Si4735 Arduino Library

AUTHOR

Version 1.1.8

Thu Apr 2 2020

Table of Contents

Class Index

Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

File Index

File List

Here is a list of all files with brief descriptions:

Class Documentation

SI4735 Class Reference

```
#include <SI4735.h>
```

Public Member Functions

- **SI4735** ()
- void **reset** (void)
- void **waitToSend** (void)
- void **setup** (uint8_t **resetPin**, uint8_t defaultFunction)
- void **setup** (uint8_t **resetPin**, int **interruptPin**, uint8_t defaultFunction, uint8_t audioMode=**SI473X_ANALOG_AUDIO**)
- void **setPowerUp** (uint8_t CTSIEN, uint8_t GPO2OEN, uint8_t PATCH, uint8_t XOSCEN, uint8_t FUNC, uint8_t OPMODE)
- void **radioPowerUp** (void)
- void **analogPowerUp** (void)
- void **powerDown** (void)
- void **setFrequency** (uint16_t)
- void **getStatus** ()
- void **getStatus** (uint8_t, uint8_t)
- uint16_t **getFrequency** (void)
- uint16_t **getCurrentFrequency** ()
- bool **getSignalQualityInterrupt** ()
- bool **getRadioDataSystemInterrupt** ()
  Gets Received Signal Quality Interrupt(RSQINT)
- bool **getTuneCompleteTriggered** ()
  Gets Radio Data System (RDS) Interrupt.
- bool **getStatusError** ()
  Seek/Tune Complete Interrupt; 1 = Tune complete has been triggered.
- bool **getStatusCTS** ()
  Return the Error flag (true or false) of status of the least Tune or Seek.
- bool **getACFIndicator** ()
  Gets the Error flag of status response.
- bool **getBandLimit** ()
  Returns true if the AFC rails (AFC Rail Indicator).
- bool **getStatusValid** ()
  Returns true if a seek hit the band limit (WRAP = 0 in FM_START_SEEK) or wrapped to the original frequency(WRAP = 1).
- uint8_t **getReceivedSignalStrengthIndicator** ()
  Returns true if the channel is currently valid as determined by the seek/tune properties (0x1403, 0x1404, 0x1108)

- uint8_t **getStatusSNR** ()
  Returns integer Received Signal Strength Indicator (dBÎ¼V).

- uint8_t **getStatusMULT** ()
  Returns integer containing the SNR metric when tune is complete (dB).

- uint8_t **getAntennaTuningCapacitor** ()
  Returns integer containing the multipath metric when tune is complete.

- void **getAutomaticGainControl** ()
  Returns integer containing the current antenna tuning capacitor value.

- void **setAvcAmMaxGain** (uint8_t gain)
- void **setAvcAmMaxGain** ()
- uint8_t **getCurrentAvcAmMaxGain** ()
- void **setAmSoftMuteMaxAttenuation** (uint8_t smattn)
- void **setAmSoftMuteMaxAttenuation** ()
- void **setSsbSoftMuteMaxAttenuation** (uint8_t smattn)
- void **setSsbSoftMuteMaxAttenuation** ()
- bool **isAgcEnabled** ()
- uint8_t **getAgcGainIndex** ()
- void **setAutomaticGainControl** (uint8_t AGCDIS, uint8_t AGCIDX)
- void **getCurrentReceivedSignalQuality** (uint8_t INTACK)
- void **getCurrentReceivedSignalQuality** (void)
- uint8_t **getCurrentRSSI** ()
- uint8_t **getCurrentSNR** ()
  current receive signal strength (0â€“127 dBÎ¼V).

- bool **getCurrentRssiDetectLow** ()
  current SNR metric (0–127 dB).

- bool **getCurrentRssiDetectHigh** ()
  RSSI Detect Low.

- bool **getCurrentSnrDetectLow** ()
  RSSI Detect High.

- bool **getCurrentSnrDetectHigh** ()
  SNR Detect Low.

- bool **getCurrentValidChannel** ()
  SNR Detect High.

- bool **getCurrentAfcRailIndicator** ()
  Valid Channel.

- bool **getCurrentSoftMuteIndicator** ()
  AFC Rail Indicator.

- uint8_t **getCurrentStereoBlend** ()
  Soft Mute Indicator. Indicates soft mute is engaged.

- bool **getCurrentPilot** ()
  Indicates amount of stereo blend in % (100 = full stereo, 0 = full mono).

- uint8_t **getCurrentMultipath** ()
  Indicates stereo pilot presence.

- uint8_t **getCurrentSignedFrequencyOffset** ()
  Contains the current multipath metric. (0 = no multipath; 100 = full multipath)

- bool **getCurrentMultipathDetectLow** ()
  Signed frequency offset (kHz).

- bool **getCurrentMultipathDetectHigh** ()
  Multipath Detect Low.

- bool **getCurrentBlendDetectInterrupt** ()
  Multipath Detect High.

- uint8_t **getFirmwarePN** ()
  Blend Detect Interrupt.

- uint8_t **getFirmwareFWMAJOR** ()
  RESP1 - Part Number (HEX)

- uint8_t **getFirmwareFWMINOR** ()
  RESP2 - Returns the Firmware Major Revision (ASCII).

- uint8_t **getFirmwarePATCHH** ()
  RESP3 - Returns the Firmware Minor Revision (ASCII).

- uint8_t **getFirmwarePATCHL** ()
  RESP4 - Returns the Patch ID High byte (HEX).

- uint8_t **getFirmwareCMPMAJOR** ()
  RESP5 - Returns the Patch ID Low byte (HEX).

- uint8_t **getFirmwareCMPMINOR** ()
  RESP6 - Returns the Component Major Revision (ASCII).

- uint8_t **getFirmwareCHIPREV** ()
  RESP7 - Returns the Component Minor Revision (ASCII).

- void **setVolume** (uint8_t **volume**)

RESP8 - Returns the Chip Revision (ASCII).

- uint8_t **getVolume** ()
- void **volumeDown** ()
- void **volumeUp** ()
- uint8_t **getCurrentVolume** ()
- void **setAudioMute** (bool off)
  Returns the current volume level.

- void **digitalOutputFormat** (uint8_t OSIZE, uint8_t OMONO, uint8_t OMODE, uint8_t OFALL)
- void **digitalOutputSampleRate** (uint16_t DOSR)
- void **setAM** ()
- void **setFM** ()
- void **setAM** (uint16_t fromFreq, uint16_t toFreq, uint16_t intialFreq, uint16_t step)
- void **setFM** (uint16_t fromFreq, uint16_t toFreq, uint16_t initialFreq, uint16_t step)
- void **setBandwidth** (uint8_t AMCHFLT, uint8_t AMPLFLT)
- void **setFrequencyStep** (uint16_t step)
- uint8_t **getTuneFrequencyFast** ()
- void **setTuneFrequencyFast** (uint8_t FAST)
  Returns the FAST tuning status.

- uint8_t **getTuneFrequencyFreeze** ()
  FAST Tuning. If set, executes fast and invalidated tune. The tune status will not be accurate.

- void **setTuneFrequencyFreeze** (uint8_t FREEZE)
  Returns the FREEZE status.

- void **setTuneFrequencyAntennaCapacitor** (uint16_t capacitor)
  Onlye FM. Freeze Metrics During Alternate Frequency Jump.

- void **frequencyUp** ()
- void **frequencyDown** ()
- bool **isCurrentTuneFM** ()
- void **getFirmware** (void)
- void **setFunction** (uint8_t FUNC)
- void **seekStation** (uint8_t SEEKUP, uint8_t WRAP)
- void **seekStationUp** ()
- void **seekStationDown** ()
- void **setSeekAmLimits** (uint16_t bottom, uint16_t top)
- void **setSeekAmSpacing** (uint16_t spacing)
- void **setSeekSrnThreshold** (uint16_t value)
- void **setSeekRssiThreshold** (uint16_t value)
- void **setFmBlendStereoThreshold** (uint8_t parameter)
- void **setFmBlendMonoThreshold** (uint8_t parameter)
- void **setFmBlendRssiStereoThreshold** (uint8_t parameter)
- void **setFmBLendRssiMonoThreshold** (uint8_t parameter)
- void **setFmBlendSnrStereoThreshold** (uint8_t parameter)
- void **setFmBLendSnrMonoThreshold** (uint8_t parameter)
- void **setFmBlendMultiPathStereoThreshold** (uint8_t parameter)
- void **setFmBlendMultiPathMonoThreshold** (uint8_t parameter)
- void **setFmStereoOn** ()
- void **setFmStereoOff** ()
- void **RdsInit** ()

- void **setRdsIntSource** (uint8_t RDSNEWBLOCKB, uint8_t RDSNEWBLOCKA, uint8_t RDSSYNCFOUND, uint8_t RDSSYNCLOST, uint8_t RDSRECV)
- void **getRdsStatus** (uint8_t INTACK, uint8_t MTFIFO, uint8_t STATUSONLY)
- void **getRdsStatus** ()
- bool **getRdsReceived** ()
- bool **getRdsSyncLost** ()
  1 = FIFO filled to minimum number of groups

- bool **getRdsSyncFound** ()
  1 = Lost RDS synchronization

- bool **getRdsNewBlockA** ()
  1 = Found RDS synchronization

- bool **getRdsNewBlockB** ()
  1 = Valid Block A data has been received.

- bool **getRdsSync** ()
  1 = Valid Block B data has been received.

- bool **getGroupLost** ()
  1 = RDS currently synchronized.

- uint8_t **getNumRdsFifoUsed** ()
  1 = One or more RDS groups discarded due to FIFO overrun.

- void **setRdsConfig** (uint8_t RDSEN, uint8_t BLETHA, uint8_t BLETHB, uint8_t BLETHC, uint8_t BLETHD)
  RESP3 - RDS FIFO Used; Number of groups remaining in the RDS FIFO (0 if empty).

- uint16_t **getRdsPI** (void)
- uint8_t **getRdsGroupType** (void)
- uint8_t **getRdsFlagAB** (void)
- uint8_t **getRdsVersionCode** (void)
- uint8_t **getRdsProgramType** (void)
- uint8_t **getRdsTextSegmentAddress** (void)
- char * **getRdsText** (void)
- char * **getRdsText0A** (void)
- char * **getRdsText2A** (void)
- char * **getRdsText2B** (void)
- char * **getRdsTime** (void)
- void **getNext2Block** (char *)
- void **getNext4Block** (char *)
- void **ssbSetup** ()
- void **setSSBBfo** (int offset)
- void **setSSBConfig** (uint8_t AUDIOBW, uint8_t SBCUTFLT, uint8_t AVC_DIVIDER, uint8_t AVCEN, uint8_t SMUTESEL, uint8_t DSP_AFCDIS)
- void **setSSB** (uint16_t fromFreq, uint16_t toFreq, uint16_t intialFreq, uint16_t step, uint8_t usblsb)
- void **setSSB** (uint8_t usblsb)
- void **setSSBAudioBandwidth** (uint8_t AUDIOBW)
- void **setSSBAutomaticVolumeControl** (uint8_t AVCEN)

- void **setSBBSidebandCutoffFilter** (uint8_t SBCUTFLT)
- void **setSSBAvcDivider** (uint8_t AVC_DIVIDER)
- void **setSSBDspAfc** (uint8_t DSP_AFCDIS)
- void **setSSBSoftMute** (uint8_t SMUTESEL)
- **si47x_firmware_query_library queryLibraryId** ()
- void **patchPowerUp** ()
- bool **downloadPatch** (const uint8_t *ssb_patch_content, const uint16_t ssb_patch_content_size)
- bool **downloadPatch** (int eeprom_i2c_address)
- void **ssbPowerUp** ()
- void **setI2CLowSpeedMode** (void)
- void **setI2CStandardMode** (void)
  Sets I2C buss to 10KHz.
- void **setI2CFastMode** (void)
  Sets I2C buss to 100KHz.
- void **setI2CFastModeCustom** (long value=500000)
  Sets I2C buss to 400KHz.
- void **setDeviceI2CAddress** (uint8_t senPin)
- int16_t **getDeviceI2CAddress** (uint8_t **resetPin**)
- void **setDeviceOtherI2CAddress** (uint8_t i2cAddr)

Protected Member Functions

- void **waitInterrupr** (void)
- void **sendProperty** (uint16_t propertyValue, uint16_t param)
- void **sendSSBModeProperty** ()
- void **disableFmDebug** ()
- void **clearRdsBuffer2A** ()
- void **clearRdsBuffer2B** ()
- void **clearRdsBuffer0A** ()

Protected Attributes

- char **rds_buffer2A** [65]
- char **rds_buffer2B** [33]
  RDS Radio Text buffer - Program Information.
- char **rds_buffer0A** [9]
  RDS Radio Text buffer - Station Informaation.
- char **rds_time** [20]
  RDS Basic tuning and switching information (Type 0 groups)
- int **rdsTextAdress2A**
- int **rdsTextAdress2B**
- int **rdsTextAdress0A**
- int16_t **deviceAddress** = **SI473X_ADDR_SEN_LOW**
- uint8_t **lastTextFlagAB**
- uint8_t **resetPin**
- uint8_t **interruptPin**
- uint8_t **currentTune**
- uint16_t **currentMinimumFrequency**

- uint16_t **currentMaximumFrequency**
- uint16_t **currentWorkFrequency**
- uint16_t **currentStep**
- uint8_t **lastMode** = -1
- uint8_t **currentAvcAmMaxGain** = 48
  Store the last mode used.

- **si47x_frequency currentFrequency**
- **si47x_set_frequency currentFrequencyParams**
- **si47x_rqs_status currentRqsStatus**
- **si47x_response_status currentStatus**
- **si47x_firmware_information firmwareInfo**
- **si47x_rds_status currentRdsStatus**
- **si47x_agc_status currentAgcStatus**
- **si47x_ssb_mode currentSSBMode**
- **si473x_powerup powerUp**
- uint8_t **volume** = 32
- uint8_t **currentSsbStatus**

Detailed Description

Definition at line 776 of file SI4735.h.

Constructor & Destructor Documentation

SI4735::SI4735 ()

This is a library for the **SI4735**, BROADCAST AM/FM/SW RADIO RECEIVER, IC from Silicon Labs for the Arduino development environment. It works with I2C protocol. This library is intended to provide an easier interface for controlling the **SI4735**.

See also

documentation on `https://github.com/pu2clr/SI4735`.

also: Si47XX PROGRAMMING GUIDE; AN332 AN332 REV 0.8 UNIVERSAL PROGRAMMING GUIDE; AMENDMENT FOR SI4735-D60 SSB AND NBFM PATCHES

Pay attention: According to Si47XX PROGRAMMING GUIDE; AN332; page 207, "For write operations, the system controller next sends a data byte on SDIO, which is captured by the device on rising edges of SCLK. The device acknowledges each data byte by driving SDIO low for one cycle on the next falling edge of SCLK. The system controller may write up to 8 data bytes in a single 2-wire transaction. The first byte is a command, and the next seven bytes are arguments. Writing more than 8 bytes results in unpredictable device behavior". So, If you are extending this library, consider that restriction presented earlier.

ATTENTION: Some methods were implemented usin inline resource. Inline methods are implemented in **SI4735.h**

By Ricardo Lima Caratti, Nov 2019. Construct a new **SI4735::SI4735** object

Definition at line 30 of file SI4735.cpp.

Member Function Documentation

void SI4735::analogPowerUp (void)

Powerup in Analog Mode. It will be deprecated. Consider use radioPowerUp instead. Actually this function works fo Digital and Analog modes. You have to call setPowerUp method before.

Definition at line 225 of file SI4735.cpp.

void SI4735::clearRdsBuffer0A () `[protected]`

Clear RDS buffer 0A (text)

Definition at line 1232 of file SI4735.cpp.

void SI4735::clearRdsBuffer2A () `[protected]`

Clear RDS buffer 2A (text)

Definition at line 1213 of file SI4735.cpp.

void SI4735::clearRdsBuffer2B () `[protected]`

Clear RDS buffer 2B (text)

Definition at line 1223 of file SI4735.cpp.

void SI4735::digitalOutputFormat (uint8_t *OSIZE*, uint8_t *OMONO*, uint8_t *OMODE*, uint8_t *OFALL*)

Configures the digital audio output format. Options: DCLK edge, data format, force mono, and sample precision.

See also

Si47XX PROGRAMMING GUIDE; AN332; page 195.

Parameters

uint8_t	OSIZE Digital Output Audio Sample Precision (0=16 bits, 1=20 bits, 2=24 bits, 3=8bits).
uint8_t	OMONO Digital Output Mono Mode (0=Use mono/stereo blend).
uint8_t	OMODE Digital Output Mode (0=I2S, 6 = Left-justified, 8 = MSB at second DCLK after DFS pulse, 12 = MSB at first DCLK after DFS pulse).
uint8_t	OFALL Digital Output DCLK Edge (0 = use DCLK rising edge, 1 = use DCLK falling edge)

Definition at line 777 of file SI4735.cpp.

void SI4735::digitalOutputSampleRate (uint16_t *DOSR*)

Enables digital audio output and configures digital audio output sample rate in samples per second (sps).

See also

Si47XX PROGRAMMING GUIDE; AN332; page 196.

Parameters

uint16_t	DOSR Digital Output Sample Rate(32–48 ksps .0 to disable digital audio output).

Definition at line 794 of file SI4735.cpp.

void SI4735::disableFmDebug () `[protected]`

There is a debug feature that remains active in Si4704/05/3x-D60 firmware which can create periodic noise in audio. Silicon Labs recommends you disable this feature by sending the following bytes (shown here in hexadecimal form): 0x12 0x00 0xFF 0x00 0x00 0x00.

See also

Si47XX PROGRAMMING GUIDE; AN332; page 299.

