Group No: 94

Microprocessor Programming and Interfacing

Design assignment

**Batch Weighing Machine**

**(Problem No. 8)**

by

Aniruddha Mahajan 2017A7PS0145P

Ravindra Singh Shekhawat 2017A7PS0146P

Deepak Chahar 2017A7PS0147P

Kunal Mohta 2017A7PS0148P

An assignment submitted in

partial fulfilment of the requirement of the course

CS F241: MICROPROCESSOR PROGRAMMING AND INTERFACING



BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE

**PILANI, PILANI CAMPUS**

**RAJASTHAN-333031**

27 APRIL 2019

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:

The following are the assumptions made regarding the system:

* Each load cell gives voltage output of the range 0-10mV. But for simulation, we have taken the value of K = 1 (**Note:** This can be changed).
* If the load cells of different rating are to be used then the voltage range is scaled down or up accordingly.
* We have used potentiometer instead of load cells, in the simulation.
* We have used LOGICSTATE instead of switches, because switches make the simulation very slow.

System Description:

The weight is sensed by three load cells. The load cells we have used have an output range of 0-10mV.

When simulation starts, the analog voltage value already loaded using potentiometer is converted to its digital equivalent by means of an A/D converter (as the 10-bit ADC is not available in Proteus, we have used 8-bit ADC for simulation purpose, but chart paper design has 10-bit ADC). This value is then multiplied by the conversion factor which is dependant on the relation between output voltage and the weight (and on the scaling factor, if present). The process is repeated for each of the three load cells and the average is calculated (up-to two decimal places).

The calculated weight is then compared with the limiting value of the weight which is 50 Kg. If the weight is above this limiting value an alarm is sounded. The weight is then displayed in four seven-segment displays. The integer part is displayed in two of the seven-segment displays and the decimal part of the weights is displayed in the other two displays. Then the system moves to a polling state, where it keeps looking at the stop-alarm switch. If the switch is pressed, the alarm is stopped, but the system still remains in the polling phase.

For further weight reads, user can change the weights using the potentiometers, and turn the read-weight switch ON, which triggers an NMI, restarting the process of weight reading and display.

**Load Cell**:

It is a component used to convert weight into analog voltage. It works on the principle of Wheat-stone bridge, where resistances (strain gauge) are dependent on the strain caused by the weight.

The output voltage is given by the following relation:

Output Voltage = (Excitation Voltage) x (Sensitivity) x (Weight)

Max Weight measurable by load cell

We have used an Excitation Voltage of 5V, the load cell has sensitivity of 2mV/V and maximum measurable weight of 100kg.

Hence, the conversion factor or the multiplier is calculated as follows:

Multiplier = 5 x 0.002 = 0.0001 V/kg.

100

(This is the value of K mentioned in the problem statement)

List of components:

The following is the list of components used in the system:

|  |  |  |  |
| --- | --- | --- | --- |
| **Chip Number** | **Quantity** | **Chip** | **Purpose** |
| 8086 | 1 | Microprocessor | Central Processing Unit |
| 6116 | 2 | RAM | Read Write Memory to house Data segment and Stack segment |
| 2732 | 2 | EPROM | Read Only Erasable Programmable memory to house the code |
| 8255 | 2 | Programmable Peripheral Interface | Provides I/O port for the other devices |
| 7447 | 4 | BCD to 7-segment decoder | Converting BCD to 7-segment code for display |
| 7SEG-MPX1-CA | 4 | Seven Segment Display | Display the output values |
| ADC 0808 | 1 | 8-bit Analog to Digital Converter | Converts the analog voltage to its digital equivalent |
| 74LS245 | 2 | 8-bit bidirectional buffer | Buffering data bus |
| 74LS373 | 3 | 8-bit latches | Demultiplexing and Latching the address bus |

Apart from the above chips, two logic states, one relay, one buzzer, one NPN transistor, three potentiometers and logic gates are used.

Read-weight Switch:

Initially the loaded weight is read as soon as weighing machine is run without any user interference. But for subsequent weight reads, user has to trigger the switch (for optimizing the simulation performance we have used a logic state instead of a switch).

Stop-alarm Switch:

In the case where the weight has exceeded 50 kg, the buzzer is activated using a relay. It is continuously sounded till the Stop-alarm switch is pressed, or a new weight is read (for optimizing the simulation performance we have used a logic state instead of a switch).

Memory Organization:

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

**Read Only Memory:**

Starting Address: 00000h

Ending Address: 01fffh

**Random Access Memory:**

Starting Address: ff000h

Ending Address: fffffh

I/O organization:

Two 8255(Programmable Peripheral Interface) are used to communicate with other input and output devices. It is organized in the following manner.

