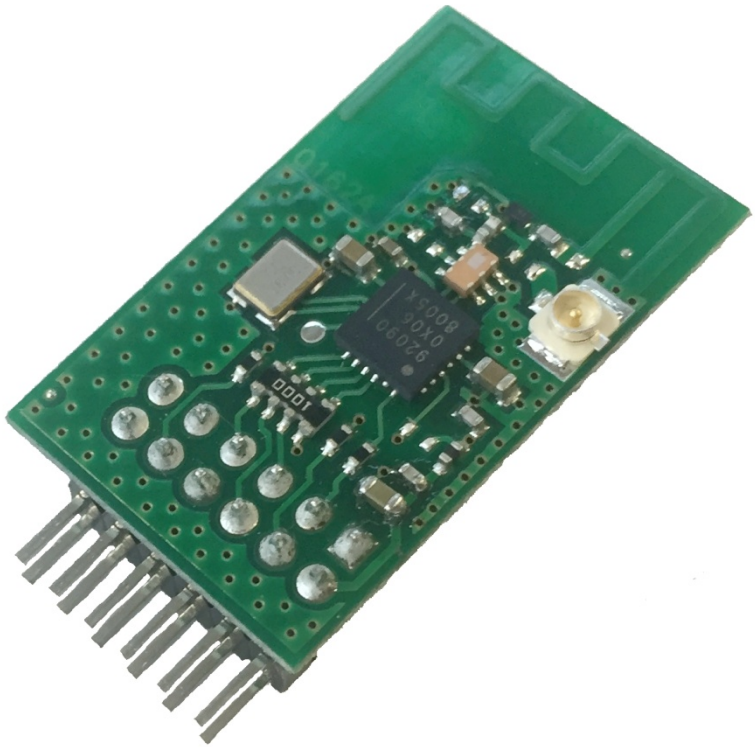


Contents

1. General description

The RTX-24EM is a 1.5kbps to 72kbps low-power, low-voltage 2.4GHz ISM band RF transceiver module ideal for battery operated wireless applications such as wireless sensors and control.
The RTX-24EM module embeds the EM Microelectronic EM9209 Radio SOC device.
The EM9209's built in custom low power microcontroller supports the proprietary wireless protocol links in the license free 2.4000GHz to 2.4835GHz ISM band. It includes a low-IF receiver architecture and uses FSK modulation. A SPI interface provides a simple control of the baseband using an external host microcontroller.
The EM9209 provides two communication modes with normal sensitivity (NS) or high sensitivity (HS) and programmable bit rate from 1.5kbps to 72kbps.

The RTX-24EM module is pin to pin compatible with Aurel previous models XTR CYP 2.4 GHz, XTR VF 2.4 LP, XTR VF 2.4PA-LNA and XTR VF 2.4 HP.

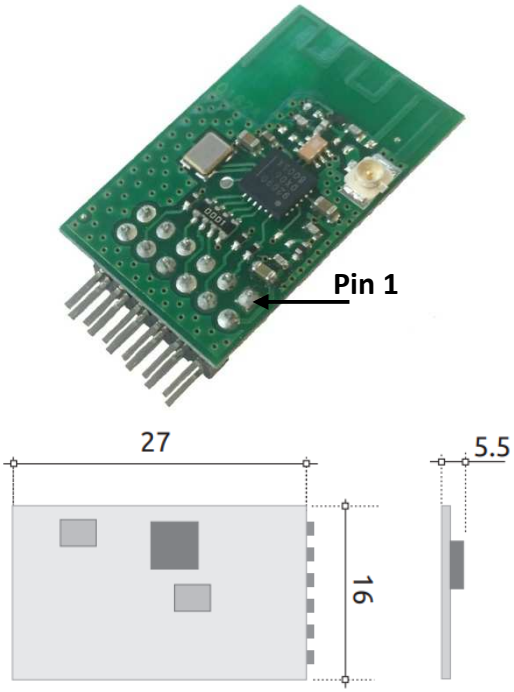
The RTX-24EM module is available in the following variants:

Aurel Code	Technical name reference	Description
650201476G	AUREL RTX-24EM-AI/V	Integrated PIFA antenna, vertical mounting
650201495G	AUREL RTX-24EM-AE/V	UFL connector for external antenna, vertical mounting
650201496G	AUREL RTX-24EM-AI/H	Integrated PIFA antenna, horizontal mounting
650201497G	AUREL RTX-24EM-AE/H	UFL connector for external antenna, horizontal mounting

1.1 Features

- Low voltage: 1,9 to 3,6 V
- Low power: 7mA in RX normal sensitivity mode (NS), 8mA in RX high sensitivity mode (HS).
- TX mode: 11mA at -1dBm, 36mA at +10dBm.
- < 1µA in power down mode.
- High performance:
  - -115dBm sensitivity at 1.5kbps,
  - Programmable output power from -20dBm to +10dBm
- Operating temperatures: -20°C to +70°C
- Flexible interface:
  - SPI interface for microcontrollers
  - Fully programmable link layer

2. RTX-24EM pin out



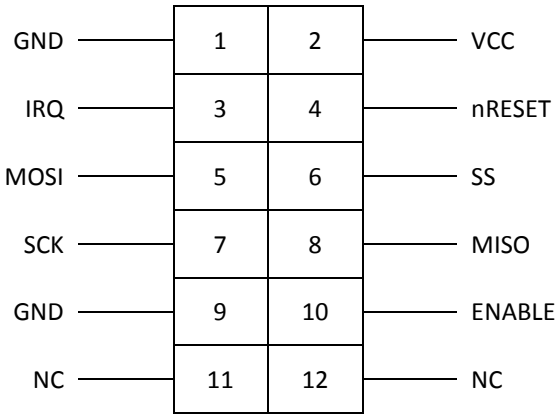


Figure 1

Pin		Description
1	GND	GND connection.
2	VCC	Positive Voltage supply connection.
3	IRQ	Interrupt output signal for external host microcontroller.
4	nRESET[1]	Reset signal (active low) from the external host microcontroller to the radio module. If this pin is forced to low level, the device is resetted and in power down mode.
5	MOSI	Master Out Slave In. SPI signal from the external host microcontroller to the radio module.
6	SS	Slave Select signal (active high) from external host microcontroller to the radio module.
7	SCK	SPI clock from the external host microcontroller to the radio module.
8	MISO	Master In Slave Out. SPI signal from the radio module to the external host microcontroller.
9	GND	GND connection.
10	ENABLE[1]	Enable Pin (active high). If this pin is forced to low level, the device is resetted and in power down mode.

Table 1: Pin description

Note 1: nReset (Pin 4) and ENABLE (Pin 10) are present on the XTR-24EM to maintain the hardware compatibility with both XTR CYP 2.4 GHz and XTR VF 2.4 Aurel modules. They are used for the same functionality (force the RTX-24EM in power down mode) and they are both connected to EN_REG pin of EM9209 radio SoC.
It is therefore recommended to manage only one of them with the host microcontroller.

3. Reference Documentation

The RTX-24EM module embeds the EM Microelectronic EM9209 Radio SOC device.
For information on technical details of the Radio SOC such as register settings, timing, application interfaces and clocking refer to the device data sheet available on the website:
http://www.emmicroelectronic.com

Here below the link of the documentation:
http://www.emmicroelectronic.com/products/wireless-rf/proprietary-protocols/em9209

4. European Reference Standards

The module **RTX-24EM** complies with the European standards **EN 300 440** and **EN 301 489-3** with maximum power supply of 3.6V.

The transceiver must be supplied by a very low voltage safety source protected against short circuits.
The usage of the transceiver is foreseen inside enclosures that assure the overcoming of the rule EN 61000-4-2, not directly applicable to the module itself.
This device is compliant with EN 62479, connected to the electromagnetic field human exposition.

5. Overview

The EM9209 is a low-power, low-voltage, single chip 2.4GHz RF transceiver ideal for battery operated wireless applications. The EM9209 employs a FSK modulation scheme which is directly applied to the 2.4GHz transmitter. RF output power is digitally tuned over a wide range (-20dBm to +10dBm) to optimize current consumption and transmitted power for the application. The on-air transmission rate is digitally programmable from 1.5kbps to 72kbps.
The EM9209 features a fully integrated low-noise, high-sensitivity 2.4GHz front end with -115dBm at 1.5kbps in high sensitivity mode. The integration of an agile frequency synthesizer makes the EM9209 well suited for frequency hopping applications.

The major blocks that build the EM9209 are:

1. the RF transceiver;
2. the digital interface including custom microcontroller;
3. the power management circuitry.

An overview of each of these blocks is provided in this section.

5.1 RF transceiver

The highly integrated multi-channel RF transceiver is ideal for wireless applications in the world-wide, license-free, ISM frequency band at 2.4000GHz to 2.4835GHz. Its robust low-IF architecture and direct FSK modulation scheme are designed for proprietary communication protocols. The EM9209 supports data transmission rates of 1.5kbps to 72kbps for up to 20 channels.

The RF transceiver can be programmed to one of two primary modes:

Transmit mode: the entire transmit-chain is active and the digital baseband data can be up-converted to a 2.4GHz FSK modulated signal.

