A Project on

BATCH WEIGHING SYSTEM

BY

Group Number 80

- 2015A7PS102P - K S Sanjay Srivastav
- 2015A7PS103P - Aman Raj
- 2015A7PS104P - Aadi Jain
- 2015A7PS105P - Shrey Goyal

FOR

Microprocessor And Interfacing Course Assignment

Semester II

Year - 2016-17

ACKNOWLEDGEMENT :

We would like to express our sincere gratitude and deep regards to the faculty of Microprocessor And Interfacing course for their guidance, support and encouragement throughout this course. We would take this occasion to thank our family and friends who were a constant encouragement through the making of this course project.

Problem Statement :

A microprocessor system is to be designed as a batch weighing machine. The system is interfaced to three load cells by means of a 10 bit A/D converter. The conditioned output of the load cells is given by the equation: Vout = K x weight (Kgs.)

Where K is dependent on the property of the sensor.

The system monitors the output of the load cells and finds out the total weight by taking the average of the three values that are sensed by each load cell. This value is displayed on a seven segment display. When this value exceeds 50 kgs, an output port, which is connected to a relay, is switched on to sound an alarm. Design the necessary hardware and software for implementing the above-mentioned task.

Assumptions :

These are the following assumptions which we made implementing the problem statement :

- All load cell are different and hence will give different values of weight
- Weight is equivalent to the input voltage of load cell
- We have used 8 bit AD converter instead of 10 bit AD converter.
- Vout of the load cell is the voltage given by amplifier.

Brief System Description :

Three load cells are used for sensing the weights. The voltage output of the load cells is scaled to 5 volt range with the help of resistance. When the start switch is pressed, the analog voltage value is converted to its digital equivalent by means of an A/D converter **ADC 0808**. This value is then multiplied by the conversion factor which is used to calculate the real value of weight calculated by load cell . The calculated weight is then compared with the limiting value of the weight which is 50 Kg. If the weight is below this limiting value, it is displayed in the seven segment display. If not, an alarm is sounded and the system waits for the user to reset the system using the reset switch.

Chip Number	**Quantity**	**Chip**	**Purpose**
8086	1	Microprocessor	Central Processing Unit
6116	2(1 for even and 1 for odd)	RAM	For data segment and stack which is used for sub-routines
2732	2(1 for even and 1 for odd)	EPROM	Read Only Programmable Memory to house code
8255	1	Programmable peripheral Interface	Provides I/O ports for other devices
74138	1	3:8 Decoder	Decode for 8255 (even though its not required its generally used in any scenarios if other I/O devices may be used)
7-seg mpx2-ca	1	Seven segment Display	Display the output values
ADC 0808	1	Analog to Digital Converter	Converts the analog voltage to its digital equivalent
7447	1	BCD to 7-segment converter	Coverts a BCD value to a value required by 7 –segment Display
74LS245	2	8-bit bi-directional buffers	Buffering of data bus
74LS373	3	8-bit Latches	Latching the address bus

Apart from the above mentioned chips , these components are also interfaced:

1. Two switches
2. One LED –Red
3. Alarm Device
4. Logic gates and resistors
5. 1Hz Clock Generator for 8255
6. A Relay for LED
7. Amplifiers
8. Voltage Sources

Memory Organization:

The system uses 4KB of RAM and 8KB of ROM. Both consist of two chips of 2KB and 4 kb size each. They are organized into odd and even bank to facilitate both byte and word size data transfers.

