

Report **UART test software**

Date: 30 April 2022

Written by: René Weimar

Copy to: Kumkeo

Enter name here

| | | |
|----------|--|----------|
| 1 | Introduction | 2 |
| 2 | Protocol..... | 2 |
| 2.1 | Receiving..... | 2 |
| 2.2 | Transmitting..... | 2 |
| 2.3 | CRC | 2 |
| 3 | Equipment needed | 3 |
| 3.1 | STM32 Cube IDE..... | 3 |
| 3.2 | HC-06..... | 3 |
| 4 | Setting up | 3 |
| 4.1 | UART connections | 3 |
| 4.2 | HC-06 module | 3 |
| 4.2.1 | Set baud rate..... | 3 |
| 4.2.2 | Install APP | 3 |
| 4.2.3 | Setting up APP | 4 |
| 5 | Software explanation | 5 |
| 5.1 | Receiving & transmitting commands..... | 5 |
| 5.2 | Receiving recipes..... | 5 |

1 Introduction

This document describes the test software for UART supplied to Kumkeo for testing.

2 Protocol

The protocol used is described in detail in the document “80482201_UART protocol V1.0.xlsx”
All equipment follows the same protocol:

2.1 Receiving

| Byte Nr | Item | Bytes | Start byte | End byte | Min | Max | Required | Value | Remark |
|---------|--------------------------|-------|------------|----------|-----|-----|----------|-------------|---|
| | Total bytes 1 recipe max | 11 | | | | | | | |
| 0 | StartByte | 1 | 0 | 0 | 0 | | Yes | / | Start character |
| 1 | Length MSB | 1 | 1 | 1 | 0 | | Yes | 0 | Length of command (Include CRC length) |
| 2 | Length LSB | 1 | 2 | 2 | 8 | | Yes | 11 | |
| 3 | Type of message MSB | 1 | 3 | 3 | 0 | | Yes | See options | 01 = CoffeeRecipe, 02 = MilkRecipe, 03 = Water recipe 11 = StartCoffee, 12 = StartMilk, 13 = StartWater, 14 = StopCoffee, 15 = StopMilk, 16 = StopWater, 21 = PartitionUpdate, 22 = Upgrade partition, 23 = Reboot, 24 = ResetFactory partition, 31 = GetProcessInfo |
| 4 | Type of message LSB | 1 | 4 | 4 | 0 | | Yes | | |
| 5 | CRC MSB | 1 | 5 | 5 | 0 | | Yes | | CRC value for data integrity check |
| 6 | CRC LSB | 1 | 6 | 6 | 0 | | Yes | | |
| 7 | Nr Of Blocks MSB | 1 | 7 | 7 | 0 | | Yes | | Nr of blocks that will be sent |
| 8 | Nr Of Blocks LSB | 1 | 8 | 8 | 0 | | Yes | | |
| 9 | Current Block MSB | 1 | 9 | 9 | 0 | | Yes | | Current block that is being sent |
| 10 | Current Block LSB | 1 | 10 | 10 | 0 | | Yes | | |
| 11 - | Data block(s) | | | | | | No | | Data (Values * 2 bytes) |

