MPA-3

Multiparameter Data Acquisition System

(Sistema de Aquisição de Dados Multi-parâmetros)

Manual

2009_NC

CALIBRAÇÃO DOS CANAIS DOS ADC'S

(Usando geradores de impulsos)

1. Seleccionar a zona com o botão direito do rato.
2. Clicar no botão "create a new Roi".
3. (opcional) duplo clique abaixo do histograma, sobra a marcação da região a analizar, para fazer zoom a zona.
4. Clicar no botão "Gaussian Fit"
5. (opcional) Guardar os valores de *Pos* (posição) e *FWHM* (Resolução de energia).
6. Clicar no botão "Calibratio".
   Nesta Janela podemos colocar manualmente os valores para os quais queremos fazer a calibração, ou então usar os valores obtidos pelo ajuste gausiano, "Gaussian Fit". È esta última que passamos a descrever.
7. Carregar em "Fit", para colocar o valor "Pos" do "Gaussian Fit" na primeira Caixa.
8. Na segunda caixa colocamos o valor pretendido.
9. Clicar em "Add>>".
10. Carregar em "Ok" e repetir os passos 4 a 10 para todos os pontos de interesse.
11. Clicar em "Calibrate" para o calculo do ajuste.
    O programa calcula os valores de *p0* e *p1*, assim como o erre do ajuste.
12. No topo da janela "Calibrate" tem a opção "Use calibratio", se estiver activa, no espectro o cursor é convertido para a unidade calibrada, assim como as unidades, "Unit", seleccionada.
13. Finalmente podemos gravar a calibração, para futuras utilizações. Para isso clicar no botão "Save as…", e o ficheiro terá a extensão "*.CTL" de um "Control File" do MPA-Software.

O FICHEIRO DMPA3.DLL

Este ficheiro permite-nos adicionar novas funções e a manipulação dos dados recebidos pelos ADC's antes destes serem armazenados, com algumas restrições, sendo a principal delas a velocidade de processamento.

- Para corrigir o facto de o programa receber canais, o que significa inteiros, e nos precisarmos de fazer divisão entre eles, vai acontecer que alguns números não vão aparecer, principalmente na divisão pelos canais mais baixos. Assim decidi-mos fazer divisão sobre reais, e acrescentar as canais um numero aleatório (0 a 1), para eliminar o vazios da divisão.

  Assim começamos por criar, no ficheiro "custom.c" que serve de código fonte para o DMPA3.DLL, uma função geradora de números aleatórios.

```
float ranqd2(void)
{
  float rand;
  static unsigned long idum = 0;
  unsigned long itemp;
  static unsigned long jflone = 0x3f800000;
  static unsigned long jflmsk = 0x007fffff;

  idum = 1664525L*idum + 1013904223L;
  itemp = jflone | (jflmsk & idum);
  rand = (*(float *)&itemp)-1.0;
  return rand;
}
```

