

Arduino Serial Messaging Library

Generated by Doxygen 1.8.1.2

Sat Nov 2 2013 18:22:41

Contents

1	Todo List	1
2	Class Index	1
2.1	Class List	1
3	File Index	1
3.1	File List	2
4	Class Documentation	2
4.1	SerialMessage::message Struct Reference	2
4.1.1	Detailed Description	2
4.1.2	Member Data Documentation	2
4.2	SerialMessage Class Reference	3
4.2.1	Detailed Description	6
4.2.2	Constructor & Destructor Documentation	6
4.2.3	Member Function Documentation	6
4.2.4	Member Data Documentation	16
5	File Documentation	17
5.1	defines.h File Reference	17
5.1.1	Macro Definition Documentation	18
5.2	SerialMessage.cpp File Reference	19
5.2.1	Macro Definition Documentation	20
5.3	SerialMessage.h File Reference	20

1 Todo List

Member `SerialMessage::parseSerialData ()`

: error handling: illegal payloads is now handled by char position (G2 -> 02, 2G -> 20, GG -> 00), requires hexdec reimplementatation (when HEX_DEC_ERROR=0)

2 Class Index

2.1 Class List

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

<code>SerialMessage::message</code>	2
<code>SerialMessage</code>	3

3 File Index

3.1 File List

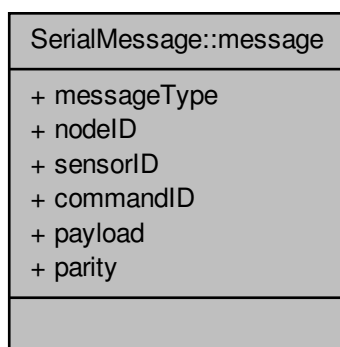
Here is a list of all files with brief descriptions:

defines.h	17
SerialMessage.cpp	19
SerialMessage.h	20

4 Class Documentation

4.1 SerialMessage::message Struct Reference

Collaboration diagram for SerialMessage::message:



Public Attributes

- `uint8_t` [messageType](#)
- `uint8_t` [nodeID](#)
- `uint8_t` [sensorID](#)
- `uint8_t` [commandID](#)
- `uint8_t` [payload](#)
- `uint8_t` [parity](#)

4.1.1 Detailed Description

Definition at line 65 of file `SerialMessage.h`.

4.1.2 Member Data Documentation

4.1.2.1 `uint8_t` `SerialMessage::message::commandID`

Definition at line 70 of file `SerialMessage.h`.

Referenced by `SerialMessage::getCommandID()`, `SerialMessage::parseSerialData()`, `SerialMessage::printInfo()`, and `SerialMessage::SerialMessage()`.

4.1.2.2 `uint8_t SerialMessage::message::messageType`

Definition at line 67 of file `SerialMessage.h`.

Referenced by `SerialMessage::parseSerialData()`, `SerialMessage::printInfo()`, and `SerialMessage::SerialMessage()`.

4.1.2.3 `uint8_t SerialMessage::message::nodeID`

Definition at line 68 of file `SerialMessage.h`.

Referenced by `SerialMessage::parseSerialData()`, `SerialMessage::printInfo()`, and `SerialMessage::SerialMessage()`.

4.1.2.4 `uint8_t SerialMessage::message::parity`

Definition at line 72 of file `SerialMessage.h`.

Referenced by `SerialMessage::checkParity()`, `SerialMessage::parseSerialData()`, `SerialMessage::printInfo()`, and `SerialMessage::SerialMessage()`.

4.1.2.5 `uint8_t SerialMessage::message::payload`

Definition at line 71 of file `SerialMessage.h`.

Referenced by `SerialMessage::getPayload()`, `SerialMessage::parseSerialData()`, `SerialMessage::printInfo()`, and `SerialMessage::SerialMessage()`.

4.1.2.6 `uint8_t SerialMessage::message::sensorID`

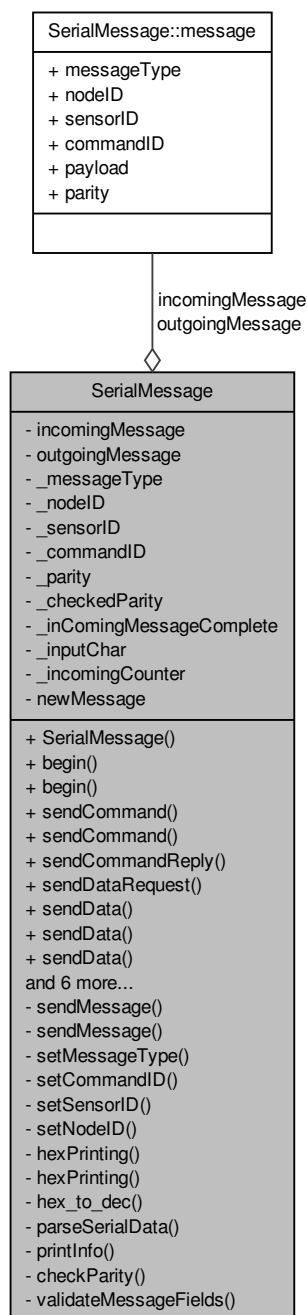
Definition at line 69 of file `SerialMessage.h`.

