Min	Max	Unit
VDDmr	Supply Voltage	-0.5	3.9	V
Tmr	Temperature	-55	+115	° C
Tj	Junction temperature	-	+125	° C
Pmr	RF Input Level	-	+10	dBm

Note Specific ratings apply to +20 dBm operation (see Section 5.4.3).

2.3. Operating Range

Table 4 Operating Range

Symbol	Description	Min	Max	Unit
VDDop	Supply voltage	1.8	3.7	V
Top	Operational temperature range	-40	+85	°C
Clop	Load capacitance on digital ports	-	25	pF
ML	RF Input Level	-	+10	dBm

Note A specific supply voltage range applies to +20 dBm operation (see Section 5.4.3).

2.4. Thermal Properties

Table 5 Thermal Properties

Symbol	Description	Min	Typ	Max	Unit
THETA_JA	Package θ_{ja} (Junction to ambient)	-	22.185	-	°C/W
THETA_JC	Package θ_{jc} (Junction to case ground paddle)	-	0.757	-	°C/W

2.5. Chip Specification

The tables below give the electrical specifications of the transceiver under the following conditions: Supply voltage VDD=3.3 V, temperature = 25 °C, *FXOSC* = 32 MHz, F_{RF} = 169/434/868/915 MHz (see specific indication), Pout = +13dBm, 2-level FSK modulation without pre-filtering, FDA = 5 kHz, Bit Rate = 4.8 kb/s and terminated in a matched 50 Ohm impedance, shared Rx and Tx path matching, unless otherwise specified.

Note *Specification whose symbol is appended with "_LF" corresponds to the performance in Band 2 and/or Band 3, as described in section 5.3.3. "_HF" refers to the upper Band 1*

2.5.1. Power Consumption

Table 6 Power Consumption Specification

Symbol	Description	Conditions	Min	Typ	Max	Unit
IDDSL	Supply current in Sleep mode		-	0.2	1	uA
IDDIDLE	Supply current in Idle mode	RC oscillator enabled	-	1.5	-	uA
IDDST	Supply current in Standby mode	Crystal oscillator enabled	-	1.6	1.8	mA
IDDFS	Supply current in Synthesizer mode	FSRx	-	5.8	-	mA
IDDR	Supply current in Receive mode	*LnaBoost* Off, band 1 *LnaBoost* On, band 1 Bands 2&3	- - -	10.8 11.5 12.0	- - -	mA
IDDT	Supply current in Transmit mode with impedance matching	RFOP = +20 dBm, on PA_BOOST RFOP = +17 dBm, on PA_BOOST RFOP = +13 dBm, on RFO_LF/HF pin RFOP = + 7 dBm, on RFO_LF/HF pin	- - - -	120 87 29 20	- - - -	mA mA mA mA

2.5.2. Frequency Synthesis

Table 7 Frequency Synthesizer Specification

Symbol	Description	Conditions		Min	Typ	Max	Unit
FR	Synthesizer frequency range	Programmable (*for SX1279)	Band 3 Band 2 Band 1	137 410 862 (*779)	- - -	175 (*160) 525 (*480) 1020 (*960)	MHz
FXOSC	Crystal oscillator frequency			-	32	-	MHz
TS_OSC	Crystal oscillator wake-up time			-	250	-	us
TS_FS	Frequency synthesizer wake-up time to PllLock signal	From Standby mode		-	60	-	us

TS_HOP	Frequency synthesizer hop time at most 10 kHz away from the target frequency	200 kHz step 1 MHz step 5 MHz step 7 MHz step 12 MHz step 20 MHz step 25 MHz step	- - - - - - -	20 20 50 50 50 50 50	- - - - - - -	us us us us us us us
FSTEP	Frequency synthesizer step	FSTEP = FXOSC/2^{19}	-	61.0	-	Hz
FRC	RC Oscillator frequency	After calibration	-	62.5	-	kHz
BRF	Bit rate, FSK	Programmable values (1)	1.2	-	300	kbps
BRA	Bit rate Accuracy, FSK	ABS(wanted BR - available BR)	-	-	250	ppm
BRO	Bit rate, OOK	Programmable	1.2	-	32.768	kbps
BR_L	Bit rate, LoRa Mode	From SF6, BW=500kHz to SF12, BW=7.8kHz	0.018	-	37.5	kbps
FDA	Frequency deviation, FSK (1)	Programmable FDA + BRF/2 =< 250 kHz	0.6	-	200	kHz

Note: For Maximum Bit rate, the maximum modulation index is 0.5.

2.5.3. FSK/OOK Mode Receiver

All receiver tests are performed with RxBw = 10 kHz (Single Side Bandwidth) as programmed in *RegRxBw*, receiving a PN15 sequence. Sensitivities are reported for a 0.1% BER (with Bit Synchronizer enabled), unless otherwise specified. Blocking tests are performed with an unmodulated interferer. The wanted signal power for the Blocking Immunity, ACR, IIP2, IIP3 and AMR tests is set 3 dB above the receiver sensitivity level.

Table 8 FSK/OOK Receiver Specification

Symbol	Description	Conditions	Min	Typ	Max	Unit
RFS_F_LF	Direct tie of RFI and RFO pins, shared Rx, Tx paths FSK sensitivity, highest LNA gain. Bands 2&3	FDA = 5 kHz, BR = 1.2 kb/s FDA = 5 kHz, BR = 4.8 kb/s FDA = 40 kHz, BR = 38.4 kb/s* FDA = 20 kHz, BR = 38.4 kb/s** FDA = 62.5 kHz, BR = 250 kb/s***	- - - - -	-121 -117 -107 -108 -95	- - - - -	dBm dBm dBm dBm dBm
	Split RF paths, the RF switch insertion loss is not accounted for. Bands 2&3	FDA = 5 kHz, BR = 1.2 kb/s FDA = 5 kHz, BR = 4.8 kb/s FDA = 40 kHz, BR = 38.4 kb/s* FDA = 20 kHz, BR = 38.4 kb/s** FDA = 62.5 kHz, BR = 250 kb/s***	- - - - -	-123 -119 -109 -110 -97	- - - - -	dBm dBm dBm dBm dBm
RFS_F_HF	Direct tie of RFI and RFO pins, shared Rx, Tx paths FSK sensitivity, highest LNA gain. Band 1	FDA = 5 kHz, BR = 1.2 kb/s FDA = 5 kHz, BR = 4.8 kb/s FDA = 40 kHz, BR = 38.4 kb/s* FDA = 20 kHz, BR = 38.4 kb/s** FDA = 62.5 kHz, BR = 250 kb/s***	- - - - -	-119 -115 -105 -105 -92	- - - - -	dBm dBm dBm dBm dBm
	Split RF paths, *LnaBoost* is turned on, the RF switch insertion loss is not accounted for. Band 1	FDA = 5 kHz, BR = 1.2 kb/s FDA = 5 kHz, BR = 4.8 kb/s FDA = 40 kHz, BR = 38.4 kb/s* FDA = 20 kHz, BR = 38.4 kb/s** FDA = 62.5 kHz, BR = 250 kb/s***	- - - - -	-123 -119 -109 -109 -96	- - - - -	dBm dBm dBm dBm dBm
RFS_O	OOK sensitivity, highest LNA gain shared Rx, Tx paths	BR = 4.8 kb/s BR = 32 kb/s	- -	-117 -108	- -	dBm dBm
CCR	Co-Channel Rejection, FSK		-	-9	-	dB
ACR	Adjacent Channel Rejection	FDA = 5 kHz, BR=4.8kb/s Offset = +/- 25 kHz or +/- 50kHz Band 1 Band 2 Band 3	 - - -	 50 56 60	 - - -	 dB dB dB
BI_HF	Blocking Immunity, Band 1	Offset = +/- 1 MHz Offset = +/- 2 MHz Offset = +/- 10 MHz	- - -	71 76 84	- - -	dB dB dB
BI_LF	Blocking Immunity, Bands 2&3	Offset = +/- 1 MHz Offset = +/- 2 MHz Offset = +/- 10 MHz	- - -	71 72 78	- - -	dB dB dB

IIP2	2nd order Input Intercept Point Unwanted tones are 20 MHz above the LO	Highest LNA gain	-	+55	-	dBm
IIP3_HF	3rd order Input Intercept point Unwanted tones are 1MHz and 1.995 MHz above the LO	Band 1 Highest LNA gain G1 LNA gain G2, 5dB sensitivity hit	- -	-11 -6	- -	dBm dBm
IIP3_LF	3rd order Input Intercept point Unwanted tones are 1MHz and 1.995 MHz above the LO	Band 2 Highest LNA gain G1 LNA gain G2, 2.5dB sensitivity hit	- -	-22 -15	- -	dBm dBm
		Band 3 Highest LNA gain G1 LNA gain G2, 2.5dB sensitivity hit	- -	-15 -11	- -	dBm dBm
BW_SSB	Single Side channel filter BW	Programmable	2.7	-	250	kHz
IMR	Image Rejection	Wanted signal 3dB over sensitivity BER=0.1%	-	50	-	dB
IMA	Image Attenuation		-	57	-	dB
DR_RSSI	RSSI Dynamic Range	AGC enabled Min Max	- -	-127 0	- -	dBm dBm

* *RxBw = 83 kHz (Single Side Bandwidth)*

** *RxBw = 50 kHz (Single Side Bandwidth)*

*** *RxBw = 250 kHz (Single Side Bandwidth)*

2.5.4. FSK/OOK Mode Transmitter

Table 9 Transmitter Specification

Symbol	Description	Conditions	Min	Typ	Max	Unit
RF_OP	RF output power in 50 ohms on RFO pin (High efficiency PA).	Programmable with steps Max Min	 - -	 +14 -1	 - -	 dBm dBm
ΔRF_OP_V	RF output power stability on RFO pin versus voltage supply.	VDD = 2.5 V to 3.3 V VDD = 1.8 V to 3.7 V	- -	3 8	- -	dB dB
RF_OPH	RF output power in 50 ohms, on PA_BOOST pin (Regulated PA).	Programmable with 1dB steps Max Min	- -	+17 +2	- -	dBm dBm
RF_OPH_MAX	Max RF output power, on PA_BOOST pin	High power mode	-	+20	-	dBm
ΔRF_OPH_V	RF output power stability on PA_-BOOST pin versus voltage supply.	VDD = 2.4 V to 3.7 V	-	+/-1	-	dB
ΔRF_T	RF output power stability versus temperature on PA_BOOST pin.	From T = -40 °C to +85 °C	-	+/-1	-	dB

PHN	Transmitter Phase Noise	169 MHz, Band 3 10kHz Offset 50kHz Offset 400kHz Offset 1MHz Offset	 - - - -	 -118 -118 -128 -134	 - - - -	dBc/Hz
		433 MHz, Band 2 10kHz Offset 50kHz Offset 400kHz Offset 1MHz Offset	 - - - -	 -110 -110 -122 -129	 - - - -	dBc/Hz
		868/915 MHz, Band 1 10kHz Offset 50kHz Offset 400kHz Offset 1MHz Offset	 - - - -	 -103 -103 -115 -122	 - - - -	dBc/Hz
ACP	Transmitter adjacent channel power (measured at 25 kHz offset)	BT=1. Measurement conditions as defined by EN 300 220-1 V2.3.1	-	-	-37	dBm
TS_TR	Transmitter wake up time, to the first rising edge of DCLK	Frequency Synthesizer enabled, *PaR-amp* = 10us, BR = 4.8 kb/s	-	120	-	us

2.5.5. Electrical Specification for LoRa™ Modulation

The table below gives the electrical specifications for the transceiver operating with LoRa™ modulation. Following conditions apply unless otherwise specified:

- Supply voltage = 3.3 V
- Temperature = 25° C
- f_{XOSC} = 32 MHz
- bandwidth (BW) = 125 kHz
- Spreading Factor (SF) = 12
- Error Correction Code (EC) = 4/6
- Packet Error Rate (PER)= 1%
- CRC on payload enabled
- Output power = 13 dBm in transmission
- Payload length = 64 bytes
- Preamble Length = 12 symbols (programmed register *PreambleLength=8*)
- With matched impedances

Table 10 LoRa Receiver Specification

Symbol	Description	Conditions	Min.	Typ	Max	Unit
IDDR_L	Supply current in receiver LoRa™ mode, *LnaBoost* off	Bands 2&3, BW=7.8 to 62.5 kHz Bands 2&3, BW = 125 kHz Bands 2&3, BW = 250 kHz Bands 2&3, BW = 500 kHz	- - - -	11.0 11.5 12.4 13.8	- - - -	mA mA mA mA
		Band 1, BW=7.8 to 62.5 kHz Band 1, BW = 125 kHz Band 1, BW = 250 kHz Band 1, BW = 500 kHz	- - - -	9.9 10.3 11.1 12.6	- - - -	mA mA mA mA
IDDT_L	Supply current in transmitter mode	RFOP = 13 dBm RFOP = 7 dBm	- -	28 20	- -	mA mA
IDDT_H_L	Supply current in transmitter mode with an external impedance transformation	Using PA_BOOST pin RFOP = 17 dBm	-	90	-	mA
BI_L	Blocking immunity, CW interferer	offset = +/- 1 MHz offset = +/- 2 MHz offset = +/- 10 MHz	-	89 94 100	-	dB dB dB
IIP2_L	2nd order Input Intercept Point Unwanted tones are 20 MHz above the LO	Highest LNA gain	-	+55	-	dBm
IIP3_L_HF	3rd order Input Intercept point Unwanted tones are 1MHz and 1.995 MHz above the LO	Band 1 Highest LNA gain G1 LNA gain G2, 5dB sensitivity hit	 - -	 -11 -6	 - -	 dBm dBm

Symbol	Description	Conditions	Min.	Typ	Max	Unit
IIP3_L_LF	3rd order Input Intercept point Unwanted tones are 1MHz and 1.995 MHz above the LO	Band 2 Highest LNA gain G1 LNA gain G2,2.5dB sensitivity hit	 - -	 -22 -15	 - -	 dBm dBm
RFS_L10_HF	RF sensitivity, Long-Range Mode, highest LNA gain, *LnaBoost* for Band 1, using split Rx/Tx path 10.4 kHz bandwidth	SF = 6 SF = 7 SF = 8 SF = 11	- - - -	-131 -134 -138 -146	- - - -	dBm dBm dBm dBm
RFS_L62_HF	RF sensitivity, Long-Range Mode, highest LNA gain, *LnaBoost* for Band 1, using split Rx/Tx path 62.5 kHz bandwidth	SF = 6 SF = 7 SF = 8 SF = 9 SF = 10 SF = 11 SF = 12	- - - - - - -	-121 -126 -129 -132 -135 -137 -139	- - - - - - -	dBm dBm dBm dBm dBm dBm dBm
RFS_L125_HF	RF sensitivity, Long-Range Mode, highest LNA gain, *LnaBoost* for Band 1, using split Rx/Tx path 125 kHz bandwidth	SF = 6 SF = 7 SF = 8 SF = 9 SF = 10 SF = 11 SF = 12	- - - - - - -	-118 -123 -126 -129 -132 -133 -136	- - - - - - -	dBm dBm dBm dBm dBm dBm dBm
RFS_L250_HF	RF sensitivity, Long-Range Mode, highest LNA gain, *LnaBoost* for Band 1, using split Rx/Tx path 250 kHz bandwidth	SF = 6 SF = 7 SF = 8 SF = 9 SF = 10 SF = 11 SF = 12	- - - - - - -	-115 -120 -123 -125 -128 -130 -133	- - - - - - -	dBm dBm dBm dBm dBm dBm dBm
RFS_L500_HF	RF sensitivity, Long-Range Mode, highest LNA gain, *LnaBoost* for Band 1, using split Rx/Tx path 500 kHz bandwidth	SF = 6 SF = 7 SF = 8 SF = 9 SF = 10 SF = 11 SF = 12	- - - - - - -	-111 -116 -119 -122 -125 -128 -130	- - - - - - -	dBm dBm dBm dBm dBm dBm dBm
RFS_L7.8_LF	RF sensitivity, Long-Range Mode, highest LNA gain, Band 2 or 3, using split Rx/Tx path 7.8 kHz bandwidth	SF = 12 SF = 11	- -	-148 -145	- -	dBm dBm
RFS_L10_LF	RF sensitivity, Long-Range Mode, highest LNA gain, Band 3, 10.4 kHz bandwidth	SF = 6 SF = 7 SF = 8	- - -	-132 -136 -138	- - -	dBm dBm dBm
RFS_L62_LF	RF sensitivity, Long-Range Mode, highest LNA gain, Band 3, 62.5 kHz bandwidth	SF = 6 SF = 7 SF = 8 SF = 9 SF = 10 SF = 11 SF = 12	- - - - - - -	-123 -128 -131 -134 -135 -137 -140	- - - - - - -	dBm dBm dBm dBm dBm dBm dBm

Symbol	Description	Conditions	Min.	Typ	Max	Unit
RFS_L125_LF	RF sensitivity, Long-Range Mode, highest LNA gain, Band 3, 125 kHz bandwidth	SF = 6 SF = 7 SF = 8 SF = 9 SF = 10 SF = 11 SF = 12	- - - - - - -	-121 -125 -128 -131 -134 -136 -137	- - - - - - -	dBm dBm dBm dBm dBm dBm dBm
RFS_L250_LF	RF sensitivity, Long-Range Mode, highest LNA gain, Band 3 250 kHz bandwidth	SF = 6 SF = 7 SF = 8 SF = 9 SF = 10 SF = 11 SF = 12	- - - - - - -	-118 -122 -125 -128 -131 -133 -134	- - - - - - -	dBm dBm dBm dBm dBm dBm dBm
RFS_L500_LF	RF sensitivity, Long-Range Mode, highest LNA gain, Band 3 500 kHz bandwidth	SF = 6 SF = 7 SF = 8 SF = 9 SF = 10 SF = 11 SF = 12	- - - - - - -	-112 -118 -121 -124 -127 -129 -130	- - - - - - -	dBm dBm dBm dBm dBm dBm dBm
CCR_LCW	Co-channel rejection Single CW tone = Sens +6 dB 1% PER	SF = 7 SF = 8 SF = 9 SF = 10 SF = 11 SF = 12	- - - - - -	5 9.5 12 14.4 17 19.5	- - - - - -	dB dB dB dB dB dB
CCR_LL	Co-channel rejection	Interferer is a LoRa™ signal using same BW and same SF. Pw = Sensitivity + 3 dB		-6		dB
ACR_LCW	Adjacent channel rejection	Interferer is 1.5*BW_L from the wanted signal center frequency 1% PER, Single CW tone = Sens + 3 dB SF = 7 SF = 12	 - -	 60 72	 - -	 dB dB
IMR_LCW	Image rejection after calibration.	1% PER, Single CW tone = Sens +3 dB	-	66	-	dB
FERR_L	Maximum tolerated frequency offset between transmitter and receiver, no sensitivity degradation, SF6 thru 12	All BW, +/-25% of BW The tighter limit applies (see below)		+/-25%		BW
	Maximum tolerated frequency offset between transmitter and receiver, no sensitivity degradation, SF10 thru 12	SF = 12 SF = 11 SF = 10	-50 -100 -200	- - -	50 100 200	ppm ppm ppm

2.5.6. Digital Specification

Conditions: Temp = 25° C, VDD = 3.3 V, FXOSC = 32 MHz, unless otherwise specified.

Table 11 Digital Specification

Symbol	Description	Conditions	Min	Typ	Max	Unit
V_{IH}	Digital input level high		0.8	-	-	VDD
V_{IL}	Digital input level low		-	-	0.2	VDD
V_{OH}	Digital output level high	Imax = 1 mA	0.9	-	-	VDD
V_{OL}	Digital output level low	Imax = -1 mA	-	-	0.1	VDD
F_{SCK}	SCK frequency		-	-	10	MHz
t_{ch}	SCK high time		50	-	-	ns
t_{cl}	SCK low time		50	-	-	ns
t_{rise}	SCK rise time		-	5	-	ns
t_{fall}	SCK fall time		-	5	-	ns
t_{setup}	MOSI setup time	From MOSI change to SCK rising edge.	30	-	-	ns
t_{hold}	MOSI hold time	From SCK rising edge to MOSI change.	20	-	-	ns
t_{nsetup}	NSS setup time	From NSS falling edge to SCK rising edge.	30	-	-	ns
t_{nhold}	NSS hold time	From SCK falling edge to NSS rising edge, normal mode.	100	-	-	ns
t_{nhigh}	NSS high time between SPI accesses		20	-	-	ns
T_DATA	DATA hold and setup time		250	-	-	ns

3. SX1276/77/78/79 Features

This section gives a high-level overview of the functionality of the SX1276/77/78/79 low-power, highly integrated transceiver. The following figure shows a simplified block diagram of the SX1276/77/78/79.

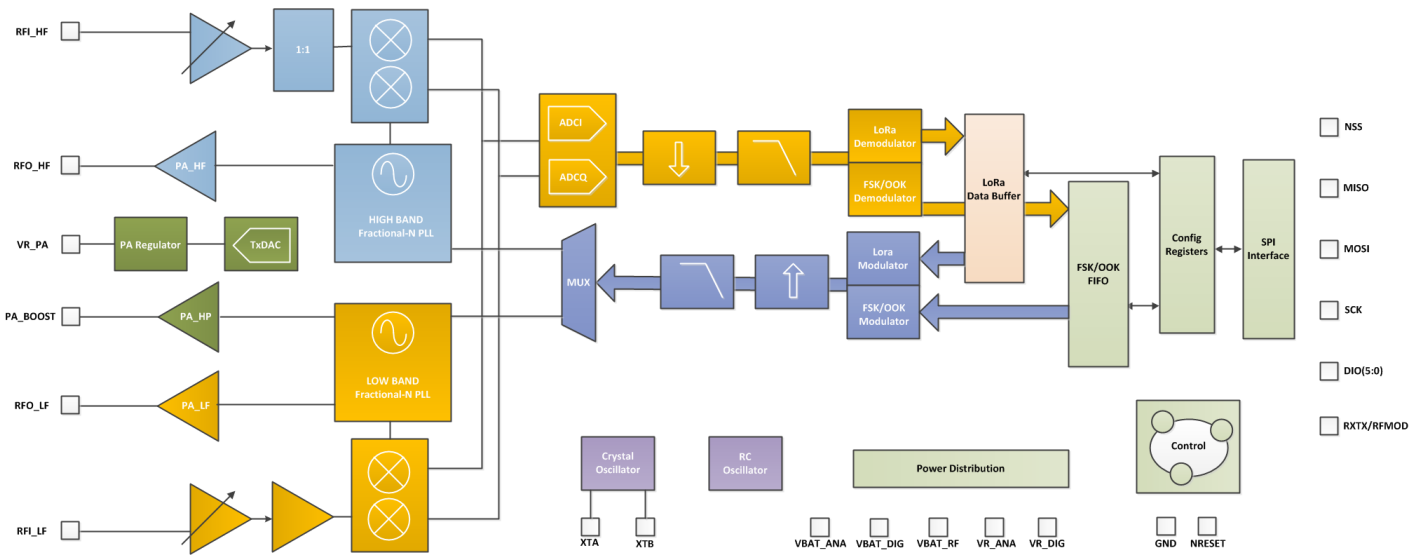


Figure 4. SX1276/77/78/79 Block Schematic Diagram

SX1276/77/78/79 is a half-duplex, low-IF transceiver. Here the received RF signal is first amplified by the LNA. The LNA inputs are single ended to minimize the external BoM and for ease of design. Following the LNA inputs, the conversion to differential is made to improve the second order linearity and harmonic rejection. The signal is then down-converted to in-phase and quadrature (I&Q) components at the intermediate frequency (IF) by the mixer stage. A pair of sigma delta ADCs then perform data conversion, with all subsequent signal processing and demodulation performed in the digital domain. The digital state machine also controls the automatic frequency correction (AFC), received signal strength indicator (RSSI) and automatic gain control (AGC). It also features the higher-level packet and protocol level functionality of the top level sequencer (TLS), only available with traditional FSK and OOK modulation schemes.

The frequency synthesizers generate the local oscillator (LO) frequency for both receiver and transmitter, one covering the lower UHF bands (up to 525 MHz), and the other one covering the upper UHF bands (from 779 MHz). The PLLs are optimized for user-transparent low lock time and fast auto-calibrating operation. In transmission, frequency modulation is performed digitally within the PLL bandwidth. The PLL also features optional pre-filtering of the bit stream to improve spectral purity.

SX1276/77/78/79 feature three distinct RF power amplifiers. Two of those, connected to RFO_LF and RFO_HF, can deliver up to +14 dBm, are unregulated for high power efficiency and can be connected directly to their respective RF receiver inputs via a pair of passive components to form a single antenna port high efficiency transceiver. The third PA, connected to the PA_BOOST pin and can deliver up to +20 dBm via a dedicated matching network. Unlike the high efficiency PAs, this high-stability PA covers all frequency bands that the frequency synthesizer addresses.

SX1276/77/78/79 also include two timing references, an RC oscillator and a 32 MHz crystal oscillator.

All major parameters of the RF front end and digital state machine are fully configurable via an SPI interface which gives access to SX1276/77/78/79's configuration registers. This includes a mode auto sequencer that oversees the transition and calibration of the SX1276/77/78/79 between intermediate modes of operation in the fastest time possible.

The SX1276/77/78/79 are equipped with both standard FSK and long range spread spectrum (LoRa™) modems. Depending upon the mode selected either conventional OOK or FSK modulation may be employed or the LoRa™ spread spectrum modem.

3.1. LoRa™ Modem

The LoRa™ modem uses a proprietary spread spectrum modulation technique. This modulation, in contrast to legacy modulation techniques, permits an increase in link budget and increased immunity to in-band interference. At the same time the frequency tolerance requirement of the crystal reference oscillator is relaxed - allowing a performance increase for a reduction in system cost. For a detailed description of the design trade-offs and operation of the SX1276/77/78/79 please consult Section 4.1 of the datasheet.

3.2. FSK/OOK Modem

In FSK/OOK mode the SX1276/77/78/79 supports standard modulation techniques including OOK, FSK, GFSK, MSK and GMSK. The SX1276/77/78/79 is especially suited to narrow band communication thanks the low-IF architecture employed and the built-in AFC functionality. For full information on the FSK/OOK modem please consult Section 4.2 of this document.

4. SX1276/77/78/79 Digital Electronics

4.1. The LoRa™ Modem

The LoRa™ modem uses spread spectrum modulation and forward error correction techniques to increase the range and robustness of radio communication links compared to traditional FSK or OOK based modulation. Examples of the performance improvement possible, for several possible settings, are summarised in the table below. Here the spreading factor and error correction rate are design variables that allow the designer to optimise the trade-off between occupied bandwidth, data rate, link budget improvement and immunity to interference.

*Table 12 Example **LoRa™** Modem Performances, 868MHz Band*

Bandwidth (kHz)	Spreading Factor	Coding rate	Nominal Rb (bps)	Sensitivity indication (dBm)	Frequency Reference
10.4	6	4/5	782	-131	TCXO
	12	4/5	24	-147	
20.8	6	4/5	1562	-128	
	12	4/5	49	-144	
62.5	6	4/5	4688	-121	XTAL
	12	4/5	146	-139	
125	6	4/5	9380	-118	
	12	4/5	293	-136	

Notes - for all bandwidths lower than 62.5 kHz, it is advised to use a TCXO as a frequency reference. This is required to meet the frequency error tolerance specifications given in the Electrical Specification

- Higher spreading factors and longer transmission times impose more stringent constraints on the short term frequency stability of the reference. Please get in touch with a Semtech representative to implement extremely low sensitivity products.

For European operation the range of crystal tolerances acceptable for each sub-band (of the ERC 70-03) is given in the specifications table. For US based operation a frequency hopping mode is available that automates both the LoRa™ spread spectrum and frequency hopping spread spectrum processes.

Another important facet of the LoRa™ modem is its increased immunity to interference. The LoRa™ modem is capable of co-channel GMSK rejection of up to 20 dB. This immunity to interference permits the simple coexistence of LoRa™ modulated systems either in bands of heavy spectral usage or in hybrid communication networks that use LoRa™ to extend range when legacy modulation schemes fail.

4.1.1. Link Design Using the LoRa™ Modem

4.1.1.1. Overview

The LoRa™ modem is setup as shown in the following figure. This configuration permits the simple replacement of the FSK modem with the LoRa™ modem via the configuration register setting *RegOpMode*. This change can be performed on the fly (in Sleep operating mode) thus permitting the use of both standard FSK or OOK in conjunction with the long range capability. The LoRa™ modulation and demodulation process is proprietary, it uses a form of spread spectrum modulation combined with cyclic error correction coding. The combined influence of these two factors is an increase in link budget and enhanced immunity to interference.

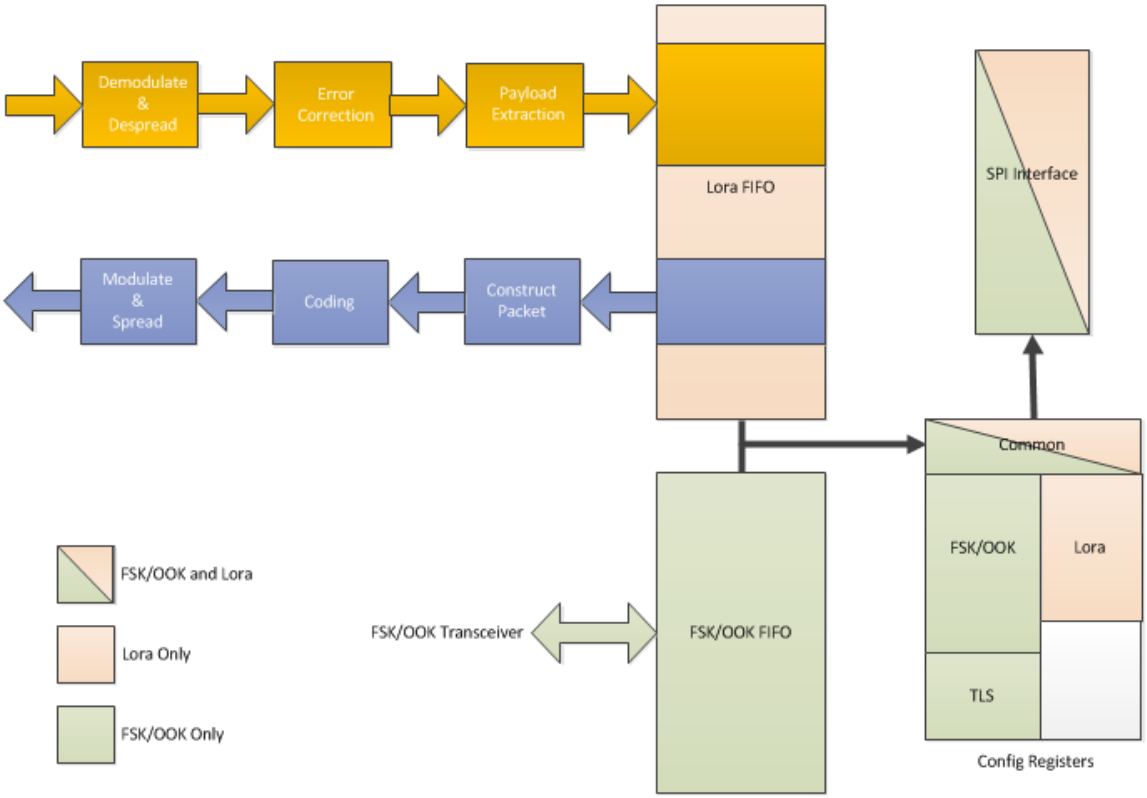


Figure 5. LoRa™ Modem Connectivity

A simplified outline of the transmit and receive processes is also shown above. Here we see that the LoRa™ modem has an independent dual port data buffer FIFO that is accessed through an SPI interface common to all modes. Upon selection of LoRa™ mode, the configuration register mapping of the SX1276/77/78/79 changes. For full details of this change please consult the register description of Section 6.

So that it is possible to optimise the LoRa™ modulation for a given application, access is given to the designer to three critical design parameters. Each one permitting a trade off between link budget, immunity to interference, spectral occupancy and nominal data rate. These parameters are spreading factor, modulation bandwidth and error coding rate.

4.1.1.2. Spreading Factor

The spread spectrum LoRa™ modulation is performed by representing each bit of payload information by multiple chips of information. The rate at which the spread information is sent is referred to as the symbol rate (Rs), the ratio between the nominal symbol rate and chip rate is the spreading factor and represents the number of symbols sent per bit of information. The range of values accessible with the LoRa™ modem are shown in the following table.

Table 13 Range of Spreading Factors

SpreadingFactor (RegModulationCfg)	Spreading Factor (Chips / symbol)	LoRa Demodulator SNR
6	64	-5 dB
7	128	-7.5 dB
8	256	-10 dB
9	512	-12.5 dB
10	1024	-15 dB
11	2048	-17.5 dB
12	4096	-20 dB

Note that the spreading factor, *SpreadingFactor*, must be known in advance on both transmit and receive sides of the link as different spreading factors are orthogonal to each other. Note also the resulting signal to noise ratio (SNR) required at the receiver input. It is the capability to receive signals with negative SNR that increases the sensitivity, so link budget and range, of the LoRa receiver.

Spreading Factor 6

SF = 6 Is a special use case for the highest data rate transmission possible with the LoRa modem. To this end several settings must be activated in the SX1276/77/78/79 registers when it is in use. These settings are only valid for SF6 and should be set back to their default values for other spreading factors:

- Set *SpreadingFactor* = 6 in *RegModemConfig2*
- The header must be set to Implicit mode.
- Set the bit field *DetectionOptimize* of register *RegLoRaDetectOptimize* to value "0b101".
- Write 0x0C in the register *RegDetectionThreshold*.

4.1.1.3. Coding Rate

To further improve the robustness of the link the LoRa™ modem employs cyclic error coding to perform forward error detection and correction. Such error coding incurs a transmission overhead - the resultant additional data overhead per transmission is shown in the table below.

Table 14 Cyclic Coding Overhead

CodingRate (RegTxCfg1)	Cyclic Coding Rate	Overhead Ratio
1	4/5	1.25
2	4/6	1.5
3	4/7	1.75
4	4/8	2

Forward error correction is particularly efficient in improving the reliability of the link in the presence of interference. So that the coding rate (and so robustness to interference) can be changed in response to channel conditions - the coding rate can optionally be included in the packet header for use by the receiver. Please consult Section 4.1.1.6 for more information on the LoRa™ packet and header.

4.1.1.4. Signal Bandwidth

An increase in signal bandwidth permits the use of a higher effective data rate, thus reducing transmission time at the expense of reduced sensitivity improvement. There are of course regulatory constraints in most countries on the permissible occupied bandwidth. Contrary to the FSK modem which is described in terms of the single sideband bandwidth, the LoRa™ modem bandwidth refers to the double sideband bandwidth (or total channel bandwidth). The range of bandwidths relevant to most regulatory situations is given in the LoRa™ modem specifications table (see Section 2.5.5).

Table 15 LoRa Bandwidth Options

Bandwidth (kHz)	Spreading Factor	Coding rate	Nominal Rb (bps)
7.8	12	4/5	18
10.4	12	4/5	24
15.6	12	4/5	37
20.8	12	4/5	49
31.2	12	4/5	73
41.7	12	4/5	98
62.5	12	4/5	146
125	12	4/5	293
250	12	4/5	586
500	12	4/5	1172

Note In the lower band (169 MHz), the 250 kHz and 500 kHz bandwidths are not supported.

4.1.1.5. LoRa™ Transmission Parameter Relationship

With a knowledge of the key parameters that can be controlled by the user we define the LoRa™ symbol rate as:

$$Rs = \frac{BW}{2^{SF}}$$

where BW is the programmed bandwidth and SF is the spreading factor. The transmitted signal is a constant envelope signal. Equivalently, one chip is sent per second per Hz of bandwidth.

4.1.1.6. LoRa™ Packet Structure

The LoRa™ modem employs two types of packet format, explicit and implicit. The explicit packet includes a short header that contains information about the number of bytes, coding rate and whether a CRC is used in the packet. The packet format is shown in the following figure.

The LoRa™ packet comprises three elements:

- A preamble.
- An optional header.
- The data payload.

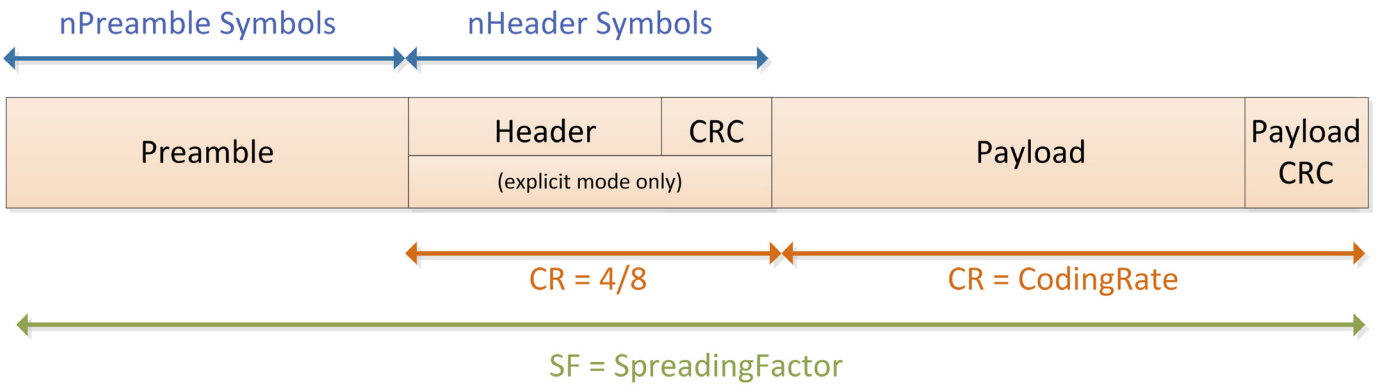


Figure 6. LoRa™ Packet Structure

Preamble

The preamble is used to synchronize receiver with the incoming data flow. By default the packet is configured with a 12 symbol long sequence. This is a programmable variable so the preamble length may be extended, for example in the interest of reducing to receiver duty cycle in receive intensive applications. However, the minimum length suffices for all communication. The transmitted preamble length may be changed by setting the register *PreambleLength* from 6 to 65535, yielding total preamble lengths of 6+4 to 65535+4 symbols, once the fixed overhead of the preamble data is considered. This permits the transmission of a near arbitrarily long preamble sequence.

The receiver undertakes a preamble detection process that periodically restarts. For this reason the preamble length should be configured identical to the transmitter preamble length. Where the preamble length is not known, or can vary, the maximum preamble length should be programmed on the receiver side.

Header

Depending upon the chosen mode of operation two types of header are available. The header type is selected by the *ImplicitHeaderModeOn* bit found within the *RegModemConfig1* register.

Explicit Header Mode

This is the default mode of operation. Here the header provides information on the payload, namely:

- The payload length in bytes.
- The forward error correction code rate
- The presence of an optional 16-bits CRC for the payload.

The header is transmitted with maximum error correction code (4/8). It also has its own CRC to allow the receiver to discard invalid headers.

Implicit Header Mode

In certain scenarios, where the payload, coding rate and CRC presence are fixed or known in advance, it may be advantageous to reduce transmission time by invoking implicit header mode. In this mode the header is removed from the packet. In this case the payload length, error coding rate and presence of the payload CRC must be manually configured on both sides of the radio link.

Note With SF = 6 selected, implicit header mode is the only mode of operation possible.

Explicit Header Mode:

In Explicit Header Mode, the presence of the CRC at the end of the payload in selected only on the transmitter side through the bit *RxPayloadCrcOn* in the register *RegModemConfig1*.

On the receiver side, the bit *RxPayloadCrcOn* in the register *RegModemConfig1* is not used and once the payload has been received, the user should check the bit *CrcOnPayload* in the register *RegHopChannel*. If the bit *CrcOnPayload* is at '1', the user should then check the Irq Flag *PayloadCrcError* to make sure the CRC is valid.

If the bit *CrcOnPayload* is at '0', it means there was no CRC on the payload and thus the IRQ Flag *PayloadCrcError* will not be trigged even if the payload has errors.

Explicit Header	Transmitter	Receiver	CRC Status
Value of the bit RxPayloadCrcOn	0	0	CRC is not checked
	0	1	CRC is not checked
	1	0	CRC is checked
	1	1	CRC is checked

Implicit Header Mode;

In Implicit Header Mode, it is necessary to set the bit RxPayloadCrcOn in the register *RegModemConfig1* on both sides (TX and RX)

Implicit Header	Transmitter	Receiver	CRC Status
Value of the bit RxPayloadCrcOn	0	0	CRC is not checked
	0	1	CRC is always wrong
	1	0	CRC is not checked
	1	1	CRC is checked

Low Data Rate Optimization

Given the potentially long duration of the packet at high spreading factors the option is given to improve the robustness of the transmission to variations in frequency over the duration of the packet transmission and reception. The bit *LowDataRateOptimize* increases the robustness of the LoRa link at these low effective data rates. Its use is mandated when the symbol duration exceeds 16ms. Note that both the transmitter and the receiver must have the same setting for *LowDataRateOptimize*.

Payload

The packet payload is a variable-length field that contains the actual data coded at the error rate either as specified in the header in explicit mode or in the register settings in implicit mode. An optional CRC may be appended. For more information on the payload and how it is loaded from the data buffer FIFO please see Section 4.1.2.3.

4.1.1.7. Time on air

For a given combination of spreading factor (SF), coding rate (CR) and signal bandwidth (BW) the total on-the-air transmission time of a LoRa™ packet can be calculated as follows. From the definition of the symbol rate it is convenient to define the symbol rate:

$$Ts = \frac{1}{Rs}$$

The LoRa packet duration is the sum of the duration of the preamble and the transmitted packet. The preamble length is calculated as follows:

$$T_{preamble} = (n_{preamble} + 4.25)T_{sym}$$

where $n_{preamble}$ is the programmed preamble length, taken from the registers *RegPreambleMsb* and *RegPreambleLsb*.The payload duration depends upon the header mode that is enabled. The following formula gives the number of payload symbols.

$$n_{payload} = 8 + max\left(ceil\left[\frac{(8PL - 4SF + 28 + 16CRC - 20IH)}{4(SF - 2DE)}\right](CR + 4), 0\right)$$

With the following dependencies:

- PL is the number of Payload bytes (1 to 255)
- SF is the spreading factor (6 to 12)
- IH=0 when the header is enabled, IH=1 when no header is present
- DE=1 when *LowDataRateOptimize*=1, DE=0 otherwise
- CR is the coding rate (1 corresponding to 4/5, 4 to 4/8)

The Payload duration is then the symbol period multiplied by the number of Payload symbols

$$T_{payload} = n_{payload} \times T_s$$

The time on air, or packet duration, in simply then the sum of the preamble and payload duration.

$$T_{packet} = T_{preamble} + T_{payload}$$

4.1.1.8. Frequency Hopping with LoRa™

Frequency hopping spread spectrum (FHSS) is typically employed when the duration of a single packet could exceed regulatory requirements relating to the maximum permissible channel dwell time. This is most notably the case in US operation where the 902 to 928 MHz ISM band which makes provision for frequency hopping operation. To ease the implementation of FHSS systems the frequency hopping mode of the LoRa™ modem can be enabled by setting *FreqHoppingPeriod* to a non-zero value in register *RegHopPeriod*.

Principle of Operation

The principle behind the FHSS scheme is that a portion of each LoRa™ packet is transmitted on each hopping channel from a look up table of frequencies managed by the host microcontroller. After a predetermined hopping period the transmitter and receiver change to the next channel in a predefined list of hopping frequencies to continue transmission and reception of the next portion of the packet. The time which the transmission will dwell in any given channel is determined by *FreqHoppingPeriod* which is an integer multiple of symbol periods:

$$HoppingPeriod = Ts \times FreqHoppingPeriod$$

The frequency hopping transmission and reception process starts at channel 0. The preamble and header are transmitted first on channel 0. At the beginning of each transmission the channel counter *FhssPresentChannel* (located in the register *RegHopChannel*) is incremented and the interrupt signal *FhssChangeChannel* is generated. The new frequency must then be programmed within the hopping period to ensure it is taken into account for the next hop, the interrupt *ChangeChannelFhss* is then to be cleared by writing a logical ‘1’.

FHSS Reception always starts on channel 0. The receiver waits for a valid preamble detection before starting the frequency hopping process as described above. Note that in the eventuality of header CRC corruption, the receiver will automatically request channel 0 and recommence the valid preamble detection process.

Timing of Channel Updates

The interrupt requesting the channel change, *FhssChangeChannel,* is generated upon transition to the new frequency. The frequency hopping process is illustrated in the diagram below:

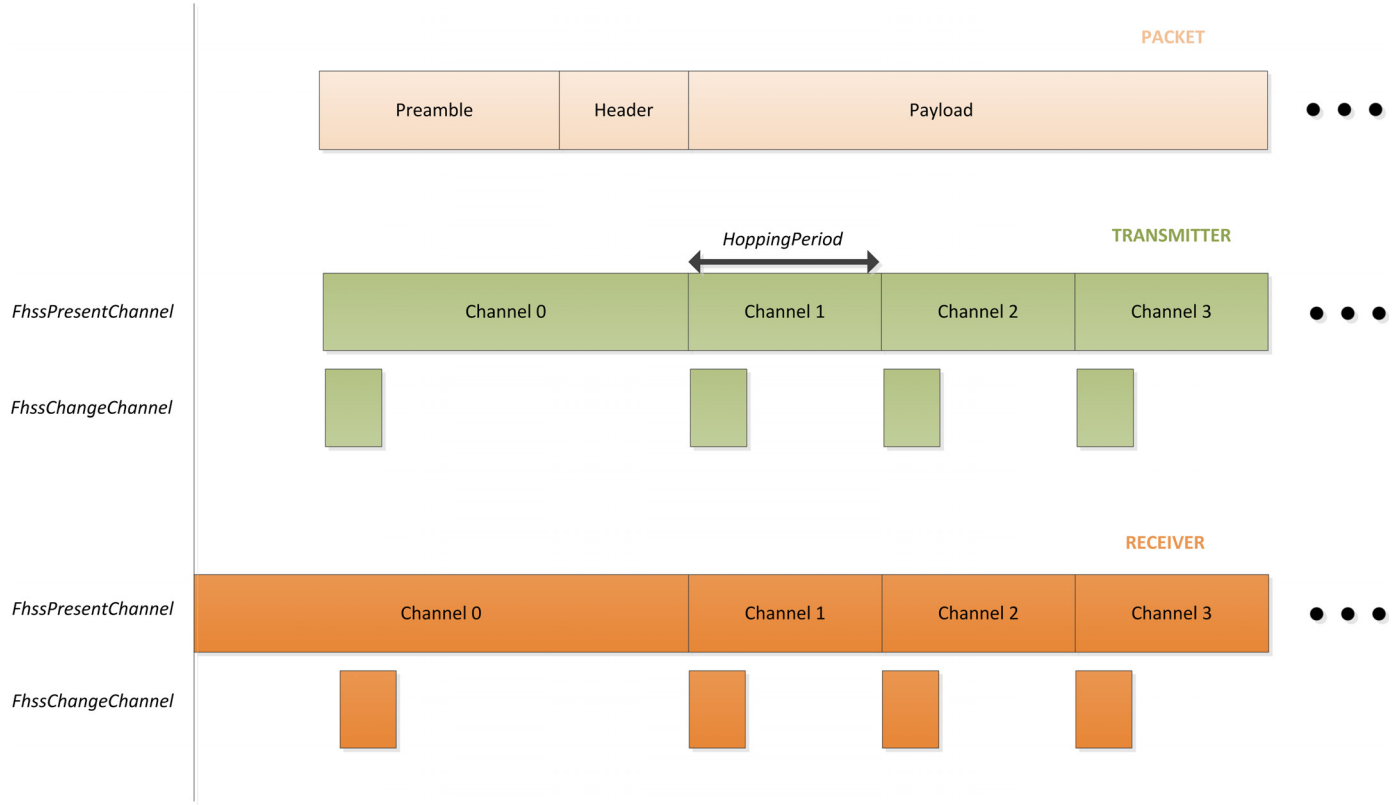


Figure 7. Interrupts Generated in the Case of Successful Frequency Hopping Communication.

4.1.2. LoRa™ Digital Interface

The LoRa™ modem comprises three types of digital interface, static configuration registers, status registers and a FIFO data buffer. All are accessed through the SX1276/77/78/79's SPI interface - full details of each type of register are given below. Full listings of the register addresses used for SPI access are given in Section 6.4.

4.1.2.1. LoRa™ Configuration Registers

Configuration registers are accessed through the SPI interface. Registers are readable in all device mode including Sleep. However, they should be **written only in Sleep and Standby modes**. Please note that **the automatic top level sequencer (TLS modes) are not available in** LoRa™ **mode** and **the configuration register mapping changes as shown in Table** 41**.** The content of the LoRa™ configuration registers is retained in FSK/OOK mode. For the functionality of mode registers common to both FSK/OOK and LoRa™ mode, please consult the Analog and RF Front End section of this document (Section 5).

4.1.2.2. Status Registers

Status registers provide status information during receiver operation.

4.1.2.3. LoRa™ Mode FIFO Data Buffer

Overview

The SX1276/77/78/79 is equipped with a 256 byte RAM data buffer which is uniquely accessible in LoRa mode. This RAM area, herein referred to as the FIFO Data buffer, is fully customizable by the user and allows access to the received, or to be transmitted, data. All access to the LoRa™ FIFO data buffer is done via the SPI interface. A diagram of the user defined memory mapping of the FIFO data buffer is shown below. These FIFO data buffer can be read in all operating modes except sleep and store data related to the last receive operation performed. It is automatically cleared of old content upon each new transition to receive mode.

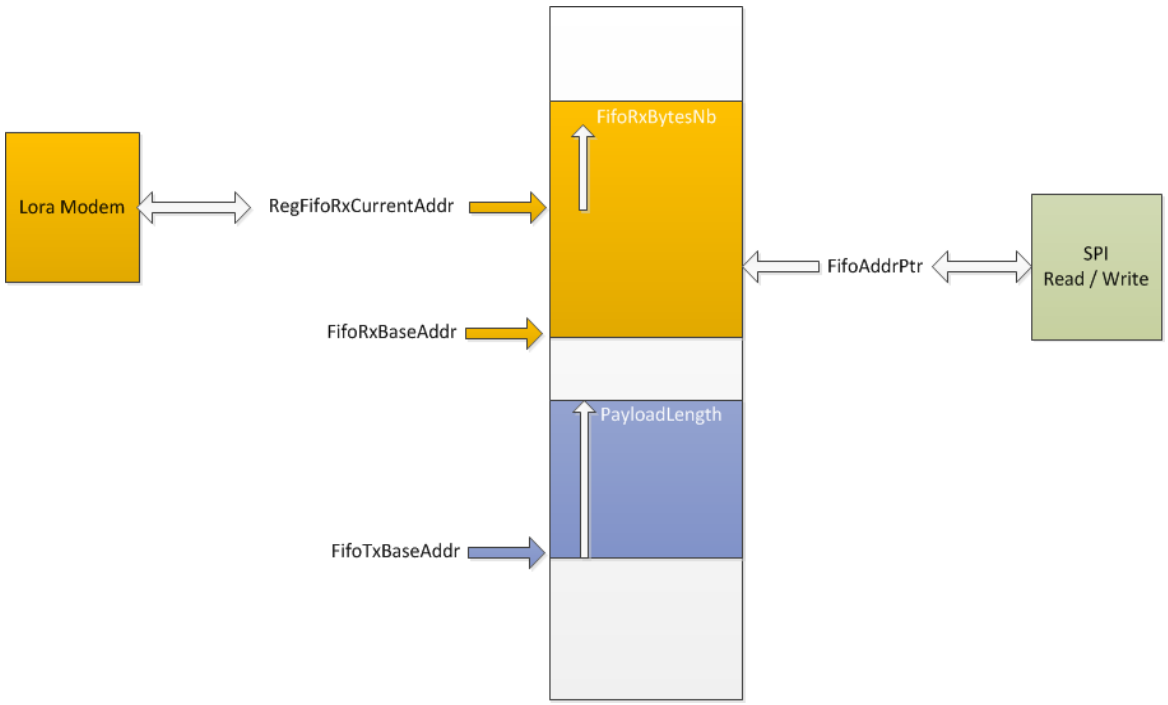


Figure 8. LoRa™ Data Buffer

Principle of Operation

Thanks to its dual port configuration, it is possible to simultaneously store both transmit and receive information in the FIFO data buffer. The register *RegFifoTxBaseAddr* specifies the point in memory where the transmit information is stored. Similarly, for receiver operation, the register *RegFifoRxBaseAddr* indicates the point in the data buffer where information will be written to in event of a receive operation.

By default, the device is configured at power up so that half of the available memory is dedicated to Rx (*RegFifoRxBaseAddr* initialized at address 0x00) and the other half is dedicated for Tx (*RegFifoTxBaseAddr* initialized at address 0x80).

However, due to the contiguous nature of the FIFO data buffer, the base addresses for Tx and Rx are fully configurable across the 256 byte memory area. Each pointer can be set independently anywhere within the FIFO. To exploit the maximum FIFO data buffer size in transmit or receive mode, the whole FIFO data buffer can be used in each mode by setting the base addresses *RegFifoTxBaseAddr* and *RegFifoRxBaseAddr* at the bottom of the memory (0x00).

The FIFO data buffer is cleared when the device is put in SLEEP mode, consequently no access to the FIFO data buffer is possible in sleep mode. However, the data in the FIFO data buffer are retained when switching across the other LoRa™ modes of operation, so that a received packet can be retransmitted with minimum data handling on the controller side. The FIFO data buffer is not self-clearing (unless if the device is put in sleep mode) and the data will only be “erased” when a new set of data is written into the occupied memory location.

The FIFO data buffer location to be read from, or written to, via the SPI interface is defined by the address pointer *RegFifoAddrPtr*. Before any read or write operation it is hence necessary to initialize this pointer to the corresponding base value. Upon reading or writing to the FIFO data buffer (*RegFifo*) the address pointer will then increment automatically.

The register *RegRxNbBytes* defines the size of the memory location to be written in the event of a successful receive operation. The register *RegPayloadLength* indicates the size of the memory location to be transmitted. In implicit header mode, the register *RegRxNbBytes* is not used as the number of payload bytes is known. Otherwise, in explicit header mode, the initial size of the receive buffer is set to the packet length in the received header. The register *RegFifoRxCurrentAddr* indicates the location of the last packet received in the FIFO so that the last packet received can be easily read by pointing the register *RegFifoAddrPtr* to this register.

It is important to notice that all the received data will be written to the FIFO data buffer even if the CRC is invalid, permitting user defined post processing of corrupted data. It is also important to note that when receiving, if the packet size exceeds the buffer memory allocated for the Rx, it will overwrite the transmit portion of the data buffer.

4.1.2.4. Interrupts in LoRa Mode

Two registers are used to control the IRQ in LoRa mode, the register *RegIrqFlagsMask* which is used to mask the interrupts and the register *RegIrqFlags* which indicates which IRQ has been trigged.
In the register *RegIrqFlagsMask,* setting a bit to ‘1’ will mask the interrupt, meaning this interrupt is disactivated. By default all the interrupt are available.
In the register *RegIrqFlags*, a ‘1’ indicates a given IRQ has been trigged and then the IRQ must be clear by writing a ‘1’.

4.1.3. Operation of the LoRa™ Modem

4.1.3.1. Operating Mode Control

The operating modes of the LoRa™ modem are accessed by enabling LoRa™ mode (setting the *LongRangeMode* bit of *RegOpMode*). Depending upon the operating mode selected the range of functionality and register access is given by the following table:

Table 16 LoRa™ Operating Mode Functionality

Operating Mode	Description
SLEEP	Low-power mode. In this mode only SPI and configuration registers are accessible. Lora FIFO is not accessible. Note that this is the only mode permissible to switch between FSK/OOK mode and LoRa mode.
STANDBY	both Crystal oscillator and Lora baseband blocks are turned on.RF part and PLLs are disabled
FSTX	This is a frequency synthesis mode for transmission. The PLL selected for transmission is locked and active at the transmit frequency. The RF part is off.
FSRX	This is a frequency synthesis mode for reception. The PLL selected for reception is locked and active at the receive frequency. The RF part is off.
TX	When activated the SX1276/77/78/79 powers all remaining blocks required for transmit, ramps the PA, transmits the packet and returns to Standby mode.
RXCONTINUOUS	When activated the SX1276/77/78/79 powers all remaining blocks required for reception, processing all received data until a new user request is made to change operating mode.
RXSINGLE	When activated the SX1276/77/78/79 powers all remaining blocks required for reception, remains in this state until a valid packet has been received and then returns to Standby mode.
CAD	When in CAD mode, the device will check a given channel to detect LoRa preamble signal

It is possible to access any mode from any other mode by changing the value in the *RegOpMode* register.

4.1.4. Frequency Settings

Recalling that the frequency step is given by:

$$F_{STEP} = \frac{F_{XOSC}}{2^{19}}$$

In order to set LO frequency values following registers are available.

Frf is a 24-bit register which defines carrier frequency. The carrier frequency relates to the register contents by following formula:

$$F_{RF} = F_{STEP} \times Frf(23,0)$$

4.1.5. Frequency Error Indication

The SX1276/77/78/79 derives its RF centre frequency from a crystal reference oscillator which has a finite frequency precision. Errors in reference frequency will manifest themselves as errors of the same proportion from the RF centre frequency.

In LoRa receive mode the SX1276/77/78/79 is capable of measuring the frequency offset between the receiver centre frequency and that of an incoming LoRa signal. The modem is intolerant of frequency offsets in the region of +/- 25% of the bandwidth and will accurately report the error over this same range.

The error is read by reading the three *RegFei* registers. The contents of which are a signed 20 bit two's compliment word, *FreqError*. The frequency error is determined from the register contents by:

$$F_{Error} = \frac{FreqError \times 2^{24}}{F_{xtal}} \times \frac{BW[kHz]}{500}$$

Where Fxtal is the crystal frequency.

To correct the measured frequency error there are two steps to be taken. First the frequency error is subtracted from the RF centre frequency. This calculation must be performed locally (or in a look-up-table), no provision is made in the circuit to apply the correction automatically.

Secondly, assuming that the frequency error is due to reference oscillator drift, the data rate of the LoRa modem must also be compensated accordingly. This is done by

*PpmOffset = 0.95 * measured Offset [PPM]*

Where PpmOffset is the value to be programmed into register 0x27 and the measured Offset is the PPM drift equivalent to the frequency error reported by the LoRa frequency error indicator. The *PpmOffset* value is a signed two's compliment value.

4.1.6. LoRa™ Modem State Machine Sequences

The sequence for transmission and reception of data to and from the LoRa™ modem, together with flow charts of typical sequences of operation, are detailed below.

Data Transmission Sequence

In transmit mode power consumption is optimized by enabling RF, PLL and PA blocks only when packet data needs to be transmitted. Figure 9 shows a typical LoRa™ transmit sequence.

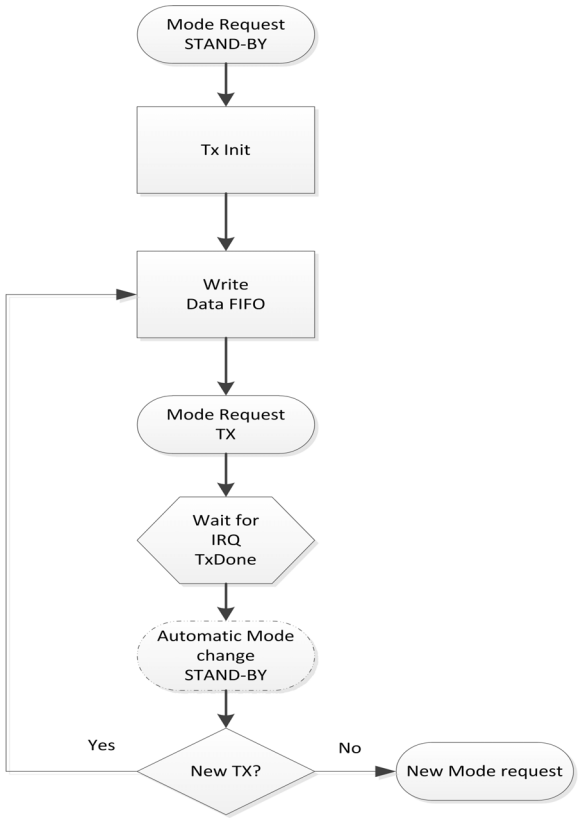


Figure 9. LoRa™ Modulation Transmission Sequence.

- Static configuration registers can only be accessed in Sleep mode, Standby mode or FSTX mode.
- The LoRa™ FIFO can only be filled in Standby mode.
- Data transmission is initiated by sending TX mode request.
- Upon completion the *TxDone* interrupt is issued and the radio returns to Standby mode.
- Following transmission the radio can be manually placed in Sleep mode or the FIFO refilled for a subsequent Tx operation.

LoRa™ Transmit Data FIFO Filling

In order to write packet data into FIFO user should:

1 Set *FifoPtrAddr* to *FifoTxPtrBase*.

2 Write *PayloadLength* bytes to the FIFO (*RegFifo*)

Data Reception Sequence

Figure 10 shows typical LoRa™ receive sequences for both single and continuous receiver modes of operation.

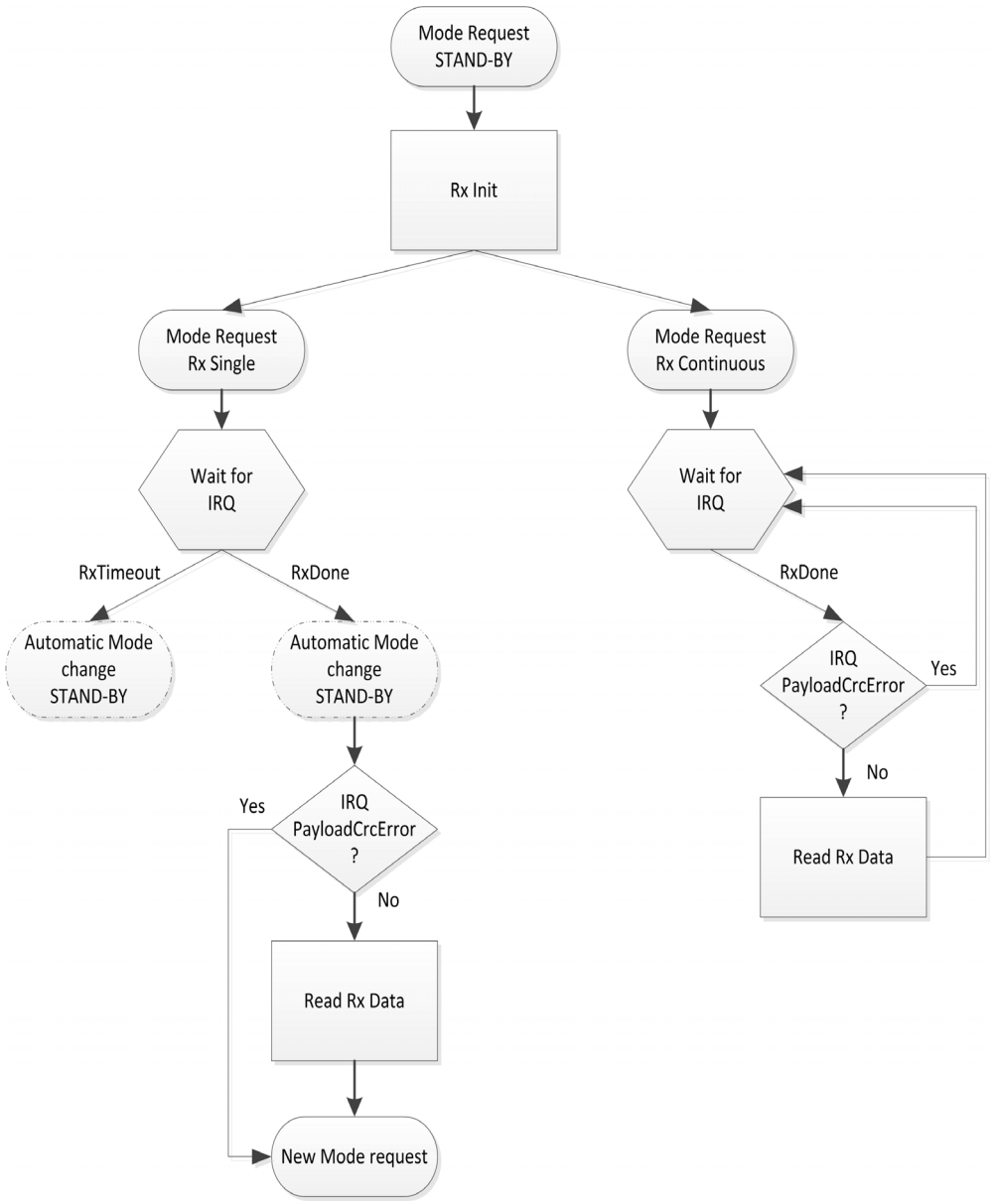


Figure 10. LoRa™ Receive Sequence.

The LoRa receive modem can work in two distinct mode

1. Single receive mode
2. Continuous receive mode

Those two modes correspond to different use cases and it is important to understand the subtle differences between them.

Single Reception Operating Mode

In this mode, the modem searches for a preamble during a given period of time. If a preamble hasn't been found at the end of the time window, the chip generates the *RxTimeout* interrupt and goes back to Standby mode. The length of the reception window (in symbols) is defined by the *RegSymbTimeout* register and should be in the range of 4 (minimum time for the modem to acquire lock on a preamble) up to 1023 symbols.

At the end of the payload, the *RxDone* interrupt is generated together with the interrupt *PayloadCrcError* if the payload CRC is not valid. However, even when the CRC is not valid, the data are written in the FIFO data buffer for post processing. Following the *RxDone* interrupt the radio goes to Standby mode.

The modem will also automatically return in Standby mode when the interrupts *RxDone* is generated. Therefore, this mode should only be used when the time window of arrival of the packet is known. In other cases, the RX continuous mode should be used.

In Rx single mode, low-power is achieved by turning off PLL and RF blocks as soon as a packet has been received. The flow is as follows:

1 *Set FifoAddrPtr* to *FifoRxBaseAddr*.

2 Static configuration register device can be written in either Sleep mode, Standby mode or FSRX mode.

3 A single packet receive operation is initiated by selecting the operating mode RXSINGLE.

4 The receiver will then await the reception of a valid preamble. Once received, the gain of the receive chain is set. Following the ensuing reception of a valid header, indicated by the *ValidHeader* interrupt in explicit mode. The packet reception process commences. Once the reception process is complete the *RxDone* interrupt is set. The radio then returns automatically to Standby mode to reduce power consumption.

5 The receiver status register *PayloadCrcError* should be checked for packet payload integrity.

6 If a valid packet payload has been received then the FIFO should be read (See Payload Data Extraction below). Should a subsequent single packet reception need to be triggered, then the RXSINGLE operating mode must be re-selected to launch the receive process again - taking care to reset the SPI pointer (*FifoAddrPtr*) to the base location in memory (*FifoRxBaseAddr*).

Continuous Reception Operating Mode

In continuous receive mode, the modem scans the channel continuously for a preamble. Each time a preamble is detected the modem tracks it until the packet is received and then carries on waiting for the next preamble.

If the preamble length exceeds the anticipated value set by the registers *RegPreambleMsb* and *RegPreambleLsb* (measured in symbol periods) the preamble will be dropped and the search for a preamble restarted. However, this scenario will not be flagged by any interrupt. In continuous RX mode, opposite to the single RX mode, the RxTimeout interrupt will never occur and the device will never go in Standby mode automatically.

It is also important to note that the demodulated bytes are written in the data buffer memory in the order received. Meaning, the first byte of a new packet is written just after the last byte of the preceding packet. The RX modem address pointer is never reset as long as this mode is enabled. It is therefore necessary for the companion microcontroller to handle the address pointer to make sure the FIFO data buffer is never full.

In continuous mode the received packet processing sequence is given below.

1 Whilst in Sleep or Standby mode select RXCONT mode.

2 Upon reception of a valid header CRC the *RxDone i*nterrupt is set. The radio remains in RXCONT mode waiting for the next RX LoRa™ packet.

3 The *PayloadCrcError* flag should be checked for packet integrity.

4 If packet has been correctly received the FIFO data buffer can be read (see below).

5 The reception process (steps 2 - 4) can be repeated or receiver operating mode exited as desired.

In continuous mode status information are available only for the last packet received, i.e. the corresponding registers should be read before the next *RxDone* arrives.

Rx Single and Rx Continuous Use Cases

The LoRa single reception mode is used mainly in battery operated systems or in systems where the companion microcontroller has a limited availability of timers. In such systems, the use of the timeout present in Rx Single reception mode allows the end user to limit the amount of time spent in reception (and thus limiting the power consumption) while not using any of the companion MCU timers (the MCU can then be in sleep mode while the radio is in the reception mode). The RxTimeout interrupt generated at the end of the reception period is then used to wake-up the companion MCU. One of the advantages of the RxSingle mode is that the interrupt RxTimeout will not be triggered if the device is currently receiving data, thus giving the priority to the reception of the data over the timeout. However, if during the reception, the device loses track of the data due to external perturbation, the device will drop the reception, flag the interrupt RxTimeout and go in StandBy mode to decrease the power consumption of the system.

On the other hand, The LoRa continuous reception mode is used in systems which do not have power restrictions or on system where the use of a companion MCU timer is preferred over the radio embedded timeout system. In RxContinuous mode, the radio will track any LoRa signal present in the air and carry on the reception of packets until the companion MCU sets the radio into another mode of operation. Upon reception the interrupt RxDone will be trigged but the device will stay in Rx Mode, ready for the reception of the next packet.

Payload Data Extraction from FIFO

In order to retrieve received data from FIFO the user must ensure that *ValidHeader*, *PayloadCrcError, RxDone* and *RxTimeout* interrupts in the status register RegIrqFlags are not asserted to ensure that packet reception has terminated successfully (i.e. no flags should be set).

In case of errors the steps below should be skipped and the packet discarded. In order to retrieve valid received data from the FIFO the user must:

- *RegRxNbBytes* Indicates the number of bytes that have been received thus far.
- *RegFifoAddr*Ptr is a dynamic pointer that indicates precisely where the Lora modem received data has been written up to.
- Set *RegFifoAddr*Ptr to *RegFifoRxCurrentAddr*. This sets the FIFO pointer to the location of the last packet received in the FIFO. The payload can then be extracted by reading the register *RegFifo, RegRxNbBytes* times.
- Alternatively, it is possible to manually point to the location of the last packet received, from the start of the current packet, by setting *RegFifoAddrPtr* to *RegFifoRxByteAddr* minus *RegRxNbBytes*. The payload bytes can then be read from the FIFO by reading the *RegFifo* address *RegRxNbBytes* times.

Packet Filtering based on Preamble Start

The LoRa modem does automatically filter received packets based upon any addressing. However the SX1276/77/78/79 permit software filtering of the received packets based on the contents of the first few bytes of payload. A brief example is given below for a 4 byte address, however, the address length can be selected by the designer.

The objective of the packet filtering process is to determine the presence, or otherwise, of a valid packet designed for the receiver. If the packet is not for the receiver then the radio returns to sleep mode in order to improve battery life.

The software packet filtering process follows the steps below:

- Each time the RxDone interrupt is received, latch the *RegFifoRxByteAddr*[7:0] register content in a variable, this variable will be called start_address. The *RegFifoRxByteAddr*[7:0] register of the SX1276/77/78/79 gives in real time the address of the last byte written in the data buffer + 1 (or the address at which the next byte will be written) by the receive LoRa modem. So by doing this, we make sure that the variable *start_address* always contains the start address of the next packet.

- Upon reception of the interrupt *ValidHeader*, start polling the *RegFifoRxByteAddr*[7:0] register until it begins to increment. The speed at which this register will increment depends on the Spreading factor, the error correction code and the modulation bandwidth. (Note that this interrupt is still generated in implicit mode).

- As soon as *RegFifoRxByteAddr*[7:0] >= start address + 4, the first 4 bytes (address) are stored in the FIFO data buffer. These can be read and tested to see if the packet is destined for the radio and either remaining in Rx mode to receive the packet or returning to sleep mode if not.

Receiver Timeout Operation

In LoRa™ Rx Single mode, a receiver timeout functionality is available that permits the receiver to listen for a predetermined period of time before generating an interrupt signal to indicate that no valid packets have been received. The timer is absolute and commences as soon as the radio is placed in single receive mode. The interrupt itself, *RxTimeout*, can be found in the interrupt register *RegIrqFlags*. In Rx Single mode, the device will return to Standby mode as soon as the interrupt occurs. The user must then clear the interrupt or go into Sleep mode before returning into Rx Single mode. The programmed timeout value is expressed as a multiple of the symbol period and is given by:

$$TimeOut = LoraRxTimeout \cdot Ts$$

Channel Activity Detection

The use of a spread spectrum modulation technique presents challenges in determining whether the channel is already in use by a signal that may be below the noise floor of the receiver. The use of the RSSI in this situation would clearly be impracticable. To this end the channel activity detector is used to detect the presence of other LoRa™ signals. Figure 11 shows the channel activity detection (CAD) process:

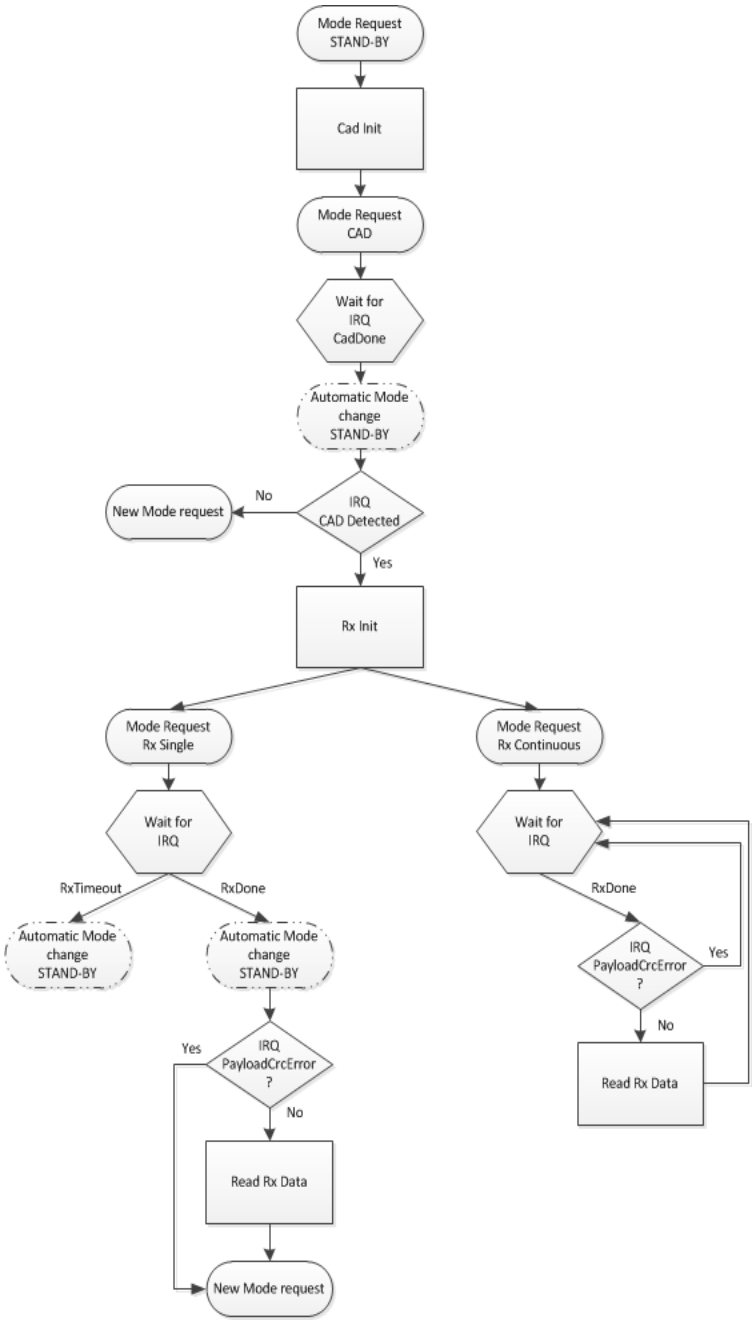


Figure 11. LoRa™ CAD Flow

Principle of Operation

The channel activity detection mode is designed to detect a LoRa preamble on the radio channel with the best possible power efficiency. Once in CAD mode, the SX1276/77/78/79 will perform a very quick scan of the band to detect a LoRa packet preamble.

During a CAD the following operations take place:

- The PLL locks
- The radio receiver captures LoRa preamble symbol of data from the channel. The radio current consumption during that phase corresponds to the specified Rx mode current
- The radio receiver and the PLL turn off, and the modem digital processing starts.
- The modem searches for a correlation between the radio captured samples and the ideal preamble waveform. This correlation process takes a little bit less than a symbol period to perform. The radio current consumption during that phase is greatly reduced.
- Once the calculation is finished the modem generates the CadDone interrupt. If the correlation was successful, CadDetected is generated simultaneously.
- The chip goes back to Standby mode.
- If a preamble was detected, clear the interrupt, then initiate the reception by putting the radio in RX single mode or RX continuous mode.

