

Embarcadero Conference 2024

Inovação faz parte do nosso DNA!

Comunicação Industrial Modbus com RAD Studio

Emerson Maurício de Almeida Alves

Uso da biblioteca libmodbus em C++ Builder para desenvolver aplicações robustas de comunicação industrial. Exemplo medidor de energia.



Comunicação Industrial

- No passado cada fabricante tinha seu protocolo proprietário.

MODBUS era o protocolo da empresa **MODICOM**

- Atualmente OPC veio resolver o problema.

- OPC

- Padrão de comunicação industrial atual adotado por toda industria.

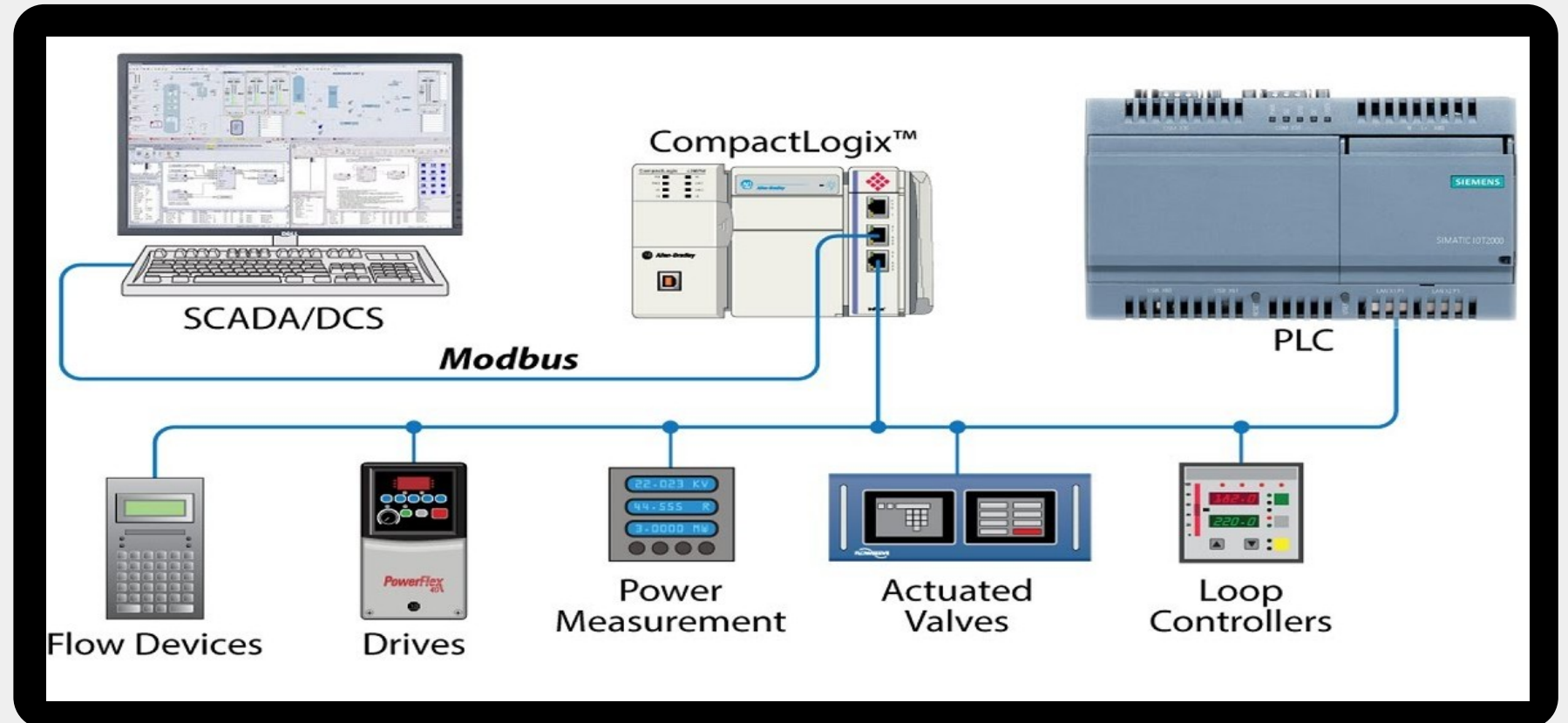
- OPC – **Ole for Process Control** – originário da microsoft com DCOM (aplicações distribuídas)