Receive mode: the frequency synthesizer and the entire receive-chain are active and ready to receive a packet.

The RF transceiver consists of three major subsystems:

- the frequency synthesizer/phase-locked loop (PLL)
- the receiver
- the transmitter.

Each of these is described below.

5.1.1 Frequency synthesizer / Phase-Locked Loop (PLL)

The frequency synthesizer provides an accurate, low jitter (-100 dBc @ 1MHz offset) 2.4GHz RF signal used for both up conversion (in Transmit mode) and down-conversion (in Receive mode). Up to 20 different RF channel frequencies can be synthesized in high sensitivity mode. Additionally, the PLL supports direct FSK modulation for use in the Transmit mode.
An auto-calibration mechanism is included in the PLL to center the VCO control voltage.

5.1.2 Receiver

The receiver achieves high sensitivity (-115dBm at 1.5kbps in high sensitivity (HS) mode) and supports a wide input signal range (up to +10dBm at 2.4GHz). It is comprised of a low noise amplifier (LNA), followed by a down-conversion mixer and an IF-filter. The output of the IF-filter is fed to a limiting-amplifier which feeds the digital FSK demodulators (normal and high sensitivity).
The receiver includes a Received Signal Strength Indicator (RSSI), which can measure the down-converted RF power after the IF filter. The average power on the channel or burst power of a packet can be read via the SPI after the single-shot RSSI measurement has been completed (see Section 9.11).

5.1.3 Transmitter

The transmitter consists of an FSK modulator with a programmable bit-rate (1.5kbps to 72kbps) which is included in the frequency synthesizer and a programmable Power Amplifier (PA) output stage.

5.2 Digital interface

The Digital Interface is shown in Figure 2. It includes:

- A four pin Serial Peripheral Interface (SPI).
- A Custom microcontroller with built in CODEC, FIFO's and timers.
- Two **RAMs** (Program and Registers).
- One **ROM** and its Boot machine.

The SPI can operate at up to 10MHz for reading and writing to the configuration (**RAM2**) and program (**RAM1**) memories.
The custom microcontroller drives an interrupt pin (IRQ) which can be programmed to indicate the status of the EM9209 (e.g., that a packet has been sent or received or that auto calibration has finished). This functionality allows the host microcontroller to complete other operations or even enter its own low power mode.
The RAM memories are reset through internal POR.

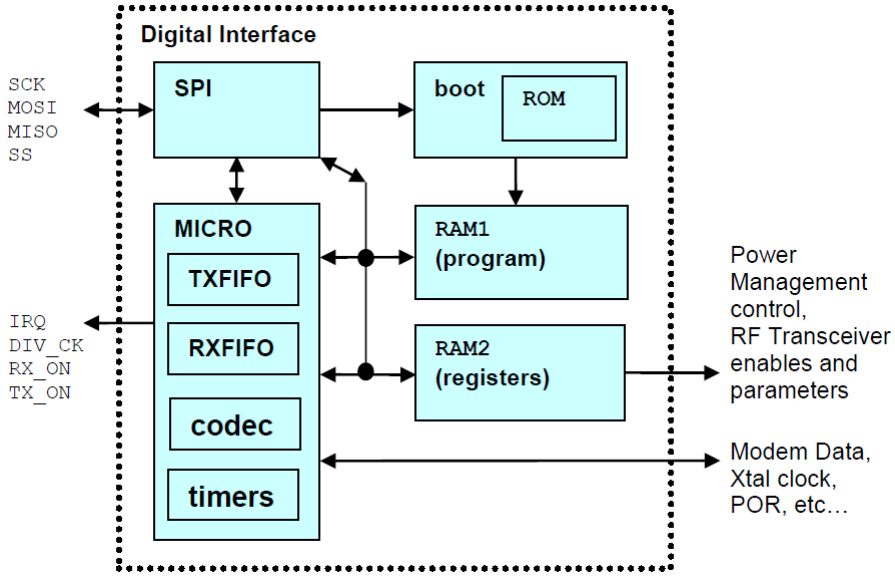


Figure 2: Digital interface

5.2.1 Baseband microcontroller

The baseband custom microcontroller is the central digital control system of the EM9209. It manages all modes of the EM9209 through RAM2 register memory and controls the RF transceiver and TXFIFO or RXFIFO operations. Furthermore, it configures digital data for transmission and processes packets received from the demodulator (what is commonly referred to as the link layer). The microcontroller is able to execute different subroutines which are handling auto-calibration, communication (and FIFO control). Those different subroutines are stored in a ROM memory and are loaded in the RAM1 program memory and activated through SPI interface.

There are various communication subroutines available for EM9209 (either high sensitivity or normal sensitivity). Most communication subroutines will set the EM9209 frequency synthesizer in Receive mode and turn on the Receiver. A simple communication subroutine allows the EM9209 to transmit and to go back to Standby mode with crystal oscillator enabled.

5.2.1.1 In Communication mode:

The **TXFIFO** can be written at any time by the host microcontroller. When the SPI command **Send_TXFIFO** is activated, the internal microcontroller will transmit all the content of the **TXFIFO** (or, depending on the subroutine used, a predefined number of bytes, stored in **RAM2**) on the selected channel at the selected data rate.
The EM9209 will wait for any packet on the selected channel at the selected data rate. When a packet is received, the EM9209 examines the packet size header, and stores the corresponding number of bytes in the **RXFIFO** (or, depending on the subroutine used, a predefined number of bytes, stored in **RAM2**) and sets the IRQ signal Pin high.
The EM9209 **RXFIFO** and **RXFIFO_Size** can be read through SPI.
The IRQ bit can be reset through the SPI **Clear_IRQ** command.

5.2.1.2 In Auto-calibration mode:

The center frequency of the VCO is tuned for a chosen channel frequency.
The result of the auto-calibration is directly written in the VCO center frequency register **VCO_Code[3:0]**.

5.2.1.3 In Standby mode:

The EM9209 control registers (**RAM2**) can be read or written.

5.2.1.4 In RAM2 initialization mode

The EM9209 configures it's **RAM2** to default value and sets IRQ pin high when this action is finished.

5.3 Power management

The power management system of the EM9209 provides the necessary supplies, voltage and current references for reliable operation in all modes. It includes low drop-out voltage regulators (LDO) for the RF transceiver.

5.3.1 RF transceiver supply

There are 2 on-chip regulators, for the transceiver and the synthesizer's analog part, which supply all analog circuits in the RF transceiver. The voltage reference for these regulators is derived from a low noise bandgap circuit. The regulators are enabled individually when needed.

5.3.2 Digital supply

A low power regulator generates the supply for all digital parts in the system (base-band, frequency synthesizer logic and demodulator).

5.3.3 Bias generator

The EM9209 features a bias generator that utilizes a temperature compensated on-chip bandgap reference and a calibrated, temperature dependent, PTAT reference current.

6. Electrical characteristics

6.1 General Operating Conditions

Parameters	Symbol	Min	Typ	Max	Unit
Supply Voltage	V_{CC}	1.9	3.0	3.6	V
Temperature Range	T_A	-20		+70	°C

Table 2: General Operating condition

6.2 Supply currents on Vcc

Operating Mode	Notes	Symbol	Conditions	Min	Typ	Max	Unit
Power Down		I_{CC_PWDOWN}	ENABLE (Pin 10) or nRESET (Pin 4) = 0			1	µA
Standby		I_{CC_STDBY}	ENABLE (Pin 10) or nRESET (Pin 4) = 1 and 26MHz crystal oscillator disabled		140		µA
Auto-calibration		$I_{AUTOCAL}$	Auto-calibration mode		4.2		mA
Transmit	1,2	I_{CC_TX3}	P_{OUT} = 1.1 dBm, 2440MHz		11		mA
		I_{CC_TX7}	P_{OUT} = 10 dBm, 2440MHz		36		mA
Receive	Normal sensitivity,2	I_{CC_RXNS}	2440 MHz		7		mA
	High sensitivity,2	I_{CC_RXHS}	2440 MHz		8		mA

Table 2: Supply currents on V_{CC}

Note 1: See Table 10 for further clarifications.
Note 2: for the version with external antenna the current consumption is measured with 50 ohm load connected to UFL connector. Connecting an antenna the current consumption may be different.