The time taken for the channel activity detection is dependent upon the LoRa modulation settings used. For a given configuration the typical CAD detection time is shown in the graph below, expressed as a multiple of the LoRa symbol period. Of this period the radio is in receiver mode for (2^{SF} + 32) / BW seconds. For the remainder of the CAD cycle the radio is in a reduced consumption state.

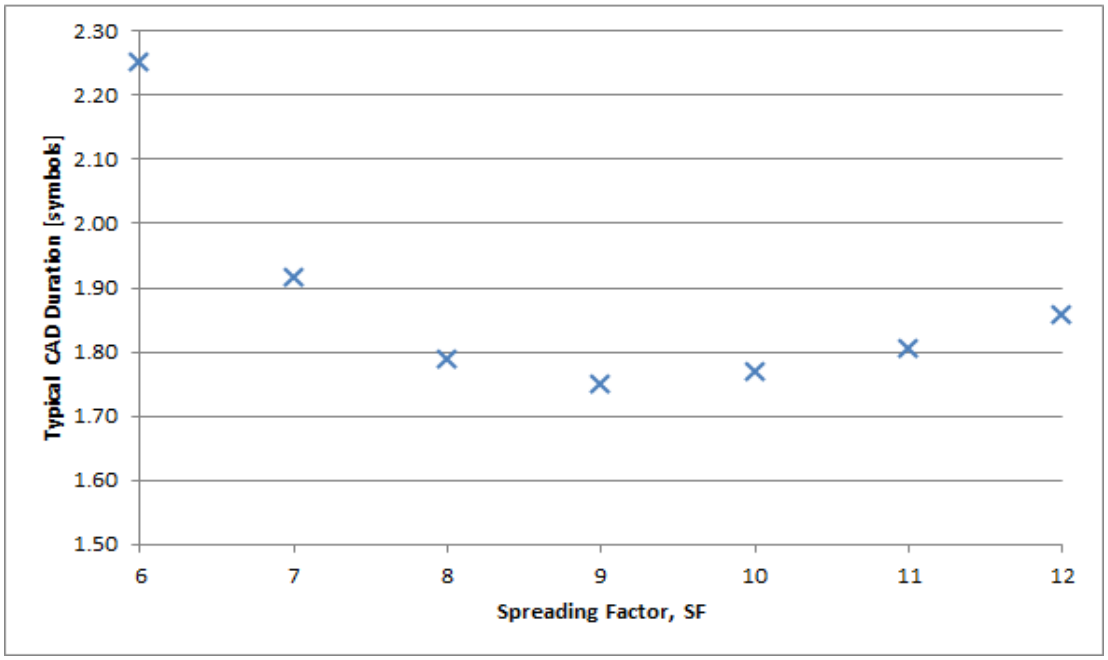


Figure 12. CAD Time as a Function of Spreading Factor

To illustrate this process and the respective consumption in each mode, the CAD process follows the sequence of events outlined below:

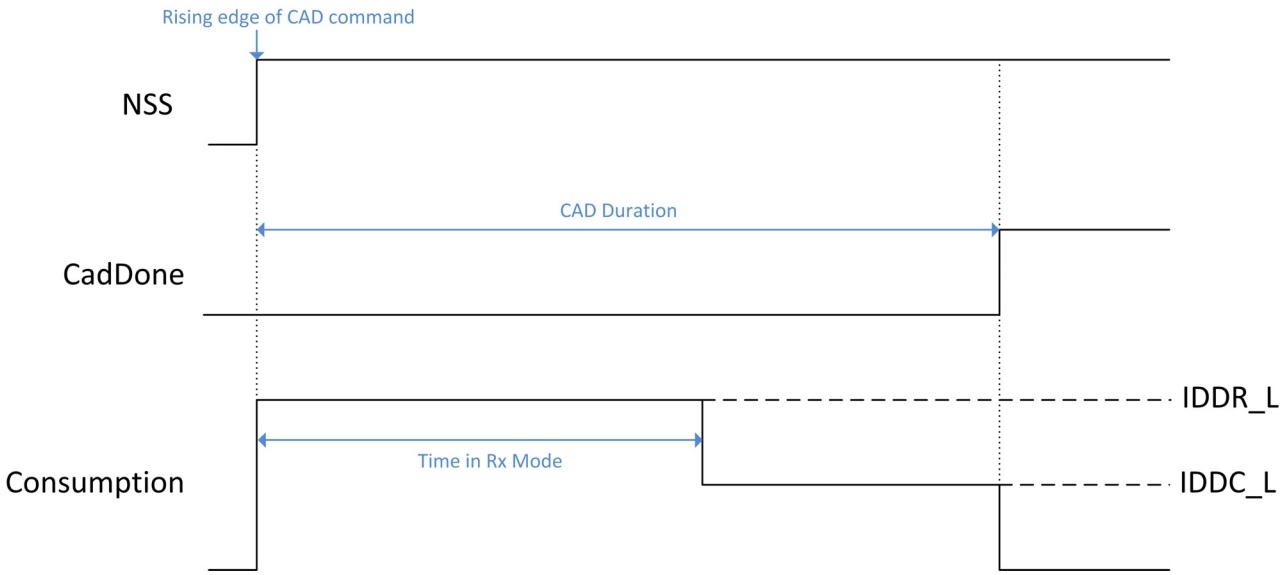


Figure 13. Consumption Profile of the LoRa CAD Process

The receiver is then in full receiver mode for just over half of the activity detection, followed by a reduced consumption processing phase where the consumption varies with the LoRa bandwidth as shown in the table below.

Table 17 LoRa CAD Consumption Figures

Bandwidth (kHz)	Full Rx, IDDR_L (mA)	Processing, IDDC_L (mA)
7.8 to 41.7	11	5.2
62.5	11	5.6
125	11.5	6
250	12.4	6.8
500	13.8	8.3

Note These numbers can be slightly lower when using Band 2 and 3, on the low frequency port.

4.1.6.1. Digital IO Pin Mapping

Six of SX1276/77/78/79's general purpose IO pins are available used in LoRa™ mode. Their mapping is shown below and depends upon the configuration of registers *RegDioMapping1* and *RegDioMapping2.*

Table 18 DIO Mapping LoRa™ Mode

Operating Mode	DIOx Mapping	DIO5	DIO4	DIO3	DIO2	DIO1	DIO0
ALL	00	ModeReady	CadDetected	CadDone	FhssChangeChannel	RxTimeout	RxDone
	01	ClkOut	PllLock	ValidHeader	FhssChangeChannel	FhssChangeChannel	TxDone
	10	ClkOut	PllLock	PayloadCrcError	FhssChangeChannel	CadDetected	CadDone
	11	-	-	-	-	-	-

4.1.7. Modem Status Indicators

The state of the LoRa modem is accessible with the *ModemStatus* bits in *RegModemStat.* They can mostly used for debug in Rx mode and the useful indicators are:

- Bit 0: *Signal Detected* indicates that a valid LoRa preamble has been detected
- Bit 1: *Signal Synchronized* indicates that the end of Preamble has been detected, the modem is in lock
- Bit 3: Header Info Valid toggles high when a valid Header (with correct CRC) is detected

4.2. FSK/OOK Modem

4.2.1. Bit Rate Setting

The bitrate setting is referenced to the crystal oscillator and provides a precise means of setting the bit rate (or equivalently chip) rate of the radio. In continuous transmit mode (Section 4.2.12) the data stream to be transmitted can be input directly to the modulator via pin 10 (DIO2/DATA) in an asynchronous manner, unless Gaussian filtering is used, in which case the DCLK signal on pin 9 (DIO1/DCLK) is used to synchronize the data stream. See section 4.2.2.3 for details on the Gaussian filter.

In Packet mode or in Continuous mode with Gaussian filtering enabled, the Bit Rate (BR) is controlled by bits *Bitrate* in *RegBitrateMsb and RegBitrateLsb*

$$BitRate = \frac{FXOSC}{BitRate(15,0) + \frac{BitrateFrac}{16}}$$

*Note: BitrateFrac bits have **no effect** (i.e may be considered equal to 0) **in OOK** modulation mode.*

The quantity *BitrateFrac* is hence designed to allow very high precision (max. 250 ppm programing resolution) for any bitrate in the programmable range. Table 19 below shows a range of standard bitrates and the accuracy to within which they may be reached.

Table 19 Bit Rate Examples

Type	BitRate (15:8)	BitRate (7:0)	(G)FSK (G)MSK	OOK	Actual BR (b/s)
Classical modem baud rates (multiples of 1.2 kbps)	0x68	0x2B	1.2 kbps	1.2 kbps	1200.015
	0x34	0x15	2.4 kbps	2.4 kbps	2400.060
	0x1A	0x0B	4.8 kbps	4.8 kbps	4799.760
	0x0D	0x05	9.6 kbps	9.6 kbps	9600.960
	0x06	0x83	19.2 kbps	19.2 kbps	19196.16
	0x03	0x41	38.4 kbps		38415.36
	0x01	0xA1	76.8 kbps		76738.60
	0x00	0xD0	153.6 kbps		153846.1
Classical modem baud rates (multiples of 0.9 kbps)	0x02	0x2C	57.6 kbps		57553.95
	0x01	0x16	115.2 kbps		115107.9
Round bit rates (multiples of 12.5, 25 and 50 kbps)	0x0A	0x00	12.5 kbps	12.5 kbps	12500.00
	0x05	0x00	25 kbps	25 kbps	25000.00
	0x80	0x00	50 kbps		50000.00
	0x01	0x40	100 kbps		100000.0
	0x00	0xD5	150 kbps		150234.7
	0x00	0xA0	200 kbps		200000.0
	0x00	0x80	250 kbps		250000.0
	0x00	0x6B	300 kbps		299065.4
Watch Xtal frequency	0x03	0xD1	32.768 kbps	32.768 kbps	32753.32

4.2.2. FSK/OOK Transmission

4.2.2.1. FSK Modulation

FSK modulation is performed inside the PLL bandwidth, by changing the fractional divider ratio in the feedback loop of the PLL. The high resolution of the sigma-delta modulator, allows for very narrow frequency deviation. The frequency deviation F_{DEV} is given by:

$$F_{DEV} = F_{STEP} \times Fdev(13,0)$$

To ensure correct modulation, the following limit applies:

$$F_{DEV} + \frac{BR}{2} \leq (250)kHz$$

Note No constraint applies to the modulation index of the transmitter, but the frequency deviation must be set between 600 Hz and 200 kHz.

4.2.2.2. OOK Modulation

OOK modulation is applied by switching on and off the power amplifier. Digital control and ramping are available to improve the transient power response of the OOK transmitter.

4.2.2.3. Modulation Shaping

Modulation shaping can be applied in both OOK and FSK modulation modes, to improve the narrow band response of the transmitter. Both shaping features are controlled with *PaRamp* bits in *RegPaRamp*.

- In FSK mode, a Gaussian filter with BT = 0.5 or 1 is used to filter the modulation stream, at the input of the sigma-delta modulator. If the Gaussian filter is enabled when the SX1276/77/78/79 is in Continuous mode, DCLK signal on pin 10 (DIO1/DCLK) will trigger an interrupt on the uC each time a new bit has to be transmitted. Please refer to section 4.2.12.2 for details.
- When OOK modulation is used, the PA bias voltages are ramped up and down smoothly when the PA is turned on and off, to reduce spectral splatter.

Note *The transmitter must be restarted if the ModulationShaping setting is changed, in order to recalibrate the built-in filter.*

4.2.3. FSK/OOK Reception

4.2.3.1. FSK Demodulator

The FSK demodulator of the SX1276/77/78/79 is designed to demodulate FSK, GFSK, MSK and GMSK modulated signals. It is most efficient when the modulation index of the signal is greater than 0.5 and below 10:

$$0.5 \leq \beta = \frac{2 \times F_{DEV}}{BR} \leq 10$$

The output of the FSK demodulator can be fed to the Bit Synchronizer to provide the companion processor with a synchronous data stream in Continuous mode.

4.2.3.2. OOK Demodulator

The OOK demodulator performs a comparison of the RSSI output and a threshold value. Three different threshold modes are available, configured through bits *OokThreshType* in *RegOokPeak*.

The recommended mode of operation is the “Peak” threshold mode, illustrated in Figure 14:

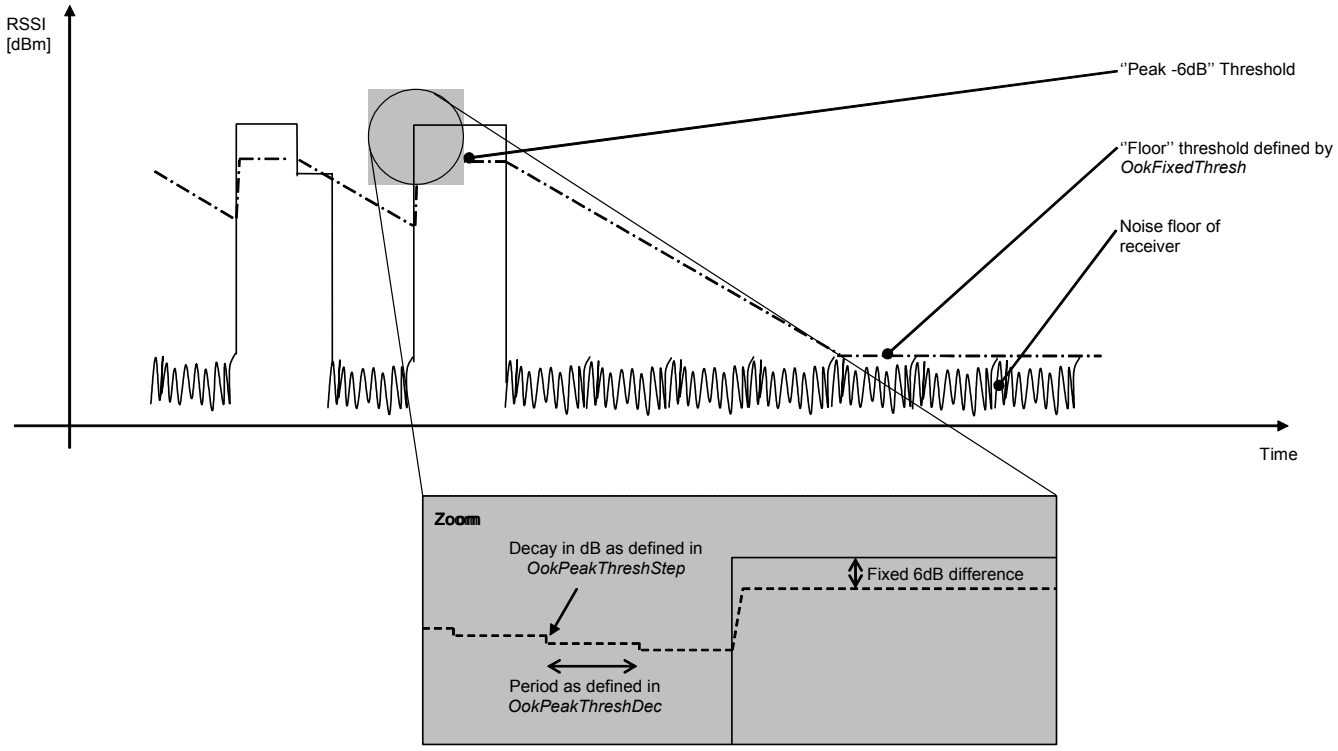


Figure 14. OOK Peak Demodulator Description

In peak threshold mode the comparison threshold level is the peak value of the RSSI, reduced by 6dB. In the absence of an input signal, or during the reception of a logical '0', the acquired peak value is decremented by one *OokPeakThreshStep* every *OokPeakThreshDec* period.

When the RSSI output is null for a long time (for instance after a long string of "0" received, or if no transmitter is present), the peak threshold level will continue falling until it reaches the "Floor Threshold", programmed in *OokFixedThresh*.

The default settings of the OOK demodulator lead to the performance stated in the electrical specification. However, in applications in which sudden signal drops are awaited during a reception, the three parameters should be optimized accordingly.

Optimizing the Floor Threshold

OokFixedThresh determines the sensitivity of the OOK receiver, as it sets the comparison threshold for weak input signals (i.e. those close to the noise floor). Significant sensitivity improvements can be generated if configured correctly.

Note that the noise floor of the receiver at the demodulator input depends on:

- The noise figure of the receiver.
- The gain of the receive chain from antenna to base band.
- The matching - including SAW filter if any.
- The bandwidth of the channel filters.

It is therefore important to note that the setting of *OokFixedThresh* will be application dependent. The following procedure is recommended to optimize *OokFixedThresh*.

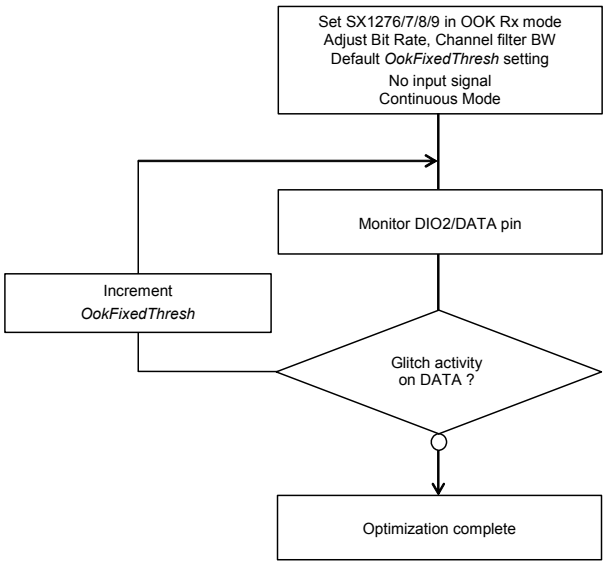


Figure 15. Floor Threshold Optimization

The new floor threshold value found during this test should be used for OOK reception with those receiver settings.

Optimizing OOK Demodulator for Fast Fading Signals

A sudden drop in signal strength can cause the bit error rate to increase. For applications where the expected signal drop can be estimated, the following OOK demodulator parameters *OokPeakThreshStep* and *OokPeakThreshDec* can be optimized as described below for a given number of threshold decrements per bit. Refer to *RegOokPeak* to access those settings.

Alternative OOK Demodulator Threshold Modes

In addition to the Peak OOK threshold mode, the user can alternatively select two other types of threshold detectors:

- Fixed Threshold: The value is selected through *OokFixedThresh*
- Average Threshold: Data supplied by the RSSI block is averaged, and this operation mode should only be used with DC-free encoded data.

4.2.3.3. Bit Synchronizer

The bit synchronizer provides a clean and synchronized digital output based upon timing recovery information gleaned from the received data edge transitions. Its output is made available on pin DIO1/DCLK in Continuous mode and can be disabled through register settings. However, for optimum receiver performance, especially in Continuous receive mode, its use is strongly advised.

The Bit Synchronizer is automatically activated in Packet mode. Its bit rate is controlled by *BitRateMsb* and *BitRateLsb* in *RegBitrate.*

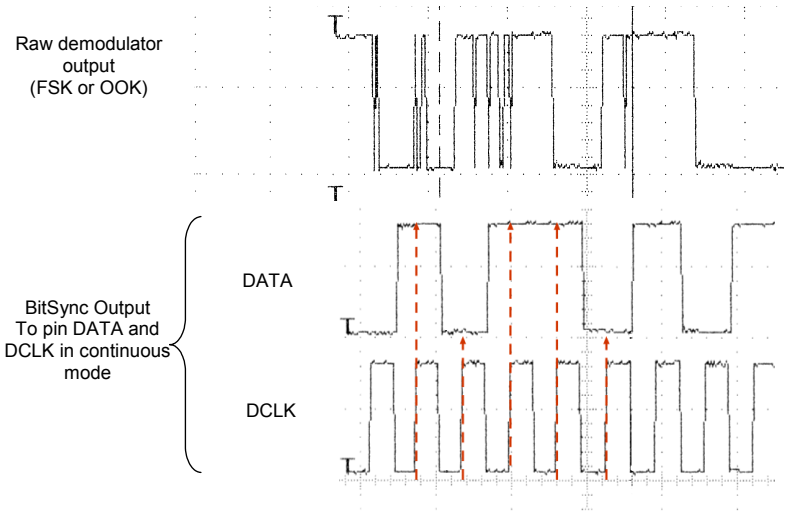


Figure 16. Bit Synchronizer Description

To ensure correct operation of the Bit Synchronizer, the following conditions have to be satisfied:

- A preamble (0x55 or 0xAA) of at least 12 bits is required for synchronization, the longer the synchronization phase is the better the ensuing packet detection rate will be.
- The subsequent payload bit stream must have at least one edge transition (either rising or falling) every 16 bits during data transmission.
- The absolute error between transmitted and received bit rate must not exceed 6.5%.

4.2.3.4. Frequency Error Indicator

This frequency error indicator measures the frequency error between the programmed RF centre frequency and the carrier frequency of the modulated input signal to the receiver. When the FEI is performed, the frequency error is measured and the signed result is loaded in *FeiValue* in *RegFei*, in 2's complement format. The time required for an FEI evaluation is 4 bit periods.

To ensure correct operation of the FEI:

- The measurement must be launched during the reception of preamble.
- The sum of the frequency offset and the 20 dB signal bandwidth must be lower than the base band filter bandwidth. i.e. The whole modulated spectrum must be received.

The 20 dB bandwidth of the signal can be evaluated as follows (double-side bandwidth):

$$BW_{20dB} = 2 \times \left(F_{DEV} + \frac{BR}{2} \right)$$

The frequency error, in Hz, can be calculated with the following formula:

$$FEI = F_{STEP} \times FeiValue$$

The FEI is enabled automatically upon the transition to receive mode and automatically updated every 4 bits.

4.2.3.5. AFC

The AFC is based on the FEI measurement, therefore the same input signal and receiver setting conditions apply. When the AFC procedure is performed the *AfcValue* is directly subtracted from the register that defines the frequency of operation of the chip, F_{RF}. The AFC is executed each time the receiver is enabled, if *AfcAutoOn* = 1.

When the AFC is enabled (*AfcAutoOn* = 1), the user has the option to:

- Clear the former AFC correction value, if *AfcAutoClearOn* = 1. Allowing the next frequency correction to be performed from the initial centre frequency.
- Start the AFC evaluation from the previously corrected frequency. This may be useful in systems in which the centre frequency experiences cumulative drift - such as the ageing of a crystal reference.

The SX1276/77/78/79 offers an alternate receiver bandwidth setting during the AFC phase allowing the accommodation of larger frequency errors. The setting *RegAfcBw* sets the receive bandwidth during the AFC process. In a typical receiver application the, once the AFC is performed, the radio will revert to the receiver communication or channel bandwidth (*RegRxBw*) for the ensuing communication phase.

Note that the FEI measurement is valid only during the reception of preamble. The provision of the *PreambleDetect* flag can hence be used to detect this condition and allow a reliable AFC or FEI operation to be triggered. This process can be performed automatically by using the appropriate options in *StartDemodOnPreamble* found in the *RegRxConfig* register.

A detailed description of the receiver setup to enable the AFC is provided in section 4.2.6.

4.2.3.6. Preamble Detector

The Preamble Detector indicates the reception of a carrier modulated with a 0101...sequence. It is insensitive to the frequency offset, as long as the receiver bandwidth is large enough. The size of detection can be programmed from 1 to 3 bytes with *PreambleDetectorSize* in *RegPreambleDetect* as defined in the next table.

Table 20 Preamble Detector Settings

PreambleDetectorSize	# of Bytes
00	1
01	2 (recommended)
10	3
11	reserved

For normal operation, *PreambleDetectTol* should be set to be set to 10 (0x0A), with a qualifying preamble size of 2 bytes.

The *PreambleDetect* interrupt (either in *RegIrqFlags1* or mapped to a specific DIO) then goes high every time a valid preamble is detected, assuming *PreambleDetectorOn*=1.

The preamble detector can also be used as a gate to ensure that AFC and AGC are performed on valid preamble. See section 4.2.6. for details.

4.2.3.7. Image Rejection Mixer

The SX1276/77/78/79 employs an image rejection mixer (IRM) which, uncalibrated, 35 dB image rejection. A low phase noise PLL is used to perform calibration of the receiver chain. This increases the typical image rejection to 48 dB.

4.2.3.8. Image and RSSI Calibration

An automated process is implemented to calibrate the phase and gain imbalances of I and Q receive paths. This calibration enhances image rejection and improves RSSI precision. It is launched under the following circumstances:

- Automatically at Power On Reset or after a Manual Reset of the chip (refer to section 7.2), only for the Low Frequency front-end, and is performed at 434MHz
- Automatically when a pre-defined temperature change is observed, if the option is enabled. A selectable temperature change, set with *TempThreshold* (5, 10, 15 or 20°C), is detected and reported in *TempChange*, if the temperature monitoring is turned On with *TempMonitorOff*=0.This interrupt flag can be used by the application to launch a new image calibration at a convenient time if *AutoImageCalOn*=0, or immediately when this temperature variation is detected, if *AutoImageCalOn*=1
- Upon user request, by setting bit *ImageCalStart* in *RegImageCal*, when the device is in Standby mode

Notes - The calibration procedure takes approximately 10ms. It is recommended to disable the fully automated (temperature-dependent) calibration, to better control when it is triggered (and avoid unexpected packet losses)

- To perform the calibration, the radio must be temporarily returned to FSK/OOK mode

- The automatic IQ and RSSI calibration done at POR and Reset is only valid at 434 MHz (the value of RegFrf at POR). To improve accuracy of RSSI and image rejection, this calibration should be replicated at the frequency (ies)

of interest, for instance a calibration should be launched with Frf set to 868.3 MHz if the high frequency port supports communication in this frequency band. Conversely if the product is used at 169 MHz, the calibration should be repeated with Frf=169MHz

- FormerTemp and TempChange in SX1276/77/79 are frequency-specific and the IC keeps a copy of these variables when switching between the low frequency and the high frequency domains (along with the corresponding calibration values, stored in test registers)

- FormerTemp and TempChange cannot be read in Sleep mode (although they are saved). They should be read in Standby mode

4.2.3.9. Timeout Function

The SX1276/77/78/79 includes a Timeout function, which allows it to automatically shut-down the receiver after a receive sequence and therefore save energy.

- *Timeout* interrupt is generated *TimeoutRxRssi x 16 x Tbit* after switching to Rx mode if the *Rssi* flag does not raise within this time frame (*RssiValue* > *RssiThreshold*)
- *Timeout* interrupt is generated *TimeoutRxPreamble x 16 x Tbit* after switching to Rx mode if the *PreambleDetect* flag does not raise within this time frame
- *Timeout* interrupt is generated *TimeoutSignalSync x 16 x Tbit* after switching to Rx mode if the *SyncAddress* flag does not raise within this time frame

This timeout interrupt can be used to warn the companion processor to shut down the receiver and return to a lower power mode. To become active, these timeouts must also be enabled by setting the correct *RxTrigger* parameters in *RegRxConfig:*

Table 21 RxTrigger Settings to Enable Timeout Interrupts

Receiver Triggering Event	*RxTrigger (2:0)*	*Timeout on Rssi*	*Timeout on Preamble*	*Timeout on SyncAddress*
None	000	Off	Off	Active
Rssi Interrupt	001	Active	Off	
PreambleDetect	110	Off	Active	
Rssi Interrupt & *PreambleDetect*	111	Active	Active	

4.2.4. Operating Modes in FSK/OOK Mode

The SX1276/77/78/79 has several working modes, manually programmed in *RegOpMode*. Fully automated mode selection, packet transmission and reception is also possible using the Top Level Sequencer described in Section 4.2.8.

Table 22 Basic Transceiver Modes

Mode	Selected mode	Symbol	Enabled blocks
000	Sleep mode	Sleep	None
001	Standby mode	Stdby	Top regulator and crystal oscillator
010	Frequency synthesiser to Tx frequency	FSTx	Frequency synthesizer at Tx frequency (Frf)
011	Transmit mode	Tx	Frequency synthesizer and transmitter

Mode	Selected mode	Symbol	Enabled blocks
100	Frequency synthesiser to Rx frequency	FSRx	Frequency synthesizer at frequency for reception (Frf-IF)
101	Receive mode	Rx	Frequency synthesizer and receiver

When switching from a mode to another the sub-blocks are woken up according to a pre-defined optimized sequence.

4.2.5. Startup Times

The startup time of the transmitter or the receiver is Dependant upon which mode the transceiver was in at the beginning. For a complete description, Figure 17 below shows a complete startup process, from the lower power mode “Sleep”.

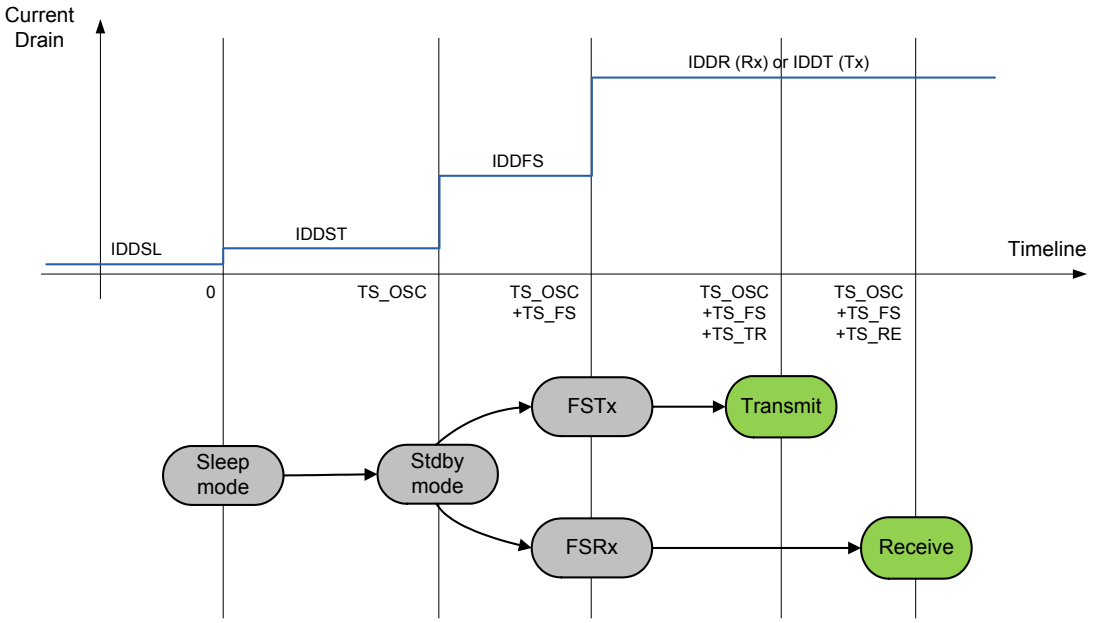


Figure 17. Startup Process

TS_OSC is the startup time of the crystal oscillator which depends on the electrical characteristics of the crystal. TS_FS is the startup time of the PLL including systematic calibration of the VCO.

Typical values of TS_OSC and TS_FS are given in Section 2.5.2.

4.2.5.1. Transmitter Startup Time

The transmitter startup time, TS_TR, is calculated as follows in FSK mode:

$$TS\_TR = 5\mu s + 1.25 \times PaRamp + \frac{1}{2} \times Tbit$$

,

where *PaRamp* is the ramp-up time programmed in *RegPaRamp* and *Tbit* is the bit time.

In OOK mode, this equation can be simplified to the following:

$$TS\_TR = 5\mu s + \frac{1}{2} \times Tbit$$

4.2.5.2. Receiver Startup Time

The receiver startup time, TS_RE, only depends upon the receiver bandwidth effective at the time of startup. When AFC is enabled (*AfcAutoOn*=1), *AfcBw* should be used instead of *RxBw* to extract the receiver startup time:

Table 23 Receiver Startup Time Summary

RxBw if *AfcAutoOn=0* *RxBwAfc* if *AfcAutoOn=1*	TS_RE (+/-5%)
2.6 kHz	2.33 ms
3.1 kHz	1.94 ms
3.9 kHz	1.56 ms
5.2 kHz	1.18 ms
6.3 kHz	984 us
7.8 kHz	791 us
10.4 kHz	601 us
12.5 kHz	504 us
15.6 kHz	407 us
20.8 kHz	313 us
25.0 kHz	264 us
31.3 kHz	215 us
41.7 kHz	169 us
50.0 kHz	144 us
62.5 kHz	119 us
83.3 kHz	97 us
100.0 kHz	84 us
125.0 kHz	71 us
166.7 kHz	85 us
200.0 kHz	74 us
250.0 kHz	63 us

TS_RE or later after setting the device in Receive mode, any incoming packet will be detected and demodulated by the transceiver.

4.2.5.3. Time to RSSI Evaluation

The first RSSI sample will be available TS_RSSI after the receiver is ready, in other words TS_RE + TS_RSSI after the receiver was requested to turn on.

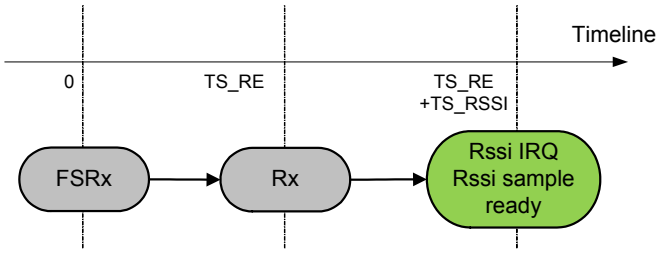


Figure 18. Time to RSSI Sample

TS_RSSI depends on the receiver bandwidth, as well as the *RssiSmoothing* option that was selected. The formula used to calculate TS_RSSI is provided in section 5.5.4.

4.2.5.4. Tx to Rx Turnaround Time

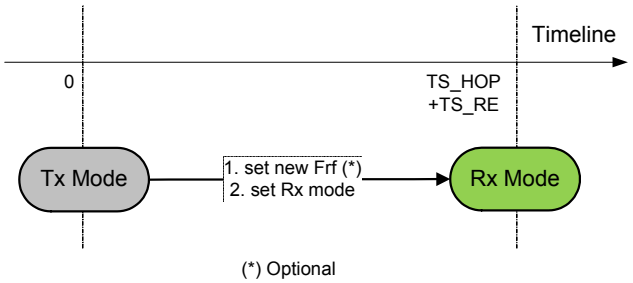


Figure 19. Tx to Rx Turnaround

Note The SPI instruction times are omitted, as they can generally be very small as compared to other timings (up to 10MHz SPI clock).

4.2.5.5. Rx to Tx

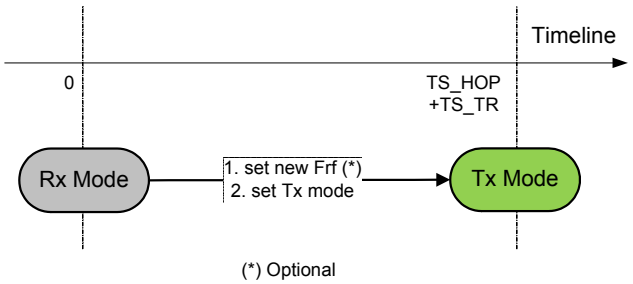


Figure 20. Rx to Tx Turnaround

4.2.5.6. Receiver Hopping, Rx to Rx

Two methods are possible:

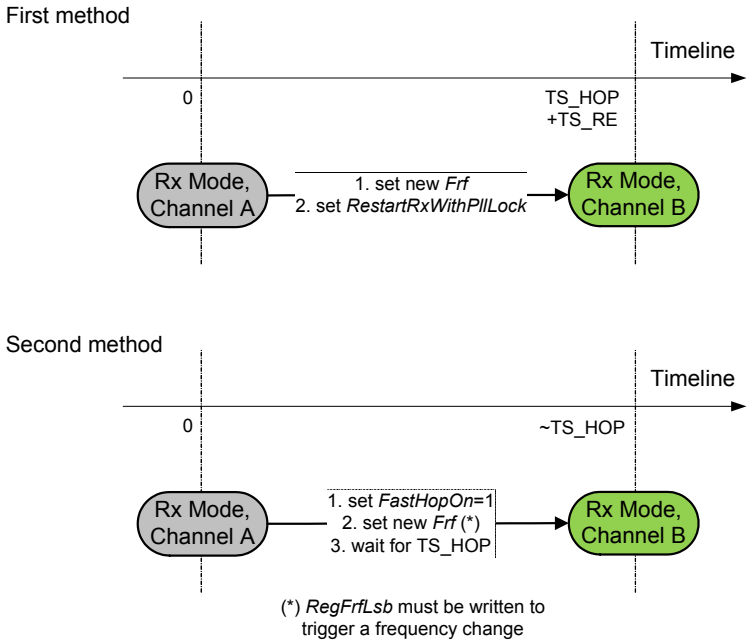


Figure 21. Receiver Hopping

The second method is quicker, and should be used if a very quick RF sniffing mechanism is to be implemented.

4.2.5.7. Tx to Tx

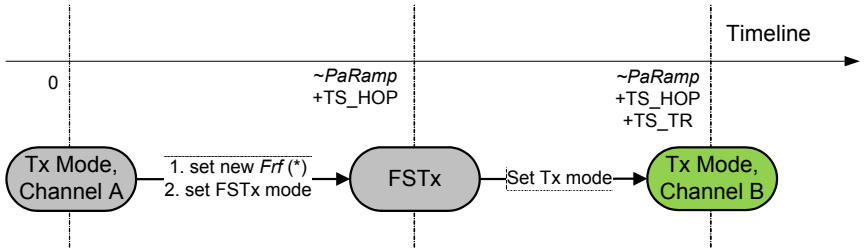


Figure 22. Transmitter Hopping

4.2.6. Receiver Startup Options

The SX1276/77/78/79 receiver can automatically control the gain of the receive chain (AGC) and adjust the receiver LO frequency (AFC). Those processes are carried out on a packet-by-packet basis. They occur:

- When the receiver is turned On.
- When the Receiver is restarted upon user request, through the use of trigger bits *RestartRxWithoutPllLock* or *RestartRxWithPllLock*, in *RegRxConfig*.
- When the receiver is automatically restarted after the reception of a valid packet, or after a packet collision.

Automatic restart capabilities are detailed in Section 4.2.7.

The receiver startup options available in SX1276/77/78/79 are described in Table 24.

Table 24 Receiver Startup Options

Triggering Event	Realized Function	*AgcAutoOn*	*AfcAutoOn*	*RxTrigger (2:0)*
None	None	0	0	000
Rssi Interrupt	AGC	1	0	001
	AGC & AFC	1	1	001
PreambleDetect	AGC	1	0	110
	AGC & AFC	1	1	110
Rssi Interrupt & *PreambleDetect*	AGC	1	0	111
	AGC & AFC	1	1	111

When *AgcAutoOn*=0, the LNA gain is manually selected by choosing *LnaGain* bits in *RegLna.*

4.2.7. Receiver Restart Methods

The options for restart of the receiver are covered below. This is typically of use to prepare for the reception of a new signal whose strength or carrier frequency is different from the preceding packet to allow the AGC or AFC to be re-evaluated.

4.2.7.1. Restart Upon User Request

In Receive mode the user can request a receiver restart - this can be useful in conjunction with the use of a Timeout interrupt following a period of inactivity in the channel of interest. Two options are available:

- No change in the Local Oscillator upon restart: the AFC is disabled, and the *Frf* register has not been changed through SPI before the restart instruction: set bit *RestartRxWithoutPllLock* in *RegRxConfig* to 1.
- Local Oscillator change upon restart: if AFC is enabled (*AfcAutoOn*=1), and/or the *Frf* register had been changed during the last Rx period: set bit *RestartRxWithPllLock* in *RegRxConfig* to 1.

Note ModeReady must be at logic level 1 for a new RestartRx command to be taken into account.

4.2.7.2. Automatic Restart after valid Packet Reception

The bits *AutoRestartRxMode* in *RegSyncConfig* control the automatic restart feature of the SX1276/77/78/79 receiver, when a valid packet has been received:

- If *AutoRestartRxMode* = 00, the function is off, and the user should manually restart the receiver upon valid packet reception (see section 4.2.7.1*).*
- If *AutoRestartRxMode* = 01, after the user has emptied the FIFO following a *PayloadReady* interrupt, the receiver will automatically restart itself after a delay of *InterPacketRxDelay*, allowing for the distant transmitter to ramp down, hence avoiding a false RSSI detection on the ‘tail’ of the previous packet.
- If *AutoRestartRxMode* = 10 should be used if the next reception is expected on a new frequency, i.e. *Frf* is changed after the reception of the previous packet. An additional delay is systematically added, in order for the PLL to lock at a new frequency.

4.2.7.3. Automatic Restart when Packet Collision is Detected

In receive mode the SX1276/77/78/79 is able to detect packet collision and restart the receiver. Collisions are detected by a sudden rise in received signal strength, detected by the RSSI. This functionality can be useful in network configurations where many asynchronous slaves attempt periodic communication with a single a master node.

The collision detector is enabled by setting bit *RestartRxOnCollision* to 1.

The decision to restart the receiver is based on the detection of RSSI change. The sensitivity of the system can be adjusted in 1 dB steps by using register *RssiCollisionThreshold* in *RegRxConfig*.

4.2.8. Top Level Sequencer

Depending on the application, it is desirable to be able to change the mode of the circuit according to a predefined sequence without access to the serial interface. In order to define different sequences or scenarios, a user-programmable state machine, called Top Level Sequencer (Sequencer in short), can automatically control the chip modes.

NOTE THAT THIS FUNCTIONALITY IS ONLY AVAILABLE IN FSK/OOK MODE.

The Sequencer is activated by setting the *SequencerStart* bit in *RegSeqConfig1* to 1 in Sleep or Standby mode (called initial mode).

It is also possible to force the Sequencer off by setting the *Stop* bit in *RegSeqConfig1* to 1 at any time.

Note SequencerStart and Stop bit must never be set at the same time.

4.2.8.1. Sequencer States

As shown in the table below, with the aid of a pair of interrupt timers (T1 and T2), the sequencer can take control of the chip operation in all modes.

Table 25 Sequencer States

Sequencer State	Description
SequencerOff State	The Sequencer is not activated. Sending a *SequencerStart* command will launch it. When coming from **LowPowerSelection** state, the Sequencer will be Off, whilst the chip will return to its initial mode (either Sleep or Standby mode).
Idle State	The chip is in low-power mode, either *Standby* or *Sleep*, as defined by *IdleMode* in *RegSeqConfig1*. The Sequencer waits only for the *T1* interrupt.
Transmit State	The transmitter in on.
Receive State	The receiver in on.
PacketReceived	The receiver is on and a packet has been received. It is stored in the FIFO.
LowPowerSelection	Selects low power state (**SequencerOff** or **Idle** State)
RxTimeout	Defines the action to be taken on a RxTimeout interrupt. RxTimeout interrupt can be a *TimeoutRxRssi*, *TimeoutRxPreamble* or *TimeoutSignalSync* interrupt.

4.2.8.2. Sequencer Transitions

The transitions between sequencer states are listed in the forthcoming table.

Table 26 Sequencer Transition Options

Variable	Transition
IdleMode	Selects the chip mode during **Idle** state: 0: *Standby* mode 1: *Sleep* mode
FromStart	Controls the Sequencer transition when the *SequencerStart* bit is set to 1 in *Sleep* or *Standby* mode: 00: to **LowPowerSelection** 01: to **Receive** state 10: to **Transmit** state 11: to **Transmit** state on a *FifoThreshold* interrupt
LowPowerSelection	Selects Sequencer LowPower state after a *to LowPowerSelection* transition 0: **SequencerOff** state with chip on Initial mode 1: **Idle** state with chip on *Standby* or *Sleep* mode depending on **IdleMode** Note: Initial mode is the chip LowPower mode at Sequencer start.
FromIdle	Controls the Sequencer transition from the **Idle** state on a *T1* interrupt: 0: to **Transmit** state 1: to **Receive** state
FromTransmit	Controls the Sequencer transition from the **Transmit** state: 0: to **LowPowerSelection** on a *PacketSent* interrupt 1: to **Receive** state on a *PacketSent* interrupt
FromReceive	Controls the Sequencer transition from the **Receive** state: 000 and 111: unused 001: to **PacketReceived** state on a *PayloadReady* interrupt 010: to **LowPowerSelection** on a *PayloadReady* interrupt 011: to **PacketReceived** state on a *CrcOk* interrupt. If CRC is wrong (corrupted packet, with CRC on but CrcAutoClearOn is off), the PayloadReady interrupt will drive the sequencer to RxTimeout state. 100: to **SequencerOff** state on a *Rssi* interrupt 101: to **SequencerOff** state on a *SyncAddress* interrupt 110: to **SequencerOff** state on a *PreambleDetect* interrupt Irrespective of this setting, transition to **LowPowerSelection** on a *T2* interrupt
FromRxTimeout	Controls the state-machine transition from the **Receive** state on a *RxTimeout* interrupt (and on *PayloadReady* if **FromReceive** = 011): 00: to **Receive** state via *ReceiveRestart* 01: to **Transmit** state 10: to **LowPowerSelection** 11: to **SequencerOff** state Note: RxTimeout interrupt is a *TimeoutRxRssi*, *TimeoutRxPreamble* or *TimeoutSignalSync* interrupt.
FromPacketReceived	Controls the state-machine transition from the **PacketReceived** state: 000: to **SequencerOff** state 001: to **Transmit** on a *FifoEmpty* interrupt 010: to **LowPowerSelection** 011: to **Receive** via *FS* mode, if frequency was changed 100: to **Receive** state (no frequency change)

4.2.8.3. Timers

Two timers (Timer1 and Timer2) are also available in order to define periodic sequences. These timers are used to generate interrupts, which can trigger transitions of the Sequencer.

T1 interrupt is generated (Timer1Resolution * Timer1Coefficient) after ***T2* interrupt** or ***SequencerStart***. command.
T2 interrupt is generated (Timer2Resolution * Timer2Coefficient) after ***T1* interrupt**.

The timers' mechanism is summarized on the following diagram.

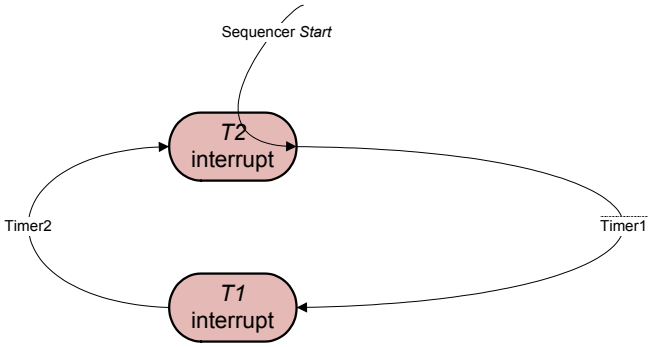


Figure 23. Timer1 and Timer2 Mechanism

Note The timer sequence is completed independently of the actual Sequencer state. Thus, both timers need to be on to achieve periodic cycling.

Table 27 Sequencer Timer Settings

Variable	Description
Timer1Resolution	Resolution of Timer1 00: disabled 01: 64 us 10: 4.1 ms 11: 262 ms
Timer2Resolution	Resolution of Timer2 00: disabled 01: 64 us 10: 4.1 ms 11: 262 ms
Timer1Coefficient	Multiplying coefficient for Timer1
Timer2Coefficient	Multiplying coefficient for Timer2

4.2.8.4. Sequencer State Machine

The following graphs summarize every possible transition between each Sequencer state. The Sequencer states are highlighted in grey. The transitions are represented by arrows. The condition activating them is described over the transition arrow. For better readability, the start transitions are separated from the rest of the graph.

Transitory states are highlighted in light grey, and exit states are represented in red. It is also possible to force the Sequencer off by setting the *Stop* bit in *RegSeqConfig1* to 1 at any time.

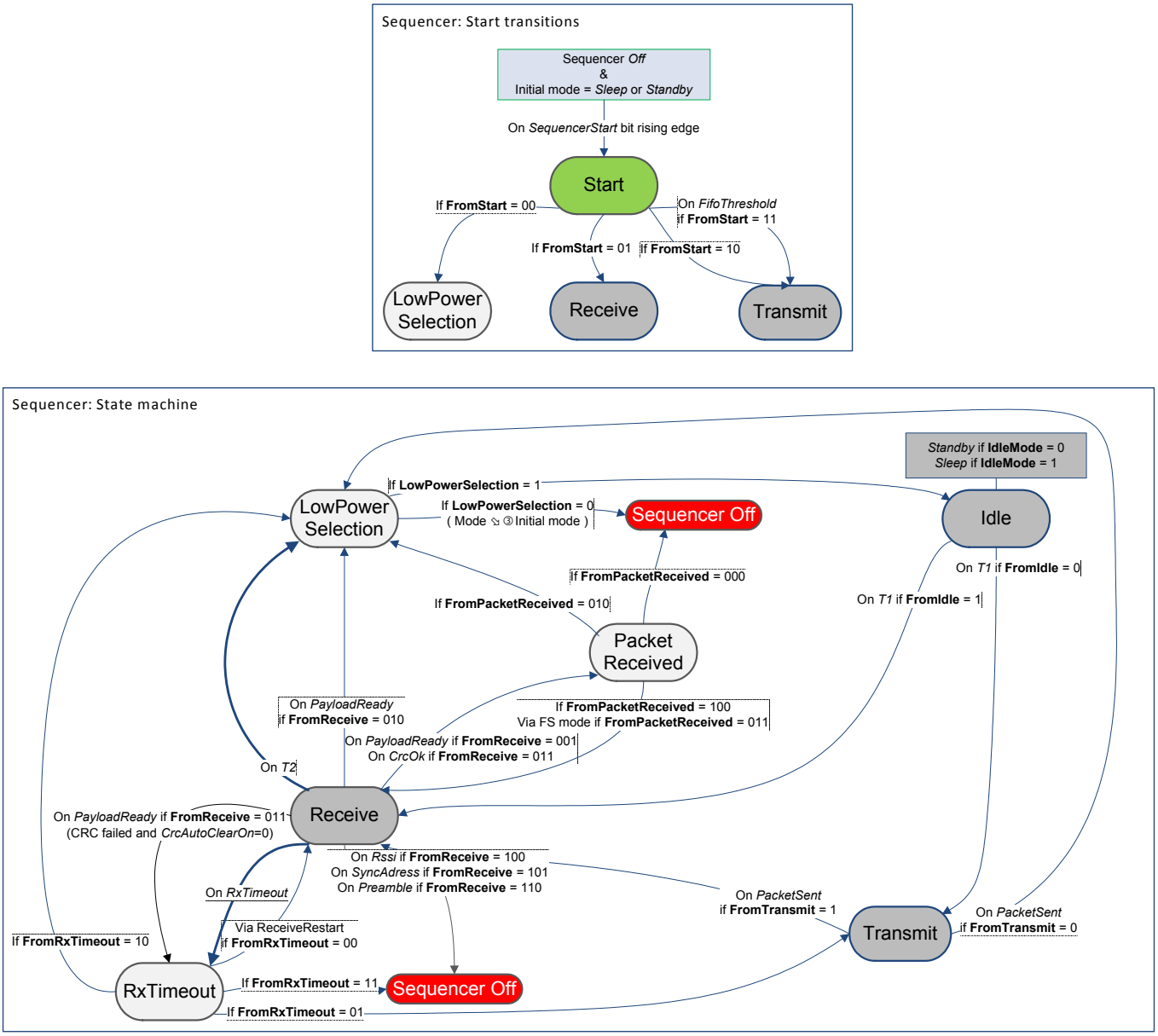


Figure 24. Sequencer State Machine

4.2.9. Data Processing in FSK/OOK Mode

4.2.9.1. Block Diagram

Figure below illustrates the SX1276/77/78/79 data processing circuit. Its role is to interface the data to/from the modulator/ demodulator and the uC access points (SPI and DIO pins). It also controls all the configuration registers.

The circuit contains several control blocks which are described in the following paragraphs.

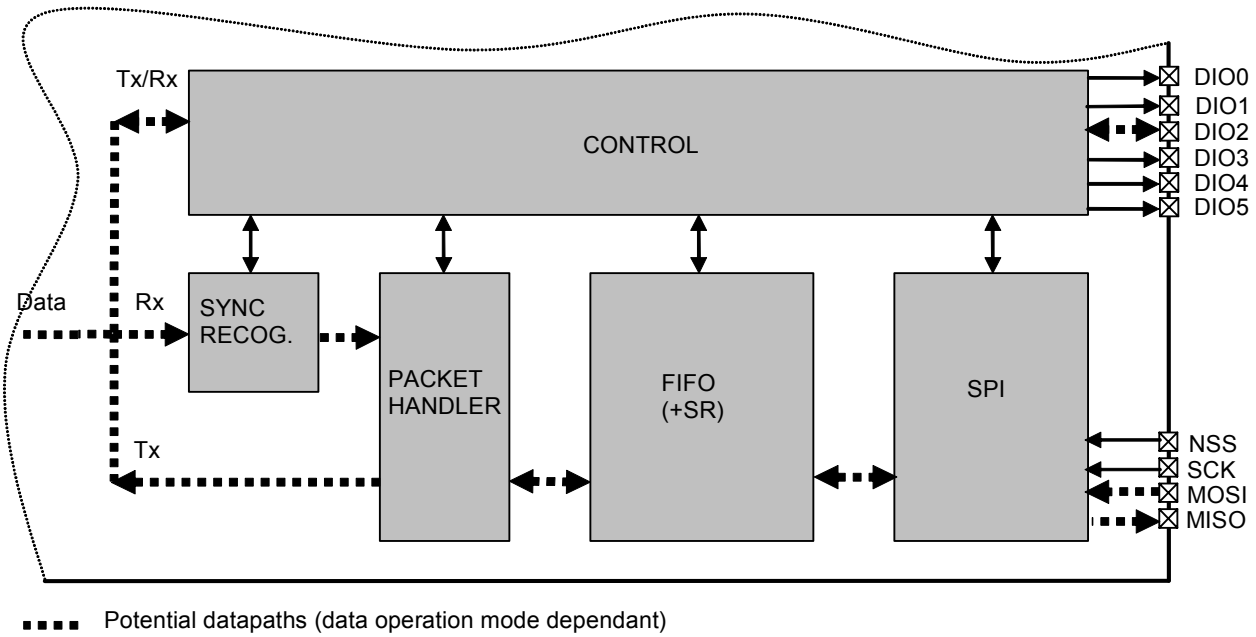


Figure 25. SX1276/77/78/79 Data Processing Conceptual View

The SX1276/77/78/79 implements several data operation modes, each with their own data path through the data processing. Depending on the data operation mode selected, some control blocks are active whilst others remain disabled.

4.2.9.2. Data Operation Modes

The SX1276/77/78/79 has two different data operation modes selectable by the user:

- Continuous mode: each bit transmitted or received is accessed in real time at the DIO2/DATA pin. This mode may be used if adequate external signal processing is available.
- Packet mode (recommended): user only provides/retrieves payload bytes to/from the FIFO. The packet is automatically built with preamble, Sync word, and optional CRC and DC-free encoding schemes The reverse operation is performed in reception. The uC processing overhead is hence significantly reduced compared to Continuous mode. Depending on the optional features activated (CRC, etc) the maximum payload length is limited to 255, 2047 bytes or unlimited.

Each of these data operation modes is fully described in the following s.

4.2.10. FIFO

Overview and Shift Register (SR)

In packet mode of operation, both data to be transmitted and that has been received are stored in a configurable FIFO (First In First Out) device. It is accessed via the SPI interface and provides several interrupts for transfer management.

The FIFO is 1 byte wide hence it only performs byte (parallel) operations, whereas the demodulator functions serially. A shift register is therefore employed to interface the two devices. In transmit mode it takes bytes from the FIFO and outputs them serially (MSB first) at the programmed bit rate to the modulator. Similarly, in Rx the shift register gets bit by bit data from the demodulator and writes them byte by byte to the FIFO. This is illustrated in figure below.

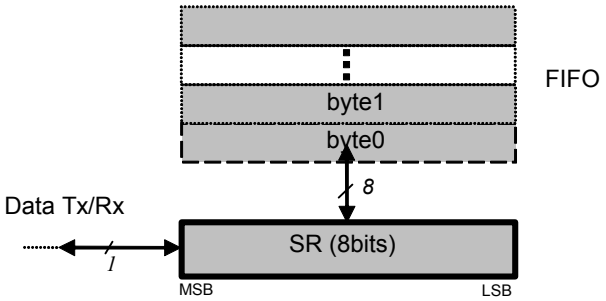


Figure 26. FIFO and Shift Register (SR)

Note When switching to Sleep mode, the FIFO can only be used once the ModeReady flag is set (quasi immediate from all modes except from Tx)

The FIFO size is fixed to 64 bytes.

Interrupt Sources and Flags

- *FifoEmpty*: *FifoEmpty* interrupt source is high when byte 0, i.e. whole FIFO, is empty. Otherwise it is low. Note that when retrieving data from the FIFO, *FifoEmpty* is updated on NSS falling edge, i.e. when *FifoEmpty* is updated to low state the currently started read operation must be completed. In other words, *FifoEmpty* state must be checked after each read operation for a decision on the next one (*FifoEmpty* = 0: more byte(s) to read; *FifoEmpty* = 1: no more byte to read).
- *FifoFull*: *FifoFull* interrupt source is high when the last FIFO byte, i.e. the whole FIFO, is full. Otherwise it is low.
- *FifoOverrunFlag*: *FifoOverrunFlag* is set when a new byte is written by the user (in Tx or Standby modes) or the SR (in Rx mode) while the FIFO is already full. Data is lost and the flag should be cleared by writing a 1, note that the FIFO will also be cleared.
- *PacketSent*: *PacketSent* interrupt source goes high when the SR's last bit has been sent.
- *FifoLevel*: Threshold can be programmed by *FifoThreshold* in *RegFifoThresh*. Its behavior is illustrated in figure below.

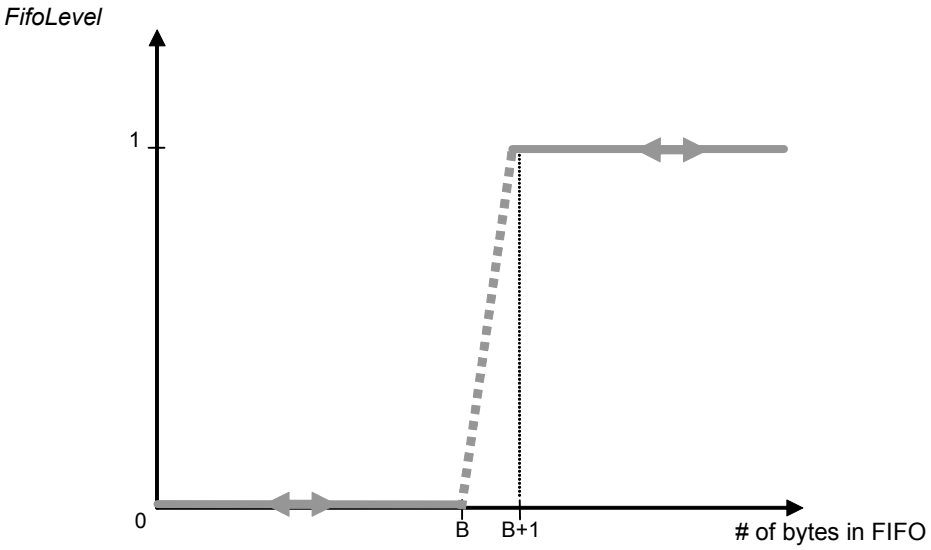


Figure 27. FifoLevel IRQ Source Behavior

Notes - FifoLevel interrupt is updated only after a read or write operation on the FIFO. Thus the interrupt cannot be dynamically updated by only changing the FifoThreshold parameter

- FifoLevel interrupt is valid as long as FifoFull does not occur. An empty FIFO will restore its normal operation

FIFO Clearing

Table below summarizes the status of the FIFO when switching between different modes

Table 28 Status of FIFO when Switching Between Different Modes of the Chip

From	To	FIFO status	Comments
Stdby	Sleep	Not cleared	
Sleep	Stdby	Not cleared	
Stdby/Sleep	Tx	Not cleared	To allow the user to write the FIFO in Stdby/Sleep before Tx
Stdby/Sleep	Rx	Cleared	
Rx	Tx	Cleared	
Rx	Stdby/Sleep	Not cleared	To allow the user to read FIFO in Stdby/Sleep mode after Rx
Tx	Any	Cleared	

4.2.10.1. Sync Word Recognition

Overview

Sync word recognition (also called Pattern recognition) is activated by setting *SyncOn* in *RegSyncConfig*. The bit synchronizer must also be activated in Continuous mode (automatically done in Packet mode).

The block behaves like a shift register; it continuously compares the incoming data with its internally programmed Sync word and sets *SyncAddressMatch* when a match is detected. This is illustrated in Figure 28 below.

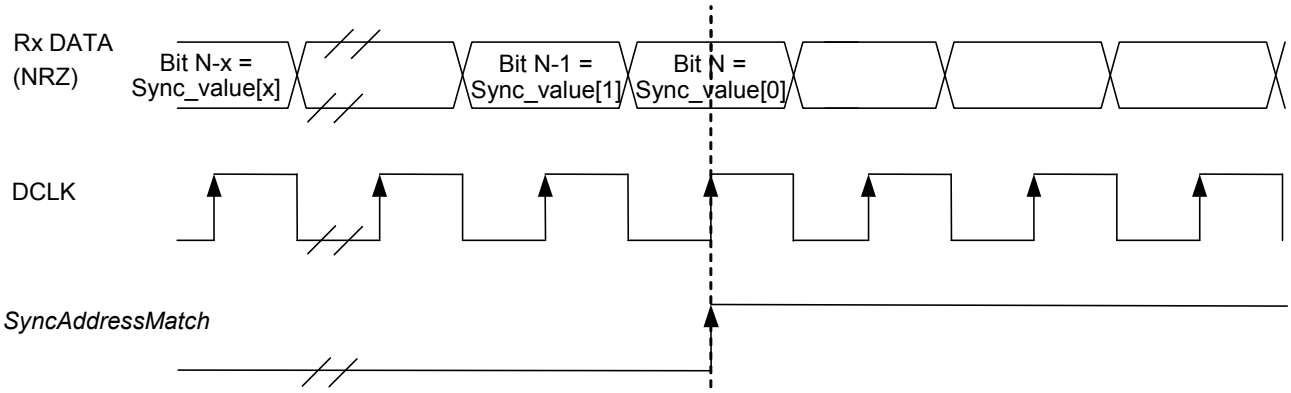


Figure 28. Sync Word Recognition

During the comparison of the demodulated data, the first bit received is compared with bit 7 (MSB) of *RegSyncValue1* and the last bit received is compared with bit 0 (LSB) of the last byte whose address is determined by the length of the Sync word.

When the programmed Sync word is detected the user can assume that this incoming packet is for the node and can be processed accordingly.

SyncAddressMatch is cleared when leaving Rx or FIFO is emptied.

Configuration

- Size: Sync word size can be set from 1 to 8 bytes (i.e. 8 to 64 bits) via *SyncSize* in *RegSyncConfig*. In Packet mode this field is also used for Sync word generation in Tx mode.
- Value: The Sync word value is configured in *SyncValue(63:0)*. In Packet mode this field is also used for Sync word generation in Tx mode.

Note SyncValue choices containing 0x00 bytes are not allowed

Packet Handler

The packet handler is the block used in Packet mode. Its functionality is fully described in Section 4.2.13.

Control

The control block configures and controls the full chip's behavior according to the settings programmed in the configuration registers.

4.2.11. Digital IO Pins Mapping

Six general purpose IO pins are available on the SX1276/77/78/79, and their configuration in Continuous or Packet mode is controlled through *RegDioMapping1* and *RegDioMapping2.*

Table 29 DIO Mapping, Continuous Mode

	DIOx Mapping	Sleep	Standby	FSRx/Tx	Rx	Tx
DIO0	00	-			SyncAddress	TxReady
	01	-			Rssi / PreambleDetect	-
	10	-			RxReady	TxReady
	11	-				
DIO1	00	-			Dclk	
	01	-			Rssi / PreambleDetect	-
	10	-				
	11	-				
DIO2	00	-			Data	
	01	-			Data	
	10	-			Data	
	11	-			Data	
DIO3	00	-			Timeout	-
	01	-			Rssi / PreambleDetect	-
	10	-				
	11	-	TempChange / LowBat		TempChange / LowBat	
DIO4	00	-		TempChange / LowBat		
	01	-		PllLock		
	10	-			TimeOut	-
	11	-	ModeReady		ModeReady	
DIO5	00	ClkOut if RC	ClkOut		ClkOut	
	01	-		PllLock		
	10	-			Rssi / PreambleDetect	-
	11	-	ModeReady		ModeReady	

Table 30 DIO Mapping, Packet Mode

	DIOx Mapping	Sleep	Standby	FSRx/Tx	Rx	Tx
DIO0	00	-			PayloadReady	PacketSent
	01	-			CrcOk	-
	10	-				
	11	-	TempChange / LowBat		TempChange / LowBat	
DIO1	00	FifoLevel		FifoLevel	FifoLevel	
	01	FifoEmpty		FifoEmpty	FifoEmpty	
	10	FifoFull		FifoFull	FifoFull	
	11	-				
DIO2	00	FifoFull		FifoFull	FifoFull	
	01	-			RxReady	-
	10	FifoFull			TimeOut	FifoFull
	11	FifoFull			SyncAddress	FifoFull
DIO3	00	FifoEmpty		FifoEmpty	FifoEmpty	
	01	-				TxReady
	10	FifoEmpty		FifoEmpty	FifoEmpty	
	11	FifoEmpty		FifoEmpty	FifoEmpty	
DIO4	00	-	TempChange / LowBat		TempChange / LowBat	
	01	-		PllLock		
	10	-			TimeOut	-
	11	-			Rssi / PreambleDetect	-
DIO5	00	ClkOut if RC	ClkOut		ClkOut	
	01	-		PllLock		
	10	-			Data	
	11	-	ModeReady		ModeReady	

4.2.12. Continuous Mode

4.2.12.1. General Description

As illustrated in Figure 29, in Continuous mode the NRZ data to (from) the (de)modulator is directly accessed by the uC on the bidirectional DIO2/DATA pin. The FIFO and packet handler are thus inactive.

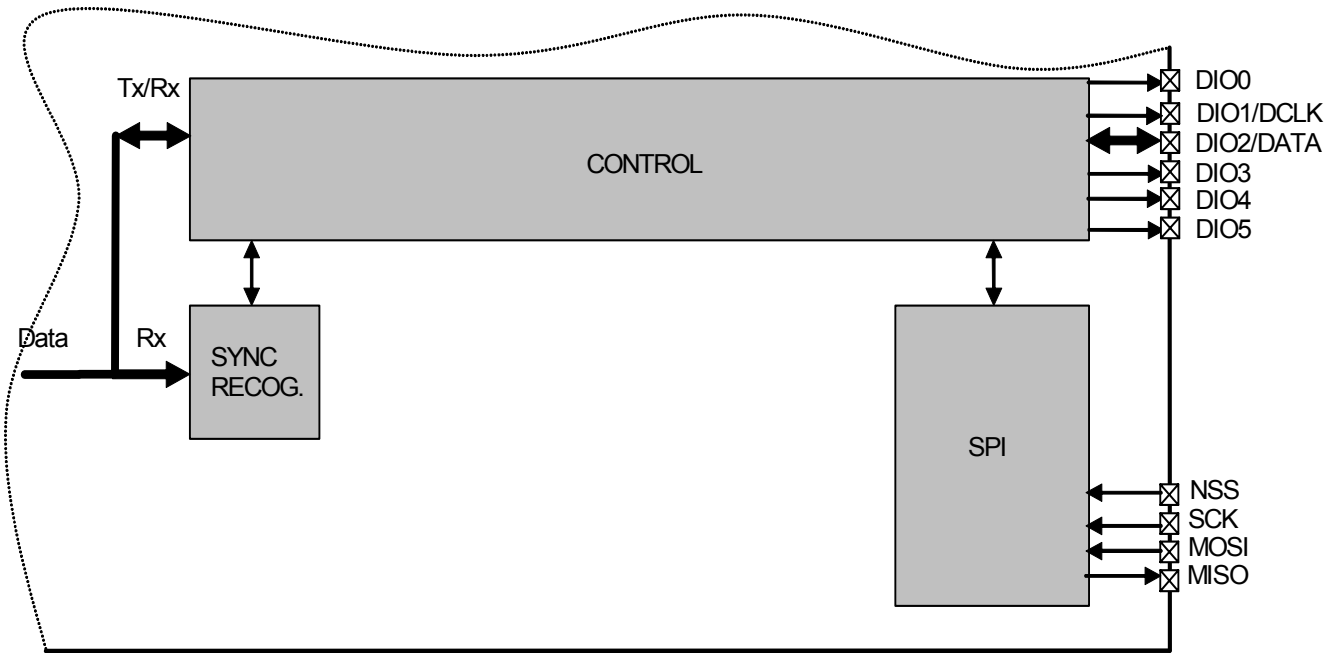


Figure 29. Continuous Mode Conceptual View

4.2.12.2. Tx Processing

In Tx mode, a synchronous data clock for an external uC is provided on DIO1/DCLK pin. Clock timing with respect to the data is illustrated in Figure 30. DATA is internally sampled on the rising edge of DCLK so the uC can change logic state anytime outside the grayed out setup/hold zone.

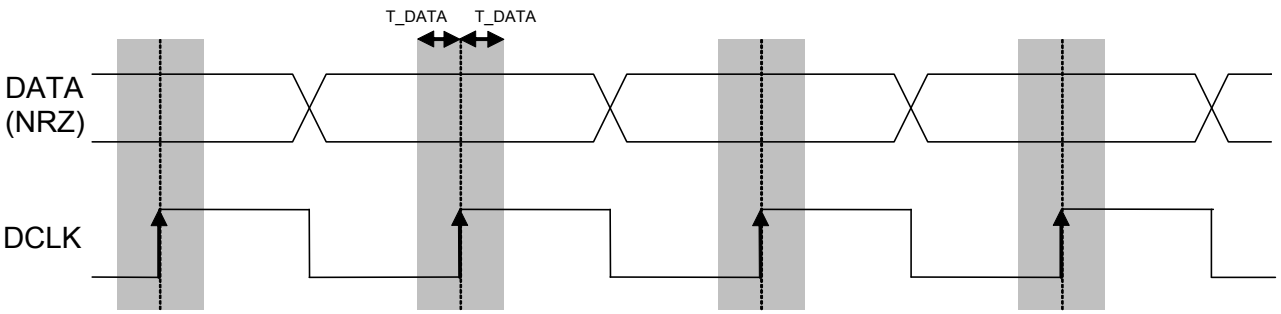


Figure 30. Tx Processing in Continuous Mode

Note the use of DCLK is required when the modulation shaping is enabled.

4.2.12.3. Rx Processing

If the bit synchronizer is disabled, the raw demodulator output is made directly available on DATA pin and no DCLK signal is provided.

Conversely, if the bit synchronizer is enabled, synchronous cleaned data and clock are made available respectively on DIO2/DATA and DIO1/DCLK pins. DATA is sampled on the rising edge of DCLK and updated on the falling edge as illustrated below.

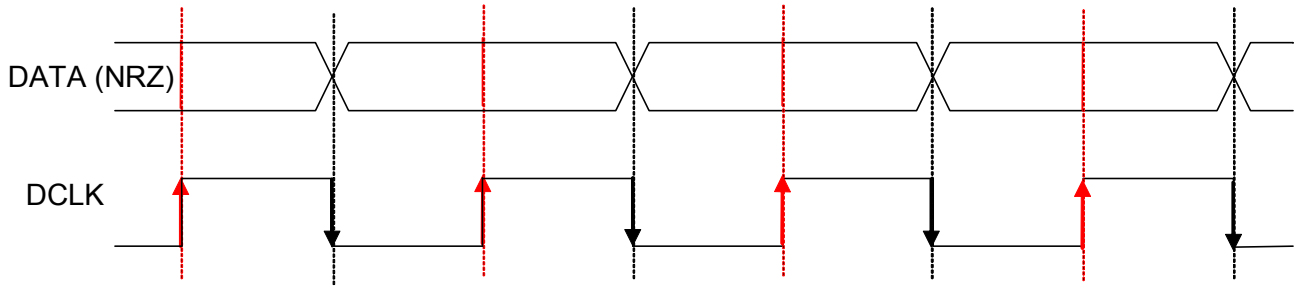


Figure 31. Rx Processing in Continuous Mode

Note In Continuous mode it is always recommended to enable the bit synchronizer to clean the DATA signal even if the DCLK signal is not used by the uC (bit synchronizer is automatically enabled in Packet mode).

4.2.13. Packet Mode

4.2.13.1. General Description

In Packet mode the NRZ data to (from) the (de)modulator is not directly accessed by the uC but stored in the FIFO and accessed via the SPI interface.

In addition, the SX1276/77/78/79 packet handler performs several packet oriented tasks such as Preamble and Sync word generation, CRC calculation/check, whitening/dewhitening of data, Manchester encoding/decoding, address filtering, etc. This simplifies software and reduces uC overhead by performing these repetitive tasks within the RF chip itself.

Another important feature is ability to fill and empty the FIFO in Sleep/Stdby mode, ensuring optimum power consumption and adding more flexibility for the software.

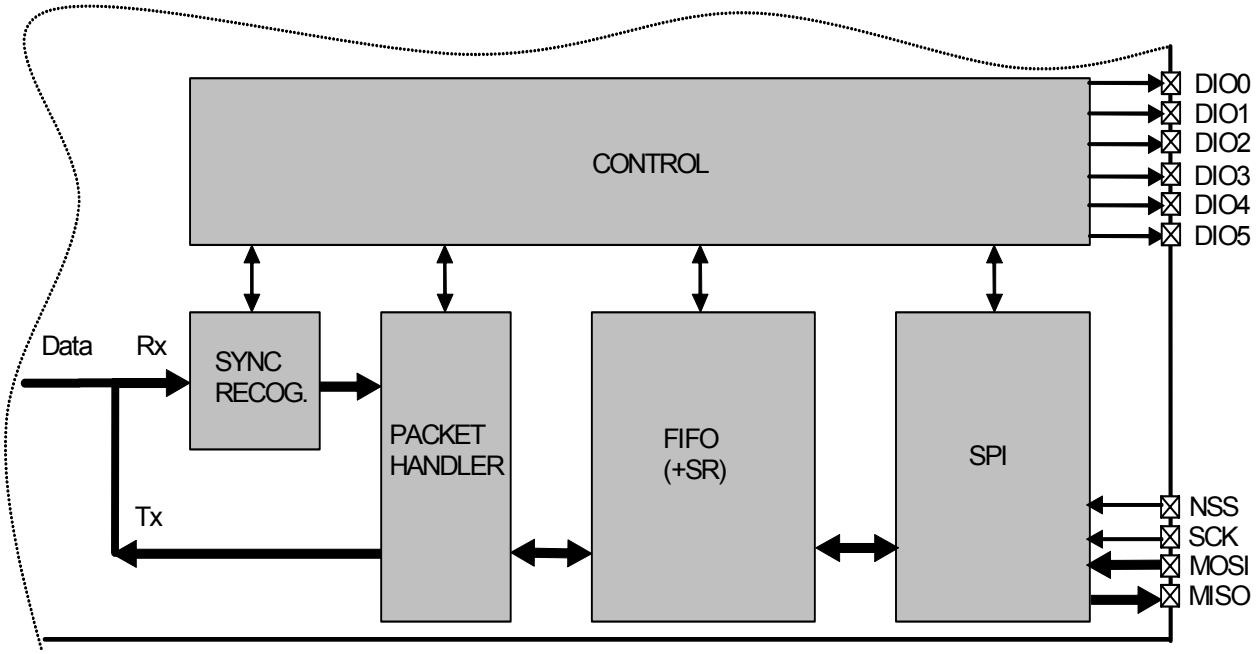


Figure 32. Packet Mode Conceptual View

Note The Bit Synchronizer is automatically enabled in Packet mode.

4.2.13.2. Packet Format

Fixed Length Packet Format

Fixed length packet format is selected when bit *PacketFormat* is set to 0 and *PayloadLength* is set to any value greater than 0.

In applications where the packet length is fixed in advance, this mode of operation may be of interest to minimize RF overhead (no length byte field is required). All nodes, whether Tx only, Rx only, or Tx/Rx should be programmed with the same packet length value.

The length of the payload is limited to 2047 bytes.

The length programmed in *PayloadLength* relates only to the payload which includes the message and the optional address byte. In this mode, the payload must contain at least one byte, i.e. address or message byte.

An illustration of a fixed length packet is shown below. It contains the following fields:

- Preamble (1010...)
- Sync word (Network ID)
- Optional Address byte (Node ID)
- Message data
- Optional 2-bytes CRC checksum

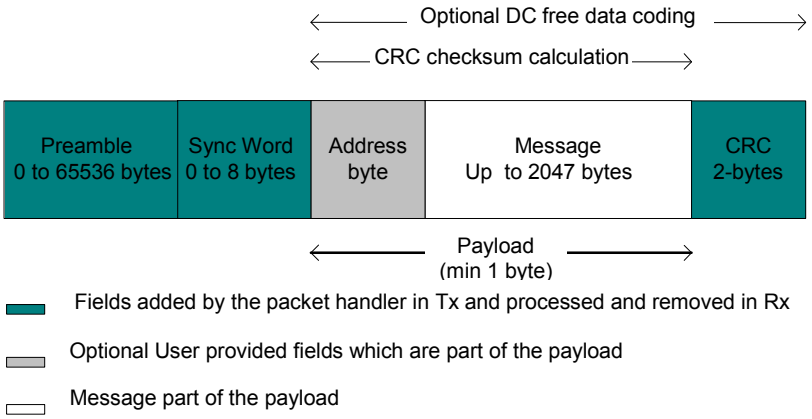


Figure 33. Fixed Length Packet Format

Variable Length Packet Format

Variable length packet format is selected when bit *PacketFormat* is set to 1.

