INTRUMENTAÇÃO
LAB 06
Determinação da constante de tempo de um sensor de temperatura com a placa de aquisição de dados Data Translation DT2814
(*http://diana.uceh.ualg.pt/Inst/lab06.pdf***)**

1. Introdução

Com este projecto determinar a constante de tempo de um sensor de temperatura e simultaneamente familiarizar o aluno com uma placa de aquisição de dados típica: a placa Data Translation DT2814.

Para estudar o tempo de resposta de um sensor e formalmente obter a constante de tempo é necessário submeter o sensor a uma variação brusca da grandeza que este mede (neste caso temperatura). Dado que um sensor de temperatura (exemplo, termistor) pode ter um tempo de resposta na ordem das dezenas de milisegundos, é necessário fazer uma aquisição automática e rápida do sinal. É pois necessária um sistema de aquisição de dados.

A placa DT2814 é composta basicamente por
- um conversor analógico digital de 12 bits de resolução
- um multiplexer de 16 canais de entrada
- um relógio de tempo real

É pré-requisito para este trabalho que o aluno sabe fazer o circuito condicionador de um sensor de temperatura, por exemplo o termistor (estudado no lab03: http//diana.uceh.ualg.pt/Inst/lab03.pdf)
ou sabe como por a funcionar um sensor de temperatura em circuito integrado, por exemplo o LM35 (estudado no lab02: http//diana.uceh.ualg.pt/Inst/lab02.pdf)

2. Montagem experimental

Realize o circuito condicionador do sensor que pretende estudar a constante de tempo.

O circuito condicionador tem que ter um sinal de saída entre 0V e +5V na gama de variação de temperatura em estudo. ***Os sinais de tensão não podem ultrapassar os valores*** **[0 V, +5V]** ***sob risco do conversor A/D ficar destruído.***

As entradas do multiplexer encontram-se no painel de contactos DT757.
O painel tem a seguinte configuração:

TB1 (em baixo)		TB2 (em cima)	
terminal	Descrição	terminal	Descrição
1	canal 8	1	canal 0
2	canal 9	2	canal 1
3	canal 10	3	canal 2
4	canal 11	4	canal 3
5	canal 12	5	canal 4
6	canal 13	6	canal 5
7	canal 14	7	canal 6
8	canal 15	8	canal 7
9	Signal	9	Power
Ground		Ground	
10	- 12V Out	10	+ 12V Out

Mais informações sobre esta placa encontram-se no manual [1].

Ligue o sinal proveniente do sensor a um dos canais do multiplexer (exemplo, canal 0, TB2 terminal 1). Não se esqueça de ligar a linha de referência do sensor a Signal Ground (TB1 terminal 9)

3. Aquisição do sinal

O programa de aquisição de dados pode ser feito em QuickBasic (`c:\QB`) ou em Borland C (`c:\bc`)

3.1 Programa em Quick Basic

```
REM DT2814 example program
REM input channel is channel 0
20 N = 0
REM write in  control register and start conversion
30 OUT &H220, N
REM read control register and wait for end of conversion (bit 7)
40 a = INP(&H220)
50 IF a >= 128 THEN 40 ELSE 60
REM read low byte and high byte
60 hbyte = INP(&H221)
70 lbyte = INP(&H221)
REM convert to decimal
REM shift hbyte 4 positions to the right
```

```
REM shift lbyte 4 positions to the left
80 result = hbyte * 16 + lbyte / 16
90 PRINT "result= "; result
REM wait until key press
95 PRINT "press any key..."
100 SLEEP
REM do it all over again
110 GOTO 30
120 END
```

