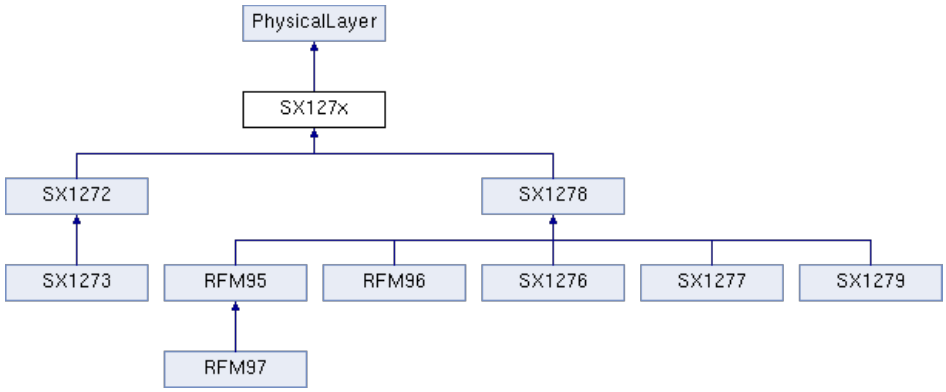
SX127x Class Reference abstract

Base class for **SX127x** series. All derived classes for **SX127x** (e.g. **SX1278** or **SX1272**) inherit from this base class. This class should not be instantiated directly from Arduino sketch, only from its derived classes. More...

`#include <SX127x.h>`

Inheritance diagram for SX127x:



Public Member Functions

	SX127x (**Module** *mod) Default constructor. Called internally when creating new LoRa instance. More...
Module *	**getMod** ()
int16_t	**begin** (uint8_t chipVersion, uint8_t syncWord, uint16_t preambleLength) Initialization method. Will be called with appropriate parameters when calling initialization method from derived class. More...
virtual void	**reset** ()=0 Reset method. Will reset the chip to the default state using RST pin. Declared pure virtual since **SX1272** and **SX1278** implementations differ.
int16_t	**beginFSK** (uint8_t chipVersion, float br, float freqDev, float rxBw, uint16_t preambleLength, bool enableOOK) Initialization method for FSK modem. Will be called with appropriate parameters when calling FSK initialization method from derived class. More...
int16_t	**transmit** (uint8_t *data, size_t len, uint8_t addr=0) override Binary transmit method. Will transmit arbitrary binary data up to 255 bytes long using LoRa or up to 63 bytes using FSK modem. For overloads to transmit Arduino String or C-string, see **PhysicalLayer::transmit**. More...
int16_t	**receive** (uint8_t *data, size_t len) override Binary receive method. Will attempt to receive arbitrary binary data up to 255 bytes long using LoRa or up to 63 bytes using FSK modem. For overloads to receive Arduino String, see **PhysicalLayer::receive**. More...
int16_t	**scanChannel** () Performs scan for valid LoRa preamble in the current channel. More...
int16_t	**sleep** () Sets the LoRa module to sleep to save power. Module will not be able to transmit or receive any data while in sleep mode. Module will wake up automatically when methods like transmit or receive are called. More...
int16_t	**standby** () override Sets the LoRa module to standby. More...
int16_t	**transmitDirect** (uint32_t frf=0) override Enables direct transmission mode on pins DIO1 (clock) and DIO2 (data). While in direct mode, the module will not be able to transmit or receive packets. Can only be activated in FSK mode. More...
int16_t	**receiveDirect** () override Enables direct reception mode on pins DIO1 (clock) and DIO2 (data). While in direct mode, the module will not be able to transmit or receive packets. Can only be activated in FSK mode. More...

int16_t **packetMode** ()
Disables direct mode and enables packet mode, allowing the module to receive packets. Can only be activated in FSK mode. More...

void **setDio0Action** (void(*func)(void))
Set interrupt service routine function to call when DIO0 activates. More...

void **clearDio0Action** ()
Clears interrupt service routine to call when DIO0 activates.

void **setDio1Action** (void(*func)(void))
Set interrupt service routine function to call when DIO1 activates. More...

void **clearDio1Action** ()
Clears interrupt service routine to call when DIO1 activates.

int16_t **startTransmit** (uint8_t *data, size_t len, uint8_t addr=0) override
Interrupt-driven binary transmit method. Will start transmitting arbitrary binary data up to 255 bytes long using LoRa or up to 63 bytes using FSK modem. More...

