

## Arduino Serial Communication Library

Generated by Doxygen 1.8.1.2

Tue Dec 4 2012 14:43:24

## Contents

<b>1</b>	<b><a href="#">Class Index</a></b>	<b>1</b>
1.1	<a href="#">Class List</a>	1
<b>2</b>	<b><a href="#">File Index</a></b>	<b>1</b>
2.1	<a href="#">File List</a>	1
<b>3</b>	<b><a href="#">Class Documentation</a></b>	<b>2</b>
3.1	<a href="#">SerialPacket::packet Struct Reference</a>	2
3.1.1	<a href="#">Detailed Description</a>	2
3.1.2	<a href="#">Member Data Documentation</a>	2
3.2	<a href="#">SerialPacket Class Reference</a>	3
3.2.1	<a href="#">Detailed Description</a>	6
3.2.2	<a href="#">Constructor &amp; Destructor Documentation</a>	6
3.2.3	<a href="#">Member Function Documentation</a>	6
3.2.4	<a href="#">Member Data Documentation</a>	15
<b>4</b>	<b><a href="#">File Documentation</a></b>	<b>15</b>
4.1	<a href="#">defines.h File Reference</a>	16
4.1.1	<a href="#">Macro Definition Documentation</a>	16
4.2	<a href="#">SerialPacket.cpp File Reference</a>	17
4.2.1	<a href="#">Macro Definition Documentation</a>	18
4.3	<a href="#">SerialPacket.h File Reference</a>	18

## 1 Class Index

### 1.1 Class List

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

<a href="#">SerialPacket::packet</a>	<b>2</b>
<a href="#">SerialPacket</a>	<b>3</b>

## 2 File Index

### 2.1 File List

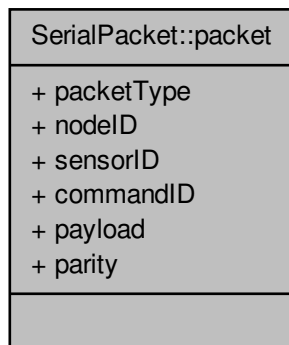
Here is a list of all files with brief descriptions:

<a href="#">defines.h</a>	<b>16</b>
<a href="#">SerialPacket.cpp</a>	<b>17</b>
<a href="#">SerialPacket.h</a>	<b>18</b>

## 3 Class Documentation

### 3.1 SerialPacket::packet Struct Reference

Collaboration diagram for SerialPacket::packet:



#### Public Attributes

- uint8\_t [packetType](#)
- uint8\_t [nodeID](#)
- uint8\_t [sensorID](#)
- uint8\_t [commandID](#)
- uint8\_t [payload](#)
- uint8\_t [parity](#)

#### 3.1.1 Detailed Description

Definition at line 65 of file SerialPacket.h.

#### 3.1.2 Member Data Documentation

##### 3.1.2.1 uint8\_t SerialPacket::packet::commandID

Definition at line 70 of file SerialPacket.h.

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

##### 3.1.2.2 uint8\_t SerialPacket::packet::nodeID

Definition at line 68 of file SerialPacket.h.

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

##### 3.1.2.3 uint8\_t SerialPacket::packet::packetType

Definition at line 67 of file SerialPacket.h.

