```
E:\Dropbox\Grau en Eng. Electrònica i Automàtica\4 - Quart\TREBALL FINAL DE GRAU\Memòri
#include <SoundLocator.h>
#include <inputRaw.h>
#include <outputRaw.h>
#include <math.h>

/*********************************************************************
*                                  CONFIGURACIONS
*********************************************************************
//CONFIGURACIÓ DEL MODUL UART I DELS SEUS REGISTRES
#use rs232(UART1A,baud=19200,parity=N,bits=8)
#word U1TXREG = 0x210      //UART1 Transmit Register
#word U1RXREG = 0x212      //UART1 Receive Register
#word U1STA = getenv("SFR:U1STA")         //UART1 Status and Control Regist

//DEFINICIÓ DEL REGISTRES DEL TIMER 2
#word T2CON = 0x0110
#bit TCS    = T2CON.1
#bit T32    = T2CON.3
#bit TCKPS0 = T2CON.4
#bit TCKPS1 = T2CON.5
#bit TON    = T2CON.15
#word TMR2  = 0x0106

//DECLARACIÓ DELS REGISTRES DEL MODUL ADC
#word ADCON1  = getenv("SFR:ADCON1")    //A/D Control Register 1
#bit  ADON    = ADCON1.15
#word ADCON2  = getenv("SFR:ADCON2")    //A/D Control Register 2
#word ADCON3  = getenv("SFR:ADCON3")    //A/D Control Register 3
#word ADCHS   = getenv("SFR:ADCHS")     //A/D Input Channel Select Register
#word ADPCFG  = getenv("SFR:ADPCFG")    //A/D Port Configuration Register
#word ADCSSL  = getenv("SFR:ADCSSL")    //A/D Input Scan Select Register
#word ADCBUF0 = getenv("SFR:ADCBUF0")   //ADC Data Buffer 0
#word ADCBUF1 = getenv("SFR:ADCBUF1")   //ADC Data Buffer 1
#word ADCBUF2 = getenv("SFR:ADCBUF2")   //ADC Data Buffer 2
#word ADCBUF3 = getenv("SFR:ADCBUF3")   //ADC Data Buffer 3

//CONFIGURACIÓ DEL MODUL I2C
#use i2c(Master,Fast,sda=PIN_F2,scl=PIN_F3)

//DEFINICIÓ DE FUNCIONS
void configuracio();
float32 filterH1(float32 in);
float32 filterH2(float32 in);
unsigned int16 ad5282_read_val(unsigned int8 adr);
void ad5282_write_val(unsigned int8 adr, int1 pot, unsigned int8 data);
float32 tempsDelay(unsigned int16 temp);
float32 angleEmissor(float32 temp, float32 soundVel, float32 dist);

//DEFINCIÓ DE CONSTANTS DEL PROGRAMA
#define POT_STEP 784.31         //ve donat per 200000/255 resistencia/nºpas
//const float32 VREF = 0.065;
float32 VREF = 0.5;
const float32 SOUNDVEL = 1435.00; //1550.00; quan es vagi al mar canviar-ho
const float32 DISTH = 0.047;
const float32 Tcy = 0.000000125;
const unsigned int8 AD5282_1_ADDR = 0b00101101; //POT H3-H4
const unsigned int8 AD5282_2_ADDR = 0b00101110; //POT H1-H2
const float32    ADC_LUT[1024] =
    {-2.50000,-2.49511,-2.49022,-2.48534,-2.48045,-2.47556,-2.47067,-2.4657
     -2.46090,-2.45601,-2.45112,-2.44624,-2.44135,-2.43646,-2.43157,-2.4266
     -2.42180,-2.41691,-2.41202,-2.40714,-2.40225,-2.39736,-2.39247,-2.3875
     -2.38270,-2.37781,-2.37292,-2.36804,-2.36315,-2.35826,-2.35337,-2.3484
     -2.34360,-2.33871,-2.33382,-2.32893,-2.32405,-2.31916,-2.31427,-2.3093
     -2.30450,-2.29961,-2.29472,-2.28983,-2.28495,-2.28006,-2.27517,-2.2702
     -2.26540,-2.26051,-2.25562,-2.25073,-2.24585,-2.24096,-2.23607,-2.2311
```

E:\Dropbox\Grau en Eng. Electrònica i Automàtica\4 - Quart\TREBALL FINAL DE GRAU\Memòri
-2.22630,-2.22141,-2.21652,-2.21163,-2.20674,-2.20186,-2.19697,-2.1920
-2.18719,-2.18231,-2.17742,-2.17253,-2.16764,-2.16276,-2.15787,-2.1529
