Aquisição e Filtragem Digital de Sinais Analógicos

Guia de Prática Laboratorial

PAULO GARRIDO, PAULO CARVALHAL, LUÍS GONÇALVES, PEDRO VIEIRA, LUÍS LOURO

Mestrado Integrado em

Engenharia Electrónica Industrial e Computadores

Escola de Engenharia da Universidade do Minho

Versão 8.6

1. Introdução

Um conversor analógico-digital (ADC) permite tirar amostras de tensões elétricas presentes na sua entrada $v_i(t)$ e atribuir-lhes valores digitais. Como exemplo, considere-se um ADC com *resolução* de 3 bits, e uma tensão de entrada $0 \leq v_i < 1$. Os valores digitais, escritos em binário, atribuídos pelo ADC seriam nominalmente:

v_i pertence ao intervalo	ponto médio do intervalo	valores digitais em binário
[0.00000, 0.06125[		000
[0.06125, 0.18625[	0.125	001
[0.18625, 0.31125[	0.250	010
[0.31125, 0.43625[	0.375	011
[0.43625, 0.56125[	0.500	100
[0.56125, 0.68625[	0.625	101
[0.68625, 0.81125[	0.750	110
[0.81125, 1.00000[		111

Utilizando em conjunto um ADC e um computador (microprocessador, microcontrolador) é possível retirar repetidamente amostras da tensão presente na entrada do ADC e atribuir-lhes valores digitais. Este processo cria uma imagem *discreta no tempo* (porque só os valores nos instantes das amostras estão definidos) e *digitalizada em valor* da evolução da tensão elétrica ou sinal[1] analógico na entrada do ADC.

Este é o processo fundamental de *aquisição digital de um sinal analógico*. Este processo tem uma utilidade imensa, porque é possível a partir dele:

- Registar em memória digital do computador o sinal na entrada do ADC, ou seja, a evolução da tensão elétrica ou da grandeza física que ela representa. A partir do registo, é possível comunicar, visualizar e processar o sinal das mais variadas formas.
- Processar digitalmente os valores amostrados de forma a realizar diferentes funções como monitorização, filtragem, cálculo de valores de comando, etc. O processamento digital apresenta diversas vantagens em relação ao equivalente analógico: flexibilidade, programabilidade, fácil realização de quaisquer valores de parâmetros, estabilidade dos parâmetros, realização do sistema eletrónico com um número imensamente reduzido de componentes em relação ao equivalente analógico e um número imensamente aumentado de funcionalidades possível.

[1] Entende-se por *sinal*, uma grandeza elétrica (tensão ou corrente) que interpretamos como contendo *informação*.

– Uma vez realizado o processamento digital, converter os valores digitais calculados em valores analógicos atuantes por meio de conversão digital-analógica, de uma forma eficaz e eficiente.

A viabilidade do registo ou processamento digital de sinais analógicos depende da capacidade de os amostrar a uma frequência suficientemente elevada para não perder informação significativa e de converter os valores amostrados para formato digital com uma resolução suficiente para não introduzir ruído com valor significativo. Estas condições podem ser satisfeitas virtualmente para todas as aplicações de eletrónica, o que faz dos sistemas digitais uma ferramenta universal de implementação.

No processo de aquisição de um sinal analógico por um dispositivo digital, como um microcontrolador, encontraremos sempre *amostragem* e *conversão*, e, se necessário, *filtragem analógica ou digital*, *linearização* e *registo em memória*. O processo de amostragem define os intervalos de tempo entre as amostras da tensão elétrica, a conversão transforma o valor da grandeza num valor digital, a filtragem elimina componentes indesejáveis, a linearização permite obter uma correspondência linear entre os valores digitais obtidos e os valores da grandeza física, o registo guarda em memória os valores adquiridos. São estes aspetos da aquisição digital de sinais analógicos que trataremos nesta prática laboratorial e cujos fundamentos revemos a seguir.

Amostragem

A amostragem define a sequência de instantes de tempo em que o sinal proveniente da medida da grandeza física dada pelo sensor analógico é convertido para valores digitais através do ADC. Habitualmente, a amostragem é periódica pelo que o intervalo de tempo entre duas leituras consecutivas do ADC é constante e chamado de *período de amostragem*. O inverso do período de amostragem é a *frequência de amostragem*. Nas condições ideais do teorema de Nyquist, se a frequência de amostragem for pelo menos *duas vezes* superior à frequência máxima presente no sinal a adquirir, então é possível reconstruir o sinal analógico a partir das amostras digitais – não existe perda de informação. Tomando em conta que na realidade só é possível aproximar as condições do teorema de Nyquist, segue-se a "regra do polegar" de utilizar uma frequência de amostragem pelo menos *quatro* a *dez* vezes superior à frequência máxima presente no sinal a adquirir (ou que se pretende recuperar do sinal a adquirir).

Seja um processo de amostragem com período h ou frequência $f_s = 1/h$ aplicado a um sinal no tempo $x(t)$. Do processo de amostragem resulta um sinal amostrado $x^*(t)$ em tempo contínuo, ver a Figura 1, que tem a seguinte expressão:

$$\begin{cases} t = nh \rightarrow x^*(t) = x(nh) \\ t \neq nh \rightarrow x^*(t) = 0 \end{cases} \tag{1}$$

No domínio das frequências, o sinal $x^*(t)$ contém todas as frequências do sinal $x(t)$ somadas de kf_s, k = ..., –2, –1, 0, 1, 2, ... A Figura 2 mostra um exemplo.