- OPC - **Open Platform Communications** - multiplataforma – definição atual



Modbus

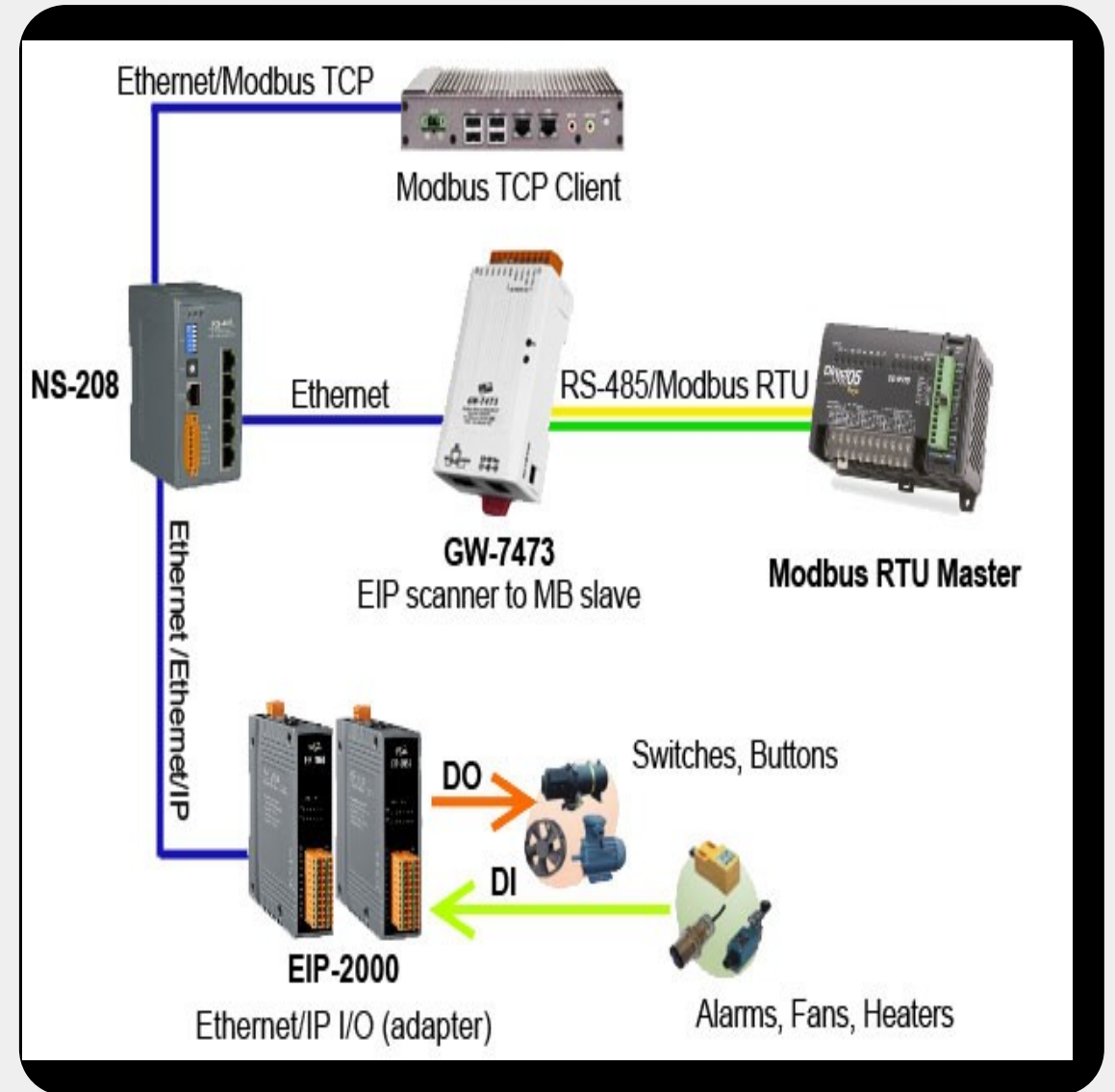
- Modbus é um protocolo de comunicação de dados utilizado em sistemas de automação industrial.
- Desenvolvido pela Modicon (atualmente parte da Schneider Electric) em 1979.
- Conecta e comunica com dispositivos em um sistema de controle.



