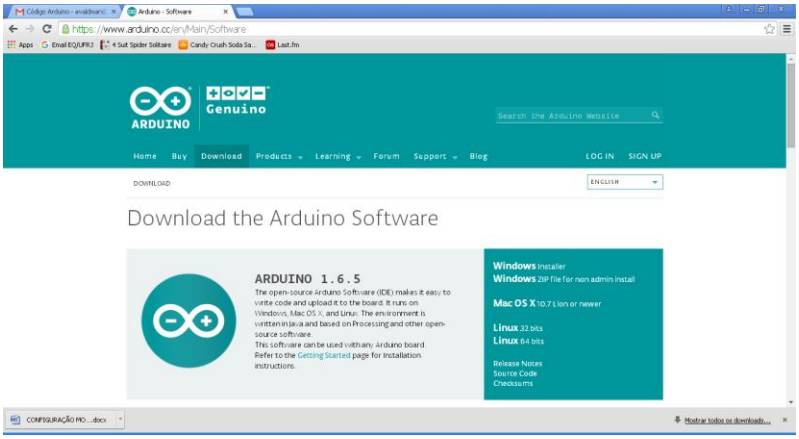
Arduino

Software de Configuração

Arduino Software 1.6.5 - https://www.arduino.cc/

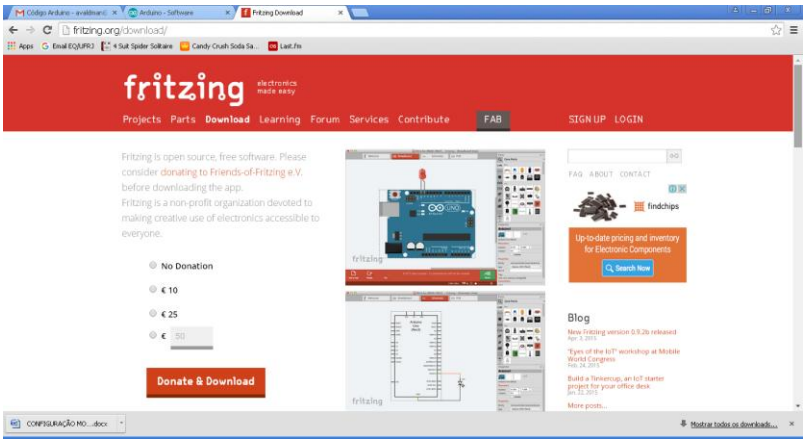


Valdman, A., Folly, R., 2015

Arduino

Software Auxiliar – Diagrama Elétrico

Fritzing Software - http://fritzing.org/home/



Valdman, A., Folly, R., 2015

Código Básico

Bibliotecas Externas

- Registro de utilização das bibliotecas <#include>:

  Exs.:

  – Medidor de Temperatura:

  ```
  #include <OneWire.h>
  #include <DallasTemperature.h>
  ```

  – Comunicação Modbus:

  ```
  #include <SimpleModbusSlave.h>
  ```


Código Básico

Setup

- Definição de variáveis internas <#define>:

  Exs.:

  – Medidor de Temperatura:

Posição do primeiro sensor	#define ONE_WIRE_BUS 3
Precisao	#define TEMPERATURE_PRECISION 12

  – Comunicação Modbus (Range em UE):

Range do sensor para compatibilidade com o driver Modbus	#define tempMax 125 #define tempMin 0 #define sinMax 9999 #define sinMin 0

Código Básico

Setup

- Definições de variáveis iniciais (conexões físicas; comunicação):
  - Medidor de Temperatura:

via comm OneWire;	OneWire oneWire(ONE_WIRE_BUS)
Inicialização TT;	DallasTemperature sensors(&oneWire) – ***inicialização TT-OneWire***
	DeviceAddress t1 = {0x28, 0x13, 0xA3, 0xBC, 0x05, 0x00, 0x00, 0xD7};
Associa Num.Série	DeviceAddress t2 = {0x28, 0x81, 0x26, 0xBE, 0x05, 0x00, 0x00, 0x16};

  - Relés:

Pino do relé;	int relay1 = 4;

  - Comunicação Modbus (Definindo Variáveis/Endereços Driver):

A Ordem de declaração no código define os endereços no driver	enum
	{ //*
	TEMP_1_VAL, //40001
	TEMP_2_VAL, //40002
	TEMP_1_CONNECTED, //40003
	RELAY_1, //…
	HOLDING_REGS_SIZE //
	};
	unsigned int holdingRegs[HOLDING_REGS_SIZE];

Valdman, A., Folly, R., 2015 EQE-737 – Instrumentação e Automação Industrial de Processo Avançadas

Código Básico

Setup

- Setup de Inicialização:

```
void setup()
{
```

Relés	pinMode(relay1, OUTPUT);

Sensores de Temperatura	sensors.begin();
	sensors.setResolution(t1, TEMPERATURE_PRECISION);
	sensors.setResolution(t2, TEMPERATURE_PRECISION);

COMM Modbus	modbus_configure(&Serial, **9600, SERIAL_8N2, 2, 2**, HOLDING_REGS_SIZE, holdingRegs);
Valor Inicial(Saída)	holdingRegs[RELAY_1] = 1; //*

```
}
```

OBS.: Vide documento de configuração do driver de comunicação MB1 - iFix

Código Básico

Loop

- Loop de Atualização:

```
void loop()
{
```

Leitura das Temperaturas

```
sensors.requestTemperatures();
```

Atualiza entradas Modbus

```
holdingRegs[TEMP_1_VAL] =
   normalizaTemp(sensors.getTempC(t1));
holdingRegs[TEMP_2_VAL] =
   normalizaTemp(sensors.getTempC(t2));
holdingRegs[TEMP_1_CONNECTED] = sensors.isConnected(t1);
holdingRegs[TEMP_2_CONNECTED] = sensors.isConnected(t2);
```

Atualiza driver Modbus-iFix

```
modbus_update();
```

Atualiza saídas Modbus

```
digitalWrite(relay1, holdingRegs[RELAY_1]); //lowByte(
```

```
}
```


Código Básico

Loop

- Funções Auxiliares:

Converte Valores de Unidade de Engenharia para Faixa de COMM Modbus

```
float normalizaTemp(float tempIn) {

float sinOut = (((tempIn - tempMin) / ( tempMax) ) * (sinMax )) +
  sinMin;

return sinOut;
}
```