This mode is useful in applications where the length of the packet is not known in advance and can vary over time. It is then necessary for the transmitter to send the length information together with each packet in order for the receiver to operate properly.

In this mode the length of the payload, indicated by the length byte, is given by the first byte of the FIFO and is limited to 255 bytes. Note that the length byte itself is not included in its calculation. In this mode, the payload must contain at least 2 bytes, i.e. length + address or message byte.

An illustration of a variable length packet is shown below. It contains the following fields:

- Preamble (1010...)
- Sync word (Network ID)
- Length byte
- Optional Address byte (Node ID)
- Message data

- Optional 2-bytes CRC checksum

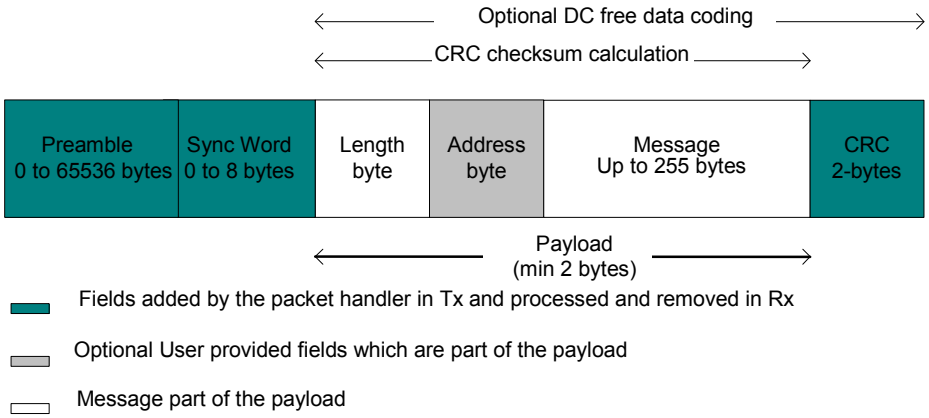


Figure 34. Variable Length Packet Format

Unlimited Length Packet Format

Unlimited length packet format is selected when bit *PacketFormat* is set to 0 and *PayloadLength* is set to 0. The user can then transmit and receive packet of arbitrary length and *PayloadLength* register is not used in Tx/Rx modes for counting the length of the bytes transmitted/received.

In Tx the data is transmitted depending on the *TxStartCondition* bit. On the Rx side the data processing features like Address filtering, Manchester encoding and data whitening are not available if the sync pattern length is set to zero (*SyncOn = 0*). The CRC detection in Rx is also not supported in this mode of the packet handler, however CRC generation in Tx is operational. The interrupts like *CrcOk* & *PayloadReady* are not available either.

An unlimited length packet shown below is made up of the following fields:

- Preamble (1010...).
- Sync word (Network ID).
- Optional Address byte (Node ID).
- Message data
- Optional 2-bytes CRC checksum (Tx only)

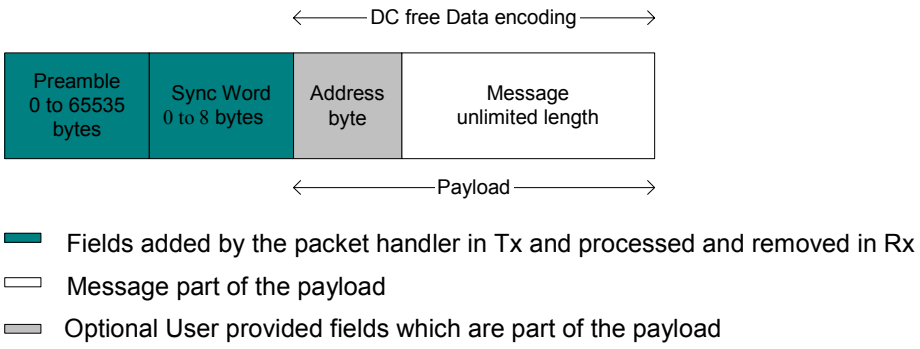


Figure 35. Unlimited Length Packet Format

4.2.13.3. Tx Processing

In Tx mode the packet handler dynamically builds the packet by performing the following operations on the payload available in the FIFO:

- Add a programmable number of preamble bytes
- Add a programmable Sync word
- Optionally calculating CRC over complete payload field (optional length byte + optional address byte + message) and appending the 2 bytes checksum.
- Optional DC-free encoding of the data (Manchester or whitening)

Only the payload (including optional address and length fields) is required to be provided by the user in the FIFO.

The transmission of packet data is initiated by the Packet Handler only if the chip is in Tx mode and the transmission condition defined by *TxStartCondition* is fulfilled. If transmission condition is not fulfilled then the packet handler transmits a preamble sequence until the condition is met. This happens only if the preamble length /= 0, otherwise it transmits a zero or one until the condition is met to transmit the packet data.

The transmission condition itself is defined as:

- if *TxStartCondition* = 1, the packet handler waits until the first byte is written into the FIFO, then it starts sending the preamble followed by the sync word and user payload
- If *TxStartCondition* = 0, the packet handler waits until the number of bytes written in the FIFO is equal to the number defined in *RegFifoThresh* + 1
- If the condition for transmission was already fulfilled i.e. the FIFO was filled in Sleep/Stdby then the transmission of packet starts immediately on enabling Tx

4.2.13.4. Rx Processing

In Rx mode the packet handler extracts the user payload to the FIFO by performing the following operations:

- Receiving the preamble and stripping it off
- Detecting the Sync word and stripping it off
- Optional DC-free decoding of data
- Optionally checking the address byte
- Optionally checking CRC and reflecting the result on *CrcOk.*

Only the payload (including optional address and length fields) is made available in the FIFO.

When the Rx mode is enabled the demodulator receives the preamble followed by the detection of sync word. If fixed length packet format is enabled then the number of bytes received as the payload is given by the *PayloadLength* parameter.

In variable length mode the first byte received after the sync word is interpreted as the length of the received packet. The internal length counter is initialized to this received length. The *PayloadLength* register is set to a value which is greater than the maximum expected length of the received packet. If the received length is greater than the maximum length stored in *PayloadLength* register the packet is discarded otherwise the complete packet is received.

If the address check is enabled then the second byte received in case of variable length and first byte in case of fixed length is the address byte. If the address matches to the one in the *NodeAddress* field, reception of the data continues otherwise it's stopped. The CRC check is performed if *CrcOn* = 1 and the result is available in *CrcOk* indicating that the

CRC was successful. An interrupt (*PayloadReady*) is also generated on DIO0 as soon as the payload is available in the FIFO. The payload available in the FIFO can also be read in Sleep/Standby mode.

If the CRC fails the *PayloadReady* interrupt is not generated and the FIFO is cleared. This function can be overridden by setting *CrcAutoClearOff* = 1, forcing the availability of *PayloadReady* interrupt and the payload in the FIFO even if the CRC fails.

4.2.13.5. Handling Large Packets

When *PayloadLength* exceeds FIFO size (64 bytes) whether in fixed, variable or unlimited length packet format, in addition to *PacketSent* in Tx and *PayloadReady* or *CrcOk* in Rx, the FIFO interrupts/flags can be used as described below:

- For Tx:

FIFO can be prefilled in Sleep/Standby but must be refilled “on-the-fly” during Tx with the rest of the payload.

1) Pre-fill FIFO (in Sleep/Standby first or directly in Tx mode) until *FifoThreshold* or *FifoFull* is set

2) In Tx, wait for *FifoThreshold* or *FifoEmpty* to be set (i.e. FIFO is nearly empty)

3) Write bytes into the FIFO until *FifoThreshold* or *FifoFull* is set.

4) Continue to step 2 until the entire message has been written to the FIFO (*PacketSent* will fire when the last bit of the packet has been sent).

- For Rx:

FIFO must be unfilled “on-the-fly” during Rx to prevent FIFO overrun.

1) Start reading bytes from the FIFO when *FifoEmpty* is cleared or *FifoThreshold* becomes set.

2) Suspend reading from the FIFO if *FifoEmpty* fires before all bytes of the message have been read

3) Continue to step 1 until *PayloadReady* or *CrcOk* fires

4) Read all remaining bytes from the FIFO either in Rx or Sleep/Standby mode

4.2.13.6. Packet Filtering

The SX1276/77/78/79 packet handler offers several mechanisms for packet filtering, ensuring that only useful packets are made available to the uC, reducing significantly system power consumption and software complexity.

Sync Word Based

Sync word filtering/recognition is used for identifying the start of the payload and also for network identification. As previously described, the Sync word recognition block is configured (size, value) in *RegSyncConfig* and *RegSyncValue(i)* registers. This information is used, both for appending Sync word in Tx, and filtering packets in Rx.

Every received packet which does not start with this locally configured Sync word is automatically discarded and no interrupt is generated.

When the Sync word is detected, payload reception automatically starts and *SyncAddressMatch* is asserted.

Note Sync Word values containing 0x00 byte(s) are forbidden

Address Based

Address filtering can be enabled via the *AddressFiltering* bits. It adds another level of filtering, above Sync word (i.e. Sync must match first), typically useful in a multi-node networks where a network ID is shared between all nodes (Sync word) and each node has its own ID (address).

Two address based filtering options are available:

- *AddressFiltering = 01*: Received address field is compared with internal register *NodeAddress*. If they match then the packet is accepted and processed, otherwise it is discarded.
- *AddressFiltering = 10*: Received address field is compared with internal registers *NodeAddress* and *BroadcastAddress*. If either is a match, the received packet is accepted and processed, otherwise it is discarded. This additional check with a constant is useful for implementing broadcast in a multi-node networks

Please note that the received address byte, as part of the payload, is not stripped off the packet and is made available in the FIFO. In addition, *NodeAddress* and *AddressFiltering* only apply to Rx. On Tx side, if address filtering is expected, the address byte should simply be put into the FIFO like any other byte of the payload.

As address filtering requires a Sync word match, both features share the same interrupt flag *SyncAddressMatch*.

Length Based

In variable length Packet mode, *PayloadLength* must be programmed with the maximum payload length permitted. If received length byte is smaller than this maximum then the packet is accepted and processed, otherwise it is discarded.

Please note that the received length byte, as part of the payload, is not stripped off the packet and is made available in the FIFO.

To disable this function the user should set the value of the *PayloadLength* to 2047.

CRC Based

The CRC check is enabled by setting bit *CrcOn* in *RegPacketConfig1*. It is used for checking the integrity of the message.

- On Tx side a two byte CRC checksum is calculated on the payload part of the packet and appended to the end of the message
- On Rx side the checksum is calculated on the received payload and compared with the two checksum bytes received. The result of the comparison is stored in bit *CrcOk.*

By default, if the CRC check fails then the FIFO is automatically cleared and no interrupt is generated. This filtering function can be disabled via *CrcAutoClearOff* bit and in this case, even if CRC fails, the FIFO is not cleared and only *PayloadReady* interrupt goes high. Please note that in both cases, the two CRC checksum bytes are stripped off by the packet handler and only the payload is made available in the FIFO. Two CRC implementations are selected with bit *CrcWhiteningType*.

Table 31 CRC Description

Crc Type	*CrcWhiteningType*	Polynomial	Seed Value	Complemented
CCITT	0 (default)	$X^{16} + X^{12} + X^5 + 1$	0x1D0F	Yes
IBM	1	$X^{16} + X^{15} + X^2 + 1$	0xFFFF	No

A C code implementation of each CRC type is proposed in Application Section 7.

4.2.13.7. DC-Free Data Mechanisms

The payload to be transmitted may contain long sequences of 1's and 0's, which introduces a DC bias in the transmitted signal. The radio signal thus produced has a non uniform power distribution over the occupied channel bandwidth. It also introduces data dependencies in the normal operation of the demodulator. Thus it is useful if the transmitted data is random and DC free.

For such purposes, two techniques are made available in the packet handler: Manchester encoding and data whitening.

Note Only one of the two methods can be enabled at a time.

Manchester Encoding

Manchester encoding/decoding is enabled if *DcFree = 01* and can only be used in Packet mode.

The NRZ data is converted to Manchester code by coding '1' as “10” and '0' as “01”.

In this case, the maximum chip rate is the maximum bit rate given in the specifications and the actual bit rate is half the chip rate.

Manchester encoding and decoding is only applied to the payload and CRC checksum while preamble and Sync word are kept NRZ. However, the chip rate from preamble to CRC is the same and defined by *BitRate* in *RegBitRate* (Chip Rate = Bit Rate NRZ = 2 x Bit Rate Manchester).

Manchester encoding/decoding is thus made transparent for the user, who still provides/retrieves NRZ data to/from the FIFO.

	1/BR ...Sync								1/BR Payload...											
RF chips @ BR	**...**	**1**	**1**	**1**	**0**	**1**	**0**	**0**	**1**	**0**	**0**	**1**	**0**	**1**	**1**	**0**	**1**	**0**	**...**	**t**
User/NRZ bits Manchester OFF	...	1	1	1	0	1	0	0	1	0	0	1	0	1	1	0	1	0	...	
User/NRZ bits Manchester ON	...	1	1	1	0	1	0	0	1		0		0		1		1		...	

Figure 36. Manchester Encoding/Decoding

Data Whitening

Another technique called whitening or scrambling is widely used for randomizing the user data before radio transmission. The data is whitened using a random sequence on the Tx side and de-whitened on the Rx side using the same sequence. Comparing to Manchester technique it has the advantage of keeping NRZ data rate i.e. actual bit rate is not halved.

The whitening/de-whitening process is enabled if *DcFree = 10*. A 9-bit LFSR is used to generate a random sequence. The payload and 2-byte CRC checksum is then XORed with this random sequence as shown below. The data is de-whitened on the receiver side by XORing with the same random sequence.

Payload whitening/de-whitening is thus made transparent for the user, who still provides/retrieves NRZ data to/from the FIFO.

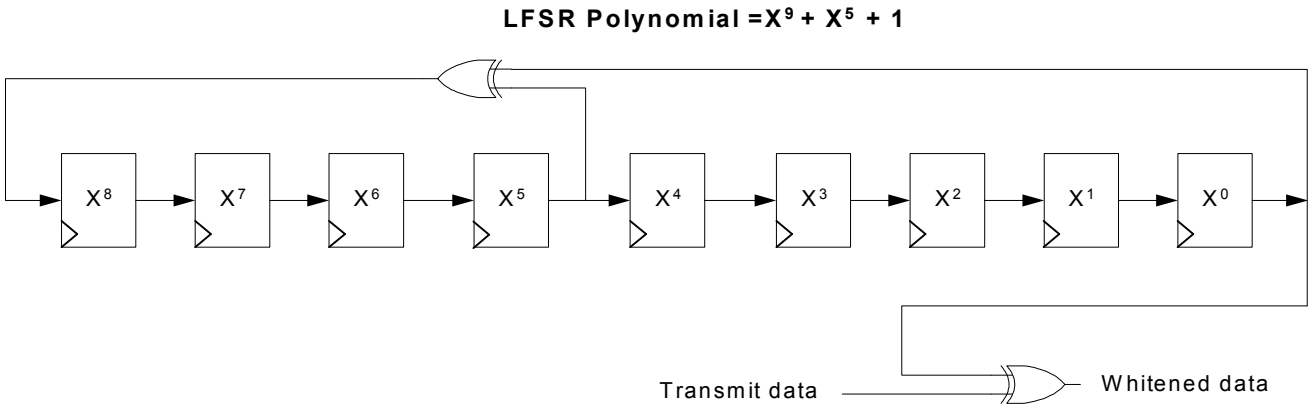


Figure 37. Data Whitening Polynomial

4.2.13.8. Beacon Tx Mode

In some short range wireless network topologies a repetitive message, also known as beacon, is transmitted periodically by a transmitter. The Beacon Tx mode allows for the re-transmission of the same packet without having to fill the FIFO multiple times with the same data.

When *BeaconOn* in *RegPacketConfig2* is set to 1, the FIFO can be filled only once in Sleep or Stdby mode with the required payload. After a first transmission, *FifoEmpty* will go high as usual, but the FIFO content will be restored when the chip exits Transmit mode. *FifoEmpty*, *FifoFull* and *FifoLevel* flags are also restored.

This feature is only available in Fixed packet format, with the Payload Length smaller than the FIFO size. The control of the chip modes (Tx-Sleep-Tx....) can either be undertaken by the microcontroller, or be automated in the Top Sequencer. See example in Section 4.2.13.8.

The Beacon Tx mode is exited by setting *BeaconOn* to 0, and clearing the FIFO by setting *FifoOverrun* to 1.

4.2.14. io-homecontrol® Compatibility Mode

The SX1276/77/78/79 features a io-homecontrol® compatibility mode. Please contact your local Semtech representative for details on its implementation.

4.3. SPI Interface

The SPI interface gives access to the configuration register via a synchronous full-duplex protocol corresponding to CPOL = 0 and CPHA = 0 in Motorola/Freescale nomenclature. Only the slave side is implemented.

Three access modes to the registers are provided:

- SINGLE access: an address byte followed by a data byte is sent for a write access whereas an address byte is sent and a read byte is received for the read access. The NSS pin goes low at the beginning of the frame and goes high after the data byte.
- BURST access: the address byte is followed by several data bytes. The address is automatically incremented internally between each data byte. This mode is available for both read and write accesses. The NSS pin goes low at the beginning of the frame and stay low between each byte. It goes high only after the last byte transfer.
- FIFO access: if the address byte corresponds to the address of the FIFO, then succeeding data byte will address the FIFO. The address is not automatically incremented but is memorized and does not need to be sent between each data byte. The NSS pin goes low at the beginning of the frame and stay low between each byte. It goes high only after the last byte transfer.

The figure below shows a typical SPI single access to a register.

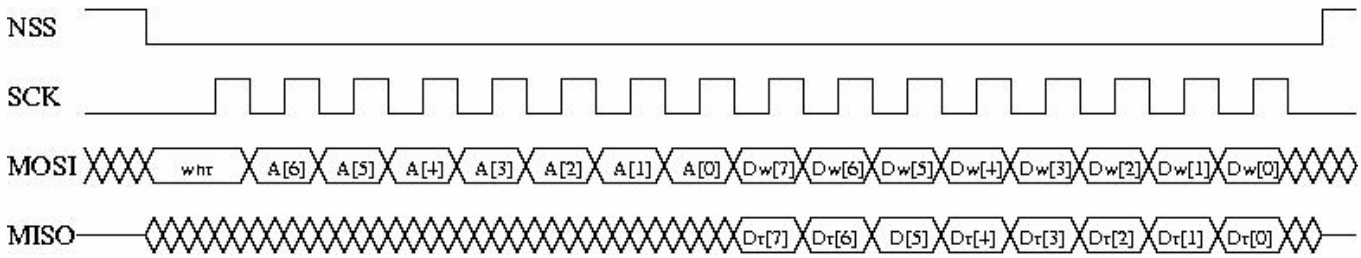


Figure 38. SPI Timing Diagram (single access)

MOSI is generated by the master on the falling edge of SCK and is sampled by the slave (i.e. this SPI interface) on the rising edge of SCK. MISO is generated by the slave on the falling edge of SCK.

A transfer is always started by the NSS pin going low. MISO is high impedance when NSS is high.

The first byte is the address byte. It is comprises:

- A wnr bit, which is 1 for write access and 0 for read access.
- Then 7 bits of address, MSB first.

The second byte is a data byte, either sent on MOSI by the master in case of a write access or received by the master on MISO in case of read access. The data byte is transmitted MSB first.

Proceeding bytes may be sent on MOSI (for write access) or received on MISO (for read access) without a rising NSS edge and re-sending the address. In FIFO mode, if the address was the FIFO address then the bytes will be written / read at the FIFO address. In Burst mode, if the address was not the FIFO address, then it is automatically incremented for each new byte received.

The frame ends when NSS goes high. The next frame must start with an address byte. The SINGLE access mode is therefore a special case of FIFO / BURST mode with only 1 data byte transferred.

During the write access, the byte transferred from the slave to the master on the MISO line is the value of the written register before the write operation.

5. SX1276/77/78/79 Analog & RF Frontend Electronics

5.1. Power Supply Strategy

The SX1276/77/78/79 employs an internal voltage regulation scheme which provides stable operating voltage, and hence device characteristics, over the full industrial temperature and operating voltage range of operation. This includes up to +17 dBm of RF output power which is maintained from 1.8 V to 3.7 V and +20 dBm from 2.4 V to 3.7 V.

The SX1276/77/78/79 can be powered from any low-noise voltage source via pins VBAT_ANA, VBAT_RF and VBAT_DIG. Decoupling capacitors should be connected, as suggested in the reference design of the applications section of this document, on VR_PA, VR_DIG and VR_ANA pins to ensure correct operation of the built-in voltage regulators.

5.2. Low Battery Detector

A low battery detector is also included allowing the generation of an interrupt signal in response to the supply voltage dropping below a programmable threshold that is adjustable through the register *RegLowBat*. The interrupt signal can be mapped to any of the DIO pins by programming *RegDioMapping*.

5.3. Frequency Synthesis

5.3.1. Crystal Oscillator

The crystal oscillator is the main timing reference of the SX1276/77/78/79. It is used as the reference for the PLL's frequency synthesis and as the clock signal for all digital processing.

The crystal oscillator startup time, TS_OSC, depends on the electrical characteristics of the crystal reference used, for more information on the electrical specification of the crystal see section 7.1. The crystal connects to the Pierce oscillator on pins XTA and XTB. The SX1276/77/78/79 optimizes the startup time and automatically triggers the PLL when the oscillator signal is stable.

Optionally, an external clock can be used to replace the crystal oscillator. This typically takes the form of a tight tolerance temperature compensated crystal oscillator (TCXO). When using an external clock source the bit *TcxoInputOn* of register *RegTcxo* should be set to 1 and the external clock has to be provided on XTA (pin 5). XTB (pin 6) should be left open.

The peak-peak amplitude of the input signal must never exceed 1.8 V. Please consult your TCXO supplier for an appropriate value of decoupling capacitor, C_D.

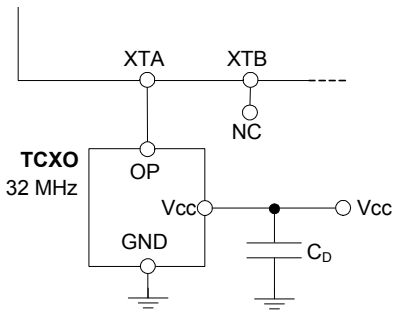


Figure 39. TCXO Connection

5.3.2. CLKOUT Output

The reference frequency, or a fraction of it, can be provided on DIO5 (pin 13) by modifying bits *ClkOut* in *RegDioMapping2*. Two typical applications of the CLKOUT output include:

- To provide a clock output for a companion processor, thus saving the cost of an additional oscillator. CLKOUT can be made available in any operation mode except Sleep mode and is automatically enabled at power on reset.
- To provide an oscillator reference output. Measurement of the CLKOUT signal enables simple software trimming of the initial crystal tolerance.

Note *To minimize the current consumption of the SX1276/77/78/79, please ensure that the CLKOUT signal is disabled when not required.*

5.3.3. PLL

The local oscillator of the SX1276/77/78/79 is derived from two almost identical fractional-N PLLs that are referenced to the crystal oscillator circuit. Both PLLs feature a programmable bandwidth setting where one of four discrete preset bandwidths may be accessed.

The SX1276/77/78/79 PLL uses a 19-bit sigma-delta modulator whose frequency resolution, constant over the whole frequency range, is given by:

$$F_{STEP} = \frac{F_{XOSC}}{2^{19}}$$

The carrier frequency is programmed through *RegFrf*, split across addresses 0x06 to 0x08:

$$F_{RF} = F_{STEP} \times Frf(23,0)$$

Note *The Frf setting is split across 3 bytes. A change in the center frequency will only be taken into account when the least significant byte FrfLsb in RegFrfLsb is written. This allows the potential for user generation of m-ary FSK at very low bit rates. This is possible where frequency modulation is achieved by direct programming of the programmed RF centre frequency. To enable this functionality set the FastHopOn bit of register RegPllHop.*

Three frequency bands are supported, defined as follows:

Table 32 Frequency Bands

Name	Frequency Limits	Products
Band 1 (HF)	862 (*779)-1020 (*960) MHz	SX1276/77/79
Band 2 (LF)	410-525 (*480) MHz	SX1276/77/78/79
Band 3 (LF)	137-175 (*160)MHz	SX1276/77/78/79

** For SX1279*

5.3.4. RC Oscillator

All timing operations in the low-power Sleep state of the Top Level Sequencer rely on the accuracy of the internal low-power RC oscillator. This oscillator is automatically calibrated at the device power-up not requiring any user input.

5.4. Transmitter Description

The transmitter of SX1276/77/78/79 comprises the frequency synthesizer, modulator (both LoRa™ and FSK/OOK) and power amplifier blocks, together with the DC biasing and ramping functionality that is provided through the VR_PA block.

5.4.1. Architecture Description

The architecture of the RF front end is shown in the following diagram:

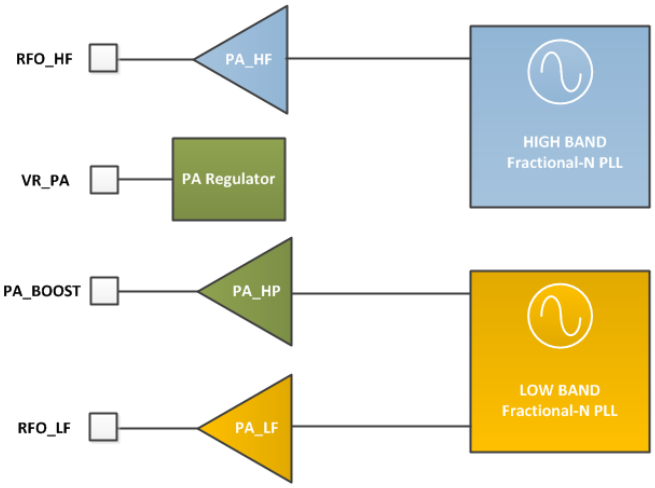


Figure 40. RF Front-end Architecture Shows the Internal PA Configuration.

5.4.2. RF Power Amplifiers

PA_HF and PA_LF are high efficiency amplifiers capable of yielding RF power programmable in 1 dB steps from -4 to +14dBm directly into a 50 ohm load with low current consumption. PA_LF covers the lower bands (up to 525 MHz), whilst PA_HF will cover the upper bands (from 779 MHz). The output power is sensitive to the power supply voltage, and typically their performance is expressed at 3.3V.

PA_HP (High Power), connected to the PA_BOOST pin, covers all frequency bands that the chip addresses. It permits continuous operation at up to +17 dBm and duty cycled operation at up to +20dBm. For full details of operation at +20dBm please consult section 5.4.3

Table 33 Power Amplifier Mode Selection Truth Table

PaSelect	Mode	Power Range	Pout Formula
0	PA_HF or PA_LF on RFO_HF or RFO_LF	-4 to +15dBm	Pout=Pmax-(15-OutputPower) Pmax=10.8+0.6*MaxPower [dBm]
1	PA_HP on PA_BOOST, any frequency	+2 to +17dBm	Pout=17-(15-OutputPower) [dBm]

Notes - For +20 dBm restrictions on operation please consult the following .

- To ensure correct operation at the highest power levels ensure that the current limiter OcpTrim is adjusted to permit delivery of the requisite supply current.

- If the PA_BOOST pin is not used it may be left floating.

5.4.3. High Power +20 dBm Operation

The SX1276/77/78/79 have a high power +20 dBm capability on PA_BOOST pin, with the following settings:

Table 34 High Power Settings

Register	*Address*	*Value for High Power*	*Default value PA_HF/LF or +17dBm*	Description
RegPaDac	0x4d	0x87	0x84	Set Pmax to +20dBm for PA_HP

Notes - High Power settings must be turned off when using PA_LF or PA_HF

- The Over Current Protection limit should be adapted to the actual power level, in RegOcp

Specific Absolute Maximum Ratings and Operating Range restrictions apply to the +20 dBm operation. They are listed in Table 35 and Table 36.

Table 35 Operating Range, +20dBm Operation

Symbol	Description	Min	Max	Unit
DC_20dBm	Duty Cycle of transmission at +20 dBm output	-	1	%
VSWR_20dBm	Maximum VSWR at antenna port, +20 dBm output	-	3:1	-

Table 36 Operating Range, +20dBm Operation

Symbol	Description	Min	Max	Unit
VDDop_20dBm	Supply voltage, +20 dBm output	2.4	3.7	V

The duty cycle of transmission at +20 dBm is limited to 1%, with a maximum VSWR of 3:1 at antenna port, over the standard operating range [-40;+85°C]. For any other operating condition, contact your Semtech representative.

5.4.4. Over Current Protection

The power amplifiers of SX1276/77/78/79 are protected against current over supply in adverse RF load conditions by the over current protection block. This has the added benefit of protecting battery chemistries with limited peak current capability and minimising worst case PA consumption in battery life calculation. The current limiter value is controlled by the *OcpTrim* bits in *RegOcp*, and is calculated according to the following formulae:

Table 37 Trimming of the OCP Current

OcpTrim	I_{MAX}	Imax Formula
0 to 15	45 to 120 mA	45 + 5**OcpTrim* [mA]
16 to 27	130 to 240 mA	-30 + 10**OcpTrim* [mA]
27+	240 mA	*240 mA*

Note Imax sets a limit on the current drain of the Power Amplifier only, hence the maximum current drain of the SX1276/77/78/79 *is equal to Imax + IDDFS*.

5.5. Receiver Description

5.5.1. Overview

The SX1276/77/78/79 features a digital receiver with the analog to digital conversion process being performed directly following the LNA-Mixers block. In addition to the LoRa™ modulation scheme the low-IF receiver is able to demodulate ASK, OOK, (G)FSK and (G)MSK modulation. All filtering, demodulation, gain control, synchronization and packet handling is performed digitally allowing a high degree of programmable flexibility. The receiver also has automatic gain calibration, this improves the precision of RSSI measurement and enhances image rejection.

5.5.2. Receiver Enabled and Receiver Active States

In the receiver operating mode two states of functionality are defined. Upon initial transition to receiver operating mode the receiver is in the ‘receiver-enabled’ state. In this state the receiver awaits for either the user defined valid preamble or RSSI detection criterion to be fulfilled. Once met the receiver enters ‘receiver-active’ state. In this second state the received signal is processed by the packet engine and top level sequencer. For a complete description of the digital functions of the SX1276/77/78/79 receiver please see section 4 of the datasheet.

5.5.3. Automatic Gain Control In FSK/OOK Mode

The AGC feature allows receiver to handle a wide Rx input dynamic range from the sensitivity level up to maximum input level of 0dBm or more, whilst optimizing the system linearity.

The following table shows typical NF and IIP3 performances for the SX1276/77/78/79 LNA gains available.

Table 38 LNA Gain Control and Performances

RX input level (Pin)	*Gain Setting*	*LnaGain*	*Relative LNA Gain [dB]*	NF Band 3/2/1 [dB]	IIP3 Band 3/2/1 [dBm]
Pin <= AgcThresh1	G1	'001'	0 dB	4/5.5/7	-15/-22/-11
AgcThresh1 < Pin <= AgcThresh2	G2	'010'	-6 dB	6.5/8/12	-11/-15/-6
AgcThresh2 < Pin <= AgcThresh3	G3	'011'	-12 dB	11/12/17	-11/-12/0
AgcThresh3 < Pin <= AgcThresh4	G4	'100'	-24 dB	20/21/27	2/3/9
AgcThresh4 < Pin <= AgcThresh5	G5	'110'	-26 dB	32/33/35	10/10/14
AgcThresh5 < Pin	G6	'111'	-48 dB	44/45/43	11/12/14

5.5.4. RSSI in FSK/OOK Mode

The RSSI provides a measure of the incoming signal power at RF input port, measured within the receiver bandwidth. The signal power is available in *RssiValue*. This value is absolute in units of dBm and with a resolution of 0.5 dB. The formula below relates the register value to the absolute input signal level at the RF input port:

$$RssiValue = -2 \cdot RF\ level\left[dBm\right] + RssiOffset\ \left[dB\right]$$

The RSSI value can be compensated to take into account the loss in the matching network or even the gain of an additional LNA by using *RssiOffset*. The offset can be chosen in 1 dB steps from -16 to +15 dB. When compensation is applied, the effective signal strength is read as follows:

$$RSSI\left[dBm\right] = -\frac{RssiValue}{2}$$

The RSSI value is smoothed on a user defined number of measured RSSI samples. The precision of the RSSI value is related to the number of RSSI samples used. *RssiSmoothing* selects the number of RSSI samples from a minimum of 2 samples up to 256 samples in increments of power of 2. Table 39 gives the estimation of the RSSI accuracy for a 10 dB SNR and response time versus the number of RSSI samples programmed in *RssiSmoothing*.

Table 39 RssiSmoothing Options

RssiSmoothing	Number of Samples	Estimated Accuracy	Response Time
'000'	2	± 6 dB	$\frac{2^{(RssiSmoothing+1)}}{4 \cdot RxBw\left[kHz\right]}\left[ms\right]$
'001'	4	± 5 dB	
'010'	8	± 4 dB	
'011'	16	± 3 dB	
'100'	32	± 2 dB	
'101'	64	± 1.5 dB	
'110'	128	± 1.2 dB	
'111'	256	± 1.1 dB	

The RSSI is calibrated when the image and RSSI calibration process is launched.

5.5.5. RSSI and SNR in LoRa™ Mode

The RSSI values reported by the LoRa™ modem differ from those expressed by the FSK/OOK modem. The following formula shows the method used to interpret the LoRa™ RSSI values:

RSSI (dBm) = -157 + *Rssi*, (when using the High Frequency (HF) port)

or

RSSI (dBm) = -164 + *Rssi*, (when using the Low Frequency (LF) port)

The same formula can be re-used to evaluate the signal strength of the received packet:

Packet Strength (dBm) = -157 + *Rssi*, (when using the High Frequency (HF) port)

or

Packet Strength (dBm) = -164 + *Rssi*, (when using the Low Frequency (LF) port)

Due to the nature of the LoRa modulation, it is possible to receive packets below the noise floor. In this situation, the SNR is used in conjunction of the PacketRssi to compute the signal strength of the received packet:

Packet Strength (dBm) = -157 + *PacketRssi* + *PacketSnr* * 0.25 (when using the HF port and SNR < 0)

or

Packet Strength (dBm) = -164 + *PacketRssi* + *PacketSnr* * 0.25 (when using the LF port and SNR < 0)

Note:

1. *PacketRssi* (in RegPktRssiValue), is an averaged version of *Rssi* (in RegRssiValue). *Rssi* can be read at any time (during packet reception or not), and should be averaged to give more precise results.

2. The constants, -157 and -164, may vary with the front-end setup of the SX1276/77/78/79 (*LnaBoost* =1 or 0, presence of an external LNA, mismatch at the LNA input…). It is recommended to adjust these values with a single-point calibration procedure to increase RSSI accuracy.

3. As signal strength increases (RSSI>-100dBm), the linearity of PacketRssi is not guaranteed and results will diverge from the ideal 1dB/dB ideal curve. When very good RSSI precision is required over the whole dynamic range of the receiver, two options are proposed:

- *Rssi* in RegRssiValue offers better linearity. *Rssi* can be sampled during the reception of the payload (between ValidHeader and RxDone IRQ), and used to extract a more high-signal RSSI measurement
- When SNR>=0, the standard formula can be adjusted to correct the slope:

RSSI = -157+16/15 * PacketRssi (or RSSI = -164+16/15 * PacketRssi)

5.5.6. Channel Filter

The role of the channel filter is to reject noise and interference outside of the wanted channel. The SX1276/77/78/79 channel filtering is implemented with a 16-tap finite impulse response (FIR) filter. Rejection of the filter is high enough that the filter stop-band performance is not the dominant influence on adjacent channel rejection performance. This is instead limited by the SX1276/77/78/79 local oscillator phase noise.

Note *To respect sampling criterion in the decimation chain of the receiver, the communication bit rate cannot be set at a higher than twice the single side receiver bandwidth (BitRate < 2 x RxBw)*

The single-side channel filter bandwidth *RxBw* is controlled by the parameters *RxBwMant* and *RxBwExp* in *RegRxBw:*

$$RxBw = \frac{FXOSC}{RxBwMant \times 2^{RxBwExp+2}}$$

The following channel filter bandwidths are hence accessible in the case of a 32 MHz reference oscillator:

Table 40 Available RxBw Settings

RxBwMant (binary/value)	*RxBwExp* (decimal)	*RxBw (kHz)* FSK/OOK
10b / 24	7	2.6
01b / 20	7	3.1
00b / 16	7	3.9
10b / 24	6	5.2
01b / 20	6	6.3
00b / 16	6	7.8
10b / 24	5	10.4
01b / 20	5	12.5
00b / 16	5	15.6
10b / 24	4	20.8
01b / 20	4	25.0
00b / 16	4	31.3
10b / 24	3	41.7
01b / 20	3	50.0
00b / 16	3	62.5
10b / 24	2	83.3
01b / 20	2	100.0
00b / 16	2	125.0
10b / 24	1	166.7
01b / 20	1	200.0
00b / 16	1	250.0
Other settings		reserved

5.5.7. Temperature Measurement

A stand alone temperature measurement block is used in order to measure the temperature in any mode except Sleep and Standby. It is enabled by default, and can be stopped by setting *TempMonitorOff* to 1. The result of the measurement is stored in *TempValue* in *RegTemp*.

Due to process variations, the absolute accuracy of the result is +/- 10 °C. Higher precision requires a calibration procedure at a known temperature. The figure below shows the influence of just such a calibration process. For more information, including source code, please consult the applications section of this document.

Example temperature curve, typical device

	Correction Factor	15	
Actual Temp [Celsius]	RegTemp [Dec]	Temp before calibration [°C]	Temp after calibration [°C]
85	181	74	89
75	190	65	80
65	201	54	69
55	211	44	59
45	222	33	48
35	232	23	38
25	245	10	25
15	0	0	15
5	10	-10	5
-5	21	-21	-6
-15	33	-33	-18
-25	44	-44	-29
-35	56	-56	-41
-40	63	-63	-48

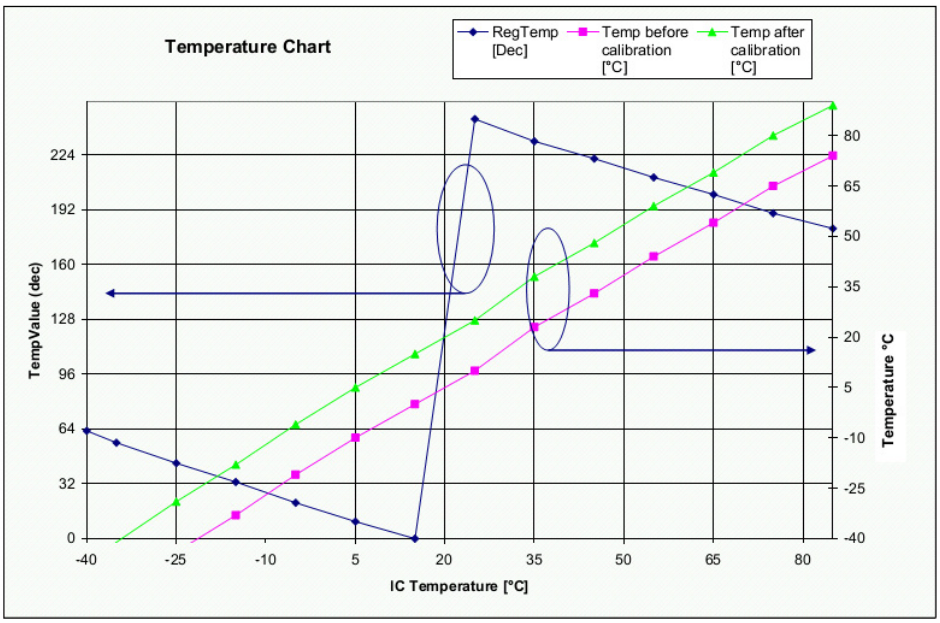


Figure 41. Temperature Sensor Response

When using the temperature sensor in the application, the following sequence should be followed:

- Set the device to Standby and wait for oscillator startup
- Set the device to FSRx mode
- Set *TempMonitorOff* = 0 (enables the sensor). It is not required to wait for the PLL Lock indication
- Wait for 140 microseconds
- Set *TempMonitorOff* = 1
- Set device back to Sleep of Standby mode
- Access temperature value in *RegTemp*

6. Description of the Registers

The register mapping depends upon whether FSK/OOK or LoRa™ mode has been selected. The following table summarises the location and function of each register and gives an overview of the changes in register mapping between both modes of operation.

6.1. Register Table Summary

Table 41 Registers Summary

Address	Register Name		Reset (POR)	Default (FSK)	Description	
	FSK/OOK Mode	LoRa™ Mode			FSK Mode	LoRa™ Mode
0x00	RegFifo		0x00		FIFO read/write access	
0x01	RegOpMode		0x01		Operating mode & LoRa™ / FSK selection	
0x02	RegBitrateMsb	Unused	0x1A		Bit Rate setting, Most Significant Bits	
0x03	RegBitrateLsb		0x0B		Bit Rate setting, Least Significant Bits	
0x04	RegFdevMsb		0x00		Frequency Deviation setting, Most Significant Bits	
0x05	RegFdevLsb		0x52		Frequency Deviation setting, Least Significant Bits	
0x06	RegFrfMsb		0x6C		RF Carrier Frequency, Most Significant Bits	
0x07	RegFrfMid		0x80		RF Carrier Frequency, Intermediate Bits	
0x08	RegFrfLsb		0x00		RF Carrier Frequency, Least Significant Bits	
0x09	RegPaConfig		0x4F		PA selection and Output Power control	
0x0A	RegPaRamp		0x09		Control of PA ramp time, low phase noise PLL	
0x0B	RegOcp		0x2B		Over Current Protection control	
0x0C	RegLna		0x20		LNA settings	
0x0D	RegRxConfig	RegFifoAddrPtr	0x08	0x0E	AFC, AGC, ctrl	FIFO SPI pointer
0x0E	RegRssiConfig	RegFifoTxBaseAddr	0x02		RSSI	Start Tx data
0x0F	RegRssiCollision	RegFifoRxBaseAddr	0x0A		RSSI Collision detector	Start Rx data
0x10	RegRssiThresh	FifoRxCurrentAddr	0xFF		RSSI Threshold control	Start address of last packet received
0x11	RegRssiValue	RegIrqFlagsMask	n/a	n/a	RSSI value in dBm	Optional IRQ flag mask
0x12	RegRxBw	RegIrqFlags	0x15		Channel Filter BW Control	IRQ flags
0x13	RegAfcBw	RegRxNbBytes	0x0B		AFC Channel Filter BW	Number of received bytes
0x14	RegOokPeak	RegRxHeaderCntValueMsb	0x28		OOK demodulator	Number of valid headers received
0x15	RegOokFix	RegRxHeaderCntValueLsb	0x0C		Threshold of the OOK demod	
0x16	RegOokAvg	RegRxPacketCntValueMsb	0x12		Average of the OOK demod	Number of valid packets received
0x17	Reserved17	RegRxPacketCntValueLsb	0x47		-	
0x18	Reserved18	RegModemStat	0x32		-	Live LoRa™ modem status
0x19	Reserved19	RegPktSnrValue	0x3E		-	Espimation of last packet SNR
0x1A	RegAfcFei	RegPktRssiValue	0x00		AFC and FEI control	RSSI of last packet

Address	Register Name		Reset (POR)	Default (FSK)	Description	
	FSK/OOK Mode	LoRa™ Mode			FSK Mode	LoRa™ Mode
0x1B	RegAfcMsb	RegRssiValue	0x00	n/a	Frequency correction value of the AFC	Current RSSI
0x1C	RegAfcLsb	RegHopChannel	0x00	n/a		FHSS start channel
0x1D	RegFeiMsb	RegModemConfig 1	0x00	n/a	Value of the calculated frequency error	Modem PHY config 1
0x1E	RegFeiLsb	RegModemConfig 2	0x00	n/a		Modem PHY config 2
0x1F	RegPreambleDetect	RegSymbTimeoutLsb	0x40	0xAA	Settings of the Preamble Detector	Receiver timeout value
0x20	RegRxTimeout1	RegPreambleMsb	0x00		Timeout Rx request and RSSI	Size of preamble
0x21	RegRxTimeout2	RegPreambleLsb	0x00		Timeout RSSI and *PayloadReady*	
0x22	RegRxTimeout3	RegPayloadLength	0x00		Timeout RSSI and *SyncAddress*	LoRaTM payload length
0x23	RegRxDelay	RegMaxPayloadLength	0x00		Delay between Rx cycles	LoRaTM maximum payload length
0x24	RegOsc	RegHopPeriod	0x05	0x07	RC Oscillators Settings, CLKOUT frequency	FHSS Hop period
0x25	RegPreambleMsb	RegFifoRxByteAddr	0x00		Preamble length, MSB	Address of last byte written in FIFO
0x26	RegPreambleLsb	RegModemConfig3	0x03		Preamble length, LSB	Modem PHY config 3
0x27	RegSyncConfig	RESERVED	0x93		Sync Word Recognition control	RESERVED
0x28	RegSyncValue1	RegFeiMsb	0x55	0x01	Sync Word bytes 1	Estimated frequency error
0x29	RegSyncValue2	RegFeiMid	0x55	0x01	Sync Word bytes 2	
0x2A	RegSyncValue3	RegFeiLsb	0x55	0x01	Sync Word bytes 3	
0x2B	RegSyncValue4	RESERVED	0x55	0x01	Sync Word bytes 4	RESERVED
0x2C	RegSyncValue5	RegRssiWideband	0x55	0x01	Sync Word bytes 5	Wideband RSSI measurement
0x2D-0x2F	RegSyncValue6-8	RESERVED	0x55	0x01	Sync Word bytes, 6 to 8	RESERVED
0x30	RegPacketConfig1	RESERVED	0x90		Packet mode settings	
0x31	RegPacketConfig2	RegDetectOptimize	0x40		Packet mode settings	LoRa detection Optimize for SF6
0x32	RegPayloadLength	RESERVED	0x40		Payload length setting	RESERVED
0x33	RegNodeAdrs	RegInvertIQ	0x00		Node address	Invert LoRa I and Q signals
0x34	RegBroadcastAdrs	RESERVED	0x00		Broadcast address	RESERVED
0x35	RegFifoThresh		0x0F	0x1F	Fifo threshold, Tx start condition	
0x36	RegSeqConfig1		0x00		Top level Sequencer settings	
0x37	RegSeqConfig2	RegDetectionThreshold	0x00		Top level Sequencer settings	LoRa detection threshold for SF6
0x38	RegTimerResol	RESERVED	0x00		Timer 1 and 2 resolution control	RESERVED
0x39	RegTimer1Coef	RegSyncWord	0xF5	0x12	Timer 1 setting	LoRa Sync Word

Address	Register Name		Reset (POR)	Default (FSK)	Description	
	FSK/OOK Mode	LoRa™ Mode			FSK Mode	LoRa™ Mode
0x3A	RegTimer2Coef	RESERVED	0x20		Timer 2 setting	RESERVED
0x3B	RegImageCal		0x82	0x02	Image calibration engine control	
0x3C	RegTemp		-		Temperature Sensor value	
0x3D	RegLowBat		0x02		Low Battery Indicator Settings	
0x3E	RegIrqFlags1		0x80		Status register: PLL Lock state, Timeout, RSSI	
0x3F	RegIrqFlags2		0x40		Status register: FIFO handling flags, Low Battery	
0x40	RegDioMapping1		0x00		Mapping of pins DIO0 to DIO3	
0x41	RegDioMapping2		0x00		Mapping of pins DIO4 and DIO5, ClkOut frequency	
0x42	RegVersion		0x12		Semtech ID relating the silicon revision	
0x44	RegPllHop	Unused	0x2D		Control the fast frequency hopping mode	Unused
0x4B	RegTcxo		0x09		TCXO or XTAL input setting	
0x4D	RegPaDac		0x84		Higher power settings of the PA	
0x5B	RegFormerTemp		-		Stored temperature during the former IQ Calibration	
0x5D	RegBitRateFrac	Unused	0x00		Fractional part in the Bit Rate division ratio	Unused
0x61	RegAgcRef		0x13		Adjustment of the AGC thresholds	
0x62	RegAgcThresh1		0x0E			
0x63	RegAgcThresh2		0x5B			
0x64	RegAgcThresh3		0xDB			
0x70	RegPll		0xD0		Control of the PLL bandwidth	
others	RegTest		-		Internal test registers. Do not overwrite	

Note - Reset values are automatically refreshed in the chip at Power On Reset

- Default values are the Semtech recommended register values, optimizing the device operation

*- Registers for which the Default value differs from the Reset value are denoted by a * in the tables of section 6.2*

6.2. FSK/OOK Mode Register Map

This section details the SX1276/77/78/79 register mapping and the precise contents of each register in FSK/OOK mode.

Convention: r: read, w: write, t:trigger, c: clear

Table 42 Register Map

Name (Address)	Bits	Variable Name	Mode	Default value	FSK/OOK Description
RegFifo (0x00)	7-0	Fifo	rw	0x00	FIFO data input/output
Registers for Common settings					
RegOpMode (0x01)	7	LongRangeMode	r	0x00	0 → FSK/OOK Mode 1→ LoRa™ Mode This bit can be modified only in Sleep mode. A write operation on other device modes is ignored.
	6-5	ModulationType	rw	0x00	Modulation scheme: 00 → FSK 01 → OOK 10 → 11 → reserved
	4	reserved	r	0x0	reserved
	3	LowFrequencyModeOn	rw	0x01	Access Low Frequency Mode registers (from address 0x61 on) 0 → High Frequency Mode (access to HF test registers) 1 → Low Frequency Mode (access to LF test registers)
	2-0	Mode	rw	0x01	Transceiver modes 000 → Sleep mode 001 → Stdby mode 010 → FS mode TX (FSTx) 011 → Transmitter mode (Tx) 100 → FS mode RX (FSRx) 101 → Receiver mode (Rx) 110 → reserved 111 → reserved
RegBitrateMsb (0x02)	7-0	BitRate(15:8)	rw	0x1a	MSB of Bit Rate (chip rate if Manchester encoding is enabled)
RegBitrateLsb (0x03)	7-0	BitRate(7:0)	rw	0x0b	LSB of bit rate (chip rate if Manchester encoding is enabled) $BitRate = \frac{FXOSC}{BitRate(15,0) + \frac{BitrateFrac}{16}}$ Default value: 4.8 kb/s
RegFdevMsb (0x04)	7-6	reserved	rw	0x00	reserved
	5-0	Fdev(13:8)	rw	0x00	MSB of the frequency deviation
RegFdevLsb (0x05)	7-0	Fdev(7:0)	rw	0x52	LSB of the frequency deviation $Fdev = Fstep \times Fdev(15,0)$ Default value: 5 kHz

Name (Address)	Bits	Variable Name	Mode	Default value	FSK/OOK Description
RegFrfMsb (0x06)	7-0	Frf(23:16)	rw	0x6c	MSB of the RF carrier frequency
RegFrfMid (0x07)	7-0	Frf(15:8)	rw	0x80	MSB of the RF carrier frequency
RegFrfLsb (0x08)	7-0	Frf(7:0)	rw	0x00	LSB of RF carrier frequency $Frf = Fstep \times Frf(23;0)$ Default value: 434.000 MHz The RF frequency is taken into account internally only when: - entering FSRX/FSTX modes - re-starting the receiver
Registers for the Transmitter					
RegPaConfig (0x09)	7	PaSelect	rw	0x00	Selects PA output pin 0 → RFO pin. Maximum power of +14 dBm 1 → PA_BOOST pin. Maximum power of +20 dBm
	6-4	MaxPower	rw	0x04	Select max output power: Pmax=10.8+0.6*MaxPower [dBm]
	3-0	OutputPower	rw	0x0f	Pout=Pmax-(15-OutputPower) if PaSelect = 0 (RFO pins) Pout=17-(15-OutputPower) if PaSelect = 1 (PA_BOOST pin)
RegPaRamp (0x0A)	7	unused	r	0x00	unused
	6-5	ModulationShaping	rw	0x00	Data shaping: In FSK: 00 → no shaping 01 → Gaussian filter BT = 1.0 10 → Gaussian filter BT = 0.5 11 → Gaussian filter BT = 0.3 In OOK: 00 → no shaping 01 → filtering with fcutoff = bit_rate 10 → filtering with fcutoff = 2*bit_rate (for bit_rate < 125 kb/s) 11 → reserved
	4	reserved	rw	0x00	reserved
	3-0	PaRamp	rw	0x09	Rise/Fall time of ramp up/down in FSK 0000 → 3.4 ms 0001 → 2 ms 0010 → 1 ms 0011 → 500 us 0100 → 250 us 0101 → 125 us 0110 → 100 us 0111 → 62 us 1000 → 50 us 1001 → 40 us (d) 1010 → 31 us 1011 → 25 us 1100 → 20 us 1101 → 15 us 1110 → 12 us 1111 → 10 us

Name (Address)	Bits	Variable Name	Mode	Default value	FSK/OOK Description
RegOcp (0x0B)	7-6	unused	r	0x00	unused
	5	OcpOn	rw	0x01	Enables overload current protection (OCP) for the PA: 0 → OCP disabled 1 → OCP enabled
	4-0	OcpTrim	rw	0x0b	Trimming of OCP current: I_{max} = 45+5*OcpTrim [mA] if OcpTrim <= 15 (120 mA) / I_{max} = -30+10*OcpTrim [mA] if 15 < OcpTrim <= 27 (130 to 240 mA) I_{max} = 240mA for higher settings Default I_{max} = 100mA
Registers for the Receiver					
RegLna (0x0C)	7-5	LnaGain	rw	0x01	LNA gain setting: 000 → reserved 001 → G1 = highest gain 010 → G2 = highest gain – 6 dB 011 → G3 = highest gain – 12 dB 100 → G4 = highest gain – 24 dB 101 → G5 = highest gain – 36 dB 110 → G6 = highest gain – 48 dB 111 → reserved Note: Reading this address always returns the current LNA gain (which may be different from what had been previously selected if AGC is enabled.
	4-3	LnaBoostLf	rw	0x00	Low Frequency (RFI_LF) LNA current adjustment 00 → Default LNA current Other → Reserved
	2	reserved	rw	0x00	reserved
	1-0	LnaBoostHf	rw	0x00	High Frequency (RFI_HF) LNA current adjustment 00 → Default LNA current 11 → Boost on, 150% LNA current

Name (Address)	Bits	Variable Name	Mode	Default value	FSK/OOK Description
RegRxConfig (0x0d)	7	RestartRxOnCollision	rw	0x00	Turns on the mechanism restarting the receiver automatically if it gets saturated or a packet collision is detected 0 → No automatic Restart 1 → Automatic restart On
	6	RestartRxWithoutPllLock	wt	0x00	Triggers a manual Restart of the Receiver chain when set to 1. Use this bit when there is no frequency change, RestartRxWithPllLock otherwise.
	5	RestartRxWithPllLock	wt	0x00	Triggers a manual Restart of the Receiver chain when set to 1. Use this bit when there is a frequency change, requiring some time for the PLL to re-lock.
	4	AfcAutoOn	rw	0x00	0 → No AFC performed at receiver startup 1 → AFC is performed at each receiver startup
	3	AgcAutoOn	rw	0x01	0 → LNA gain forced by the LnaGain Setting 1 → LNA gain is controlled by the AGC
	2-0	RxTrigger	rw	0x06 *	Selects the event triggering AGC and/or AFC at receiver startup. See Table 24 for a description.
RegRssiConfig (0x0e)	7-3	RssiOffset	rw	0x00	Signed RSSI offset, to compensate for the possible losses/gains in the front-end (LNA, SAW filter...) 1dB / LSB, 2's complement format
	2-0	RssiSmoothing	rw	0x02	Defines the number of samples taken to average the RSSI result: 000 → 2 samples used 001 → 4 samples used 010 → 8 samples used 011 → 16 samples used 100 → 32 samples used 101 → 64 samples used 110 → 128 samples used 111 → 256 samples used
RegRssiCollision (0x0f)	7-0	RssiCollisionThreshold	rw	0x0a	Sets the threshold used to consider that an interferer is detected, witnessing a packet collision. 1dB/LSB (only RSSI increase) Default: 10dB
RegRssiThresh (0x10)	7-0	RssiThreshold	rw	0xff	RSSI trigger level for the Rssi interrupt: - RssiThreshold / 2 [dBm]
RegRssiValue (0x11)	7-0	RssiValue	r	-	Absolute value of the RSSI in dBm, 0.5dB steps. RSSI = - RssiValue/2 [dBm]
RegRxBw (0x12)	7	unused	r	-	unused
	6-5	reserved	rw	0x00	reserved
	4-3	RxBwMant	rw	0x02	Channel filter bandwidth control: 00 → RxBwMant = 16 10 → RxBwMant = 24 01 → RxBwMant = 20 11 → reserved
	2-0	RxBwExp	rw	0x05	Channel filter bandwidth control
RegAfcBw (0x13)	7-5	reserved	rw	0x00	reserved
	4-3	RxBwMantAfc	rw	0x01	RxBwMant parameter used during the AFC
	2-0	RxBwExpAfc	rw	0x03	RxBwExp parameter used during the AFC

Name (Address)	Bits	Variable Name	Mode	Default value	FSK/OOK Description
RegOokPeak (0x14)	7-6	reserved	rw	0x00	reserved
	5	BitSyncOn	rw	0x01	Enables the Bit Synchronizer. 0 → Bit Sync disabled (not possible in Packet mode) 1 → Bit Sync enabled
	4-3	OokThreshType	rw	0x01	Selects the type of threshold in the OOK data slicer: 00 → fixed threshold 10 → average mode 01 → peak mode (default) 11 → reserved
	2-0	OokPeakTheshStep	rw	0x00	Size of each decrement of the RSSI threshold in the OOK demodulator: 000 → 0.5 dB 001 → 1.0 dB 010 → 1.5 dB 011 → 2.0 dB 100 → 3.0 dB 101 → 4.0 dB 110 → 5.0 dB 111 → 6.0 dB
RegOokFix (0x15)	7-0	OokFixedThreshold	rw	0x0C	Fixed threshold for the Data Slicer in OOK mode Floor threshold for the Data Slicer in OOK when Peak mode is used
RegOokAvg (0x16)	7-5	OokPeakThreshDec	rw	0x00	Period of decrement of the RSSI threshold in the OOK demodulator: 000 → once per chip 001 → once every 2 chips 010 → once every 4 chips 011 → once every 8 chips 100 → twice in each chip 101 → 4 times in each chip 110 → 8 times in each chip 111 → 16 times in each chip
	4	reserved	rw	0x01	reserved
	3-2	OokAverageOffset	rw	0x00	Static offset added to the threshold in average mode in order to reduce glitching activity (OOK only): 00 → 0.0 dB 10 → 4.0 dB 01 → 2.0 dB 11 → 6.0 dB
	1-0	OokAverageThreshFilt	rw	0x02	Filter coefficients in average mode of the OOK demodulator: 00 → f_C ≈ chip rate / 32.π 01 → f_C ≈ chip rate / 8.π 10 → f_C ≈ chip rate / 4.π 11 → f_C ≈ chip rate / 2.π
RegRes17 to RegRes19	7-0	reserved	rw	0x47 0x32 0x3E	reserved. Keep the Reset values.
RegAfcFei (0x1a)	7-5	unused	r	-	unused
	4	AgcStart	wt	0x00	Triggers an AGC sequence when set to 1.
	3	reserved	rw	0x00	reserved
	2	unused	-	-	unused
	1	AfcClear	wc	0x00	Clear AFC register set in Rx mode. Always reads 0.
	0	AfcAutoClearOn	rw	0x00	Only valid if AfcAutoOn is set 0 → AFC register is not cleared at the beginning of the automatic AFC phase 1 → AFC register is cleared at the beginning of the automatic AFC phase

Name (Address)	Bits	Variable Name	Mode	Default value	FSK/OOK Description
RegAfcMsb (0x1b)	7-0	AfcValue(15:8)	rw	0x00	MSB of the AfcValue, 2's complement format. Can be used to overwrite the current AFC value
RegAfcLsb (0x1c)	7-0	AfcValue(7:0)	rw	0x00	LSB of the AfcValue, 2's complement format. Can be used to overwrite the current AFC value
RegFeiMsb (0x1d)	7-0	FeiValue(15:8)	rw	-	MSB of the measured frequency offset, 2's complement. Must be read before RegFeiLsb.
RegFeiLsb (0x1e)	7-0	FeiValue(7:0)	rw	-	LSB of the measured frequency offset, 2's complement *Frequency error* = FeiValue x Fstep
RegPreambleDetect (0x1f)	7	PreambleDetectorOn	rw	0x01 *	Enables Preamble detector when set to 1. The AGC settings supersede this bit during the startup / AGC phase. 0 → Turned off 1 → Turned on
	6-5	PreambleDetectorSize	rw	0x01 *	Number of Preamble bytes to detect to trigger an interrupt 00 → 1 byte 10 → 3 bytes 01 → 2 bytes 11 → Reserved
	4-0	PreambleDetectorTol	rw	0x0A *	Number or chip errors tolerated over PreambleDetectorSize. 4 chips per bit.
RegRxTimeout1 (0x20)	7-0	TimeoutRxRssi	rw	0x00	*Timeout* interrupt is generated *TimeoutRxRssi**16*T_{bit} after switching to Rx mode if *Rssi* interrupt doesn't occur (i.e. *RssiValue* > *RssiThreshold)* 0x00: *TimeoutRxRssi* is disabled
RegRxTimeout2 (0x21)	7-0	TimeoutRxPreamble	rw	0x00	*Timeout* interrupt is generated *TimeoutRxPreamble**16*T_{bit} after switching to Rx mode if *Preamble* interrupt doesn't occur 0x00: *TimeoutRxPreamble* is disabled
RegRxTimeout3 (0x22)	7-0	TimeoutSignalSync	rw	0x00	*Timeout* interrupt is generated *TimeoutSignalSync**16*T_{bit} after the Rx mode is programmed, if *SyncAddress* doesn't occur 0x00: *TimeoutSignalSync* is disabled
RegRxDelay (0x23)	7-0	InterPacketRxDelay	rw	0x00	Additional delay before an automatic receiver restart is launched: Delay = InterPacketRxDelay*4*Tbit
RC Oscillator registers					
RegOsc (0x24)	7-4	unused	r	-	unused
	3	RcCalStart	wt	0x00	Triggers the calibration of the RC oscillator when set. Always reads 0. RC calibration must be triggered in Standby mode.
	2-0	ClkOut	rw	0x07 *	Selects CLKOUT frequency: 000 → FXOSC 001 → FXOSC / 2 010 → FXOSC / 4 011 → FXOSC / 8 100 → FXOSC / 16 101 → FXOSC / 32 110 → RC (automatically enabled) 111 → OFF
Packet Handling registers					

Name (Address)	Bits	Variable Name	Mode	Default value	FSK/OOK Description
RegPreambleMsb (0x25)	7-0	PreambleSize(15:8)	rw	0x00	Size of the preamble to be sent (from *TxStartCondition* fulfilled). (MSB byte)
RegPreambleLsb (0x26)	7-0	PreambleSize(7:0)	rw	0x03	Size of the preamble to be sent (from *TxStartCondition* fulfilled). (LSB byte)
RegSyncConfig (0x27)	7-6	AutoRestartRxMode	rw	0x02	Controls the automatic restart of the receiver after the reception of a valid packet (PayloadReady or CrcOk): 00 → Off 01 → On, without waiting for the PLL to re-lock 10 → On, wait for the PLL to lock (frequency changed) 11 → reserved
	5	PreamblePolarity	rw	0x00	Sets the polarity of the Preamble 0 → 0xAA (default) 1 → 0x55
	4	SyncOn	rw	0x01	Enables the Sync word generation and detection: 0 → Off 1 → On
	3	reserved	rw	0x00	reserved
	2-0	SyncSize	rw	0x03	Size of the Sync word: (*SyncSize* + 1) bytes, (*SyncSize*) bytes if *ioHomeOn*=1
RegSyncValue1 (0x28)	7-0	SyncValue(63:56)	rw	0x01 *	1st byte of Sync word. (MSB byte) Used if *SyncOn* is set.
RegSyncValue2 (0x29)	7-0	SyncValue(55:48)	rw	0x01 *	2nd byte of Sync word Used if *SyncOn* is set and *(SyncSize +1)* >= 2.
RegSyncValue3 (0x2a)	7-0	SyncValue(47:40)	rw	0x01 *	3rd byte of Sync word. Used if *SyncOn* is set and *(SyncSize +1)* >= 3.
RegSyncValue4 (0x2b)	7-0	SyncValue(39:32)	rw	0x01 *	4th byte of Sync word. Used if *SyncOn* is set and *(SyncSize +1)* >= 4.
RegSyncValue5 (0x2c)	7-0	SyncValue(31:24)	rw	0x01 *	5th byte of Sync word. Used if *SyncOn* is set and *(SyncSize +1)* >= 5.
RegSyncValue6 (0x2d)	7-0	SyncValue(23:16)	rw	0x01 *	6th byte of Sync word. Used if *SyncOn* is set and *(SyncSize +1)* >= 6.
RegSyncValue7 (0x2e)	7-0	SyncValue(15:8)	rw	0x01 *	7th byte of Sync word. Used if *SyncOn* is set and *(SyncSize +1)* >= 7.
RegSyncValue8 (0x2f)	7-0	SyncValue(7:0)	rw	0x01 *	8th byte of Sync word. Used if *SyncOn* is set and *(SyncSize +1)* = 8.

Name (Address)	Bits	Variable Name	Mode	Default value	FSK/OOK Description
RegPacketConfig1 (0x30)	7	PacketFormat	rw	0x01	Defines the packet format used: 0 → Fixed length 1 → Variable length
	6-5	DcFree	rw	0x00	Defines DC-free encoding/decoding performed: 00 → None (Off) 01 → Manchester 10 → Whitening 11 → reserved
	4	CrcOn	rw	0x01	Enables CRC calculation/check (Tx/Rx): 0 → Off 1 → On
	3	CrcAutoClearOff	rw	0x00	Defines the behavior of the packet handler when CRC check fails: 0 → Clear FIFO and restart new packet reception. No *PayloadReady* interrupt issued. 1 → Do not clear FIFO. *PayloadReady* interrupt issued.
	2-1	AddressFiltering	rw	0x00	Defines address based filtering in Rx: 00 → None (Off) 01 → Address field must match *NodeAddress* 10 → Address field must match *NodeAddress* or *BroadcastAddress* 11 → reserved
	0	CrcWhiteningType	rw	0x00	Selects the CRC and whitening algorithms: 0 → CCITT CRC implementation with standard whitening 1 → IBM CRC implementation with alternate whitening
RegPacketConfig2 (0x31)	7	unused	r	-	unused
	6	DataMode	rw	0x01	Data processing mode: 0 → Continuous mode 1 → Packet mode
	5	IoHomeOn	rw	0x00	Enables the io-homecontrol® compatibility mode 0 → Disabled 1 → Enabled
	4	IoHomePowerFrame	rw	0x00	reserved - Linked to io-homecontrol® compatibility mode
	3	BeaconOn	rw	0x00	Enables the Beacon mode in Fixed packet format
	2-0	PayloadLength(10:8)	rw	0x00	Packet Length Most significant bits
RegPayloadLength (0x32)	7-0	PayloadLength(7:0)	rw	0x40	If PacketFormat = 0 (fixed), payload length. If PacketFormat = 1 (variable), max length in Rx, not used in Tx.
RegNodeAdrs (0x33)	7-0	NodeAddress	rw	0x00	Node address used in address filtering.
RegBroadcastAdrs (0x34)	7-0	BroadcastAddress	rw	0x00	Broadcast address used in address filtering.

Name (Address)	Bits	Variable Name	Mode	Default value	FSK/OOK Description
RegFifoThresh (0x35)	7	TxStartCondition	rw	0x01 *	Defines the condition to start packet transmission: 0 → *FifoLevel* (i.e. the number of bytes in the FIFO exceeds *FifoThreshold)* 1 → *FifoEmpty goes low*(i.e. at least one byte in the FIFO)
	6	unused	r	-	unused
	5-0	FifoThreshold	rw	0x0f	Used to trigger *FifoLevel* interrupt, when: number of bytes in FIFO >= FifoThreshold + 1
Sequencer registers					
RegSeqConfig1 (0x36)	7	SequencerStart	wt	0x00	Controls the top level Sequencer When set to '1', executes the "Start" transition. The sequencer can only be enabled when the chip is in Sleep or Standby mode.
	6	SequencerStop	wt	0x00	Forces the Sequencer Off. Always reads '0'
	5	IdleMode	rw	0x00	Selects chip mode during the state: 0: Standby mode 1: Sleep mode
	4-3	FromStart	rw	0x00	Controls the Sequencer transition when *SequencerStart* is set to 1 in Sleep or Standby mode: 00: to LowPowerSelection 01: to Receive state 10: to Transmit state 11: to Transmit state on a *FifoLevel* interrupt
	2	LowPowerSelection	rw	0x00	Selects the Sequencer LowPower state after a *to LowPowerSelection* transition: 0: SequencerOff state with chip on Initial mode 1: Idle state with chip on *Standby* or *Sleep* mode depending on *IdleMode* *Note: Initial mode is the chip LowPower mode at Sequencer Start.*
	1	FromIdle	rw	0x00	Controls the Sequencer transition from the Idle state on a T1 interrupt: 0: to Transmit state 1: to Receive state
	0	FromTransmit	rw	0x00	Controls the Sequencer transition from the Transmit state: 0: to LowPowerSelection on a *PacketSent* interrupt 1: to Receive state on a *PacketSent* interrupt

Name (Address)	Bits	Variable Name	Mode	Default value	FSK/OOK Description
RegSeqConfig2 (0x37)	7-5	FromReceive	rw	0x00	Controls the Sequencer transition from the Receive state 000 and 111: unused 001: to PacketReceived state on a *PayloadReady* interrupt 010: to LowPowerSelection on a *PayloadReady* interrupt 011: to PacketReceived state on a *CrcOk* interrupt (1) 100: to SequencerOff state on a *Rssi* interrupt 101: to SequencerOff state on a *SyncAddress* interrupt 110: to SequencerOff state on a *PreambleDetect* interrupt Irrespective of this setting, transition to LowPowerSelection on a T2 interrupt (1) If the CRC is wrong (corrupted packet, with CRC on but *CrcAutoClearOn*=0), the *PayloadReady* interrupt will drive the sequencer to RxTimeout state.
	4-3	FromRxTimeout	rw	0x00	Controls the state-machine transition from the Receive state on a *RxTimeout* interrupt (and on *PayloadReady* if FromReceive = 011): 00: to Receive State, via ReceiveRestart 01: to Transmit state 10: to LowPowerSelection 11: to SequencerOff state *Note: RxTimeout interrupt is a TimeoutRxRssi, TimeoutRxPreamble or TimeoutSignalSync interrupt*
	2-0	FromPacketReceived	rw	0x00	Controls the state-machine transition from the PacketReceived state: 000: to SequencerOff state 001: to Transmit state on a *FifoEmpty* interrupt 010: to LowPowerSelection 011: to Receive via FS mode, if frequency was changed 100: to Receive state (no frequency change)
RegTimerResol (0x38)	7-4	unused	r	-	unused
	3-2	Timer1Resolution	rw	0x00	Resolution of Timer 1 00: Timer1 disabled 01: 64 us 10: 4.1 ms 11: 262 ms
	1-0	Timer2Resolution	rw	0x00	Resolution of Timer 2 00: Timer2 disabled 01: 64 us 10: 4.1 ms 11: 262 ms
RegTimer1Coef (0x39)	7-0	Timer1Coefficient	rw	0xf5	Multiplying coefficient for Timer 1
RegTimer2Coef (0x3a)	7-0	Timer2Coefficient	rw	0x20	Multiplying coefficient for Timer 2

Name (Address)	Bits	Variable Name	Mode	Default value	FSK/OOK Description
Service registers					
RegImageCal (0x3b)	7	AutoImageCalOn	rw	0x00 *	Controls the Image calibration mechanism 0 → Calibration of the receiver depending on the temperature is disabled 1 → Calibration of the receiver depending on the temperature enabled.
	6	ImageCalStart	wt	-	Triggers the IQ and RSSI calibration when set in Standby mode.
	5	ImageCalRunning	r	0x00	Set to 1 while the Image and RSSI calibration are running. Toggles back to 0 when the process is completed
	4	unused	r	-	unused
	3	TempChange	r	0x00	IRQ flag witnessing a temperature change exceeding TempThreshold since the last Image and RSSI calibration: 0 → Temperature change lower than TempThreshold 1 → Temperature change greater than TempThreshold
	2-1	TempThreshold	rw	0x01	Temperature change threshold to trigger a new I/Q calibration 00 → 5 °C 01 → 10 °C 10 → 15 °C 11 → 20 °C
	0	TempMonitorOff	rw	0x00	Controls the temperature monitor operation: 0 → Temperature monitoring done in all modes except Sleep and Standby 1 → Temperature monitoring stopped.
RegTemp (0x3c)	7-0	TempValue	r	-	Measured temperature -1°C per Lsb Needs calibration for absolute accuracy
RegLowBat (0x3d)	7-4	unused	r	-	unused
	3	LowBatOn	rw	0x00	Low Battery detector enable signal 0 → LowBat detector disabled 1 → LowBat detector enabled
	2-0	LowBatTrim	rw	0x02	Trimming of the LowBat threshold: 000 → 1.695 V 001 → 1.764 V 010 → 1.835 V (d) 011 → 1.905 V 100 → 1.976 V 101 → 2.045 V 110 → 2.116 V 111 → 2.185 V
Status registers					

Name (Address)	Bits	Variable Name	Mode	Default value	FSK/OOK Description
RegIrqFlags1 (0x3e)	7	ModeReady	r	-	Set when the operation mode requested in *Mode*, is ready - Sleep: Entering Sleep mode - Standby: XO is running - FS: PLL is locked - Rx: RSSI sampling starts - Tx: PA ramp-up completed Cleared when changing the operating mode.
	6	RxReady	r	-	Set in Rx mode, after RSSI, AGC and AFC. Cleared when leaving Rx.
	5	TxReady	r	-	Set in Tx mode, after PA ramp-up. Cleared when leaving Tx.
	4	PllLock	r	-	Set (in FS, Rx or Tx) when the PLL is locked. Cleared when it is not.
	3	Rssi	rwc	-	Set in Rx when the *RssiValue* exceeds *RssiThreshold.* Cleared when leaving Rx or setting this bit to 1.
	2	Timeout	r	-	Set when a timeout occurs Cleared when leaving Rx or FIFO is emptied.
	1	PreambleDetect	rwc	-	Set when the Preamble Detector has found valid Preamble. bit clear when set to 1
	0	SyncAddressMatch	rwc	-	Set when Sync and Address (if enabled) are detected. Cleared when leaving Rx or FIFO is emptied. This bit is read only in Packet mode, rwc in Continuous mode
RegIrqFlags2 (0x3f)	7	FifoFull	r	-	Set when FIFO is full (i.e. contains 66 bytes), else cleared.
	6	FifoEmpty	r	-	Set when FIFO is empty, and cleared when there is at least 1 byte in the FIFO.
	5	FifoLevel	r	-	Set when the number of bytes in the FIFO strictly exceeds *FifoThreshold*, else cleared.
	4	FifoOverrun	rwc	-	Set when FIFO overrun occurs. (except in Sleep mode) Flag(s) and FIFO are cleared when this bit is set. The FIFO then becomes immediately available for the next transmission / reception.
	3	PacketSent	r	-	Set in Tx when the complete packet has been sent. Cleared when exiting Tx
	2	PayloadReady	r	-	Set in Rx when the payload is ready (i.e. last byte received and CRC, if enabled and *CrcAutoClearOff* is cleared, is Ok). Cleared when FIFO is empty.
	1	CrcOk	r	-	Set in Rx when the CRC of the payload is Ok. Cleared when FIFO is empty.
	0	LowBat	rwc	-	Set when the battery voltage drops below the Low Battery threshold. Cleared only when set to 1 by the user.
IO control registers					

Name (Address)	Bits	Variable Name	Mode	Default value	FSK/OOK Description
RegDioMapping1 (0x40)	7-6	Dio0Mapping	rw	0x00	Mapping of pins DIO0 to DIO5 See Table 18 for mapping in LoRa mode See Table 29 for mapping in Continuous mode SeeTable 30 for mapping in Packet mode
	5-4	Dio1Mapping	rw	0x00	
	3-2	Dio2Mapping	rw	0x00	
	1-0	Dio3Mapping	rw	0x00	
RegDioMapping2 (0x41)	7-6	Dio4Mapping	rw	0x00	
	5-4	Dio5Mapping	rw	0x00	
	3-1	reserved	rw	0x00	reserved. Retain default value
	0	MapPreambleDetect	rw	0x00	Allows the mapping of either *Rssi* Or *PreambleDetect* to the DIO pins, as summarized on Table 29 and Table 30 0 → *Rssi* interrupt 1 → *PreambleDetect* interrupt
Version register					
RegVersion (0x42)	7-0	Version	r	0x12	Version code of the chip. Bits 7-4 give the full revision number; bits 3-0 give the metal mask revision number.
Additional registers					
RegPllHop (0x44)	7	FastHopOn	rw	0x00	Bypasses the main state machine for a quick frequency hop. Writing RegFrfLsb will trigger the frequency change. 0 → Frf is validated when FSTx or FSRx is requested 1 → Frf is validated triggered when RegFrfLsb is written
	6-0	reserved	rw	0x2d	reserved
RegTcxo (0x4b)	7-5	reserved	rw	0x00	reserved. Retain default value
	4	TcxoInputOn	rw	0x00	Controls the crystal oscillator 0 → Crystal Oscillator with external Crystal 1 → External clipped sine TCXO AC-connected to XTA pin
	3-0	reserved	rw	0x09	Reserved. Retain default value.
RegPaDac (0x4d)	7-3	reserved	rw	0x10	reserved. Retain default value
	2-0	PaDac	rw	0x04	Enables the +20dBm option on PA_BOOST pin 0x04 → Default value 0x07 → +20dBm on PA_BOOST when OutputPower=1111
RegFormerTemp (0x5b)	7-0	FormerTemp	rw	-	Temperature saved during the latest IQ (RSSI and Image) calibration. Same format as *TempValue* in *RegTemp*.
RegBitrateFrac (0x5d)	7-4	unused	r	0x00	unused
	3-0	BitRateFrac	rw	0x00	Fractional part of the bit rate divider (Only valid for FSK) If *BitRateFrac*> 0 then: $BitRate = \frac{FXOSC}{BitRate(15,0) + \frac{BitrateFrac}{16}}$

Name (Address)	Bits	Variable Name	Mode	Default value	FSK/OOK Description
RegAgcRef (0x61)	7-6	unused	r	-	unused
	5-0	AgcReferenceLevel	rw	0x19	Sets the floor reference for all AGC thresholds: AGC Reference[dBm]= -174dBm+10*log(2**RxBw*)+SNR+*AgcReferenceLevel* SNR = 8dB, fixed value
RegAgcThresh1 (0x62)	7-5	unused	r	-	unused
	4-0	AgcStep1	rw	0x0c	Defines the 1st AGC Threshold
RegAgcThresh2 (0x63)	7-4	AgcStep2	rw	0x04	Defines the 2nd AGC Threshold:
	3-0	AgcStep3	rw	0x0b	Defines the 3rd AGC Threshold:
RegAgcThresh3 (0x64)	7-4	AgcStep4	rw	0x0c	Defines the 4th AGC Threshold:
	3-0	AgcStep5	rw	0x0c	Defines the 5th AGC Threshold:

6.3. Band Specific Additional Registers

The registers in the address space from 0x61 to 0x73 are specific for operation in the lower frequency bands (below 525 MHz), or in the upper frequency bands (above 779 MHz). Their programmed value may differ, and are retained when switching from lower to high frequency and vice-versa. The access to the band specific registers is granted by enabling or disabling the bit 3 *LowFrequencyModeOn* of the *RegOpMode* register. By default, the bit *LowFrequencyModeOn* is at '1' indicating that the registers are configured for the low frequency band.