Definition at line 749 of file SI4735.cpp.

bool SI4735::downloadPatch (const uint8_t * *ssb_patch_content*, const uint16_t *ssb_patch_content_size*)

Transfers the content of a patch stored in a array of bytes to the **SI4735** device. You must mount an array as shown below and know the size of that array as well.

It is importante to say that patches to the **SI4735** are distributed in binary form and have to be transferred to the internal RAM of the device by the host MCU (in this case Arduino). Since the RAM is volatile memory, the patch stored into the device gets lost when you turn off the system. Consequently, the content of the patch has to be transferred again to the device each time after turn on the system or reset the device.

The disadvantage of this approach is the amount of memory used by the patch content. This may limit the use of other radio functions you want implemented in Arduino.

See also

Si47XX PROGRAMMING GUIDE; AN332; pages 64 and 215-220.

Example of content: const PROGMEM uint8_t ssb_patch_content_full[] = { // SSB patch for whole SSBRX full download 0x15, 0x00, 0x0F, 0xE0, 0xF2, 0x73, 0x76, 0x2F, 0x16, 0x6F, 0x26, 0x1E, 0x00, 0x4B, 0x2C, 0x58, 0x16, 0xA3, 0x74, 0x0F, 0xE0, 0x4C, 0x36, 0xE4, 0x16, 0x3B, 0x1D, 0x4A, 0xEC, 0x36, 0x28, 0xB7, 0x16, 0x00, 0x3A, 0x47, 0x37, 0x00, 0x00, 0x00, 0x15, 0x00, 0x00, 0x00, 0x00, 0x00, 0x9D, 0x29};

const int size_content_full = sizeof ssb_patch_content_full;

Parameters

ssb_patch_content	point to array of bytes content patch.
ssb_patch_content_size	array size (number of bytes). The maximum size allowed for a patch is 15856 bytes

Returns

false if an error is found.

Definition at line 2168 of file SI4735.cpp.

bool SI4735::downloadPatch (int *eeprom_i2c_address*)

Under construction... Transfers the content of a patch stored in a eeprom to the **SI4735** device.

TO USE THIS METHOD YOU HAVE TO HAVE A EEPROM WRITEN WITH THE PATCH CONTENT

See also

the sketch write_ssb_patch_eeprom.ino (TO DO)

Parameters

eeprom_i2c_address	

Returns

false if an error is found.

Definition at line 2227 of file SI4735.cpp.

void SI4735::frequencyDown ()

Decrements the current frequency on current band/function by using the current step.

See also

setFrequencyStep

Definition at line 442 of file SI4735.cpp.

void SI4735::frequencyUp ()

Increments the current frequency on current band/function by using the current step.

See also

setFrequencyStep()

Definition at line 427 of file SI4735.cpp.

bool SI4735::getACFIndicator () `[inline]`

Gets the Error flag of status response.

Definition at line 866 of file SI4735.h.

uint8_t SI4735::getAgcGainIndex () `[inline]`

Definition at line 888 of file SI4735.h.

uint8_t SI4735::getAntennaTuningCapacitor () `[inline]`

Returns integer containing the multipath metric when tune is complete.

Definition at line 872 of file SI4735.h.

void SI4735::getAutomaticGainControl ()

Returns integer containing the current antenna tuning capacitor value.

Queries AGC STATUS

See also

Si47XX PROGRAMMING GUIDE; AN332; For FM page 80; for AM page 142.

AN332 REV 0.8 Universal Programming Guide Amendment for SI4735-D60 SSB and NBFM patches; page 18.

After call this method, you can call isAgcEnabled to know the AGC status and getAgcGainIndex to know the gain index value.

Definition at line 885 of file SI4735.cpp.

bool SI4735::getBandLimit () `[inline]`

Returns true if the AFC rails (AFC Rail Indicator).

Definition at line 867 of file SI4735.h.

bool SI4735::getCurrentAfcRailIndicator () `[inline]`

Valid Channel.

Definition at line 902 of file SI4735.h.

uint8_t SI4735::getCurrentAvcAmMaxGain () `[inline]`

Definition at line 878 of file SI4735.h.

bool SI4735::getCurrentBlendDetectInterrupt () `[inline]`

Multipath Detect High.

Definition at line 911 of file SI4735.h.

uint16_t SI4735::getCurrentFrequency ()

Gets the current frequency saved in memory. Unlike getFrequency, this method gets the current frequency recorded after the last setFrequency command. This method avoids bus traffic and CI processing. However, you can not get others status information like RSSI.

See also

getFrequency()

Definition at line 829 of file SI4735.cpp.

uint8_t SI4735::getCurrentMultipath () `[inline]`

Indicates stereo pilot presence.

Definition at line 907 of file SI4735.h.

bool SI4735::getCurrentMultipathDetectHigh () `[inline]`

Multipath Detect Low.

Definition at line 910 of file SI4735.h.

bool SI4735::getCurrentMultipathDetectLow () `[inline]`

Signed frequency offset (kHz).

Definition at line 909 of file SI4735.h.

bool SI4735::getCurrentPilot () `[inline]`

Indicates amount of stereo blend in % (100 = full stereo, 0 = full mono).

Definition at line 906 of file SI4735.h.

void SI4735::getCurrentReceivedSignalQuality (uint8_t *INTACK*)

Queries the status of the Received Signal Quality (RSQ) of the current channel. This method sould be called berore call **getCurrentRSSI()**, **getCurrentSNR()** etc. Command FM_RSQ_STATUS

See also

Si47XX PROGRAMMING GUIDE; AN332; pages 75 and 141

Parameters

INTACK	Interrupt Acknowledge. 0 = Interrupt status preserved; 1 = Clears RSQINT, BLENDINT, SNRHINT, SNRLINT, RSSIHINT, RSSILINT, MULTHINT, MULTLINT.

Definition at line 974 of file SI4735.cpp.

void SI4735::getCurrentReceivedSignalQuality (void)

Queries the status of the Received Signal Quality (RSQ) of the current channel Command FM_RSQ_STATUS

See also

Si47XX PROGRAMMING GUIDE; AN332; pages 75 and 141

Parameters

INTACK	Interrupt Acknowledge. 0 = Interrupt status preserved; 1 = Clears RSQINT, BLENDINT, SNRHINT, SNRLINT, RSSIHINT, RSSILINT, MULTHINT, MULTLINT.

Definition at line 1020 of file SI4735.cpp.

uint8_t SI4735::getCurrentRSSI () `[inline]`

Definition at line 895 of file SI4735.h.

bool SI4735::getCurrentRssiDetectHigh () `[inline]`

RSSI Detect Low.

Definition at line 898 of file SI4735.h.

bool SI4735::getCurrentRssiDetectLow () `[inline]`

current SNR metric (0–127 dB).

Definition at line 897 of file SI4735.h.

uint8_t SI4735::getCurrentSignedFrequencyOffset () `[inline]`

Contains the current multipath metric. (0 = no multipath; 100 = full multipath)

Definition at line 908 of file SI4735.h.

uint8_t SI4735::getCurrentSNR () `[inline]`

current receive signal strength (0â€“127 dBÎ¼V).

Definition at line 896 of file SI4735.h.

bool SI4735::getCurrentSnrDetectHigh () `[inline]`

SNR Detect Low.

Definition at line 900 of file SI4735.h.

bool SI4735::getCurrentSnrDetectLow () `[inline]`

RSSI Detect High.

Definition at line 899 of file SI4735.h.

bool SI4735::getCurrentSoftMuteIndicator () `[inline]`

AFC Rail Indicator.

Definition at line 903 of file SI4735.h.

uint8_t SI4735::getCurrentStereoBlend () `[inline]`

Soft Mute Indicator. Indicates soft mute is engaged.

Definition at line 905 of file SI4735.h.

bool SI4735::getCurrentValidChannel () `[inline]`

SNR Detect High.

Definition at line 901 of file SI4735.h.

uint8_t SI4735::getCurrentVolume () `[inline]`

Definition at line 933 of file SI4735.h.

int16_t SI4735::getDeviceI2CAddress (uint8_t *resetPin*)

Scans for two possible addresses for the Si47XX (0x11 or 0x63) This function also sets the system to the found I2C bus address of Si47XX.

You do not need to use this function if the SEN PIN is configured to ground (GND). The default I2C address is 0x11. Use this function if you do not know how the SEN pin is configured.

Parameters

uint8_t	resetPin MCU Mater (Arduino) reset pin

Returns

int16_t 0x11 if the SEN pin of the Si47XX is low or 0x63 if the SEN pin of the Si47XX is HIGH or 0x0 if error.

Definition at line 63 of file SI4735.cpp.

void SI4735::getFirmware (void)

Gets firmware information

See also

Si47XX PROGRAMMING GUIDE; AN332; pages 66, 131

Definition at line 250 of file SI4735.cpp.

uint8_t SI4735::getFirmwareCHIPREV () `[inline]`

RESP7 - Returns the Component Minor Revision (ASCII).

Definition at line 926 of file SI4735.h.

uint8_t SI4735::getFirmwareCMPMAJOR () `[inline]`

RESP5 - Returns the Patch ID Low byte (HEX).

Definition at line 924 of file SI4735.h.

uint8_t SI4735::getFirmwareCMPMINOR () `[inline]`

RESP6 - Returns the Component Major Revision (ASCII).

Definition at line 925 of file SI4735.h.

uint8_t SI4735::getFirmwareFWMAJOR () `[inline]`

RESP1 - Part Number (HEX)

Definition at line 920 of file SI4735.h.

uint8_t SI4735::getFirmwareFWMINOR () `[inline]`

RESP2 - Returns the Firmware Major Revision (ASCII).

Definition at line 921 of file SI4735.h.

uint8_t SI4735::getFirmwarePATCHH () `[inline]`

RESP3 - Returns the Firmware Minor Revision (ASCII).

Definition at line 922 of file SI4735.h.

uint8_t SI4735::getFirmwarePATCHL () `[inline]`

RESP4 - Returns the Patch ID High byte (HEX).

Definition at line 923 of file SI4735.h.

uint8_t SI4735::getFirmwarePN () `[inline]`

Blend Detect Interrupt.

Definition at line 919 of file SI4735.h.

uint16_t SI4735::getFrequency (void)

Gets the current frequency of the Si4735 (AM or FM) The method status do it an more. See getStatus below.

See also

Si47XX PROGRAMMING GUIDE; AN332; pages 73 (FM) and 139 (AM)

Definition at line 809 of file SI4735.cpp.

bool SI4735::getGroupLost ()[inline]

1 = RDS currently synchronized.

Definition at line 992 of file SI4735.h.

void SI4735::getNext2Block (char * *c*)

Process data received from group 2B

Parameters

c	char array reference to the "group 2B" text

Definition at line 1506 of file SI4735.cpp.

void SI4735::getNext4Block (char * *c*)

Process data received from group 2A

Parameters

c	char array reference to the "group 2A" text

Definition at line 1538 of file SI4735.cpp.

uint8_t SI4735::getNumRdsFifoUsed ()[inline]

1 = One or more RDS groups discarded due to FIFO overrun.

Definition at line 993 of file SI4735.h.

bool SI4735::getRadioDataSystemInterrupt ()[inline]

Gets Received Signal Quality Interrupt(RSQINT)

Definition at line 862 of file SI4735.h.

uint8_t SI4735::getRdsFlagAB (void)

Returns the current Text Flag A/B

Returns

uint8_t

Definition at line 1440 of file SI4735.cpp.

uint8_t SI4735::getRdsGroupType (void)

Returns the Group Type (extracted from the Block B)

Definition at line 1424 of file SI4735.cpp.

bool SI4735::getRdsNewBlockA ()[inline]

1 = Found RDS synchronization

Definition at line 989 of file SI4735.h.

bool SI4735::getRdsNewBlockB ()[inline]

1 = Valid Block A data has been received.

Definition at line 990 of file SI4735.h.

uint16_t SI4735::getRdsPI (void)

Returns the programa type. Read the Block A content

See also

Si47XX PROGRAMMING GUIDE; AN332; pages 77 and 78

Returns

BLOCKAL

Definition at line 1412 of file SI4735.cpp.

uint8_t SI4735::getRdsProgramType (void)

Returns the Program Type (extracted from the Block B)

See also

`https://en.wikipedia.org/wiki/Radio_Data_System`

Returns

program type (an integer betwenn 0 and 31)

Definition at line 1491 of file SI4735.cpp.

bool SI4735::getRdsReceived ()`[inline]`

Definition at line 986 of file SI4735.h.

void SI4735::getRdsStatus ()

Gets RDS Status. Same result of calling getRdsStatus(0,0,0);

See also

SI4735::getRdsStatus(uint8_t INTACK, uint8_t MTFIFO, uint8_t STATUSONLY)

Please, call **getRdsStatus(uint8_t INTACK, uint8_t MTFIFO, uint8_t STATUSONLY)** instead **getRdsStatus()** if you want other behaviour

Definition at line 1397 of file SI4735.cpp.

void SI4735::getRdsStatus (uint8_t *INTACK*, uint8_t *MTFIFO*, uint8_t *STATUSONLY*)

Gets the RDS status. Store the status in currentRdsStatus member. RDS COMMAND FM_RDS_STATUS

See also

Si47XX PROGRAMMING GUIDE; AN332; pages 55 and 77

Parameters

INTACK	Interrupt Acknowledge; 0 = RDSINT status preserved. 1 = Clears RDSINT.
MTFIFO	0 = If FIFO not empty, read and remove oldest FIFO entry; 1 = Clear RDS Receive FIFO.
STATUSONLY	Determines if data should be removed from the RDS FIFO.

Definition at line 1350 of file SI4735.cpp.

bool SI4735::getRdsSync ()`[inline]`

1 = Valid Block B data has been received.

Definition at line 991 of file SI4735.h.

bool SI4735::getRdsSyncFound ()[inline]

1 = Lost RDS synchronization

Definition at line 988 of file SI4735.h.

bool SI4735::getRdsSyncLost ()[inline]

1 = FIFO filled to minimum number of groups

Definition at line 987 of file SI4735.h.

char * SI4735::getRdsText (void)

Gets the RDS Text when the message is of the Group Type 2 version A

Returns

char* The string (char array) with the content (Text) received from group 2A

Definition at line 1572 of file SI4735.cpp.

char * SI4735::getRdsText0A (void)

Gets the station name and other messages.

Returns

char* should return a string with the station name. However, some stations send other kind of messages

Definition at line 1594 of file SI4735.cpp.

char * SI4735::getRdsText2A (void)

Gets the Text processed for the 2A group

Returns

char* string with the Text of the group A2

Definition at line 1625 of file SI4735.cpp.

char * SI4735::getRdsText2B (void)

Gets the Text processed for the 2B group

Returns

char* string with the Text of the group AB

Definition at line 1657 of file SI4735.cpp.

uint8_t SI4735::getRdsTextSegmentAddress (void)

Returns the address of the text segment. 2A - Each text segment in version 2A groups consists of four characters. A messages of this group can be have up to 64 characters. 2B - In version 2B groups, each text segment consists of only two characters. When the current RDS status is using this version, the maximum message length will be 32 characters.

Returns

uint8_t the address of the text segment.

Definition at line 1460 of file SI4735.cpp.

char * SI4735::getRdsTime (void)

Gets the RDS time and date when the Group type is 4

Returns

char* a string with hh:mm +/- offset

Definition at line 1688 of file SI4735.cpp.

uint8_t SI4735::getRdsVersionCode (void)

Gets the version code (extracted from the Block B)

Returns

0=A or 1=B

Definition at line 1474 of file SI4735.cpp.

uint8_t SI4735::getReceivedSignalStrengthIndicator ()`[inline]`

Returns true if the channel is currently valid as determined by the seek/tune properties (0x1403, 0x1404, 0x1108)

Definition at line 869 of file SI4735.h.

bool SI4735::getSignalQualityInterrupt ()`[inline]`

STATUS RESPONSE Set of methods to get current status information. Call them after getStatus or getFrequency or seekStation See Si47XX PROGRAMMING GUIDE; AN332; pages 63

Definition at line 861 of file SI4735.h.

void SI4735::getStatus ()

Gets the current status of the Si4735 (AM or FM)

See also

Si47XX PROGRAMMING GUIDE; AN332; pages 73 (FM) and 139 (AM)

Definition at line 872 of file SI4735.cpp.

void SI4735::getStatus (uint8_t *INTACK*, uint8_t *CANCEL*)

Gets the current status of the Si4735 (AM or FM)

See also

Si47XX PROGRAMMING GUIDE; AN332; pages 73 (FM) and 139 (AM)

Parameters

uint8_t	INTACK Seek/Tune Interrupt Clear. If set, clears the seek/tune complete interrupt status indicator;
uint8_t	CANCEL Cancel seek. If set, aborts a seek currently in progress;

Definition at line 841 of file SI4735.cpp.

bool SI4735::getStatusCTS ()`[inline]`

Return the Error flag (true or false) of status of the least Tune or Seek.

Definition at line 865 of file SI4735.h.

bool SI4735::getStatusError () `[inline]`

Seek/Tune Complete Interrupt; 1 = Tune complete has been triggered.

Definition at line 864 of file SI4735.h.

uint8_t SI4735::getStatusMULT () `[inline]`

Returns integer containing the SNR metric when tune is complete (dB).

Definition at line 871 of file SI4735.h.

uint8_t SI4735::getStatusSNR () `[inline]`

Returns integer Received Signal Strength Indicator (dBÎ¼V).

Definition at line 870 of file SI4735.h.

bool SI4735::getStatusValid () `[inline]`

Returns true if a seek hit the band limit (WRAP = 0 in FM_START_SEEK) or wrapped to the original frequency(WRAP = 1).

Definition at line 868 of file SI4735.h.

bool SI4735::getTuneCompleteTriggered () `[inline]`

Gets Radio Data System (RDS) Interrupt.

