PRESSURE SENSOR

PROJECT OF ELCE721 EMBEDDED SYSTEMS

Tang Qi (M-B5-5515-2)
Department of Electrical and Computer Engineering
Faculty of Science and Technology
mb55515@umac.mo

I. ABSTRACT

This project is to build scheduler on Altera DE2-based 8051 IP core and implement some tasks into it. Here, a pressure sensor is implemented. Pressure sensors are used for control and monitoring in thousands of everyday applications. They can also be used to indirectly measure other variables such as fluid/gas flow, water level.

This report contains four parts. The first part briefly introduced DE2 board and related peripherals, which are 16*2 LCD, piezoelectric transducer, and HX711 ADC. The second part illustrated the process to embed an 8051 IP core in Quartus and implement it into development board of Altera DE2. Next, a scheduler program scheduling the tasks of weight sensing & processing, as well as LCD display is built in Keil and compiled into DE2 via Quartus. In the fourth part, difficulties and solutions are illustrated.

II. INTRODUCTION

1. *Altera DE2 board*

This board provides Altera Cyclone II (2C35) FPGA with 35,000 Les, 16*2 LCD panel and expansion headers GPIO for this project. The layouts of corresponding modules are as shown in Fig.02.

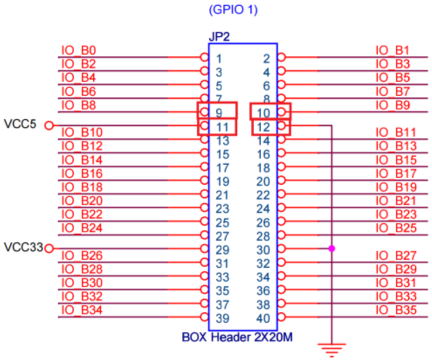


Fig.01 Schematic of header (GPIO1)

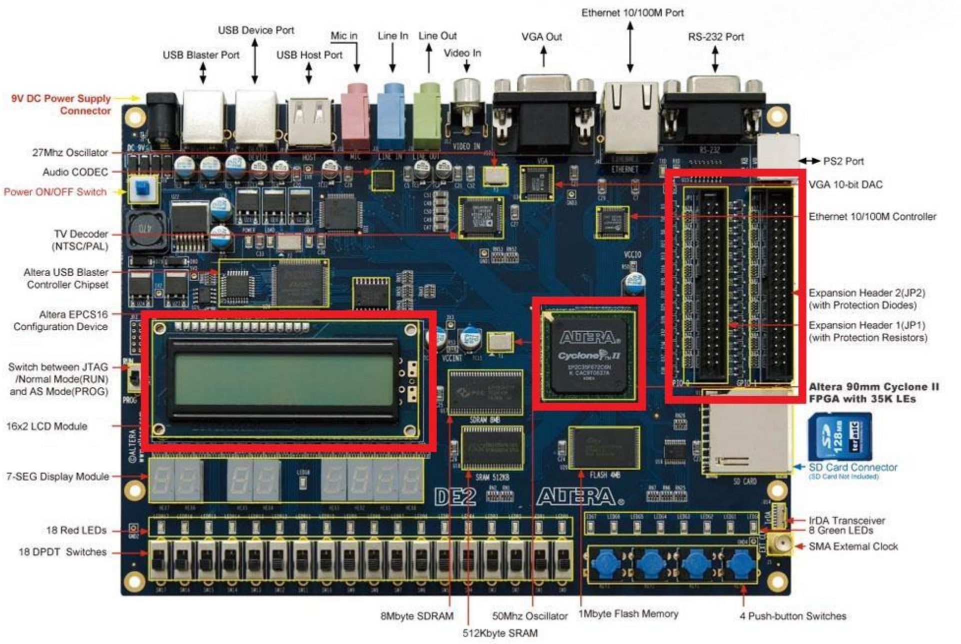


Fig.02 Layout of modules provided in Altera DE2

2. *Peripherals*

2.1 LCD1602

To display the output of the pressure sensor, 1602 LCD panel on board is used. The pins of LCD module is usually controlled by a chip of HD44780 and we only need to assign the pins of GND, Vcc, V0, RS, RW, En, and D0-7 in Keil and connect corresponding pins in "Pin Planner" of Quartus.

