Mestrado Integrado em Engenharia Eletrónica Industrial e Computadores

Microprocessadores II

2020/2021

Guia 4

I2C : interface com E2PROM e encriptação de dados

Vitor Silva
vsilva@dei.uminho.pt

Tiago Gomes
mr.gomes@dei.uminho.pt

Sofia Paiva

José Mendes
jose.mendes@dei.uminho.pt

Departamento de Eletrónica Industrial

Universidade do Minho

Introdução

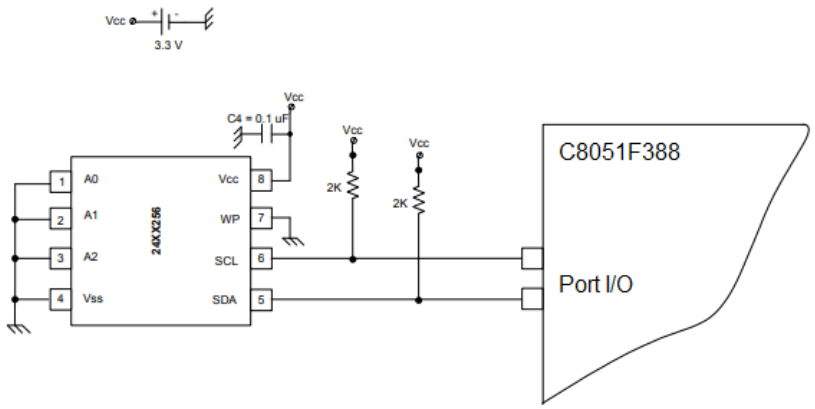
O protocolo inter-integrated circuit (I2C) que aprendeu nas aulas teóricas é um protocolo série, síncrono, e bidirecional (half-duplex), criado pela Philips no início dos anos 80 para interligar periféricos de baixa velocidade, tendo ganho enorme popularidade e utilização em diversos sistemas computorizados atuais (smart TVs, smartphones, computadores, etc). Dado que utiliza um barramento (dados e clock) partilhado por todos os dispositivos ligados, uma comunicação entre dois dispositivos pode ser facilmente "escutada" através de um dispositivo sniffer, sem que este interfira com o normal funcionamento do sistema. Como tal, em determinadas aplicações utilizam-se técnicas de encriptação sobre os dados transmitidos, o que impossibilita a sua interpretação imediata.

Neste trabalho pretende-se ler e escrever mensagens de texto enciptadas num periférico I2C (memória E2PROM). As mensagens antes de serem armazenadas na memória EEPROM devem ser previamente encriptadas com uma chave de encriptação de 4 bytes e o seu conteúdo só estará legível depois da sua desencriptação, impedindo assim que os dados transmitidos possam ser interpretados por terceiros.

Problema

Implemente um programa para o microprocessador C8051F388 que permita ler e escrever dados encriptados numa memória EEPROM da família 24CXX (24C01/2/4/8/16), recebidos pela porta série. Apesar de existir um periférico two-wire interface TWI (que implementa o protocolo I2C) no microprocessador C8051F388, este não é suportado pelo simulador do Keil, pelo que o programa a desenvolver deve fazer uso dos pinos de GPIO para implementar protocolo através de software, usando uma técnica conhecida como *bit banging*. Desta forma é possível a implementação e debug do protocolo através do ambiente de simulação do Keil onde, e com o auxílio da ferramenta "logic analyzer", torna-se mais fácil a interpretação e visualização das tramas enviadas para a memória através dos pinos GPIO selecionados, bem como a manipulação e simulação dos dados recebidos.

A título de exemplo, a próxima figura apresenta um esquema das ligações entre o periférico EEPROM (slave) e o microcontrolador C8051F388 (master), com os pinos de endereço A2, A1, e A0 ligados ao sinal *ground*.



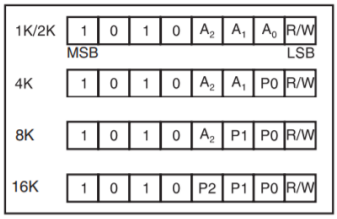
A mensagem em texto limpo deverá ser recebida através do interface série, previamente desenvolvido nos guias anteriores, e uma vez armazenada na memória interna do microcontrolador, deve ser encriptada aplicando a chave de encriptação ao array de memória correspondente, antes de ser posteriormente enviada via I2C para armazenamento na memória EEPROM. Após uma sequência de leitura de uma mensagem armazenada na memória EEPROM, é necessário voltar a aplicar a chave de encriptação de forma a ser possível obter-se novamente a mensagem em texto limpo.

