BIRLA INSTITUTE OF SCIENCE AND TECHNOLOGY, PILANI

COURSE TITLE:
MICROPROCESSOR PROGRAMMING AND INTERFACING

DESIGN ASSIGNMENT
CHOCOLATE VENDING MACHINE

Group No. 72   Project No. 17

Rishabh Garg – 2014A7PS065P
Shivam Gupta – 2014A7PS066P
Devansh Patel – 2014A7PS069P
Ankit Sultana – 2014A7PS070P

P17: System to be Designed : Chocolate Vending System

Description: This automatic machine vends three different types of chocolates.

Perk: Rs. 5.00

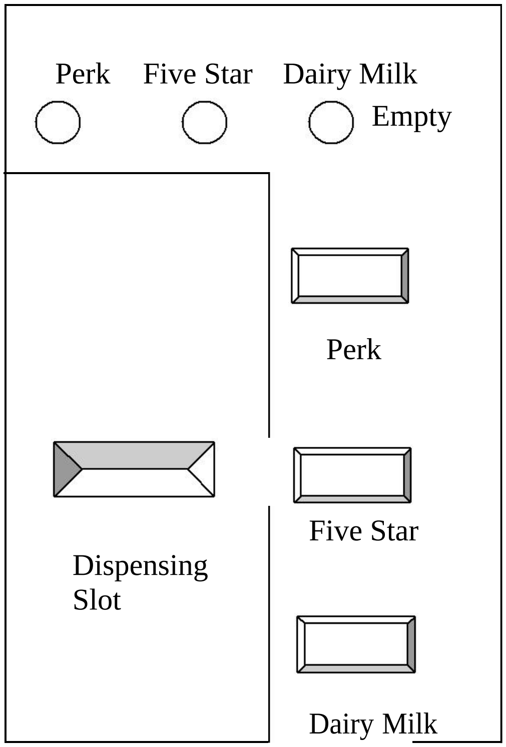
Five-Star: Rs10

Dairy Milk: Rs20.

The currency has to be given in terms of 5 Rupee coin. A weight sensor is used to detect whether the coin is an Rs5 coin or not. There are three buttons available for the selection of the chocolate. After the chocolate has been selected user has to put the correct currency into the coin slot. When the user has dropped the entire amount into the slot, the machine dispenses the correct chocolate. Whenever the chocolate is dispensed successfully, a buzzer is sounded.

LED's are used as indicators to show if any of the chocolates being vended are not available.

User Interface:



System to be designed: Chocolate Vending Machine

System Requirements:

- Coins denomination should be 5.
- System is a vending machine which give chocolates of three types i.e. Dairy Milk, Five Star and Perk.
- The prices of the chocolates are as follows:

  Dairy Milk – Rs. 20
  Five Star – Rs. 10
  Perk – Rs. 5
- User presses the button for chocolate selection and then puts in money (currency in terms of 5 Rupee coin only).

System Specifications:

- 3 LEDs are used to indicate if chocolate is available in the machine. Each LED is of 5 Volt.
- Motor is used to dispense the correct chocolate.
- Motor used is of 12V.
- Pressure Sensor (with conversion factor of 1KPa = 20 mV) is used to sense the pressure of the input coin.
- Analog To Digital Converter is used to digitize the reading taken by pressure sensor . The resolution of the ADC is 5V/256 = 19.53125 mV. Unipolar Stepper Motor is used to serve the purpose of the dispensing slot.

Assumptions:

- Maximum 50 chocolates of each type are available.
- In each transaction, the user can get only one chocolate of a particular type (i.e. dairy milk, perk, five star).

- If the user enters invalid number of coins then the coins are supposed to be returned automatically, the user is supposed to pick them up back and no chocolate comes out.

- At most 4 coins can be put on the machine for Dairy Milk.

  If the user puts more than 4 coins, it would be considered as an invalid transaction and no chocolate would be dispensed.
- The pressure of a 5 rupee coin is 1KPa which gives a 20mV voltage.

Hardware Description:

SR No.	Component	Description	No.
1	8086	Microprocessor	1
2	6116	RAM	2
3	2732	ROM	2
4	8255	Programmable Peripheral Interface	1
5	74LS245	Bidirectional Buffer	2
6	74LS373	Octal Latch	3
7	74154	Decoder IC	1
	74LS138	3:8 Decoder	1
8	ADC0808	Analog to Digital Converter	1
9	OR Gate	2 input	6
10	OR Gate	3 input	3
11	NOT Gate	-	5
12	LED	-	3
	NAND Gate	3 input	1
13	Unipolar Stepper	-	1
14	MPX 4250	Pressure Sensor	1
15	SPST Push Buttons	Buttons	3
16	Buzzer	Using PC sound Card	1
17	AND GATE	2 input	1