-2.14809,-2.14321,-2.13832,-2.13343,-2.12854,-2.12366,-2.11877,-2.1138
-2.10899,-2.10411,-2.09922,-2.09433,-2.08944,-2.08456,-2.07967,-2.0747
-2.06989,-2.06500,-2.06012,-2.05523,-2.05034,-2.04545,-2.04057,-2.0356
-2.03079,-2.02590,-2.02102,-2.01613,-2.01124,-2.00635,-2.00147,-1.9965
-1.99169,-1.98680,-1.98192,-1.97703,-1.97214,-1.96725,-1.96237,-1.9574
-1.95259,-1.94770,-1.94282,-1.93793,-1.93304,-1.92815,-1.92326,-1.9183
-1.91349,-1.90860,-1.90371,-1.89883,-1.89394,-1.88905,-1.88416,-1.8792
-1.87439,-1.86950,-1.86461,-1.85973,-1.85484,-1.84995,-1.84506,-1.8401
-1.83529,-1.83040,-1.82551,-1.82063,-1.81574,-1.81085,-1.80596,-1.8010
-1.79619,-1.79130,-1.78641,-1.78152,-1.77664,-1.77175,-1.76686,-1.7619
-1.75709,-1.75220,-1.74731,-1.74242,-1.73754,-1.73265,-1.72776,-1.7228
-1.71799,-1.71310,-1.70821,-1.70332,-1.69844,-1.69355,-1.68866,-1.6837
-1.67889,-1.67400,-1.66911,-1.66422,-1.65934,-1.65445,-1.64956,-1.6446
-1.63978,-1.63490,-1.63001,-1.62512,-1.62023,-1.61535,-1.61046,-1.6055
-1.60068,-1.59580,-1.59091,-1.58602,-1.58113,-1.57625,-1.57136,-1.5664
-1.56158,-1.55670,-1.55181,-1.54692,-1.54203,-1.53715,-1.53226,-1.5273
-1.52248,-1.51760,-1.51271,-1.50782,-1.50293,-1.49804,-1.49316,-1.4882
-1.48338,-1.47849,-1.47361,-1.46872,-1.46383,-1.45894,-1.45406,-1.4491
-1.44428,-1.43939,-1.43451,-1.42962,-1.42473,-1.41984,-1.41496,-1.4100
-1.40518,-1.40029,-1.39541,-1.39052,-1.38563,-1.38074,-1.37586,-1.3709
-1.36608,-1.36119,-1.35630,-1.35142,-1.34653,-1.34164,-1.33675,-1.3318
-1.32698,-1.32209,-1.31720,-1.31232,-1.30743,-1.30254,-1.29765,-1.2927
-1.28788,-1.28299,-1.27810,-1.27322,-1.26833,-1.26344,-1.25855,-1.2536
-1.24878,-1.24389,-1.23900,-1.23412,-1.22923,-1.22434,-1.21945,-1.2145
-1.20968,-1.20479,-1.19990,-1.19501,-1.19013,-1.18524,-1.18035,-1.1754
-1.17058,-1.16569,-1.16080,-1.15591,-1.15103,-1.14614,-1.14125,-1.1363
-1.13148,-1.12659,-1.12170,-1.11681,-1.11193,-1.10704,-1.10215,-1.0972
-1.09238,-1.08749,-1.08260,-1.07771,-1.07283,-1.06794,-1.06305,-1.0581
-1.05327,-1.04839,-1.04350,-1.03861,-1.03372,-1.02884,-1.02395,-1.0190
-1.01417,-1.00929,-1.00440,-0.99951,-0.99462,-0.98974,-0.98485,-0.9799
-0.97507,-0.97019,-0.96530,-0.96041,-0.95552,-0.95064,-0.94575,-0.9408
-0.93597,-0.93109,-0.92620,-0.92131,-0.91642,-0.91153,-0.90665,-0.9017
-0.89687,-0.89198,-0.88710,-0.88221,-0.87732,-0.87243,-0.86755,-0.8626
-0.85777,-0.85288,-0.84800,-0.84311,-0.83822,-0.83333,-0.82845,-0.8235
-0.81867,-0.81378,-0.80890,-0.80401,-0.79912,-0.79423,-0.78935,-0.7844
-0.77957,-0.77468,-0.76979,-0.76491,-0.76002,-0.75513,-0.75024,-0.7453
-0.74047,-0.73558,-0.73069,-0.72581,-0.72092,-0.71603,-0.71114,-0.7062
-0.70137,-0.69648,-0.69159,-0.68671,-0.68182,-0.67693,-0.67204,-0.6671
-0.66227,-0.65738,-0.65249,-0.64761,-0.64272,-0.63783,-0.63294,-0.6280
-0.62317,-0.61828,-0.61339,-0.60850,-0.60362,-0.59873,-0.59384,-0.5889
-0.58407,-0.57918,-0.57429,-0.56940,-0.56452,-0.55963,-0.55474,-0.5498
-0.54497,-0.54008,-0.53519,-0.53030,-0.52542,-0.52053,-0.51564,-0.5107
-0.50587,-0.50098,-0.49609,-0.49120,-0.48631,-0.48143,-0.47654,-0.4716
-0.46676,-0.46188,-0.45699,-0.45210,-0.44721,-0.44233,-0.43744,-0.4325
-0.42766,-0.42278,-0.41789,-0.41300,-0.40811,-0.40323,-0.39834,-0.3934