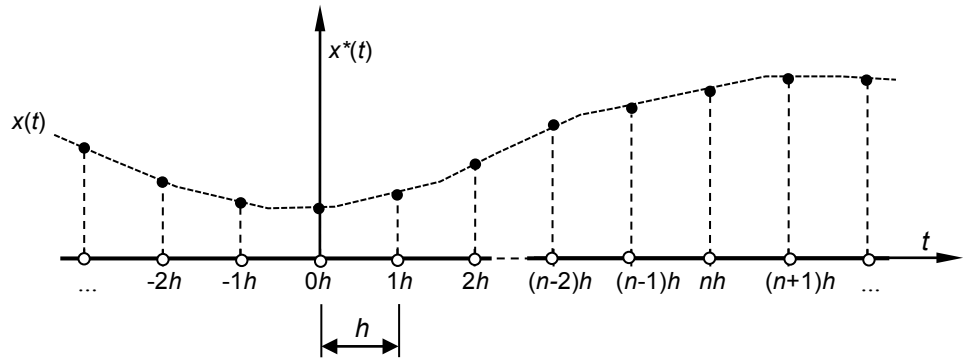

Figura 1 Sinal em tempo contínuo $x^*(t)$ resultante do processo de amostragem do sinal $x(t)$.

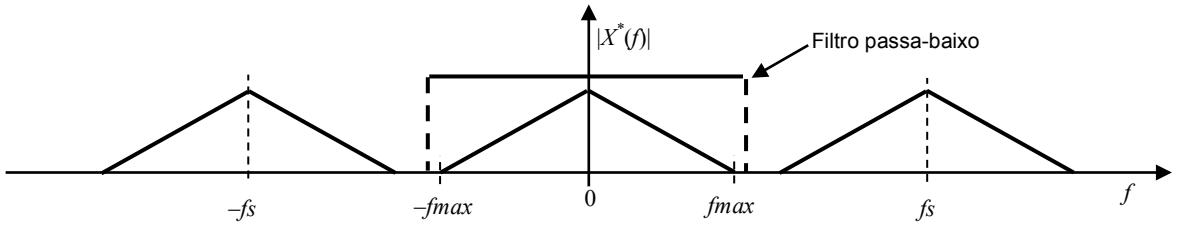

Figura 2 Espectro de amplitude, $X^*(f)$ de um sinal $x^*(t)$. Assume-se que o espectro de $x(t)$ está compreendido entre *fmax* e *–fmax*. Assume-se também que *fmax* < *fs*/2. O filtro passa-baixo ideal representado permite recuperar em teoria o espectro do sinal contínuo.

Retentor de ordem 0

Fisicamente não é o sinal amostrado $x^*(t)$ que é processado, mas sim o seu *holding* ou a sua retenção. Os valores $x(nh)$ são guardados em memória resultando o sinal $x_h(t)$:

$$nh \le t < nh + h \rightarrow x_h(t) = x(nh) \tag{2}$$

A expressão (2) realiza o chamado retentor de ordem 0. Um exemplo de sinal $x_h(t)$ é mostrado na Figura 3.

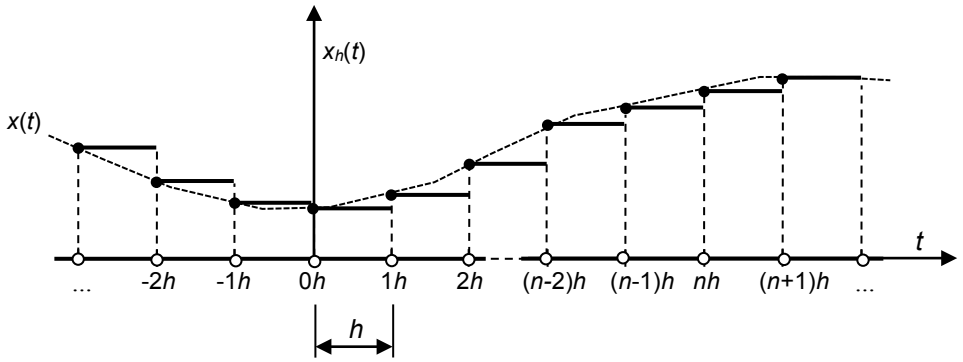

Figura 3 Sinal após amostragem e retenção de ordem 0. No intervalo de tempo [*nh*, *nh*+*h*[o valor de $x_h(t)$ é igual $x(nh)$.

Outros retentores são teoricamente possíveis, mas, na prática, são muito pouco utilizados.

Aliasing e filtragem analógica

Na Figura 4 ilustra-se o problema conhecido por *aliasing* na literatura inglesa com um sinal $x(t)$ que contém uma só componente sinusoidal de frequência 100 Hz. As amostras deste sinal, obtidas com uma frequência de amostragem de 90 Hz, indicam a existência de uma sinusoide de 10 Hz – que na realidade não existe, mas que será a informação resultante de passar o sinal amostrado num retentor de ordem 0.

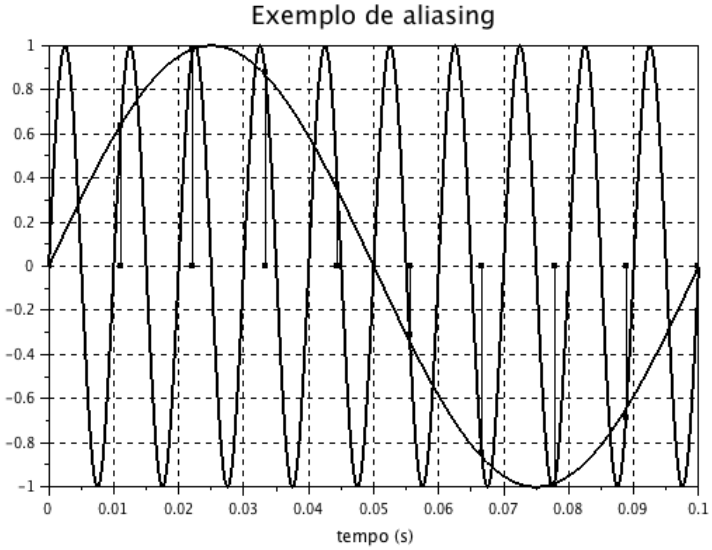

Figura 4 As amostras de uma sinusoide com 100 Hz de frequência, retiradas a uma frequência de 90 Hz coincidem com os valores que seriam retirados de uma sinusoide de 10 Hz. Note-se que o sinal $x(t)$ a ser amostrado contém uma só componente sinusoidal com frequência $f = 100$. O sinal amostrado $x^*(t)$corresponde a $k = -1$ na expressão kf_s dos *aliases*, visto que $10 = 100 - 90$.

