Arduino Frequency Modulation Receiver

Generated by Doxygen 1.8.11

Contents

1 Hierarchical Index

1.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

2 Class Index

2.1 Class List

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

3 File Index

3.1 File List

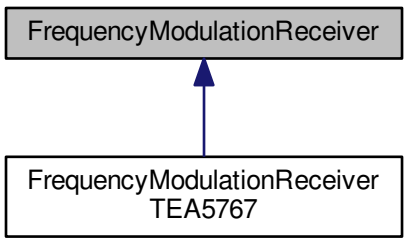
Here is a list of all files with brief descriptions:

4 Class Documentation

4.1 FrequencyModulationReceiver Class Reference

```
#include <FrequencyModulationReceiver.h>
```

Inheritance diagram for FrequencyModulationReceiver:



Public Types

- enum SerachDirection { SD_DOWN, SD_UP }
- enum Side { SIDE_LEFT = 0x01, SIDE_RIGHT = 0x02, SIDE_BOTH = SIDE_LEFT | SIDE_RIGHT }
- enum SideInjection { SI_LOW = 0x00, SI_HIGH = 0x01 }
- enum SearchStopLevel { SSL_LOW = 0x01, SSL_MID = 0x02, SSL_HIGH = 0x03 }

Public Member Functions

- FrequencyModulationReceiver ()
- virtual ~FrequencyModulationReceiver ()
- virtual void setFrequency (long frequency)=0
- virtual long getFrequency ()=0
- virtual void setStation (float station)=0
- virtual float getStation ()=0
- virtual void setStereo (bool stereo)=0
- virtual bool isStereo ()=0
- virtual void mute (Side size)=0
- virtual void unmute (Side size)=0
- virtual void setSoftmute (bool softmute)=0
- virtual bool isSoftmute ()=0
- virtual void setSearchDirection (SerachDirection direction)=0
- virtual void setStandby (bool standby)=0
- virtual bool isStandby ()=0
- virtual void setSearchStopLevel (SearchStopLevel level)=0
- virtual SearchStopLevel getSearchStopLevel ()=0
- virtual void setSearchMode (bool mode)=0
- virtual long searchNextFrequency ()=0
- virtual unsigned char getSignalLevel ()=0
- virtual bool isBandLimitReached ()=0
- virtual bool searchFinished ()=0
- virtual long getFoundStationFrequency ()=0
- virtual unsigned char getIntermediateFrequency ()=0

4.1.1 Detailed Description

Arduino - Radio Receiver.

Author

Dalmir da Silva dalmirdasilva@gmail.com

Definition at line 10 of file FrequencyModulationReceiver.h.

4.1.2 Member Enumeration Documentation

4.1.2.1 enum FrequencyModulationReceiver::SearchStopLevel

Enumerator

SSL_LOW

SSL_MID

SSL_HIGH

Definition at line 30 of file FrequencyModulationReceiver.h.

4.1.2.2 enum FrequencyModulationReceiver::SerachDirection

Enumerator

SD_DOWN

SD_UP

Definition at line 14 of file FrequencyModulationReceiver.h.

4.1.2.3 enum FrequencyModulationReceiver::Side

Enumerator

SIDE_LEFT

SIDE_RIGHT

SIDE_BOTH

Definition at line 19 of file FrequencyModulationReceiver.h.

4.1.2.4 enum FrequencyModulationReceiver::SideInjection

Enumerator

SI_LOW

SI_HIGH

Definition at line 25 of file FrequencyModulationReceiver.h.

4.1.3 Constructor & Destructor Documentation

4.1.3.1 FrequencyModulationReceiver::FrequencyModulationReceiver ( )

Definition at line 3 of file FrequencyModulationReceiver.cpp.

4.1.3.2 FrequencyModulationReceiver::~FrequencyModulationReceiver ( ) [virtual]

Definition at line 6 of file FrequencyModulationReceiver.cpp.

4.1.4 Member Function Documentation

4.1.4.1 virtual long FrequencyModulationReceiver::getFoundStationFrequency ( ) [pure virtual]

Implemented in FrequencyModulationReceiverTEA5767.

4.1.4.2 virtual long FrequencyModulationReceiver::getFrequency ( ) [pure virtual]

Implemented in FrequencyModulationReceiverTEA5767.

4.1.4.3 virtual unsigned char FrequencyModulationReceiver::getIntermediateFrequency ( ) [pure virtual]

Implemented in FrequencyModulationReceiverTEA5767.

4.1.4.4 virtual SearchStopLevel FrequencyModulationReceiver::getSearchStopLevel ( ) [pure virtual]

Implemented in FrequencyModulationReceiverTEA5767.

4.1.4.5 virtual unsigned char FrequencyModulationReceiver::getSignalLevel ( ) [pure virtual]

Implemented in FrequencyModulationReceiverTEA5767.

4.1.4.6 virtual float FrequencyModulationReceiver::getStation ( ) [pure virtual]

Implemented in FrequencyModulationReceiverTEA5767.

4.1.4.7 virtual bool FrequencyModulationReceiver::isBandLimitReached ( ) [pure virtual]

Implemented in FrequencyModulationReceiverTEA5767.

4.1.4.8 virtual bool FrequencyModulationReceiver::isSoftmute ( ) [pure virtual]

Implemented in FrequencyModulationReceiverTEA5767.

4.1.4.9 virtual bool FrequencyModulationReceiver::isStandby ( ) [pure virtual]

Implemented in FrequencyModulationReceiverTEA5767.

4.1.4.10 virtual bool FrequencyModulationReceiver::isStereo ( ) [pure virtual]

Implemented in FrequencyModulationReceiverTEA5767.

4.1.4.11 virtual void FrequencyModulationReceiver::mute (Side *size*) [pure virtual]

Implemented in FrequencyModulationReceiverTEA5767.

4.1.4.12 virtual bool FrequencyModulationReceiver::searchFinished ( ) [pure virtual]

Implemented in FrequencyModulationReceiverTEA5767.

4.1.4.13 virtual long FrequencyModulationReceiver::searchNextFrequency ( ) [pure virtual]

Implemented in FrequencyModulationReceiverTEA5767.

4.1.4.14 virtual void FrequencyModulationReceiver::setFrequency (long *frequency*) [pure virtual]

Implemented in FrequencyModulationReceiverTEA5767.

4.1.4.15 virtual void FrequencyModulationReceiver::setSearchDirection (SerachDirection *direction*) [pure virtual]

Implemented in FrequencyModulationReceiverTEA5767.

4.1.4.16 virtual void FrequencyModulationReceiver::setSearchMode (bool *mode*) [pure virtual]

Implemented in FrequencyModulationReceiverTEA5767.

4.1.4.17 virtual void FrequencyModulationReceiver::setSearchStopLevel (SearchStopLevel *level*) [pure virtual]

Implemented in FrequencyModulationReceiverTEA5767.

4.1.4.18 virtual void FrequencyModulationReceiver::setSoftmute (bool *softmute*) [pure virtual]

Implemented in FrequencyModulationReceiverTEA5767.

4.1.4.19 virtual void FrequencyModulationReceiver::setStandby (bool *standby*) [pure virtual]

Implemented in FrequencyModulationReceiverTEA5767.

4.1.4.20 virtual void FrequencyModulationReceiver::setStation (float *station*) [pure virtual]

Implemented in FrequencyModulationReceiverTEA5767.

4.1.4.21 virtual void FrequencyModulationReceiver::setStereo (bool *stereo*) [pure virtual]

Implemented in FrequencyModulationReceiverTEA5767.

4.1.4.22 virtual void FrequencyModulationReceiver::unmute (Side *size*) [pure virtual]

Implemented in FrequencyModulationReceiverTEA5767.