int16_t **startReceive** (uint8_t len=0, uint8_t mode=RADIOLIB_SX127X_RXCONTINUOUS)
Interrupt-driven receive method. DIO0 will be activated when full valid packet is received. More...

int16_t **readData** (uint8_t *data, size_t len) override
Reads data that was received after calling startReceive method. This method reads len characters. More...

int16_t **startChannelScan** ()
Interrupt-driven channel activity detection method. DIO0 will be activated when LoRa preamble is detected. DIO1 will be activated if there's no preamble detected before timeout. More...

int16_t **setSyncWord** (uint8_t syncWord)
Sets LoRa sync word. Only available in LoRa mode. More...

int16_t **setCurrentLimit** (uint8_t currentLimit)
Sets current limit for over current protection at transmitter amplifier. Allowed values range from 45 to 120 mA in 5 mA steps and 120 to 240 mA in 10 mA steps. More...

int16_t **setPreambleLength** (uint16_t preambleLength)
Sets LoRa or FSK preamble length. Allowed values range from 6 to 65535 in LoRa mode or 0 to 65535 in FSK mode. More...

float **getFrequencyError** (bool autoCorrect=false)
Gets frequency error of the latest received packet. More...

float **getAFCError** ()
Gets current AFC error. More...

float **getSNR** ()
Gets signal-to-noise ratio of the latest received packet. Only available in LoRa mode. More...

float **getDataRate** () const
Get data rate of the latest transmitted packet. More...

int16_t **setBitRate** (float br)
Sets FSK bit rate. Allowed values range from 1.2 to 300 kbps. Only available in FSK mode. More...

int16_t **setFrequencyDeviation** (float freqDev) override
Sets FSK frequency deviation from carrier frequency. Allowed values depend on bit rate setting and must be lower than 200 kHz. Only available in FSK mode. More...

int16_t **setRxBandwidth** (float rxBw)
Sets FSK receiver bandwidth. Allowed values range from 2.6 to 250 kHz. Only available in FSK mode. More...

int16_t **setAFCBandwidth** (float afcBw)
Sets FSK automatic frequency correction bandwidth. Allowed values range from 2.6 to 250 kHz. Only available in FSK mode. More...

int16_t **setAFC** (bool isEnabled)
Enables or disables FSK automatic frequency correction(AFC) More...

int16_t **setAFCAGCTrigger** (uint8_t trigger)
Controls trigger of AFC and AGC. More...

int16_t **setSyncWord** (uint8_t *syncWord, size_t len)
Sets FSK sync word. Allowed sync words are up to 8 bytes long and can not contain null bytes. Only available in FSK mode. More...

int16_t **setNodeAddress** (uint8_t nodeAddr)
Sets FSK node address. Calling this method will enable address filtering. Only available in FSK mode. More...

int16_t **setBroadcastAddress** (uint8_t broadAddr)
Sets FSK broadcast address. Calling this method will enable address filtering. Only available in FSK mode. More...

int16_t **disableAddressFiltering** ()
Disables FSK address filtering. More...

int16_t **setOOK** (bool enableOOK)
Enables/disables OOK modulation instead of FSK. More...

int16_t **setOokThresholdType** (uint8_t type)
Selects the type of threshold in the OOK data slicer. More...

int16_t **setOokPeakThresholdDecrement** (uint8_t value)
Period of decrement of the RSSI threshold in the OOK demodulator. More...

int16_t **setOokFixedOrFloorThreshold** (uint8_t value)
Fixed threshold for the Data Slicer in OOK mode or floor threshold for the Data Slicer in OOK when Peak mode is used. More...

int16_t **setOokPeakThresholdStep** (uint8_t value)
Size of each decrement of the RSSI threshold in the OOK demodulator. More...

int16_t **enableBitSync** ()
Enable Bit synchronizer. More...

int16_t **disableBitSync** ()
Disable Bit synchronizer (not allowed in Packet mode). More...

size_t **getPacketLength** (bool update=true) override
Query modem for the packet length of received payload. More...

int16_t **fixedPacketLengthMode** (uint8_t len=RADIOLIB_SX127X_MAX_PACKET_LENGTH_FSK)
Set modem in fixed packet length mode. Available in FSK mode only. More...

int16_t **variablePacketLengthMode** (uint8_t maxLen=RADIOLIB_SX127X_MAX_PACKET_LENGTH_FSK)
Set modem in variable packet length mode. Available in FSK mode only. More...