Definition at line 863 of file SI4735.h.

uint8_t SI4735::getTuneFrequencyFast () `[inline]`

Definition at line 950 of file SI4735.h.

uint8_t SI4735::getTuneFrequencyFreeze () `[inline]`

FAST Tuning. If set, executes fast and invalidated tune. The tune status will not be accurate.

Definition at line 952 of file SI4735.h.

uint8_t SI4735::getVolume ()

Gets the current volume level.

See also

setVolume()

Returns

volume (domain: 0 - 63)

Definition at line 1165 of file SI4735.cpp.

bool SI4735::isAgcEnabled () `[inline]`

Definition at line 887 of file SI4735.h.

bool SI4735::isCurrentTuneFM ()

Returns true if the current function is FM (FM_TUNE_FREQ).

Returns

true if the current function is FM (FM_TUNE_FREQ).

Definition at line 592 of file SI4735.cpp.

void SI4735::patchPowerUp ()

This method can be used to prepare the device to apply SSBRX patch Call queryLibraryId before call this method. Powerup the device by issuing the POWER_UP command with FUNC = 1 (AM/SW/LW Receive)

See also

Si47XX PROGRAMMING GUIDE; AN332; pages 64 and 215-220 and

AN332 REV 0.8 UNIVERSAL PROGRAMMING GUIDE AMENDMENT FOR SI4735-D60 SSB AND NBFM PATCHES; page 7.

Definition at line 2089 of file SI4735.cpp.

void SI4735::powerDown (void)

Moves the device from powerup to powerdown mode. After Power Down command, only the Power Up command is accepted.

See also

Si47XX PROGRAMMING GUIDE; AN332; pages 67, 132

Definition at line 236 of file SI4735.cpp.

si47x_firmware_query_library SI4735::queryLibraryId ()

Call it first if you are applying a patch on **SI4735**. Used to confirm if the patch is compatible with the internal device library revision. See Si47XX PROGRAMMING GUIDE; AN332; pages 64 and 215-220.

Returns

a struct **si47x_firmware_query_library** (see it in **SI4735.h**) Query the library information

You have to call this function if you are applying a patch on SI47XX (SI4735-D60)

The first command that is sent to the device is the POWER_UP command to confirm that the patch is compatible with the internal device library revision. The device moves into the powerup mode, returns the reply, and moves into the powerdown mode. The POWER_UP command is sent to the device again to configure the mode of the device and additionally is used to start the patching process. When applying the patch, the PATCH bit in ARG1 of the POWER_UP command must be set to 1 to begin the patching process. [AN332 page 219].

See also

Si47XX PROGRAMMING GUIDE; AN332; pages 214, 215, 216, 219

si47x_firmware_query_library in **SI4735.h**

Returns

si47x_firmware_query_library Library Identification

Definition at line 2053 of file SI4735.cpp.

void SI4735::radioPowerUp (void)

Powerup the Si47XX Before call this function call the setPowerUp to set up the parameters. Parameters you have to set up with setPowerUp

CTSIEN Interrupt anabled or disabled; GPO2OEN GPO2 Output Enable or disabled; PATCH Boot normally or patch; XOSCEN Use external crystal oscillator; FUNC defaultFunction = 0 = FM Receive; 1 = AM (LW/MW/SW) Receiver. OPMODE SI473X_ANALOG_AUDIO (B00000101) or SI473X_DIGITAL_AUDIO (B00001011)

See also

SI4735::setPowerUp()

Si47XX PROGRAMMING GUIDE; AN332; pages 64, 129

Definition at line 206 of file SI4735.cpp.

void SI4735::RdsInit ()

Starts the control variables for RDS.

Definition at line 1201 of file SI4735.cpp.

void SI4735::reset (void)

Reset the SI473X

See also

Si47XX PROGRAMMING GUIDE; AN332;

Definition at line 126 of file SI4735.cpp.

void SI4735::seekStation (uint8_t *SEEKUP*, uint8_t *WRAP*)

Look for a station

See also

Si47XX PROGRAMMING GUIDE; AN332; pages 55, 72, 125 and 137

Parameters

SEEKUP	Seek Up/Down. Determines the direction of the search, either UP = 1, or DOWN = 0.
Wrap/Halt.	Determines whether the seek should Wrap = 1, or Halt = 0 when it hits the band limit.

Definition at line 1033 of file SI4735.cpp.

void SI4735::seekStationDown ()

Search the previous station

See also

seekStation(uint8_t SEEKUP, uint8_t WRAP)

Definition at line 1078 of file SI4735.cpp.

void SI4735::seekStationUp ()

Search for the next station

See also

seekStation(uint8_t SEEKUP, uint8_t WRAP)

Definition at line 1066 of file SI4735.cpp.

void SI4735::sendProperty (uint16_t *propertyValue*, uint16_t *parameter*) `[protected]`

Sends (sets) property to the SI47XX This method is used for others to send generic properties and params to SI47XX

See also

Si47XX PROGRAMMING GUIDE; AN332; pages 68, 124 and 133.

Definition at line 603 of file SI4735.cpp.

void SI4735::sendSSBModeProperty () `[protected]`

Just send the property SSB_MOD to the device. Internal use (privete method).

Definition at line 2006 of file SI4735.cpp.

void SI4735::setAM ()

Sets the radio to AM function. It means: LW MW and SW.

See also

Si47XX PROGRAMMING GUIDE; AN332; page 129.

Definition at line 458 of file SI4735.cpp.

void SI4735::setAM (uint16_t *fromFreq*, uint16_t *toFreq*, uint16_t *initialFreq*, uint16_t *step*)

Sets the radio to AM (LW/MW/SW) function.

See also

setAM()

Parameters

fromFreq	minimum frequency for the band
toFreq	maximum frequency for the band
initialFreq	initial frequency
step	step used to go to the next channel

Definition at line 499 of file SI4735.cpp.

void SI4735::setAmSoftMuteMaxAttenuation () `[inline]`

Definition at line 881 of file SI4735.h.

void SI4735::setAmSoftMuteMaxAttenuation (uint8_t *smattn*) `[inline]`

Definition at line 880 of file SI4735.h.

void SI4735::setAudioMute (bool *off*)

Returns the current volume level.

Sets the audio on or off

See also

See Si47XX PROGRAMMING GUIDE; AN332; pages 62, 123, 171

Parameters

value	if true, mute the audio; if false unmute the audio.

Definition at line 1153 of file SI4735.cpp.

void SI4735::setAutomaticGainControl (uint8_t *AGCDIS*, uint8_t *AGCIDX*)

If FM, overrides AGC setting by disabling the AGC and forcing the LNA to have a certain gain that ranges between 0 (minimum attenuation) and 26 (maximum attenuation); If AM/SSB, Overrides the AM AGC setting by disabling the AGC and forcing the gain index that ranges between 0 (minimum attenuation) and 37+ATTN_BACKUP (maximum attenuation);

See also

Si47XX PROGRAMMING GUIDE; AN332; For FM page 81; for AM page 143

Parameters

uint8_t	AGCDIS This param selects whether the AGC is enabled or disabled (0 = AGC enabled; 1 = AGC disabled);
uint8_t	AGCIDX AGC Index (0 = Minimum attenuation (max gain); 1 – 36 = Intermediate attenuation); if >greater than 36 - Maximum attenuation (min gain)).

Definition at line 926 of file SI4735.cpp.

void SI4735::setAvcAmMaxGain () `[inline]`

Definition at line 877 of file SI4735.h.

void SI4735::setAvcAmMaxGain (uint8_t *gain*)

Sets the maximum gain for automatic volume control. If no parameter is sent, it will be consider 48dB.

See also

Si47XX PROGRAMMING GUIDE; AN332; page 152

Parameters

uint8_t	gain Select a value between 12 and 192. Defaul value 48dB.

Definition at line 956 of file SI4735.cpp.

void SI4735::setBandwidth (uint8_t *AMCHFLT*, uint8_t *AMPLFLT*)

Selects the bandwidth of the channel filter for AM reception. The choices are 6, 4, 3, 2, 2.5, 1.8, or 1 (kHz). The default bandwidth is 2 kHz. Works only in AM / SSB (LW/MW/SW)

See also

Si47XX PROGRAMMING GUIDE; AN332; pages 125, 151, 277, 181.

Parameters

AMCHFLT	the choices are: 0 = 6 kHz Bandwidth 1 = 4 kHz Bandwidth 2 = 3 kHz Bandwidth 3 = 2 kHz Bandwidth 4 = 1 kHz Bandwidth 5 = 1.8 kHz Bandwidth 6 = 2.5 kHz Bandwidth, gradual roll off 7–15 = Reserved (Do not use).
AMPLFLT	Enables the AM Power Line Noise Rejection Filter.

Definition at line 557 of file SI4735.cpp.

void SI4735::setDeviceI2CAddress (uint8_t *senPin*)

Sets the I2C Bus Address

ATTENTION: The parameter senPin is not the I2C bus address. It is the SEN pin setup of the schematic (eletronic circuit). If it is connected to the ground, call this function with senPin = 0; else senPin = 1. You do not need to use this function if the SEN PIN configured to ground (GND).

The default value is 0x11 (senPin = 0). In this case you have to ground the pin SEN of the SI473X. If you want to change this address, call this function with senPin = 1

Parameters

senPin	0 - when the pin SEN (16 on SSOP version or pin 6 on QFN version) is set to low (GND - 0V) 1 - when the pin SEN (16 on SSOP version or pin 6 on QFN version) is set to high (+3.3V)

Definition at line 108 of file SI4735.cpp.

void SI4735::setDeviceOtherI2CAddress (uint8_t *i2cAddr*)

Sets the onther I2C Bus Address (for Si470X) You can set another I2C address different of 0x11 and 0x63

Parameters

uint8_t	i2cAddr (example 0x10)

Definition at line 117 of file SI4735.cpp.

void SI4735::setFM ()

Sets the radio to FM function

See also

Si47XX PROGRAMMING GUIDE; AN332; page 64.

Definition at line 478 of file SI4735.cpp.

void SI4735::setFM (uint16_t *fromFreq*, uint16_t *toFreq*, uint16_t *initialFreq*, uint16_t *step*)

Sets the radio to FM function.

See also

setFM()

Parameters

fromFreq	minimum frequency for the band
toFreq	maximum frequency for the band
initialFreq	initial frequency (default frequency)
step	step used to go to the next channel

Definition at line 524 of file SI4735.cpp.

void SI4735::setFmBlendMonoThreshold (uint8_t *parameter*)

Sets RSSI threshold for mono blend (Full mono below threshold, blend above threshold). To force stereo set this to 0. To force mono set this to 127. Default value is 30 dBÎ¼V.

See also

Si47XX PROGRAMMING GUIDE; AN332; page 56.

Parameters

parameter	valid values: 0 to 127

Definition at line 644 of file SI4735.cpp.

void SI4735::setFmBlendMultiPathMonoThreshold (uint8_t *parameter*)

Sets Multipath threshold for mono blend (Full mono above threshold, blend below threshold). To force stereo, set to 100. To force mono, set to 0. The default is 60.

See also

Si47XX PROGRAMMING GUIDE; AN332; page 60.

Parameters

parameter	valid values: 0 to 100

Definition at line 721 of file SI4735.cpp.

void SI4735::setFmBlendMultiPathStereoThreshold (uint8_t *parameter*)

Sets multipath threshold for stereo blend (Full stereo below threshold, blend above threshold). To force stereo, set this to 100. To force mono, set this to 0. Default value is 20.

See also

Si47XX PROGRAMMING GUIDE; AN332; page 60.

Parameters

parameter	valid values: 0 to 100

Definition at line 708 of file SI4735.cpp.

void SI4735::setFmBLendRssiMonoThreshold (uint8_t *parameter*)

Sets RSSI threshold for mono blend (Full mono below threshold, blend above threshold). To force stereo, set this to 0. To force mono, set this to 127. Default value is 30 dBÎ¼V.

See also

Si47XX PROGRAMMING GUIDE; AN332; page 59.

Parameters

parameter	valid values: 0 to 127

Definition at line 669 of file SI4735.cpp.

void SI4735::setFmBlendRssiStereoThreshold (uint8_t *parameter*)

Sets RSSI threshold for stereo blend. (Full stereo above threshold, blend below threshold.) To force stereo, set this to 0. To force mono, set this to 127. Default value is 49 dBÎ¼V.

See also

Si47XX PROGRAMMING GUIDE; AN332; page 59.

Parameters

parameter	valid values: 0 to 127

Definition at line 656 of file SI4735.cpp.

void SI4735::setFmBLendSnrMonoThreshold (uint8_t *parameter*)

Sets SNR threshold for mono blend (Full mono below threshold, blend above threshold). To force stereo, set this to 0. To force mono, set this to 127. Default value is 14 dB.

See also

Si47XX PROGRAMMING GUIDE; AN332; page 59.

Parameters

parameter	valid values: 0 to 127

Definition at line 695 of file SI4735.cpp.

void SI4735::setFmBlendSnrStereoThreshold (uint8_t *parameter*)

Sets SNR threshold for stereo blend (Full stereo above threshold, blend below threshold). To force stereo, set this to 0. To force mono, set this to 127. Default value is 27 dB.

See also

Si47XX PROGRAMMING GUIDE; AN332; page 59.

Parameters

parameter	valid values: 0 to 127

Definition at line 682 of file SI4735.cpp.

void SI4735::setFmBlendStereoThreshold (uint8_t *parameter*)

Sets RSSI threshold for stereo blend (Full stereo above threshold, blend below threshold). To force stereo, set this to 0. To force mono, set this to 127.

See also

Si47XX PROGRAMMING GUIDE; AN332; page 90.

Parameters

parameter	valid values: 0 to 127

Definition at line 631 of file SI4735.cpp.

void SI4735::setFmStereoOff ()

Turn Off Stereo operation.

Definition at line 729 of file SI4735.cpp.

void SI4735::setFmStereoOn ()

Turn Off Stereo operation.

Definition at line 737 of file SI4735.cpp.

void SI4735::setFrequency (uint16_t *freq*)

Set the frequency to the corrent function of the Si4735 (FM, AM or SSB) You have to call setup or setPowerUp before call setFrequency.

See also

Si47XX PROGRAMMING GUIDE; AN332; pages 70, 135

AN332 REV 0.8 UNIVERSAL PROGRAMMING GUIDE; page 13

Parameters

uint16_t	freq Is the frequency to change. For example, FM => 10390 = 103.9 MHz; AM => 810 = 810 KHz.

Definition at line 376 of file SI4735.cpp.

void SI4735::setFrequencyStep (uint16_t *step*)

Sets the current step value.

ATTENTION: This function does not check the limits of the current band. Please, don't take a step bigger than your legs.

Parameters

step	if you are using FM, 10 means 100KHz. If you are using AM 10 means 10KHz For AM, 1 (1KHz) to 1000 (1MHz) are valid values. For FM 5 (50KHz) and 10 (100KHz) are valid values.

Definition at line 417 of file SI4735.cpp.

void SI4735::setFunction (uint8_t *FUNC*)

void SI4735::setI2CFastMode (void)`[inline]`

Sets I2C buss to 100KHz.

Definition at line 1046 of file SI4735.h.

void SI4735::setI2CFastModeCustom (long *value* = `500000`)`[inline]`

Sets I2C buss to 400KHz.

Sets the I2C bus to a given value.

ATTENTION: use this function with cation

Parameters

value	in Hz. For example: The values 500000 sets the bus to 500KHz.

Definition at line 1055 of file SI4735.h.

void SI4735::setI2CLowSpeedMode (void)`[inline]`

Definition at line 1044 of file SI4735.h.

void SI4735::setI2CStandardMode (void)`[inline]`

Sets I2C buss to 10KHz.

Definition at line 1045 of file SI4735.h.

void SI4735::setPowerUp (uint8_t *CTSIEN*, uint8_t *GPO2OEN*, uint8_t *PATCH*, uint8_t *XOSCEN*, uint8_t *FUNC*, uint8_t *OPMODE*)

Set the Power Up parameters for si473X. Use this method to chenge the defaul behavior of the Si473X. Use it before PowerUp()

See also

Si47XX PROGRAMMING GUIDE; AN332; pages 65 and 129

Parameters

uint8_t	CTSIEN sets Interrupt anabled or disabled (1 = anabled and 0 = disabled)
uint8_t	GPO2OEN sets GP02 Si473X pin enabled (1 = anabled and 0 = disabled)
uint8_t	PATCH Used for firmware patch updates. Use it always 0 here.
uint8_t	XOSCEN sets external Crystal enabled or disabled
uint8_t	FUNC sets the receiver function have to be used [0 = FM Receive; 1 = AM (LW/MW/SW) and SSB (if SSB patch apllied)]
uint8_t	OPMODE set the kind of audio mode you want to use.

Definition at line 163 of file SI4735.cpp.

void SI4735::setRdsConfig (uint8_t *RDSEN*, uint8_t *BLETHA*, uint8_t *BLETHB*, uint8_t *BLETHC*, uint8_t *BLETHD*)

RESP3 - RDS FIFO Used; Number of groups remaining in the RDS FIFO (0 if empty).

Sets RDS property (FM_RDS_CONFIG) Configures RDS settings to enable RDS processing (RDSEN) and set RDS block error thresholds. When a RDS Group is received, all block errors must be less than or equal the associated block error threshold for the group to be stored in the RDS FIFO.

See also

Si47XX PROGRAMMING GUIDE; AN332; page 104

MPORTANT: All block errors must be less than or equal the associated block error threshold for the group to be stored in the RDS FIFO. 0 = No errors. 1 = 1–2 bit errors detected and corrected. 2 = 3–5 bit errors detected and corrected. 3 = Uncorrectable. Recommended Block Error Threshold options: 2,2,2,2 = No group stored if any errors are uncorrected. 3,3,3,3 = Group stored regardless of errors. 0,0,0,0 = No group stored containing corrected or uncorrected errors. 3,2,3,3 = Group stored with corrected errors on B, regardless of errors on A, C, or D.