**8255(1):**

**Base Address** – 00h

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Port** | **Port Address** | **Mode** | **Input/Output** | **Connected to** |
| A  (PA0-PA2) | 00h | 0 | Output | ADC converter  (to ADD A, B, C) |
| B | 02h | 0 | Input | ADC converter (to OUT1-8) |
| C lower  (PC0, PC1) | 04h | - | Input | ADC converter  (PC0 to Start pin, PC1 to ALE pin) |
| C upper  (PC4 only) | 04h | - | Input | ADC converter  (EOC pin) |
| Control Register | 06h | - | - | - |

**8255(2):**

**Base Address** – 10h

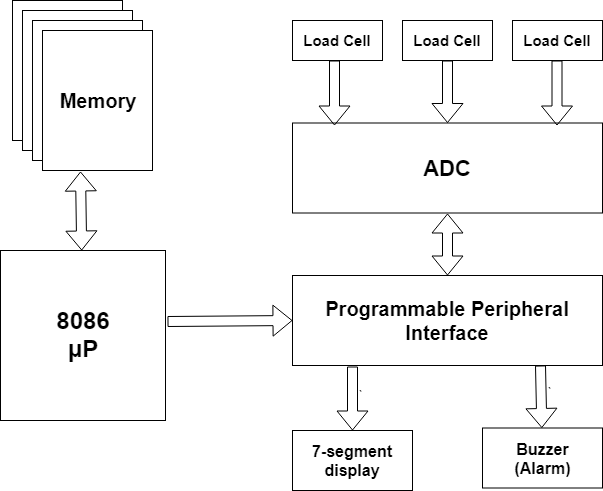
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Port** | **Port Address** | **Mode** | **Input/Output** | **Connected to** |
| A | 10h | 0 | Output | 2\*(BCD to 7seg decoder) |
| B | 12h | 0 | Output | 2\*(BCD to 7seg decoder) |
| C lower (PC0 only) | 14h | - | Output | Buzzer |
| C upper  (PC4 only) | 14h | - | Input | Stop-Alarm switch |
| Control Register | 16h | - | - | - |

IVT Table:

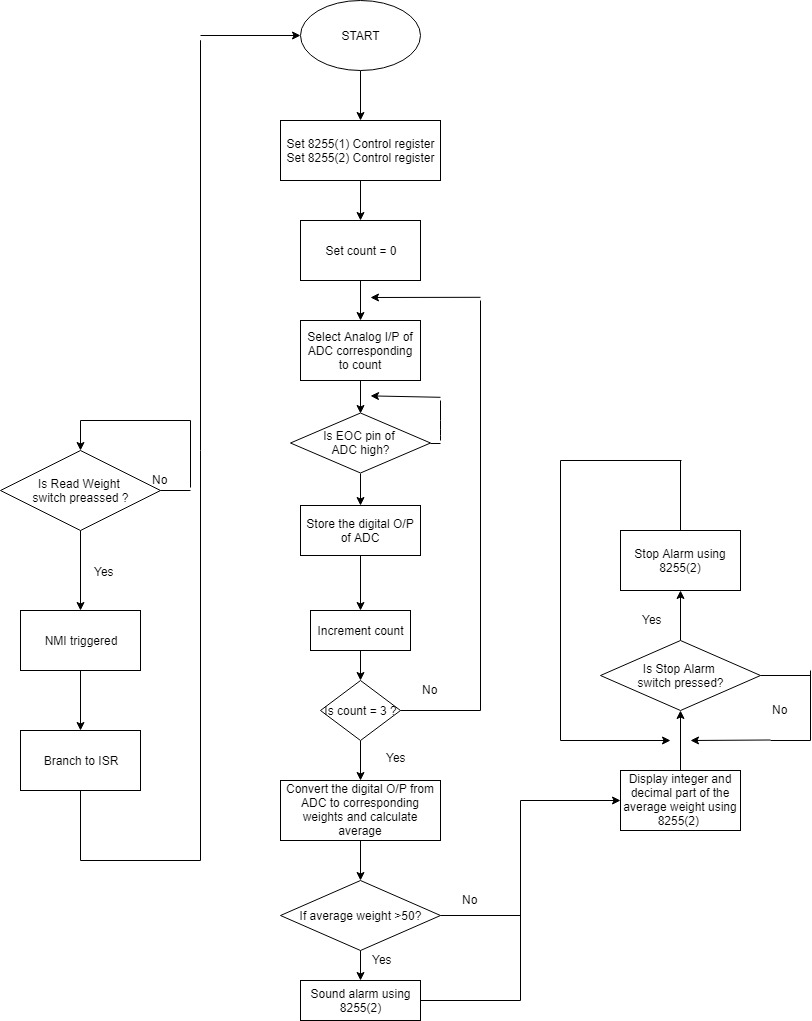
Only Non-maskable Interrupt (NMI) is used, which by default maps to the vector number 02h.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Interrupt Vector** | **Location** | | **Invocation** | **Function** |
| 02h | 00008 | IP value of ISR | Interrupt is invoked when the read-weight switch (or logic state) is pressed to ON (or high) | Runs the weight-reading code sequence |
| 00009 |
| 0000A | CS value of ISR |
| 0000B |