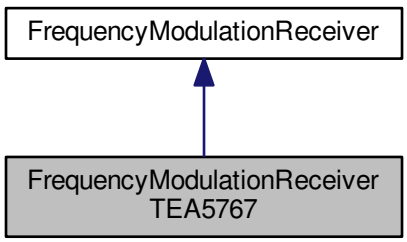
The documentation for this class was generated from the following files:

- FrequencyModulationReceiver.h
- FrequencyModulationReceiver.cpp

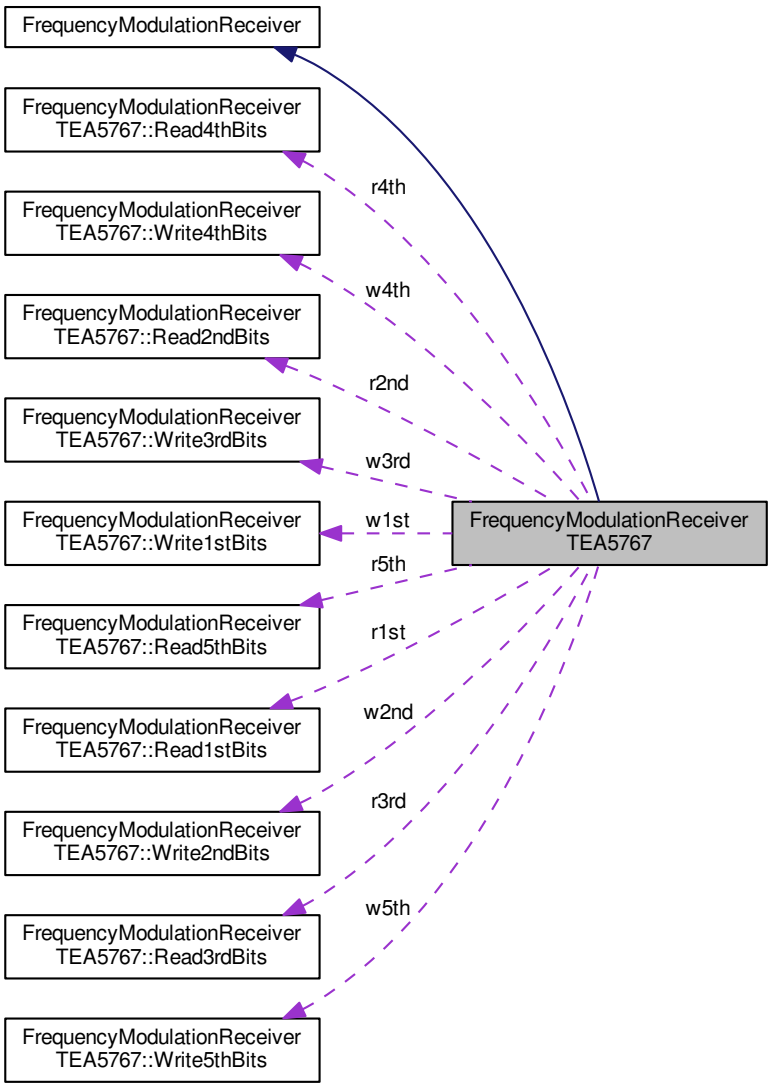
4.2 FrequencyModulationReceiverTEA5767 Class Reference

```
#include <FrequencyModulationReceiverTEA5767.h>
```

Inheritance diagram for FrequencyModulationReceiverTEA5767:



Collaboration diagram for FrequencyModulationReceiverTEA5767:



Classes

- union Read1stBits
- union Read2ndBits
- union Read3rdBits
- union Read4thBits
- union Read5thBits
- union Write1stBits
- union Write2ndBits
- union Write3rdBits
- union Write4thBits
- union Write5thBits

Public Member Functions

- FrequencyModulationReceiverTEA5767 ()
- void initialize ()
- void setFrequency (long frequency)
- long getFrequency ()
- void setStation (float station)
- float getStation ()
- void setStereo (bool stereo)
- bool isStereo ()
- void mute ()
- void mute (Side side)
- void unmute ()
- void unmute (Side side)
- void setSoftmute (bool softmute)
- bool isSoftmute ()
- void setSearchDirection (SerachDirection direction)
- void setStandby (bool standby)
- bool isStandby ()
- void setSearchStopLevel (SearchStopLevel level)
- SearchStopLevel getSearchStopLevel ()
- unsigned char getSignalLevel ()
- bool isBandLimitReached ()
- bool searchFinished ()
- float getFoundStation ()
- long getFoundStationFrequency ()
- unsigned char getIntermediateFrequency ()
- void setSearchMode (bool mode)
- long searchNextFrequency ()
- void setSideInjection (SideInjection level)
- void autoAjustSideInjection ()
- void setRawConfiguration (unsigned char *buf)
- long stationToFrequency (float station)
- float frequencyToStation (long frequency)

Private Member Functions

- void setMute (Side side, bool mute)
- long phaseLockedLoopToFrequency (unsigned int phaseLockedLoop)
- unsigned int frequencyToPhaseLockedLoop (long frequency)
- void applyFrequency ()
- void read ()
- void flush ()

Private Attributes

- long frequency
- Write1stBits w1st
- Write2ndBits w2nd
- Write3rdBits w3rd
- Write4thBits w4th
- Write5thBits w5th
- Read1stBits r1st
- Read2ndBits r2nd
- Read3rdBits r3rd
- Read4thBits r4th
- Read5thBits r5th

Additional Inherited Members

4.2.1 Detailed Description

Definition at line 28 of file FrequencyModulationReceiverTEA5767.h.

4.2.2 Constructor & Destructor Documentation

4.2.2.1 FrequencyModulationReceiverTEA5767::FrequencyModulationReceiverTEA5767 ( )

Definition at line 5 of file FrequencyModulationReceiverTEA5767.cpp.

4.2.3 Member Function Documentation

4.2.3.1 void FrequencyModulationReceiverTEA5767::applyFrequency ( ) [private]

Definition at line 209 of file FrequencyModulationReceiverTEA5767.cpp.

4.2.3.2 void FrequencyModulationReceiverTEA5767::autoAjustSideInjection ( )

Definition at line 129 of file FrequencyModulationReceiverTEA5767.cpp.

4.2.3.3 void FrequencyModulationReceiverTEA5767::flush ( ) [private]

Definition at line 227 of file FrequencyModulationReceiverTEA5767.cpp.

4.2.3.4 unsigned int FrequencyModulationReceiverTEA5767::frequencyToPhaseLockedLoop (long *frequency*) [private]

Definition at line 204 of file FrequencyModulationReceiverTEA5767.cpp.

4.2.3.5 float FrequencyModulationReceiverTEA5767::frequencyToStation (long *frequency*)

Definition at line 180 of file FrequencyModulationReceiverTEA5767.cpp.

4.2.3.6 float FrequencyModulationReceiverTEA5767::getFoundStation ( )

Definition at line 149 of file FrequencyModulationReceiverTEA5767.cpp.

4.2.3.7 long FrequencyModulationReceiverTEA5767::getFoundStationFrequency ( ) [virtual]

Implements FrequencyModulationReceiver.

Definition at line 153 of file FrequencyModulationReceiverTEA5767.cpp.

4.2.3.8 long FrequencyModulationReceiverTEA5767::getFrequency ( ) [virtual]

Implements FrequencyModulationReceiver.

Definition at line 27 of file FrequencyModulationReceiverTEA5767.cpp.

4.2.3.9 unsigned char FrequencyModulationReceiverTEA5767::getIntermediateFrequency ( ) [virtual]

Because the IF counter is not automatically updated, each time a read action is required, a previous write action must be performed.

After a write operation the result will be ready after 27ms. Wait until the result is available for the bus.

Implements FrequencyModulationReceiver.

Definition at line 161 of file FrequencyModulationReceiverTEA5767.cpp.

4.2.3.10 FrequencyModulationReceiverTEA5767::SearchStopLevel FrequencyModulationReceiverTEA5767::getSearchStopLevel ( ) [virtual]

Implements FrequencyModulationReceiver.

Definition at line 93 of file FrequencyModulationReceiverTEA5767.cpp.

4.2.3.11 unsigned char FrequencyModulationReceiverTEA5767::getSignalLevel ( ) [virtual]

Implements FrequencyModulationReceiver.

Definition at line 97 of file FrequencyModulationReceiverTEA5767.cpp.

4.2.3.12 float FrequencyModulationReceiverTEA5767::getStation ( ) [virtual]

Implements FrequencyModulationReceiver.

Definition at line 35 of file FrequencyModulationReceiverTEA5767.cpp.

4.2.3.13 void FrequencyModulationReceiverTEA5767::initialize ( )

Definition at line 9 of file FrequencyModulationReceiverTEA5767.cpp.

4.2.3.14 bool FrequencyModulationReceiverTEA5767::isBandLimitReached ( ) [virtual]

Implements FrequencyModulationReceiver.