Em geral, num espectro contínuo, altas frequências do sinal $x(t)$ aparecerão como baixas frequências no sinal $x^*(t)$, do que resulta distorção do sinal, mesmo utilizando um filtro passa-baixo ideal. Veja-se a Figura 5.

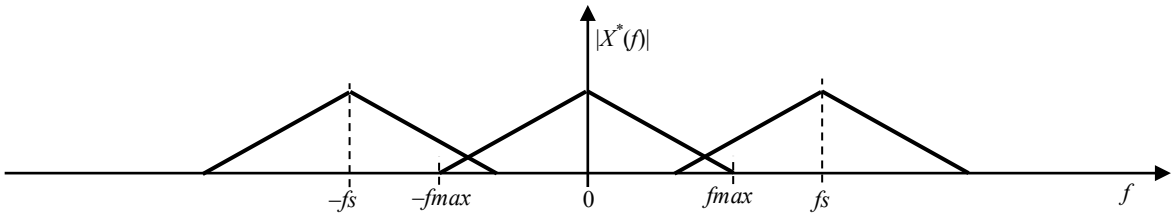

Figura 5 Fenómeno de *aliasing* mostrando o espectro de amplitude de um sinal $x^*(t)$ quando $fmax > fs/2$. Assume-se que o espectro de $x(t)$ está compreendido entre *fmax* e *–fmax*, sendo nulo fora deste intervalo.

Para resolver este problema podem usar-se duas aproximações: aumentar a frequência de amostragem ou filtrar analogicamente o sinal $x(t)$ de forma a que as frequências que produzem *aliasing* passem a ter um valor insignificante. Em relação à Figura 5, no primeiro caso resultaria o espectro para o sinal amostrado representado na Figura 2.

Conversão

A conversão analógica-digital transforma um valor amostrado de tensão num valor digital. Os dois parâmetros mais relevantes na conversão são o *número de bits* de resolução

do ADC e o seu *tempo de conversão*. Quanto maior a resolução, menor é o *ruído de quantificação* introduzido pelo ADC. O ruído de quantificação resulta do sinal amostrado e a seguir "retido" pelo retentor de ordem 0 diferir do sinal analógico por uma quantidade que é no máximo igual a $v_{max}/2^n$, em que v_{max} é a tensão máxima que o ADC é capaz de converter e n é o seu número de bits – se a frequência de amostragem for suficientemente alta. Nesta condição, a relação sinal-ruído de quantificação, *SNR* de um ADC é, aproximadamente, dada pela expressão:

$$SNR = 6.02n \text{ dB} \tag{3}$$

Um conversor de 8 bits terá uma *SNR* de aproximadamente 48 dB, enquanto que um de 10 bits terá uma *SNR* de aproximadamente 60 dB. Note-se que estamos a falar de valores aproximados, apenas para o ruído de quantificação e na condição de durante o período de amostragem a variação do sinal não exceder o *quanto* $v_{max}/2^n$.

O tempo de conversão é definido como o tempo máximo que decorre entre o instante em que o ADC recebe o impulso de início de conversão e o instante em que sinaliza que a conversão está completa e o valor digital pode ser lido. Este tempo deve ser sempre (muito) inferior ao período de amostragem e como tal estabelece um valor máximo para a frequência de amostragem.

Filtragem digital

A filtragem digital implica o cálculo numérico do valor de tensão que deve ser a saída do filtro e o seu *output*. A filtragem digital tem muitas vantagens em relação à sua equivalente analógica, como explicado acima. Tal como a filtragem analógica, a filtragem digital pode ser usada para atenuar ruído de baixa frequência (em relação à frequência de amostragem) no sinal amostrado.

Nos filtros digitais, cada valor em tempo discreto $y[n]$ de saída do filtro é calculado tendo em conta a leitura realizada no ADC, $x[n]$, os valores de entrada passados, $x[n–k]$ e os valores de saída do filtro $y[n–k]$, calculados nas iterações anteriores. Na expressão (4) de um filtro digital linear, os valores de a_k são interpretados como os pesos a dar aos valores anteriores de saída do filtro e os valores de b_k como os pesos a dar aos valores anteriores de entrada do filtro, de acordo com a expressão:

$$y[n] = a_1 y[n-1] + ... + a_N y[n-N] + b_0 x[n] + b_1 x[n-1] + ... + b_M x[n-M] \tag{4}$$

Se todos os a_k forem 0, então a resposta do filtro a um impulso unitário torna-se 0 ao fim de M períodos e o filtro é conhecido por FIR (*Finite Impulse Response*). Caso contrário, o filtro é IIR (*Infinite Impulse Response*).

Implementação digital de um filtro RC passa-baixo

Considere-se um filtro IIR de primeira ordem com a seguinte expressão:

$$y[n] = ay[n-1] + (1-a)x[n-1] \tag{5}$$

O valor que pesa a leitura do ADC é feita igual a $(1 – a)$ de forma a que o filtro tenha ganho em regime permanente igual a 1. Se a aquisição do sinal for efectuada a interva-

los de tempo constantes (amostragem periódica), o valor de a é constante.