Corresponding functions of pins are:

GND: Ground;

Vcc: Direct voltage of 5V;

V0: Brightness of LCD panel controlled by controllable voltage. 0V means the brightest;

RS: Selection on Data/ Command of register. High means data and low means command.

RW: Read/ write. High means read and low means write.

En: Enable LCD.

D0- D7: Data lines.

Because HX711 AD is used to convert the analog to digital signal, we only need to provide a DC voltage of 5V (Vcc, GND), clock (SCK) and connect the data output (DOUT) from header to piezoelectric transducer, as shown in Fig.02. The time of delay should be considered. When we need to transmit data and displays on LCD, if the delay is too short, the LCD will haven't finished write the data. If the delay is too long, the update data would not be in real-time. Thus, the program to drive it would be most essential and later becomes the toughest problem in this project.

2.2 Piezoelectric transducer

Piezoelectricity means electricity resulting from pressure. A piezoelectric sensor is a device that uses the piezoelectric effect, to measure changes in pressure or force by converting them to an electrical charge. It has very high DC output impedance and the voltage V at the source is directly proportional to the applied force, as shown in Fig.03 & 04. The output signal is related to the mechanical force.

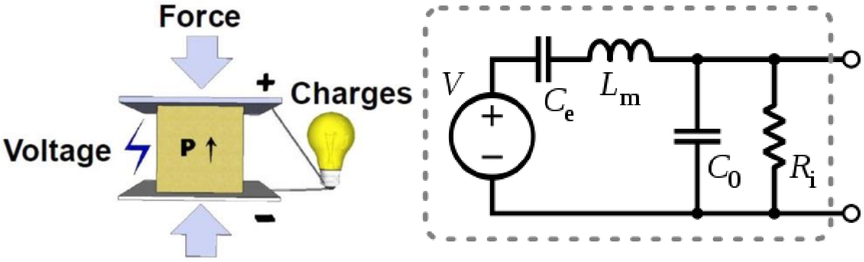


Fig.03 Electronic model of a piezoelectric sensor

After setup, the piezoelectric transducer has upper limit to 5kg, connected to HX711.

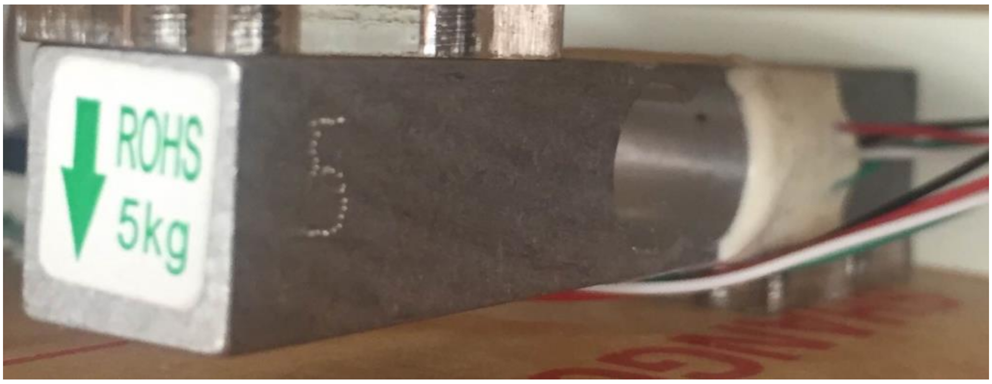


Fig.03 Piezoelectric transducer, upper limit 5Kg

2.3 HX711 AD

HX711 is a 24-bit analog to-digital converter (ADC) designed for weigh scales and industrial control applications to interface directly with a bridge sensor. This element is cheap yet not enough static that cannot be applied to very sensitive measurements. However, it is still enough for the pressure sensing precise to 0.001Kg. Main Characteristics of HX711 are that:

• Current consumption: normal operation < 1.5mA, power down < 1uA
• Operation supply voltage range: 2.6 ~ 5.5V (Here 5V is applied.)
• Operation temperature range: -40 ~ +85℃

There is no programming needed for the internal registers. All controls to the HX711 are through the pins. The schematic of typical weigh scale diagram via HX711 block is shown in Fig.04. Lines of clock, output data, and direct voltage are applied.

