PC/104 az űrtechnológiában

Bevezetés

Ebben a laboratóriumi gyakorlatban egy PC/104 formátumú ipari számítógéppel és ennek mérés-adatgyűjtésre történő alkalmazásával lehet megismerkedni. A mérés dokumentálására a mérési utasítás végén található forma használható.

Áttekintés

A PC/104 egy beágyazott számítógép szabvány, amelyiknél a kártyaméretek pontosan definiáltak (3.775”*3.550”), továbbá egy úgynevezett ’stacking’, vagyis egymásra illeszthető struktúrát képeznek. Ez egy moduláris, robusztus PC szabvány. Nincs alaplap, az egyes modulok az ISA, PCI és PCIe buszokon keresztül csatlakoznak egymáshoz. A szabványos helyeken elhelyezett szerelési csavarok biztosítják a stabil rögzítést.

Az egységes interfészeknek köszönhetően többféle gyártó eszközei is társíthatóak, a saját igényeinknek megfelelő modulok összeválogatásával.

A kompakt szerkezet stabil, vibrációmentes elrendezést nyújt, többnyire nem tartalmaznak mozgó alkatrészt és számos modul specifikált a széles sávú, -40/+85°C hőmérsékleti tartományra. A modulok teljesítményfelvétele egy szokásos PC kompatibilis rendszerhez képest igen csekély, a kisebb teljesítményű processzorokkal szerelt eszközök csupán 5V tápfeszültséget és 5-6W energiát igényelnek.

Egy alapkiépítésben egy CPU modul és egy tápegység modul már egy működőképes PC kompatibilis rendszert képez. Ez kiegészíthető további modulokkal, mint például mérés-adatgyűjtő, DSP, GPS, videó kontroller, vagy ipari kapcsoló-modulok.

Egy PC/104 stack (forrás: www.rtd.com)

A fenti tulajdonságok ismeretében nem meglepő, hogy ezeket az eszközöket számos űrkutatási programban is használják. A felhasznált alkatrészek ugyan nem űrminősítésűek, de ez számos alacsonyabb költségvetésű missziónál nem is követelmény. A széles működési hőmérséklettartomány, a mozgó alkatrészek (pl. ventilátor) hiánya, a vibráció-érzéketlenség

lehetővé teszi, hogy műholdak, űrszondák, leszálló egységek elemeiként is alkalmazzák ezeket. (például NASA Lunar Micro-Rover, Lunar Crater Observation and Sensing Satellite, stb.)

A felhasznált eszközök és műszerek

PC/104 fejlesztői platform, ISA104D
CPU modul
DM7520 mérés-adatgyűjtő modul
interface panel
oszcilloszkóp
jelgenerátor

Mérési feladatok

A mérés célja a megismerkedés a PC/104 technológiával a DM7520 mérés-adatgyűjtő modul segítségével, és néhány egyszerű program elkészítése a már rendelkezésre álló alapprogramokból.

Ezek a következők:

- Digitális I/O műveletek
- Analóg jel mintavételezése
- Analóg jel előállítása
- Időzítő használata

A PC/104-es CPU modul kompatibilis az ismert PC-s operációs rendszerekkel (Linux, Windows, DOS).

Ezek közül egy űreszköz esetében többnyire nincs szükség a grafikus felhasználói felületre, hanem az adott alkalmazásnak megfelelő feladatot ellátó szoftvert kell írni, amelyik esetleg valamilyen kommunikációs csatornán (soros port, CAN busz, Ethernet) csatlakozik az űreszköz egyéb részeihez, vagy a telemetria rendszerhez. Éppen ezért ebben a mérésben Windows konzol alatti programokat kell elkészíteni. A fejlesztési folyamatot megkönnyítendő és felgyorsítandó, a mérésben használt PC/104 fejlesztői platformon egy HDD-vel ellátott rendszer van Windows XP környezettel.

