OPTA-PROG-MODBUS7M-PROGARDUINO

PT-BR

Programação OPTA com 7M (Modbus) - Programação Arduino

1. Informações Gerais

- Versão / Revisão: Revisão 1
- Data de Desenvolvimento: 10/05/2023
- Autor: Daniel Arcos
- Linguagem: C++
- Plataforma / Hardware: Finder OPTA

2. Objetivo da Programação

Esse programação tem como finalidade estabelecer a conexão entre o OPTA e o 7M (Medidor de energia da Finder), utilizando o protocolo Modbus, permitindo a leitura de todos os parâmetros disponíveis.

3. Requisitos e Dependências

- Instalar o Arduino IDE em sua última versão disponível

- Instalar os drivers de Hardware do OPTA no Arduino IDE
- Instalar biblioteca do OPTA
- Instalar as bibliotecas descritas na programação

4. Passo a Passo da Implementação

1. Incluir bibliotecas necessárias
2. Configurar os parâmetros Modbus na programação
3. Configurar os parâmetros Modbus 7M
4. Verificar tabela de registradores Modbus do 7M
5. Colocar endereços do registradores na programação

5. Código completo

O código completo está somente no Github por conta do tamanho da programação.

6. Explicativo

Este código implementa um sistema de gerenciamento de energia que monitora e controla o consumo de energia de dispositivos conectados, utilizando um medidor de energia que se comunica via protocolo Modbus RTU.

Visão Geral

O sistema tem as seguintes funcionalidades principais:

1. Monitoramento de parâmetros elétricos (tensão, corrente, potência, energia)
2. Controle de dispositivos baseado em perfis de consumo definidos pelo usuário

3. Integração com Arduino IoT Cloud para configuração remota
4. Protocolo Modbus RTU para comunicação com medidor de energia

Estrutura Principal

Variáveis Globais

- `Device_1`, `Device_2`: Estados dos dispositivos controlados
- Variáveis para armazenar parâmetros elétricos (V_actual, A_avg, W_max, etc.)
- `user_profile`: Estrutura que armazena os limites de operação definidos pelo usuário

Funções Principais

1. **relay_Trigger()**: Controla relés baseado em condições de consumo
   - Compara valores atuais com limites definidos
   - Ativa/desativa relés conforme necessário
2. **consumption_profile()**: Define os parâmetros iniciais de operação
   - Margem de operação, limite de potência, limite de consumo de energia
3. **energy_distro_ctrl()**: Lógica principal de controle de energia
   - Gerencia dispositivos baseado nos parâmetros medidos
   - Implementa condições de segurança e override remoto
4. **Funções modbus_com_**: Conjunto de funções para comunicação com medidor de energia
   - Recuperam valores atuais, médios, máximos e mínimos
   - Monitoram consumo de energia (Wh e varh)
5. **IoT Cloud Integration**:
   - Funções de callback para atualização remota de parâmetros
   - Configuração de conexão com a nuvem

Setup e Loop Principal

- **setup()**: Inicializa portas, comunicação serial, Modbus RTU, LEDs e IoT Cloud
- **loop()**: Atualiza dados da nuvem, executa controle de energia e gerencia consumo
- **modbus_line()**: Função executada em paralelo para leitura contínua do medidor

Detalhes de Implementação

Comunicação Modbus RTU

O código implementa um cliente Modbus RTU para se comunicar com um medidor de energia (modelo 7M.24). As funções de leitura recuperam:

- Parâmetros elétricos instantâneos
- Médias, máximos e mínimos
- Consumo acumulado de energia

Controle de Energia

O sistema usa uma abordagem hierárquica:

1. Verifica se o consumo está dentro dos limites definidos
2. Aplica condições primárias e secundárias para ativação de dispositivos
3. Permite override remoto via IoT Cloud
4. Fornece alertas quando os parâmetros se aproximam dos limites

Segurança

- Margens de operação configuráveis (10% padrão)
- Desligamento automático em caso de sobrecarga
- Monitoramento contínuo de parâmetros

Fluxo de Operação

1. Inicializa hardware e comunicações
2. Configura parâmetros iniciais de operação
3. Inicia leitura contínua do medidor de energia

4. Aplica lógica de controle baseada nos valores lidos
5. Atualiza estado dos dispositivos controlados
6. Mantém sincronização com IoT Cloud

EN

OPTA Programming with 7M (Modbus) - Arduino Programming

1. General Information

- Version/Revision: Revision 1
- Development Date: 10/05/2023
- Author: Daniel Arcos
- Language: C++
- Platform/Hardware: Finder OPTA

2. Programming Objective

This programming aims to establish the connection between the OPTA and the 7M (Finder energy meter), using the Modbus protocol, allowing the reading of all available parameters.

3. Requirements and Dependencies

- Install the latest version of Arduino IDE
- Install the OPTA hardware drivers in Arduino IDE
- Install the OPTA library
- Install the libraries described in the programming

4. Implementation Step by Step

1. Include necessary libraries
2. Configure Modbus parameters in the programming
3. Configure 7M Modbus parameters
4. Check 7M Modbus register table
5. Enter register addresses in the programming

5. Full code

The complete code is only on Github due to the size of the program.

6. Explanatory

This code implements a power management system that monitors and controls the power consumption of connected devices, using a power meter that communicates via the Modbus RTU protocol.

Overview

The system has the following main functionalities:

1. Monitoring of electrical parameters (voltage, current, power, energy)

2. Control of devices based on user-defined consumption profiles
3. Integration with Arduino IoT Cloud for remote configuration
4. Modbus RTU protocol for communication with energy meter

Main Structure

Global Variables

- `Device_1`, `Device_2`: States of the controlled devices
- Variables to store electrical parameters (V_actual, A_avg, W_max, etc.)
- `user_profile`: Structure that stores the operating limits defined by the user

Main Functions

1. **relay_Trigger()**: Controls relays based on consumption conditions
- Compares current values with defined limits
- Enables/disables relays as needed
1. **consumption_profile()**: Defines initial operating parameters
- Operating margin, power limit, energy consumption limit
1. **energy_distro_ctrl()**: Main energy control logic
- Manages devices based on measured parameters
- Implements safety conditions and remote override
1. **modbus_com_Functions**: Set of functions for communicating with energy meters
- Retrieve current, average, maximum and minimum values
- Monitor energy consumption (Wh and varh)
1. **IoT Cloud Integration**:
- Callback functions for remote parameter updates
- Cloud connection configuration

Setup and Main Loop

- **setup()**: Initializes ports, serial communication, Modbus RTU, LEDs and IoT Cloud

- **loop()**: Updates data from the cloud, performs energy control and manages consumption
- **modbus_line()**: Function executed in parallel for continuous meter reading

Implementation Details

Modbus RTU Communication

The code implements a Modbus RTU client to communicate with an energy meter (model 7M.24). The reading functions retrieve:

- Instantaneous electrical parameters
- Averages, maximums and minimums
- Accumulated energy consumption

Energy Control

The system uses a hierarchical approach:

1. Checks if consumption is within defined limits
2. Applies primary and secondary conditions for device activation
3. Allows remote override via IoT Cloud
4. Provides alerts when parameters approach limits

Security

- Configurable operating margins (10% default)
- Automatic shutdown in case of overload
- Continuous monitoring of parameters

Operation Flow

1. Initializes hardware and communications
2. Configures initial operating parameters
3. Starts continuous reading of the energy meter
4. Applies control logic based on the read values
5. Updates the status of controlled devices

6. Maintains synchronization with IoT Cloud