Definition at line 139 of file FrequencyModulationReceiverTEA5767.cpp.

4.2.3.15 bool FrequencyModulationReceiverTEA5767::isSoftmute ( ) [virtual]

Implements FrequencyModulationReceiver.

Definition at line 70 of file FrequencyModulationReceiverTEA5767.cpp.

4.2.3.16 bool FrequencyModulationReceiverTEA5767::isStandby ( ) [virtual]

Implements FrequencyModulationReceiver.

Definition at line 84 of file FrequencyModulationReceiverTEA5767.cpp.

4.2.3.17 bool FrequencyModulationReceiverTEA5767::isStereo ( ) [virtual]

Implements FrequencyModulationReceiver.

Definition at line 44 of file FrequencyModulationReceiverTEA5767.cpp.

4.2.3.18 void FrequencyModulationReceiverTEA5767::mute ( )

Definition at line 49 of file FrequencyModulationReceiverTEA5767.cpp.

4.2.3.19 void FrequencyModulationReceiverTEA5767::mute (Side *side*) [virtual]

Implements FrequencyModulationReceiver.

Definition at line 53 of file FrequencyModulationReceiverTEA5767.cpp.

4.2.3.20 long FrequencyModulationReceiverTEA5767::phaseLockedLoopToFrequency (unsigned int *phaseLockedLoop*) [private]

Definition at line 199 of file FrequencyModulationReceiverTEA5767.cpp.

4.2.3.21 void FrequencyModulationReceiverTEA5767::read ( ) [private]

Definition at line 216 of file FrequencyModulationReceiverTEA5767.cpp.

4.2.3.22 bool FrequencyModulationReceiverTEA5767::searchFinished ( ) [virtual]

Implements FrequencyModulationReceiver.

Definition at line 144 of file FrequencyModulationReceiverTEA5767.cpp.

4.2.3.23 long FrequencyModulationReceiverTEA5767::searchNextFrequency ( ) [virtual]

To perform an autonomous search, MUTE and SEARCH bits should be set.

At the same time the current frequency has to be increased or decreased with one grid step.

Implements FrequencyModulationReceiver.

Definition at line 107 of file FrequencyModulationReceiverTEA5767.cpp.

4.2.3.24 void FrequencyModulationReceiverTEA5767::setFrequency (long *frequency*) [virtual]

Implements FrequencyModulationReceiver.

Definition at line 16 of file FrequencyModulationReceiverTEA5767.cpp.

4.2.3.25 void FrequencyModulationReceiverTEA5767::setMute (Side *side,* bool *mute*) [private]

Definition at line 184 of file FrequencyModulationReceiverTEA5767.cpp.

4.2.3.26 void FrequencyModulationReceiverTEA5767::setRawConfiguration (unsigned char * *buf*)

Definition at line 168 of file FrequencyModulationReceiverTEA5767.cpp.

4.2.3.27 void FrequencyModulationReceiverTEA5767::setSearchDirection (SerachDirection *direction*) [virtual]

Implements FrequencyModulationReceiver.

Definition at line 74 of file FrequencyModulationReceiverTEA5767.cpp.

4.2.3.28 void FrequencyModulationReceiverTEA5767::setSearchMode (bool *mode*) [virtual]

Implements FrequencyModulationReceiver.

Definition at line 102 of file FrequencyModulationReceiverTEA5767.cpp.

4.2.3.29 void FrequencyModulationReceiverTEA5767::setSearchStopLevel (SearchStopLevel *level*) [virtual]

Implements FrequencyModulationReceiver.

Definition at line 88 of file FrequencyModulationReceiverTEA5767.cpp.

4.2.3.30 void FrequencyModulationReceiverTEA5767::setSideInjection (SideInjection *level*)

Definition at line 124 of file FrequencyModulationReceiverTEA5767.cpp.

4.2.3.31 void FrequencyModulationReceiverTEA5767::setSoftmute (bool *softmute*) [virtual]

The RF input signal can come under weak level input so that the total noise energy in the AF spectrum can be larger than the AF signal.

This causes an unpleasant sound. When activated, the softmute will limit the amount of noise energy in the AF spectrum. The inter-station noise is then attenuated which will result in a better perception of the audio signal.

Implements FrequencyModulationReceiver.

Definition at line 65 of file FrequencyModulationReceiverTEA5767.cpp.

4.2.3.32 void FrequencyModulationReceiverTEA5767::setStandby (bool *standby*) [virtual]

By using the standby bit the IC can be switched into a low current Standby mode.

In Standby mode the IC must be in the WRITE mode. When the IC is switched to READ mode, during standby, the IC will hold the data line down. The standby current can be reduced by deactivating the bus interface (pin BUSENABLE LOW). If the bus interface is deactivated (pin BUSENABLE LOW) without the Standby mode being programmed, the IC maintains normal operation, but is isolated from the clock and data line.

Parameters

standby	Whether to activate or deactivate standby mode.

Implements FrequencyModulationReceiver.

Definition at line 79 of file FrequencyModulationReceiverTEA5767.cpp.

4.2.3.33 void FrequencyModulationReceiverTEA5767::setStation (float *station*) [virtual]

Implements FrequencyModulationReceiver.

Definition at line 31 of file FrequencyModulationReceiverTEA5767.cpp.

4.2.3.34 void FrequencyModulationReceiverTEA5767::setStereo (bool *stereo*) [virtual]

Implements FrequencyModulationReceiver.

Definition at line 39 of file FrequencyModulationReceiverTEA5767.cpp.

4.2.3.35 long FrequencyModulationReceiverTEA5767::stationToFrequency (float *station*)

Definition at line 176 of file FrequencyModulationReceiverTEA5767.cpp.

4.2.3.36 void FrequencyModulationReceiverTEA5767::unmute ( )

Definition at line 57 of file FrequencyModulationReceiverTEA5767.cpp.

4.2.3.37 void FrequencyModulationReceiverTEA5767::unmute (Side *side*) [virtual]

Implements FrequencyModulationReceiver.

Definition at line 61 of file FrequencyModulationReceiverTEA5767.cpp.

4.2.4 Member Data Documentation

4.2.4.1 long FrequencyModulationReceiverTEA5767::frequency [private]

Definition at line 197 of file FrequencyModulationReceiverTEA5767.h.

4.2.4.2 Read1stBits FrequencyModulationReceiverTEA5767::r1st [private]

Definition at line 205 of file FrequencyModulationReceiverTEA5767.h.

4.2.4.3 Read2ndBits FrequencyModulationReceiverTEA5767::r2nd [private]

Definition at line 206 of file FrequencyModulationReceiverTEA5767.h.

4.2.4.4 Read3rdBits FrequencyModulationReceiverTEA5767::r3rd [private]

Definition at line 207 of file FrequencyModulationReceiverTEA5767.h.

4.2.4.5 Read4thBits FrequencyModulationReceiverTEA5767::r4th [private]

Definition at line 208 of file FrequencyModulationReceiverTEA5767.h.

4.2.4.6 Read5thBits FrequencyModulationReceiverTEA5767::r5th [private]

Definition at line 209 of file FrequencyModulationReceiverTEA5767.h.

4.2.4.7 Write1stBits FrequencyModulationReceiverTEA5767::w1st [private]

Definition at line 199 of file FrequencyModulationReceiverTEA5767.h.

4.2.4.8 Write2ndBits FrequencyModulationReceiverTEA5767::w2nd [private]

Definition at line 200 of file FrequencyModulationReceiverTEA5767.h.

4.2.4.9 Write3rdBits FrequencyModulationReceiverTEA5767::w3rd [private]

Definition at line 201 of file FrequencyModulationReceiverTEA5767.h.

4.2.4.10 Write4thBits FrequencyModulationReceiverTEA5767::w4th [private]

Definition at line 202 of file FrequencyModulationReceiverTEA5767.h.

4.2.4.11 Write5thBits FrequencyModulationReceiverTEA5767::w5th [private]

Definition at line 203 of file FrequencyModulationReceiverTEA5767.h.

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

- FrequencyModulationReceiverTEA5767.h
- FrequencyModulationReceiverTEA5767.cpp

4.3 FrequencyModulationReceiverTEA5767::Read1stBits Union Reference

Public Attributes

- struct {
    unsigned char PLL:6
    unsigned char BLF:1
    unsigned char RF:1
  };

- unsigned char value

4.3.1 Detailed Description

Definition at line 132 of file FrequencyModulationReceiverTEA5767.h.