Table 43 Low Frequency Additional Registers

Name (Address)	Bits	Variable Name	Mode	Default value	Low Frequency Additional Registers
RegAgcRefLf (0x61)	7-6	unused	r	-	unused
	5-0	AgcReferenceLevel	rw	0x19	Sets the floor reference for all AGC thresholds: AGC Reference[dBm]= -174dBm+10*log(2**RxBw*)+SNR+*AgcReferenceLevel* SNR = 8dB, fixed value
RegAgcThresh1Lf (0x62)	7-5	unused	r	-	unused
	4-0	AgcStep1	rw	0x0c	Defines the 1st AGC Threshold
RegAgcThresh2Lf (0x63)	7-4	AgcStep2	rw	0x04	Defines the 2nd AGC Threshold:
	3-0	AgcStep3	rw	0x0b	Defines the 3rd AGC Threshold:
RegAgcThresh3Lf (0x64)	7-4	AgcStep4	rw	0x0c	Defines the 4th AGC Threshold:
	3-0	AgcStep5	rw	0x0c	Defines the 5th AGC Threshold:
RegPllLf (0x70)	7-6	PllBandwidth	rw	0x03	Controls the PLL bandwidth: 00 → 75 kHz 10 → 225 kHz 01 → 150 kHz 11 → 300 kHz
	5-0	reserved	rw	0x10	reserved. Retain default value

Table 44 High Frequency Additional Registers

Name (Address)	Bits	Variable Name	Mode	Default value	Low Frequency Additional Registers
RegAgcRefHf (0x61)	7-6	unused	r	-	unused
	5-0	AgcReferenceLevel	rw	0x1c	Sets the floor reference for all AGC thresholds: AGC Reference[dBm]= -174dBm+10*log(2**RxBw*)+SNR+*AgcReferenceLevel* SNR = 8dB, fixed value
RegAgcThresh1Hf (0x62)	7-5	unused	r	-	unused
	4-0	AgcStep1	rw	0x0e	Defines the 1st AGC Threshold
RegAgcThresh2Hf (0x63)	7-4	AgcStep2	rw	0x05	Defines the 2nd AGC Threshold:
	3-0	AgcStep3	rw	0x0b	Defines the 3rd AGC Threshold:
RegAgcThresh3Hf (0x64)	7-4	AgcStep4	rw	0x0c	Defines the 4th AGC Threshold:
	3-0	AgcStep5	rw	0x0c	Defines the 5th AGC Threshold:
RegPllHf (0x70)	7-6	PllBandwidth	rw	0x03	Controls the PLL bandwidth: 00 → 75 kHz 10 → 225 kHz 01 → 150 kHz 11 → 300 kHz
	5-0	reserved	rw	0x10	reserved. Retain default value

6.4. LoRa™ Mode Register Map

This details the SX1276/77/78/79 register mapping and the precise contents of each register in LoRa™ mode.

It is essential to understand that the LoRa™ modem is controlled independently of the FSK modem. Therefore, care should be taken when accessing the registers, especially as some register may have the same name in LoRa™ or FSK mode.

The LoRa registers are only accessible when the device is set in Lora mode (and, in the same way, the FSK register are only accessible in FSK mode). However, in some cases, it may be necessary to access some of the FSK register while in LoRa mode. To this aim, the *AccesSharedReg* bit was created in the *RegOpMode* register. This bit, when set to '1', will grant access to the FSK register 0x0D up to the register 0x3F. Once the setup has been done, it is strongly recommended to clear this bit so that LoRa register can be accessed normally.

Convention: r: read, w: write, c : set to clear and t: trigger.

Name (Address)	Bits	Variable Name	Mode	Reset	LoRa™ Description
RegFifo (0x00)	7-0	Fifo	rw	0x00	LoRa™ base-band FIFO data input/output. FIFO is cleared an not accessible when device is in SLEEP mode
Common Register Settings					
RegOpMode (0x01)	7	LongRangeMode	rw	0x0	0 → FSK/OOK Mode 1 → LoRa™ Mode This bit can be modified only in Sleep mode. A write operation on other device modes is ignored.
	6	AccessSharedReg	rw	0x0	This bit operates when device is in Lora mode; if set it allows access to FSK registers page located in address space (0x0D:0x3F) while in LoRa mode 0 → Access LoRa registers page 0x0D: 0x3F 1 → Access FSK registers page (in mode LoRa) 0x0D: 0x3F
	5-4	reserved	r	0x00	reserved
	3	LowFrequencyModeOn	rw	0x01	Access Low Frequency Mode registers 0 → High Frequency Mode (access to HF test registers) 1 → Low Frequency Mode (access to LF test registers)
	2-0	Mode	rwt	0x01	Device modes 000 → SLEEP 001 → STDBY 010 → Frequency synthesis TX (FSTX) 011 → Transmit (TX) 100 → Frequency synthesis RX (FSRX) 101 → Receive continuous (RXCONTINUOUS) 110 → receive single (RXSINGLE) 111 → Channel activity detection (CAD)
(0x02)	7-0	reserved	r	0x00	-
(0x03)	7-0	reserved	r	0x00	-
(0x04)	7-0	reserved	rw	0x00	-
(0x05)	7-0	reserved	r	0x00	-
RegFrMsb (0x06)	7-0	Frf(23:16)	rw	0x6c	MSB of RF carrier frequency

Name (Address)	Bits	Variable Name	Mode	Reset	LoRa™ Description
RegFrMid (0x07)	7-0	Frf(15:8)	rw	0x80	MSB of RF carrier frequency
RegFrLsb (0x08)	7-0	Frf(7:0)	rwt	0x00	LSB of RF carrier frequency $f_{RF} = \frac{F(XOSC) \cdot Frf}{2^{19}}$ Resolution is 61.035 Hz if F(XOSC) = 32 MHz. Default value is 0x6c8000 = 434 MHz. Register values must be modified only when device is in SLEEP or STAND-BY mode.
Registers for RF blocks					
RegPaConfig (0x09)	7	PaSelect	rw	0x00	Selects PA output pin 0 → RFO pin. Output power is limited to +14 dBm. 1 → PA_BOOST pin. Output power is limited to +20 dBm
	6-4	MaxPower	rw	0x04	Select max output power: Pmax=10.8+0.6*MaxPower [dBm]
	3-0	OutputPower	rw	0x0f	Pout=Pmax-(15-OutputPower) if PaSelect = 0 (RFO pin) Pout=17-(15-OutputPower) if PaSelect = 1 (PA_BOOST pin)
RegPaRamp (0x0A)	7-5	unused	r	-	unused
	4	reserved	rw	0x00	reserved
	3-0	PaRamp(3:0)	rw	0x09	Rise/Fall time of ramp up/down in FSK 0000 → 3.4 ms 0001 → 2 ms 0010 → 1 ms 0011 → 500 us 0100 → 250 us 0101 → 125 us 0110 → 100 us 0111 → 62 us 1000 → 50 us 1001 → 40 us 1010 → 31 us 1011 → 25 us 1100 → 20 us 1101 → 15 us 1110 → 12 us 1111 → 10 us
RegOcp (0x0B	7-6	unused	r	0x00	unused
	5	OcpOn	rw	0x01	Enables overload current protection (OCP) for PA: 0 → OCP disabled 1 → OCP enabled
	4-0	OcpTrim	rw	0x0b	Trimming of OCP current: Imax = 45+5*OcpTrim [mA] if OcpTrim <= 15 (120 mA) / Imax = -30+10*OcpTrim [mA] if 15 < OcpTrim <= 27 (130 to 240 mA) Imax = 240mA for higher settings Default Imax = 100mA

Name (Address)	Bits	Variable Name	Mode	Reset	LoRa™ Description
RegLna (0x0C)	7-5	LnaGain	rwx	0x01	LNA gain setting: 000 → not used 001 → G1 = maximum gain 010 → G2 011 → G3 100 → G4 101 → G5 110 → G6 = minimum gain 111 → not used
	4-3	LnaBoostLf	rw	0x00	Low Frequency (RFI_LF) LNA current adjustment 00 → Default LNA current Other → Reserved
	2	reserved	rw	0x00	reserved
	1-0	LnaBoostHf	rw	0x00	High Frequency (RFI_HF) LNA current adjustment 00 → Default LNA current 11 → Boost on, 150% LNA current
Lora page registers					
RegFifoAddrPtr (0x0D)	7-0	FifoAddrPtr	rw	0x00	SPI interface address pointer in FIFO data buffer.
RegFifoTxBaseAddr (0x0E)	7-0	FifoTxBaseAddr	rw	0x80	write base address in FIFO data buffer for TX modulator
RegFifoRxBaseAddr (0x0F)	7-0	FifoRxBaseAddr	rw	0x00	read base address in FIFO data buffer for RX demodulator
RegFifoRxCurrentAddr (0x10)	7-0	FifoRxCurrentAddr	r	n/a	Start address (in data buffer) of last packet received
RegIrqFlagsMask (0x11)	7	RxTimeoutMask	rw	0x00	Timeout interrupt mask: setting this bit masks the corresponding IRQ in RegIrqFlags
	6	RxDoneMask	rw	0x00	Packet reception complete interrupt mask: setting this bit masks the corresponding IRQ in RegIrqFlags
	5	PayloadCrcErrorMask	rw	0x00	Payload CRC error interrupt mask: setting this bit masks the corresponding IRQ in RegIrqFlags
	4	ValidHeaderMask	rw	0x00	Valid header received in Rx mask: setting this bit masks the corresponding IRQ in RegIrqFlags
	3	TxDoneMask	rw	0x00	FIFO Payload transmission complete interrupt mask: setting this bit masks the corresponding IRQ in RegIrqFlags
	2	CadDoneMask	rw	0x00	CAD complete interrupt mask: setting this bit masks the corresponding IRQ in RegIrqFlags
	1	FhssChangeChannelMask	rw	0x00	FHSS change channel interrupt mask: setting this bit masks the corresponding IRQ in RegIrqFlags
	0	CadDetectedMask	rw	0x00	Cad Detected Interrupt Mask: setting this bit masks the corresponding IRQ in RegIrqFlags

Name (Address)	Bits	Variable Name	Mode	Reset	LoRa™ Description
RegIrqFlags (0x12)	7	RxTimeout	rc	0x00	Timeout interrupt: writing a 1 clears the IRQ
	6	RxDone	rc	0x00	Packet reception complete interrupt: writing a 1 clears the IRQ
	5	PayloadCrcError	rc	0x00	Payload CRC error interrupt: writing a 1 clears the IRQ
	4	ValidHeader	rc	0x00	Valid header received in Rx: writing a 1 clears the IRQ
	3	TxDone	rc	0x00	FIFO Payload transmission complete interrupt: writing a 1 clears the IRQ
	2	CadDone	rc	0x00	CAD complete: write to clear: writing a 1 clears the IRQ
	1	FhssChangeChannel	rc	0x00	FHSS change channel interrupt: writing a 1 clears the IRQ
	0	CadDetected	rc	0x00	Valid Lora signal detected during CAD operation: writing a 1 clears the IRQ
RegRxNbBytes (0x13)	7-0	FifoRxBytesNb	r	n/a	Number of payload bytes of latest packet received
RegRxHeaderCntValueMsb (0x14)	7-0	ValidHeaderCntMsb(15:8)	r	n/a	Number of valid headers received since last transition into Rx mode, MSB(15:8). Header and packet counters are reseted in Sleep mode.
RegRxHeaderCntValueLsb (0x15)	7-0	ValidHeaderCntLsb(7:0)	r	n/a	Number of valid headers received since last transition into Rx mode, LSB(7:0). Header and packet counters are reseted in Sleep mode.
RegRxPacketCntValueMsb (0x16)	7-0	ValidPacketCntMsb(15:8)	rc	n/a	Number of valid packets received since last transition into Rx mode, MSB(15:8). Header and packet counters are reseted in Sleep mode.
RegRxPacketCntValueLsb (0x17)	7-0	ValidPacketCntLsb(7:0)	r	n/a	Number of valid packets received since last transition into Rx mode, LSB(7:0). Header and packet counters are reseted in Sleep mode.
RegModemStat (0x18)	7-5	RxCodingRate	r	n/a	Coding rate of last header received
	4	ModemStatus	r	‘1’	Modem clear
	3		r	‘0’	Header info valid
	2		r	‘0’	RX on-going
	1		r	‘0’	Signal synchronized
	0		r	‘0’	Signal detected
RegPktSnrValue (0x19)	7-0	PacketSnr	r	n/a	Estimation of SNR on last packet received.In two’s compliment format mutiplied by 4. $$SNR[dB] = \frac{PacketSnr[twos\ complement]}{4}$$

Name (Address)	Bits	Variable Name	Mode	Reset	LoRa™ Description
RegPktRssiValue (0x1A)	7-0	PacketRssi	r	n/a	RSSI of the latest packet received (dBm): RSSI[dBm] = -157 + Rssi (using HF output port, SNR >= 0) or RSSI[dBm] = -164 + Rssi (using LF output port, SNR >= 0) (see section 5.5.5 for details)
RegRssiValue (0x1B)	7-0	Rssi	r	n/a	Current RSSI value (dBm) RSSI[dBm] = -157 + Rssi (using HF output port) or RSSI[dBm] = -164 + Rssi (using LF output port) (see section 5.5.5 for details)
RegHopChannel (0x1C)	7	PllTimeout	r	n/a	PLL failed to lock while attempting a TX/RX/CAD operation 1 → PLL did not lock 0 → PLL did lock
	6	CrcOnPayload	r	n/a	CRC Information extracted from the received packet header (Explicit header mode only) 0 → Header indicates CRC off 1 → Header indicates CRC on
	5-0	FhssPresentChannel	r	n/a	Current value of frequency hopping channel in use.
RegModemConfig 1 (0x1D)	7-4	Bw	rw	0x07	Signal bandwidth: 0000 → 7.8 kHz 0001 → 10.4 kHz 0010 → 15.6 kHz 0011 → 20.8kHz 0100 → 31.25 kHz 0101 → 41.7 kHz 0110 → 62.5 kHz 0111 → 125 kHz 1000 → 250 kHz 1001 → 500 kHz other values → reserved In the lower band (169MHz), signal bandwidths 8&9 are not supported)
	3-1	CodingRate	rw	'001'	Error coding rate 001 → 4/5 010 → 4/6 011 → 4/7 100 → 4/8 All other values → reserved In implicit header mode should be set on receiver to determine expected coding rate. See 4.1.1.3
	0	ImplicitHeaderModeOn	rw	0x0	0 → Explicit Header mode 1 → Implicit Header mode

Name (Address)	Bits	Variable Name	Mode	Reset	LoRa™ Description
RegModemConfig 2 (0x1E)	7-4	SpreadingFactor	rw	0x07	SF rate (expressed as a base-2 logarithm) 6 → 64 chips / symbol 7 → 128 chips / symbol 8 → 256 chips / symbol 9 → 512 chips / symbol 10 → 1024 chips / symbol 11 → 2048 chips / symbol 12 → 4096 chips / symbol other values reserved.
	3	TxContinuousMode	rw	0	0 → normal mode, a single packet is sent 1 → continuous mode, send multiple packets across the FIFO (used for spectral analysis)
	2	RxPayloadCrcOn	rw	0x00	Enable CRC generation and check on payload: 0 → CRC disable 1 → CRC enable If CRC is needed, RxPayloadCrcOn should be set: - in Implicit header mode: on Tx and Rx side - in Explicit header mode: on the Tx side alone (recovered from the header in Rx side)
	1-0	SymbTimeout(9:8)	rw	0x00	RX Time-Out MSB
RegSymbTimeoutLsb (0x1F)	7-0	SymbTimeout(7:0)	rw	0x64	RX Time-Out LSB RX operation time-out value expressed as number of symbols: $TimeOut = SymbTimeout \cdot Ts$
RegPreambleMsb (0x20)	7-0	PreambleLength(15:8)	rw	0x0	Preamble length MSB, = PreambleLength + 4.25 Symbols See 4.1.1 for more details.
RegPreambleLsb (0x21)	7-0	PreambleLength(7:0)	rw	0x8	Preamble Length LSB
RegPayloadLength (0x22)	7-0	PayloadLength(7:0)	rw	0x1	Payload length in bytes. The register needs to be set in implicit header mode for the expected packet length. A 0 value is not permitted
RegMaxPayloadLength (0x23)	7-0	PayloadMaxLength(7:0)	rw	0xff	Maximum payload length; if header payload length exceeds value a header CRC error is generated. Allows filtering of packet with a bad size.
RegHopPeriod (0x24)	7-0	FreqHoppingPeriod(7:0)	rw	0x0	Symbol periods between frequency hops. (0 = disabled). 1st hop always happen after the 1st header symbol
RegFifoRxByteAddr (0x25)	7-0	FifoRxByteAddrPtr	r	n/a	Current value of RX databuffer pointer (address of last byte written by Lora receiver)

Name (Address)	Bits	Variable Name	Mode	Reset	LoRa™ Description
RegModemConfig 3 (0x26)	7-4	Unused	r	0x00	
	3	LowDataRateOptimize	rw	0x00	0 → Disabled 1 → Enabled; mandated for when the symbol length exceeds 16ms
	2	AgcAutoOn	rw	0x00	0 → LNA gain set by register LnaGain 1 → LNA gain set by the internal AGC loop
	1-0	Reserved	rw	0x00	Reserved
(0x27)	7-0	PpmCorrection	rw	0x00	Data rate offset value, used in conjunction with AFC
RegFeiMsb (0x28)	7-4	Reserved	r	n/a	Reserved
	3-0	FreqError(19:16)	r	0x0	Estimated frequency error from modem MSB of RF Frequency Error $F_{Error} = \frac{FreqError \times 2^{24}}{F_{xtal}} \times \frac{BW[kHz]}{500}$
RegFeiMid (0x29)	7-0	FreqError(15:8)	r	0x0	Middle byte of RF Frequency Error
RegFeiLsb (0x2A)	7-0	FreqError(7:0)	r	0x0	LSB of RF Frequency Error
(0x2B)	-	Reserved	r	n/a	Reserved
RegRssiWideband (0x2C)	7-0	RssiWideband(7:0)	r	n/a	Wideband RSSI measurement used to locally generate a random number
(0x2D) - (0x30)	-	Reserved	r	n/a	Reserved
RegDetectOptimize (0x31)	7-3	Reserved	r	0xC0	Reserved
	2-0	DetectionOptimize	rw	0x03	LoRa Detection Optimize 0x03 → SF7 to SF12 0x05 → SF6
(0x32)	-	Reserved	r	n/a	Reserved
RegInvertIQ (0x33)	7	Reserved	rw	0x0	Reserved
	6	InvertIQ	rw	0x0	Invert the LoRa I and Q signals 0 → normal mode 1 → I and Q signals are inverted
	5-0	Reserved	rw	0x27	Reserved
(0x34) - (0x36)	7-0	Reserved	r	n/a	Reserved
RegDetectionThreshold (0x37)	7-0	DetectionThreshold	rw	0x0A	LoRa detection threshold 0x0A → SF7 to SF12 0x0C → SF6
(0x38)	-	Reserved	r	n/a	Reserved
RegSyncWord (0x39)	7-0	SyncWord	rw	0x12	LoRa Sync Word Value 0x34 is reserved for LoRaWAN networks

Name (Address)	Bits	Variable Name	Mode	Reset	LoRa™ Description
(0x3A) - (0x3F)	-	Reserved	r	n/a	Reserved

7. Application Information

7.1. Crystal Resonator Specification

Table 45 shows the crystal resonator specification for the crystal reference oscillator circuit of the SX1276/77/78/79. This specification covers the full range of operation of the SX1276/77/78/79 and is employed in the reference design.

Table 45 Crystal Specification

Symbol	Description	Conditions	Min	Typ	Max	Unit
FXOSC	XTAL Frequency		-	32	-	MHz
RS	XTAL Serial Resistance		-	15	100	ohms
C0	XTAL Shunt Capacitance		-	1	3	pF
CFOOT	External Foot Capacitance	On each pin XTA and XTB	10	15	22	pF
CLOAD	Crystal Load Capacitance		6	-	12	pF

Notes - the initial frequency tolerance, temperature stability and aging performance should be chosen in accordance with the target operating temperature range and the receiver bandwidth selected.

- the loading capacitance should be applied externally, and adapted to the actual Cload specification of the XTAL.

7.2. Reset of the Chip

A power-on reset of the SX1276/77/78/79 is triggered at power up. Additionally, a manual reset can be issued by controlling pin 7.

7.2.1. POR

If the application requires the disconnection of VDD from the SX1276/77/78/79, despite of the extremely low Sleep Mode current, the user should wait for 10 ms from of the end of the POR cycle before commencing communications over the SPI bus. Pin 7 (NRESET) should be left floating during the POR sequence.

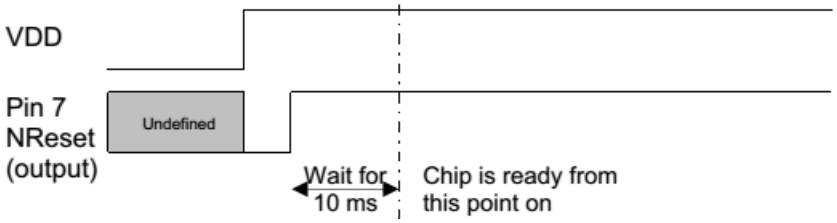


Figure 42. POR Timing Diagram

Please note that any CLKOUT activity can also be used to detect that the chip is ready.

7.2.2. Manual Reset

A manual reset of the SX1276/77/78/79 is possible even for applications in which VDD cannot be physically disconnected. Pin 7 should be pulled low for a hundred microseconds, and then released. The user should then wait for 5 ms before using the chip.

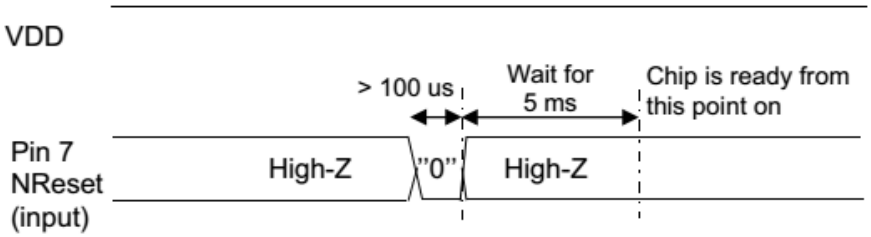


Figure 43. Manual Reset Timing Diagram

Note whilst pin 7 is driven low, an over current consumption of up to one milliampere can be seen on VDD.

7.3. Top Sequencer: Listen Mode Examples

In this scenario, the circuit spends most of the time in Idle mode, during which only the RC oscillator is on. Periodically the receiver wakes up and looks for incoming signal. If a wanted signal is detected, the receiver is kept on and data are analyzed. Otherwise, if there was no wanted signal for a defined period of time, the receiver is switched off until the next receive period.

During Listen mode, the Radio stays most of the time in a Low Power mode, resulting in very low average power consumption. The general timing diagram of this scenario is given in Figure 44.

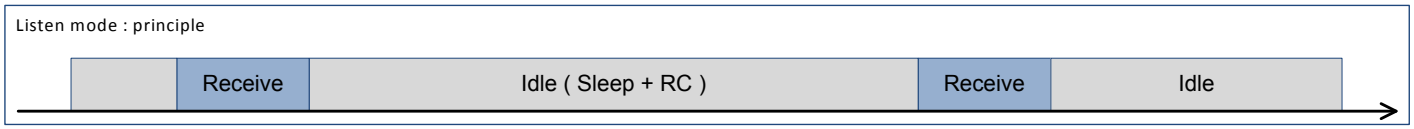


Figure 44. Listen Mode: Principle

An interrupt request is generated on a packet reception. The user can then take appropriate action.

Depending on the application and environment, there are several ways to implement Listen mode:

- Wake on a *PreambleDetect* interrupt
- Wake on *a SyncAddress* interrupt
- Wake on *a PayloadReady* interrupt

7.3.1. Wake on Preamble Interrupt

In one possible scenario, the sequencer polls for a Preamble detection. If a preamble signal is detected, the sequencer is switched off and the circuit stays in Receive mode until the user switches modes. Otherwise, the receiver is switched off until the next Rx period.

7.3.1.1. Timing Diagram

When no signal is received, the circuit wakes every Timer1 + Timer2 and switches to Receive mode for a time defined by Timer2, as shown on the following diagram. If no Preamble is detected, it then switches back to Idle mode, i.e. Sleep mode with RC oscillator on.

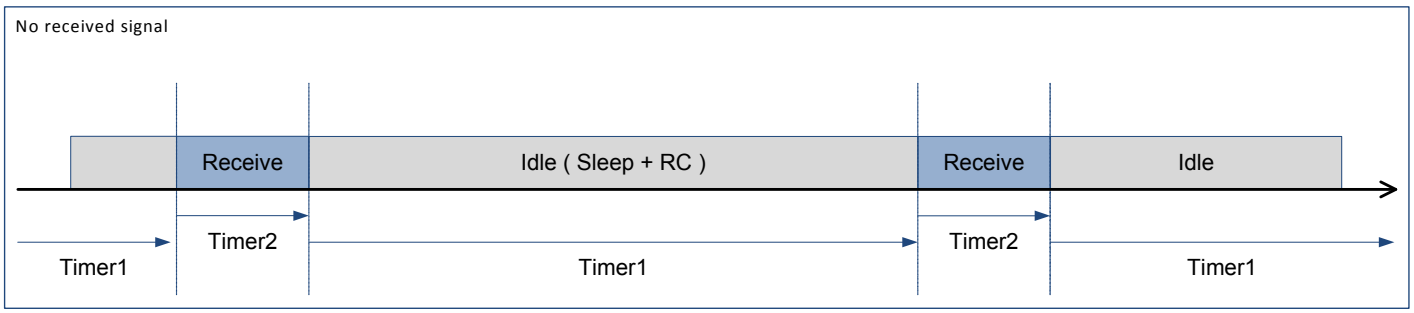


Figure 45. Listen Mode with No Preamble Received

If a Preamble signal is detected, the Sequencer is switched off. The *PreambleDetect* signal can be mapped to DIO4, in order to request the user's attention. The user can then take appropriate action.

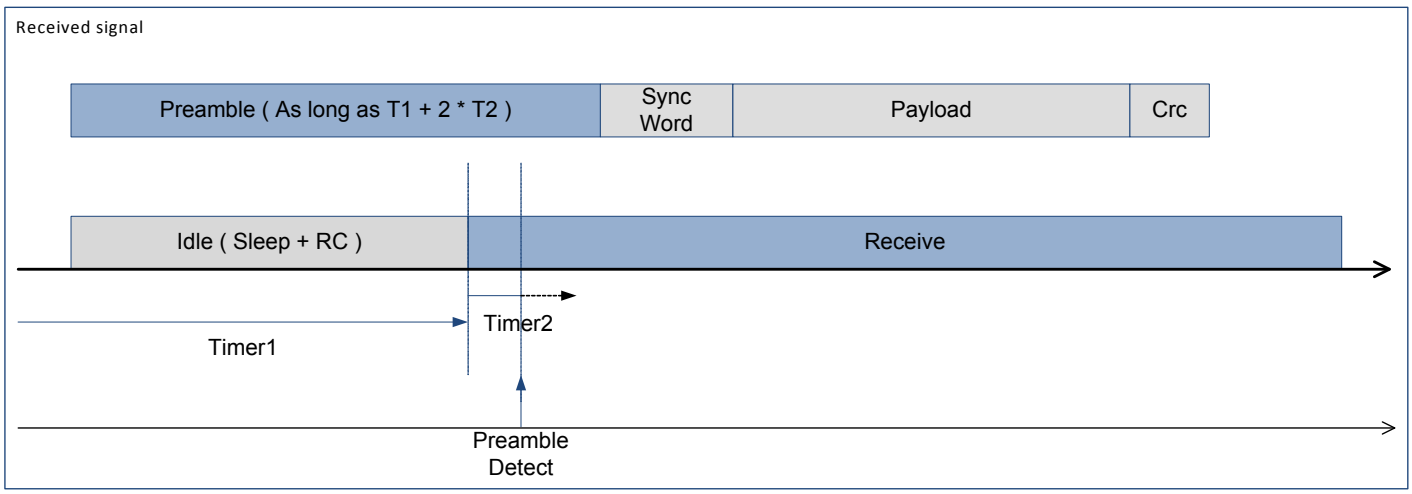


Figure 46. Listen Mode with Preamble Received

7.3.1.2. Sequencer Configuration

The following graph shows Listen mode - Wake on *PreambleDetect* state machine:

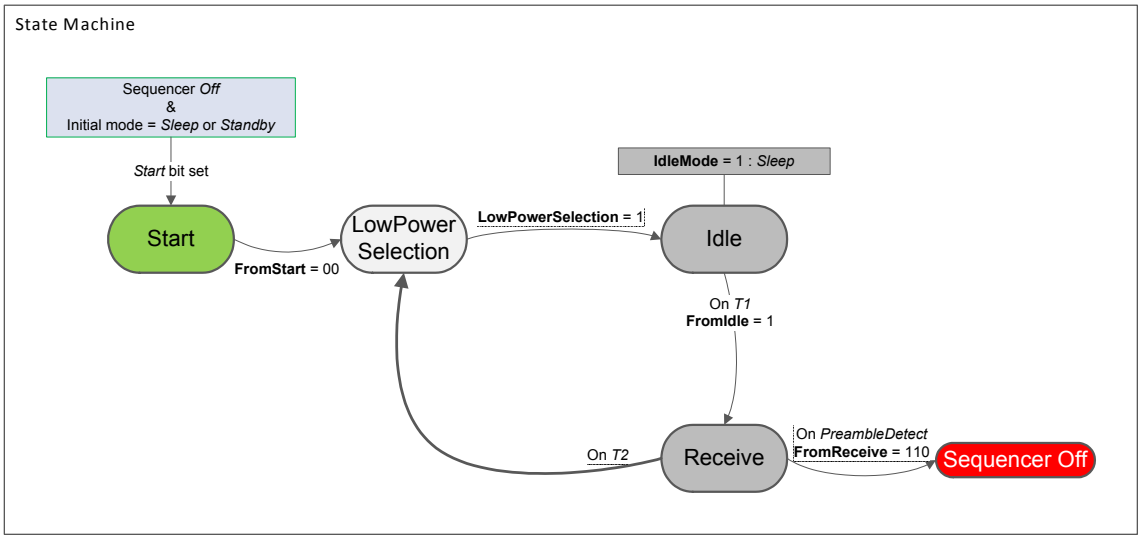


Figure 47. Wake On PreambleDetect State Machine

This example configuration is achieved as follows:

Table 46 Listen Mode with PreambleDetect Condition Settings

Variable	Effect
IdleMode	1: **Sleep** mode
FromStart	00: To **LowPowerSelection**
LowPowerSelection	1: To **Idle** state
FromIdle	1: To **Receive** state on *T1* interrupt
FromReceive	110: To **Sequencer Off** on *PreambleDetect* interrupt

T_{Timer2} defines the maximum duration the chip stays in Receive mode as long as no Preamble is detected. In order to optimize power consumption, Timer2 must be set just long enough for Preamble detection.

T_{Timer1} + T_{Timer2} defines the cycling period, i.e. time between two Preamble polling starts. In order to optimize average power consumption, Timer1 should be relatively long. However, increasing Timer1 also extends packet reception duration.

In order to insure packet detection and optimize the receiver's power consumption, the received packet Preamble should be as long as T_{Timer1} + 2 x T_{Timer2}.

An example of DIO configuration for this mode is described in the following table:

Table 47 Listen Mode with PreambleDetect Condition Recommended DIO Mapping

DIO	Value	Description
0	01	CrcOk
1	00	FifoLevel
3	00	FifoEmpty
4	11	PreambleDetect – Note: *MapPreambleDetect* bit should be set.

7.3.2. Wake on SyncAddress Interrupt

In another possible scenario, the sequencer polls for a Preamble detection and then for a valid *SyncAddress* interrupt. If events occur, the sequencer is switched off and the circuit stays in Receive mode until the user switches modes. Otherwise, the receiver is switched off until the next Rx period.

7.3.2.1. Timing Diagram

Most of the sequencer running time is spent while no wanted signal is received. As shown by the timing diagram in Figure 48, the circuit wakes periodically for a short time, defined by RxTimeout. The circuit is in a Low Power mode for the rest of Timer1 + Timer2 (i.e. Timer1 + Timer2 - TrxTimeout)

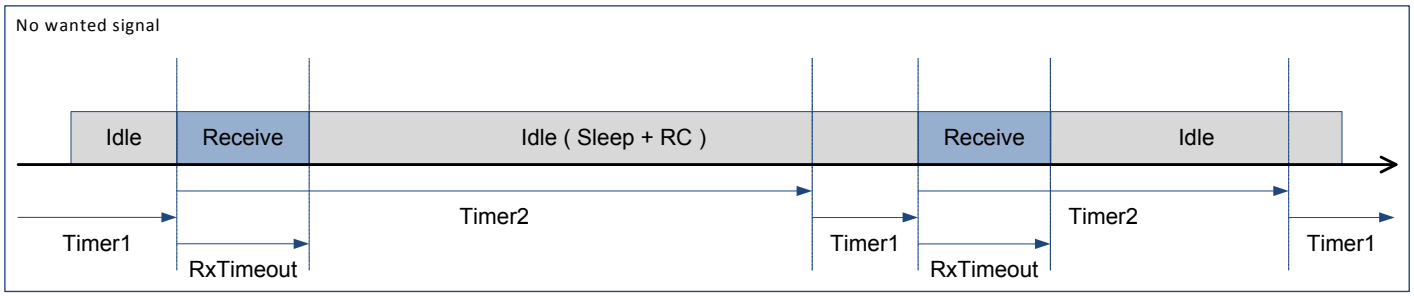


Figure 48. Listen Mode with no SyncAddress Detected

If a preamble is detected before *RxTimeout* timer ends, the circuit stays in Receive mode and waits for a valid *SyncAddress* detection. If none is detected by the end of Timer2, Receive mode is deactivated and the polling cycle resumes, without any user intervention.

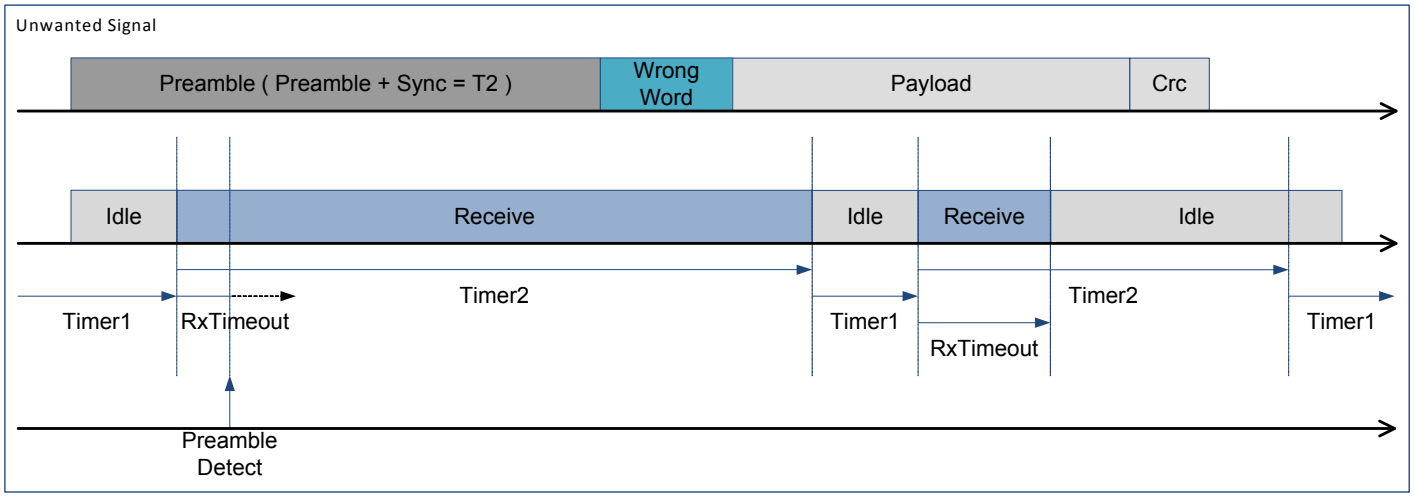


Figure 49. Listen Mode with Preamble Received and no SyncAddress

But if a valid Sync Word is detected, a *SyncAddress* interrupt is fired, the Sequencer is switched off and the circuit stays in Receive mode as long as the user doesn't switch modes.

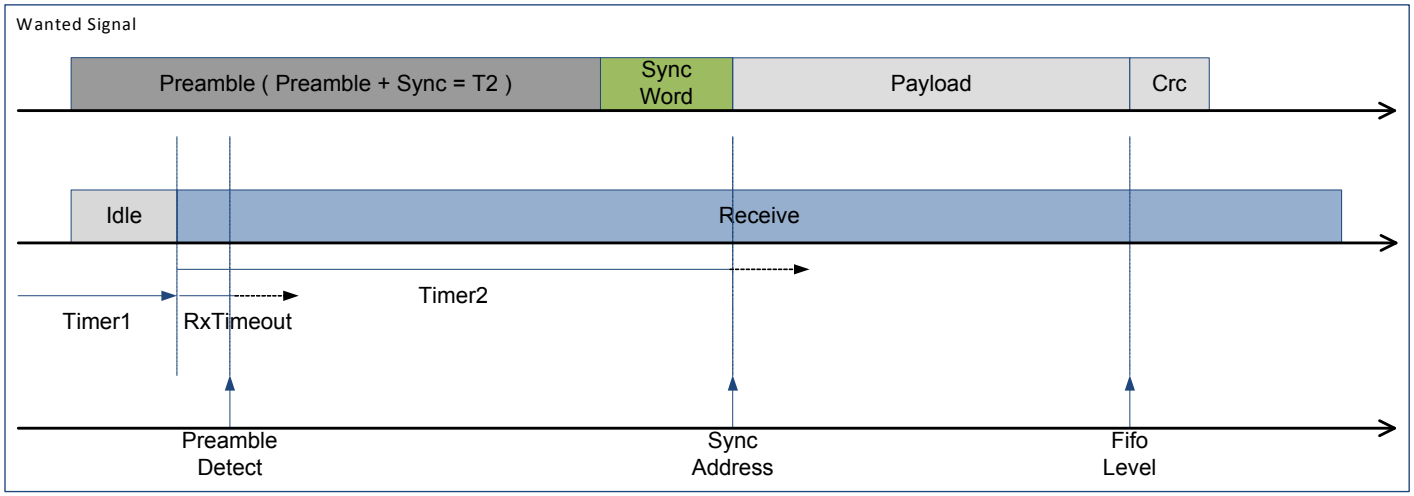


Figure 50. Listen Mode with Preamble Received & Valid SyncAddress

7.3.2.2. Sequencer Configuration

The following graph shows Listen mode - Wake on SyncAddress state machine:

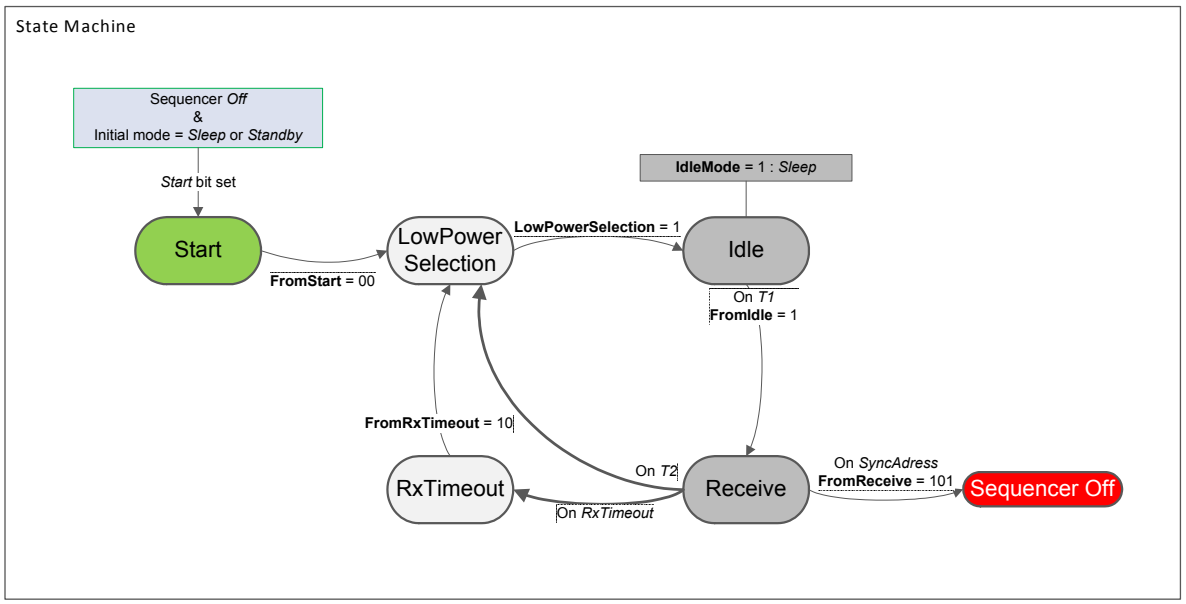


Figure 51. Wake On SyncAddress State Machine

This example configuration is achieved as follows:

Table 48 Listen Mode with SyncAddress Condition Settings

Variable	Effect
IdleMode	1: **Sleep** mode
FromStart	00: To **LowPowerSelection**
LowPowerSelection	1: To **Idle** state
FromIdle	1: To **Receive** state on *T1* interrupt
FromReceive	101: To **Sequencer off** on *SyncAddress* interrupt
FromRxTimeout	10: To **LowPowerSelection**

$T_{TimeoutRxPreamble}$ should be set to just long enough to catch a preamble (depends on *PreambleDetectSize* and *BitRate*).

T_{Timer1} should be set to 64 µs (shortest possible duration).

T_{Timer2} is set so that $T_{Timer1} + T_{Timer2}$ defines the time between two start of reception.

In order to insure packet detection and optimize the receiver power consumption, the received packet Preamble should be defined so that $T_{Preamble} = T_{Timer2} - T_{SyncAddress}$ with $T_{SyncAddress} = (SyncSize + 1)*8/BitRate$.

An example of DIO configuration for this mode is described in the following table:

Table 49 Listen Mode with PreambleDetect Condition Recommended DIO Mapping

DIO	Value	Description
0	01	CrcOk
1	00	FifoLevel
2	11	SyncAddress
3	00	FifoEmpty
4	11	PreambleDetect – Note: *MapPreambleDetect* bit should be set.

7.4. Top Sequencer: Beacon Mode

In this mode, a repetitive message is transmitted periodically. If the Payload being sent is always identical, and *PayloadLength* is smaller than the FIFO size, the use of the *BeaconOn* bit in *RegPacketConfig2* together with the Sequencer permit to achieve periodic beacon without any user intervention.

7.4.1. Timing diagram

In this mode, the Radio is switched to Transmit mode every T_{Timer1} + T_{Timer2} and back to Idle mode after *PacketSent*, as shown in the diagram below. The Sequencer insures minimal time is spent in Transmit mode, and therefore power consumption is optimized.

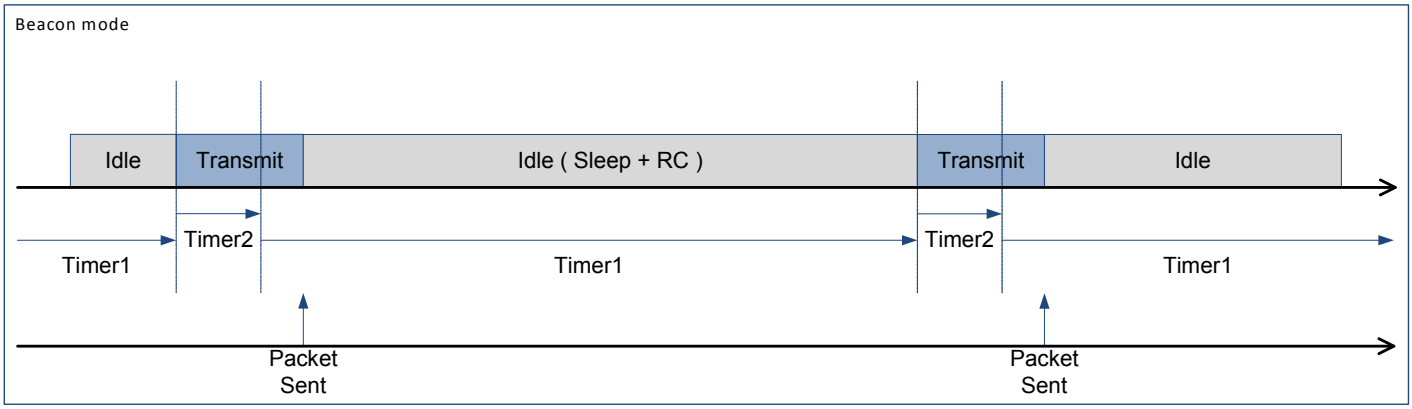


Figure 52. Beacon Mode Timing Diagram

7.4.2. Sequencer Configuration

The Beacon mode state machine is presented in the following graph. It is noticeable that the sequencer enters an infinite loop and can only be stopped by setting *SequencerStop* bit in *RegSeqConfig1*.

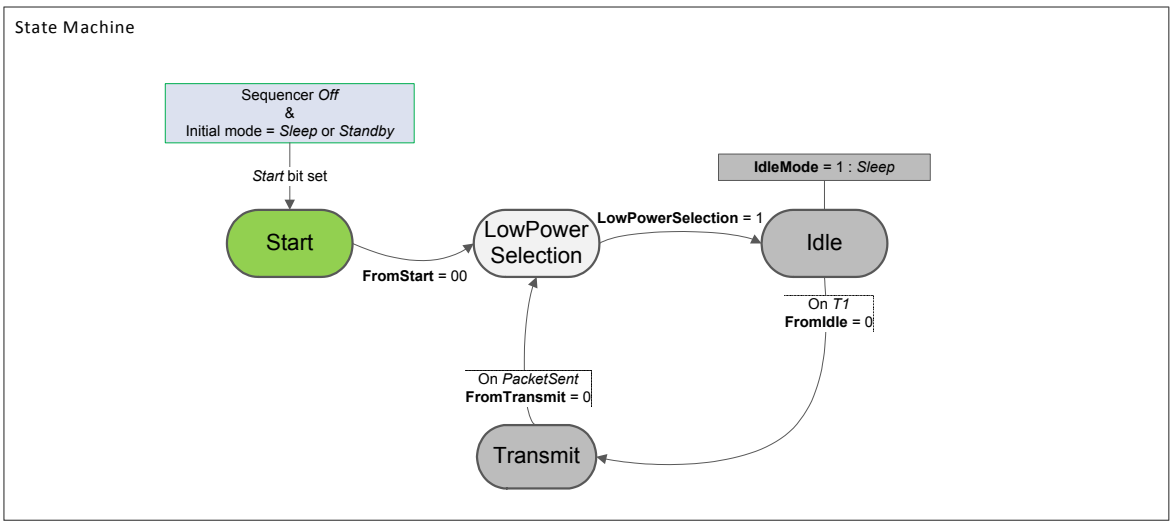


Figure 53. Beacon Mode State Machine

This example is achieved by programming the Sequencer as follows:

Table 50 Beacon Mode Settings

Variable	Effect
IdleMode	1: **Sleep** mode
FromStart	00: To **LowPowerSelection**
LowPowerSelection	1: To **Idle** state
FromIdle	0: To **Transmit** state on *T1* interrupt
FromTransmit	0: To **LowPowerSelection** on *PacketSent* interrupt

$T_{Timer1} + T_{Timer2}$ define the time between the start of two transmissions.

7.5. Example CRC Calculation

The following routine(s) may be implemented to mimic the CRC calculation of the SX1276/77/78/79:

```
// CRC types
#define CRC_TYPE_CCITT                              0
#define CRC_TYPE_IBM                                1

// Polynomial = X^16 + X^12 + X^5 + 1
#define POLYNOMIAL_CCITT                            0x1021
// Polynomial = X^16 + X^15 + X^2 + 1
#define POLYNOMIAL_IBM                              0x8005

// Seeds
#define CRC_IBM_SEED                                0xFFFF
#define CRC_CCITT_SEED                              0x1D0F

/*
 * CRC algorithm implementation
 *
 * \param[IN] crc Previous CRC value
 * \param[IN] data New data to be added to the CRC
 * \param[IN] polynomial CRC polynomial selection [CRC_TYPE_CCITT, CRC_TYPE_IBM]
 *
 * \retval crc New computed CRC
 */
U16 ComputeCrc( U16 crc, U8 data, U16 polynomial )
{
    U8 i;
    for( i = 0; i < 8; i++ )
    {
        if( ( ( ( crc & 0x8000 ) >> 8 ) ^ ( data & 0x80 ) ) != 0 )
        {
            crc <<= 1;              // shift left once
            crc ^= polynomial;      // XOR with polynomial
        }
        else
        {
            crc <<= 1;              // shift left once
        }
        data <<= 1;                 // Next data bit
    }
    return crc;
}

/*
 * CRC algorithm implementation
 *
 * \param[IN] buffer Array containing the data
 * \param[IN] bufferLength Buffer length
 * \param[IN] crcType Selects the CRC polynomial[CRC_TYPE_CCITT, CRC_TYPE_IBM]
 *
 * \retval crc Buffer computed CRC
 */
U16 RadioPacketComputeCrc( U8 *buffer, U8 bufferLength, U8 crcType )
{
    U8 i;
    U16 crc;
    U16 polynomial;

    polynomial = ( crcType == CRC_TYPE_IBM ) ? POLYNOMIAL_IBM : POLYNOMIAL_CCITT;
    crc = ( crcType == CRC_TYPE_IBM ) ? CRC_IBM_SEED : CRC_CCITT_SEED;

    for( i = 0; i < bufferLength; i++ )
    {
        crc = ComputeCrc( crc, buffer[i], polynomial );
    }

    if( crcType == CRC_TYPE_IBM )
    {
        return crc;
    }
    else
    {
        return ( U16 )( ~crc );
    }
}
```

Figure 54. Example CRC Code

7.6. Example Temperature Reading

The following routine(s) may be implemented to read the temperature and calibrate the sensor:

Temperature.c

```
/*!
 * Reads the raw temperature
 * \retval temperature New raw temperature reading in 2's complement format
 */
int8 RadioGetRawTemp( void )
{
    int8_t temp = 0;
    uint8_t previousOpMode;

    // Save current Operation Mode
    SX1276Read( REG_OPMODE, &SX1276->RegOpMode );
    previousOpMode = SX1276->RegOpMode;

    // Pass through LoRa sleep only necessary if reading temperature while in LoRa Mode
    if( ( previousOpMode & RFLR_OPMODE_LONGRANGEMODE_ON ) == RFLR_OPMODE_LONGRANGEMODE_ON )
    {
        SX1276->RegOpMode = RFLR_OPMODE_SLEEP;
        SX1276Write( REG_OPMODE, SX1276->RegOpMode );   // put device in LoRa Sleep Mode
    }

    // Put device in FSK Sleep Mode
    SX1276->RegOpMode = RF_OPMODE_SLEEP;
    SX1276Write( REG_OPMODE, SX1276->RegOpMode );
    // Put device in FSK RxSynth
    SX1276->RegOpMode = RF_OPMODE_SYNTHESIZER_RX;
    SX1276Write( REG_OPMODE, SX1276->RegOpMode );
    // Enable Temperature reading
    SX1276Read( REG_IMAGECAL, &SX1276->RegImageCal );
    SX1276->RegImageCal = ( SX1276->RegImageCal & RF_IMAGECAL_TEMPMONITOR_MASK ) | RF_IMAGECAL_TEMPMONITOR_ON;
    SX1276Write( REG_IMAGECAL, SX1276->RegImageCal );

    // Wait 150us
    Delay( 150 );

    // Disable Temperature reading
    SX1276Read( REG_IMAGECAL, &SX1276->RegImageCal );
    SX1276->RegImageCal = ( SX1276->RegImageCal & RF_IMAGECAL_TEMPMONITOR_MASK ) | RF_IMAGECAL_TEMPMONITOR_OFF;
    SX1276Write( REG_IMAGECAL, SX1276->RegImageCal );

    // Put device in FSK Sleep Mode
    SX1276->RegOpMode = RF_OPMODE_SLEEP;
    SX1276Write( REG_OPMODE, SX1276->RegOpMode );

    // Read temperature
    SX1276Read( REG_TEMP, &SX1276->RegTemp );

    if( ( SX1276->RegTemp & 0x80 ) == 0x80 )
    {
        temp = 255 - SX1276->RegTemp;
    }
    else
    {
        temp = SX1276->RegTemp;
        temp *= -1;
    }
    // We were in LoRa Mode prior to the temperature reading
    if( ( previousOpMode & RFLR_OPMODE_LONGRANGEMODE_ON ) == RFLR_OPMODE_LONGRANGEMODE_ON )
    {
        SX1276->RegOpMode = RFLR_OPMODE_SLEEP;
        SX1276Write( REG_OPMODE, SX1276->RegOpMode );   // put device in LoRa Sleep Mode
    }

    // Reload previous Op Mode
    SX1276Write( REG_OPMODE, previousOpMode );
    return temp;
}
```

Figure 55. Example Temperature Reading

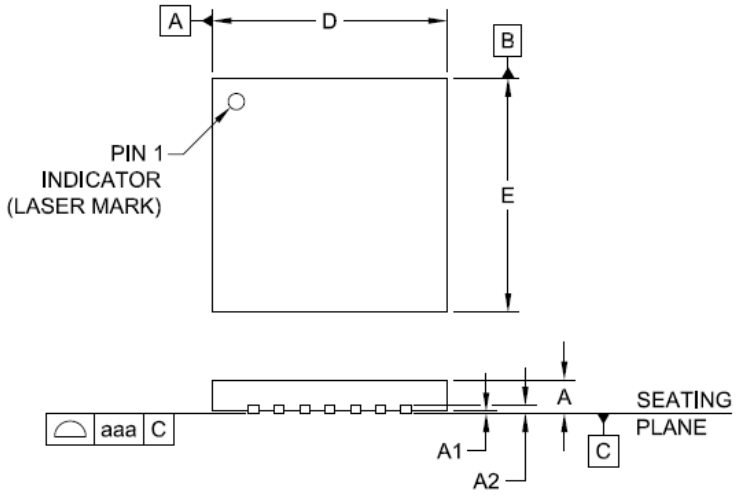
```
68
69  /*!
70   * Computes the temperature compensation factor
71   * \param [IN] actualTemp Actual temperature measured by an external device
72   * \retval compensationFactor Computed compensation factor
73   */
74  S8 RadioCalibreateTemp( S8 actualTemp )
75  {
76      return actualTemp - RadioGetRawTemp( );
77  }
78
79  /*!
80   * Gets the actual compensated temperature
81   * \param [IN] compensationFactor Return value of the calibration function
82   * \retval New compensated temperature value
83   */
84  S8 RadioGetTemp( S8 compensationFactor )
85  {
86      return RadioGetRawTemp( ) + compensationFactor;
87  }
88
89  /*!
90   * Usage example
91   */
92  void main( void )
93  {
94      S8 temp;
95      S8 actualTemp = 0;
96      S8 compensationFactor = 0;
97
98      // Ask user for the temperature during calibration
99      actualTemp = AskUserTemperature( );
100     compensationFactor = RadioCalibreateTemp( actualTemp );
101
102     while( True )
103     {
104         temp = RadioGetTemp( compensationFactor );
105     }
106 }
```

Figure 56. Example Temperature Reading (continued)

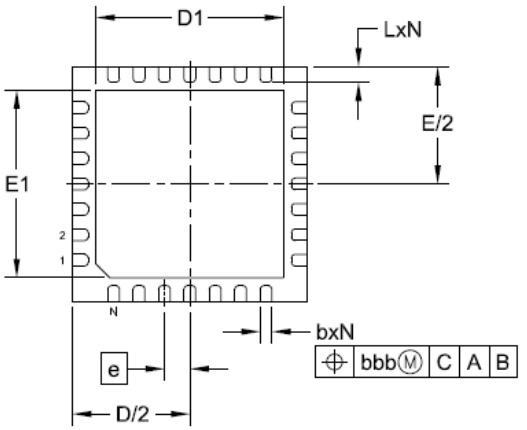
8. Packaging Information

8.1. Package Outline Drawing

The SX1276/77/78/79 is available in a 28-lead QFN package as shown in Figure 57.



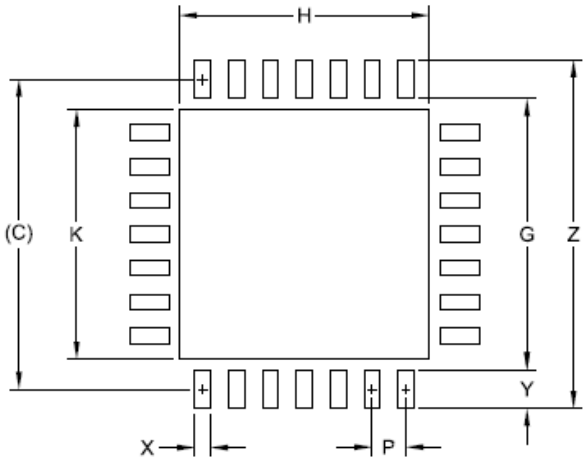
DIMENSIONS			
DIM	MILLIMETERS		
	MIN	NOM	MAX
A	0.80	—	1.00
A1	0.00	—	0.05
A2		(0.20)	
b	0.25	0.30	0.35
D	5.90	6.00	6.10
D1	4.65	4.80	4.90
E	5.90	6.00	6.10
E1	4.65	4.80	4.90
e		0.65 BSC	
L	0.30	0.40	0.50
N		28	
aaa		0.08	
bbb		0.10	



NOTES:
1. CONTROLLING DIMENSIONS ARE IN MILLIMETERS (ANGLES IN DEGREES).
2. COPLANARITY APPLIES TO THE EXPOSED PAD AS WELL AS THE TERMINALS.

Figure 57. Package Outline Drawing

8.2. Recommended Land Pattern



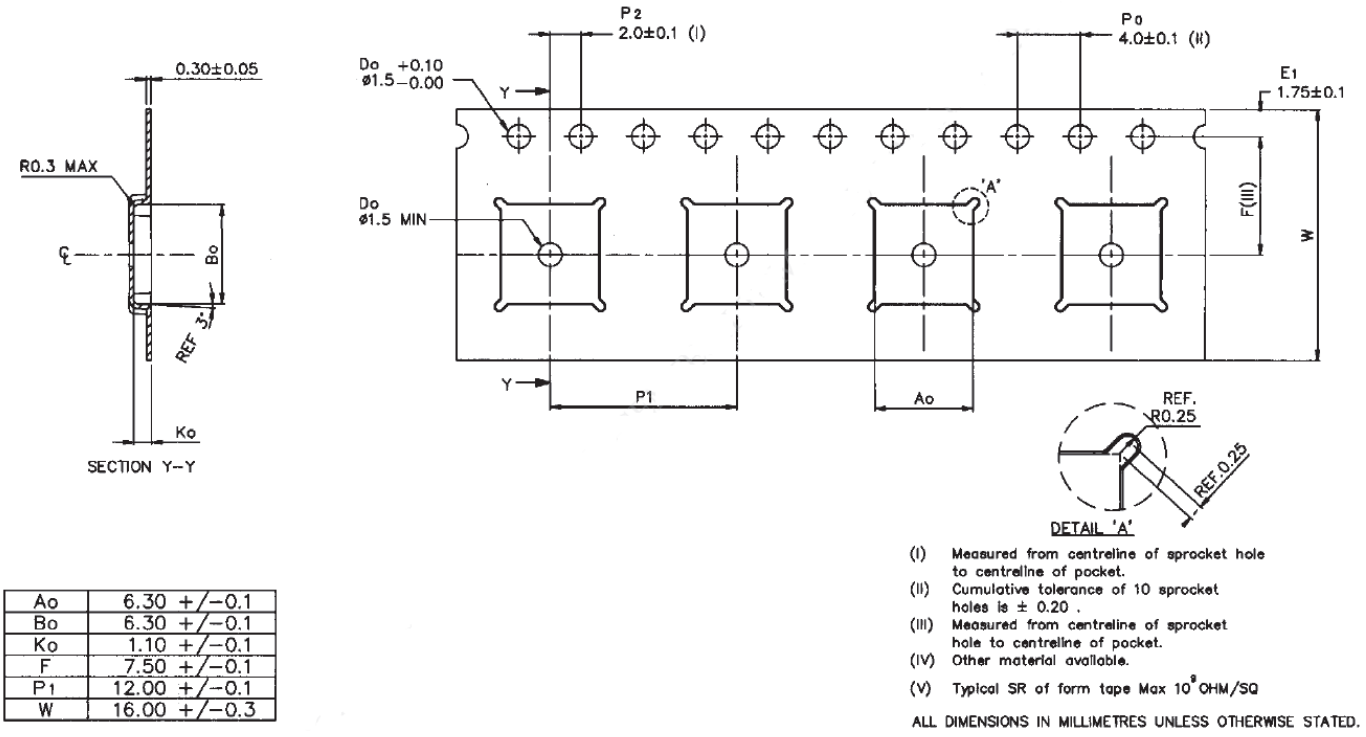
DIMENSIONS	
DIM	MILLIMETERS
C	(5,95)
G	5.20
H	4.80
K	4.80
P	0.65
X	0.30
Y	0.75
Z	6.70

NOTES:

1. CONTROLLING DIMENSIONS ARE IN MILLIMETERS (ANGLES IN DEGREES).
2. THIS LAND PATTERN IS FOR REFERENCE PURPOSE ONLY.
   CONSULT YOUR MANUFACTURING GROUP TO ENSURE YOUR COMPANY'S MANUFACTURING GUIDELINES ARE MET.
3. THERMAL VIAS IN THE LAND PATTERN OF THE EXPOSED PAD SHALL BE CONNECTED TO A SYSTEM GROUND PLANE. FAILURE TO DO SO MAY COMPROMISE THE THERMAL AND/OR FUNCTIONAL PERFORMANCE OF THE DEVICE.
4. SQUARE PACKAGE - DIMENSIONS APPLY IN BOTH " X " AND " Y " DIRECTIONS.

Figure 58. Recommended Land Pattern

8.3. Tape & Reel Information



Ao	6.30 +/–0.1
Bo	6.30 +/–0.1
Ko	1.10 +/–0.1
F	7.50 +/–0.1
P1	12.00 +/–0.1
W	16.00 +/–0.3

Figure 59. Tape and Reel Information

9. Revision History

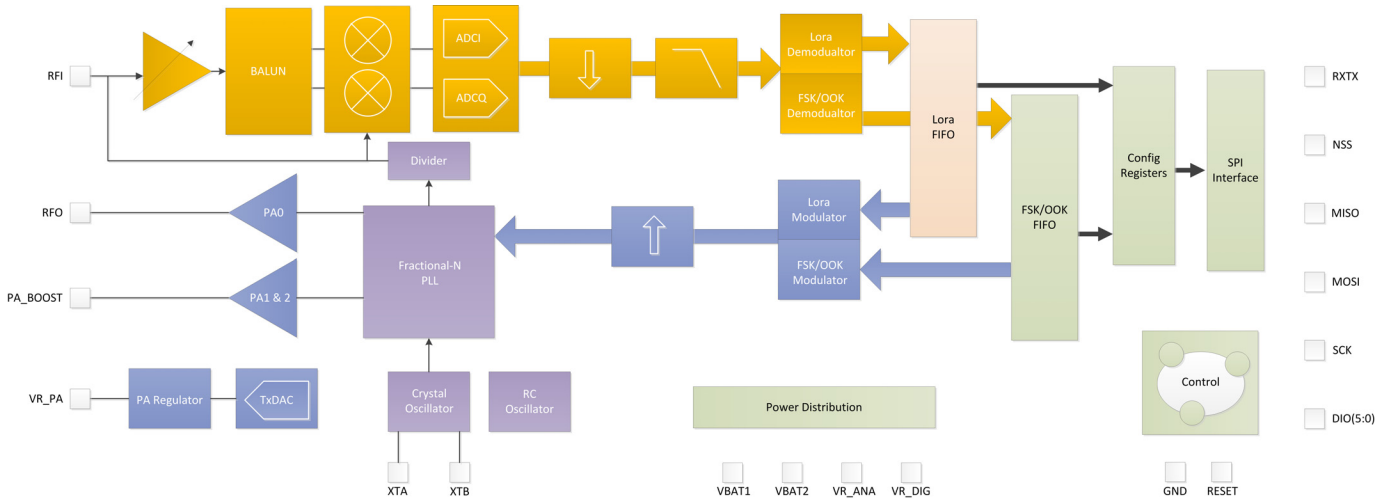
Table 51 Revision History

Revision	Date	Comment
1	Sept 2013	First FINAL release
2	Nov 2014	Miscellaneous typographical corrections Correction of *RxPayloadCrcOn* description Improve description in the RSSI and IQ calibration mechanism Correction of ToA formulae Inclusion of FEI and automatic frequency correction for LoRa Corrected Rssi Formula in Lora mode
3	Nov 2014	Addition of part SX1279
4	March 2015	Clarified operation modes for Rx Single and Rx Continuous mode in LoRa Added use cases for Rx Single and Rx Continuous mode in LoRa mode Clarified used of LoRa RxPayloadCrcOn in Register Table Added description of register RegSyncWord in LoRa register table Changed Stand-By typo into Standby

Contact information

Semtech Corporation
Wireless,Sensing & Timing Products Division
200 Flynn Road, Camarillo, CA 93012
Phone: (805) 498-2111 Fax: (805) 498-3804
E-mail: sales@semtech.com
support_rf@semtech.com
Internet: http://www.semtech.com

SX1272/73 - 860 MHz to 1020 MHz Low Power Long Range Transceiver



GENERAL DESCRIPTION

The SX1272/73 transceivers feature the LoRa™ long range modem that provides ultra-long range spread spectrum communication and high interference immunity whilst minimising current consumption.

Using Semtech's patented LoRa™ modulation technique SX1272/73 can achieve a sensitivity of over -137 dBm using a low cost crystal and bill of materials. The high sensitivity combined with the integrated +20 dBm power amplifier yields industry leading link budget making it optimal for any application requiring range or robustness. LoRa™ also provides significant advantages in both blocking and selectivity over conventional modulation techniques, solving the traditional design compromise between range, interference immunity and energy consumption.

These devices also support high performance (G)FSK modes for systems including WMBus, IEEE802.15.4g. The SX1272/73 deliver exceptional phase noise, selectivity, receiver linearity and IIP3 for significantly lower current consumption than competing devices.

ORDERING INFORMATION

Part Number	Delivery	MOQ / Multiple
SX1272IMLTRT	T&R	3000 pieces
SX1273IMLTRT	T&R	3000 pieces

- QFN 28 Package - Operating Range from -40 to +85°C
- Pb-free, Halogen free, RoHS/WEEE compliant product

KEY PRODUCT FEATURES

- LoRa™ Modem
- 157 dB maximum link budget
- +20 dBm at 100 mW constant RF output vs. V supply
- +14 dBm high efficiency PA
- Programmable bit rate up to 300 kbps
- High sensitivity: down to -137 dBm
- Bullet-proof front end: IIP3 = -12.5 dBm
- 89 dB blocking immunity
- Low RX current of 10 mA, 100 nA register retention
- Fully integrated synthesizer with a resolution of 61 Hz
- FSK, GFSK, MSK, GMSK, LoRa™ and OOK modulation
- Built-in bit synchronizer for clock recovery
- Preamble detection
- 127 dB Dynamic Range RSSI
- Automatic RF Sense and CAD with ultra-fast AFC
- Packet engine up to 256 bytes with CRC
- Built-in temperature sensor and low battery indicator

APPLICATIONS

- Automated Meter Reading
- Home and Building Automation
- Wireless Alarm and Security Systems
- Industrial Monitoring and Control
- Long range Irrigation Systems

Table of contents

Section **Page**

1. General Description 10
1.1. Simplified Block Diagram 10
1.2. Product Versions 11
1.3. Pin Diagram 11
1.4. Pin Description 12
1.5. Package Marking 13
2. Electrical Characteristics 14
2.1. ESD Notice 14
2.2. Absolute Maximum Ratings 14
2.3. Operating Range 14
2.4. Thermal Properties 14
2.5. Chip Specification 15
2.5.1. Power Consumption 15
2.5.2. Frequency Synthesis 15
2.5.3. FSK/OOK Mode Receiver 16
2.5.4. FSK/OOK Mode Transmitter 17
2.5.5. Electrical specification for LoRaTM modulation 18
2.5.6. Digital Specification 20
3. SX1272/73 Features 21
3.1. LoRaTM Modem 22
3.2. FSK/OOK Modem 22
4. SX1272/73 Digital Electronics 23
4.1. The LoRaTM Modem 23
4.1.1. Link Design Using the LoRaTM Modem 24
4.1.1.1. Overview 24
4.1.1.2. Spreading Factor 25
4.1.1.3. Coding Rate 25
4.1.1.4. Signal Bandwidth 26
4.1.1.5. LoRaTM Transmission Parameter Relationship 26
4.1.1.6. LoRaTM Packet Structure 27
4.1.1.7. Time on air 29
4.1.1.8. Frequency Hopping with LoRaTM 29
4.1.2. LoRaTM Digital Interface 31
4.1.2.1. LoRaTM Configuration Registers 31
4.1.2.2. Status Registers 31
4.1.2.3. LoRaTM Mode FIFO Data Buffer 31
4.1.2.4. Interrupts in LoRa Mode 32
4.1.3. Operation of the LoRaTM Modem 33
4.1.3.1. Operating Mode Control 33
4.1.4. Frequency Settings 34
4.1.5. Frequency Error Indication 34
4.1.6. LoRaTM Modem State Machine Sequences 35

Table of contents

Section **Page**

4.1.6.1. Digital IO Pin Mapping 42
4.2. FSK/OOK Modem 43
4.2.1. Bit Rate Setting 43
4.2.2. FSK/OOK Transmission 44
4.2.2.1. FSK Modulation 44
4.2.2.2. OOK Modulation 44
4.2.2.3. Modulation Shaping 44
4.2.3. FSK/OOK Reception 45
4.2.3.1. FSK Demodulator 45
4.2.3.2. OOK Demodulator 45
4.2.3.3. Bit Synchronizer 47
4.2.3.4. Frequency Error Indicator 48
4.2.3.5. AFC 48
4.2.3.6. Preamble Detector 49
4.2.3.7. Image Rejection Mixer 49
4.2.3.8. Image and RSSI Calibration 49
4.2.3.9. Timeout Function 50
4.2.4. Operating Modes in FSK/OOK Mode 50
4.2.5. General Overview 50
4.2.6. Startup Times 51
4.2.6.1. Transmitter Startup Time 51
4.2.6.2. Receiver Startup Time 51
4.2.6.3. Time to RSSI Evaluation 52
4.2.6.4. Tx to Rx Turnaround Time 53
4.2.6.5. Rx to Tx 53
4.2.6.6. Receiver Hopping, Rx to Rx 54
4.2.6.7. Tx to Tx 54
4.2.7. Receiver Startup Options 54
4.2.8. Receiver Restart Methods 55
4.2.8.1. Restart Upon User Request 55
4.2.8.2. Automatic Restart after valid Packet Reception 55
4.2.8.3. Automatic Restart when Packet Collision is Detected 56
4.2.9. Top Level Sequencer 56
4.2.9.1. Sequencer States 56
4.2.9.2. Sequencer Transitions 57
4.2.9.3. Timers 58
4.2.9.4. Sequencer State Machine 60
4.2.10. Data Processing in FSK/OOK Mode 62
4.2.10.1. Block Diagram 62
4.2.10.2. Data Operation Modes 62
4.2.11. FIFO 63
4.2.11.1. Sync Word Recognition 64

Table of contents

Section **Page**

4.2.12. Digital IO Pins Mapping 66
4.2.13. Continuous Mode 67
4.2.13.1. General Description 67
4.2.13.2. Tx Processing 67
4.2.13.3. Rx Processing 68
4.2.14. Packet Mode 68
4.2.14.1. General Description 68
4.2.14.2. Packet Format 69
4.2.14.3. Tx Processing 72
4.2.14.4. Rx Processing 72
4.2.14.5. Handling Large Packets 73
4.2.14.6. Packet Filtering 73
4.2.14.7. DC-Free Data Mechanisms 75
4.2.14.8. Beacon Tx Mode 76
4.2.15. io-homecontrol® Compatibility Mode 76
4.3. SPI Interface 77
5. SX1272/73 Analog & RF Frontend Electronics 79
5.1. Power Supply Strategy 79
5.2. Low Battery Detector 79
5.3. Frequency Synthesis 79
5.3.1. Crystal Oscillator 79
5.3.2. CLKOUT Output 80
5.3.3. PLL 80
5.3.4. RC Oscillator 82
5.4. Transmitter Description 83
5.4.1. Architecture Description 83
5.4.2. RF Power Amplifiers 83
5.4.3. High Power +20 dBm Operation 84
5.4.4. Over Current Protection 85
5.5. Receiver Description 85
5.5.1. Overview 85
5.5.2. Receiver Enabled and Receiver Active States 85
5.5.3. Automatic Gain Control In FSK/OOK Mode 86
5.5.4. RSSI in FSK/OOK Mode 87
5.5.5. RSSI and SNR in LoRaTM Mode 88
5.5.6. Channel Filter 89
5.5.7. Temperature Measurement 89
6. Description of the Registers 91
6.1. Register Table Summary 91
6.2. FSK/OOK Mode Register Map 94
6.3. LoRaTM Mode Register Map 108
7. Application Information 115

Table of contents

Section **Page**

7.1. Crystal Resonator Specification 115
7.2. Reset of the Chip 115
7.2.1. POR 115
7.2.2. Manual Reset 116
7.3. Top Sequencer: Listen Mode Examples 116
7.3.1. Wake on Preamble Interrupt 116
7.3.1.1. Timing Diagram 117
7.3.1.2. Sequencer Configuration 118
7.3.2. Wake on SyncAddress Interrupt 119
7.3.2.1. Timing Diagram 119
7.3.2.2. Sequencer Configuration 120
7.4. Top Sequencer: Beacon Mode 122
7.4.1. Timing diagram 122
7.4.2. Sequencer Configuration 122
7.5. Example CRC Calculation 124
7.6. Example Temperature Reading 125
8. Packaging Information 126
8.1. Package Outline Drawing 126
8.2. Recommended Land Pattern 127
8.3. Tape and Reel Information 128
9. Revision History 129

Table of contents

Section **Page**

Table 1. SX1272/73 Device Variants and Key Parameters 11
Table 2. Pin Description 12
Table 3. Absolute Maximum Ratings 14
Table 4. Operating Range 14
Table 5. Operating Range 14
Table 6. Power Consumption Specification 15
Table 7. Frequency Synthesizer Specification 15
Table 8. Receiver Specification 16
Table 9. Transmitter Specification 17
Table 10. Electrical specifications: LoraTM mode 18
Table 11. Digital Specification 20
Table 12. Example LoRaTM Modem Performances 23
Table 13. Range of Spreading Factors 25
Table 14. Cyclic Coding Overhead 25
Table 15. LoRaTM Operating Mode Functionality 33
Table 16. LoRa CAD Consumption Figures 42
Table 17. DIO Mapping LoRaTM Mode 42
Table 18. Bit Rate Examples 43
Table 19. Preamble Detector Settings 49
Table 20. RxTrigger Settings to Enable Timeout Interrupts 50
Table 21. Basic Transceiver Modes 50
Table 22. Receiver Startup Time Summary 52
Table 23. Receiver Startup Options 55
Table 24. Sequencer States 56
Table 25. Sequencer Transition Options 57
Table 26. Sequencer Timer Settings 59
Table 27. Status of FIFO when Switching Between Different Modes of the Chip 64
Table 28. DIO Mapping, Continuous Mode 66
Table 29. DIO Mapping, Packet Mode 66
Table 30. CRC Description 74
Table 31. Power Amplifier Mode Selection Truth Table 83
Table 32. High Power Settings 84
Table 33. Operating Range, +20 dBm Operation 84
Table 34. Operating Range, +20 dBm Operation 84
Table 35. Trimming of the OCP Current 85
Table 36. LNA Gain Control and Performances 86
Table 37. RssiSmoothing Options 88
Table 38. Available RxBw Settings 89
Table 39. Registers Summary 91

Table of contents

Section **Page**

Table 40. Register Map 94
Table 41. Register Map, LoRa Mode 108
Table 42. Crystal Specification 115
Table 43. Listen Mode with PreambleDetect Condition Settings 118
Table 44. Listen Mode with PreambleDetect Condition Recommended DIO Mapping 118
Table 45. Listen Mode with SyncAddress Condition Settings 120
Table 46. Listen Mode with PreambleDetect Condition Recommended DIO Mapping 121
Table 47. Beacon Mode Settings 123
Table 48. Revision History 129

Table of contents

Section **Page**

Figure 1. SX1272/73 Block Diagram 10
Figure 2. Pin Diagram 11
Figure 3. Package Marking 13
Figure 4. Simplified SX1272 Block Schematic Diagram 21
Figure 5. LoRaTM Modem Connectivity 24
Figure 6. LoRaTM Packet Structure 27
Figure 7. Interrupts generated in the case of successful frequency hopping communication. 30
Figure 8. LoRaTM data buffer 31
Figure 9. Applied versus measured frequency offset and influence on PER. 34
Figure 10. LoRaTM modulation transmission sequence. 35
Figure 11. LoRaTM receive sequence. 36
Figure 12. LoRaTM CAD flow 40
Figure 13. Channel activity detection (CAD) time as a function of spreading factor. 41
Figure 14. Consumption Profile of the LoRa CAD Process 42
Figure 15. OOK Peak Demodulator Description 45
Figure 16. Floor Threshold Optimization 46
Figure 17. Bit Synchronizer Description 47
Figure 18. Startup Process 51
Figure 19. Time to Rssi Sample 52
Figure 20. Tx to Rx Turnaround 53
Figure 21. Rx to Tx Turnaround 53
Figure 22. Receiver Hopping 54
Figure 23. Transmitter Hopping 54
Figure 24. Timer1 and Timer2 Mechanism 58
Figure 25. Sequencer State Machine 60
Figure 26. SX1272/73 Data Processing Conceptual View 62
Figure 27. FIFO and Shift Register (SR) 63
Figure 28. FifoLevel IRQ Source Behavior 64
Figure 29. Sync Word Recognition 65
Figure 30. Continuous Mode Conceptual View 67
Figure 31. Tx Processing in Continuous Mode 67
Figure 32. Rx Processing in Continuous Mode 68
Figure 33. Packet Mode Conceptual View 69
Figure 34. Fixed Length Packet Format 70
Figure 35. Variable Length Packet Format 71
Figure 36. Unlimited Length Packet Format 71
Figure 37. Manchester Encoding/Decoding 75
Figure 38. Data Whitening Polynomial 76
Figure 39. SPI Timing Diagram (single access) 77

Table of contents

Section **Page**

Figure 40. TCXO Connection 79
Figure 41. Typical Phase Noise Performances of the Low Consumption and Low Phase Noise PLLs. 81
Figure 42. RF Front-end Architecture Shows the Internal PA Configuration. 83
Figure 43. Receiver Block Diagram 86
Figure 44. AGC Steps Definition 87
Figure 45. Temperature Sensor Response 90
Figure 46. POR Timing Diagram 115
Figure 47. Manual Reset Timing Diagram 116
Figure 48. Listen Mode: Principle 116
Figure 49. Listen Mode with No Preamble Received 117
Figure 50. Listen Mode with Preamble Received 117
Figure 51. Wake On PreambleDetect State Machine 118
Figure 52. Listen Mode with no SyncAddress Detected 119
Figure 53. Listen Mode with Preamble Received and no SyncAddress 119
Figure 54. Listen Mode with Preamble Received & Valid SyncAddress 120
Figure 55. Wake On SyncAddress State Machine 120
Figure 56. Beacon Mode Timing Diagram 122
Figure 57. Beacon Mode State Machine 122
Figure 58. Example CRC Code 124
Figure 59. Example Temperature Reading 125
Figure 60. Package Outline Drawing 126
Figure 61. Recommended Land Pattern 127
Figure 62. Tape and Reel Information 128

1. General Description

The SX1272/73 incorporates the LoRa™ spread spectrum modem which is capable of achieving significantly longer range than existing systems based on FSK or OOK modulation. With this new modulation scheme sensitivities 8 dB better than equivalent data rate FSK can be achieved with a low-cost, low-tolerance crystal reference. This increase in link budget provides much longer range and robustness without the need for a TCXO or external amplification. LoRa™ Also provides significant advances in selectivity and blocking performance, further improving communication reliability. For maximum flexibility the user may decide on the spread spectrum modulation bandwidth (BW), spreading factor (SF) and error correction rate (CR). Another benefit of the spread modulation is that each spreading factor is orthogonal - thus multiple transmitted signals can occupy the same channel without interfering. This also permits simple coexistence with existing FSK based systems. Standard GFSK, FSK, OOK, and GMSK modulation is also provided to allow compatibility with existing systems or standards such as wireless MBUS and IEEE 802.15.4g.

