1. Program configuration

Configuration flags allow to select functionality for master, slave or both. Please comment/uncomment corresponding #define in .ino file.

By default, master and slave shall use UART2 (RX2 and TX2 pins on NodeMCU board) . In case of loopback, Slave will use UART1 on GPIO21 (RxD) and GPIO22 (TxD) pins. Please uncomment #define LOOPBACK in order to use it.

Verbosity level (DEBUG, INFO, WARNING, ERROR) is controlled by defines in the same file too.

The interval at which the recurrent commands are send is set in POLL\_INTERVAL in milliseconds. REPLY\_TIMEOUT specify the time window in which the master will wait for reply.

1. Communication system

Designed to implement lightweight bidirectional Command-Response style protocol over RS485 unidirectional bus. Architecture consist of one master (0x0) and up to 14 slave boards (0x1 – 0xE). Address 0xF is reserved for broadcast purposes. Every transaction consists of:

1. message containing command plus optional payload
2. reply message with response code for the command plus optional payload).

Transactions can be originated by the master only. Up to 8 commands (0x0 – 0x7) are supported and there are corresponding 8 reply codes. The reply code is the same as with 4-th bit set (0x8 – 0xF).

For commands encoding and transmission, Nick Gammon RS485 lib is used, with few things hacked like message receive timeout support, embedded errors generation for testing, errors reporting, etc.

Commands and replies are delivered by messages, one message per command/reply. Message frame consist of up to 128bytes. The size is selected in order to match ESP32 FIFO size. Every message starts with one-byte STX (0x3), up to 124 bytes encoded payload, two bytes encoded 8bit CRC and ends one-byte ETX. Message payload and CRC are encoded for errors detection, every 4bits are shifted and complimented with they inverse values. As a result, every byte is sent as two bytes and STX and ETX markers will not be present into the message payload, leading to more reliable detection of the message frame.

Transmit of the message is blocking, but receive of the message is not blocking.

1. Master operation

Master loop is in master.h file which is included in loop() function if corresponding #define (MASTER) is #define-ed.

* 1. Receiving command reply

The process of receiving message is non-blocking, allowing while waiting for message to do other staff. It is implemented by calling check4msg() which returns with no delay with:

1. either message available;
2. no message available yet;
3. error or timeout while receiving message.

If message is available, it is retrieved and processed by masterProcessMsg(). In case of error or reply timeout has expired, the error is printed to console (if enabled). The timeout parameter to function determines for how long check4msg() shall wait for message to be received before give-up. Other potential errors which can occur while receiving message like bad CRC, bad payload size, etc are handled locally by check4msg() and lower level functions. See chapter for errors handling for more details.

* 1. Sending command

If there is no message available, the program checks if it is time to send recurrent command like POLL. The check is done by calling function isTimeFor with parameters command code and time interval in milliseconds. In case it is time to send, actually send is performed by sendCmd function, which receives as parameters command ID, destination address, and message payload. As payload length is determined by command ID, no need to provide payload length. The function is blocking, but because by design the message frame shall completely fit in ESP32 Tx FIFO, this should not lead to delays. Error handling while sending is handled internally in sendCmd function. sendCmd function also sets global waiting\_for\_reply flag, used to determine if command is send and master has to wait for reply and records the time when the transmission is done (in RS485 class) for reply timeout checking purposes.

1. Slave operation

Slave loop is in slave.h file which is included in loop() function if corresponding #define (SLAVE) is #define-ed.

* 1. Receiving command

Receiving command is implemented again with non-blocking check4msg(). The timeout parameter is zero as the slave can receive commands any time. If check4msg() returns that command has received, the corresponding action is taken. If check4msg() returns error, error is printed to console or MQTT.