4.3.2 Member Data Documentation

4.3.2.1 struct { ... }

4.3.2.2 unsigned char FrequencyModulationReceiverTEA5767::Read1stBits::BLF

Definition at line 140 of file FrequencyModulationReceiverTEA5767.h.

4.3.2.3 unsigned char FrequencyModulationReceiverTEA5767::Read1stBits::PLL

Definition at line 137 of file FrequencyModulationReceiverTEA5767.h.

4.3.2.4 unsigned char FrequencyModulationReceiverTEA5767::Read1stBits::RF

Definition at line 143 of file FrequencyModulationReceiverTEA5767.h.

4.3.2.5 unsigned char FrequencyModulationReceiverTEA5767::Read1stBits::value

Definition at line 145 of file FrequencyModulationReceiverTEA5767.h.

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

- FrequencyModulationReceiverTEA5767.h

4.4 FrequencyModulationReceiverTEA5767::Read2ndBits Union Reference

Public Attributes

- struct {
  unsigned char PLL:8
  };
- unsigned char value

4.4.1 Detailed Description

Definition at line 148 of file FrequencyModulationReceiverTEA5767.h.

4.4.2 Member Data Documentation

4.4.2.1 struct { ... }

4.4.2.2 unsigned char FrequencyModulationReceiverTEA5767::Read2ndBits::PLL

Definition at line 153 of file FrequencyModulationReceiverTEA5767.h.

4.4.2.3 unsigned char FrequencyModulationReceiverTEA5767::Read2ndBits::value

Definition at line 155 of file FrequencyModulationReceiverTEA5767.h.

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

- FrequencyModulationReceiverTEA5767.h

4.5 FrequencyModulationReceiverTEA5767::Read3rdBits Union Reference

Public Attributes

- struct {
    unsigned char IF:7
    unsigned char STEREO:1
  };
- unsigned char value

4.5.1 Detailed Description

Definition at line 158 of file FrequencyModulationReceiverTEA5767.h.

4.5.2 Member Data Documentation

4.5.2.1 struct { ... }

4.5.2.2 unsigned char FrequencyModulationReceiverTEA5767::Read3rdBits::IF

Definition at line 163 of file FrequencyModulationReceiverTEA5767.h.

4.5.2.3 unsigned char FrequencyModulationReceiverTEA5767::Read3rdBits::STEREO

Definition at line 166 of file FrequencyModulationReceiverTEA5767.h.

4.5.2.4 unsigned char FrequencyModulationReceiverTEA5767::Read3rdBits::value

Definition at line 168 of file FrequencyModulationReceiverTEA5767.h.

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

- FrequencyModulationReceiverTEA5767.h

4.6 FrequencyModulationReceiverTEA5767::Read4thBits Union Reference

Public Attributes

- struct {
  unsigned char:1
  unsigned char CI:3
  unsigned char LEV:4
  };
- unsigned char value

4.6.1 Detailed Description

Definition at line 171 of file FrequencyModulationReceiverTEA5767.h.

4.6.2 Member Data Documentation

4.6.2.1 struct { ... }

4.6.2.2 unsigned FrequencyModulationReceiverTEA5767::Read4thBits::char

Definition at line 176 of file FrequencyModulationReceiverTEA5767.h.

4.6.2.3 unsigned char FrequencyModulationReceiverTEA5767::Read4thBits::CI

Definition at line 179 of file FrequencyModulationReceiverTEA5767.h.

4.6.2.4 unsigned char FrequencyModulationReceiverTEA5767::Read4thBits::LEV

Definition at line 182 of file FrequencyModulationReceiverTEA5767.h.

4.6.2.5 unsigned char FrequencyModulationReceiverTEA5767::Read4thBits::value

Definition at line 184 of file FrequencyModulationReceiverTEA5767.h.

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

- FrequencyModulationReceiverTEA5767.h

4.7 FrequencyModulationReceiverTEA5767::Read5thBits Union Reference

Public Attributes

- struct {
  unsigned char:8
  };
- unsigned char value

4.7.1 Detailed Description

Definition at line 187 of file FrequencyModulationReceiverTEA5767.h.

4.7.2 Member Data Documentation

4.7.2.1 struct { ... }

4.7.2.2 unsigned FrequencyModulationReceiverTEA5767::Read5thBits::char

Definition at line 192 of file FrequencyModulationReceiverTEA5767.h.

4.7.2.3 unsigned char FrequencyModulationReceiverTEA5767::Read5thBits::value

Definition at line 194 of file FrequencyModulationReceiverTEA5767.h.

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

- FrequencyModulationReceiverTEA5767.h

4.8 FrequencyModulationReceiverTEA5767::Write1stBits Union Reference

Public Attributes

- struct {
    unsigned char PLL:6
    unsigned char SM:1
    unsigned char MUTE:1
  };
- unsigned char value

4.8.1 Detailed Description

Definition at line 30 of file FrequencyModulationReceiverTEA5767.h.

4.8.2 Member Data Documentation

4.8.2.1 struct { ... }

4.8.2.2 unsigned char FrequencyModulationReceiverTEA5767::Write1stBits::MUTE

Definition at line 41 of file FrequencyModulationReceiverTEA5767.h.

4.8.2.3 unsigned char FrequencyModulationReceiverTEA5767::Write1stBits::PLL

Definition at line 35 of file FrequencyModulationReceiverTEA5767.h.

4.8.2.4 unsigned char FrequencyModulationReceiverTEA5767::Write1stBits::SM

Definition at line 38 of file FrequencyModulationReceiverTEA5767.h.

4.8.2.5 unsigned char FrequencyModulationReceiverTEA5767::Write1stBits::value

Definition at line 43 of file FrequencyModulationReceiverTEA5767.h.

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

- FrequencyModulationReceiverTEA5767.h

4.9 FrequencyModulationReceiverTEA5767::Write2ndBits Union Reference

Public Attributes

- struct {
    unsigned char PLL:8
  };
- unsigned char value

4.9.1 Detailed Description

Definition at line 46 of file FrequencyModulationReceiverTEA5767.h.

4.9.2 Member Data Documentation

4.9.2.1 struct { ... }

4.9.2.2 unsigned char FrequencyModulationReceiverTEA5767::Write2ndBits::PLL

Definition at line 51 of file FrequencyModulationReceiverTEA5767.h.

4.9.2.3 unsigned char FrequencyModulationReceiverTEA5767::Write2ndBits::value

Definition at line 53 of file FrequencyModulationReceiverTEA5767.h.

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

- FrequencyModulationReceiverTEA5767.h

4.10 FrequencyModulationReceiverTEA5767::Write3rdBits Union Reference

Public Attributes

- struct {
  unsigned char SWP1:1
  unsigned char ML:1
  unsigned char MR:1
  unsigned char MS:1
  unsigned char HLSI:1
  unsigned char SSL:2
  unsigned char SUD:1
  };
- unsigned char value

4.10.1 Detailed Description

Definition at line 56 of file FrequencyModulationReceiverTEA5767.h.

4.10.2 Member Data Documentation

4.10.2.1 struct { ... }

4.10.2.2 unsigned char FrequencyModulationReceiverTEA5767::Write3rdBits::HLSI

Definition at line 73 of file FrequencyModulationReceiverTEA5767.h.

4.10.2.3 unsigned char FrequencyModulationReceiverTEA5767::Write3rdBits::ML

Definition at line 64 of file FrequencyModulationReceiverTEA5767.h.

4.10.2.4 unsigned char FrequencyModulationReceiverTEA5767::Write3rdBits::MR

Definition at line 67 of file FrequencyModulationReceiverTEA5767.h.

4.10.2.5 unsigned char FrequencyModulationReceiverTEA5767::Write3rdBits::MS

Definition at line 70 of file FrequencyModulationReceiverTEA5767.h.

4.10.2.6 unsigned char FrequencyModulationReceiverTEA5767::Write3rdBits::SSL

Definition at line 76 of file FrequencyModulationReceiverTEA5767.h.

4.10.2.7 unsigned char FrequencyModulationReceiverTEA5767::Write3rdBits::SUD

Definition at line 79 of file FrequencyModulationReceiverTEA5767.h.