Alguns comentários ajudam a compreender o programa

- a placa DT2814 está instalada no PC com o endereço 0x220 (em hexadecimal)
- Esse endereço coincide com o endereço do registo de controlo da placa. É um registo de 8 bits em que cada bit tem a função indicada na Figura 1. De particular interesse são os bits C0 C1 C2 C3 que permitem escolher o canal do multiplexer.
- Sempre que se escreve neste registo o conversor A/D inicia imediatamente uma conversão
- Para se saber quando a conversão acabou lê-se o Bit 7 desse mesmo registo
- Para se saber o resultado da conversão lê-se o registo de dados no endereço 0x221 (em hexadecimal) duas vezes consecutivas: na primeira obtem-se os bits mais significativos e na segunda os bits menos significativos no formato indicado na Figura 2. Logo é necessário fazer alguns "shifts" para a esquerda e para a direita para pôr os bits no sitio certo…

B7	B6	B5	B4	B3	B2	B1	B0
F2	F1	F0	ENB	C3	C2	C1	C0

(a) escrita

B7	B6	B5	B4	B3	B2	B1	B0
FINISH	ERR	BUSY	ENB	C3	C2	C1	C0

(b) leitura

Figura 1: Registo de Controlo da placa DT2814 (endereço 0x220)

B7	B6	B5	B4	B3	B2	B1	B0
D11	D10	D9	D8	D7	D6	D5	D4

(a) primeira leitura - high byte

B7	B6	B5	B4	B3	B2	B1	B0
D3	D2	D1	D0	0	0	0	0

(a) segunda leitura -low byte

Figura 2: Formato dos bits no registo de dados (endereço 0x221)

3.2 Programa em C

O programa de aquisição dos dados feito na linguagem Borland C (`c:\bc`) serve-se de uma livraria (e de um device-driver) distribuidos com a placa, que fornece um conjunto de subrotinas de alto nível que escondem do programador todos os detalhes do hardware.

De um conjunto de subrotinas muito completo, é importante destacar as seguintes:

```
lp_initialize()   inicializa a placa DT2814
lp_adc_value(channel,gain,&analog_data_value)   adquire o resultado de
     uma conversão
lp_measure_volts(channel,&analog_data_value)    adquire o resultado de
     uma conversão e retorna o valor em Volts
lp_setup_adc(timing_source,start_chan,end_chan,gain) prepara o ADC para
     uma série de conversões continuas
lp_adc_series(number_of_values,&analog_data_array)    faz
     number_of_values conversões e coloca o resultado  em
     analog_data_array
lp_wait_adc(&analog_data_array)     faz com que o programa espere que a
     série de conversões termine
```

Há muito mais rotinas na livraria. Consulta o manual [3], ***mas não leves o manual para fora do laboratório!***

Mostra-se de seguida um programa exemplo:

```
#include <stdio.h>
#include <stdlib.h>
#include "lpclerrs.h" /* include LPCLAB error codes                */
#include "lpcldefs.h" /* include LPCLAB function declarations      */

#define NUMBER_OF_DATA_POINTS 3

main ()
{
   FILE *file_handle;              /* Pointer to file structure */
   unsigned short analog_data_value [NUMBER_OF_DATA_POINTS] ;
   unsigned short iterations;
   unsigned short channel = 0 , gain = 1 ;

   lp_initialize();                 /* initialize the LPCLAB
subroutines */
   lp_select_board( 1 );            /* address board 1, the first unit
*/
   lp_set_timeout( 50 );            /* establish the board timeout
period */



      /* Open an empty data file for writing.
      If the given file exists, its contents are destroyed. */

   if ((file_handle = fopen("DATA.DAT","w")) == NULL)
```

```
   {
      printf ("\r\n ERROR opening file DATA.DAT.");
      exit( 0);    /* terminate program */
   }

   iterations = 0 ;

   do
   {
        lp_adc_value(channel,gain,&analog_data_value[iterations]);


        printf("\nValor lido:%d",analog_data_value[iterations]);

        fprintf(file_handle, "%d\r\n", analog_data_value[iterations]);

        iterations++;

   } while (iterations < NUMBER_OF_DATA_POINTS);



   //close file
   if (fclose(file_handle) != 0)
      printf("\r\n\n Error closing DATA.DAT file.");


   lp_terminate();

}
```

O programa é tão simples que dispensa comentários…

Talvez mais importante é dar algumas dicas sobre como compilar correctamente o programa.