-0.38856,-0.38368,-0.37879,-0.37390,-0.36901,-0.36413,-0.35924,-0.3543
-0.34946,-0.34457,-0.33969,-0.33480,-0.32991,-0.32502,-0.32014,-0.3152
-0.31036,-0.30547,-0.30059,-0.29570,-0.29081,-0.28592,-0.28104,-0.2761
-0.27126,-0.26637,-0.26149,-0.25660,-0.25171,-0.24682,-0.24194,-0.2370
-0.23216,-0.22727,-0.22239,-0.21750,-0.21261,-0.20772,-0.20283,-0.1979
-0.19306,-0.18817,-0.18328,-0.17840,-0.17351,-0.16862,-0.16373,-0.1588
-0.15396,-0.14907,-0.14418,-0.13930,-0.13441,-0.12952,-0.12463,-0.1197
-0.11486,-0.10997,-0.10508,-0.10020,-0.09531,-0.09042,-0.08553,-0.0806
-0.07576,-0.07087,-0.06598,-0.06109,-0.05621,-0.05132,-0.04643,-0.0415
-0.03666,-0.03177,-0.02688,-0.02199,-0.01711,-0.01222,-0.00733,-0.0024
0.00244,0.00733,0.01222,0.01711,0.02199,0.02688,0.03177,0.03666,0.0415
0.04643,0.05132,0.05621,0.06109,0.06598,0.07087,0.07576,0.08065,0.0855
0.09042,0.09531,0.10020,0.10508,0.10997,0.11486,0.11975,0.12463,0.1295
0.13441,0.13930,0.14418,0.14907,0.15396,0.15885,0.16373,0.16862,0.1735
0.17840,0.18328,0.18817,0.19306,0.19795,0.20283,0.20772,0.21261,0.2175
0.22239,0.22727,0.23216,0.23705,0.24194,0.24682,0.25171,0.25660,0.2614
0.26637,0.27126,0.27615,0.28104,0.28592,0.29081,0.29570,0.30059,0.3054
0.31036,0.31525,0.32014,0.32502,0.32991,0.33480,0.33969,0.34457,0.3494

```
E:\Dropbox\Grau en Eng. Electrònica i Automàtica\4 - Quart\TREBALL FINAL DE GRAU\Memòri
     0.35435,0.35924,0.36413,0.36901,0.37390,0.37879,0.38368,0.38856,0.3934
     0.39834,0.40323,0.40811,0.41300,0.41789,0.42278,0.42766,0.43255,0.4374
     0.44233,0.44721,0.45210,0.45699,0.46188,0.46676,0.47165,0.47654,0.4814
     0.48631,0.49120,0.49609,0.50098,0.50587,0.51075,0.51564,0.52053,0.5254
     0.53030,0.53519,0.54008,0.54497,0.54985,0.55474,0.55963,0.56452,0.5694
     0.57429,0.57918,0.58407,0.58895,0.59384,0.59873,0.60362,0.60850,0.6133
     0.61828,0.62317,0.62805,0.63294,0.63783,0.64272,0.64761,0.65249,0.6573
     0.66227,0.66716,0.67204,0.67693,0.68182,0.68671,0.69159,0.69648,0.7013
     0.70626,0.71114,0.71603,0.72092,0.72581,0.73069,0.73558,0.74047,0.7453
     0.75024,0.75513,0.76002,0.76491,0.76979,0.77468,0.77957,0.78446,0.7893
     0.79423,0.79912,0.80401,0.80890,0.81378,0.81867,0.82356,0.82845,0.8333
     0.83822,0.84311,0.84800,0.85288,0.85777,0.86266,0.86755,0.87243,0.8773
     0.88221,0.88710,0.89198,0.89687,0.90176,0.90665,0.91153,0.91642,0.9213
     0.92620,0.93109,0.93597,0.94086,0.94575,0.95064,0.95552,0.96041,0.9653
     0.97019,0.97507,0.97996,0.98485,0.98974,0.99462,0.99951,1.00440,1.0092
     1.01417,1.01906,1.02395,1.02884,1.03372,1.03861,1.04350,1.04839,1.0532
     1.05816,1.06305,1.06794,1.07283,1.07771,1.08260,1.08749,1.09238,1.0972
     1.10215,1.10704,1.11193,1.11681,1.12170,1.12659,1.13148,1.13636,1.1412
     1.14614,1.15103,1.15591,1.16080,1.16569,1.17058,1.17546,1.18035,1.1852
     1.19013,1.19501,1.19990,1.20479,1.20968,1.21457,1.21945,1.22434,1.2292
     1.23412,1.23900,1.24389,1.24878,1.25367,1.25855,1.26344,1.26833,1.2732
     1.27810,1.28299,1.28788,1.29277,1.29765,1.30254,1.30743,1.31232,1.3172
     1.32209,1.32698,1.33187,1.33675,1.34164,1.34653,1.35142,1.35630,1.3611
     1.36608,1.37097,1.37586,1.38074,1.38563,1.39052,1.39541,1.40029,1.4051