Referenced by `SerialMessage::parseSerialData()`, `SerialMessage::printInfo()`, and `SerialMessage::SerialMessage()`.

4.2 SerialMessage Class Reference

```
#include <SerialMessage.h>
```

Collaboration diagram for SerialMessage:



Classes

- struct [message](#)

Public Member Functions

- [SerialMessage](#) ()

[SerialMessage.cpp](#) - Library for sending sensor data messages over UART. For more information: variable declaration, changelog,... see [SerialMessage.h](#)

- void [begin](#) ()
Begin using default settings:
- void [begin](#) (long speed, uint8_t nodeID)
Begin using custom settings
- void [sendCommand](#) (uint8_t commandID, uint8_t payload)
Send a single command
- void [sendCommand](#) (uint8_t payload)
Send a single command, reuses commandID from previous messages
- void [sendCommandReply](#) (uint8_t commandID, uint8_t payload)
Send a reply to a command
- void [sendDataRequest](#) (uint8_t sensorID, uint8_t payload)
Request a single data value
- void [sendData](#) (uint8_t sensorID, uint8_t payload)
Send a single data value
- void [sendData](#) (uint8_t sensorID, int16_t payload)
Send a single data value
- void [sendData](#) (uint8_t payload)
Send a single 8-bit data value (Arduino 'byte' type), reuses sensorID from previous messages
- void [sendData](#) (int16_t payload)
Send a single 16-bit data value (Arduino 'int' type), reuses sensorID from previous messages
- void [sendDataArrayRequest](#) (uint8_t arrayID, uint8_t length)
Request an array of data values
- void [sendDataArray](#) (uint8_t *dataArray, uint8_t length)
Send multiple data samples in one message by passing an array and its length
- boolean [readSerialData](#) ()
Set readSerialData
- uint8_t [getCommandID](#) ()
Get commandID
- uint8_t [getPayload](#) ()
Get getPayload

Private Member Functions

- void [sendMessage](#) (uint8_t &payload)
Send out the actual 8-bit data message (called from other 'send' functions)
- void [sendMessage](#) (int16_t &payload)
Send out the actual 16-bit data message (called from other 'send' functions)
- void [setMessageType](#) (uint8_t type)
Set message type
- void [setCommandID](#) (uint8_t &commandID)
Set commandID
- void [setSensorID](#) (uint8_t &sensorID)
Set sensorID
- void [setNodeID](#) (uint8_t &nodeID)
Set nodeID
- void [hexPrinting](#) (uint8_t &data)
HexPrinting: helper function to print data with a constant field width (2 hex values)
- void [hexPrinting](#) (int16_t &data)
HexPrinting: helper function to print data with a constant field width (2 hex values)

- uint8_t [hex_to_dec](#) (uint8_t in)
- boolean [parseSerialData](#) ()
Set parseSerialData
- void [printlnInfo](#) ()
printlnInfo:
- boolean [checkParity](#) ()
Check parity
- boolean [validateMessageFields](#) ()
Check if all the field in the message have acceptable value : implement this

Private Attributes

- struct [SerialMessage::message](#) incomingMessage
- struct [SerialMessage::message](#) outgoingMessage
- uint8_t [_messageType](#)
- uint8_t [_nodeID](#)
- uint8_t [_sensorID](#)
- uint8_t [_commandID](#)
- uint8_t [_parity](#)
- uint8_t [_checkedParity](#)
- boolean [_inComingMessageComplete](#)
- char [_inputChar](#) [20]
- uint8_t [_incomingCounter](#)
- boolean [newMessage](#)

4.2.1 Detailed Description

Definition at line 38 of file [SerialMessage.h](#).

4.2.2 Constructor & Destructor Documentation

4.2.2.1 SerialMessage::SerialMessage ()

[SerialMessage.cpp](#) - Library for sending sensor data messages over UART. For more information: variable declaration, changelog,... see [SerialMessage.h](#)

Constructor

Definition at line 31 of file [SerialMessage.cpp](#).

4.2.3 Member Function Documentation

4.2.3.1 void SerialMessage::begin ()

Begin using default settings:

- speed: 115200 baud
- nodeID: 0

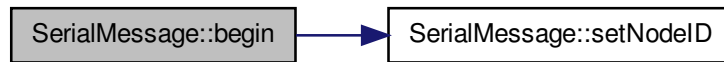
Definition at line 49 of file [SerialMessage.cpp](#).

4.2.3.2 void SerialMessage::begin (long *speed*, uint8_t *nodeID*)

Begin using custom settings

Definition at line 57 of file SerialMessage.cpp.

Here is the call graph for this function:



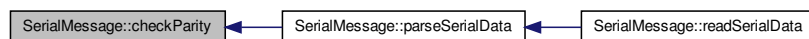
4.2.3.3 boolean SerialMessage::checkParity () [private]

Check parity

Definition at line 396 of file SerialMessage.cpp.