int16_t **setRSSIConfig** (uint8_t smoothingSamples, int8_t offset=0)
Sets RSSI measurement configuration in FSK mode. More...

int16_t **setEncoding** (uint8_t encoding) override
Sets transmission encoding. Only available in FSK mode. Allowed values are RADIOLIB_ENCODING_NRZ, RADIOLIB_ENCODING_MANCHESTER and RADIOLIB_ENCODING_WHITENING. More...

uint16_t **getIRQFlags** ()
Reads currently active IRQ flags, can be used to check which event caused an interrupt. In LoRa mode, this is the content of SX127X_REG_IRQ_FLAGS register. In FSK mode, this is the contents of SX127X_REG_IRQ_FLAGS_2 (MSB) and SX127X_REG_IRQ_FLAGS_1 (LSB) registers. More...

uint8_t **getModemStatus** ()
Reads modem status. Only available in LoRa mode. More...

int8_t **getTempRaw** ()
Reads uncalibrated temperature value. This function will change operating mode and should not be called during Tx, Rx or CAD. More...

void **setRfSwitchPins** (RADIOLIB_PIN_TYPE rxEn, RADIOLIB_PIN_TYPE txEn)
Some modules contain external RF switch controlled by two pins. This function gives RadioLib control over those two pins to automatically switch Rx and Tx state. When using automatic RF switch control, DO NOT change the pin mode of rxEn or txEn from Arduino sketch! More...

uint8_t **randomByte** ()
Get one truly random byte from RSSI noise. More...

int16_t **getChipVersion** ()

	Read version SPI register. Should return SX1278_CHIP_VERSION (0x12) or SX1272_CHIP_VERSION (0x22) if **SX127x** is connected and working. More...
int16_t	**invertIQ** (bool invertIQ) Enables/disables Invert the LoRa I and Q signals. More...
void	**setDirectAction** (void(*func)(void)) Set interrupt service routine function to call when data bit is receveid in direct mode. More...
void	**readBit** (RADIOLIB_PIN_TYPE pin) Function to read and process data bit in direct reception mode. More...
int16_t	**setFHSSHoppingPeriod** (uint8_t freqHoppingPeriod) Sets the hopping period and enables FHSS. More...
uint8_t	**getFHSSHoppingPeriod** (void) Gets FHSS hopping period. More...
uint8_t	**getFHSSChannel** (void) Gets the FHSS channel in use. More...
void	**clearFHSSInt** (void) Clear the FHSS interrupt.
int16_t	**transmit** (__FlashStringHelper *fstr, uint8_t addr=0) Arduino Flash String transmit method. More...
int16_t	**transmit** (String &str, uint8_t addr=0) Arduino String transmit method. More...
int16_t	**transmit** (const char *str, uint8_t addr=0) C-string transmit method. More...
virtual int16_t	**transmit** (uint8_t *data, size_t len, uint8_t addr=0)=0 Binary transmit method. Must be implemented in module class. More...
int16_t	**receive** (String &str, size_t len=0) Arduino String receive method. More...
virtual int16_t	**receive** (uint8_t *data, size_t len)=0 Binary receive method. Must be implemented in module class. More...
int16_t	**startTransmit** (String &str, uint8_t addr=0) Interrupt-driven Arduino String transmit method. Unlike the standard transmit method, this one is non-blocking. Interrupt pin will be activated when transmission finishes. More...
int16_t	**startTransmit** (const char *str, uint8_t addr=0) Interrupt-driven Arduino String transmit method. Unlike the standard transmit method, this one is non-blocking. Interrupt pin will be activated when transmission finishes. More...
virtual int16_t	**startTransmit** (uint8_t *data, size_t len, uint8_t addr=0)=0 Interrupt-driven binary transmit method. More...
int16_t	**readData** (String &str, size_t len=0) Reads data that was received after calling startReceive method. More...
virtual int16_t	**readData** (uint8_t *data, size_t len)=0 Reads data that was received after calling startReceive method. More...

▸ **Public Member Functions inherited from PhysicalLayer**

Detailed Description

Base class for **SX127x** series. All derived classes for **SX127x** (e.g. **SX1278** or **SX1272**) inherit from this base class. This class should not be instantiated directly from Arduino sketch, only from its derived classes.

Constructor & Destructor Documentation

◆ SX127x()

SX127x::SX127x (**Module** * mod)

Default constructor. Called internally when creating new LoRa instance.

