Trabalho prático N.º 2

Objetivos

- Montar, em placa branca, dispositivos simples de interação com o utilizador ligados a portos de I/O do PIC32: 4 LEDs, 1 *dip-switch* de 4 posições, 2 *displays* de 7 segmentos e 1 potenciómetro.
- Executar programas de teste fornecidos para verificar o bom funcionamento dos dispositivos montados.
- Utilizar o *core timer* do MIPS para gerar atrasos programáveis.

Trabalho a realizar

Parte I

1. A Figura 1 apresenta a ligação de 4 LEDs a 4 portos do PIC32 (Porto E: **`RE0`**, **`RE1`**, **`RE2`** e **`RE3`**). Cada LED tem uma resistência em série que limita o valor da corrente que o atravessa. Para a polarização de um LED deste tipo, devemos usar um valor de corrente inferior a 10mA. O valor da corrente no LED é dado por **`(3.3-1.5)/R`**, em que 3.3V é o valor da tensão correspondente ao nível lógico '**`1`**' no porto e 1.5V é o valor aproximado da tensão aos terminais do LED quando este está no estado ON. Para uma resistência de 270Ω, a corrente é, aproximadamente, 6.7mA.

   a) Observe a sugestão de colocação dos componentes na placa branca fornecida na Figura 5 e a disposição dos portos na placa DETPIC32 fornecida na Figura 6 (em anexo). De seguida, monte na placa branca os componentes do esquema da Figura 1.

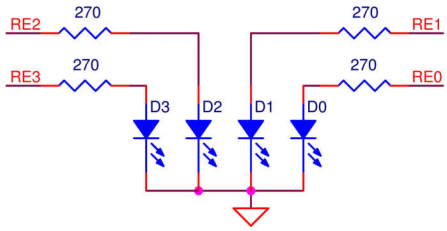

Figura 1. Ligação de 4 LEDs a portos do PIC32.

   b) Transfira para a placa DETPIC32 o programa de teste "**`test_leds.hex`**" disponível no *moodle* da UC. Deverá observar a ativação sequencial dos LEDs, do menos significativo (ligado ao porto **`RE0`**), para o mais significativo (ligado ao porto **`RE3`**). Se tal não acontecer, verifique a origem do erro e faça as correções devidas.

2. A Figura 2 apresenta o esquema de ligação de um *dip-switch* de 4 posições a portos do PIC32 (Porto B: **`RB0`**, **`RB1`**, **`RB2`** e **`RB3`**). Em cada um dos portos, a resistência de 10kΩ destina-se a fixar a tensão a zero quando o *switch* está aberto. A resistência de 470Ω foi incluída para evitar uma possível destruição do porto do microcontrolador no caso de ser configurado, por engano, como uma saída.

   a) Monte na placa branca os componentes do esquema da Figura 2.

   b) Transfira para a placa DETPIC32 o programa de teste "**`test_dsw.hex`**". Com este programa de teste cada um dos *switches* deverá ativar um LED: **`SW1`** ativa o LED ligado ao porto **`RE0`**,…, **`SW4`** ativa o LED ligado ao porto **`RE3`**.

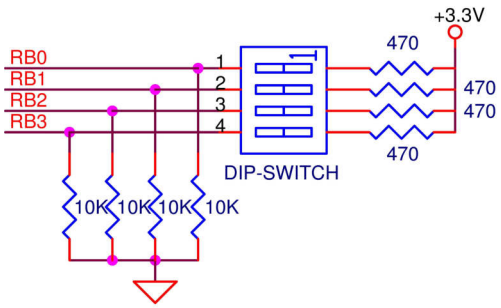

Figura 2. Ligação de um *dip-switch* de 4 posições a portos do PIC32.

3. A Figura 3 apresenta o esquema elétrico de ligação de dois *displays* de 7 segmentos a portos do PIC32 (Porto B: **`RB8`** a **`RB15`**; Porto D: **`RD5`** e **`RD6`**). Cada um dos *displays* pode ser ativado ou desativado através dos transístores Q1 e Q2 ligados aos portos **`RD5`** e **`RD6`**, respetivamente. Com esta configuração, os *displays* funcionam em modo alternado, uma vez que o barramento de dados é comum aos dois.