3.3 Compilação do programa

Primeiro cria uma pasta `c:\users\a999999` (onde a999999 é o teu numero) onde vais trabalhar. *Este pormenor é importante para não se instaurar a anarquia total no disco duro*…

Copia para essa pasta a livraria `LPCLTCS.LIB` (que se encontra em `c:\LPCLAB`) pois vais precisar dela…

Abre um novo projecto (Project -> Open…) e dá-lhe um nome apropriado. Inclui no novo projecto o teu programa em C e a livraria, como se mostra na Figura 3.

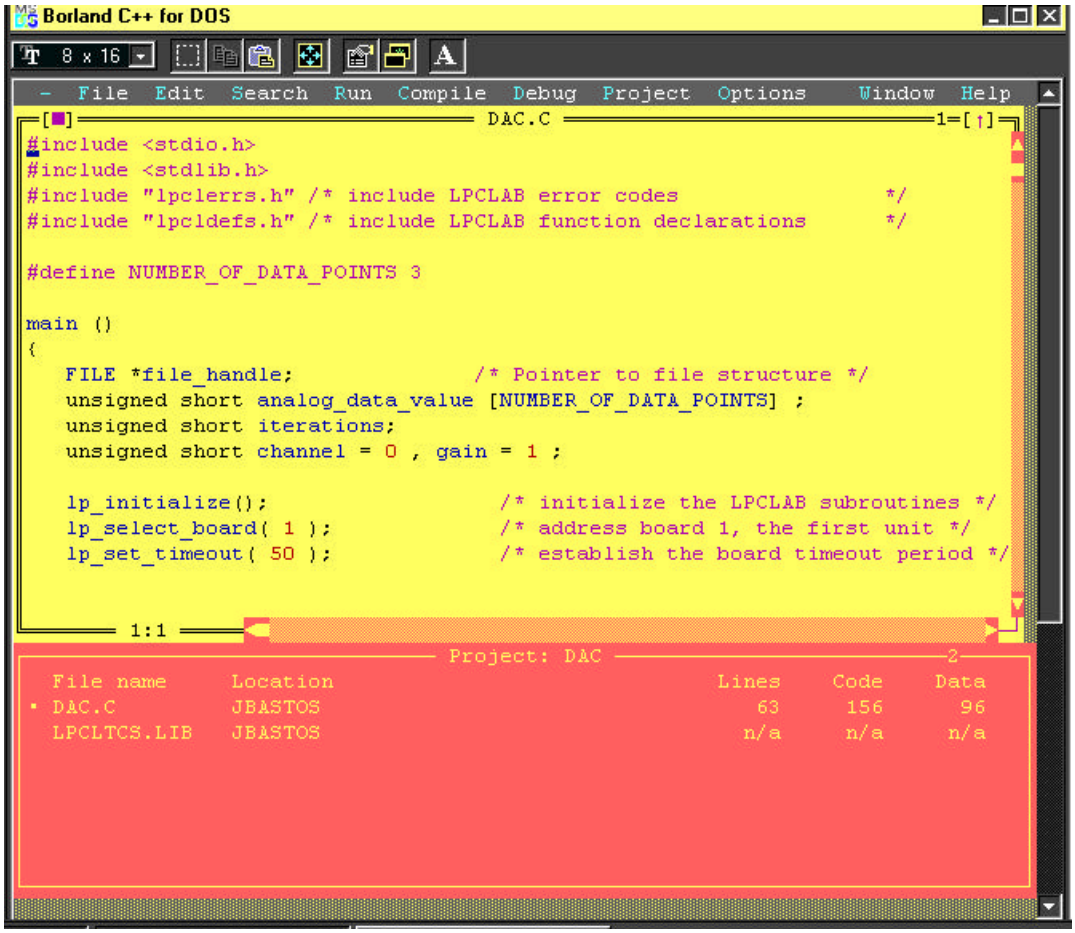


Figura 3: O ambiente de trabalho em Borland C (versão DOS)

Agora vai a Options->Directories e actualiza o conteúdo com o nome das pastas pertinentes, como se mostra na Figura 4.