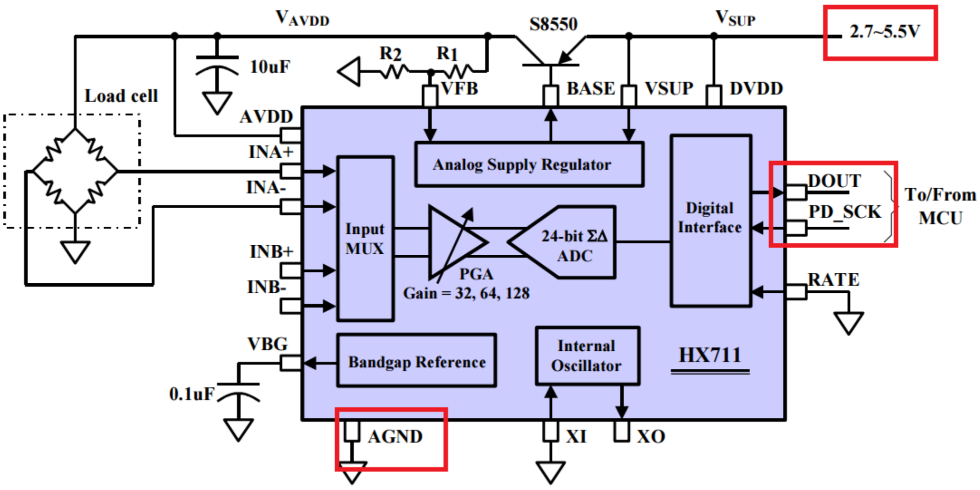


Fig.04 Typical weigh scale application block diagram

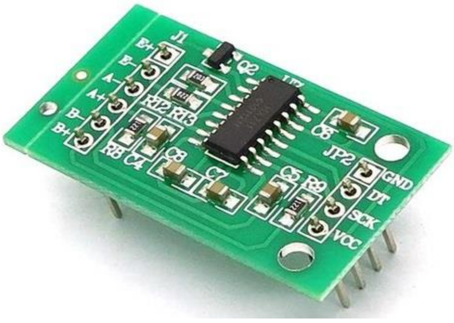


Fig.05 Entity of HX711

Gnd is connected to GPIO1-11 on DE2 board which provides ground. DT is connected to GPIO1-9, SCK connected to GPIO1-10 and Vcc connected to the header GPIO1-12which provides 5V DC on Altera DE2 board.

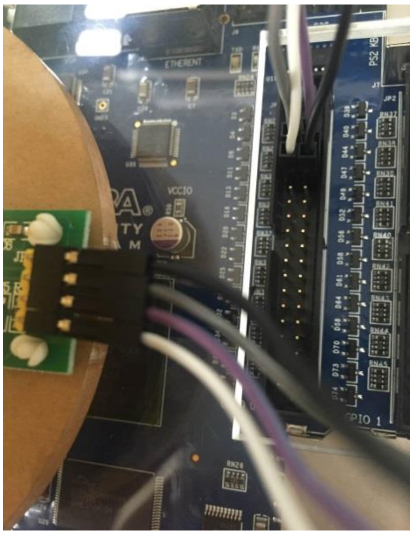


Fig.06 connection of HX711 and GPIO1

III. PROGRAMS

To build a real-time pressure sensor, we need to set the initial pressure (or weight) value to be 0, do the update in every 0.02s and update the LCD result display within less than 0.02s. It is worth mentioning that because the transmission data from HX711 to LCD is not in real-time, delay time should be counted into the LCD display program. The program of scheduler is firstly implemented.

3.1 SCHEDULER PROGRAMS WITH WEIGH AND DISPLAY FUNCTIONS IN KEIL

A scheduler to embedded system is like OS to normal desktop systems. The difference among scheduler, interrupt and delay is that each interrupt and delay is called once a time but a scheduler can re-call the interrupt periodically after initialization. Thus, function of delay would be not necessary.

