# A5 Annex 5: Pulse-rate meter instrument

## A5.1 Introduction

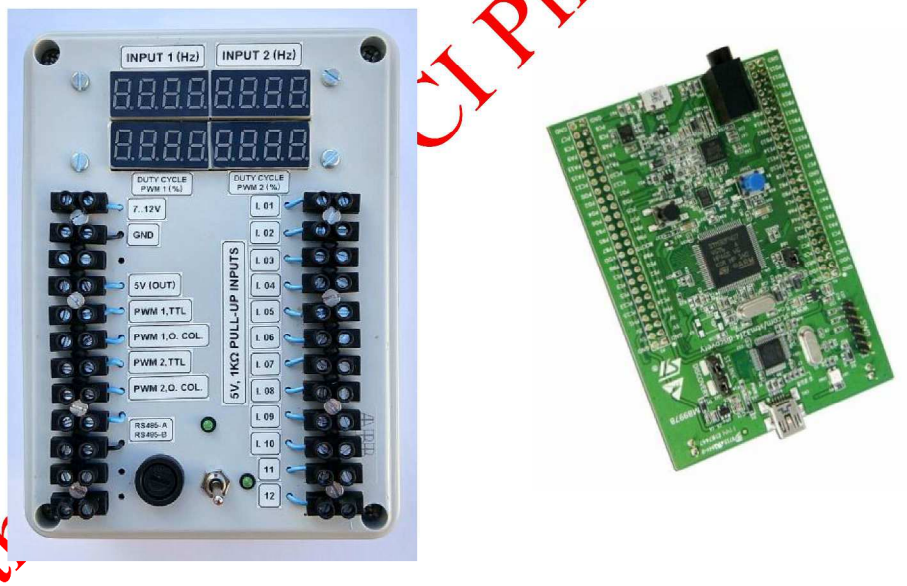
This section presents a custom-designed instrument that has two main features:

1. measurement of the pulse-rate of a number of digital input signals,
2. generation of pulse-width modulation (PWM) signals.

Regarding the first functionality, the device computes the pulse-rate of a number of digital inputs and transmits the information through a communication channel to a master device. More specifically, the device counts the number of rising edges of twelve digital inputs simultaneously. The counting process takes place during a series of consecutive one-second time frames. At the end of each time frame, the device updates the registers with the last pulse-counts. Since the time window duration is one second, the values of the registers are equal to the frequency of each input (in Hz). The master device (e.g. a computer) accesses the device's registers through a polling process using Modbus-RTU protocol.

As for the second functionality, the device generates two independent PWM signals, whose parameters (i.e. frequency and duty cycle) are set by the master device.

Fig. A5.1 (a) depicts a front view of the instrument and Fig. A5.1 (b) shows the microcontroller board used in the device.




(a) (b)

Fig. A5.1. Prototype: pulse rate meter device. (a) Front view of the device. (b) Microcontroller board used to implement the device.

The device has 13 pulse-rate inputs (from 0 to 12). The first input is connected to the internal board's push-button and it is used for testing purposes. The rest of the inputs (1 to 12) are connected to the device's front terminal blocks.

The Modbus communication link is establish through a RS-485 line. For this purpose, a 'TTL-RS485' mini board was utilized.

| Pulse-rate meter | |
|---|---|
| **Brand/model** | Custom-designed<br>Microcontroller board model: STM32F4DISCOVERY<br>(32bits ARM Cortex-M4, 164MHz) |
| **Power source** | Two options:<br>a) through USB connector<br>b) 7..12VDC input (fuse protected) |
| **Inputs** | 12 pull-up digital inputs 5VDC, 1kΩ |
| **PWM outputs** | 2 PWM independent signals, each of with two outputs:<br>• 1 x TTL<br>• 1 x open collector (1A max)<br>PWM frequencies (Hz): 100, 250, 500, 1k, 2.5k, 5k, 10k, 25k<br>PWM duty cycle: 0 to 100%, in steps of 1% |
| **Communication** | Modbus-RTU over RS-485<br>• Modbus slave address: 0x04<br>• Baud rate: 9600, data bits: 8, parity: none, stop bits: 1<br>• Available Modbus functions:<br>'0x04': Read Input Registers. Start address: 700<br>'0x10': Write Multiple Registers. Start address: 600 |
| **Display** | Four 7-segments, four digits displays<br>• Top-left display: input 1 pulse-rate (Hz)<br>• Top-right display: input 2 pulse-rate (Hz)<br>• Bottom-left display: PWM output 1 duty cycle (0..100%)<br>• Bottom-right display: PWM output 2 duty cycle (0..100%) |
| **Connectors** | • Two lines of terminal blocks (power, inputs, outputs, RS-485)<br>• USB port (accessible from the side of the device) used to program and debug the microcontroller board and also to power the device. |
| **Dimensions** | 15x12x7cm (height x width x depth)<br>DIN-riel clamp at the back of the device |

Table A5.1. Pulse-rate meter's characteristics.

