Introduction

Atmel® wireless chat application based on Atmel AT86RF215 provides flexibility to explore features and capabilities of the AT86RF215 device along with simple transmission of text messages between nodes.

Features

- AT86RF215 transceiver is a multi-band sub-1GHz/2.4GHz device compliant to IEEE® 802.15.4g-2012™ and ETSI TS 102 887-1
- AT86RF215 offers very high flexibility by supporting a variety of data rates with three modulation schemes:
  - Multi-Rate and Multi-Regional Frequency Shift Keying (MR-FSK)
  - Orthogonal Frequency Division Multiplexing (MR-OFDM)
  - Offset Quadrature Phase-Shift Keying (MR-O-QPSK)
- Bi-directional differential RF signal ports, one for sub-1GHz and one for 2.4GHz
- Simultaneous operation of sub-1GHz and 2.4GHz transceiver
- Separate 2KB RX and TX frame buffer
1. **AT86RF215 Architecture**

AT86RF215 device features two independent radio systems; one for sub-1GHz and one for 2.4GHz. Each radio is paired with a baseband core optimized to modulate/demodulate IEEE 802.15.4g-2012 and ETSI TS 102 887-1 compliant signals. The internal baseband cores support MR-FSK, O-QPSK/MR-O-QPSK, and MR-OFDM modulation schemes.

AT86RF215 includes baseband core CORE0 connected with the sub-1GHz radio and baseband core CORE1 connected with the 2.4GHz radio. CORE0 and CORE1 are identical and can be independently used.

Figure 1-1 AT86RF215 Block Diagram
2. Modulation Schemes

The AT86RF215 features dedicated baseband core functionality using the following modulation schemes:

- MR-FSK PHY
- MR-OFDM PHY
- O-QPSK PHY

Each baseband core contains PHY implementations for MR-FSK, MR-OFDM, and O-QPSK. For each core, only one of the PHYs can be selected at a time.

BBCn_PC.PT – sub register of PHY Control register will be used to select the desired Modulation.

Table 2-1 Modulation Configuration

<table>
<thead>
<tr>
<th>Sub-register</th>
<th>Name</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PT</td>
<td>BB_PHYOFF</td>
<td>0x0</td>
<td>OFF</td>
</tr>
<tr>
<td></td>
<td>BB_MRFSK</td>
<td>0x1</td>
<td>MR-FSK</td>
</tr>
<tr>
<td></td>
<td>BB_MROFDM</td>
<td>0x2</td>
<td>MR-OFDM</td>
</tr>
<tr>
<td></td>
<td>BB_MROQPSK</td>
<td>0x3</td>
<td>MR-O-QPSK or legacy O-QPSK</td>
</tr>
</tbody>
</table>
3. **Hardware Platform**

The chat application has been implemented on the AT86RF215 development platform which consists of the ATREB215-XPRO extension board connected with the SAM4S/L-XPRO Evaluation Kit.

Figure 3-1 **ATREB215-XPRO Extension Board**
3.1. **Hardware Requirements**

The following hardware is required to demonstrate the capabilities of the AT86RF215.

- ATREB215-XPRO extension board – Three pieces.
- SAM4S/L-XPRO Evaluation Kit – Three pieces.
- Micro-USB cable – Three pieces.
- Antenna (sub GHZ and 2.4GHz) – Three pieces each.
4. **Chat Application**

The chat application provides a facility to test the basic functionality of the transceiver and possibility to change the configurations. Chat application contains four predefined modulation configurations that can be used for simple transmit and receive functionality.

(0) Off

(1) FSK, 50kbit/s, mod idx 1.0, dw=1, fec=0, CRC32

(2) MR-OFDM, Opt. 1, MCS3, CRC32, interl=0

(3) MR-OQPSK, chip rate 100kchip/s, rate mode 0, CRC32

(4) Legacy OQPSK

Custom configuration can be defined as option ‘5’ for validation. The procedures for code modifications are described in the later sections.

4.1. **Flashing Procedure**

The RF215_Chat.hex file available in the package can be programmed using the Atmel Studio → Device programming tool.

**Figure 4-1 Device Programming**
For ATSAM4SD32C device, set the Boot Mode bit true in the GPNVM bits available in the Device programming. During debugging of the program the GPNVM bit settings will be modified by the Atmel Studio without any need of user intervention.

Figure 4-2 Fuse Setting - ATSAM4SD32C Device

4.2. Demo Procedure

To execute the demonstration,

1. Open terminal program with the configuration 115200, 8, n, 1.
2. Application description is provided on the terminal console on the MCU reset as shown in following figure.
3. The device receives data at both frequency bands.
4. The transmitting band is selected by “Ctrl-S” for the sub-1GHz band and “Ctrl-D” for the 2.4GHz band.
5. Type any characters and press enter; by pressing return the line is sent to other device configured in the PEER_SHORT_ADDR available in the app_common.h file.
6. The received message is printed to the terminal window with information about receiver band, RSSI value, and sender’s address.
7. CSMA and re-transmissions are enabled in chat message transmissions. If transmission was not successful, an error message is printed to the terminal window.
8. PHY modulation can be changed by Modulation configuration menu, press “/” to access Modulation setting menu.

Figure 4-4 Modulation Configuration
9. Four PHY Modulation configuration presets are available.
10. For custom modulation configuration, changes need to be done directly in the source code, which will be covered in section 6.
11. Change the PEER_SHORT_ADDR (destination address) and OWN_SHORT_ADDR (source address) available in the app_common.h file unique for each device used in the demo.

Note: For the purpose of the demonstration, PEER_SHORT_ADDR and OWN_SHORT_ADDR are configured as same values.

4.3. Task 1: Sub-1GHz Data Communication

The intention of Task 1 is to demonstrate the sub-1GHz communication using two SAM4S+AT86RF215 devices.
1. Generate two hex files with unique address configuration as mentioned in section demo procedure.
2. Flash two devices with generated hex file after modification.
3. Open one HyperTerminal window for transmission and another HyperTerminal window for reception with configuration -115200,8,N,1.
4. Press “Ctrl-S” to select transmitting band as sub-1GHz.
5. Start typing the chat message on the transmitting device and press Enter.
6. Transmitted data will be displayed on the receiving device (PEER_SHORT_ADDR) along with receiver band information as RF09, RSSI value, and sender’s address (OWN_SHORT_ADDR) as shown in following figure.

Figure 4-5 Task 1: Sub-1GHz Data Communication

4.4. Task 2: 2.4GHz Data Communication

The intention of Task 2 is to demonstrate the 2.4GHz communication using two SAM4S+AT86RF215 devices.
1. Generate two hex files with unique address configuration as mentioned in section demo procedure.
2. Flash two devices with RF215_Chat.hex file.
3. Open one HyperTerminal window for transmission and another HyperTerminal window for reception with configuration -115200,8,N,1.
4. Press “Ctrl-D” to select transmitting band as 2.4GHz.
5. Start typing the chat message on the transmitting device and press Enter.
6. Transmitted data will be displayed on the receiving device (PEER_SHORT_ADDR) along with receiver band information as RF24, RSSI value, and sender’s address (OWN_SHORT_ADDR) as shown in following figure.
4.5. Task 3: Simultaneous Data Reception in both sub-1GHz and 2.4GHz

The intention of Task 3 is to demonstrate the simultaneous data reception in both sub-1GHz and 2.4GHz communication using three SAM4S+AT86RF215 devices.