Random Access Memory:
Starting Address: 02000h
Ending Address: 02fffh

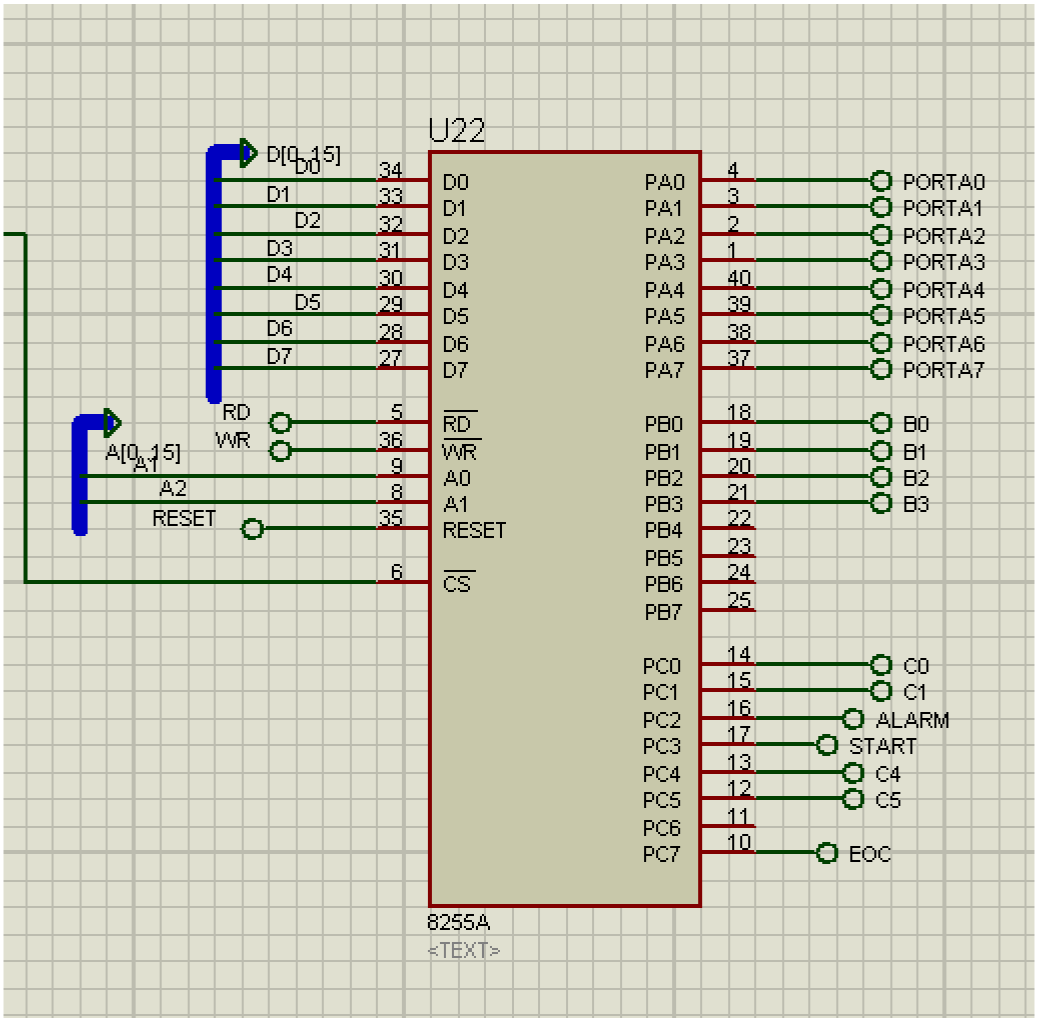
The data segment starts at the address of 02700h and is of size 1KB. The stack segment starts at the address 02810h and is of 1008 bytes.

Read Only Memory:
Starting Address: 00000h
Ending Address: 01fffh

I/O Organization:

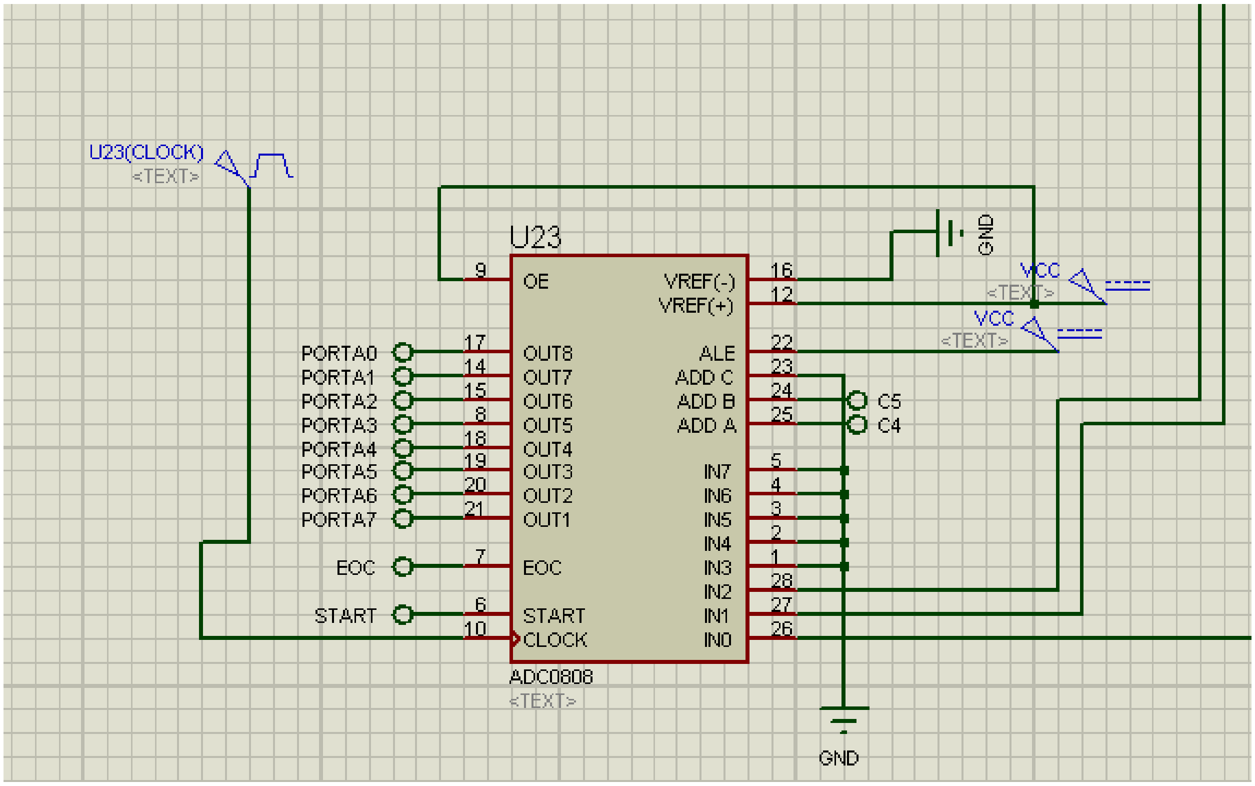
An **8255** is used to communicate with input/output devices . It's interfacing has been shown below :

PORT	PORT ADDRESS	MODE	INPUT/OUTPUT	CONNECTED TO
A	00H	0	Input	ADC
B	02H	0	Output	7447
C Lower	04H		Output	PC0 – 7 Segment Decoder PC1 – 7 Segment Decoder PC2 – Relay for alarm and led PC3 – Start of ADC
C Higher	04H		Output	PC4, PC5- Select lines of ADC's Input(Here these will be used only in BSR-Bitset Reset Mode) PC7 – EOC of ADC
Control Register	06H			


U22
D[0..15]
D0
D1
D2
D3
D4
D5
D6
D7
34
33
32
31
30
29
28
27
RD
WR
A[0..15]
A1
A2
RESET
5
36
9
8
35
6
D0
D1
D2
D3
D4
D5
D6
D7
RD
WR
A0
A1
RESET
CS
PA0
PA1
PA2
PA3
PA4
PA5
PA6
PA7
PB0
PB1
PB2
PB3
PB4
PB5
PB6
PB7
PC0
PC1
PC2
PC3
PC4
PC5
PC6
PC7
4
3
2
1
40
39
38
37
18
19
20
21
22
23
24
25
14
15
16
17
13
12
11
10
PORTA0
PORTA1
PORTA2
PORTA3
PORTA4
PORTA5
PORTA6
PORTA7
B0
B1
B2
B3
C0
C1
ALARM
START
C4
C5
EOC
8255A
<TEXT>

