INSTITUTO FEDERAL
Santa Catarina

I2C
Inter-integrated circuit

Matéria: Sistemas Embarcados

Alunos: Marcone Louzada e Guilherme Roque

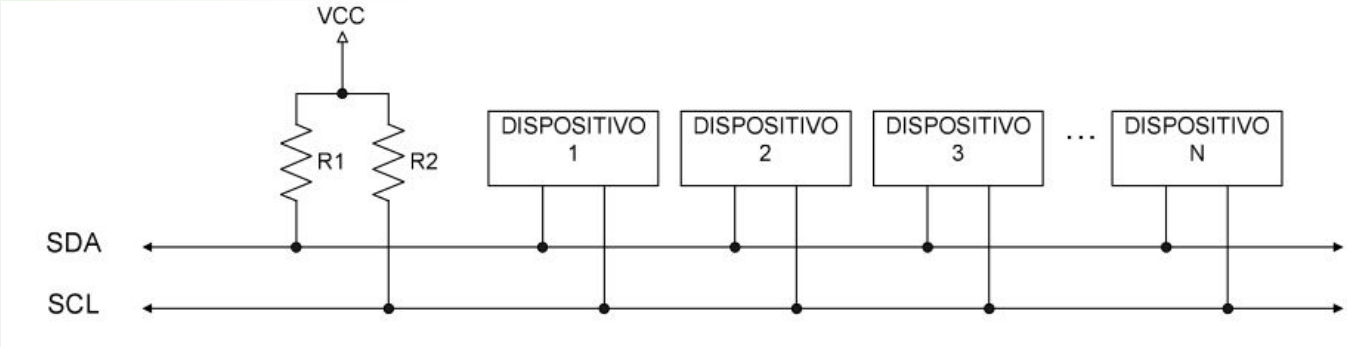
INSTITUTO FEDERAL
Santa Catarina

Revisão

- Protocolo de comunicação desenvolvido pela Philips nos anos 80;
- Nos microcontroladores da família ATmega, é denominado TWI (Tho-Wire Interface), para evitar o pagamento de royalties pelo uso do termo I2C;
- É encontrado em um grande número de circuitos integrados e periféricos, que são, em sua maioria:
  - Memórias EEPROM.
  - Sensores de temperatura.
  - Relógios de tempo real.
  - Conversores AD e DA;

Revisão

- A via de dados SDA transporta endereços, controle e dados, enquanto a via de clock SCL sincroniza o transmissor e receptor durante a transferência.
- O mestre gera o sinal de clock, inicia a transmissão e busca pelo do endereço do escravo. Ao escravo cabe responder e obedecer aos comandos do mestre.



INSTITUTO FEDERAL
Santa Catarina

Modelos de transmissão

Modelos de transmissão descritos no datasheet do ATmega 2560, as velocidades de transmissão disponíveis são de 100KHz e 400KHz.

- **S** - Condição de START;
- **Rs** - Condição de REPEATED START;
- **R** - Bit de leitura : nível lógico “0”;
- **W** - Bit de escrita : nível lógico “1”;
- **A** - Bit ACK : nível lógico “0”;
- **Ā** - Bit NACK : nível lógico “1”;
- **Data** - 8 bits de dados;
- **P** - Condição de STOP;
- **SLA** - Endereço do escravo;

INSTITUTO FEDERAL
Santa Catarina

Modelagem das classes

Durante a modelagem, foi observada a possibilidade de dividir a implementação em duas classes:

- TWI - Master
  - write();
  - receive();

- TWI - Slave
  - init();