1. Generate two hex files with unique address configuration as mentioned in section demo procedure.
   - Change the PEER_SHORT_ADDR (destination address) of two devices (device1 and device 2) to the third device (device 3) address.
2. Flash three devices with generated RF215_Chat.hex file.
3. Open two HyperTerminal windows for transmission using device 1 and device 2 and open another HyperTerminal window for reception using device 3 with configuration -115200,8,N,1.
4. Press “Ctrl-D” to select transmitting band as 2.4GHz on device 1 and press “Ctrl-S” to select transmitting band as sub-1GHz on device 2.
5. Device 3 will be listening for chat messages on both 2.4GHz and sub-1GHz frequency bands.
6. Start typing the chat message on the transmitting device 1 and 2 and press Enter.
7. Transmitted data will be displayed on the receiving device 3 (PEER_SHORT_ADDR) along with receiver band information as RF24 or RF09, RSSI value, and sender's address (OWN_SHORT_ADDR) as shown in the following screens.

Table 4-1 Device Address Configuration

<table>
<thead>
<tr>
<th>Task</th>
<th>Device</th>
<th>OWN_SHORT_ADDR</th>
<th>PEER_SHORT_ADDR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task 1 and Task 2</td>
<td>Device 1</td>
<td>0x1234</td>
<td>0x1235</td>
</tr>
<tr>
<td></td>
<td>Device 2</td>
<td>0x1235</td>
<td>0x1234</td>
</tr>
<tr>
<td>Task</td>
<td>Device</td>
<td>OWN_SHORT_ADDR</td>
<td>PEER_SHORT_ADDR</td>
</tr>
<tr>
<td>--------</td>
<td>--------</td>
<td>----------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>Task 3</td>
<td>Device 1</td>
<td>0x1234</td>
<td>0x1236</td>
</tr>
<tr>
<td></td>
<td>Device 2</td>
<td>0x1235</td>
<td>0x1236</td>
</tr>
<tr>
<td></td>
<td>Device 3</td>
<td>0x1236</td>
<td>0x1234/0x1235</td>
</tr>
</tbody>
</table>
5. **Configuration Options**

AT86RF215 provides a rich set of configuration possibilities like modulation, data rate, bandwidth, etc. to the user. Upcoming topics cover the different configuration registers that are required for changing or selecting a particular modulation scheme.

The chat application project folder organization is as follows:

- AT86RF215 folder contains implementation of AT86RF215 functionalities.
- Chat folder contains the application logic related to the chat application.
Chat application internally handles the transceiver register level configuration based on user inputs to these variables.
The structures responsible for configuring AT86RF215 are provided below.

```c
/**
 * PHY mode structure
 */
typedef union phy_mode_tag
{
    fsk_t fsk;
    ofdm_t ofdm;
    oqpsk_t oqpsk;
    leg_oqpsk_t leg_oqpsk;
} phy_mode_t;

typedef struct phy_tag
{
    modulation_t modulation;
    sun_freq_band_t freq_band;
    uint32_t freq_f0;
    uint32_t ch_spacing;
    phy_mode_t phy_mode;
} phy_t;

typedef union rate_tag
{
    fsk_sym_rate_t fsk_rate;
    ofdm_mcs_t ofdm_mcs;
    oqpsk_rate_mode_t oqpsk_rate_mod;
} rate_t;

typedef struct new_phy_tag
{
    modulation_t modulation;
    phy_mode_t phy_mode;
    rate_t rate;
    bool_t fec_enabled;
} new_phy_t;
```

- Structure `phy_t` - Contains modulation and operating channel related parameters.
- Structure `phy_mode_t` - Contains parameter related to the modulation scheme based on the modulation available in the Structure `phy_t`.
- Structure `new_phy_t` and the `rate_t` - Contains parameter related to mode switch feature for storing the new modulation parameters.

The possible values for the variables are depend on the modulation scheme and other parameters. Refer the `ieee_154g.h` file structures and enumeration values for further information.

Reference for modulation configuration settings are available in the file `phy_conf.c`.

Basic steps to configure transceiver modulation and frequency based on variables are:

1. Set modulation using the `phy.modulation`.
2. Set modulation mode related parameters using `phy.phy_mode.<modulation_mode>`.
3. Set the frequency band using the `phy.freq_band`.
4. Set the channel spacing using the `phy.ch_spacing`.
5. Set the center frequency using the `phy.freq_f0`.
6. Set the data rate, transmit power and channel using the corresponding PIB values based on the modulation scheme.

Refer the function `set_mod()` and sub-functions available in the `phy_conf.c` file for specific modulation schemes.

The parameters defined in the `app_common.h` will be used in data transfer and reception.

5.1. Channel Configuration

The frequency synthesizer generates the RF frequency which is used to convert the baseband signal to an RF signal during transmission and used to mix down the received RF signal to a low IF signal during reception process.

Two separate frequency synthesizers (sub-1GHz and 2.4GHz transceiver) are implemented to generate the required frequency.

The PLLs operate at the TXPREP, StateTX, and State RX states alone. The PLL locking and unlocking process will be intimated through the IRQS.TRXRDY and IRQS.TRXERR interrupts respectively.

The channel frequency must be configured in State TRXOFF using registers Channel Spacing register (RFn_CS) with resolution of 25kHz, Channel Center Frequency registers (RFn_CCF0H, RFn_CCF0L) and the Channel Number and Mode registers (RFn_CNL, RFn_CNM).

The Channels can be configured in the following two modes using the Channel Setting Mode bits (CNM.CM):

- IEEE-compliant scheme
- Fine resolution scheme

<table>
<thead>
<tr>
<th>Channel mode</th>
<th>Channel setting mode (CNM.CM) value</th>
<th>Description</th>
<th>Frequency resolution</th>
<th>2.4GHz transceiver (RF24)</th>
<th>sub-1GHz transceiver (RF09)</th>
</tr>
</thead>
<tbody>
<tr>
<td>IEEE compliant channel scheme</td>
<td>0x00</td>
<td>[f=(CCF0+CNC*CS)*25kHz+f offset ] (f offset = 0Hz for sub-1GHz transceiver; f offset = 1.5GHz for 2.4GHz transceiver)</td>
<td>≈25kHz</td>
<td>Valid</td>
<td>No effect</td>
</tr>
<tr>
<td>Fine resolution scheme</td>
<td>0x01</td>
<td>Fine resolution (389.5-510.0)MHz with 99.182Hz channel stepping</td>
<td>≈100Hz (6.5MHz/216)</td>
<td>No effect</td>
<td>Valid</td>
</tr>
<tr>
<td>Fine resolution scheme</td>
<td>0x02</td>
<td>Fine resolution (779-1020)MHz with 198.364Hz channel stepping</td>
<td>≈200Hz (13.0MHz/216)</td>
<td>No effect</td>
<td>Valid</td>
</tr>
<tr>
<td>Fine resolution scheme</td>
<td>0x03</td>
<td>Fine resolution (2400-2483.5)MHz with 396.728Hz channel stepping</td>
<td>≈400Hz (26.0MHz/216)</td>
<td>Valid</td>
<td>No effect</td>
</tr>
</tbody>
</table>
5.1.1. **Configuration Mapping**