4.10.2.8 unsigned char FrequencyModulationReceiverTEA5767::Write3rdBits::SWP1

Definition at line 61 of file FrequencyModulationReceiverTEA5767.h.

4.10.2.9 unsigned char FrequencyModulationReceiverTEA5767::Write3rdBits::value

Definition at line 81 of file FrequencyModulationReceiverTEA5767.h.

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

- FrequencyModulationReceiverTEA5767.h

4.11 FrequencyModulationReceiverTEA5767::Write4thBits Union Reference

Public Attributes

- struct {
  unsigned char SI:1
  unsigned char SNC:1
  unsigned char HCC:1
  unsigned char SMUTE:1
  unsigned char XTAL:1
  unsigned char BL:1
  unsigned char STBY:1
  unsigned char SWP2:1
  };
- unsigned char value

4.11.1 Detailed Description

Definition at line 84 of file FrequencyModulationReceiverTEA5767.h.

4.11.2 Member Data Documentation

4.11.2.1 struct { ... }

4.11.2.2 unsigned char FrequencyModulationReceiverTEA5767::Write4thBits::BL

Definition at line 104 of file FrequencyModulationReceiverTEA5767.h.

4.11.2.3 unsigned char FrequencyModulationReceiverTEA5767::Write4thBits::HCC

Definition at line 95 of file FrequencyModulationReceiverTEA5767.h.

4.11.2.4 unsigned char FrequencyModulationReceiverTEA5767::Write4thBits::SI

Definition at line 89 of file FrequencyModulationReceiverTEA5767.h.

4.11.2.5 unsigned char FrequencyModulationReceiverTEA5767::Write4thBits::SMUTE

Definition at line 98 of file FrequencyModulationReceiverTEA5767.h.

4.11.2.6 unsigned char FrequencyModulationReceiverTEA5767::Write4thBits::SNC

Definition at line 92 of file FrequencyModulationReceiverTEA5767.h.

4.11.2.7 unsigned char FrequencyModulationReceiverTEA5767::Write4thBits::STBY

Definition at line 107 of file FrequencyModulationReceiverTEA5767.h.

4.11.2.8 unsigned char FrequencyModulationReceiverTEA5767::Write4thBits::SWP2

Definition at line 110 of file FrequencyModulationReceiverTEA5767.h.

4.11.2.9 unsigned char FrequencyModulationReceiverTEA5767::Write4thBits::value

Definition at line 112 of file FrequencyModulationReceiverTEA5767.h.

4.11.2.10 unsigned char FrequencyModulationReceiverTEA5767::Write4thBits::XTAL

Definition at line 101 of file FrequencyModulationReceiverTEA5767.h.

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

- FrequencyModulationReceiverTEA5767.h

4.12 FrequencyModulationReceiverTEA5767::Write5thBits Union Reference

Public Attributes

- struct {
  unsigned char:5
  unsigned char DTC:1
  unsigned char PLLREF:1
  };
- unsigned char value

4.12.1 Detailed Description

Definition at line 115 of file FrequencyModulationReceiverTEA5767.h.

4.12.2 Member Data Documentation

4.12.2.1 struct { ... }

4.12.2.2 unsigned FrequencyModulationReceiverTEA5767::Write5thBits::char

Definition at line 120 of file FrequencyModulationReceiverTEA5767.h.

4.12.2.3 unsigned char FrequencyModulationReceiverTEA5767::Write5thBits::DTC

Definition at line 123 of file FrequencyModulationReceiverTEA5767.h.

4.12.2.4 unsigned char FrequencyModulationReceiverTEA5767::Write5thBits::PLLREF

Definition at line 127 of file FrequencyModulationReceiverTEA5767.h.

4.12.2.5 unsigned char FrequencyModulationReceiverTEA5767::Write5thBits::value

Definition at line 129 of file FrequencyModulationReceiverTEA5767.h.

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

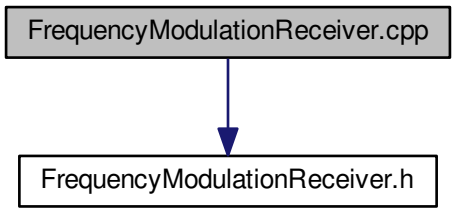
- FrequencyModulationReceiverTEA5767.h

5 File Documentation

5.1 FrequencyModulationReceiver.cpp File Reference

```
#include "FrequencyModulationReceiver.h"
```

Include dependency graph for FrequencyModulationReceiver.cpp:

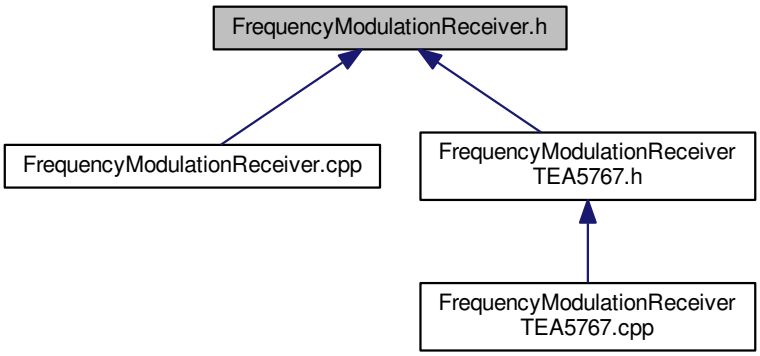


5.2 FrequencyModulationReceiver.cpp

```
00001 #include "FrequencyModulationReceiver.h"
00002
00003 FrequencyModulationReceiver::FrequencyModulationReceiver
      () {
00004 }
00005
00006 FrequencyModulationReceiver::~FrequencyModulationReceiver
      () {
00007 }
```

5.3 FrequencyModulationReceiver.h File Reference

This graph shows which files directly or indirectly include this file:



Classes

- class FrequencyModulationReceiver

5.4 FrequencyModulationReceiver.h

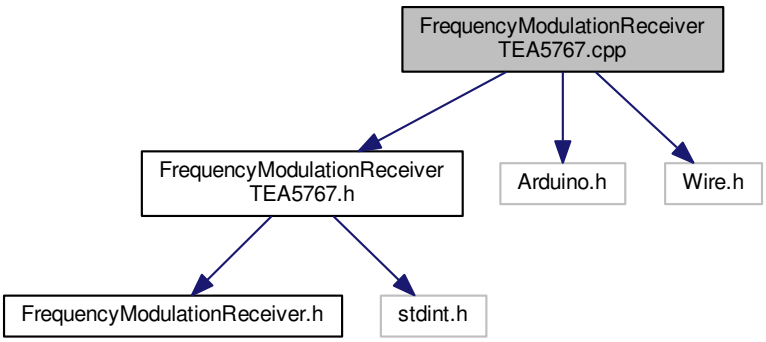
```
00001
00007 #ifndef __ARDUINO_FREQUENCY_MODULATION_RECEIVER_H__
00008 #define __ARDUINO_FREQUENCY_MODULATION_RECEIVER_H__ 1
00009
00010 class FrequencyModulationReceiver {
00011
00012 public:
00013
00014     enum SerachDirection {
00015         SD_DOWN,
00016         SD_UP
00017     };
00018
00019     enum Side {
00020         SIDE_LEFT = 0x01,
00021         SIDE_RIGHT = 0x02,
00022         SIDE_BOTH = SIDE_LEFT | SIDE_RIGHT
00023     };
00024
00025     enum SideInjection {
00026         SI_LOW = 0x00,
00027         SI_HIGH = 0x01
00028     };
00029
00030     enum SearchStopLevel {
00031         SSL_LOW = 0x01,
00032         SSL_MID = 0x02,
00033         SSL_HIGH = 0x03
00034     };
00035
00036     FrequencyModulationReceiver();
00037
00038     virtual ~FrequencyModulationReceiver();
00039
00040     virtual void setFrequency(long frequency) = 0;
00041
00042     virtual long getFrequency() = 0;
00043
00044     virtual void setStation(float station) = 0;
```

```
00045
00046     virtual float getStation() = 0;
00047
00048     virtual void setStereo(bool stereo) = 0;
00049
00050     virtual bool isStereo() = 0;
00051
00052     virtual void mute(Side size) = 0;
00053
00054     virtual void unmute(Side size) = 0;
00055
00056     virtual void setSoftmute(bool softmute) = 0;
00057
00058     virtual bool isSoftmute() = 0;
00059
00060     virtual void setSearchDirection(SerachDirection direction) = 0;
00061
00062     virtual void setStandby(bool standby) = 0;
00063
00064     virtual bool isStandby() = 0;
00065
00066     virtual void setSearchStopLevel(SearchStopLevel level) = 0;
00067
00068     virtual SearchStopLevel getSearchStopLevel() = 0;
00069
00070     virtual void setSearchMode(bool mode) = 0;
00071
00072     virtual long searchNextFrequency() = 0;
00073
00074     virtual unsigned char getSignalLevel() = 0;
00075
00076     virtual bool isBandLimitReached() = 0;
00077
00078     virtual bool searchFinished() = 0;
00079
00080     virtual long getFoundStationFrequency() = 0;
00081
00082     virtual unsigned char getIntermediateFrequency() = 0;
00083 };
00084
00085 #endif // __ARDUINO_FREQUENCY_MODULATION_RECEIVER_H__
```