2.2 Transmitting

| | Item | Bytes | | | Min | Max | Required | | Remark |
|-----|--------------------------|-------|---|---|-----|-----|----------|-------------|---|
| | Total bytes 1 recipe max | 9 | | | | | | | |
| 0 | StartByte | 1 | 0 | 0 | 0 | | Yes | / | Start character |
| 1 | Length MSB | 1 | 1 | 1 | 0 | | Yes | 0 | Length of command (Include CRC length) |
| 2 | Length LSB | 1 | 2 | 2 | 8 | | Yes | 9 | |
| 3 | Type of message MSB | 1 | 3 | 3 | 0 | | Yes | See options | 01 = CoffeeRecipe, 02 = MilkRecipe, 03 = Water recipe 11 = StartCoffee, 12 = StartMilk, 13 = StartWater, 14 = StopCoffee, 15 = StopMilk, 16 = StopWater, 21 = PartitionUpdate, 22 = Upgrade partition, 23 = Reboot, 24 = ResetFactory partition, 31 = GetProcessInfo |
| 4 | Type of message LSB | 1 | 4 | 4 | 0 | | Yes | | |
| 5 | CRC MSB | 1 | 5 | 5 | 0 | | Yes | | CRC value for data integrity check |
| 6 | CRC LSB | 1 | 6 | 6 | 0 | | Yes | | |
| 7 | Status MSB | 1 | 7 | 7 | 1 | | Yes | | 01 = Success, |
| 8 | Status LSB | 1 | 8 | 8 | 1 | | Yes | | 11 = CRC error, 12 = Unknown type, 13 = Unable to execute, |
| 9 - | Data block(s) | | | | | | No | | |

2.3 CRC

All messages include a CRC, for checking for transmission errors.

3 Equipment needed

3.1 STM32 Cube IDE

The software is written for the Nucleo-F030R8 in the STM32 Cube IDE

3.2 HC-06

For testing the BT connection an HC-06 module is needed, that can be bought for example here:

<https://www.roboter-bausatz.de/p/hc-06-wireless-bluetooth-arduino-modul-master-slave?number=RBS11778&sPartner=7>

4 Setting up

4.1 UART connections

The software can handle two UART:

- UART1 [PA9 and PA10]
The received info is put in Message[0] buffer
- UART2 [PA2 and PA3]
The received info is put in Message[1] buffer

One of the UART can be connected to the Linux system, whilst the other can be connected to the HC-06 module, simulating the grinder BT connection. (The final grinder will use this module too)

4.2 HC-06 module


4.2.1 Set baud rate

The software is written for a baudrate of 115200 and the HC-06 is standard configured for 9600 baud. Therefore it will need to be configured by sending AT+BAUD8 to the module.

Simultaneously, also the name can be changed using the AT+NAME command. For example AT+NAMENunc_12345678 will rename the module to 'Nunc_12345678'

4.2.2 Install APP

On Google an APP called Serial Bluetooth terminal can be downloaded:

 [Serial Bluetooth Terminal - Apps on Google Play](https://play.google.com/store/apps/details?id=de.kai_morich...)
https://play.google.com/store/apps/details?id=de.kai_morich...