The SX1272 offers three bandwidth options of 125 kHz, 250 kHz, and 500 kHz with spreading factors ranging from 6 to 12. The SX1273 offers the same bandwidth options with spreading factors from 6 to 9.

1.1. Simplified Block Diagram

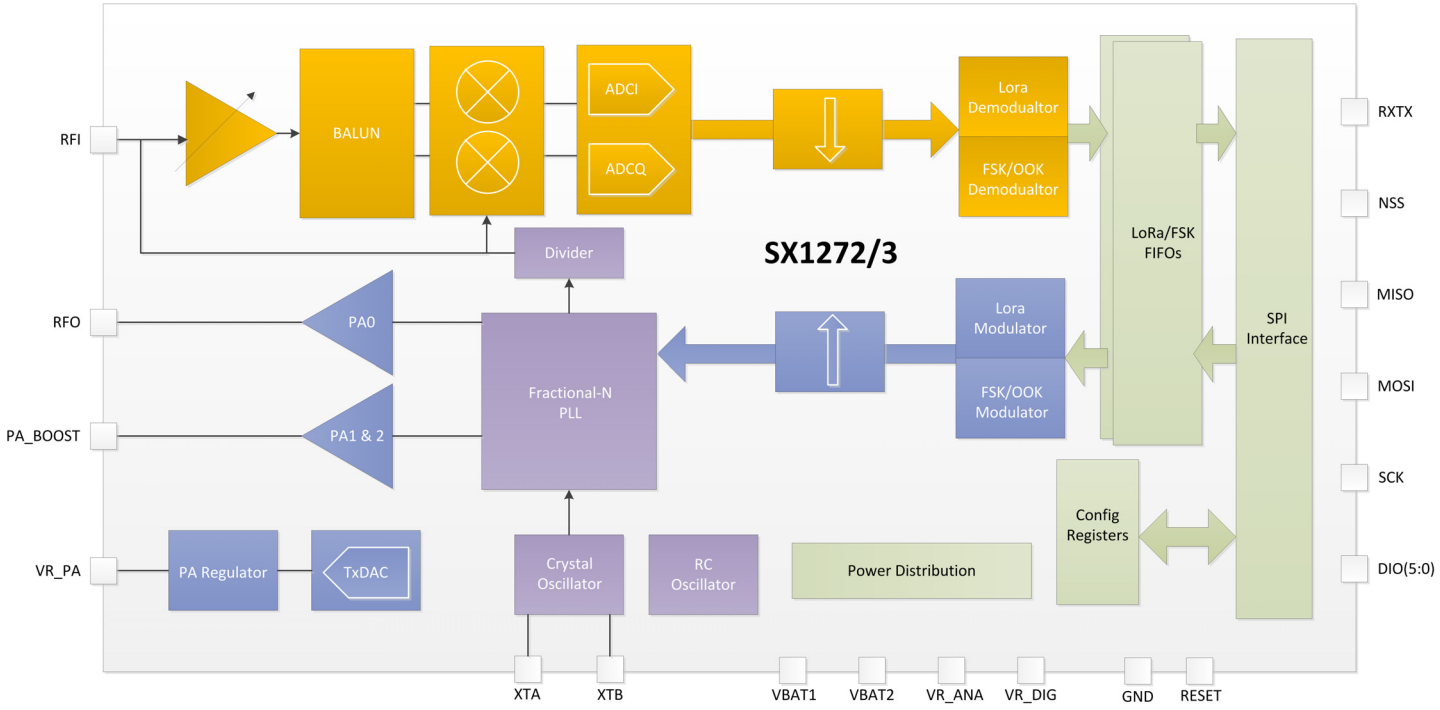


Figure 1. SX1272/73 Block Diagram

1.2. Product Versions

The features of the two product variants SX1272 and SX1273 are detailed in the following table.

Table 1 SX1272/73 Device Variants and Key Parameters

Part Number	Frequency Range	LoRa™ Parameters			
		Spreading Factor	Bandwidth	Effective Bitrate	Sensitivity
SX1272	860 - 1020 MHz	6 - 12	125 - 500 kHz	0.24 - 37.5 kbps	-117 to -137 dBm
SX1273	860 - 1020 MHz	6 - 9	125 - 500 kHz	1.7 - 37.5 kbps	-117 to -130 dBm

1.3. Pin Diagram

The following diagram shows the pin arrangement of the QFN package, top view.

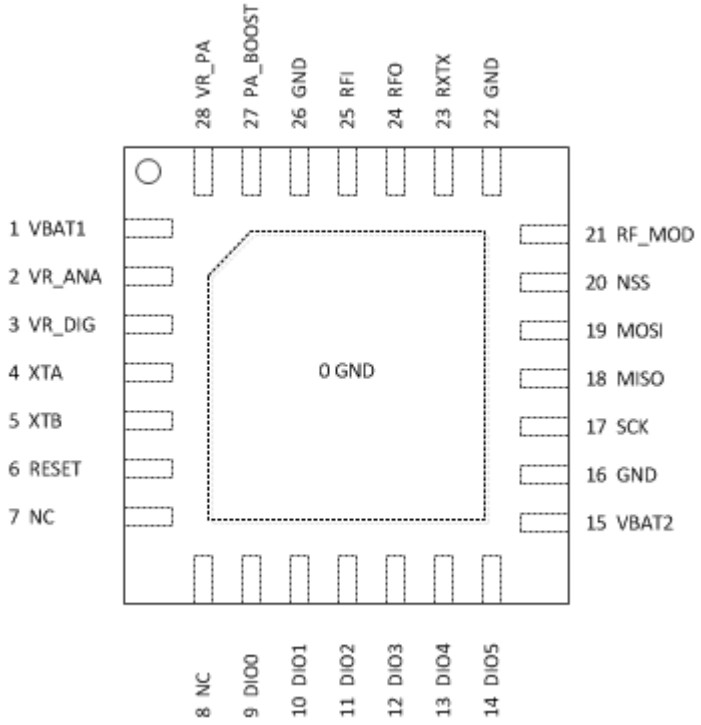


Figure 2. Pin Diagram

1.4. Pin Description

Table 2 Pin Description

Number	Name	Type	Description
0	GROUND	-	Exposed ground pad
1	VBAT1	-	Supply voltage
2	VR_ANA	-	Regulated supply voltage for analogue circuitry
3	VR_DIG	-	Regulated supply voltage for digital blocks
4	XTA	I/O	XTAL connection or TCXO input
5	XTB	I/O	XTAL connection
6	RESET	I/O	Reset trigger input
7	NC	-	Can be connected to Ground
8	NC	-	Can be connected to Ground
9	DIO0	I/O	Digital I/O, software configured
10	DIO1/DCLK	I/O	Digital I/O, software configured
11	DIO2/DATA	I/O	Digital I/O, software configured
12	DIO3	I/O	Digital I/O, software configured
13	DIO4	I/O	Digital I/O, software configured
14	DIO5	I/O	Digital I/O, software configured
15	VBAT2	-	Supply voltage
16	GND	-	Ground
17	SCK	I	SPI Clock input
18	MISO	O	SPI Data output
19	MOSI	I	SPI Data input
20	NSS	I	SPI Chip select input
21	RF_MOD	O	NC
22	GND	O	Ground
23	RXTX	O	Rx/Tx switch control: high in Tx
24	RFO	O	RF output
25	RFI	I	RF input
26	GND	O	Ground
27	PA_BOOST	O	Optional high-power PA output
28	VR_PA	O	Regulated supply for the PA

1.5. Package Marking

TOP MARK	
CHAR	ROWS
7/7/7/7/7	5

Marking for the 6 x 6 mm MLPQ 28 Lead package:

nnnn = Part Number (Example: 1272)
yyww = Date Code (Example: 1352)
xxxxxxx = Semtech Lot No. (Example: E901010
xxxxxxx 0101-10)

Figure 3. Package Marking

2. Electrical Characteristics

2.1. ESD Notice

The SX1272/73 is a high performance radio frequency device. It satisfies:

- Class II of the JEDEC standard JESD22-A114-B (Human Body Model) on all pins.
- Class III of the JEDEC standard JESD22-C101C (Charged Device Model) on all pins

It should thus be handled with all the necessary ESD precautions to avoid any permanent damage.

2.2. Absolute Maximum Ratings

Stresses above the values listed below may cause permanent device failure. Exposure to absolute maximum ratings for extended periods may affect device reliability.

Table 3 Absolute Maximum Ratings

Symbol	Description	Min	Max	Unit
VDDmr	Supply Voltage	-0.5	3.9	V
Tmr	Temperature	-55	+115	° C
Tj	Junction temperature	-	+125	° C
Pmr	RF Input Level	-	+10	dBm

Note Specific ratings apply to +20 dBm operation (see Section 5.4.3).

2.3. Operating Range

Table 4 Operating Range

Symbol	Description	Min	Max	Unit
VDDop	Supply voltage	1.8	3.7	V
Top	Operational temperature range	-40	+85	°C
Clop	Load capacitance on digital ports	-	25	pF
ML	RF Input Level	-	+10	dBm

Note A specific supply voltage range applies to +20 dBm operation (see Section 5.4.3).

2.4. Thermal Properties

Table 5 Operating Range

Symbol	Description	Min	Typ	Max	Unit
THETA_JA	Package θ_{ja} (Junction to ambient)	-	22.185	-	°C/W
THETA_JC	Package θ_{jc} (Junction to case ground paddle)	-	0.757	-	°C/W

2.5. Chip Specification

The tables below give the electrical specifications of the transceiver under the following conditions: Supply voltage VBAT1 = VBAT2 = VDD = 3.3 V, temperature = 25 °C, *FXOSC* = 32 MHz, F_{RF} = 915 MHz, Pout = +13 dBm, 2 level FSK modulation without pre-filtering, FDA = 5 kHz, Bit Rate = 4.8 kbps and terminated in a matched 50 Ohm impedance, unless otherwise specified. Shared Rx and Tx path matching.

Note Unless otherwise specified, the performance in the 868 MHz band is identical or better.

2.5.1. Power Consumption

Table 6 Power Consumption Specification

Symbol	Description	Conditions	Min	Typ	Max	Unit
IDDSL	Supply current in Sleep mode		-	0.1	1	uA
IDDIDLE	Supply current in Idle mode	RC oscillator enabled	-	1.5	-	uA
IDDST	Supply current in Standby mode	Crystal oscillator enabled	-	1.4	1.6	mA
IDDFS	Supply current in Synthesizer mode	FSRx	-	4.5	-	mA
IDDR	Supply current in Receive mode	*LnaBoost* Off *LnaBoost* On	- -	10.5 11.2	- -	mA
IDDT	Supply current in Transmit mode with impedance matching	RFOP = +20 dBm on PA_BOOST RFOP = +17 dBm on PA_BOOST RFOP = +13 dBm on RFO pin RFOP = + 7 dBm on RFO pin	- - - -	125 90 28 18	- - - -	mA mA mA mA

2.5.2. Frequency Synthesis

Table 7 Frequency Synthesizer Specification

Symbol	Description	Conditions	Min	Typ	Max	Unit
FRF	Synthesizer frequency range	Programmable	860	-	1020	MHz
FXOSC	Crystal oscillator frequency		-	32	-	MHz
TS_OSC	Crystal oscillator wake-up time		-	250	-	us
TS_FS	Frequency synthesizer wake-up time to PllLock signal	From Standby mode	-	60	-	us
TS_HOP	Frequency synthesizer hop time at most 10 kHz away from the target frequency	200 kHz step 1 MHz step 5 MHz step 7 MHz step 12 MHz step 20 MHz step 25 MHz step	- - - - - - -	20 20 50 50 50 50 50	- - - - - - -	us us us us us us us
FSTEP	Frequency synthesizer step	FSTEP = FXOSC/2^{19}	-	61.0	-	Hz
FRC	RC Oscillator frequency	After calibration	-	62.5	-	kHz

BRF	Bit rate, FSK	Programmable values (1)	1.2	-	300	kbps
BRO	Bit rate, OOK	Programmable	1.2	-	32.768	kbps
BRA	Bit Rate Accuracy	ABS(wanted BR - available BR)	-	-	250	ppm
FDA	Frequency deviation, FSK (1)	Programmable FDA + BRF/2 =< 250 kHz	0.6	-	200	kHz

Note For Maximum Bit Rate the maximum modulation index is 0.5.

2.5.3. FSK/OOK Mode Receiver

All receiver tests are performed with RxBw = 10 kHz (Single Side Bandwidth) as programmed in *RegRxBw*, receiving a PN15 sequence. Sensitivities are reported for a 0.1% BER (with Bit Synchronizer enabled), unless otherwise specified. Blocking tests are performed with an unmodulated interferer. The wanted signal power for the Blocking Immunity, ACR, IIP2, IIP3 and AMR tests is set 3 dB above the receiver sensitivity level.

Table 8 Receiver Specification

Symbol	Description	Conditions	Min	Typ	Max	Unit
RFS_F	Direct tie of RFI and RFO pins, shared Rx, Tx paths FSK sensitivity, highest LNA gain.	FDA = 5 kHz, BR = 1.2 kbps FDA = 5 kHz, BR = 4.8 kbps FDA = 40 kHz, BR = 38.4 kbps* FDA = 20 kHz, BR = 38.4 kbps** FDA = 62.5 kHz, BR = 250 kbps***	- - - - -	-119 -115 -105 -106 -92	- - - - -	dBm dBm dBm dBm dBm
	Split RF paths, LnaBoost is turned on, the RF switch insertion loss is not accounted for.	FDA = 5 kHz, BR = 1.2 kbps FDA = 5 kHz, BR = 4.8 kbps FDA = 40 kHz, BR = 38.4 kbps* FDA = 20 kHz, BR = 38.4 kbps** FDA = 62.5 kHz, BR = 250 kbps***	- - - - -	-123 -119 -110 -110 -97	- - - - -	dBm dBm dBm dBm dBm
RFS_O	OOK sensitivity, highest LNA gain shared Rx, Tx paths	BR = 4.8 kbps BR = 32 kbps	- -	-117 -108	- -	dBm dBm
CCR	Co-Channel Rejection		-	-9	-	dB
ACR	Adjacent Channel Rejection	FDA = 2 kHz, BR = 1.2 kbps, RxBw = 5.2 kHz Offset = +/- 25 kHz	-	54	-	dB
		FDA = 5 kHz, BR=4.8kbps Offset = +/- 25 kHz Offset = +/- 50 kHz	- -	50 50	- -	dB dB
BI	Blocking Immunity	Offset = +/- 1 MHz Offset = +/- 2 MHz Offset = +/- 10 MHz	- - -	73 78 87	- - -	dB dB dB
AMR	AM Rejection, AM modulated interferer with 100% modulation depth, fm = 1 kHz, square	Offset = +/- 1 MHz Offset = +/- 2 MHz Offset = +/- 10 MHz	- - -	73 78 87	- - -	dB dB dB

IIP2	2nd Order input intercept point unwanted tones are 20 MHz above the LO	Highest LNA gain	-	+57	-	dBm
IIP3	3rd Order input intercept point unwanted tones are 1 MHz and 1.995 MHz above the LO	Highest LNA gain G1 LNA gain G2, 4dB sensitivity reduction.	- -	-12.5 -8.5	- -	dBm dBm
BW_SSB	Single Side channel filter BW	Programmable	2.7	-	250	kHz
IMR	Image Rejection	Wanted signal power sensitivity +3 dB BER = 0.1%	-	48	-	dB
IMA	Image Attenuation		-	57	-	dB
DR_RSSI	RSSI Dynamic Range	AGC enabled Min Max	- -	-127 0	- -	dBm dBm

* *RxBw = 83 kHz (Single Side Bandwidth)*

** *RxBw = 50 kHz (Single Side Bandwidth)*

*** *RxBw = 250 kHz (Single Side Bandwidth)*

2.5.4. FSK/OOK Mode Transmitter

Table 9 Transmitter Specification

Symbol	Description	Conditions	Min	Typ	Max	Unit
RF_OP	RF output power in 50 ohms on RFO pin (High efficiency PA).	Programmable with steps Max Min	 +11 -	 +14 -1	 - -	 dBm dBm
ΔRF_OP_V	RF output power stability on RFO pin versus voltage supply	VDD = 2.5 V to 3.3 V VDD = 1.8 V to 3.7 V	- -	3 8	- -	dB dB
RF_OPH	RF output power in 50 ohms, on PA_BOOST pin (Regulated PA)	Programmable with 1dB steps Max Min	- -	+17 +2	- -	dBm dBm
RF_OPH_MAX	Max RF output power, on PA_BOOST pin	High power mode	-	+20	-	dBm
ΔRF_OPH_V	RF output power stability on PA_-BOOST pin versus voltage supply.	VDD = 2.4 V to 3.7 V	-	+/-1	-	dB
ΔRF_T	RF output power stability versus temperature on both RF pins.	From T = -40 °C to +85 °C	-	+/-1	-	dB
PHN	Transmitter Phase Noise	Low Consumption PLL, 915 MHz 50 kHz offset 400 kHz offset 1 MHz offset	 - - -	 -102 -114 -120	 - - -	dBc/Hz
		Low Phase Noise PLL, 915 MHz 50 kHz offset 400 kHz offset 1 MHz offset	 - - -	 -106 -117 -122	 - - -	dBc/Hz

ACP	Transmitter adjacent channel power (measured at 25 kHz offset)	BT = 1. Measurement conditions as defined by EN 300 220-1 V2.3.1	-	-	-37	dBm
TS_TR	Transmitter wake up time, to the first rising edge of DCLK	Frequency Synthesizer enabled, *PaRamp* = 10 us, BR = 4.8 kbps	-	120	-	us

2.5.5. Electrical specification for LoRa™ modulation

The table below gives the electrical specifications for the transceiver operating with Lora™ modulation. Following conditions apply unless otherwise specified:

- Supply voltage = 3.3 V.
- Temperature = 25° C.
- f_{XOSC} = 32 MHz.
- Band: f_{RF} = 915 MHz.
- bandwidth (BW) = 125 kHz.
- Spreading Factor (SF) = 12.
- Error Correction Code (EC) = 4/6.
- Packet Error Rate (PER)= 1%
- CRC on payload enabled.
- Output power = 13 dBm in transmission.
- Payload length = 10 bytes.
- Preamble Length = 12 symbols (programmed register *PreambleLength=8*)
- With matched impedances

Symbol	Description	Conditions	Min.	Typ	Max	Unit
IDDR_L	Supply current in receiver LoRa™ mode	LnaBoost Off, BW = 125 kHz LnaBoost Off, BW = 250 kHz LnaBoost Off, BW = 500 kHz	- - -	9.7 10.5 12	- - -	mA mA mA
		LnaBoost On, BW = 125 kHz LnaBoost On, BW = 250 kHz LnaBoost On, BW = 500 kHz	- - -	10.8 11.6 13	- - -	mA mA mA
IDDT_L	Supply current in transmitter mode	RFOP = 13 dBm RFOP = 7 dBm	- -	28 18	- -	mA mA
IDDT_H_L	Supply current in transmitter mode with an external impedance transformation	Using PA_BOOST pin RFOP = 17 dBm	-	90	-	mA
BI_L	Blocking immunity, FRF=868 MHz CW interferer	offset = +/- 1 MHz offset = +/- 2 MHz offset = +/- 10 MHz	-	82.5 86.5 89		dB dB dB
IIP3_L	3rd order input intercept point, highest LNA gain, FRF=868 MHz, CW interferer	F1 = FRF + 1 MHz F2 = FRF + 1.995 MHz	-	-12.5	-	dBm

Table 10 Electrical specifications: Lora™ *mode*

Symbol	Description	Conditions	Min.	Typ	Max	Unit
IIP2_L	2nd order input intercept point, highest LNA gain, FRF = 868 MHz, CW interferer.	F1 = FRF + 20 MHz F2 = FRF+ 20 MHz + Δf	-	57	-	dBm
BR_L	Bit rate, Long-Range Mode	From SF6, CR = 4/5, BW = 500 kHz to SF12, CR = 4/8, BW = 125 kHz	0.24	-	37.5	kbps
RFS_L125	RF sensitivity, Long-Range Mode, highest LNA gain, LNA boost, 125 kHz bandwidth using split Rx/Tx path	SF = 6 SF = 7 SF = 8 SF = 9 SF = 10 SF = 11 SF = 12	- - - - - - -	-121 -124 -127 -130 -133 -135 -137	- - - - - - -	dBm dBm dBm dBm dBm dBm dBm
RFS_L250	RF sensitivity, Long-Range Mode, highest LNA gain, LNA boost, 250 kHz bandwidth using split Rx/Tx path	SF = 6 SF = 7 SF = 8 SF = 9 SF = 10 SF = 11 SF = 12	- - - - - - -	-118 -122 -125 -128 -130 -132 -135	- - - - - - -	dBm dBm dBm dBm dBm dBm dBm
RFS_L500	RF sensitivity, Long-Range Mode, highest LNA gain, LNA boost, 500 kHz bandwidth using split Rx/Tx path	SF = 6 SF = 7 SF = 8 SF = 9 SF = 10 SF = 11 SF = 12	- - - - - - -	-111 -116 -119 -122 -125 -128 -129	- - - - - - -	dBm dBm dBm dBm dBm dBm dBm
CCR_LCW	Co-channel rejection Single CW tone = Sens +6 dB 1% PER	SF = 7 SF = 8 SF = 9 SF = 10 SF = 11 SF = 12	- - - - - -	5 9.5 12 14.4 17 19.5	- - - - - -	dB dB dB dB dB dB
CCR_LL	Co-channel rejection	Interferer is a LoRa™ signal using same BW and same SF. Pw = Sensitivity +3 dB		-6		dB
ACR_LCW	Adjacent channel rejection FRF = 868 MHz	Interferer is 1.5*BW_L from the wanted signal center frequency 1% PER, Single CW tone = Sensitivity + 3 dB SF = 7 SF = 12	 - -	 60 72	 - -	 dB dB
IMR_LCW	Image rejection after calibration	1% PER, Single CW tone = Sens +3 dB	-	66	-	dB
FERR_L	Maximum tolerated frequency offset between transmitter and receiver, no sensitivity degradation	BW_L = 125 kHz BW_L = 250 kHz BW_L = 500 kHz	-30 -60 -120	- - -	30 60 120	kHz kHz kHz

Table 10 Electrical specifications: Lora™ *mode*

2.5.6. Digital Specification

Conditions: Temp = 25° C, VDD = 3.3 V, FXOSC = 32 MHz, unless otherwise specified.

Table 11 Digital Specification

Symbol	Description	Conditions	Min	Typ	Max	Unit
V_{IH}	Digital input level high		0.8	-	-	VDD
V_{IL}	Digital input level low		-	-	0.2	VDD
V_{OH}	Digital output level high	Imax = 1 mA	0.9	-	-	VDD
V_{OL}	Digital output level low	Imax = -1 mA	-	-	0.1	VDD
F_{SCK}	SCK frequency		-	-	10	MHz
t_{ch}	SCK high time		50	-	-	ns
t_{cl}	SCK low time		50	-	-	ns
t_{rise}	SCK rise time		-	5	-	ns
t_{fall}	SCK fall time		-	5	-	ns
t_{setup}	MOSI setup time	From MOSI change to SCK rising edge	30	-	-	ns
t_{hold}	MOSI hold time	From SCK rising edge to MOSI change	20	-	-	ns
t_{nsetup}	NSS setup time	From NSS falling edge to SCK rising edge	30	-	-	ns
t_{nhold}	NSS hold time	From SCK falling edge to NSS rising edge, normal mode	100	-	-	ns
t_{nhigh}	NSS high time between SPI accesses		20	-	-	ns
T_DATA	DATA hold and setup time		250	-	-	ns

3. SX1272/73 Features

This section gives a high-level overview of the functionality of the SX1272/73 low-power, highly integrated transceiver. The following figure shows a simplified block diagram of the SX1272/73.

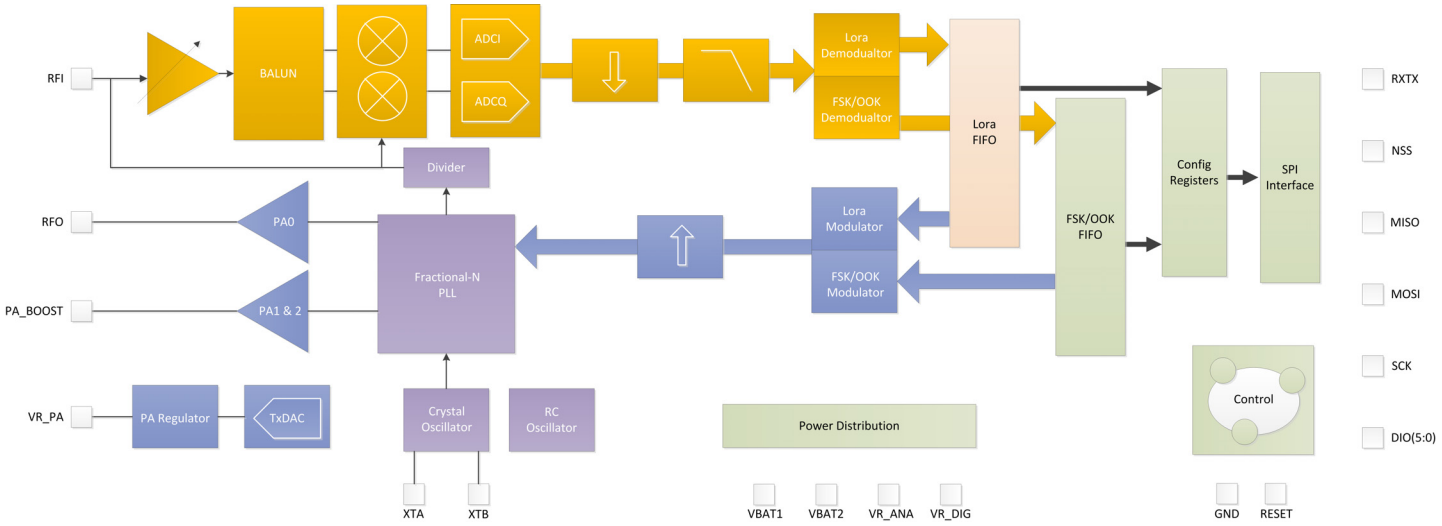


Figure 4. Simplified SX1272 Block Schematic Diagram

SX1272/73 Is a half-duplex, low-IF transceiver. Here the received RF signal is first amplified by the LNA. The LNA input is single ended to minimize the external BoM and for ease of design. Following the LNA output, the conversion to differential is made to improve the second order linearity and harmonic rejection. The signal is then down-converted to in-phase and quadrature (I&Q) components at the intermediate frequency (IF) by the mixer stage. A pair of sigma delta ADCs then perform data conversion, with all subsequent signal processing and demodulation performed in the digital domain. The digital state machine also controls the automatic frequency correction (AFC), received signal strength indicator (RSSI) and automatic gain control (AGC). It also features the higher-level packet and protocol level functionality of the top level sequencer (TLS).

The frequency synthesizer generates the local oscillator (LO) frequency for both receiver and transmitter. The PLL is optimized for user-transparent low lock time and fast auto-calibrating operation. In transmission, frequency modulation is performed digitally within the PLL bandwidth. The PLL also features optional prefiltering of the bit stream to improve spectral purity.

SX1272/73 feature a pair of RF power amplifiers. The first, connected to RFO, can deliver up to +14 dBm, is unregulated for high power efficiency and can be connected directly to the RF receiver input via a pair of passive components to form a single antenna port high efficiency transceiver. The second PA, connected to the PA_BOOST pin, can deliver up to +20 dBm via a dedicated matching network.

SX1272/73 also includes two timing references, an RC oscillator and a 32 MHz crystal oscillator.

All major parameters of the RF front end and digital state machine are fully configurable via an SPI interface which gives access to SX1272/73's configuration registers. This includes a mode auto sequencer that oversees the transition and calibration of the SX1272/73 between intermediate modes of operation in the fastest time possible.

The SX1272/73 are equipped with both standard FSK and long range spread spectrum (LoRa™) modems. Depending upon the mode selected either conventional OOK or FSK modulation may be employed or the LoRa™ spread spectrum modem.

3.1. LoRa™ Modem

The LoRa™ modem uses a proprietary spread spectrum modulation technique. This modulation, in contrast to legacy modulation techniques, permits an increase in link budget and increased immunity to in-band interference. At the same time the frequency tolerance requirement of the crystal reference oscillator is relaxed - allowing a performance increase for a reduction in system cost. For a fuller description of the design trade-offs and operation of the SX1272/73 please consult Section 4.1 of the datasheet.

3.2. FSK/OOK Modem

In FSK/OOK mode the SX1272/73 supports standard modulation techniques including OOK, FSK, GFSK, MSK and GMSK. The SX1272/73 is especially suited to narrow band communication thanks the low-IF architecture employed and the built-in AFC functionality. For full information on the FSK/OOK modem please consult Section 4.2 of this document.

4. SX1272/73 Digital Electronics

4.1. The LoRa™ Modem

The LoRa™ modem uses spread spectrum modulation and forward error correction techniques to increase the range and robustness of radio communication links compared to traditional FSK or OOK based modulation. Examples of the performance improvement possible for several settings are summarised in the table below. The spreading factor and error correction rate are design variables that allow the designer to optimise the trade-off between occupied bandwidth, data rate, link budget improvement and immunity to interference. In the table below a coding rate of 4/5 is used.

Table 12 Example LoRa™ Modem Performances

Bandwidth (kHz)	Spreading Factor	Nominal Rb (bps)	Sensitivity (dBm)
125	6	9380	-122
125	12	293	-137
250	6	18750	-119
250	12	586	-134
500	6	3750	-116
500	12	1172	-131

Typically such performance gains require high stability frequency references, with LoRa™ this is not the case. Low crystal tolerances are easily accommodated reducing the overall BoM cost for a given increase in link budget.

For European operation the range of crystal tolerances acceptable for each sub-band (of the ERC 70-03) is given in the specifications table. For US based operation a frequency hopping mode is available that automates both the LoRa™ spread spectrum and frequency hopping spread spectrum processes.

Another important facet of the LoRa™ modem is its increased immunity to interference. The LoRa™ modem is capable of co-channel GMSK rejection of up to 25 dB. This immunity to interference permits the simple coexistence of LoRa™ modulated systems either in bands of heavy spectral usage or in hybrid communication networks that use LoRa™ to extend range when legacy modulation schemes fail.

4.1.1. Link Design Using the LoRa™ Modem

4.1.1.1. Overview

The LoRa™ modem is setup as shown in the following figure. This configuration permits the simple replacement of the FSK modem with the LoRa™ modem via the configuration register setting *RegOpMode*. This change can be performed on the fly (in Sleep operating mode) thus permitting the use of both standard FSK or OOK in conjunction with the long range capability. The LoRa™ modulation and demodulation process is proprietary, it uses a form of spread spectrum modulation combined with cyclic error correction coding. The combined influence of these two factors is an increase in link budget and enhanced immunity to interference.

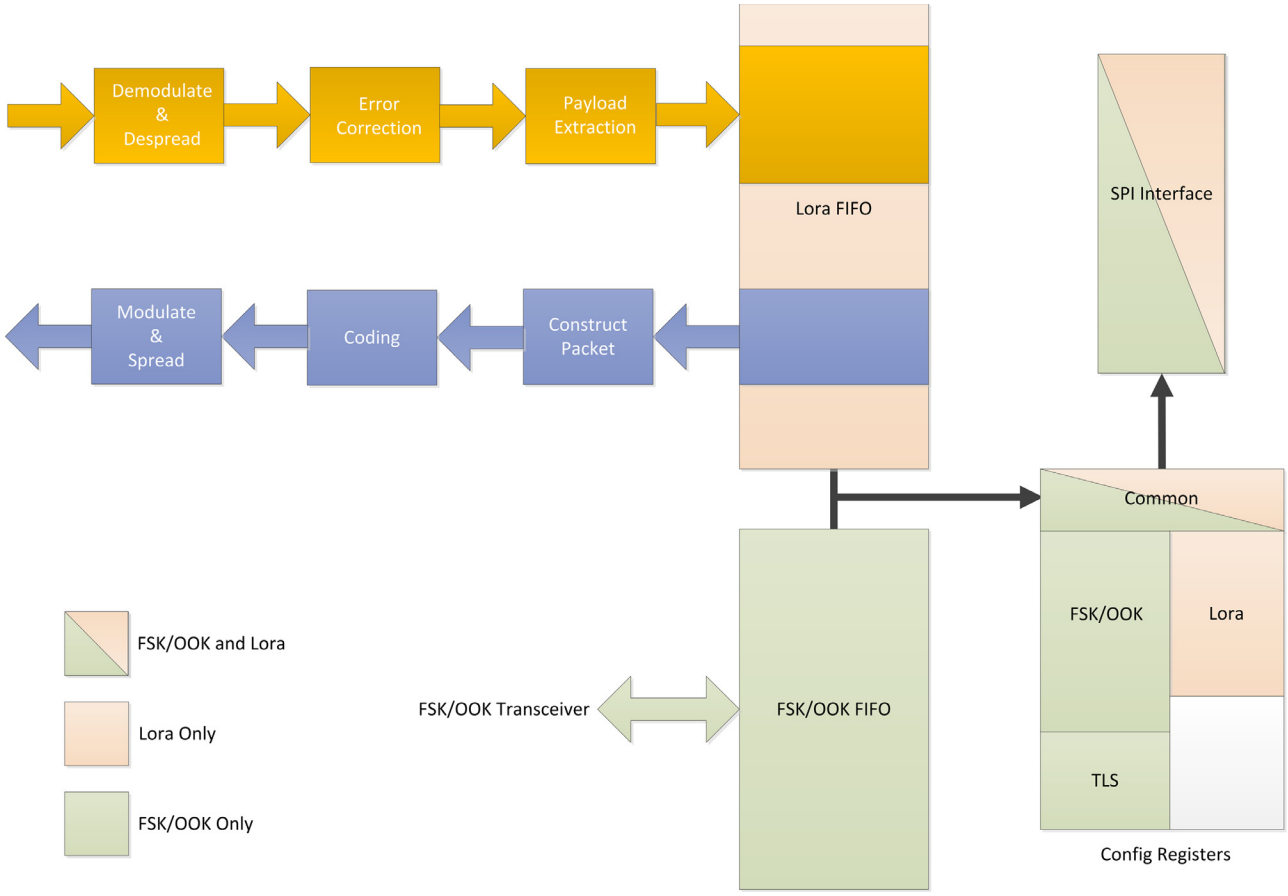


Figure 5. LoRa™ Modem Connectivity

A simplified outline of the transmit and receive processes is also shown above. Here we see that the LoRa™ modem has an independent dual port data buffer FIFO that is accessed through an SPI interface common to all modes. Upon selection of LoRa™ mode, the configuration register mapping of the SX1272/73 changes. For full details of this change please consult the register description of Section 6.

So that it is possible to optimise the LoRa™ modulation for a given application, access is given to the designer to three critical design parameters. Each one permitting a trade off between link budget, immunity to interference, spectral occupancy and nominal data rate. These parameters are spreading factor, modulation bandwidth and error coding rate.

4.1.1.2. Spreading Factor

The spread spectrum LoRa™ modulation is performed by representing each bit of payload information by multiple chips of information. The rate at which the spread information is sent is referred to as the symbol rate (*Rs*), the ratio between the nominal symbol rate and chip rate is the spreading factor and represents the number of symbols sent per bit of information. The range of values accessible with the LoRa™ modem are shown in the following table.

Table 13 Range of Spreading Factors

SpreadingFactor (RegModemConfig2)	Spreading Factor (Chips / symbol)	LoRa Demodulator SNR
6	64	-5 dB
7	128	-7.5 dB
8	256	-10 dB
9	512	-12.5 dB
10	1024	-15 dB
11	2048	-17.5 dB
12	4096	-20 dB

Note that the spreading factor, *SpreadingFactor*, must be known in advance on both transmit and receive sides of the link as different spreading factors are orthogonal to each other. Note also the resulting signal to noise ratio (SNR) required at the receiver input. It is the capability to receive signals with negative SNR that increases the sensitivity, so link budget and range, of the LoRa receiver.

Spreading Factor 6

SF = 6 Is a special use case for the highest data rate transmission possible with the LoRa modem. To this end several settings must be activated in the SX1272/73 registers when it is in use. These settings are only valid for SF6 and should be set back to their default values for other spreading factors:

- Set *SpreadingFactor* = 6 in *RegModemConfig2*
- The header must be set to Implicit mode.
- Set the bit field DetectionOptimize of register *RegLoRaDetectOptimize* to value "0b101".
- Write 0x0C in the register *RegDetectionThreshold*.

4.1.1.3. Coding Rate

To further improve the robustness of the link the LoRa™ modem employs cyclic error coding to perform forward error detection and correction. Such error coding incurs a transmission overhead - the resultant additional data overhead per transmission is shown in the table below.

Table 14 Cyclic Coding Overhead

CodingRate (RegModemConfig1)	Cyclic Coding Rate	Overhead Ratio
1	4/5	1.25
2	4/6	1.5
3	4/7	1.75
4	4/8	2

Forward error correction is particularly efficient in improving the reliability of the link in the presence of interference. So that the coding rate (and so robustness to interference) can be changed in response to channel conditions - the coding rate can optionally be included in the packet header for use by the receiver. Please consult Section 4.1.1.6 for more information on the LoRa™ packet and header.

4.1.1.4. Signal Bandwidth

An increase in signal bandwidth permits the use of a higher effective data rate, thus reducing transmission time at the expense of reduced sensitivity improvement. There are of course regulatory constraints in most countries on the permissible occupied bandwidth. Contrary to the FSK modem, which is described in terms of the single sideband bandwidth, the LoRa™ modem bandwidth refers to the double sideband bandwidth (or total channel bandwidth). The range of bandwidths relevant to most regulatory situations is given in the LoRa™ modem specifications table (see Section 2.5.5).

Bandwidth (kHz)	Spreading Factor	Coding rate	Nominal Rb (bps)	Sensitivity (dBm)
125	12	4/5	293	-136
250	12	4/5	586	-133
500	12	4/5	1172	-130

4.1.1.5. LoRa™ Transmission Parameter Relationship

With a knowledge of the key parameters that can be controlled by the user we define the LoRa™ symbol rate as:

$$Rs = \frac{BW}{2^{SF}}$$

where BW is the programmed bandwidth and SF is the spreading factor. The transmitted signal is a constant envelope signal. Equivalently, one chip is sent per second per Hz of bandwidth.

4.1.1.6. *LoRa™ Packet Structure*

The LoRa™ modem employs two types of packet format, explicit and implicit. The explicit packet includes a short header that contains information about the number of bytes, coding rate and whether a CRC is used in the packet. The packet format is shown in the following figure.

The LoRa™ packet comprises three elements:

- A preamble.
- An optional header.
- The data payload.

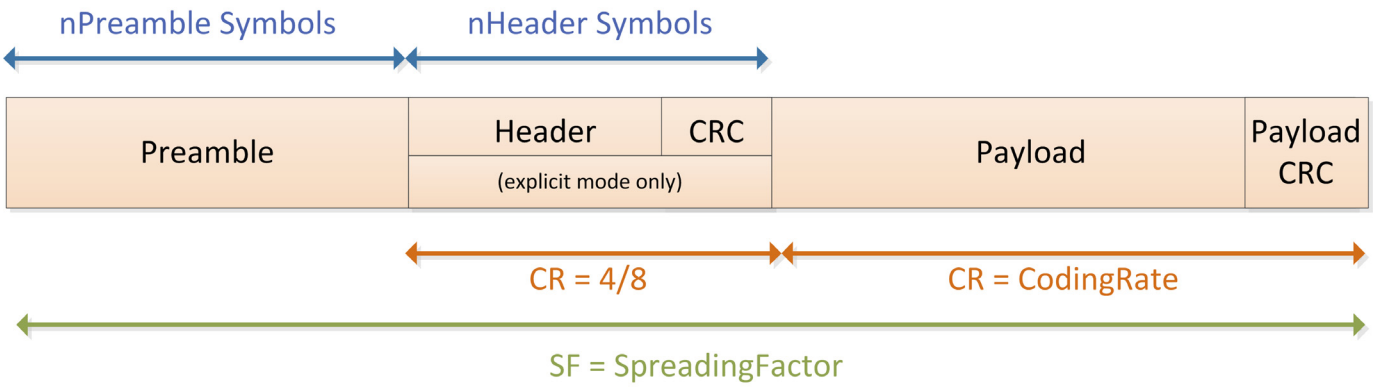


Figure 6. LoRa™ *Packet Structure*

Preamble

The preamble is used to synchronize receiver with the incoming data flow. By default the packet is configured with a 12 symbol long sequence. This is a programmable variable so the preamble length may be extended, for example in the interest of reducing to receiver duty cycle in receive intensive applications. The transmitted preamble length may be changed by setting the registers *RegPreambleMsb* and *RegPreambleLsb* from 6 to 65535, yielding total preamble lengths of 6 + 4 to 65535 + 4 symbols, once the fixed overhead of the preamble data is considered. This permits the transmission of near arbitrarily long preamble sequences.

The receiver undertakes a preamble detection process that periodically restarts. For this reason the preamble length should be configured identical to the transmitter preamble length. Where the preamble length is not known, or can vary, the maximum preamble length should be programmed on the receiver side.

Header

Depending upon the chosen mode of operation two types of header are available. The header type is selected by the *ImplictHeaderModeOn* bit found within the *RegModemConfig1* register.

Explicit Header Mode

This is the default mode of operation. Here the header provides information on the payload, namely:

- The payload length in bytes.
- The forward error correction code rate
- The presence of an optional 16-bits CRC for the payload.

The header is transmitted with maximum error correction code (4/8). It also has its own CRC to allow the receiver to discard invalid headers.

In Explicit Header Mode the presence of the payload CRC selected on the transmit side through the use of the bit RxPayloadCrcOn locatind in the register *RegModemConfig1*.

The corresponding bit (RxPayloadCrcOn) is hence unused on the receive side. Instead, upon reception of the payload, may consult the bit CrcOnPayload in the register *RegHopChannel.* If the bit CrcOnPayload is at '1' then the user should then check the Irq Flag PayloadCrcError to ensure that the CRC is valid.

If the bit CrcOnPayload is at '0', it means there was no CRC on the payload and thus the IRQ Flag PayloadCrcError will not be trigged in the event of payload errors.

Explicit Header	Transmitter	Receiver	CRC Status
Value of the bit RxPayloadCrcOn	0	0	CRC is not checked
	0	1	CRC is not checked
	1	0	CRC is checked
	1	1	CRC is checked

Implicit Header Mode

In certain scenarios, where the payload, coding rate and CRC presence are fixed or known in advance, it may be advantageous to reduce transmission time by invoking implicit header mode. In this mode the header is removed from the packet. In this case the payload length, error coding rate and presence of the payload CRC must be manually configured on both sides of the radio link.

Note that with SF = 6 selected implicit header mode is the only mode of operation possible.

To avail of the payload CRC in Implicit Header Mode, it is necessary to set the bit RxPayloadCrcOn in the register *RegModemConfig1* on both sides (TX and RX).

Implicit Header	Transmitter	Receiver	CRC Status
Value of the bit RxPayloadCrcOn	0	0	CRC is not checked
	0	1	CRC is always wrong
	1	0	CRC is not checked
	1	1	CRC is checked

Low Data Rate Optimization

Given the potentially long duration of the packet at high spreading factors the option is given to improve the robustness of the transmission to variations in frequency over the duration of the packet transmission and reception. The bit *LowDataRateOptimize* increases the robustness of the LoRa link at these low effective data rates, its use is mandated with spreading factors of 11 and 12 at 125 kHz bandwidth.

Payload

The packet payload is a variable-length field that contains the actual data coded at the error rate either as specified in the header in explicit mode or in the register settings in implicit mode. An optional CRC may be appended. For more information on the payload and how it is loaded from the data buffer FIFO please see Section 4.1.2.3.

4.1.1.7. Time on air

For a given combination of spreading factor (SF), coding rate (CR) and signal bandwidth (BW) the total on-the-air transmission time of a LoRa™ packet can be calculated as follows. From the definition of the symbol rate it is convenient to define the symbol period:

$$Ts = \frac{1}{Rs}$$

The LoRa packet duration is the sum of the duration of the preamble and the transmitted packet. The preamble length is calculated as follows:

$$T_{preamble} = (n_{preamble} + 4.25)T_{sym}$$

where $n_{preamble}$ is the programmed preamble length, taken from the registers *RegPreambleMsb* and *RegPreambleLsb*. The payload duration depends upon the header mode that is enabled. The following formula gives the number of payload symbols.

$$n_{payload} = 8 + max\left(ceil\left[\frac{(8PL - 4SF + 28 + 16CRC - 20IH)}{4(SF - 2DE)}\right](CR + 4), 0\right)$$

Where PL is the number of bytes of payload, SF is the spreading factor, IH = 1 when implicit header mode is enabled and IH = 0 when explicit header mode is used. DE set to 1 indicates the use of the low data rate optimization, 0 when disabled. CRC indicates the presence of the payload CRC = 1 when on 0 when off. CR is the programmed coding rate from 1 to 4.

The *ceil* function indicates that the portion of the equation in square brackets should be rounded up to the next integer value. The *max* function compares the evaluated *ceil* function result and returns 0 or the result - whichever is higher.

$$T_{payload} = n_{payload} \times T_s$$

Addition of the preamble and payload durations gives the total packet time on air.

$$T_{packet} = T_{preamble} + T_{payload}$$

4.1.1.8. Frequency Hopping with LoRa™

Frequency hopping spread spectrum (FHSS) is typically employed when the duration of a single packet could exceed regulatory requirements relating to the maximum permissible channel dwell time. This is most notably the case in US operation where the 902 to 928 MHz ISM band which makes provision for frequency hopping operation. To ease the implementation of FHSS systems the frequency hopping mode of the LoRa™ modem can be enabled by setting *FreqHoppingPeriod* to a non-zero value in register *RegHopPeriod*.

Principle of Operation

The principle behind the FHSS scheme is that a portion of each LoRa™ packet is transmitted on each hopping channel from a look up table of frequencies managed by the host microcontroller. After a predetermined hopping period the transmitter and receiver change to the next channel in a predefined list of hopping frequencies to continue transmission and reception of the next portion of the packet. The time which the transmission will dwell in any given channel is determined by *FreqHoppingPeriod* which is an integer multiple of symbol periods:

$$HoppingPeriod[s] = Ts \times FreqHoppingPeriod$$

The frequency hopping transmission and reception process starts at channel 0. The preamble and header are transmitted first on channel 0. At the beginning of each transmission the channel counter *FhssPresentChannel* (located in the register *RegHopChannel*) is incremented and the interrupt signal *FhssChangeChannel* is generated. The new frequency must then be programmed within the hopping period to ensure it is taken into account for the next hop, the interrupt *ChangeChannelFhss* is then to be cleared by writing a logical '1'.

FHSS Reception always starts on channel 0. The receiver waits for a valid preamble detection before starting the frequency hopping process as described above. Note that in the eventuality of header CRC corruption, the receiver will automatically request channel 0 and recommence the valid preamble detection process.

Timing of Channel Updates

The interrupt requesting the channel change, *FhssChannelChange,* is generated upon transition to the new frequency. The frequency hopping process is illustrated in the diagram below:

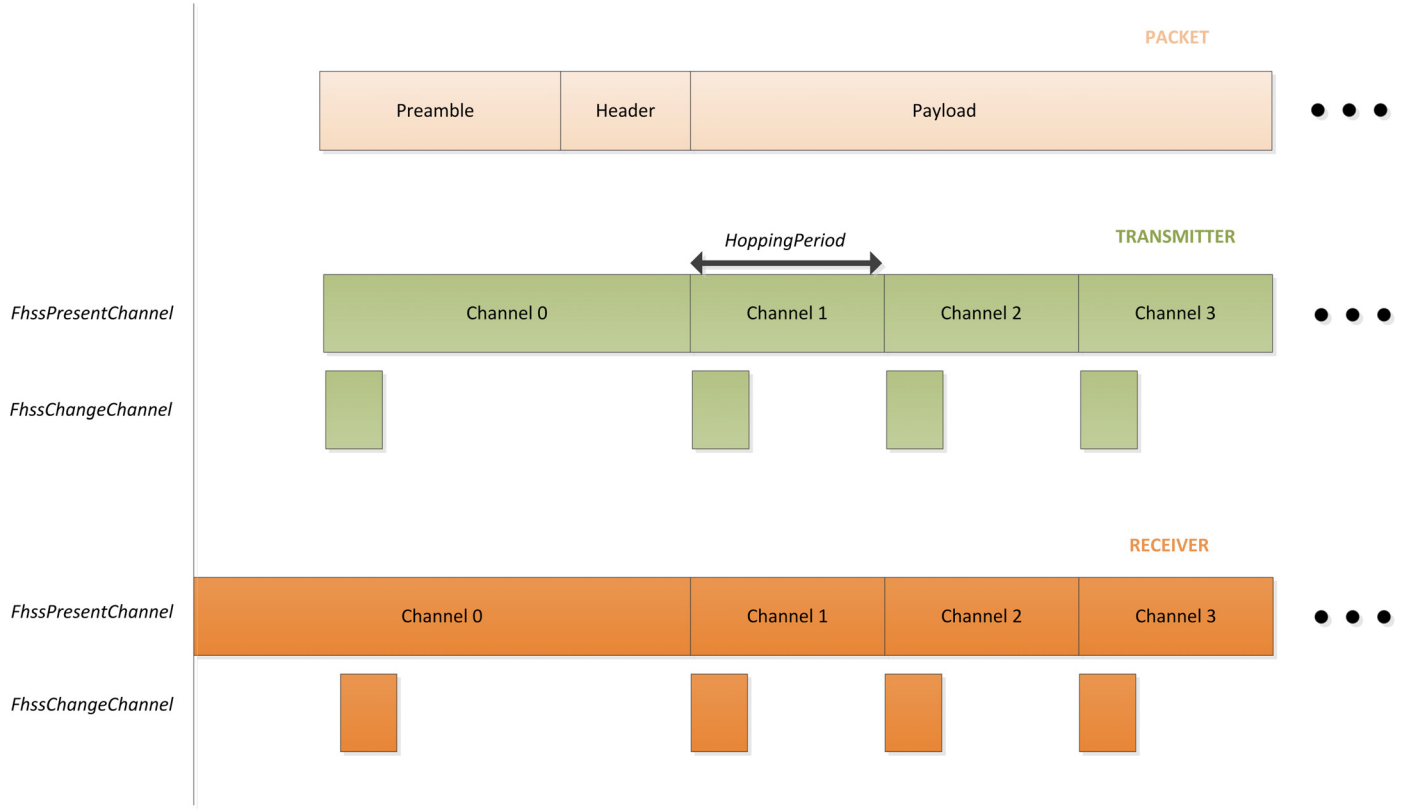


Figure 7. Interrupts generated in the case of successful frequency hopping communication.

4.1.2. LoRa™ Digital Interface

The LoRa™ modem comprises three types of digital interface, static configuration registers, status registers and a FIFO data buffer. All are accessed through the SX1272/73's SPI interface - full details of each type of register are given below. Full listings of the register addresses used for SPI access are given in Section 6.3.

4.1.2.1. LoRa™ Configuration Registers

Configuration registers are accessed through the SPI interface. Registers are readable in all device mode including Sleep. However, they should be **written only in Sleep and Standby modes**. Please note that **the automatic top level sequencer (TLS modes) are not available in** LoRa™ **mode** and **the configuration register mapping changes as shown in Table** 39**.** The content of the LoRa™ configuration registers is retained in FSK/OOK mode. For the functionality of mode registers common to both FSK/OOK and LoRa™ mode, please consult the Analog and RF Front End section of this document (Section 5).

4.1.2.2. Status Registers

Status registers provide status information during receiver operation.

4.1.2.3. LoRa™ Mode FIFO Data Buffer

Overview

The SX1272/73 is equipped with a 256 byte RAM data buffer which is uniquely accessible in LoRa mode. This RAM area, herein referred to as the FIFO Data buffer, is fully customizable by the user and allows access to the received, or to be transmitted, data. All access to the LoRa™ FIFO data buffer is done via the SPI interface. A diagram of the user defined memory mapping of the FIFO data buffer is shown below. These FIFO data buffer can be read in all operating modes except sleep and store data related to the last receive operation performed. It is automatically cleared of old content upon each new transition to receive mode.

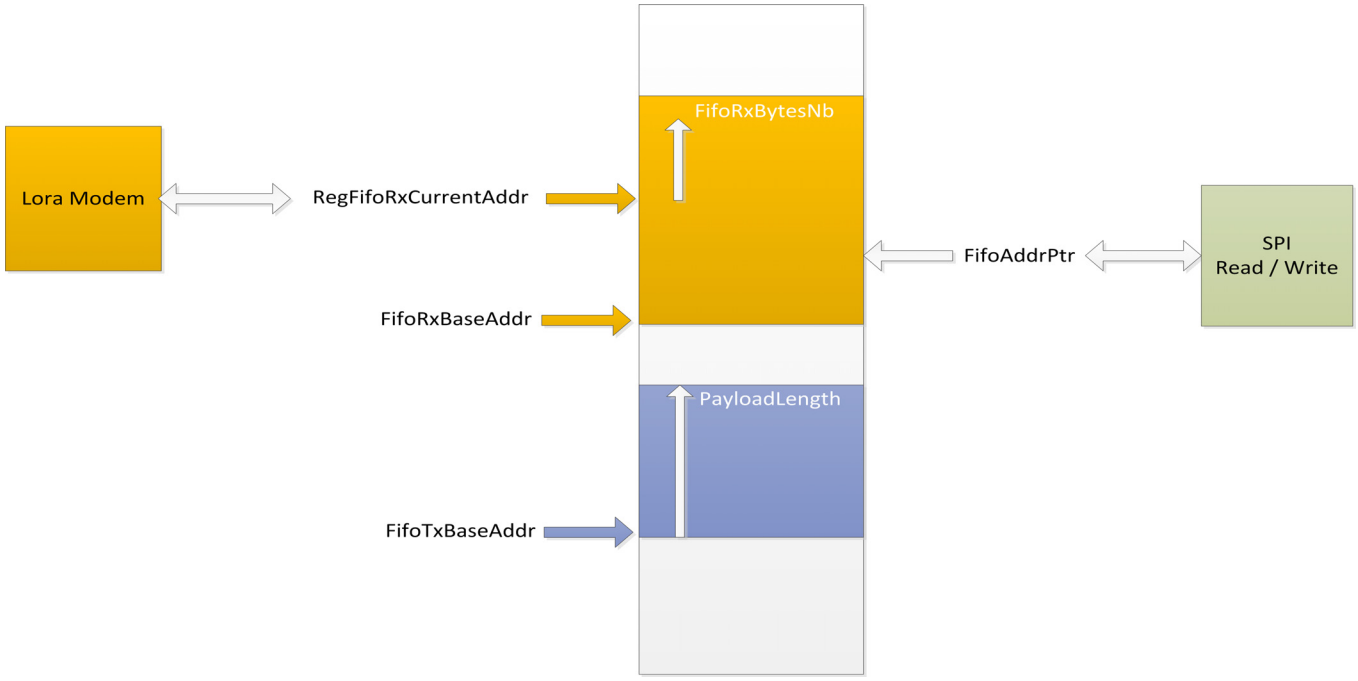


Figure 8. LoRa™ data buffer

Principle of Operation

Thanks to its dual port configuration, it is possible to simultaneously store both transmit and receive information in the FIFO data buffer. The register *RegFifoTxBaseAddr* specifies the point in memory where the transmit information is stored. Similarly, for receiver operation, the register *RegFifoRxBaseAddr* indicates the point in the data buffer where information will be written to in event of a receive operation.

By default, the device is configured at power up so that half of the available memory is dedicated to Rx (*RegFifoRxBaseAddr* initialized at address 0x00) and the other half is dedicated for Tx (*RegFifoTxBaseAddr* initialized at address 0x80).

However, due to the contiguous nature of the FIFO data buffer, the base addresses for Tx and Rx are fully configurable across the 256 byte memory area. Each pointer can be set independently anywhere within the FIFO. To exploit the maximum FIFO data buffer size in transmit or receive mode, the whole FIFO data buffer can be used in each mode by setting the base addresses *RegFifoTxBaseAddr* and *RegFifoRxBaseAddr* at the bottom of the memory (0x00).

The FIFO data buffer is cleared when the device is put in SLEEP mode, consequently no access to the FIFO data buffer is possible in sleep mode. However, the data in the FIFO data buffer are retained when switching across the other LoRa™ modes of operation, so that a received packet can be retransmitted with minimum data handling on the controller side. The FIFO data buffer is not self-clearing (unless if the device is put in sleep mode) and the data will only be “erased” when a new set of data is written into the occupied memory location.

The FIFO data buffer location to be read from, or written to, via the SPI interface is defined by the address pointer *RegFifoAddrPtr*. Before any read or write operation it is hence necessary to initialize this pointer to the corresponding base value. Upon reading or writing to the FIFO data buffer (*RegFifo*) the address pointer will then increment automatically.

The register *RegRxNbBytes* defines the size of the memory location to be written in the event of a successful receive operation. The register *RegPayloadLength* indicates the size of the memory location to be transmitted. In implicit header mode, the register *RegRxNbBytes* is not used as the number of payload bytes is known. Otherwise, in explicit header mode, the initial size of the receive buffer is set to the packet length in the received header. The register *RegFifoRxCurrentAddr* indicates the location of the last packet received in the FIFO so that the last packet received can be easily read by pointing the register *RegFifoAddrPtr* to this register.

It is important to note that all the received data will be written to the FIFO data buffer even if the CRC is invalid, permitting user defined post processing of corrupted data. It is also important to note that when receiving, if the packet size exceeds the buffer memory allocated for the Rx, it will overwrite the transmit portion of the data buffer.

4.1.2.4. Interrupts in LoRa Mode

Two registers are used to control the interrupt signals (IRQ) available in LoRa mode: the register *RegIrqFlags* contains the state of the interrupts themselves and the register *RegIrqFlagsMask* which can be used to mask the interrupts.
The interrupt mask *RegIrqFlagsMask,* allows the user to mask individual interrupts, in this case setting a bit to ‘1’ will mask - so deactivate the interrupt at the same position in the *RegIrqFlags* register. By default all the interrupts are unmasked, so available.
In the register *RegIrqFlags*, a ‘1’ indicates a given IRQ has been trigged and then the IRQ must be clear by writing a ‘1’.

4.1.3. Operation of the LoRa™ Modem

4.1.3.1. Operating Mode Control

The operating modes of the LoRa™ modem are accessed by enabling LoRa™ mode (setting the *LongRangeMode* bit of *RegOpMode*). Depending upon the operating mode selected the range of functionality and register access is given by the following table:

Table 15 LoRa™ Operating Mode Functionality

Operating Mode	Description
SLEEP	Low-power mode. In this mode only SPI and configuration registers are accessible. Lora FIFO is not accessible. Note that this is the only mode permissible to switch between FSK/OOK mode and LoRa mode.
STANDBY	Both crystal oscillator and LoRa baseband blocks are turned on. RF front-end and PLLs are disabled
FSTX	This is a frequency synthesis mode for transmission. The PLL selected for transmission is locked and active at the transmit frequency. The RF front-end is off.
FSRX	This is a frequency synthesis mode for reception. The PLL selected for reception is locked and active at the receive frequency. The RF front-end is off.
TX	When activated the SX1272/73 powers all remaining blocks required for transmit, ramps the PA, transmits the packet and returns to Standby mode.
RXCONTINUOUS	When activated the SX1272/73 powers all remaining blocks required for reception, processing all received data until a new user request is made to change operating mode.
RXSINGLE	When activated the SX1272/73 powers all remaining blocks required for reception, remains in this state until a valid packet has been received and then returns to Standby mode.
CAD	When in CAD mode, the device will check a given channel to detect LoRa preamble signal

It is possible to access any mode from any other mode by changing the value in the *RegOpMode* register.

4.1.4. Frequency Settings

Recalling that the frequency step is given by:

$$F_{STEP} = \frac{F_{XOSC}}{2^{19}}$$

In order to set LO frequency values following registers are available.

Frf is a 24-bit register which defines carrier frequency. The carrier frequency relates to the register contents by following formula:

$$F_{RF} = F_{STEP} \times Frf(23,0)$$

4.1.5. Frequency Error Indication

The SX1272 derives its RF centre frequency from a crystal reference oscillator which has a finite frequency precision. Errors in reference frequency will manifest themselves as errors of the same proportion from the RF centre frequency.

In LoRa receive mode the SX1272 is capable of measuring the frequency offset between the receiver centre frequency and that of an incoming LoRa signal. The image below shows the estimated frequency offset and corresponding PER for various coding rates as a function of frequency offset. Here we can see that the modem is intolerant of frequency offsets in the region of +/- 20% of the bandwidth and will accurately report the error over this same range.

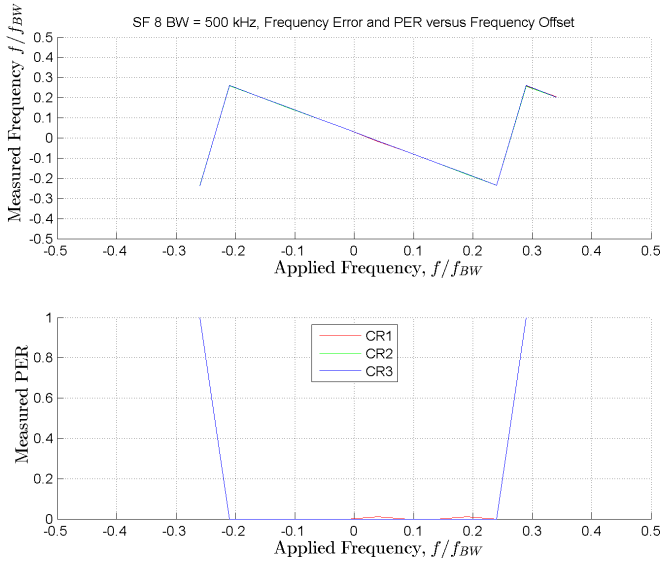


Figure 9. Applied versus measured frequency offset and influence on PER.

The error is read by reading the three *RegFei* registers. The contents of which are a signed 20 bit two's compliment word, *FreqError*. The frequency error is determined from the register contents by:

$$F_{Error} = \frac{FreqError \times 2^{24}}{F_{xtal}}$$

Where Fxtal is the crystal frequency.

Please note that the measured FEI should not be applied to the RF centre frequency to perform AFC. The FEI measurement is provided for information only.

4.1.6. LoRa™ Modem State Machine Sequences

The sequence for transmission and reception of data to and from the LoRa™ modem, together with flow charts of typical sequences of operation, are detailed below.

Data Transmission Sequence

In transmit mode power consumption is optimized by enabling RF, PLL and PA blocks only when packet data needs to be transmitted. Figure 10 shows a typical LoRa™ transmit sequence.

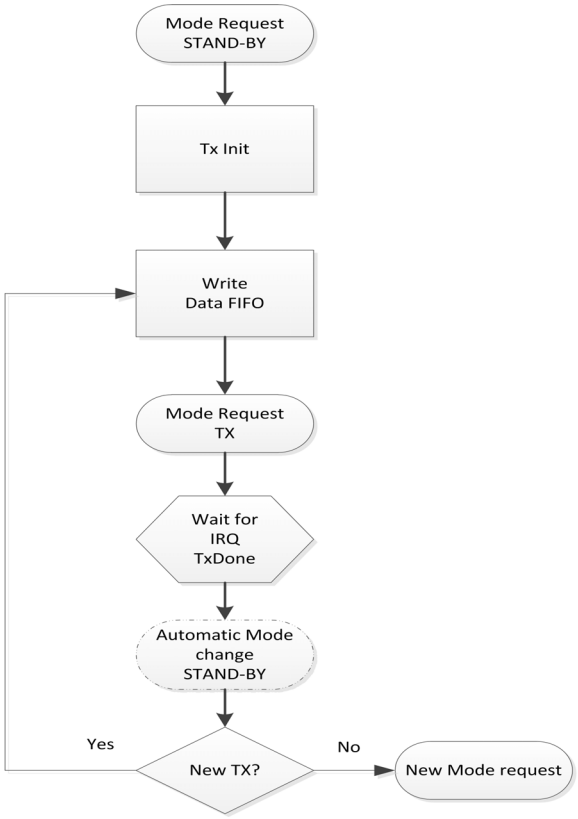


Figure 10. LoRa™ modulation transmission sequence.

- Static configuration registers can only be accessed in Sleep mode, Standby mode or FSTX mode.
- The LoRa™ FIFO should only be filled in Standby mode and cannot be filled in Sleep mode.
- Data transmission is initiated by sending TX mode request.
- Upon completion the *TxDone* interrupt is issued and the radio returns to Standby mode.
- Following transmission the radio can be manually placed in Sleep mode or the FIFO refilled for a subsequent Tx operation.

LoRa™ Transmit Data FIFO Filling

In order to write packet data into FIFO user should:

1 Set *FifoAddrPtr* to *FifoTxBaseAddrs*.

2 Write *PayloadLength* bytes to the FIFO (*RegFifo*)

Data Reception Sequence

Figure 11 shows typical LoRa™ receive sequences for both single and continuous receiver modes of operation.

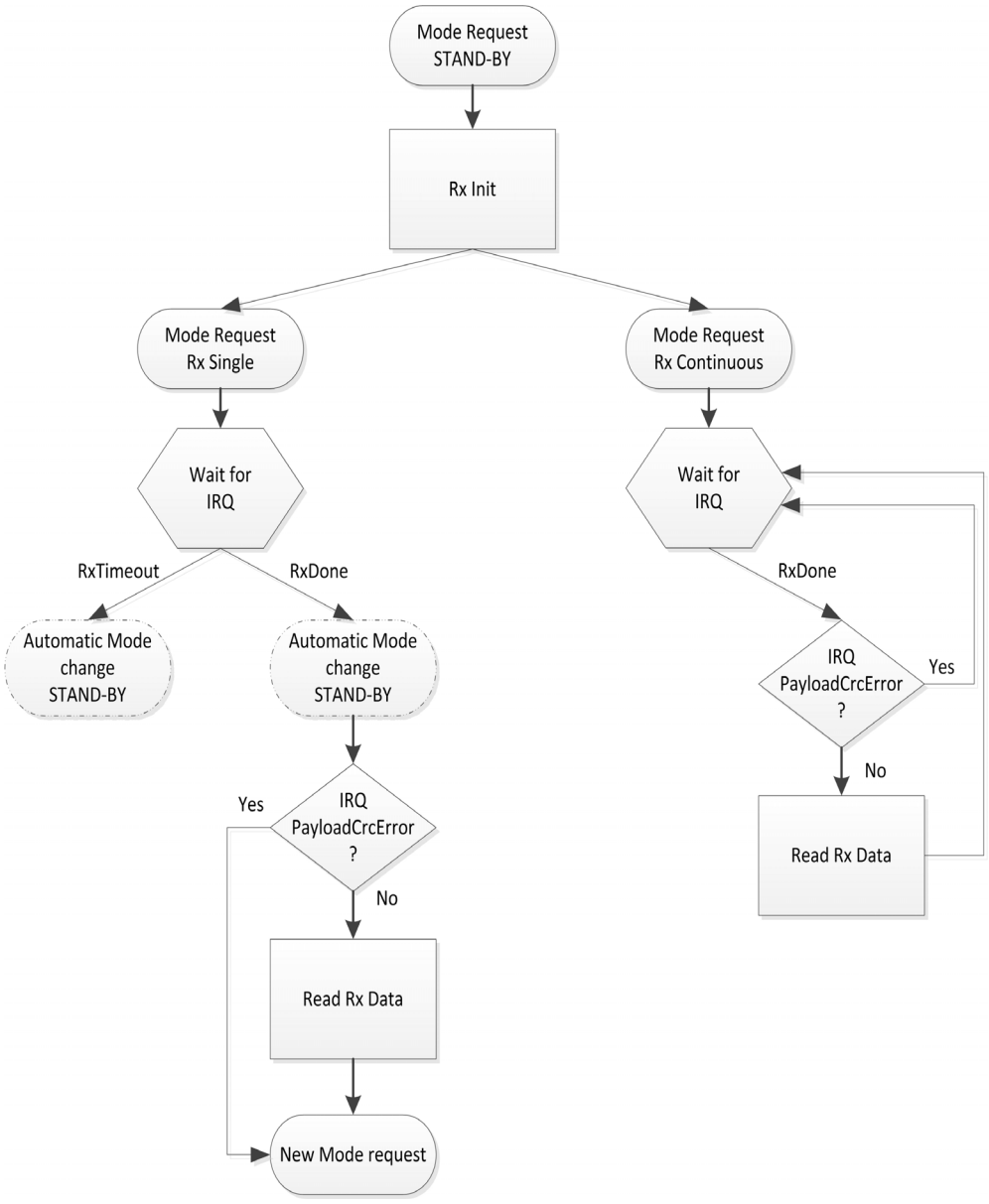


Figure 11. LoRa™ receive sequence.

The LoRa™ modem can work in two distinct reception modes:

1. Single receive mode
2. Continuous receive mode

Single Reception Operating Mode

In this mode, the modem searches for a preamble during a given period of time. If a preamble hasn't been found at the end of the time window, the chip generates the *RxTimeout* interrupt and goes back to Standby mode. The length of the reception window (in symbols) is defined by the *RegSymbTimeout* register and should be in the range of 4 (minimum time for the modem to acquire lock on a preamble) up to 1023 symbols.

At the end of the payload, the *RxDone* interrupt is generated together with the interrupt *PayloadCrcError* if the payload CRC is not valid. However, even when the CRC is not valid, the data are written in the FIFO data buffer for post processing. Following the *RxDone* interrupt the radio goes to Standby mode.

The modem will also automatically return in Standby mode when the interrupts *RxDone* is generated. Therefore, this mode should only be used when the time window of arrival of the packet is known. In other cases, the RX continuous mode should be used.

In Rx single mode low-power is achieved by turning off PLL and RF blocks as soon as a packet has been received. The flow is as follows:

1 *Set FifoAddrPtr* to *FifoRxBaseAddr*.

2 Static configuration register device can be written in either Sleep mode, Standby mode or FSRX mode.

3 A single packet receive operation is initiated by selecting the operating mode RXSINGLE.

4 The receiver will then await the reception of a valid preamble. Once received, the gain of the receive chain is set. Following the ensuing reception of a valid header, indicated by the *ValidHeader* interrupt in explicit mode. The packet reception process commences. Once the reception process is complete the *RxDone* interrupt is set. The radio then returns automatically to Standby mode to reduce power consumption.