RTU e Ethernet

Modbus RTU: Serial RS232 ou RS485 codificação binária, compacta e eficiente.

Modbus TCP: Ethernet, usando o protocolo TCP/IP.



Protocolo

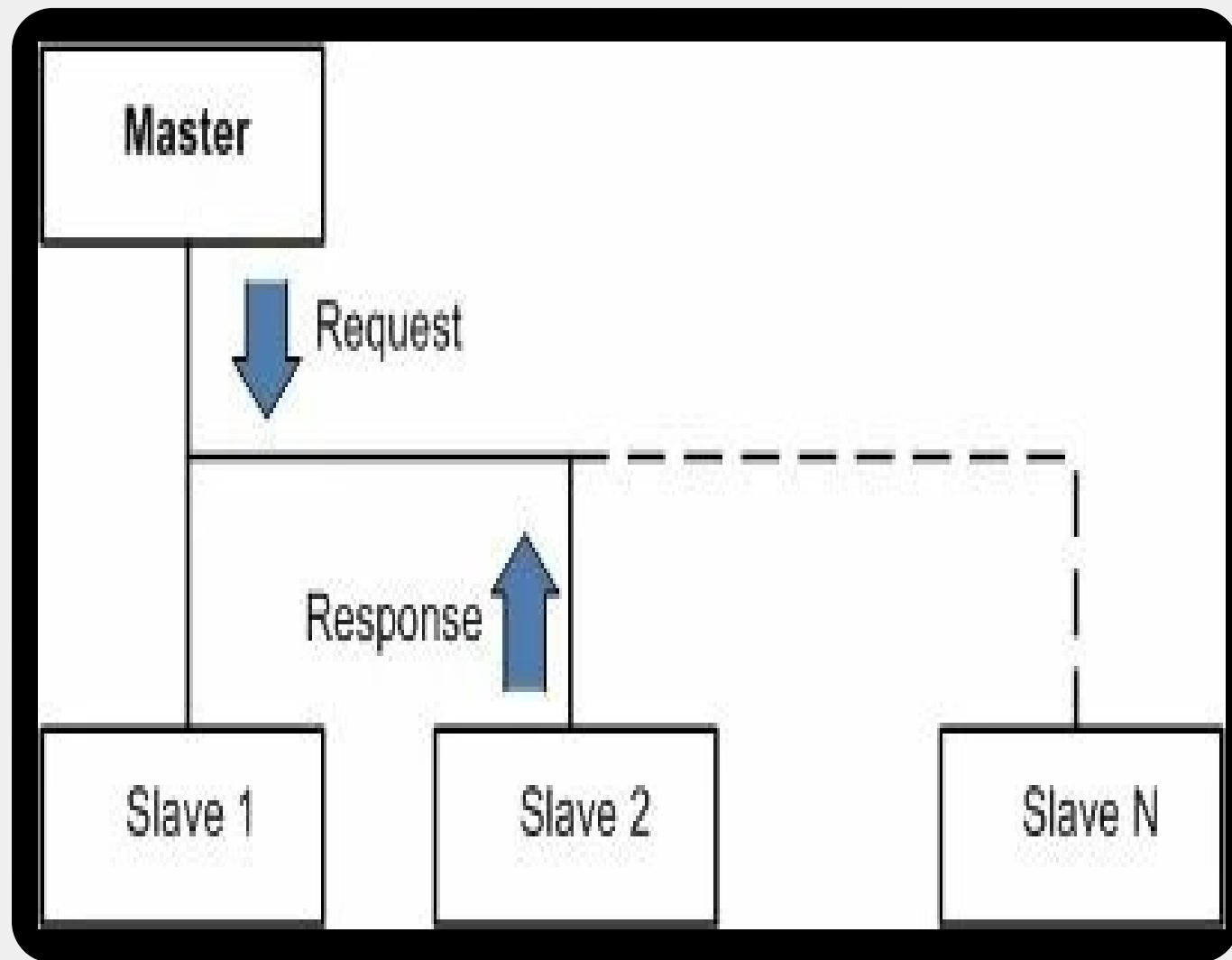
Cada dispositivo possui um endereço único. Mestre inicia a comunicação, o escravo do endereço responde.

Tipos de Dados

Holding Registers: Registros de leitura e escrita para armazenar dados.

Input Registers: Registros de leitura apenas para dados de entrada.