## A5.2 Device's Modbus register mapping

Table A5.1 details the main characteristics of the instrument and Fig. A5.2 shows a timing diagram. The device stores the pulse-rate value of each input in a 32-bits register. These registers are accessed by a Modbus-RTU master device through a pooling process. Since Modbus is a 16-bits register protocol, each register is fragmented into two. Table A5.2 shows the read-only register address map.

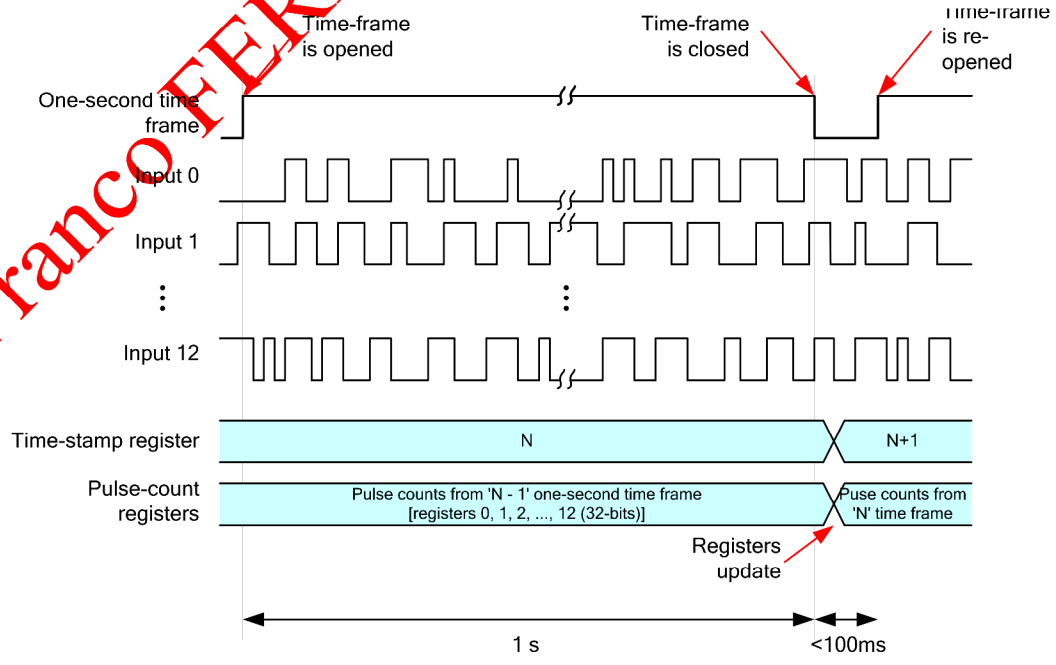



Fig. A5.2. Pulse-rate meter instrument. Timing diagram.

The device also offers a 16-bit 'time-stamp' register which is monotonically increased with every new one-second time frame. This gives the ability to the master device to identify if two sets of polled registers belong to the same 'one-second' time frame (since the polling process can be performed several times per second).

| Modbus "Read Input Registers" (0x04) address mapping (16-bits registers) | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 700 | 701 | 702 | 703 | 704 | 705 | 706 | 707 | 708 | 709 |
| PUSH BUTTON (HIGH) | PUSH BUTTON (LOW) | INPUT 1 (HIGH) | INPUT 1 (LOW) | INPUT 2 (HIGH) | INPUT 2 (LOW) | INPUT 3 (HIGH) | INPUT 3 (LOW) | INPUT 4 (HIGH) | INPUT 4 (LOW) |
| 710 | 711 | 712 | 713 | 714 | 715 | 716 | 717 | 718 | 719 |
| INPUT 5 (HIGH) | INPUT 5 (LOW) | INPUT 6 (HIGH) | INPUT 6 (LOW) | INPUT 7 (HIGH) | INPUT 7 (LOW) | INPUT 8 (HIGH) | INPUT 8 (LOW) | INPUT 9 (HIGH) | INPUT 9 (LOW) |
| 720 | 721 | 722 | 723 | 724 | 725 | 726 | | | |
| INPUT 10 (HIGH) | INPUT 10 (LOW) | INPUT 11 (HIGH) | INPUT 11 (LOW) | INPUT 12 (HIGH) | INPUT 12 (LOW) | TIME STAMP | | | |

Table A5.2. Pulse-rate meter. Modbus "Read Input Registers" (0x04) address mapping.

