Trabalho prático N.º 5

Objetivos

- Familiarização com o modo de funcionamento de um periférico com capacidade de produzir informação.
- Utilização da técnica de *polling* para detetar a ocorrência de um evento e efetuar o consequente processamento.
- Efetuar a conversão analógica/digital de um sinal de entrada e mostrar o resultado no sistema de visualização implementado anteriormente.

Introdução

Um conversor analógico-digital (A/D) é um dispositivo eletrónico que efetua a conversão de uma grandeza contínua (uma tensão) numa representação digital com **n** bits (uma quantidade numérica com **n** bits). O número de bits que o conversor usa para a representação numérica designa-se por resolução e representa o logaritmo na base 2 do número de níveis em que o conversor divide a gama de tensão de entrada (quanto maior for a resolução, melhor será, idealmente, a exatidão da conversão). Por exemplo, um conversor A/D de 10 bits divide a gama de tensão de entrada em 1024 níveis (2^{10}), fazendo corresponder à tensão mínima o valor **`0x000`** e à tensão máxima admissível o valor **`0x3FF`** (1023). Se os valores mínimo e máximo dessas tensões forem 0V e 3.3V, respetivamente, as correspondentes representações digitais serão **`0x000`** e **`0x3FF`**. A Figura 1 mostra um exemplo de codificação de uma gama de tensão Vmax com 3 bits, isto é, com 8 níveis.

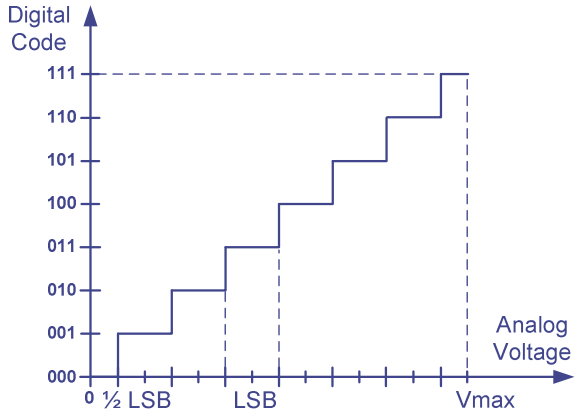

Figura 1. Conversão de uma gama de tensão 0 - Vmax com 3 bits (8 níveis).

O incremento na tensão de entrada a que corresponde uma alteração no bit menos significativo da codificação designa-se por LSB e é dado por LSB = VR / (2^N-1), em que VR é a gama de tensão de entrada e N o número de bits usado para a codificação. No exemplo da Figura 1, se Vmax for igual a 7V então LSB será LSB=7 / (2^3-1) = 1 V. Para um conversor de 10 bits com uma gama de tensão de entrada de 3.3V, o valor do LSB é dado por LSB = 3.3V / (2^{10} - 1), aproximadamente 3.2 mV.

O PIC32 disponibiliza um módulo de conversão analógico-digital, com um modelo de programação que permite múltiplas possibilidades de configuração. No essencial, o módulo A/D é constituído por um conversor analógico-digital de 10 bits (que utiliza para a conversão o método das aproximações sucessivas), um *multiplexer* analógico de 16 entradas e um conjunto de registos onde o conversor coloca o(s) resultado(s) da conversão, tal como esquematizado no diagrama de blocos da Figura 2.

A zona de armazenamento dos resultados (designada por *buffer*) é constituída por 16 registos de 32 bits, referenciados pelos nomes **`ADC1BUF0`**, **`ADC1BUF1`**, ..., **`ADC1BUFF`**, que podem ser acedidos nos endereços **`0xBF809070`**, **`0xBF809080`**, .**`0xBF809090`**, ..., **`0xBF809160`**.