Before we use scheduler to add tasks, we should first call the initialization of scheduler, to rest the global error variable, and reload using Timer0, to initialize the timing and interrupt source of scheduler. Corresponding codes are as shown:

```
void SCH_Init_T0(void)
    {
    tByte i;
    for (i = 0; i < SCH_MAX_TASKS; i++)
        {
        SCH_Delete_Task(i);
        }
    Error_code_G = 0;
    TMOD &= 0xF0;
    TMOD |= 0x01;
    SCH_Manual_Timer0_Reload();
    ET0 = 1;
    }
```

Then we should active the timer by function of **SCH_start()** and dispatch a task when the task should be called by **SCH_Dispatch_Tasks()**. Report the status and display error codes by **SCH_Report_Status()**. Also, there should be functions of add tasks and delete tasks to ass and remove tasks from scheduler. Corresponding codes are in the appendix.

3.2 Designed program for pressure sensing

The LCD panel should be initialized first. RW of LCD turned high to get ready for write command and a sentence of “Welcome to use!” is displayed. Get weight function is defined and import into HX711

```
void Get_Weight()
{       Weight_Shiwu = HX711_Read();
        if(Weight_Shiwu >= 0)
        {
                Weight_Shiwu = (unsigned long)((float)Weight_Shiwu/GapValue);
        }
        else
        {
                Weight_Shiwu = 0;
        }
}
```

As we want the pressure sensing to be real-time, that the weighing and LCD display function should be updated in a short time by scheduler. The codes for the update function are as shown:

```
void weight_update(void )
{               Get_Weight();
                LCD1602_write_com(0x80);
                LCD1602_write_word("Weight=");//character corresponded to ASCII code
                LCD1602_write_data(Weight_Shiwu%10000/1000 + 0x30);
                LCD1602_write_data('.');
                LCD1602_write_data(Weight_Shiwu%1000/100 + 0x30);
                LCD1602_write_data(Weight_Shiwu%100/10 + 0x30);
                LCD1602_write_data(Weight_Shiwu%10 + 0x30);
                LCD1602_write_word("Kg");
        }
```

For sub-functions are called, which are LCD1602_write_com, LCD1602_write_word, LCD1602_write_data. As mentioned in LCD’s functions of RS, RW, EN. For instance, the **LCD1602_write_data** is called when we need to write data on LCD, then RS is set to high, R/W is set to low, E is set to high, then data is transmitted to data-bus and set E to low (the write data action is executed when EN goes through negative edge). Detail operation can be referred below:

```
void LCD1602_delay_ms()
{
        unsigned int    i,j;
        for(i=0;i<50;i++)
                for(j=0;j<11000;j++);
        //      for(j=0;j<123;j++);
}
```

```
void        LCD1602_write_com(unsigned
char com)
{
    LCD1602_delay_ms();
        LCD1602_EN = 0;
        LCD1602_RS = 0;
    LCD1602_RW = 0;
        LCD1602_PORT = com;
        LCD1602_EN = 1;
        LCD1602_EN = 0;
}
```

```
void        LCD1602_write_data(unsigned
char dat)
{
        LCD1602_delay_ms();
        LCD1602_EN = 0;
        LCD1602_RS = 1;
        LCD1602_RW = 0;
        LCD1602_PORT = dat;
        LCD1602_EN = 1;
        LCD1602_EN = 0;
}
```

```
void       LCD1602_write_word(unsigned
char *s)
{
//      addr=addr|0x80;
//      LCD1602_write_com(addr);
        while(*s!='\0')
        {
                LCD1602_write_data(*s++);
        }
}
```

After building corresponding weighing and LCD displaying functions, we need to add task to the main function of scheduler:

```
void main(void)
                      {
                          weight_initial();
                                  SCH_Init_T0();
                                  SCH_Add_Task(weight_update, 0, 200);
                          SCH_Start();
                      while(1)
                            {
                            SCH_Dispatch_Tasks();
                            }
                      }
```

3.3 Project of embedded 8051 on Altera DE2 via Quartus

After building pressure sensing function in Keil and compile into hex.file, we should store in the ROM of 8051 core. The core itself is made up of the submodules timer/counter, ALU, serial interface, and control unit. RAM or ROM blocks are most often generated corresponding to the selected target technology and are therefore instantiated in the highest design hierarchy. The schematic of embedded 8051 IP core built in Quartus is shown in Fig.07. It is worth mentioning that the difference between this embedded 8051 and 8051 single-chip is that the former one uses the same line as both input and output and the latter one has two separate input and output lines.We build the top level signal exactly the same of MC8051core, Corresponding functions are as shown in Fig.08.