Parameters

uint8_t	RDSEN RDS Processing Enable; 1 = RDS processing enabled.
uint8_t	BLETHA Block Error Threshold BLOCKA.
uint8_t	BLETHB Block Error Threshold BLOCKB.
uint8_t	BLETHC Block Error Threshold BLOCKC.
uint8_t	BLETHD Block Error Threshold BLOCKD.

Definition at line 1265 of file SI4735.cpp.

void SI4735::setRdsIntSource (uint8_t *RDSNEWBLOCKB*, uint8_t *RDSNEWBLOCKA*, uint8_t *RDSSYNCFOUND*, uint8_t *RDSSYNCLOST*, uint8_t *RDSRECV*)

Configures interrupt related to RDS

Use this method if want to use interrupt

See also

Si47XX PROGRAMMING GUIDE; AN332; page 103

Parameters

RDSRECV	If set, generate RDSINT when RDS FIFO has at least FM_RDS_INT_FIFO_COUNT entries.
RDSSYNCLOST	If set, generate RDSINT when RDS loses synchronization.
RDSSYNCFOUND	set, generate RDSINT when RDS gains synchronization.
RDSNEWBLOCKA	If set, generate an interrupt when Block A data is found or subsequently changed
RDSNEWBLOCKB	If set, generate an interrupt when Block B data is found or subsequently changed

Definition at line 1309 of file SI4735.cpp.

void SI4735::setSBBSidebandCutoffFilter (uint8_t *SBCUTFLT*)

Sets SBB Sideband Cutoff Filter for band pass and low pass filters: 0 = Band pass filter to cutoff both the unwanted side band and high frequency components > 2.0 kHz of the wanted side band. (default) 1 = Low pass filter to cutoff the unwanted side band. Other values = not allowed.

See also

AN332 REV 0.8 UNIVERSAL PROGRAMMING GUIDE; page 24

Parameters

SBCUTFLT	0 or 1; see above

Definition at line 1914 of file SI4735.cpp.

void SI4735::setSeekAmLimits (uint16_t *bottom*, uint16_t *top*)

Sets the bottom frequency and top frequency of the AM band for seek. Default is 520 to 1710.

See also

Si47XX PROGRAMMING GUIDE; AN332; pages 127, 161, and 162

Parameters

uint16_t	bottom - the bottom of the AM band for seek
uint16_t	top - the top of the AM band for seek

Definition at line 1093 of file SI4735.cpp.

void SI4735::setSeekAmSpacing (uint16_t *spacing*)

Selects frequency spacingfor AM seek. Default is 10 kHz spacing.

See also

Si47XX PROGRAMMING GUIDE; AN332; pages 163, 229 and 283

Parameters

uint16_t	spacing - step in KHz

Definition at line 1106 of file SI4735.cpp.

void SI4735::setSeekRssiThreshold (uint16_t *value*)

Sets the RSSI threshold for a valid AM Seek/Tune. If the value is zero then RSSI threshold is not considered when doing a seek. Default value is 25 dBÎ¼V.

See also

Si47XX PROGRAMMING GUIDE; AN332; page 127

Definition at line 1128 of file SI4735.cpp.

void SI4735::setSeekSrnThreshold (uint16_t *value*)

Sets the SNR threshold for a valid AM Seek/Tune. If the value is zero then SNR threshold is not considered when doing a seek. Default value is 5 dB.

See also

Si47XX PROGRAMMING GUIDE; AN332; page 127

Definition at line 1117 of file SI4735.cpp.

void SI4735::setSSB (uint16_t *fromFreq*, uint16_t *toFreq*, uint16_t *intialFreq*, uint16_t *step*, uint8_t *usblsb*)

Definition at line 1986 of file SI4735.cpp.

void SI4735::setSSB (uint8_t *usblsb*)

Set the radio to AM function. It means: LW MW and SW.

See also

AN332 REV 0.8 UNIVERSAL PROGRAMMING GUIDE; pages 13 and 14

setAM()

void **SI4735::setFrequency(uint16_t freq)**

Parameters

usblsb	upper or lower side band; 1 = LSB; 2 = USB

Definition at line 1960 of file SI4735.cpp.

void SI4735::setSSBAudioBandwidth (uint8_t *AUDIOBW*)

SSB Audio Bandwidth for SSB mode

0 = 1.2 kHz low-pass filter* . (default) 1 = 2.2 kHz low-pass filter* . 2 = 3.0 kHz low-pass filter. 3 = 4.0 kHz low-pass filter. 4 = 500 Hz band-pass filter for receiving CW signal, i.e. [250 Hz, 750 Hz] with center frequency at 500 Hz when USB is selected or [-250 Hz, -750 1Hz] with center frequency at -500Hz when LSB is selected* . 5 = 1 kHz band-pass filter for receiving CW signal, i.e. [500 Hz, 1500 Hz] with center frequency at 1 kHz when USB is selected or [-500 Hz, -1500 1 Hz] with center frequency at -1kHz when LSB is selected* . Other values = reserved. Note: If audio bandwidth selected is about 2 kHz or below, it is recommended to set SBCUTFLT[3:0] to 0 to enable the band pass filter for better high- cut performance on the wanted side band. Otherwise, set it to 1.

See also

AN332 REV 0.8 UNIVERSAL PROGRAMMING GUIDE; page 24

Parameters

AUDIOBW	the valid values are 0, 1, 2, 3, 4 or 5; see description above

Definition at line 1943 of file SI4735.cpp.

void SI4735::setSSBAutomaticVolumeControl (uint8_t *AVCEN*)

Sets SSB Automatic Volume Control (AVC) for SSB mode

See also

AN332 REV 0.8 UNIVERSAL PROGRAMMING GUIDE; page 24

Parameters

AVCEN	0 = Disable AVC; 1 = Enable AVC (default).

Definition at line 1885 of file SI4735.cpp.

void SI4735::setSSBAvcDivider (uint8_t *AVC_DIVIDER*)

Sets AVC Divider

See also

AN332 REV 0.8 UNIVERSAL PROGRAMMING GUIDE; page 24

Parameters

AVC_DIVIDER	SSB mode, set divider = 0; SYNC mode, set divider = 3; Other values = not allowed.

Definition at line 1898 of file SI4735.cpp.

void SI4735::setSSBBfo (int *offset*)

Sets the SSB Beat Frequency Offset (BFO).

See also

AN332 REV 0.8 UNIVERSAL PROGRAMMING GUIDE; pages 5 and 23

Parameters

offset	16-bit signed value (unit in Hz). The valid range is -16383 to +16383 Hz.

Definition at line 1790 of file SI4735.cpp.

void SI4735::setSSBConfig (uint8_t *AUDIOBW*, uint8_t *SBCUTFLT*, uint8_t *AVC_DIVIDER*, uint8_t *AVCEN*, uint8_t *SMUTESEL*, uint8_t *DSP_AFCDIS*)

Set the SSB receiver mode details: 1) Enable or disable AFC track to carrier function for receiving normal AM signals; 2) Set the audio bandwidth; 3) Set the side band cutoff filter; 4) Set soft-mute based on RSSI or SNR; 5) Enable or disbable automatic volume control (AVC) function.

See also

AN332 REV 0.8 UNIVERSAL PROGRAMMING GUIDE; page 24

Parameters

AUDIOBW	SSB Audio bandwidth; 0 = 1.2KHz (default); 1=2.2KHz; 2=3KHz; 3=4KHz; 4=500Hz; 5=1KHz.
SBCUTFLT	SSB side band cutoff filter for band passand low pass filter if 0, the band pass filter to cutoff both the unwanted side band and high frequency component > 2KHz of the wanted side band (default).
AVC_DIVIDER	set 0 for SSB mode; set 3 for SYNC mode.
AVCEN	SSB Automatic Volume Control (AVC) enable; 0=disable; 1=enable (default).
SMUTESEL	SSB Soft-mute Based on RSSI or SNR.
DSP_AFCDIS	DSP AFC Disable or enable; 0=SYNC MODE, AFC enable; 1=SSB MODE, AFC disable.

Definition at line 1835 of file SI4735.cpp.

void SI4735::setSSBDspAfc (uint8_t *DSP_AFCDIS*)

Sets DSP AFC disable or enable

See also

AN332 REV 0.8 UNIVERSAL PROGRAMMING GUIDE; page 24

Parameters

DSP_AFCDIS	0 = SYNC mode, AFC enable; 1 = SSB mode, AFC disable

Definition at line 1858 of file SI4735.cpp.

void SI4735::setSSBSoftMute (uint8_t *SMUTESEL*)

Sets SSB Soft-mute Based on RSSI or SNR Selection:

See also

AN332 REV 0.8 UNIVERSAL PROGRAMMING GUIDE; page 24

Parameters

SMUTESEL	0 = Soft-mute based on RSSI (default); 1 = Soft-mute based on SNR.

Definition at line 1872 of file SI4735.cpp.

void SI4735::setSsbSoftMuteMaxAttenuation ()`[inline]`

Definition at line 884 of file SI4735.h.

void SI4735::setSsbSoftMuteMaxAttenuation (uint8_t *smattn*)`[inline]`

Definition at line 883 of file SI4735.h.

void SI4735::setTuneFrequencyAntennaCapacitor (uint16_t *capacitor*)

Onlye FM. Freeze Metrics During Alternate Frequency Jump.

Selects the tuning capacitor value.

For FM, Antenna Tuning Capacitor is valid only when using TXO/LPI pin as the antenna input.

See also

Si47XX PROGRAMMING GUIDE; AN332; pages 71 and 136

Parameters

capacitor	If zero, the tuning capacitor value is selected automatically. If the value is set to anything other than 0: AM - the tuning capacitance is manually set as 95 fF x ANTCAP + 7 pF. ANTCAP manual range is 1–6143; FM - the valid range is 0 to 191. According to Silicon Labs, automatic capacitor tuning is recommended (value 0).

Definition at line 343 of file SI4735.cpp.

void SI4735::setTuneFrequencyFast (uint8_t *FAST*)`[inline]`

Returns the FAST tuning status.

Definition at line 951 of file SI4735.h.

void SI4735::setTuneFrequencyFreeze (uint8_t *FREEZE*)`[inline]`

Returns the FREEZE status.

Definition at line 953 of file SI4735.h.

void SI4735::setup (uint8_t *resetPin*, int *interruptPin*, uint8_t *defaultFunction*, uint8_t *audioMode* = `SI473X_ANALOG_AUDIO`)

Starts the Si473X device.

If the audio mode parameter is not entered, analog mode will be considered.

Parameters

uint8_t	resetPin Digital Arduino Pin used to RESET command
uint8_t	interruptPin interrupt Arduino Pin (see your Arduino pinout). If less than 0, iterrupt disabled
uint8_t	defaultFunction
uint8_t	audioMode default SI473X_ANALOG_AUDIO (Analog Audio). Use SI473X_ANALOG_AUDIO or SI473X_DIGITAL_AUDIO

Definition at line 279 of file SI4735.cpp.

void SI4735::setup (uint8_t *resetPin*, uint8_t *defaultFunction*)

Starts the Si473X device.

Use this setup if you are not using interrupt resource

Parameters

uint8_t	resetPin Digital Arduino Pin used to RESET command
uint8_t	defaultFunction

Definition at line 322 of file SI4735.cpp.

void SI4735::setVolume (uint8_t *volume*)

RESP8 - Returns the Chip Revision (ASCII).

Sets volume level (0 to 63)

See also

Si47XX PROGRAMMING GUIDE; AN332; pages 62, 123, 170, 173 and 204

Parameters

uint8_t	volume (domain: 0 - 63)

Definition at line 1140 of file SI4735.cpp.

void SI4735::ssbPowerUp ()

This function can be useful for debug and teste.

Definition at line 2115 of file SI4735.cpp.

void SI4735::ssbSetup ()

Starts the Si473X device on SSB (same AM Mode). Same **SI4735::setup** optimized to improve loading patch performance

Definition at line 2104 of file SI4735.cpp.

void SI4735::volumeDown ()

Set sound volume level Down

See also

setVolume()

Definition at line 1187 of file SI4735.cpp.

void SI4735::volumeUp ()

Set sound volume level Up

See also

setVolume()

Definition at line 1175 of file SI4735.cpp.

void SI4735::waitInterrupr (void) `[protected]`

If you setup interrupt, this function will be called whenever the Si4735 changes.

Definition at line 45 of file SI4735.cpp.

void SI4735::waitToSend (void)

Wait for the si473x is ready (Clear to Send (CTS) status bit have to be 1).

This function should be used before sending any command to a SI47XX device.

See also

Si47XX PROGRAMMING GUIDE; AN332; pages 63, 128

Definition at line 141 of file SI4735.cpp.

Member Data Documentation

si47x_agc_status SI4735::currentAgcStatus `[protected]`

Definition at line 814 of file SI4735.h.

uint8_t SI4735::currentAvcAmMaxGain = 48 `[protected]`

Store the last mode used.

Definition at line 806 of file SI4735.h.

si47x_frequency SI4735::currentFrequency `[protected]`

Definition at line 808 of file SI4735.h.

si47x_set_frequency SI4735::currentFrequencyParams `[protected]`

Definition at line 809 of file SI4735.h.

uint16_t SI4735::currentMaximumFrequency `[protected]`

Definition at line 799 of file SI4735.h.

uint16_t SI4735::currentMinimumFrequency `[protected]`

Definition at line 798 of file SI4735.h.

si47x_rds_status SI4735::currentRdsStatus `[protected]`

Definition at line 813 of file SI4735.h.

si47x_rqs_status SI4735::currentRqsStatus `[protected]`

Definition at line 810 of file SI4735.h.

si47x_ssb_mode SI4735::currentSSBMode `[protected]`

Definition at line 815 of file SI4735.h.

uint8_t SI4735::currentSsbStatus `[protected]`

Definition at line 821 of file SI4735.h.

si47x_response_status SI4735::currentStatus `[protected]`

Definition at line 811 of file SI4735.h.

uint16_t SI4735::currentStep `[protected]`

Definition at line 802 of file SI4735.h.

uint8_t SI4735::currentTune `[protected]`

Definition at line 796 of file SI4735.h.

uint16_t SI4735::currentWorkFrequency `[protected]`

Definition at line 800 of file SI4735.h.

int16_t SI4735::deviceAddress = SI473X_ADDR_SEN_LOW `[protected]`

Definition at line 790 of file SI4735.h.

si47x_firmware_information SI4735::firmwareInfo `[protected]`

Definition at line 812 of file SI4735.h.

uint8_t SI4735::interruptPin `[protected]`

Definition at line 794 of file SI4735.h.

uint8_t SI4735::lastMode = -1 `[protected]`

Definition at line 804 of file SI4735.h.

uint8_t SI4735::lastTextFlagAB `[protected]`

Definition at line 792 of file SI4735.h.

si473x_powerup SI4735::powerUp `[protected]`

Definition at line 817 of file SI4735.h.

char SI4735::rds_buffer0A[9] `[protected]`

RDS Radio Text buffer - Station Informaation.

Definition at line 783 of file SI4735.h.

char SI4735::rds_buffer2A[65] `[protected]`

Definition at line 781 of file SI4735.h.

char SI4735::rds_buffer2B[33] `[protected]`

RDS Radio Text buffer - Program Information.

Definition at line 782 of file SI4735.h.

char SI4735::rds_time[20] `[protected]`

RDS Basic tuning and switching information (Type 0 groups)

Definition at line 784 of file SI4735.h.

int SI4735::rdsTextAdress0A `[protected]`

Definition at line 788 of file SI4735.h.

int SI4735::rdsTextAdress2A `[protected]`

Definition at line 786 of file SI4735.h.

int SI4735::rdsTextAdress2B `[protected]`

Definition at line 787 of file SI4735.h.

uint8_t SI4735::resetPin `[protected]`

Definition at line 793 of file SI4735.h.

uint8_t SI4735::volume = 32 `[protected]`

Definition at line 819 of file SI4735.h.

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

- **SI4735.h**
- **SI4735.cpp**

si4735_digital_output_format Union Reference

```
#include <SI4735.h>
```

Public Attributes

- struct {
- uint8_t **OSIZE**: 2
- uint8_t **OMONO**: 1
  Digital Output Audio Sample Precision (0=16 bits, 1=20 bits, 2=24 bits, 3=8bits).
- uint8_t **OMODE**: 4
  Digital Output Mono Mode (0=Use mono/stereo blend).
- uint8_t **OFALL**: 1
  Digital Output Mode (0000=I2S, 0110 = Left-justified, 1000 = MSB at second DCLK after DFS pulse, 1100 = MSB at first DCLK after DFS pulse).
- uint8_t **dummy**: 8
  Digital Output DCLK Edge (0 = use DCLK rising edge, 1 = use DCLK falling edge)
- } **refined**
- uint16_t **raw**

Detailed Description

Digital audio output format data structure (Property 0x0102. DIGITAL_OUTPUT_FORMAT). Useed to configure: DCLK edge, data format, force mono, and sample precision.

See also

Si47XX PROGRAMMING GUIDE; AN332; page 195.

Definition at line 734 of file SI4735.h.

Member Data Documentation

uint8_t si4735_digital_output_format::dummy

Digital Output DCLK Edge (0 = use DCLK rising edge, 1 = use DCLK falling edge)

Definition at line 740 of file SI4735.h.

uint8_t si4735_digital_output_format::OFALL

Digital Output Mode (0000=I2S, 0110 = Left-justified, 1000 = MSB at second DCLK after DFS pulse, 1100 = MSB at first DCLK after DFS pulse).

Definition at line 739 of file SI4735.h.

uint8_t si4735_digital_output_format::OMODE

Digital Output Mono Mode (0=Use mono/stereo blend).

