Arduino MIDI Library

version 2.5

Generated by Doxygen 1.5.8

Mon Dec 14 14:45:29 2009

Contents

Chapter 1

Class Index

1.1 Class List

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

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

Chapter 3

Class Documentation

3.1 MIDI_Class Class Reference

`#include <MIDI.h>`

Public Member Functions

- MIDI_Class ()
- ~MIDI_Class ()
- void begin (byte inChannel=1)
- bool read ()
- bool read (byte channel)
- void send (byte type, byte param1, byte param2, byte canal)
- void sendNoteOn (byte NoteNumber, byte Velocity, byte Channel)
- void sendNoteOff (byte NoteNumber, byte Velocity, byte Channel)
- void sendProgramChange (byte ProgramNumber, byte Channel)
- void sendControlChange (byte ControlNumber, byte ControlValue, byte Channel)
- void sendPitchBend (int PitchValue, byte Channel)
- void sendPolyPressure (byte NoteNumber, byte Pressure, byte Channel)
- void sendAfterTouch (byte Pressure, byte Channel)
- void sendSysEx (byte length, byte *array)
- void turnThruOn ()
- void turnThruOff ()
- void turnThru (bool state)
- byte getType ()
- byte getChannel ()
- byte getData1 ()
- byte getData2 ()
- byte * getSysExArray ()
- bool check ()
- byte getInputChannel ()
- byte getFilterMode ()
- bool getThruState ()
- void setDeviceID (byte sysID)
- void delMsg ()

- void delSysEx ()
- void setInputChannel (byte channel)
- void setFilter (byte filter)

3.1.1 Detailed Description

Definition at line 69 of file MIDI.h.

3.1.2 Constructor & Destructor Documentation

3.1.2.1 MIDI_Class::MIDI_Class ()

Default constructor for MIDI_Class.

Definition at line 34 of file MIDI.cpp.

3.1.2.2 MIDI_Class::~MIDI_Class ()

Default destructor for MIDI_Class.

This is not really useful for the Arduino, as it is never called...

Definition at line 38 of file MIDI.cpp.

3.1.3 Member Function Documentation

3.1.3.1 void MIDI_Class::begin (byte *inChannel* = 1)

Call the begin method in the setup() function of the Arduino. All parameters are set to their default values:

- Full thru mirroring
- Input channel set to 1 if no value specified

Definition at line 46 of file MIDI.cpp.

3.1.3.2 bool MIDI_Class::check ()

Check if a valid message is stored in the structure.

Definition at line 148 of file MIDI.cpp.

3.1.3.3 void MIDI_Class::delMsg ()

Use this method to delete the message stored in the structure.

Definition at line 152 of file MIDI.cpp.

3.1.3.4 void MIDI_Class::delSysEx ()

Definition at line 161 of file MIDI.cpp.

3.1.3.5 byte MIDI_Class::getChannel ()

Getter method: access to the channel of the message stored in the structure.

Definition at line 140 of file MIDI.cpp.

3.1.3.6 byte MIDI_Class::getData1 ()

Getter method: access to the first data byte of the message stored in the structure.

If the message is SysEx, the length of the array is stocked there.

Definition at line 142 of file MIDI.cpp.

3.1.3.7 byte MIDI_Class::getData2 ()

Getter method: access to the second data byte of the message stored in the structure.

Definition at line 144 of file MIDI.cpp.

3.1.3.8 byte MIDI_Class::getFilterMode () [inline]

Definition at line 105 of file MIDI.h.

3.1.3.9 byte MIDI_Class::getInputChannel () [inline]

Definition at line 104 of file MIDI.h.

3.1.3.10 byte * MIDI_Class::getSysExArray ()

Getter method: access to the System Exclusive byte array. Array length is stocked in Data1.

Definition at line 146 of file MIDI.cpp.

3.1.3.11 bool MIDI_Class::getThruState () [inline]

Definition at line 106 of file MIDI.h.

3.1.3.12 byte MIDI_Class::getType ()

Getter method: access to the message type stored in the structure.

Definition at line 138 of file MIDI.cpp.

3.1.3.13 bool MIDI_Class::read (byte *channel*)

Reading/mirroring method, the same as read() with a given input channel to read on.

Definition at line 334 of file MIDI.cpp.

3.1.3.14 bool MIDI_Class::read ()

Read a MIDI message from the serial port using the main input channel (see setInputChannel() for reference).

Returned value: true if any valid message has been stored in the structure, false if not. A valid message is a message that matches the input channel.

If the Thru is enabled and the messages matches the filter, it is sent back on the MIDI output.

Definition at line 330 of file MIDI.cpp.

3.1.3.15 void MIDI_Class::send (byte *type*, byte *data1*, byte *data2*, byte *channel*)

Generate and send a custom MIDI mMessage.