Address Mapping

RAM - 2k + 2K = 4K

ROM - 4K + 4K = 8K

Random Access Memory (RAM):-

Starting address – FD000H

Ending address – FDFFFH

Even bank begins at FD000H and ends at FDFFEH

Odd bank begins at FD001H and ends at FDFFFH

Read Only Memory (ROM):-

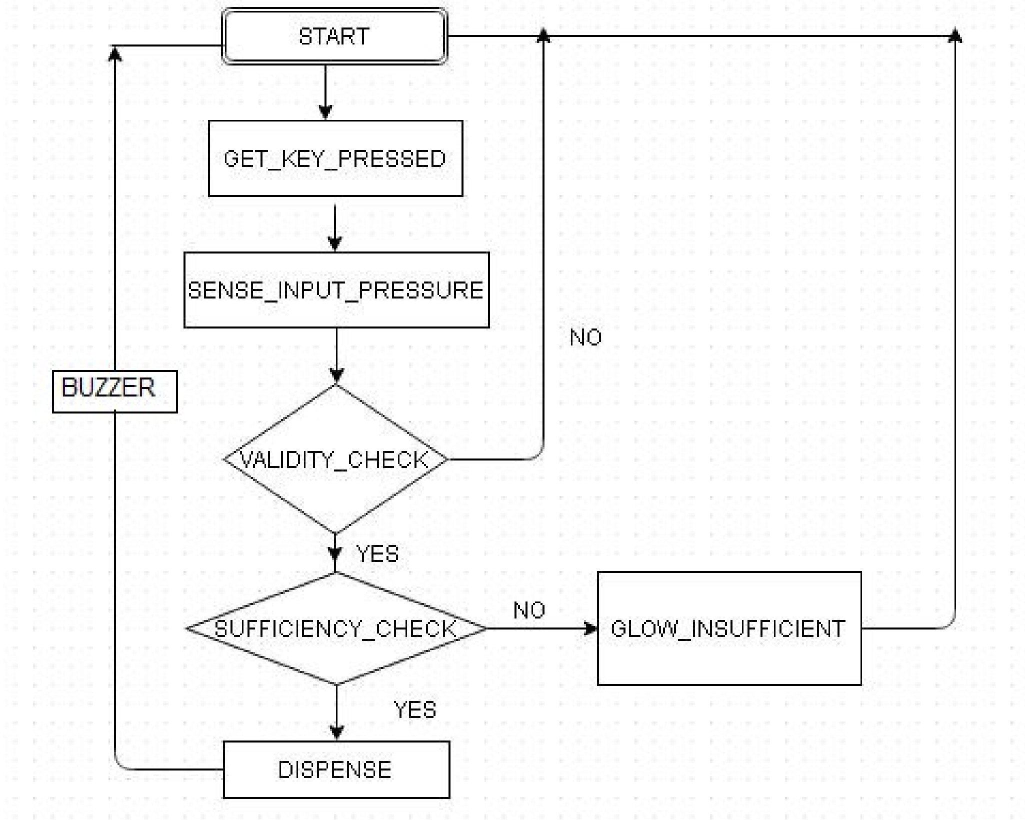
Starting address – FE000H
Ending address – FFFFFH

Even bank begins at FE000H and ends at FFFFEH
Odd bank begins at FE001H and ends at FFFFFH

I/O MAPPING

8255A	PORT	ADDRESS
	A	00h
	B	02h
	C	04h
	Control Register	06h

Flow Chart:



Legend for reading flowchart:

GET_KEY_PRESSED: Gets the key pressed by the user.

SENSE_INPUT_PRESSURE: Converts the input pressure to the number of coins.

VALIDITY_ CHECK: Checks if the number of coins placed on the tray are equal to the one required for the chocolate button pressed.

SUFFICIENCY_CHECK: Checks if the selected chocolate is available in sufficient quantity or not.

GLOW_ INSUFFICIENT: Glow the LED for the corresponding chocolate which is found insufficient.

DISPENSE: Uses motor to dispense the corresponding number of chocolates.

BUZZER: Buzzer is sounded on successful dispense

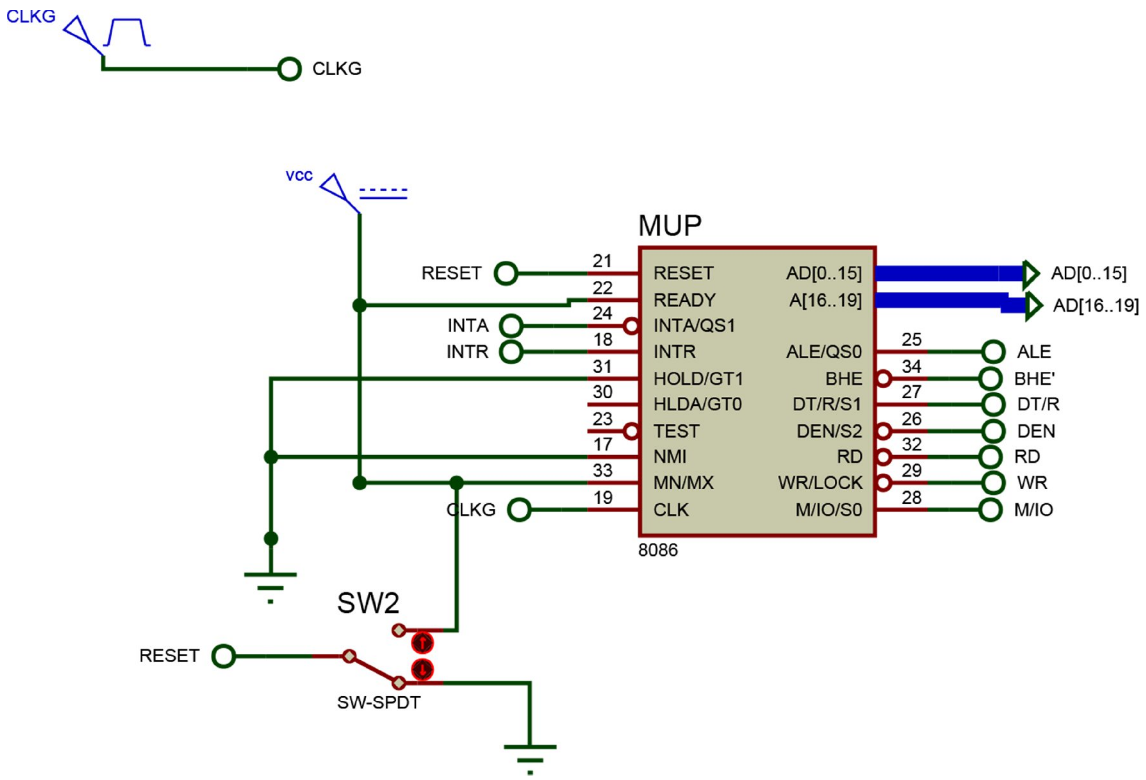
ALP PROGRAM

```
.Model Tiny
.data
ORG  00

; KEYPAD LOOKUP TABLE
KEYPAD_TABLE    DB    060H,050H,030H
KEYPAD_TABLE_LENGTH   EQU   3

; PORT ADDRESSES OF 8255
PORTA EQU 00H
PORTB EQU 02H
PORTC EQU 04H
CTRL_ADDR EQU 06H
IO_MODE EQU 80H

;DELAY AND KEYBOARD VARIABLES
KEYPRESSED DB   ?
DELAY20MSCOUNT  EQU   1000h

; KEY IDs
KEYID_DAIRY_MILK      EQU        1
KEYID_FIVE_STAR       EQU        2
KEYID_PERK            EQU        3

; STACK
STACK1          DW    100 DUP(0)
TOP_STACK1 LABEL WORD

;PRESSURE SENSOR VARIABLES
IS_VALID db ?
NO_OF_COINS db ?

;VALIDITY CONDITION VARIABLES
COINS_FOR_DAIRY_MILK equ 4
COINS_FOR_FIVE_STAR equ 2
COINS_FOR_PERK equ 1
NUM_OF_DAIRY_MILK_LEFT db 50
NUM_OF_FIVE_STAR_LEFT db 50
NUM_OF_PERK_LEFT db 50

;STEPPER MOTOR ROTATION SEQUENCE VARIABLES
STEPPER_MOTOR_SEQUENCE1 EQU 00000100B            ;motor sequence
with PB2=1
STEPPER_MOTOR_SEQUENCE2 EQU 00001000B            ;motor sequence
with PB3=1
STEPPER_MOTOR_SEQUENCE3 EQU 00010000B            ;motor sequence
with PB4=1
STEPPER_MOTOR_SEQUENCE4 EQU 00100000B            ;motor sequence
with PB5=1
```