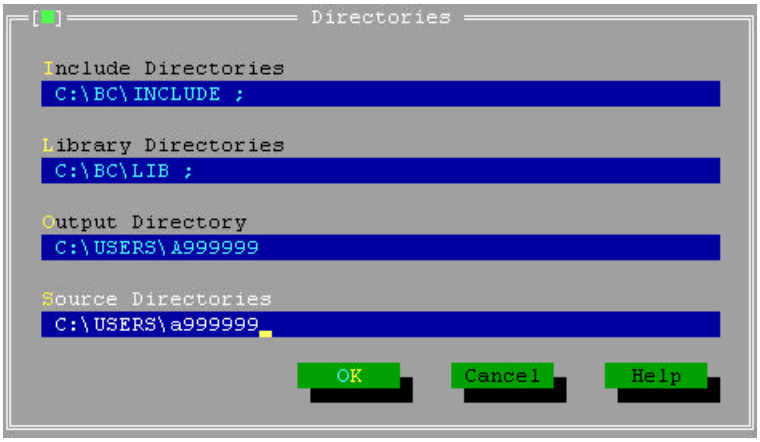


Figura 4: Actualização das pastas

De seguida actualiza as opções de compilação, como se mostra na Figura 5.

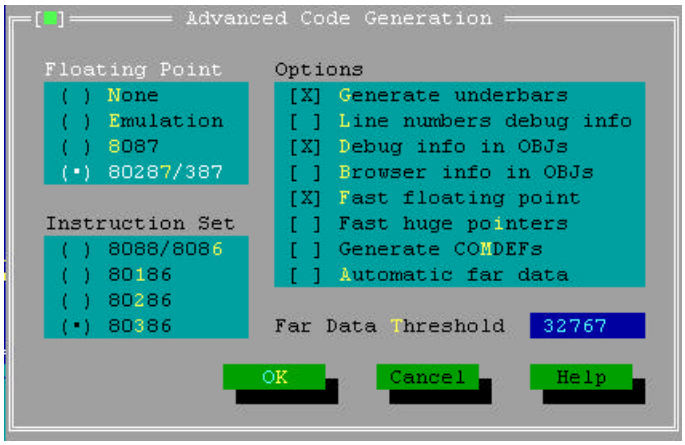


Figura 5: Opções de compilação

Não te esqueces de seleccionar a geração de código SMALL, como se mostra na Figura 6.

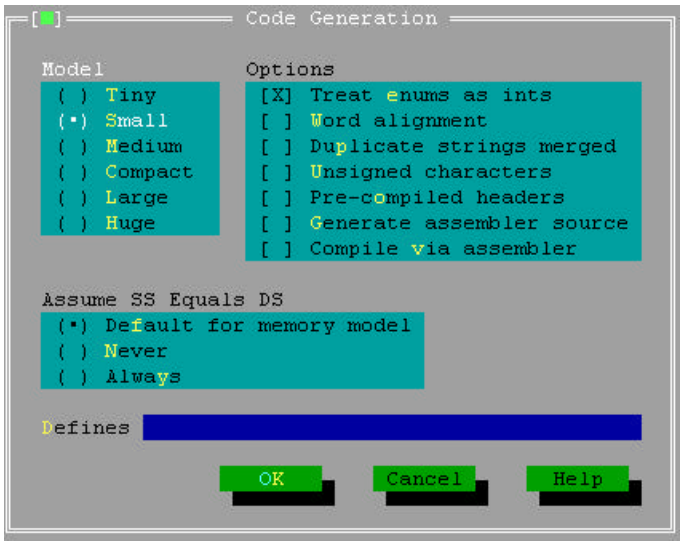


Figura 6: Escolha de código SMALL

Por fim guarda (Options-> Save) o teu ambiente de trabalho. Se não te esqueceste de nenhum detalhe, o teu programa deve agora compilar sem problemas…

Corre o teu programa numa janela de DOS (File -> DOS Shell).

