Trabalho 1

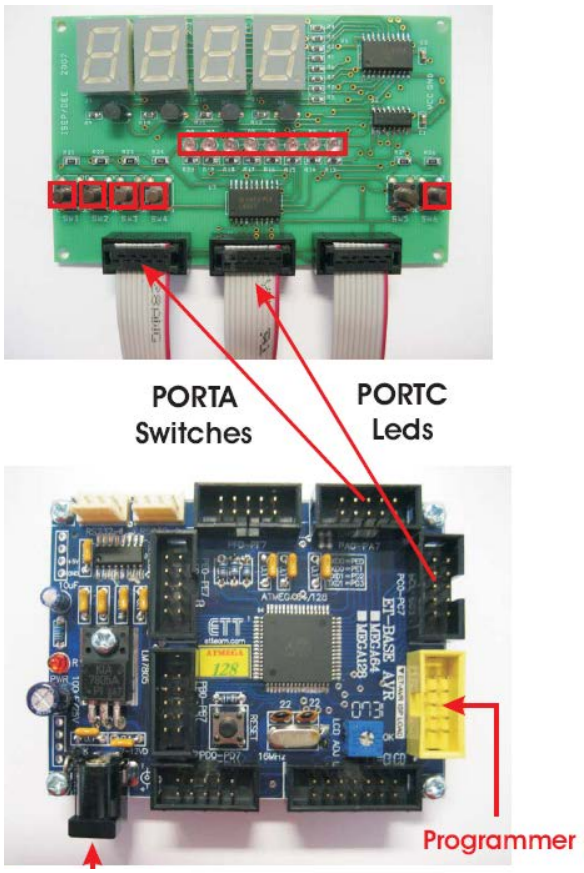
Objectivo: implementação de operações simples de entrada/saída de dados.

Necessário: conhecimentos de arquitectura do μC, arquitectura de memória e portos de I/O

Funcionamento 1: pretende-se controlar o estado dos LEDs D1…D8 utilizando para isso os interruptores SW. O estado dos LEDs deve obedecer à seguinte tabela:

Interruptor activo	LEDs ligados
SW1	D1, D2, D3, D4, D5, D6, D7, D8
SW2	D2, D3, D4, D5, D6, D7
SW3	D3, D4, D5, D6
SW4	D4, D5
SW6	(LEDs todos desligados)

Hardware a utilizar:



Porto dos Interruptores

Pino nº	Função	Pino nº	Função
1	SW1 (PA.0)	2	SW2 (PA.1)
3	SW3 (PA.2)	4	SW4 (PA.3)
5	SW5 (PA.4)	6	SW6 (PA.5)
7	MUX.0	8	MUX.1
9	Vcc	10	Ground

Nota: de acordo com o hardware da placa de I/O, o estado de repouso dos interruptores corresponde ao valor lógico 1, o estado de accionado corresponde ao valor lógico 0.

Porto dos LEDs

Pino nº	Função	Pino nº	Função
1	D1 (PC.0)	2	D2 (PC.1)
3	D3 (PC.2)	4	D4 (PC.3)
5	D5 (PC.4)	6	D6 (PC.5)
7	D7 (PC.6)	8	D8 (PC.7)
9	Vcc	10	Ground

Nota: de acordo do o hardware da placa de I/O, para acender um LED deve ser colocado no respectivo pino o valor lógico 0, para apagar um LED deve ser colocado o valor lógico 1.

Implementação do software

- Utilizando **linguagem Assembly**

Funcionamento 2: partindo da situação em que os 8 LEDs (D1 .. D8) se encontram apagados, pretende-se que, ao acionar **SW1**, os LEDs sejam ativados sequencialmente, começando pelo LED **D1** até **D8**. A sequência dos diferentes passos deve ser feita de, aproximadamente, **500 em 500 ms**. Durante o tempo em que o LED **D8** estiver activado (**500 ms**) pode ser accionado o **SW6** para inverter a sequência de accionamento dos LEDs (**D8 → D1**). Durante o tempo em que o LED **D1** estiver activado (**500 ms**) pode ser accionado o **SW1** para inverter novamente a sequência de accionamento dos LEDs (**D1 → D8**). Caso **SW1**, ou **SW6 (sequência D8 → D1),** não seja accionado deve terminar a sequência e piscar os LEDs **D4** e **D5** a uma frequência de **1 Hz** durante **3 s**. No final deste tempo, deve desligar todos os LEDs e iniciar nova sequência caso **SW1** seja accionado.