6.3 RF characteristics

Parameter	Conditions	Notes	Symbol	Min	Typ	Max	Unit
General RF conditions							
Operating frequency			fop	2400		2484	MHz
Modulation			mod		FSK		
Data rate	HS 1.5	1,2	DR1	1.2	1.5	2	kbps
	HS 3	1,2	DR2	2	3	4	kbps
	HS 6	1,2	DR3	4	6	8	kbps
	HS 12	1,2	DR4	8	12	16	kbps
	NS 24	1,2	DR5		24		kbps
	NS 48	1,2	DR6		48		kbps
	NS 72	1,2	DR7		72		kbps
Channel spacing			F_{CHW}		4		MHz
Transmitter Operation							
RF Output power on UFL connector (RTX-24EM-AE/H and RTX-24EM-AE/V versions)	Power level = 7	3	P_{RF7}		+10		dBm
	Power level = 6	3	P_{RF6}		+9		dBm
	Power level = 5	3	P_{RF5}		+6		dBm
	Power level = 4	3	P_{RF4}		+2.5		dBm
	Power level = 3	3	P_{RF3}		-1		dBm
	Power level = 2	3	P_{RF2}		-3		dBm
	Power level = 1	3	P_{RF1}		-10		dBm
E.R.P. RF power (RTX-24EM-AI/H and RTX-24EM-AI/V versions)	Power level = 7	3	P_{ERP7}		+8.5		dBm
	Power level = 6	3	P_{ERP6}		+7.5		dBm
	Power level = 5	3	P_{ERP5}		+4.5		dBm
	Power level = 4	3	P_{ERP4}		+1.2		dBm
	Power level = 3	3	P_{ERP3}		-2.5		dBm
	Power level = 2	3	P_{ERP2}		-4.5		dBm
	Power level = 1	3	P_{ERP1}		-11.5		dBm
Receiver Operation							
Sensitivity	HS 1.5	1	$S_{HS1.5}$		-115		dBm
	HS 3	1	S_{HS3}		-113		dBm
	HS 6	1	S_{HS6}		-111		dBm
	HS 12	1	S_{HS12}		-107		dBm
	NS 24	1	S_{NS24}		-100		dBm
	NS 48	1	S_{NS48}		-98		dBm
	NS 72	1	S_{NS72}		-97		dBm

Table 5: RF characteristics

Note 1: HS = High Sensitivity , NS = Normal Sensitivity.
Note 2: Data rate "on air". In case of more than 4 consecutive identical symbols, bit stuffing can reduce this data rate from 100% down to 80% of this value.
Note 3: See Table 10 for more detailed power amplifier settings.

7. Functional modes

This section describes the operational modes of the RTX-24EM module. An example state diagram is given in Figure 3, and each mode is described below. The SPI interface is used to set or change the mode by loading and running the corresponding subroutine. Most transitions are immediate, shorter than the SPI transactions, except for those marked in the figure and listed in Table 5: Timing Characteristics.

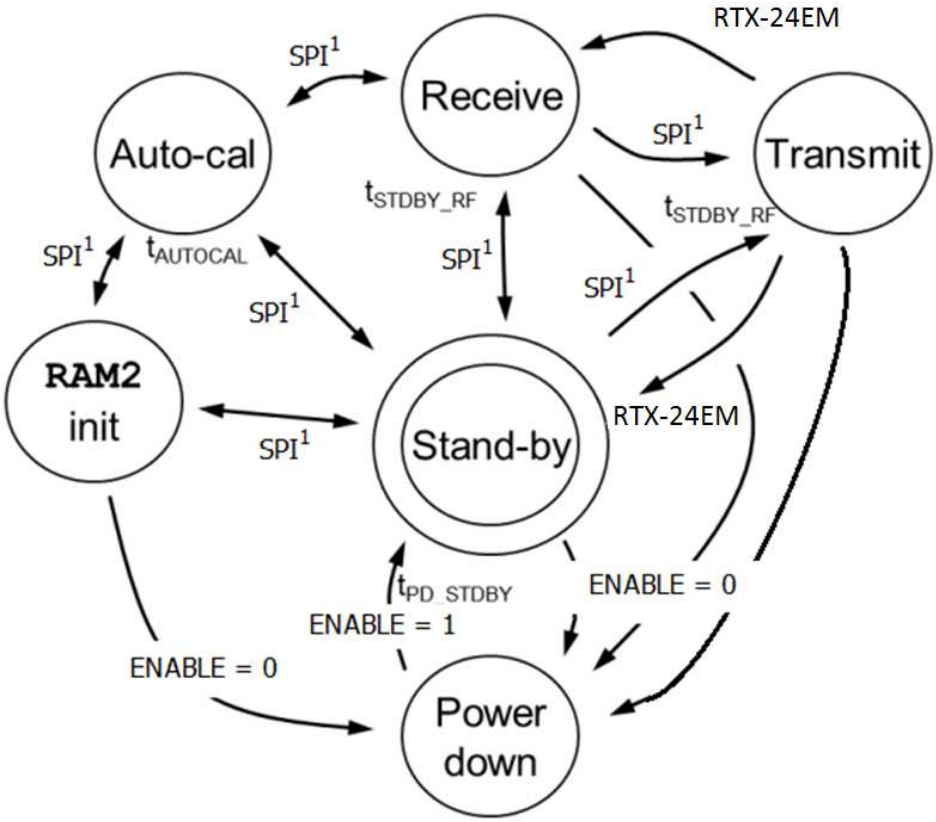


Figure 1: RTX-24EM Functional Overview

Notes:
1. See SPI Operation chapter.

7.1 Power Down

This mode is enabled when ENABLE (Pin 10) or nRESET (Pin 4) is tied to GND. EM9209 internal regulators and the voltage reference are disabled and the supply current is < 1uA.

7.2 Standby Mode

When ENABLE (Pin 10) or nRESET (Pin 4) is high the RTX-24EM wakes up in standby mode at about $t_{PD-STBY}$ seconds (see Table 6). The SPI register memory is then set to 0 and the SPI waits for the host programming. In standby mode all internal circuits are disabled and can be accessed, including the crystal oscillator. The host microcontroller can program the RTX-24EM for any operational mode.

7.3 RAM2 Init

The RTX-24EM can be configured using a 16 registers memory RAM2. RAM2 is resetted to 0 when ENABLE (Pin 10) or nRESET (Pin 4) is set from GND to Vcc. In order to avoid 16 different SPI write cycles, a dedicated subroutine located at **ROM_BOOT_Address = 0** will initialize most RAM2 addresses to their default values. This procedure takes about t_{RAM2_INIT} seconds (see Table 6).

7.4 Auto-cal

VCO center frequency

The RTX-24EM frequency synthesizer has an Auto-calibration mode that must be run periodically by the host microcontroller. This keeps the channel frequency and FSK modulator operating within specification. Analog components in this block are sensitive to temperature variation, therefore performance may degrade or the link may fail if not run periodically. Typically, Auto-calibration should be run when changing channels or if the operating temperature changes by more than 10 to 20 °C. This procedure takes about t_{VCO_CAL} seconds (see Table 6).

PTAT reference current

The internally generated PTAT current can be self calibrated using an internal PTAT generator. This procedure takes about t_{PTAT_CAL} seconds (see Table 6).

7.5 Transmit

In Transmit mode, the RTX-24EM outputs a FSK-modulated packet, it returns to receive mode or standby mode with the crystal oscillator enabled and sets the interrupt pin IRQ high. Depending on the choosen subroutine the RTX-24EM can transmit the whole TXFIFO (till TXFIFO size = 0) or a predefined number of bytes programmed in RAM2.
Transmit mode is activated from Receive mode or from Standby mode (with crystal oscillator enabled) using the SPI command **Send_TXFIFO**.

7.6 Receive

The RTX-24EM in receive mode waits for a FSK-modulated packet. After receiving a suitable packet it sets the IRQ pin high. Depending on the choosen subroutine, it can either read the size of the packet to be received in the header or in RAM2.

Parameter	Symbol	Min	Typ	Max	Unit
Power Down → Standby mode (crystal oscillator OFF)	$t_{PD-STBY}$		100		µs
Standby mode (crystal OFF) → Standby mode Crystal oscillator ON	$t_{STBY-OSC\_ON}$		900		µs
RAM2 Init	t_{RAM2_INIT}		200		µs
VCO Auto calibration	t_{VCO_CAL}	340	500		µs
PTAT current Auto calibration	t_{PTAT_CAL}		150		µs
Standby mode crystal oscillator ON → TX/RX mode	$t_{STBY-RF}$	0.8	1		ms

Table 6: Timing characteristics (Vcc = 3V)

8. Digital interface

The RTX-24EM can be controlled with a 4-wire serial peripheral interface (SPI). The four wires are:
SS Slave Select
SCK Serial Clock
MOSI Serial data in to RTX-24EM
MISO Serial data out of RTX-24EM

The RTX-24EM has a programmable interrupt pin (IRQ).

All internal enables signals and parameters of the RTX-24EM are mapped in a small 16x12 bits memory called RAM2. RAM2 can directly be accessed through SPI and no crystal clock is required.

8.1 SPI operations

The SPI interface is used to read from and to write all the register of the EM9209 embedded on the RTX-24EM module.

SPI operations allow various accesses:

- Memories write and read actions
- EM9209 internal microcontroller commands
- Loading of subroutines in RAM1

A SPI transaction is defined as all of the activity on SCK, MOSI and MISO that occurs between one rising edge of SS and its next falling edge. All the data shall be sent starting with the MSB first.
Not all the commands are encoded on a number of bits multiple of 8. Additional clocks can be sent after the command with no impact on the command decoding. Thus, the EM9209 can be accessed without

problems using an 8-bit wide SPI interface. Each change to MOSI is latched on the rising edge of SCK, and each change to MISO is available on the falling edge of SCK. A timing diagram is shown in Figure 2. Complete timing specifications are given in Table .