Parameters

mod Instance of **Module** that will be used to communicate with the LoRa chip.

Member Function Documentation

◆ begin()

```
int16_t SX127x::begin ( uint8_t  chipVersion,
                        uint8_t  syncWord,
                        uint16_t preambleLength
                      )
```

Initialization method. Will be called with appropriate parameters when calling initialization method from derived class.

Parameters

chipVersion	Value in SPI version register. Used to verify the connection and hardware version.
syncWord	LoRa sync word.
preambleLength	Length of LoRa transmission preamble in symbols.

Returns

Status Codes

◆ beginFSK()

```
int16_t SX127x::beginFSK ( uint8_t  chipVersion,
                           float    br,
                           float    freqDev,
                           float    rxBw,
                           uint16_t preambleLength,
                           bool     enableOOK
                         )
```

Initialization method for FSK modem. Will be called with appropriate parameters when calling FSK initialization method from derived class.

Parameters

chipVersion	Value in SPI version register. Used to verify the connection and hardware version.
br	Bit rate of the FSK transmission in kbps (kilobits per second).
freqDev	Frequency deviation of the FSK transmission in kHz.
rxBw	Receiver bandwidth in kHz.
preambleLength	Length of FSK preamble in bits.
enableOOK	Flag to specify OOK mode. This modulation is similar to FSK.

Returns

Status Codes

◆ disableAddressFiltering()

int16_t SX127x::disableAddressFiltering ( )

Disables FSK address filtering.

Returns

Status Codes

◆ disableBitSync()

int16_t SX127x::disableBitSync ( )

Disable Bit synchronizer (not allowed in Packet mode).

Returns

Status Codes

◆ enableBitSync()

int16_t SX127x::enableBitSync ( )

Enable Bit synchronizer.

Returns

Status Codes

◆ fixedPacketLengthMode()

int16_t SX127x::fixedPacketLengthMode (uint8_t len = `RADIOLIB_SX127X_MAX_PACKET_LENGTH_FSK`)

Set modem in fixed packet length mode. Available in FSK mode only.

Parameters

len Packet length.

Returns

Status Codes

◆ getAFCError()

float SX127x::getAFCError ( )

Gets current AFC error.

Returns

Frequency offset from RF in Hz if AFC is enabled and triggered, zero otherwise.

◆ getChipVersion()

int16_t SX127x::getChipVersion ( )

Read version SPI register. Should return SX1278_CHIP_VERSION (0x12) or SX1272_CHIP_VERSION (0x22) if **SX127x** is connected and working.

Returns

Version register contents or **Status Codes**

◆ getDataRate()

float SX127x::getDataRate ( ) const

Get data rate of the latest transmitted packet.

Returns

Last packet data rate in bps (bits per second).

◆ getFHSSChannel()

uint8_t SX127x::getFHSSChannel (void)

Gets the FHSS channel in use.

Returns

6 bit channel number

◆ getFHSSHoppingPeriod()

uint8_t SX127x::getFHSSHoppingPeriod (void)

Gets FHSS hopping period.

Returns

8 bit period

◆ getFrequencyError()

float SX127x::getFrequencyError (bool autoCorrect = `false`)

Gets frequency error of the latest received packet.

Parameters

autoCorrect When set to true, frequency will be automatically corrected.

Returns

Frequency error in Hz.

◆ getIRQFlags()

uint16_t SX127x::getIRQFlags ( )

Reads currently active IRQ flags, can be used to check which event caused an interrupt. In LoRa mode, this is the content of SX127X_REG_IRQ_FLAGS register. In FSK mode, this is the contents of SX127X_REG_IRQ_FLAGS_2 (MSB) and SX127X_REG_IRQ_FLAGS_1 (LSB) registers.

Returns

IRQ flags.

◆ getModemStatus()

uint8_t SX127x::getModemStatus ( )

Reads modem status. Only available in LoRa mode.

Returns

Modem status.

◆ getPacketLength()

size_t SX127x::getPacketLength (bool update = `true`) override virtual

Query modem for the packet length of received payload.

Parameters

update Update received packet length. Will return cached value when set to false.

Returns

Length of last received packet in bytes.

Implements **PhysicalLayer**.

◆ getSNR()

float SX127x::getSNR ( )

Gets signal-to-noise ratio of the latest received packet. Only available in LoRa mode.

Returns

Last packet signal-to-noise ratio (SNR).