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

#### 3.1.2.4 `uint8_t SerialPacket::packet::parity`

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

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

#### 3.1.2.5 `uint8_t SerialPacket::packet::payload`

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

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

#### 3.1.2.6 `uint8_t SerialPacket::packet::sensorID`

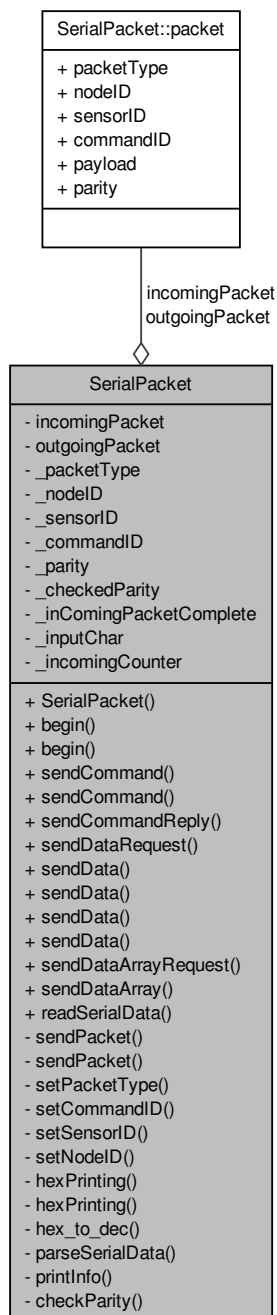
Definition at line 69 of file `SerialPacket.h`.

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

## 3.2 SerialPacket Class Reference

```
#include <SerialPacket.h>
```

Collaboration diagram for SerialPacket:



#### Classes

- struct [packet](#)

#### Public Member Functions

- [SerialPacket](#) ()

[SerialPacket.cpp](#) - Library for sending sensor data packets over UART. For more information: variable declaration, changelog,... see [SerialPacket.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 packets*
- 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 packets*
- void [sendData](#) (int16\_t payload)  
*Send a single 16-bit data value (Arduino 'int' type), reuses sensorID from previous packets*
- 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 packet by passing an array and its length*
- void [readSerialData](#) ()  
*Set readSerialData*

#### Private Member Functions

- void [sendPacket](#) (uint8\_t &payload)  
*Send out the actual 8-bit data packet (called from other 'send' functions)*
- void [sendPacket](#) (int16\_t &payload)  
*Send out the actual 16-bit data packet (called from other 'send' functions)*
- void [setPacketType](#) (uint8\_t type)  
*Set packet 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)
- void [parseSerialData](#) ()  
*Set parseSerialData*
- void [printInfo](#) ()  
*printInfo:*
- boolean [checkParity](#) ()  
*Convert HEX to DecimalT12N00I00P1CQ0E*

## Private Attributes

- struct [SerialPacket::packet incomingPacket](#)
- struct [SerialPacket::packet outgoingPacket](#)
- uint8\_t [\\_packetType](#)
- uint8\_t [\\_nodeID](#)
- uint8\_t [\\_sensorID](#)
- uint8\_t [\\_commandID](#)
- uint8\_t [\\_parity](#)
- uint8\_t [\\_checkedParity](#)
- boolean [\\_inComingPacketComplete](#)
- char [\\_inputChar](#) [20]
- uint8\_t [\\_incomingCounter](#)

## 3.2.1 Detailed Description

Definition at line 40 of file [SerialPacket.h](#).

## 3.2.2 Constructor &amp; Destructor Documentation

## 3.2.2.1 SerialPacket::SerialPacket ( )

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

Constructor

Definition at line 42 of file [SerialPacket.cpp](#).

## 3.2.3 Member Function Documentation

## 3.2.3.1 void SerialPacket::begin ( )

Begin using default settings:

- speed: 115200 baud
- nodeID: 0

Definition at line 55 of file [SerialPacket.cpp](#).

3.2.3.2 void SerialPacket::begin ( long *speed*, uint8\_t *nodeID* )

Begin using custom settings

Definition at line 63 of file [SerialPacket.cpp](#).

Here is the call graph for this function:



**3.2.3.3** `boolean SerialPacket::checkParity ( ) [private]`

Convert HEX to DecimalT12N00I00P1CQ0E

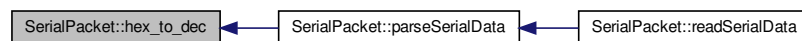
Definition at line 384 of file SerialPacket.cpp.

**3.2.3.4** `uint8_t SerialPacket::hex_to_dec ( uint8_t in ) [private]`

Definition at line 344 of file SerialPacket.cpp.

Referenced by `parseSerialData()`.