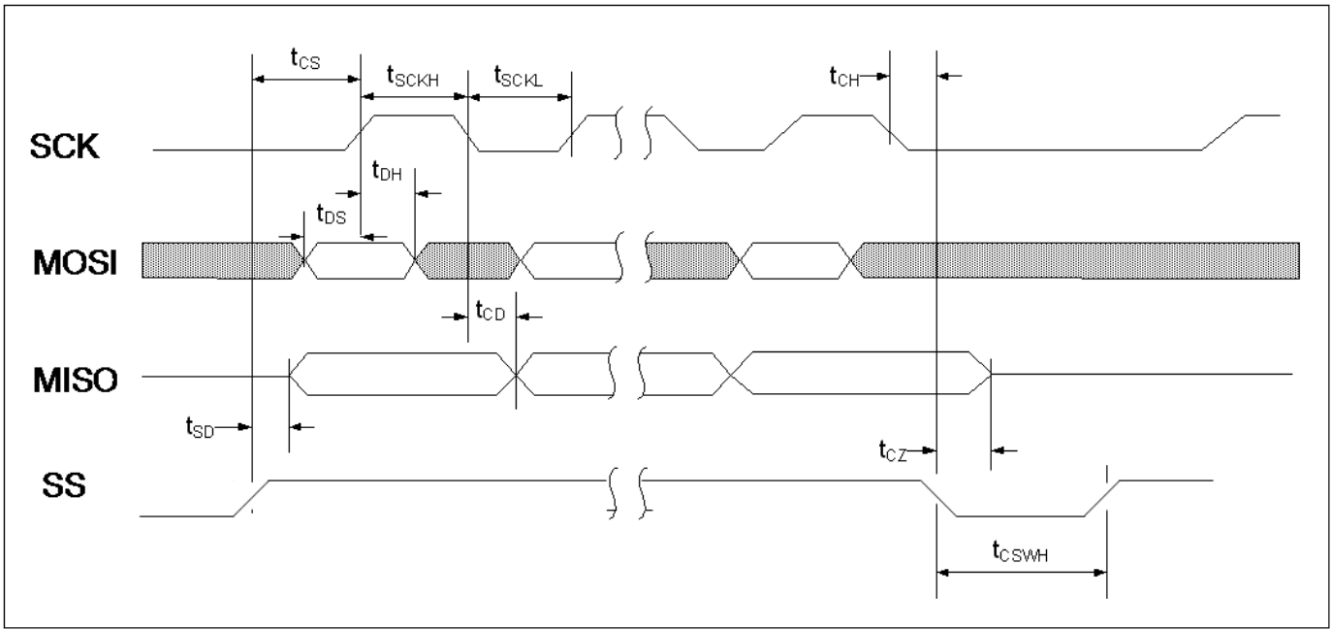


Figure 2: SPI timing diagram

Symbol	Parameters	Min	Max	Units
t_{DS}	MOSI to SCK Setup	20		ns
t_{DH}	SCK to MOSI hold	20		ns
t_{SD}	SS to MISO Valid		30	ns
t_{CD}	SCK to MISO Valid		30	ns
t_{SCKL}	SCK low time	40		ns
t_{SCKH}	SCK high time	40		ns
f_{sck}	SCK frequency	0	10	MHz
t_{CS}	SS to SCK Setup	20		ns
t_{CH}	SCK to SS Hold	20		ns
t_{CSWH}	SS Inactive Time	20		ns
t_{CZ}	SS to MISO High Z		30	ns

Table 7: SPI timing values

For each SPI command, MISO will always give three status bits on the first three SCK cycles.

- As soon as SS goes high, the first status bit (Status[2]) is available on the MISO terminal. This bit is called “Previous_FIFO_Order_Pending” and is high when the microcontroller has not yet processed the previous FIFO order. This process takes a maximum of 8 clock cycles and starts on the falling edge of the SS signal.

- Status[1] reflects the inactivity of the crystal oscillator (0: running).
- Status[0] shows the unlock state of the 2.4GHz LO frequency synthesizer (0: main LO PLL locked).

For connect transmission operation, status[2..0] must be equal to '000'.

8.2 SPI Commands

Read_RXFIFO

MOSI	1	1	0	0	x	x	x	x	x	x	x	x	x	x	x	x
MISO	Status[2..0]			RXFIFO_Size[4..0]					RXFIFO_Data[7..0]							

This command returns the next byte out of the RXFIFO. It also returns the total number of bytes currently available in the RXFIFO (including the one being read).
This SPI operation works together with the internal microcontroller and is functional only when this latter has been started (SPI command Start_Micro) and when the master clock is active (Crystal oscillator must be enabled). The order is taken into account only when SS signal goes down and the RXFIFO size information are sampled when SS is low.

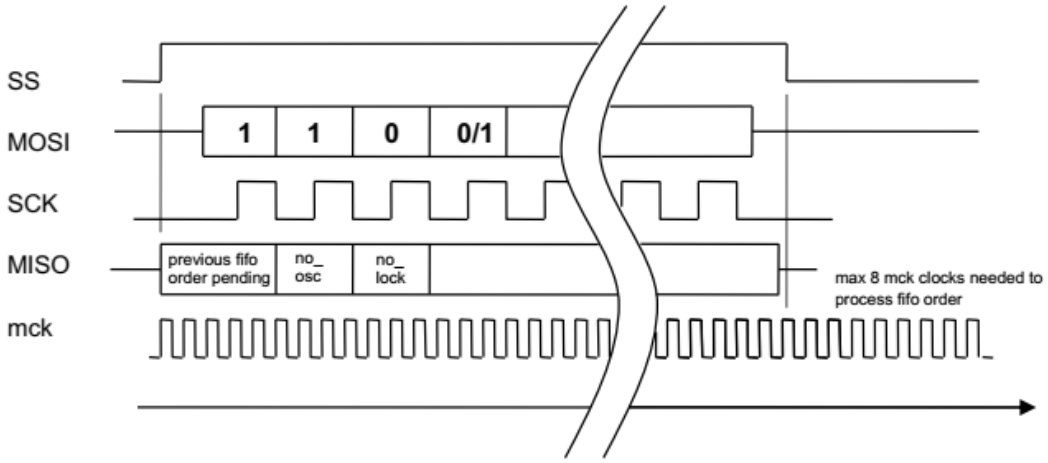


Figure 3: Timing of the SPI Read_RXFIFO / Write_TXFIFO command

Write_TXFIFO

MOSI	1	1	0	1	TXFIFO_Data[7..0]				x	x	x	x
MISO	Status[2..0]			RXFIFO_Size[4..0]				TXFIFO_Size[4..0]			x	x	x

This command writes a byte to the TXFIFO. It also returns the total number of bytes in both FIFOs, not including this one. This SPI operation works together whit the internal microcontroller and is functional only when this latter has been started (SPI command Start_Micro) and when the master clock is active (Crystal oscillator must be enabled). The order is taken into account only when SS signal goes down and the FIFO size information are sampled by *mck* when SS is low.

Read_RXFIFO_Size

MOSI	1	1	1	0	0	0	0	x	x	x	x	x	x	x	x	x
MISO	**Status[2..0]**			x	x	x	x	**RXFIFO_Size[4..0]**					x	x	x	x

This command reads the total number of bytes currently available in the RXFIFO.

Read_TXFIFO_Size

MOSI	1	1	1	0	0	0	1	x	x	x	x	x	x	x	x	x
MISO	**Status[2..0]**			x	x	x	x	**TXFIFO_Size[4..0]**					x	x	x	x

This command reads the total number of bytes currently available in the TXFIFO.

Read_RAM1

MOSI	0	0	1	address[5..0]						x	x	x	x	x	x	x	x	x	x	x
MISO	**Status[2..0]**			x	x	x	x	x	x	data_read[11..0]									x	x	x

This command reads the 12 bits word from the specified address (6bits) of RAM1. This command will put the microcontroller on hold and reset state, until last bit has been processed.

Write_RAM1

MOSI	0	0	0	address[5..0]						data_write[11..0]											x	x	x
MISO	**Status[2..0]**			x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x

This command writes a 12 bits word to the specified address (6 bits) of RAM1. This command will put the microcontroller on hold and reset state until last bit has been processed.

Read_RAM2

MOSI	0	1	1	address[3..0]				x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
MISO	**Status[2..0]**			x	x	x	x	data_read[11..0]											x	x	x	x	x

This command reads the 12 bits word to the specific address (4 bits) of RAM2. This command will put the microcontroller on hold until last bit has been processed.

Write_RAM2

MOSI	0	1	0	address[3..0]				data_write[11..0]													x	x	x	x	x
MISO	**Status[2..0]**			x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x

This command writes a 12 bits word to the specified address (4 bits) of RAM2. This command will put the microcontroller on hold until last bit has been processed.

Reset_Micro

MOSI	1	1	1	0	0	1	0	x
MISO	**Status[2..0]**			x	x	x	x	x

This instruction allows an asynchronous reset of the microcontroller. Never use this command when the micro is running (RAM2 and FIFO's content could be corrupted). Always first stop the micro using SPI command Stop_Micro prior to use Reset_Micro.