◆ getTempRaw()

int8_t SX127x::getTempRaw ( )

Reads uncalibrated temperature value. This function will change operating mode and should not be called during Tx, Rx or CAD.

Returns

Uncalibrated temperature sensor reading.

◆ invertIQ()

int16_t SX127x::invertIQ (bool invertIQ)

Enables/disables Invert the LoRa I and Q signals.

Parameters

invertIQ Enable (true) or disable (false) LoRa I and Q signals.

Returns

Status Codes

◆ packetMode()

int16_t SX127x::packetMode ( )

Disables direct mode and enables packet mode, allowing the module to receive packets. Can only be activated in FSK mode.

Returns

Status Codes

◆ randomByte()

uint8_t SX127x::randomByte ( ) virtual

Get one truly random byte from RSSI noise.

Returns

TRNG byte.

Implements **PhysicalLayer**.

◆ readBit()

void SX127x::readBit (RADIOLIB_PIN_TYPE pin) virtual

Function to read and process data bit in direct reception mode.

Parameters

pin Pin on which to read.

Implements **PhysicalLayer**.

◆ readData() [1/3]

int16_t PhysicalLayer::readData

Reads data that was received after calling startReceive method.

Parameters

str Address of Arduino String to save the received data.

len Expected number of characters in the message. When set to 0, the packet length will be retreived automatically. When more bytes than received are requested, only the number of bytes requested will be returned.

Returns

Status Codes

◆ readData() [2/3]

int16_t SX127x::readData (uint8_t * data,
size_t len
) override virtual

Reads data that was received after calling startReceive method. This method reads len characters.

Parameters

data Pointer to array to save the received binary data.

len Number of bytes that will be read. When set to 0, the packet length will be retreived automatically. When more bytes than received are requested, only the number of bytes requested will be returned.

Returns

Status Codes

Implements **PhysicalLayer**.

◆ readData() [3/3]

virtual int16_t PhysicalLayer::readData

Reads data that was received after calling startReceive method.

Parameters

data Pointer to array to save the received binary data.

len Number of bytes that will be read. When set to 0, the packet length will be retreived automatically. When more bytes than received are requested, only the number of bytes requested will be returned.

Returns

Status Codes

◆ receive() [1/3]

int16_t PhysicalLayer::receive

Arduino String receive method.

Parameters

str Address of Arduino String to save the received data.

len Expected number of characters in the message. Leave as 0 if expecting a unknown size packet

Returns

Status Codes

◆ receive() [2/3]

int16_t SX127x::receive (uint8_t * data,
size_t len
) override virtual

Binary receive method. Will attempt to receive arbitrary binary data up to 255 bytes long using LoRa or up to 63 bytes using FSK modem. For overloads to receive Arduino String, see **PhysicalLayer::receive**.

Parameters

data Pointer to array to save the received binary data.

len Number of bytes that will be received. Must be known in advance for binary transmissions.

Returns

Status Codes

Implements **PhysicalLayer**.

◆ receive() [3/3]

virtual int16_t PhysicalLayer::receive

Binary receive method. Must be implemented in module class.

Parameters

data Pointer to array to save the received binary data.

len Number of bytes that will be received. Must be known in advance for binary transmissions.

Returns

Status Codes

◆ receiveDirect()

int16_t SX127x::receiveDirect ( ) override virtual

Enables direct reception mode on pins DIO1 (clock) and DIO2 (data). While in direct mode, the module will not be able to transmit or receive packets. Can only be activated in FSK mode.

Returns

Status Codes

Implements **PhysicalLayer**.

◆ scanChannel()

int16_t SX127x::scanChannel ( )

Performs scan for valid LoRa preamble in the current channel.

Returns

Status Codes

◆ setAFC()

int16_t SX127x::setAFC (bool isEnabled)

Enables or disables FSK automatic frequency correction(AFC)

Parameters

isEnabled AFC enabled or disabled

Returns

Status Codes

◆ setAFCAGCTrigger()

int16_t SX127x::setAFCAGCTrigger (uint8_t trigger)

Controls trigger of AFC and AGC.

Parameters

trigger one from SX127X_RX_TRIGGER_NONE, SX127X_RX_TRIGGER_RSSI_INTERRUPT, SX127X_RX_TRIGGER_PREAMBLE_DETECT, SX127X_RX_TRIGGER_BOTH

Returns

Status Codes

◆ setAFCBandwidth()

int16_t SX127x::setAFCBandwidth (float afcBw)

Sets FSK automatic frequency correction bandwidth. Allowed values range from 2.6 to 250 kHz. Only available in FSK mode.