4. Aquisição de dados

O teu programa em C deve ter um ciclo de espera, em que apenas se sai se for detectado uma variação brusca na grandeza medida pelo sensor.

Isto acontece, por exemplo, quando o sensor é movido rápidamente dum ponto "quente" (água a ferver[1]) para um ponto "frio" (ar ambiente).

Mal é detectada a transição, o programa de aquisição de dados deve começar a adquirir rápidamente dados a uma cadência conhecida (tem que se saber o intervalo de tempo entre cada ponto!).

É razoável adquirir-se 25 a 100 pontos entre o valor inicial e o valor final.

Guarda os dados num ficheiro (exemplo, dados.dat) em que a primeira coluna é o tempo, e a segunda coluna é o valor da conversão.

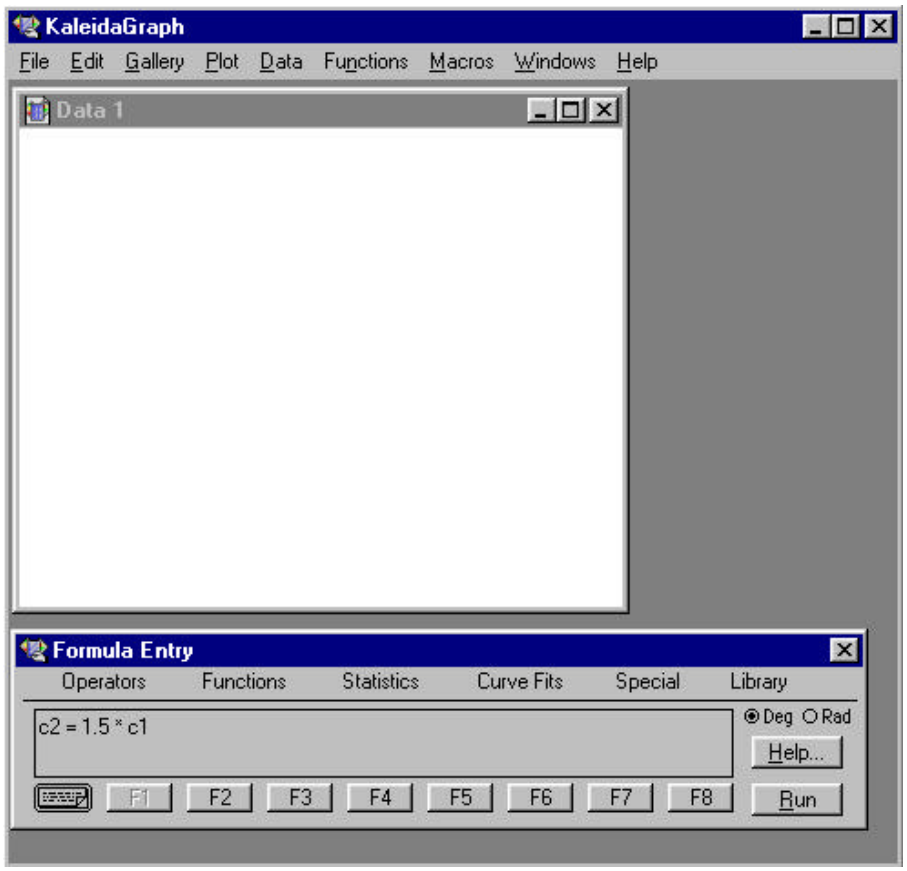
5. Processamento dos dados

Uma vez guardados os dados, o seu processamento pode ser feito em Matlab, Excel, etc. Apenas deves ter tido o cuidado de ter escolhido o caracter separador de colunas conveniente para o programa preferido (tab, virgula, espaço, etc.)