Table A5.3 shows the set of read/write registers used to control the PWM outputs, starting at address 600. Each of the two PWM generators has an enable/disable register (see Table A5.4), a frequency register and a duty-cycle register. Table A5.5 shows the correspondence between the registers' value and the output frequency. The 'PWM *X* DUTY CYCLE' registers (X = 1 or 2) can be set to any value between 0 and 100 (inclusive), which corresponds to the duty cycle, in percentage.

| Modbus "Write Multiple Registers" (0x10) address mapping (16-bits registers) | | | | | |
|---|---|---|---|---|---|
| 600 | 601 | 602 | 603 | 604 | 605 |
| PWM 1 STATUS CODE | PWM 1 FREQUENCY CODE | PWM 1 DUTY CYCLE | PWM 2 STATUS CODE | PWM 2 FREQUENCY CODE | PWM 2 DUTY CYCLE |

Table A5.3. Pulse-rate meter. Modbus "Write Multiple Registers" (0x10) address mapping.

| PWM status code | |
|---|---|
| **Register value** | **Status** |
| 0x0000 | disabled |
| any other | enabled |

Table A5.4. Pulse-rate meter. Registers '600' & '603' (PWM *X* STATUS CODE).

| PWM frequency code | |
|---|---|
| **Register value** | **Frequency (Hz)** |
| 0x0000 | 100 |
| 0x0001 | 250 |
| 0x0002 | 500 |
| 0x0003 | 1k |
| 0x0004 | 2.5k |
| 0x0005 | 5.0k |
| 0x0006 | 10k |
| 0x0007 | 25k |

Table A5.5. Pulse-rate meter. Registers '601' & '604' (PWM *X* FREQUENCY CODE).

Fig. A5.3 shows an example of communication between a computer and the device using the computer program "QModMaster" [1].

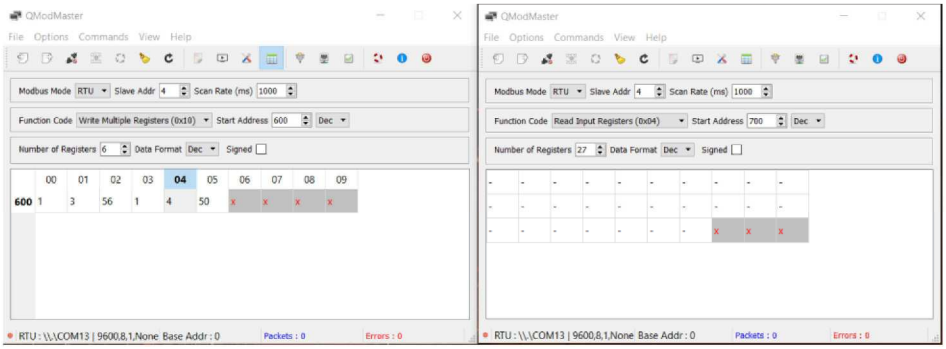



Fig. A5.3. Pulse-rate meter. Example of a Modbus-RTU communication between the device and a computer using the software "QModMaster".

## A5.3 Device programming

The device's board was programmed in C language. The Modbus-RTU stack was based on the open-source library 'FreeModbus'[2]. The library was ported to the STM32F4 DISCOVERY board with the help of 'STM32_HAL_FREEMODBUS_RTU' project [3].

The following development tools (IDE) were used to program and test the device:

- 'STMCube32MX', to configure the STM32F4 DISCOVERY board's peripherals (free license) [5]
- 'IAR Embedded Workbench', to program, compile and debug the code (free license) [4]
- 'Realterm', to debug the communication protocol (open-source) [5]
- 'QModMaster', to test the Modbus-RTU protocol (open-source) [1]
- 'KiCad', to create the device's schematics (open-source) [7]

The complete project has been uploaded into a GitHub repository [8]. Additionally, the following page shows the schematics of the device.



## References

[1] QModMaster open-source software. Available online at https://sourceforge.net/projects/qmodmaster/

[2] FreeModbus: A free Modbus RTU/ASCII implementation for embedded systems (C library). Available online at https://sourceforge.net/projects/freemodbus.berlios/

[3] FreeMODBUS RTU port for STM32 HAL library (C library). Available online at https://github.com/eziya/STM32_HAL_FREEMODBUS_RTU

[4] IAR Embedded Workbench - IAR Systems. Available online at https://www.iar.com/iar-embedded-workbench/

[5] STM32Cube initialization code generator: graphical tool for the configuration of STM32 microcontrollers. Available online at https://www.st.com/en/development-tools/stm32cubemx.html

[6] RealTerm: Serial/TCP terminal program for engineering. Available online at https://sourceforge.net/projects/realterm/

[7] KiCad EDA, A Cross Platform and Open Source Electronics Design Automation Suite. Available online at http://www.kicad-pcb.org/

[8] Pulse-rate meter measurement & PWM generator. Modbus-RTU communication. Github repository. Available online at https://github.com/ferrucci-franco/pulse_rate_meter_-_pwm_generator