5 The receiver status register *PayloadCrcError* should be checked for packet payload integrity.

6 If a valid packet payload has been received then the FIFO should be read (See Payload Data Extraction below). Should a subsequent single packet reception need to be triggered, then the RXSINGLE operating mode must be re-selected to launch the receive process again - taking care to reset the SPI pointer (*FifoAddrPtr*) to the base location in memory (*FifoRxBaseAddr*).

Continuous Reception Operating Mode

In continuous receive mode, the modem scans the channel continuously for a preamble. Each time a preamble is detected the modem tracks it until the packet is received and then carries on waiting for the next preamble.

If the preamble length exceeds the anticipated value set by the registers *RegPreambleMsb* and *RegPreambleLsb* (measured in symbol periods) the preamble will be dropped and the search for a preamble restarted. However, this scenario will not be flagged by any interrupt. In continuous RX mode, opposite to the single RX mode, the RxTimeout interrupt will never occur and the device will never go in Standby mode automatically.

It is also important to note that the demodulated bytes are written in the data buffer memory in the order received. Meaning, the first byte of a new packet is written just after the last byte of the preceding packet. The RX modem address pointer is never reset as long as this mode is enabled. It is therefore necessary for the companion microcontroller controller to handle the address pointer to make sure the FIFO data buffer is never full.

In continuous mode the received packet processing sequence is given below.

1 Whilst in Sleep or Standby mode select RXCONT mode.

2 Upon reception of a valid header CRC the *RxDone i*nterrupt is set. The radio remains in RXCONT mode waiting for the next RX LoRa™ packet.

3 The *PayloadCrcError* flag should be checked for packet integrity.

4 If packet has been correctly received the FIFO data buffer can be read (see below).

5 The reception process (steps 2 - 4) can be repeated or receiver operating mode exited as desired.

In continuous mode status information are available only for the last packet received, i.e. the corresponding registers should be read before the next *RxDone* arrives.

Rx Single and Rx Continuous Use Cases

The LoRa single reception mode is used mainly in battery operated systems or in systems where the companion microcontroller has a limited availability of timers. In such systems, the use of the timeout present in Rx Single reception mode allows the end user to limit the amount of time spent in reception (and thus limiting the power consumption) while not using any of the companion MCU timers (the MCU can then be in sleep mode while the radio is in the reception mode). The RxTimeout interrupt generated at the end of the reception period is then used to wake-up the companion MCU. One of the advantages of the RxSingle mode is that the interrupt RxTimeout will not be triggered if the device is currently receiving data, thus giving the priority to the reception of the data over the timeout. However, if during the reception, the device loses track of the data due to external perturbation, the device will drop the reception, flag the interrupt RxTimeout and go in Standby mode to decrease the power consumption of the system.

On the other hand, The LoRa continuous reception mode is used in systems which do not have power restrictions or on system where the use of a companion MCU timer is preferred over the radio embedded timeout system. In RxContinuous mode, the radio will track any LoRa signal present in the air and carry on the reception of packets until the companion MCU sets the radio into another mode of operation. Upon reception the interrupt RxDone will be trigged but the device will stay in Rx Mode, ready for the reception of the next packet.

Payload Data Extraction from FIFO

In order to retrieve received data from FIFO the user must ensure that *ValidHeader*, *PayloadCrcError, RxDone* and *RxTimeout* interrupts in the status register RegIrqFlags are not asserted to ensure that packet reception has terminated successfully (i.e. no flags should be set).

In case of errors the steps below should be skipped and the packet discarded. In order to retrieve valid received data from the FIFO the user must:

- *RegRxNbBytes* Indicates the number of bytes that have been received thus far.
- *RegFifoAddr*Ptr is a dynamic pointer that indicates precisely where the Lora modem received data has been written up to.
- Set *RegFifoAddr*Ptr to *RegFifoRxCurrentAddr*. This sets the FIFO pointer to the location of the last packet received in the FIFO. The payload can then be extracted by reading the register *RegFifo, RegRxNbBytes* times.

  Alternatively, it is possible to manually point to the location of the last packet received, from the start of the current packet, by setting *RegFifoAddrPtr* to *RegFifoRxByteAddr* minus *RegRxNbBytes*. The payload bytes can then be read from the FIFO by reading the *RegFifo* address *RegRxNbBytes* times.

Packet Filtering based on Preamble Start

The LoRa™ modem does not automatically filter received packets based upon an address. However, the SX1272/73 permits software filtering of the received packets based on the contents of the first few bytes of payload. A brief example is given below for a 4 byte address, however, the address length can be selected by the designer.

The objective of the packet filtering process is to determine the presence, or otherwise, of a valid packet designed for the receiver. If the packet is not for the receiver then the radio returns to sleep mode in order to improve battery life.The software packet filtering process follows the steps below:

- Each time the *RxDone* interrupt is received, latch the RegFifoRxByteAddr[7:0] register content in a variable, this variable will be called start_address. The *RegFifoRxByteAddr[7:0]* register of the SX1272 gives in real time the address of the last byte written in the data buffer + 1 (or the address at which the next byte will be written by the receive LoRa™ modem). So by doing this, we make sure that the variable start_address always contains the start address of the next packet.

- Upon reception of the interrupt ValidHeader, start polling the *RegFifoRxByteAddr[7:0]* register until it begins to increment. The speed at which this register will increment depends on the spreading factor, the error correction code and the modulation bandwidth. (Note that this interrupt is still generated in implicit mode).

- As soon as *RegFifoRxByteAddr[7:0]* >= start address + 4, the first 4 bytes (address) are stored in the FIFO data buffer. These can be read and tested to see if the packet is destined for the radio and either remaining in Rx mode to receive the packet or returning to sleep mode if not.

Receiver Timeout Operation

In LoRa™ Rx Single mode, a receiver timeout functionality is available that permits the receiver to listen for a predetermined period of time before generating an interrupt signal to indicate that no valid packets have been received. The timer is absolute and commences as soon as the radio is placed in single receive mode. The interrupt itself, *RxTimeout*, can be found in the interrupt register *RegIrqFlags*. In Rx Single mode, the device will return to Standby mode as soon as the interrupt occurs. The user must then clear the interrupt or go into Sleep mode before returning into Rx Single mode. The programmed timeout value is expressed as a multiple of the symbol period and is given by:

$$TimeOut = LoraRxTimeout \cdot Ts$$

Channel activity detection

The use of a spread spectrum modulation technique presents challenges in determining whether the channel is already in use by a signal that may be below the noise floor of the receiver. The use of the RSSI in this situation would clearly be impracticable. To this end the channel activity detector is used to detect the presence of other LoRa™ signals. Figure 12 shows the channel activity detection (CAD) process:

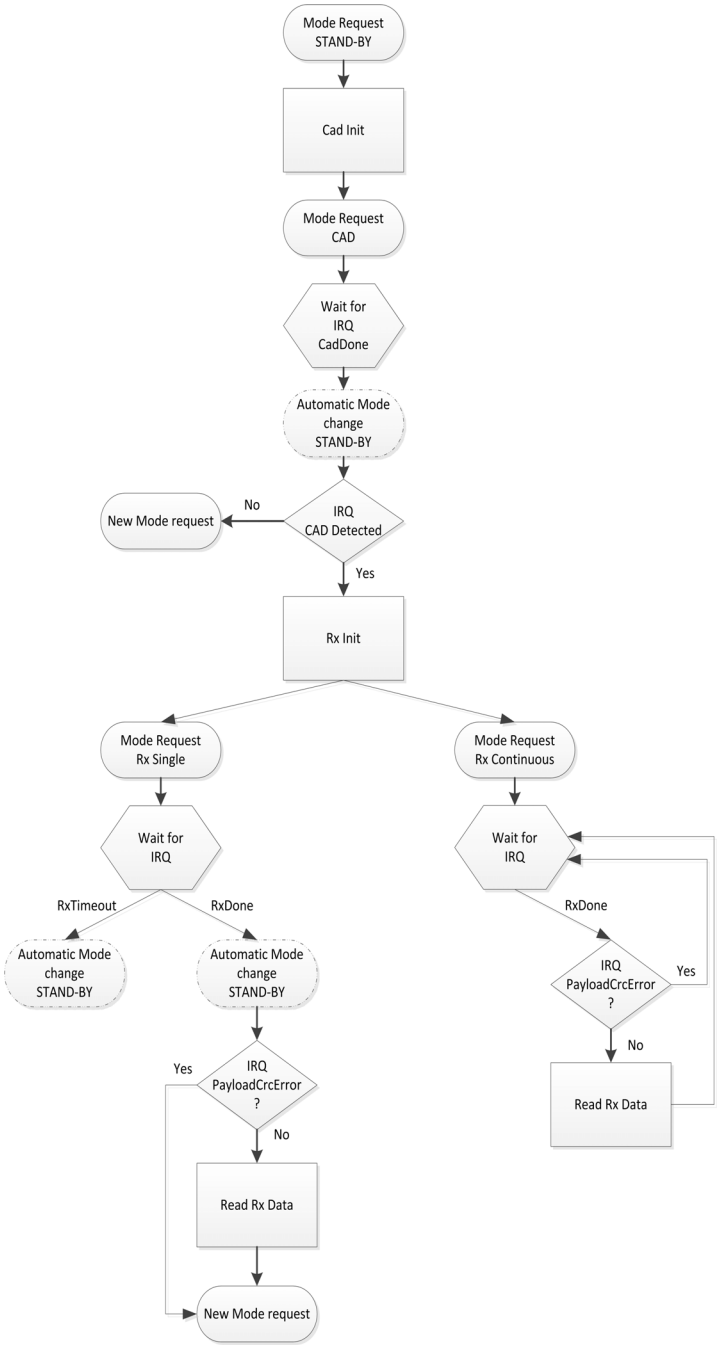


Figure 12. LoRa™ CAD flow

Principle of Operation

The channel activity detection mode is designed to detect a LoRa preamble on the radio channel with the best possible power efficiency. Once in CAD mode, the SX1272/73 will perform a very quick scan of the band to detect a LoRa™ packet preamble.

During a CAD the following operations take place:

- The PLL locks
- The radio receiver captures LoRa™ preamble symbol of data from the channel. The radio current consumption during that phase is approximately 10 mA.
- The radio receiver and the PLL turn off and the modem digital processing starts.
- The modem searches for a correlation between the radio captured samples and the ideal preamble waveform. This correlation process takes a little bit less than a symbol period to perform. The radio current consumption during that phase is greatly reduced.
- Once the calculation is finished the modem generates the *CadDone* interrupt. If the correlation was successful, the CadDetected is generated simultaneously.
- The chip goes back to Standby mode.
- If a preamble was detected, clear the interrupt, then initiate the reception by putting the radio in RX single mode or RX continuous mode.

The time taken for the channel activity detection is dependent upon the LoRa™ modulation settings used. For a given configuration the typical CAD detection time is shown in the graph below, expressed as a multiple of the LoRa™ symbol period. Of this period the radio is in receiver mode for (2^{SF} + 32) / BW seconds. For the remainder of the CAD cycle the radio is in a reduced consumption state.

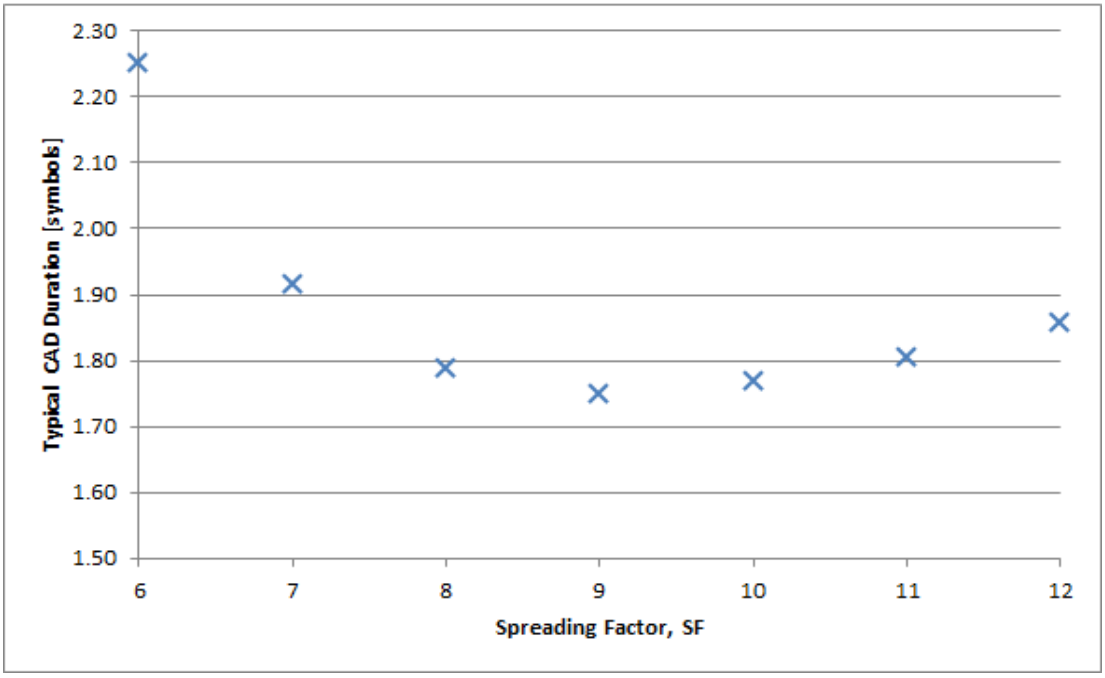


Figure 13. Channel activity detection (CAD) time as a function of spreading factor.

To illustrate this process and the respective consumption in each mode, the CAD process follows the sequence of events outlined below:

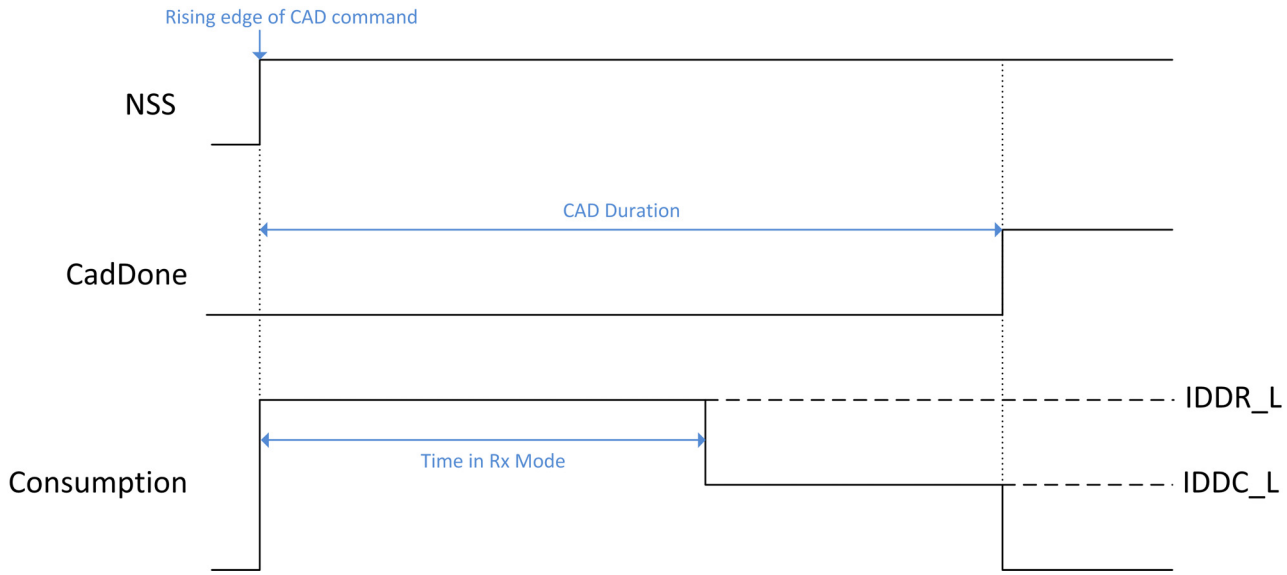


Figure 14. Consumption Profile of the LoRa CAD Process

The receiver is then in full receiver mode for just over half of the activity detection, followed by a reduced consumption processing phase where the consumption varies with the LoRa bandwidth as shown in the table below.

Table 16 LoRa CAD Consumption Figures

Bandwidth (kHz)	Full Rx, IDDR_L (mA)	Processing, IDDC_L (mA)
125	10.8	5.6
250	11.6	6.5
500	13	8

4.1.6.1. Digital IO Pin Mapping

Six of SX1272/73's general purpose IO pins are available used in LoRa™ mode. Their mapping is shown below and depends upon the configuration of registers *RegDioMapping1* and *RegDioMapping2.*

Table 17 DIO Mapping LoRa™ *Mode*

Operating Mode	DIOx Mapping	DIO5	DIO4	DIO3	DIO2	DIO1	DIO0
ALL	00	ModeReady	CadDetected	CadDone	FhssChangeChannel	RxTimeout	RxDone
	01	ClkOut	PllLock	ValidHeader	FhssChangeChannel	FhssChangeChannel	TxDone
	10	ClkOut	PllLock	PayloadCrcError	FhssChangeChannel	CadDetected	CadDone
	11	-	-	-	-	-	-

4.2. FSK/OOK Modem

4.2.1. Bit Rate Setting

The bit rate setting is referenced to the crystal oscillator and provides a precise means of setting the bit rate (or equivalently chip) rate of the radio. In continuous transmit mode (Section 4.2.13) the data stream to be transmitted can be input directly to the modulator via pin 9 (DIO2/DATA) asynchronously, unless Gaussian filtering is used, in which case the DCLK signal on pin 10 (DIO1/DCLK) is used to clock-in the data stream. See section 4.2.2.3 for details of the Gaussian filter.

In Packet mode or in Continuous mode with Gaussian filtering enabled, the Bit Rate (BR) is controlled by bits *BitRate* in *RegBitrateMsb and RegBitrateLsb*

$$BitRate = \frac{FXOSC}{BitRate(15,0) + \frac{BitrateFrac}{16}}$$

*Note: BitrateFrac bits have **no effect** (i.e may be considered equal to 0) **in OOK** modulation mode.*

The quantity *BitrateFrac* is hence designed to allow very high precision (max. 250 ppm programing resolution) for any bitrate in the programmable range. Table 18 below shows a range of standard bitrates and the accuracy to within which they may be attained.

Table 18 Bit Rate Examples

Type	BitRate (15:8)	BitRate (7:0)	(G)FSK (G)MSK	OOK	Actual BR (b/s)
Classical modem baud rates (multiples of 1.2 kbps)	0x68	0x2B	1.2 kbps	1.2 kbps	1200.015
	0x34	0x15	2.4 kbps	2.4 kbps	2400.060
	0x1A	0x0B	4.8 kbps	4.8 kbps	4799.760
	0x0D	0x05	9.6 kbps	9.6 kbps	9600.960
	0x06	0x83	19.2 kbps	19.2 kbps	19196.16
	0x03	0x41	38.4 kbps		38415.36
	0x01	0xA1	76.8 kbps		76738.60
	0x00	0xD0	153.6 kbps		153846.1
Classical modem baud rates (multiples of 0.9 kbps)	0x02	0x2C	57.6 kbps		57553.95
	0x01	0x16	115.2 kbps		115107.9

Type	BitRate (15:8)	BitRate (7:0)	(G)FSK (G)MSK	OOK	Actual BR (b/s)
Round bit rates (multiples of 12.5, 25 and 50 kbps)	0x0A	0x00	12.5 kbps	12.5 kbps	12500.00
	0x05	0x00	25 kbps	25 kbps	25000.00
	0x80	0x00	50 kbps		50000.00
	0x01	0x40	100 kbps		100000.0
	0x00	0xD5	150 kbps		150234.7
	0x00	0xA0	200 kbps		200000.0
	0x00	0x80	250 kbps		250000.0
	0x00	0x6B	300 kbps		299065.4
Watch Xtal frequency	0x03	0xD1	32.768 kbps	32.768 kbps	32753.32

4.2.2. FSK/OOK Transmission

4.2.2.1. FSK Modulation

FSK modulation is performed inside the PLL bandwidth by changing the fractional divider ratio in the feedback loop of the PLL. The high resolution of the sigma-delta modulator allows for very narrow frequency deviation. The frequency deviation F_{DEV} is given by:

$$F_{DEV} = F_{STEP} \times Fdev(13,0)$$

To ensure correct modulation the following limit applies:

$$F_{DEV} + \frac{BR}{2} \leq (250)kHz$$

Note *No constraint applies to the modulation index of the transmitter, but the frequency deviation must be set between 600 Hz and 200 kHz.*

4.2.2.2. OOK Modulation

OOK modulation is applied by switching on and off the power amplifier. Digital control and ramping are available to improve the transient power response of the OOK transmitter.

4.2.2.3. Modulation Shaping

Modulation shaping can be applied in both OOK and FSK modulation modes to improve the narrowband response of the transmitter. Both shaping features are controlled with *PaRamp* bits in *RegPaRamp*.

- In FSK mode, a Gaussian filter with BT = 0.5 or 1 can be used to filter the modulation stream, at the input of the sigma-delta modulator. If the Gaussian filter is enabled when the SX1272/73 is in Continuous mode, DCLK signal on pin 10 (DIO1/DCLK) will trigger an interrupt on the uC each time a new bit has to be transmitted. Please refer to section 4.2.13.2 for details.
- When OOK modulation is used the PA bias voltages are ramped up and down smoothly when the PA is turned on and off to reduce spectral splatter.

Note The transmitter must be restarted if the ModulationShaping setting is changed in order to recalibrate the built-in filter.

4.2.3. FSK/OOK Reception

4.2.3.1. FSK Demodulator

The FSK demodulator of the SX1272/73 is designed to demodulate FSK, GFSK, MSK and GMSK modulated signals. It is most efficient when the modulation index (β) of the signal is greater than 0.5 and below 10:

$$0.5 \leq \beta = \frac{2 \times F_{DEV}}{BR} \leq 10$$

The output of the FSK demodulator can be fed to the Bit Synchronizer to provide the companion processor with a synchronous data stream in Continuous mode.

4.2.3.2. OOK Demodulator

The OOK demodulator performs a comparison of the RSSI output and a threshold value. Three different threshold modes are available, configured through bits *OokThreshType* in *RegOokPeak*.

The recommended mode of operation is the “Peak” threshold mode, illustrated in Figure 15:

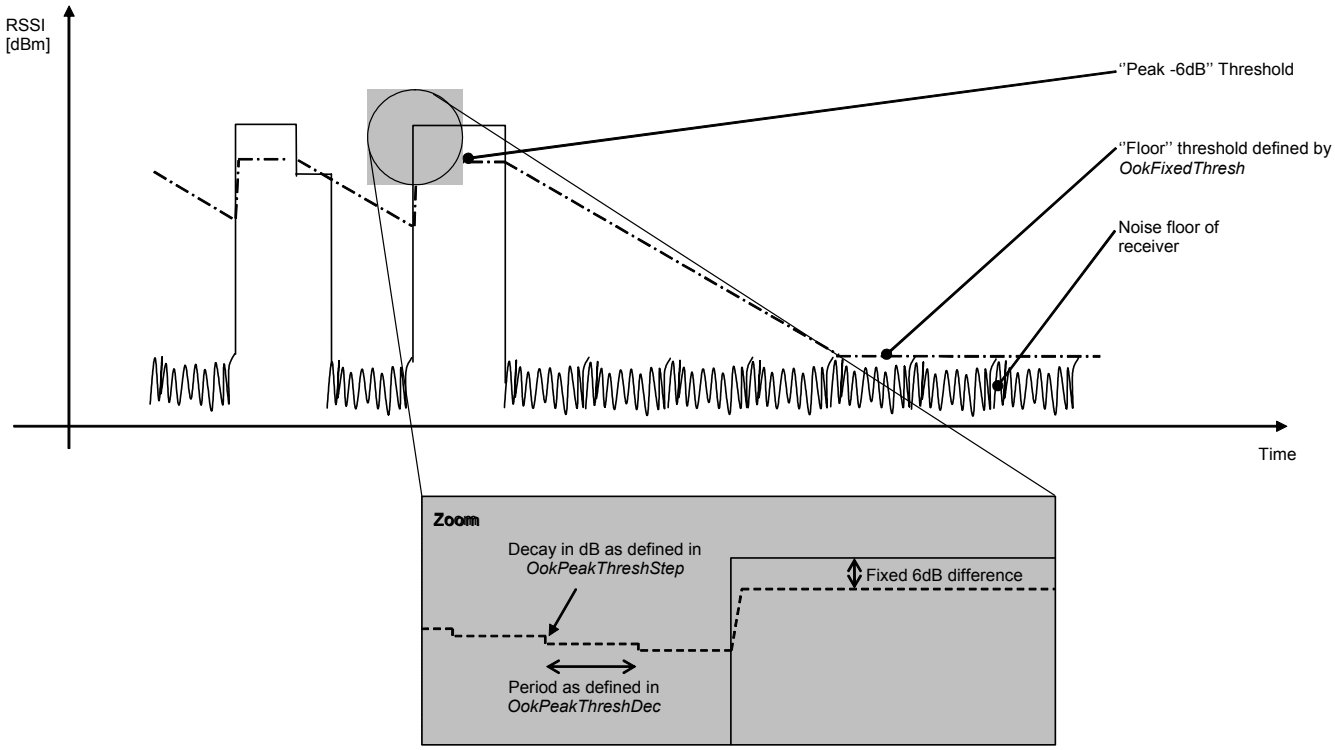


Figure 15. OOK Peak Demodulator Description

In peak threshold mode the comparison threshold level is the peak value of the RSSI reduced by 6 dB. In the absence of an input signal, or during the reception of a logical ‘0’, the acquired peak value is decremented by one *OokPeakThreshStep* every *OokPeakThreshDec* period.

When the RSSI output is null for a long time (for instance after a long string of “0” received or if no transmitter is present) the peak threshold level will continue falling until it reaches the “Floor Threshold” programmed in *OokFixedThresh*.

The default settings of the OOK demodulator lead to the performance stated in the electrical specification. However, in applications in which sudden received signal power reduction is possible, the three parameters should be optimized accordingly.

Optimizing the Floor Threshold

OokFixedThresh determines the sensitivity of the OOK receiver, as it sets the comparison threshold for weak input signals (i.e. those close to the noise floor). Significant sensitivity improvements can be generated if configured correctly.

Note that the noise floor of the receiver at the demodulator input depends on:

- The noise figure of the receiver.
- The gain of the receive chain from antenna to base band.
- The matching - including SAW filter if any.
- The bandwidth of the channel filters.

It is therefore important to note that the setting of *OokFixedThresh* will be application dependant. The following procedure is recommended to optimize *OokFixedThresh*.

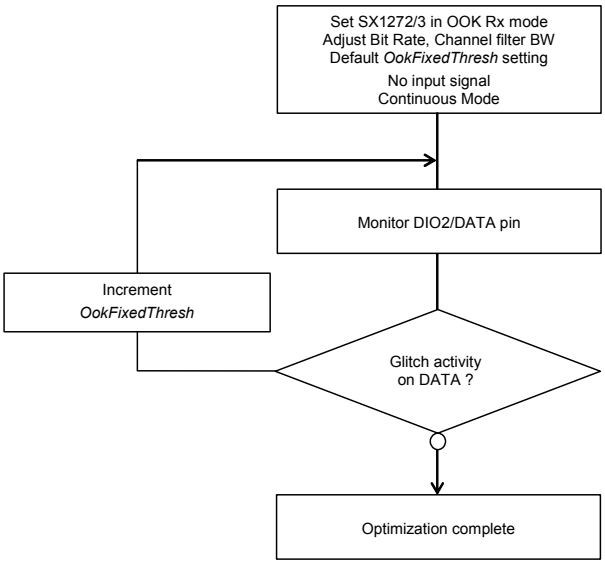


Figure 16. Floor Threshold Optimization

The new floor threshold value found during this test should be used for OOK reception with those receiver settings.

Optimizing OOK Demodulator for Fast Fading Signals

A sudden drop in signal strength can cause the bit error rate to increase. For applications where the expected signal drop can be estimated, the following OOK demodulator parameters *OokPeakThreshStep* and *OokPeakThreshDec* can be optimized as described below for a given number of threshold decrements per bit. Refer to *RegOokPeak* to access those settings.

Alternative OOK Demodulator Threshold Modes

In addition to the Peak OOK threshold mode, the user can alternatively select two other types of threshold detectors:

- Fixed Threshold: The value is selected through *OokFixedThresh*
- Average Threshold: Data supplied by the RSSI block is averaged (this operation mode should only be used with DC-free encoded data).

4.2.3.3. Bit Synchronizer

The bit synchronizer provides a clean and synchronized digital output based upon timing recovery information gleaned from the received data edge transitions. Its output is made available on pin DIO1/DCLK in Continuous mode and can be disabled through register settings. However, for optimum receiver performance, especially in Continuous receive mode, its use is strongly advised.

The Bit Synchronizer is automatically activated in Packet mode. Its bit rate is controlled by *BitRateMsb* and *BitRateLsb* in *RegBitrate.*

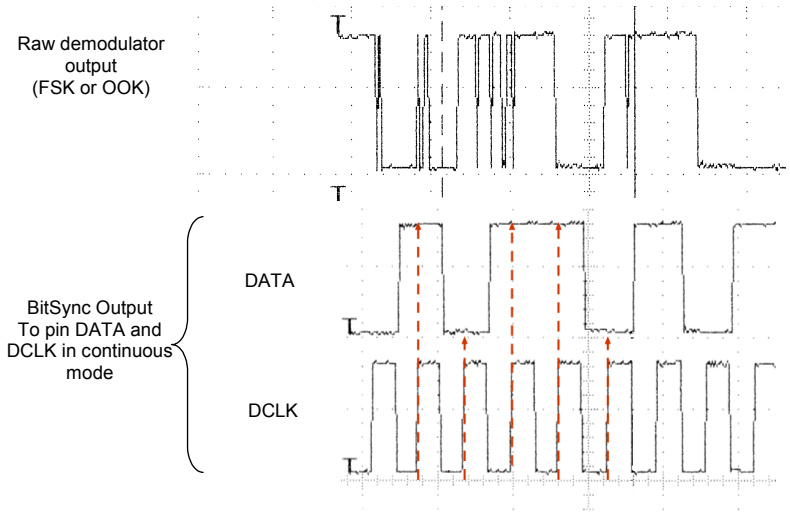


Figure 17. Bit Synchronizer Description

To ensure correct operation of the Bit Synchronizer the following conditions have to be satisfied:

- A preamble (0x55 or 0xAA) of at least 12 bits is required for synchronization, the longer the synchronization phase is the better the ensuing packet detection rate will be.
- The subsequent payload bit stream must have at least one edge transition (either rising or falling) every 16 bits during data transmission.
- The absolute error between transmitted and received bit rate must not exceed 6.5%.

4.2.3.4. Frequency Error Indicator

This frequency error indicator measures the frequency error between the programmed RF centre frequency and the carrier frequency of the modulated input signal to the receiver. When the FEI is performed the frequency error is measured and the signed result is loaded in *FeiValue* in *RegFei* in 2's complement format. The time required for an FEI evaluation is 4 bit periods.

To ensure correct operation of the FEI:

- The measurement must be launched during the reception of preamble.
- The sum of the frequency offset and the 20 dB signal bandwidth must be lower than the base band filter bandwidth. i.e. The whole modulated spectrum must be received.

The 20 dB bandwidth of the signal can be evaluated as follows (double-side bandwidth):

$$BW_{20dB} = 2 \times \left(F_{DEV} + \frac{BR}{2} \right)$$

The frequency error, in Hz, can be calculated with the following formula:

$$FEI = F_{STEP} \times FeiValue$$

The FEI is enabled automatically upon the transition to receive mode and automatically updated every 4 bits.

4.2.3.5. AFC

The AFC is based on the FEI measurement therefore the same input signal and receiver setting conditions apply. When the AFC procedure is performed the *AfcValue* is directly subtracted from the register that defines the frequency of operation of the chip, F_{RF}. The AFC is executed each time the receiver is enabled, if *AfcAutoOn* = 1.

When the AFC is enabled (*AfcAutoOn* = 1) the user has the option to:

- Clear the former AFC correction value if *AfcAutoClearOn* = 1. Allowing the next frequency correction to be performed from the initial centre frequency.
- Start the AFC evaluation from the previously corrected frequency. This may be useful in systems in which the centre frequency experiences cumulative drift - such as the ageing of a crystal reference.

The SX1272/73 offers an alternate receiver bandwidth setting during the AFC phase allowing the accommodation of larger frequency errors. The setting *RegAfcBw* sets the receive bandwidth during the AFC process. In a typical receiver application, once the AFC is performed, the radio will revert to the receiver communication or channel bandwidth (*RegRxBw*) for the ensuing communication phase.

Note that the FEI measurement is valid only during the reception of preamble. The provision of the *PreambleDetect* flag can hence be used to detect this condition and allow a reliable AFC or FEI operation to be triggered. This process can be performed automatically by using the appropriate options in *StartDemodOnPreamble* found in the *RegRxConfig* register.

A detailed description of the receiver setup to enable the AFC is provided in section 4.2.7.

4.2.3.6. Preamble Detector

The Preamble Detector indicates the reception of a carrier modulated with a 0101...sequence. It is insensitive to the frequency offset, as long as the receiver bandwidth is large enough. The size of detection can be programmed from 1 to 3 bytes with *PreambleDetectorSize* in *RegPreambleDetect* as defined in the next table.

Table 19 Preamble Detector Settings

PreambleDetectorSize	# of Bytes
00	1
01	2 (recommended)
10	3
11	reserved

For normal operation, *PreambleDetectTol* should be set to be set to 10 (0x0A) with a qualifying preamble size of 2 bytes.

The *PreambleDetect* interrupt (either in *RegIrqFlags1* or mapped to a specific DIO) then goes high every time a valid preamble is detected assuming *PreambleDetectorOn*=1.

The preamble detector can also be used as a gate to ensure that AFC and AGC are performed on valid preamble. See section 4.2.7. for details.

4.2.3.7. Image Rejection Mixer

The SX1272/73 employs an image rejection mixer (IRM) which, uncalibrated, gives 35 dB image rejection. The low phase noise PLL is used to perform calibration of the receiver chain. Which increases the typical image rejection to 48 dB. This process is fully automated in FSK/OOK mode and radio power-up.

4.2.3.8. Image and RSSI Calibration

An automatic calibration process is used to calibrate the phase and gain of both I and Q receive paths. This calibration allows enhanced image frequency rejection and improves the RSSI precision. This calibration process is launched under the following circumstances:

- Automatically at Power On Reset or after a Manual Reset of the chip (refer to section 7.2). For applications where the temperature remains stable, or if the Image Rejection is not a major concern, this single calibration will suffice.
- Automatically when a pre-defined temperature change is observed.
- Upon User request, by setting bit *ImageCalStart* in *RegImageCal*, when the device is in Standby mode. Note that in LoRa™ mode the calibration command is inaccessible. To perform the calibration the radio must be returned temporarily to FSK/OOK mode.

A selectable temperature change, set with *TempThreshold* (5, 10, 15 or 20°C), is detected and reported in *TempChange* if the temperature monitoring is turned On with *TempMonitorOff* = 0.

This interrupt flag can be used by the application to launch a new image calibration at a convenient time if *AutoImageCalOn*=0, or immediately when this temperature variation is detected, if *AutoImageCalOn*=1.

The calibration process takes approximately 10 ms.

4.2.3.9. Timeout Function

The SX1272/73 includes a Timeout function, which allows the automation of a duty-cycled recceive oprtation where the radio periodically wakes from sleep mode into receiver mode.

- *Timeout* interrupt is generated *TimeoutRxRssi x 16 x Tbit* after switching to Rx mode if the *Rssi* flag does not raise within this time frame (*RssiValue* > *RssiThreshold*).
- *Timeout* interrupt is generated *TimeoutRxPreamble x 16 x Tbit* after switching to Rx mode if the *PreambleDetect* flag does not raise within this time frame.
- *Timeout* interrupt is generated *TimeoutSignalSync x 16 x Tbit* after switching to Rx mode if the *SyncAddress* flag does not raise within this time frame.

This timeout interrupt can be used to warn the companion processor to shut down the receiver and return to a lower power mode. To become active, these timeouts must also be enabled by setting the correct *RxTrigger* parameters in *RegRxConfig:*

Table 20 RxTrigger Settings to Enable Timeout Interrupts

Receiver Triggering Event	*RxTrigger (2:0)*	*Timeout on Rssi*	*Timeout on Preamble*	*Timeout on SyncAddress*
None	000	Off	Off	Active
Rssi Interrupt	001	Active	Off	
PreambleDetect	110	Off	Active	
Rssi Interrupt & *PreambleDetect*	111	Active	Active	

4.2.4. Operating Modes in FSK/OOK Mode

4.2.5. General Overview

The SX1272/73 has several working modes, manually programmed in *RegOpMode*. Fully automated mode selection, packet transmission and reception is also possible using the Top Level Sequencer described in Section 4.2.9.

Table 21 Basic Transceiver Modes

Mode	Selected mode	Symbol	Enabled blocks
000	Sleep mode	Sleep	None
001	Standby mode	Stdby	Top regulator and crystal oscillator
010	Frequency synthesiser to Tx frequency	FSTx	Frequency synthesizer at Tx frequency (Frf)
011	Transmit mode	Tx	Frequency synthesizer and transmitter
100	Frequency synthesiser to Rx frequency	FSRx	Frequency synthesizer at frequency for reception (Frf-IF)
101	Receive mode	Rx	Frequency synthesizer and receiver

When switching from a mode to another the sub-blocks are woken up according to a pre-defined optimized sequence.

4.2.6. Startup Times

The startup time of the transmitter or the receiver is dependent upon which mode the transceiver was in at the beginning. For a complete description, Figure 18 below shows a complete startup process, from the lower power mode “Sleep”.

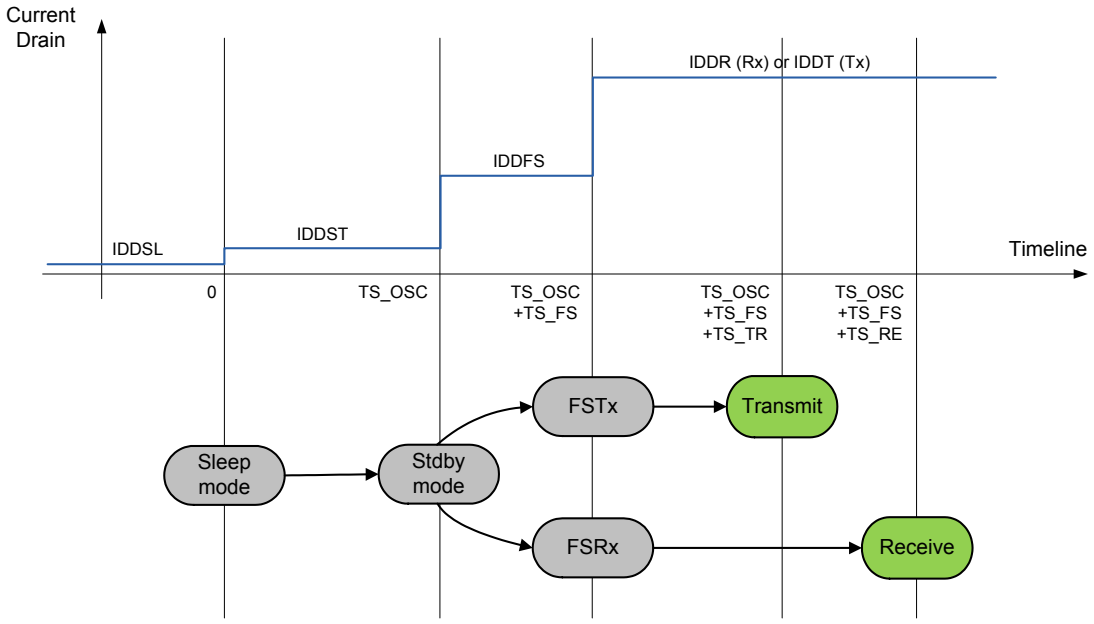


Figure 18. Startup Process

TS_OSC is the startup time of the crystal oscillator which depends on the electrical characteristics of the crystal. TS_FS is the startup time of the PLL including systematic calibration of the VCO.

Typical values of TS_OSC and TS_FS are given in Section 2.5.2.

4.2.6.1. Transmitter Startup Time

The transmitter startup time, TS_TR, is calculated as follows in FSK mode:

$$TS\_TR = 5\mu s + 1.25 \times PaRamp + \frac{1}{2} \times Tbit$$

,

where *PaRamp* is the ramp-up time programmed in *RegPaRamp* and *Tbit* is the bit time.

In OOK mode, this equation can be simplified to the following:

$$TS\_TR = 5\mu s + \frac{1}{2} \times Tbit$$

4.2.6.2. Receiver Startup Time

The receiver startup time, TS_RE, only depends upon the receiver bandwidth effective at the time of startup. When AFC is enabled (*AfcAutoOn*=1) the *AfcBw* should be used instead of *RxBw* to extract the receiver startup time:

Table 22 Receiver Startup Time Summary

RxBw if *AfcAutoOn=0* *RxBwAfc* if *AfcAutoOn=1*	TS_RE (+/-5%)
2.6 kHz	2.33 ms
3.1 kHz	1.94 ms
3.9 kHz	1.56 ms
5.2 kHz	1.18 ms
6.3 kHz	984 us
7.8 kHz	791 us
10.4 kHz	601 us
12.5 kHz	504 us
15.6 kHz	407 us
20.8 kHz	313 us
25.0 kHz	264 us
31.3 kHz	215 us
41.7 kHz	169 us
50.0 kHz	144 us
62.5 kHz	119 us
83.3 kHz	97 us
100.0 kHz	84 us
125.0 kHz	71 us
166.7 kHz	85 us
200.0 kHz	74 us
250.0 kHz	63 us

TS_RE or later after setting the device in Receive mode, any incoming packet will be detected and demodulated by the transceiver.

4.2.6.3. Time to RSSI Evaluation

The first RSSI sample will be available TS_RSSI after the receiver is ready: equivalently TS_RE + TS_RSSI after the receive mode instruction was issued.

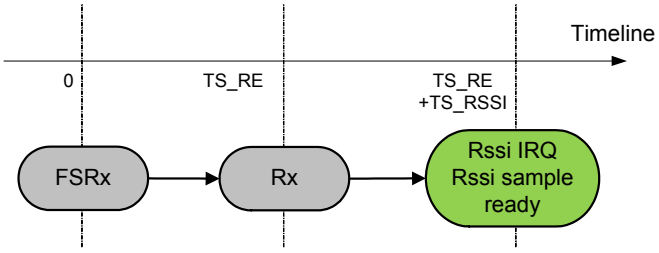


Figure 19. Time to Rssi Sample

TS_RSSI depends on the receiver bandwidth as well as the *RssiSmoothing* option that was selected. The formula used to calculate TS_RSSI is provided in section 5.5.4.

4.2.6.4. Tx to Rx Turnaround Time

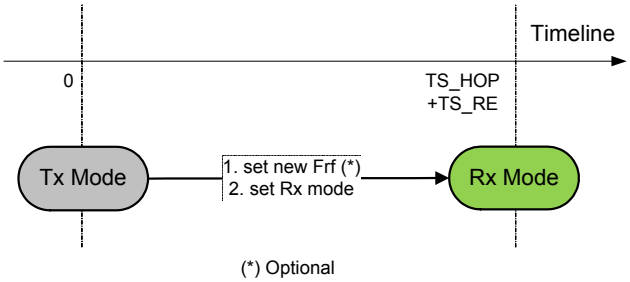


Figure 20. Tx to Rx Turnaround

Note The SPI instruction times are omitted, as they can generally be very small as compared to other timings (up to 10 MHz SPI clock).

4.2.6.5. Rx to Tx

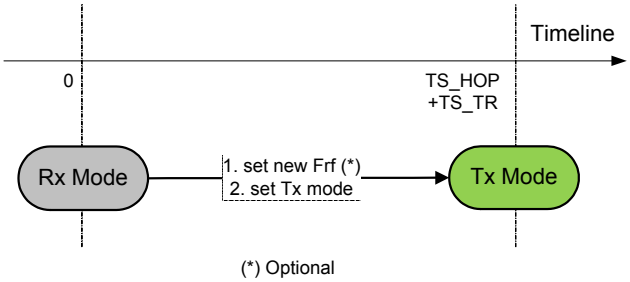


Figure 21. Rx to Tx Turnaround

4.2.6.6. Receiver Hopping, Rx to Rx

Two methods are possible:

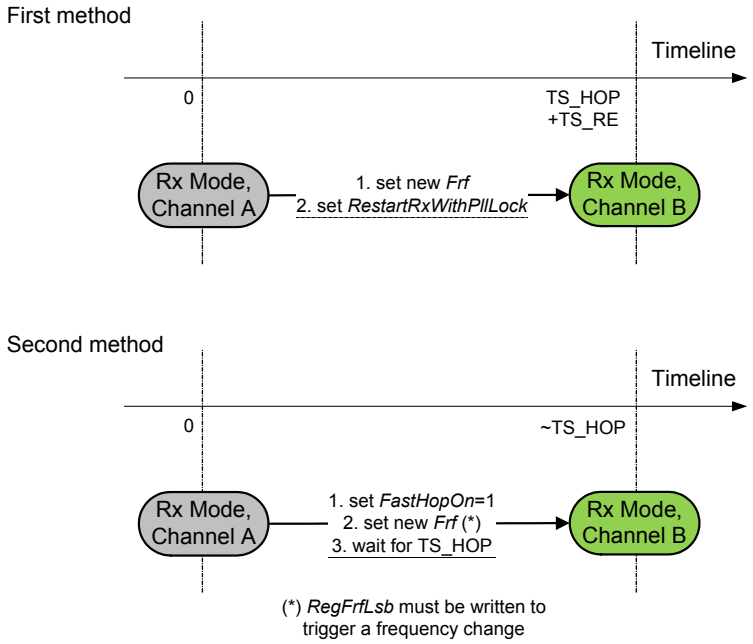


Figure 22. Receiver Hopping

The second method is quicker and should be used if a very quick RF sniffing mechanism is to be implemented.

4.2.6.7. Tx to Tx

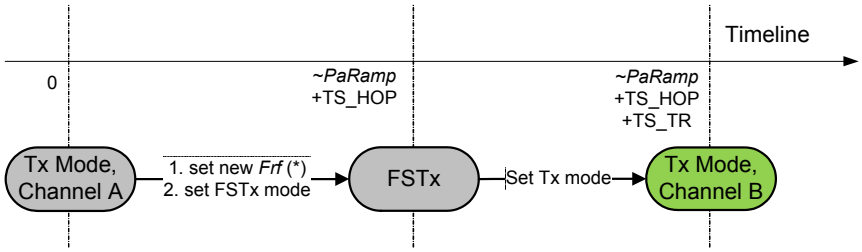


Figure 23. Transmitter Hopping

4.2.7. Receiver Startup Options

The SX1272/73 receiver can automatically control the gain of the receive chain (AGC) and adjust the receiver LO frequency (AFC). Those processes are carried out on a packet-by-packet basis. They occur:

- When the receiver is turned On.
- When the Receiver is restarted upon user request, through the use of trigger bits *RestartRxWithoutPllLock* or *RestartRxWithPllLock* in *RegRxConfig*.
- When the receiver is automatically restarted after the reception of a valid packet or after a packet collision.

Automatic restart capabilities are detailed in Section 4.2.8.

The receiver startup options available in SX1272/73 are described in Table 23.

Table 23 Receiver Startup Options

Triggering Event	Realized Function	*AgcAutoOn*	*AfcAutoOn*	*RxTrigger (2:0)*
None	None	0	0	000
Rssi Interrupt	AGC	1	0	001
	AGC & AFC	1	1	001
PreambleDetect	AGC	1	0	110
	AGC & AFC	1	1	110
Rssi Interrupt & *PreambleDetect*	AGC	1	0	111
	AGC & AFC	1	1	111

When *AgcAutoOn*=0, the LNA gain is manually selected by choosing *LnaGain* bits in *RegLna.*

4.2.8. Receiver Restart Methods

The options for restart of the receiver are covered below. This is typically of use to prepare for the reception of a new signal whose strength or carrier frequency is different from the preceding packet to allow the AGC or AFC to be re-evaluated.

4.2.8.1. Restart Upon User Request

In Receive mode the user can request a receiver restart - this can be useful in conjunction with the use of a Timeout interrupt following a period of inactivity in the channel of interest. Two options are available:

- No change in the Local Oscillator upon restart: the AFC is disabled, and the *Frf* register has not been changed through SPI before the restart instruction: set bit *RestartRxWithoutPllLock* in *RegRxConfig* to 1.
- Local Oscillator change upon restart: if AFC is enabled (*AfcAutoOn*=1), and/or the *Frf* register had been changed during the last Rx period: set bit *RestartRxWithPllLock* in *RegRxConfig* to 1.

Note ModeReady must be at logic level 1 for a new RestartRx command to be taken into account.

4.2.8.2. Automatic Restart after valid Packet Reception

The bits *AutoRestartRxMode* in *RegSyncConfig* control the automatic restart feature of the SX1272/73 receiver, when a valid packet has been received:

- If *AutoRestartRxMode* = 00, the function is off, and the user should manually restart the receiver upon valid packet reception (see section 4.2.8.1*).*
- If *AutoRestartRxMode* = 01, after the user has emptied the FIFO following a *PayloadReady* interrupt, the receiver will automatically restart itself after a delay of *InterPacketRxDelay*, allowing for the distant transmitter to ramp down, hence avoiding a false RSSI detection on the ‘tail’ of the previous packet.
- If *AutoRestartRxMode* = 10 should be used if the next reception is expected on a new frequency, i.e. *Frf* is changed after the reception of the previous packet. An additional delay is systematically added, in order for the PLL to lock at a new frequency.

4.2.8.3. Automatic Restart when Packet Collision is Detected

In receive mode the SX1272/73 is able to detect packet collision and restart the receiver. Collisions are detected by a sudden rise in received signal strength, detected by the RSSI. This functionality can be useful in network configurations where many asynchronous slaves attempt periodic communication with a single a master node.

The collision detector is enabled by setting bit *RestartRxOnCollision* to 1.

The decision to restart the receiver is based on the detection of RSSI change. The sensitivity of the system can be adjusted in 1 dB steps by using register *RssiCollisionThreshold* in *RegRxConfig*.

4.2.9. Top Level Sequencer

Depending on the application it may be desirable to be able to change the mode of the circuit according to a predefined sequence without access to the serial interface. In order to define different sequences or scenarios a user-programmable state machine called the Top Level Sequencer (herein reffered to as the Sequencer) can automatically control the chip modes.

NOTE THAT THIS FUNCTIONALITY IS ONLY AVAILABLE IN FSK/OOK MODE.

The Sequencer is activated by setting the *SequencerStart* bit in *RegSeqConfig1* to 1 in Sleep or Standby mode (called initial mode).

It is also possible to force the Sequencer off by setting the *Stop* bit in *RegSeqConfig1* to 1 at any time.

Note SequencerStart and Stop bit must never be set at the same time.

4.2.9.1. Sequencer States

As shown in the table below, with the aid of a pair of interrupt timers (T1 and T2), the sequencer can take control of the chip operation in all modes.

Table 24 Sequencer States

Sequencer State	Description
SequencerOff State	The Sequencer is not activated. Sending a *SequencerStart* command will launch it. When coming from **LowPowerSelection** state, the Sequencer will be Off, whilst the chip will return to its initial mode (either Sleep or Standby mode).
Idle State	The chip is in low-power mode, either *Standby* or *Sleep*, as defined by *IdleMode* in *RegSeqConfig1*. The Sequencer waits only for the *T1* interrupt.
Transmit State	The transmitter in on.
Receive State	The receiver in on.
PacketReceived	The receiver is on and a packet has been received. It is stored in the FIFO.
LowPowerSelection	Selects low power state (**SequencerOff** or **Idle** State)

RxTimeout	Defines the action to be taken on a RxTimeout interrupt. RxTimeout interrupt can be a *TimeoutRxRssi*, *TimeoutRxPreamble* or *TimeoutSignalSync* interrupt.

4.2.9.2. Sequencer Transitions

The transitions between sequencer states are listed in the forthcoming table.

Table 25 Sequencer Transition Options

Variable	Transition
IdleMode	Selects the chip mode during **Idle** state: 0: *Standby* mode 1: *Sleep* mode
FromStart	Controls the Sequencer transition when the *SequencerStart* bit is set to 1 in *Sleep* or *Standby* mode: 00: to **LowPowerSelection** 01: to **Receive** state 10: to **Transmit** state 11: to **Transmit** state on a *FifoThreshold* interrupt
LowPowerSelection	Selects Sequencer LowPower state after a *to LowPowerSelection* transition 0: **SequencerOff** state with chip on Initial mode 1: **Idle** state with chip on *Standby* or *Sleep* mode depending on **IdleMode** Note: Initial mode is the chip LowPower mode at Sequencer start.
FromIdle	Controls the Sequencer transition from the **Idle** state on a *T1* interrupt: 0: to **Transmit** state 1: to **Receive** state
FromTransmit	Controls the Sequencer transition from the **Transmit** state: 0: to **LowPowerSelection** on a *PacketSent* interrupt 1: to **Receive** state on a *PacketSent* interrupt
FromReceive	Controls the Sequencer transition from the **Receive** state: 000 and 111: unused 001: to **PacketReceived** state on a *PayloadReady* interrupt 010: to **LowPowerSelection** on a *PayloadReady* interrupt 011: to **PacketReceived** state on a *CrcOk* interrupt. If CRC is wrong (corrupted packet, with CRC on but CrcAutoClearOn is off), the PayloadReady interrupt will drive the sequencer to RxTimeout state. 100: to **SequencerOff** state on a *Rssi* interrupt 101: to **SequencerOff** state on a *SyncAddress* interrupt 110: to **SequencerOff** state on a *PreambleDetect* interrupt Irrespective of this setting, transition to **LowPowerSelection** on a *T2* interrupt
FromRxTimeout	Controls the state-machine transition from the **Receive** state on a *RxTimeout* interrupt (and on *PayloadReady* if **FromReceive** = 011): 00: to **Receive** state via *ReceiveRestart* 01: to **Transmit** state 10: to **LowPowerSelection** 11: to **SequencerOff** state Note: RxTimeout interrupt is a *TimeoutRxRssi*, *TimeoutRxPreamble* or *TimeoutSignalSync* interrupt.

FromPacketReceived	Controls the state-machine transition from the **PacketReceived** state: 000: to **SequencerOff** state 001: to **Transmit** on a *FifoEmpty* interrupt 010: to **LowPowerSelection** 011: to **Receive** via *FS* mode, if frequency was changed 100: to **Receive** state (no frequency change)

4.2.9.3. Timers

Two timers (Timer1 and Timer2) are also available in order to define periodic sequences. These timers are used to generate interrupts, which can trigger transitions of the Sequencer.

T1 interrupt is generated (Timer1Resolution * Timer1Coefficient) after ***T2* interrupt** or ***SequencerStart***. command.

T2 interrupt is generated (Timer2Resolution * Timer2Coefficient) after ***T1* interrupt**.

The timer mechanism is summarized on the following diagram.

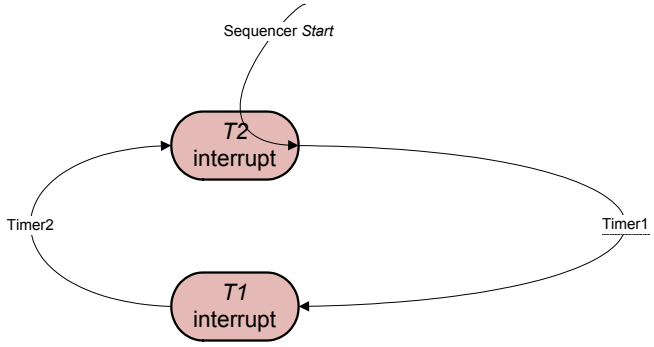


Figure 24. Timer1 and Timer2 Mechanism

Note The timer sequence is completed independently of the actual Sequencer state. Thus, both timers need to be on to achieve periodic cycling.

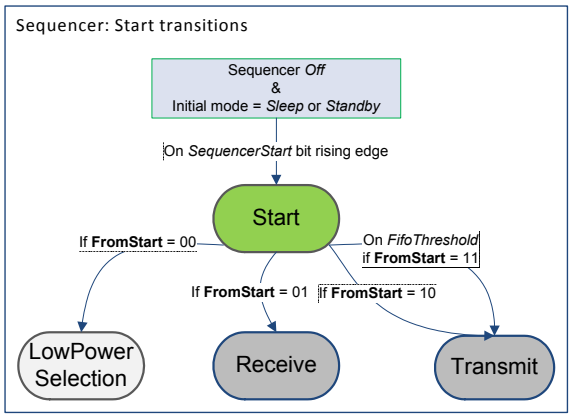
Table 26 Sequencer Timer Settings

Variable	Description
Timer1Resolution	Resolution of Timer1 00: disabled 01: 64 us 10: 4.1 ms 11: 262 ms
Timer2Resolution	Resolution of Timer2 00: disabled 01: 64 us 10: 4.1 ms 11: 262 ms
Timer1Coefficient	Multiplying coefficient for Timer1
Timer2Coefficient	Multiplying coefficient for Timer2

4.2.9.4. Sequencer State Machine

The following graphs summarize every possible transition between each Sequencer state. The Sequencer states are highlighted in grey. The transitions are represented by arrows. The condition activating them is described over the transition arrow. For better readability, the start transitions are separated from the rest of the graph.

Transitory states are highlighted in light grey, and exit states are represented in red. It is also possible to force the Sequencer off by setting the *Stop* bit in *RegSeqConfig1* to 1 at any time.



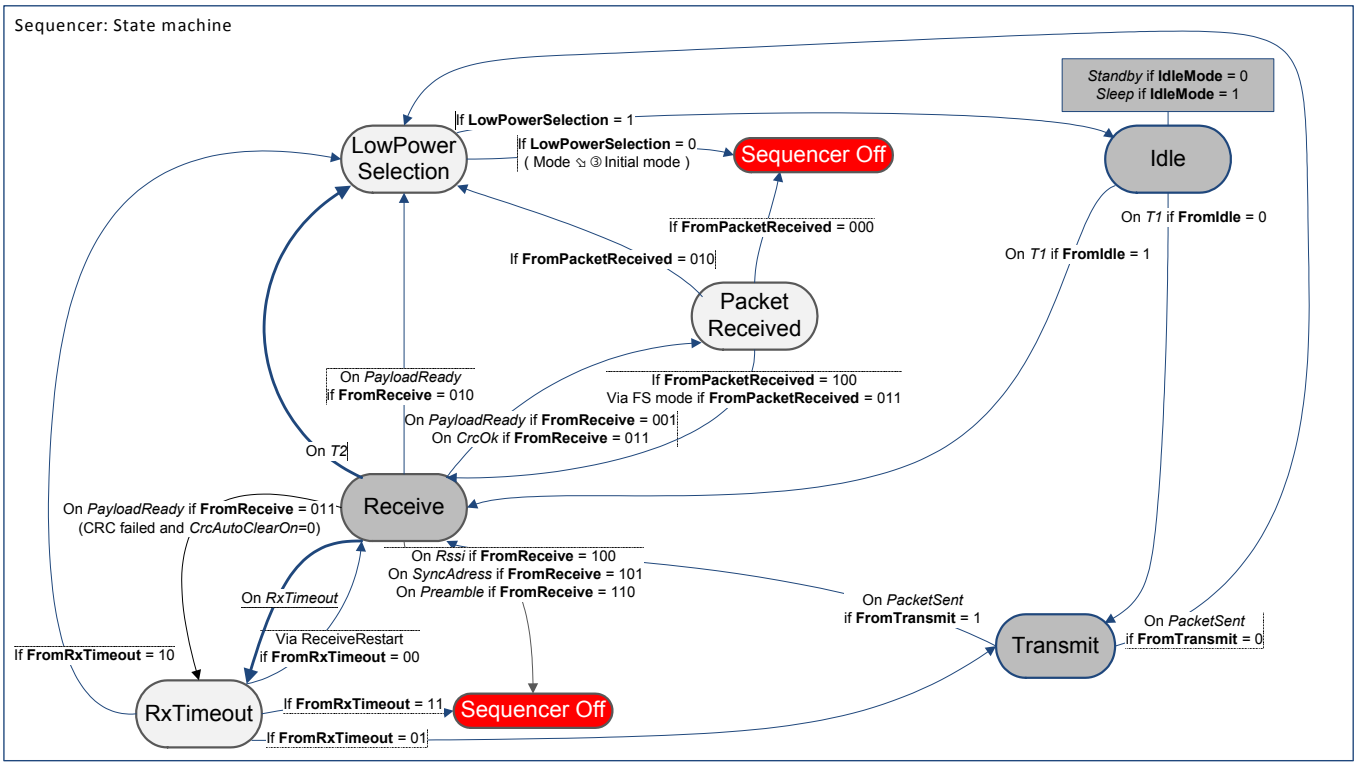


Figure 25. Sequencer State Machine

4.2.10. Data Processing in FSK/OOK Mode

4.2.10.1. Block Diagram

Figure below illustrates the SX1272/73 data processing circuit. Its role is to interface the data to/from the modulator/ demodulator and the uC access points (SPI and DIO pins). It also controls all the configuration registers.

The circuit contains several control blocks which are described in the following paragraphs.

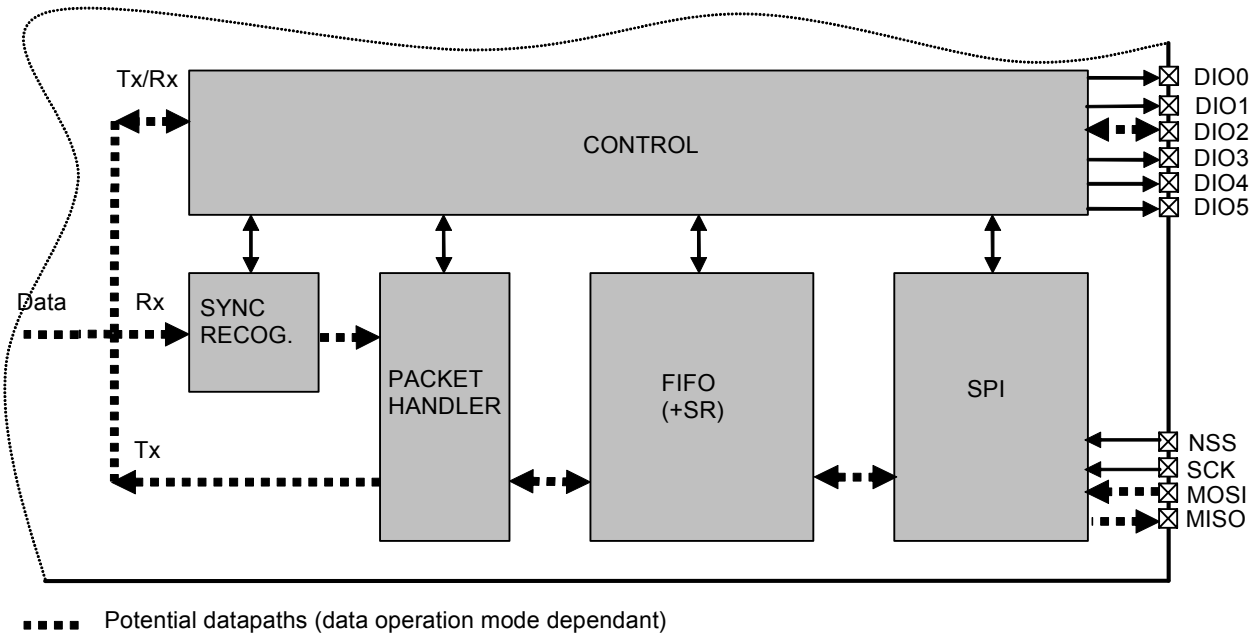


Figure 26. SX1272/73 Data Processing Conceptual View

The SX1272/73 implements several data operation modes each with their own data path through the data processing section. Depending on the data operation mode selected some control blocks are active whilst others remain disabled.

4.2.10.2. Data Operation Modes

The SX1272/73 has two different data operation modes selectable by the user:

- Continuous mode: each bit transmitted or received is accessed in real time at the DIO2/DATA pin. This mode may be used if adequate external signal processing is available.
- Packet mode (recommended): user only provides/retrieves payload bytes to/from the FIFO. The packet is automatically built with preamble, Sync word, and optional CRC and DC-free encoding schemes The reverse operation is performed in reception. The uC processing overhead is hence significantly reduced compared to Continuous mode. Depending on the optional features activated (CRC, etc) the maximum payload length is limited to 255, 2047 bytes or unlimited.

Each of these data operation modes is fully described in the following sections.

4.2.11. FIFO

Overview and Shift Register (SR)

In packet mode of operation both data to be transmitted and that has been received are stored in a configurable FIFO (First In First Out). It is accessed via the SPI interface and provides several interrupts for transfer management.

The FIFO is 1 byte wide hence it only performs byte (parallel) operations, whereas the demodulator functions serially. A shift register is therefore employed to interface the two devices. In transmit mode it takes bytes from the FIFO and outputs them serially (MSB first) at the programmed bit rate to the modulator. Similarly, in Rx the shift register gets bit by bit data from the demodulator and writes them byte by byte to the FIFO. This is illustrated in figure below.

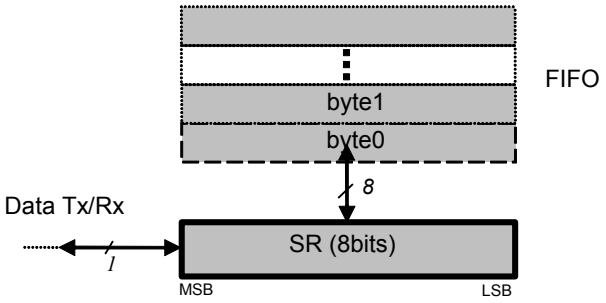


Figure 27. FIFO and Shift Register (SR)

Note When switching to Sleep mode, the FIFO can only be used once the ModeReady flag is set (immediately from all modes except from Tx)

The FIFO size is fixed to 64 bytes.

Interrupt Sources and Flags

- *FifoEmpty*: *FifoEmpty* interrupt source is high when byte 0, i.e. whole FIFO, is empty. Otherwise it is low. Note that when retrieving data from the FIFO, *FifoEmpty* is updated on NSS falling edge, i.e. when *FifoEmpty* is updated to low state the currently started read operation must be completed. In other words, *FifoEmpty* state must be checked after each read operation for a decision on the next one (*FifoEmpty* = 0: more byte(s) to read; *FifoEmpty* = 1: no more byte to read).
- *FifoFull*: *FifoFull* interrupt source is high when the last FIFO byte, i.e. the whole FIFO, is full. Otherwise it is low.
- *FifoOverrunFlag*: *FifoOverrunFlag* is set when a new byte is written by the user (in Tx or Standby modes) or the SR (in Rx mode) while the FIFO is already full. Data is lost and the flag should be cleared by writing a 1, note that the FIFO will also be cleared.
- *PacketSent*: *PacketSent* interrupt source goes high when the SR's last bit has been sent.
- *FifoLevel*: Threshold can be programmed by *FifoThreshold* in *RegFifoThresh*. Its behavior is illustrated in figure below.

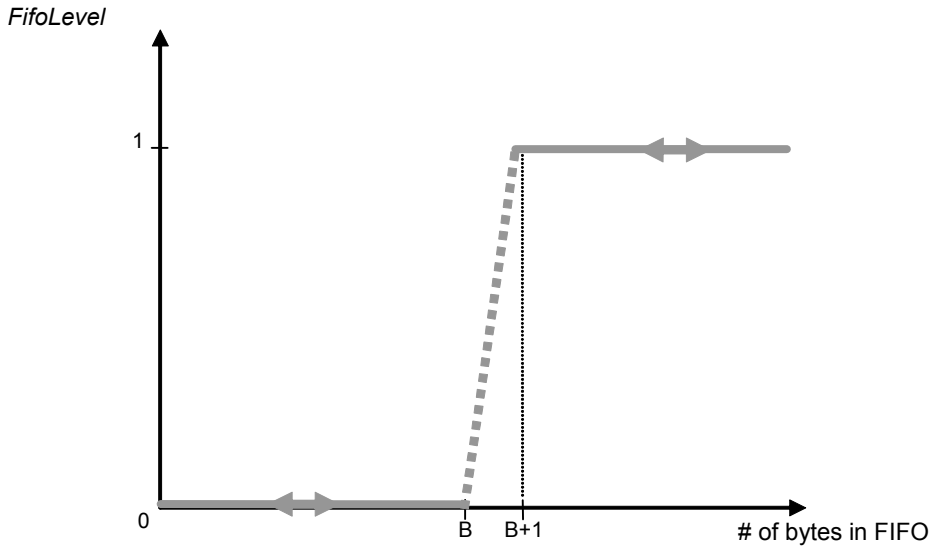


Figure 28. FifoLevel IRQ Source Behavior

Notes - FifoLevel interrupt is updated only after a read or write operation on the FIFO. Thus the interrupt cannot be dynamically updated by only changing the FifoThreshold parameter

- FifoLevel interrupt is valid as long as FifoFull does not occur. An empty FIFO will restore its normal operation

FIFO Clearing

Table below summarizes the status of the FIFO when switching between different modes

Table 27 Status of FIFO when Switching Between Different Modes of the Chip

From	To	FIFO status	Comments
Stdby	Sleep	Not cleared	
Sleep	Stdby	Not cleared	
Stdby/Sleep	Tx	Not cleared	To allow the user to write the FIFO in Stdby/Sleep before Tx
Stdby/Sleep	Rx	Cleared	
Rx	Tx	Cleared	
Rx	Stdby/Sleep	Not cleared	To allow the user to read FIFO in Stdby/Sleep mode after Rx
Tx	Any	Cleared	

4.2.11.1. Sync Word Recognition

Overview

Sync word recognition (also called Pattern recognition) is activated by setting *SyncOn* in *RegSyncConfig*. The bit synchronizer must also be activated in Continuous mode (automatically done in Packet mode).

The block behaves like a shift register as it continuously compares the incoming data with its internally programmed Sync word and sets *SyncAddressMatch* when a match is detected. This is illustrated in Figure 29 below.

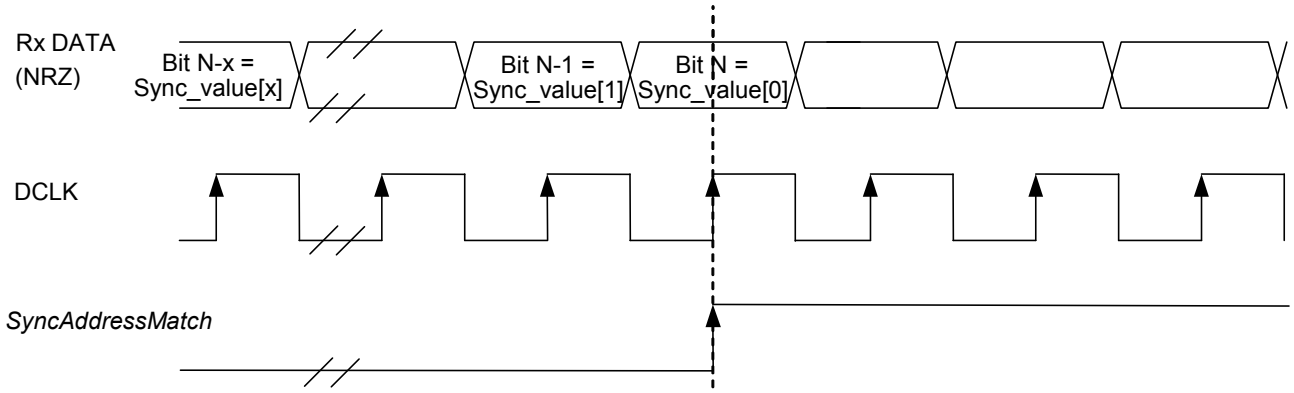


Figure 29. Sync Word Recognition

During the comparison of the demodulated data, the first bit received is compared with bit 7 (MSB) of *RegSyncValue1* and the last bit received is compared with bit 0 (LSB) of the last byte whose address is determined by the length of the Sync word.

When the programmed Sync word is detected the user can assume that this incoming packet is for the node and can be processed accordingly.

SyncAddressMatch is cleared when leaving Rx or FIFO is emptied.

Configuration

- Size: Sync word size can be set from 1 to 8 bytes (i.e. 8 to 64 bits) via *SyncSize* in *RegSyncConfig*. In Packet mode this field is also used for Sync word generation in Tx mode.
- Value: The Sync word value is configured in *SyncValue(63:0)*. In Packet mode this field is also used for Sync word generation in Tx mode.

Note SyncValue choices containing 0x00 bytes are not allowed

Packet Handler

The packet handler is the block used in Packet mode. Its functionality is fully described in section 4.2.14.

Control

The control block configures and controls the full chip's behavior according to the settings programmed in the configuration registers.

4.2.12. Digital IO Pins Mapping

Six general purpose IO pins are available on the SX1272/73 and their configuration in Continuous or Packet mode is controlled through *RegDioMapping1* and *RegDioMapping2.*

Table 28 DIO Mapping, Continuous Mode

	DIOx Mapping	Sleep	Standby	FSRx/Tx	Rx	Tx
DIO0	00	-			SyncAddress	TxReady
	01	-			Rssi / PreambleDetect	-
	10	-			RxReady	TxReady
	11	-				
DIO1	00	-			Dclk	
	01	-			Rssi / PreambleDetect	-
	10	-				
	11	-				
DIO2	00	-			Data	
	01	-			Data	
	10	-			Data	
	11	-			Data	
DIO3	00	-			Timeout	-
	01	-			Rssi / PreambleDetect	-
	10	-				
	11	-	TempChange / LowBat		TempChange / LowBat	
DIO4	00	-		TempChange / LowBat		
	01	-		PllLock		
	10	-			TimeOut	-
	11	-	ModeReady		ModeReady	
DIO5	00	ClkOut if RC	ClkOut		ClkOut	
	01	-		PllLock		
	10	-			Rssi / PreambleDetect	-
	11	-	ModeReady		ModeReady	

Table 29 DIO Mapping, Packet Mode

	DIOx Mapping	Sleep	Standby	FSRx/Tx	Rx	Tx
DIO0	00	-			PayloadReady	PacketSent
	01	-			CrcOk	-
	10	-				
	11	-	TempChange / LowBat		TempChange / LowBat	
DIO1	00	FifoLevel		FifoLevel	FifoLevel	
	01	FifoEmpty		FifoEmpty	FifoEmpty	
	10	FifoFull		FifoFull	FifoFull	
	11	-				
DIO2	00	FifoFull		FifoFull	FifoFull	
	01	-			RxReady	-
	10	FifoFull			TimeOut	FifoFull
	11	FifoFull			SyncAddress	FifoFull
DIO3	00	FifoEmpty		FifoEmpty	FifoEmpty	
	01	-				TxReady
	10	FifoEmpty		FifoEmpty	FifoEmpty	
	11	FifoEmpty		FifoEmpty	FifoEmpty	
DIO4	00	-	TempChange / LowBat		TempChange / LowBat	
	01	-		PllLock		
	10	-			TimeOut	-
	11	-			Rssi / PreambleDetect	-
DIO5	00	ClkOut if RC	ClkOut		ClkOut	
	01	-		PllLock		
	10	-			Data	
	11	-	ModeReady		ModeReady	

4.2.13. Continuous Mode

4.2.13.1. General Description

As illustrated in Figure 30 in Continuous mode the NRZ data to (from) the (de)modulator is directly accessed by the uC on the bidirectional DIO2/DATA pin. The FIFO and packet handler are thus inactive.