Referenced by `parseSerialData()`.

Here is the caller graph for this function:



4.2.3.4 uint8_t SerialMessage::getCommandID ()

Get commandID

Definition at line 416 of file SerialMessage.cpp.

4.2.3.5 uint8_t SerialMessage::getPayload ()

Get getPayload

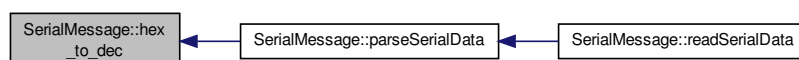
Definition at line 424 of file SerialMessage.cpp.

4.2.3.6 uint8_t SerialMessage::hex_to_dec (uint8_t *in*) [private]

Definition at line 362 of file SerialMessage.cpp.

Referenced by `parseSerialData()`.

Here is the caller graph for this function:



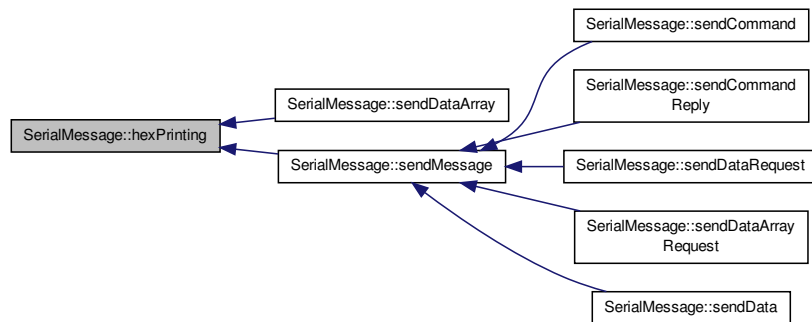
4.2.3.7 void SerialMessage::hexPrinting (uint8_t & data) [private]

HexPrinting: helper function to print data with a constant field width (2 hex values)

Definition at line 243 of file SerialMessage.cpp.

Referenced by sendDataArray(), and sendMessage().

Here is the caller graph for this function:



4.2.3.8 void SerialMessage::hexPrinting (int16_t & data) [private]

HexPrinting: helper function to print data with a constant field width (2 hex values)

Definition at line 254 of file SerialMessage.cpp.

4.2.3.9 boolean SerialMessage::parseSerialData () [private]

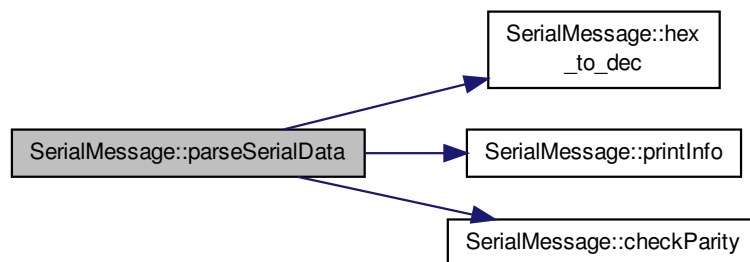
Set parseSerialData

Todo : error handling: illegal payloads is now handled by char position (G2 -> 02, 2G -> 20, GG -> 00), requires hexto dec reimplementation (when HEX_DEC_ERROR=0)

Definition at line 311 of file SerialMessage.cpp.

Referenced by readSerialData().

Here is the call graph for this function:



Here is the caller graph for this function:



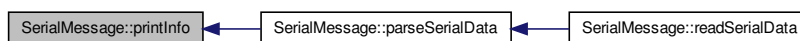
4.2.3.10 void SerialMessage::printInfo () [private]

printInfo:

Definition at line 373 of file SerialMessage.cpp.

Referenced by parseSerialData().

Here is the caller graph for this function:

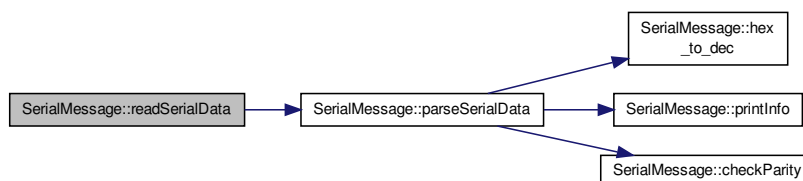


4.2.3.11 boolean SerialMessage::readSerialData ()

Set readSerialData

Definition at line 287 of file SerialMessage.cpp.

Here is the call graph for this function:

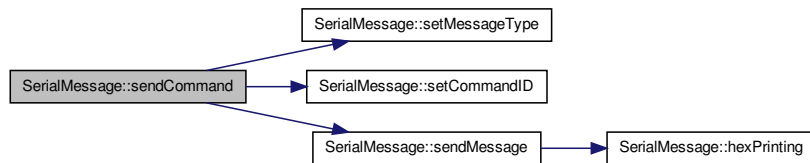


4.2.3.12 void SerialMessage::sendCommand (uint8_t commandID, uint8_t payload)

Send a single command

Definition at line 66 of file SerialMessage.cpp.