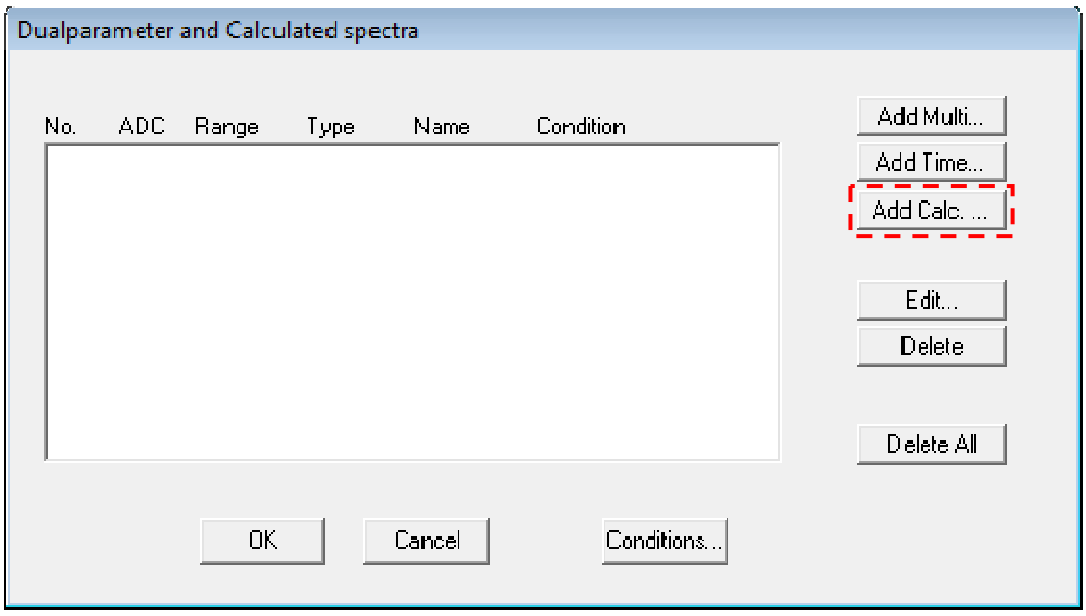
Esta função ao ser chamada por: "`ranqd2()`" gera um valor real arbitrário entre 0.0 e 1.0

Para testar a função geradora de números aleatórios, criou-se uma função para os representar no programa MPA, a função é a seguinte

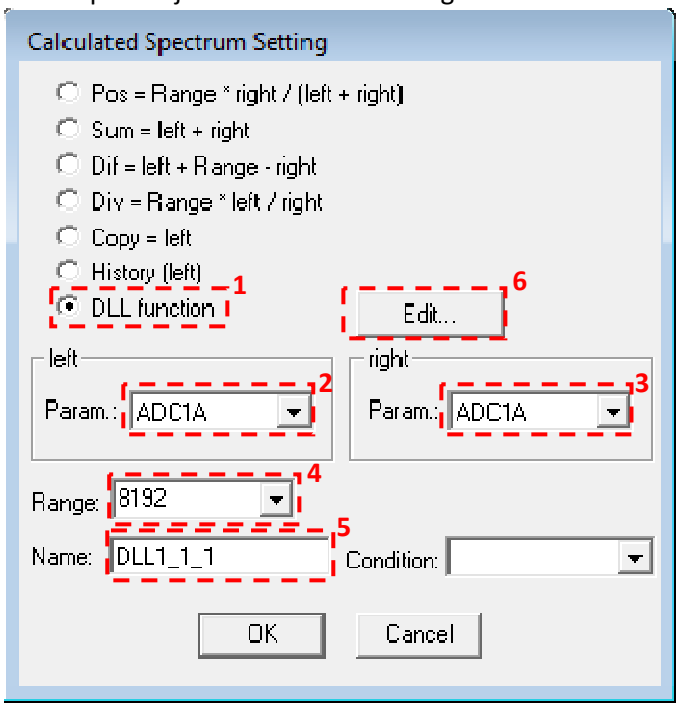
```
unsigned long APIENTRY RanPos(long x, long y, long z)
{
  float rand;
  rand = ranqd2();
  return (unsigned long) (rand*8192);
}
```

Para chamar o função no software, ir ao menu "Options", e selecionar a opção "Spectra…", ou clicar no botão: 