O filtro em (5) realiza um filtro digital similar[2] a um filtro analógico RC. A resposta do filtro a um degrau de tensão corresponde à amostragem da resposta de um circuito RC, cuja constante de tempo $T = RC$ satisfaz as expressões:

$$a = \exp(-h/T) \Leftrightarrow T = -h/\ln a \tag{6}$$

O valor de a varia no intervalo]0, 1[. Com valores de a perto de 0, o valor de $y[n]$ acompanha facilmente o valor de $x[n]$. Com valores de a perto da unidade, as variações no valor de $x[n]$ refletem-se lentamente no valor de $y[n]$. A Figura 6 mostra a resposta do filtro a um degrau de tensão na entrada do ADC, com valores de a de 0.1, 0.3 e 0.9.

Este filtro tem a característica de ser passa-baixo. Como tal permite reduzir o efeito de sinais aleatórios de frequência elevada (ruído). Na Figura 7 apresenta-se a resposta do filtro a um sinal sinusoidal, ao qual foi sobreposto um pico de ruído. O valor de a deve ser escolhido em função da frequência máxima prevista na variação da grandeza a medir, de forma a não interferir significativamente nas variações desta, e reduzir os efeitos de sinais de frequência mais elevada.

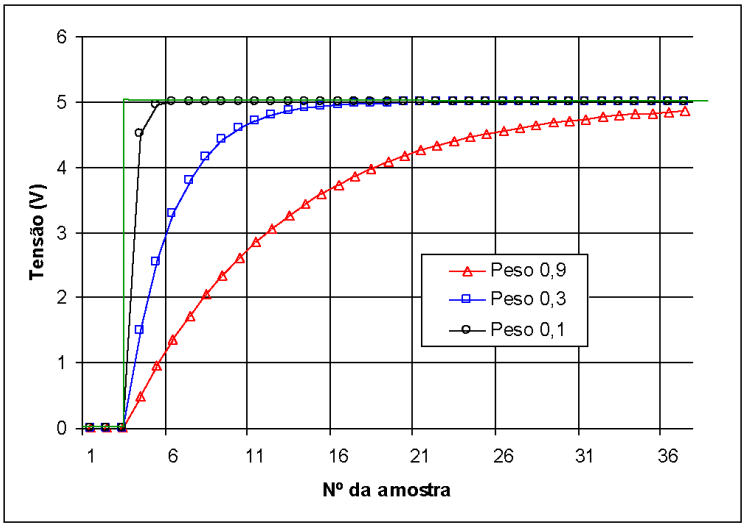

Figura 6 Resposta do filtro passa-baixo a um degrau de tensão na entrada do ADC, com valores de a de 0.1, 0.3 e 0.9.

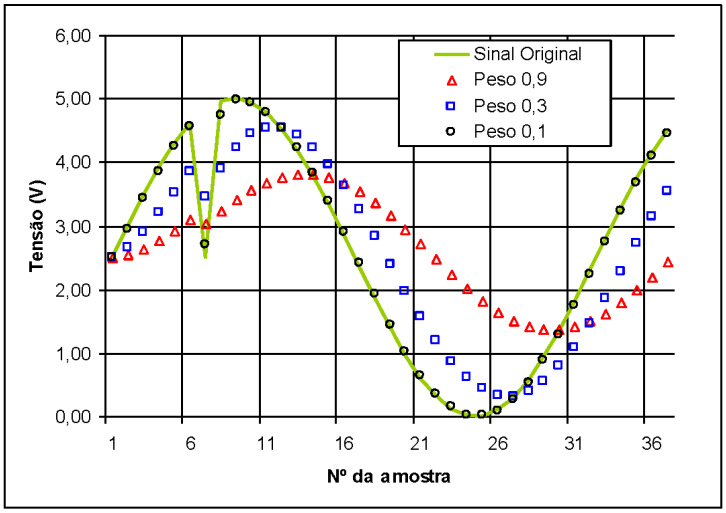

Figura 7 Resposta do filtro passa-baixo a um impulso de ruído sobreposto a um sinal sinusoidal, na entrada do ADC, com valores de a de 0.1, 0.3 e 0.9.

[2] A resposta em frequência de um filtro digital difere da resposta do filtro analógico de que foi mapeado pelo warping.

Filtros FIR

Os filtros FIR (*Finite Impulse Response*) são conhecidos por terem uma resposta impulsional de duração finita porque a saída de um filtro FIR iguala a soma ponderada de um número finito de valores passados e do valor presente de entrada do filtro. Uma vez que estes filtros apresentam resposta em frequência com fase linear, o seu projeto resume-se na aproximação da resposta em módulo que se deseja.

Um filtro passa-baixo tem como objetivo passar sinais com frequência abaixo da frequência de corte e atenuar sinais com frequência acima da frequência de corte. Na Figura 8 é possível ver a resposta habitual em frequência a um filtro deste tipo, sendo a banda passante o conjunto de frequências até a resposta começar a diminuir; a banda de transição o conjunto de frequências onde a resposta tem um decréscimo e a banda de rejeição o conjunto de frequências que serão rejeitadas pelo filtro. Além das três bandas de frequências existe também o *ripple*, que consiste na variação de amplitude que ocorre nas bandas. No caso de um filtro FIR, o valor máximo do *ripple* é o mesmo nas bandas passante e de rejeição. Um filtro é tanto melhor quanto menores o *ripple* e a largura da banda de transição.