Coils: Dados de saída binária para controle de dispositivos.



Libmodbus

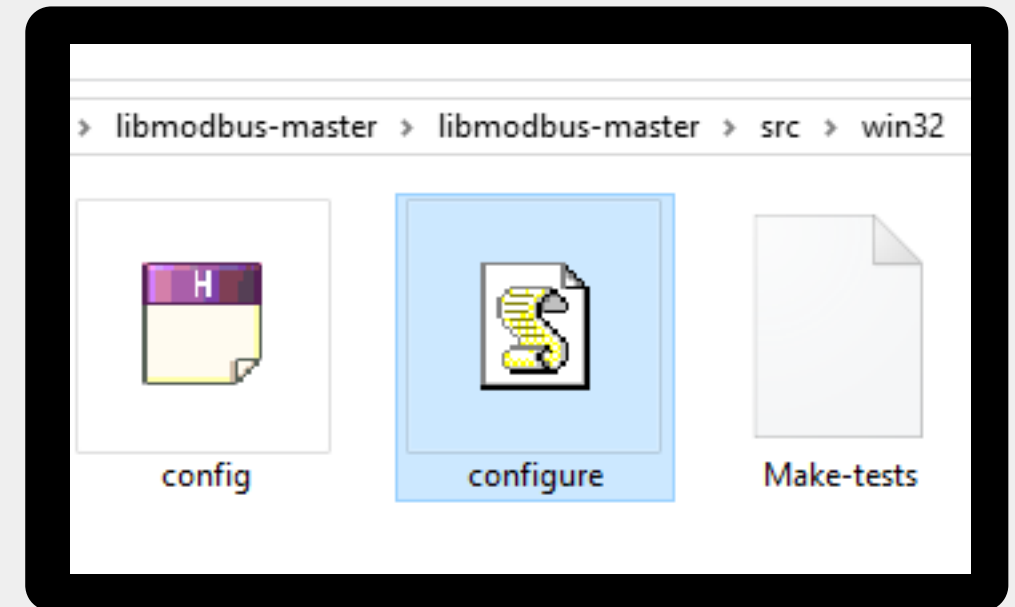
- **Libmodbus:** é uma biblioteca em C++ de código aberto para implementar o protocolo Modbus, suporta Modbus RTU e TCP.
- **Operações de Leitura e Escrita:** Permite realizar operações básicas como leitura e escrita em registros e coils.
- **Gestão de Erros:** Inclui mecanismos para lidar com erros e exceções na comunicação.
- **Facilidade de Uso:** Simplifica a implementação do Modbus com funções e ferramentas para comunicação.