Definition at line 738 of file SI4735.h.

uint8_t si4735_digital_output_format::OMONO

Digital Output Audio Sample Precision (0=16 bits, 1=20 bits, 2=24 bits, 3=8bits).

Definition at line 737 of file SI4735.h.

uint8_t si4735_digital_output_format::OSIZE

Definition at line 736 of file SI4735.h.

uint16_t si4735_digital_output_format::raw

Definition at line 742 of file SI4735.h.

struct { ... }   si4735_digital_output_format::refined

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

- **SI4735.h**

si4735_digital_output_sample_rate Struct Reference

```
#include <SI4735.h>
```

Public Attributes

- uint16_t **DOSR**

Detailed Description

Digital audio output sample structure (Property 0x0104. DIGITAL_OUTPUT_SAMPLE_RATE). Used to enable digital audio output and to configure the digital audio output sample rate in samples per second (sps).

See also

Si47XX PROGRAMMING GUIDE; AN332; page 196.

Definition at line 751 of file SI4735.h.

Member Data Documentation

uint16_t si4735_digital_output_sample_rate::DOSR

Definition at line 752 of file SI4735.h.

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

- **SI4735.h**

si473x_powerup Union Reference

```
#include <SI4735.h>
```

Public Attributes

- struct {
-   uint8_t **FUNC**: 4
-   uint8_t **XOSCEN**: 1
  Function (0 = FM Receive; 1–14 = Reserved; 15 = Query Library ID)
-   uint8_t **PATCH**: 1
  Crystal Oscillator Enable (0 = crystal oscillator disabled; 1 = Use crystal oscillator and and OPMODE=ANALOG AUDIO) .
-   uint8_t **GPO2OEN**: 1
  Patch Enable (0 = Boot normally; 1 = Copy non-volatile memory to RAM).
-   uint8_t **CTSIEN**: 1
  GPO2 Output Enable (0 = GPO2 output disabled; 1 = GPO2 output enabled).
-   uint8_t **OPMODE**
  CTS Interrupt Enable (0 = CTS interrupt disabled; 1 = CTS interrupt enabled).
- } **arg**
- uint8_t **raw** [2]

Detailed Description

Power Up arguments data type

See also

Si47XX PROGRAMMING GUIDE; AN332; pages 64 and 65

Definition at line 160 of file SI4735.h.

Member Data Documentation

struct { ... } si473x_powerup::arg

uint8_t si473x_powerup::CTSIEN

GPO2 Output Enable (0 = GPO2 output disabled; 1 = GPO2 output enabled).

Definition at line 174 of file SI4735.h.

uint8_t si473x_powerup::FUNC

Definition at line 170 of file SI4735.h.

uint8_t si473x_powerup::GPO2OEN

Patch Enable (0 = Boot normally; 1 = Copy non-volatile memory to RAM).

Definition at line 173 of file SI4735.h.

uint8_t si473x_powerup::OPMODE

CTS Interrupt Enable (0 = CTS interrupt disabled; 1 = CTS interrupt enabled).

Definition at line 176 of file SI4735.h.

uint8_t si473x_powerup::PATCH

Crystal Oscillator Enable (0 = crystal oscillator disabled; 1 = Use crystal oscillator and and OPMODE=ANALOG AUDIO) .

Definition at line 172 of file SI4735.h.

uint8_t si473x_powerup::raw[2]

Definition at line 175 of file SI4735.h.

uint8_t si473x_powerup::XOSCEN

Function (0 = FM Receive; 1–14 = Reserved; 15 = Query Library ID)

Definition at line 171 of file SI4735.h.

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

- **SI4735.h**

si47x_agc_overrride Union Reference

`#include <SI4735.h>`

Public Attributes

- struct {
-   uint8_t **AGCDIS**: 1
-   uint8_t **DUMMY**: 7
-   uint8_t **AGCIDX**
- } **arg**
- uint8_t **raw** [2]

Detailed Description

If FM, Overrides AGC setting by disabling the AGC and forcing the LNA to have a certain gain that ranges between 0 (minimum attenuation) and 26 (maximum attenuation). If AM, overrides the AGC setting by disabling the AGC and forcing the gain index that ranges between 0

See also

Si47XX PROGRAMMING GUIDE; AN332; For FM page 81; for AM page 143

Definition at line 673 of file SI4735.h.

Member Data Documentation

uint8_t si47x_agc_overrride::AGCDIS

Definition at line 677 of file SI4735.h.

uint8_t si47x_agc_overrride::AGCIDX

Definition at line 680 of file SI4735.h.

struct { ... }   si47x_agc_overrride::arg

uint8_t si47x_agc_overrride::DUMMY

Definition at line 678 of file SI4735.h.

uint8_t si47x_agc_overrride::raw[2]

Definition at line 682 of file SI4735.h.

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

- **SI4735.h**

si47x_agc_status Union Reference

`#include <SI4735.h>`

Public Attributes

- struct {
- uint8_t **STCINT**: 1
- uint8_t **DUMMY1**: 1
- uint8_t **RDSINT**: 1
- uint8_t **RSQINT**: 1
- uint8_t **DUMMY2**: 2
- uint8_t **ERR**: 1
- uint8_t **CTS**: 1
- uint8_t **AGCDIS**: 1
- uint8_t **DUMMY**: 7
- uint8_t **AGCIDX**
- } **refined**
- uint8_t **raw** [3]

Detailed Description

AGC data types FM / AM and SSB structure to AGC

See also

Si47XX PROGRAMMING GUIDE; AN332; For FM page 80; for AM page 142

AN332 REV 0.8 Universal Programming Guide Amendment for SI4735-D60 SSB and NBFM patches; page 18.

Definition at line 646 of file SI4735.h.

Member Data Documentation

uint8_t si47x_agc_status::AGCDIS

Definition at line 658 of file SI4735.h.

uint8_t si47x_agc_status::AGCIDX

Definition at line 661 of file SI4735.h.

uint8_t si47x_agc_status::CTS

Definition at line 656 of file SI4735.h.

uint8_t si47x_agc_status::DUMMY

Definition at line 659 of file SI4735.h.

uint8_t si47x_agc_status::DUMMY1

Definition at line 651 of file SI4735.h.

uint8_t si47x_agc_status::DUMMY2

Definition at line 654 of file SI4735.h.

uint8_t si47x_agc_status::ERR

Definition at line 655 of file SI4735.h.

uint8_t si47x_agc_status::raw[3]

Definition at line 663 of file SI4735.h.

uint8_t si47x_agc_status::RDSINT

Definition at line 652 of file SI4735.h.

struct { ... }   si47x_agc_status::refined

uint8_t si47x_agc_status::RSQINT

Definition at line 653 of file SI4735.h.

uint8_t si47x_agc_status::STCINT

Definition at line 650 of file SI4735.h.

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

- **SI4735.h**

si47x_antenna_capacitor Union Reference

`#include <SI4735.h>`

Public Attributes

- struct {
-   uint8_t **ANTCAPL**
-   uint8_t **ANTCAPH**
  Antenna Tuning Capacitor High byte.
- } **raw**
- uint16_t **value**

Detailed Description

Antenna Tuning Capacitor data type manupulation

Definition at line 191 of file SI4735.h.

Member Data Documentation

uint8_t si47x_antenna_capacitor::ANTCAPH

Antenna Tuning Capacitor High byte.

Definition at line 195 of file SI4735.h.

uint8_t si47x_antenna_capacitor::ANTCAPL

Definition at line 194 of file SI4735.h.

struct { ... } si47x_antenna_capacitor::raw

uint16_t si47x_antenna_capacitor::value

Definition at line 197 of file SI4735.h.

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

- **SI4735.h**

si47x_bandwidth_config Union Reference

```
#include <SI4735.h>
```

Public Attributes

- struct {
-   uint8_t **AMCHFLT**: 4
-   uint8_t **DUMMY1**: 4

  Selects the bandwidth of the AM channel filter.

-   uint8_t **AMPLFLT**: 1
-   uint8_t **DUMMY2**: 7

  Enables the AM Power Line Noise Rejection Filter.

- } **param**
- uint8_t **raw** [2]

Detailed Description

The bandwidth of the AM channel filter data type AMCHFLT values: 0 = 6 kHz Bandwidth

1 = 4 kHz Bandwidth 2 = 3 kHz Bandwidth 3 = 2 kHz Bandwidth 4 = 1 kHz Bandwidth 5 = 1.8 kHz Bandwidth 6 = 2.5 kHz Bandwidth, gradual roll off 7–15 = Reserved (Do not use)

See also

Si47XX PROGRAMMING GUIDE; AN332; pages 125 and 151

Definition at line 698 of file SI4735.h.

Member Data Documentation

uint8_t si47x_bandwidth_config::AMCHFLT

Definition at line 701 of file SI4735.h.

uint8_t si47x_bandwidth_config::AMPLFLT

Definition at line 703 of file SI4735.h.

uint8_t si47x_bandwidth_config::DUMMY1

Selects the bandwidth of the AM channel filter.

Definition at line 702 of file SI4735.h.

uint8_t si47x_bandwidth_config::DUMMY2

Enables the AM Power Line Noise Rejection Filter.

Definition at line 704 of file SI4735.h.

struct { ... }   si47x_bandwidth_config::param

uint8_t si47x_bandwidth_config::raw[2]

Definition at line 706 of file SI4735.h.

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

- **SI4735.h**

si47x_firmware_information Union Reference

`#include <SI4735.h>`

Public Attributes

- struct {
-   uint8_t **STCINT**: 1
-   uint8_t **DUMMY1**: 1
-   uint8_t **RDSINT**: 1
-   uint8_t **RSQINT**: 1
-   uint8_t **DUMMY2**: 2
-   uint8_t **ERR**: 1
-   uint8_t **CTS**: 1
-   uint8_t **PN**
-   uint8_t **FWMAJOR**
  RESP1 - Final 2 digits of Part Number (HEX).
-   uint8_t **FWMINOR**
  RESP2 - Firmware Major Revision (ASCII).
-   uint8_t **PATCHH**
  RESP3 - Firmware Minor Revision (ASCII).
-   uint8_t **PATCHL**
  RESP4 - Patch ID High byte (HEX).
-   uint8_t **CMPMAJOR**
  RESP5 - Patch ID Low byte (HEX).
-   uint8_t **CMPMINOR**
  RESP6 - Component Major Revision (ASCII).
-   uint8_t **CHIPREV**
  RESP7 - Component Minor Revision (ASCII).
- } **resp**
- uint8_t **raw** [9]

Detailed Description

Data representation for Firmware Information (GET_REV) The part number, chip revision, firmware revision, patch revision and component revision numbers.

See also

Si47XX PROGRAMMING GUIDE; AN332; pages 66 and 131

Definition at line 278 of file SI4735.h.

Member Data Documentation

uint8_t si47x_firmware_information::CHIPREV

RESP7 - Component Minor Revision (ASCII).

Definition at line 296 of file SI4735.h.

uint8_t si47x_firmware_information::CMPMAJOR

RESP5 - Patch ID Low byte (HEX).

Definition at line 294 of file SI4735.h.

uint8_t si47x_firmware_information::CMPMINOR

RESP6 - Component Major Revision (ASCII).

Definition at line 295 of file SI4735.h.

uint8_t si47x_firmware_information::CTS

Definition at line 288 of file SI4735.h.

uint8_t si47x_firmware_information::DUMMY1

Definition at line 283 of file SI4735.h.

uint8_t si47x_firmware_information::DUMMY2

Definition at line 286 of file SI4735.h.

uint8_t si47x_firmware_information::ERR

Definition at line 287 of file SI4735.h.

uint8_t si47x_firmware_information::FWMAJOR

RESP1 - Final 2 digits of Part Number (HEX).

Definition at line 290 of file SI4735.h.

uint8_t si47x_firmware_information::FWMINOR

RESP2 - Firmware Major Revision (ASCII).

Definition at line 291 of file SI4735.h.

uint8_t si47x_firmware_information::PATCHH

RESP3 - Firmware Minor Revision (ASCII).

Definition at line 292 of file SI4735.h.

uint8_t si47x_firmware_information::PATCHL

RESP4 - Patch ID High byte (HEX).

Definition at line 293 of file SI4735.h.

uint8_t si47x_firmware_information::PN

Definition at line 289 of file SI4735.h.

uint8_t si47x_firmware_information::raw[9]

Definition at line 299 of file SI4735.h.

uint8_t si47x_firmware_information::RDSINT

Definition at line 284 of file SI4735.h.

struct { ... } si47x_firmware_information::resp

uint8_t si47x_firmware_information::RSQINT

Definition at line 285 of file SI4735.h.

uint8_t si47x_firmware_information::STCINT

Definition at line 282 of file SI4735.h.

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

- **SI4735.h**

si47x_firmware_query_library Union Reference

`#include <SI4735.h>`

Public Attributes

- struct {
- uint8_t **STCINT**: 1
- uint8_t **DUMMY1**: 1
- uint8_t **RDSINT**: 1
- uint8_t **RSQINT**: 1
- uint8_t **DUMMY2**: 2
- uint8_t **ERR**: 1
- uint8_t **CTS**: 1
- uint8_t **PN**
- uint8_t **FWMAJOR**
  RESP1 - Final 2 digits of Part Number (HEX).
- uint8_t **FWMINOR**
  RESP2 - Firmware Major Revision (ASCII).
- uint8_t **RESERVED1**
  RESP3 - Firmware Minor Revision (ASCII).
- uint8_t **RESERVED2**
  RESP4 - Reserved, various values.
- uint8_t **CHIPREV**
  RESP5 - Reserved, various values.
- uint8_t **LIBRARYID**
  RESP6 - Chip Revision (ASCII).
- } **resp**
- uint8_t **raw** [8]

Detailed Description

Firmware Query Library ID response. Used to represent the response of a power up command with FUNC = 15 (patch)

To confirm that the patch is compatible with the internal device library revision, the library revision should be confirmed by issuing the POWER_UP command with Function = 15 (query library ID)

See also

Si47XX PROGRAMMING GUIDE; AN332; page 12

Definition at line 311 of file SI4735.h.

Member Data Documentation

uint8_t si47x_firmware_query_library::CHIPREV

RESP5 - Reserved, various values.

Definition at line 327 of file SI4735.h.

uint8_t si47x_firmware_query_library::CTS

Definition at line 321 of file SI4735.h.

uint8_t si47x_firmware_query_library::DUMMY1

Definition at line 316 of file SI4735.h.

uint8_t si47x_firmware_query_library::DUMMY2

Definition at line 319 of file SI4735.h.

uint8_t si47x_firmware_query_library::ERR

Definition at line 320 of file SI4735.h.

uint8_t si47x_firmware_query_library::FWMAJOR

RESP1 - Final 2 digits of Part Number (HEX).

Definition at line 323 of file SI4735.h.

uint8_t si47x_firmware_query_library::FWMINOR

RESP2 - Firmware Major Revision (ASCII).

Definition at line 324 of file SI4735.h.

uint8_t si47x_firmware_query_library::LIBRARYID

RESP6 - Chip Revision (ASCII).

Definition at line 328 of file SI4735.h.

uint8_t si47x_firmware_query_library::PN

Definition at line 322 of file SI4735.h.

uint8_t si47x_firmware_query_library::raw[8]

Definition at line 331 of file SI4735.h.

uint8_t si47x_firmware_query_library::RDSINT

Definition at line 317 of file SI4735.h.

uint8_t si47x_firmware_query_library::RESERVED1

RESP3 - Firmware Minor Revision (ASCII).
Definition at line 325 of file SI4735.h.

uint8_t si47x_firmware_query_library::RESERVED2

RESP4 - Reserved, various values.
Definition at line 326 of file SI4735.h.

struct { ... } si47x_firmware_query_library::resp

uint8_t si47x_firmware_query_library::RSQINT

Definition at line 318 of file SI4735.h.

uint8_t si47x_firmware_query_library::STCINT

Definition at line 315 of file SI4735.h.

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

- **SI4735.h**

si47x_frequency Union Reference

```
#include <SI4735.h>
```

Public Attributes

- struct {
-   uint8_t **FREQL**
-   uint8_t **FREQH**

  Tune Frequency High byte.

- } **raw**
- uint16_t **value**

Detailed Description

Represents how the frequency is stored in the si4735. It helps to convert frequency in uint16_t to two bytes (uint8_t) (FREQL and FREQH)

Definition at line 179 of file SI4735.h.

Member Data Documentation

uint8_t si47x_frequency::FREQH

Tune Frequency High byte.

Definition at line 183 of file SI4735.h.

uint8_t si47x_frequency::FREQL

Definition at line 182 of file SI4735.h.

struct { ... } si47x_frequency::raw

uint16_t si47x_frequency::value

Definition at line 185 of file SI4735.h.

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

- **SI4735.h**

si47x_property Union Reference

```
#include <SI4735.h>
```

Public Attributes

- struct {
-   uint8_t **byteLow**
-   uint8_t **byteHigh**
- } **raw**
- uint16_t **value**

Detailed Description

Property Data type (help to deal with SET_PROPERTY command on si473X)

Definition at line 353 of file SI4735.h.

Member Data Documentation

uint8_t si47x_property::byteHigh

Definition at line 357 of file SI4735.h.

uint8_t si47x_property::byteLow

Definition at line 356 of file SI4735.h.

struct { ... } si47x_property::raw

uint16_t si47x_property::value

Definition at line 359 of file SI4735.h.

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

- **SI4735.h**

si47x_rds_blocka Union Reference

`#include <SI4735.h>`

Public Attributes

- struct {
-   uint16_t **pi**
- } **refined**
- struct {
-   uint8_t **highValue**
-   uint8_t **lowValue**
- } **raw**

Detailed Description

Block A data type

Definition at line 531 of file SI4735.h.