   Cada LED do *display* de 7 segmentos tem em série uma resistência que limita a corrente que nele circula (como já referido no ponto 1); a não colocação desta resistência tem como consequência a destruição do LED e/ou da saída do microcontrolador à qual o LED estiver ligado. A corrente consumida por cada LED pode ser calculada como: **`I=(3.3-1.5)/R`**; para uma resistência de 330Ω, a corrente é, aproximadamente, 5.5mA.

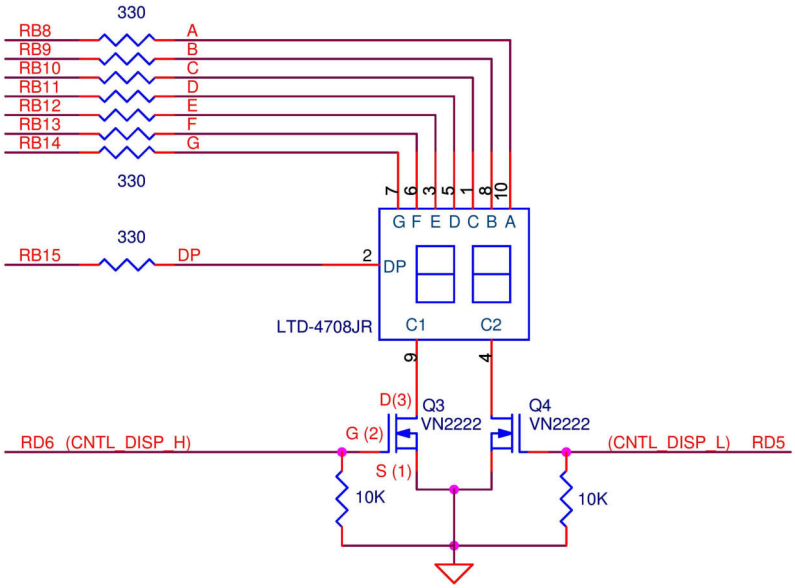

Figura 3. Ligação de dois *displays* de 7 segmentos ao porto B do PIC32.

a) Monte os 2 *displays* de 7 segmentos e os dois transístores, de acordo com o esquema da Figura 3. Verifique no *datasheet* do *display* de 7 segmentos (disponível no *moodle* da UC) qual a correspondência entre cada segmento e o respetivo pino físico.

b) Ligue os dois transístores ao circuito. Verifique previamente quais os pinos físicos correspondentes à *gate* (G), *drain* (D) e *source* (S) (note que a montagem incorreta deste componente impede, ou limita, o bom funcionamento do circuito de visualização).

c) Transfira para a placa DETPIC32 o programa de teste "**`test_displays.hex`**". Com este programa deverá observar a ativação sequencial dos LEDs de cada *display*, do segmento "a" até ao segmento "g", seguida de uma contagem de **`0`** a **`F`**.

4. No circuito da figura seguinte uma resistência variável (designada por potenciómetro) está ligada ao bit **`AN4`** do PIC32 (**`RB4`**). Na configuração em que está montada, a resistência variável permite variar, de forma linear, a tensão no seu ponto intermédio entre 0 V e 3.3 V.

   a) Monte esse circuito e ligue o ponto intermédio do potenciómetro à entrada **`RB4`** da placa DETPIC32.

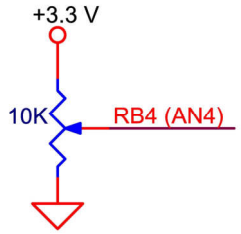

Figura 4. Ligação de uma resistência variável à placa DETPIC32.

   b) Transfira para a placa DETPIC32 o programa de teste "**`test_all.hex`**". Com este programa de teste poderá verificar se todo o hardware da placa está a funcionar corretamente. Deverá observar que: i) com todos os *switches* na posição OFF nos *displays* deverá aparecer um valor entre 0.0 e 3.3 correspondente ao valor da tensão no potenciómetro e nos LEDs uma indicação da amplitude dessa tensão ii) nas restantes combinações dos *switches*, os LEDs acendem sequencialmente e os *displays* mostram o valor hexadecimal da combinação binária presente nos *switches*.

Parte II

O *core* MIPS disponível no microcontrolador PIC32 implementa, no coprocessador 0, um contador crescente de 32 bits (designado por *core timer*) atualizado a cada dois ciclos de relógio do CPU. Na placa DETPIC32 o relógio do CPU está configurado a 40 MHz, pelo que o contador é incrementado a uma frequência de relógio de 20 MHz. Isto significa que o tempo necessário para incrementar o contador desde o valor 0 até 20.000.000 é 1 segundo.

