PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO GRANDE DO SUL

FACULDADE DE ENGENHARIA
DEPARTAMENTO DE ENGENHARIA ELÉTRICA

Professor Juliano D'Ornelas Benfica
**Laboratório de Processadores e Programação de Periféricos**

# Tutorial de uso do I2C por pinos no MSP430
# Exemplos para o Sensor de Temperatura TMP112 e para o Smart-Card

2011

## 1 Exemplo para o Sensor de Temperatura TMP112

### 1.1 Exemplo de software em C para o MSP430 conforme diagrama de escrita e configuração do sensor de temperatura

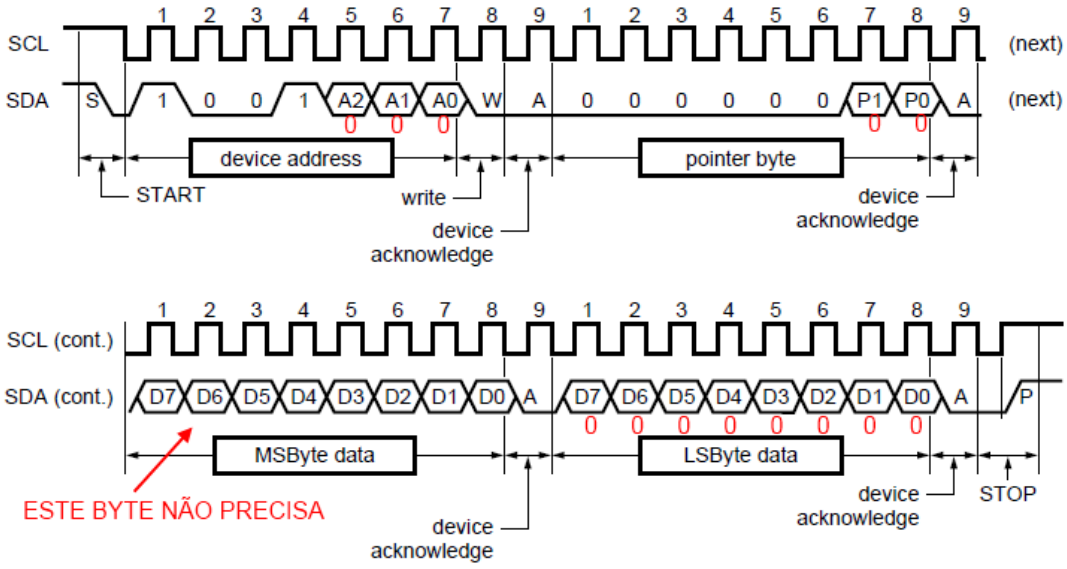


Figura 1: Escrita de um Byte

## 1.2 Conforme mostrado no diagrama anterior, é necessário construir algumas funções para a implementação do protocolo para a escrita e configuração do sensor, que são:

1. configura_sensor();
2. START_I2C();
3. STOP_I2C();
4. ENVIA_1_I2C();
5. ENVIA_0_I2C();
6. ACK_I2C();
7. NACK_I2C();
8. ENVIA_BYTE_I2C();
9. LE_BYTE_I2C();

### 1.2.1 Exemplo de como enviar a configuração completa para o sensor de temperatura:

```
void configura_sensor(void)
{
  char x=0;

  start_i2c();
  envia_1_i2c();
  envia_0_i2c();
  envia_0_i2c();
  envia_1_i2c();
  envia_0_i2c();
  envia_0_i2c();
  envia_0_i2c();
  envia_0_i2c(); //0 pois é escrita
  x = ack_i2c();
  envia_byte_i2c(0x00);
  x = ack_i2c();
  envia_byte_i2c(0x00);
  x = ack_i2c();
  stop_i2c();
}
```