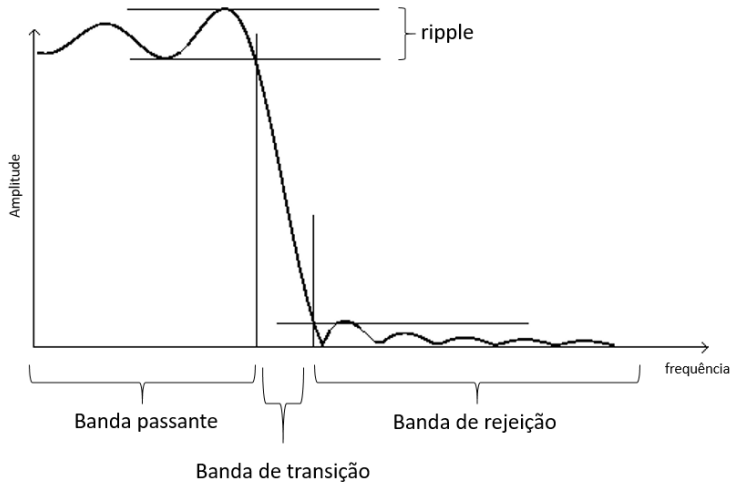

Figura 8 Resposta padrão em frequência de um filtro passa-baixo com a delimitação das três bandas e do ripple.

Uma das estratégias para determinar um filtro FIR é o método das janelas. A Janela de Kaiser é uma das janelas mais utilizadas, sendo descrita pela seguinte equação, onde $I_0(.)$ é uma função de Bessel modificada do 1º tipo e de ordem zero

$$b[k] = \begin{cases} \dfrac{I_0\left[\beta\left(1-\left[\dfrac{k-M/2}{M/2}\right]^2\right)^{1/2}\right]}{I_0(\beta)}; & 0 \le n \le M \\ 0; \quad outros\ casos \end{cases} \tag{7}$$

A expressão anterior fornece os coeficientes b_0 a b_M. Para se utilizar a janela de Kaiser é necessário estabelecer:

1. A frequência de corte (Ω_c) e a largura da banda de transição ($\Delta\Omega$). Não esquecer que estes valores se referem às frequências normalizadas, ou seja $\Omega = \omega h$, onde ω representa a frequência angular (rad/s) e h o período de amostragem do sinal (s).
2. O inverso do *ripple* pretendido (A) em dB.

Os parâmetros desta janela são obtidos de forma empírica, tendo em conta as características pretendidas para o filtro:

$$A = -20\log\delta \tag{8}$$

$$\Delta\Omega = \Omega_s - \Omega_p \tag{9}$$

$$\Omega_c = \frac{\Omega_s + \Omega_p}{2} \tag{10}$$

$$M = \frac{A-8}{2.285\Delta\Omega} \tag{11}$$

$$\beta = \begin{cases} 0.1102\left(A-8.7\right); & A > 50 \\ 0.5842\left(A-21\right)^{0.4} + 0.07886\left(A-21\right); & 21 \le A \le 50 \\ 0.0; & A < 21 \end{cases} \tag{12}$$

Como se pode ver na Equação (11), a ordem do filtro (M) é tanto maior, quanto menor o ripple (inverso de A) e quanto menor a largura da banda de transição. Por isso, logicamente, quanto maior a ordem do filtro, melhor a qualidade do mesmo.

Estes parâmetros podem ser calculados usando o Matlab, tal como é referido no Objetivo 2.

Computação em tempo real

Numa primeira aproximação, entende-se por computação em tempo real operações de processamento computacional submetidas a uma restrição de serem realizadas num intervalo de tempo máximo, de forma que a resposta a um certo evento seja garantidamente dada no intervalo de tempo.

Suponha-se que se pretende realizar um filtro digital com uma frequência de amostragem de 1 kHz. Isto significa que o sistema eletrónico deve amostrar o sinal analógico de entrada de 1 ms em 1 ms. Se além disso especificarmos que o atraso entre a recolha da amostra e a saída do filtro não deve ser superior a 1/10 do período de amostragem, isto significa que o intervalo de tempo máximo permissível para obter a amostra, converter para valor digital, calcular o valor de saída e fazer o seu *output*, é de 100 µs, submetido à restrição adicional de o processo iniciar à passagem de cada milissegundo.

2. Arquitetura de software a utilizar

O objetivo geral da PL é integrar na interface, desenvolvida na prática laboratorial anterior, a funcionalidade de aquisição de valores de sinais analógicos utilizando amostragem periódica com possibilidade de filtragem linear e envio dos valores adquiridos pela interface série.

Dessa forma, espera-se que sejam desenvolvidos módulos, à imagem do guia anterior, baseados no modelo produtor/consumidor e em programação concorrente. Neste caso, o processo produtor será baseado em interrupções invocadas por temporizador, que despoletará a aquisição de sinal por parte do ADC do microcontrolador. O valor adquirido será disponibilizado por partilha de memória com o módulo main. A este módulo será acrescentado o código funcional deste guia.

Descrição detalhada do software a desenvolver

A necessidade de amostrar periodicamente o sinal a uma frequência bem definida impõe que o processo de aquisição seja programado numa rotina de serviço de interrupção com prioridade máxima, invocada por um temporizador. A manipulação de parâmetros associados à rotina e às operações que a rotina executa é realizada no programa principal ou de interface com o utilizador.

Suponha-se que se pretende amostrar um sinal analógico com um período *h* de 10 ms, logo com frequência de amostragem de 100 Hz.

Na secção do programa principal que inicializa o sistema, programa-se um temporizador do microprocessador para provocar uma interrupção a cada 10 ms com prioridade máxima. Se não for possível o temporizador fazer *auto-reload* do valor do período, então a primeira operação da rotina será programar o temporizador para produzir uma nova interrupção ao fim de 10 ms.

Desta forma, de 10 em 10 ms, o código da rotina de serviço de interrupção é executado.

Descrito em alto nível, este código terá no mínimo a operação de leitura do ADC

```
x = read(ADC_channel)
```

sendo `x` uma variável para a qual se lê o resultado da operação de conversão de uma amostra do sinal analógico para um valor digital, realizada no canal `ADC_channel`. A expressão "canal de um ADC" advém de um conversor analógico-digital ter usualmente associado um multiplexador analógico que lhe permite converter tantas tensões diferentes quantas as que estão presentes nas entrada do multiplexador.