     1.41007,1.41496,1.41984,1.42473,1.42962,1.43451,1.43939,1.44428,1.4491
     1.45406,1.45894,1.46383,1.46872,1.47361,1.47849,1.48338,1.48827,1.4931
     1.49804,1.50293,1.50782,1.51271,1.51760,1.52248,1.52737,1.53226,1.5371
     1.54203,1.54692,1.55181,1.55670,1.56158,1.56647,1.57136,1.57625,1.5811
     1.58602,1.59091,1.59580,1.60068,1.60557,1.61046,1.61535,1.62023,1.6251
     1.63001,1.63490,1.63978,1.64467,1.64956,1.65445,1.65934,1.66422,1.6691
     1.67400,1.67889,1.68377,1.68866,1.69355,1.69844,1.70332,1.70821,1.7131
     1.71799,1.72287,1.72776,1.73265,1.73754,1.74242,1.74731,1.75220,1.7570
     1.76197,1.76686,1.77175,1.77664,1.78152,1.78641,1.79130,1.79619,1.8010
     1.80596,1.81085,1.81574,1.82063,1.82551,1.83040,1.83529,1.84018,1.8450
     1.84995,1.85484,1.85973,1.86461,1.86950,1.87439,1.87928,1.88416,1.8890
     1.89394,1.89883,1.90371,1.90860,1.91349,1.91838,1.92326,1.92815,1.9330
     1.93793,1.94282,1.94770,1.95259,1.95748,1.96237,1.96725,1.97214,1.9770
     1.98192,1.98680,1.99169,1.99658,2.00147,2.00635,2.01124,2.01613,2.0210
     2.02590,2.03079,2.03568,2.04057,2.04545,2.05034,2.05523,2.06012,2.0650
     2.06989,2.07478,2.07967,2.08456,2.08944,2.09433,2.09922,2.10411,2.1089
     2.11388,2.11877,2.12366,2.12854,2.13343,2.13832,2.14321,2.14809,2.1529
     2.15787,2.16276,2.16764,2.17253,2.17742,2.18231,2.18719,2.19208,2.1969
     2.20186,2.20674,2.21163,2.21652,2.22141,2.22630,2.23118,2.23607,2.2409
     2.24585,2.25073,2.25562,2.26051,2.26540,2.27028,2.27517,2.28006,2.2849
     2.28983,2.29472,2.29961,2.30450,2.30938,2.31427,2.31916,2.32405,2.3289
     2.33382,2.33871,2.34360,2.34848,2.35337,2.35826,2.36315,2.36804,2.3729
     2.37781,2.38270,2.38759,2.39247,2.39736,2.40225,2.40714,2.41202,2.4169
     2.42180,2.42669,2.43157,2.43646,2.44135,2.44624,2.45112,2.45601,2.4609
     2.46579,2.47067,2.47556,2.48045,2.48534,2.49022,2.49511,2.50000};

//DEFINICIÓ DE VARIABLES GLOBALS
//int8 parameter[16];      //Per ajudar a enviar dades pel port RS232
int8 estat = 0;
unsigned int8 VAL_POT = 128;
float32 H1_ant = 0;
float32 H1_Vmax = 0;
float32 H1_Vmin = 0;
float32 H2_ant = 0;
float32 H2_Vmax = 0;
float32 H2_Vmin = 0;
unsigned int16 H2_temp = 0;
unsigned int8 ERROR = 0.1;
int1 pujaBaixaH1;
int1 pujaBaixaH2;
float32 yt1,yt2,yt3,yt4,xt2,xt3,xt4;
```

```
E:\Dropbox\Grau en Eng. Electrònica i Automàtica\4 - Quart\TREBALL FINAL DE GRAU\Memòri
float32 yt1h2,yt2h2,yt3h2,yt4h2,xt2h2,xt3h2,xt4h2;

/*****************************************************************************
*                                    INTERRUPCIONS
*****************************************************************************
//INTERRUPCIÓ QUE S'ACTIVA QUAN ES DETECTEN DADES EN EL PORT RS232 (RX)
#INT_RDA
void  RDA_isr(void)
{
   putInputRaw(U1RXREG);      //Guardem la dada rebuda del port RS232
}

//INTERRUCPICÓ QUE S'ACTIVA QUAN EL BUFER DE TRANSMISIÓ DEL RS232 ESTÀ LLIU
#INT_TBE
void  TBE_isr(void)
{
   if(outputRawEmpty())      //Si tenim el buffer de transmissió buit...
   {
      disable_interrupts(INT_TBE);   //desactivem la interrupció de transmi
   }
   else                  //Si te dades...
   {
      U1TXREG = getFromOutputRaw();   //Transmetem la dada
   }
}