Basic system layout:



Flow Chart:



Limitations:

The main limitations of the system are:

* The system does not automatically check for the new weights. User intervention is necessary whenever a weight is to be measured.
* The system provides an accuracy of up to two decimal digits only.
* The least weight that can be measured is dependant on the rating of the load cell and number of output lines of ADC.
* The alarm cannot be stopped automatically. It can be stopped only by using the Stop-alarm switch.
* Some error occurs for subsequent weight reads after the first one-
  + If the Read-weight switch is pressed without changing weights, then there is some inaccuracy associated with results displayed.
  + Trap debugging revealed that this is because of some issue with OUT instruction for 8255(2), responsible for 7-segment displays.

Code:

#make\_bin#

#LOAD\_SEGMENT=FFFFh#

#LOAD\_OFFSET=0000h#

#CS=0000h#

#IP=0000h#

#DS=0000h#

#ES=0000h#

#SS=0000h#

#SP=FFFEh#

#AX=0000h#

#BX=0000h#

#CX=0000h#

#DX=0000h#

#SI=0000h#

#DI=0000h#

#BP=0000h#

JMP START

; DEFINED DATA

DB 6 DUP(0)

DW NMIS

DW 0000

PORTA1 EQU 00H

PORTB1 EQU 02H

PORTC1 EQU 04H

CREG1 EQU 06H

PORTA2 EQU 10H

PORTB2 EQU 12H

PORTC2 EQU 14H

CREG2 EQU 16H

MAXWEIGHT EQU 100

ANLG DB 00H, 01H, 02H

WEIGHTS DB 3 DUP(?)

START:

; KEEPING ALARM OFF INITIALLY

MOV AL, 10001000B

OUT CREG2, AL

MOV AL, 00H

OUT PORTC2, AL

READWEIGHT:

; PROPERLY INITIALIZING REGISTERS

MOV AX, 0

MOV BX, 0

MOV BP, 0

MOV SI, OFFSET ANLG

MOV DI, OFFSET WEIGHTS

WEIGHTLOOP:

; SETTING CONTROL REGISTER FOR 8255-1

MOV AL, 10001010B

OUT CREG1, AL

; SELECTING ANALOG I/P CORRESPONDING TO THE CURRENT ITERATION

MOV AL, [SI]

OUT PORTA1, AL

INC SI

; HANDLING THE ALE AND START PINS OF ADC

MOV AL, 02H

OUT PORTC1, AL

MOV AL, 01H

OUT PORTC1, AL

; WAITING FOR THE EOC OF ADC TO GO LOW INITIALLY

LEOC:

IN AL, PORTC1

AND AL, 10H

CMP AL, 10H

JZ LEOC

; HANDLING THE ALE AND START PINS OF ADC

MOV AL, 00H

OUT PORTC1, AL

; WATING FOR THE EOC OF ADC TO GO HIGH INDICATING CONVERSION COMPLETION

HEOC:

IN AL, PORTC1

AND AL, 10H

CMP AL, 10H

JNZ HEOC

; GETTING DIGITAL O/P FROM ADC

IN AL, PORTB1

; RECORDING WEIGHT CORRESPONDING TO THE CURRENT ITERATION

MOV [DI], AL

INC DI

INC BP

CMP BP, 3

JNZ WEIGHTLOOP

; TAKING THE AVERAGE OF THE 3 WEIGHTS RECORDED

MOV AX, 0

MOV BX, 0

MOV CX, 0

MOV DX, 0

MOV DI, OFFSET WEIGHTS

MOV BL, [DI]

ADD AX, BX

INC DI

MOV BL, [DI]

ADD AX, BX

INC DI

MOV BL, [DI]

ADD AX, BX

MOV CL, 3

DIV CL

MOV AH, 0

; CONVERTING DIGITAL O/P TO ACTUAL WEIGHT

; AND GETTING INTEGER AND DECIMAL PART OF THE DATA, I.E. AVG WEIGHT

MOV CL, MAXWEIGHT

MUL CL

MOV CX, 100

MUL CX

MOV CX, 256

DIV CX

MOV CL, 100

DIV CL

; NOW REGISTER AH HAS THE DECIMAL VALUE AND REGISTER AL HAS THE VALUE FOR BEFORE DECIMAL

MOV BX, AX

; ALARM IF WEIGHT GREATER THAN 50

CMP AL, 50

JB DISPLAY

MOV AL, 01H

OUT PORTC2, AL

; DISPLAYING WEIGHT ON 7-SEG DISPLAY

DISPLAY:

; DISPLAYING INTEGER PART ON 7-SEG DISPLAY

MOV AX, 0

MOV AL, BL

MOV CL, 10

DIV CL

MOV DL, AL

MOV AL, AH

SHL AL, 4

ADD AL, DL

OUT PORTA2, AL

; DISPLAYING DECIMAL PART ON 7-SEG DISPLAY

MOV AX, 0

MOV AL, BH

MOV CL, 10

DIV CL

MOV DL, AL

MOV AL, AH

SHL AL, 4

ADD AL, DL

OUT PORTB2, AL

; WAITING FOR 'STOP ALARM' SWITCH TO BE PRESSED (POLLING)

READSTOPALARM:

MOV AX, 0

MOV BX, 0

IN AL, PORTC2

MOV BL, AL

AND BL, 10H

CMP BL, 10H

JNZ X

MOV AL, 00H

OUT PORTC2, AL

X:

JMP READSTOPALARM

; ISR OF NMI INTERRUPT - USED TO START READING WEIGHT

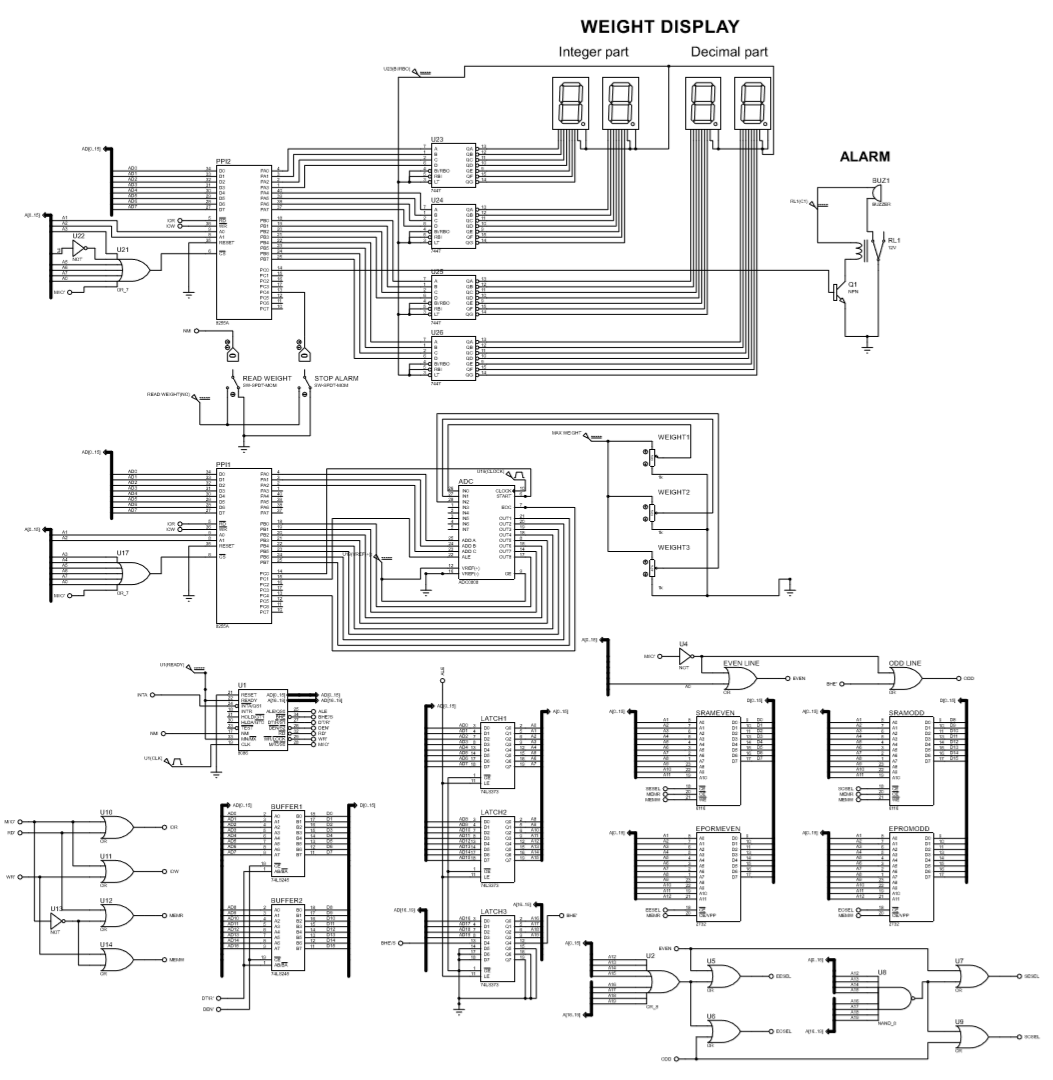
NMIS:

JMP START

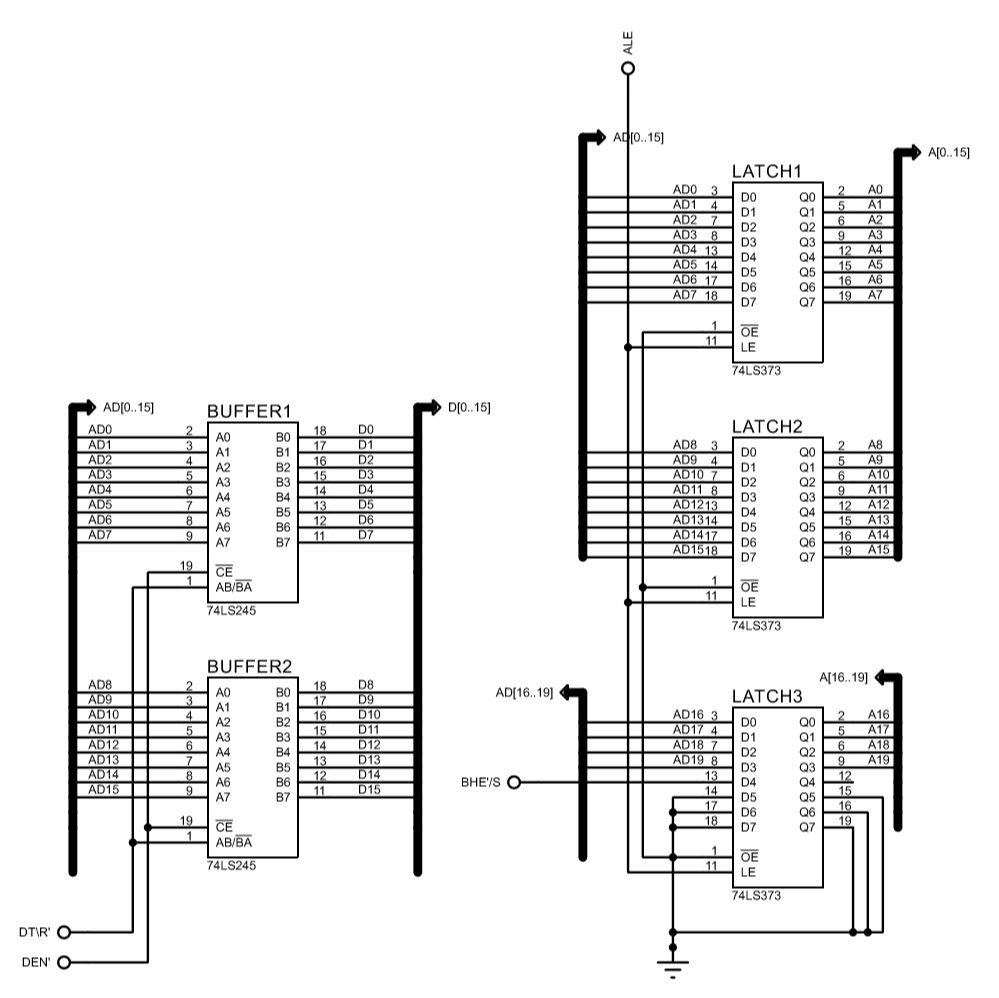
ret

System Design

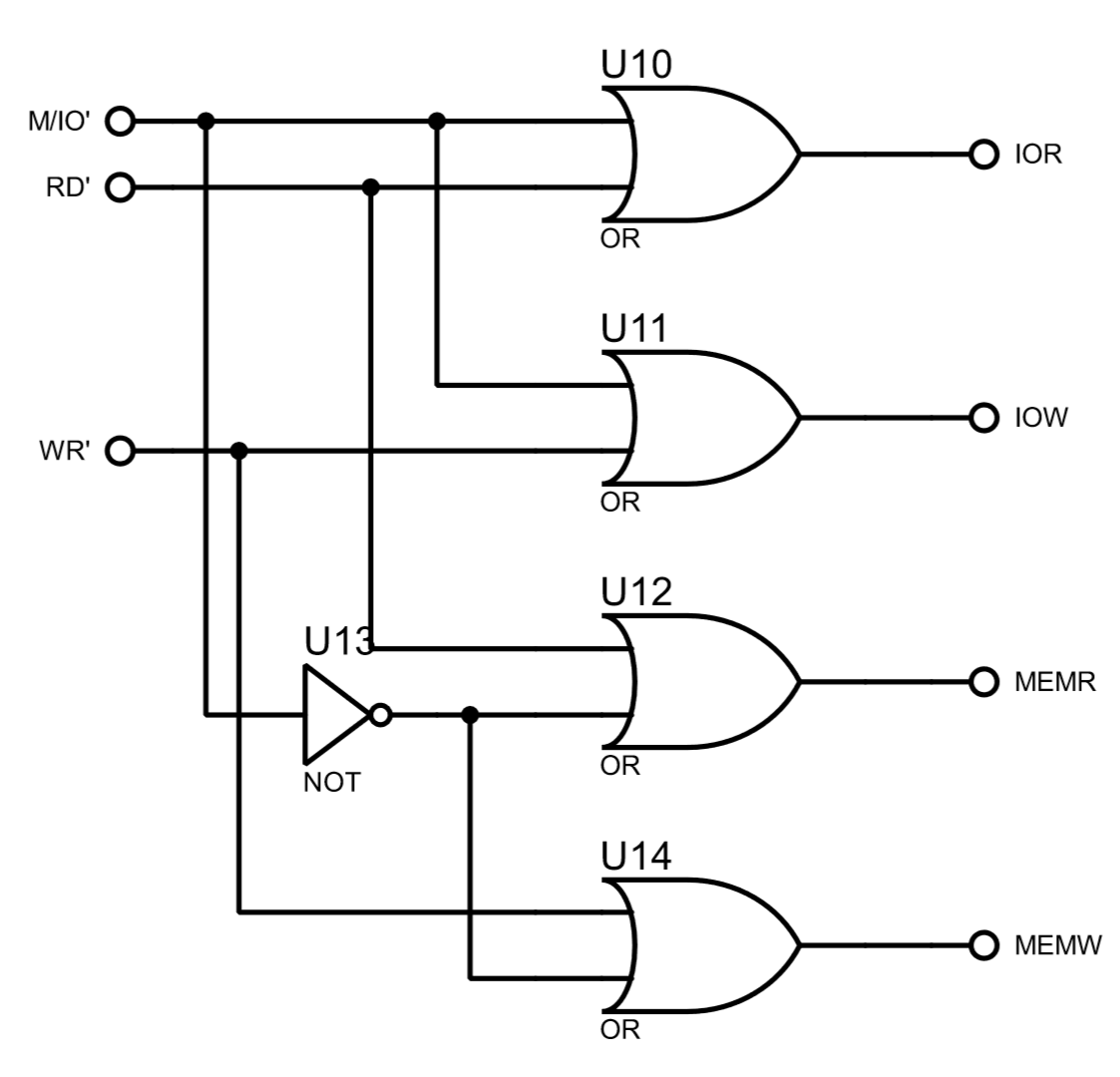