Parameters

rxBw Receiver AFC bandwidth to be set (in kHz).

Returns

Status Codes

◆ setBitRate()

int16_t SX127x::setBitRate (float br)

Sets FSK bit rate. Allowed values range from 1.2 to 300 kbps. Only available in FSK mode.

Parameters

br Bit rate to be set (in kbps).

Returns

Status Codes

Todo:

fractional part of bit rate setting (not in OOK)

◆ setBroadcastAddress()

int16_t SX127x::setBroadcastAddress (uint8_t broadAddr)

Sets FSK broadcast address. Calling this method will enable address filtering. Only available in FSK mode.

Parameters

broadAddr Broadcast address to be set.

Returns

Status Codes

◆ setCurrentLimit()

int16_t SX127x::setCurrentLimit (uint8_t currentLimit)

Sets current limit for over current protection at transmitter amplifier. Allowed values range from 45 to 120 mA in 5 mA steps and 120 to 240 mA in 10 mA steps.

Parameters

currentLimit Current limit to be set (in mA).

Returns

Status Codes

◆ setDio0Action()

void SX127x::setDio0Action (void(*)(void) func)

Set interrupt service routine function to call when DIO0 activates.

Parameters

func Pointer to interrupt service routine.

◆ setDio1Action()

void SX127x::setDio1Action (void(*)(void) func)

Set interrupt service routine function to call when DIO1 activates.

Parameters

func Pointer to interrupt service routine.

◆ setDirectAction()

void SX127x::setDirectAction (void(*)(void) func) virtual

Set interrupt service routine function to call when data bit is receveid in direct mode.

Parameters

func Pointer to interrupt service routine.

Implements **PhysicalLayer**.

◆ setEncoding()

int16_t SX127x::setEncoding (uint8_t encoding) override virtual

Sets transmission encoding. Only available in FSK mode. Allowed values are RADIOLIB_ENCODING_NRZ, RADIOLIB_ENCODING_MANCHESTER and RADIOLIB_ENCODING_WHITENING.

Parameters

encoding Encoding to be used.

Returns

Status Codes

Implements **PhysicalLayer**.

◆ setFHSSHoppingPeriod()

int16_t SX127x::setFHSSHoppingPeriod (uint8_t freqHoppingPeriod)

Sets the hopping period and enables FHSS.

Parameters

freqHoppingPeriod Integer multiple of symbol periods between hops

Returns

Status Codes

◆ setFrequencyDeviation()

int16_t SX127x::setFrequencyDeviation (float freqDev) override virtual

Sets FSK frequency deviation from carrier frequency. Allowed values depend on bit rate setting and must be lower than 200 kHz. Only available in FSK mode.

Parameters

freqDev Frequency deviation to be set (in kHz).

Returns

Status Codes

Implements **PhysicalLayer**.

◆ setNodeAddress()

int16_t SX127x::setNodeAddress (uint8_t nodeAddr)

Sets FSK node address. Calling this method will enable address filtering. Only available in FSK mode.

Parameters

nodeAddr Node address to be set.

Returns

Status Codes

◆ setOOK()

int16_t SX127x::setOOK (bool enableOOK)

Enables/disables OOK modulation instead of FSK.

Parameters

enableOOK Enable (true) or disable (false) OOK.

Returns

Status Codes

◆ setOokFixedOrFloorThreshold()

int16_t SX127x::setOokFixedOrFloorThreshold (uint8_t value)

Fixed threshold for the Data Slicer in OOK mode or floor threshold for the Data Slicer in OOK when Peak mode is used.

Parameters

value Threshold level in steps of 0.5 dB.

Returns

Status Codes

◆ setOokPeakThresholdDecrement()

int16_t SX127x::setOokPeakThresholdDecrement (uint8_t value)

Period of decrement of the RSSI threshold in the OOK demodulator.

Parameters

value Use defines RADIOLIB_SX127X_OOK_PEAK_THRESH_DEC_X_X_CHIP

Returns

Status Codes

◆ setOokPeakThresholdStep()

int16_t SX127x::setOokPeakThresholdStep (uint8_t value)

Size of each decrement of the RSSI threshold in the OOK demodulator.