5.5 FrequencyModulationReceiverTEA5767.cpp File Reference

```
#include <FrequencyModulationReceiverTEA5767.h>
#include <Arduino.h>
#include <Wire.h>
```

Include dependency graph for FrequencyModulationReceiverTEA5767.cpp:



5.6 FrequencyModulationReceiverTEA5767.cpp

```
00001 #include <FrequencyModulationReceiverTEA5767.h>
```

```
00002 #include <Arduino.h>
00003 #include <Wire.h>
00004
00005 FrequencyModulationReceiverTEA5767::FrequencyModulationReceiverTEA5767
      ()
00006         : FrequencyModulationReceiver(), frequency(0) {
00007 }
00008
00009 void FrequencyModulationReceiverTEA5767::initialize() {
00010     unsigned char raw[] = { 0x00, 0x00, 0xb0, 0x10, 0x00 };
00011     Wire.begin();
00012     read();
00013     setRawConfiguration(raw);
00014 }
00015
00016 void FrequencyModulationReceiverTEA5767::setFrequency(long
      frequency) {
00017     if (frequency < TEA5767_LOW_BAND_LIMIT_FREQ) {
00018         frequency = TEA5767_LOW_BAND_LIMIT_FREQ;
00019     } else if (frequency > TEA5767_HIGH_BAND_LIMIT_FREQ) {
00020         frequency = TEA5767_HIGH_BAND_LIMIT_FREQ;
00021     }
00022     this->frequency = frequency;
00023     applyFrequency();
00024     autoAjustSideInjection();
00025 }
00026
00027 long FrequencyModulationReceiverTEA5767::getFrequency() {
00028     return frequency;
00029 }
00030
00031 void FrequencyModulationReceiverTEA5767::setStation(float
      station) {
00032     setFrequency(stationToFrequency(station));
00033 }
00034
00035 float FrequencyModulationReceiverTEA5767::getStation() {
00036     return frequencyToStation(frequency);
00037 }
00038
00039 void FrequencyModulationReceiverTEA5767::setStereo(bool stereo
      ) {
00040     w3rd.MS = stereo;
00041     flush();
00042 }
00043
00044 bool FrequencyModulationReceiverTEA5767::isStereo() {
00045     read();
00046     return r3rd.STEREO;
00047 }
00048
00049 void FrequencyModulationReceiverTEA5767::mute() {
00050     mute(SIDE_BOTH);
00051 }
00052
00053 void FrequencyModulationReceiverTEA5767::mute(
      Side side) {
00054     setMute(side, 1);
00055 }
00056
00057 void FrequencyModulationReceiverTEA5767::unmute() {
00058     unmute(SIDE_BOTH);
00059 }
00060
00061 void FrequencyModulationReceiverTEA5767::unmute(
      Side side) {
00062     setMute(side, 0);
00063 }
00064
00065 void FrequencyModulationReceiverTEA5767::setSoftmute(bool
      softmute) {
00066     w4th.SMUTE = softmute;
00067     flush();
00068 }
00069
00070 bool FrequencyModulationReceiverTEA5767::isSoftmute() {
00071     return w4th.SMUTE;
00072 }
00073
00074 void FrequencyModulationReceiverTEA5767::setSearchDirection
      (SerachDirection direction) {
00075     w3rd.SUD = direction;
00076     flush();
00077 }
00078
00079 void FrequencyModulationReceiverTEA5767::setStandby(bool
      standby) {
```

```
00080     w4th.STBY = standby;
00081     flush();
00082 }
00083
00084 bool FrequencyModulationReceiverTEA5767::isStandby() {
00085     return w4th.STBY;
00086 }
00087
00088 void FrequencyModulationReceiverTEA5767::setSearchStopLevel
      (SearchStopLevel level) {
00089     w3rd.SSL = level;
00090     flush();
00091 }
00092
00093 FrequencyModulationReceiverTEA5767::SearchStopLevel
      FrequencyModulationReceiverTEA5767::getSearchStopLevel
      () {
00094     return (SearchStopLevel) w3rd.SSL;
00095 }
00096
00097 unsigned char FrequencyModulationReceiverTEA5767::getSignalLevel
      () {
00098     read();
00099     return r4th.LEV;
00100 }
00101
00102 void FrequencyModulationReceiverTEA5767::setSearchMode(
      bool mode) {
00103     w1st.SM = mode;
00104     flush();
00105 }
00106
00107 long FrequencyModulationReceiverTEA5767::searchNextFrequency
      () {
00108     long nextFrequency = 0;
00109     long oneGridStep = (w3rd.SUD == SD_UP) ? TEA5767_ONE_GRID_STEP : -(
      TEA5767_ONE_GRID_STEP);
00110     mute();
00111     setFrequency(frequency + oneGridStep);
00112     setSearchMode(true);
00113     while (!searchFinished())
00114         ;
00115     if (!isBandLimitReached()) {
00116         nextFrequency = getFoundStationFrequency();
00117     }
00118     setSearchMode(false);
00119     setFrequency(frequency - oneGridStep);
00120     unmute();
00121     return nextFrequency;
00122 }
00123
00124 void FrequencyModulationReceiverTEA5767::setSideInjection
      (SideInjection level) {
00125     w3rd.HLSI = level;
00126     flush();
00127 }
00128
00129 void FrequencyModulationReceiverTEA5767::autoAjustSideInjection
      () {
00130     unsigned char signal;
00131     setSideInjection(SI_LOW);
00132     signal = getSignalLevel();
00133     setSideInjection(SI_HIGH);
00134     if (signal > getSignalLevel()) {
00135         setSideInjection(SI_LOW);
00136     }
00137 }
00138
00139 bool FrequencyModulationReceiverTEA5767::isBandLimitReached
      () {
00140     read();
00141     return r1st.BLF;
00142 }
00143
00144 bool FrequencyModulationReceiverTEA5767::searchFinished()
       {
00145     read();
00146     return r1st.RF;
00147 }
00148
00149 float FrequencyModulationReceiverTEA5767::getFoundStation
      () {
00150     return frequencyToStation(getFoundStationFrequency());
00151 }
00152
00153 long FrequencyModulationReceiverTEA5767::getFoundStationFrequency
      () {
```