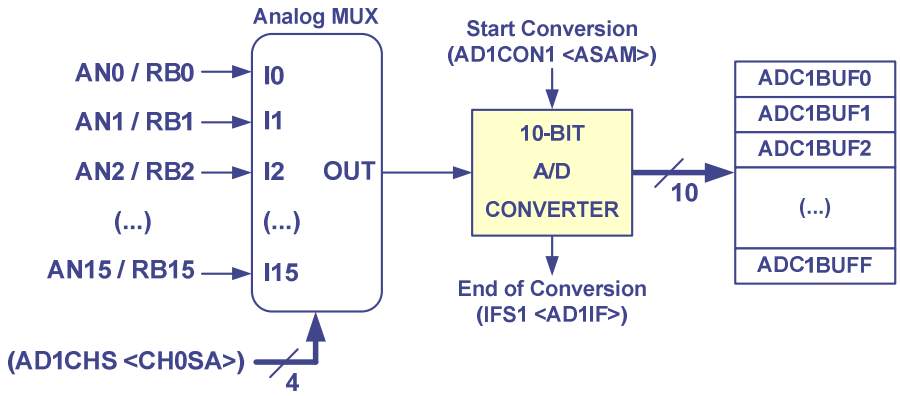

Figura 2. Diagrama de blocos simplificado do módulo A/D do PIC32.

O objetivo deste trabalho prático não é o estudo aprofundado do módulo A/D, pelo que vamos usar apenas um dos vários modos de funcionamento possíveis[1]. Para esse modo de funcionamento, são necessários os seguintes passos de configuração:

1) configurar um dos portos de I/O do porto B como entrada analógica (registos **`TRISB`** e **`AD1PCFG`**);
2) selecionar a entrada analógica a converter, isto é, escolher o canal de entrada (registo **`AD1CHS`**, bits **`<CH0SA>`**);
3) configurar o número de conversões consecutivas do mesmo canal (registo **`AD1CON2`**, bits **`<SMPI>`**;
4) configurar o *trigger* do processo de início de conversão para "auto convert" (registo **`AD1CON1`**, bits **`SSRC`**);
5) determinar que o processo de conversão apenas ocorre quando é dada ordem de início de conversão (registo **`AD1CON1`**, bit **`CLRASAM`**)
6) configurar a duração do tempo de amostragem (registo **`AD3CON`**, bits **`SAMC`**).

Após a configuração dos parâmetros de funcionamento, o processo de conversão envolve:

1) dar ordem de início de conversão ao conversor A/D (registo **`AD1CON1`**, bit **`<ASAM>`**);
2) esperar que o sinal que indica fim de conversão fique ativo (registo **`IFS1`**, bit **`<AD1IF>`**);
3) ler o resultado da conversão em **`SMPI+1`** registos **`ADC1BUFx`**.

Configuração das entradas analógicas

As 16 entradas analógicas do PIC 32 são fisicamente coincidentes com os 16 bits do porto B. Qualquer um destes 16 pinos pode ser configurado como entrada analógica ou como porto digital, sendo que na placa DETPIC32 o porto B está configurado, por defeito, como porto digital[2]. A configuração completa de um dado bit **`n`** do porto B como entrada analógica envolve sempre dois passos: i) desligar a componente digital de saída do porto, isto é, fazer **`TRISBn=1`** e ii) configurar o porto como entrada analógica. A configuração como entrada analógica é feita através do registo **`AD1PCFG`**.

[1] Algumas das indicações que serão dadas ao longo deste texto são válidas apenas para o modo de funcionamento escolhido (para informações mais detalhadas deve ser consultado o manual do fabricante disponível no *moodle* de AC2).

[2] A configuração do porto B como porto digital é feita no *firmware* da placa DETPIC32, após *reset* ou *power*-up. No PIC32 todos os pinos que partilham funções analógicas estão configurados, por defeito, como entradas analógicas.