Parameters

value Step size: RADIOLIB_SX127X_OOK_PEAK_THRESH_STEP_0_5_DB (default),
RADIOLIB_SX127X_OOK_PEAK_THRESH_STEP_1_0_DB, RADIOLIB_SX127X_OOK_PEAK_THRESH_STEP_1_5_DB,
RADIOLIB_SX127X_OOK_PEAK_THRESH_STEP_2_0_DB, RADIOLIB_SX127X_OOK_PEAK_THRESH_STEP_3_0_DB,
RADIOLIB_SX127X_OOK_PEAK_THRESH_STEP_4_0_DB, RADIOLIB_SX127X_OOK_PEAK_THRESH_STEP_5_0_DB,
RADIOLIB_SX127X_OOK_PEAK_THRESH_STEP_6_0_DB

Returns

Status Codes

◆ setOokThresholdType()

int16_t SX127x::setOokThresholdType (uint8_t type)

Selects the type of threshold in the OOK data slicer.

Parameters

type Threshold type: SX127X_OOK_THRESH_PEAK(default), SX127X_OOK_THRESH_FIXED, SX127X_OOK_THRESH_AVERAGE

Returns

Status Codes

◆ setPreambleLength()

```
int16_t SX127x::setPreambleLength ( uint16_t  preambleLength )
```

Sets LoRa or FSK preamble length. Allowed values range from 6 to 65535 in LoRa mode or 0 to 65535 in FS

Parameters

preambleLength Preamble length to be set (in symbols when in LoRa mode or bits in FSK mode).

Returns

Status Codes

◆ setRfSwitchPins()

```
void SX127x::setRfSwitchPins ( RADIOLIB_PIN_TYPE  rxEn,
                               RADIOLIB_PIN_TYPE  txEn
                             )
```

Some modules contain external RF switch controlled by two pins. This function gives RadioLib control over those two pins to automatically switch Rx and Tx state. When using automatic RF switch control, DO NOT change the pin mode of rxEn or txEn from Arduino sketch!

Parameters

rxEn RX enable pin.

txEn TX enable pin.

◆ setRSSIConfig()

```
int16_t SX127x::setRSSIConfig ( uint8_t  smoothingSamples,
                                int8_t   offset = 0
                              )
```

Sets RSSI measurement configuration in FSK mode.

Parameters

smoothingSamples Number of samples taken to average the RSSI result. numSamples = 2 ^ (1 + smoothingSamples), allowed values are in range 0 (2 samples) - 7 (256 samples)

offset Signed RSSI offset that will be automatically compensated. 1 dB per LSB, defaults to 0, allowed values are in range -16 dB to +15 dB.

Returns

Status Codes

◆ setRxBandwidth()

int16_t SX127x::setRxBandwidth (float rxBw)

Sets FSK receiver bandwidth. Allowed values range from 2.6 to 250 kHz. Only available in FSK mode.

Parameters

rxBw Receiver bandwidth to be set (in kHz).

Returns

Status Codes

◆ setSyncWord() [1/2]

int16_t SX127x::setSyncWord (uint8_t * syncWord,
size_t len
)

Sets FSK sync word. Allowed sync words are up to 8 bytes long and can not contain null bytes. Only available in FSK mode.

Parameters

syncWord Sync word array.

len Sync word length (in bytes).

Returns

Status Codes

◆ setSyncWord() [2/2]

int16_t SX127x::setSyncWord (uint8_t syncWord)

Sets LoRa sync word. Only available in LoRa mode.

Parameters

syncWord Sync word to be set.

Returns

Status Codes

◆ sleep()

int16_t SX127x::sleep ( )

Sets the LoRa module to sleep to save power. Module will not be able to transmit or receive any data while in sleep mode. Module will wake up automatically when methods like transmit or receive are called.

Returns

Status Codes

◆ standby()

int16_t SX127x::standby ( ) override virtual

Sets the LoRa module to standby.

Returns

Status Codes

Implements **PhysicalLayer**.

◆ startChannelScan()

int16_t SX127x::startChannelScan ( )

Interrupt-driven channel activity detection method. DIO0 will be activated when LoRa preamble is detected. DIO1 will be activated if there's no preamble detected before timeout.

Returns

Status Codes

◆ startReceive()

```
int16_t SX127x::startReceive ( uint8_t  len = 0,
                               uint8_t  mode = RADIOLIB_SX127X_RXCONTINUOUS
                             )
```

Interrupt-driven receive method. DIO0 will be activated when full valid packet is received.