Stop_Micro

MOSI	1	1	1	0	0	1	1	x
MISO	**Status[2..0]**			x	x	x	x	x

This command stops the microcontroller.

Start_Micro

MOSI	1	1	1	0	1	0	0	x
MISO	**Status[2..0]**			x	x	x	x	x

This command start the microcontroller and executes the program currently stored in RAM1.

Clear_IRQ

MOSI	1	1	1	0	1	0	1	x
MISO	**Status[2..0]**			x	x	x	x	x

Use this command to reset the IRQ signal. It works only when microcontroller is running.

Send_TXFIFO

MOSI	1	1	1	0	1	1	0	x
MISO	**Status[2..0]**			x	x	x	x	x

This command will send the current contents of the TXFIFO. Depending on the selected subroutine, the program either sends the full content of the FIFO, or the number of bytes specified in RAM2.

Aux_com

MOSI	1	1	1	0	1	1	1	x
MISO	**Status[2..0]**			x	x	x	x	x

This command allows the Channel RSSI to be read and stored to Limit_RSSI[3:0].

ROM_Boot

MOSI	1	1	1	1	0	0	0	ROM_Boot_Address[8..0]								
MISO	Status[2..0]			x	x	x	x	x	x	x	x	x	x	x	x	x

This command copies the 64 12 bits instructions from the specified ROM address to RAM1. This allows for fast initialization of the microcontroller subroutines.
The crystal oscillator must be enabled to perform this operation. Additionally, ROM_Boot command stops and resets the microcontroller.

ROM_Boot0_and_Start

MOSI	1	1	1	1	1	1	1	x
MISO	Status[2..0]			x	x	x	x	x

This command copies the 64 12-bits instructions from the ROM address 0 to RAM1. This allows for fast initialization of the microcontroller subroutines.
The crystal oscillator must be enabled to perform this operation. Additionally, ROM_Boot0_and_Start command resets and starts the microcontroller. It goes in standby mode.

9. Programming interface

This section describes how to program the RTX-24EM by writing to the EM9209 RAM2 or by booting and running in ROM stored subroutines.

9.1 RAM2, RAM1 reset

The EM9209 automatically performs a power on reset to RAM1 and RAM2 after ENABLE (Pin 10) or nRESET (Pin 4) is set to Vcc from GND.
After reset, all RF communication parameters must be reconfigured and the PLL auto calibration cycle must be initiated again.

9.2 RAM2 Initialization

The EM9209 has a dedicated subroutine located at the **ROM_Boot_Address** = **0** which will initialize the **RAM2** memory to its default state (see Section 12).
To perform the **RAM2** initialization:

- ENABLE (Pin 10) or nRESET (Pin 4) must be set to Vcc.
- Crystal oscillator must be started by writing 0b111000000100 (0xE04) in the **RAM2** at the address **0**.
- Quartz activity can be monitored through **Status[1]** by using a simple **Stop_Micro** SPI command, for example. When **Status[1]** has gone low, the **RAM2** initialization subroutine can be executed.
- Use SPI command **Write_RAM1** with address = **13** and data_write = **1184 .** This will initialize RAM2 with RB_Inst_Dis = 1 and this allows the user to manually select the subroutine to be stored in RAM1.
- Use SPI command **ROM_Boot** with argument **ROM_Boot_Address** = **0.**
- Use then the SPI command **Start_Micro**.
- The end of initialization will be signaled with IRQ going high. SPI command **Clear_IRQ** allows clearing the interrupt.

9.3 Internal PTAT current auto calibration

This auto-calibration is used to calibrate the current delivered by the internal PTAT generator (Proportional To Absolute Temperature).
The EM9209 has a dedicated subroutine located at the **ROM_Boot_Address** = **33** which will perform the calibration.
To perform the auto calibration:

- ENABLE (Pin 10) or nRESET (Pin 4) must be set to Vcc.
- Crystal oscillator must be started (see section 9.2)
- RB_Inst_Dis must be set to 1 (if RB_Inst_Dis has been previously set = 1, see section 9.2, this step can be omitted).
- Use SPI command **ROM_Boot** with argument **ROM_Boot_Address** = **33.**
- Use then the SPI command **Start_Micro**.
- The end of auto calibration will be signaled with IRQ going high. SPI command **Clear_IRQ** allows clearing the interrupt.

The calibration of the PTAT is independent of the temperature and only needs to be executed once when the chip is powered. Because this value should be constant during product life, it is possible to do calibration only once and to store **RAM2@5<3:0>** value somewhere in a non-volatile memory. **Write_RAM2** SPI command with this pre-stored value could then be used to set the correct internal PTAT current.

9.4 VCO auto calibration

This auto-calibration is used to calibrate the analog circuits of the PLL. For correct transmission and reception, the PLL should be calibrated at each channel frequency to be used before the link is established or if the operating temperature changes by more than 10 to 20 °C.

The EM9209 has a dedicated subroutine located at the **ROM_Boot_Address** = **64** which will perform the VCO auto calibration.
Auto calibration frequency is set through register **VcoCalibFreq[11:0]** which is located in **RAM1** at the address **53**. It must be programmed through the SPI **Write_RAM1** command to fit the required frequency operation.

VcoCalibFreq[11:0] is:

VcoCalibFreq[7:0] = (round(4259840 / fo) – 1618).
VcoCalibFreq[11:8] = 0b1011 (0xB).

where fo is the RF operating frequency in MHz.

Examples:

fo = 2480, VcoCalibFreq[11:0] = 0xB64.
fo = 2440, VcoCalibFreq[11:0] = 0xB80.
fo = 2400, VcoCalibFreq[11:0] = 0xB9D.

To perform the auto calibration:

- ENABLE (Pin 10) or nRESET (Pin 4) must be set to Vcc.
- Crystal oscillator must be started (see section 9.2)
- RB_Inst_Dis must be set to 1 (if RB_Inst_Dis has been previously set = 1, see section 9.2, this step can be omitted).
- Use SPI command **ROM_Boot** with argument **ROM_Boot_Address** = **64.**
- Use SPI command **Write_RAM1** with address = **53** and data_write = **VcoCalibFreq[11:0].**
- Use then the SPI command **Start_Micro**.
- The end of auto calibration will be signaled with IRQ going high. SPI command **Clear_IRQ** allows clearing the interrupt.

The calibration of the PLL may vary if the external conditions change (e.g., temperature), therefore calibration should be repeated periodically.

9.5 Channel Data Rate

The EM9209 has a programmable channel data rate of 1.5kbps to 72kbps for transmission and reception in normal sensitivity mode.
The channel data rate is set by **R_Bit_Clk[8:0]** (**RAM2@12[8:0]**) and **Ch_Rate[2:0]** (**RAM2@12[11:9]**) as shown in Table 8**.** The complete typical values **RAM2@12[11:0]** is also reported.
In high sensitivity mode, only the 4 slower data rates are available (**Ch_Rate[2:0]** = '000' to '011').

On air bit rate (kbps)	Ch_Rate[2:0]	R_Bit_Clk[8:0]	RAM2@12[11:0]
1.5	000	110000000	0x180
2.99	001	011000000	0x2C0
6.02	010	001011111	0x45F
12.037	011	000101111	0x62F
24.074	100	000010111	0x817
48.15	101	000001011	0xA0B
72.22	110	000000111	0xC07

Table 8: Channel Data Rate

To establish a communication, both linked devices must be set to the same data rate.

9.5.1 Bit stuffing

To improve the receiver's clock recovery, the data transmitted is automatically bit stuffed with a hardwired algorithm. The internal bit stuffing procedure is allowing a maximum of 4 consecutive same symbols to be transmitted. It corresponds to a minimum efficiency of 80%, or a minimum bit rate described by Table 9.

On air bit rate (kbps)	**Minimum bit stuffed data rate (kbps)**
1.5	1.2
3	2.4
6	4.8
12	9.6
24	19.2
48	38.4
72	57.6

Table 9: On air VS Minimum bit stuffed data rates

9.6 RF operating frequency

The channel register sets the center frequency of the transmission channel used by the EM9209.
The channel is set by the **Frequ[16:0]**register.
Frequency should be between 2400MHz and 2484MHz

Frequ[16..0] = (32768 x ((Frequency / fref) – 92)) - FreqCorrection

with

Frequency = desired center frequency in MHz.
fref = 26
FreqCorrection = 125

Examples:

Frequency = 2401.5 Frequ[16..0] = 11848 = 0x2E48.
Frequency = 2441.5, Frequ[16..0]= 62260 = 0xF334.
Frequency = 2477.5, Frequ[16..0]= 107631 = 0x1A46F.

FreqCorrection is used to compensate the 26MHz crystal oscillator (internal to the RTX-24EM) deviation.

The channel step is given by 26MHz / 32768 and is approximately equal to 793Hz.

To establish a communication, both linked devices must be set to the same channel. The host microcontroller can program a channel change, which is validated when SPI signal SS goes down.
Channel spacing of 4 MHz is recommended to limit interference with other RTX-24EM devices operating on adjacent channels.