//INTERRUPCIÓ QUE S'ACTIVA QUAN EL ADC HA ACABAT LA CONVERSIÓ
#int_ADC1
void  ADC1_isr(void)
{
   float32 H1_act = 0;
   float32 H2_act = 0;
   //float32 H3_act = 0;
   //float32 H4_act = 0;

   H1_act = filterH1(ADC_LUT[ADCBUF0]);      //si no funciona treure el fil
   //printf("H1:%f\n\r",H1_act);
   //H4_act = ADC_LUT[ADCBUF1];
   //printf("H4:%f\n\r",H4_act);
   //H3_act = ADC_LUT[ADCBUF2];
   //printf("H3:%f\n\r",H3_act);
   H2_act = filterH2(ADC_LUT[ADCBUF3]);
   //printf("H2:%f\n\r",H2_act);

   //MÀQUINA D'ESTATS
   switch (estat)
   {
      case 1:     //Comprovem que el valor d'H1 passi un valor de referenci

         if(H1_act >= VREF)
         {
            if(H1_act < H1_ant)
            {
               pujaBaixaH1 = 1;
            }
            else
            {
               pujaBaixaH1 = 0;
            }

            H1_ant = H1_act;

            //printf("PBH1:%d\n\r",pujaBaixaH1);
            //printf("H1_ant:%e\n\r",H1_ant);
```

```
            TMR2 = 0;
            enable_interrupts(INT_TIMER2);   //Habilitem el temporitzador 2
            TON = 1;
            estat = 2;
         }

         H1_ant = H1_act;
         H2_ant = H2_act;

         break;

      case 2:     //Mirem si el valor d'H1 i d'H2 coincideixen

         //printf("H2_act:%e\n\r",H2_act);
         //printf("if(%e >= %e)=%d)\n\r",H2_act,H1_ant,(H2_act >= H1_ant));

         if(H2_act >= H1_ant)       //if(H2_act >= H1_ant)
         //if(H2_act > (H1_ant - H1_ant*ERROR) && H2_act < (H1_ant + H1_ant
         {
            H2_ant = H2_act;
            H2_temp = TMR2;   //Llegim el valor acutal del temporitzador
            estat = 3;
         }

         //H1_ant = H1_act;   //Comentar un cop finalitzades les proves
         H2_ant = H2_act;

         break;

      case 3:  //Comprovem el sentit dels hidròfons

         if(H2_act < H2_ant)
         {
            pujaBaixaH2 = 1;
         }
         else
         {
            pujaBaixaH2 = 0;
         }

         //printf("PBH2:%d\n\r",pujaBaixaH2);

         if(pujaBaixaH1 == pujaBaixaH2)
         {
            estat = 4;
         }
         else
         {
            estat = 2;
         }

         break;
   }

   //COMPROVEM ELS VALORS MÀXIMS I MÍNIM DE CADA HIDRÒFON
   if(H1_act > H1_Vmax)
   {
      H1_Vmax = H1_act;
   }
   else
   {
      if(H1_act < H1_Vmin)
      {
         H1_Vmin = H1_act;
      }
```

```
   }

   if(H2_act > H2_Vmax)
   {
      H2_Vmax = H2_act;
   }
   else
   {
      if(H2_act < H2_Vmin)
      {
         H2_Vmin = H2_act;
      }
   }
}

//INTERRUPCIÓ DEL TIMER 2
#int_TIMER2
void  TIMER2_isr(void)
{
   //output_toggle(PIN_E2);
   //printf("wop\n\r");
   TMR2 = 0;
   estat = 0;
}

/*******************************************************************************
*                                PROGRAMA PRINCIPAL
*******************************************************************************/

void main()
{
   configuracio();

   //DEFINICIÓ DE VARIABLES LOCALS
   float32 AngleH12;
   int8 caracter;
   float32 H2_tempDelay;

   printf ("INICIALITZANT...\n\r");

   //ad5282_write_val(AD5282_1_ADDR,0,VAL_POT);
   //ad5282_write_val(AD5282_1_ADDR,1,VAL_POT);

   ad5282_write_val(AD5282_2_ADDR,0,VAL_POT);