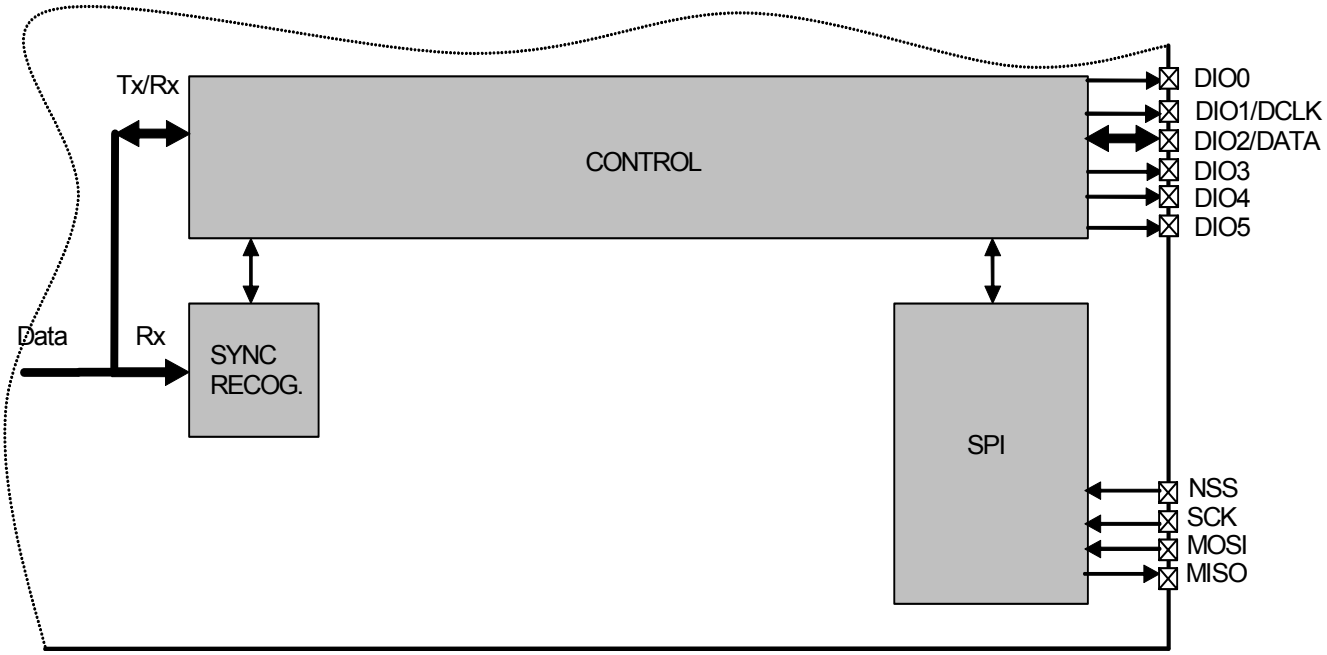


Figure 30. Continuous Mode Conceptual View

4.2.13.2. Tx Processing

In Tx mode a synchronous data clock for an external uC is provided on DIO1/DCLK pin. Clock timing with respect to the data is illustrated in Figure 31. DATA is internally sampled on the rising edge of DCLK so the uC can change logic state anytime outside the grayed out setup/hold zone.

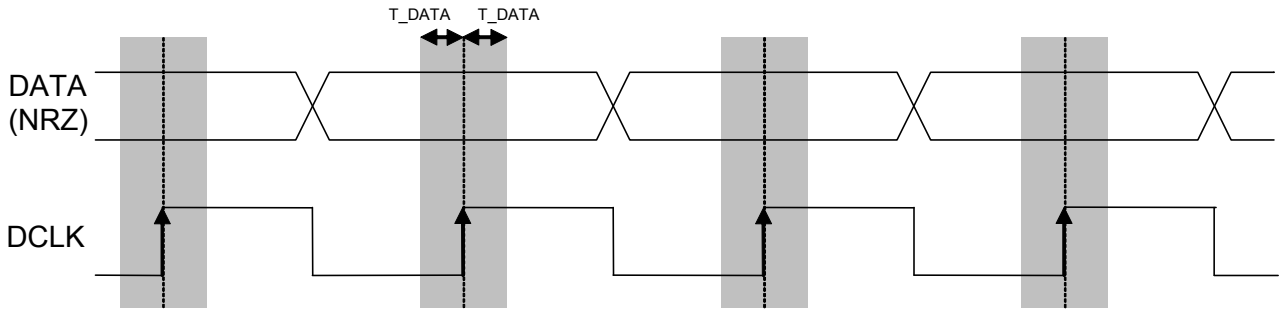


Figure 31. Tx Processing in Continuous Mode

Note the use of DCLK is required when the modulation shaping is enabled (see section 4.2.13.2).

4.2.13.3. Rx Processing

If the bit synchronizer is disabled the raw demodulator output is made directly available on DATA pin and no DCLK signal is provided.

Conversely, if the bit synchronizer is enabled, synchronous cleaned data and clock are made available respectively on DIO2/DATA and DIO1/DCLK pins. DATA Is sampled on the rising edge of DCLK and updated on the falling edge as illustrated below.

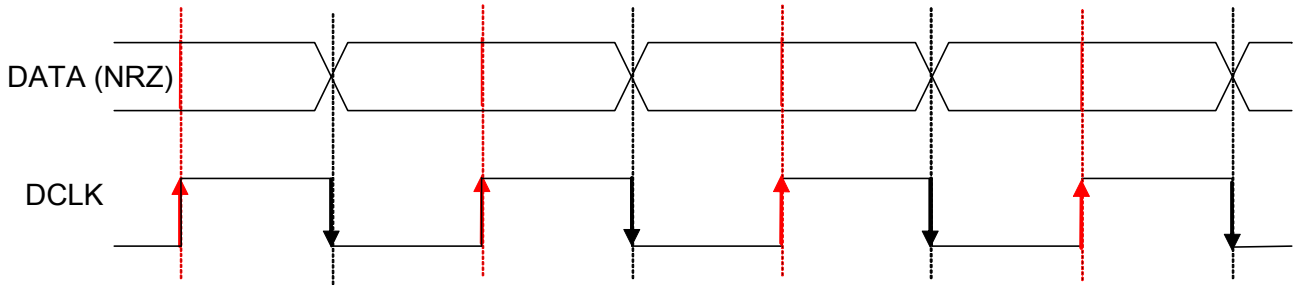


Figure 32. Rx Processing in Continuous Mode

Note In Continuous mode it is always recommended to enable the bit synchronizer to clean the DATA signal even if the DCLK signal is not used by the uC (bit synchronizer is automatically enabled in Packet mode).

4.2.14. Packet Mode

4.2.14.1. General Description

In Packet mode the NRZ data to (from) the (de)modulator is not directly accessed by the uC but stored in the FIFO and accessed via the SPI interface.

In addition the SX1272/73 packet handler performs several packet oriented tasks such as Preamble and Sync word generation, CRC calculation/check, whitening/dewhitening of data, Manchester encoding/decoding, address filtering, etc. This simplifies software and reduces uC overhead by performing these repetitive tasks within the RF chip itself.

Another important feature is ability to fill and empty the FIFO in Sleep/Stdby mode to ensure minimum power consumption when accessing payload data.

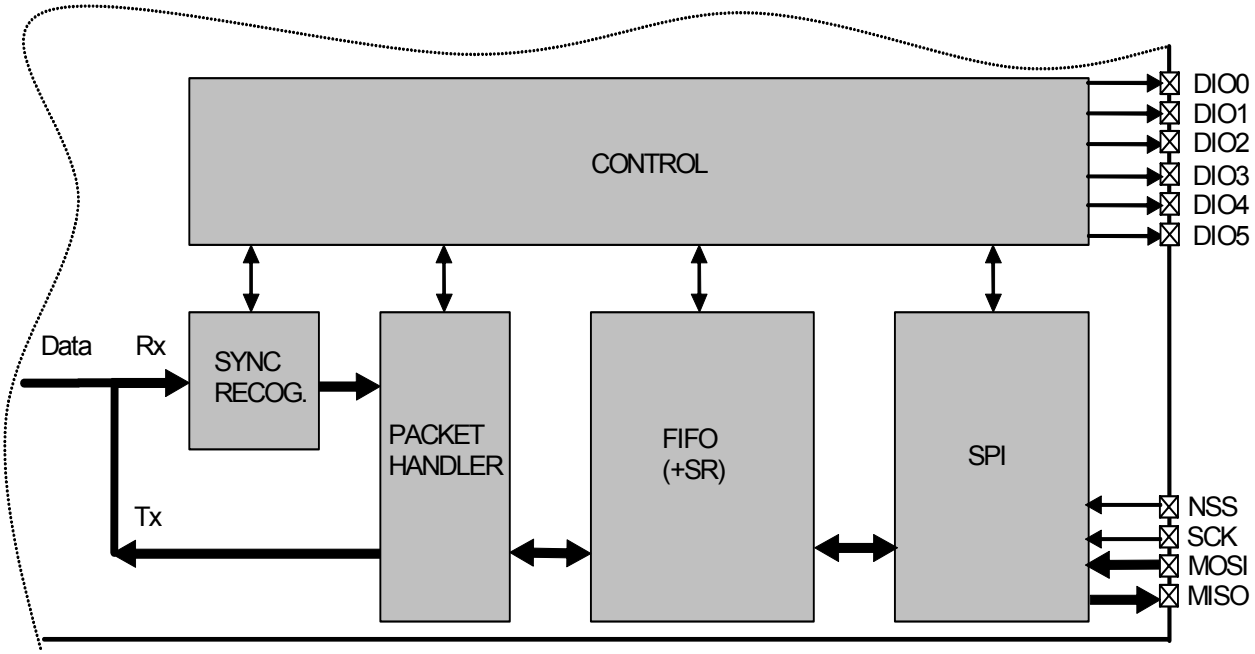


Figure 33. Packet Mode Conceptual View

Note The Bit Synchronizer is automatically enabled in Packet mode.

4.2.14.2. Packet Format

Fixed Length Packet Format

Fixed length packet format is selected when bit *PacketFormat* is set to 0 and *PayloadLength* is set to any value greater than 0.

In applications where the packet length is fixed in advance, this mode of operation may be of interest to minimize RF overhead (no length byte field is required). All nodes, whether Tx only, Rx only or Tx/Rx should be programmed with the same packet length value.

The length of the payload is limited to 2047 bytes.

The length programmed in *PayloadLength* relates only to the payload which includes the message and the optional address byte. In this mode the payload must contain at least one byte i.e. address or message byte.

An illustration of a fixed length packet is shown below. It contains the following fields:

- Preamble (1010...)
- Sync word (Network ID)
- Optional Address byte (Node ID)
- Message data
- Optional 2-bytes CRC checksum

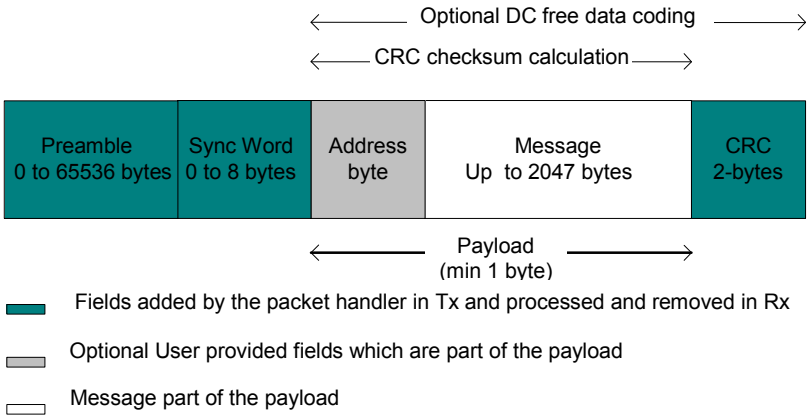


Figure 34. Fixed Length Packet Format

Variable Length Packet Format

Variable length packet format is selected when bit *PacketFormat* is set to 1.

This mode is useful in applications where the length of the packet is not known in advance and can vary over time. It is then necessary for the transmitter to send the length information together with each packet in order for the receiver to operate properly.

In this mode the length of the payload, indicated by the length byte, is given by the first byte of the FIFO and is limited to 255 bytes. Note that the length byte itself is not included in its calculation. In this mode the payload must contain at least 2 bytes i.e. length + address or message byte.

An illustration of a variable length packet is shown below. It contains the following fields:

- Preamble (1010...)
- Sync word (Network ID)
- Length byte
- Optional Address byte (Node ID)
- Message data

- Optional 2-bytes CRC checksum

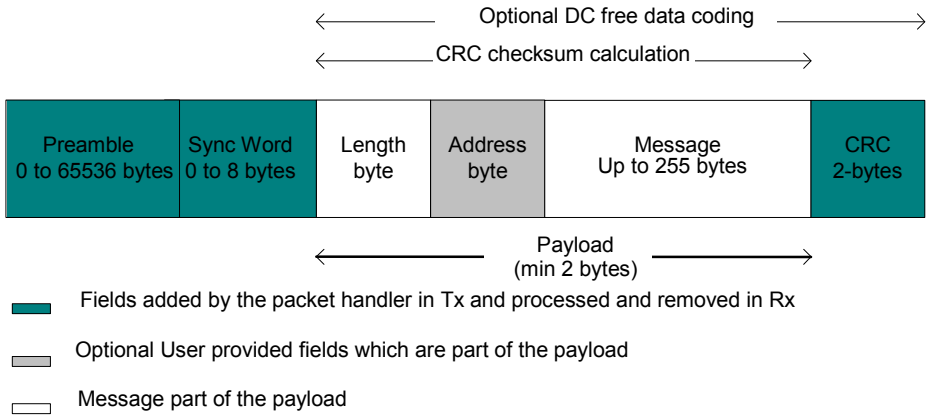


Figure 35. Variable Length Packet Format

Unlimited Length Packet Format

Unlimited length packet format is selected when bit *PacketFormat* is set to 0 and *PayloadLength* is set to 0. The user can then transmit and receive packet of arbitrary length and *PayloadLength* register is not used in Tx/Rx modes for counting the length of the bytes transmitted/received.

In Tx the data is transmitted depending on the *TxStartCondition* bit. On the Rx side the data processing features like Address filtering, Manchester encoding and data whitening are not available if the sync pattern length is set to zero (*SyncOn = 0*). The filling of the FIFO in this case can be controlled by the bit *FifoFillCondition*. The CRC detection in Rx is also not supported in this mode of the packet handler, however CRC generation in Tx is operational. The interrupts like *CrcOk* & *PayloadReady* are not available either.

An unlimited length packet shown below is made up of the following fields:

- Preamble (1010...).
- Sync word (Network ID).
- Optional Address byte (Node ID).
- Message data
- Optional 2-bytes CRC checksum (Tx only)

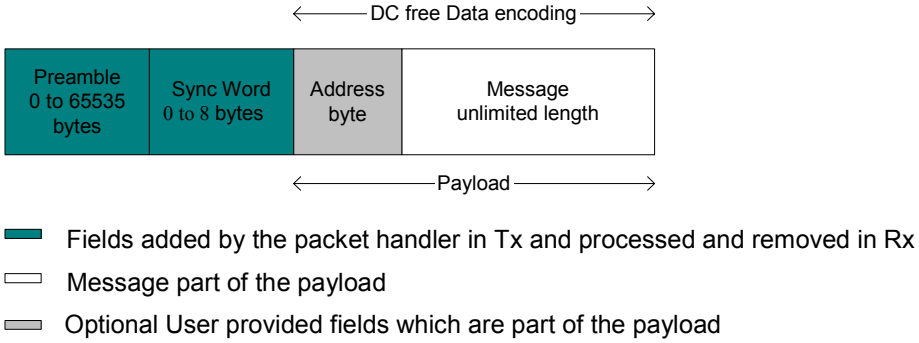


Figure 36. Unlimited Length Packet Format

4.2.14.3. Tx Processing

In Tx mode the packet handler dynamically builds the packet by performing the following operations on the payload available in the FIFO:

- Add a programmable number of preamble bytes.
- Add a programmable Sync word.
- Optionally calculating CRC over complete payload field (optional length byte + optional address byte + message) and appending the 2 bytes checksum.
- Optional DC-free encoding of the data (Manchester or whitening).

Only the payload (including optional address and length fields) is required to be provided by the user in the FIFO.

The transmission of packet data is initiated by the Packet Handler only if the chip is in Tx mode and the transmission condition defined by *TxStartCondition* is fulfilled. If transmission condition is not fulfilled then the packet handler transmits a preamble sequence until the condition is met. This happens only if the preamble length /= 0, otherwise it transmits a zero or one until the condition is met to transmit the packet data.

The transmission condition itself is defined as:

- if *TxStartCondition* = 1, the packet handler waits until the first byte is written into the FIFO, then it starts sending the preamble followed by the sync word and user payload
- If *TxStartCondition* = 0, the packet handler waits until the number of bytes written in the FIFO is equal to the number defined in *RegFifoThresh* + 1
- If the condition for transmission was already fulfilled i.e. the FIFO was filled in Sleep/Stdby then the transmission of packet starts immediately on enabling Tx

4.2.14.4. Rx Processing

In Rx mode the packet handler extracts the user payload to the FIFO by performing the following operations:

- Receiving the preamble and stripping it off.
- Detecting the Sync word and stripping it off.
- Optional DC-free decoding of data.
- Optionally checking the address byte.
- Optionally checking CRC and reflecting the result on *CrcOk*..

Only the payload (including optional address and length fields) is made available in the FIFO.

When the Rx mode is enabled the demodulator receives the preamble followed by the detection of sync word. If fixed length packet format is enabled then the number of bytes received as the payload is given by the *PayloadLength* parameter.

In variable length mode the first byte received after the sync word is interpreted as the length of the received packet. The internal length counter is initialized to this received length. The *PayloadLength* register is set to a value which is greater than the maximum expected length of the received packet. If the received length is greater than the maximum length stored in *PayloadLength* register the packet is discarded otherwise the complete packet is received.

If the address check is enabled then the second byte received in case of variable length and first byte in case of fixed length is the address byte. If the address matches to the one in the *NodeAddress* field reception of the data continues otherwise it's stopped. The CRC check is performed if *CrcOn* = 1 and the result is available in *CrcOk* indicating that the

CRC was successful. An interrupt (*PayloadReady*) is also generated on DIO0 as soon as the payload is available in the FIFO. The payload available in the FIFO can also be read in Sleep/Standby mode.

If the CRC fails the *PayloadReady* interrupt is not generated and the FIFO is cleared. This function can be overridden by setting *CrcAutoClearOff* = 1, forcing the availability of *PayloadReady* interrupt and the payload in the FIFO even if the CRC fails.

4.2.14.5. Handling Large Packets

When *PayloadLength* exceeds FIFO size (64 bytes) whether in fixed, variable or unlimited length packet format, in addition to *PacketSent* in Tx and *PayloadReady* or *CrcOk* in Rx, the FIFO interrupts/flags can be used as described below:

- For Tx:

FIFO can be prefilled in Sleep/Standby but must be refilled “on-the-fly” during Tx with the rest of the payload.

1) Pre-fill FIFO (in Sleep/Standby first or directly in Tx mode) until *FifoThreshold* or *FifoFull* is set

2) In Tx, wait for *FifoThreshold* or *FifoEmpty* to be set (i.e. FIFO is nearly empty)

3) Write bytes into the FIFO until *FifoThreshold* or *FifoFull* is set.

4) Continue to step 2 until the entire message has been written to the FIFO (*PacketSent* will fire when the last bit of the packet has been sent).

- For Rx:

FIFO must be emptied “on-the-fly” during Rx to prevent FIFO overrun.

1) Start reading bytes from the FIFO when *FifoEmpty* is cleared or *FifoThreshold* becomes set.

2) Suspend reading from the FIFO if *FifoEmpty* fires before all bytes of the message have been read

3) Continue to step 1 until *PayloadReady* or *CrcOk* fires

4) Read all remaining bytes from the FIFO either in Rx or Sleep/Standby mode

4.2.14.6. Packet Filtering

The SX1272/73 packet handler offers several mechanisms for packet filtering, ensuring that only useful packets are made available to the uC, significantly reducing system power consumption and software complexity.

Sync Word Based

Sync word filtering/recognition is used for identifying the start of the payload and also for network identification. As previously described, the Sync word recognition block is configured (size, value) in *RegSyncConfig* and *RegSyncValue(i)* registers. This information is used both for appending Sync word in Tx and filtering packets in Rx.

Every received packet which does not start with this locally configured Sync word is automatically discarded and no interrupt is generated.

When the Sync word is detected payload reception automatically starts and *SyncAddressMatch* is asserted.

Note Sync Word values containing 0x00 are forbidden.

Address Based

Address filtering can be enabled via the *AddressFiltering* bits. It adds another level of filtering above Sync word (i.e. Sync must match first) and is typically useful in a multi-node networks where a network ID is shared between all nodes (Sync word) and each node has its own ID (address).

Two address based filtering options are available:

- *AddressFiltering* = *01*: Received address field is compared with internal register *NodeAddress*. If they match then the packet is accepted and processed, otherwise it is discarded.
- *AddressFiltering* = *10*: Received address field is compared with internal registers *NodeAddress* and *BroadcastAddress*. If either is a match, the received packet is accepted and processed, otherwise it is discarded. This additional check with a constant is useful for implementing broadcast in a multi-node networks

Please note that the received address byte, as part of the payload, is not stripped off the packet and is made available in the FIFO. In addition, *NodeAddress* and *AddressFiltering* only apply to Rx. On Tx side, if address filtering is expected, the address byte should simply be put into the FIFO like any other byte of the payload.

As address filtering requires a Sync word match hence both features share the same interrupt flag *SyncAddressMatch*.

Length Based

In variable length Packet mode, *PayloadLength* must be programmed with the maximum payload length permitted. If received length byte is smaller than this maximum then the packet is accepted and processed, otherwise it is discarded.

Please note that the received length byte, as part of the payload, is not stripped off the packet and is made available in the FIFO.

To disable this function the user should set the value of the *PayloadLength* to 2047.

CRC Based

The CRC check is enabled by setting bit *CrcOn* in *RegPacketConfig1*. It is used for checking the integrity of the message.

- On Tx side a two byte CRC checksum is calculated on the payload part of the packet and appended to the end of the message
- On Rx side the checksum is calculated on the received payload and compared with the two checksum bytes received. The result of the comparison is stored in bit *CrcOk.*

By default, if the CRC check fails then the FIFO is automatically cleared and no interrupt is generated. This filtering function can be disabled via *CrcAutoClearOff* bit and in this case, even if CRC fails, the FIFO is not cleared and only *PayloadReady* interrupt goes high. Please note that in both cases, the two CRC checksum bytes are stripped off by the packet handler and only the payload is made available in the FIFO. Two CRC implementations are selected with bit *CrcWhiteningType*.

Table 30 CRC Description

Crc Type	*CrcWhiteningType*	Polynomial	Seed Value	Complemented
CCITT	0 (default)	$X^{16} + X^{12} + X^5 + 1$	0x1D0F	Yes
IBM	1	$X^{16} + X^{15} + X^2 + 1$	0xFFFF	No

A C code implementation of each CRC type is proposed in Application Section 7.

4.2.14.7. DC-Free Data Mechanisms

The payload to be transmitted may contain long sequences of 1's and 0's, which introduces a DC bias in the transmitted signal. The radio signal thus produced has a non uniform power distribution over the occupied channel bandwidth. It also introduces data dependencies in the normal operation of the demodulator. Thus it is useful if the transmitted data is random and DC free.

For such purposes, two techniques are made available in the packet handler: Manchester encoding and data whitening.

Note Only one of the two methods can be enabled at a time.

Manchester Encoding

Manchester encoding/decoding is enabled if *DcFree = 01* and can only be used in Packet mode.

The NRZ data is converted to Manchester code by coding '1' as “10” and '0' as “01”.

In this case, the maximum chip rate is the maximum bit rate given in the specifications section and the actual bit rate is half the chip rate.

Manchester encoding and decoding is only applied to the payload and CRC checksum while preamble and Sync word are kept NRZ. However, the chip rate from preamble to CRC is the same and defined by *BitRate* in *RegBitRate* (Chip Rate = Bit Rate NRZ = 2 x Bit Rate Manchester).

Manchester encoding/decoding is thus transparent with NRZ transferred between FIFO and MCU.

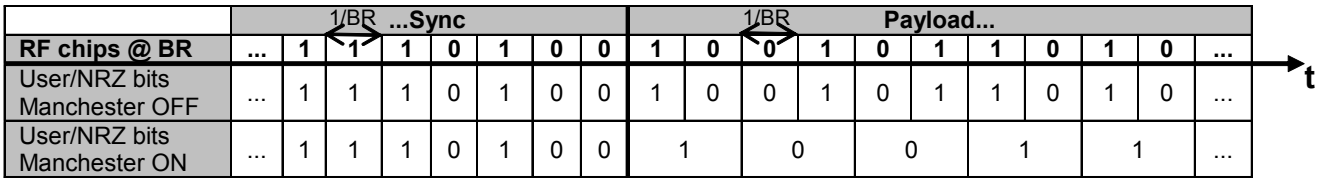


Figure 37. Manchester Encoding/Decoding

Data Whitening

Another technique called whitening or scrambling is widely used for randomizing the user data before radio transmission. The data is whitened using a random sequence on the Tx side and de-whitened on the Rx side using the same sequence. Compared to Manchester coding, whitening has the added advantage that the NRZ data rate is retained i.e. the effective actual bit rate is not halved.

The whitening/de-whitening process is enabled if *DcFree = 10*. A 9-bit LFSR is used to generate a random sequence. The payload and 2-byte CRC checksum is then XORed with this random sequence as shown below. The data is de-whitened on the receiver side by XORing with the same random sequence.

Payload whitening/de-whitening is thus made transparent to the user who still provides/retrieves NRZ data to/from the FIFO.

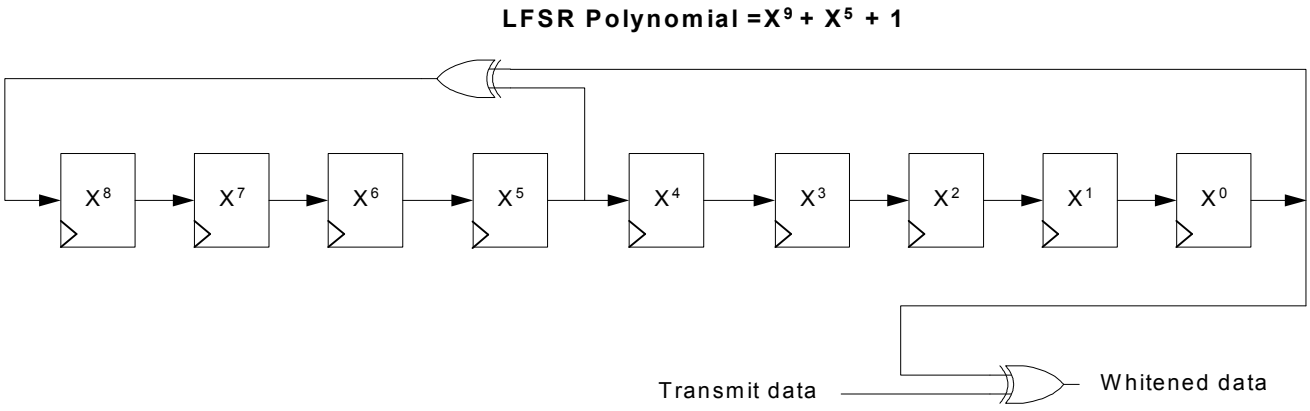


Figure 38. Data Whitening Polynomial

4.2.14.8. Beacon Tx Mode

In some short range wireless network topologies a repetitive message, also known as beacon, is transmitted periodically by a transmitter. The Beacon Tx mode allows for the re-transmission of the same packet without having to fill the FIFO multiple times with the same data.

When *BeaconOn* in *RegPacketConfig2* is set to 1 the FIFO can be filled only once in Sleep or Stdby mode with the required payload. After a first transmission, *FifoEmpty* will go high as usual, but the FIFO content will be restored when the chip exits Transmit mode. *FifoEmpty*, *FifoFull* and *FifoLevel* flags are also restored.

This feature is only available in Fixed packet format with the Payload Length smaller than the FIFO size. The control of the chip modes (Tx-Sleep-Tx....) can either be undertaken by the microcontroller, or be automated in the Top Sequencer. See example in section 4.2.14.8.

The Beacon Tx mode is exited by setting *BeaconOn* to 0 and clearing the FIFO by setting *FifoOverrun* to 1.

4.2.15. io-homecontrol® Compatibility Mode

The SX1272/73 features a io-homecontrol® compatibility mode. Please contact your local Semtech representative for details on its implementation.

4.3. SPI Interface

The SPI interface gives access to the configuration register via a synchronous full-duplex protocol corresponding to CPOL = 0 and CPHA = 0 in Motorola/Freescale nomenclature. Only the slave side is implemented.

Three access modes to the registers are provided:

- SINGLE access: an address byte followed by a data byte is sent for a write access whereas an address byte is sent and a read byte is received for the read access. The NSS pin goes low at the beginning of the frame and goes high after the data byte.
- BURST access: the address byte is followed by several data bytes. The address is automatically incremented internally between each data byte. This mode is available for both read and write accesses. The NSS pin goes low at the beginning of the frame and stay low between each byte. It goes high only after the last byte transfer.
- FIFO access: if the address byte corresponds to the address of the FIFO, then succeeding data byte will address the FIFO. The address is not automatically incremented but is memorized and does not need to be sent between each data byte. The NSS pin goes low at the beginning of the frame and stay low between each byte. It goes high only after the last byte transfer.

The figure below shows a typical SPI single access to a register.

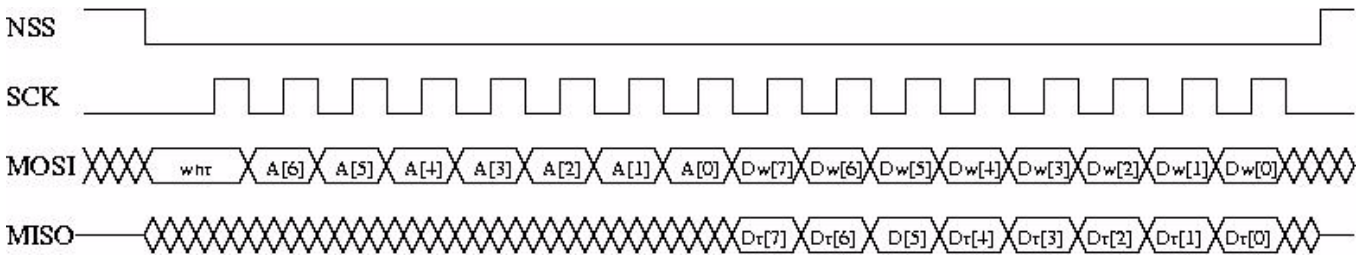


Figure 39. SPI Timing Diagram (single access)

MOSI is generated by the master on the falling edge of SCK and is sampled by the slave (i.e. this SPI interface) on the rising edge of SCK. MISO is generated by the slave on the falling edge of SCK.

A transfer is always started by the NSS pin going low. MISO is high impedance when NSS is high.

The first byte is the address byte. It is comprises:

- A wnr bit, which is 1 for write access and 0 for read access.
- Then 7 bits of address, MSB first.

The second byte is a data byte, either sent on MOSI by the master in case of a write access or received by the master on MISO in case of read access. The data byte is transmitted MSB first.

Proceeding bytes may be sent on MOSI (for write access) or received on MISO (for read access) without a rising NSS edge and re-sending the address. In FIFO mode, if the address was the FIFO address then the bytes will be written / read at the FIFO address. In Burst mode, if the address was not the FIFO address, then it is automatically incremented for each new byte received.

The frame ends when NSS goes high. The next frame must start with an address byte. The SINGLE access mode is therefore a special case of FIFO / BURST mode with only 1 data byte transferred.

During the write access the byte transferred from the slave to the master on the MISO line is the value of the written register before the write operation.

5. SX1272/73 Analog & RF Frontend Electronics

5.1. Power Supply Strategy

The SX1272/73 employs an internal voltage regulation scheme which provides stable operating voltage, and hence device characteristics, over the full industrial temperature and operating voltage range. This includes up to +17 dBm of RF output power which is maintained from 1.8 V to 3.7 V and +20 dBm from 2.4 V to 3.7 V.

The SX1272/73 can be powered from any low-noise voltage source via pins VBAT1 and VBAT2. Decoupling capacitors should be connected, as suggested in the reference design of the applications section of this document, on VR_PA, VR_DIG and VR_ANA pins to ensure correct operation of the built-in voltage regulators.

5.2. Low Battery Detector

A low battery detector is also included allowing the generation of an interrupt signal in response to the supply voltage dropping below a programmable threshold that is adjustable through the register *RegLowBat*. The interrupt signal can be mapped to any of the DIO pins by programming *RegDioMapping*.

5.3. Frequency Synthesis

5.3.1. Crystal Oscillator

The crystal oscillator is the main timing reference of the SX1272/73. It is used as the reference for the PLL's frequency synthesis and as the clock signal for all digital processing.

The crystal oscillator startup time, TS_OSC, depends on the electrical characteristics of the crystal reference used, for more information on the electrical specification of the crystal see section 7.1. The crystal connects to the Pierce oscillator of pins XTA and XTB. The SX1272/73 optimizes the startup time and automatically triggers the PLL when the oscillator signal is stable.

Optionally, an external clock can be used to replace the crystal oscillator. This typically takes the form of a tight tolerance temperature compensated crystal oscillator (TCXO). When using an external clock source the bit *TcxoInputOn* of register *RegTcxo* should be set to 1 and the external clock has to be provided on XTA (pin 4). XTB (pin 5) should be left open.

The peak-peak amplitude of the input signal must never exceed 1.8 V. Please consult your TCXO supplier for an appropriate value of decoupling capacitor, C_D.

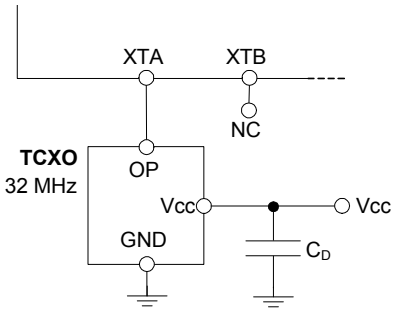


Figure 40. TCXO Connection

5.3.2. CLKOUT Output

The reference frequency, or a fraction of it, can be provided on DIO5 (pin 12) by modifying bits *ClkOut* in *RegDioMapping2*. Two typical applications of the CLKOUT output include:

- To provide a clock output for a companion processor, thus saving the cost of an additional oscillator. CLKOUT can be made available in any operation mode except Sleep mode and is automatically enabled at power on reset.
- To provide an oscillator reference output. Measurement of the CLKOUT signal enables simple software trimming of the initial crystal tolerance.

Note *To minimize the current consumption of the SX1272/73, please ensure that the CLKOUT signal is disabled when not required.*

5.3.3. PLL

The local oscillator of the SX1272/73 is derived from a fractional-N PLL that is referenced to the crystal oscillator circuit. Two PLLs are available for transmit mode operation - either low phase noise or low current consumption to maximize either transmit power consumption or transmit spectral purity respectively. Both PLLs feature a programmable bandwidth setting where one of four discrete preset bandwidths may be accessed. For reference the relative performance of both low consumption and low phase noise PLLs, for each programmable bandwidth setting, is shown in the following figure.

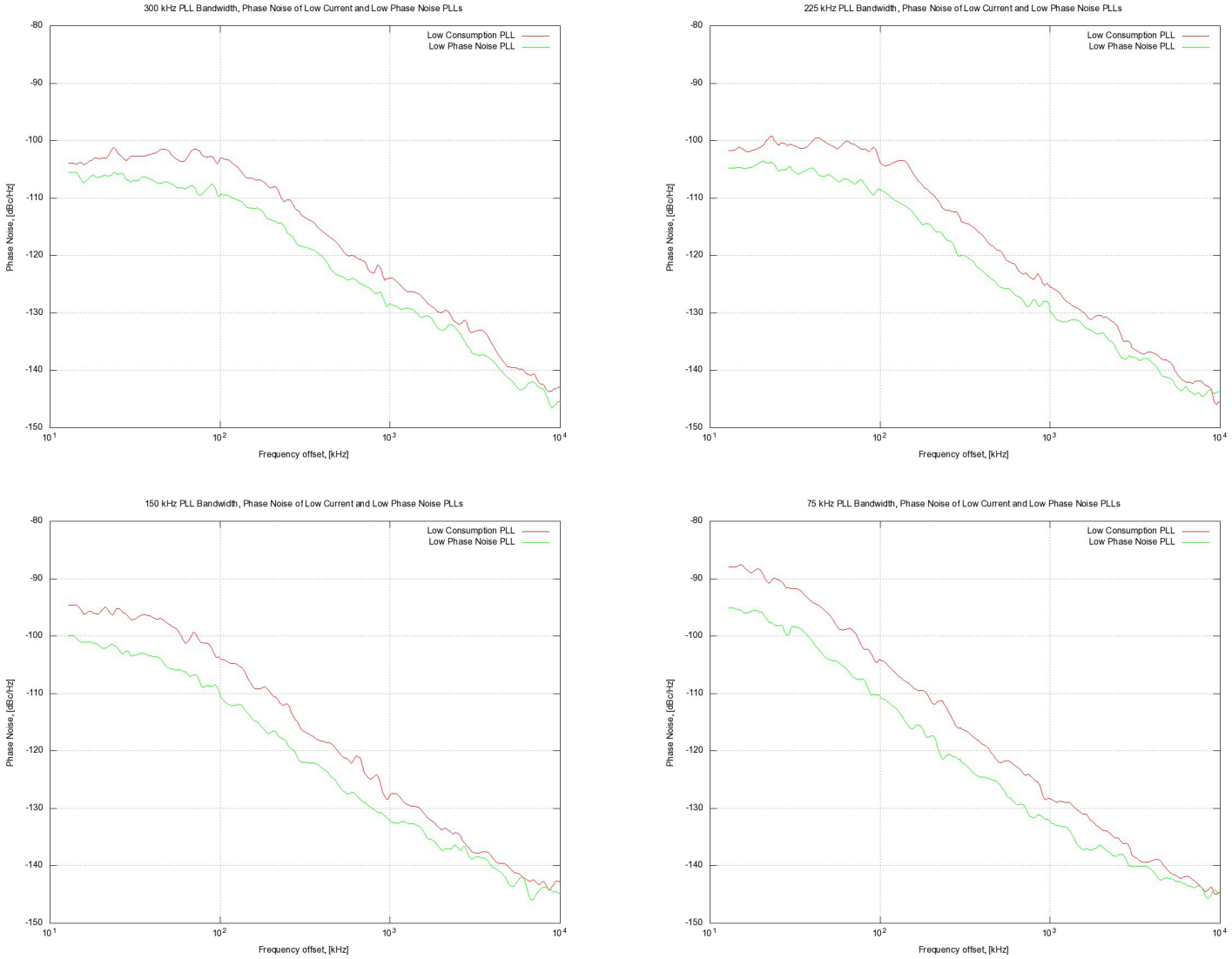


Figure 41. Typical Phase Noise Performances of the Low Consumption and Low Phase Noise PLLs.

Note In receive mode only the low consumption PLL is available.

The SX1272/73 PLL uses a 19-bit sigma-delta modulator whose frequency resolution, constant over the whole frequency range, is given by:

$$F_{STEP} = \frac{F_{XOSC}}{2^{19}}$$

The carrier frequency is programmed through *RegFrf*, split across addresses 0x06 to 0x08:

$$F_{RF} = F_{STEP} \times Frf(23,0)$$

Note *The Frf setting is split across 3 bytes. A change in the center frequency will only be taken into account when the least significant byte FrfLsb in RegFrfLsb is written. This allows the potential for user generation of m-ary FSK at very low bit rates. This is possible where frequency modulation is achieved by direct programming of the programmed RF centre frequency. To enable this functionality set the FastHopOn bit of register RegPllHop.*

5.3.4. RC Oscillator

All timing operations in the low-power Sleep state of the Top Level Sequencer rely on the accuracy of the internal low-power RC oscillator. This oscillator is automatically calibrated at the device power-up not requiring any user input.

5.4. Transmitter Description

The transmitter of SX1272/73 comprises the frequency synthesizer, modulator (both LoRa™ and FSK/OOK) and power amplifier blocks, together with the DC biasing and ramping functionality that is provided through the VR_PA block.

5.4.1. Architecture Description

The architecture of the RF front end is shown in the following diagram. Here we see that the unregulated PA0 is connected to the RFO pin features a single low power amplifier device. The PA_BOOST pin is connected to the internally regulated PA1 and PA2 circuits. Here PA2 is a high power amplifier that permits continuous operation up to +17 dBm and duty cycled operation up to +20 dBm. For full details of operation at +20 dBm please consult Section 5.4.3.

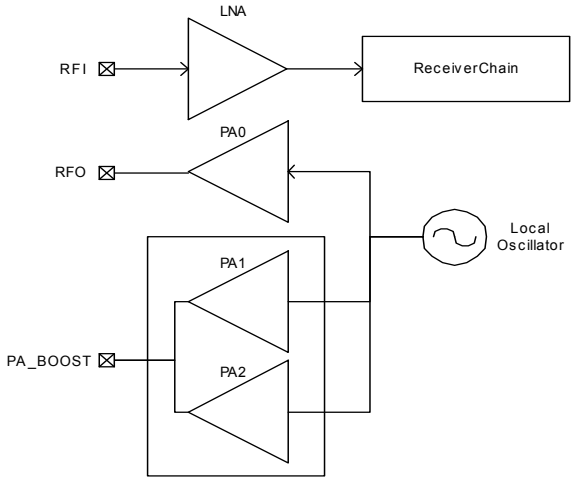


Figure 42. RF Front-end Architecture Shows the Internal PA Configuration.

5.4.2. RF Power Amplifiers

Three power amplifier blocks, PA0 - PA2, are available in the SX1272/73. PA0 is a high efficiency amplifier capable of yielding RF power programmable in 1 dB steps from -1 dBm to +14 dBm directly into a 50 ohm load with low current consumption. PA0 is connected to pin RFO (pin 24).

PA1 and PA2 are both connected to pin PA_BOOST (pin 27). There are two potential configurations of these power amplifiers, fixed or programmable. In the fixed configuration they can deliver up to +20 dBm. In programmable configuration they can provide from +17 dBm to +2 dBm in 1 dB programmable steps. Naturally, low impedance matching and harmonic filtering is required to ensure RF power delivery and regulatory compliance. (See the applications section of this document for more details).

Table 31 Power Amplifier Mode Selection Truth Table

PaSelect	Mode	Power Range	Pout Formula
0	PA0 output on pin RFO	-1 to +14 dBm	-1 dBm + *OutputPower*
1	PA1 and PA2 combined on pin PA_-BOOST	+2 to +17 dBm	+2 dBm + *OutputPower*
1	PA1+PA2 on PA_BOOST with high output power +20 dBm settings (see 5.4.3)	+5 to +20 dBm	+5 dBm + *OutputPower*

Notes - For +20 dBm restrictions on operation please consult the following section.

- To ensure correct operation at the highest power levels ensure that the current limiter OcpTrim is adjusted to permit delivery of the requisite supply current.

- If the PA_BOOST pin is not used it may be left floating.

5.4.3. High Power +20 dBm Operation

The SX1272/73 has a high power +20 dBm capability on PA_BOOST pin, with the following settings:

Table 32 High Power Settings

Register	*Address*	*Value for High Power*	*Default value PA0 or +17dBm*	Description
RegPaDac	0x5A	0x87	0x84	High power PA control

Notes - High Power settings must be turned off when using PA0

- The Over Current Protection limit should be adapted to the actual power level, in RegOcp

Specific Absolute Maximum Ratings and Operating Range restrictions apply to the +20 dBm operation. They are listed in Table 33 and Table 34.

Table 33 Operating Range, +20 dBm Operation

Symbol	Description	Min	Max	Unit
DC_20dBm	Duty Cycle of transmission at +20 dBm output	-	1	%
VSWR_20dBm	Maximum VSWR at antenna port, +20 dBm output	-	3:1	-

Table 34 Operating Range, +20 dBm Operation

Symbol	Description	Min	Max	Unit
VDDop_20dBm	Supply voltage, +20 dBm output	2.4	3.7	V

The duty cycle of transmission at +20 dBm is limited to 1%, with a maximum VSWR of 3:1 at antenna port, over the standard operating range (-40 to +85 °C). For any other operating conditions, contact your Semtech representative.

5.4.4. Over Current Protection

The power amplifiers of SX1272/73 are protected against current over supply in adverse RF load conditions by the over current protection block. This has the added benefit of protecting battery chemistries with limited peak current capability and minimising worst case PA consumption in battery life calculations. The current limiter value is controlled by the *OcpTrim* bits in *RegOcp* and is calculated according to the following formulas:

Table 35 Trimming of the OCP Current

OcpTrim	I_{MAX}	Imax Formula
0 to 15	45 to 120 mA	45 + 5**OcpTrim* [mA]
16 to 27	130 to 240 mA	-30 + 10**OcpTrim* [mA]
27+	240 mA	*240* [mA]

Note Imax sets a limit on the current drain of the Power Amplifier only, hence the maximum current drain of the SX1272/73 *is equal to Imax + IDDFS*.

5.5. Receiver Description

5.5.1. Overview

The SX1272/73 features a digital receiver with the analog to digital conversion process performed directly following the LNA-Mixer block. In addition to the LoRa™ modulation scheme the low-IF receiver is able to demodulate ASK, OOK, (G)FSK and (G)MSK modulation. All filtering, demodulation, gain control, synchronization and packet handling is performed digitally allowing a high degree of programmable flexibility. The receiver also has automatic gain calibration, this improves the precision of RSSI measurement and enhances image rejection.

5.5.2. Receiver Enabled and Receiver Active States

In the receiver operating mode two states of functionality are defined. Upon initial transition to receiver operating mode the receiver is in the ‘receiver-enabled’ state. In this state the receiver awaits for either the user defined valid preamble or RSSI detection criterion to be fulfilled. Once met the receiver enters ‘receiver-active’ state. In this second state the received signal is processed by the packet engine and top level sequencer. For a complete description of the digital functions of the SX1272/73 receiver please see Section 5.5 of the datasheet.

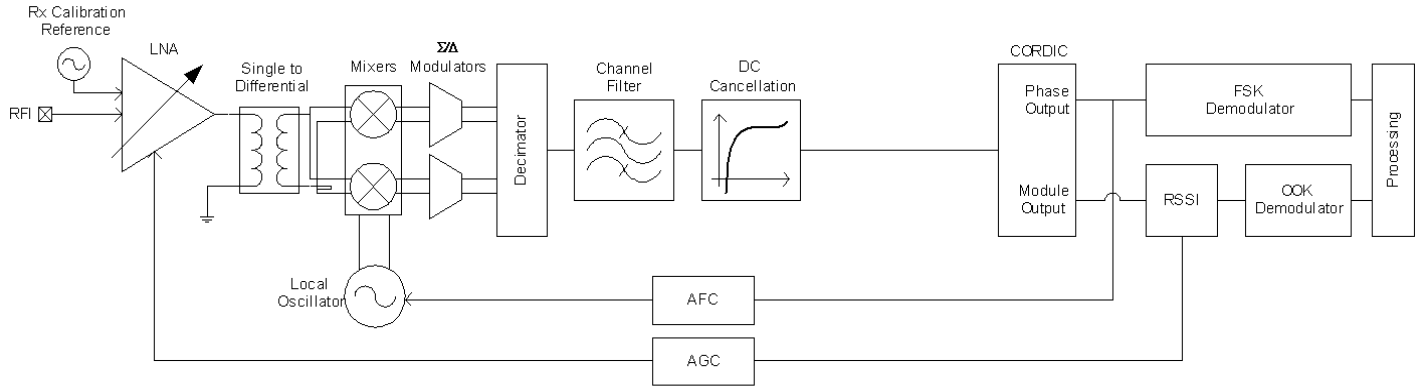


Figure 43. Receiver Block Diagram

5.5.3. Automatic Gain Control In FSK/OOK Mode

The AGC feature allows receiver to handle a wide Rx input dynamic range from the sensitivity level up to maximum input level of 0 dBm or more, whilst optimizing the system linearity.

The following table shows typical NF and IIP3 performances for the SX1272/73 LNA gains available.

Table 36 LNA Gain Control and Performances

RX input level (Pin)	*Gain Setting*	*LnaGain*	*Relative LNA Gain [dB]*	NF [dB]	IIP3 [dBm]
Pin <= AgcThresh1	G1	'001'	0 dB	7	-12
AgcThresh1 < Pin <= AgcThresh2	G2	'010'	-6 dB	11	-8
AgcThresh2 < Pin <= AgcThresh3	G3	'011'	-12 dB	16	-5
AgcThresh3 < Pin <= AgcThresh4	G4	'100'	-24 dB	26	5
AgcThresh4 < Pin <= AgcThresh5	G5	'110'	-26 dB	34	10
AgcThresh5 < Pin	G6	'111'	-48 dB	44	10

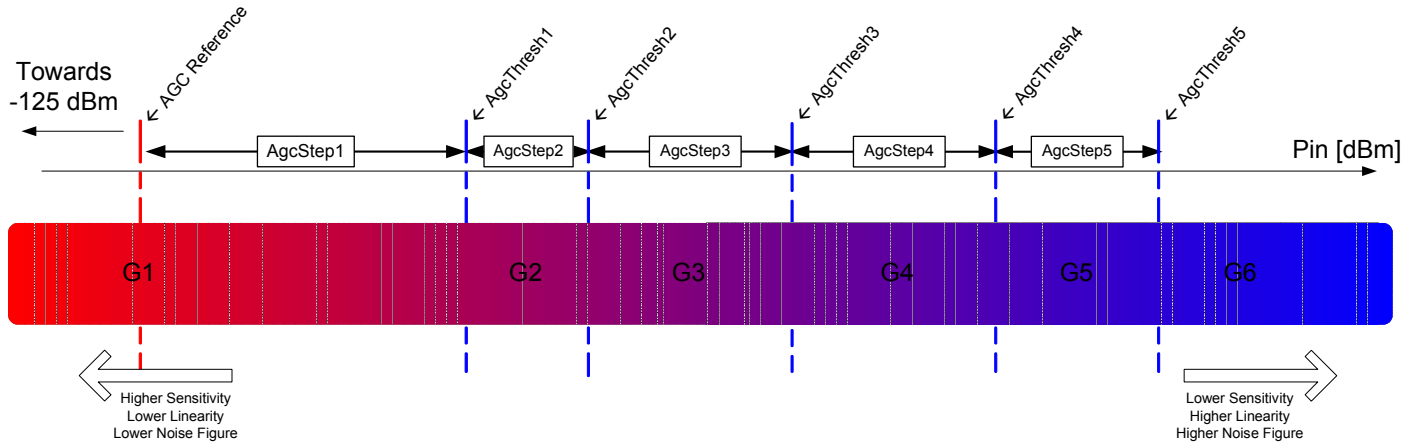


Figure 44. AGC Steps Definition

The AGC reference power level is determined as follows:

AGC Reference [dBm]=-174 dBm + 10 * log(2 * *RxBw*) + SNR + *AgcReferenceLevel*

with SNR = 8 dB (considered a fixed value).

A detailed description of the receiver setup to enable the AGC is provided in section 4.2.7.

5.5.4. RSSI in FSK/OOK Mode

The RSSI provides a measure of the incoming signal power at RF input port measured within the receiver bandwidth. The signal power is available in *RssiValue*. This value is absolute in units of dBm and with a resolution of 0.5 dB. The formula below relates the register value to the absolute input signal level at the RF input port:

$$RssiValue = -2 \cdot RF\ level\left[dBm\right] + RssiOffset\ \left[dB\right]$$

The RSSI value can be compensated to take into account the loss in the matching network or even the gain of an additional LNA by using *RssiOffset*. The offset can be chosen in 1 dB steps from -16 to +15 dB. When compensation is applied the effective signal strength is read as follows:

$$RSSI\left[dBm\right] = -\frac{RssiValue}{2}$$

The RSSI value is smoothed on a user defined number of measured RSSI samples. The precision of the RSSI value is related to the number of RSSI samples used. *RssiSmoothing* selects the number of RSSI samples from a minimum of 2 samples up to 256 samples in increments of power of 2. Table 37 gives the estimation of the RSSI accuracy for a 10 dB SNR and response time versus the number of RSSI samples programmed in *RssiSmoothing*.

Table 37 RssiSmoothing Options

RssiSmoothing	Number of Samples	Estimated Accuracy	Response Time
'000'	2	± 6 dB	$\frac{2^{(RssiSmoothing+1)}}{4 \cdot RxBw[kHz]}[ms]$
'001'	4	± 5 dB	
'010'	8	± 4 dB	
'011'	16	± 3 dB	
'100'	32	± 2 dB	
'101'	64	± 1.5 dB	
'110'	128	± 1.2 dB	
'111'	256	± 1.1 dB	

The RSSI is calibrated when the image and RSSI calibration process is launched. Please see Section Table 4.2.3.8 for details.

5.5.5. RSSI and SNR in LoRa™ Mode

The RSSI values reported by the LoRa™ modem differ from those expressed by the FSK/OOK modem. The following formula shows the method used to interpret the LoRa™ RSSI values:

RSSI (dBm) = -139 + *Rssi*, (with LnaBoost On)

The same formula can be re-used to evaluate the signal strength of the received packet:

Packet Strength (dBm) = -139 + *PacketRssi * 0.25*, (with LnaBoost On and SNR >= 0)

Due to the nature of the LoRa modulation, it is possible to receive packets below the noise floor. In this situation, the SNR is used in conjunction of the PacketRssi to compute the signal strength of the received packet:

Packet Strength (dBm) = -139 + *PacketRssi* + *PacketSnr * 0.25,* (with LnaBoost On and SNR < 0)

Note:

1. *PacketRssi* (in RegPktRssiValue), is an averaged version of *Rssi* (in RegRssiValue). *Rssi* can be read at any time (during packet reception or not), and should be averaged to give more precise results.

2. The constants, -139, may vary with the front-end setup of the SX1272 (*LnaBoost* On or Off, presence of an external LNA, mismatch at the LNA input…). It is recommended to adjust these values with a single-point calibration procedure to increase RSSI accuracy.

3. As signal strength increases (RSSI>-100dBm), the linearity of *PacketRssi* is not guaranteed and results will diverge from the ideal 1dB/dB curve. When very good RSSI precision is required over the whole dynamic range of the receiver, two options are proposed:

- *Rssi* in RegRssiValue offers better linearity. *Rssi* can be sampled during the reception of the payload (between ValidHeader and RxDone IRQ), and used to extract a more high-signal RSSI measurement. *Rssi* is updated every 1/BW (i.e. 8us in 125kHz mode, 4us in 250kHz, etc…)
- When SNR>=0, the standard formula can be adjusted to correct the slope:

RSSI = -139+16/15 * PacketRssi

5.5.6. Channel Filter

The role of the channel filter is to reject noise and interference outside of the wanted channel. The SX1272/73 channel filtering is implemented with a 16-tap finite impulse response (FIR) filter. Rejection of the filter is high enough that the filter stop-band performance is not the dominant influence on adjacent channel rejection performance. This is instead limited by the SX1272/73 PLL phase noise.

Note *To respect sampling criterion in the decimation chain of the receiver, the communication bit rate cannot be set at a higher than twice the single side receiver bandwidth (BitRate < 2 x RxBw)*

The programmed single side bandwidth *RxBw* of the channel filter is determined by the parameters *RxBwMant* and *RxBwExp* in *RegRxBw:*

$$RxBw = \frac{FXOSC}{RxBwMant \times 2^{RxBwExp+2}}$$

The following channel filter bandwidths are hence accessible in the case of a 32 MHz reference oscillator.

Table 38 Available RxBw Settings

RxBwMant (binary/value)	*RxBwExp* (decimal)	*RxBw (kHz)* FSK / OOK
10b / 24	7	2.6
01b / 20	7	3.1
00b / 16	7	3.9
10b / 24	6	5.2
01b / 20	6	6.3
00b / 16	6	7.8
10b / 24	5	10.4
01b / 20	5	12.5
00b / 16	5	15.6
10b / 24	4	20.8
01b / 20	4	25.0
00b / 16	4	31.3
10b / 24	3	41.7
01b / 20	3	50.0
00b / 16	3	62.5
10b / 24	2	83.3
01b / 20	2	100.0
00b / 16	2	125.0
10b / 24	1	166.7
01b / 20	1	200.0
00b / 16	1	250.0
Other settings		reserved

5.5.7. Temperature Measurement

A stand alone temperature measurement block is used in order to measure the temperature in all mode except Sleep and Standby. It is enabled by default and can be stopped by setting *TempMonitorOff* to 1. The result of the measurement is stored in *TempValue* in *RegTemp*.

Due to process variations the absolute accuracy of the result is +/- 10 °C. Higher precision requires a calibration procedure at a known temperature. The figure below shows the influence of just such a calibration process. For more information, including source code, please consult the applications section of this document.

Example temperature curve, typical device

	Correction Factor		15
Actual Temp [Celsius]	RegTemp [Dec]	Temp before calibration [°C]	Temp after calibration [°C]
85	181	74	89
75	190	65	80
65	201	54	69
55	211	44	59
45	222	33	48
35	232	23	38
25	245	10	25
15	0	0	15
5	10	-10	5
-5	21	-21	-6
-15	33	-33	-18
-25	44	-44	-29
-35	56	-56	-41
-40	63	-63	-48

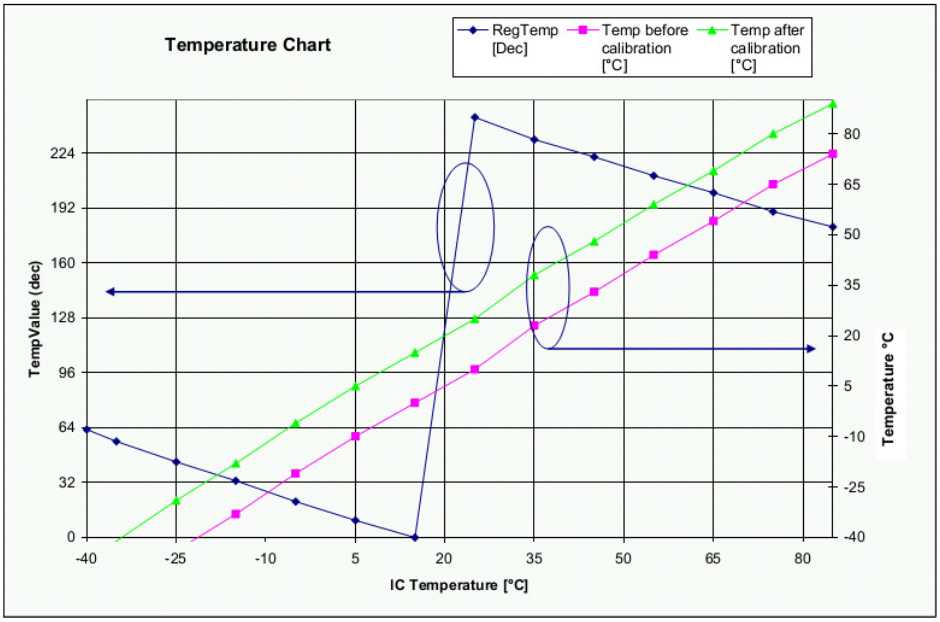


Figure 45. Temperature Sensor Response

6. Description of the Registers

The register mapping depends upon whether FSK/OOK or LoRa™ mode has been selected. The following table summarises the location and function of each register and gives an overview of the changes in register mapping between both modes of operation.

6.1. Register Table Summary

Table 39 Registers Summary

Address	Register Name		Reset (POR)	Default (FSK)	Description	
	FSK/OOK Mode	LoRa™ Mode			FSK Mode	LoRa™ Mode
0x00	RegFifo		0x00		FIFO read/write access	
0x01	RegOpMode		0x01		Operating mode & LoRa™ / FSK selection	
0x02	RegBitrateMsb	Unused	0x1A		Bit Rate setting, Most Significant Bits	
0x03	RegBitrateLsb		0x0B		Bit Rate setting, Least Significant Bits	
0x04	RegFdevMsb		0x00		Frequency Deviation setting, Most Significant Bits	
0x05	RegFdevLsb		0x52		Frequency Deviation setting, Least Significant Bits	
0x06	RegFrfMsb		0xE4		RF Carrier Frequency, Most Significant Bits	
0x07	RegFrfMid		0xC0		RF Carrier Frequency, Intermediate Bits	
0x08	RegFrfLsb		0x00		RF Carrier Frequency, Least Significant Bits	
0x09	RegPaConfig		0x0F		PA selection and Output Power control	
0x0A	RegPaRamp		0x19		Control of PA ramp time, low phase noise PLL	
0x0B	RegOcp		0x2B		Over Current Protection control	
0x0C	RegLna		0x20		LNA settings	
0x0D	RegRxConfig	RegFifoAddrPtr	0x08	0x1E	AFC, AGC, ctrl	FIFO SPI pointer
0x0E	RegRssiConfig	RegFifoTxBaseAddr	0x02		RSSI	Start Tx data
0x0F	RegRssiCollision	RegFifoRxBaseAddr	0x0A		RSSI Collision detector	Start Rx data
0x10	RegRssiThresh	FifoRxCurrentAddr	0xFF		RSSI Threshold control	Start address of last packet received
0x11	RegRssiValue	RegIrqFlagsMask	n/a	n/a	RSSI value in dBm	Optional IRQ flag mask
0x12	RegRxBw	RegIrqFlags	0x15		Channel Filter BW Control	IRQ flags
0x13	RegAfcBw	RegRxNbBytes	0x0B		AFC Channel Filter BW	Number of received bytes
0x14	RegOokPeak	RegRxHeaderCntValueMsb	0x28		OOK demodulator	Number of valid headers received
0x15	RegOokFix	RegRxHeaderCntValueLsb	0x0C		Threshold of the OOK demod	
0x16	RegOokAvg	RegRxPacketCntValueMsb	0x12		Average of the OOK demod	Number of valid packets received
0x17	Reserved17	RegRxPacketCntValueLsb	0x47		-	
0x18	Reserved18	RegModemStat	0x32		-	Live LoRa™ modem status
0x19	Reserved19	RegPktSnrValue	0x3E		-	Espimation of last packet SNR
0x1A	RegAfcFei	RegPktRssiValue	0x00	n/a	AFC and FEI control	RSSI of last packet

Address	Register Name		Reset (POR)	Default (FSK)	Description	
	FSK/OOK Mode	LoRa™ Mode			FSK Mode	LoRa™ Mode
0x1B	RegAfcMsb	RegRssiValue	0x00	n/a	Frequency correction value of the AFC	Current RSSI
0x1C	RegAfcLsb	RegHopChannel	0x00	n/a		FHSS start channel
0x1D	RegFeiMsb	RegModemConfig 1	0x00	n/a	Value of the calculated frequency error	Modem PHY config 1
0x1E	RegFeiLsb	RegModemConfig 2	0x00	n/a		Modem PHY config 2
0x1F	RegPreambleDetect	RegSymbTimeoutLsb	0x40	0xAA	Settings of the Preamble Detector	Receiver timeout value
0x20	RegRxTimeout1	RegPreambleMsb	0x00		Timeout Rx request and RSSI	Size of preamble
0x21	RegRxTimeout2	RegPreambleLsb	0x00		Timeout RSSI and *PayloadReady*	
0x22	RegRxTimeout3	RegPayloadLength	0x00		Timeout RSSI and *SyncAddress*	LoRaTM payload length
0x23	RegRxDelay	RegMaxPayloadLength	0x00		Delay between Rx cycles	LoRaTM maximum payload length
0x24	RegOsc	RegHopPeriod	0x05	0x07	RC Oscillators Settings, CLKOUT frequency	FHSS Hop period
0x25	RegPreambleMsb	RegFifoRxByteAddr	0x00		Preamble length, MSB	Address of last byte written in FIFO
0x26	RegPreambleLsb	RESERVED	0x03		Preamble length, LSB	LoRa™ rx data pointer
0x27	RegSyncConfig		0x93		Sync Word Recognition control	RESERVED
0x28	RegSyncValue1	RegFeiMsb	0x55	0x01	Sync Word bytes, 1 through 8	Estimated frequency error
0x29	RegSyncValue2	RegFeiMib	0x55	0x01		
0x2A	RegSyncValue3	RegFeiLsb	0x55	0x01		
0x2B-0x2F	RegSyncValue4	RESERVED	0x55	0x01		RESERVED
0x2C	RegSyncValue5	RegRssiWideband	0x55	0x01		Wideband RSSI measurement
0x2D-2F	RegSyncValue6-8	RESERVED	0x55	0x01		RESERVED
0x30	RegPacketConfig1		0x90		Packet mode settings	
0x31	RegPacketConfig2	RegDetectOptimize	0x40		Packet mode settings	LoRa detection Optimize for SF6
0x32	RegPayloadLength	RESERVED	0x40		Payload length setting	RESERVED
0x33	RegNodeAdrs	RegInvertIQ	0x00		Node address	Invert LoRa I and Q signals
0x34	RegBroadcastAdrs	RESERVED	0x00		Broadcast address	RESERVED
0x35	RegFifoThresh		0x0F	0x8F	Fifo threshold, Tx start condition	
0x36	RegSeqConfig1		0x00		Top level Sequencer settings	
0x37	RegSeqConfig2	RegDetectionThreshold	0x00		Top level Sequencer settings	Change the LoRa Detection threshold for SF6
0x38	RegTimerResol	RESERVED	0x00		Timer 1 and 2 resolution control	RESERVED

Address	Register Name		Reset (POR)	Default (FSK)	Description	
	FSK/OOK Mode	LoRa™ Mode			FSK Mode	LoRa™ Mode
0x39	RegTimer1Coef	RegSyncWord	0xF5	0x12	Timer 1 setting	LoRa Sync Word
0x3A	RegTimer2Coef	RESERVED	0x20		Timer 2 setting	RESERVED
0x3B	RegImageCal		0x82	0x02	Image calibration engine control	
0x3C	RegTemp		-		Temperature Sensor value	
0x3D	RegLowBat		0x02		Low Battery Indicator Settings	
0x3E	RegIrqFlags1		0x80	n/a	Status register: PLL Lock state, Timeout, RSSI	
0x3F	RegIrqFlags2		0x40	n/a	Status register: FIFO handling flags, Low Battery	
0x40	RegDioMapping1		0x00		Mapping of pins DIO0 to DIO3	
0x41	RegDioMapping2		0x00		Mapping of pins DIO4 and DIO5, ClkOut frequency	
0x42	RegVersion		0x22		Semtech ID relating the silicon revision	
0x43	RegAgcRef		0x13		Adjustment of the AGC thresholds	
0x44	RegAgcThresh1		0x0E			
0x45	RegAgcThresh2		0x5B			
0x46	RegAgcThresh3		0xDB			
0x4B	RegPllHop		0x2E		Control the fast frequency hopping mode	
0x58	RegTcxo		0x09		TCXO or XTAL input setting	
0x5A	RegPaDac		0x84		Higher power settings of the PA	
0x5C	RegPll		0xD0		Control of the PLL bandwidth	
0x5E	RegPllLowPn		0xD0		Control of the Low Phase Noise PLL bandwidth	
0x6C	RegFormerTemp		-		Stored temperature during the former IQ Calibration	
0x70	RegBitRateFrac		0x00		Fractional part in the Bit Rate division ratio	

Notes - Reset values are automatically refreshed in the chip at Power On Reset

- Default values are the Semtech recommended register values, optimizing the device operation

*- Registers for which the Default value differs from the Reset value are denoted by an * in the tables of section 6.2*

6.2. FSK/OOK Mode Register Map

This section details the SX1272/73 register mapping and the precise contents of each register in FSK/OOK mode.

Convention: r: read, w: write, t:trigger, c: clear

Table 40 Register Map

Name (Address)	Bits	Variable Name	Mode	Default value	FSK/OOK Description
RegFifo (0x00)	7-0	Fifo	rw	0x00	FIFO data input/output
Registers for Common settings					
RegOpMode (0x01)	7	LongRangeMode	r	0x00	0 → FSK/OOK Mode 1→ LoRa™ Mode This bit can be modified only in Sleep mode. A write operation on other device modes is ignored.
	6-5	ModulationType	rw	0x00	Modulation scheme: 00 → FSK 01 → OOK 10 -11 → reserved
	4-3	ModulationShaping	rw	0x00	Data shaping: In FSK: 00 → no shaping 01 → Gaussian filter BT = 1.0 10 → Gaussian filter BT = 0.5 11 → Gaussian filter BT = 0.3 In OOK: 00 → no shaping 01 → filtering with f_{cutoff} = bit_rate 10 → filtering with f_{cutoff} = 2*bit_rate (for bit_rate < 125 kbps) 11 → reserved
	2-0	Mode	rw	0x01	Transceiver modes 000 → Sleep mode 001 → Stdby mode 010 → FS mode TX (FSTx) 011 → Transmitter mode (Tx) 100 → FS mode RX (FSRx) 101 → Receiver mode (Rx) 110 → reserved 111 → reserved
RegBitrateMsb (0x02)	7-0	BitRate(15:8)	rw	0x1A	MSB of Bit Rate (chip rate if Manchester encoding is enabled)
RegBitrateLsb (0x03)	7-0	BitRate(7:0)	rw	0x0B	LSB of bit rate (chip rate if Manchester encoding is enabled) $BitRate = \frac{FXOSC}{BitRate(15,0) + \frac{BitrateFrac}{16}}$ Default value: 4.8 kbps

Name (Address)	Bits	Variable Name	Mode	Default value	FSK/OOK Description
RegFdevMsb (0x04)	7-6	unused	r	0x00	unused
	5-0	Fdev(13:8)	rw	0x00	MSB of the frequency deviation
RegFdevLsb (0x05)	7-0	Fdev(7:0)	rw	0x52	LSB of the frequency deviation $Fdev = Fstep \times Fdev(15,0)$ Default value: 5 kHz
RegFrfMsb (0x06)	7-0	Frf(23:16)	rw	0xE4	MSB of the RF carrier frequency
RegFrfMid (0x07)	7-0	Frf(15:8)	rw	0xC0	MSB of the RF carrier frequency
RegFrfLsb (0x08)	7-0	Frf(7:0)	rw	0x00	LSB of RF carrier frequency $Frf = Fstep \times Frf(23;0)$ Default value: 915.000 MHz The RF frequency is taken into account internally only when: - entering FSRX/FSTX modes - re-starting the receiver
Registers for the Transmitter					
RegPaConfig (0x09)	7	PaSelect	rw	0x00	Selects PA output pin 0 → RFO pin. Maximum power of +13 dBm 1 → PA_BOOST pin. Maximum power of +20 dBm
	6-4	unused	r	0x00	unused
	3-0	OutputPower	rw	0x0F	Output power setting, with 1dB steps Pout = 2 + *OutputPower* [dBm], on PA_BOOST pin Pout = -1 + *OutputPower* [dBm], on RFO pin
RegPaRamp (0x0A)	7-5	unused	r	-	unused
	4	LowPnTxPllOff	rw	0x01	Select a higher power, lower phase noise PLL only when the transmitter is used: 0 → Standard PLL used in Rx mode, Lower PN PLL in Tx 1 → Standard PLL used in both Tx and Rx modes
	3-0	PaRamp	rw	0x09	Rise/Fall time of ramp up/down in FSK 0000 → 3.4 ms 0001 → 2 ms 0010 → 1 ms 0011 → 500 us 0100 → 250 us 0101 → 125 us 0110 → 100 us 0111 → 62 us 1000 → 50 us 1001 → 40 us (d) 1010 → 31 us 1011 → 25 us 1100 → 20 us 1101 → 15 us 1110 → 12 us 1111 → 10 us

Name (Address)	Bits	Variable Name	Mode	Default value	FSK/OOK Description
RegOcp (0x0B)	7-6	unused	r	0x00	unused
	5	OcpOn	rw	0x01	Enables overload current protection (OCP) for the PA: 0 → OCP disabled 1 → OCP enabled
	4-0	OcpTrim	rw	0x0B	Trimming of OCP current: I_{max} = 45+5*OcpTrim [mA] if OcpTrim <= 15 (120 mA) / I_{max} = -30+10*OcpTrim [mA] if 15 < OcpTrim <= 27 (130 to 240 mA) I_{max} = 240mA for higher settings Default I_{max} = 100mA
Registers for the Receiver					
RegLna (0x0C)	7-5	LnaGain	rw	0x01	LNA gain setting: 000 → reserved 001 → G1 = highest gain 010 → G2 = highest gain – 6 dB 011 → G3 = highest gain – 12 dB 100 → G4 = highest gain – 24 dB 101 → G5 = highest gain – 36 dB 110 → G6 = highest gain – 48 dB 111 → reserved Note: Reading this address always returns the current LNA gain (which may be different from what had been previously selected if AGC is enabled.
	4-2	-	r	0x00	unused
	1-0	LnaBoost	rw	0x00	Improves the system Noise Figure at the expense of Rx current consumption: 00 → Default setting, meeting the specification 11 → Improved sensitivity
RegRxConfig (0x0D)	7	RestartRxOnCollision	rw	0x00	Turns on the mechanism restarting the receiver automatically if it gets saturated or a packet collision is detected 0 → No automatic Restart 1 → Automatic restart On
	6	RestartRxWithoutPllLock	wt	0x00	Triggers a manual Restart of the Receiver chain when set to 1. Use this bit when there is no frequency change, RestartRxWithPllLock otherwise.
	5	RestartRxWithPllLock	wt	0x00	Triggers a manual Restart of the Receiver chain when set to 1. Use this bit when there is a frequency change, requiring some time for the PLL to re-lock.
	4	AfcAutoOn	rw	0x00	0 → No AFC performed at receiver startup 1 → AFC is performed at each receiver startup
	3	AgcAutoOn	rw	0x01	0 → LNA gain forced by the LnaGain Setting 1 → LNA gain is controlled by the AGC
	2-0	RxTrigger	rw	0x06 *	Selects the event triggering AGC and/or AFC at receiver startup. See Table Table 23 for a description.