9.7 Address byte

For proper communication between two devices, the receiving device must set the **Address[7:0]** register to match the transmitting device's **Address[7:0]** register.
The Address byte is used in the packet preamble in order to set the byte start in the bit to byte built-in reconstruction algorithm.
Decimal value of **Address[7:0]** must be different from **0, 48, 51, 99, 102, 146, 153, 204**. To prevent wrong packet synchronization when address Rx and Tx differ by a right or left shift, even address values must be avoided and the most significant bit (MSB) of the first payload byte should be the invert of the address most-significant bit (MSB).
As the preamble to start packet detection consists of three successive address bytes, the reception can be triggered by a payload containing such data, even if the transmit and receive addresses are not equal. For applications where the integrity of the address or the payload is critical, it is recommended to include CRC or error correction bits inside the payload.

9.8 TX Power level

The PA output power can be adjusted to many different levels from -20dBm to +10 dBm.
Table 10 shows 7 typical levels with optimum efficiency.
These levels are set by **I_Pre_PA[4:0]** and **I_PA[4:0]** register bits.
Typical current consumption for each of these power levels is also shown.

			RTX-24EM-AI/H and RTX-24EM-AI/V		RTX-24EM-AE/H and RTX-24EM-AE/V	
Power Level	I_Pre_PA [4:0]	I_PA [4:0]	E.R.P. RF Power (dBm)	Current consumption (mA)	RF Output Power on UFL connector (dBm)	Current consumption (mA)
7	29	18	+8.5	36	+10	36
6	21	5	+7.5	30	+9	30
5	10	2	+4.5	21	+6	21
4	7	1	+1.2	14	+2.5	14
3	4	1	-2.5	11	-1	11
2	3	1	-4.5	10	-3	10
1	1	1	-11.5	8	-10	8

Table 10: TX power level

Note 1 : the European standard EN 300 440 allows maximum +10dBm RF radiated power. This means that, to be compliant with the standard, the device with external antenna requires a 0dB antenna maximum gain for the power level 7, a 1dB antenna maximum gain for power level 6, a 4dB antenna maximum gain for power level 5 and so on.
Note 2 : for the version with external antenna the current consumption is measured with 50 ohm load connected to UFL connector. Connecting an antenna the current consumption may be different.

9.9 Packet (TX and RX) payload

9.9.1 Mode payload size in the header

In this mode (defined by the SPI Command **ROM_Boot** with argument **ROM_Boot_Address** = **256** for HS Mode or **320** for NS Mode), the transmit payload can be up to 31 bytes. A header byte which defines the packet size is added (See Section 13). The payload (and header) are read and written through the SPI command **Read_RXFIFO** and **Write_TXFIFO**.

9.9.2 Mode payload size in RAM2

In this mode (defined by the SPI Command **ROM_Boot** with argument **ROM_Boot_Address** = **128** for HS Mode or **192** for NS Mode), the transmit payload can be up to 32 bytes, but the first byte following the Address byte has to be different from **Address[7:0]**. Payload size is defined in the register **N_Pay[4:0]**

(**RAM2@14[4:0]**). The payload is read and written through the SPI command **Read_RXFIFO** and **Write_TXFIFO**.

9.10 Register TXFIFO and RXFIFO

The EM9209 has two 32 Bytes FIFO's (**TXFIFO** and **RXFIFO**). The transmit **TXFIFO** can be accessed through the SPI command **Write_TXFIFO**. The receive **RXFIFO** can be accessed through the SPI command **Read_RXFIFO**.

The transmit **TXFIFO** must be loaded prior to transmission. The size of the transmit **TXFIFO** is monitored at each write FIFO command or can be viewed at any time through the command **Read_TXFIFO_Size**.

The receive **RXFIFO** can be read after an incoming packet has been received. The size of the receive **RXFIFO** is monitored at each **Read_FIFO** command or can be viewed at any time through the command **Read_RXFIFO_Size**

.

Note: Both **TXFIFO** and **RXFIFO** are managed by the internal microcontroller, which means that this latter must be enabled (**Start_Micro**) and running (crystal oscillator enabled, **Status[1]** = '0'). Also, the SPI command **Reset_Micro** will reset all internal **TXFIFO** and **RXFIFO** pointers to 0.

9.11 Received Signal Strength Indicator (RSSI)

A received signal strength indicator (RSSI) is available through the register **Limit_RSSI[3:0]** located in **RAM2@8[3:0]**. A RSSI measurement can be activated in two different ways:

Packet RSSI

The communication subroutines defined by SPI command **ROM_Boot** with **ROM_Boot_Address** = **128, 192, 256** and **320** will write the RSSI measured during the packet reception to **RAM2** register **Limit_RSSI[3:0]**. Header + Payload size (see Section 13) must be greater than 1 byte to get a correct **Limit_RSSI[3:0]** value.

Channel RSSI

The special subroutine defined by **ROM_Boot** with **ROM_Boot_Address** = **248** will write the measured RSSI value to **RAM2** register **Limit_RSSI[3:0]** when SPI **Aux_com** is activated. This allows the user to determine the amount of RF activity on a given channel. This is useful for determining if there is other RF activity on the channel (e.g., WiFi).
The relationship between the applied RF power at the antenna (PIN) and the value given by the RSSI **Limit_RSSI[3:0]** can be expressed as:

PIN [dBm] = -120dBm + unsigned (**Limit_RSSI[3:0])** * 8dB, for -89 dBm < PIN < -48 dBm.

The low limit of RSSI[3:0] is determined by channel thermic noise at typically 2, corresponding to -89 dBm, while the maximum saturates at 9, at about -48 dBm.
The accuracy of the RSSI is not guaranteed, and is provided for test purposes only.

9.12 Frequency error register : DFT_Mes[7:0]

This Register located in **RAM2@9[7:0]** is updated by the subroutine in High Sensitivity communication mode each time that a packet is received. **DFT_Mes[7:0]** contains the signed frequency error between the transmitter and the receiver RF frequencies.
1LSB corresponds to an offset frequency of 26MHz/32768 = 793Hz.

10 Transmission programming flow

10.1 Transmission flow, mode payload size defined in header

This section describes the entire flow for transmitting RF data, in mode payload size in the header, starting from the RTX-24EM in power down mode (ENABLE (Pin 10) or nRESET (Pin 4) = GND).

1. Start the RTX-24EM by setting ENABLE (Pin 10) or nRESET (Pin 4) to Vcc.
2. Start the crystal oscillator by writing 0b111000000100 (0xE04) in **RAM2** at the address **0** (SPI command **Write_RAM2**).
3. Wait for crystal oscillator ON: quartz activity can be monitored through **Status[1]** by using a simple **Stop_Micro** SPI command. When **Status[1]** has gone low, the oscillator is ON.
4. Send the SPI command **ROM_Boot** with **ROM_Boot_Address = 0** to perform the **RAM2** initialization.
5. Send the SPI command **Write_RAM1** with address = **13** and data write = **1184** (to set RB_Inst_Dis = 1).
6. Send the SPI command **Start_Micro**.
7. Wait the IRQ pin to go to high : this will mean that the RAM2 initialization subroutine has been executed.
8. Clear IRQ using the SPI command **Clear_IRQ**.
9. Send the SPI command **ROM_Boot** with **ROM_Boot_Address = 33** to perform the PTAT auto calibration.
10. Send the SPI command **Start_Micro**.
11. Wait the IRQ pin to go to high : this will mean that the PTAT auto calibration subroutine has been executed.
12. Clear IRQ using the SPI command **Clear_IRQ**.
13. Use the SPI command **Write_RAM2** to configure the Address byte **Address[7:0]** (see section 9.7).
14. Use the SPI command **Write_RAM2** to configure the data rate (**R_Bit_Clk[8:0]** and **Ch_Rate[2:0]**), see section 9.5).
15. Use the SPI command **Write_RAM2** to configure the output power (**I_Pre_PA[4:0]** and **I_PA[4:0]**), see section 9.8).
16. Use the SPI command **Write_RAM2** to configure the channel frequency (**Frequ[16:0]**), see section 9.6).
17. Send the SPI command **ROM_Boot** with **ROM_Boot_Address = 64** to perform the VCO auto calibration.
18. Send the SPI command **Write_RAM1** with address = **53** and data_write = **VcoCalibFreq[11:0]** to program the center frequency for the auto calibration, see section 9.4.
19. Send the SPI command **Start_Micro**.
20. Wait the IRQ pin to go to high : this will mean that the VCO auto calibration subroutine has been executed.
21. Clear IRQ using the SPI command **Clear_IRQ**.

22. Send the SPI command **ROM_Boot** with **ROM_Boot_Address = 256** for High sensitivity mode or **320** for Normal sensitivity mode.
23. Send the SPI command **Start_Micro**.
24. Write the **TXFIFO** through SPI command **Write_TXFIFO**. The first byte written is the header and contents the payload size in its 5 LSBs.
25. Send the SPI command **Send_TXFIFO** to transmit the packet: the RTX-24EM will send the RF packet until **TXFIFO** is empty.
26. Wait the IRQ pin to go to high: this will mean that the RF packet transmission has been completed.
27. Clear IRQ using the SPI command **Clear_IRQ**.