INSTITUTO FEDERAL
Santa Catarina

Modelagem das classes

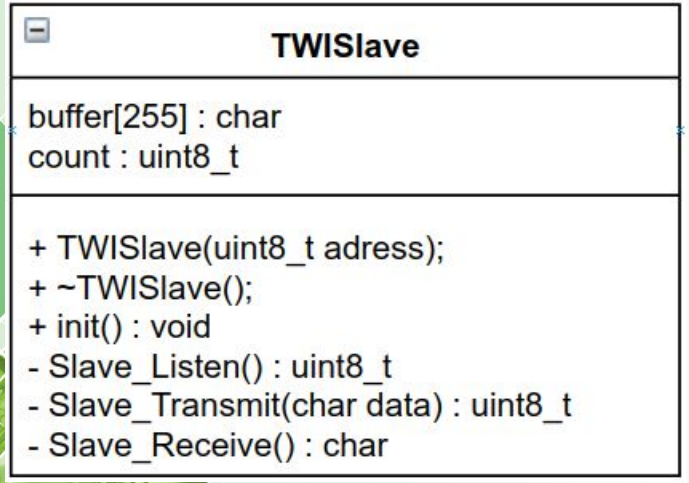
TWIMaster

+ TWIMaster(unsigned long freq);
+ ~TWIMaster();
+ write(const char* data,uint8_t bytes, uint8_t addr,uint8_t reg) : void
+ receive(char* data,uint8_t bytes, uint8_t addr,uint8_t reg) : void
- Start(char write_address) : uint8_t
- Repeated_Start(char read_address) : uint8_t
- Stop() : void
- Start_Wait(char write_address) : void
- Write(char data) : uint8_t
- Read_Ack() : char
- Read_Nack() : char

```
#ifndef TWIMASTER_H
#define TWIMASTER_H
#define F_CPU 16000000UL    /* Define CPU clock Frequency e.g. here its 16MHz */

#include <avr/io.h>
#include <util/delay.h>
#include <util/twi.h>

#define SLOW_TWI 100000UL
#define FAST_TWI 400000UL

class TWIMaster{
public:
    TWIMaster(unsigned long freq);
    ~TWIMaster();

    void write(const char* data,uint8_t bytes, uint8_t addr,uint8_t reg);

    void receive(char* data,uint8_t bytes, uint8_t addr,uint8_t reg);

private:
    uint8_t Start(char write_address);
    uint8_t Repeated_Start(char read_address);
    void Stop();
    void Start_Wait(char write_address);
    uint8_t Write(char data);
    char Read_Ack();
    char Read_Nack();
};

#endif  /* TWIMASTER_H */
```

INSTITUTO FEDERAL
Santa Catarina

Modelagem das classes




```
#ifndef TWISLAVE_H
#define TWISLAVE_H
#define F_CPU 16000000UL    /* Define CPU clock Frequency e.g. here its 16MHz */

#include <avr/io.h>
#include <util/delay.h>
#include <util/twi.h>
#include <string.h>
#include "UART.h"

class TWISlave{
public:
    TWISlave(uint8_t adress);
    ~TWISlave();
    void init();

private:
    uint8_t Slave_Listen();
    uint8_t Slave_Transmit(char data);
    char Slave_Receive();
    char buffer[255];
    uint8_t count;
};

#endif  /* TWISLAVE_H */
```

INSTITUTO FEDERAL
Santa Catarina

Teste da classe TWIMaster

```
int main(int argc, char** argv){
    EEPROM_1.set(0);                    /* Controla o bit A2 da EEPROM 1 */
    TWIMaster master(SLOW_TWI);
    char  input[] = "ARDUINO";
    char  input2[] = "EEPROM";
    char  output[10];
    char c;

    sei();
    while (1) {
        if (uart.has_data()) {
            c = uart.get();
            uart.puts("Escrevendo: ");
            uart.puts(input);
            _delay_ms(100);
            master.write(input,7,SLAVE_Address,0x00);
            master.receive(output,7,SLAVE_Address,0x00);
            _delay_ms(100);
            uart.put('\n');
            uart.puts("Leu: ");
            uart.puts(output);
            uart.put('\n');
            uart.puts("Escrevendo: ");
            uart.puts(input2);
            _delay_ms(100);
            master.write(input2,7,EEPROM_1_Address,0x00);
            master.receive(output,7,EEPROM_1_Address,0x00);
            uart.put('\n');
            uart.puts("Leu: ");
            uart.puts(output);
            _delay_ms(100);
        }
    }
    return 0;
}
```

INSTITUTO FEDERAL
Santa Catarina

Teste da classe TWISlave

```
#define SLAVE_Address      0xB0              /* 10110000 = 0xB0*/
int main(int argc, char** argv){

    TWISlave slave(SLAVE_Address);

    slave.init();

    return 0;
}
```

INSTITUTO FEDERAL
Santa Catarina

Referências

- https://ww1.microchip.com/downloads/en/devicedoc/atmel-2549-8-bit-avr-microcontroller-atmega640-1280-1281-2560-2561_datasheet.pdf
- https://www.dropbox.com/s/3tawx6im9az0wz7/Livro_AVR_2a_ed.pdf?dl=0