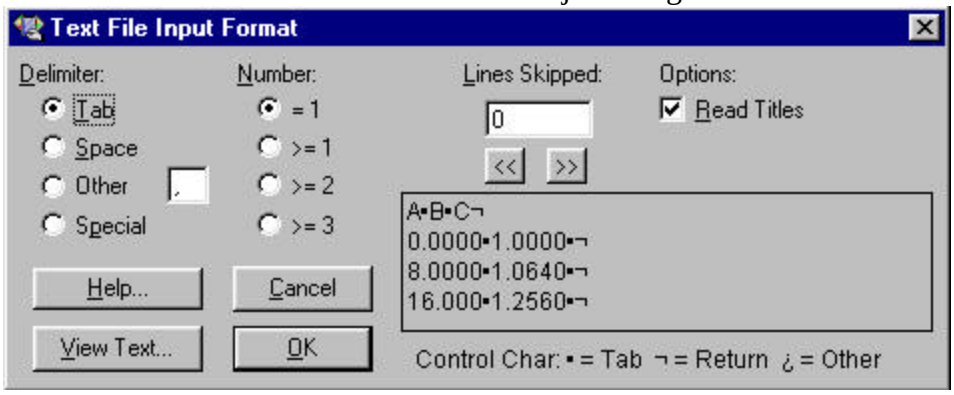
Essencialmente o que se pretende é, a partir de uma tabela com os dados, construir um gráfico e dele extrair (*pelo método da tangente na origem*) a constante de tempo do sensor em estudo.

A titulo de exemplo vai-se utilizar um programa muito simples: **kaleidagraph.** É semelhante ao Excel mas mais simples e mais *user friendly*.

[1] Notar que provavelmente o sensor tem que estar em "banho maria", isto é não pode estar em contacto directo com a água, mas sim imerso num recipiente com liquido não condutor eléctrico (mas bom condutor térmico) que por sua vez está imerso em água…



Seleccione **File > Import > Text** para escolher o ficheiro onde guardou os resultados e escolha o caracter limitador das colunas de dados na janela seguinte:



Faça um gráfico **Gallery > Linear > Line** escolhendo para eixo dos x os resultados do sensor em estudos e para eixo dos y os resultados do sensor de referencia.

Mostra-se de seguida na **Figura 7** um gráfico a título de exemplo…

O teu gráfico será com certeza diferente porque este foi feito com dados artificiais (e um pouco de ruído gaussiano à mistura para dar mais realismo…)

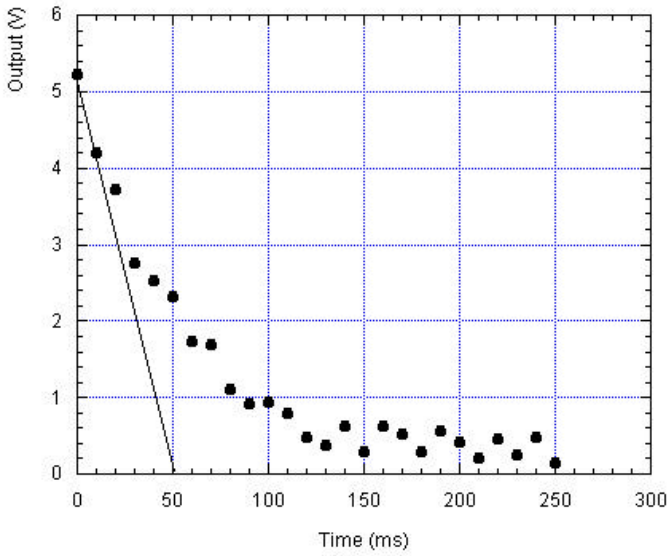


Figura 7: Resposta no tempo de um sensor (dados artificiais)

A constante de tempo é dada pela intercepção da tangente à curva na origem com a linha horizontal que passa pelo valor final.

De preferência a constante de tempo é determinada por um método numérico. Encontra a melhor recta (método dos mínimos quadrados) que passa pelos 5 ~ 10 primeiros pontos.

Determina o par (x, y) desta recta para o qual y é o valor final. O valor de x correspondente é a constante de tempo desejada.

6. Bibliografia

[1] DT2814 User Manual
[2] DT2814 Device Driver
[3] LPCLAB User Manual