At the end of the transmission the RTX-24EM goes to reception mode therefore, if low power consumption is required, it will be necessary to set ENABLE (Pin 10) or nRESET (Pin 4) to GND to force the RTX-24EM in power down mode. The next packet transmission requires to execute all steps from 1 to 27.

If the low power consumption is not required, at the end of the transmission the RTX-24EM can stay in reception mode. In this case the next packet transmission requires to execute only steps 24 to 27.

Note 1: Transmission mode is started from reception mode. So, it is not possible to exclude that an incoming packet has triggered the IRQ before Sent Packet IRQ is activated (IRQ is the same for all operational modes). The SPI command **Read_TXFIFO_Size** and **Read_RXFIFO_Size** allows the user to look at both **TXFIFO** and **RXFIFO** to determine the origin of the interrupt.

Note 2: Transmission mode is operated by the internal microcontroller, which operates by taking control of **RAM2** and **TXFIFO**. Read and Write SPI accesses to **RAM2** will put the microcontroller on hold. If the SPI transaction time is too long (if SCK frequency is close to channel data rate), correct transmission operation can be corrupted. SPI commands to retrieve **RAM2** values such as **Limit_RSSI[3:0]** or **DFT_Mes[7:0]** should be sent immediately after IRQ signal has gone high rather than after a **Send_TXFIFO** request.

10.2 Transmission flow, mode payload size defined in RAM2

This section describes the entire flow for transmitting RF data, in mode payload size in RAM2, starting from the RTX-24EM in power down mode (ENABLE (Pin 10) or nRESET (Pin 4) = GND).

The steps are the same as described in section 10.1 excepted:

- the payload size **N_Pay[4:0]** must be write in **RAM2** after step 12;
- the header byte no longer defines the number of byte of the payload;
- the **ROM_Boot_Address** must be set **= 128** for High sensitivity mode or **192** for Normal sensitivity mode, in step 22.

11 Reception programming flow

11.1 Reception flow, mode payload size defined in header

This section describes the entire flow for receiving RF data, in mode payload size in the header, starting from the RTX-24EM in power down mode (ENABLE (Pin 10) or nRESET (Pin 4) = GND).

28. Start the RTX-24EM by setting the ENABLE (Pin 10) or nRESET (Pin 4) to Vcc.
1. Start the crystal oscillator by writing 0b111000000100 (0xE04) in **RAM2** at the address **0** (SPI command **Write_RAM2**).
2. Wait for crystal oscillator ON: quartz activity can be monitored through **Status[1]** by using a simple **Stop_Micro** SPI command. When **Status[1]** has gone low, the oscillator is ON.
3. Send the SPI command **ROM_Boot** with **ROM_Boot_Address = 0** to perform the **RAM2** initialization.
4. Send the SPI command **Write_RAM1** with address = **13** and data write = **1184** (to set RB_Inst_Dis = 1).
5. Send the SPI command **Start_Micro**.
6. Wait the IRQ pin to go to high : this will mean that the RAM2 initialization subroutine has been executed.
7. Clear IRQ using the SPI command **Clear_IRQ**.
8. Send the SPI command **ROM_Boot** with **ROM_Boot_Address = 33** to perform the PTAT auto calibration.
9. Send the SPI command **Start_Micro**.
10. Wait the IRQ pin to go to high : this will mean that the PTAT auto calibration subroutine has been executed.
11. Clear IRQ using the SPI command **Clear_IRQ**.
12. Use the SPI command **Write_RAM2** to configure the Address byte **Address[7:0]** (see section 9.7).
13. Use the SPI command **Write_RAM2** to configure the data rate (**R_Bit_Clk[8:0]** and **Ch_Rate[2:0]**), see section 9.5).
14. Use the SPI command **Write_RAM2** to configure the output power (**I_Pre_PA[4:0]** and **I_PA[4:0]**), see section 9.8).
15. Use the SPI command **Write_RAM2** to configure the channel frequency (**Frequ[16:0]**), see section 9.6).
16. Send the SPI command **ROM_Boot** with **ROM_Boot_Address = 64** to perform the VCO auto calibration.
17. Send the SPI command **Write_RAM1** with address = **53** and data_write = **VcoCalibFreq[11:0]** to program the center frequency for the auto calibration, see section 9.4.
18. Send the SPI command **Start_Micro**.
19. Wait the IRQ pin to go to high : this will mean that the VCO auto calibration subroutine has been executed.
20. Clear IRQ using the SPI command **Clear_IRQ**.
21. Send the SPI command **ROM_Boot** with **ROM_Boot_Address = 256** for High sensitivity mode or **320** for Normal sensitivity mode.
22. Send the SPI command **Start_Micro**: the RTX-24EM will wait an incoming RF packet and store the number of byte stored in the header in the **RXFIFO** (including the header).
23. Wait the IRQ pin to go to high: this will mean that a RF packet has been stored in the **RXFIFO**.
24. Clear IRQ using the SPI command **Clear_IRQ**.
25. Read the **RXFIFO** using the SPI command **Read_RXFIFO**.

Note 1: Reception mode is operated by the internal microcontroller, which operates by taking control of **RAM2** and **RXFIFO**. Read and Write SPI accesses to **RAM2** will put the microcontroller on hold. If the SPI transaction time is too long (if SCK frequency is close to channel data rate), correct reception operation can be corrupted.SPI commands to retrieve **RAM2** values such as **Limit_RSSI[3:0]** or **DFT_Mes[7:0]** should be sent immediately after IRQ signal has gone high.

11.2 Reception flow, mode payload size defined in RAM2

This section describes the entire flow for receiving RF data, in mode payload size defined in RAM2, starting from the RTX-24EM in power down mode (ENABLE (Pin 10) or nRESET (Pin 4) = GND).

The steps are the same as described in section 11.1 excepted:

- the payload size **N_Pay[4:0]** must be write in **RAM2** after step 12;
- the header byte no longer defines the number of byte of the payload;
- the **ROM_Boot_Address** must be set **= 128** for High sensitivity mode or **192** for Normal sensitivity mode, in step 22.

12 RAM2 Registers

In this section are shown all the most relevant RAM2 Register in order to properly configure the RTX-24EM. Any register not specifically mentioned here is reserved and its contents must be set according to defined Default value.

RAM2[11:0]@ Address 0

	Bit	Default Value	Reset Value	Description
VDD_Synth_En	11	1	0	VDD_Synth Voltage Regulator enable.
VDD_RXTX_En	10	1	0	VDD_RXTX Voltage Regulator enable.
Xtal_En	9	1	0	Crystal oscillator enable.
Reserved	8	1	0	Reserved.
Reserved	7	1	0	
Reserved	6	1	0	
Reserved	5	0	0	
Div_Ck_Freq[0]	4	0	0	Select the frequency on the clock output on DIV_CK.
Div_Ck_Freq[1]	3	0	0	
Reserved	2	1	0	Reserved bit. Bit 2 should be set to '1' each time this register is written.
Reserved	1	0	0	
Reserved	0	0	0	

RAM2[11:0]@ Address 1

	Bit	Default Value	Reset Value	Description
Reserved	11	0	0	Reserved.
Reserved	10	0	0	
Reserved	9	0	0	
Reserved	8	0	0	
Reserved	7	0	0	
Reserved	6	0	0	
Reserved	5	0	0	
Reserved	4	0	0	
Reserved	3	0	0	
Tx_On_Pad	2	0	0	TX_On Pad control. Set by internal microcontroller.
Rx_On_Pad	1	0	0	RX_On Pad control. Set by internal microcontroller.
Reserved	0	0	0	Reserved.

RAM2[11:0]@ Address 2

	Bit	Default Value	Reset Value	Description
Reserved	11	1	0	Reserved.
Reserved	10	0	0	
Reserved	9	1	0	
Reserved	8	0	0	
Reserved	7	0	0	
Reserved	6	0	0	
Reserved	5	0	0	
Reserved	4	1	0	
Reserved	3	0	0	
Reserved	2	1	0	
Reserved	1	0	0	
Reserved	0	1	0	

RAM2[11:0]@ Address 3

	Bit	Default Value	Reset Value	Description
Reserved	11	0	0	Reserved
Reserved	10	0	0	
Reserved	9	1	0	
Reserved	8	0	0	
Reserved	7	1	0	
Reserved	6	1	0	
Reserved	5	0	0	
Reserved	4	0	0	
VCO_Code[3]	3	1	0	The VCO tuning code is determined automatically by auto calibration procedure.
VCO_Code[2]	2	0	0	
VCO_Code[1]	1	0	0	
VCO_Code[0]	0	0	0	

RAM2[11:0]@ Address 4

	Bit	Default Value	Reset Value	Description
I_Pre_PA[4]	11	0	0	Current bias of the PA preamplifier. Defines RF output power in Transmit mode.
I_Pre_PA[3]	10	1	0	
I_Pre_PA[2]	9	1	0	
I_Pre_PA[1]	8	0	0	
I_Pre_PA[0]	7	1	0	
Reserved	6	0	0	Reserved.
Reserved	5	0	0	
Reserved	4	0	0	
Reserved	3	0	0	
Reserved	2	0	0	
Reserved	1	0	0	
Reserved	0	0	0	