Table 5-2 Channel Configuration Register - Variable Map

<table>
<thead>
<tr>
<th>Register</th>
<th>Variable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Channel Spacing register (RFn_CS) with resolution of 25kHz</td>
<td>phy.ch_spacing</td>
</tr>
<tr>
<td>Channel Center Frequency registers (RFn_CCF0H, RFn_CCF0L)</td>
<td>phy.freq_f0</td>
</tr>
</tbody>
</table>

Channel Number(RFn_CNL) can be set by setting phyCurrentChannel Pib using the `tal_pib_set()` function.

5.2. **Frontend Configuration**

The transceiver must be in the State TRXOFF for configuration of the transmitter and receiver frontend.

5.2.1. **Transmitter Digital Frontend**

The Transmitter Digital Frontend (TX_DFE) performs discrete time sampling rate conversion of the complex baseband signal based on the sample rate configured in the TX Sample Rate (TXDFE.SR). For this conversion, a zero-intermediate frequency (zero-IF) architecture is applied. The baseband signal at sampling frequency (fs) can be pre-filtered at block PRE_FLT based on the cut-off frequencies (f cut) configured in the TX filter relative to the cut-off frequency (TXDFE.RCUT).

![Figure 5-2 Transmitter Digital Frontend Block Diagram](image)

5.2.2. **Transmit Control**

The Baseband mode is used to control the transmitter based on the Chip Mode (IQIFC1.CHPM):

Table 5-3 Transmitter Status

<table>
<thead>
<tr>
<th>Operating Mode</th>
<th>TRX09</th>
<th>TRX24</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>RF09</td>
<td>BBC0</td>
</tr>
<tr>
<td>IQIFC1.CHPM=RF_MODE_BBRF</td>
<td>on</td>
<td>on</td>
</tr>
</tbody>
</table>

|                | Baseband Mode | Baseband Mode |

5.2.3. **Receiver Analog Frontend**

The received signal is down-converted to a low intermediate frequency (IF) based on the Receiver Bandwidth (RXBWC.BW) after passing a low-noise amplifier (LNA). Depending on the RX Sample Rate configuration register (RXDFE.SR), the signal is further down-sampled to the target receive sampling...
frequency. RX filter relative cut-off frequency (RXDFE.RCUT) value will be used to further filtering by block POST_FLT.

Figure 5-3 Receiver Analog Frontend Block Diagram

5.2.4. Configuration Mapping

There are few parameters like chip rate, modulation index, symbol rate, etc. are obtained based on the modulation schemes.

typedef struct oqpsk_tag
{
    oqpsk_chip_rate_t chip_rate;
} oqpsk_t;

typedef struct ofdm_tag
{
    ofdm_option_t option;
} ofdm_t;

typedef struct leg_oqpsk_tag
{
    oqpsk_chip_rate_t chip_rate;
} leg_oqpsk_t;

typedef struct fsk_tag
{
    fsk_mod_type_t mod_type;
    mod_idx_t mod_idx;
    fsk_sym_rate_t sym_rate;
    fsk_op_mode_t op_mode;
} fsk_t;

The front end configurations parameters are derived from the modulation related parameters and the same has been configured in the tal_fe.c file.

The TX Sample Rate (TXDFE.SR), TX filter relative to the cut-off frequency (TXDFE.RCUT), Receiver Bandwidth (RXBWC.BW), RX Sample Rate configuration register (RXDFE.SR), RX filter relative cut-off frequency (RXDFE.RCUT) are derived based on the modulation.

- For OFDM the parameters are available in the RF215 Datasheet, table 'Recommended Transmitter Frontend Configuration' and table 'Recommended PHY Receiver and Digital Frontend Configuration' and they are applied in the function ofdm_rfcfg().

Ref: Atmel AT13282: AT86RF215 Chat Application [APPLICATION NOTE]
• For OQPSK the parameters are derived based on the Chip rate provided in the `oqpsk_t.chip_rate` and used in the function `oqpsk_rfcfg()`.
• For FSK the parameters are derived based on the modulation order, Symbol rate and modulation index provided in the `fsk_t` and used in the function `fsk_rfcfg()`.

5.3. **Energy Measurement and AGC Configuration**

The energy measurement module can be used as part of a channel selection algorithm, such as energy detection (ED) scan procedure, or clear channel assessment procedure (CCA). The AT86RF215 energy detection module is characterized by:

- Wide measurement range
- 1dB resolution
- Three different energy measurement modes
- Configurable energy detection averaging duration
- Current Received Signal Strength Indicator (RSSI) value

Discrete time regulation loop Automatic Gain Control (AGC) is employed in the AT86RF215. Variance of the signal (AGCC.AGCI set to 0 or 1) is measured and compared with a target variance. Gain of the LNA is set to aim for a vanishing control deviation if the AGC Enabled (AGCC.EN).

AGC measures the energy on the configured channel when the receiver is enabled and the result is stored to the RSSI register (RFn_RSSI). The RSSI value will be used to generate an average ED value based on the duration set in the Receiver Energy Detection Averaging Duration register (RFn_EDD) and stored in the Receiver Energy Detection Value (RFn_EDV).

The energy measurement module supports three different modes based on the value set in the Energy Detection Configuration register (RFn_EDC):

1. Automatic measurement (default mode) – Updated for every data reception.
2. Single measurement – Used for CCA.
3. Continuous measurement – Used for ED.

The measurement is triggered by writing the value 1 (for single mode) or 2 (for continuous mode) to the sub-register EDC.EDM for the Single and Continuous measurement mode.

5.4. **Baseband Configuration**

5.4.1. **MR-FSK**

The multi-rate and multi-regional frequency shift keying (MR-FSK) PHY.

AT86RF215 implemented the MR-FSK supporting the operation specified in IEEE Std 802.15.4g™ GFSK PHY.

The information bits of a packet can be optionally processed by Forward Error Correction (FEC) based on the FEC Scheme (FSKC2.FECS), interleaving (ILV) based on the FEC Interleaving Enable (FSKC2.FECIE), and data whitening (DW) based on the Data Whitening (FSKPHRTX.DW). Bit-To-Symbol-Mapping (B2S) depends on the FSK Modulation Order (FSKC0.MORD) either 2FSK or 4FSK produces the discrete-time frequency symbol sequence.