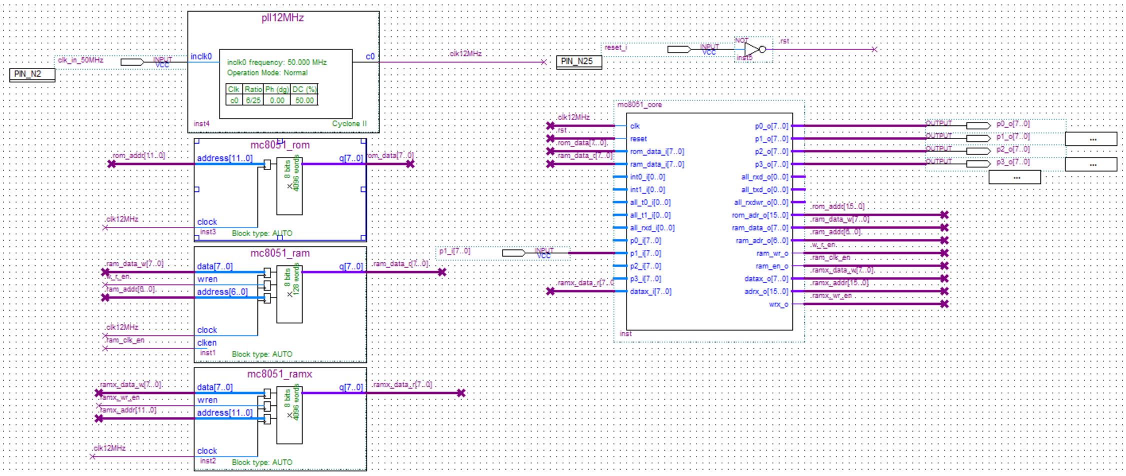
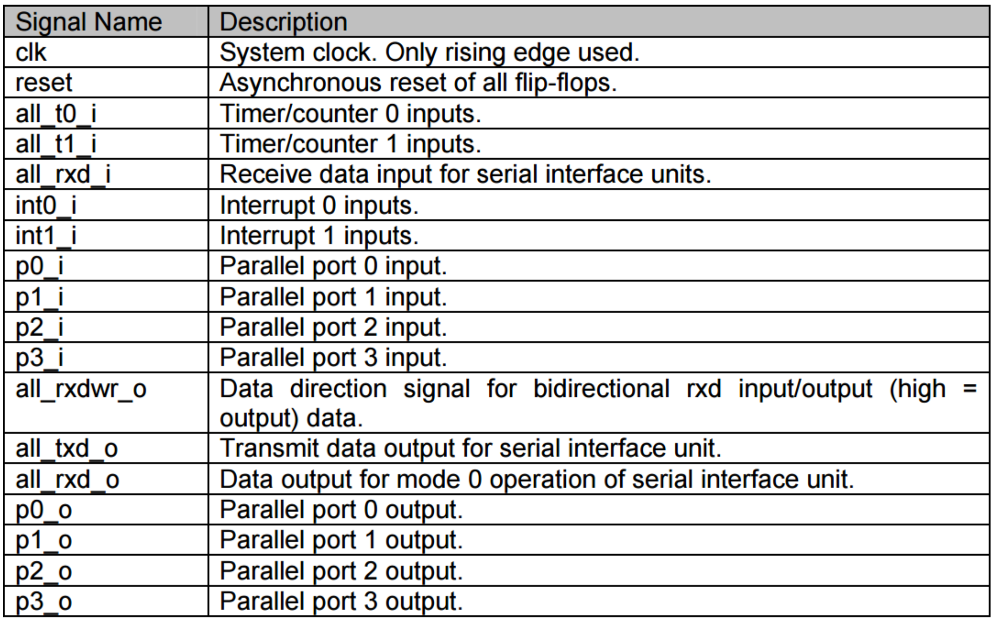