   ad5282_write_val(AD5282_2_ADDR,1,VAL_POT);

   //INICI DEL BUCLE INFINIT
   while(true)
   {

      //printf("E:%d\n\r",estat);
      switch (estat)
      {
         case 0:  //Inicialitzem el convertidor ADC

            ADON = 1;
            enable_interrupts(INT_ADC1);

            estat = 1;

            break;

         case 4:  //Calculem el valor de l'angle

            disable_interrupts(INT_TIMER2);
```

```
            disable_interrupts(INT_ADC1);
            ADON = 0;
            TON = 0;
            H2_tempDelay = tempsDelay(H2_temp);
            printf("T2:%Ld,H2D:%e\n\r",H2_temp,H2_tempDelay);
            angleH12 = angleEmissor(H2_tempDelay, SOUNDVEL, DISTH);
            printf("AH12:%f\n\r\n\r",angleH12);

            estat = 0;

            break;
      }

      if(!inputRawEmpty())             //En cas que el buffer 232 no estigu
      {
         caracter=getFromInputRaw();   //Guardem un byte dins de car
         //printf("Caracter rebut: %c\n\r",caracter);

         if(caracter == 'I')
         {
            H1_Vmax = 0;
            H1_Vmin = 0;
            H2_Vmax = 0;
            H2_Vmin = 0;
         }

         if(caracter == '0')
         {
            estat = 0;
         }

         if(caracter == 'E')
         {
            printf ("E:%d\n\r",estat);
         }

         if(caracter == 'V')
         {
            VREF = VREF + 0.01;
            printf ("V:%e\n\r",VREF);
         }

         if(caracter == 'B')
         {
            VREF = VREF - 0.01;
            printf ("V:%e\n\r",VREF);
         }

         if(caracter == 'M')
         {
            printf ("H1+:%e\n\r",H1_Vmax);
            printf ("H1-:%e\n\r",H1_Vmin);
            printf ("H2+:%e\n\r",H2_Vmax);
            printf ("H2-:%e\n\r",H2_Vmin);
         }

         if(caracter == 'A')
         {
            printf("H1:%Ld,%e\n\r",ADCBUF0,ADC_LUT[ADCBUF0]);
            printf("H2:%Ld,%e\n\r",ADCBUF3,ADC_LUT[ADCBUF3]);
         }

         if(caracter == 'F')
         {
            printf("H1f:%e\n\r",filterH1(ADC_LUT[ADCBUF0]));
```

```
            printf("H2f:%e\n\r",filterH2(ADC_LUT[ADCBUF3]));
         }

         if(caracter == '+')
         {
            VAL_POT = VAL_POT+5;
            //ad5282_write_val(AD5282_1_ADDR,0,VAL_POT);
            //ad5282_write_val(AD5282_1_ADDR,1,VAL_POT);
            //printf("Valor de resistencia P1:%f\n\r",(ad5282_read_val(AD52
            ad5282_write_val(AD5282_2_ADDR,0,VAL_POT);
            ad5282_write_val(AD5282_2_ADDR,1,VAL_POT);
            printf ("P2:%e\n\r",ad5282_read_val(AD5282_2_ADDR)*POT_STEP);
         }

         if(caracter == '-')
         {
            VAL_POT = VAL_POT-5;
            //ad5282_write_val(AD5282_1_ADDR,0,VAL_POT);
            //ad5282_write_val(AD5282_1_ADDR,1,VAL_POT);
            //printf("Valor de resistencia P1:%f\n\r",ad5282_read_val(AD528
            ad5282_write_val(AD5282_2_ADDR,0,VAL_POT);
            ad5282_write_val(AD5282_2_ADDR,1,VAL_POT);
            printf ("P2:%e\n\r",ad5282_read_val(AD5282_2_ADDR)*POT_STEP);
         }
      }

      //COMPROVACIÓ QUE NO HI HAGI OVERRUN ERROR
      if (U1STA & 0x02)   //Si hi ha overrun error en el port RS232...