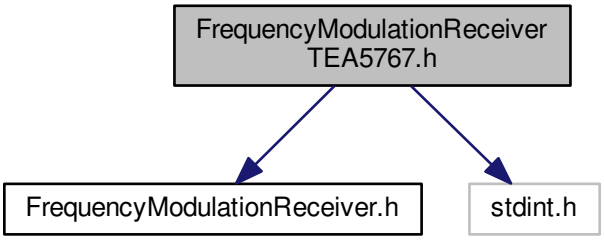
```
00154     unsigned int phaseLockedLoop = 0;
00155     phaseLockedLoop = r1st.PLL;
00156     phaseLockedLoop <<= 8;
00157     phaseLockedLoop |= r2nd.PLL;
00158     return phaseLockedLoopToFrequency(phaseLockedLoop);
00159 }
00160
00161 unsigned char FrequencyModulationReceiverTEA5767::getIntermediateFrequency
      () {
00162     flush();
00163     delay(TEA5767_IF_LOAD_DELAY_MS);
00164     read();
00165     return r3rd.IF;
00166 }
00167
00168 void FrequencyModulationReceiverTEA5767::setRawConfiguration
      (unsigned char *buf) {
00169     unsigned char* values[] = { &w1st.value, &w2nd.value, &
      w3rd.value, &w4th.value, &w5th.value };
00170     for (unsigned char i = 0; i < TEA5767_BYTES_COUNT; i++) {
00171         *values[i] = buf[i];
00172     }
00173     flush();
00174 }
00175
00176 long FrequencyModulationReceiverTEA5767::stationToFrequency
      (float station) {
00177     return station * TEA5767_STATION_TO_FREQ;
00178 }
00179
00180 float FrequencyModulationReceiverTEA5767::frequencyToStation
      (long frequency) {
00181     return frequency / TEA5767_STATION_TO_FREQ;
00182 }
00183
00184 void FrequencyModulationReceiverTEA5767::setMute(
      Side side, bool mute) {
00185     switch (side) {
00186     case SIDE_LEFT:
00187         w3rd.ML = mute;
00188         break;
00189     case SIDE_RIGHT:
00190         w3rd.MR = mute;
00191         break;
00192     case SIDE_BOTH:
00193         w1st.MUTE = mute;
00194         break;
00195     }
00196     flush();
00197 }
00198
00199 long FrequencyModulationReceiverTEA5767::phaseLockedLoopToFrequency
      (unsigned int phaseLockedLoop) {
00200     long intFreq = (w3rd.HLSI) ? TEA5767_INT_FREQ : -(
      TEA5767_INT_FREQ);
00201     return (phaseLockedLoop * TEA5767_REF_FREQ) / 4 - intFreq;
00202 }
00203
00204 unsigned int FrequencyModulationReceiverTEA5767::frequencyToPhaseLockedLoop
      (long frequency) {
00205     long intFreq = (w3rd.HLSI) ? TEA5767_INT_FREQ : -(
      TEA5767_INT_FREQ);
00206     return 4 * (frequency + intFreq) / TEA5767_REF_FREQ;
00207 }
00208
00209 void FrequencyModulationReceiverTEA5767::applyFrequency()
       {
00210     unsigned int phaseLockedLoop = frequencyToPhaseLockedLoop(
      frequency);
00211     w1st.PLL = (phaseLockedLoop >> 8) & 0x3f;
00212     w2nd.PLL = phaseLockedLoop & 0xff;
00213     flush();
00214 }
00215
00216 void FrequencyModulationReceiverTEA5767::read() {
00217     unsigned char* input[] = { &r1st.value, &r2nd.value, &r3rd.
      value, &r4th.value, &r5th.value };
00218     Wire.requestFrom(TEA5767_I2C_ADDRESS, TEA5767_BYTES_COUNT);
00219     if (Wire.available()) {
00220         for (unsigned char i = 0; i < TEA5767_BYTES_COUNT; i++) {
00221             *input[i] = Wire.read();
00222         }
00223     }
00224     delay(50);
00225 }
00226
00227 void FrequencyModulationReceiverTEA5767::flush() {
```

```
00228     unsigned char output[] = { w1st.value, w2nd.value, w3rd.
      value, w4th.value, w5th.value };
00229     Wire.beginTransmission(TEA5767_I2C_ADDRESS);
00230     for (unsigned char i = 0; i < TEA5767_BYTES_COUNT; i++) {
00231         Wire.write(output[i]);
00232     }
00233     Wire.endTransmission();
00234     delay(50);
00235 }
```

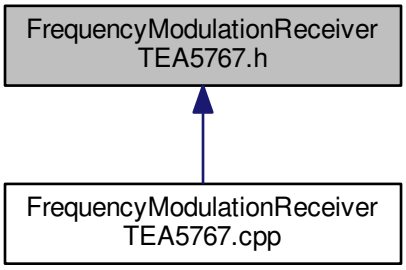
5.7 FrequencyModulationReceiverTEA5767.h File Reference

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

Include dependency graph for FrequencyModulationReceiverTEA5767.h:



This graph shows which files directly or indirectly include this file:



Classes

- class FrequencyModulationReceiverTEA5767
- union FrequencyModulationReceiverTEA5767::Write1stBits
- union FrequencyModulationReceiverTEA5767::Write2ndBits
- union FrequencyModulationReceiverTEA5767::Write3rdBits

- union FrequencyModulationReceiverTEA5767::Write4thBits
- union FrequencyModulationReceiverTEA5767::Write5thBits
- union FrequencyModulationReceiverTEA5767::Read1stBits
- union FrequencyModulationReceiverTEA5767::Read2ndBits
- union FrequencyModulationReceiverTEA5767::Read3rdBits
- union FrequencyModulationReceiverTEA5767::Read4thBits
- union FrequencyModulationReceiverTEA5767::Read5thBits

Macros

- #define TEA5767_I2C_ADDRESS 0x60
- #define TEA5767_REF_FREQ 32768.0
- #define TEA5767_INT_FREQ 225000.0
- #define TEA5767_BYTES_COUNT 5
- #define TEA5767_STATION_TO_FREQ 1000000.0
- #define TEA5767_IF_LOAD_DELAY_MS 27
- #define TEA5767_LOW_BAND_LIMIT_FREQ 87.5 ∗ TEA5767_STATION_TO_FREQ
- #define TEA5767_HIGH_BAND_LIMIT_FREQ 108 ∗ TEA5767_STATION_TO_FREQ
- #define TEA5767_ONE_GRID_STEP 98304.0

5.7.1 Macro Definition Documentation

5.7.1.1 #define TEA5767_BYTES_COUNT 5

Definition at line 16 of file FrequencyModulationReceiverTEA5767.h.

5.7.1.2 #define TEA5767_HIGH_BAND_LIMIT_FREQ 108 ∗ TEA5767_STATION_TO_FREQ

Definition at line 20 of file FrequencyModulationReceiverTEA5767.h.

5.7.1.3 #define TEA5767_I2C_ADDRESS 0x60

Arduino - Frequency Modulation Receiver.

Author

Dalmir da Silva dalmirdasilva@gmail.com

Definition at line 13 of file FrequencyModulationReceiverTEA5767.h.

5.7.1.4 #define TEA5767_IF_LOAD_DELAY_MS 27

Definition at line 18 of file FrequencyModulationReceiverTEA5767.h.

5.7.1.5 #define TEA5767_INT_FREQ 225000.0

Definition at line 15 of file FrequencyModulationReceiverTEA5767.h.

5.7.1.6 #define TEA5767_LOW_BAND_LIMIT_FREQ 87.5 * TEA5767_STATION_TO_FREQ

Definition at line 19 of file FrequencyModulationReceiverTEA5767.h.

5.7.1.7 #define TEA5767_ONE_GRID_STEP 98304.0

Definition at line 26 of file FrequencyModulationReceiverTEA5767.h.

5.7.1.8 #define TEA5767_REF_FREQ 32768.0

Definition at line 14 of file FrequencyModulationReceiverTEA5767.h.

5.7.1.9 #define TEA5767_STATION_TO_FREQ 1000000.0

Definition at line 17 of file FrequencyModulationReceiverTEA5767.h.

