Null Packet Comms Arduino

Release 0.2.3

Steve Richardson (Creating Null)

Apr 16, 2023

DOCUMENTATION:

This is an arduino library that implements a binary packet-based communication protocol on top of the arduino Serial library. This wrapper allows for robust generic data transfer between systems, it is heavily relied on for the Arduino UOS implementation.

The library exposes a class which is designed to mimic the Serial API, but using formatted packets.

CHAPTER

ONE

NPC ARDUINO API

The following is the class structure for the library. This documents all the user-facing functions and members.

class **NullPacketComms**

Serial connection object to be handled via NPC protocol.

Public Functions

NullPacketComms()

Constructor to initialise the wrapper.

bool **begin**(uint32_t baud_rate)

Opens the connection to the UART port.

Parameters
baud_rate – Standard UART baud rate to use.

Returns
`true` if opened, `false` if unsupported baud.

void **end**()

Closes the connection to the UART port.

int **available**()

Number bytes waiting in the serial buffer.

Returns
number of bytes.

bool **readPacket**(bool manual_ack = false)

Reads the next packet from the serial buffer.

Parameters
manual_ack – (optional) Set `true` to respond ACK in client code.

Returns
`true` if read success, `false` otherwise.

uint8_t **writePacket**(uint8_t target, uint8_t data[], uint8_t data_len)

Builds a data packet and sends it to the host system.

Parameters

- **target** – Of the system to respond to.
- **data** – Payload array to populate into the packet.

- **data_len** – Length of the payload array.

Returns
Number of bytes successfully sent.

bool **writeAck**(uint8_t target, uint8_t ack_code)

Sends an ack packet in response to a resolved rx packet.

Parameters

- **target** – Address of system we are responding from.
- **ack_code** – NACK error code or `0` for ACK.

Returns
`true` if ack was sent successfully, else `false`.

Public Members

uint8_t **target_**

Stores the to address of the last rx packet.

uint8_t **len_**

Stores the payload length populated with the last rx packet.

uint8_t **payload_**[58]

Stores the payload from the last rx packet.

CHAPTER

TWO

CHANGELOG

2.1 Version 0.2.3

Date
16-April-2023

- Googletest framework updated to 1.13.0
- Correcting incorrect syntax used for rst external hyperlink in docs.
- Adding a check on CI build against the `simple_example` sketch to verify integration.

2.2 Version 0.2.2

Date
31-December-2022

- Fixing doc shields which inspected github workflows (see: issue).
- Defining test data as constant where it is used as a constant.

2.3 Version 0.2.1

Date
2-October-2022

- Used `Serial.available` calls in preference to the nested `NullPacketComms.available` internally.
- Sourcing doc metadata from `library.properties`. This fixes a issues from me forgetting to maintain `conf.py`.

2.4 Version 0.2.0

Date
2-October-2022

- Adding support for higher baud-rates 128000 to 2000000.
- Googletest framework updated to 1.12.1.
- Added `waitOnAvailable` private function to allow for dynamic delays pending next bytes.
- Reduced idle time from 25ms to 1ms while waiting on buffer to accumulate. This improves responsiveness, especially at higher baud rates.
- Error case for loss of start packet sync set to dynamically delay rather than blindly delay.

2.5 Version 0.1.2

Date
18-April-2022

- Explicit casting used on `Serial.write` calls to add support for platform cores with ambiguous prototypes.
- Improvements to distributed docs.

2.6 Version 0.1.1

Date
14-February-2022

- Removing inclusion of files not required for the dist.
- Fixing permalinks to docs.

2.7 Version 0.1.0

Date
12-February-2022

- Initial release of the redesigned API.

INDEX

N