RAM2[11:0]@ Address 5

	Bit	Default Value	Reset Value	Description
Reserved	11	0	0	Reserved.
Reserved	10	1	0	
Reserved	9	0	0	
Reserved	8	0	0	
Reserved	7	0	0	
Reserved	6	1	0	
Reserved	5	0	0	
Reserved	4	0	0	
Main_PTAT[3]	3	1	0	Control of the main chip PTAT current bias.
Main_PTAT[2]	2	0	0	
Main_PTAT[1]	1	0	0	
Main_PTAT[0]	0	0	0	

RAM2[11:0]@ Address 6

	Bit	Default Value	Reset Value	Description
I_PA[4]	11	0	0	Current bias of the PA. Defines RF output in Transmit mode.
I_PA[3]	10	1	0	
I_PA[2]	9	0	0	
I_PA[1]	8	1	0	
I_PA[0]	7	1	0	
Reserved	6	0	0	Reserved.
Reserved	5	0	0	
Reserved	4	0	0	
Reserved	3	0	0	
Reserved	2	0	0	
Reserved	1	0	0	
Reserved	0	0	0	

RAM2[11:0]@ Address 7

	Bit	Default Value	Reset Value	Description
Reserved	11	1	0	Reserved.
Reserved	10	0	0	
Reserved	9	0	0	
Reserved	8	1	0	
Reserved	7	0	0	
Reserved	6	0	0	
Reserved	5	0	0	
Reserved	4	0	0	
Reserved	3	0	0	
Reserved	2	0	0	
Reserved	1	0	0	
Reserved	0	0	0	

RAM2[11:0]@ Address 8

	Bit	Default Value	Reset Value	Description
Reserved	11	0	0	Reserved.
Reserved	10	1	0	
Reserved	9	0	0	
Reserved	8	0	0	
Reserved	7	1	0	
Reserved	6	0	0	
RB_Inst_Dis	5	0	0	ROMboot Instruction Disable.
Reserved	4	0	0	Reserved.
Limit_RSSI[3]	3	0	0	RSSI value.
Limit_RSSI[2]	2	0	0	
Limit_RSSI[1]	1	0	0	
Limit_RSSI[0]	0	0	0	

RAM2[11:0]@ Address 9

	Bit	Default Value	Reset Value	Description
Reserved	11	1	0	Reserved.
Reserved	10	0	0	
Reserved	9	1	0	
Reserved	8	0	0	
DFT_Mes[7]	7	0	0	Error frequency measured by DFT in High Sensitivity mode.
DFT_Mes[6]	6	0	0	
DFT_Mes[5]	5	0	0	
DFT_Mes[4]	4	0	0	
DFT_Mes[3]	3	0	0	
DFT_Mes[2]	2	0	0	
DFT_Mes[1]	1	0	0	
DFT_Mes[0]	0	0	0	

RAM2[11:0]@ Address 10

	Bit	Default Value	Reset Value	Description
Reserved	11	0	0	Reserved.
Reserved	10	0	0	
Reserved	9	1	0	
Reserved	8	0	0	
Reserved	7	1	0	
Reserved	6	0	0	
Reserved	5	0	0	
Reserved	4	0	0	
Reserved	3	0	0	
Reserved	2	1	0	
Reserved	1	0	0	
Reserved	0	0	0	

RAM2[11:0]@ Address 11

	Bit	Default Value	Reset Value	Description
Reserved	11	0	0	Reserved.
Reserved	10	0	0	
Reserved	9	0	0	
Reserved	8	0	0	
Reserved	7	1	0	
Reserved	6	1	0	
Reserved	5	0	0	
Reserved	4	0	0	
Reserved	3	0	0	
Reserved	2	0	0	
Reserved	1	0	0	
Reserved	0	0	0	

RAM2[11:0]@ Address 12

	Bit	Default Value	Reset Value	Description
Ch_Rate[2]	11	0	0	Bandwidth of the normal sensitivity demodulator.
Ch_Rate[1]	10	0	0	
Ch_Rate[0]	9	0	0	
R_Bit_Ck[8]	8	1	0	CODEC Bit clock frequency.
R_Bit_Ck[7]	7	1	0	
R_Bit_Ck[6]	6	0	0	
R_Bit_Ck[5]	5	0	0	
R_Bit_Ck[4]	4	0	0	
R_Bit_Ck[3]	3	0	0	
R_Bit_Ck[2]	2	0	0	
R_Bit_Ck[1]	1	0	0	
R_Bit_Ck[0]	0	0	0	

RAM2[11:0]@ Address 13

	Bit	Default Value	Reset Value	Description
Reserved	11	0	0	Reserved.
Reserved	10	0	0	
Reserved	9	0	0	
Reserved	8	0	0	
Address[7]	7	1	0	Address Byte Value.
Address[6]	6	0	0	
Address[5]	5	1	0	
Address[4]	4	0	0	
Address[3]	3	1	0	
Address[2]	2	0	0	
Address[1]	1	0	0	
Address[0]	0	1	0	

RAM2[11:0]@ Address 14

	Bit	Default Value	Reset Value	Description
Reserved	11	0	0	Reserved.
Reserved	10	0	0	
Frequ[4]	9	0	0	Synthesizer's RF Frequency LSB's
Frequ[3]	8	1	0	
Frequ[2]	7	1	0	
Frequ[1]	6	1	0	
Frequ[0]	5	0	0	
N_Pay[4]	4	0	0	Payload size of the Packet: N_Pay + 1
N_Pay[3]	3	0	0	
N_Pay[2]	2	0	0	
N_Pay[1]	1	1	0	
N_Pay[0]	0	1	0	

RAM2[11:0]@ Address 15

	Bit	Default Value	Reset Value	Description
Frequ[16]	11	0	0	Synthesizer's RF Frequency MSB's.
Frequ[15]	10	1	0	
Frequ[14]	9	1	0	
Frequ[13]	8	1	0	
Frequ[12]	7	0	0	
Frequ[11]	6	1	0	
Frequ[10]	5	1	0	
Frequ[9]	4	0	0	
Frequ[8]	3	0	0	
Frequ[7]	2	0	0	
Frequ[6]	1	1	0	
Frequ[5]	0	0	0	

13 RF Packet format

In normal sensitivity mode, each packet contains the following information:

Preamble	3 * Address Byte	Header	Payload

Packet information	Length	Description
Preamble = 5 * '11001100'	5 byte	Clock recovery and data-slicer initialization.
3 * Address Byte	3 byte	This field consists of 3 Address byte Address[7:0].
Header	0 or 1 byte	The 5 LSB's of the header represent the payload size of the packet for "payload size defined in header" mode (ROM_Boot_Address = 320). For "payload size defined in RAM2" mode (ROM_Boot_Address = 192) the header is not existing.
Payload	0 -32 bytes	Data

In high sensitivity mode, each packet contains the following information:

Marking	Preamble	3 * Address Byte	Header	Payload

Packet information	Length	Description
Marking	14ms	Initial frequency step used to lock high sensitivity demodulator.
Preamble = 2 * '11001100'	2 byte	Clock recovery initialization.
3 * Address Byte	3 byte	This field consists of 3 Address byte Address[7:0].
Header	0 or 1 byte	The 5 LSB's of the header represent the payload size of the packet for "payload size defined in header" mode (ROM_Boot_Address = 256). For "payload size defined in RAM2" mode (ROM_Boot_Address = 128) the header is not existing.
Payload	0 -32 bytes	Data

The size of the received packet including header and payload (stored in **RXFIFO**) can be read with the SPI command **Read_RXFIFO_Size**.
When **RXFIFO** has been completely read (until **RXFIFO_Size[4..0]** = 0) after the previous reception, **RXFIFO_Size[4..0]** of the current packet is then the total packet size including the header.
The header byte is treated like a standard payload byte and must be stored in the **TX_FIFO** before the payload data prior to transmit the packet in any mode.
For each byte, the most-significant bit (MSB) is sent first.

14 Device usage

In order to obtain the performances described in the technical specifications and to comply with the operating conditions which characterize the certification, the device must be mounted on a printed circuit taking into account the following:

Voltage Supply

1. The module must be powered by a low voltage safety source protected against short circuits. Maximum allowed voltage variations: 1.9÷3.6V.
2. Decoupling near the transmitter with a ceramic capacitor of at least 100 nF.

Ground

It must surround in the best possible way the welding area of the transceiver. The ground plane should be made in the lower face and must not be present near the antenna integrated in order to avoid coupling.

Other components:

1. When mounting the module parallel to the PCB (RTX-24EM-AI/H) do not include tracks near integrated antenna.
2. Keep the device away from all other components of the circuit (more than 5 mm)
3. Keep particularly far away and shielded all microprocessors and their clock circuits.

15 Revision History

Date	Description	Revision
25/02/2019	First Release	Preliminary
16/09/2019	Rev A	A
12/11/2019	Added some notes related to RF power and current consumption.	B
26/05/2020	Corrected the value of the parameter FreqCorrection.	C