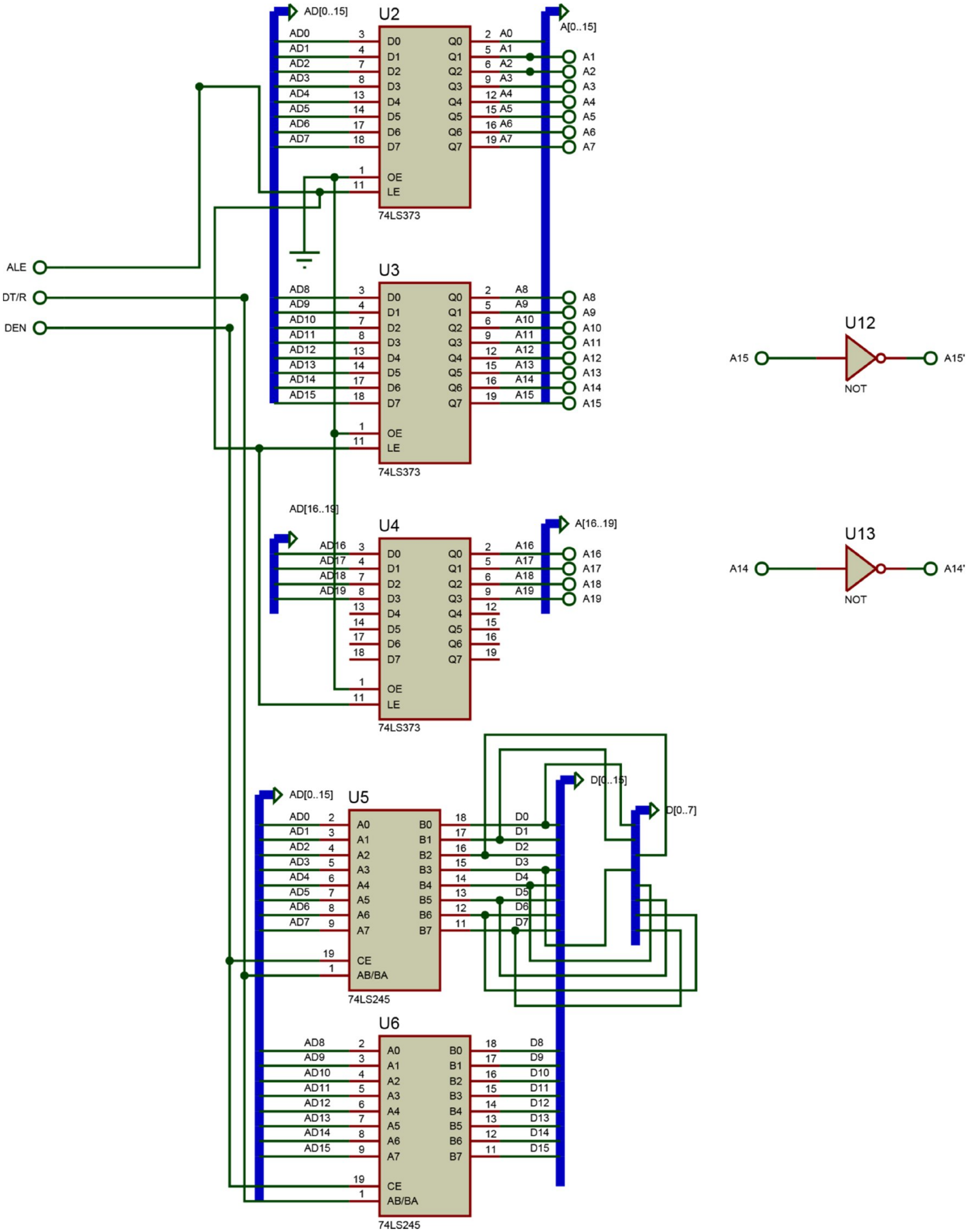
```
.code
.startup
      MAIN1:
                ;set all ports to zero
                CALL GLOW_NOTHING

                ;Get the key pressed in the variable KEYPRESSED
                CALL GET_KEY_PRESSED

                ;Start sensing pressure
                CALL SENSE_INPUT_PRESSURE

                CMP IS_VALID,00h
                JZ MAIN1    ; if yes then discard and start fresh
                                 ; else go to MAIN2 where you see
the key press.

                ;checks if number of coins placed matches with the
coins required for the key pressed
                CALL VALIDITY_CHECK
                CMP IS_VALID,00H
                JZ MAIN1

      DAIRY_MILK:
                CMP KEYPRESSED,KEYID_DAIRY_MILK
                JNZ FIVE_STAR
                CMP NUM_OF_DAIRY_MILK_LEFT,00H
                JZ   GLOW_DM
                SUB NUM_OF_DAIRY_MILK_LEFT,01H
                CALL START_MOTOR_DAIRYMILK
                CALL DELAY_20MS
                CALL SOUND_BUZZER
                JMP MAIN_END

      FIVE_STAR:
                CMP KEYPRESSED,KEYID_FIVE_STAR
                JNZ PERK
                CMP NUM_OF_FIVE_STAR_LEFT,00H
                JZ   GLOW_FS
                SUB NUM_OF_FIVE_STAR_LEFT,01H
                CALL START_MOTOR_FIVESTAR
                CALL DELAY_20MS
                CALL SOUND_BUZZER
                JMP MAIN_END

      PERK:
                CMP KEYPRESSED,KEYID_PERK
                JNZ MAIN_END
                CMP NUM_OF_PERK_LEFT,00H
                JZ   GLOW_PK
                SUB NUM_OF_PERK_LEFT,01H
                CALL START_MOTOR_PERK
                CALL DELAY_20MS
                CALL SOUND_BUZZER
                JMP MAIN_END
                ;if number of chocolates of particular type are 0
then the LED will be switched on

                      GLOW_DM:
                                      CALL GLOW_BLUE
                                      CALL DELAY_20MS
                                      CALL DELAY_20MS
                                      JMP   MAIN_END

                      GLOW_FS:
                                      CALL GLOW_RED
                                      CALL DELAY_20MS
                                      CALL DELAY_20MS
                                      JMP   MAIN_END

                      GLOW_PK:
                                      CALL GLOW_BLUE
                                      CALL DELAY_20MS
                                      CALL DELAY_20MS

     MAIN_END:
                    JMP MAIN1
.exit

GET_KEY_PRESSED PROC NEAR
           PUSHF
           PUSH  AX
           PUSH  BX
           PUSH  CX
           PUSH  DX

           ;Setting 8255 PC lower(0-3) as output and PC upper(4-7)
is input
           MOV AL,10011000b
           OUT CTRL_ADDR,AL
           ;check for key release
           XXX0:
                      MOV AL,01110000b
                      OUT PORTC,AL
           ; Checking if all keys are released
           XXX1: IN  AL,PORTC
                      AND AL,70h
                      CMP AL,70h
                      JNZ XXX1
                      CALL DELAY_20MS

                      MOV AL,01110000b
                      OUT PORTC,AL
           ; checking for key pressed
           XXX2:      IN AL,PORTC
                           AND AL,70h
                           CMP AL,70h
                           JZ XXX2
                           CALL DELAY_20MS
                           ; decoding key pressed
                           MOV AL,01110000b
                           OUT PORTC,AL
                           IN AL,PORTC
                           AND AL,70h
                           CMP AL,70h
                           JZ XXX2
                           CALL DELAY_20MS

           XXX3:
                           CMP AL,KEYPAD_TABLE[0]