A programírás WinXP alatt, Microsoft Visual Studio 6.0 környezetben történik, amely szintén telepítve van a mérés céljára szolgáló PC/104 számítógépen. Az elkészült programokat konzol ablakban futtatva ellenőrizzük. Ezzel jól szimulálható egy olyan rendszer, amelyik például egy SSD diszkről bootolva elindítja az operációs rendszert és az erre írt applikációt, amely lehet akár egy űreszköz fedélzeti számítógépének vagy mérés-adatgyűjtőjének a központi programja is.

Az alábbiakban a négyféle programozási feladat legfontosabb kódrészleteit mutatjuk be. A programok a gyártó által a fejlesztők részére rendelkezésre bocsájtott driver függvények használatával kezelik a különféle hardware funkciókat. A megnyitás az OpenBoard függvénnyel történik, ezt a kezdeti állapotot beállító InitBoard függvény követi. Ezután az egyes funkciókhoz tartozó specifikus függvények hívása történik.

1. feladat: Digitális I/O műveletek

A program a billentyűzeten beírt számjegyet kiírja az 1-es digitális I/O portra, valamint beolvassa a 0-s port értékét.

```
#include <stdio.h>
#include <conio.h>
#include "..\dm7520.h"

/***************************************************************************
  Defines
***************************************************************************/
#define BOARD_NO     1                 // We use the first DM7520 board
/***************************************************************************/

BoardConfig7520 boardconfig;           // Get board configuration from the driver
int DEVICE_NO;                         // board handle returned by OpenBoard7520
int  board_id;

void main(void)
{
   int      DOData;     // Data for digital output
   int      ExitProgram;

   char     C;
   char szBuf[512];

   // Load driver and open board DM7520 for use
   board_id = 0x7520;
   if ( OpenBoard7520(0x7520,BOARD_NO,&boardconfig) )
      {
      printf("\nBoard Open Error");
      return;
      }
        else
      printf("\nBoard Opened");

   DEVICE_NO = boardconfig.device_no;

   InitBoard7520(DEVICE_NO);     // Board initializing

   SetupPort07520(DEVICE_NO, 0, 0, 0, 0, 0, 0);   // Initializing port 0 for input.  P1.0-P1.7
   SetupPort17520(DEVICE_NO, 1, 0, 0, 0, 0, 0);   // Initializing port 1 for output. P0.0-P0.7

   ExitProgram = 0;
   while (!ExitProgram) {              // Run until <Esc> is pressed
   printf( "\n%6d", ReadDIO07520(DEVICE_NO) );

        // Read the digital input lines

   if (kbhit()) {             // If key pressed get value to output
                C = getch();
                if (C == 27)            // If <Esc> then quit
                    ExitProgram = 1;
                else if ((C >= 0x30)
                                & (C <= 0x39))   // C is a digit
         {
        printf("\n");
                  putch(C);
                  ungetch(C);
                  cscanf("%d", &DOData);
        WriteDIO17520(DEVICE_NO,DOData);         // Write data to digital output
                }
         }
     Sleep(500);
  }
  CloseBoard7520(DEVICE_NO,szBuf);
  printf("\nBoard Closed");
}
```