Parameters:

type The message type (see type defines for reference)

data1 The first data byte

data2 The second data byte (if the message contains only 1 data byte, set this one to 0)

channel The output channel on which the message will be sent

Definition at line 74 of file MIDI.cpp.

3.1.3.16 void MIDI_Class::sendAfterTouch (byte *Pressure*, byte *Channel*)

Monophonic AfterTouch

Definition at line 125 of file MIDI.cpp.

3.1.3.17 void MIDI_Class::sendControlChange (byte *ControlNumber*, byte *ControlValue*, byte *Channel*)

Send a Control Change message

Definition at line 121 of file MIDI.cpp.

3.1.3.18 void MIDI_Class::sendNoteOff (byte *NoteNumber*, byte *Velocity*, byte *Channel*)

Send a Note Off message (a real Note Off, not a Note On with null velocity)

Definition at line 117 of file MIDI.cpp.

3.1.3.19 void MIDI_Class::sendNoteOn (byte *NoteNumber*, byte *Velocity*, byte *Channel*)

Send a Note On message

Definition at line 115 of file MIDI.cpp.

3.1.3.20 void MIDI_Class::sendPitchBend (int *PitchValue*, byte *Channel*)

3.1.3.21 void MIDI_Class::sendPolyPressure (byte *NoteNumber*, byte *Pressure*, byte *Channel*)

Polyphonic AfterTouch (carries the information of pressure of the given key/note)

Definition at line 123 of file MIDI.cpp.

3.1.3.22 void MIDI_Class::sendProgramChange (byte *ProgramNumber*, byte *Channel*)

Send a Program Change message

Definition at line 119 of file MIDI.cpp.

3.1.3.23 void MIDI_Class::sendSysEx (byte *length*, byte * *array*)

Generate and send a System Exclusive frame.

Parameters:

length The size of the array to send

array The byte array containing the data to send

Don't put F0 or F7 (start & stop SysEx codes), but put destination device's Manufacturer's ID and/or Device ID in the buffer array.

Definition at line 131 of file MIDI.cpp.

3.1.3.24 void MIDI_Class::setDeviceID (byte *sysID*)

Set the System Exclusive Device ID (for listening to SysEx messages)

Definition at line 165 of file MIDI.cpp.

3.1.3.25 void MIDI_Class::setFilter (byte *filter*)

Set the filter for thru mirroring

Parameters:

filter a filter mode:

- MIDI_FILTER_FULL Every incoming message is mirrored.
- MIDI_FILTER_CANAL Every message matching the input channel is mirrored.
- MIDI_FILTER_ANTICANAL Every message non-matching the input channel is mirrored.

Definition at line 177 of file MIDI.cpp.

3.1.3.26 void MIDI_Class::setInputChannel (byte *channel*)

Set the value for the input MIDI channel

Parameters:

channel the channel value. Valid values are 1 to 16, MIDI_CHANNEL_OMNI if you want to listen all channels, and MIDI_CHANNEL_OFF to disable MIDI input.

Definition at line 170 of file MIDI.cpp.

3.1.3.27 void MIDI_Class::turnThru (bool *val*)

Setter method: set message mirroring to the specified value.

Definition at line 184 of file MIDI.cpp.

3.1.3.28 void MIDI_Class::turnThruOff ()

Setter method: turn message mirroring off.

Definition at line 182 of file MIDI.cpp.

3.1.3.29 void MIDI_Class::turnThruOn ()

Setter method: turn message mirroring on.

Definition at line 180 of file MIDI.cpp.

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

- /Users/franky/DIY/Arduino/Arduino.app/Contents/Resources/Java/hardware/libraries/MIDI/MIDI.h
- /Users/franky/DIY/Arduino/Arduino.app/Contents/Resources/Java/hardware/libraries/MIDI/MIDI.cpp

3.2 midimsg Struct Reference

`#include <MIDI.h>`

Public Attributes

- byte channel
- byte type
- byte data1
- byte data2
- byte sysex_array [MIDI_SYSEX_ARRAY_SIZE]
- bool OK

3.2.1 Detailed Description

This structure contains decoded data of a MIDI message read from the serial port with read() or thru().

Definition at line 53 of file MIDI.h.

3.2.2 Member Data Documentation

3.2.2.1 byte midimsg::channel

The MIDI channel on which the message was recieved.

Value goes from 1 to 16.

Definition at line 55 of file MIDI.h.

3.2.2.2 byte midimsg::data1

The first data byte.

Value goes from 0 to 127.

If the message is SysEx, this byte contains the array length.

Definition at line 59 of file MIDI.h.

3.2.2.3 byte midimsg::data2

The second data byte. If the message is only 2 bytes long, this one is null.

Value goes from 0 to 127.

Definition at line 61 of file MIDI.h.

3.2.2.4 bool midimsg::OK

This boolean indicates if the message is valid or not. There is no channel consideration here, validity means the message respects the MIDI norm.

Definition at line 65 of file MIDI.h.