The MR-FSK Symbol Rate, Modulation Index and FSK Bandwidth Time Product can be configured using the FSKC1.SRATE, FSKC0.MIDX, FSKC0.BT fields respectively.
Supported IEEE 802.15.4g MR-FSK modes

- Filtered 2FSK
  - 50 kbit/s (1) with modulation index 1.0
  - 100 kbit/s (2) with modulation index 1.0
  - 150 kbit/s (2) with modulation index 0.5
  - 200 kbit/s (2) with modulation index 0.5 and 1.0

- Filtered 4FSK (2)
  - 200 kbit/s with modulation index 0.33
  - 400 kbit/s with modulation index 0.33

Parameter set

- 2FSK and 4FSK
- Symbol rates: 50, 100, 150, 200, 300, 400 ksymbol/s
- Rate ½ FEC: RSC and NRNSC, with and without interleaving
- Gaussian filtering: BT = 0.5/1.0/1.5/2.0
- Modulation index: 0.33 – 2.5
- Preamble length: 4 – 1000

Modulation parameters for European bands according to ETSI-TS-102-887-1

- GFSK PHY
  - 2FSK with modulation index 0.5, 100 ksymbol/s, 100 and 50 kbit/s
  - 2FSK with modulation index 0.5, 200 ksymbol/s, 200 and 100 kbit/s
  - 2FSK with modulation index 1, 50 ksymbol/s, 50 and 25 kbit/s
  - 2FSK with modulation index 0.5, 150 ksymbol/s, 150 and 75 kbit/s
  - 4FSK with modulation index 0.33, 100 ksymbol/s, 200 and 100 kbit/s

Table 5-4 MR-FSK Register Variable Map

<table>
<thead>
<tr>
<th>Register</th>
<th>Variable</th>
</tr>
</thead>
<tbody>
<tr>
<td>BBCn_PC</td>
<td>phy.modulation set to 0</td>
</tr>
<tr>
<td>FEC Scheme (FSKC2.FECS)</td>
<td>tal_pib.FSKFECScheme</td>
</tr>
<tr>
<td>Interleaving (ILV) based on the</td>
<td>tal_pib.FSKFECInterleavingRSC</td>
</tr>
<tr>
<td>FEC Interleaving Enable (FSKC2.FECSIE),</td>
<td></td>
</tr>
<tr>
<td>Data whitening (DW) based on the</td>
<td>tal_pib.FSKScramblePSDU</td>
</tr>
<tr>
<td>Data Whitening (FSKPHRTX.DW).</td>
<td></td>
</tr>
<tr>
<td>Bit-To-Symbol-Mapping (B2S)</td>
<td>tal_pib.phy.phy_mode.fsk.mod_type</td>
</tr>
<tr>
<td>depends on the FSK Modulation</td>
<td></td>
</tr>
<tr>
<td>Order (FSKC0.MORD)</td>
<td></td>
</tr>
<tr>
<td>MR-FSK Symbol Rate (FSKC1.SRATE)</td>
<td>tal_pib.phy.phy_mode.fsk.sym_rate</td>
</tr>
<tr>
<td>Modulation Index (FSKC0.MIDX)</td>
<td>tal_pib.phy.phy_mode.fsk.mod_idx</td>
</tr>
</tbody>
</table>

5.4.2. MR-OFDM

The multi-rate and multi-regional orthogonal frequency division multiplexing (MR-OFDM) PHY.
This PHY includes four MR-OFDM Bandwidth Options based on the OFDMC.OPT fields, each one characterized by the number of active tones during the PHR or PSDU. The modulation and coding scheme (MCS - OFDMPHRTX.MCS for transmission and OFDMPHRRX.MCS for reception) allows selecting different PSDU data rates per option.

Table 5-5 OFDM Operating Modes

<table>
<thead>
<tr>
<th>OFDM parameters</th>
<th>MCS</th>
<th>Option 1</th>
<th>Option 2</th>
<th>Option 3</th>
<th>Option 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>OFDMC.OPT</td>
<td></td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Nominal bandwidth [kHz]</td>
<td></td>
<td>1094</td>
<td>552</td>
<td>281</td>
<td>156</td>
</tr>
<tr>
<td>DFT Size</td>
<td></td>
<td>128</td>
<td>64</td>
<td>32</td>
<td>16</td>
</tr>
<tr>
<td>Active tones</td>
<td></td>
<td>104</td>
<td>52</td>
<td>26</td>
<td>14</td>
</tr>
<tr>
<td>Pilot Tones</td>
<td></td>
<td>8</td>
<td>4</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>Data tones</td>
<td></td>
<td>96</td>
<td>48</td>
<td>24</td>
<td>12</td>
</tr>
<tr>
<td>PSDU data rates [kb/s]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BPSK, rate ½, 4 x frequency repetition</td>
<td></td>
<td>0</td>
<td>100</td>
<td>50</td>
<td>-</td>
</tr>
<tr>
<td>BPSK, rate ½, 2 x frequency repetition</td>
<td></td>
<td>1</td>
<td>200</td>
<td>100</td>
<td>50</td>
</tr>
<tr>
<td>QPSK, rate ½, 2 x frequency repetition</td>
<td></td>
<td>2</td>
<td>400</td>
<td>200</td>
<td>100</td>
</tr>
<tr>
<td>QPSK, rate ½</td>
<td></td>
<td>3</td>
<td>800</td>
<td>400</td>
<td>200</td>
</tr>
<tr>
<td>QPSK, rate ¾</td>
<td></td>
<td>4</td>
<td>1200(*)</td>
<td>600</td>
<td>300</td>
</tr>
<tr>
<td>16-QAM, rate ½</td>
<td></td>
<td>5</td>
<td>1600(*)</td>
<td>800</td>
<td>400</td>
</tr>
<tr>
<td>16-QAM, rate ¾</td>
<td></td>
<td>6</td>
<td>2400(*)</td>
<td>1200(*)</td>
<td>600</td>
</tr>
</tbody>
</table>

Transmitter Configuration Steps

Table 5-6 OFDM Transmitter Configuration

<table>
<thead>
<tr>
<th>S.No</th>
<th>Action</th>
<th>Register</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Activate the MR-OFDM Physical Layer</td>
<td>PC.PT</td>
</tr>
<tr>
<td>2</td>
<td>Configure MR-OFDM bandwidth option</td>
<td>OFDMC.OPT</td>
</tr>
<tr>
<td>3</td>
<td>Configure modulation and coding scheme</td>
<td>OFDMPHRTX.MCS</td>
</tr>
<tr>
<td>4</td>
<td>Select scrambler seed</td>
<td>OFDMC.SSTX</td>
</tr>
<tr>
<td>5</td>
<td>Configure PIB attribute phyOFDMInterleaving</td>
<td>OFDMC.POI</td>
</tr>
<tr>
<td>6</td>
<td>Specify I/Q data interface rate of the Transmitter Digital Frontend (dependent on OFDMC.OPT)</td>
<td>TXDFE.SR</td>
</tr>
<tr>
<td>7</td>
<td>Specify cut-off frequency of the Transmitter Digital Frontend filter (dependent on OFDMC.OPT)</td>
<td>TXDFE.RCUT</td>
</tr>
<tr>
<td>8</td>
<td>Specify cut-off frequency of the Transmitter Analog Frontend filter (dependent on OFDMC.OPT)</td>
<td>TXCUTC.LPFCUT</td>
</tr>
<tr>
<td>9</td>
<td>Configure transmit power</td>
<td>PAC.TXPWR</td>
</tr>
</tbody>
</table>
Receiver Configuration Steps