                           JNZ XXX4
                           MOV KEYPRESSED,KEYID_DAIRY_MILK
                           JMP GET_KEY_PRESSED_END
           XXX4:
                           CMP AL,KEYPAD_TABLE[1]
                           JNZ XXX5
                           MOV KEYPRESSED,KEYID_FIVE_STAR
                           JMP GET_KEY_PRESSED_END
           XXX5:
                           CMP AL,KEYPAD_TABLE[2]
                           JNZ GET_KEY_PRESSED_END
                           MOV KEYPRESSED,KEYID_PERK
                           JMP GET_KEY_PRESSED_END
           GET_KEY_PRESSED_END:
           POP   DX
           POP   CX
           POP   BX
           POP   AX
           POPF
           RET
GET_KEY_PRESSED ENDP

SENSE_INPUT_PRESSURE PROC NEAR
                PUSHF
                PUSH  AX
                PUSH  BX
                PUSH  CX
                PUSH  DX
                ;setting PORTC upper(4-7) as input and PORTC
lower(0-3) as output,PORTA as input,PORTB as output
                MOV AL,10011000B
                OUT CTRL_ADDR,AL
                ;Selecting input 0 connected to ADC
                MOV AL,00H
                OUT PORTB,AL
                ;Giving start of conversion signal using PC3
                MOV AL,00000110B
                OUT CTRL_ADDR,AL
                MOV AL,00000111B
                OUT CTRL_ADDR,AL
                MOV AL,00000110B
                OUT CTRL_ADDR,AL

                ;waiting for end of conversion signal
     WAITT:     IN AL,PORTC
                ROL AL,01H
                JNC WAITT
                IN AL,PORTA

                ;setting number of coins placed on the pressure sensor
                CMP AL,04H
                JNZ XX1
                MOV IS_VALID,1H
                MOV NO_OF_COINS,04H
                JMP PRESSURE_FINISH

     XX1:  CMP AL,02H
                JNZ XX2
                MOV IS_VALID,1H
                MOV NO_OF_COINS,02H
                JMP PRESSURE_FINISH

     XX2:  CMP AL,01H
                JNZ XX3
                MOV IS_VALID,1H
                MOV NO_OF_COINS,01H
                JMP PRESSURE_FINISH

                ;if more than 4 coins are placed or no coin is placed, is_valid will be 0
     XX3:  MOV IS_VALID,00H
                MOV NO_OF_COINS,00H
                JMP PRESSURE_FINISH

     PRESSURE_FINISH:
           POP  DX
           POP  CX
           POP  BX
           POP  AX
           POPF
           RET
SENSE_INPUT_PRESSURE ENDP

VALIDITY_CHECK PROC NEAR
           PUSHF
           PUSH AX
           PUSH BX
           PUSH CX
           PUSH DX

           MOV IS_VALID,00H

           DAIRY_MILK_PRESSED:
                CMP KEYPRESSED,KEYID_DAIRY_MILK
                JNZ FIVE_STAR_PRESSED
                CMP NO_OF_COINS,COINS_FOR_DAIRY_MILK
                JNZ  ENDING
                MOV IS_VALID,01H
                JMP ENDING


           FIVE_STAR_PRESSED:
                CMP KEYPRESSED,KEYID_FIVE_STAR
                JNZ PERK_PRESSED
                CMP NO_OF_COINS,COINS_FOR_FIVE_STAR
                JNZ  ENDING
                MOV IS_VALID,01H
                JMP ENDING

           PERK_PRESSED:
                CMP KEYPRESSED,KEYID_PERK
                JNZ ENDING
                CMP NO_OF_COINS,COINS_FOR_PERK
                JNZ  ENDING
                MOV IS_VALID,01H

           ENDING:
           POP  DX
           POP  CX
           POP  BX
           POP  AX
           POPF
           RET
VALIDITY_CHECK ENDP

SOUND_BUZZER PROC NEAR
           PUSHF
           PUSH AX
           PUSH BX
           PUSH CX
           PUSH DX

           ;here port 42h,43h,61h are the defaut pc sound card ports
           MOV     DX,5         ; Number of times to repeat whole
routine.
           MOV     BX,1             ; Frequency value.
           MOV     AL, 10110110B
           OUT     43H, AL