3.2.2.5 byte midimsg::sysex_array[MIDI_SYSEX_ARRAY_SIZE]

System Exclusive dedicated byte array.

Array length is stocked in data1.

Definition at line 63 of file MIDI.h.

3.2.2.6 byte midimsg::type

The type of the message (see the define section for types reference)

Definition at line 57 of file MIDI.h.

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

- /Users/franky/DIY/Arduino/Arduino.app/Contents/Resources/Java/hardware/libraries/MIDI/MIDI.h

Chapter 4

File Documentation

4.1 /Users/franky/DIY/Arduino/Arduino.app/Contents/Resources/Java/hardware/ File Reference

```
#include <inttypes.h>
#include <stdlib.h>
#include "WConstants.h"
#include "MIDI.h"
#include "HardwareSerial.h"
```

Defines

- #define UART Serial
- #define Channel_Refused 0

Variables

- MIDI_Class MIDI

4.1.1 Define Documentation

4.1.1.1 #define Channel_Refused 0

Definition at line 27 of file MIDI.cpp.

4.1.1.2 #define UART Serial

to ATmega 644p users: this library uses the serial port #1 for MIDI.

to any other ATmega users: this library uses the serial port #0 for MIDI (if you have more than one UART).

Definition at line 16 of file MIDI.cpp.

4.1.2 Variable Documentation

4.1.2.1 MIDI_Class MIDI

Main instance (the class comes pre-instantiated).

Definition at line 30 of file MIDI.cpp.

4.2 /Users/franky/DIY/Arduino/Arduino.app/Contents/Resources/Java/hardware/ File Reference

```
#include <inttypes.h>
```

Classes

- struct midimsg
- class MIDI_Class

Defines

- #define MIDI_rate 31250
- #define NoteOff 0
- #define NoteOn 1
- #define ATPoly 2
- #define CC 3
- #define PC 4
- #define ATCanal 5
- #define PitchBend 6
- #define SysEx 7
- #define MIDI_FILTER_OFF 0
- #define MIDI_FILTER_FULL 1
- #define MIDI_FILTER_CANAL 2
- #define MIDI_FILTER_ANTICANAL 3
- #define MIDI_CHANNEL_OMNI 0
- #define MIDI_CHANNEL_OFF 17
- #define MIDI_SYSEX_ARRAY_SIZE 256
- #define MANUFACTURER_ID_H 0x7D

Typedefs

- typedef uint8_t byte

Variables

- MIDI_Class MIDI

4.2.1 Define Documentation

4.2.1.1 #define ATCanal 5

Message type AfterTouch Channel

Definition at line 26 of file MIDI.h.

4.2.1.2 #define ATPoly 2

Message type AfterTouch Poly

Definition at line 20 of file MIDI.h.

4.2.1.3 #define CC 3

Message type Control Change

Definition at line 22 of file MIDI.h.

4.2.1.4 #define MANUFACTURER_ID_H 0x7D

Definition at line 47 of file MIDI.h.

4.2.1.5 #define MIDI_CHANNEL_OFF 17

Definition at line 42 of file MIDI.h.

4.2.1.6 #define MIDI_CHANNEL_OMNI 0

Definition at line 41 of file MIDI.h.

4.2.1.7 #define MIDI_FILTER_ANTICANAL 3

Definition at line 39 of file MIDI.h.

4.2.1.8 #define MIDI_FILTER_CANAL 2

Definition at line 38 of file MIDI.h.

4.2.1.9 #define MIDI_FILTER_FULL 1

Definition at line 37 of file MIDI.h.

4.2.1.10 #define MIDI_FILTER_OFF 0

Definition at line 36 of file MIDI.h.

4.2.1.11 #define MIDI_rate 31250

The basic baudrate for MIDI communications.

Definition at line 14 of file MIDI.h.

4.2.1.12 #define MIDI_SYSEX_ARRAY_SIZE 256

Definition at line 45 of file MIDI.h.

4.2.1.13 #define NoteOff 0

Message type Note Off (equivalent to a NoteOn with null velocity)

Definition at line 16 of file MIDI.h.

4.2.1.14 #define NoteOn 1

Message type Note On

Definition at line 18 of file MIDI.h.

4.2.1.15 #define PC 4

Message type Program Change

Definition at line 24 of file MIDI.h.

4.2.1.16 #define PitchBend 6

Message type Pitch Bend

Definition at line 28 of file MIDI.h.

4.2.1.17 #define SysEx 7

Message type System Exclusive

Definition at line 30 of file MIDI.h.

4.2.2 Typedef Documentation

4.2.2.1 typedef uint8_t byte

Type definition for practical use (because "unsigned char" is a bit long to write..)

Definition at line 50 of file MIDI.h.

4.2.3 Variable Documentation

4.2.3.1 MIDI_Class MIDI

Main instance (the class comes pre-instantiated).

Definition at line 30 of file MIDI.cpp.

Index