Member Data Documentation

uint8_t si47x_rds_blocka::highValue

Definition at line 538 of file SI4735.h.

uint8_t si47x_rds_blocka::lowValue

Definition at line 539 of file SI4735.h.

uint16_t si47x_rds_blocka::pi

Definition at line 534 of file SI4735.h.

struct { ... } si47x_rds_blocka::raw

struct { ... } si47x_rds_blocka::refined

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

- **SI4735.h**

si47x_rds_blockb Union Reference

`#include <SI4735.h>`

Public Attributes

- struct {
- uint16_t **address**: 2
- uint16_t **DI**: 1
- uint16_t **MS**: 1
- uint16_t **TA**: 1
- uint16_t **programType**: 5
- uint16_t **trafficProgramCode**: 1
- uint16_t **versionCode**: 1
- uint16_t **groupType**: 4
- } **group0**
- struct {
- uint16_t **address**: 4
- uint16_t **textABFlag**: 1
- uint16_t **programType**: 5
- uint16_t **trafficProgramCode**: 1
- uint16_t **versionCode**: 1
- uint16_t **groupType**: 4
- } **group2**
- struct {
- uint16_t **content**: 4
- uint16_t **textABFlag**: 1
- uint16_t **programType**: 5
- uint16_t **trafficProgramCode**: 1
- uint16_t **versionCode**: 1
- uint16_t **groupType**: 4
- } **refined**
- struct {
- uint8_t **lowValue**
- uint8_t **highValue**
- } **raw**

Detailed Description

Block B data type

For GCC on System-V ABI on 386-compatible (32-bit processors), the following stands: 1) Bit-fields are allocated from right to left (least to most significant). 2) A bit-field must entirely reside in a storage unit appropriate for its declared type. Thus a bit-field never crosses its unit boundary. 3) Bit-fields may share a storage unit with other struct/union members, including members that are not bit-fields. Of course, struct members occupy different parts of the storage unit. 4) Unnamed bit-fields' types do not affect the alignment of a structure or union, although individual bit-fields' member offsets obey the alignment constraints.

See also

also Si47XX PROGRAMMING GUIDE; AN332; pages 78 and 79

also `https://en.wikipedia.org/wiki/Radio_Data_System`

Definition at line 558 of file SI4735.h.

Member Data Documentation

uint16_t si47x_rds_blockb::address

Definition at line 561 of file SI4735.h.

uint16_t si47x_rds_blockb::content

Definition at line 581 of file SI4735.h.

uint16_t si47x_rds_blockb::DI

Definition at line 562 of file SI4735.h.

struct { ... } si47x_rds_blockb::group0

struct { ... } si47x_rds_blockb::group2

uint16_t si47x_rds_blockb::groupType

Definition at line 568 of file SI4735.h.

uint8_t si47x_rds_blockb::highValue

Definition at line 591 of file SI4735.h.

uint8_t si47x_rds_blockb::lowValue

Definition at line 590 of file SI4735.h.

uint16_t si47x_rds_blockb::MS

Definition at line 563 of file SI4735.h.

uint16_t si47x_rds_blockb::programType

Definition at line 565 of file SI4735.h.

struct { ... } si47x_rds_blockb::raw

struct { ... } si47x_rds_blockb::refined

uint16_t si47x_rds_blockb::TA

Definition at line 564 of file SI4735.h.

uint16_t si47x_rds_blockb::textABFlag

Definition at line 573 of file SI4735.h.

uint16_t si47x_rds_blockb::trafficProgramCode

Definition at line 566 of file SI4735.h.

uint16_t si47x_rds_blockb::versionCode

Definition at line 567 of file SI4735.h.

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

- **SI4735.h**

si47x_rds_command Union Reference

`#include <SI4735.h>`

Public Attributes

- struct {
-   uint8_t **INTACK**: 1
-   uint8_t **MTFIFO**: 1
-   uint8_t **STATUSONLY**: 1
-   uint8_t **dummy**: 5
- } **arg**
- uint8_t **raw**

Detailed Description

FM_RDS_STATUS (0x24) command Data type for command and response information

See also

Si47XX PROGRAMMING GUIDE; AN332; pages 77 and 78

Also `https://en.wikipedia.org/wiki/Radio_Data_System`

Definition at line 417 of file SI4735.h.

Member Data Documentation

struct { ... } si47x_rds_command::arg

uint8_t si47x_rds_command::dummy

Definition at line 423 of file SI4735.h.

uint8_t si47x_rds_command::INTACK

Definition at line 420 of file SI4735.h.

uint8_t si47x_rds_command::MTFIFO

Definition at line 421 of file SI4735.h.

uint8_t si47x_rds_command::raw

Definition at line 425 of file SI4735.h.

uint8_t si47x_rds_command::STATUSONLY

Definition at line 422 of file SI4735.h.

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

- **SI4735.h**

si47x_rds_config Union Reference

```
#include <SI4735.h>
```

Public Attributes

- struct {
-   uint8_t **RDSEN**: 1
-   uint8_t **DUMMY1**: 7
  1 = RDS Processing Enable.
-   uint8_t **BLETHD**: 2
-   uint8_t **BLETHC**: 2
  Block Error Threshold BLOCKD.
-   uint8_t **BLETHB**: 2
  Block Error Threshold BLOCKC.
-   uint8_t **BLETHA**: 2
  Block Error Threshold BLOCKB.
- } **arg**
- uint8_t **raw** [2]

Detailed Description

Data type for FM_RDS_CONFIG Property

IMPORTANT: all block errors must be less than or equal the associated block error threshold for the group to be stored in the RDS FIFO. 0 = No errors; 1 = 1–2 bit errors detected and corrected; 2 = 3–5 bit errors detected and corrected; 3 = Uncorrectable. Recommended Block Error Threshold options: 2,2,2,2 = No group stored if any errors are uncorrected. 3,3,3,3 = Group stored regardless of errors. 0,0,0,0 = No group stored containing corrected or uncorrected errors. 3,2,3,3 = Group stored with corrected errors on B, regardless of errors on A, C, or D.

See also

Si47XX PROGRAMMING GUIDE; AN332; pages 58 and 104

Definition at line 515 of file SI4735.h.

Member Data Documentation

struct { ... } si47x_rds_config::arg

uint8_t si47x_rds_config::BLETHA

Block Error Threshold BLOCKB.

Definition at line 523 of file SI4735.h.

uint8_t si47x_rds_config::BLETHB

Block Error Threshold BLOCKC.

Definition at line 522 of file SI4735.h.

uint8_t si47x_rds_config::BLETHC

Block Error Threshold BLOCKD.

Definition at line 521 of file SI4735.h.

uint8_t si47x_rds_config::BLETHD

Definition at line 520 of file SI4735.h.

uint8_t si47x_rds_config::DUMMY1

1 = RDS Processing Enable.

Definition at line 519 of file SI4735.h.

uint8_t si47x_rds_config::raw[2]

Definition at line 525 of file SI4735.h.

uint8_t si47x_rds_config::RDSEN

Definition at line 518 of file SI4735.h.

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

- **SI4735.h**

si47x_rds_date_time Union Reference

```
#include <SI4735.h>
```

Public Attributes

- struct {
-   uint8_t **offset**: 5
-   uint8_t **offset_sense**: 1
-   uint8_t **minute1**: 2
-   uint8_t **minute2**: 4
-   uint8_t **hour1**: 4
-   uint8_t **hour2**: 1
-   uint32_t **mjd**: 17
- } **refined**
- uint8_t **raw** [6]

Detailed Description

Group type 4A (RDS Date and Time) When group type 4A is used by the station, it shall be transmitted every minute according to EN 50067. This Structure uses blocks 2,3 and 5 (B,C,D)

ATTENTION: To make it compatible with 8, 16 and 32 bits platforms and avoid Crosses boundary, it was necessary to split minute and hour representation.

Definition at line 625 of file SI4735.h.

Member Data Documentation

uint8_t si47x_rds_date_time::hour1

Definition at line 632 of file SI4735.h.

uint8_t si47x_rds_date_time::hour2

Definition at line 633 of file SI4735.h.

uint8_t si47x_rds_date_time::minute1

Definition at line 630 of file SI4735.h.

uint8_t si47x_rds_date_time::minute2

Definition at line 631 of file SI4735.h.

uint32_t si47x_rds_date_time::mjd

Definition at line 634 of file SI4735.h.

uint8_t si47x_rds_date_time::offset

Definition at line 628 of file SI4735.h.

uint8_t si47x_rds_date_time::offset_sense

Definition at line 629 of file SI4735.h.

uint8_t si47x_rds_date_time::raw[6]

Definition at line 636 of file SI4735.h.

struct { ... } si47x_rds_date_time::refined

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

- **SI4735.h**

si47x_rds_int_source Union Reference

`#include <SI4735.h>`

Public Attributes

- struct {
- uint8_t **RDSRECV**: 1
- uint8_t **RDSSYNCLOST**: 1
  If set, generate RDSINT when RDS FIFO has at least FM_RDS_INT_FIFO_COUNT entries.
- uint8_t **RDSSYNCFOUND**: 1
  If set, generate RDSINT when RDS loses synchronization.
- uint8_t **DUMMY1**: 1
  f set, generate RDSINT when RDS gains synchronization.
- uint8_t **RDSNEWBLOCKA**: 1
  Always write to 0.
- uint8_t **RDSNEWBLOCKB**: 1
  If set, generate an interrupt when Block A data is found or subsequently changed.
- uint8_t **DUMMY2**: 5
  If set, generate an interrupt when Block B data is found or subsequently changed.
- uint8_t **DUMMY3**: 5
  Reserved - Always write to 0.
- } **refined**
- uint8_t **raw** [2]

Detailed Description

FM_RDS_INT_SOURCE property data type

See also

Si47XX PROGRAMMING GUIDE; AN332; page 103

also `https://en.wikipedia.org/wiki/Radio_Data_System`

Definition at line 486 of file SI4735.h.

Member Data Documentation

uint8_t si47x_rds_int_source::DUMMY1

f set, generate RDSINT when RDS gains synchronization.

Definition at line 492 of file SI4735.h.

uint8_t si47x_rds_int_source::DUMMY2

If set, generate an interrupt when Block B data is found or subsequently changed.

Definition at line 495 of file SI4735.h.

uint8_t si47x_rds_int_source::DUMMY3

Reserved - Always write to 0.

Definition at line 496 of file SI4735.h.

uint8_t si47x_rds_int_source::raw[2]

Definition at line 498 of file SI4735.h.

uint8_t si47x_rds_int_source::RDSNEWBLOCKA

Always write to 0.

Definition at line 493 of file SI4735.h.

uint8_t si47x_rds_int_source::RDSNEWBLOCKB

If set, generate an interrupt when Block A data is found or subsequently changed.

Definition at line 494 of file SI4735.h.

uint8_t si47x_rds_int_source::RDSRECV

Definition at line 489 of file SI4735.h.

uint8_t si47x_rds_int_source::RDSSYNCFOUND

If set, generate RDSINT when RDS loses synchronization.

Definition at line 491 of file SI4735.h.

uint8_t si47x_rds_int_source::RDSSYNCLOST

If set, generate RDSINT when RDS FIFO has at least FM_RDS_INT_FIFO_COUNT entries.

Definition at line 490 of file SI4735.h.

struct { ... }   si47x_rds_int_source::refined

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

- **SI4735.h**

si47x_rds_status Union Reference

```
#include <SI4735.h>
```

Public Attributes

- struct {
-   uint8_t **STCINT**: 1
-   uint8_t **DUMMY1**: 1
-   uint8_t **RDSINT**: 1
-   uint8_t **RSQINT**: 1
-   uint8_t **DUMMY2**: 2
-   uint8_t **ERR**: 1
-   uint8_t **CTS**: 1
-   uint8_t **RDSRECV**: 1
-   uint8_t **RDSSYNCLOST**: 1

  RDS Received; 1 = FIFO filled to minimum number of groups set by RDSFIFOCNT.

-   uint8_t **RDSSYNCFOUND**: 1

  RDS Sync Lost; 1 = Lost RDS synchronization.

-   uint8_t **DUMMY3**: 1

  RDS Sync Found; 1 = Found RDS synchronization.

-   uint8_t **RDSNEWBLOCKA**: 1
-   uint8_t **RDSNEWBLOCKB**: 1

  RDS New Block A; 1 = Valid Block A data has been received.

-   uint8_t **DUMMY4**: 2

  RDS New Block B; 1 = Valid Block B data has been received.

-   uint8_t **RDSSYNC**: 1
-   uint8_t **DUMMY5**: 1

  RDS Sync; 1 = RDS currently synchronized.

-   uint8_t **GRPLOST**: 1
-   uint8_t **DUMMY6**: 5

  Group Lost; 1 = One or more RDS groups discarded due to FIFO overrun.

-   uint8_t **RDSFIFOUSED**
-   uint8_t **BLOCKAH**

  RESP3 - RDS FIFO Used; Number of groups remaining in the RDS FIFO (0 if empty).

-   uint8_t **BLOCKAL**

  RESP4 - RDS Block A; HIGH byte.

-   uint8_t **BLOCKBH**

  RESP5 - RDS Block A; LOW byte.

- uint8_t **BLOCKBL**
  RESP6 - RDS Block B; HIGH byte.

- uint8_t **BLOCKCH**
  RESP7 - RDS Block B; LOW byte.

- uint8_t **BLOCKCL**
  RESP8 - RDS Block C; HIGH byte.

- uint8_t **BLOCKDH**
  RESP9 - RDS Block C; LOW byte.

- uint8_t **BLOCKDL**
  RESP10 - RDS Block D; HIGH byte.

- uint8_t **BLED**: 2
  RESP11 - RDS Block D; LOW byte.

- uint8_t **BLEC**: 2
- uint8_t **BLEB**: 2
- uint8_t **BLEA**: 2
- } **resp**
- uint8_t **raw** [13]

Detailed Description

Response data type for current channel and reads an entry from the RDS FIFO.

See also

Si47XX PROGRAMMING GUIDE; AN332; pages 77 and 78

Definition at line 433 of file SI4735.h.

Member Data Documentation

uint8_t si47x_rds_status::BLEA

Definition at line 475 of file SI4735.h.

uint8_t si47x_rds_status::BLEB

Definition at line 474 of file SI4735.h.

uint8_t si47x_rds_status::BLEC

Definition at line 473 of file SI4735.h.

uint8_t si47x_rds_status::BLED

RESP11 - RDS Block D; LOW byte.

Definition at line 472 of file SI4735.h.

uint8_t si47x_rds_status::BLOCKAH

RESP3 - RDS FIFO Used; Number of groups remaining in the RDS FIFO (0 if empty).

Definition at line 459 of file SI4735.h.

uint8_t si47x_rds_status::BLOCKAL

RESP4 - RDS Block A; HIGH byte.

Definition at line 460 of file SI4735.h.

uint8_t si47x_rds_status::BLOCKBH

RESP5 - RDS Block A; LOW byte.

Definition at line 461 of file SI4735.h.

uint8_t si47x_rds_status::BLOCKBL

RESP6 - RDS Block B; HIGH byte.

Definition at line 462 of file SI4735.h.

uint8_t si47x_rds_status::BLOCKCH

RESP7 - RDS Block B; LOW byte.

Definition at line 463 of file SI4735.h.

uint8_t si47x_rds_status::BLOCKCL

RESP8 - RDS Block C; HIGH byte.

Definition at line 464 of file SI4735.h.

uint8_t si47x_rds_status::BLOCKDH

RESP9 - RDS Block C; LOW byte.

Definition at line 465 of file SI4735.h.

uint8_t si47x_rds_status::BLOCKDL

RESP10 - RDS Block D; HIGH byte.

Definition at line 466 of file SI4735.h.

uint8_t si47x_rds_status::CTS

Definition at line 443 of file SI4735.h.

uint8_t si47x_rds_status::DUMMY1

Definition at line 438 of file SI4735.h.

uint8_t si47x_rds_status::DUMMY2

Definition at line 441 of file SI4735.h.

uint8_t si47x_rds_status::DUMMY3

RDS Sync Found; 1 = Found RDS synchronization.

Definition at line 448 of file SI4735.h.

uint8_t si47x_rds_status::DUMMY4

RDS New Block B; 1 = Valid Block B data has been received.

Definition at line 451 of file SI4735.h.

uint8_t si47x_rds_status::DUMMY5

RDS Sync; 1 = RDS currently synchronized.

Definition at line 454 of file SI4735.h.

uint8_t si47x_rds_status::DUMMY6

Group Lost; 1 = One or more RDS groups discarded due to FIFO overrun.

Definition at line 456 of file SI4735.h.

uint8_t si47x_rds_status::ERR

Definition at line 442 of file SI4735.h.

uint8_t si47x_rds_status::GRPLOST

Definition at line 455 of file SI4735.h.

uint8_t si47x_rds_status::raw[13]

Definition at line 477 of file SI4735.h.

uint8_t si47x_rds_status::RDSFIFOUSED

Definition at line 458 of file SI4735.h.

uint8_t si47x_rds_status::RDSINT

Definition at line 439 of file SI4735.h.

uint8_t si47x_rds_status::RDSNEWBLOCKA

Definition at line 449 of file SI4735.h.

uint8_t si47x_rds_status::RDSNEWBLOCKB

RDS New Block A; 1 = Valid Block A data has been received.

Definition at line 450 of file SI4735.h.

uint8_t si47x_rds_status::RDSRECV

Definition at line 445 of file SI4735.h.

uint8_t si47x_rds_status::RDSSYNC

Definition at line 453 of file SI4735.h.

uint8_t si47x_rds_status::RDSSYNCFOUND

RDS Sync Lost; 1 = Lost RDS synchronization.

Definition at line 447 of file SI4735.h.

uint8_t si47x_rds_status::RDSSYNCLOST

RDS Received; 1 = FIFO filled to minimum number of groups set by RDSFIFOCNT.

