Contents

Introduction:

The PCB18.16 is a sensor board with some GPIO’s and a 7-segment display.

The board uses a hybrid RP2040 development board for IO’s, this is the wiznet w5100s-evb-pico, it is identical for all intentional purposes to the RP2040 board with the exception that is has ethernet on SPI1.

Visual studio example:

https://docs.wiznet.io/Product/iEthernet/W5100S/w5100s-evb-pico

Arduino example:

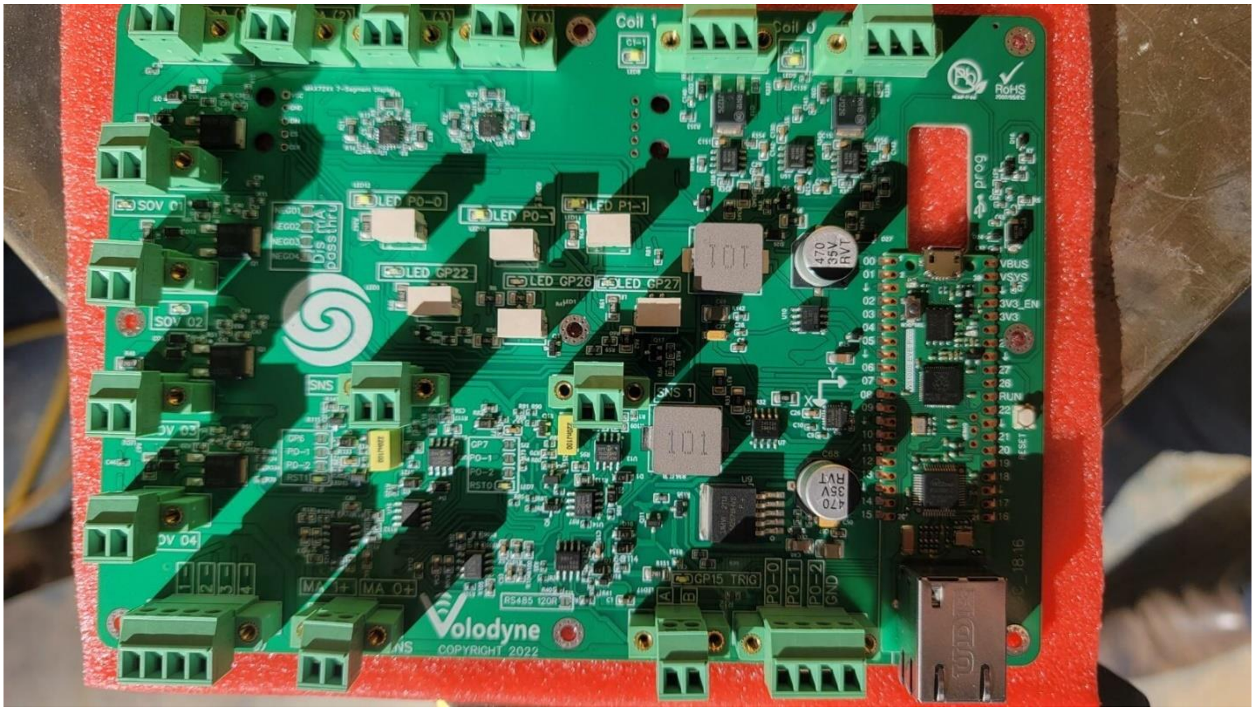
https://maker.wiznet.io/2021/11/29/w5100s-evb-pico-with-the-arduino-ide/

Sample code for the ethernet board:

https://github.com/WIZnet-ArduinoEthernet/Ethernet/archive/master.zip

HARDWARE:

PCB 18.16



A PCB was developed using the Wiznet w5100s-evb-pico board, the PCB 18.16 has the following,

I2C

GPIO8 – SDA, GPIO9 - SDL.

Eeprom

Model: CAT24C512WI-GT3

ADDRESS: 0x50

The CAT24C512 is a 512KB EEPROM unit used to store non-volatile global values for reading and writing to by the RP2040. Memory usage shall be bits for state settings , words (2 byte) integers for general settings and Dword (4 bytes) long for counter that is displayed periodically on the 7 segment 8 digit display.

Gyroscope

Model: MPU6050

ADDRESS: 0x68

Gyroscope is used to provide a calculated relative acceleration value, the pre processing is bon on the board and the output is provided as an integer or float.