5.8 FrequencyModulationReceiverTEA5767.h

```
00001
00007 #ifndef __ARDUINO_FREQUENCY_MODULATION_RECEIVER_TEA5767_H__
00008 #define __ARDUINO_FREQUENCY_MODULATION_RECEIVER_TEA5767_H__ 1
00009
00010 #include <FrequencyModulationReceiver.h>
00011 #include <stdint.h>
00012
00013 #define TEA5767_I2C_ADDRESS             0x60
00014 #define TEA5767_REF_FREQ                32768.0
00015 #define TEA5767_INT_FREQ                225000.0
00016 #define TEA5767_BYTES_COUNT             5
00017 #define TEA5767_STATION_TO_FREQ         1000000.0
00018 #define TEA5767_IF_LOAD_DELAY_MS        27
00019 #define TEA5767_LOW_BAND_LIMIT_FREQ     87.5 * TEA5767_STATION_TO_FREQ
00020 #define TEA5767_HIGH_BAND_LIMIT_FREQ    108  * TEA5767_STATION_TO_FREQ
00021
00022 // Because the tuning system is internally provided with 100kHz grid step, care should be taken when the
       tuner is
00023 // clocked with the 32768Hz reference frequency. The grid step is then 98.304kHz (3*32768Hz).
00024 // In that case, when performing a search and a station is found, the PLL word of the programmable divider
       will
00025 // be read. The value of this word will be rounded and sent back to the tuner.
00026 #define TEA5767_ONE_GRID_STEP    98304.0
00027
00028 class FrequencyModulationReceiverTEA5767: public
     FrequencyModulationReceiver {
00029
00030     union Write1stBits {
00031
00032         struct {
00033
00034             // Setting of synthesizer programmable counter for search or preset.
00035             unsigned char PLL :6;
00036
00037             // Search mode: if SM = 1 then in search mode; if SM = 0 then not in search mode.
00038             unsigned char SM :1;
00039
00040             // If MUTE = 1 then L and R audio are muted; if MUTE = 0 then L and R audio are not muted.
00041             unsigned char MUTE :1;
00042         };
00043         unsigned char value;
00044     };
00045
00046     union Write2ndBits {
00047
00048         struct {
00049
00050             // Setting of synthesizer programmable counter for search or preset.
00051             unsigned char PLL :8;
00052         };
00053         unsigned char value;
00054     };
00055
00056     union Write3rdBits {
```

```
00057
00058         struct {
00059
00060             // Software programmable port 1: if SWP1 = 1 then port 1 is HIGH; if SWP1 = 0 then port 1 is
      LOW.
00061             unsigned char SWP1 :1;
00062
00063             // Mute Left: if ML = 1 then the left audio channel is muted and forced mono; if ML = 0 then
      the left audio channel is not muted.
00064             unsigned char ML :1;
00065
00066             // Mute Right: if MR = 1 then the right audio channel is muted and forced mono; if MR = 0 then
      the right audio channel is not muted.
00067             unsigned char MR :1;
00068
00069             // Mono to Stereo: if MS = 1 then forced mono; if MS = 0 then stereo ON.
00070             unsigned char MS :1;
00071
00072             // High/Low Side Injection: if HLSI = 1 then high side LO injection; if HLSI = 0 then low side
      LO injection.
00073             unsigned char HLSI :1;
00074
00075             // Search Stop Level.
00076             unsigned char SSL :2;
00077
00078             // Search Up/Down: if SUD = 1 then search up; if SUD = 0 then search down.
00079             unsigned char SUD :1;
00080         };
00081         unsigned char value;
00082     };
00083
00084     union Write4thBits {
00085
00086         struct {
00087
00088             // Search Indicator: if SI = 1 then pin SWPORT1 is output for the ready flag; if SI = 0 then
      pin SWPORT1 is software programmable port 1.
00089             unsigned char SI :1;
00090
00091             // Stereo Noise Cancelling: if SNC = 1 then stereo noise cancelling is ON; if SNC = 0 then
      stereo noise cancelling is OFF.
00092             unsigned char SNC :1;
00093
00094             // High Cut Control: if HCC = 1 then high cut control is ON; if HCC = 0 then high cut control
      is OFF.
00095             unsigned char HCC :1;
00096
00097             // Soft Mute: if SMUTE = 1 then soft mute is ON; if SMUTE = 0 then soft mute is OFF.
00098             unsigned char SMUTE :1;
00099
00100             // Clock frequency.
00101             unsigned char XTAL :1;
00102
00103             // Band Limits: if BL = 1 then Japanese FM band; if BL = 0 then US/Europe FM band.
00104             unsigned char BL :1;
00105
00106             // Standby: if STBY = 1 then in Standby mode; if STBY = 0 then not in Standby mode.
00107             unsigned char STBY :1;
00108
00109             // Software programmable port 2: if SWP2 = 1 then port 2 is HIGH; if SWP2 = 0 then port 2 is
      LOW.
00110             unsigned char SWP2 :1;
00111         };
00112         unsigned char value;
00113     };
00114
00115     union Write5thBits {
00116
00117         struct {
00118
00119             // Not used; position is don't care.
00120             unsigned char :5;
00121
00122             // If DTC = 1 then the de-emphasis time constant is 75 us; if DTC = 0 then the de-emphasis time
      constant is 50 us.
00123             unsigned char DTC :1;
00124
00125             // If PLLREF = 1 then the 6.5 MHz reference frequency for the PLL is enabled;
00126             // If PLLREF = 0 then the 6.5 MHz reference frequency for the PLL is disabled.
00127             unsigned char PLLREF :1;
00128         };
00129         unsigned char value;
00130     };
00131
00132     union Read1stBits {
00133
00134         struct {
```

```
00135
00136             // PLL[13:8] setting of synthesizer programmable counter after search or preset.
00137             unsigned char PLL :6;
00138
00139             // Band Limit Flag: if BLF = 1 then the band limit has been reached; if BLF = 0 then the band
      limit has not been reached.
00140             unsigned char BLF :1;
00141
00142             // Ready Flag: if RF = 1 then a station has been found or the band limit has been reached; if
      RF = 0 then no station has been found.
00143             unsigned char RF :1;
00144         };
00145         unsigned char value;
00146     };
00147
00148     union Read2ndBits {
00149
00150         struct {
00151
00152             // PLL[7:0] setting of synthesizer programmable counter after search or preset.
00153             unsigned char PLL :8;
00154         };
00155         unsigned char value;
00156     };
00157
00158     union Read3rdBits {
00159
00160         struct {
00161
00162             // IF counter result.
00163             unsigned char IF :7;
00164
00165             // Stereo indication: if STEREO = 1 then stereo reception; if STEREO = 0 then mono reception.
00166             unsigned char STEREO :1;
00167         };
00168         unsigned char value;
00169     };
00170
00171     union Read4thBits {
00172
00173         struct {
00174
00175             // This bit is internally set to logic 0.
00176             unsigned char :1;
00177
00178             // Chip Identification: these bits have to be set to logic 0.
00179             unsigned char CI :3;
00180
00181             // Level ADC output.
00182             unsigned char LEV :4;
00183         };
00184         unsigned char value;
00185     };
00186
00187     union Read5thBits {
00188
00189         struct {
00190
00191             // This bit is internally set to logic 0.
00192             unsigned char :8;
00193         };
00194         unsigned char value;
00195     };
00196
00197     long frequency;
00198
00199     Write1stBits w1st;
00200     Write2ndBits w2nd;
00201     Write3rdBits w3rd;
00202     Write4thBits w4th;
00203     Write5thBits w5th;
00204
00205     Read1stBits r1st;
00206     Read2ndBits r2nd;
00207     Read3rdBits r3rd;
00208     Read4thBits r4th;
00209     Read5thBits r5th;
00210 public:
00211
00212     FrequencyModulationReceiverTEA5767();
00213
00214     void initialize();
00215
00216     void setFrequency(long frequency);
00217
00218     long getFrequency();
00219
```

```
00220     void setStation(float station);
00221
00222     float getStation();
00223
00224     void setStereo(bool stereo);
00225
00226     bool isStereo();
00227
00228     void mute();
00229
00230     void mute(Side side);
00231
00232     void unmute();
00233
00234     void unmute(Side side);
00235
00242     void setSoftmute(bool softmute);
00243
00244     bool isSoftmute();
00245
00246     void setSearchDirection(SerachDirection direction);
00247
00259     void setStandby(bool standby);
00260
00261     bool isStandby();
00262
00263     void setSearchStopLevel(SearchStopLevel level);
00264
00265     SearchStopLevel getSearchStopLevel();
00266
00267     unsigned char getSignalLevel();
00268
00269     bool isBandLimitReached();
00270
00271     bool searchFinished();
00272
00273     float getFoundStation();
00274
00275     long getFoundStationFrequency();
00276
00281     unsigned char getIntermediateFrequency();
00282
00283     void setSearchMode(bool mode);
00284
00289     long searchNextFrequency();
00290
00291     void setSideInjection(SideInjection level);
00292
00293     void autoAjustSideInjection();
00294
00295     void setRawConfiguration(unsigned char *buf);
00296
00297     long stationToFrequency(float station);
00298
00299     float frequencyToStation(long frequency);
00300
00301 private:
00302
00303     void setMute(Side side, bool mute);
00304
00305     long phaseLockedLoopToFrequency(unsigned int phaseLockedLoop);
00306
00307     unsigned int frequencyToPhaseLockedLoop(long frequency);
00308
00309     void applyFrequency();
00310
00311     void read();
00312
00313     void flush();
00314 };
00315
00316 #endif // __ARDUINO_FREQUENCY_MODULATION_RECEIVER_TEA5767_H__
```

Index