Definition at line 446 of file SI4735.h.

struct { ... } si47x_rds_status::resp

uint8_t si47x_rds_status::RSQINT

Definition at line 440 of file SI4735.h.

uint8_t si47x_rds_status::STCINT

Definition at line 437 of file SI4735.h.

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

- **SI4735.h**

si47x_response_status Union Reference

```
#include <SI4735.h>
```

Public Attributes

- struct {
- uint8_t **STCINT**: 1
- uint8_t **DUMMY1**: 1
  Seek/Tune Complete Interrupt; 1 = Tune complete has been triggered.

- uint8_t **RDSINT**: 1
- uint8_t **RSQINT**: 1
  Radio Data System (RDS) Interrup; 0 = interrupt has not been triggered.

- uint8_t **DUMMY2**: 2
  Received Signal Quality Interrupt; 0 = interrupt has not been triggered.

- uint8_t **ERR**: 1
- uint8_t **CTS**: 1
  Error. 0 = No error 1 = Error.

- uint8_t **VALID**: 1
  Clear to Send.

- uint8_t **AFCRL**: 1
  Valid Channel.

- uint8_t **DUMMY3**: 5
  AFC Rail Indicator.

- uint8_t **BLTF**: 1
- uint8_t **READFREQH**
  Reports if a seek hit the band limit.

- uint8_t **READFREQL**
  Read Frequency High byte.

- uint8_t **RSSI**
  Read Frequency Low byte.

- uint8_t **SNR**
  Received Signal Strength Indicator (dBÎ¼V)

- uint8_t **MULT**
  This byte contains the SNR metric when tune is complete (dB).

- uint8_t **READANTCAP**

Contains the multipath metric when tune is complete.

- } **resp**
- uint8_t **raw** [8]

Detailed Description

Response status command

See also

Si47XX PROGRAMMING GUIDE; pages 73 and

Definition at line 239 of file SI4735.h.

Member Data Documentation

uint8_t si47x_response_status::AFCRL

Valid Channel.

Definition at line 252 of file SI4735.h.

uint8_t si47x_response_status::BLTF

Definition at line 254 of file SI4735.h.

uint8_t si47x_response_status::CTS

Error. 0 = No error 1 = Error.

Definition at line 249 of file SI4735.h.

uint8_t si47x_response_status::DUMMY1

Seek/Tune Complete Interrupt; 1 = Tune complete has been triggered.

Definition at line 244 of file SI4735.h.

uint8_t si47x_response_status::DUMMY2

Received Signal Quality Interrupt; 0 = interrupt has not been triggered.

Definition at line 247 of file SI4735.h.

uint8_t si47x_response_status::DUMMY3

AFC Rail Indicator.

Definition at line 253 of file SI4735.h.

uint8_t si47x_response_status::ERR

Definition at line 248 of file SI4735.h.

uint8_t si47x_response_status::MULT

This byte contains the SNR metric when tune is complete (dB).

Definition at line 264 of file SI4735.h.

uint8_t si47x_response_status::raw[8]

Definition at line 268 of file SI4735.h.

uint8_t si47x_response_status::RDSINT

Definition at line 245 of file SI4735.h.

uint8_t si47x_response_status::READANTCAP

Contains the multipath metric when tune is complete.

Definition at line 266 of file SI4735.h.

uint8_t si47x_response_status::READFREQH

Reports if a seek hit the band limit.

Definition at line 256 of file SI4735.h.

uint8_t si47x_response_status::READFREQL

Read Frequency High byte.

Definition at line 258 of file SI4735.h.

struct { ... } si47x_response_status::resp

uint8_t si47x_response_status::RSQINT

Radio Data System (RDS) Interrup; 0 = interrupt has not been triggered.

Definition at line 246 of file SI4735.h.

uint8_t si47x_response_status::RSSI

Read Frequency Low byte.

Definition at line 260 of file SI4735.h.

uint8_t si47x_response_status::SNR

Received Signal Strength Indicator (dBÎ¼V)

Definition at line 262 of file SI4735.h.

uint8_t si47x_response_status::STCINT

Definition at line 243 of file SI4735.h.

uint8_t si47x_response_status::VALID

Clear to Send.

Definition at line 251 of file SI4735.h.

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

- **SI4735.h**

si47x_rqs_status Union Reference

`#include <SI4735.h>`

Public Attributes

- struct {
- uint8_t **STCINT**: 1
- uint8_t **DUMMY1**: 1
- uint8_t **RDSINT**: 1
- uint8_t **RSQINT**: 1
- uint8_t **DUMMY2**: 2
- uint8_t **ERR**: 1
- uint8_t **CTS**: 1
- uint8_t **RSSIILINT**: 1
- uint8_t **RSSIHINT**: 1
  RSSI Detect Low.
- uint8_t **SNRLINT**: 1
  RSSI Detect High.
- uint8_t **SNRHINT**: 1
  SNR Detect Low.
- uint8_t **MULTLINT**: 1
  SNR Detect High.
- uint8_t **MULTHINT**: 1
  Multipath Detect Low.
- uint8_t **DUMMY3**: 1
  Multipath Detect High.
- uint8_t **BLENDINT**: 1
- uint8_t **VALID**: 1
  Blend Detect Interrupt.
- uint8_t **AFCRL**: 1
  Valid Channel.
- uint8_t **DUMMY4**: 1
  AFC Rail Indicator.
- uint8_t **SMUTE**: 1
- uint8_t **DUMMY5**: 4
  Soft Mute Indicator. Indicates soft mute is engaged.
- uint8_t **STBLEND**: 7
- uint8_t **PILOT**: 1

Indicates amount of stereo blend in% (100 = full stereo, 0 = full mono).

- uint8_t **RSSI**
  Indicates stereo pilot presence.

- uint8_t **SNR**
  RESP4 - Contains the current receive signal strength (0â€“127 dBÎ¼V).

- uint8_t **MULT**
  RESP5 - Contains the current SNR metric (0–127 dB).

- uint8_t **FREQOFF**
  RESP6 - Contains the current multipath metric. (0 = no multipath; 100 = full multipath)

- } **resp**
- uint8_t **raw** [8]

Detailed Description

Data type for status information about the received signal quality FM_RSQ_STATUS and AM_RSQ_STATUS

See also

Si47XX PROGRAMMING GUIDE; AN332; pages 75 and

Definition at line 372 of file SI4735.h.

Member Data Documentation

uint8_t si47x_rqs_status::AFCRL

Valid Channel.

Definition at line 398 of file SI4735.h.

uint8_t si47x_rqs_status::BLENDINT

Definition at line 395 of file SI4735.h.

uint8_t si47x_rqs_status::CTS

Definition at line 386 of file SI4735.h.

uint8_t si47x_rqs_status::DUMMY1

Definition at line 381 of file SI4735.h.

uint8_t si47x_rqs_status::DUMMY2

Definition at line 384 of file SI4735.h.

uint8_t si47x_rqs_status::DUMMY3

Multipath Detect High.

Definition at line 394 of file SI4735.h.

uint8_t si47x_rqs_status::DUMMY4

AFC Rail Indicator.

Definition at line 399 of file SI4735.h.

uint8_t si47x_rqs_status::DUMMY5

Soft Mute Indicator. Indicates soft mute is engaged.

Definition at line 401 of file SI4735.h.

uint8_t si47x_rqs_status::ERR

Definition at line 385 of file SI4735.h.

uint8_t si47x_rqs_status::FREQOFF

RESP6 - Contains the current multipath metric. (0 = no multipath; 100 = full multipath)

Definition at line 409 of file SI4735.h.

uint8_t si47x_rqs_status::MULT

RESP5 - Contains the current SNR metric (0–127 dB).

Definition at line 408 of file SI4735.h.

uint8_t si47x_rqs_status::MULTHINT

Multipath Detect Low.

Definition at line 393 of file SI4735.h.

uint8_t si47x_rqs_status::MULTLINT

SNR Detect High.

Definition at line 392 of file SI4735.h.

uint8_t si47x_rqs_status::PILOT

Indicates amount of stereo blend in% (100 = full stereo, 0 = full mono).

Definition at line 404 of file SI4735.h.

uint8_t si47x_rqs_status::raw[8]

Definition at line 409 of file SI4735.h.

uint8_t si47x_rqs_status::RDSINT

Definition at line 382 of file SI4735.h.

struct { ... }   si47x_rqs_status::resp

uint8_t si47x_rqs_status::RSQINT

Definition at line 383 of file SI4735.h.

uint8_t si47x_rqs_status::RSSI

Indicates stereo pilot presence.
Definition at line 406 of file SI4735.h.

uint8_t si47x_rqs_status::RSSIHINT

RSSI Detect Low.
Definition at line 389 of file SI4735.h.

uint8_t si47x_rqs_status::RSSIILINT

Definition at line 388 of file SI4735.h.

uint8_t si47x_rqs_status::SMUTE

Definition at line 400 of file SI4735.h.

uint8_t si47x_rqs_status::SNR

RESP4 - Contains the current receive signal strength (0â€“127 dBÎ¼V).
Definition at line 407 of file SI4735.h.

uint8_t si47x_rqs_status::SNRHINT

SNR Detect Low.
Definition at line 391 of file SI4735.h.

uint8_t si47x_rqs_status::SNRLINT

RSSI Detect High.
Definition at line 390 of file SI4735.h.

uint8_t si47x_rqs_status::STBLEND

Definition at line 403 of file SI4735.h.

uint8_t si47x_rqs_status::STCINT

Definition at line 380 of file SI4735.h.

uint8_t si47x_rqs_status::VALID

Blend Detect Interrupt.

Definition at line 397 of file SI4735.h.

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

- **SI4735.h**

si47x_seek Union Reference

`#include <SI4735.h>`

Public Attributes

- struct {
-   uint8_t **RESERVED1**: 2
-   uint8_t **WRAP**: 1
-   uint8_t **SEEKUP**: 1
  Determines whether the seek should Wrap = 1, or Halt = 0 when it hits the band limit.
-   uint8_t **RESERVED2**: 4
  Determines the direction of the search, either UP = 1, or DOWN = 0.
- } **arg**
- uint8_t **raw**

Detailed Description

Represents searching for a valid frequency data type.

Definition at line 223 of file SI4735.h.

Member Data Documentation

struct { ... }   si47x_seek::arg

uint8_t si47x_seek::raw

Definition at line 231 of file SI4735.h.

uint8_t si47x_seek::RESERVED1

Definition at line 226 of file SI4735.h.

uint8_t si47x_seek::RESERVED2

Determines the direction of the search, either UP = 1, or DOWN = 0.

Definition at line 229 of file SI4735.h.

uint8_t si47x_seek::SEEKUP

Determines whether the seek should Wrap = 1, or Halt = 0 when it hits the band limit.

Definition at line 228 of file SI4735.h.

uint8_t si47x_seek::WRAP

Definition at line 227 of file SI4735.h.

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

- **SI4735.h**

si47x_set_frequency Union Reference

`#include <SI4735.h>`

Public Attributes

- struct {
-   uint8_t **FAST**: 1
-   uint8_t **FREEZE**: 1
  ARG1 - FAST Tuning. If set, executes fast and invalidated tune. The tune status will not be accurate.
-   uint8_t **DUMMY1**: 4
  Valid only for FM (Must be 0 to AM)
-   uint8_t **USBLSB**: 2
  Always set 0.
-   uint8_t **FREQH**
  SSB Upper Side Band (USB) and Lower Side Band (LSB) Selection. 10 = USB is selected; 01 = LSB is selected.
-   uint8_t **FREQL**
  ARG2 - Tune Frequency High byte.
-   uint8_t **ANTCAPH**
  ARG3 - Tune Frequency Low byte.
-   uint8_t **ANTCAPL**
  ARG4 - Antenna Tuning Capacitor High byte.
- } **arg**
- uint8_t **raw** [5]

Detailed Description

AM_TUNE_FREQ data type command

See also

Si47XX PROGRAMMING GUIDE; AN332; pages 135

Definition at line 205 of file SI4735.h.

Member Data Documentation

uint8_t si47x_set_frequency::ANTCAPH

ARG3 - Tune Frequency Low byte.

Definition at line 214 of file SI4735.h.

uint8_t si47x_set_frequency::ANTCAPL

ARG4 - Antenna Tuning Capacitor High byte.

Definition at line 215 of file SI4735.h.

struct { ... } si47x_set_frequency::arg

uint8_t si47x_set_frequency::DUMMY1

Valid only for FM (Must be 0 to AM)

Definition at line 210 of file SI4735.h.

uint8_t si47x_set_frequency::FAST

Definition at line 208 of file SI4735.h.

uint8_t si47x_set_frequency::FREEZE

ARG1 - FAST Tuning. If set, executes fast and invalidated tune. The tune status will not be accurate.

Definition at line 209 of file SI4735.h.

uint8_t si47x_set_frequency::FREQH

SSB Upper Side Band (USB) and Lower Side Band (LSB) Selection. 10 = USB is selected; 01 = LSB is selected.

Definition at line 212 of file SI4735.h.

uint8_t si47x_set_frequency::FREQL

ARG2 - Tune Frequency High byte.

Definition at line 213 of file SI4735.h.

uint8_t si47x_set_frequency::raw[5]

Definition at line 217 of file SI4735.h.

uint8_t si47x_set_frequency::USBLSB

Always set 0.

Definition at line 211 of file SI4735.h.

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

- **SI4735.h**

si47x_ssb_mode Union Reference

`#include <SI4735.h>`

Public Attributes

- struct {
-   uint8_t **AUDIOBW**: 4
-   uint8_t **SBCUTFLT**: 4
  0 = 1.2KHz (default); 1=2.2KHz; 2=3KHz; 3=4KHz; 4=500Hz; 5=1KHz
-   uint8_t **AVC_DIVIDER**: 4
  SSB side band cutoff filter for band passand low pass filter.
-   uint8_t **AVCEN**: 1
  set 0 for SSB mode; set 3 for SYNC mode;
-   uint8_t **SMUTESEL**: 1
  SSB Automatic Volume Control (AVC) enable; 0=disable; 1=enable (default);.
-   uint8_t **DUMMY1**: 1
  SSB Soft-mute Based on RSSI or SNR.
-   uint8_t **DSP_AFCDIS**: 1
  Always write 0;.
- } **param**
- uint8_t **raw** [2]

Detailed Description

SSB - datatype for SSB_MODE (property 0x0101)

See also

AN332 REV 0.8 UNIVERSAL PROGRAMMING GUIDE; page 24

Definition at line 714 of file SI4735.h.

Member Data Documentation

uint8_t si47x_ssb_mode::AUDIOBW

Definition at line 717 of file SI4735.h.

uint8_t si47x_ssb_mode::AVC_DIVIDER

SSB side band cutoff filter for band passand low pass filter.

Definition at line 719 of file SI4735.h.

uint8_t si47x_ssb_mode::AVCEN

set 0 for SSB mode; set 3 for SYNC mode;

Definition at line 720 of file SI4735.h.

uint8_t si47x_ssb_mode::DSP_AFCDIS

Always write 0;.

Definition at line 723 of file SI4735.h.

uint8_t si47x_ssb_mode::DUMMY1

SSB Soft-mute Based on RSSI or SNR.

Definition at line 722 of file SI4735.h.

struct { ... }   si47x_ssb_mode::param

uint8_t si47x_ssb_mode::raw[2]

Definition at line 725 of file SI4735.h.

uint8_t si47x_ssb_mode::SBCUTFLT

0 = 1.2KHz (default); 1=2.2KHz; 2=3KHz; 3=4KHz; 4=500Hz; 5=1KHz

Definition at line 718 of file SI4735.h.

uint8_t si47x_ssb_mode::SMUTESEL

SSB Automatic Volume Control (AVC) enable; 0=disable; 1=enable (default);.

Definition at line 721 of file SI4735.h.

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

- **SI4735.h**

si47x_tune_status Union Reference

`#include <SI4735.h>`

Public Attributes

- struct {
-   uint8_t **INTACK**: 1
-   uint8_t **CANCEL**: 1
  If set, clears the seek/tune complete interrupt status indicator.
-   uint8_t **RESERVED2**: 6
  If set, aborts a seek currently in progress.
- } **arg**
- uint8_t **raw**

Detailed Description

Status of FM_TUNE_FREQ or FM_SEEK_START commands or Status of AM_TUNE_FREQ or AM_SEEK_START commands.

See also

Si47XX PROGRAMMING GUIDE; AN332; pages 73 and 139

Definition at line 340 of file SI4735.h.

Member Data Documentation

struct { ... }   si47x_tune_status::arg

uint8_t si47x_tune_status::CANCEL

If set, clears the seek/tune complete interrupt status indicator.

Definition at line 344 of file SI4735.h.

uint8_t si47x_tune_status::INTACK

Definition at line 343 of file SI4735.h.

uint8_t si47x_tune_status::raw

Definition at line 347 of file SI4735.h.

uint8_t si47x_tune_status::RESERVED2

If set, aborts a seek currently in progress.

Definition at line 345 of file SI4735.h.