Realizada a operação de leitura do ADC, o valor de `x` pode ser processado. O processamento de `x` pode consistir apenas no seu valor ser enviado para um *buffer* de dados. Ou então, o valor a ser enviado é calculado a partir de `x` e de outros valores, como quando se usa filtragem ou linearização. No primeiro caso, a operação pode ser descrita como

```
output(x)
```

Em que `output` significa a operação de enviar `x` para o *buffer*.

No segundo caso, a operação pode ser descrita como

```
output(y)
```

Em que `y` é o valor a ser enviado após filtragem. Vejamos agora como calcular este valor.

Implementação de um filtro digital

O instante atual é considerado ser o instante *n*. Segue-se que o valor `x` lido do ADC será representado como *x*[*n*] e o valor de saída do filtro `y` será representado como *y*[*n*]. Suponha-se que o filtro é linear. Como descrito anteriormente, teremos:

$$y[n] = a_1 y[n-1] + ... + a_N y[n-N] + b_0 x[n] + b_1 x[n-1] + ... + b_M x[n-M] \qquad (13)$$

O valor de *y* para o instante *n* (atual) é calculado como uma soma de:

– Uma média pesada dos *N* valores anteriores de *y*,

– Uma média pesada do valor atual de *x* (lido do ADC) e dos *M* valores anteriores de *x*.

Para calcular o valor de `y` será necessário ter em memória, devidamente definidas e com valores consistentes, as seguintes variáveis:

– Os valores de *M* e *N*: `M,N`

– *Arrays* em que são guardados os coeficientes a_1 a a_N e b_0 a b_M: `a,b`

– *Arrays* em que são guardados os valores anteriores de *y* e *x*: `y_ant`, `x_ant`.

O elemento `x_ant(0)` terá o significado especial de ser o valor de `x` lido do ADC no instante *n*. O elemento `x_ant(1)` terá o significado de ser o valor de `x` lido do ADC no instante *n*–1 e assim sucessivamente.

O elemento `y_ant(0)` terá o significado especial de ser o valor de `y` calculado no instante *n*. O elemento `y_ant(1)` terá o significado de ser o valor de `y` calculado no instante *n*–1 e assim sucessivamente.

Quando a rotina de interrupção começa a executar no instante *n*, os valores nos *arrays* referem-se ao instante *n*–1, logo a rotina deverá atualizá-los.

Assumindo que estas variáveis são globais na rotina de interrupção, a seguinte subrotina calculará o valor de `y`:

```
calc_y()
//Atualizar os valores de x_ant de 0 a M e de y_ant de 1 a N para o
instante presente n
for i = M to 1    //Ir do valor mais antigo para o mais recente
      x_ant(i)=x_ant(i-1)
      end
x_ant(0)=x
```

```
for i = N to 1
      y_ant(i)=y_ant(i-1)
      end
//Calcular y
y=0
for i = 0 to M
      y=y+b(i)*x_ant(i)
      end
for i = 1 to N
      y=y+a(i)*y_ant(i)
      end
//Atualizar o valor y(0)
y(0)=y
return
```

Para a descrição das operações da rotina de serviço da interrupção, convém assumir a existência de uma *flag* que indica se o filtro está ativo: `filter_on`. Ter-se-á:

```
begin
reload(timer) // Se necessário
x=read(ADC_channel)
if filter_on == 0 output(x) else calc_y()
output(y)
return
```

Saída (*output*) de um filtro digital

O valor `y` resultante da rotina de cálculo do valor $y[n]$ pode ser enviado para dois tipos de destino diferentes:

A) Para um porto ou dispositivo que o transforma num valor de analógico de tensão, por exemplo, um conversor digital-analógico (DAC).

B) Para memória onde ficará registado para posterior transmissão e/ou armazenamento.

No caso A, estar-se-á a realizar um filtro digital no sentido mais comum do termo (*on-line*). Considere-se um filtro analógico: a sua tensão de entrada evolui como uma função $x(t)$; a sua tensão de saída evolui como uma função $y(t)$, a qual é a pretendida versão filtrada de $x(t)$. A versão filtrada de $x(t)$ é obtida com recurso a eletrónica analógica.

Se o filtro for digital, tudo se passa da mesma forma do ponto de vista do exterior. Muda é a forma de obter a tensão de saída: primeiro, a tensão de entrada é amostrada com período h. Segundo, em cada período n, o valor da tensão de saída para, nominalmente,

o instante *nh* é calculado numericamente. Este valor é enviado para um DAC que o converte numa tensão contínua.

No caso B, estar-se-á a adquirir dados numa forma já filtrada (antes de proceder ao seu armazenamento).

No seguimento, indica-se como programar o processo de aquisição. Mas sugere-se que os alunos que disponham de microcontroladores com canais de conversão DAC implementem também um filtro digital *on-line* (caso A) nos objetivos 1 e 2.

Buffers: estrutura e valores

Os valores adquiridos (valores de saída do filtro ou os diretamente lidos do ADC, se o filtro estiver inativo) deverão ser colocados num *buffer* de memória do microcontrolador, a que chamaremos o *buffer* de valores. Num segundo *buffer*, colocar-se-ão os números *n* de sequência dos valores, sendo contados a partir da recepção do comando de início de amostragem. Resulta uma estrutura que pode ser descrita como:

Buffer de valores adquiridos	Val1	Val2	Val3	...
Buffer de valores de *n*	1	2	3	...

O registo nos *buffers* acima dos valores adquiridos será realizado pela programação do módulo `output(.)`, cujos detalhes não se mostram aqui.

Envio dos valores adquiridos pela interface série

Os valores adquiridos serão enviados do *buffer* para o PC através da interface série. Imediatamente antes de cada valor adquirido, deverá ser enviado o valor *n* associado. Os valores adquiridos e os valores de *n* deverão ser enviados como números em base 10 codificados em carateres ASCII. Os valores lidos diretamente do ADC, se este tiver 10 bits, estarão no intervalo [0, 1023].