**Complete design**:



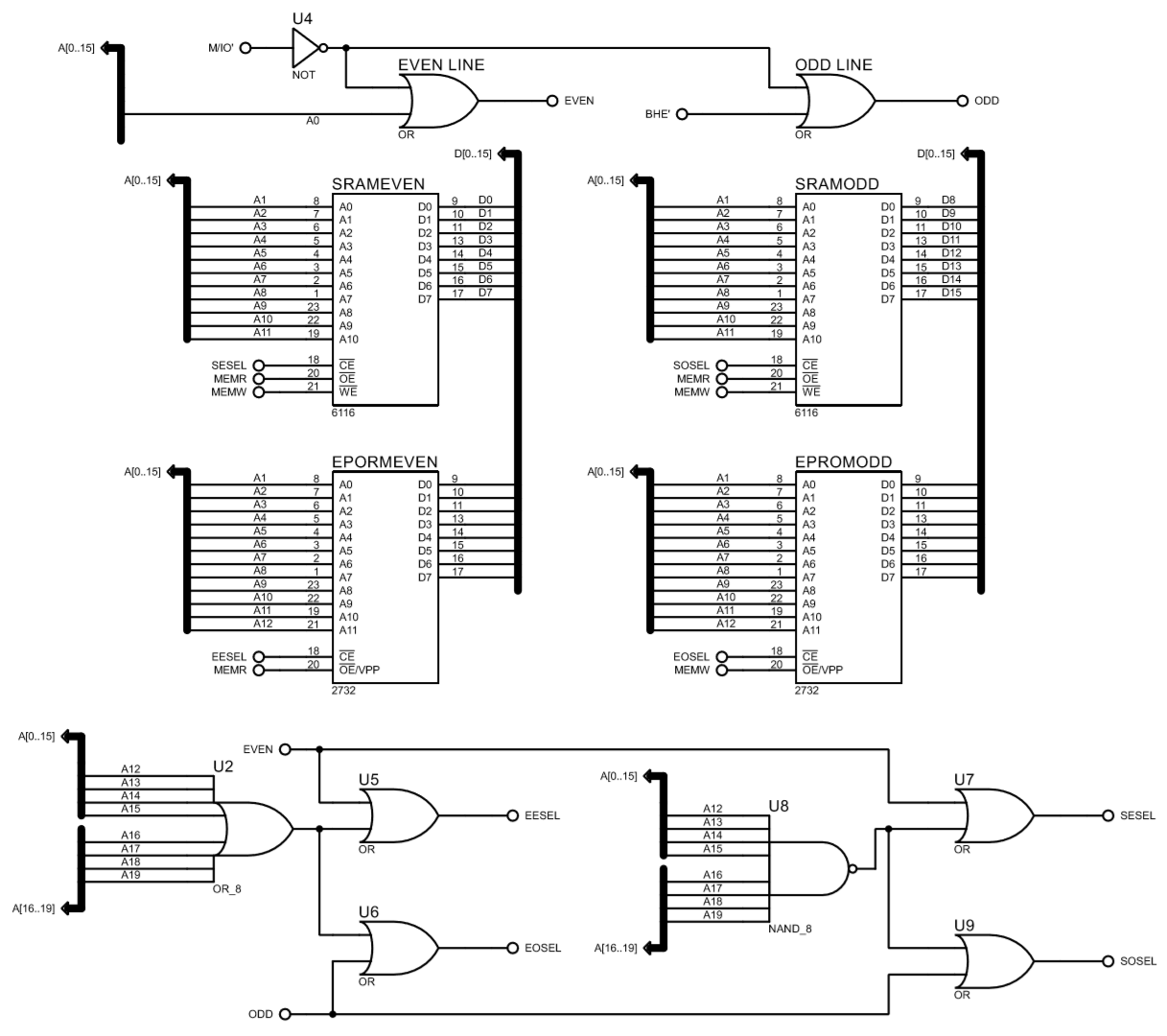
**Demultiplexing Address and Data lines:**