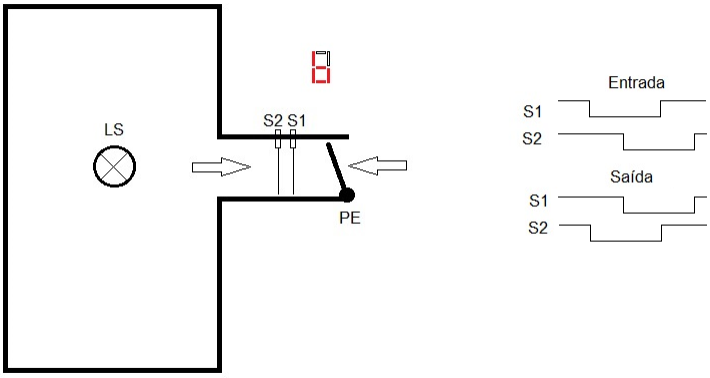
Implementação do software

- Utilizando **linguagem Assembly**

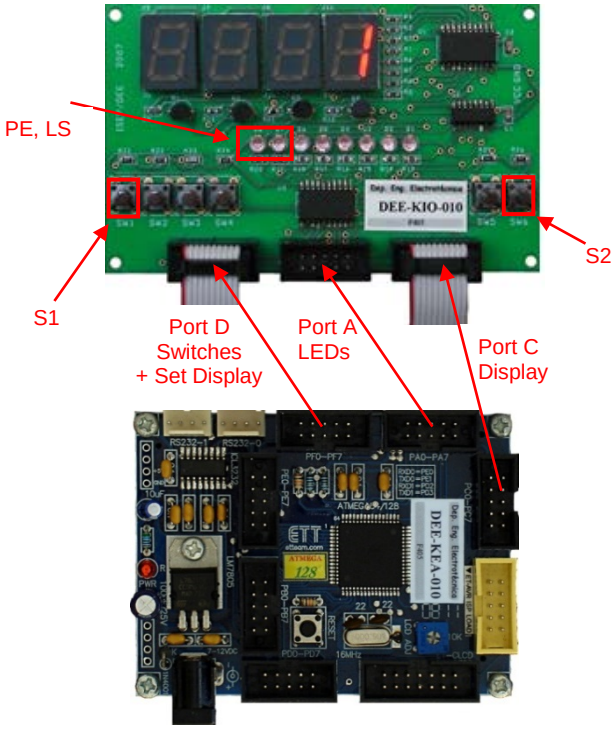
Funcionamento 3: Pretende-se simular o controlo do acesso de pessoas a uma sala com uma lotação máxima de **9** pessoas. A entrada e saída de pessoas é efectuada pelo mesmo local sendo a passagem das pessoas detectada pelos sensores **S1** e **S2.** Sempre que a lotação atingir o máximo, a porta **PE (D8)** deve ser fechada para impedir a entrada de mais pessoas. O display mais à direita deve mostrar o **nº de lugares vagos** na sala. A luz da sala **LS (D7)** deve ser desligada sempre que a sala estiver vazia.

Os sensores **S1** e **S**2 estão colocados com um desfasamento de 90º para permitir detectar a entrada e saída de pessoas. Na entrada de uma pessoa, primeiro é activado **S1** e depois **S2**, enquanto na saída, primeiro é activado **S2** e depois **S1**.

Para determinar o valor dos sensores **S1** e **S2** devem ser feitas, para cada sensor, duas leituras com um intervalo de **1 ms** e ambas devem ter o mesmo valor lógico para que seja considerado uma leitura válida.



Hardware a utilizar:



Porto dos Interruptores

Pino nº	Função	Pino nº	Função
1	**SW1 (PD.0)**	2	SW2 (PD.1)
3	SW3 (PD.2)	4	SW4 (PD.3)
5	SW5 (PD.4)	6	**SW6 (PD.5)**
7	**MUX.0 (PD.6)**	8	**MUX.1 (PD.7)**
9	Vcc	10	Ground

Nota: PD.0 .. PD.5 devem ser programados como entrada de dados para a leitura dos interruptores. PD.6 e PD.7 devem ser programados como saída de dados e o seu valor deve ser **1** para que o dígito a utilizar seja o dígito da direita.

Porto dos Displays

Pino nº	Função	Pino nº	Função
1	Seg a (PC.0)	2	Seg b (PC.1)
3	Seg c (PC.2)	4	Seg d (PC.3)
5	Seg e (PC.4)	6	Seg f (PC.5)
7	Seg g (PC.6)	8	DP (PC.7)
9	Vcc	10	Ground

Nota: para acender um segmento deve ser colocado no respectivo pino o valor lógico 0, para apagar um segmento deve ser colocado o valor lógico 1.

Tabela dos segmentos

Dígito	DP	Seg g	Seg f	Sege	Seg d	Seg c	Seg b	Seg a	PORTC
0	1	1	0	0	0	0	0	0	0xC0
1	1	1	1	1	1	0	0	1	0xF9
2	1	0	1	0	0	1	0	0	0xA4
3	1	0	1	1	0	0	0	0	0xB0
4	1	0	0	1	1	0	0	1	0x99
5	1	0	0	1	0	0	1	0	0x92
6	1	0	0	0	0	0	1	0	0x82
7	1	1	1	1	1	0	0	0	0xF8
8	1	0	0	0	0	0	0	0	0x80
9	1	0	0	1	0	0	0	0	0x90

Implementação do software

- Implementar o software utilizando **linguagem Assembly**