Here is the caller graph for this function:

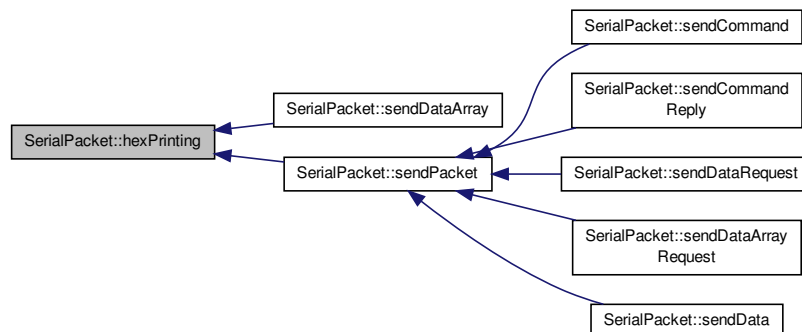
**3.2.3.5** `void SerialPacket::hexPrinting ( uint8_t & data ) [private]`

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

Definition at line 254 of file SerialPacket.cpp.

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

Here is the caller graph for this function:

**3.2.3.6** `void SerialPacket::hexPrinting ( int16_t & data ) [private]`

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

Definition at line 266 of file SerialPacket.cpp.

**3.2.3.7** `void SerialPacket::parseSerialData ( ) [private]`

Set `parseSerialData`

Definition at line 322 of file SerialPacket.cpp.

Referenced by `readSerialData()`.



Here is the call graph for this function:



Here is the caller graph for this function:



#### 3.2.3.8 void SerialPacket::printInfo ( ) [private]

printInfo:

Definition at line 356 of file SerialPacket.cpp.

Referenced by readSerialData().

Here is the caller graph for this function:

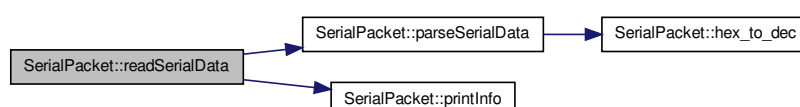


#### 3.2.3.9 void SerialPacket::readSerialData ( )

Set readSerialData

Definition at line 302 of file SerialPacket.cpp.

Here is the call graph for this function:

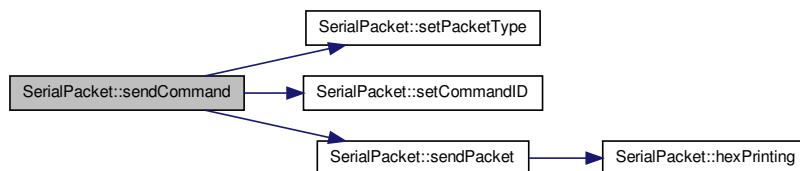


### 3.2.3.10 void SerialPacket::sendCommand ( uint8\_t *commandID*, uint8\_t *payload* )

Send a single command

Definition at line 74 of file SerialPacket.cpp.

Here is the call graph for this function:

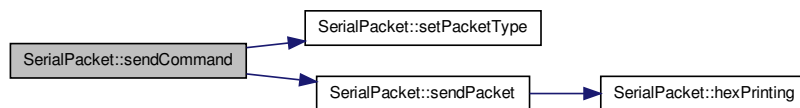


### 3.2.3.11 void SerialPacket::sendCommand ( uint8\_t *payload* )

Send a single command, reuses commandID from previous packets

Definition at line 84 of file SerialPacket.cpp.

Here is the call graph for this function:

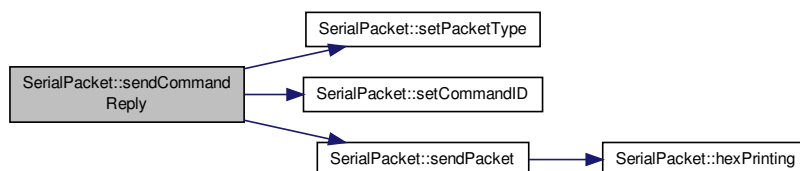


### 3.2.3.12 void SerialPacket::sendCommandReply ( uint8\_t *commandID*, uint8\_t *payload* )

Send a reply to a command

Definition at line 93 of file SerialPacket.cpp.