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

- **SI4735.h**

File Documentation

SI4735.cpp File Reference

```
#include <SI4735.h>
```

SI4735.h File Reference

```
#include <Arduino.h>
#include <Wire.h>
```

Classes

- union **si473x_powerup**
- union **si47x_frequency**
- union **si47x_antenna_capacitor**
- union **si47x_set_frequency**
- union **si47x_seek**
- union **si47x_response_status**
- union **si47x_firmware_information**
- union **si47x_firmware_query_library**
- union **si47x_tune_status**
- union **si47x_property**
- union **si47x_rqs_status**
- union **si47x_rds_command**
- union **si47x_rds_status**
- union **si47x_rds_int_source**
- union **si47x_rds_config**
- union **si47x_rds_blocka**
- union **si47x_rds_blockb**
- union **si47x_rds_date_time**
- union **si47x_agc_status**
- union **si47x_agc_overrride**
- union **si47x_bandwidth_config**
- union **si47x_ssb_mode**
- union **si4735_digital_output_format**
- struct **si4735_digital_output_sample_rate**
- class **SI4735**

Macros

- #define **POWER_UP_FM** 0
- #define **POWER_UP_AM** 1
- #define **POWER_UP_WB** 3
- #define **POWER_PATCH** 15
- #define **SI473X_ADDR_SEN_LOW** 0x11
- #define **SI473X_ADDR_SEN_HIGH** 0x63
- #define **POWER_UP** 0x01
- #define **GET_REV** 0x10
- #define **POWER_DOWN** 0x11
- #define **SET_PROPERTY** 0x12
- #define **GET_PROPERTY** 0x13
- #define **GET_INT_STATUS** 0x14
- #define **FM_TUNE_FREQ** 0x20
- #define **FM_SEEK_START** 0x21
- #define **FM_TUNE_STATUS** 0x22
- #define **FM_AGC_STATUS** 0x27
- #define **FM_AGC_OVERRIDE** 0x28
- #define **FM_RSQ_STATUS** 0x23
- #define **FM_RDS_STATUS** 0x24
- #define **FM_RDS_INT_SOURCE** 0x1500
- #define **FM_RDS_INT_FIFO_COUNT** 0x1501
- #define **FM_RDS_CONFIG** 0x1502

- #define **FM_RDS_CONFIDENCE** 0x1503
- #define **FM_BLEND_STEREO_THRESHOLD** 0x1105
- #define **FM_BLEND_MONO_THRESHOLD** 0x1106
- #define **FM_BLEND_RSSI_STEREO_THRESHOLD** 0x1800
- #define **FM_BLEND_RSSI_MONO_THRESHOLD** 0x1801
- #define **FM_BLEND_SNR_STEREO_THRESHOLD** 0x1804
- #define **FM_BLEND_SNR_MONO_THRESHOLD** 0x1805
- #define **FM_BLEND_MULTIPATH_STEREO_THRESHOLD** 0x1808
- #define **FM_BLEND_MULTIPATH_MONO_THRESHOLD** 0x1809
- #define **AM_TUNE_FREQ** 0x40
- #define **AM_SEEK_START** 0x41
- #define **AM_TUNE_STATUS** 0x42
- #define **AM_RSQ_STATUS** 0x43
- #define **AM_AGC_STATUS** 0x47
- #define **AM_AGC_OVERRIDE** 0x48
- #define **GPIO_CTL** 0x80
- #define **GPIO_SET** 0x81
- #define **SSB_TUNE_FREQ** 0x40
- #define **SSB_TUNE_STATUS** 0x42
- #define **SSB_RSQ_STATUS** 0x43
- #define **SSB_AGC_STATUS** 0x47
- #define **SSB_AGC_OVERRIDE** 0x48
- #define **DIGITAL_OUTPUT_FORMAT** 0x0102
- #define **DIGITAL_OUTPUT_SAMPLE_RATE** 0x0104
- #define **REFCLK_FREQ** 0x0201
- #define **REFCLK_PRESCALE** 0x0202
- #define **AM_DEEMPHASIS** 0x3100
- #define **AM_CHANNEL_FILTER** 0x3102
- #define **AM_AUTOMATIC_VOLUME_CONTROL_MAX_GAIN** 0x3103
- #define **AM_MODE_AFC_SW_PULL_IN_RANGE** 0x3104
- #define **AM_MODE_AFC_SW_LOCK_IN_RANGE** 0x3105
- #define **AM_RSQ_INTERRUPTS** 0x3200
- #define **AM_RSQ_SNR_HIGH_THRESHOLD** 0x3201
- #define **AM_RSQ_SNR_LOW_THRESHOLD** 0x3202
- #define **AM_RSQ_RSSI_HIGH_THRESHOLD** 0x3203
- #define **AM_RSQ_RSSI_LOW_THRESHOLD** 0x3204
- #define **AM_SOFT_MUTE_RATE** 0x3300
- #define **AM_SOFT_MUTE_SLOPE** 0x3301
- #define **AM_SOFT_MUTE_MAX_ATTENUATION** 0x3302
- #define **AM_SOFT_MUTE_SNR_THRESHOLD** 0x3303
- #define **AM_SOFT_MUTE_RELEASE_RATE** 0x3304
- #define **AM_SOFT_MUTE_ATTACK_RATE** 0x3305
- #define **AM_SEEK_BAND_BOTTOM** 0x3400
- #define **AM_SEEK_BAND_TOP** 0x3401
- #define **AM_SEEK_FREQ_SPACING** 0x3402
- #define **AM_SEEK_SNR_THRESHOLD** 0x3403
- #define **AM_SEEK_RSSI_THRESHOLD** 0x3404
- #define **AM_AGC_ATTACK_RATE** 0x3702
- #define **AM_AGC_RELEASE_RATE** 0x3703
- #define **AM_FRONTEND_AGC_CONTROL** 0x3705
- #define **AM_NB_DETECT_THRESHOLD** 0x3900
- #define **AM_NB_INTERVAL** 0x3901
- #define **AM_NB_RATE** 0x3902
- #define **AM_NB_IIR_FILTER** 0x3903
- #define **AM_NB_DELAY** 0x3904
- #define **RX_VOLUME** 0x4000
- #define **RX_HARD_MUTE** 0x4001

- #define **GPO_IEN** 0x0001
- #define **SSB_BFO** 0x0100
- #define **SSB_MODE** 0x0101
- #define **SSB_RSQ_INTERRUPTS** 0x3200
- #define **SSB_RSQ_SNR_HI_THRESHOLD** 0x3201
- #define **SSB_RSQ_SNR_LO_THRESHOLD** 0x3202
- #define **SSB_RSQ_RSSI_HI_THRESHOLD** 0x3203
- #define **SSB_RSQ_RSSI_LO_THRESHOLD** 0x3204
- #define **SSB_SOFT_MUTE_RATE** 0x3300
- #define **SSB_SOFT_MUTE_MAX_ATTENUATION** 0x3302
- #define **SSB_SOFT_MUTE_SNR_THRESHOLD** 0x3303
- #define **SSB_RF_AGC_ATTACK_RATE** 0x3700
- #define **SSB_RF_AGC_RELEASE_RATE** 0x3701
- #define **SSB_RF_IF_AGC_ATTACK_RATE** 0x3702
- #define **SSB_RF_IF_AGC_RELEASE_RATE** 0x3703
- #define **LSB_MODE** 1
- #define **USB_MODE** 2
- #define **SI473X_ANALOG_AUDIO** 0b00000101
- #define **SI473X_DIGITAL_AUDIO1** 0b00001011
- #define **SI473X_DIGITAL_AUDIO2** 0b10110000
- #define **SI473X_DIGITAL_AUDIO3** 0b10110101
- #define **FM_CURRENT_MODE** 0
- #define **AM_CURRENT_MODE** 1
- #define **SSB_CURRENT_MODE** 2
- #define **MAX_DELAY_AFTER_SET_FREQUENCY** 30
- #define **MIN_DELAY_WAIT_SEND_LOOP** 300

Macro Definition Documentation

#define AM_AGC_ATTACK_RATE 0x3702

Definition at line 99 of file SI4735.h.

#define AM_AGC_OVERRIDE 0x48

Definition at line 60 of file SI4735.h.

#define AM_AGC_RELEASE_RATE 0x3703

Definition at line 100 of file SI4735.h.

#define AM_AGC_STATUS 0x47

Definition at line 59 of file SI4735.h.

#define AM_AUTOMATIC_VOLUME_CONTROL_MAX_GAIN 0x3103

Definition at line 80 of file SI4735.h.

#define AM_CHANNEL_FILTER 0x3102

Definition at line 79 of file SI4735.h.

#define AM_CURRENT_MODE   1

Definition at line 145 of file SI4735.h.

#define AM_DEEMPHASIS   0x3100

Definition at line 78 of file SI4735.h.

#define AM_FRONTEND_AGC_CONTROL   0x3705

Definition at line 101 of file SI4735.h.

#define AM_MODE_AFC_SW_LOCK_IN_RANGE   0x3105

Definition at line 82 of file SI4735.h.

#define AM_MODE_AFC_SW_PULL_IN_RANGE   0x3104

Definition at line 81 of file SI4735.h.

#define AM_NB_DELAY   0x3904

Definition at line 106 of file SI4735.h.

#define AM_NB_DETECT_THRESHOLD   0x3900

Definition at line 102 of file SI4735.h.

#define AM_NB_IIR_FILTER   0x3903

Definition at line 105 of file SI4735.h.

#define AM_NB_INTERVAL   0x3901

Definition at line 103 of file SI4735.h.

#define AM_NB_RATE   0x3902

Definition at line 104 of file SI4735.h.

#define AM_RSQ_INTERRUPTS   0x3200

Definition at line 83 of file SI4735.h.

#define AM_RSQ_RSSI_HIGH_THRESHOLD   0x3203

Definition at line 86 of file SI4735.h.

#define AM_RSQ_RSSI_LOW_THRESHOLD 0x3204

Definition at line 87 of file SI4735.h.

#define AM_RSQ_SNR_HIGH_THRESHOLD 0x3201

Definition at line 84 of file SI4735.h.

#define AM_RSQ_SNR_LOW_THRESHOLD 0x3202

Definition at line 85 of file SI4735.h.

#define AM_RSQ_STATUS 0x43

Definition at line 58 of file SI4735.h.

#define AM_SEEK_BAND_BOTTOM 0x3400

Definition at line 94 of file SI4735.h.

#define AM_SEEK_BAND_TOP 0x3401

Definition at line 95 of file SI4735.h.

#define AM_SEEK_FREQ_SPACING 0x3402

Definition at line 96 of file SI4735.h.

#define AM_SEEK_RSSI_THRESHOLD 0x3404

Definition at line 98 of file SI4735.h.

#define AM_SEEK_SNR_THRESHOLD 0x3403

Definition at line 97 of file SI4735.h.

#define AM_SEEK_START 0x41

Definition at line 56 of file SI4735.h.

#define AM_SOFT_MUTE_ATTACK_RATE 0x3305

Definition at line 93 of file SI4735.h.

#define AM_SOFT_MUTE_MAX_ATTENUATION 0x3302

Definition at line 90 of file SI4735.h.

#define AM_SOFT_MUTE_RATE   0x3300

Definition at line 88 of file SI4735.h.

#define AM_SOFT_MUTE_RELEASE_RATE   0x3304

Definition at line 92 of file SI4735.h.

#define AM_SOFT_MUTE_SLOPE   0x3301

Definition at line 89 of file SI4735.h.

#define AM_SOFT_MUTE_SNR_THRESHOLD   0x3303

Definition at line 91 of file SI4735.h.

#define AM_TUNE_FREQ   0x40

Definition at line 55 of file SI4735.h.

#define AM_TUNE_STATUS   0x42

Definition at line 57 of file SI4735.h.

#define DIGITAL_OUTPUT_FORMAT   0x0102

Definition at line 74 of file SI4735.h.

#define DIGITAL_OUTPUT_SAMPLE_RATE   0x0104

Definition at line 75 of file SI4735.h.

#define FM_AGC_OVERRIDE   0x28

Definition at line 35 of file SI4735.h.

#define FM_AGC_STATUS   0x27

Definition at line 34 of file SI4735.h.

#define FM_BLEND_MONO_THRESHOLD   0x1106

Definition at line 46 of file SI4735.h.

#define FM_BLEND_MULTIPATH_MONO_THRESHOLD   0x1809

Definition at line 52 of file SI4735.h.

#define FM_BLEND_MULTIPATH_STEREO_THRESHOLD   0x1808

Definition at line 51 of file SI4735.h.

#define FM_BLEND_RSSI_MONO_THRESHOLD   0x1801

Definition at line 48 of file SI4735.h.

#define FM_BLEND_RSSI_STEREO_THRESHOLD   0x1800

Definition at line 47 of file SI4735.h.

#define FM_BLEND_SNR_MONO_THRESHOLD   0x1805

Definition at line 50 of file SI4735.h.

#define FM_BLEND_SNR_STEREO_THRESHOLD   0x1804

Definition at line 49 of file SI4735.h.

#define FM_BLEND_STEREO_THRESHOLD   0x1105

Definition at line 45 of file SI4735.h.

#define FM_CURRENT_MODE   0

Definition at line 144 of file SI4735.h.

#define FM_RDS_CONFIDENCE   0x1503

Definition at line 43 of file SI4735.h.

#define FM_RDS_CONFIG   0x1502

Definition at line 42 of file SI4735.h.

#define FM_RDS_INT_FIFO_COUNT   0x1501

Definition at line 41 of file SI4735.h.

#define FM_RDS_INT_SOURCE   0x1500

Definition at line 40 of file SI4735.h.

#define FM_RDS_STATUS   0x24

Definition at line 37 of file SI4735.h.

#define FM_RSQ_STATUS 0x23

Definition at line 36 of file SI4735.h.

#define FM_SEEK_START 0x21

Definition at line 32 of file SI4735.h.

#define FM_TUNE_FREQ 0x20

Definition at line 31 of file SI4735.h.

#define FM_TUNE_STATUS 0x22

Definition at line 33 of file SI4735.h.

#define GET_INT_STATUS 0x14

Definition at line 28 of file SI4735.h.

#define GET_PROPERTY 0x13

Definition at line 27 of file SI4735.h.

#define GET_REV 0x10

Definition at line 24 of file SI4735.h.

#define GPIO_CTL 0x80

Definition at line 61 of file SI4735.h.

#define GPIO_SET 0x81

Definition at line 62 of file SI4735.h.

#define GPO_IEN 0x0001

Definition at line 114 of file SI4735.h.

#define LSB_MODE 1

Definition at line 133 of file SI4735.h.

#define MAX_DELAY_AFTER_SET_FREQUENCY 30

Definition at line 147 of file SI4735.h.

#define MIN_DELAY_WAIT_SEND_LOOP 300

Definition at line 148 of file SI4735.h.

#define POWER_DOWN 0x11

Definition at line 25 of file SI4735.h.

#define POWER_PATCH 15

Definition at line 17 of file SI4735.h.

#define POWER_UP 0x01

Definition at line 23 of file SI4735.h.

#define POWER_UP_AM 1

Definition at line 15 of file SI4735.h.

#define POWER_UP_FM 0

Definition at line 14 of file SI4735.h.

#define POWER_UP_WB 3

Definition at line 16 of file SI4735.h.

#define REFCLK_FREQ 0x0201

Definition at line 76 of file SI4735.h.

#define REFCLK_PRESCALE 0x0202

Definition at line 77 of file SI4735.h.

#define RX_HARD_MUTE 0x4001

Definition at line 109 of file SI4735.h.

#define RX_VOLUME 0x4000

Definition at line 108 of file SI4735.h.

#define SET_PROPERTY 0x12

Definition at line 26 of file SI4735.h.

#define SI473X_ADDR_SEN_HIGH   0x63

Definition at line 21 of file SI4735.h.

#define SI473X_ADDR_SEN_LOW   0x11

Definition at line 20 of file SI4735.h.

#define SI473X_ANALOG_AUDIO   0b00000101

Definition at line 138 of file SI4735.h.

#define SI473X_DIGITAL_AUDIO1   0b00001011

Definition at line 139 of file SI4735.h.

#define SI473X_DIGITAL_AUDIO2   0b10110000

Definition at line 140 of file SI4735.h.

#define SI473X_DIGITAL_AUDIO3   0b10110101

Definition at line 141 of file SI4735.h.

#define SSB_AGC_OVERRIDE   0x48

Definition at line 70 of file SI4735.h.

#define SSB_AGC_STATUS   0x47

Definition at line 69 of file SI4735.h.

#define SSB_BFO   0x0100

Definition at line 115 of file SI4735.h.

#define SSB_CURRENT_MODE   2

Definition at line 146 of file SI4735.h.

#define SSB_MODE   0x0101

Definition at line 116 of file SI4735.h.

#define SSB_RF_AGC_ATTACK_RATE   0x3700

Definition at line 125 of file SI4735.h.

#define SSB_RF_AGC_RELEASE_RATE   0x3701

Definition at line 126 of file SI4735.h.

#define SSB_RF_IF_AGC_ATTACK_RATE   0x3702

Definition at line 129 of file SI4735.h.

#define SSB_RF_IF_AGC_RELEASE_RATE   0x3703

Definition at line 130 of file SI4735.h.

#define SSB_RSQ_INTERRUPTS   0x3200

Definition at line 117 of file SI4735.h.

#define SSB_RSQ_RSSI_HI_THRESHOLD   0x3203

Definition at line 120 of file SI4735.h.

#define SSB_RSQ_RSSI_LO_THRESHOLD   0x3204

Definition at line 121 of file SI4735.h.

#define SSB_RSQ_SNR_HI_THRESHOLD   0x3201

Definition at line 118 of file SI4735.h.

#define SSB_RSQ_SNR_LO_THRESHOLD   0x3202

Definition at line 119 of file SI4735.h.

#define SSB_RSQ_STATUS   0x43

Definition at line 68 of file SI4735.h.

#define SSB_SOFT_MUTE_MAX_ATTENUATION   0x3302

Definition at line 123 of file SI4735.h.

#define SSB_SOFT_MUTE_RATE   0x3300

Definition at line 122 of file SI4735.h.

#define SSB_SOFT_MUTE_SNR_THRESHOLD   0x3303

Definition at line 124 of file SI4735.h.

#define SSB_TUNE_FREQ   0x40

Definition at line 66 of file SI4735.h.

#define SSB_TUNE_STATUS   0x42

Definition at line 67 of file SI4735.h.

#define USB_MODE   2

Definition at line 134 of file SI4735.h.

Index

INDEX