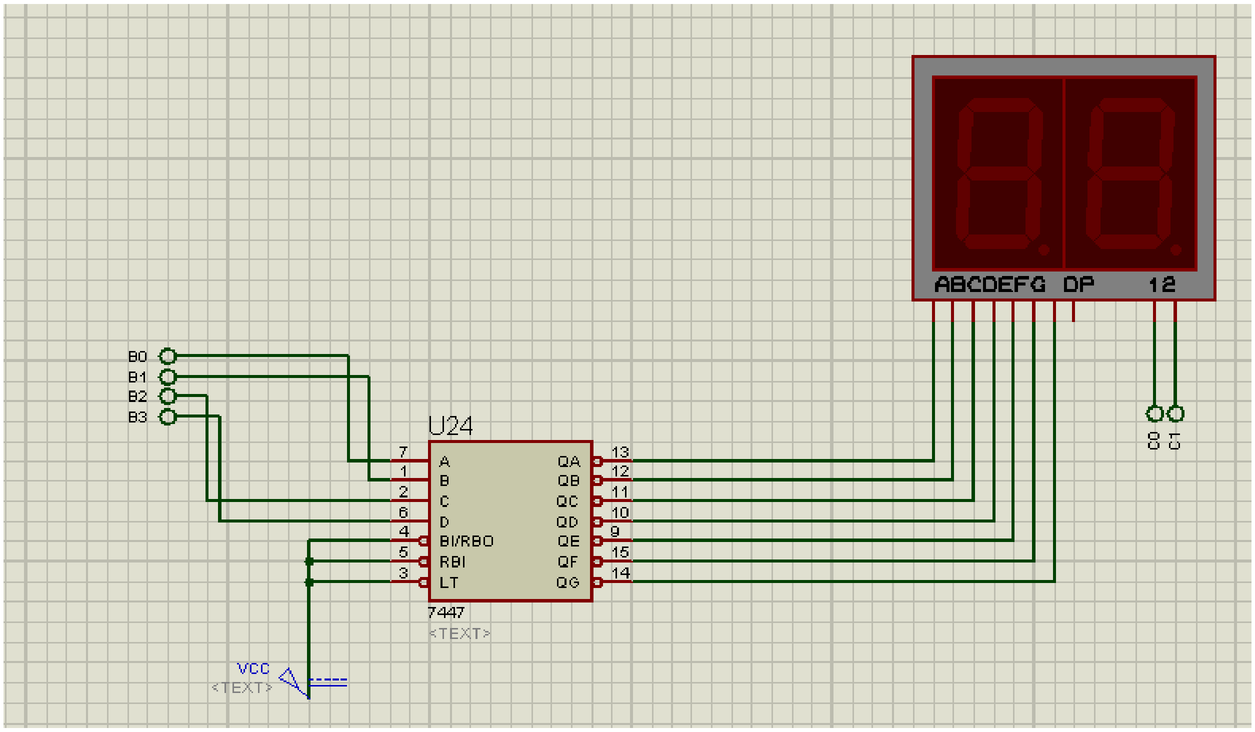
8-BIT Analog to Digital Converter：

ADC 0808 has been used with a clock connected at a pulsed high voltage of 5V and it takes 3 input signals in the form of voltages from 3 load cells as shown below. 2 select lines coming from 8255 are used to select among these 3 signals and thus the output 8-bits go into port A of 8255