Here is the call graph for this function:



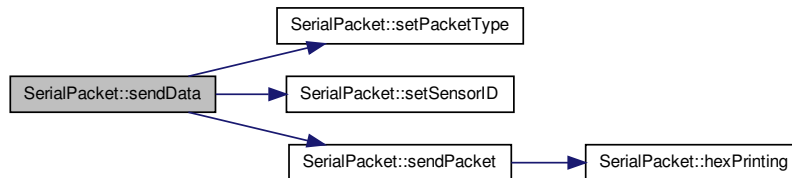
### 3.2.3.13 void SerialPacket::sendData ( uint8\_t *sensorID*, uint8\_t *payload* )

Send a single data value

T12N00I00P00Q12

Definition at line 123 of file SerialPacket.cpp.

Here is the call graph for this function:

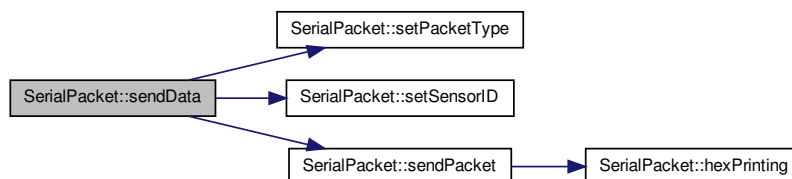


### 3.2.3.14 void SerialPacket::sendData ( uint8\_t *sensorID*, int16\_t *payload* )

Send a single data value

Definition at line 133 of file SerialPacket.cpp.

Here is the call graph for this function:

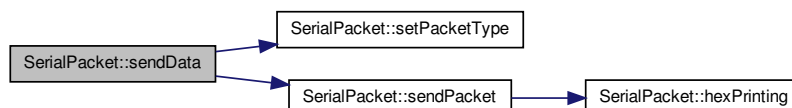


### 3.2.3.15 void SerialPacket::sendData ( uint8\_t *payload* )

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

Definition at line 143 of file SerialPacket.cpp.

Here is the call graph for this function:

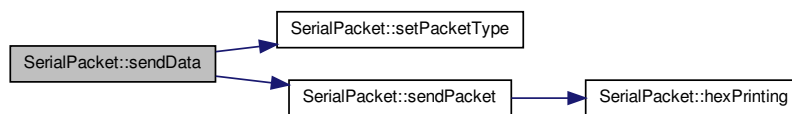


### 3.2.3.16 void SerialPacket::sendData ( int16\_t *payload* )

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

Definition at line 152 of file SerialPacket.cpp.

Here is the call graph for this function:



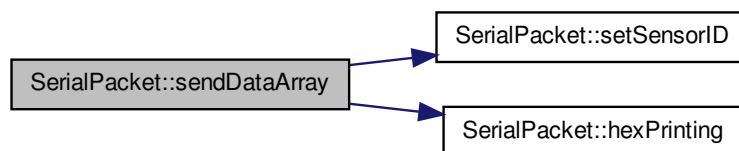
### 3.2.3.17 void SerialPacket::sendDataArray ( uint8\_t \* dataArray, uint8\_t length )

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

T12N00I00P1CQ0E

Definition at line 213 of file SerialPacket.cpp.

Here is the call graph for this function:

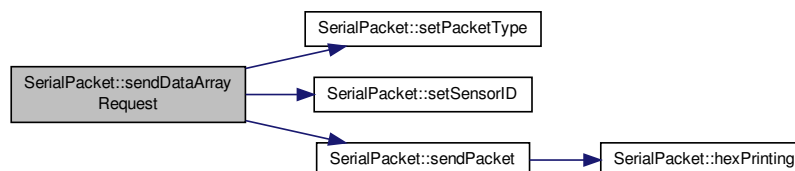


### 3.2.3.18 void SerialPacket::sendDataArrayRequest ( uint8\_t arrayID, uint8\_t payload )

Request an array of data values

Definition at line 113 of file SerialPacket.cpp.