Table 5-7 OFDM Receiver Configuration

<table>
<thead>
<tr>
<th>S.No</th>
<th>Action</th>
<th>Register</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Activate the MR-OFDM Physical Layer</td>
<td>PC.PT</td>
</tr>
<tr>
<td>1</td>
<td>Configure MR-OFDM bandwidth option</td>
<td>OFDMC.OPT</td>
</tr>
<tr>
<td>2</td>
<td>Configure PIB attribute phyOFDMInterleaving</td>
<td>OFDMC.POI</td>
</tr>
<tr>
<td>3</td>
<td>Select 'low frequency offset' option if applicable</td>
<td>OFDMC.LFO</td>
</tr>
<tr>
<td>4</td>
<td>Specify I/Q data interface rate of the Receiver Digital Frontend</td>
<td>RXDFE.SR</td>
</tr>
<tr>
<td></td>
<td>(dependent on OFDMC.OPT)</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Specify cut-off frequency of the Receiver Digital Frontend filter</td>
<td>RXDFE.RCUT</td>
</tr>
<tr>
<td></td>
<td>(dependent on OFDMC.OPT and OFDMC.LFO)</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Specify bandwidth and IF frequency of the Receiver Analog Frontend</td>
<td>RXBWC.BW and</td>
</tr>
<tr>
<td></td>
<td>filter (dependent on OFDMC.OPT and OFDMC.LFO)</td>
<td>RXBWC.IFS</td>
</tr>
<tr>
<td>7</td>
<td>Specify AGC measurement period (dependent on OFDMC.OPT)</td>
<td>AGCC.AVGS</td>
</tr>
<tr>
<td>8</td>
<td>Specify AGC speed (dependent on OFDMC.OPT)</td>
<td>AGCC.AGCI</td>
</tr>
<tr>
<td>9</td>
<td>Set AGC target level to 3 (default)</td>
<td>AGCS.TGT</td>
</tr>
<tr>
<td>10</td>
<td>Adjust preamble detector threshold</td>
<td>OFDMSW.PDT</td>
</tr>
</tbody>
</table>

The OFDM modes supported by AT86RF215 are:
- MR-OFDM (2)
  - DFT size 128 with 100, 200, 400, 800, 1200(3), 1600(3), 2400(3) kbit/s
  - DFT size 64 with 50, 100, 200, 400, 600, 800, 1200(3) kbit/s
  - DFT size 32 with 50, 100, 200, 300, 400, 600 kbit/s
  - DFT size 16 with 50, 100, 150, 200, 300 kbit/s
- OFDM PHY
  - Options 3 and 4 of IEEE 802.15.4g as per ETSI TS 102 887-1

Table 5-8 MR-OFDM Register Variable Map

<table>
<thead>
<tr>
<th>Register</th>
<th>Variable</th>
</tr>
</thead>
<tbody>
<tr>
<td>BBCn_PC</td>
<td>phy.modulation set to 1</td>
</tr>
<tr>
<td>Bandwidth Options (OFDMC.OPT)</td>
<td>phy.phy_mode.ofdm.option</td>
</tr>
<tr>
<td>Interleaving (ILV)</td>
<td>tal_pib.OFDMInterleaving</td>
</tr>
<tr>
<td>MCS(Modulation and Coding Scheme)</td>
<td>tal_pib.OFDMMCS</td>
</tr>
</tbody>
</table>

5.4.3. MR-O-QPSK

The AT86RF215 supports simultaneous operation of MR-O-QPSK and legacy O-QPSK. In AT86RF215 O-QPSK operation mode (MR-O-QPSK or legacy O-QPSK) can be configured by setting the PC.PT value to 3.

The following table provides the details about the modes supported for MR-O-QPSK,
### Table 5-9 MR-O-QPSK Modes Support

<table>
<thead>
<tr>
<th>Chip rate [kchip/s]</th>
<th>Rate mode</th>
<th>PSDU data rate [kb/s]</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>12.5</td>
<td>O-QPSK-A-PHY-ID-1 [3]</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>50</td>
<td>O-QPSK-A-PHY-ID-3 [3]</td>
</tr>
<tr>
<td>200</td>
<td>0</td>
<td>12.5</td>
<td>O-QPSK-PHY-OPTION-2 [5]</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>25</td>
<td>Proprietary (as O-QPSK-A-PHY-ID-1; chip rate is 200kchip/s)</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>50</td>
<td>Proprietary (as O-QPSK-A-PHY-ID-2; chip rate is 200kchip/s)</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>100</td>
<td>Proprietary (as O-QPSK-A-PHY-ID-3; chip rate is 200kchip/s)</td>
</tr>
<tr>
<td>1000</td>
<td>0</td>
<td>31.25</td>
<td>O-QPSK-B-PHY-ID-0 [3]</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>125</td>
<td>O-QPSK-B-PHY-ID-1 [3]</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>250</td>
<td>O-QPSK-B-PHY-ID-2 [3]</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>500</td>
<td>O-QPSK-B-PHY-ID-3 [3]</td>
</tr>
<tr>
<td>2000</td>
<td>0</td>
<td>31.25</td>
<td>O-QPSK-C-PHY-ID-0 [3]</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>125</td>
<td>O-QPSK-C-PHY-ID-1 [3]</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>250</td>
<td>O-QPSK-C-PHY-ID-2 [3]</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>500</td>
<td>O-QPSK-C-PHY-ID-3 [3]</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>1000</td>
<td>Proprietary (as O-QPSK-C-PHY-ID-3; spreading bypassed)</td>
</tr>
</tbody>
</table>

The desired chip rate for the transmission can be selected with sub-register OQPSKC0.FCHIP. The desired impulse response of the pulse shaping filter can be selected with sub-register OQPSKC0.MOD.

O-QPSK baseband sub-core of the AT86RF215 contains two specific receiver units, RX_0 for the receiving of MR-O-QPSK frames and RX_1 for the receiving of legacy O-QPSK frames but the reception is based on the Receive Mode (OQPSKC2.RXM) value. The sensitivity level for preamble detection can be configured with sub-registers OQPSKC1.PDT0 and OQPSKC1.PDT1 and SFD words can be configured with sub-register OQPSKC3.NSFD.
Table 5-10  MR-O-QPSK Register Variable Map

<table>
<thead>
<tr>
<th>Register</th>
<th>Variable</th>
</tr>
</thead>
<tbody>
<tr>
<td>BBCn_PC</td>
<td>phy.modulation set to 2</td>
</tr>
<tr>
<td>Chip rate (OQPSKC0.FCHIP)</td>
<td>phy.phy_mode.oqpsk.chip_rate</td>
</tr>
<tr>
<td>Receive Mode (OQPSKC2.RXM)</td>
<td>conf_leg_oqpsk() in tal_phyCfg.c</td>
</tr>
</tbody>
</table>

5.5. State Machine Control

The AT86RF215 state machine forms the basis for both Baseband mode and I/Q radio operating modes.