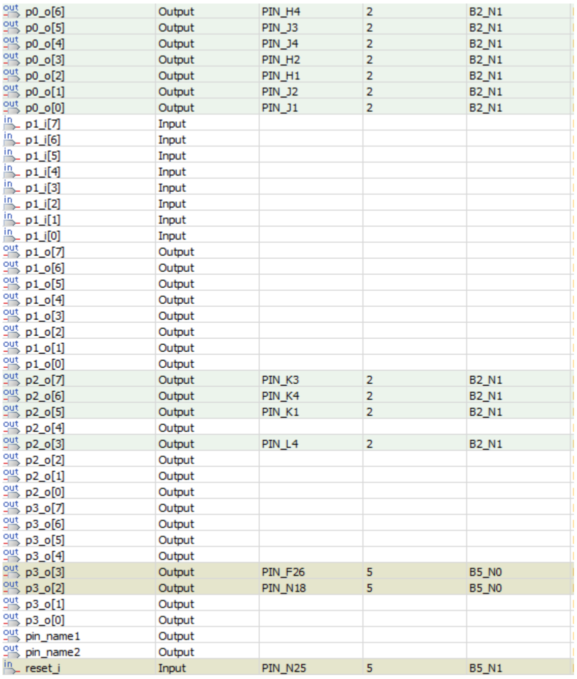
Fig.07 schematic of embedded 8051 IP core



Signal Name	Description
clk	System clock. Only rising edge used.
reset	Asynchronous reset of all flip-flops.
all_t0_i	Timer/counter 0 inputs.
all_t1_i	Timer/counter 1 inputs.
all_rxd_i	Receive data input for serial interface units.
int0_i	Interrupt 0 inputs.
int1_i	Interrupt 1 inputs.
p0_i	Parallel port 0 input.
p1_i	Parallel port 1 input.
p2_i	Parallel port 2 input.
p3_i	Parallel port 3 input.
all_rxdwr_o	Data direction signal for bidirectional rxd input/output (high = output) data.
all_txd_o	Transmit data output for serial interface unit.
all_rxd_o	Data output for mode 0 operation of serial interface unit.
p0_o	Parallel port 0 output.
p1_o	Parallel port 1 output.
p2_o	Parallel port 2 output.
p3_o	Parallel port 3 output.

Fig.08 Top level signal name & functions

After storing the hex. file into the ROM of embedded 8051 and successfully compiling into the program, corresponding pins should be assigned to pins of peripherals in pin planner:



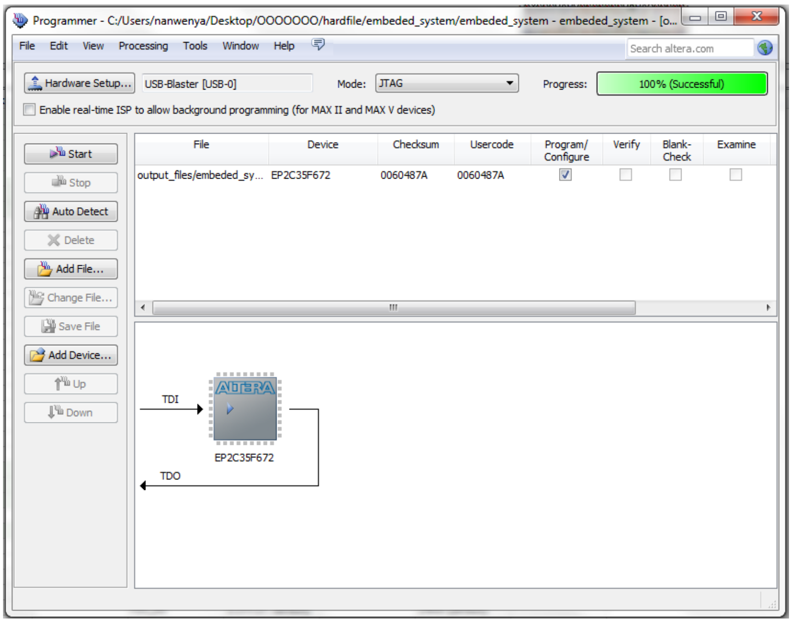
p0_o[6]	Output	PIN_H4	2	B2_N1
p0_o[5]	Output	PIN_J3	2	B2_N1
p0_o[4]	Output	PIN_J4	2	B2_N1
p0_o[3]	Output	PIN_H2	2	B2_N1
p0_o[2]	Output	PIN_H1	2	B2_N1
p0_o[1]	Output	PIN_J2	2	B2_N1
p0_o[0]	Output	PIN_J1	2	B2_N1
p1_i[7]	Input			
p1_i[6]	Input			
p1_i[5]	Input			
p1_i[4]	Input			
p1_i[3]	Input			
p1_i[2]	Input			
p1_i[1]	Input			
p1_i[0]	Input			
p1_o[7]	Output			
p1_o[6]	Output			
p1_o[5]	Output			
p1_o[4]	Output			
p1_o[3]	Output			
p1_o[2]	Output			
p1_o[1]	Output			
p1_o[0]	Output			
p2_o[7]	Output	PIN_K3	2	B2_N1
p2_o[6]	Output	PIN_K4	2	B2_N1
p2_o[5]	Output	PIN_K1	2	B2_N1
p2_o[4]	Output			
p2_o[3]	Output	PIN_L4	2	B2_N1
p2_o[2]	Output			
p2_o[1]	Output			
p2_o[0]	Output			
p3_o[7]	Output			
p3_o[6]	Output			
p3_o[5]	Output			
p3_o[4]	Output			
p3_o[3]	Output	PIN_F26	5	B5_N0
p3_o[2]	Output	PIN_N18	5	B5_N0
p3_o[1]	Output			
p3_o[0]	Output			
pin_name1	Output			
pin_name2	Output			
reset_i	Input	PIN_N25	5	B5_N1

Fig.09 Pin assignments

Then we need to connect t

IV. Result

After debugging the program in Keil and compile with the project in Quartus, the sof. File generated is put into the programmer, and connected with the hard device of Altera DE2 via usb-blaster, as shown in Fig.10. We get the final result of weighing function via the display of LCD on Altera DE2 board.



Programmer - C:/Users/nanwenya/Desktop/OOOOOOO/hardfile/embeded_system/embeded_system - embeded_system - [o...

Hardware Setup... USB-Blaster [USB-0] Mode: JTAG Progress: 100% (Successful)

□ Enable real-time ISP to allow background programming (for MAX II and MAX V devices)

File	Device	Checksum	Usercode	Program/ Configure	Verify	Blank-Check	Examine
output_files/embeded_sy...	EP2C35F672	0060487A	0060487A	☑	□	□	□

Fig.10 Programmer

Downloading the program to the Altera DE2 board, if we pressure the piezoelectric transducer, we can read the real-time force we applied from the LCD panel. The data refreshes in every 0.02s.

V. Difficulties & Solutions

In this project, the greatest problem is that the LCD doesn’t work, where the exact program works normally on MC8051single chip board. Considering the initial of LCD, even there is no input signal, the LCD panel should display a sentence of “Welcome to use!” However, in the beginning, there is no respond on the Altera DE2 LCD. It is believed that the problem is the calling of LCD instead of pin assignments. After several trials, a hex. file of TESTLCD is generated for purely testing the LCD. And it works on board, “welcome to use!” is displayed..

When we add the whole program again, after successfully testing the LCD, there is still no respond when we built the whole program into DE2, not even the initialization of LCD. Thus, there are other problems in the program. after several tests, I think the problem is the clk of my DE2 board.

VI. Future Work

Currently, only real-time pressure limited to 5Kg is measured. In the future, piezoelectric transducer with higher Frequency response in usable region can be applied. Thus, greater pressure could be measured. In addition, we can add some functions of “add” “minus” “multiply” “divide” into the program, making the project more than pressure sensor.