Here is the call graph for this function:



4.2.3.13 void SerialMessage::sendCommand (uint8_t payload)

Send a single command, reuses commandID from previous messages

Definition at line 76 of file SerialMessage.cpp.

Here is the call graph for this function:

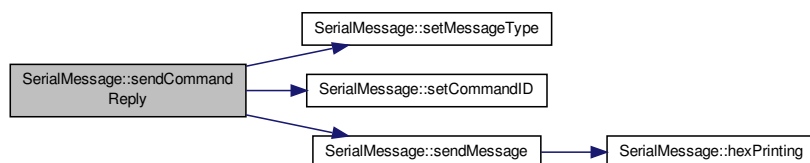


4.2.3.14 void SerialMessage::sendCommandReply (uint8_t commandID, uint8_t payload)

Send a reply to a command

Definition at line 85 of file SerialMessage.cpp.

Here is the call graph for this function:

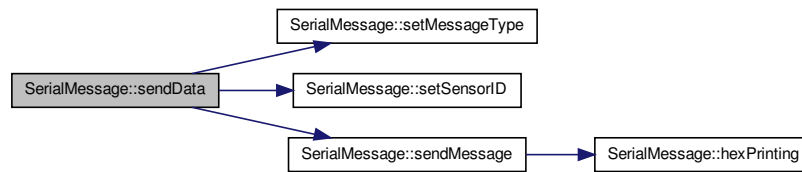


4.2.3.15 void SerialMessage::sendData (uint8_t sensorID, uint8_t payload)

Send a single data value

Definition at line 115 of file SerialMessage.cpp.

Here is the call graph for this function:

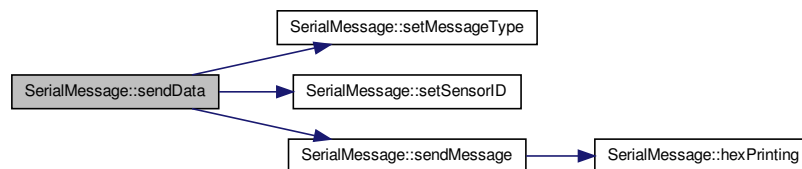


4.2.3.16 void SerialMessage::sendData (uint8_t *sensorID*, int16_t *payload*)

Send a single data value

Definition at line 125 of file `SerialMessage.cpp`.

Here is the call graph for this function:



4.2.3.17 void SerialMessage::sendData (uint8_t *payload*)

Send a single 8-bit data value (Arduino 'byte' type), reuses sensorID from previous messages

Definition at line 135 of file `SerialMessage.cpp`.

Here is the call graph for this function:

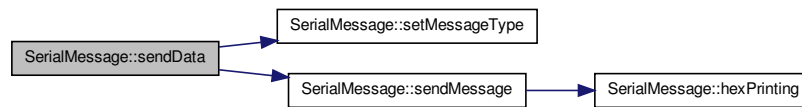


4.2.3.18 void SerialMessage::sendData (int16_t *payload*)

Send a single 16-bit data value (Arduino 'int' type), reuses sensorID from previous messages

Definition at line 144 of file `SerialMessage.cpp`.

Here is the call graph for this function:

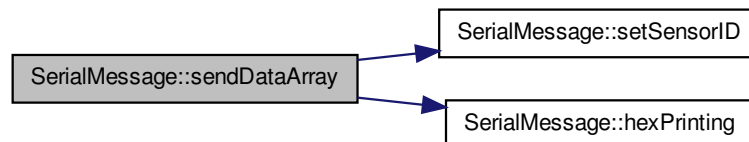


4.2.3.19 void SerialMessage::sendDataArray (uint8_t * dataArray, uint8_t length)

Send multiple data samples in one message by passing an array and its length

Definition at line 203 of file `SerialMessage.cpp`.

Here is the call graph for this function:

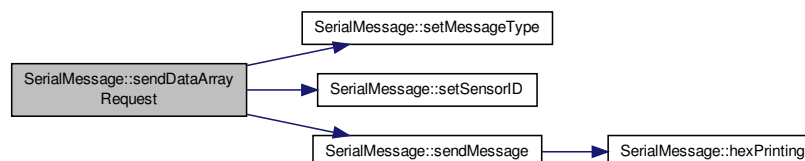


4.2.3.20 void SerialMessage::sendDataArrayRequest (uint8_t arrayID, uint8_t payload)

Request an array of data values

Definition at line 105 of file `SerialMessage.cpp`.

Here is the call graph for this function:

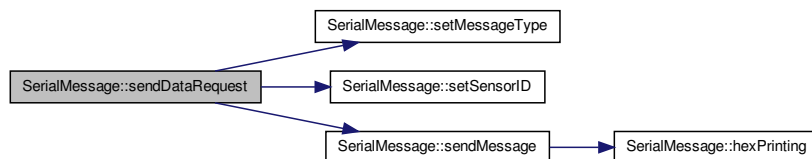


4.2.3.21 void SerialMessage::sendDataRequest (uint8_t sensorID, uint8_t payload)

Request a single data value

Definition at line 95 of file `SerialMessage.cpp`.