           MOV     AX, BX           ; Move our Frequency value in AX
           OUT     42H, AL          ; Send LSB to port 42H.
           MOV     AL, AH           ; Move MSB into AL
           OUT     42H, AL          ; Send MSB to port 42H.
           IN      AL, 61H          ; Get current value of port 61H.
           OR      AL, 00000011B    ; OR AL to this value, forcing
first two bits high.
           OUT     61H, AL          ; Copy it to port 61H of the PPI
Chip
           MOV     CX, 100          ; Repeat loop 100 times

           DELAY_LOOP:              ; Here is where we loop back
too.

           LOOP    DELAY_LOOP       ; Jump repeatedly to DELAY_LOOP
until CX = 0
           INC     BX              ; Incrementing the value of BX
lowers
                                      ; whole routine
           DEC     DX              ; Decrement repeat routine count
           CMP     DX, 0            ; Is DX (repeat count) = to 0
           JNZ     NEXT_FREQUENCY   ; If not jump to NEXT_FREQUENCY
                                      ; and do whole routine again.
           ; Else DX = 0 time to turn speaker OFF
           IN      AL,61H
           AND     AL,11111100B
           OUT     61H,AL
           POP  DX
           POP  CX
           POP  BX
           POP  AX
           POPF
           RET
SOUND_BUZZER ENDP

GLOW_NOTHING PROC NEAR
           PUSHF
           PUSH AX
           PUSH BX
           PUSH CX
           PUSH DX

           MOV AL,80h
           OUT CTRL_ADDR,AL
           MOV AL,00000000b
           OUT PORTC,AL

           POP  DX
           POP  CX
           POP  BX
           POP  AX
           POPF
           RET
GLOW_NOTHING ENDP

GLOW_BLUE PROC NEAR
           PUSHF
           PUSH AX
           PUSH BX
           PUSH CX
           PUSH DX

           MOV AL,10000000b
           OUT CTRL_ADDR,AL
      ;SET PC1 TO 0 AND PC0 TO 0
           MOV AL,00000000b
           OUT CTRL_ADDR,AL
           MOV AL,00000010b
           OUT CTRL_ADDR,AL
           MOV AL,00000101b
           OUT CTRL_ADDR,AL
           POP  DX
           POP  CX
           POP  BX
           POP  AX
           POPF
           RET
GLOW_BLUE ENDP

GLOW_GREEN PROC NEAR
           PUSHF
           PUSH  AX
           PUSH  BX
           PUSH  CX
           PUSH  DX
           MOV AL,10000000b
           OUT CTRL_ADDR,AL
      ;SET PC1 TO 0 AND PC0 TO 1
           MOV AL,00000001b
           OUT CTRL_ADDR,AL
           MOV AL,00000010b
           OUT CTRL_ADDR,AL
           MOV AL,00000101b
           OUT CTRL_ADDR,AL
           POP   DX
           POP   CX
           POP   BX
           POP   AX
           POPF
           RET
GLOW_GREEN ENDP

GLOW_RED PROC NEAR
           PUSHF
           PUSH  AX
           PUSH  BX
           PUSH  CX
           PUSH  DX
           MOV AL,10000000b
           OUT CTRL_ADDR,AL
      ;SET PC1 TO 1 AND PC0 TO 0
           MOV AL,00000000b
           OUT CTRL_ADDR,AL
           MOV AL,00000011b
           OUT CTRL_ADDR,AL
           MOV AL,00000101b
           OUT CTRL_ADDR,AL
           POP   DX
           POP   CX
           POP   BX
           POP   AX
           POPF
           RET
GLOW_RED ENDP

START_MOTOR_PERK PROC NEAR

           PUSHF
           PUSH  AX
           PUSH  BX
           PUSH  CX
           PUSH  DX
           CALL STEPPER_MOTOR_OPEN
           CALL DELAY_20MS
           CALL STEPPER_MOTOR_CLOSE

           POP   DX
           POP   CX
           POP   BX
           POP   AX
           POPF

           RET
START_MOTOR_PERK ENDP

START_MOTOR_FIVESTAR PROC NEAR

           PUSHF
           PUSH  AX
           PUSH  BX
           PUSH  CX
           PUSH  DX

           MOV CX,02H
           rotate:
           CALL STEPPER_MOTOR_OPEN
           CALL DELAY_20MS
           loop rotate

           MOV CX,02H
           rotate2:
           CALL STEPPER_MOTOR_CLOSE
           CALL DELAY_20MS