A placa DETPIC32 disponibiliza dois *system calls* para interagir com esse contador: ler o valor atual do contador (**`readCoreTimer()`**) e reiniciar a zero o seu valor (**`resetCoreTimer()`**).

1. O programa seguinte incrementa o valor de uma variável à frequência de 100 Hz. De cada vez que a variável é atualizada, o seu valor é apresentado no ecrã do PC.

```
void main(void)
{
   int counter = 0;
   while(1)
   {
      while(readCoreTimer() < 200000);
      resetCoreTimer();
      printInt(++counter, 10);
      putChar(' ');  // space
   }
}
```

   a) Complete a tradução para código *assembly* do MIPS que se apresenta de seguida e teste-o na placa DETPIC32.

```
.equ    READ_CORE_TIMER,11
.equ    RESET_CORE_TIMER,?
.equ    PUT_CHAR,?
.equ    PRINT_INT,?
```

```
        .data
        .text
        .globl  main
main:   li      $t0,0
while:  li      $v0,READ_CORE_TIMER # while (1) {
        syscall                     #
        b??     $v0,200000,???      #   while(readCoreTimer() < 20000);
        li      $v0,???             #
        syscall                     #   resetCoreTimer();
        li      $a0,' '             #
        li      $v0,PUT_CHAR        #
        syscall                     #   putChar(' ');
        addi    $t0,$t0,???         #
        move    $a0,$t0             #
        li      $a1,10              #
        li      $v0,???             #
        syscall                     #   printInt(++counter, 10);
        ?       ???                 # }
        jr      $ra                 #
```

   b) Altere o código de forma a que a variável seja incrementada com uma frequência de 10 Hz, 5 HZ e de 1Hz.

2. O objetivo da função **`delay()`**, apresentada a seguir, é gerar um atraso temporal programável múltiplo de 1ms.

```
void delay(int ms)
{
   for(; ms > 0; ms--)
   {
      resetCoreTimer();
      while(readCoreTimer() < K);
   }
}
```

   a) Determine o valor da constante K, de modo a que para "ms" igual a 1 o atraso gerado seja 1ms.

   b) Traduza para *assembly* do MIPS a função **`delay()`** e teste-a com diferentes valores de entrada (para o teste utilize como base o código C fornecido no ponto 1).

3. Usando a sequência de instruções *assembly* "**`lui $t1,0xBF88`**", "**`lw $t2,0x6050($t1)`**" pode ler o valor binário dos 4 *switches* que montou na sua placa. Esse valor fica disponível nos bits 3 a 0 do registo **`$t2`**.

   a) Escreva um programa em *assembly* que leia, com uma frequência de 2 Hz, o valor lógico imposto pelos *switches* e que o mostre, em binário[1], no ecrã do PC.

   b) Altere o programa anterior de modo a que o valor dos *switches* seja usado para impor a frequência de leitura, entre 1 Hz e 16 Hz; a constante de tempo para a função **`delay()`** pode ser calculada como: k=**`1000/(valSwitch + 1)`**.

Elementos de apoio

- *Datasheets* do *display* LTD-4708JR e do transístor VN2222 (disponíveis no *moodle* de AC2).

[1] O *system call* **`printInt`** permite especificar o número mínimo de dígitos com que o valor é impresso. Essa configuração é feita nos 16 bits mais significativos do registo usado para a base da representação (e.g., para a impressão em binário com 4 bits, o valor a colocar no registo **`$a1`** é **`0x00040002`**).