2. feladat: Analóg jel mintavételezése

A program folyamatos A/D konverziót végez és kiírja a konvertált értéket a képernyőre.

```
#include <stdio.h>
#include <conio.h>
#include "..\dm7520.h"

/**************************************************************************
  Defines
**************************************************************************/
#define BOARD_NO     1                // We use the first DM7520 board
#define CHANNEL      0                // A/D channel
#define GAIN         0                // Gain
#define RANGE        0                // Range = +/- 5 volts
#define SE_DIFF      0                // Single ended
#define ADSLOPE      (4096.0/10.0)    // Number of bits divided by AD range
/**************************************************************************/

BoardConfig7520 boardconfig;          // Get board configuration from the driver
int DEVICE_NO;                        // Board handle returned by OpenBoard7520
int  board_id;

void main(void)
{
   int      ADData;  // Variable for acquired analog input data
   double   Data;    // Variable for voltage
   char szBuf[512];

   // Load driver and open board DM7520 for use
   board_id = 0x7520;
   if ( OpenBoard7520(0x7520,BOARD_NO,&boardconfig) )
      {
      printf("\nBoard Open Error");
      return;
      }
        else
      printf("\nBoard Opened");

   DEVICE_NO = boardconfig.device_no;

   InitBoard7520(DEVICE_NO);     // Board initializing

   SetConversionSelect7520(DEVICE_NO, ADC_START_SOFTWARE);  // Set conversion select to software
   SetChannelGain7520(DEVICE_NO, CHANNEL, GAIN, RANGE, SE_DIFF, NRSE_AGND); // Set channel and gain
   ClearADFifo7520(DEVICE_NO);                              // Clear A/D fifo

   while ( !kbhit() ) {
   StartConversion7520(DEVICE_NO);        // Perform a Single A/D conversion
   while ( IsADFifoEmpty7520(DEVICE_NO) );// Wait until data is ready to read

   ADData = ReadADData7520(DEVICE_NO);    // Read acquired data
   Data = ADData / ADSLOPE;               // Convert Data to voltages

   printf("\n%2.2f V",Data);
   Sleep(1000);
   }
   CloseBoard7520(DEVICE_NO,szBuf);
   printf("\nBoard Closed");
}
```

3. feladat: Analóg jel előállítása

A program egy folyamatosan változó feszültségszintet állít be a D/A konverter kimenetén, amelyet az A/D konverter bemenetére visszakötve meg is mérhetünk.

```
#include <stdio.h>
#include <conio.h>
#include "..\dm7520.h"

/*************************************************************************
  Defines
*************************************************************************/
#define BOARD_NO     1                // We use the first DM7520 board
#define CHANNEL      0                // A/D channel
#define GAIN         0                // Gain
#define RANGE        0                // Range = +/- 5 volts
#define SE_DIFF      0                // Single ended
#define ADSLOPE      (4096.0/10.0)    // Number Bits divided by AD Range.
#define DAC_SLOPE    (4095.0/10.0)    // Number Bits divided by DAC Range
#define DAC_OFFSET   0                // Offset of the output line signal
/*************************************************************************/

BoardConfig7520 boardconfig;          // Get board configuration from the driver
int DEVICE_NO;                        // board handle returned by OpenBoard7520
int  board_id;

void main(void)
{
   int      ADData;     // Variable for acquired analog input data
   int      DAData;     // Data for analog output.
   float    mV;

   double   Data;       // Variable for voltage
   char szBuf[512];

   // Load driver and open board DM7520 for use
   board_id = 0x7520;
   if ( OpenBoard7520(0x7520,BOARD_NO,&boardconfig) )
      {
      printf("\nBoard Open Error");
      return;
      }
        else
      printf("\nBoard Opened");

   DEVICE_NO = boardconfig.device_no;

   InitBoard7520(DEVICE_NO);     // Board initializing

   SetConversionSelect7520(DEVICE_NO, ADC_START_SOFTWARE);  // Set conversion select to software
   SetChannelGain7520(DEVICE_NO, CHANNEL, GAIN, RANGE, SE_DIFF, NRSE_AGND); // Set channel and gain
   ClearADFifo7520(DEVICE_NO);                              // Clear A/D fifo

   // DAC 0, unipolar 5V, no cycle, sw. start
   SetupDAC7520(DEVICE_NO,0,2,DAC_CYCLE_SINGLE,DAC_START_SOFTWARE);
   ClearDAC1Fifo7520(DEVICE_NO);                // Clear A/D fifo

   mV = -5000;                                  // Start DAC scan at -5000 mvolts
   while ( !kbhit() ) {                         // Run until any key is pressed

   DAData = (int)((mV / 1000.0) * DAC_SLOPE) + DAC_OFFSET;

   LoadDAC17520(DEVICE_NO, DAData);             // Sending analog output data

   UpdateDAC17520(DEVICE_NO);

   StartConversion7520(DEVICE_NO);        // Perform a Single A/D conversion
   while ( IsADFifoEmpty7520(DEVICE_NO) );// Wait until data is ready to read

   ADData = ReadADData7520(DEVICE_NO);    // Read acquired data
   Data = ADData / ADSLOPE;               // Convert Data to voltages
```

```
   printf("\nOut=%2.2f mV In=%2.2f V",mV,Data);
   Sleep(1000);

   mV += 500;                              // Increment data
   if (mV > 5000)                          // If more than 5,000 mvolts
   mV = -5000;                             // Start over at -5,000 mvolts
   } //while

   CloseBoard7520(DEVICE_NO,szBuf);
   printf("\nBoard Closed");
}
```

