# Modbus Reading for Schneider PowerLogic PM8000

This is a specification and implementation of the Arduino-based Modbus data logger for the Schneider PowerLogic PM8000. This software is designed for Vivarox EMS and only Vivarox has right to use and modify this software.

## Arduino Implementation:

This project uses an Arduino MEGA 2560 to connect to Modbus devices, read information, and log it onto an SD card with timestamps. The MEGA 2560 is particularly well-suited for this project due to its increased memory capacity and multiple hardware serial ports.

### Hardware needed:

1. Arduino Board Required: Arduino MEGA 2560 (selected for its 256KB flash memory, 8KB SRAM, and multiple hardware serial ports)
    - Arduino MEGA @ R377.20
2. RS485 to TTL Module Allows communication between the Arduino and Modbus devices using the RS485 protocol.
    - RS485 Module (TTL -> RS485) @ R25.30
    - MAX485 Bus Transceiver (4 Pack) @ R16.00
3. SD Card Module Allows the Arduino to read from and write data to an SD card.
    - Micro SD Card Module @ R25.00
4. RTC Module To keep track of the current date and time, even when the Arduino is powered off.
    - DS3231 Real Time Clock Module @ R55.20
5. Power Supply To power the Arduino and connected peripherals.
    - AC Adapter 9V with barrel jack @ R60
6. LED Indicators Two LEDs for status indication (not included in original cost estimate).

### Wiring for MEGA 2560

#### RS485 Module to Arduino MEGA:

1. RO (Receiver Output) to MEGA RX1 (pin 19) - Using Hardware Serial1
2. DI (Driver Input) to MEGA TX1 (pin 18) - Using Hardware Serial1
3. DE (Driver Enable) & RE (Receiver Enable) to MEGA digital pin 4
4. VCC to 5V on MEGA
5. GND to GND on MEGA
6. A & B (RS485 differential pair) to Modbus device

#### SD Card Module to Arduino MEGA:

1. VCC to 5V on MEGA
2. GND to GND on MEGA
3. MOSI to MOSI (pin 51)
4. MISO to MISO (pin 50)
5. SCK to SCK (pin 52)
6. CS (Chip Select) to digital pin 53

#### RTC Module to Arduino MEGA:

1. VCC to 5V on the MEGA
2. GND to GND on the MEGA
3. SDA to SDA (pin 20)
4. SCL to SCL (pin 21)

#### LED Indicators:

1. LED A to digital pin 3
2. LED B to digital pin 5

### Software

- Modbus Library: ModbusMaster
- SD Library: SdFat (more advanced than the standard SD library)
- RTC Library: RTClib by Adafruit

### Implementation Details

1. Modbus Configuration:
    - Slave ID: 1
    - Baud Rate: 9600
    - Register map: Defined in separate "register_map.h" file
    - Using Hardware Serial1 for improved reliability
2. Data Logging:
    - Frequency: Readings taken every second
    - File Format: CSV (Comma-Separated Values)
    - Filename: "log_YYYYMMDD.csv" (generated daily based on current date)
    - Data Structure: Timestamp, followed by register values
    - Header Row: Includes register addresses for easy identification
    - Larger buffer sizes possible due to MEGA's increased memory
3. Register Types Supported:
    - Float (32-bit)
    - Integer (32-bit)
    - Long (64-bit)
    - String (up to 20 characters)
    - Multiple register reads supported simultaneously due to larger memory
4. Error Handling and Status Indication:
    - LED A: Indicates successful data writing and transmission
    - LED B: Indicates errors (e.g., SD card issues, RTC problems, Modbus communication errors)
    - Serial output for debugging (115200 baud possible due to hardware serial)
5. Special Features:
    - Automatic creation of new log file on date change
    - Header row written only once per file
    - Robust error handling for SD card, RTC, and Modbus communication
    - Support for larger register maps due to increased memory
    - Possibility to implement multiple Modbus device communication using additional hardware serial ports

### Programming Workflow

1. Initialize hardware (RTC, SD card, RS485 module)
2. Set up Modbus communication parameters using Hardware Serial1
3. Enter main loop:
    - Read current time from RTC
    - Read data from Modbus registers (larger batches possible)
    - Write timestamped data to SD card
    - Handle any errors and provide status indication via LEDs
    - Delay for 1 second before next reading

## MEGA-Specific Advantages

- More memory allows reading more registers simultaneously
- Hardware serial ports provide more reliable communication
- Additional I/O pins available for expansion
- Possibility to monitor multiple Modbus devices using different serial ports
- Larger program space allows for more complex error handling and data processing

- No need to be selective about registers due to memory constraints
- Can implement additional features like local display or network connectivity

## Best Practices

- Use Hardware Serial1 (pins $^{18}/_{19}$) for primary Modbus communication
- Additional Modbus devices can use Serial2 (pins $^{16}/_{17}$) or Serial3 (pins $^{14}/_{15}$)
- Take advantage of the extra memory to implement robust error checking
- Consider using the additional I/O pins for status displays or control interfaces
- You can include all registers from your register map without memory concerns
- Consider implementing a circular buffer for temporary data storage