AMASP Arduino Library

Generated by Doxygen 1.8.20

Chapter 1

README

AMASP Arduino Library

This library implements the AMASP (ASCII Master/Slave Protocol) for Arduino boards, a simple way to exchange messages between two computers using serial communication.

AMASP is free and uses four different packets:

MASTER -> SLAVE:

MRP - Master Request Packet CEP - Communication Error Packet

SLAVE -> MASTER:

SRP - Slave Response Packet SIP - Slave Interruption Packet CEP - Communication Error Packet

The protocol is transparent to the user that only needs to use the AMASP Arduino Library functions to implement his own applications. Please, take a look at the example codes.

AMASPArduinoLib is under test and improvements, if you have any problem using it, please send a mail to the author (Spanish, Portuguese or English) adelai@gmail.com.

Contributors will be welcome!

Do you want to design an AMASP library to other platforms? Be my guest!

Documentation about AMASP available here: https://doi.org/10.14209/jcis.2019.1

Author:

Andre L. Delai adelai@gmail.com

Enjoy, it's free! :)

Chapter 2

Class Index

2.1 Class List

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

Chapter 3

Class Documentation

3.1 AMASPSerialMaster Class Reference

Public Member Functions

- void begin (HardwareSerial &serial)

  Initializes the master connecting it to the serial port.
- void end ()

  Finalizes the master disconnect it from the serial port.
- int sendRequest (int deviceID, byte message[ ], int msgLength)

  Send a MRP (Master Request Packet).
- int sendError (int device, int errorCode)

  Send a CEP (Communication Error Packet).
- PacketType readPacket (int &deviceID, byte message[ ], int &codeLength)

  Read the incoming AMASP packet.
- PacketType readPacket (int &deviceID, byte message[ ], int &codeLength, ErrorCheck &eca, int &ecd)

  Read the incoming AMASP packet.
- void setErrorCheck (ErrorCheck eca)

  Sets the ECA (Error Check Algorithm).

3.1.1 Member Function Documentation

3.1.1.1 begin()

```
void AMASPSerialMaster::begin (
            HardwareSerial & serial )
```

Initializes the master connecting it to the serial port.

Parameters

serial	Serial communication object (input).

3.1.1.2 end()

```
void AMASPSerialMaster::end ( )
```

Finalizes the master disconnect it from the serial port.

3.1.1.3 readPacket() [1/2]

```
PacketType AMASPSerialMaster::readPacket (
            int & deviceID,
            byte message[],
            int & codeLength )
```

Read the incoming AMASP packet.

Parameters

deviceID	The device identification (output).
message	The message read from the associated device (output).
msgLength	The Message length in bytes (output).

Returns

Returns a PacketType enumeration (MRP, SRP, SIP, CEP or timeout). If timeout is returned, no valid AMASP packet was found.

3.1.1.4 readPacket() [2/2]

```
PacketType AMASPSerialMaster::readPacket (
            int & deviceID,
            byte message[],
            int & codeLength,
            ErrorCheck & eca,
            int & ecd )
```

Read the incoming AMASP packet.

Parameters

deviceID	The device identification (output).
message	The message read from the associated device (output).
msgLength	The Message length in bytes (output).
eca	The error Check Algorithm used by the packet (output).
ecd	The error Check Data of the packet (output).

Returns

Returns a PacketType enumeration (MRP, SRP, SIP, CEP or timeout). If timeout is returned, no valid AMASP packet was found.

3.1.1.5 sendError()

```
int AMASPSerialMaster::sendError (
            int device,
            int errorCode )
```

Send a CEP (Communication Error Packet).

Parameters

deviceID	The device identification (input).
errorCode	The communication error code (input).

Returns

Returns the generated error check data.

3.1.1.6 sendRequest()

```
int AMASPSerialMaster::sendRequest (
            int deviceID,
            byte message[],
            int msgLength )
```

Send a MRP (Master Request Packet).

Parameters

deviceID	The device identification (input).
message	The message to be send to the associated device (input).
msgLength	Message length in bytes (input).

Returns

Returns the generated error check data.

3.1.1.7 setErrorCheck()

```
void AMASPSerialMaster::setErrorCheck (
            ErrorCheck eca )
```

Sets the ECA (Error Check Algorithm).