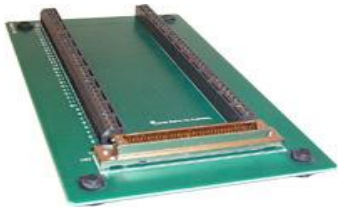
4. feladat: Időzítő használata

A program a kártyára épített időzítők használatát demonstrálja.

```
#include <stdio.h>
#include <conio.h>
#include "..\dm7520.h"

/*************************************************************************
  Defines
*************************************************************************/
#define BOARD_NO     1                 // We use the first DM7520 board
/*************************************************************************/

BoardConfig7520 boardconfig;           // Get board configuration from the driver
int DEVICE_NO;                         // board handle returned by OpenBoard7520
int  board_id;

void main(void)
{
   uint16   count;
   uint8    mode;

   char szBuf[512];

   // Load driver and open board DM7520 for use
   board_id = 0x7520;
   if ( OpenBoard7520(0x7520,BOARD_NO,&boardconfig) )
      {
      printf("\nBoard Open Error");
      return;
      }
        else
      printf("\nBoard Opened");

   DEVICE_NO = boardconfig.device_no;

   InitBoard7520(DEVICE_NO);     // Board initializing

   SetUtc0Clock7520 (DEVICE_NO, CUTC0_8MHZ);
   SetUtc1Clock7520 (DEVICE_NO, CUTC1_UTC0_OUT);
   SetupTimerCounter7520(DEVICE_NO,TC_UTC0,M8254_SQUARE_WAVE,40000);
   SetupTimerCounter7520(DEVICE_NO,TC_UTC1,M8254_EVENT_COUNTER,0xFFFF);

   while (!kbhit()) {             // Run until any key pressed

   ReadTimerCounter7520( DEVICE_NO, TC_UTC1, &mode, &count );

   printf("\nElapsed %2.2f seconds",(float)(65535.0-(float)count) / 200.0);
   }
  CloseBoard7520(DEVICE_NO,szBuf);
  printf("\nBoard Closed");
}
```

A programok hardware jeleket generálnak, melyeket az adatgyűjtő kártyához csatlakoztatott terminál-kártyán oszcilloszkóppal lehet ellenőrizni. Az egyes jeleket a következő számozás szerint lehet megtalálni:

AIN 9 / AIN1-	2	1	AIN 1 / AIN1+
AIN 10 / AIN2-	4	3	AIN 2 / AIN2+
AIN 11 / AIN3-	6	5	AIN 3 / AIN3+
AIN 12 / AIN4-	8	7	AIN 4 / AIN4+
AGND	10	9	AINSENSE
AIN 13 / AIN5-	12	11	AIN 5 / AIN5+
AIN 14 / AIN6-	14	13	AIN 6 / AIN6+
AIN 15 / AIN7-	16	15	AIN 7 / AIN7+
AIN 16 / AIN8-	18	17	AIN 8 / AIN8+
AGND	20	19	AGND
Reserved	22	21	AOUT 1
Reserved	24	23	AOUT 2
AGND	26	25	Reserved
AGND	28	27	AGND
D/A 2 DATA MARKER 0	30	29	D/A 1 DATA MARKER 0
P1.7 / DIG TABLE 7	32	31	HIGH SPEED INPUT 7 / P0.7 / A/D DM2
P1.6 / DIG TABLE 6	34	33	HIGH SPEED INPUT 6 / P0.6 / A/D DM1
P1.5 / DIG TABLE 5	36	35	HIGH SPEED INPUT 5 / P0.5 / A/D DM0
P1.4 / DIG TABLE 4	38	37	HIGH SPEED INPUT 4 / P0.4
P1.3 / DIG TABLE 3	40	39	HIGH SPEED INPUT 3 / P0.3
P1.2 / DIG TABLE 2	42	41	HIGH SPEED INPUT 2 / P0.2
P1.1 / DIG TABLE 1	44	43	HIGH SPEED INPUT 1 / P0.1
P1.0 / DIG TABLE 0	46	45	HIGH SPEED INPUT 0 / P0.0
DGND	48	47	TRIGGER INPUT
RESET	50	49	EXTERNAL PACER CLOCK INPUT
DGND	52	51	EXTERNAL INTERRUPT INPUT
USER INPUT 1	54	53	USER INPUT 0
USER OUTPUT 1	56	55	USER OUTPUT 0
DGND	58	57	TC OUT 0
EXTERNAL TC GATE 1	60	59	EXTERNAL TC CLOCK 1
TC OUT 2	62	61	TC OUT 1
EXTERNAL TC GATE 2	64	63	EXTERNAL TC CLOCK 2
DGND	66	65	+5 VOLTS
DGND	68	67	+5 VOLTS



Terminal board és a jelek kiosztása (forrás: www.rtd.com)

A mérés helye: V1-215 labor	A mérés időpontja:
Mérésvezető:	
A mérést végzi:	

1) A mérésvezető útmutatásai alapján töltsük be az egyes példaprogramokat, majd fordítás után futtassuk azokat. Ellenőrizzük a működést jelgenerátor és oszcilloszkóp segítségével.

2) A feladatok:

Digitális I/O:
Hurkoljunk vissza néhány bitet a digitális kimenetről a bemenetre. Ellenőrizzük a visszaolvasott értékeket oszcilloszkóppal.
További lehetséges feladatok:
A billentyűzetre való várakozást kivéve, mi a maximálisan elérhető négyszögjel-frekvencia a kimeneten?

Analóg jel mintavételezése:
Jelgenerátorból adjunk egy lassan változó színuszos feszültséget az analóg bemenetre, és ellenőrizzük a mért értéket. Ügyeljünk a bemeneti feszültségtartományra.
További lehetséges feladatok:
Módosítsuk úgy a programot, hogy 1 másodpercenként automatikusan vegyen egy mintát és azt egy fileban tárolja el.

Analóg jel előállítása:
Ellenőrizzük az analóg kimenet állapotát oszcilloszkóppal. Kössük vissza az analóg kimenetet az analóg bemenetre. Ellenőrizzük a mért értéket.
További lehetséges feladatok:
Állítsunk elő egy színuszjelet az analóg kimeneten.

Időzítő használata:
Futtassuk a programot és teszteljük az időzítő pontosságát 1 perces méréssel.
További lehetséges feladatok:
Végezzünk időzített digitális I/O műveleteket, például 5 másodpercenként változtassuk a Port0 alsó bitjének az értékét.

Irodalom:

Az Űrtechnológia (VIHVBV06) tantárgy előadási anyagai:
http://home.hvt.bme.hu/~csurgai/urtech/urtech.htm

A mérésben használt PC/104 modulok gyártói honlapja: www.rtd.com