Link: <https://libmodbus.org/>



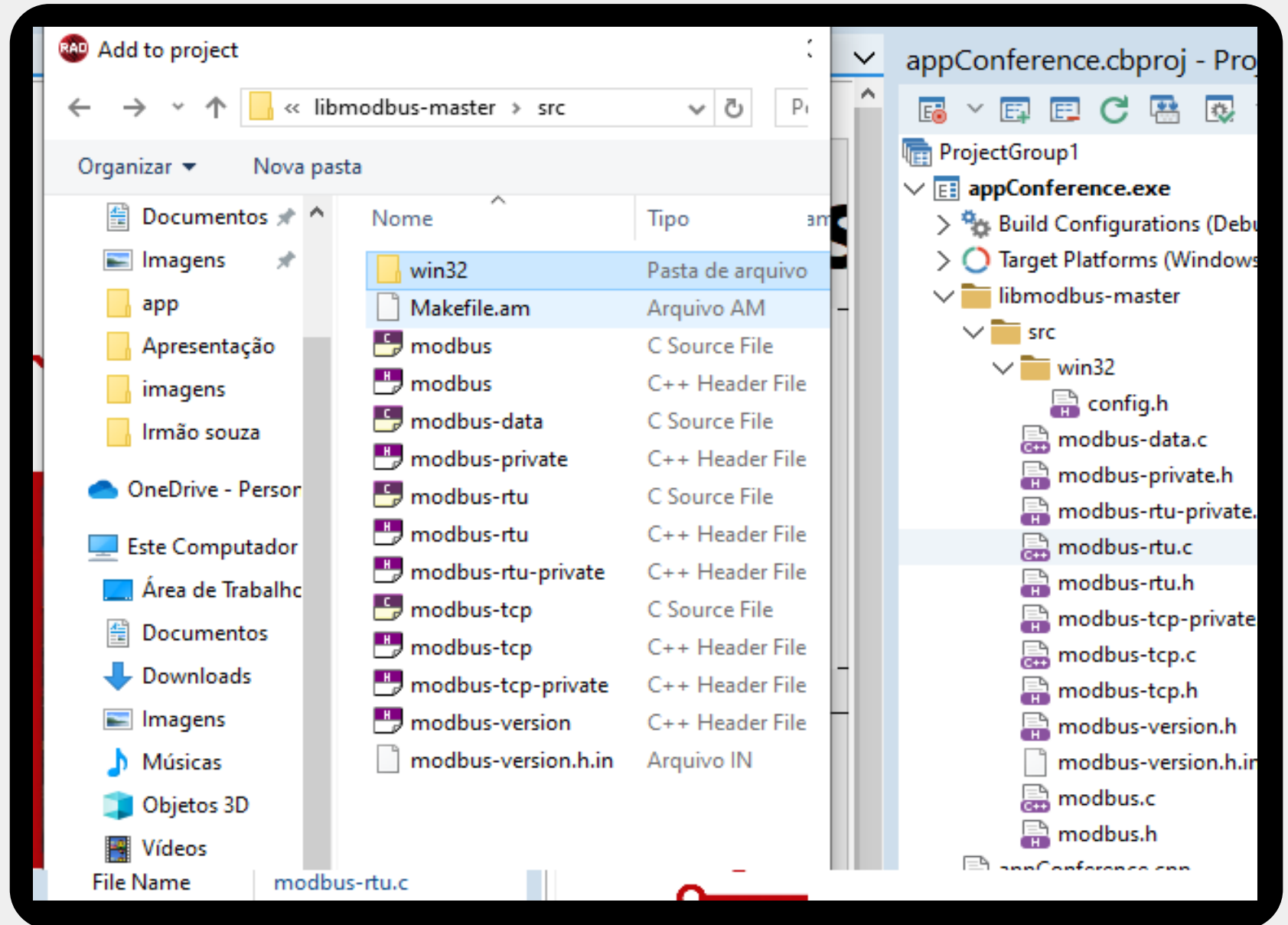
Libmodbus em C++ Builder - Preparo

- Baixe a biblioteca compactada no Github:
- <https://github.com/stephane/libmodbus>
- Descompacte, no diretório **libmodbus-master\src\win32**
- Execute o arquivo **configure.js** para gerar o **config.h**



Libmodus em C++ Builder – Adicionar bibliotecas

- Adicione ao projeto todos arquivos **.C** e **.h** da pasta **\src**
- Adicione o arquivo **config.h** da pasta **src\win32**