           loop rotate2

           POP   DX
           POP   CX
           POP   BX
           POP   AX
           POPF

           RET
START_MOTOR_FIVESTAR ENDP

START_MOTOR_DAIRYMILK PROC NEAR

           PUSHF
           PUSH  AX
           PUSH  BX
           PUSH  CX
           PUSH  DX

           MOV CX,04H
           rotate3:
           CALL STEPPER_MOTOR_OPEN
           CALL DELAY_20MS
           loop rotate3
           MOV CX,04H
           rotate4:
           CALL STEPPER_MOTOR_CLOSE
           CALL DELAY_20MS
           loop rotate4

           POP   DX
           POP   CX
           POP   BX
           POP   AX
           POPF

           RET
START_MOTOR_DAIRYMILK ENDP

STEPPER_MOTOR_OPEN PROC NEAR
;give the sequence to stepper motor such that at a time one input is
1,others are 0.
;clockwise rotation is taken as opening of motor slot.

           PUSHF
           PUSH  AX
           PUSH  BX
           PUSH  CX
           PUSH  DX

           MOV AL,10011000B           ;setting PORTC upper(4-7) as
input and PORTC lower(0-3) as output,PORTA as input,PORTB as output
           OUT CTRL_ADDR,AL
           ;to disable the decoder putting PC2=0
           IN AL,PORTC
           MOV DL,AL
           MOV BL,DL
           AND BL,11111011B
           MOV AL,BL
           OUT PORTC,AL

           MOV AL,STEPPER_MOTOR_SEQUENCE1
           OUT PORTB,AL
           CALL DELAY_20MS

           MOV AL,STEPPER_MOTOR_SEQUENCE2
           OUT PORTB,AL
           CALL DELAY_20MS

           MOV AL,STEPPER_MOTOR_SEQUENCE3
           OUT PORTB,AL
           CALL DELAY_20MS

           MOV AL,STEPPER_MOTOR_SEQUENCE4
           OUT PORTB,AL
           CALL DELAY_20MS


           ;restore state of PORTC
           MOV AL,10011000B           ;setting PORTC upper(4-7) as
input and PORTC lower(0-3) as output,PORTA as input,PORTB as output
           OUT CTRL_ADDR,AL
           MOV AL,DL
           OUT PORTC,AL

           POP   DX
           POP   CX
           POP   BX
           POP   AX
           POPF

           RET

STEPPER_MOTOR_OPEN ENDP

STEPPER_MOTOR_CLOSE PROC NEAR
;give the sequence to stepper motor such that at a time one input is
1,others are 0.
;anti-clockwise rotation is taken as closing of motor slot.

           PUSHF
           PUSH  AX
           PUSH  BX
           PUSH  CX
           PUSH  DX

           MOV AL,10011000B           ;setting PORTC upper(4-7) as
input and PORTC lower(0-3) as output,PORTA as input,PORTB as output
           OUT CTRL_ADDR,AL
           ;to disable the decoder putting PC2=0
           IN AL,PORTC
           MOV DL,AL
           MOV BL,DL
           AND BL,11111011B
           MOV AL,BL
           OUT PORTC,AL

           MOV AL,STEPPER_MOTOR_SEQUENCE1
           OUT PORTB,AL
           CALL DELAY_20MS

           MOV AL,STEPPER_MOTOR_SEQUENCE4
           OUT PORTB,AL
           CALL DELAY_20MS

           MOV AL,STEPPER_MOTOR_SEQUENCE3
           OUT PORTB,AL
           CALL DELAY_20MS

           MOV AL,STEPPER_MOTOR_SEQUENCE2
           OUT PORTB,AL
           CALL DELAY_20MS

           ;restore state of PORTC
           MOV AL,10011000B           ;setting PORTC upper(4-7) as
input and PORTC lower(0-3) as output,PORTA as input,PORTB as output
           OUT CTRL_ADDR,AL
           MOV AL,DL
           OUT PORTC,AL

           POP  DX
           POP  CX
           POP  BX
           POP  AX
           POPF

           RET

STEPPER_MOTOR_CLOSE ENDP

DELAY_20MS PROC NEAR
           PUSHF
           PUSH AX
           PUSH BX
           PUSH CX
           PUSH DX

           MOV  CX, 1000h                             ; MACHINE
CYCLES COUNT FOR 20ms
     DELAYLOOP: NOP
           NOP
           NOP
           NOP
           NOP
           LOOP DELAYLOOP

           POP  DX
           POP  CX
           POP  BX
           POP  AX
           POPF
           RET
DELAY_20MS ENDP

End
```