Current Sensor

Model: INA3221AIRGVR

ADDRESS: 0x40 and 41

2 three channel INA3221 current sensors are provided using a 1 ohm shunt for the external 4-20mA Sensing and the internal SNS output from the amplification circuit.

SPI0

Ethernet

Model: W5100s

The W5100s ethernet module from wiznet is incorporated into the W5100s-EVB-PICO board and uses SPI0

SPI1

7-segment 8 digit display

Model: MAX7219

DN: SPI1 TX GP11

CS: SPI1 Csn GP13

CLK: SPI1 SCK GP10

The MAX7219 IC is used to drive the 7-segement-8-digit display and display data periodically.

UART:

GPIO4 - DI, GPIO5 - RO, GPIO14 - DE

RS-485 unit

Model:MAX3485ESA

RE/DE: GP4

RO: GP5

DI: GP14

RS-485 is used for Modbus RTU, the Modbus function is described later.

GPIO:

GPIO0

Solid-state relay

Pin output, SOV1. Set via Modbus writable coil.

GPIO1

Solid-state relay

Pin output, SOV2. Set via Modbus writable coil.

GPIO2

Solid-state relay

Pin output, SOV3. Set via Modbus writable coil.

GPIO3

Solid-state relay

Pin output, SOV4. Set via Modbus writable coil.

GPIO6

Reset 0

Pin output, RST0. Set via Modbus writable coil, when output set to 1 the application circuit on SNS0 is enabled and the peak hold circuit used the 220uF cap to hold the value. When pin is down the 220uF cap is grounded.

GPIO7

Reset 1

Pin output, RST1. Set via Modbus writable coil, when output set to 1 the application circuit on SNS1 is enabled and the peak hold circuit used the 220uF cap to hold the value. When pin is down the 220uF cap is grounded.

GPIO8

I2C SDA

GPIO9

I2C SDL

GPIO15

Trigger PO-2 (write)

Pin output, Trig. Set via Modbus writable coil, when output set to 1 the Trigger PO-2 set true. This is bypass of the PO-02 input. When manually set the GPIO28 is used to read the status of PO-2.

GPIO22

LED-GPIO22

Pin output, LED. Set via Code / Modbus writable coil, when output set to 1 LED provides visual indication.

GPIO26

LED-GPIO26

Pin output, LED. Set via Code / Modbus writable coil, when output set to 1 LED provides visual indication.

GPI027

LED-GPIO27

Pin output, LED. Set via Code / Modbus writable coil, when output set to 1 LED provides visual indication.

GPIO28

Trigger PO-2 (read)

Pin input, Trig. read coil, used to read the status of PO-2.

SCHEMATIC

A schematic is attached Schematic 18.16.

SOFTWARE

IDE / Language

The preferred IDE is VS Code as recommended by the PI Organization with C++ code.

Arduino is supported and many examples exist written using the Arduino IDE, if it is not possible to use the IDE then the Arduino IDE can be used. Provided that the code is documented accordingly.

Sensor values

Sensors shall be configured to return values that can be stored as global variables. These variables are poled by the Modbus class.

MODBUS

Modbus RTU

Using the MAX485 chip and a Modbus library the firmware should report coil status, report registers from sensors and software variables and be able to be writer to coils and registers from a master PLC over RS485.

Modbus RTU Address

The Modbus RTU address shall be static and set in firmware, the address shall be periodical displayed on the display.

TEST CASE

For the initial test case global variables will be set on the RP2040 for bits and integers. These values will be displayed via serial from the while loop and be accessible as read coils / registers in Modbus. When the PLC reads the Modbus value and output to the serial shall print the activity.

For write coils the serial shall print Modbus function and the value of the coil register

Modbus TCP

Using the W5100s ethernet chipset and a Modbus library the firmware should report coil status, report registers from sensors and software variables and be able to be writer to coils and registers from a master PLC over TCP.

Modbus TCP Address

The Modbus TCP address shall be static in the range of 192.168.1.2 – 192.168.1.100, subnet 255.255.255.0 and set in firmware, the address shall be periodical displayed on the display.

TEST CASE

For the initial test case global variables will be set on the RP2040 for bits and integers. These values will be displayed via serial from the while loop and be accessible as read coils / registers in Modbus. When the PLC reads the Modbus value and output to the serial shall print the activity.

For write coils the serial shall print Modbus function and the value of the coil register

LIBRARIES

Libraries should be installed for all sensors and tested, where possible functions should be written to seamlessly read sensors, read EEPROM, write EPPROM. This can be expanded further.

RTOS

RTOS or dual core application will be considered once the Bare Metal application has been completed.