Here is the call graph for this function:

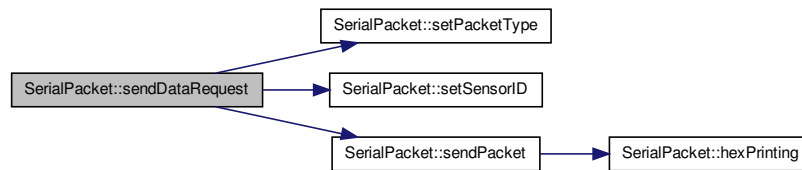


### 3.2.3.19 void SerialPacket::sendDataRequest ( uint8\_t sensorID, uint8\_t payload )

Request a single data value

Definition at line 103 of file SerialPacket.cpp.

Here is the call graph for this function:



### 3.2.3.20 void SerialPacket::sendPacket ( uint8\_t & payload ) [private]

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

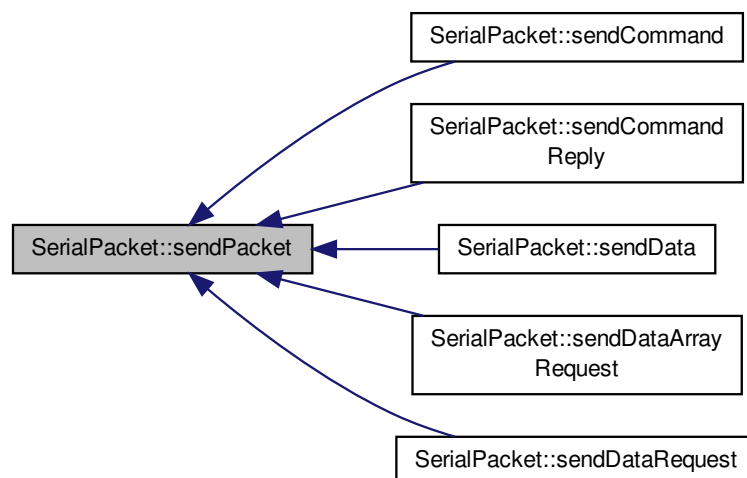
Definition at line 161 of file SerialPacket.cpp.

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

Here is the call graph for this function:



Here is the caller graph for this function:



**3.2.3.21** void SerialPacket::sendPacket ( int16\_t & *payload* ) [private]

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

Definition at line 187 of file SerialPacket.cpp.

Here is the call graph for this function:

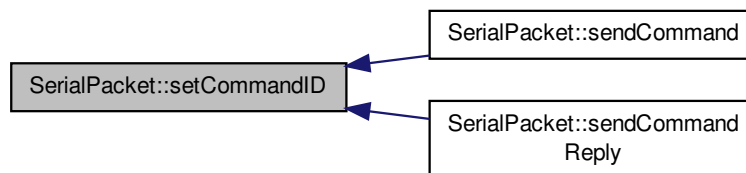
**3.2.3.22** void SerialPacket::setCommandID ( uint8\_t & *commandID* ) [private]

Set commandID

Definition at line 238 of file SerialPacket.cpp.

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

Here is the caller graph for this function:

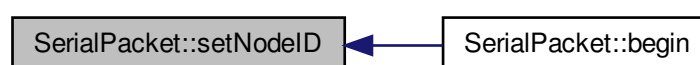
**3.2.3.23** void SerialPacket::setNodeID ( uint8\_t & *nodeID* ) [private]

Set nodeID

Definition at line 286 of file SerialPacket.cpp.

Referenced by `begin()`.