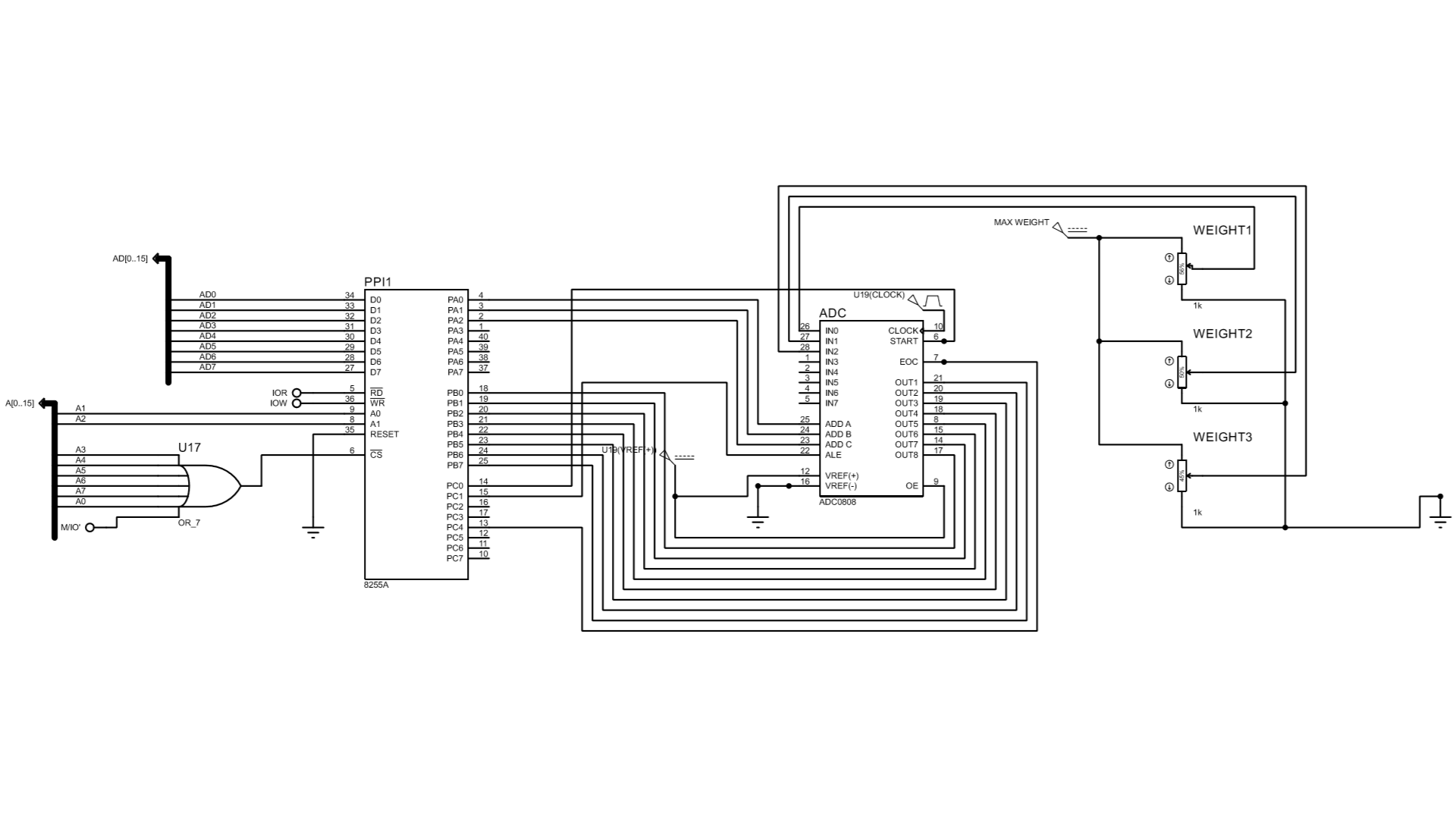
**Control signals:**



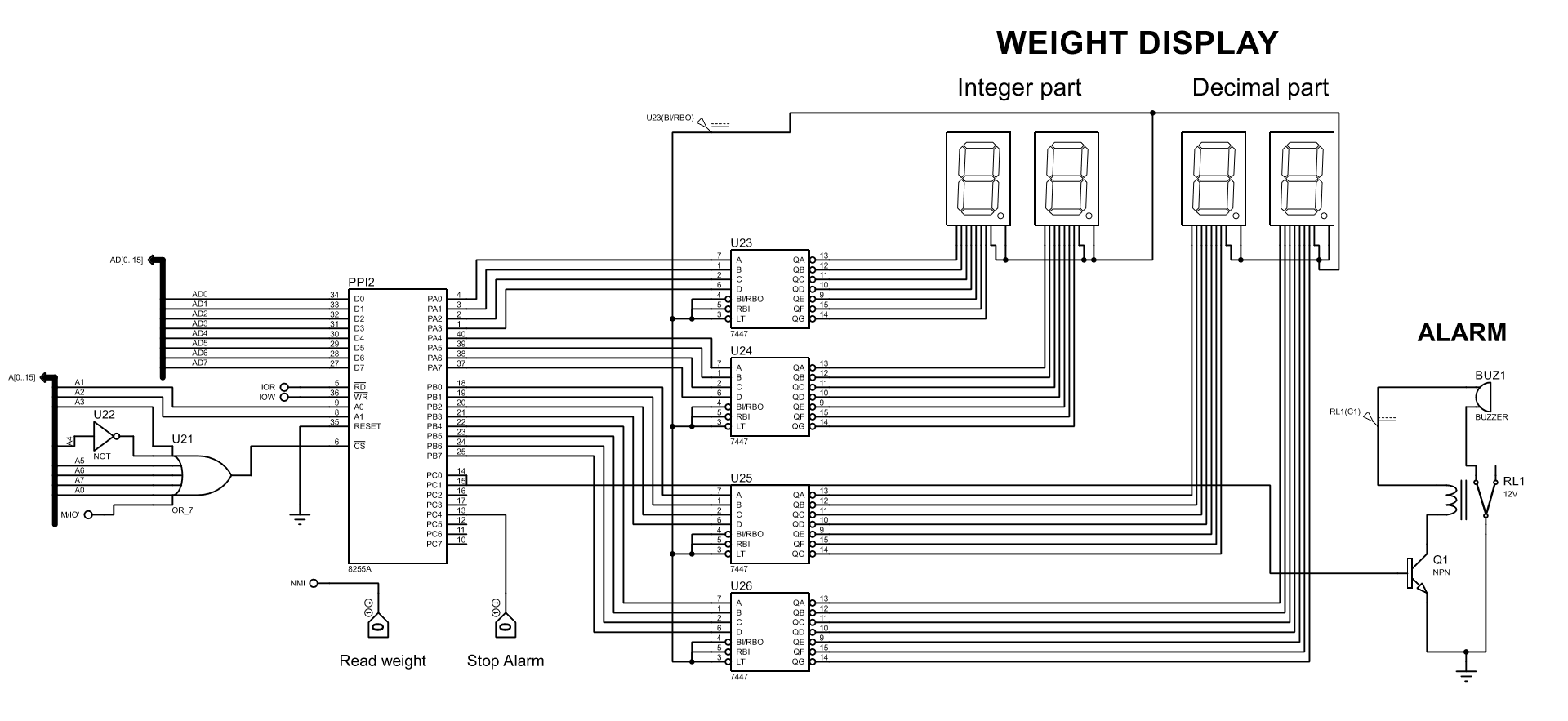
**Memory Interfacing:**



**I/O Interfacing (1):**



**I/O interfacing (2):**



**Note**: The Chart Paper design is different from these designs. The use of 10-bit ADC and Load Cells have been demonstrated in the design on Chart.

References

* **Load Cell**
  + **Model 125 LH (100kg)**
  + <https://sensing.honeywell.com/honeywell-test-and-measurement-model-125-load-cell-produt-sheet-008665-2-en.pdf>
* **8-bit ADC**
  + **ADC 0808**
  + <http://www.ti.com/lit/ds/symlink/adc0808-n.pdf>
* **Clock Generator**
  + **8284**
  + <http://home.etf.rs/~vm/os/mips/razno/datasheets/8284.pdf>