Sugere-se o uso da seguinte "trama" para o envio de cada valor:

n<dig>$^{+}$βv<dig>$^{+}$<CR>

Como exemplo, o resultado da visualização de quatro tramas poderia aparecer no emulador de terminal correndo no PC como:

n20 v500
n21 v502
n22 v508
n23 v490

As tramas podem ser implementadas com comprimento fixo (os valores são sempre apresentados no terminal como uma coluna) ou variável (minimiza o tempo de transmissão). Pode ser necessário usar inteiros (ou fracionários) com sinal para os valores de saída do filtro.

O processo de envio pela interface série deve ser programado *fora* do módulo `output(.)` Esta restrição explica-se porque, pertencendo este módulo à rotina de inter-

rupção, o envio da informação pela porta série aumentaria o tempo de processamento da rotina e diminuiria o valor máximo possível da frequência de amostragem.

Indica-se um possível esquema para implementar o processo de envio. Criar no programa principal um teste de uma *flag* (colocada a 1 pela função `output(.)` em cada escrita no *buffer*) que indica existirem valores no *buffer* para enviar. Se o teste for positivo, o programa principal cria uma trama com o primeiro par de valores, *n* e de aquisição, que não foram enviados e passa a trama à rotina de interrupção de envio da USART.

Comandos a implementar

Os comandos a implementar deverão aumentar os comandos interpretados pelo *parser* e executor de comandos no programa principal. São os seguintes.

a) Definir o período de amostragem:

(i) **Sampling Period:** <char>$^{+}$="SP β <timeunit> β <units>"

b) Definir o canal a ser amostrado:

(ii) **Analog Channel:** <char>$^{+}$="AC β <addr3>"

c) Ativar ou desativar o filtro:

(iii) **Filter oN:** <char>$^{+}$="FN"
(iv) **Filter ofF:** <char>$^{+}$="FF"

O comando FN deve colocar `filter_on` a 1 e deve colocar a 0 os conteúdos dos *arrays* `x_ant` e `y_ant` , de forma a que a filtragem comece sempre numa situação bem definida.

Os parâmetros e variáveis do filtro serão definidos na compilação do programa não sendo objeto de modificação através da interface.

d) Iniciar ou parar a aquisição de dados:

(v) **Sample:** <char>$^{+}$="S"
(vi) **Sample only K values:** <char>$^{+}$="S β <dig>$^{+}$"
(vii) **Stop sampling:** <char>$^{+}$="ST"

O comando ST parará qualquer processo de aquisição que esteja ativo. O comando S inicia uma sequência ilimitada de aquisição e envio. O comando S seguido de um valor *K* inicia uma sequência de aquisição e envio que terminará após ser adquirida e enviada a *K*–ésima amostra. Em qualquer caso, o valor de *n*, número de ordem das amostras, deve ser inicializado de forma a que cada sequência comece sempre em 1.

Para tomar em conta os casos em que o número de valores a adquirir será superior ao número de valores que o *buffer* pode conter, a escrita neste, feita pelo módulo `output(.)`, deve ser feita de modo "circular". Quando `output(.)` escreve na última posição disponível do *buffer*, aponta a próxima escrita para o início. Isto implica destruir os valores registados nas posições em que se escreverá – estes deverão ter sido enviados pela interface série, entretanto.

A função do *buffer* é impedir que os valores adquiridos se percam no caso de o tempo de envio de uma trama pela interface série ser superior ao período de amostragem: a variável com o valor adquirido seria atualizada para o instante *n* antes que o valor do instante *n*–1 tivesse sido transmitido e este seria perdido. Com o *buffer*, não existirão perdas se o número de valores a enviar, *K*, for inferior ou igual à capacidade do mesmo. Se *K* for maior ou a sequência de aquisição for ilimitada, não existirão perdas se a frequência média de transmissão de valores no canal série for igual ou superior à frequência de amostragem.

Sugere-se que todos estes comandos funcionem utilizando um contexto comum. Isto quer dizer que se se executar a sequência de comandos a seguir indicada:

SP s 1
AC 1
FN
S 200

O microcontrolador vai adquirir e enviar pela interface série 200 amostras do canal número 1 do ADC com uma taxa de amostragem de 1 Hz, utilizando o filtro digital cujos coeficientes foram especificados na compilação do programa.

Relembra-se a gramática geral que regula a troca de dados ao nível da interface com utilizador, com os acrescentos relativos a esta PL.

COMANDO = <char>$^{+}$<CR>|<char>$^{+}$<control char>
<char> = {'a'..'z','A'..'Z'} U [0..9] U { β, <bckspc>}
<control char> = {<esc>,'$'}
´β´ =20h
<esc>=1Bh
<bckspc>=08h
´$´=24h
<CR>=0Dh

<start>,<end>,<addr>,<org>,<dest>,<value>,<units> = <16 bit value>
<timeunits> = micro|ms|s
<dig> = {"0".."9"}
<16 bit value> = {"0000".."FFFF" |"0".."65536" | "0".."1111 1111 1111 1111"}
<bitvalue> = {"0","1"}
<length>,<byte>,<addr8> = {"0".."255" | "00".."FF" | "0" .. "1111 1111"}
<addr3> = {"0".."7"}

3. Objectivos a avaliar

Objetivo 1

Compilar o programa para realizar um filtro como em (5) com $a = 0.4$, $(1–a) = 0.6$. Adquirir sinais analógicos produzidos por um gerador de sinal (ondas quadrada, triangular e sinusoidal). Atenção a não aplicar tensões negativas na entrada do ADC – usar a função de polarização positiva do gerador. A frequência de amostragem a usar deverá ser no mínimo 10 vezes superior à frequência do sinal a amostrar. Os valores recebidos no terminal poderão ser passados por *copy paste* para uma folha Excel, o que facilitará a sua visualização em gráfico.