Libmodus em C++ Builder – Correção Identificador Compilador

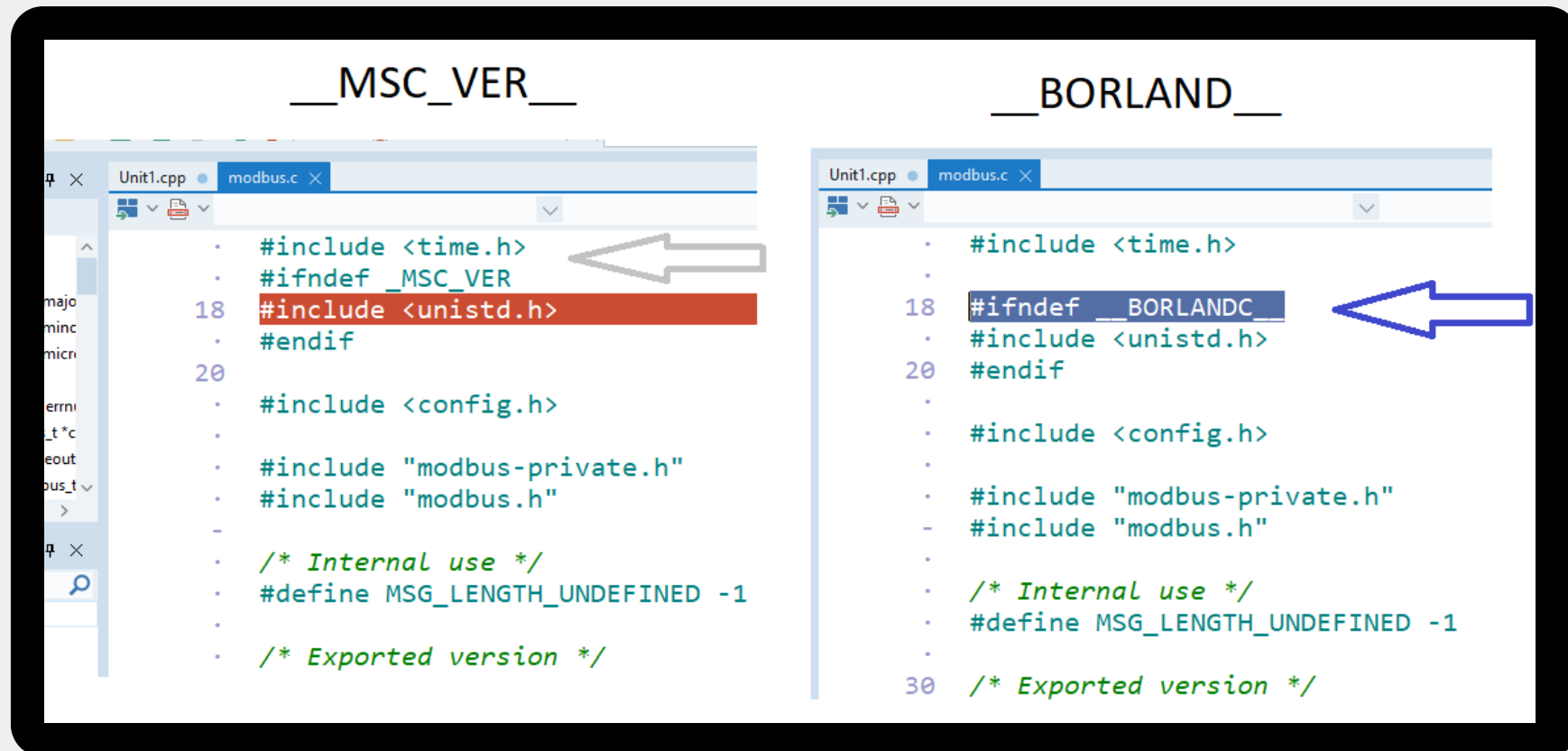
- Na compilação da biblioteca ocorrerá um erro de arquivo **unist.h** inexistente.

É necessário trocar, na macro, o identificador de compilador :

__MSC_VER

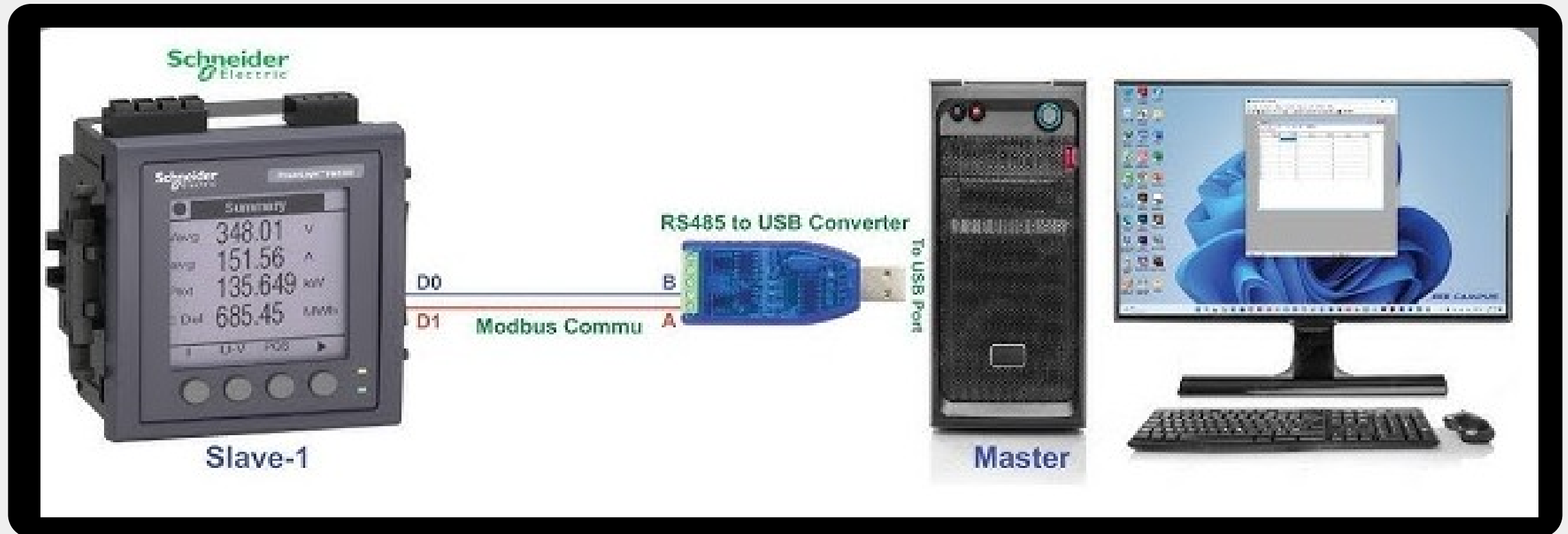
por:

__BORLAND__



Aplicação Exemplo – Medidor de Energia Industrial PM5330

- **PM5330:** Medidor industrial da [Schneider Electric](https://www.schneider-electric.com) - Empresa que adquiriu a Modicon
- Conversor USB para RS485 para barramento de até 1200m.



Aplicação Exemplo – Endereço Registradores

- Endereço das memórias é fornecido pelo fabricante do equipamento, mas a forma de consulta é padrão no protocolo, nesse caso será leitura de **Holding Registers**.

registradores PM5330.ods — LibreOffice Calc

1	Sub Cat 2	Description	Register	Units	Size (INT 16)	Data Type	Access
201		Current Avg	3010	A	2	FLOAT32	R
202	Current Unbalance		3012	—		—	—
203		Current Unbalance A	3012	%	2	FLOAT32	R
204		Current Unbalance B	3014	%	2	FLOAT32	R
205		Current Unbalance C	3016	%	2	FLOAT32	R
206		Current Unbalance Worst	3018	%	2	FLOAT32	R
207	Voltage		3020	—		—	—
208		Voltage A-B	3020	V	2	FLOAT32	R
209		Voltage B-C	3022	V	2	FLOAT32	R
210		Voltage C-A	3024	V	2	FLOAT32	R

Endereços registradores fornecidos pela Schneider Electric – Medidor PM5330

Aplicação Exemplo – Função Conecta

- Chama as funções:
- **modbus_new_rtu** ou **modbis_new_tcp**

- **Importante:**

- **try...catch**

- Bloco protegido evita travamento em conexão serial.

```
bool TformMed::conectaModbus(bool RTU){  
    try{           //bloco protegido  
        char szBuffer[200];  
        int rc;  
  
        //API que cria nova conexão MODBUS  
        if (RTU)  
            ctx = modbus_new_rtu("COM4", 19200, 'N', 8, 1);  
        else  
            ctx = modbus_new_tcp("127.0.0.1", 502);  
  
        //Executa a conexão  
        if (modbus_connect(ctx) == -1) {  
            sprintf(szBuffer, "Falha na conexão: %s\n", modbus_strerror(errno));  
            modbus_free(ctx);  
            return false;↓  
        }  
    }  
    catch(...)  
    {  
        desConecta();  
    }  
}
```

Aplicação Exemplo – Função Ler Registradores

Holding Registers: Leitura dos registradores

- Chama a função:
- `modbus_read_registers`
- **CUIDADO:**
- **Timeout Resposta**
- pode gerar travamentos na aplicação.

```
void __fastcall TFormMed::readModBus(int add,int n, uint16_t *reg ){
    int rc;
    char szBuffer[200];
    /*          Atenção - Risco de atrasos na Aplicação

    Função pode gerar atrasos pelo timeout do Modbus
    Não deve ser chamadas em eventos de timer
    Deve ser chamada em Threads - ou processo de segundo plano
    */
    rc =modbus_read_registers(ctx, add,n, reg); //carrega os valores no vetor
                                              //passado como parâmetro da função

    if (rc == -1) {
        sprintf(szBuffer, "Connection failed: %s\n", modbus_strerror(errno));
        Memo1->Lines->Add(szBuffer );
        Memo1->Lines->Add("err= "+ IntToStr(errno));
        return ;↓
    }

    for (int i=0; i < rc; i++) {
        Memo1->Lines->Add("Reg: " + IntToStr(i+add)+" - "+FormatFloat("00", reg[i]));
    }
}
```