Here is the call graph for this function:



4.2.3.22 void SerialMessage::sendMessage (uint8_t & payload) [private]

Send out the actual 8-bit data message (called from other 'send' functions)

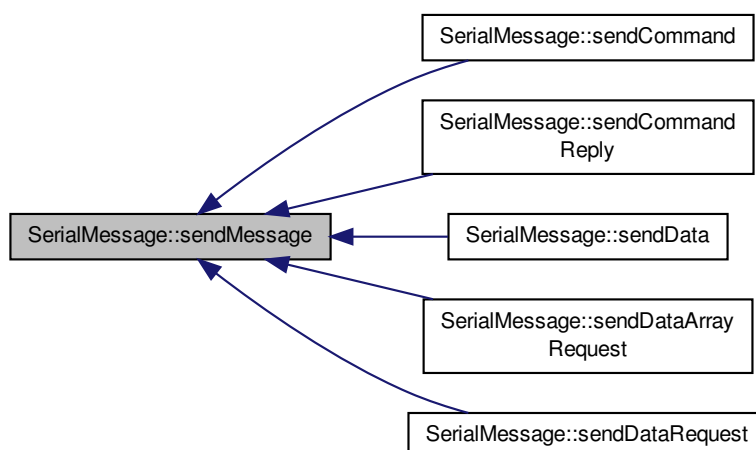
Definition at line 153 of file SerialMessage.cpp.

Referenced by `sendCommand()`, `sendCommandReply()`, `sendData()`, `sendDataArrayRequest()`, and `sendDataRequest()`.

Here is the call graph for this function:



Here is the caller graph for this function:



4.2.3.23 void SerialMessage::sendMessage (int16_t & payload) [private]

Send out the actual 16-bit data message (called from other 'send' functions)

Definition at line 178 of file SerialMessage.cpp.

Here is the call graph for this function:

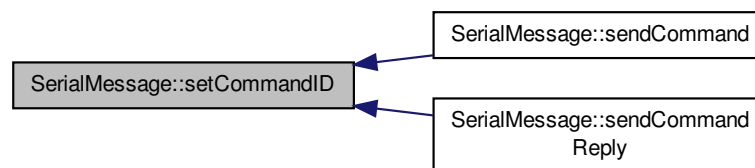
**4.2.3.24 void SerialMessage::setCommandID (uint8_t & commandID) [private]**

Set commandID

Definition at line 227 of file SerialMessage.cpp.

Referenced by `sendCommand()`, and `sendCommandReply()`.

Here is the caller graph for this function:

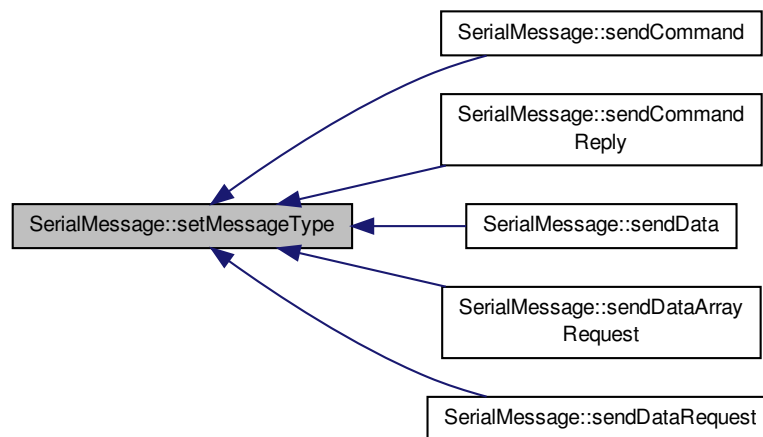
**4.2.3.25 void SerialMessage::setMessageType (uint8_t type) [private]**

Set message type

Definition at line 235 of file SerialMessage.cpp.

Referenced by `sendCommand()`, `sendCommandReply()`, `sendData()`, `sendDataArrayRequest()`, and `sendDataRequest()`.

Here is the caller graph for this function:



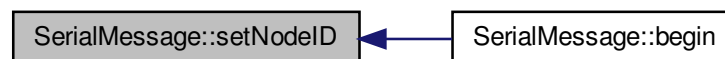
4.2.3.26 void SerialMessage::setNodeID (uint8_t & nodeID) [private]

Set nodeID

Definition at line 271 of file `SerialMessage.cpp`.

Referenced by `begin()`.

Here is the caller graph for this function:



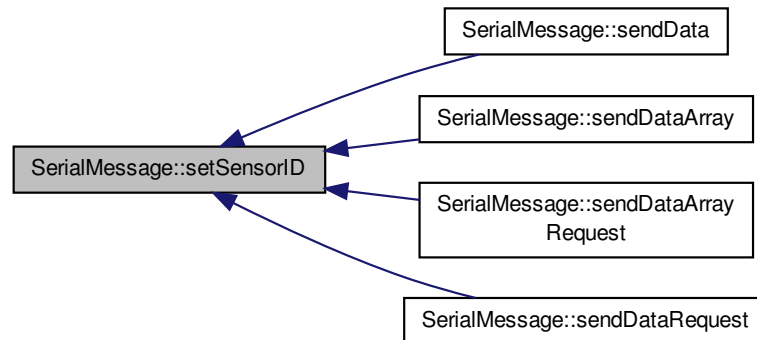
4.2.3.27 void SerialMessage::setSensorID (uint8_t & sensorID) [private]

Set sensorID

Definition at line 279 of file `SerialMessage.cpp`.