### 1.2.2 Função Start

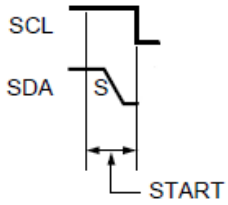


Figura 2: Diagrama do Start

```
//-- COLOCAR ESTES DEFINES NO CABEÇALHO DO PROGRAMA

#define SDA (1<<1) //P3.1 - PLACA 2011
#define SCL (1<<2) //P3.2 - PLACA 2011

#define SDA0 P3OUT &= ~SDA
#define SDA1 P3OUT |= SDA

#define SCL0 P3OUT &= ~SCL
#define SCL1 P3OUT |= SCL

void start_i2c(void) //I2C START
{
   SDA1;
   espera_us(500);//500 microsegundos
   SCL1;
   espera_us(500);
   SDA0;
   espera_us(500);
   SCL0;
   espera_us(500);
}
```

### 1.2.3 Função Stop

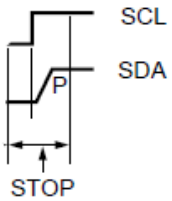


Figura 3: Diagrama do Stop

```
void stop_i2c(void) //I2C STOP
{
  SDA0;
  espera_us(500);//500 microsegundos
  SCL0;
  espera_us(500);
  SCL1;
  espera_us(500);
  SDA1;
  espera_us(500);
}
```

### 1.2.4 Função Envia 1

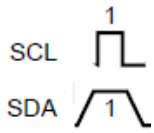


Figura 4: Diagrama do Envia 1

```
void envia_1_i2c(void) //Envia 1 pelo I2C
{
  SDA1;
  espera_us(500);
  SCL1;
  espera_us(500);
  SCL0;
  espera_us(500);
}
```

### 1.2.5 Função Envia 0

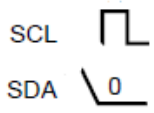


Figura 5: Diagrama do Envia 0

```
void envia_0_i2c(void) //Envia 0 pelo I2C
{
  SDA0;
  espera_us(500);
  SCL1;
  espera_us(500);
  SCL0;
  espera_us(500);
}
```

### 1.2.6 Função Lê ACK

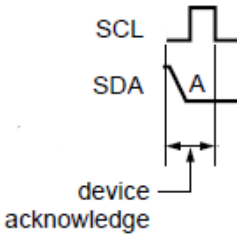


Figura 6: Diagrama do ACK

```
int ack_i2c(void)
{
  int x;
  P3DIR &= ~SDA; //FAZ SDA COMO ENTRADA
  SCL1;
  espera_us(500);
  x = (FIO0PIN & SDA)>>27; //LÊ O PINO
  SCL0;
  espera_us(500);
  P3DIR |= SDA; //FAZ SDA COMO SAÍDA
  return x; //se 0 ok, se 1 erro

}
```

### 1.2.7 Função envia NACK

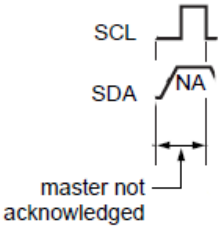


Figura 7: Diagrama do NACK

```
voi nack_i2c(void)
{
  envia_1_i2c(); // simplesmente enviar 1
}
```

### 1.2.8 Função envia Byte

**Esta função deverá ser construída pelo aluno.**

```
void envia_byte_i2c(int dado)
{
  .
  .
  .

}
```

DICA: Testar bit a bit do dado e verificar se o bit é 1 ou 0. Se o bit for 1, chamar envia_1_i2c(); Se o bit for 0, chamar envia_0_i2c();