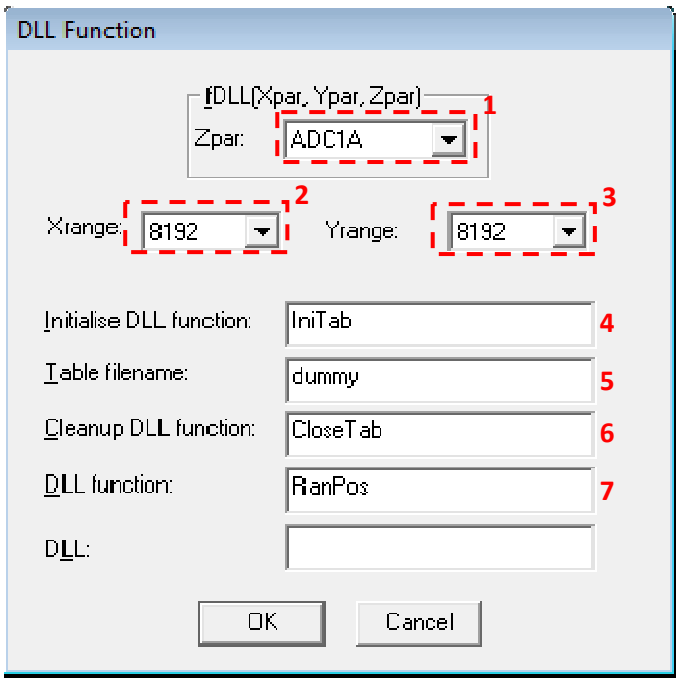
Na janela seguinte clicar no botão "Add Calc. …"



de seguida preencher os campos da janela conforme a imagem

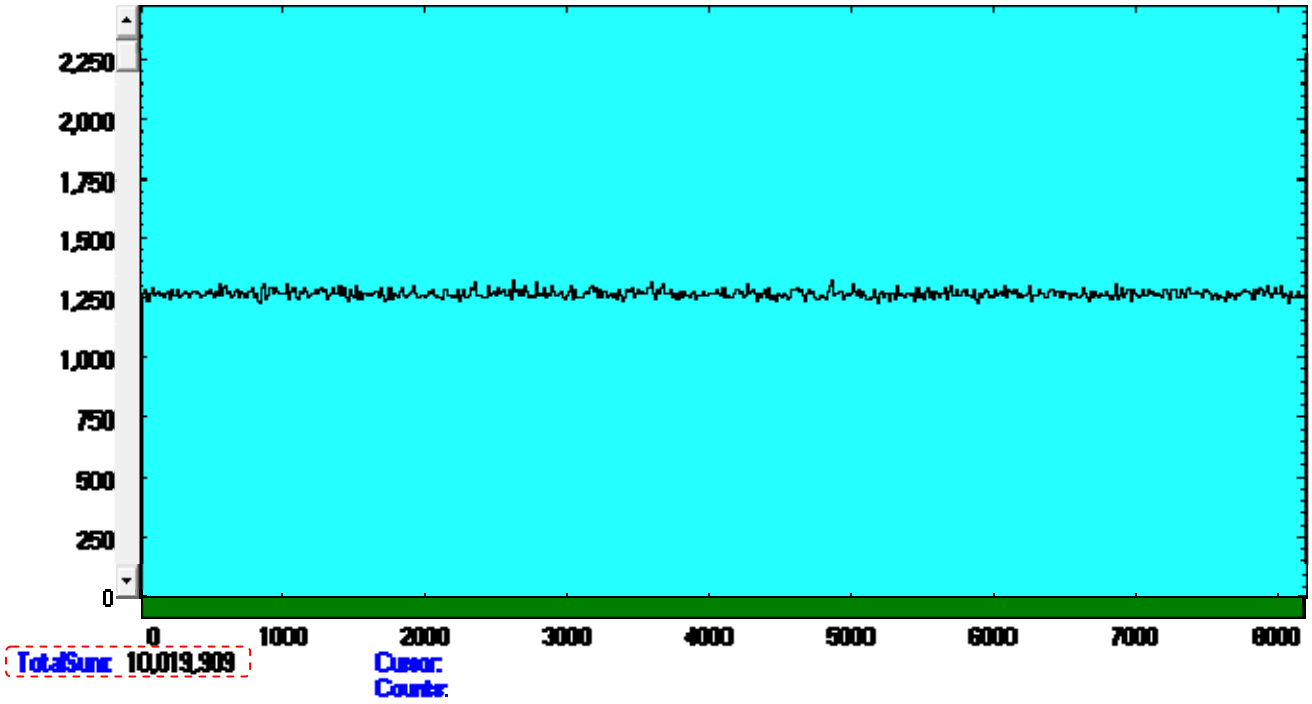


Seguida clicar em “EDIT” e preencher como a figura:



Para este caso, não necessitava-mos de todos os parâmetros, mas a função DLL, não nos permite retirar, por isso, nesta ultima a opções **1**, **2** e **3** são obrigatórias, mas não são usados, por isso podemos por o que nos apetecer. Nas opções **4** e **5** também, são obrigatórias, mas convém que sejam funções que existam no programa. Na opção **6** esta é a única opção válida e finalmente na opção **7** a nossa função **RanPos**.

O resultado desta função, aparece na forma de gráfico no nosso programa e por cada impulso que chega ao canal 1 (ADC_A), este gera um número aleatório e multiplica pelo número de canais, neste caso 8192. Depois de 10 000 000 número o resultado é o seguinte:



Que mostra que a função funciona cumpre os requisitos pretendidos.

Seguidamente criamos uma função para o calculo da posição, tal qual o que é necessário para os sensores de posição, esta função ira incluir somar também o numero aleatório a a cada um dos valores de x e y, o resultado é o seguinte.

```
unsigned long APIENTRY IncPos(long x, long y, long z)
{
  float x_real, y_real, rand;
  unsigned long idum;
  rand = ranqd2();
  x_real = (float) x + rand;
  y_real = (float) y + rand;
  return (unsigned long)((y_real*8192)/(x_real+y_real));
}
```

Neste ponto pensou-se optimizar. Pensou-se em criar um ficheiro de texto que seria lido pelo programa, ficheiro esse que iria introduzir o *range* necessário, poderíamos inserir correcção aos valores obtidos pelos adc's, tais como factores de escala, ofsets, e a quando dos testes com o equipamento, serviria também para inserir outros factores de correcção, tais como os vários pré-amplificadores e amplificadores não terem ganhos semelhantes. Assim sendo começou-se por criar uma função de leitura.

```
int APIENTRY IniCorr(LPSTR filename)
{
    FILE *f;
    char txt[256];

    if (!filename) return -1;
    if (!strnicmp("dummy", filename, 5)) return 0;
    if (!(f = fopen(filename, "rb"))) {
      sprintf(txt, "Filename %s not found!", filename);
      MessageBox(NULL, txt, "DMPA3.DLL", MB_OK);
      return -1;
    }
    while(!freadstr(f,txt,256)) {
      if(!strnicmp("range=", txt, 6)) {
        sscanf(txt+6,"%ld",&range);
      }
      else if(!strnicmp("acaloff=", txt, 8)) {
        sscanf(txt+8,"%f",&acaloff);
      }
        else if(!strnicmp("acalfac=", txt, 8)) {
        sscanf(txt+8,"%f",&acalfac);
      }
        else if(!strnicmp("bcaloff=", txt, 8)) {
        sscanf(txt+8,"%f",&bcaloff);
      }
        else if(!strnicmp("bcalfac=", txt, 8)) {
        sscanf(txt+8,"%f",&bcalfac);
      }
        else if(!strnicmp("ccaloff=", txt, 8)) {
        sscanf(txt+8,"%f",&ccaloff);
      }
        else if(!strnicmp("ccalfac=", txt, 8)) {
        sscanf(txt+8,"%f",&ccalfac);
      }
        else if(!strnicmp("dcaloff=", txt, 8)) {
        sscanf(txt+8,"%f",&dcaloff);
      }
        else if(!strnicmp("dcalfac=", txt, 8)) {
```

```
        sscanf(txt+8,"%f",&dcalfac);
      }
      }
    fclose(f);
      return 0;
}
```