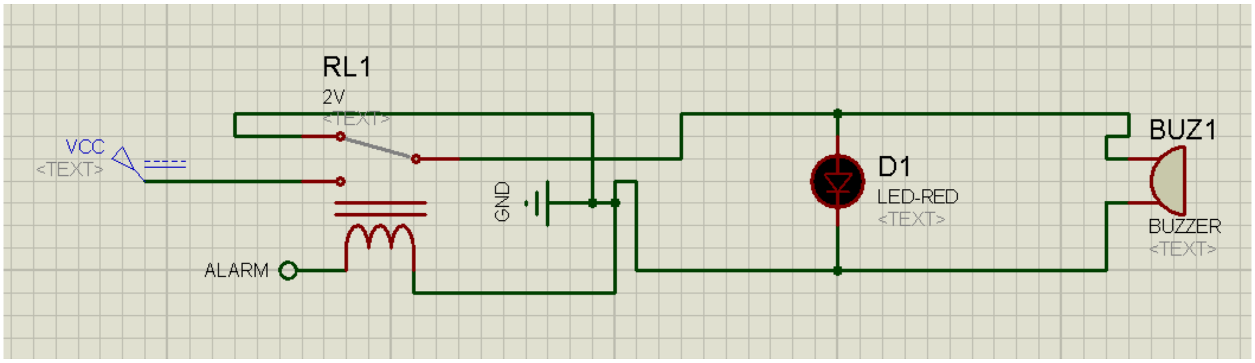


Other Devices:

- Output 7-segment display along with 7447:



- Alarm for testing of heavy weights (weight more than 50kgs) in parallel with an LED :



Limitations:

The main limitations of the system are:

- The system does not automatically check for the new weights. Human intervention is necessary whenever a weight is to be measured.
- Also the 10-bit ADC as asked in the problem statement couldn't be used as the proteus didn't have the model specifications required by it .Therefore an 8-bit ADC has been used instead.
- The system provides an accuracy of up to two integer's digits only.
- The alarm used by us wasn't working well as the sound wasn't audible. Therfore we also used an LED in parallel to it to indicate that there is some potential difference across it.
- If the load on the load cell exceeds, then the load cell may be permanently damaged. So care should be taken that load shouldn't exceed the maximum value.

Conclusion:

The batch weighing system has been implemented as a 8086 based system.