Figure 5-4  AT86RF215 State Machine
The current transceiver state is determined by reading the TransceiverState register (RFn_STATE). A state change is initiated by writing a command to the Transceiver Command register (RFn_CMD).

Table 5-11 AT86RF215 Commands

<table>
<thead>
<tr>
<th>Sub-register</th>
<th>Name</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CMD</td>
<td>RF_NOP</td>
<td>0x0</td>
<td>NO OPERATION</td>
</tr>
<tr>
<td></td>
<td>RF_SLEEPS</td>
<td>0x1</td>
<td>SLEEP</td>
</tr>
<tr>
<td></td>
<td>RF_TRXOFF</td>
<td>0x2</td>
<td>TRXOFF (Transceiver off, SPI active)</td>
</tr>
<tr>
<td></td>
<td>RF_TXPREP</td>
<td>0x3</td>
<td>TRPREP (Transmit preperation)</td>
</tr>
<tr>
<td></td>
<td>RF_TXPREP</td>
<td>0x4</td>
<td>TX (Transmit)</td>
</tr>
<tr>
<td></td>
<td>RF_RX</td>
<td>0x5</td>
<td>RX (Receive)</td>
</tr>
<tr>
<td></td>
<td>RF_RESET</td>
<td>0x7</td>
<td>RESET (transceiver reset, the transceiver state will automatically end up in state TRXOFF)</td>
</tr>
</tbody>
</table>

5.6. Interrupt Handling

The radios and the basebands of the AT86RF215 generate interrupt events. All enabled interrupt events are logically OR'd to form the single external interrupt signal at pin IRQ. The IRQ behavior and the pad driver strength can be configured by the IRQ Configuration register (RF_CFG).

An interrupt indication may be associated with some state changes or frame transmission or reception. The corresponding interrupt can be read from the Radio IRQ Status register (RFn_IRQS) or Baseband IRQ Status register (BBCn_IRQS).

Table 5-12 AT86RF215 Interrupts

<table>
<thead>
<tr>
<th>Baseband or Radio</th>
<th>Interrupt</th>
<th>Scenario</th>
</tr>
</thead>
<tbody>
<tr>
<td>Radio Interrupt</td>
<td>IRQS.WAKEUP</td>
<td>Returning from the state DEEP_SLEEP, the interrupt IRQS.WAKEUP is always set for both radios together. On the other hand, the interrupt IRQS.WAKEUP is set only for the corresponding radio that returns from the State SLEEP. Once a Chip Reset procedure is completed, the interrupt IRQS.WAKEUP is issued for both transceivers. A Transceiver Reset completion is indicated by the interrupt IRQS.WAKEUP for the corresponding transceiver only.</td>
</tr>
<tr>
<td>Radio Interrupt</td>
<td>IRQS.TRXRDY</td>
<td>IRQ TRXRDY is issued once the frequency settling is completed</td>
</tr>
<tr>
<td>Radio Interrupt</td>
<td>IRQS.EDC</td>
<td>Energy Measurement Completion</td>
</tr>
<tr>
<td>Radio Interrupt</td>
<td>IRQS.BATLOW</td>
<td>Battery Monitor (BATMON) Low indication</td>
</tr>
<tr>
<td>Radio Interrupt</td>
<td>IRQS.TRXERR</td>
<td>Transceiver error is detected</td>
</tr>
<tr>
<td>Radio Interrupt</td>
<td>IRQS.IQIFS</td>
<td>The I/Q data interface synchronization fails</td>
</tr>
</tbody>
</table>
Baseband or Radio | Interrupt | Scenario
--- | --- | ---
Baseband Interrupt | IRQS.RXFS | This interrupt is issued if a valid PHY header is detected during frame receive.
Baseband Interrupt | IRQS.RXFE | The IRQ RXFE is issued at the end of a successful frame reception.
Baseband Interrupt | IRQS.RXEM | This interrupt occurs during frame receive if the Address Filter is enabled and if the received frame is recognized as extended.
Baseband Interrupt | IRQS.RXAM | This interrupt occurs during frame receive if the Address Filter is enabled and if the received frame is recognized as matching.
Baseband Interrupt | IRQS.TXFE | The IRQ_TXFE is issued when a frame is completely transmitted.
Baseband Interrupt | IRQS.AGCH | The interrupt AGCH is issued during frame receive if a preamble of the selected PHY is detected.
Baseband Interrupt | IRQS.AGCR | The interrupt AGCR is issued during frame receive if a receive process is finished.
Baseband Interrupt | IRQS.FBLI | If the pre-programmed number of octets have been received (i.e. stored to the frame buffer), the IRQ FBLI is issued.

The interrupts are handled in the `trx_irq_handler_cb()` function available in the `tal_irq_handler.c`.

### 5.7. Auto Mode Configuration

The AT86RF215 provides embedded IEEE MAC support reducing the MCU to transceiver interaction.

The Auto mode features supported by the AT86RF215 are,
- Frame Filter
- Clear Channel Assessment with Automatic Transmit (CCATX)
- Transmit and Switch to Receive (TX2RX)
- Automatic Acknowledgment (AACK)

#### Table 5-13 Auto Mode Feature Register Map

<table>
<thead>
<tr>
<th>Register</th>
<th>Used by MAC procedure</th>
</tr>
</thead>
<tbody>
<tr>
<td>BBCn_OQPSKPHRTX</td>
<td>AACK</td>
</tr>
<tr>
<td>BBCn_OFDMPHRTX</td>
<td>AACK</td>
</tr>
<tr>
<td>BBCn_FSKPHRTX</td>
<td>AACK</td>
</tr>
<tr>
<td>RFn_CMD</td>
<td>AACK, CCATX, TX2RX</td>
</tr>
</tbody>
</table>

#### 5.7.1. Frame Filter

Frame filtering is a procedure that evaluates whether or not a received frame matches predefined criteria, like address information or frame types. If a frame passes all selected filter criteria, the interrupt receiver frame end (`IRQS.RXFE`) is generated.
Figure 5-5 Frame Filter Flow

start frame parsing

valid PHR detected?  

Y

N

generation IRQ RXFS

AFEN0.3 = 1?  

N

Y

check global rules

rules passed?  

N

Y

click 3rd level filter rules

click extended filter rules

rules passed?  

N

Y

generation IRQ RXAM

generation IRQ RXEM

2nd level of filtering

PM = 1?  

N

Y

IRQ RXAM or RXEM occurred?  