Anexo

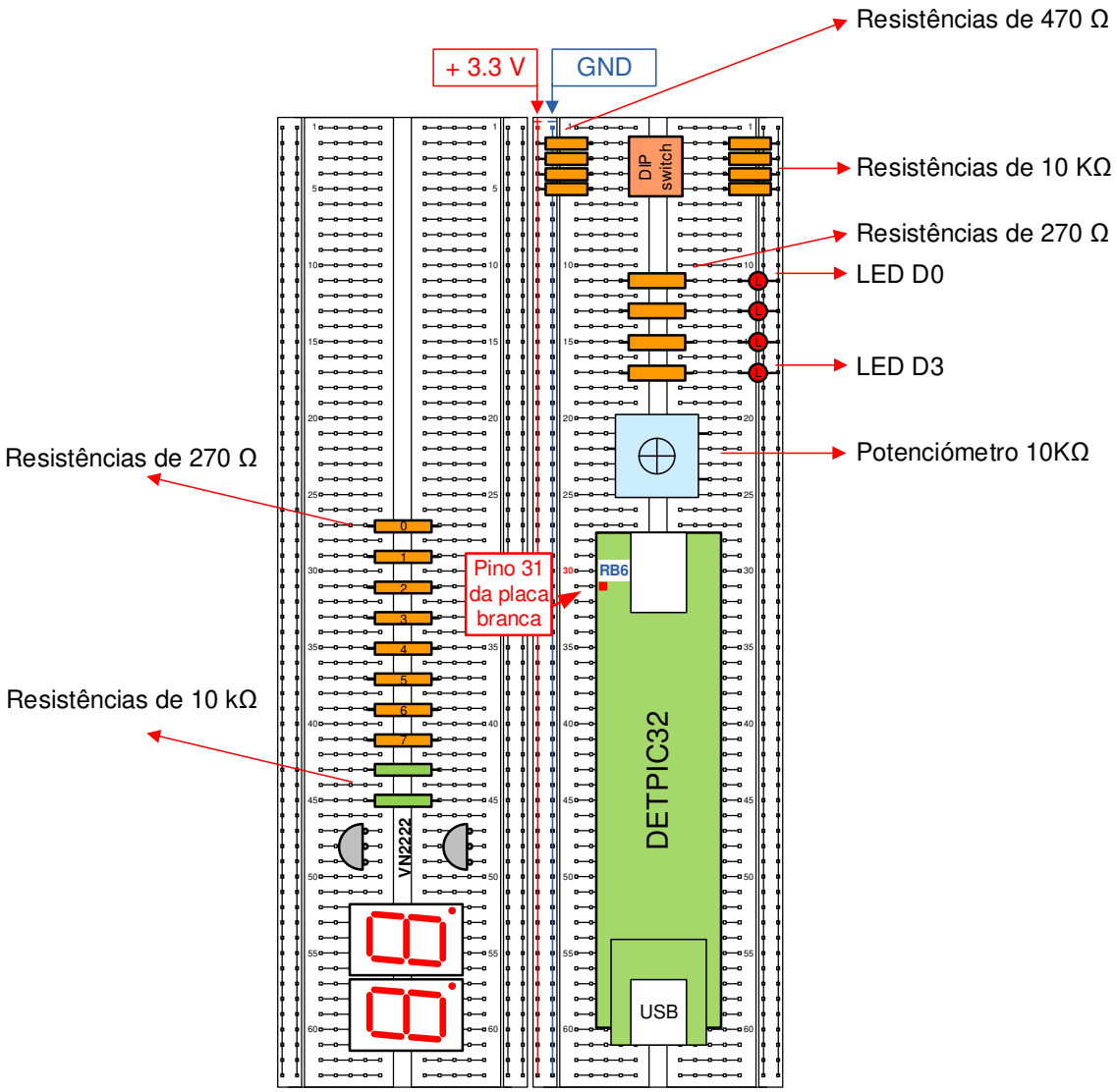

Figura 5. Sugestão de organização do espaço nas placas brancas.

Notas:

1. Não deve utilizar qualquer fonte de tensão externa. Deve-se ter, no entanto, em atenção que o consumo máximo admissível do conjunto não deverá exceder 100 mA.
2. Utilize apenas um barramento da placa branca para a ligação da massa (GND) e apenas um para a ligação da tensão positiva (+3.3V).
3. A inserção/remoção da placa DETPIC32 na placa branca deve ser realizada com muito cuidado de forma a evitar o empeno e consequente quebra dos seus pinos. A inserção deve ser feita pressionando lentamente a placa em ambas as extremidades A remoção deve ser evitada, mas sempre que houver necessidade de o fazer, deverá ser efetuada de forma a puxar simultaneamente as duas extremidades da placa DETPIC32.
4. A placa DETPIC32 deverá ser montada de tal forma que a ficha de ligação USB fique posicionada na extremidade da placa branca. Por outro lado, a ligação à placa DETPIC32 estará menos sujeita a erros se o pino 1 (correspondente ao porto **RB6**) ficar posicionado no pino 31 da placa branca. Deste modo, para se obter o número do pino da placa DETPIC32 basta subtrair 30 ao número da pista correspondente da placa branca.