Parameters

eca	Error Check Algorithm.

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

- C:/Users/delai/Documents/Repositorio/arduinoamasplib/AMASPLib/AMASP.h
- C:/Users/delai/Documents/Repositorio/arduinoamasplib/AMASPLib/AMASPSerialMaster.cpp

3.2 AMASPSerialSlave Class Reference

Public Member Functions

- void begin (HardwareSerial &serial)
  Initializes the slave connecting it to the serial port.
- void end ()
  Finalizes the slave disconnect it from the serial port.
- int sendResponse (int deviceID, byte message[ ], int msgLength)
  Send a SRP (Slave Response Packet).
- int sendInterruption (int deviceID, int code)
  Send a SIP (Slave Interruption Packet).
- int sendError (int Device, int code)
  Send a CEP (Communication Error Packet).
- PacketType readPacket (int &deviceID, byte message[ ], int &codeLength)
  Read the incoming AMASP packet.
- PacketType readPacket (int &deviceID, byte message[ ], int &codeLength, ErrorCheck &eca, int &ecd)
  Read the incoming AMASP packet.
- void setErrorCheck (ErrorCheck eca)
  Sets the ECA (Error Check Algorithm).

3.2.1 Member Function Documentation

3.2.1.1 begin()

```
void AMASPSerialSlave::begin (
            HardwareSerial & serial )
```

Initializes the slave connecting it to the serial port.

Parameters

serial	Serial communication object (input).

3.2.1.2 end()

```
void AMASPSerialSlave::end ( )
```

Finalizes the slave disconnect it from the serial port.

3.2.1.3 readPacket() [1/2]

```
PacketType AMASPSerialSlave::readPacket (
            int & deviceID,
            byte message[],
            int & codeLength )
```

Read the incoming AMASP packet.

Parameters

deviceID	The device identification (output).
message	The message read from the associated device (output).
msgLength	The Message length in bytes (output).

Returns

Returns a PacketType enumeration (MRP, SRP, SIP, CEP or timeout). If timeout is returned, no valid AMASP packet was found.

3.2.1.4 readPacket() [2/2]

```
PacketType AMASPSerialSlave::readPacket (
            int & deviceID,
            byte message[],
            int & codeLength,
            ErrorCheck & eca,
            int & ecd )
```

Read the incoming AMASP packet.

Parameters

deviceID	The device identification (output).
message	The message read from the associated device (output).
msgLength	The Message length in bytes (output).
eca	The error Check Algorithm used by the packet (output).
ecd	The error Check Data of the packet (output).

Returns

Returns a PacketType enumeration (MRP, SRP, SIP, CEP or timeout). If timeout is returned, no valid AMASP packet was found.

3.2.1.5 sendError()

```
int AMASPSerialSlave::sendError (
            int Device,
            int code )
```

Send a CEP (Communication Error Packet).

Parameters

deviceID	The device identification (input).
errorCode	The communication error code (input).

Returns

Returns the generated error check data.

3.2.1.6 sendInterruption()

```
int AMASPSerialSlave::sendInterruption (
            int deviceID,
            int code )
```

Send a SIP (Slave Interruption Packet).

Parameters

deviceID	The device identification (input).
errorCode	The interruption code (input).

Returns

Returns the generated error check data.

3.2.1.7 sendResponse()

```
int AMASPSerialSlave::sendResponse (
            int deviceID,
            byte message[],
            int msgLength )
```

Send a SRP (Slave Response Packet).

Parameters

deviceID	The device identification (input).
message	The message to be send from the associated device (input).
msgLength	The message length in bytes (input).

Returns

Returns the generated error check data.

3.2.1.8 setErrorCheck()

```
void AMASPSerialSlave::setErrorCheck (
            ErrorCheck eca )
```

Sets the ECA (Error Check Algorithm).

Parameters

eca	Error Check Algorithm (input).

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

- C:/Users/delai/Documents/Repositorio/arduinoamasplib/AMASPLib/AMASP.h
- C:/Users/delai/Documents/Repositorio/arduinoamasplib/AMASPLib/AMASPSerialSlave.cpp

Index