Referenced by `sendData()`, `sendDataArray()`, `sendDataArrayRequest()`, and `sendDataRequest()`.

Here is the caller graph for this function:



4.2.3.28 `boolean SerialMessage::validateMessageFields () [private]`

Check if all the field in the message have acceptable value : implement this

Definition at line 347 of file `SerialMessage.cpp`.

4.2.4 Member Data Documentation

4.2.4.1 `uint8_t SerialMessage::_checkedParity [private]`

Definition at line 80 of file `SerialMessage.h`.

Referenced by `checkParity()`, `parseSerialData()`, and `printInfo()`.

4.2.4.2 `uint8_t SerialMessage::_commandID [private]`

Definition at line 78 of file `SerialMessage.h`.

Referenced by `sendMessage()`, and `setCommandID()`.

4.2.4.3 `uint8_t SerialMessage::_incomingCounter [private]`

Definition at line 84 of file `SerialMessage.h`.

Referenced by `readSerialData()`, and `SerialMessage()`.

4.2.4.4 `boolean SerialMessage::_inComingMessageComplete [private]`

Definition at line 82 of file `SerialMessage.h`.

Referenced by `SerialMessage()`.

4.2.4.5 `char SerialMessage::_inputChar[20] [private]`

Definition at line 83 of file `SerialMessage.h`.

Referenced by `parseSerialData()`, and `readSerialData()`.

4.2.4.6 `uint8_t SerialMessage::_messageType [private]`

Definition at line 75 of file `SerialMessage.h`.

Referenced by `sendDataArray()`, `sendMessage()`, and `setMessageType()`.

4.2.4.7 `uint8_t SerialMessage::_nodeID` [private]

Definition at line 76 of file `SerialMessage.h`.

Referenced by `sendDataArray()`, `sendMessage()`, and `setNodeID()`.

4.2.4.8 `uint8_t SerialMessage::_parity` [private]

Definition at line 79 of file `SerialMessage.h`.

Referenced by `sendDataArray()`, and `sendMessage()`.

4.2.4.9 `uint8_t SerialMessage::_sensorID` [private]

Definition at line 77 of file `SerialMessage.h`.

Referenced by `sendDataArray()`, `sendMessage()`, and `setSensorID()`.

4.2.4.10 `struct SerialMessage::message SerialMessage::incomingMessage` [private]

Referenced by `checkParity()`, `getCommandID()`, `getPayload()`, `parseSerialData()`, `printInfo()`, and `SerialMessage()`.

4.2.4.11 `boolean SerialMessage::newMessage` [private]

Definition at line 98 of file `SerialMessage.h`.

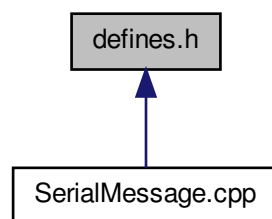
Referenced by `parseSerialData()`, `readSerialData()`, and `SerialMessage()`.

4.2.4.12 `struct SerialMessage::message SerialMessage::outgoingMessage` [private]

5 File Documentation

5.1 defines.h File Reference

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



Macros

- `#define SERIAL_ASCII`
- `#define COMMAND 0x01`
- `#define COMMAND_REPLY 0x02`
- `#define DATA_REQUEST 0x11`
- `#define DATA_BYTE 0x12`

- #define `DATA_INT` 0x13
- #define `DATA_ARRAY_REQUEST` 0x21
- #define `DATA_ARRAY` 0x22
- #define `TEMPERATURE` 0x10

Sensor Types:

- #define `HUMIDITY` 0x11
- #define `DISTANCE` 0x30
- #define `STOP_MOTOR_A` 0x10

Command IDs.

- #define `START_MOTOR_A` 0x11
- #define `SET_SPEED_MOTOR_A` 0x12
- #define `BRAKE_MOTOR_A` 0x13
- #define `STOP_MOTOR_B` 0x15
- #define `START_MOTOR_B` 0x16
- #define `SET_SPEED_MOTOR_B` 0x17
- #define `BRAKE_MOTOR_B` 0x18

5.1.1 Macro Definition Documentation

5.1.1.1 #define `BRAKE_MOTOR_A` 0x13

Definition at line 28 of file defines.h.

5.1.1.2 #define `BRAKE_MOTOR_B` 0x18

Definition at line 33 of file defines.h.

5.1.1.3 #define `COMMAND` 0x01

Definition at line 6 of file defines.h.

Referenced by `SerialMessage::parseSerialData()`, `SerialMessage::sendCommand()`, and `SerialMessage::sendMessage()`.