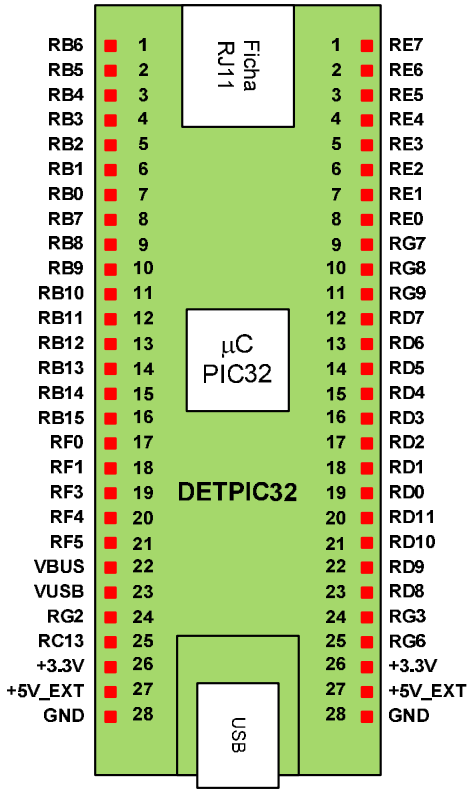

Figura 6. Disposição dos pinos de ligação à placa branca no DETPIC32.

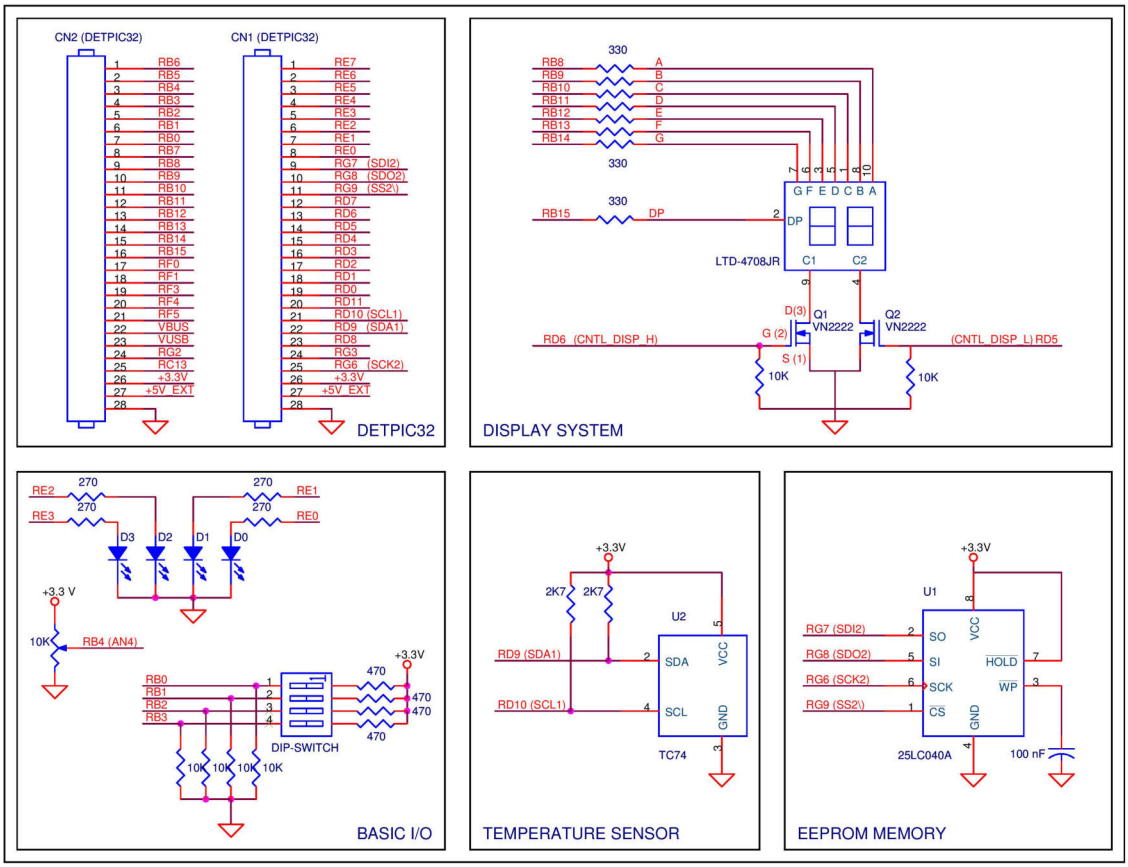

Figura 7. Esquema elétrico completo
(incluindo um sensor de temperatura e uma memória EEPROM).