A chave de encriptação é normalmente negociada, de forma segura, numa etapa de *handshake* da comunicação, mas por questões de simplicidade a respetiva chave deve ser recebida via porta série e ficar armazenada na memória interna do microcontrolador, e com o tamanho predefinido de 4 bytes. A técnica de encriptação/desencriptação a usar encontra-se no excerto de código abaixo, onde DATA_LEN representa o tamanho da mensagem recebida (ex. 16 bytes), CYPHER_LEN é 4 e *cypher* é a chave de encriptação/desencriptação previamente escolhida.:

```
for( i = 0; i < DATA_LEN; i++ )
    buf[i] ^= cypher[ i%CYPHER_LEN ];
```

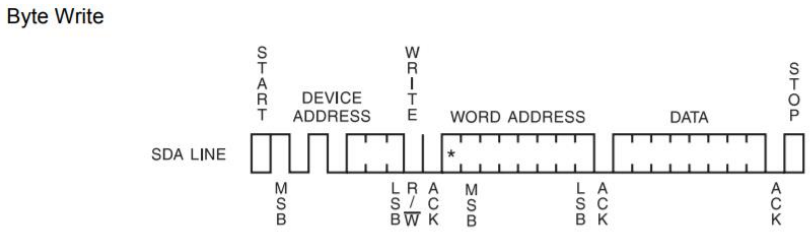
Para a configuração do periférico I2C, bem como as operações de leitura e escrita, deverá seguir-se a sequência de operações (de acordo com o *datasheet* do fabricante). Para uma memória da familia AT24C01A/02/04/08A/16A o endereço do periférico pode ser configurado da seguinte forma:

Device Address



Para a realização do guia considera-se uma memória de baixa densidade AT24C02, cujo endereço pode ser obtido da seguinte forma: "**1 0 1 0 A2 A1 A0**".

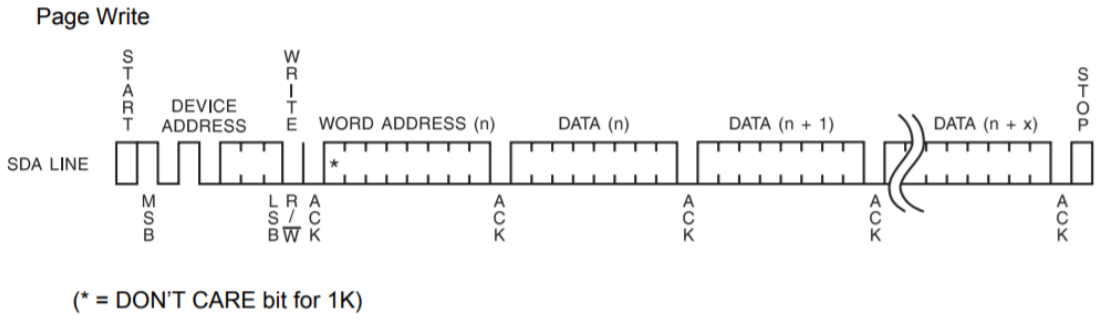
Trama completa para a escrita de **um** byte de dados, **0xAA**, na posição de memória **0x08**, do slave com o endereço **0x50**:



S	SLAVE ADDR + W	A	MEM ADDR	A	DATA	A	P
0	0xA0	0	0x01	0	0xAA	0	1

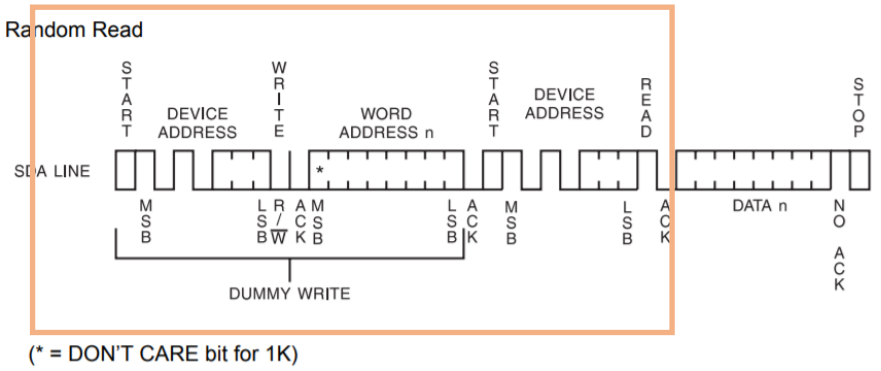
A memória AT24C02 está internamente organizada em 32 blocos (páginas) de 8 bytes, permitindo a escrita de um bloco de uma só vez até ao tamanho máximo da página e usando apenas uma vez o endereço de escrita (este é automaticamente incrementado sempre que um byte é escrito). Por questões de simplicidade, assumimos que a trama que vamos escrever, cabe sempre numa página da memória AT24C02.