Aplicação Exemplo – Função Ler Registradores

Função readValues(): função de leituras para registradores não sequenciais.



```
void __fastcall TformMed::readValues()
{
    if(!ctx)
        return;↓

    int n=2;
    uint16_t reg[2];

    readModBus( 2999, n, reg );
    iMed= int16ToFloat(reg[0], reg[1]) ;

    readModBus( 3109, n, reg );
    freq= int16ToFloat(reg[0], reg[1]) ;

    readModBus( 3019, n, reg );
    vMed= int16ToFloat(reg[0], reg[1]) ;

    readModBus( 3053, n, reg );
    potA= int16ToFloat(reg[0], reg[1]) ;

}
```


Aplicação Exemplo – Multitarefa (Threads)

Thread: Processo independente da aplicação principal para leitura no barramento RS485.

Atrasos no processo devido a aguardar tempo de resposta (timeout) não atinge a aplicação principal.



```
frmPrincipal.cpp  thdReadModbus.cpp  readModbus::Execute

*
*  __fastcall readModbus::readModbus(bool CreateSuspended)
*      : TThread(CreateSuspended)
*  {
*  }
*  //-----
*  void __fastcall readModbus::Execute()
30  {
*      FreeOnTerminate=true;  //Thread será destruída quando termina o processo
*      bool ok=true ;
*      while(!Terminated || ok)
34  {
*          Sleep(1000);
*          formMed->readValues();  //chamada de leitura modbus nesse processo
*      }
*  }
*  //-----
```

Aplicação Exemplo – Multitarefa (Threads)

Thread: Processo independente realiza a leitura no barramento RS485, Modbus RTU.

Start Thread

```
void __fastcall TformMed::startThread()
{
    if (threadStart)
        return;
    if( pThreadRead == NULL)
    {
        pThreadRead = new readModbus(false);
        threadStart = true; // set the thread flag
    }
}
```

Stop Thread

```
bool __fastcall TformMed::stopThread()
{
    DWORD dwExitCode;
    if (pThreadRead != NULL )
    {
        GetExitCodeThread((void*)(pThreadRead->Handle), &dwExitCode);
        if (dwExitCode == STILL_ACTIVE)
            TerminateThread((void*)(pThreadRead->Handle), dwExitCode);
        pThreadRead->Terminate();
        delete pThreadRead ;
        pThreadRead= NULL;
    }

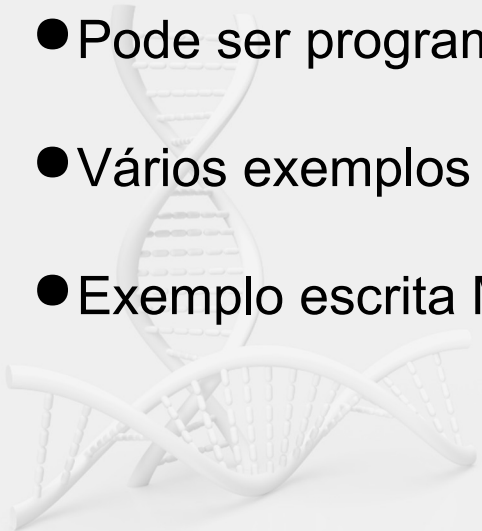
    if( pThreadRead == NULL)
    {
        threadStart = false;
        return true;
    }
    else
        return false;
}
```



IoT- Microcontrolador – ESP32 – Modbus TCP/IP

ESP32: Microcontrolador 80 MHz da Espressif

- WiFi, Bluetooth, BLE, SPI, I2C, USART- embutido
- Ideal para IoT
- Já possui biblioteca nativa para Modbus
- Custo baixo
- Pode ser programado na IDE Arduino
- Vários exemplos de códigos na Internet
- Exemplo escrita Modbus – escrita binária



Obrigado!

Embarcadero Conference 2024

Inovação faz parte do nosso DNA!




Quer me ver na
#ECON25?
Acesse o QRCode
e avalie minha palestra!



Emerson Alves

in [linkedindopalestrante](#)

 Emerson.alves@ifnmg.edu.br

 [\(38\) 9 9151 4981](tel:(38)991514981)