N

Y

1st level of filtering

FCSFE = 1  

Y

N

click FCS

FCS correct?  

N

Y

generation IRQ RXFE

frame parsing finished
• First level filtering is based on the FCS field, if Frame Check Sequence Filter Enabled (PC.FCSFE =1)
• Second level of filtering is based on the Promiscuous Mode enabling (AFC0.PM =1)
• Address filter consists of global filter, 3rd level filter and extended filter
• Global filter is based on the Address Filter Frame Type Mask (BBCn_AFFTM) value and Address Filter Frame Version Mask (BBCn_AFFVM) value which filters the frame type and the frame version respectively
• Third level filter is based on the PAN id, Short address, Extended address, Frame type (Beacon, Data, ACK), and the PAN coordinator status

Table 5-14 Frame Filter Configuration Register

<table>
<thead>
<tr>
<th>Sub-register name</th>
<th>IEEE 802.15.4-2006 attribute or name</th>
<th>Size [bits]</th>
<th>Reset value</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>AFEN*</td>
<td>1</td>
<td>0</td>
<td>If this sub-register is set to 1, the corresponding address filter unit is enabled.</td>
<td></td>
</tr>
<tr>
<td>MACEA0..7</td>
<td>macExtendedAddress</td>
<td>64</td>
<td>0</td>
<td>These registers define the macExtendedAddress.</td>
</tr>
<tr>
<td>MACSHA0F*, MACSHA1F*</td>
<td>macShortAddress</td>
<td>16</td>
<td>0xFFFF</td>
<td>These registers define the macSortAddress.</td>
</tr>
<tr>
<td>MACPID0F*, MACPID1F*</td>
<td>macPANId</td>
<td>16</td>
<td>0xFFFF</td>
<td>These registers define the macPANId.</td>
</tr>
<tr>
<td>PM</td>
<td>macPromiscuousMode</td>
<td>1</td>
<td>0</td>
<td>If this sub-register is set to 1, the frame filter operates in promiscuous mode.</td>
</tr>
<tr>
<td>PANC*</td>
<td>PANCoordinator</td>
<td>1</td>
<td>0</td>
<td>If this sub-register is set to 1, the frame filter operates as a PAN coordinator.</td>
</tr>
<tr>
<td>AFFTM</td>
<td>frame type bitmask</td>
<td>8</td>
<td>0x0B</td>
<td>Address filter frame type bitmask.</td>
</tr>
<tr>
<td>MRFT*</td>
<td>map reserved frame types</td>
<td>1</td>
<td>0</td>
<td>If this sub-register is set to 1, incoming frames with reserved frame types [4..7] are filtered like data frames. If the frame header matches the address filter, RXAM IRQ occurs. A RXEM IRQ is issued anyway.</td>
</tr>
<tr>
<td>AFFVM</td>
<td>frame version bitmask</td>
<td>4</td>
<td>3</td>
<td>Address filter frame version mask.</td>
</tr>
</tbody>
</table>
### Sub-register name | IEEE 802.15.4-2006 attribute or name | Size [bits] | Reset value | Comment
---|---|---|---|---
AM* | address filter status | 1 | 0 | If this bit is set to 1, the address filter has detected an address match. This bit is updated once the MAC header of an incoming frame is completely parsed.

| Sub-register name | IEEE 802.15.4-2006 attribute or name | Size [bits] | Reset value | Comment
---|---|---|---|---
EM | extended match status | 1 | 0 | If this bit is set to 1, the frame filter has detected an extended match. The bit is updated once the MAC header of an incoming frame is completely parsed.

**Note:** Register names marked with a star “*” are indexed with the last digit to be valid for each of the four 3rd level filter units (i.e. AFEN* is implemented as AFEN0, AFEN1, AFEN2, AFEN3).

An incoming frame which passes all global rules and all third level filter rules triggers the address match interrupt (IRQS.RXAM). If the frame passes the extended rules then the extended match interrupt (IRQS.RXEM) is generated.

Multi Address Filtering (MAF) feature is available in the AT86RF215 which used to satisfy the address filtering of the device may be connected to different networks. The Address Filter Status register (BBCn_AFS) indicates which unit has actually an address match detected.

#### 5.7.2. Clear Channel Assessment with Automatic Transmit (CCATX)

The procedure CCATX provides a clear channel assessment (CCA) mechanism with an automatic frame transmission if the channel is assessed as idle. The procedure CCATX is enabled if Measurement and automatic Transmit (AMCS.CCATX: CCA) bit is set. If measured ED value is greater than the Auto Mode Energy Detection Threshold (BBCn_AMEDT) then the TXFE is issued. Otherwise, the frame will be transmitted.

#### 5.7.3. Transmit and Switch to Receive (TX2RX)

Transmit and switch to receive is a procedure that switches the transceiver state to RX after transmission based on the AMCS.TX2RX value.

#### 5.7.4. Automatic Acknowledgment (AACK)

Automatic Acknowledgment is a procedure that transmits automatically an acknowledgment (ACK) frame for the acknowledgment requested MAC data and command frames which passes the third level frame filtering.

**AACK Features:**

- Automatic sequence number handling
- Automatic FCS generation
- Configurable timing between RX frame end and ACK frame start based on the Auto Mode Automatic ACK Time (BBCn_AMAACKTx)
- Data rate of ACK frame can be determined by incoming frame or by predefined TX settings based on the Auto Acknowledgment Data Rate (AMCS.AACKDR)
- Configuration of the ACK frame content source: Either the ACK frame content is generated internally or the ACK frame is sent from the TX frame buffer (requires controller interaction) based on the Auto Acknowledgment Source (AMCS.AACKS).
• FCS type of the ACK can be derived from incoming frame or from predefined settings based on the Auto Acknowledgment FCS Adaption (AMCS.AACKFA)
6. **Creating Custom Configuration**

Create custom configuration using the following steps,

1. Add custom configuration as part of option 5 user configuration.
2. Copy the code available inside sub-function based on the desired modulation from function `set_mod()` in `phy_conf.c`.
3. Paste the sub-function (`set_fsk/set_ofdm/set_oqpsk/set_leg_oqpsk`) code inside the `set_user_config()` function.
4. Modify the required modulation parameters such as chip rate, frequency band, channel spacing, center frequency, current channel, transmit power, etc... based on the user configuration requirements.
5. Create the hex file with the modified code.
6. Flash modified hex file in two devices.
7. Select option 5 user config and proceed with chat messages.