Demonstrar o funcionamento dos comandos de aquisição (S, sem e com *K*, e ST), de mudança de parâmetros (SP e AC), de filtragem (FN e FF) e a realização da aquisição.

Dispondo de um microcontrolador com canal de conversão digital-analógica, é de todo o interesse formativo que, na rotina de interrupção, após o cálculo de `y`, os alunos incluam um comando de envio deste para o conversor DA. Desta forma, realizam um filtro digital *on-line* (sentido usual do termo) e podem observar em osciloscópio os sinais filtrados, evitando o processo de fazer a sua visualização através do Excel.

Objetivo 2

Pretende-se desenhar e implementar um filtro FIR, usando a janela de Kaiser, que tenha um *A* de pelo menos 30 dB e uma banda de transição ($\Delta\Omega$) que não ultrapasse os 0,3π rad. Deixa-se ao critério dos alunos a escolha da frequência máxima da banda passante em Hz.

Determinar os coeficientes do filtro, usando o Matlab (ou alguma ferramenta similar) e as funções *fir1* e *kaiserord*. Estas têm por argumentos:

i) **F**, um vetor com as frequências Ω_s e Ω_p,

ii) **amp**, um vetor com as amplitudes pretendidas para as bandas do filtro, como o filtro pretendido é um filtro passa-baixo, o vetor será [1 0],

iii) **A**, o valor de desvio máximo dos *ripples* e

iv) **h**, a frequência de amostragem do sinal.

```
[M,Wn,beta,FILTYPE] = kaiserord(F,amp,A,h)
b = fir1(M,Wn,ftype,kaiser(M+1,beta),'noscale')
```

Com este filtro passa-baixo, quando a entrada apresenta sinais com frequência acima da frequência de corte especificada essas componentes são atenuadas, sendo um filtro muito útil para retirar ruído de um sinal.

Compilar o programa para realizar o filtro. Aplicar três sinais sinusoidais um em cada banda e adquirir as respostas do filtro. Uma vez adquirido o conjunto de leituras no computador portátil, analisar o seu traçado gráfico.

Dispondo de um microcontrolador com canal de conversão digital-analógica, será de todo o interesse visualizar a saída do filtro em osciloscópio. Desta forma será possível testar o filtro mudando manualmente a frequência de entrada no gerador de sinais.

Objetivo 3

O tempo de cálculo da saída `y` do filtro é um fator determinante do mínimo período de amostragem (ou máxima frequência de amostragem) que se pode obter com o microcontrolador e a programação realizada. Assim, é importante determinar experimentalmente o tempo de cálculo.

Para tal, compilar um programa com um filtro com, por exemplo, $N = 10$ e $M = 10$, os valores dos coeficientes a_k e b_k sendo quaisquer desde que não-nulos. Na entrada na rotina de interrupção, colocar um pino de I/O a 1. Na saída, pô-lo a 0. Aplicar um sinal sinusoidal e observar no osciloscópio a tensão no pino. Deverá observar-se uma tensão periódica em que os intervalos de tempo a 1 correspondem ao tempo de cálculo. Como é óbvio, o período de amostragem deverá ser superior ao tempo de cálculo.

A relação entre o intervalo de tempo de cálculo e a máxima frequência de amostragem depende de se utilizar o cálculo do filtro para os objetivos anteriormente indicados como A (filtro digital *on-line*) ou B (filtragem de valores adquiridos) e, no primeiro caso, de o coeficiente b_0 ser diferente de 0, ou não.

Na filtragem de valores adquiridos, o mínimo período de amostragem deverá ser superior ao tempo de cálculo na medida do necessário para permitir a realização de outras funções durante o período. Note-se que para minimizar este tempo, a transmissão de dados via porta série deve ser feita através de ISR.

Se o dispositivo for usado como filtro digital *on-line*, o mínimo período de amostragem depende de o coeficiente b_0 ser nulo ou não. Se o for, no instante n é possível calcular o valor de $y[n+1]$ no instante n. Considere-se o exemplo do filtro IIR dado acima:

$$y[n] = ay[n-1] + (1-a)x[n-1] \tag{14}$$

Esta expressão calcula (no período n) o valor de y para o instante n indo buscar a memória os valores de y e x do instante n–1. Mas a expressão é equivalente a:

$$y[n+1] = ay[n] + (1-a)x[n] \tag{15}$$

Com esta expressão, calcula-se (no período n) o valor de y para o instante n+1 indo buscar a memória o valor de y para o instante n (calculado no período n–1) e o valor de x acabado de ler do ADC. Notar que o uso de uma expressão deste tipo implica mudar os pseudo-códigos do cálculo do filtro e da ISR dados anteriormente.

Nesta situação, o mínimo período de amostragem vem como no caso B.

Se o coeficiente b_0 não for nulo então o valor de $y[n]$ só pode ser calculado *depois* de o valor de $x[n]$ ser lido do ADC para `x`. Devido ao tempo de cálculo, existirá sempre um atraso T_a entre o instante de tempo em que o valor calculado `y` deveria ser idealmente colocado no DAC e o instante em que ele é efetivamente colocado. Este atraso aumenta o desvio de fase do filtro em relação ao valor nominal. Se, por exemplo, se quiser limitar o atraso a 1/10 do período de amostragem, este não poderá ser inferior a 10 vezes o tempo de cálculo.

Objetivo 4

Verificar o fenómeno de *aliasing*, aparecimento de baixas frequências no sinal amostrado, quando a frequência de amostragem é inferior ao dobro da máxima frequência do sinal a amostrar. Para tal, aplicar uma onda sinusoidal com frequência de 100 Hz e recolher 100 amostras da mesma a uma frequência de 90 Hz. Verificar que se obtém a amostragem de uma onda sinusoidal de frequência igual a 10 Hz.