CODE :

```
.model tiny
.data
MLP1       DW 390
; muptiplying individual weights by .39 to convert the range
from ;0-255 to 0-100
MLP2      DW 390
; muptiplying individual weights by .39 to convert the range
from ;0-255 to 0-100
MLP3       DW 390
; muptiplying individual weights by .39 to convert the range
from ;0-255 to 0-100

DIVI      DW 1000

DSDIV     DB 10
;to seperate the unit and tens digits

AVG         DB 03
UNITS      DB ?
TENS      DB ?
WT1       DB ?
WT2       DB ?
```

```
WT3       DB ?
WTAVG     DB ?
porta     equ 00h
portb     equ 02h
portc     equ 04h
creg      equ 06h
stack     dw  100 dup(?)
top_stack  label word

.code
.startup

LEA       SP,top_stack
;initialising stack for the sub routine

MOV       AL,90h
                                        ;port A input
                                        ;port B & C outpu
OUT       creg,AL

;BSR mode to select pins of ADC
;Selecting IN0

MOV       AL,08h
OUT       creg,AL             ;PC4 logic zero
MOV       AL,0ah              ;00001010b
OUT       creg,AL             ;PC5 logic zero
;IN0 of ADC is selected

CALL      delay_soc
```

```
;sending low and high pulses with delay to ADC to convert
;analog to digital

MOV      AL,90h
;10011000b port A input & port B & C output

OUT      creg,AL
;CHECKING FOR END OF CONVERSION
;(whether EOC= LOGIC 1)
;CHECKING STARTS

X1:  IN  AL,portc
; loading in the results of port c pins

AND      AL,80h
; considering only the result of pin 7

JZ       X1


IN       AL,porta          ;dig o/p of ADC is read
MOV      WT1,AL            ;store o/p into mem as weight 1

;Select IN1
MOV      AL,09h
OUT      creg,AL                ;pc4 logic one
MOV      AL,0ah                 ;00001010b
OUT      creg,AL                ;pc5 logic zero
;IN1 is selected

CALL     delay_soc              ;100 ns low to ADC
```

```
MOV AL,90h
;10010000b port A input & port B & C output

OUT creg,AL

X2:  IN AL,portc                    ; reading port c results
AND AL,80h                          ; considering only on 7.
JZ X2
MOV AL,90h
OUT creg,AL
IN AL,porta                         ;dig o/p. of ADC is read
MOV WT2,AL

;store o/p. into mem as weight 2.

;Select IN2

MOV AL,08h
OUT creg,AL                         ;set PC4 logic zero
MOV AL,0bh
OUT creg,AL                         ;set PC5 logic one
;IN2 is selected

CALL delay_soc                      ;100ns low to ADC.
MOV AL,90h
OUT creg,AL


X3:  IN AL,portc                    ; reading port c data
AND AL,80h
; data of only pin 7 is considered.
```

```
JZ X3
MOV AL,90h
OUT creg,AL
IN AL,porta                    ;dig o/p of ADC is read
MOV WT3,AL                          ;Store o/p into mem as weight 3.



;Calculating converted Weight

MOV AH,00h
MOV AL,WT1
; AX register now holds weight 1.

MUL MLP1
DIV DIVI
;changing the range of data to 0-100

MOV WT1,AL
;i/p wt of load1 Moved to WT1

MOV AH,00h
MOV AL,WT2
; AX register now holds weight 3.

MUL MLP2
DIV DIVI
;changing the range of data to 0-100
```

```
MOV WT2,AL
;i/p wt of load2 Moved to WT2

MOV AH,00h
MOV AL,WT3
; AX register now holds weight 3.

MUL MLP3
DIV DIVI
;changing the range of data to 0-100

MOV WT3,AL
;i/p wt of load3 Moved to wt3


; Calculating average of WT1,WT2 and WT3

CLC
MOV ah,00h
MOV bh,00h
MOV Al,WT1
MOV bl,WT2
ADC AX,BX                        ; adding wt1 and wt2
MOV bl,WT3
ADC AX,BX                        ; total wt is  now stored in
ax
DIV AVG
;Avg of 3 wts Moved to AL

MOV WTAVG,AL
;Moving avg to mem in wtavg
```

```
CMP AL,50                           ;Check if AVGWT < 50kg
JB FINAL_DISPLAY
;if avg is below 50 then we have to display it on screen.


BUZZER:
MOV AL,05h
OUT creg,AL                         ;alarm if(load>50kg)
MOV cx,05h
DELAY1:     nop
loop DELAY1
JMP BUZZER
; loop for sounding the buzzer.


;display (wt<50kg)


FINAL_DISPLAY:          MOV AH,00h
MOV AL,WTAVG
DIV DSDIV                   ;Separating two digits of weight
                            ;Storing digits in mem
MOV TENS,AL
MOV UNITS,AH



Y1:         MOV     AL,01h
OUT         creg,AL                 ;(pc0)
;Switch on units digit display
```

```
MOV     AL,90h
OUT     creg,AL
;Set i/o mode to input 7447

MOV     AL,UNITS
OUT     portb,AL
MOV     AL,00h                  ;Switch off units digit
display
OUT     creg,AL


MOV     AL,03h
OUT     creg,AL                 ;(pc1)
;Switch on tens digit display

MOV     AL,90h
OUT     creg,AL
;i/o mode to input 7447

MOV     AL,TENS
OUT     portb,AL
MOV     AL,02h
OUT     creg,AL
;Switch off tens digit display

JMP     Y1



.exit
```

```
delay_soc proc near
MOV     AL,06h
OUT     creg,AL                     ;set pc3 low

MOV cx,05h
DELAY:  nop
loop DELAY

MOV     AL,07h
OUT     creg,AL                 ;set pc3 high


RET
delay_soc endp

end
```