Digital Transceiver for the Raspberry Pi

1.1

Generated by Doxygen 1.8.13

Contents

Chapter 1

Overview

The AX5043 library provides an interface to program the transceiver. The library is currently configured with the following settings:

- Carrier Frequency - 435.3 MHz
- Symbol Rate - 4.8 kS/s
- Modulation - GFSK
- Transmit Deviation - 13.6 kHz
- Transmit Power - 15.0 dBm
- Receive Bandwidth - 28.2 kHz
- Encoding: HDLC with FEC
- Error Detection: CRC-16

Next Steps

Over time, the AX5043 library will have new functions to change these settings.

Chapter 2

Data Structure Index

2.1 Data Structures

Here are the data structures with brief descriptions:

Chapter 3

File Index

3.1 File List

Here is a list of all documented files with brief descriptions:

Chapter 4

Data Structure Documentation

4.1 axradio_address Struct Reference

Structure containing a four byte X.25 address.

`#include <axradioinit_p.h>`

Data Fields

- uint8_t addr [4]
  Four byte X.25 address.

4.1.1 Detailed Description

Structure containing a four byte X.25 address.

The documentation for this struct was generated from the following file:

- axradio/axradioinit_p.h

Chapter 5

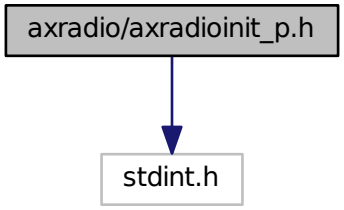
File Documentation

5.1 axradio/axradioinit_p.h File Reference

Provides an interface to initialze the AX5043 transceiver.

```
#include <stdint.h>
```

Include dependency graph for axradioinit_p.h:



Data Structures

- struct axradio_address

  Structure containing a four byte X.25 address.

Macros

- #define AXRADIO_ERR_NOERROR 0x00

  Operation successful.
- #define AXRADIO_ERR_NOTSUPPORTED 0x01

  Operation not supported.
- #define AXRADIO_ERR_BUSY 0x02

  Transceiver busy.

- #define AXRADIO_ERR_TIMEOUT 0x03

  Operation timed out.
- #define AXRADIO_ERR_INVALID 0x04

  Invalid parameter.
- #define AXRADIO_ERR_NOCHIP 0x05

  Transceiver not found.
- #define AXRADIO_ERR_RANGING 0x06

  Frequency could not be ranged.
- #define AXRADIO_ERR_LOCKLOST 0x07

  Lost PLL lock.
- #define AXRADIO_ERR_RETRANSMISSION 0x08

  Retrasnmitted packet.
- #define AXRADIO_ERR_RESYNC 0x09

  Restarts synchronization.
- #define AXRADIO_ERR_RESYNCTIMEOUT 0x0a

  Synchronization timed out.
- #define AXRADIO_ERR_RECEIVESTART 0x0b

  Receiver restarted.

Functions

- uint8_t axradio_init (void)

  Initialize the AX5043 radio transceiver.
- uint8_t axradio_setfreq (int32_t f)

  Set the receive and transmit frequency.

5.1.1 Detailed Description

Provides an interface to initialze the AX5043 transceiver.

Copyright

endorse or promote products derived from this software without specific prior written permission. 4.All advertising materials mentioning features or use of this software must display the following acknowledgement: This product includes software developed by AXSEM AG and its contributors. 5.The usage of this source code is only granted for operation with AX5043 and AX8052F143. Porting to other radio or communication devices is strictly prohibited.

THIS SOFTWARE IS PROVIDED BY AXSEM AG AND CONTRIBUTORS “AS IS" AND ANY EXPRESS OR IMPL↩ IED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL AXSEM AG AND CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CON↩ SEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEG↩ LIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

5.1.2 Function Documentation

5.1.2.1 axradio_init()

```
uint8_t axradio_init (
            void  )
```

Initialize the AX5043 radio transceiver.

Returns

AXRADIO_ERR_NOERROR on success, otherwise a value indicating an error.

See also

AXRADIO_ERR_NOERROR

5.1.2.2 axradio_setfreq()

```
uint8_t axradio_setfreq (
            int32_t f )
```

Set the receive and transmit frequency.

Parameters

f	The frequency in Hertz.

Returns

AXRADIO_ERR_NOERROR on success, otherwise a value indicating an error.

See also

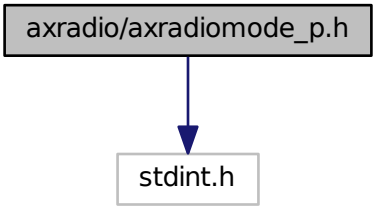
AXRADIO_ERR_NOERROR

5.2 axradio/axradiomode_p.h File Reference

Provides an interface to change the transceiver mode.

```
#include <stdint.h>
```

Include dependency graph for axradiomode_p.h:



Functions

- uint8_t mode_tx (void)

  Switch the tranceiver into transmit mode.
- uint8_t mode_rx (void)

  Switch the tranceiver into receive mode.

5.2.1 Detailed Description

Provides an interface to change the transceiver mode.

5.2.2 Function Documentation

5.2.2.1 mode_rx()

```
uint8_t mode_rx (
              void  )
```

Switch the tranceiver into receive mode.

The receive buffer may contain garbage and reading from the buffer will obtain and discard that garbage.

Returns

AXRADIO_ERROR_NOERROR on success, otherwise a value indicating an error.

See also

AXRADIO_ERR_NOERROR

5.2.2.2 mode_tx()

```
uint8_t mode_tx (
            void  )
```

Switch the tranceiver into transmit mode.