A função ira ler um ficheiro de texto i retirar os valores de range e os parâmetros de correcção dos quatro adc's, o ficheiro de texto criado foi o seguinte.

```
range=8192
acaloff=0.0
acalfac=1.0
bcaloff=0.0
bcalfac=1.0
ccaloff=0.0
ccalfac=1.0
dcaloff=0.0
dcalfac=1.0
```

corr.txt

seguidamente criou-se uma função de posição que utilizes estes dados

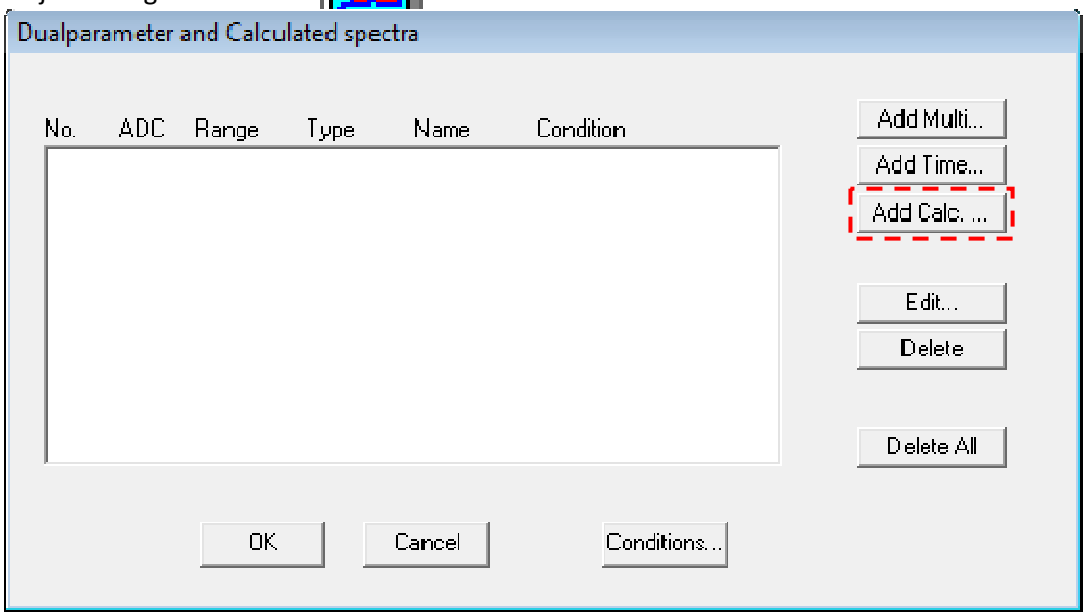
```
unsigned long APIENTRY A_PosCal(long x, long y, long z)
{
  double x_real, y_real, rand, rang_real, x_corr, y_corr;
  if (z >= 10) {
      rand = ranqd2();
      x_real = (double) (x) + rand;
      x_corr = acaloff + (x_real * acalfac);
      rand = ranqd2();
      y_real = (double) (y) + rand;
      y_corr = bcaloff + (y_real * bcalfac);
      rang_real = (double) (range);
      return (unsigned long)((y_corr * rang_real) / (x_corr +
y_corr));
  }
  else return -1;
}
```

Neste caso ira corrigir os adc's A e B, para os adc's C e D, criou-se uma função semelhante