5.1.1.4 #define `COMMAND_REPLY` 0x02

Definition at line 7 of file defines.h.

Referenced by `SerialMessage::parseSerialData()`, `SerialMessage::sendCommandReply()`, and `SerialMessage::sendMessage()`.

5.1.1.5 #define `DATA_ARRAY` 0x22

Definition at line 14 of file defines.h.

5.1.1.6 #define `DATA_ARRAY_REQUEST` 0x21

Definition at line 13 of file defines.h.

Referenced by `SerialMessage::sendDataArrayRequest()`, and `SerialMessage::sendMessage()`.

5.1.1.7 #define `DATA_BYTE` 0x12

Definition at line 10 of file defines.h.

Referenced by `SerialMessage::parseSerialData()`, `SerialMessage::sendData()`, and `SerialMessage::sendMessage()`.

5.1.1.8 #define DATA_INT 0x13

Definition at line 11 of file defines.h.

Referenced by SerialMessage::parseSerialData(), SerialMessage::sendData(), and SerialMessage::sendMessage().

5.1.1.9 #define DATA_REQUEST 0x11

Definition at line 9 of file defines.h.

Referenced by SerialMessage::parseSerialData(), SerialMessage::sendDataRequest(), and SerialMessage::sendMessage().

5.1.1.10 #define DISTANCE 0x30

Definition at line 21 of file defines.h.

5.1.1.11 #define HUMIDITY 0x11

Definition at line 19 of file defines.h.

5.1.1.12 #define SERIAL_ASCII

Definition at line 3 of file defines.h.

5.1.1.13 #define SET_SPEED_MOTOR_A 0x12

Definition at line 27 of file defines.h.

5.1.1.14 #define SET_SPEED_MOTOR_B 0x17

Definition at line 32 of file defines.h.

5.1.1.15 #define START_MOTOR_A 0x11

Definition at line 26 of file defines.h.

5.1.1.16 #define START_MOTOR_B 0x16

Definition at line 31 of file defines.h.

5.1.1.17 #define STOP_MOTOR_A 0x10

Command IDs.

Definition at line 25 of file defines.h.

5.1.1.18 #define STOP_MOTOR_B 0x15

Definition at line 30 of file defines.h.

5.1.1.19 #define TEMPERATURE 0x10

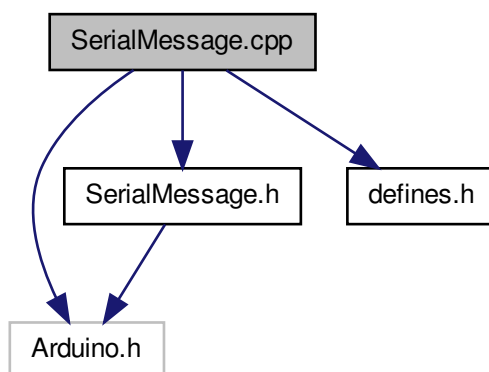
Sensor Types:

Definition at line 18 of file defines.h.

5.2 SerialMessage.cpp File Reference

```
#include <Arduino.h>
#include "SerialMessage.h"
#include "defines.h"
```

Include dependency graph for SerialMessage.cpp:



Macros

- `#define HEX_DEC_ERROR 0`
Convert HEX to Decimal

5.2.1 Macro Definition Documentation

5.2.1.1 `#define HEX_DEC_ERROR 0`

Convert HEX to Decimal

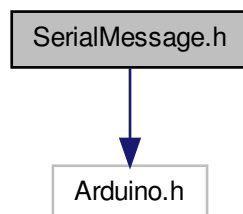
Definition at line 361 of file SerialMessage.cpp.

Referenced by SerialMessage::hex_to_dec().

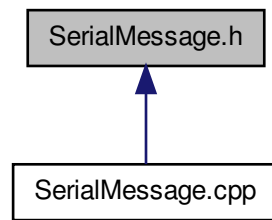
5.3 SerialMessage.h File Reference

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

Include dependency graph for SerialMessage.h:



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



Classes

- class [SerialMessage](#)
- struct [SerialMessage::message](#)