Por exemplo, para a configuração do bit 4 do porto B como entrada analógica (**AN4**) pode fazer-se:

```
TRISBbits.TRISB4  = 1;  // RB4 digital output disconnected
AD1PCFGbits.PCFG4 = 0;  // RB4 configured as analog input (AN4)
```

Seleção do canal de entrada

O *multiplexer* analógico permite selecionar, em cada instante, qual a entrada analógica que é encaminhada para o conversor A/D. A seleção do canal de entrada é efetuada através dos 4 bits do campo **CH0SA** do registo **AD1CHS**. Por exemplo, para a seleção da entrada **AN4** como entrada para o conversor A/D pode fazer-se:

```
AD1CHSbits.CH0SA = 4;   // Selects AN4 as input for the A/D converter
```

Configuração do número de conversões consecutivas do mesmo canal

O módulo A/D permite configurar o número de conversões consecutivas do mesmo canal antes de o conversor gerar o evento de fim de conversão. Essa configuração é efetuada no registo **AD1CON2** nos 4 bits do campo **SMPI**. Os valores possíveis de configuração, em binário, vão, assim, desde **0000** a **1111**, a que correspondem 1 e 16 conversões consecutivas, respetivamente. Por exemplo, para fazer 4 conversões consecutivas no canal 7 pode fazer-se:

```
AD1CHSbits.CH0SA = 7;   // Selects AN7 as input for the A/D converter
AD1CON2bits.SMPI = 3;   // 4 samples will be converted and stored
                        // in buffer locations ADC1BUF0 to ADC1BUF3
```

O *buffer* de armazenamento referido anteriormente, é preenchido em função do número de conversões que tiver sido previamente configurado, começando sempre em **ADC1BUF0**. No exemplo de cima, o sinal de fim de conversão só é gerado quando o conversor A/D tiver efetuado as 4 conversões.

Início de conversão e deteção de fim de conversão

Na configuração adotada para as aulas práticas, o módulo A/D funciona em modo manual. Significa isto que um processo de conversão só começa quando há uma ordem explícita de início. Para essa configuração, a ordem de conversão é dada através da instrução:

```
AD1CON1bits.ASAM = 1;     // Start conversion
```

Quando o módulo A/D termina uma sequência de conversão gera um pedido de interrupção (ativa o bit **AD1IF** do registo **IFS1)**. Este pedido de interrupção pode ou não ter seguimento, dependendo da configuração do sistema de interrupções. Se não estiverem a ser usadas interrupções, a deteção do evento de fim de conversão terá quer ser feita por *polling*, esperando, em ciclo, que o bit **AD1IF** transite do nível lógico 0 para o nível lógico 1. O ciclo de *polling* pode ser efetuado do seguinte modo:

```
while( IFS1bits.AD1IF == 0 );   // Wait while conversion not done
```

O bit **AD1IF** é automaticamente ativado pelo módulo A/D, mas a sua desativação é manual. Assim, terminada a operação de leitura dos valores da sequência de conversão, é sempre necessário fazer o *reset* desse bit (**IFS1bits.AD1IF = 0**).

Configuração completa do módulo A/D, incluindo a configuração do porto de entrada

Para além das apresentadas anteriormente, há ainda um conjunto de configurações adicionais que têm que ser efetuadas para que o módulo A/D funcione de acordo com o pretendido. A lista completa de configurações do módulo A/D fica então:

```
TRISBbits.TRISBx = 1;     // RBx digital output disconnected
AD1PCFGbits.PCFGx= 0;     // RBx configured as analog input (AN4)
AD1CON1bits.SSRC = 7;     // Conversion trigger selection bits: in this
                          //  mode an internal counter ends sampling and
                          //  starts conversion
AD1CON1bits.CLRASAM = 1;  // Stop conversions when the 1st A/D converter
                          //  interrupt is generated. At the same time,
                          //  hardware clears the ASAM bit
AD1CON3bits.SAMC = 16;    // Sample time is 16 TAD (TAD = 100 ns)
AD1CON2bits.SMPI = XX-1;  // Interrupt is generated after XX samples
                          //  (replace XX by the desired number of
                          //  consecutive samples)
AD1CHSbits.CH0SA = x;     // replace x by the desired input
                          //  analog channel (0 to 15)
AD1CON1bits.ON = 1;       // Enable A/D converter
                          //  This must the last command of the A/D
                          //  configuration sequence
```