Returns

AXRADIO_ERROR_NOERROR on success, otherwise a value indicating an error.

See also

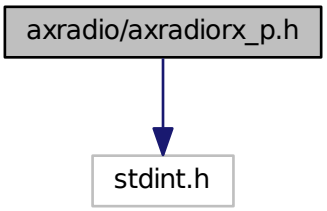
AXRADIO_ERR_NOERROR

5.3 axradio/axradiorx_p.h File Reference

Provides an interface to receive packets using the digital transceiver.

```
#include <stdint.h>
```

Include dependency graph for axradiorx_p.h:



Functions

- uint8_t receive_packet (void)

  Receive a packet from the digital transceiver receive buffer.

5.3.1 Detailed Description

Provides an interface to receive packets using the digital transceiver.

Copyright

5.3.2 Function Documentation

5.3.2.1 receive_packet()

```
uint8_t receive_packet (
            void  )
```

Receive a packet from the digital transceiver receive buffer.

Returns

AXRADIO_ERROR_NOERROR on success, otherwise a value indicating an error.

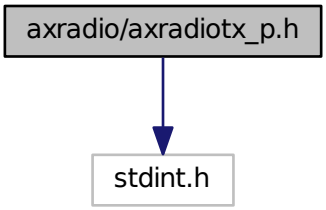
See also

AXRADIO_ERR_NOERROR

5.4 axradio/axradiotx_p.h File Reference

Provides an interface to transmit packets using the digital transceiver.

`#include <stdint.h>`
Include dependency graph for axradiotx_p.h:



Functions

- uint8_t transmit_packet (const struct axradio_address *addr, const uint8_t *pkt, uint16_t pktlen)

  Transmit a packet using the digital transceiver.

5.4.1 Detailed Description

Provides an interface to transmit packets using the digital transceiver.

Copyright

materials mentioning features or use of this software must display the following acknowledgement: This product includes software developed by AXSEM AG and its contributors. 5.The usage of this source code is only granted for operation with AX5043 and AX8052F143. Porting to other radio or communication devices is strictly prohibited.

THIS SOFTWARE IS PROVIDED BY AXSEM AG AND CONTRIBUTORS “AS IS" AND ANY EXPRESS OR IMPL↩ IED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL AXSEM AG AND CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CON↩ SEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEG↩ LIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

5.4.2 Function Documentation

5.4.2.1 transmit_packet()

```
uint8_t transmit_packet (
            const struct axradio_address * addr,
            const uint8_t * pkt,
            uint16_t pktlen )
```

Transmit a packet using the digital transceiver.

Parameters

addr	The address of the desired destiantion radio, if used.
pkt	The byte data to be transmitted.
pktlen	The number of bytes in pkt to be transmitted.

Returns

AXRADIO_ERROR_NOERROR on success, otherwise a value indicating an error.

See also

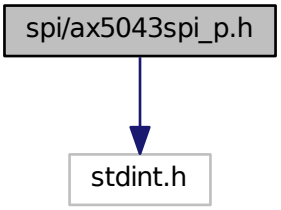
AXRADIO_ERR_NOERROR

5.5 spi/ax5043spi_p.h File Reference

Provides an abstraction layer for the SPI interface communicating to the digital transceiver.

```
#include <stdint.h>
```

Include dependency graph for ax5043spi_p.h:



Macros

- #define SPI_CHANNEL (0)

  The default SPI channel for the digital transceiver.
- #define SPI_SPEED (32000000)

  The default SPI bus speed for the digital transceiver.

Functions

- void setSpiChannel (int newSpiChannel)

  Set the SPI channel for the digital transceiver.
- void setSpiSpeed (int newSpiSpeed)

  Set the SPI bus speed for the digital transceiver.
- void initializeSpi (void)

  Initilize the SPI bus to communicate with the digital transceiver.
- void ax5043WriteReg (uint16_t reg, uint8_t val)

  Write a value to an AX5043 register.
- uint8_t ax5043ReadReg (uint16_t reg)

  Read a value from an AX5043 register.

5.5.1 Detailed Description

Provides an abstraction layer for the SPI interface communicating to the digital transceiver.

Copyright

5.5.2 Function Documentation

5.5.2.1 ax5043ReadReg()

```
uint8_t ax5043ReadReg (
            uint16_t reg )
```

Read a value from an AX5043 register.

Parameters

reg	The register to read.

Returns

The value read from the register.

5.5.2.2 ax5043WriteReg()

```
void ax5043WriteReg (
            uint16_t reg,
            uint8_t val )
```

Write a value to an AX5043 register.

Parameters

reg	The register to write.
val	The value to right to the register.

5.5.2.3 initializeSpi()

```
void initializeSpi (
            void  )
```

Initilize the SPI bus to communicate with the digital transceiver.

setSpiChannel() and setSpiSpeed() must both be called before initializeSPI().

See also

setSpiChannel
setSpiSpeed

5.5.2.4 setSpiChannel()

```
void setSpiChannel (
            int newSpiChannel )
```

Set the SPI channel for the digital transceiver.

setSpiChannel must be called before initializeSpi(). The default is SPI_CHANNEL.

Parameters

newSpiChannel	The SPI channel for the digital transceiver.

See also

SPI_CHANNEL
initializeSpi

5.5.2.5 setSpiSpeed()

```
void setSpiSpeed (
            int newSpiSpeed )
```

Set the SPI bus speed for the digital transceiver.

setSpiSpeed must be called before initializeSpi(). The default is SPI_SPEED.

Parameters

newSpiSpeed	The SPI bus speed for the digital transceiver.

See also

SPI_SPEED
initializeSpi

Index