A trama para a escrita de um bloco de dados **data[] = {'0', '1', '2', '3', '4', '5'}** na posição de memória **0x08** do slave com o endereço **0x50**:



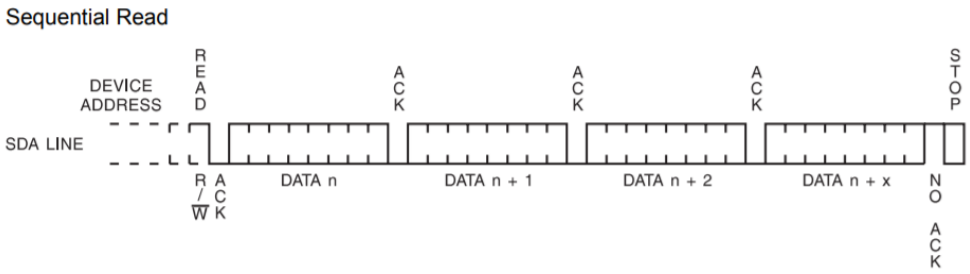
S	SLAVE ADDR + W	A	MEM ADDR	A	DATA	A	DATA	A	DATA	A	DATA	A	DATA	A	DATA	A	P
0	0xA0	0	0x08	0	0x30	0	0x31	0	0x32	0	0x33	0	0x34	0	0x35	0	1

Para uma sequência de leitura de 1 byte da memória a partir da posição 0x08, deverá ser feito inicialmente um pedido de escrita incompleta (*dummy write*), apenas para posicionar o apontador interno para os dados na posição 0x08, pelo que os pedidos de leitura subsequentes (se for necessário ler mais que um byte) não necessitam de indicar o endereço de dados, como se pode verificar na figura seguinte.



S	SLAVE ADDR + W	A	MEM ADDR	A	RS	SLAVE ADDR + R	A	DATA	NACK	P
0	0xA0	0	0x08	0	0	0xA1	0	0x30	1	1

Para uma sequência de leitura de 4 bytes da memória com o endereço 0x50, a partir da posição 0x08 deve-se utilizar a trama ilustrada na figura seguinte. De notar que, a cada leitura, e dentro da mesma página, o apontador para os dados é incrementado automaticamente sempre que o *master* envia um ACK a confirmar a receção.



S	SLAVE ADDR + W	A	MEM ADDR	A	RS	SLAVE ADDR + R	A	DATA	A	DATA	A	DATA	A	DATA	NACK	P
0	0xA0	0	0x08	0	0	0xA1	0	0x30	0	0x31	0	0x32	0	0x33	1	1

No final da leitura dos dados, deve-se aplicar novamente a chave de encriptação no texto encriptado para se obter o texto limpo. Este deve ser porteriormente enviado via interface série para o terminal do sistema *host*.

Recomenda-se a elaboração do programa seguindo sempre uma abordagem modular, abstraindo assim as operações da memória do protocolo de comunicação utilizado. Para isso podem desenvolver-se 2 módulos (respectivos .c e .h), um para a memória EEPROM, e outro para o I2C.

Como sugestão, o módulo E2PROM poderá incluir as seguintes funções: ***memory_write_byte***, ***memory_read_byte***, ***memory_write_page***, e ***memory_read_page***. Os parâmetros de entrada/saída das funções devem ser convenientemente selecionados. Este módulo, por sua vez, recorre ao módulo I2C que deverá implementar as seguintes funções: ***i2c_start_condition, i2c_stop_condition, i2c_read_ack, i2c_write, i2c_read, i2c_write_ack***.

Nota: Implementar todas as funções que considere relevantes, além destas, seguindo a abordagem de programação que achar mais indicada. Pode também reutilizar os módulos e funcionalides desenvolvidas nos guias anteriores que achar relevantes (command line para envio de tramas, chaves de encriptação, biblioteca serialIO, push button, etc.)

Na ausência de memória AT24C02, programa desenvolvido deverá ser simulado e o resultado da técnica de bit bang poderá ser controlado/visualizado pela ferramenta logic analyzer do Keil. A simulação das tramas de leitura deverá contemplar uma trama encriptada, previamente guardada na memória.

Elementos a entregar

O trabalho deve ser feito em grupo e como tal cada elemento do grupo deve ficar responsável por uma parte do trabalho. Deste modo, cada elemento deverá entregar uma identificação do grupo e o plano de trabalhos, onde se descreve a estratégia e como foi dividido o trabalho pelo grupo, a sua parte do algoritmo de programação, fluxogramas, incluindo o conjunto de variáveis de memória, configuração de periféricos, e ficheiro (s) do código fonte: extensão c e h contendo o **programa completo** na sua versão final. (Pode ser enviado um ficheiro .zip da diretoria do projeto).