Index

- _checkedParity
 - SerialMessage, [16](#)
 - _commandID
 - SerialMessage, [16](#)
 - _inComingMessageComplete
 - SerialMessage, [16](#)
 - _incomingCounter
 - SerialMessage, [16](#)
 - _inputChar
 - SerialMessage, [16](#)
 - _messageType
 - SerialMessage, [16](#)
 - _nodeID
 - SerialMessage, [17](#)
 - _parity
 - SerialMessage, [17](#)
 - _sensorID
 - SerialMessage, [17](#)
- BRAKE_MOTOR_A
 - defines.h, [18](#)
- BRAKE_MOTOR_B
 - defines.h, [18](#)
- begin
 - SerialMessage, [6](#)
- COMMAND
 - defines.h, [18](#)
- COMMAND_REPLY
 - defines.h, [18](#)
- checkParity
 - SerialMessage, [7](#)
- commandID
 - SerialMessage::message, [2](#)
- DATA_ARRAY
 - defines.h, [18](#)
- DATA_ARRAY_REQUEST
 - defines.h, [18](#)
- DATA_BYTE
 - defines.h, [18](#)
- DATA_INT
 - defines.h, [18](#)
- DATA_REQUEST
 - defines.h, [19](#)
- DISTANCE
 - defines.h, [19](#)
- defines.h, [17](#)
 - BRAKE_MOTOR_A, [18](#)
 - BRAKE_MOTOR_B, [18](#)
 - COMMAND, [18](#)
 - COMMAND_REPLY, [18](#)
 - DATA_ARRAY, [18](#)
 - DATA_ARRAY_REQUEST, [18](#)
 - DATA_BYTE, [18](#)
 - DATA_INT, [18](#)
 - DATA_REQUEST, [19](#)
 - DISTANCE, [19](#)
 - HUMIDITY, [19](#)
 - SERIAL_ASCII, [19](#)
 - SET_SPEED_MOTOR_A, [19](#)
 - SET_SPEED_MOTOR_B, [19](#)
 - START_MOTOR_A, [19](#)
 - START_MOTOR_B, [19](#)
 - STOP_MOTOR_A, [19](#)
 - STOP_MOTOR_B, [19](#)
 - TEMPERATURE, [19](#)
- getCommandID
 - SerialMessage, [7](#)
- getPayload
 - SerialMessage, [7](#)
- HEX_DEC_ERROR
 - SerialMessage.cpp, [20](#)
- HUMIDITY
 - defines.h, [19](#)
- hex_to_dec
 - SerialMessage, [7](#)
- hexPrinting
 - SerialMessage, [7](#), [8](#)
- incomingMessage
 - SerialMessage, [17](#)
- messageType
 - SerialMessage::message, [2](#)
- newMessage
 - SerialMessage, [17](#)
- nodeID
 - SerialMessage::message, [2](#)
- outgoingMessage
 - SerialMessage, [17](#)
- parity
 - SerialMessage::message, [2](#)
- parseSerialData
 - SerialMessage, [8](#)
- payload
 - SerialMessage::message, [3](#)
- printInfo
 - SerialMessage, [9](#)
- readSerialData
 - SerialMessage, [9](#)
- SERIAL_ASCII
 - defines.h, [19](#)
- SET_SPEED_MOTOR_A
 - defines.h, [19](#)
- SET_SPEED_MOTOR_B

- defines.h, [19](#)
- START_MOTOR_A
 - defines.h, [19](#)
- START_MOTOR_B
 - defines.h, [19](#)
- STOP_MOTOR_A
 - defines.h, [19](#)
- STOP_MOTOR_B
 - defines.h, [19](#)
- sendCommand
 - SerialMessage, [9, 10](#)
- sendCommandReply
 - SerialMessage, [10](#)
- sendData
 - SerialMessage, [10, 11](#)
- sendDataArray
 - SerialMessage, [12](#)
- sendDataArrayRequest
 - SerialMessage, [12](#)
- sendDataRequest
 - SerialMessage, [12](#)
- sendMessage
 - SerialMessage, [13](#)
- sensorID
 - SerialMessage::message, [3](#)
- SerialMessage, [3](#)
 - _checkedParity, [16](#)
 - _commandID, [16](#)
 - _incomingMessageComplete, [16](#)
 - _incomingCounter, [16](#)
 - _inputChar, [16](#)
 - _messageType, [16](#)
 - _nodeID, [17](#)
 - _parity, [17](#)
 - _sensorID, [17](#)
- begin, [6](#)
- checkParity, [7](#)
- getCommandID, [7](#)
- getPayload, [7](#)
- hex_to_dec, [7](#)
- hexPrinting, [7, 8](#)
- incomingMessage, [17](#)
- newMessage, [17](#)
- outgoingMessage, [17](#)
- parseSerialData, [8](#)
- printInfo, [9](#)
- readSerialData, [9](#)
- sendCommand, [9, 10](#)
- sendCommandReply, [10](#)
- sendData, [10, 11](#)
- sendDataArray, [12](#)
- sendDataArrayRequest, [12](#)
- sendDataRequest, [12](#)
- sendMessage, [13](#)
- SerialMessage, [6](#)
- SerialMessage, [6](#)
- setCommandID, [14](#)
- setMessageType, [14](#)
- setNodeID, [15](#)
- setSensorID, [15](#)
- validateMessageFields, [16](#)
- SerialMessage.cpp, [19](#)
 - HEX_DEC_ERROR, [20](#)
- SerialMessage.h, [20](#)
- SerialMessage::message, [2](#)
 - commandID, [2](#)
 - messageType, [2](#)
 - nodeID, [2](#)
 - parity, [2](#)
 - payload, [3](#)
 - sensorID, [3](#)
- setCommandID
 - SerialMessage, [14](#)
- setMessageType
 - SerialMessage, [14](#)
- setNodeID
 - SerialMessage, [15](#)
- setSensorID
 - SerialMessage, [15](#)
- TEMPERATURE
 - defines.h, [19](#)
- validateMessageFields
 - SerialMessage, [16](#)