Name (Address)	Bits	Variable Name	Mode	Default value	FSK/OOK Description
RegRssiConfig (0x0E)	7-3	RssiOffset	rw	0x00	Signed RSSI offset, to compensate for the possible losses/gains in the front-end (LNA, SAW filter...) 1dB / LSB, 2's complement format
	2-0	RssiSmoothing	rw	0x02	Defines the number of samples taken to average the RSSI result: 000 → 2 samples used 001 → 4 samples used 010 → 8 samples used 011 → 16 samples used 100 → 32 samples used 101 → 64 samples used 110 → 128 samples used 111 → 256 samples used
RegRssiCollision (0x0F)	7-0	RssiCollisionThreshold	rw	0x0A	Sets the threshold used to consider that an interferer is detected, witnessing a packet collision. 1dB/LSB (only RSSI increase) Default: 10dB
RegRssiThresh (0x10)	7-0	RssiThreshold	rw	0xFF	RSSI trigger level for the Rssi interrupt: - RssiThreshold / 2 [dBm]
RegRssiValue (0x11)	7-0	RssiValue	r	-	Absolute value of the RSSI in dBm, 0.5dB steps. RSSI = - RssiValue/2 [dBm]
RegRxBw (0x12)	7	unused	r	-	unused
	6-5	reserved	rw	0x00	reserved
	4-3	RxBwMant	rw	0x02	Channel filter bandwidth control: 00 → RxBwMant = 16 10 → RxBwMant = 24 01 → RxBwMant = 20 11 → reserved
	2-0	RxBwExp	rw	0x05	Channel filter bandwidth control: FSK Mode: $RxBw = \frac{FXOSC}{RxBwMant \times 2^{RxBwExp+2}}$
RegAfcBw (0x13)	7-5	reserved	rw	0x00	reserved
	4-3	RxBwMantAfc	rw	0x01	RxBwMant parameter used during the AFC
	2-0	RxBwExpAfc	rw	0x03	RxBwExp parameter used during the AFC
RegOokPeak (0x14)	7-6	reserved	rw	0x00	reserved
	5	BitSyncOn	rw	0x01	Enables the Bit Synchronizer. 0 → Bit Sync disabled (not possible in Packet mode) 1 → Bit Sync enabled
	4-3	OokThreshType	rw	0x01	Selects the type of threshold in the OOK data slicer: 00 → fixed threshold 10 → average mode 01 → peak mode (default) 11 → reserved
	2-0	OokPeakTheshStep	rw	0x00	Size of each decrement of the RSSI threshold in the OOK demodulator: 000 → 0.5 dB 001 → 1.0 dB 010 → 1.5 dB 011 → 2.0 dB 100 → 3.0 dB 101 → 4.0 dB 110 → 5.0 dB 111 → 6.0 dB

Name (Address)	Bits	Variable Name	Mode	Default value	FSK/OOK Description
RegOokFix (0x15)	7-0	OokFixedThreshold	rw	0x0C	Fixed threshold for the Data Slicer in OOK mode Floor threshold for the Data Slicer in OOK when Peak mode is used
RegOokAvg (0x16)	7-5	OokPeakThreshDec	rw	0x00	Period of decrement of the RSSI threshold in the OOK demodulator: 000 → once per chip 001 → once every 2 chips 010 → once every 4 chips 011 → once every 8 chips 100 → twice in each chip 101 → 4 times in each chip 110 → 8 times in each chip 111 → 16 times in each chip
	4	reserved	rw	0x01	reserved
	3-2	OokAverageOffset	rw	0x00	Static offset added to the threshold in average mode in order to reduce glitching activity (OOK only): 00 → 0.0 dB 10 → 4.0 dB 01 → 2.0 dB 11 → 6.0 dB
	1-0	OokAverageThreshFilt	rw	0x02	Filter coefficients in average mode of the OOK demodulator: 00 → f_C ≈ chip rate / 32.π 01 → f_C ≈ chip rate / 8.π 10 → f_C ≈ chip rate / 4.π 11 →f_C ≈ chip rate / 2.π
RegRes17 to RegRes19	7-0	reserved	rw	0x47 0x32 0x3E	reserved. Keep the Reset values.
RegAfcFei (0x1A)	7-5	unused	r	-	unused
	4	AgcStart	wt	0x00	Triggers an AGC sequence when set to 1.
	3	reserved	rw	0x00	reserved
	2	unused	-	-	unused
	1	AfcClear	wc	0x00	Clear AFC register set in Rx mode. Always reads 0.
	0	AfcAutoClearOn	rw	0x00	Only valid if AfcAutoOn is set 0 → AFC register is not cleared at the beginning of the automatic AFC phase 1 → AFC register is cleared at the beginning of the automatic AFC phase
RegAfcMsb (0x1B)	7-0	AfcValue(15:8)	rw	0x00	MSB of the AfcValue, 2's complement format. Can be used to overwrite the current AFC value
RegAfcLsb (0x1C)	7-0	AfcValue(7:0)	rw	0x00	LSB of the AfcValue, 2's complement format. Can be used to overwrite the current AFC value
RegFeiMsb (0x1D)	7-0	FeiValue(15:8)	rw	-	MSB of the measured frequency offset, 2's complement. Must be read before RegFeiLsb.
RegFeiLsb (0x1E)	7-0	FeiValue(7:0)	rw	-	LSB of the measured frequency offset, 2's complement *Frequency error* = FeiValue x Fstep

Name (Address)	Bits	Variable Name	Mode	Default value	FSK/OOK Description
RegPreambleDetect (0x1F)	7	PreambleDetectorOn	rw	0x01 *	Enables Preamble detector when set to 1. The AGC settings supersede this bit during the startup / AGC phase. 0 → Turned off 1 → Turned on
	6-5	PreambleDetectorSize	rw	0x01 *	Number of Preamble bytes to detect to trigger an interrupt 00 → 1 byte 10 → 3 bytes 01 → 2 bytes 11 → Reserved
	4-0	PreambleDetectorTol	rw	0x0A *	Number or chip errors tolerated over PreambleDetectorSize. 4 chips per bit.
RegRxTimeout1 (0x20)	7-0	TimeoutRxRssi	rw	0x00	*Timeout* interrupt is generated *TimeoutRxRssi**16*T_{bit} after switching to Rx mode if *Rssi* interrupt doesn't occur (i.e. *RssiValue* > *RssiThreshold)* 0x00: *TimeoutRxRssi* is disabled
RegRxTimeout2 (0x21)	7-0	TimeoutRxPreamble	rw	0x00	*Timeout* interrupt is generated *TimeoutRxPreamble**16*T_{bit} after switching to Rx mode if *Preamble* interrupt doesn't occur 0x00: *TimeoutRxPreamble* is disabled
RegRxTimeout3 (0x22)	7-0	TimeoutSignalSync	rw	0x00	*Timeout* interrupt is generated *TimeoutSignalSync**16*T_{bit} after the Rx mode is programmed, if *SyncAddress* doesn't occur 0x00: *TimeoutSignalSync* is disabled
RegRxDelay (0x23)	7-0	InterPacketRxDelay	rw	0x00	Additional delay before an automatic receiver restart is launched: Delay = InterPacketRxDelay*4*Tbit
RC Oscillator registers					
RegOsc (0x24)	7-4	unused	r	-	unused
	3	RcCalStart	wt	0x00	Triggers the calibration of the RC oscillator when set. Always reads 0. RC calibration must be triggered in Standby mode.
	2-0	ClkOut	rw	0x07 *	Selects CLKOUT frequency: 000 → FXOSC 001 → FXOSC / 2 010 → FXOSC / 4 011 → FXOSC / 8 100 → FXOSC / 16 101 → FXOSC / 32 110 → RC (automatically enabled) 111 → OFF
Packet Handling registers					
RegPreambleMsb (0x25)	7-0	PreambleSize(15:8)	rw	0x00	Size of the preamble to be sent (from *TxStartCondition* fulfilled). (MSB byte)
RegPreambleLsb (0x26)	7-0	PreambleSize(7:0)	rw	0x03	Size of the preamble to be sent (from *TxStartCondition* fulfilled). (LSB byte)

Name (Address)	Bits	Variable Name	Mode	Default value	FSK/OOK Description
RegSyncConfig (0x27)	7-6	AutoRestartRxMode	rw	0x02	Controls the automatic restart of the receiver after the reception of a valid packet (PayloadReady or CrcOk): 00 → Off 01 → On, without waiting for the PLL to re-lock 10 → On, wait for the PLL to lock (frequency changed) 11 → reserved
	5	PreamblePolarity	rw	0x00	Sets the polarity of the Preamble 0 → 0xAA (default) 1 → 0x55
	4	SyncOn	rw	0x01	Enables the Sync word generation and detection: 0 → Off 1 → On
	3	FifoFillCondition	rw	0x00	FIFO filling condition: 0 → if *SyncAddress* interrupt occurs 1 → as long as *FifoFillCondition* is set
	2-0	SyncSize	rw	0x03	Size of the Sync word: (*SyncSize* + 1) bytes, (*SyncSize*) bytes if *ioHomeOn*=1
RegSyncValue1 (0x28)	7-0	SyncValue(63:56)	rw	0x01 *	1st byte of Sync word. (MSB byte) Used if *SyncOn* is set.
RegSyncValue2 (0x29)	7-0	SyncValue(55:48)	rw	0x01 *	2nd byte of Sync word Used if *SyncOn* is set and *(SyncSize +1)* >= 2.
RegSyncValue3 (0x2A)	7-0	SyncValue(47:40)	rw	0x01 *	3rd byte of Sync word. Used if *SyncOn* is set and *(SyncSize +1)* >= 3.
RegSyncValue4 (0x2B)	7-0	SyncValue(39:32)	rw	0x01 *	4th byte of Sync word. Used if *SyncOn* is set and *(SyncSize +1)* >= 4.
RegSyncValue5 (0x2C)	7-0	SyncValue(31:24)	rw	0x01 *	5th byte of Sync word. Used if *SyncOn* is set and *(SyncSize +1)* >= 5.
RegSyncValue6 (0x2D)	7-0	SyncValue(23:16)	rw	0x01 *	6th byte of Sync word. Used if *SyncOn* is set and *(SyncSize +1)* >= 6.
RegSyncValue7 (0x2E)	7-0	SyncValue(15:8)	rw	0x01 *	7th byte of Sync word. Used if *SyncOn* is set and *(SyncSize +1)* >= 7.
RegSyncValue8 (0x2F)	7-0	SyncValue(7:0)	rw	0x01 *	8th byte of Sync word. Used if *SyncOn* is set and *(SyncSize +1)* = 8.

Name (Address)	Bits	Variable Name	Mode	Default value	FSK/OOK Description
RegPacketConfig1 (0x30)	7	PacketFormat	rw	0x01	Defines the packet format used: 0 → Fixed length 1 → Variable length
	6-5	DcFree	rw	0x00	Defines DC-free encoding/decoding performed: 00 → None (Off) 01 → Manchester 10 → Whitening 11 → reserved
	4	CrcOn	rw	0x01	Enables CRC calculation/check (Tx/Rx): 0 → Off 1 → On
	3	CrcAutoClearOff	rw	0x00	Defines the behavior of the packet handler when CRC check fails: 0 → Clear FIFO and restart new packet reception. No *PayloadReady* interrupt issued. 1 → Do not clear FIFO. *PayloadReady* interrupt issued.
	2-1	AddressFiltering	rw	0x00	Defines address based filtering in Rx: 00 → None (Off) 01 → Address field must match *NodeAddress* 10 → Address field must match *NodeAddress* or *BroadcastAddress* 11 → reserved
	0	CrcWhiteningType	rw	0x00	Selects the CRC and whitening algorithms: 0 → CCITT CRC implementation with standard whitening 1 → IBM CRC implementation with alternate whitening
RegPacketConfig2 (0x31)	7	unused	r	-	unused
	6	DataMode	rw	0x01	Data processing mode: 0 → Continuous mode 1 → Packet mode
	5	IoHomeOn	rw	0x00	Enables the io-homecontrol® compatibility mode 0 → Disabled 1 → Enabled
	4	IoHomePowerFrame	rw	0x00	reserved - Linked to io-homecontrol® compatibility mode
	3	BeaconOn	rw	0x00	Enables the Beacon mode in Fixed packet format
	2-0	PayloadLength(10:8)	rw	0x00	Packet Length Most significant bits
RegPayloadLength (0x32)	7-0	PayloadLength(7:0)	rw	0x40	If PacketFormat = 0 (fixed), payload length. If PacketFormat = 1 (variable), max length in Rx, not used in Tx.
RegNodeAdrs (0x33)	7-0	NodeAddress	rw	0x00	Node address used in address filtering.
RegBroadcastAdrs (0x34)	7-0	BroadcastAddress	rw	0x00	Broadcast address used in address filtering.

Name (Address)	Bits	Variable Name	Mode	Default value	FSK/OOK Description
RegFifoThresh (0x35)	7	TxStartCondition	rw	0x01 *	Defines the condition to start packet transmission: 0 → *FifoLevel* (i.e. the number of bytes in the FIFO exceeds *FifoThreshold)* 1 → *FifoEmpty goes low*(i.e. at least one byte in the FIFO)
	6	unused	r	-	unused
	5-0	FifoThreshold	rw	0x0f	Used to trigger *FifoLevel* interrupt, when: number of bytes in FIFO >= FifoThreshold + 1
Sequencer registers					
RegSeqConfig1 (0x36)	7	SequencerStart	wt	0x00	Controls the top level Sequencer When set to '1', executes the "Start" transition. The sequencer can only be enabled when the chip is in Sleep or Standby mode.
	6	SequencerStop	wt	0x00	Forces the Sequencer Off. Always reads '0'
	5	IdleMode	rw	0x00	Selects chip mode during the state: 0: Standby mode 1: Sleep mode
	4-3	FromStart	rw	0x00	Controls the Sequencer transition when *SequencerStart* is set to 1 in Sleep or Standby mode: 00: to LowPowerSelection 01: to Receive state 10: to Transmit state 11: to Transmit state on a *FifoLevel* interrupt
	2	LowPowerSelection	rw	0x00	Selects the Sequencer LowPower state after a *to LowPowerSelection* transition: 0: SequencerOff state with chip on Initial mode 1: Idle state with chip on *Standby* or *Sleep* mode depending on *IdleMode* *Note: Initial mode is the chip LowPower mode at Sequencer Start.*
	1	FromIdle	rw	0x00	Controls the Sequencer transition from the Idle state on a T1 interrupt: 0: to Transmit state 1: to Receive state
	0	FromTransmit	rw	0x00	Controls the Sequencer transition from the Transmit state: 0: to LowPowerSelection on a *PacketSent* interrupt 1: to Receive state on a *PacketSent* interrupt

Name (Address)	Bits	Variable Name	Mode	Default value	FSK/OOK Description
RegSeqConfig2 (0x37)	7-5	FromReceive	rw	0x00	Controls the Sequencer transition from the Receive state 000 and 111: unused 001: to PacketReceived state on a *PayloadReady* interrupt 010: to LowPowerSelection on a *PayloadReady* interrupt 011: to PacketReceived state on a *CrcOk* interrupt (1) 100: to SequencerOff state on a *Rssi* interrupt 101: to SequencerOff state on a *SyncAddress* interrupt 110: to SequencerOff state on a *PreambleDetect* interrupt Irrespective of this setting, transition to LowPowerSelection on a T2 interrupt (1) If the CRC is wrong (corrupted packet, with CRC on but *CrcAutoClearOn*=0), the *PayloadReady* interrupt will drive the sequencer to RxTimeout state.
	4-3	FromRxTimeout	rw	0x00	Controls the state-machine transition from the Receive state on a *RxTimeout* interrupt (and on *PayloadReady* if FromReceive = 011): 00: to Receive State, via ReceiveRestart 01: to Transmit state 10: to LowPowerSelection 11: to SequencerOff state *Note: RxTimeout interrupt is a TimeoutRxRssi, TimeoutRxPreamble or TimeoutSignalSync interrupt*
	2-0	FromPacketReceived	rw	0x00	Controls the state-machine transition from the PacketReceived state: 000: to SequencerOff state 001: to Transmit state on a *FifoEmpty* interrupt 010: to LowPowerSelection 011: to Receive via FS mode, if frequency was changed 100: to Receive state (no frequency change)
RegTimerResol (0x38)	7-4	unused	r	-	unused
	3-2	Timer1Resolution	rw	0x00	Resolution of Timer 1 00: Timer1 disabled 01: 64 us 10: 4.1 ms 11: 262 ms
	1-0	Timer2Resolution	rw	0x00	Resolution of Timer 2 00: Timer2 disabled 01: 64 us 10: 4.1 ms 11: 262 ms
RegTimer1Coef (0x39)	7-0	Timer1Coefficient	rw	0xF5	Multiplying coefficient for Timer 1
RegTimer2Coef (0x3A)	7-0	Timer2Coefficient	rw	0x20	Multiplying coefficient for Timer 2

Name (Address)	Bits	Variable Name	Mode	Default value	FSK/OOK Description
Service registers					
RegImageCal (0x3B)	7	AutoImageCalOn	rw	0x00 *	Controls the Image calibration mechanism 0 → Calibration of the receiver depending on the temperature is disabled 1 → Calibration of the receiver depending on the temperature enabled.
	6	ImageCalStart	wt	-	Triggers the IQ and RSSI calibration when set in Standby mode.
	5	ImageCalRunning	r	0x00	Set to 1 while the Image and RSSI calibration are running. Toggles back to 0 when the process is completed
	4	unused	r	-	unused
	3	TempChange	r	0x00	IRQ flag witnessing a temperature change exceeding TempThreshold since the last Image and RSSI calibration: 0 → Temperature change lower than TempThreshold 1 → Temperature change greater than TempThreshold
	2-1	TempThreshold	rw	0x01	Temperature change threshold to trigger a new I/Q calibration 00 → 5 °C 01 → 10 °C 10 → 15 °C 11 → 20 °C
	0	TempMonitorOff	rw	0x00	Controls the temperature monitor operation: 0 → Temperature monitoring done in all modes except Sleep and Standby 1 → Temperature monitoring stopped.
RegTemp (0x3C)	7-0	TempValue	r	-	Measured temperature -1°C per Lsb Needs calibration for absolute accuracy
RegLowBat (0x3D)	7-4	unused	r	-	unused
	3	LowBatOn	rw	0x00	Low Battery detector enable signal 0 → LowBat detector disabled 1 → LowBat detector enabled
	2-0	LowBatTrim	rw	0x02	Trimming of the LowBat threshold: 000 → 1.695 V 001 → 1.764 V 010 → 1.835 V (d) 011 → 1.905 V 100 → 1.976 V 101 → 2.045 V 110 → 2.116 V 111 → 2.185 V
Status registers					

Name (Address)	Bits	Variable Name	Mode	Default value	FSK/OOK Description
RegIrqFlags1 (0x3E)	7	ModeReady	r	-	Set when the operation mode requested in *Mode*, is ready - Sleep: Entering Sleep mode - Standby: XO is running - FS: PLL is locked - Rx: RSSI sampling starts - Tx: PA ramp-up completed Cleared when changing the operating mode.
	6	RxReady	r	-	Set in Rx mode, after RSSI, AGC and AFC. Cleared when leaving Rx.
	5	TxReady	r	-	Set in Tx mode, after PA ramp-up. Cleared when leaving Tx.
	4	PllLock	r	-	Set (in FS, Rx or Tx) when the PLL is locked. Cleared when it is not.
	3	Rssi	rwc	-	Set in Rx when the *RssiValue* exceeds *RssiThreshold.* Cleared when leaving Rx or setting this bit to 1.
	2	Timeout	r	-	Set when a timeout occurs Cleared when leaving Rx or FIFO is emptied.
	1	PreambleDetect	rwc	-	Set when the Preamble Detector has found valid Preamble. bit clear when set to 1
	0	SyncAddressMatch	rwc	-	Set when Sync and Address (if enabled) are detected. Cleared when leaving Rx or FIFO is emptied. This bit is read only in Packet mode, rwc in Continuous mode
RegIrqFlags2 (0x3F)	7	FifoFull	r	-	Set when FIFO is full (i.e. contains 66 bytes), else cleared.
	6	FifoEmpty	r	-	Set when FIFO is empty, and cleared when there is at least 1 byte in the FIFO.
	5	FifoLevel	r	-	Set when the number of bytes in the FIFO strictly exceeds *FifoThreshold*, else cleared.
	4	FifoOverrun	rwc	-	Set when FIFO overrun occurs. (except in Sleep mode) Flag(s) and FIFO are cleared when this bit is set. The FIFO then becomes immediately available for the next transmission / reception.
	3	PacketSent	r	-	Set in Tx when the complete packet has been sent. Cleared when exiting Tx
	2	PayloadReady	r	-	Set in Rx when the payload is ready (i.e. last byte received and CRC, if enabled and *CrcAutoClearOff* is cleared, is Ok). Cleared when FIFO is empty.
	1	CrcOk	r	-	Set in Rx when the CRC of the payload is Ok. Cleared when FIFO is empty.
	0	LowBat	rwc	-	Set when the battery voltage drops below the Low Battery threshold. Cleared only when set to 1 by the user.
IO control registers					

Name (Address)	Bits	Variable Name	Mode	Default value	FSK/OOK Description
RegDioMapping1 (0x40)	7-6	Dio0Mapping	rw	0x00	Mapping of pins DIO0 to DIO5 See Table 17 for mapping in LoRa mode See Table 28 for mapping in Continuous mode See Table 29 for mapping in Packet mode
	5-4	Dio1Mapping	rw	0x00	
	3-2	Dio2Mapping	rw	0x00	
	1-0	Dio3Mapping	rw	0x00	
RegDioMapping2 (0x41)	7-6	Dio4Mapping	rw	0x00	
	5-4	Dio5Mapping	rw	0x00	
	3-1	reserved	rw	0x00	reserved. Retain default value
	0	MapPreambleDetect	rw	0x00	Allows the mapping of either *Rssi* Or *PreambleDetect* to the DIO pins, as summarized on Table 28 and Table 29 0 → *Rssi* interrupt 1 → *PreambleDetect* interrupt
Version register					
RegVersion (0x42)	7-0	Version	r	0x22	Version code of the chip. Bits 7-4 give the full revision number; bits 3-0 give the metal mask revision number.
Additional registers					
RegAgcRef (0x43)	7-6	unused	r	-	unused
	5-0	AgcReferenceLevel	rw	0x13	Sets the floor reference for all AGC thresholds: AGC Reference [dBm] = -174 dBm + 10*log(2**RxBw*) + SNR + *AgcReferenceLevel* SNR = 8 dB, fixed value
RegAgcThresh1 (0x44)	7-5	unused	r	-	unused
	4-0	AgcStep1	rw	0x0E	Defines the 1st AGC Threshold
RegAgcThresh2 (0x45)	7-4	AgcStep2	rw	0x05	Defines the 2nd AGC Threshold:
	3-0	AgcStep3	rw	0x0B	Defines the 3rd AGC Threshold:
RegAgcThresh3 (0x46)	7-4	AgcStep4	rw	0x0D	Defines the 4th AGC Threshold:
	3-0	AgcStep5	rw	0x0B	Defines the 5th AGC Threshold:
RegPllHop (0x4b)	7	FastHopOn	rw	0x00	Bypasses the main state machine for a quick frequency hop. Writing RegFrfLsb will trigger the frequency change. 0 → Frf is validated when FSTx or FSRx is requested 1 → Frf is validated triggered when RegFrfLsb is written
	6-0	reserved	rw	0x2E	reserved
RegTcxo (0x58)	7-5	reserved	rw	0x00	reserved. Retain default value
	4	TcxoInputOn	rw	0x00	Controls the crystal oscillator 0 → Crystal Oscillator with external Crystal 1 → External clipped sine TCXO AC-connected to XTA pin
	3-0	reserved	rw	0x09	Reserved. Retain default value.
RegPaDac (0x5A)	7-3	reserved	rw	0x10	reserved. Retain default value
	2-0	PaDac	rw	0x04	Enables the +20 dBm option on PA_BOOST pin 0x04 → Default value 0x07 → +20 dBm on PA_BOOST when OutputPower = 1111

Name (Address)	Bits	Variable Name	Mode	Default value	FSK/OOK Description
RegPll (0x5C)	7-6	PllBandwidth	rw	0x03	Controls the PLL bandwidth: 00 → 75 kHz 10 → 225 kHz 01 → 150 kHz 11 → 300 kHz
	5-0	reserved	rw	0x10	reserved. Retain default value
RegPllLowPn (0x5E)	7-6	PllBandwidth	rw	0x03	Controls the Low Phase Noise PLL bandwidth: 00 → 75 kHz 10 → 225 kHz 01 → 150 kHz 11 → 300 kHz
	5-0	reserved	rw	0x10	reserved. Retain default value
RegFormerTemp (0x6C)	7-0	FormerTemp	rw	-	Temperature saved during the latest IQ (RSSI and Image) calibrated. Same format as *TempValue* in *RegTemp*.
RegBitrateFrac (0x70)	7-4	unused	r	0x00	unused
	3-0	BitRateFrac	rw	0x00	Fractional part of the bit rate divider (Only valid for FSK) If *BitRateFrac*> 0 then: $BitRate = \frac{FXOSC}{BitRate(15,0) + \frac{BitrateFrac}{16}}$

6.3. LoRa™ Mode Register Map

This section details the SX1272/73 register mapping and the precise contents of each register in LoRa™ mode.

It is essential to understand that the LoRa modem is controlled independently of the FSK modem. Therefore, care should be taken when accessing the registers, especially as some register may have the same name in LoRa or FSK mode.

The LoRa registers are only accessible when the device is set in Lora mode (and, in the same way, the FSK register are only accessible in FSK mode). However, in some cases, it may be necessary to access some of the FSK register while in LoRa mode. To this aim, the *AccesSharedReg* bit was created in the *RegOpMode* register. This bit, when set to '1', will grant access to the FSK register 0x0D up to the register 0x3F. Once the setup has been done, it is strongly recommended to clear this bit so that LoRa register can be access normally.

Convention: r: read, w: write, c: set to clear and t: trigger

Table 41 Register Map, LoRa Mode

Name (Address)	Bits	Variable Name	Mode	Reset	LoRa™ Description
RegFifo (0x00)	7-0	Fifo	rw	0x00	LoRa™ base-band FIFO data input/output. FIFO is cleared an not accessible when device is in SLEEP mode
Common Register Settings					
RegOpMode (0x01)	7	LongRangeMode	rw	0x0	0 → FSK/OOK Mode 1 → LoRa™ Mode This bit can be modified only in Sleep mode. A write operation on other device modes is ignored.
	6	AccesSharedReg	rw	0x0	This bit operates when device is in Lora mode; if set it allows access to FSK registers page located in address space (0x0D:0x3F) while in LoRa mode 0 → Access LoRa registers page 0x0D: 0x3F 1 → Access FSK registers page (in mode LoRa) 0x0D: 0x3F
	5-3	unused	r	0x00	
	2-0	Mode	rwt	0x01	Device modes 000 → SLEEP 001 → STDBY 010 → Frequency synthesis TX (FSTX) 011 → Transmit (TX) 100 → Frequency synthesis RX (FSRX) 101 → Receive continuous (RXCONTINUOUS) 110 → receive single (RXSINGLE) 111 → Channel activity detection (CAD)
(0x02)	7-0	reserved	r	0x00	-
(0x03)	7-0	reserved	r	0x00	-
(0x04)	7-0	reserved	r	0x00	-
(0x05)	7-0	reserved	r	0x00	-
RegFrMsb (0x06)	7-0	Frf(23:16)	rw	0xE4	MSB of RF carrier frequency
RegFrMib (0x07)	7-0	Frf(15:8)	rw	0xC0	MIB of RF carrier frequency

Name (Address)	Bits	Variable Name	Mode	Reset	LoRa™ Description
RegFrLsb (0x08)	7-0	Frf(7:0)	rwt	0x00	LSB of RF carrier frequency $f_{RF} = \frac{F(XOSC) \cdot Frf}{2^{19}}$ Resolution is 61.035 Hz if F(XOSC) = 32 MHz. Default value is 0xe4c000 = 915 MHz. Register values must be modified only when device is in SLEEP or STANDBY mode.
register for RF					
RegPaConfig (0x09)	7	PaSelect	rw	0x00	Selects PA output pin 0 → RFIO pin. Output power is limited to 13 dBm. 1 → PA_BOOST pin. Output power is limited to 20 dBm
	6-4	unused	r	-	unused
	3-0	OutputPower	rw	0x0F	power amplifier max output power: Pout = 2 + OutputPower(3:0) on PA_BOOST. Pout = -1 + OutputPower(3:0) on RFIO.
RegPaRamp (0x0A)	7-5	unused	r	-	unused
	4	LowPnTxPllOff	rw	0x01	1 → Low consumption PLL is used in receive and transmit mode 0 → Low consumption PLL in receive mode, low phase noise PLL in transmit mode.
	3-0	PaRamp(3:0)	rw	0x09	Rise/Fall time of ramp up/down in FSK 0000 → 3.4 ms 0001 → 2 ms 0010 → 1 ms 0011 → 500 us 0100 → 250 us 0101 → 125 us 0110 → 100 us 0111 → 62 us 1000 → 50 us 1001 → 40 us 1010 → 31 us 1011 → 25 us 1100 → 20 us 1101 → 15 us 1110 → 12 us 1111 → 10 us
RegOcp (0x0B)	7-6	unused	r	0x00	unused
	5	OcpOn	rw	0x01	Enables overload current protection (OCP) for PA: 0 → OCP disabled 1 → OCP enabled
	4-0	OcpTrim	rw	0x0B	Trimming of OCP current: Imax = 45+5*OcpTrim [mA] if OcpTrim <= 15 (120 mA) / Imax = -30+10*OcpTrim [mA] if 15 < OcpTrim <= 27 (130 to 240 mA) Imax = 240mA for higher settings Default Imax = 100mA

Name (Address)	Bits	Variable Name	Mode	Reset	LoRa™ Description
RegLna (0x0C)	7-5	LnaGain	rwx	0x01	LNA gain setting: 000 → not used 001 → G1 = maximum gain 010 → G2 011 → G3 100 → G4 101 → G5 110 → G6 = minimum gain 111 → not used
	4-2	reserved	r	0x00	-
	1-0	LnaBoost	rw	0x00	00 → Default LNA current 11 → Boost on, 150% LNA current.
Lora page registers					
RegFifoAddrPtr (0x0D)	7-0	FifoAddrPtr	rw	0x00	SPI interface address pointer in FIFO data buffer.
RegFifoTxBaseAddr (0x0E)	7-0	FifoTxBaseAddr	rw	0x80	write base address in FIFO data buffer for TX modulator
RegFifoRxBaseAddr (0x0F)	7-0	FifoRxBaseAddr	rw	0x00	read base address in FIFO data buffer for RX demodulator
RegFifoRxCurrentAddr (0x10)	7-0	FifoRxCurrentAddr	r	n/a	Start address (in data buffer) of last packet received
RegIrqFlagsMask (0x11)	7	RxTimeoutMask	rw	0x00	Timeout interrupt mask: setting this bit masks the corresponding IRQ in RegIrqFlags
	6	RxDoneMask	rw	0x00	Packet reception complete interrupt mask: setting this bit masks the corresponding IRQ in RegIrqFlags
	5	PayloadCrcErrorMask	rw	0x00	Payload CRC error interrupt mask: setting this bit masks the corresponding IRQ in RegIrqFlags
	4	ValidHeaderMask	rw	0x00	Valid header received in Rx mask: setting this bit masks the corresponding IRQ in RegIrqFlags
	3	TxDoneMask	rw	0x00	FIFO Payload transmission complete interrupt mask: setting this bit masks the corresponding IRQ in RegIrqFlags
	2	CadDoneMask	rw	0x00	CAD complete interrupt mask: setting this bit masks the corresponding IRQ in RegIrqFlags
	1	FhssChangeChannelMask	rw	0x00	FHSS change channel interrupt mask: setting this bit masks the corresponding IRQ in RegIrqFlags
	0	CadDetectedMask	rw	0x00	Cad Detected Interrupt Mask: setting this bit masks the corresponding IRQ in RegIrqFlags

Name (Address)	Bits	Variable Name	Mode	Reset	LoRa™ Description
RegIrqFlags (0x12)	7	RxTimeout	rc	0x00	Timeout interrupt: writing a 1 clears the IRQ
	6	RxDone	rc	0x00	Packet reception complete interrupt: writing a 1 clears the IRQ
	5	PayloadCrcError	rc	0x00	Payload CRC error interrupt: writing a 1 clears the IRQ
	4	ValidHeader	rc	0x00	Valid header received in Rx: writing a 1 clears the IRQ
	3	TxDone	rc	0x00	FIFO Payload transmission complete interrupt: writing a 1 clears the IRQ
	2	CadDone	rc	0x00	CAD complete: write to clear: writing a 1 clears the IRQ
	1	FhssChangeChannel	rc	0x00	FHSS change channel interrupt: writing a 1 clears the IRQ
	0	CadDetected	rc	0x00	Valid Lora signal detected during CAD operation: writing a 1 clears the IRQ
RegRxNbBytes (0x13)	7-0	FifoRxBytesNb	r	n/a	Number of payload bytes of latest packet received
RegRxHeaderCntValueMsb (0x14)	7-0	ValidHeaderCntMsb(15:8)	r	n/a	Number of valid headers received since last transition into Rx mode, MSB(15:8). Header and packet counters are reseted in Sleep mode.
RegRxHeaderCntValueLsb (0x15)	7-0	ValidHeaderCntLsb(7:0)	r	n/a	Number of valid headers received since last transition into Rx mode, LSB(7:0). Header and packet counters are reseted in Sleep mode.
RegRxPacketCntValueMsb (0x16)	7-0	ValidPacketCntMsb(15:8)	rc	n/a	Number of valid packets received since last transition into Rx mode, MSB(15:8). Header and packet counters are reseted in Sleep mode.
RegRxPacketCntValueLsb (0x17)	7-0	ValidPacketCntLsb(7:0)	r	n/a	Number of valid packets received since last transition into Rx mode, LSB(7:0). Header and packet counters are reseted in Sleep mode.
RegModemStat (0x18)	7-5	RxCodingRate	r	n/a	Coding rate of last header received
	4	ModemStatus	r	‘1’	Modem clear
	3		r	‘0’	Header info valid
	2		r	‘0’	RX on-going
	1		r	‘0’	Signal synchronized
	0		r	‘0’	Signal detected
RegPktSnrValue (0x19)	7-0	PacketSnr	r	n/a	Estimation of SNR on last packet received.In two’s compliment format mutiplied by 4. $$SNR[dB] = \frac{PacketSnr[twos\ complement]}{4}$$

Name (Address)	Bits	Variable Name	Mode	Reset	LoRa™ Description
RegPktRssiValue (0x1A)	7-0	PacketRssi	r	n/a	RSSI of the latest packet received (dBm) RSSI[dBm] = - 139 + *PacketRssi* (when SNR >= 0) or RSSI[dBm] = - 139 + *PacketRssi* + *PacketSnr *0.25* (when SNR < 0)
RegRssiValue (0x1B)	7-0	Rssi	r	n/a	Current RSSI value (dBm) RSSI[dBm] = - 139 + *Rssi*
RegHopChannel (0x1C)	7	PllTimeout	r	n/a	PLL failed to lock while attempting a TX/RX/CAD operation 1 → PLL did not lock 0 → PLL did lock
	6	CrcOnPayload	r	n/a	CRC Information extracted from the received packet header (Explicit header mode only) 0 → Header indicates CRC off 1 → Header indicates CRC on
	5-0	FhssPresentChannel	r	n/a	Current value of frequency hopping channel in use.
RegModemConfig 1 (0x1D)	7-6	Bw	rw	0x0	Signal bandwidth: 00 → 125 kHz 01 → 250 kHz 10 → 500 kHz 11 → reserved
	5-3	CodingRate	rw	'001'	Error coding rate 001 → 4/5 010 → 4/6 011 → 4/7 100 → 4/8 All other values → reserved In implicit header mode should be set on receiver to determine expected coding rate. See Section 4.1.1.3.
	2	ImplicitHeaderModeOn	rw	0x0	0 → Explicit Header mode 1 → Implicit Header mode
	1	RxPayloadCrcOn	rw	0x0	Enable CRC generation and check on payload: 0 → CRC disable 1 → CRC enable If CRC is needed, RxPayloadCrcOn should be set: - in Implicit header mode: on Tx and Rx side - in Explicit header mode: on the Tx side alone (recovered from the header in Rx side)
	0	LowDataRateOptimize	rw	0x0	0 → Disabled 1 → Enabled; mandated for SF11 and SF12 with BW = 125 kHz

Name (Address)	Bits	Variable Name	Mode	Reset	LoRa™ Description
RegModemConfig 2 (0x1E)	7-4	SpreadingFactor	rw	0x7	SF rate (expressed as a base-2 logarithm) 6 → 64 chips / symbol 7 → 128 chips / symbol 8 → 256 chips / symbol 9 → 512 chips / symbol 10 → 1024 chips / symbol 11 → 2048 chips / symbol 12 → 4096 chips / symbol other values reserved.
	3	TxContinuousMode	rw	0	0 → normal mode, a single packet is sent 1 → continuous mode, send multiple packets across the FIFO (used for spectral analysis)
	2	AgcAutoOn	rw	0x01	0 → LNA gain set by register LnaGain 1 → LNA gain set by the internal AGC loop
	1-0	SymbTimeout(9:8)	rw	0x00	RX Time-Out MSB
RegSymbTimeoutLsb (0x1F)	7-0	SymbTimeout(7:0)	rw	0x64	RX Time-Out LSB RX operation time-out value expressed as number of symbols: $TimeOut = SymbTimeout \cdot Ts$
RegPreambleMsb (0x20)	7-0	PreambleLength(15:8)	rw	0x0	Preamble length MSB, = PreambleLength + 4.25 Symbols See Section 4.1.1.6 for more details.
RegPreambleLsb (0x21)	7-0	PreambleLength(7:0)	rw	0x8	Preamble Length LSB
RegPayloadLength (0x22)	7-0	PayloadLength(7:0)	rw	0x1	Payload length in bytes. The register needs to be set in implicit header mode for the expected packet length. A 0 value is not permitted
RegMaxPayloadLength (0x23)	7-0	PayloadMaxLength(7:0)	rw	0xFF	Maximum payload length; if header payload length exceeds value a header CRC error is generated. Allows filtering of packet with a bad size.
RegHopPeriod (0x24)	7-0	FreqHoppingPeriod(7:0)	rw	0x0	Symbol periods between frequency hops. (0 = disabled). 1st hop always happen after the 1st header symbol
RegFifoRxByteAddr (0x25)	7-0	FifoRxByteAddrPtr	r	n/a	Current value of RX databuffer pointer (address of last byte written by Lora receiver)
(0x26) - (0x27)	-	Reserved	r	n/a	Reserved
RegFeiMsb (0x28)	7-4	Reserved	r	n/a	Reserved
	3-0	FreqError(19:16)	r	0x0	Estimated frequency error from modem in 2's compliment format. MSB of RF Frequency error $F_{Error} = \frac{FreqError \times 2^{24}}{F_{xtal}}$
(RegFeiMid (0x29)	7-0	FreqError(15:8)	r	0x0	Middle byte of RF Frequency Error

Name (Address)	Bits	Variable Name	Mode	Reset	LoRa™ Description
RegFeiLsb (0x2A)	7-0	FreqError(7:0)	r	0x0	LSB of RF Frequency Error
(0x2B)	-	Reserved	r	n/a	Reserved
RegRssiWideband (0x2C)	7-0	RssiWideband(7:0)	r	n/a	Wideband RSSI measurement used to locally generate a random number
(0x2D) - (0x30)	-	Reserved	r	n/a	Reserved
RegDetectOptimize (0x31)	7-3	Reserved	r	0xC0	Reserved
	2-0	DetectionOptimize	rw	0x03	LoRa detection Optimize 0x03 → SF7 to SF12 0x05 → SF6
(0x32)	-	Reserved	r	n/a	Reserved
RegInvertIQ (0x33)	7	Reserved	rw	0x0	Reserved
	6	InvertIQ	rw	0x0	Invert the LoRa I and Q signals 0 → normal mode 1 → I and Q signals are inverted
	5-0	Reserved	rw	0x27	Reserved
(0x34) - (0x36)	7-0	Reserved	r	n/a	Reserved
RegDetectionThreshold (0x37)	7-0	DetectionThreshold	rw	0x0A	LoRa detection threshold 0x0A → SF7 to SF12 0x0C → SF6
(0x38)	-	Reserved	r	n/a	Reserved
RegSyncWord (0x39)	7-0	SyncWord	rw	0x12	LoRa Sync Word Value 0x34 is reserved for LoRaWAN networks
(0x3A) - (0x3F)	-	Reserved	r	n/a	Reserved

7. Application Information

7.1. Crystal Resonator Specification

Table 42 shows the crystal resonator specification for the crystal reference oscillator circuit of the SX1272/73. This specification covers the full range of operation of the SX1272/73 and is employed in the reference design.

Table 42 Crystal Specification

Symbol	Description	Conditions	Min	Typ	Max	Unit
FXOSC	XTAL Frequency		-	32	-	MHz
RS	XTAL Serial Resistance		-	15	40	Ohms
C0	XTAL Shunt Capacitance		-	1.5	3	pF
CFOOT	External Foot Capacitance	On each pin XTA and XTB	8	15	22	pF
CLOAD	Crystal Load Capacitance		6	-	12	pF

Notes - the initial frequency tolerance, temperature stability and aging performance should be chosen in accordance with the target operating temperature range and the receiver bandwidth selected.

- the loading capacitance should be applied externally, and adapted to the actual Cload specification of the XTAL.

7.2. Reset of the Chip

A power-on reset of the SX1272/73 is triggered at power up. Additionally, a manual reset can be issued by controlling pin 6.

7.2.1. POR

If the application requires the disconnection of VDD from the SX1272/73, despite of the extremely low Sleep Mode current, the user should wait for 10 ms from of the end of the POR cycle before commencing communications over the SPI bus. Pin 6 (Reset) should be left floating during the POR sequence.

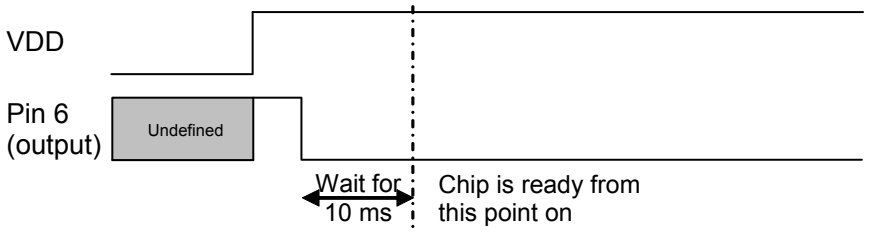


Figure 46. POR Timing Diagram

Please note that any CLKOUT activity can also be used to detect that the chip is ready.

7.2.2. Manual Reset

A manual reset of the SX1272/73 is possible even for applications in which VDD cannot be physically disconnected. Pin 6 should be pulled high for a hundred microseconds and then released. The user should then wait for 5 ms before using the chip.

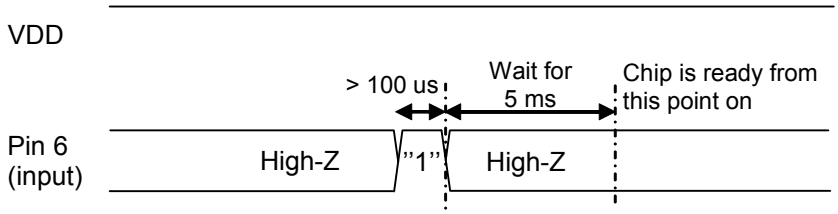


Figure 47. Manual Reset Timing Diagram

Note Whilst pin 6 is driven high an over current consumption of up to ten milliampere can be seen on VDD.

7.3. Top Sequencer: Listen Mode Examples

In this scenario the circuit spends most of the time in Idle mode during which only the RC oscillator is on. Periodically the receiver wakes up and looks for incoming signal. If a wanted signal is detected the receiver is kept on and data are analyzed. Otherwise, if there was no wanted signal for a defined period of time, the receiver is switched off until the next receive period.

During Listen mode the Radio stays most of the time in a Low Power mode resulting in very low average power consumption. The general timing diagram of this scenario is given in Figure 48.

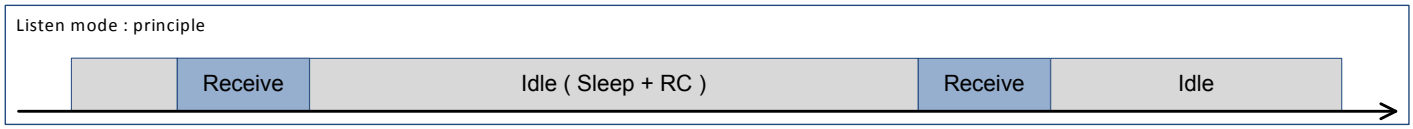


Figure 48. Listen Mode: Principle

An interrupt request is generated on a packet reception. The user can then take appropriate action.

Depending on the application and environment, there are several ways to implement Listen mode:

- Wake on a *PreambleDetect* interrupt.
- Wake on *a SyncAddress* interrupt.
- Wake on *a PayloadReady* interrupt.

7.3.1. Wake on Preamble Interrupt

In one possible scenario, the sequencer polls for a Preamble detection. If a preamble signal is detected, the sequencer is switched off and the circuit stays in Receive mode until the user switches modes. Otherwise, the receiver is switched off until the next Rx period.

7.3.1.1. Timing Diagram

When no signal is received, the circuit wakes every Timer1 + Timer2 and switches to Receive mode for a time defined by Timer2, as shown on the following diagram. If no Preamble is detected, it then switches back to Idle mode, i.e. Sleep mode with RC oscillator on.

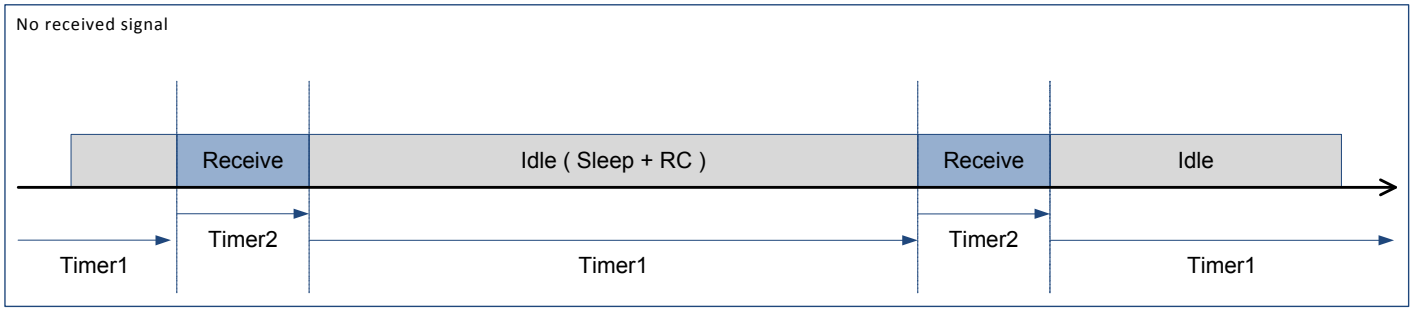


Figure 49. Listen Mode with No Preamble Received

If a Preamble signal is detected the Sequencer is switched off. The *PreambleDetect* signal can be mapped to DIO4 in order to request the user's attention.

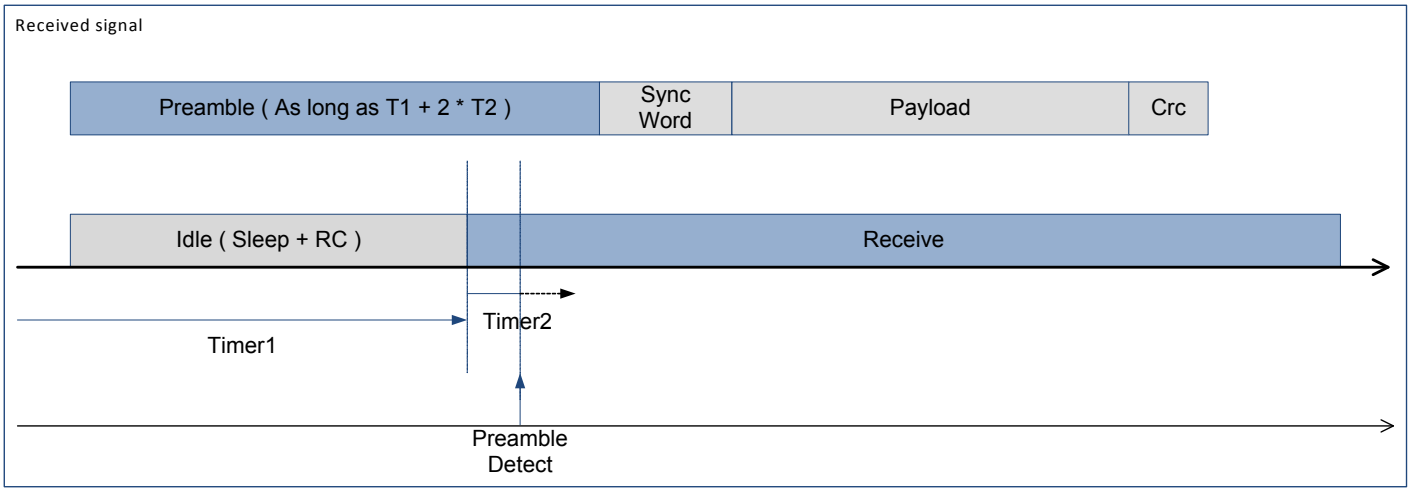


Figure 50. Listen Mode with Preamble Received

7.3.1.2. Sequencer Configuration

The following graph shows Listen mode - Wake on *PreambleDetect* state machine:

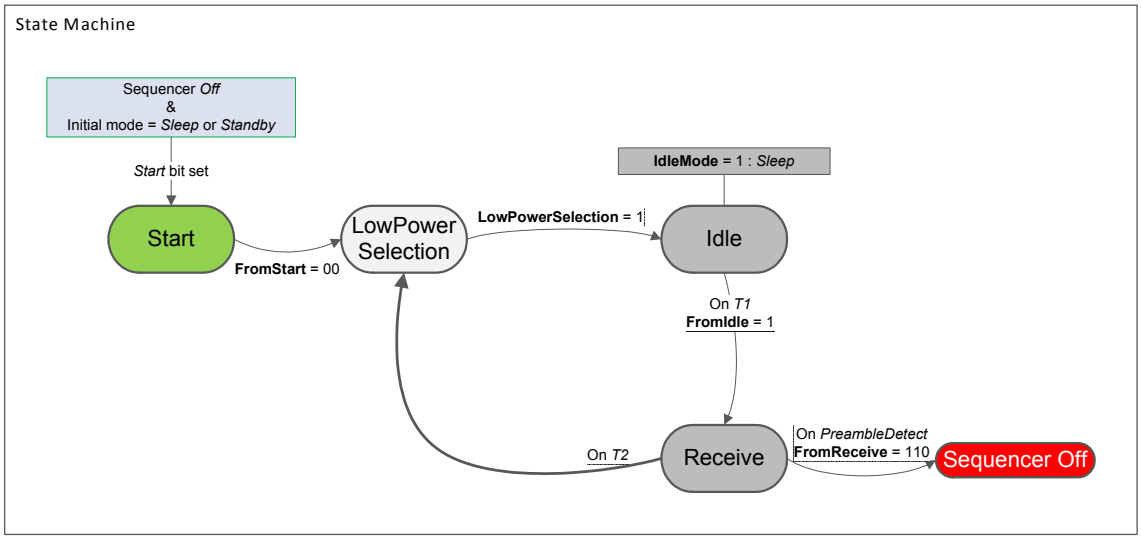


Figure 51. Wake On PreambleDetect State Machine

This example configuration is achieved as follows:

Table 43 Listen Mode with PreambleDetect Condition Settings

Variable	Effect
IdleMode	1: **Sleep** mode
FromStart	00: To **LowPowerSelection**
LowPowerSelection	1: To **Idle** state
FromIdle	1: To **Receive** state on *T1* interrupt
FromReceive	110: To **Sequencer Off** on *PreambleDetect* interrupt

T_{Timer2} defines the maximum duration the chip stays in Receive mode as long as no Preamble is detected. In order to optimize power consumption Timer2 must be set just long enough for Preamble detection.

T_{Timer1} + T_{Timer2} defines the cycling period, i.e. time between two Preamble polling starts. In order to optimize average power consumption, Timer1 should be relatively long. However, increasing Timer1 also extends packet reception duration.

In order to insure packet detection and optimize the receiver's power consumption the received packet Preamble should be as long as T_{Timer1} + 2 x T_{Timer2}.

An example of DIO configuration for this mode is described in the following table:

Table 44 Listen Mode with PreambleDetect Condition Recommended DIO Mapping

DIO	Value	Description
0	01	CrcOk
1	00	FifoLevel
3	00	FifoEmpty
4	11	PreambleDetect – Note: *MapPreambleDetect* bit should be set.

7.3.2. Wake on SyncAddress Interrupt

In another possible scenario, the sequencer polls for a Preamble detection and then for a valid *SyncAddress* interrupt. If events occur, the sequencer is switched off and the circuit stays in Receive mode until the user switches modes. Otherwise, the receiver is switched off until the next Rx period.

7.3.2.1. Timing Diagram

Most of the sequencer running time is spent duty cycling the receiver and idle modes with no wanted signal present. As shown by the timing diagram in Figure 52, the circuit wakes periodically for a short time, defined by RxTimeout. The circuit is in a Low Power mode for the rest of Timer1 + Timer2 (i.e. Timer1 + Timer2 - TrxTimeout)

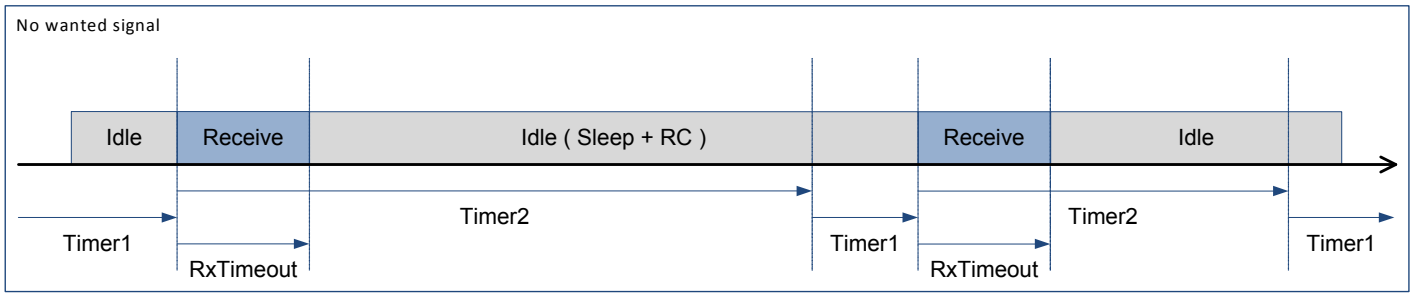


Figure 52. Listen Mode with no SyncAddress Detected

If a preamble is detected before *RxTimeout* timer ends the circuit stays in Receive mode and waits for a valid *SyncAddress* detection. If none is detected by the end of Timer2, Receive mode is deactivated and the polling cycle resumes, without any user intervention.

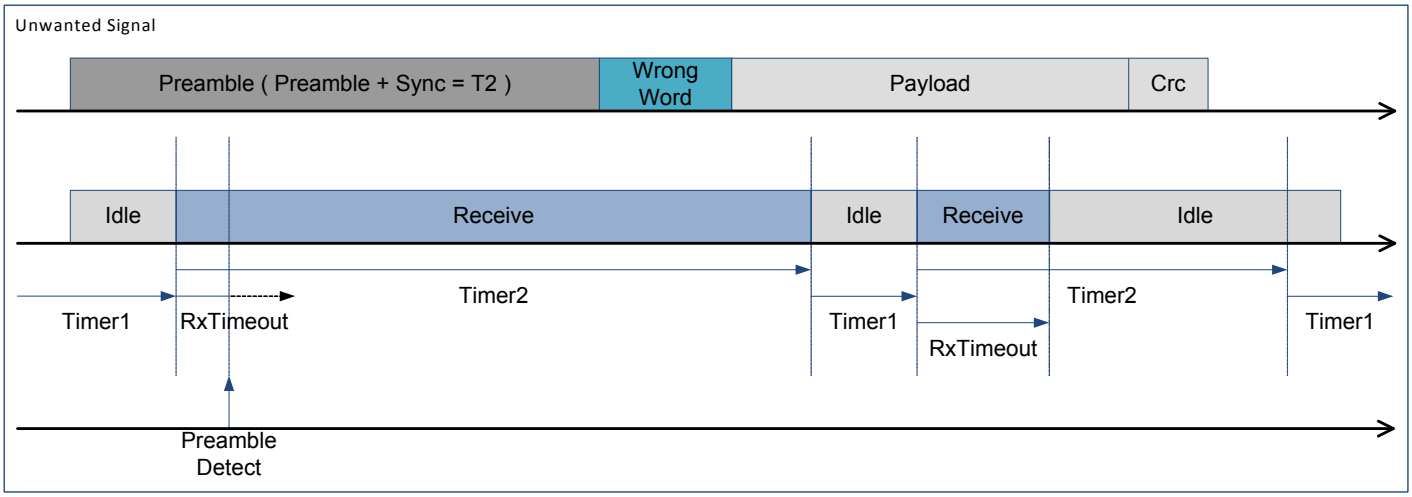


Figure 53. Listen Mode with Preamble Received and no SyncAddress

But if a valid Sync Word is detected a *SyncAddress* interrupt is fired, the Sequencer is switched off and the circuit stays in Receive mode as long as the user doesn't switch modes.

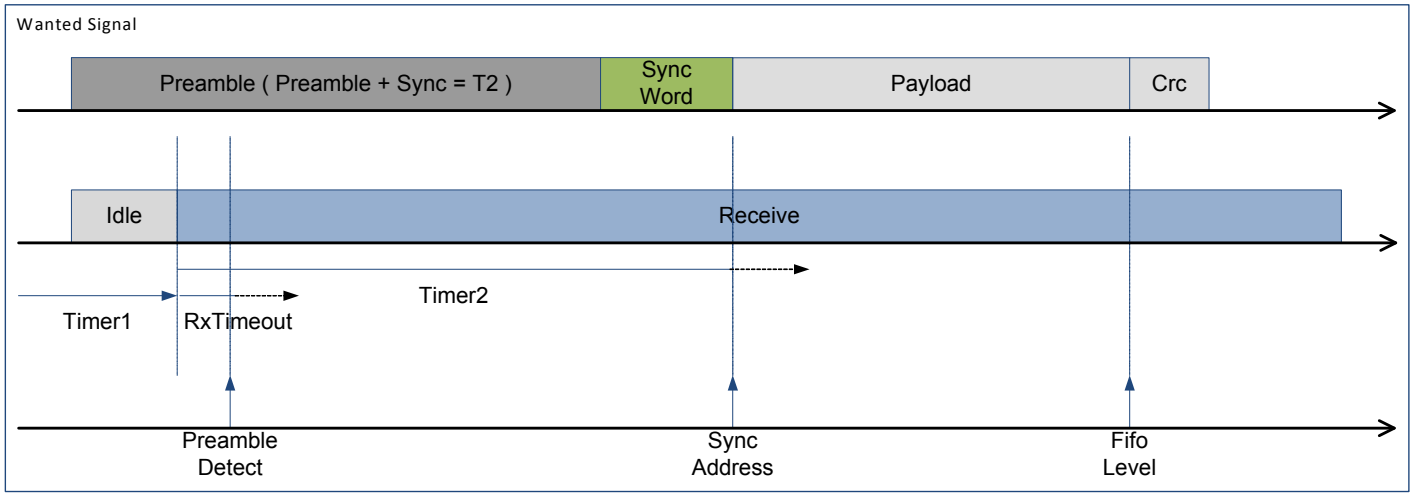


Figure 54. Listen Mode with Preamble Received & Valid SyncAddress

7.3.2.2. Sequencer Configuration

The following graph shows Listen mode - Wake on SyncAddress state machine:

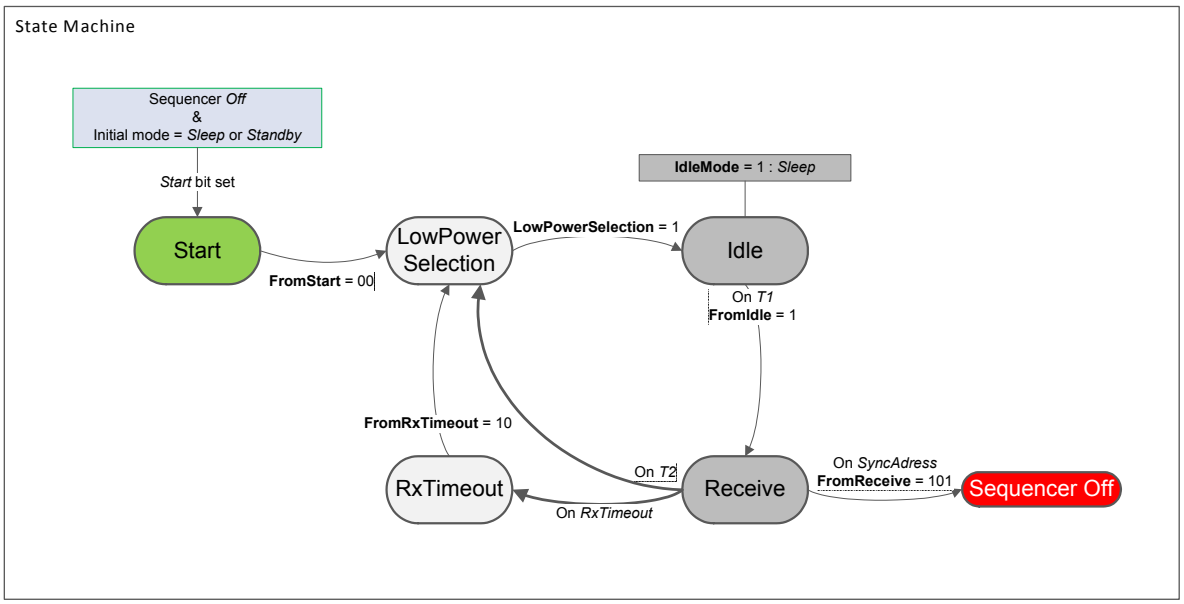


Figure 55. Wake On SyncAddress State Machine

This example configuration is achieved as follows:

Table 45 Listen Mode with SyncAddress Condition Settings

Variable	Effect
IdleMode	1: **Sleep** mode
FromStart	00: To **LowPowerSelection**

LowPowerSelection	1: To **Idle** state
FromIdle	1: To **Receive** state on *T1* interrupt
FromReceive	101: To **Sequencer off** on *SyncAddress* interrupt
FromRxTimeout	10: To **LowPowerSelection**

$T_{TimeoutRxPreamble}$ should be set to the expected transmit preamble duration (depends on *PreambleDetectSize* and *BitRate*).

T_{Timer1} should be set to 64 µs (shortest possible duration).

T_{Timer2} is set so that $T_{Timer1} + T_{Timer2}$ defines the time between two start of reception.

In order to ensure packet detection and optimize the receiver power consumption the received packet Preamble should be defined so that $T_{Preamble} = T_{Timer2} - T_{SyncAddress}$ with $T_{SyncAddress}$ = (*SyncSize* + 1)*8/*BitRate*.

An example of DIO configuration for this mode is described in the following table:

Table 46 Listen Mode with PreambleDetect Condition Recommended DIO Mapping

DIO	Value	Description
0	01	CrcOk
1	00	FifoLevel
2	11	SyncAddress
3	00	FifoEmpty
4	11	PreambleDetect – Note: *MapPreambleDetect* bit should be set.

7.4. Top Sequencer: Beacon Mode

In this mode, a single message is periodically re-transmitted. If the Payload being sent is always identical and *PayloadLength* is smaller than the FIFO size, the use of the *BeaconOn* bit in *RegPacketConfig2* together with the Sequencer permit to achieve periodic beacon without any user intervention.

7.4.1. Timing diagram

In this mode, the Radio is switched to Transmit mode every T_{Timer1} + T_{Timer2} and back to Idle mode after *PacketSent*, as shown in the diagram below. The Sequencer insures minimal time is spent in Transmit mode and therefore power consumption is optimized.

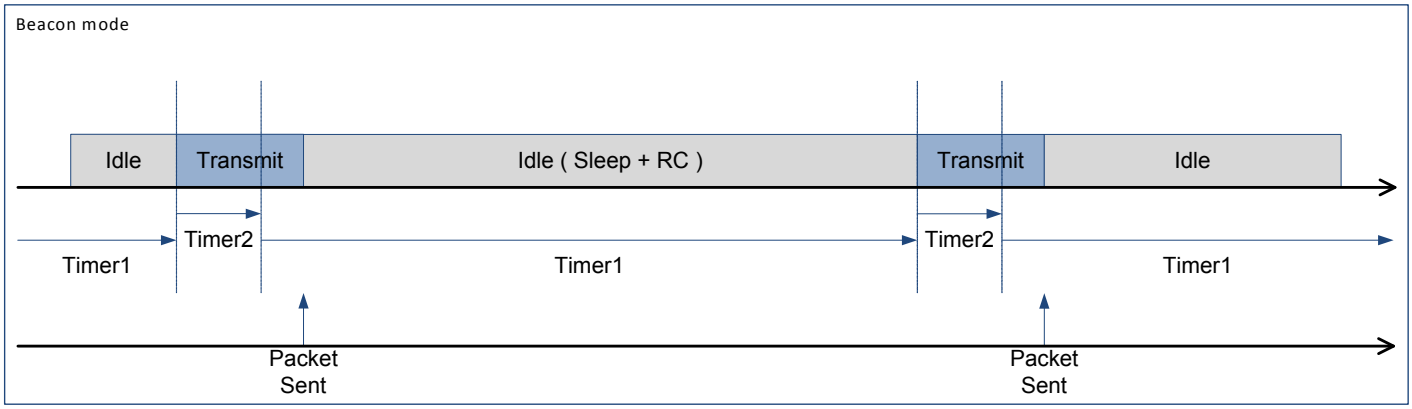


Figure 56. Beacon Mode Timing Diagram

7.4.2. Sequencer Configuration

The Beacon mode state machine is presented in the following graph. It should be noted that the sequencer enters an infinite loop and can only be stopped by setting *SequencerStop* bit in *RegSeqConfig1*.

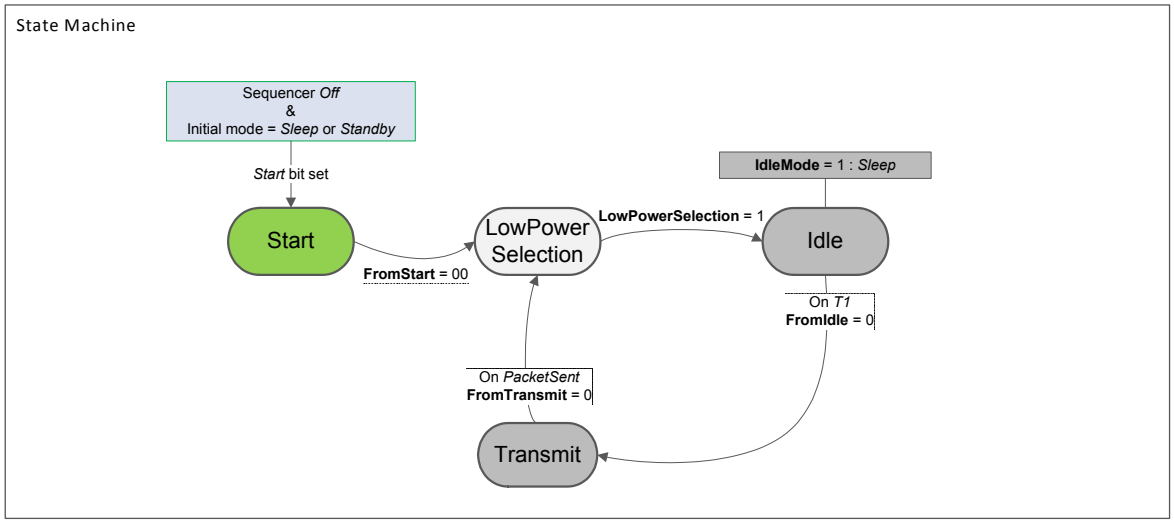


Figure 57. Beacon Mode State Machine

This example is achieved by programming the Sequencer as follows:

Table 47 Beacon Mode Settings

Variable	Effect
IdleMode	1: **Sleep** mode
FromStart	00: To **LowPowerSelection**
LowPowerSelection	1: To **Idle** state
FromIdle	0: To **Transmit** state on *T1* interrupt
FromTransmit	0: To **LowPowerSelection** on *PacketSent* interrupt

$T_{Timer1} + T_{Timer2}$ define the time between the start of two transmissions.

7.5. Example CRC Calculation

The following routine may be implemented to mimic the CRC calculation of the SX1272/73:

```
// CRC types
#define CRC_TYPE_CCITT                              0
#define CRC_TYPE_IBM                                1

// Polynomial = X^16 + X^12 + X^5 + 1
#define POLYNOMIAL_CCITT                            0x1021
// Polynomial = X^16 + X^15 + X^2 + 1
#define POLYNOMIAL_IBM                              0x8005

// Seeds
#define CRC_IBM_SEED                                0xFFFF
#define CRC_CCITT_SEED                              0x1D0F

/*
 * CRC algorithm implementation
 *
 * \param[IN] crc Previous CRC value
 * \param[IN] data New data to be added to the CRC
 * \param[IN] polynomial CRC polynomial selection [CRC_TYPE_CCITT, CRC_TYPE_IBM]
 *
 * \retval crc New computed CRC
 */
U16 ComputeCrc( U16 crc, U8 data, U16 polynomial )
{
    U8 i;
    for( i = 0; i < 8; i++ )
    {
        if( ( ( ( crc & 0x8000 ) >> 8 ) ^ ( data & 0x80 ) ) != 0 )
        {
            crc <<= 1;                  // shift left once
            crc ^= polynomial;          // XOR with polynomial
        }
        else
        {
            crc <<= 1;                  // shift left once
        }
        data <<= 1;                     // Next data bit
    }
    return crc;
}

/*
 * CRC algorithm implementation
 *
 * \param[IN] buffer Array containing the data
 * \param[IN] bufferLength Buffer length
 * \param[IN] crcType Selects the CRC polynomial[CRC_TYPE_CCITT, CRC_TYPE_IBM]
 *
 * \retval crc Buffer computed CRC
 */
U16 RadioPacketComputeCrc( U8 *buffer, U8 bufferLength, U8 crcType )
{
    U8 i;
    U16 crc;
    U16 polynomial;

    polynomial = ( crcType == CRC_TYPE_IBM ) ? POLYNOMIAL_IBM : POLYNOMIAL_CCITT;
    crc = ( crcType == CRC_TYPE_IBM ) ? CRC_IBM_SEED : CRC_CCITT_SEED;

    for( i = 0; i < bufferLength; i++ )
    {
        crc = ComputeCrc( crc, buffer[i], polynomial );
    }

    if( crcType == CRC_TYPE_IBM )
    {
        return crc;
    }
    else
    {
        return ( U16 )( ~crc );
    }
}
```

Figure 58. Example CRC Code

7.6. Example Temperature Reading

The following routine may be implemented to read the temperature and calibrate the sensor:

Temperature.c

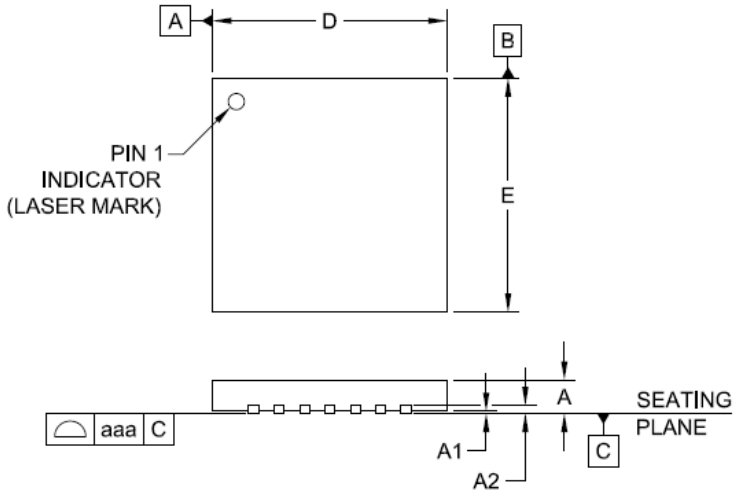
```
/*!
 * Reads the raw temperature
 * \retval temperature New raw temperature reading in 2's complement format
 */
S8 RadioGetRawTemp( void )
{
    S8 temp = 0;
    U8 regValue = 0;

    regValue = RadioRead( 0x3C );

    // 2's complements conversion
    temp = regValue & 0x7F;
    if( ( regValue & 0x80 ) == 0x80 )
    {
        temp *= -1;
    }
    return temp;
}

/*!
 * Computes the temperature compensation factor
 * \param [IN] actualTemp Actual temperature measured by an external device
 * \retval compensationFactor Computed compensation factor
 */
S8 RadioCalibrateTemp( S8 actualTemp )
{
    return actualTemp - RadioGetRawTemp( );
}

/*!
 * Gets the actual compensated temperature
 * \param [IN] compensationFactor Return value of the calibration function
 * \retval New compensated temperature value
 */
S8 RadioGetTemp( S8 compensationFactor )
{
    return RadioGetRawTemp( ) + compensationFactor;
}

/*!
 * Usage example
 */
void main( void )
{
    S8 temp;
    S8 actualTemp = 0;
    S8 compensationFactor = 0;

    // Ask user for the temperature during calibration
    actualTemp = AskUserTemperature( );
    compensationFactor = RadioCalibrateTemp( actualTemp );

    while( True )
    {
        temp = RadioGetTemp( compensationFactor );
    }
}
```

Figure 59. Example Temperature Reading

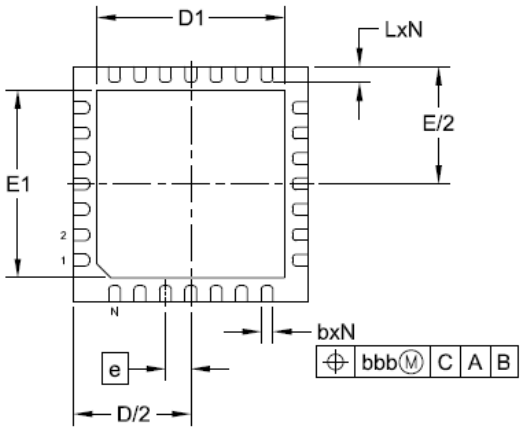
8. Packaging Information

8.1. Package Outline Drawing

The SX1272/73 is available in a 28-lead QFN package as shown in Figure 60.



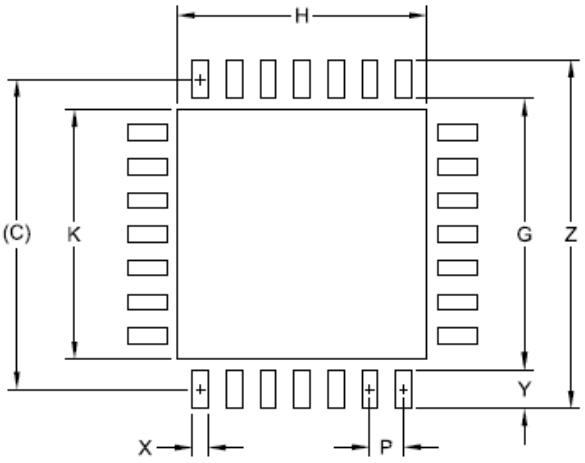
DIMENSIONS			
DIM	MILLIMETERS		
	MIN	NOM	MAX
A	0.80	—	1.00
A1	0.00	—	0.05
A2		(0.20)	
b	0.25	0.30	0.35
D	5.90	6.00	6.10
D1	4.65	4.80	4.90
E	5.90	6.00	6.10
E1	4.65	4.80	4.90
e		0.65 BSC	
L	0.30	0.40	0.50
N		28	
aaa		0.08	
bbb		0.10	



NOTES:
1. CONTROLLING DIMENSIONS ARE IN MILLIMETERS (ANGLES IN DEGREES).
2. COPLANARITY APPLIES TO THE EXPOSED PAD AS WELL AS THE TERMINALS.

Figure 60. Package Outline Drawing

8.2. Recommended Land Pattern



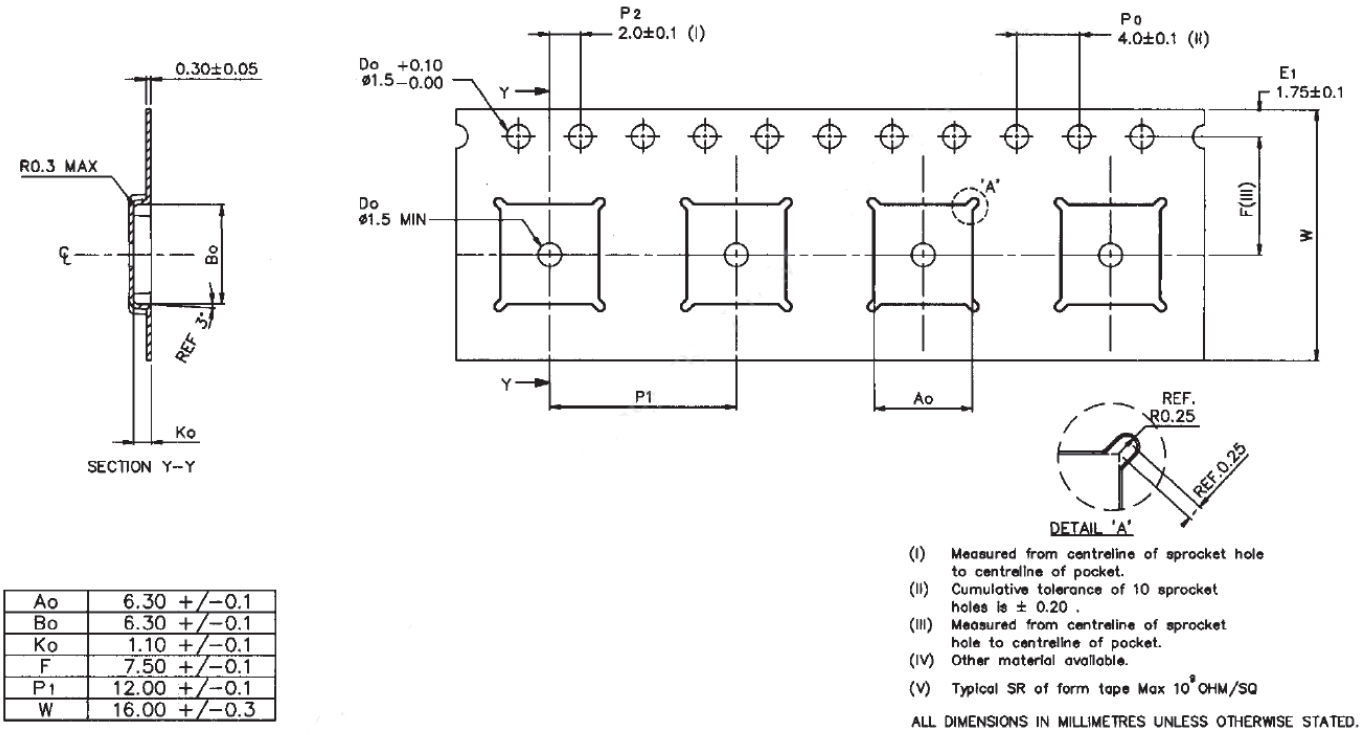
DIMENSIONS	
DIM	MILLIMETERS
C	(5,95)
G	5.20
H	4.80
K	4.80
P	0.65
X	0.30
Y	0.75
Z	6.70

NOTES:

1. CONTROLLING DIMENSIONS ARE IN MILLIMETERS (ANGLES IN DEGREES).
2. THIS LAND PATTERN IS FOR REFERENCE PURPOSE ONLY.
   CONSULT YOUR MANUFACTURING GROUP TO ENSURE YOUR COMPANY'S MANUFACTURING GUIDELINES ARE MET.
3. THERMAL VIAS IN THE LAND PATTERN OF THE EXPOSED PAD SHALL BE CONNECTED TO A SYSTEM GROUND PLANE. FAILURE TO DO SO MAY COMPROMISE THE THERMAL AND/OR FUNCTIONAL PERFORMANCE OF THE DEVICE.
4. SQUARE PACKAGE - DIMENSIONS APPLY IN BOTH " X " AND " Y " DIRECTIONS.

Figure 61. Recommended Land Pattern

8.3. Tape and Reel Information



Ao	6.30 +/−0.1
Bo	6.30 +/−0.1
Ko	1.10 +/−0.1
F	7.50 +/−0.1
P1	12.00 +/−0.1
W	16.00 +/−0.3

Figure 62. Tape and Reel Information

9. Revision History

Table 48 Revision History

Revision	Date	Comment
1	June 2013	First release.
2	July 2014	Inclusion of FEI Correction of ToA formula Improve description in the RSSI and IQ calibration mechanism Correction of default value in FSK Added undocumented register
3	October 2014	Corrected Rssi formula in LoRa mode (text +Register Table
4	March 2015	Clarified operation modes for Rx Single and Rx Continuous mode in LoRa Added use cases for Rx Single and Rx Continuous mode in LoRa mode Clarified used of LoRa RxPayloadCrcOn in Register Table Added description of register RegSyncWord in LoRa register table Changed Stand-By typo into Standby

Contact information

Semtech Corporation
Wireless, Sensing & Timing Products Division
200 Flynn Road, Camarillo, CA 93012
Phone: (805) 498-2111 Fax: (805) 498-3804
E-mail: sales@semtech.com
support_rf@semtech.com
Internet: http://www.semtech.com