Here is the caller graph for this function:



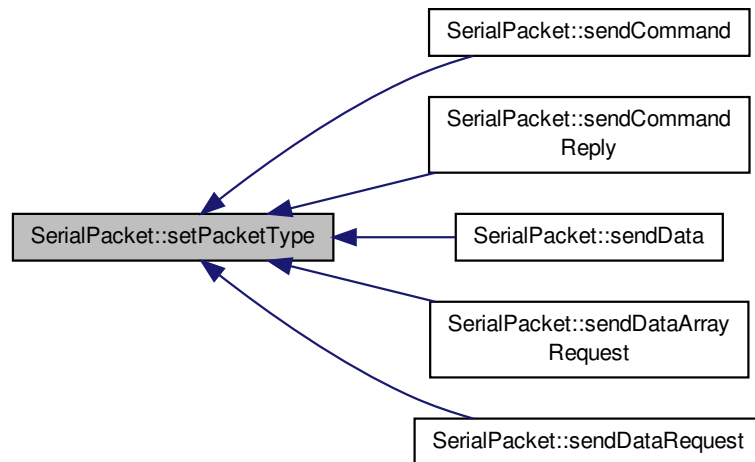
### 3.2.3.24 void SerialPacket::setPacketType ( uint8\_t type ) [private]

Set packet type

Definition at line 246 of file SerialPacket.cpp.

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

Here is the caller graph for this function:



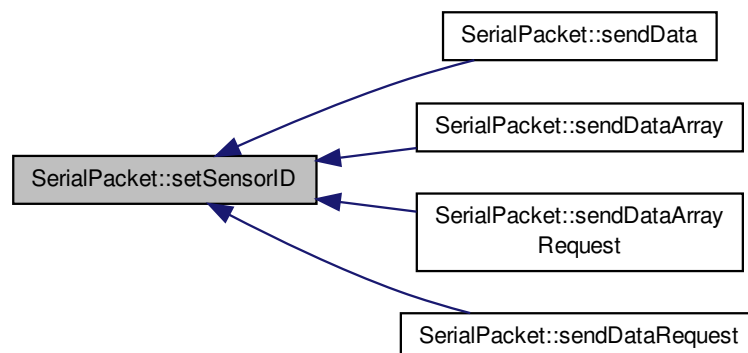
### 3.2.3.25 void SerialPacket::setSensorID ( uint8\_t & sensorID ) [private]

Set sensorID

Definition at line 294 of file SerialPacket.cpp.

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

Here is the caller graph for this function:



### 3.2.4 Member Data Documentation

#### 3.2.4.1 `uint8_t SerialPacket::_checkedParity` [private]

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

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

#### 3.2.4.2 `uint8_t SerialPacket::_commandID` [private]

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

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

#### 3.2.4.3 `uint8_t SerialPacket::_incomingCounter` [private]

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

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

#### 3.2.4.4 `boolean SerialPacket::_inComingPacketComplete` [private]

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

Referenced by `SerialPacket()`.

#### 3.2.4.5 `char SerialPacket::_inputChar[20]` [private]

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

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

#### 3.2.4.6 `uint8_t SerialPacket::_nodeID` [private]

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

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

#### 3.2.4.7 `uint8_t SerialPacket::_packetType` [private]

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

Referenced by `sendDataArray()`, `sendPacket()`, and `setPacketType()`.

#### 3.2.4.8 `uint8_t SerialPacket::_parity` [private]

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

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

#### 3.2.4.9 `uint8_t SerialPacket::_sensorID` [private]

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

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

#### 3.2.4.10 `struct SerialPacket::packet SerialPacket::incomingPacket` [private]

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

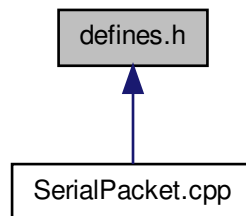
#### 3.2.4.11 `struct SerialPacket::packet SerialPacket::outgoingPacket` [private]

## 4 File Documentation



## 4.1 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`

#### 4.1.1 Macro Definition Documentation

##### 4.1.1.1 `#define COMMAND 0x01`

Definition at line 6 of file defines.h.

Referenced by `SerialPacket::parseSerialData()`, `SerialPacket::sendCommand()`, and `SerialPacket::sendPacket()`.