HARDWARE DESIGN

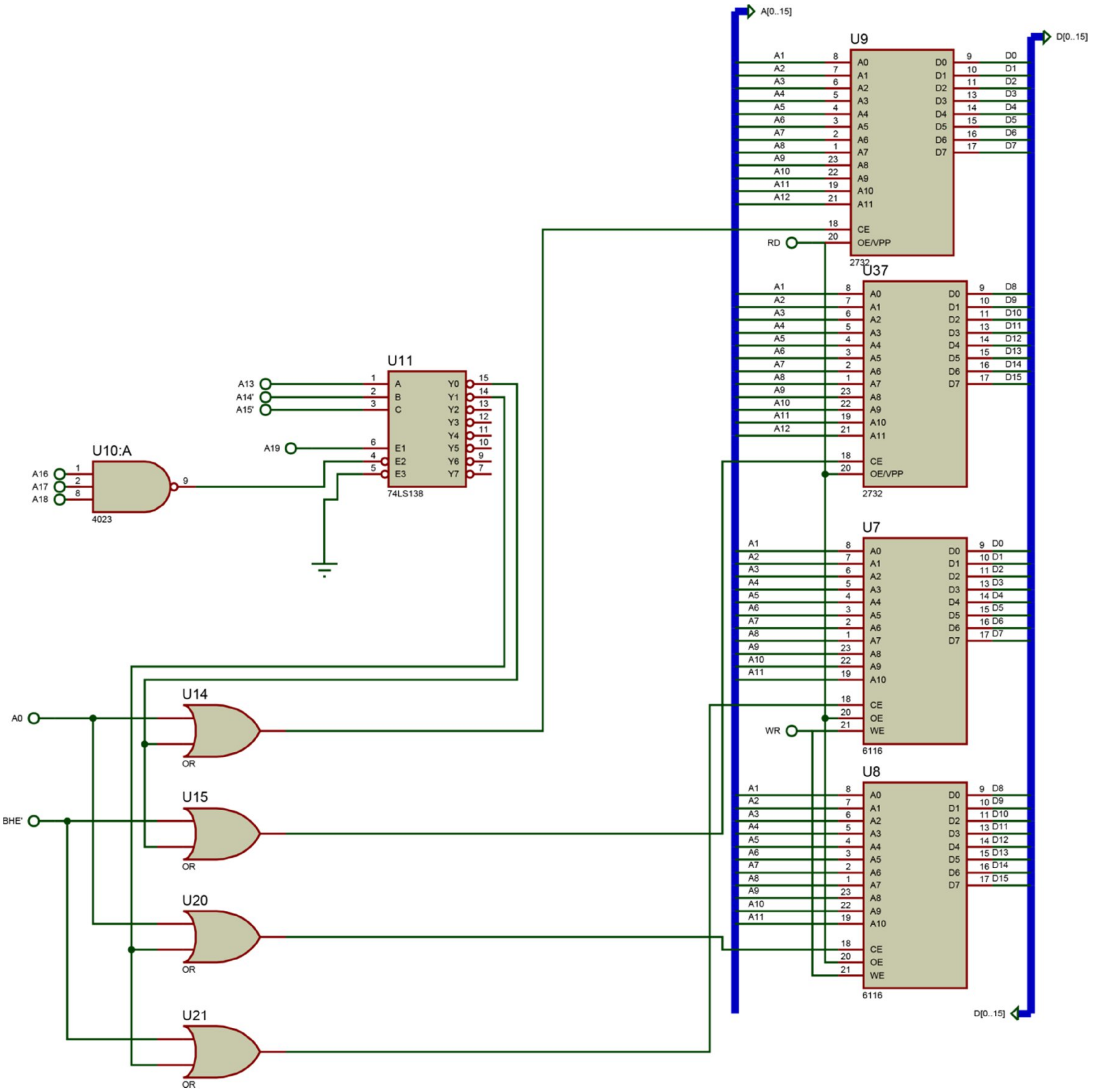
MICROPROCESSOR 8086



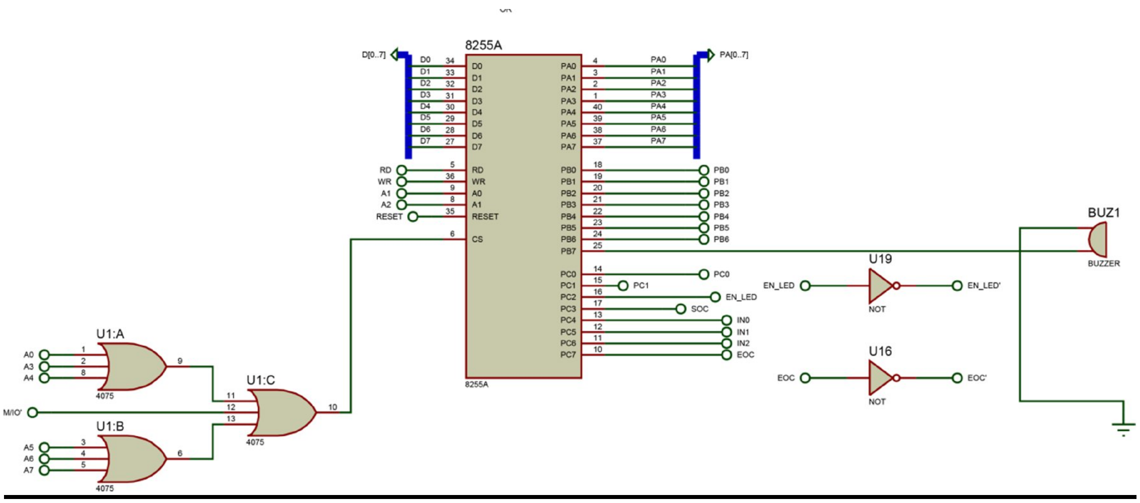
ADDRESS AND DATALINES DEMULTIPLEXING



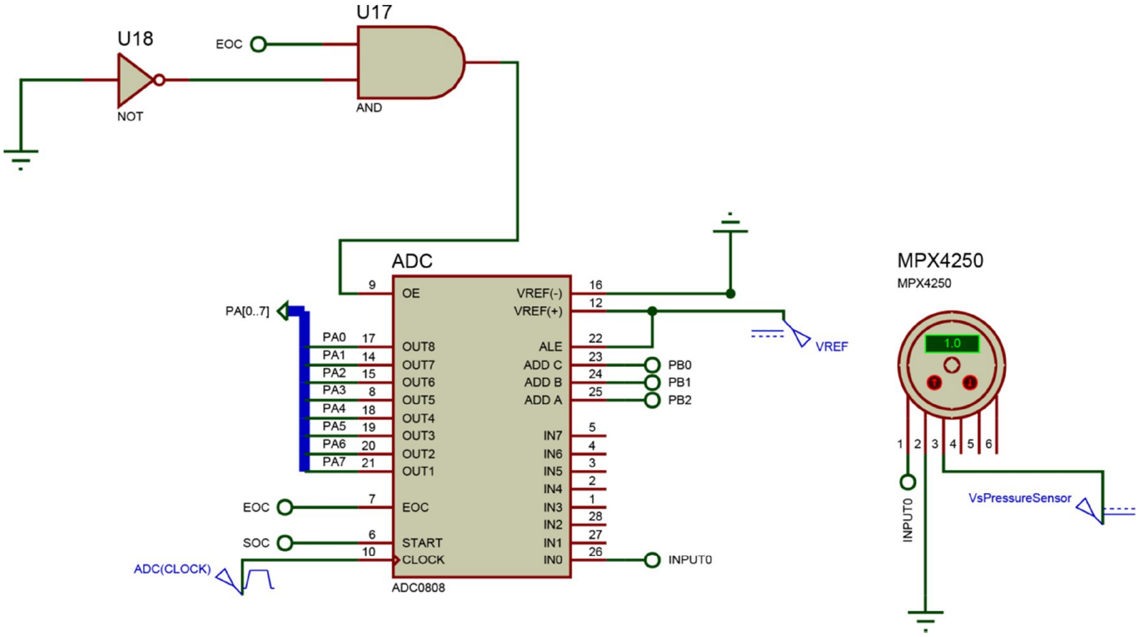
MEMORY INTERFACING



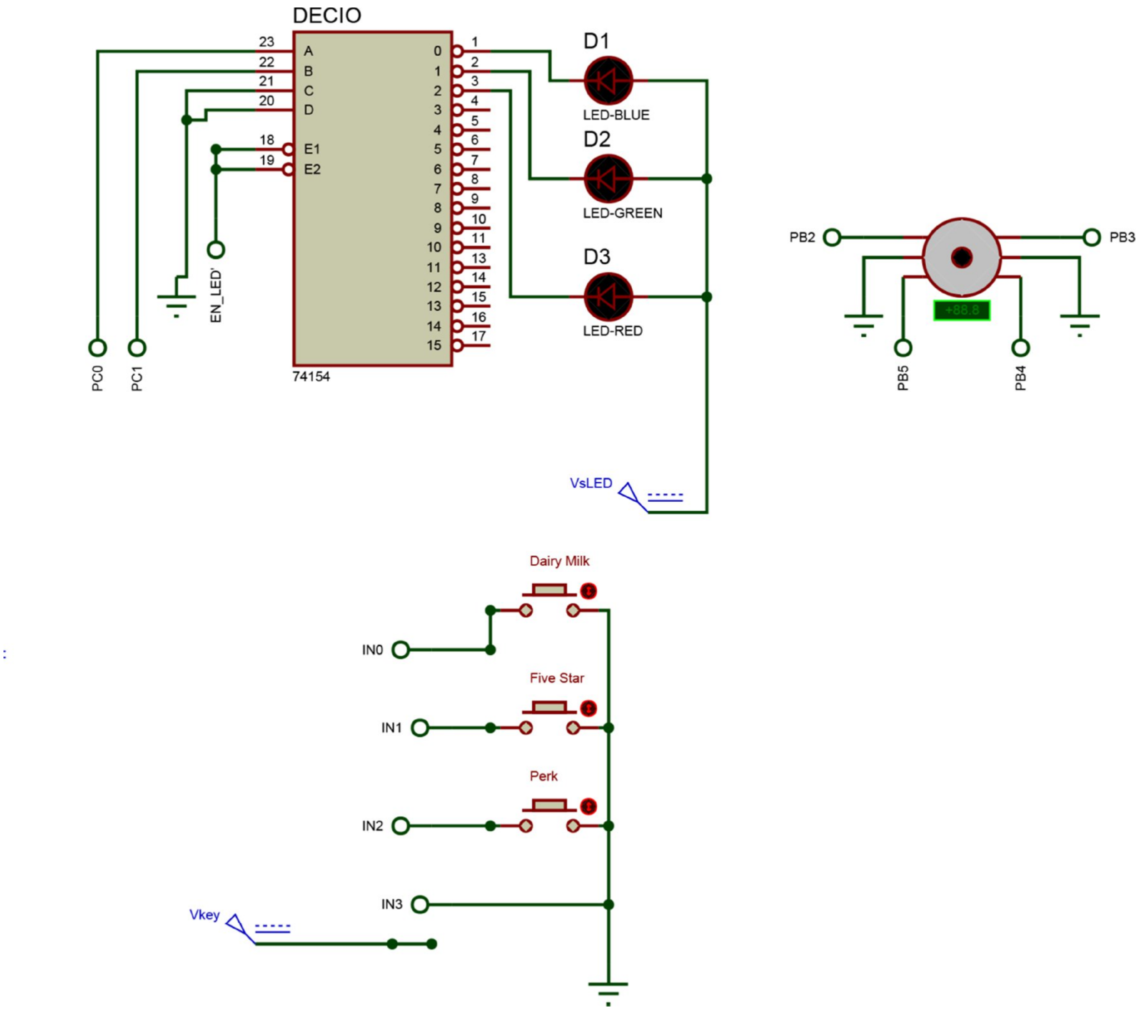
8255A WITH BUZZER CONNECTION



ADC AND PRESSURE SENSOR



LED’s AND SWITCHES



REFRENCES

1.Datasheet of MPX4250 (Pressure Sensor):

http://www.nxp.com/files/sensors/doc/data_sheet/MPX4250.pdf
http://www.datasheetcatalog.com/info_redirect/datasheet/motorola/MPX4250.pdf.shtml

2. Stepper Motor reference:

https://books.google.co.in/books?id=KJNpD2KimEsC&pg=PA228&lpg=PA228&dq=stepper+motor+interfacing+with+8086&source=bl&ots=eMysYMx2Wb&sig=CX2G1c5I_ufy-2NpoRN_Jg13Hw0&hl=en&sa=X&ved=0ahUKEwiMj6nQgKTMAhWTJI4KHQnFD9U4ChDoAQgxMAU#v=onepage&q=stepper%20motor%20interfacing%20with%208086&f=false