      {
         U1STA = 0x80;      //Netejem el CREN bit
         U1STA = 0x90;      // Posa a 1 el bit CREN i SPEN
      }
   }
}

/*************************************************************************
*                           FUNCIÓ DE CONFIGURACIÓ
*************************************************************************
*   Aquesta funcio ens permet fer la configuració dels diferents móduls del
*   dsPIC. Aquests són el módul UART, I2C i l'ADC
*************************************************************************/
void configuracio()
{
   //Inicialització dels buffers d'emegatzematge rotatius
   inputRawInit();
   outputRawInit();

   //Inicialització dels potenciometres a un determinat valor VAL_POT
   ad5282_write_val(AD5282_1_ADDR,0,VAL_POT);
   ad5282_write_val(AD5282_1_ADDR,1,VAL_POT);
   ad5282_write_val(AD5282_2_ADDR,0,VAL_POT);
   ad5282_write_val(AD5282_2_ADDR,1,VAL_POT);

   //Configuració del modul ADC i dels seus registres
   ADPCFG = 0b1111111110000111;
            /*1111111110000111 AN15-AN7 & AN2-AN0 in digital mode
              ---------0000--- AN3,AN4, AN5 & AN6 in analog mode*/
   ADCON1 = 0b0000000011101100;
           /* -X-XXX-----X---- Unimplemented, set to 0
              0--------------- ADON: A/D Operating Mode bit
              --0------------- ADSIL: Stop in Idle Mode bit
              ------00-------- FORM<1:0> Data Output Format bits
              --------111----- SSRC<2:0> Conversion Trigger Source Select b
              ------------1--- SIMSAM: Simultaneous Sample Select bit
              -------------1-- ASAM: A/D Sample Auto-Start bit
```

```
              --------------0- SAMP: A/D Sample Enable bit
              ---------------X DONE: A/D Conversion Status bit (Read only)*/
   ADCHS = 0b0000000000100110;
           /*XXXXXXXX-------- MUX B input selections, not used
             --------0x------ CH123NA<1:0> MUX A CH1-CH3 negative input is V
             ----------1----- CH123SA: MUX A CH1 positive input is AN3, CH2
             -----------0---- CH0NA: MUX A CH0 negative input is VREF-
             ------------0110 CH0SA<3:0> MUX A CH0 positive input is AN6*/

   ADCSSL = 0b0000000000000000; //No channels participate in input scan
   ADCON3 = 0b0000110000000100;
            /*XXX------X------ Unimplemented, set to 0
              ---01100-------- SAMC<4:0> Auto-Sample Time bits
              --------0------- ADRC: A/D Conversion Clock Source bit
              ----------000100 ADCS<5:0> A/D Conversion Clock Select bits*/
   ADCON2 = 0b0110001100000000;
            /*---XX----X------ Unimplemented, set to 0
              011------------- VCFG<2:0> Voltage Reference Configuration bi
              -----0---------- CSCNA: Scan Input Selections for CH0+ S/H In
              ------11-------- CHIPS<1:0> Selects Channels Utilized bits
              --------X------- BUFS: Buffer Fill Status bit
              ----------0000-- SMPI<3:0> Sample/Convert Sequences Per Inter
              --------------0- BUFM: Buffer Mode Select bit
              ---------------0 ALTS: Alternate Input Sample Mode Select bit

   ADON = 0;  //Posar a 1 per conversió ADC

   //Configuració de les interrupcions de cada módul
   setup_timer1(TMR_DISABLED);
   //Configuració del timer 2
   TCKPS0 = 0;      //Timer Input Clock Presale Select bits 0
   TCKPS1 = 0;      //Timer Input Clock Presale Select bits 1
   T32 = 0;         //32-bit Timer Mode Select bits
   TCS = 0;         //Timer Clock Source Select bit
   TON = 0;         //Timer On Control bit
   TMR2 = 0;      //Temps d'overflow = 8.191875ms
   setup_timer4(TMR_DISABLED |TMR_DIV_BY_1 ,0);
   setup_timer3(TMR_DISABLED |TMR_DIV_BY_1 ,0);
   setup_timer5(TMR_DISABLED |TMR_DIV_BY_1 ,0);
   enable_interrupts(INT_RDA);
   disable_interrupts(INT_TBE);
   disable_interrupts(INT_ADC1);
   disable_interrupts(INT_EEPROM);