##### 4.1.1.2 `#define COMMAND_REPLY 0x02`

Definition at line 7 of file defines.h.

Referenced by `SerialPacket::parseSerialData()`, `SerialPacket::sendCommandReply()`, and `SerialPacket::sendPacket()`.

##### 4.1.1.3 `#define DATA_ARRAY 0x22`

Definition at line 14 of file defines.h.

##### 4.1.1.4 `#define DATA_ARRAY_REQUEST 0x21`

Definition at line 13 of file defines.h.

Referenced by `SerialPacket::sendDataArrayRequest()`, and `SerialPacket::sendPacket()`.

#### 4.1.1.5 #define DATA\_BYTE 0x12

Definition at line 10 of file defines.h.

Referenced by SerialPacket::parseSerialData(), SerialPacket::sendData(), and SerialPacket::sendPacket().

#### 4.1.1.6 #define DATA\_INT 0x13

Definition at line 11 of file defines.h.

Referenced by SerialPacket::parseSerialData(), SerialPacket::sendData(), and SerialPacket::sendPacket().

#### 4.1.1.7 #define DATA\_REQUEST 0x11

Definition at line 9 of file defines.h.

Referenced by SerialPacket::parseSerialData(), SerialPacket::sendDataRequest(), and SerialPacket::sendPacket().

#### 4.1.1.8 #define DISTANCE 0x30

Definition at line 21 of file defines.h.

#### 4.1.1.9 #define HUMIDITY 0x11

Definition at line 19 of file defines.h.

#### 4.1.1.10 #define SERIAL\_ASCII

Definition at line 3 of file defines.h.

#### 4.1.1.11 #define TEMPERATURE 0x10

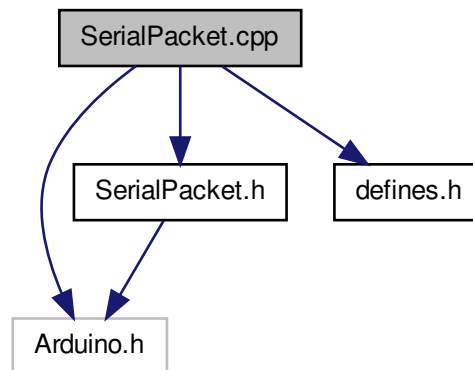
Sensor Types:

Definition at line 18 of file defines.h.

## 4.2 SerialPacket.cpp File Reference

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

Include dependency graph for SerialPacket.cpp:



#### Macros

- `#define HEX_DEC_ERROR 42`  
*Convert HEX to Decimal*

#### 4.2.1 Macro Definition Documentation

##### 4.2.1.1 `#define HEX_DEC_ERROR 42`

Convert HEX to Decimal

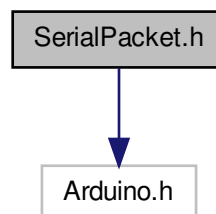
Definition at line 343 of file SerialPacket.cpp.

Referenced by SerialPacket::hex\_to\_dec().

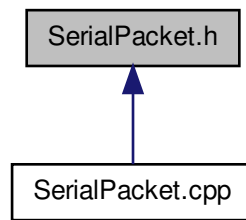
#### 4.3 SerialPacket.h File Reference

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

Include dependency graph for SerialPacket.h:



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



#### Classes

- class [SerialPacket](#)
- struct [SerialPacket::packet](#)