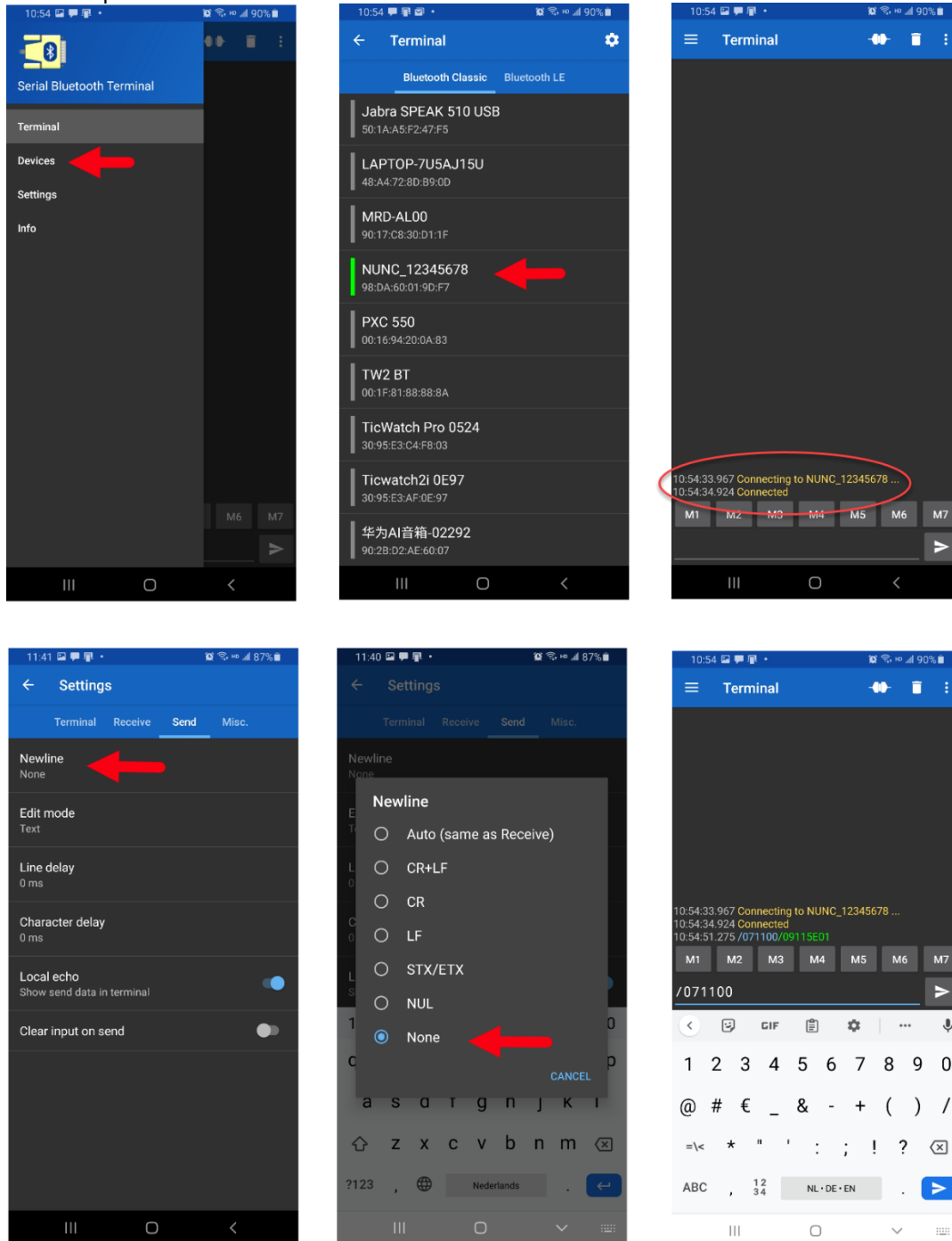


'Serial Bluetooth Terminal' is a line-oriented terminal microcontrollers, arduinos and other devices with a
4.5/5 ★★★★★ (2,2K) Developer: V
Age Rating: Everyone Size: 1.3M
Category: Tools

After installing the APP needs to be setup.

4.2.3 Setting up APP

The module can be found by going to settings and click DEVICES.
The Nunc_12345678 is connected by clicking on it.
After this press the menu button and choose Newline and set this to none.



In the last screenshot we see in blue the message sent (/071100) and in green the message received (/09115E01)

After starting the software, it receives automatically in any UART and returns answers automatically. The examples in the Excel file give you strings to send and expected return messages. Any message sent will begin with a start character ('/'), which makes the software clear the buffer.

Command 0x11 (Start coffee) is sent to the embedded board as message '071100'. The board returns the message '09110001' indicating it is received correctly and will be executed.

Command 0x11 (Start coffee) is sent to the embedded board as message '07110A'. The board returns the message '09110011' indicating it is received incorrectly, because the CRC is wrong and will not be executed.

5.2 Receiving recipes

1. Coffee (limited to max 8 blocks)
2. Milk (limited to max 2 blocks)
3. Water (limited to max 1 block)

When a correct recipe block is sent, the values are filled in a recipe structure that can be monitored in 'live expressions'. Example:

[illegible]

We see that the `RecipeBlocksCoffee[0]` container is filled with the corresponding values:

Report UART test software.docx
5 - 5