Parameters

len Expected length of packet to be received. Required for LoRa spreading factor 6.

mode Receive mode to be used. Defaults to RxContinuous.

Returns

Status Codes

◆ startTransmit() [1/4]

int16_t PhysicalLayer::startTransmit

Interrupt-driven Arduino String transmit method. Unlike the standard transmit method, this one is non-blocking. Interrupt pin will be activated when transmission finishes.

Parameters

str C-string that will be transmitted.

addr Node address to transmit the packet to. Only used in FSK mode.

Returns

Status Codes

◆ startTransmit() [2/4]

int16_t PhysicalLayer::startTransmit

Interrupt-driven Arduino String transmit method. Unlike the standard transmit method, this one is non-blocking. Interrupt pin will be activated when transmission finishes.

Parameters

str Address of Arduino String that will be transmitted.

addr Node address to transmit the packet to. Only used in FSK mode.

Returns

Status Codes

◆ startTransmit() [3/4]

```
int16_t SX127x::startTransmit ( uint8_t *  data,
                                size_t     len,
                                uint8_t    addr = 0
                              )                                override virtual
```

Interrupt-driven binary transmit method. Will start transmitting arbitrary binary data up to 255 bytes long using LoRa or up to 63 bytes using FSK modem.

Parameters

data Binary data that will be transmitted.

len Length of binary data to transmit (in bytes).

addr Node address to transmit the packet to. Only used in FSK mode.

Returns

Status Codes

Implements **PhysicalLayer**.

◆ startTransmit() [4/4]

virtual int16_t PhysicalLayer::startTransmit

Interrupt-driven binary transmit method.

Parameters

data Binary data that will be transmitted.

len Length of binary data to transmit (in bytes).

addr Node address to transmit the packet to. Only used in FSK mode.

Returns

Status Codes

◆ transmit() [1/5]

int16_t PhysicalLayer::transmit

Arduino Flash String transmit method.

Parameters

str Pointer to Arduino Flash String that will be transmitted.

addr Node address to transmit the packet to. Only used in FSK mode.

Returns

Status Codes

◆ transmit() [2/5]

int16_t PhysicalLayer::transmit

C-string transmit method.

Parameters

str C-string that will be transmitted.

addr Node address to transmit the packet to. Only used in FSK mode.

Returns

Status Codes

◆ transmit() [3/5]

int16_t PhysicalLayer::transmit

Arduino String transmit method.

Parameters

str Address of Arduino string that will be transmitted.

addr Node address to transmit the packet to. Only used in FSK mode.

Returns

Status Codes

◆ transmit() [4/5]

```
int16_t SX127x::transmit ( uint8_t *  data,
                           size_t     len,
                           uint8_t    addr = 0
                         )                                    override virtual
```

Binary transmit method. Will transmit arbitrary binary data up to 255 bytes long using LoRa or up to 63 bytes using FSK modem. For overloads to transmit Arduino String or C-string, see **PhysicalLayer::transmit**.

Parameters

data Binary data that will be transmitted.

len Length of binary data to transmit (in bytes).

addr Node address to transmit the packet to. Only used in FSK mode.

Returns

Status Codes

Implements **PhysicalLayer**.

◆ transmit() [5/5]

virtual int16_t PhysicalLayer::transmit

Binary transmit method. Must be implemented in module class.

Parameters

data Binary data that will be transmitted.

len Length of binary data to transmit (in bytes).

addr Node address to transmit the packet to. Only used in FSK mode.

Returns

Status Codes

◆ transmitDirect()

```
int16_t SX127x::transmitDirect ( uint32_t  frf = 0 )          override virtual
```

Enables direct transmission mode on pins DIO1 (clock) and DIO2 (data). While in direct mode, the module will not be able to transmit or receive packets. Can only be activated in FSK mode.

Parameters

frf 24-bit raw frequency value to start transmitting at. Required for quick frequency shifts in RTTY.

Returns

Status Codes

Implements **PhysicalLayer**.

◆ variablePacketLengthMode()

int16_t SX127x::variablePacketLengthMode (uint8_t maxLen = RADIOLIB_SX127X_MAX_PACKET_LENGTH_FSK)

Set modem in variable packet length mode. Available in FSK mode only.

Parameters

len Maximum packet length.

Returns

Status Codes

The documentation for this class was generated from the following files:

- src/modules/SX127x/**SX127x.h**
- src/modules/SX127x/SX127x.cpp