Refer the following sample code for user configuration.

```c
#ifdef USER_CONFIGURATION
/**
 * @brief Set User config
 *
 * @param trx_id Transceiver identifier
 *
 * @return MAC_SUCCESS or FAILURE
 */
retval_t set_user_config(trx_id_ttrx_id)
{
    printf("The User can add their custom configuration in phy_conf.c - > set_user_config()\n");

    phy_t phy;
    retval_t status;
    if (tal_trx_sleep(RF09, SLEEP_MODE_1) == MAC_SUCCESS)
    {
        sleep_enabled[RF09] = true;
    }

    trx_id = RF24;
    current_trx_id = RF24;
    current_mod[RF24] = USER_CONFIG;

    phy.modulation = OFDM;
    phy.phy_mode.ofdm.option = OFDM_OPT_1;

    phy.freq_band = WORLD_2450;
    phy.ch_spacing = OFDM_2450_OPT1_CH_SPAC;    //IEEE802.15.4g-2012 spec,
    Table 148-Data Rates for MR-OFDM PHY
    phy.freq_f0 = 2450000000;    //2.45GHz as center frequency

    /* Set interleaving */
    bool interl = true;
    status = tal_pib_set(trx_id, phyOFDMInterleaving, (pib_value_t *)&interl);
```

---

```
#endif
```
if (status != MAC_SUCCESS)
{
    return status;
}

/* Set data rate / MCS */
ofdm_mcs_t mcs = MCS5;
status = tal_pib_set(trx_id, phyOFDMMCS, (pib_value_t *)&mcs);
if (status != MAC_SUCCESS)
{
    return status;
}

/* CRC */
uint16_t crc_type = FCS_TYPE_4_OCTETS;
status = tal_pib_set(trx_id, macFCSType, (pib_value_t *)&crc_type);
if (status != MAC_SUCCESS)
{
    return status;
}

/* Set modulation / PHY configuration */
status = tal_pib_set(trx_id, phySetting, (pib_value_t *)&phy);
if (status != MAC_SUCCESS)
{
    return status;
}

/* Set channel */
channel_t ch = 0;
status = tal_pib_set(trx_id, phyCurrentChannel, (pib_value_t *)&ch);
if (status != MAC_SUCCESS)
{
    return status;
}

/* Set transmit power */
int pwr = 14;
status = tal_pib_set(trx_id, phyTransmitPower, (pib_value_t *)&pwr);
if (status != MAC_SUCCESS)
{
    return status;
}

return MAC_SUCCESS;

#endif /* #ifdef USER_CONFIGURATION*/
### API References

The following API's are available in the Chat application. The file name and its description are provided in this table.

**Table 7-1 AT86RF215 Chat Application API Description**

<table>
<thead>
<tr>
<th>API</th>
<th>File name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void tal_ed_end_cb(trx_id_t trx_id, uint8_t energy_level)</td>
<td>Main.c</td>
<td>Call back function indicating the result of ED Scan</td>
</tr>
<tr>
<td>void tal_rx_frame_cb(trx_id_t trx_id, frame_info_t *rx_frame)</td>
<td>Main.c</td>
<td>Call back function indicating reception of frame</td>
</tr>
<tr>
<td>void tal_tx_frame_done_cb(trx_id_t trx_id, retval_t status, frame_info_t *frame)</td>
<td>Main.c</td>
<td>Call back function indicating status of the frame transmission</td>
</tr>
<tr>
<td>retval_t set_fsk(trx_id_t trx_id)</td>
<td>phy_conf.c</td>
<td>Set FSK modulation</td>
</tr>
<tr>
<td>retval_t set_ofdm(trx_id_t trx_id)</td>
<td>phy_conf.c</td>
<td>Set OFDM modulation</td>
</tr>
<tr>
<td>retval_t set_oqpsk(trx_id_t trx_id)</td>
<td>phy_conf.c</td>
<td>Set MR-OQPSK modulation</td>
</tr>
<tr>
<td>retval_t set_leg_oqpsk(trx_id_t trx_id)</td>
<td>phy_conf.c</td>
<td>Set legacy OQPSK modulation</td>
</tr>
<tr>
<td>retval_t set_user_config(trx_id_t trx_id)</td>
<td>phy_conf.c</td>
<td>Set User config</td>
</tr>
<tr>
<td>void chat_handle_incoming_frame(trx_id_t trx_id, frame_info_t *rx_frame)</td>
<td>Chat.c</td>
<td>Frame reception callback function for the chat application</td>
</tr>
<tr>
<td>void chat_tx_done_cb(trx_id_t trx_id, retval_t status, frame_info_t *frame)</td>
<td>Chat.c</td>
<td>Frame transmission callback function for the chat application</td>
</tr>
</tbody>
</table>
8. **Reference**

- **Data Sheet:**
  

- **ATREB215 Xplained PRO extension board:**
  

- **Web Pages**
  
  - **RF215 device page:**
    

- **ATREB215 Xplained PRO extension board:**
  
9. **Revision History**

<table>
<thead>
<tr>
<th>Doc Rev.</th>
<th>Date</th>
<th>Comments</th>
</tr>
</thead>
</table>
Atmel Corporation
1600 Technology Drive, San Jose, CA 95110 USA
T: (+1)(408) 441.0311  F: (+1)(408) 436.4200  |  www.atmel.com


Atmel®, Atmel logo and combinations thereof, Enabling Unlimited Possibilities®, and others are registered trademarks or trademarks of Atmel Corporation in U.S. and other countries. Other terms and product names may be trademarks of others.

DISCLAIMER: The information in this document is provided in connection with Atmel products. No license, express or implied, by estoppel or otherwise, to any intellectual property right is granted by this document or in connection with the sale of Atmel products. EXCEPT AS SET FORTH IN THE ATMEL TERMS AND CONDITIONS OF SALES LOCATED ON THE ATMEL WEBSITE, ATMEL ASSUMES NO LIABILITY WHATSOEVER AND DISCLAIMS ANY EXPRESS, IMPLIED OR STATUTORY WARRANTY RELATING TO ITS PRODUCTS INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT. IN NO EVENT SHALL ATMEL BE LIABLE FOR ANY DIRECT, INDIRECT, CONSEQUENTIAL, PUNITIVE, SPECIAL OR INCIDENTAL DAMAGES (INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS AND PROFITS, BUSINESS INTERRUPTION, OR LOSS OF INFORMATION) ARISING OUT OF THE USE OR INABILITY TO USE THIS DOCUMENT, EVEN IF ATMEL HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. Atmel makes no representations or warranties with respect to the accuracy or completeness of the contents of this document and reserves the right to make changes to specifications and products descriptions at any time without notice. Atmel does not make any commitment to update the information contained herein. Unless specifically provided otherwise, Atmel products are not suitable for, and shall not be used in, automotive applications. Atmel products are not intended, authorized, or warranted for use as components in applications intended to support or sustain life.

SAFETY-CRITICAL, MILITARY, AND AUTOMOTIVE APPLICATIONS DISCLAIMER: Atmel products are not designed for and will not be used in connection with any applications where the failure of such products would reasonably be expected to result in significant personal injury or death ("Safety-Critical Applications") without an Atmel officer's specific written consent. Safety-Critical Applications include, without limitation, life support devices and systems, equipment or systems for the operation of nuclear facilities and weapons systems. Atmel products are not designed nor intended for use in military or aerospace applications or environments unless specifically designated by Atmel as military-grade. Atmel products are not designed nor intended for use in automotive applications unless specifically designated by Atmel as automotive-grade.