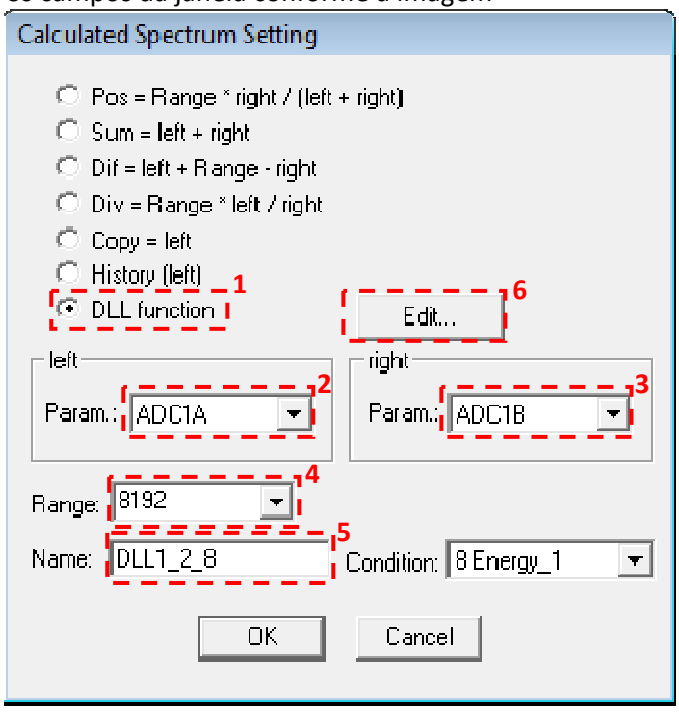
```
unsigned long APIENTRY B_PosCal(long x, long y, long z)
{
  double x_real, y_real, rand, rang_real, x_corr, y_corr;
  if (z >= 10) {
      rand = ranqd2();
      x_real = (double) (x) + rand;
      x_corr = ccaloff + (x_real * ccalfac);
      rand = ranqd2();
      y_real = (double) (y) + rand;
      y_corr = dcaloff + (y_real * dcalfac);
      rang_real = (double) (range);
      return (unsigned long)((y_corr * rang_real) / (x_corr +
y_corr));
  }
  else return -1;
}
```

Para chamar o função no software, ir ao menu “Options”, e selecionar a opção “Spectra…”, ou clicar no botão: 

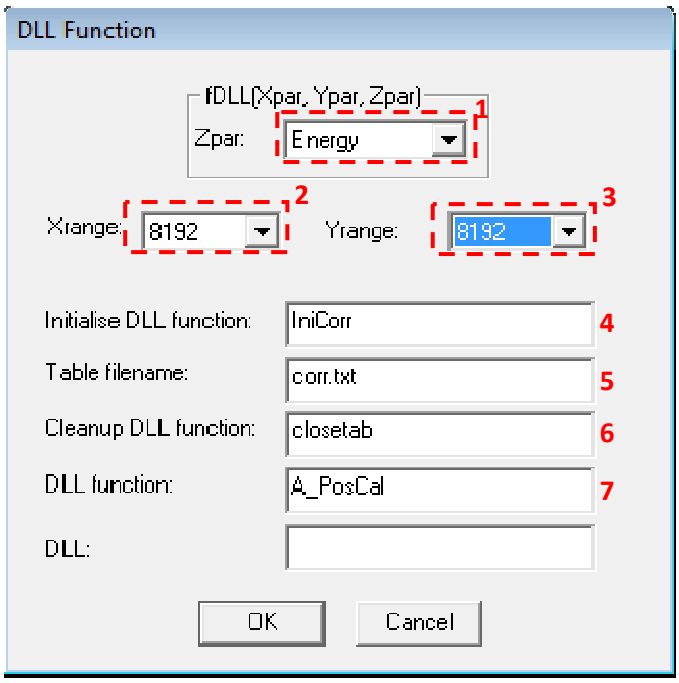
Na janela seguinte clicar no botão “Add Calc. …”



de seguida preencher os campos da janela conforme a imagem



Seguida clicar em “EDIT” e preencher como a figura:

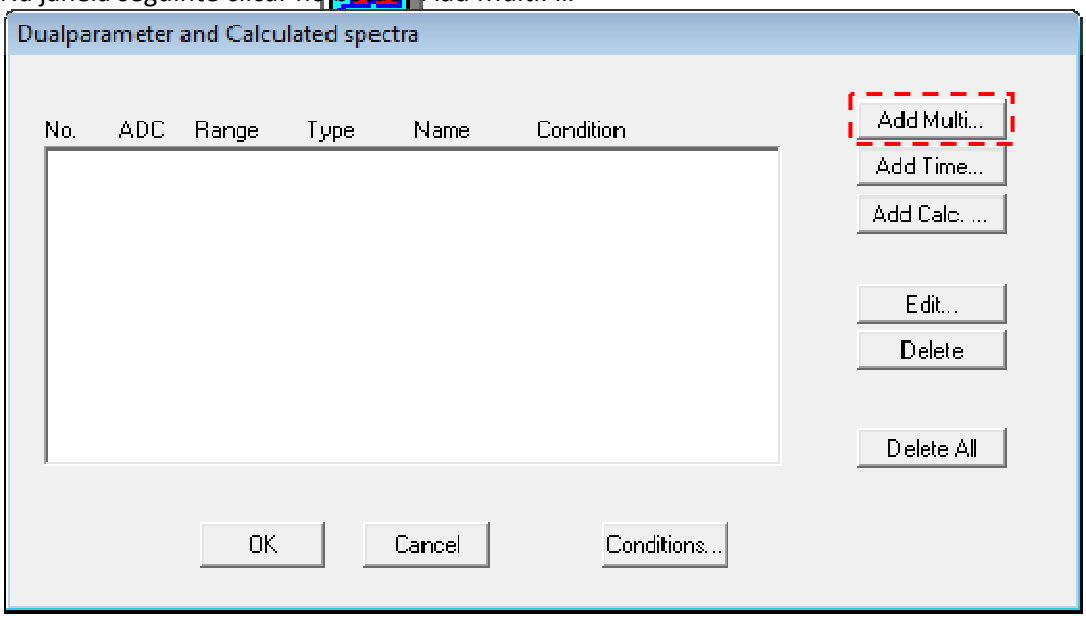


Para estas funções utilizamos todas as opções das funções DLL, incluindo o parâmetro Z onde inserimos um cálculo de energia, utilizando para isso a função de calculo de energia utilizado pelo programas, que não é mais que a soma de dois adc's, e usamos este parâmetro, para poder-mos seleccionar uma determinada banda de energia, por exemplo uma banda de um elemento num espectro de rbs.

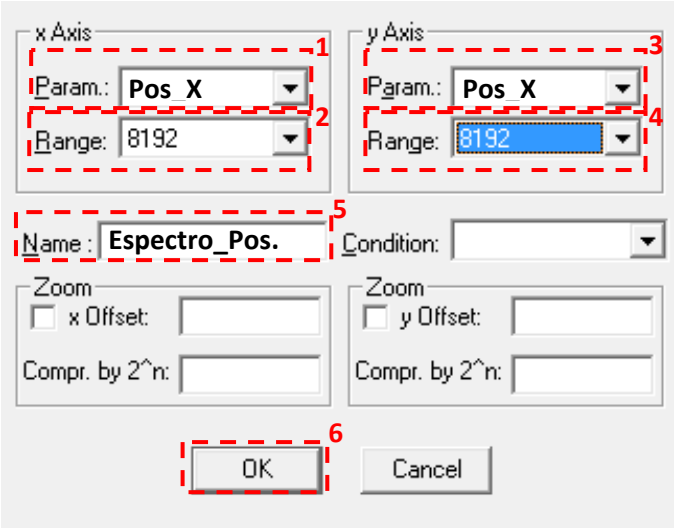
Fazemos duas funções destas para as posições X e Y do sensor, mudando o parâmetros de entrada "Xpar" e "Ypar" e a função DLL de A_PosCal para B_PosCal.

Seguidamente podemos representar as duas variáveis num gráfico com gradiente de cor, bastando para isso chamar o função no software, ir ao menu "Options", e selecionar a opção "Spectra…", ou clicar no botão: 

Na janela seguinte clicar no botão "Add Multi. …"



de seguida preencher os campos da janela conforme a imagem



Os parâmetros 1 e 3 são escolhidos consoante os nomes dados as funções de posição-

E assim obtemos os gráficos pretendidos para este projecto.

até novas correcções……