## 1.3 Exemplo de software em C para o MSP430 conforme diagrama de leitura de dois bytes do sensor de temperatura

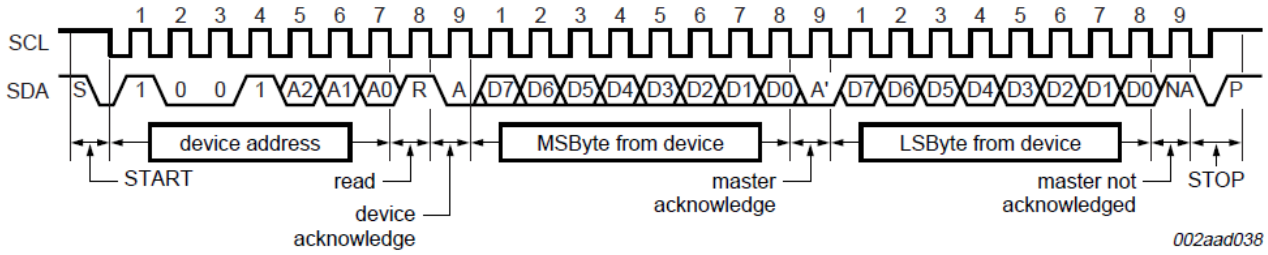


Figura 8: Leitura de dois Bytes da Temperatura

## 1.4 Conforme mostrado no diagrama anterior, é necessário construir algumas funções para a implementação do protocolo para a leitura do sensor, que são:

1. int LE_SENSOR_I2C();
2. START_I2C();
3. STOP_I2C();
4. ENVIA_1_I2C();
5. ENVIA_0_I2C();
6. ACK_I2C();
7. NACK_I2C();
8. ENVIA_BYTE_I2C(int dado);
9. int LE_BYTE_I2C();

### 1.4.1 Exemplo de como ler o sensor de temperatura:

```
int le_sensor(void)
{
  char x=0;
  int temp;

  start_i2c();
  envia_1_i2c();
  envia_0_i2c();
  envia_0_i2c();
  envia_1_i2c();
```

```
  envia_0_i2c();
  envia_0_i2c();
  envia_0_i2c();
  envia_1_i2c(); //1 pois é leitura
  x = ack_i2c();
  temp = le_byte(); //leitura de um byte da temperatura
  nack_i2c();
  stop_i2c();

  return temp;
}
```

### 1.4.2 Função Lê Byte

**Esta função deverá ser construída pelo aluno.**

```
int le_byte_i2c(void)
{
  int dado=0;
  .
  .
  .

  return dado;
}
```

DICA: Fazer a leitura da porta 8 vezes enquanto desloca para a esquerda(<<) e faz a operação ou(|) com o próximo dado.

# 2 Exemplo para o Smart-Card tipo 24LC02

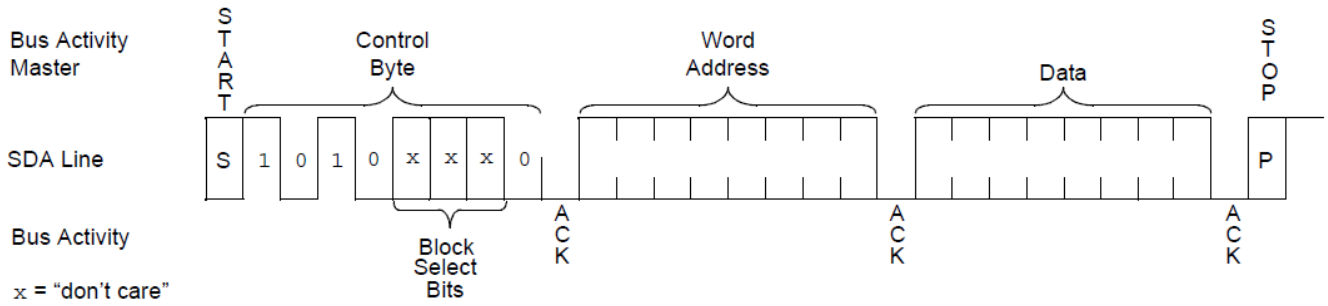


Figura 9: Escrita no Smart Card

## 2.1 Conforme mostrado no diagrama anterior, é necessário utilizar as funções mostradas anteriormente para a escrita e na memória do Smart-Card, que são:

1. ESCREVE_SMART(int endereco, int dado);
2. START_I2C();
3. STOP_I2C();
4. ENVIA_1_I2C();
5. ENVIA_0_I2C();
6. ACK_I2C();
7. NACK_I2C();
8. ENVIA_BYTE_I2C(int dado);
9. int LE_BYTE_I2C();

### 2.1.1 Exemplo de como escrever um dado em um determinado endereço do Smart-Card

```
void escreve_smart(int endereco, int dado)
{
  char x=0;

  start_i2c();
  envia_1_i2c();
  envia_0_i2c();
  envia_1_i2c();
  envia_0_i2c();
  envia_0_i2c();
  envia_0_i2c();
  envia_0_i2c();
  envia_0_i2c(); //0 pois é escrita
  x = ack_i2c();
  envia_byte_i2c(endereco);
  x = ack_i2c();
  envia_byte_i2c(dado);
  x = ack_i2c();
  stop_i2c();
}
```

—-

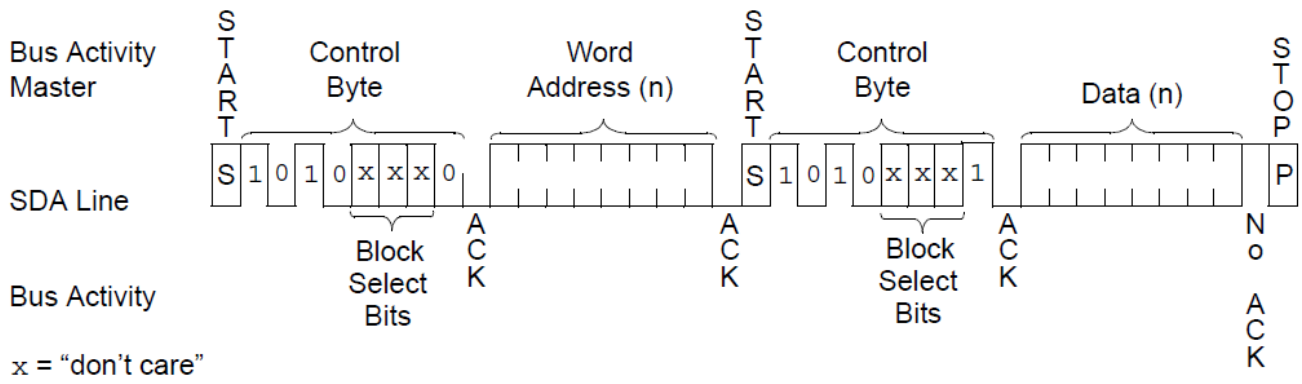


Figura 10: Leitura do Smart Card

## 2.2 Conforme mostrado no diagrama anterior, é necessário utilizar as funções mostradas anteriormente para a leitura da memória do Smart-Card, que são:

1. int LE_SMART(int endereco);
2. START_I2C();
3. STOP_I2C();
4. ENVIA_1_I2C();
5. ENVIA_0_I2C();
6. ACK_I2C();
7. NACK_I2C();
8. ENVIA_BYTE_I2C(int dado);
9. int LE_BYTE_I2C();

### 2.2.1 Exemplo de como ler um dado de um determinado endereço do Smart-Card

```
int le_smart(int endereco)
{
  char x=0;
  int dado;

  start_i2c();
  envia_1_i2c();
  envia_0_i2c();
```

```
  envia_1_i2c();
  envia_0_i2c();
  envia_0_i2c();
  envia_0_i2c();
  envia_0_i2c();
  envia_0_i2c(); //0 pois é escrita do endereço
  x = ack_i2c();
  envia_byte_i2c(endereco);
  x = ack_i2c();
  start_i2c();
  envia_1_i2c();
  envia_0_i2c();
  envia_1_i2c();
  envia_0_i2c();
  envia_0_i2c();
  envia_0_i2c();
  envia_0_i2c();
  envia_1_i2c(); //1 pois é leitura do dado
  x = ack_i2c();
  dado = le_byte(); //leitura do dado
  nack_i2c();
  stop_i2c();

  return dado;
}
```