## Index

- `_checkedParity`  
SerialPacket, 14
  - `_commandID`  
SerialPacket, 14
  - `_inComingPacketComplete`  
SerialPacket, 14
  - `_incomingCounter`  
SerialPacket, 14
  - `_inputChar`  
SerialPacket, 14
  - `_nodeID`  
SerialPacket, 14
  - `_packetType`  
SerialPacket, 14
  - `_parity`  
SerialPacket, 14
  - `_sensorID`  
SerialPacket, 14
- `begin`  
SerialPacket, 5
- `COMMAND`  
defines.h, 15
- `COMMAND_REPLY`  
defines.h, 15
- `checkParity`  
SerialPacket, 5
- `commandID`  
SerialPacket::packet, 2
- `DATA_ARRAY`  
defines.h, 15
- `DATA_ARRAY_REQUEST`  
defines.h, 15
- `DATA_BYTE`  
defines.h, 15
- `DATA_INT`  
defines.h, 16
- `DATA_REQUEST`  
defines.h, 16
- `DISTANCE`  
defines.h, 16
- defines.h, 15
  - `COMMAND`, 15
  - `COMMAND_REPLY`, 15
  - `DATA_ARRAY`, 15
  - `DATA_ARRAY_REQUEST`, 15
  - `DATA_BYTE`, 15
  - `DATA_INT`, 16
  - `DATA_REQUEST`, 16
  - `DISTANCE`, 16
  - `HUMIDITY`, 16
  - `SERIAL_ASCII`, 16
  - `TEMPERATURE`, 16
- `HEX_DEC_ERROR`  
SerialPacket.cpp, 17
- `HUMIDITY`  
defines.h, 16
- `hex_to_dec`  
SerialPacket, 6
- `hexPrinting`  
SerialPacket, 6
- `incomingPacket`  
SerialPacket, 14
- `nodeID`  
SerialPacket::packet, 2
- `outgoingPacket`  
SerialPacket, 14
- `packetType`  
SerialPacket::packet, 2
- `parity`  
SerialPacket::packet, 2
- `parseSerialData`  
SerialPacket, 6
- `payload`  
SerialPacket::packet, 2
- `printInfo`  
SerialPacket, 7
- `readSerialData`  
SerialPacket, 7
- `SERIAL_ASCII`  
defines.h, 16
- `sendCommand`  
SerialPacket, 7, 8
- `sendCommandReply`  
SerialPacket, 8
- `sendData`  
SerialPacket, 8, 9
- `sendDataArray`  
SerialPacket, 10
- `sendDataArrayRequest`  
SerialPacket, 10
- `sendDataRequest`  
SerialPacket, 10
- `sendPacket`  
SerialPacket, 11
- `sensorID`  
SerialPacket::packet, 2
- SerialPacket, 2
  - `_checkedParity`, 14
  - `_commandID`, 14
  - `_inComingPacketComplete`, 14
  - `_incomingCounter`, 14
  - `_inputChar`, 14
  - `_nodeID`, 14

- [\\_packetType](#), 14
  - [\\_parity](#), 14
  - [\\_sensorID](#), 14
- [begin](#), 5
- [checkParity](#), 5
- [hex\\_to\\_dec](#), 6
- [hexPrinting](#), 6
- [incomingPacket](#), 14
- [outgoingPacket](#), 14
- [parseSerialData](#), 6
- [printInfo](#), 7
- [readSerialData](#), 7
- [sendCommand](#), 7, 8
- [sendCommandReply](#), 8
- [sendData](#), 8, 9
- [senddataArray](#), 10
- [senddataArrayRequest](#), 10
- [sendDataRequest](#), 10
- [sendPacket](#), 11
- [SerialPacket](#), 5
- [SerialPacket](#), 5
- [setCommandID](#), 12
- [setNodeID](#), 12
- [setPacketType](#), 12
- [setSensorID](#), 13
- [SerialPacket.cpp](#), 16
  - [HEX\\_DEC\\_ERROR](#), 17
- [SerialPacket.h](#), 17
- [SerialPacket::packet](#), 1
  - [commandID](#), 2
  - [nodeID](#), 2
  - [packetType](#), 2
  - [parity](#), 2
  - [payload](#), 2
  - [sensorID](#), 2
- [setCommandID](#)
  - [SerialPacket](#), 12
- [setNodeID](#)
  - [SerialPacket](#), 12
- [setPacketType](#)
  - [SerialPacket](#), 12
- [setSensorID](#)
  - [SerialPacket](#), 13
- TEMPERATURE
  - [defines.h](#), 16