   enable_interrupts(INT_TIMER2);
}

/*************************************************************************
*                              FUNCIÓ DE FILTRATGE
*************************************************************************
*   Aquesta funció ens permet filtrar el senyal d'entrada dels hidròfons
*   a partir d'un filtre Butterworth de X ordre
*************************************************************************
float32 filterH1(float32 in)
{
   float32 yt;

   yt = 3.97450921470395*yt1-5.9350134244133876*yt2+3.9463557967817047*yt3-

    yt4 = yt3;
    yt3 = yt2;
    yt2 = yt1;
    yt1 = yt;

    xt4 = xt3;
    xt3 = xt2;
```

```
    xt2 = in;

   return yt;
}

float32 filterH2(float32 in)
{
   float32 yth2;

   yth2 = 3.97450921470395*yt1h2-5.9350134244133876*yt2h2+3.946355796781704

    yt4h2 = yt3h2;
    yt3h2 = yt2h2;
    yt2h2 = yt1h2;
    yt1h2 = yth2;

    xt4h2 = xt3h2;
    xt3h2 = xt2h2;
    xt2h2 = in;

   return yth2;
}

/*************************************************************************
*                            FUNCIÓ LLEGIR POTENCIOMETRE
*************************************************************************
*   Aquesta funció ens permet llegir els valors de resitència dels potenci-
*   ometres indicant la adreça, tenir en compte que per llegir el pot1 avan
*   s'ha d'haver escrit en el pot1 i de la mateixa forma pel pot2.
*************************************************************************
unsigned int16 ad5282_read_val(unsigned int8 adr)
{
   unsigned int8 read = 0b00000001;
   unsigned int8 data;

   adr <<= 1;
   adr |= read;
   i2c_start();
   i2c_write(adr);
   data = i2c_read();
   i2c_stop();

   return data;
}

/*************************************************************************
*                           FUNCIÓ ESCRIURE POTENCIOMETRE
*************************************************************************
*   Aquesta funció ens permet escriure un valor de resistència en els poten
*   ometres, indicant la adreça d'aquest, el pot que volem escollir i final
*   el valor de resistencia que hi volem escriure (0-255)
*************************************************************************
void ad5282_write_val(unsigned int8 adr, int1 pot, unsigned int8 data)
{
   unsigned int8 instr;

   adr <<= 1;
   adr &= 0xFE;
   instr = pot;
   instr <<= 7;
   instr &=0x80;

   i2c_start();
   i2c_write(adr);
   i2c_write(instr);
```

```
   i2c_write(data);
   i2c_stop();
}

/*************************************************************************
*                          FUNCIÓ PER DETERMINAR EL TEMPS
*************************************************************************
*   Aquesta funció el que fa és un cop aconseguida la diferència entre els
*   transforma el valor actual del timer per un valor real de temps
*************************************************************************
float32 tempsDelay(unsigned int16 stepTim)
{
   int8 prescaler = 1;   //Valor del prescaler del temporitzador
   float32 delay = 0;

   delay = Tcy*prescaler*stepTim;

   return delay;
}

/*************************************************************************
*      FUNCIÓ QUE PERMET CALCULAR L'ANGLE A PARTIR DEL DELAY DE TEMPS
*************************************************************************
*   Aquesta funció se li passa el valor de temps de delay entre hidròfons i
*   retorna el valor de l'angle real que hi ha entre aquests i l'emissor
*   acústic. Els paràmetres que se li passen són el delay de temps entre hi
*   fons, la velocitat del so a l'aigua i la distància entre els hidròfons
*************************************************************************
float32 angleEmissor(float32 temp, float32 soundVel, float32 dist)
{
   float32 angle = 0;
   float32 tempo = 0;
   tempo = (temp*soundVel)/dist;
   angle = asin(tempo);
   return angle;
}
```