Design Problem 1: h_{FE} Tester

Batch No. 36

Yash Pandya (2014B4A7914G)

Himanshu Agarwal (2014B4A3578G)

Vishal Agrawal (2014B5A7566G)

Sarthak Munjal (2014B5A7701G)

Ashutosh Saboo (2014B4A7427G)

Ankit Anand (2014B5A3672G)

Problem Statement

- Design a microprocessor based transistor h_{FE} tester.The system has to display the h_{FE} value of NPN transistors.
- The transistor under test (TUT) is to be inserted in the socket, and its base is energized with a current from a device DI.
- The current I produced by the device DI, can be controlled by supplying it with a DC voltage V.
- The relationship is as follows. **$I = V * 10^{-6}$ A**
- The emitter of the transistor is grounded, and the collector is connected to a 1K resistor, whose other end is connected to the +5V supply.
- The Voltage drop across a 1K resistor is measured and this is related to the h_{FE} by the following relation:

$h_{FE} * I * 1000$ = Voltage drop

- The h_{FE} value should be displayed on a seven segment display.
- If the h_{FE} value is less than 20, an alarm should be sounded.
- For the transistor being tested current varying from 1-10 µA is given as input in steps with a resolution of 1µA.
- A switch is provided for the user -which has to be closed after the transistor has been placed in TUT Slot.

Assumptions

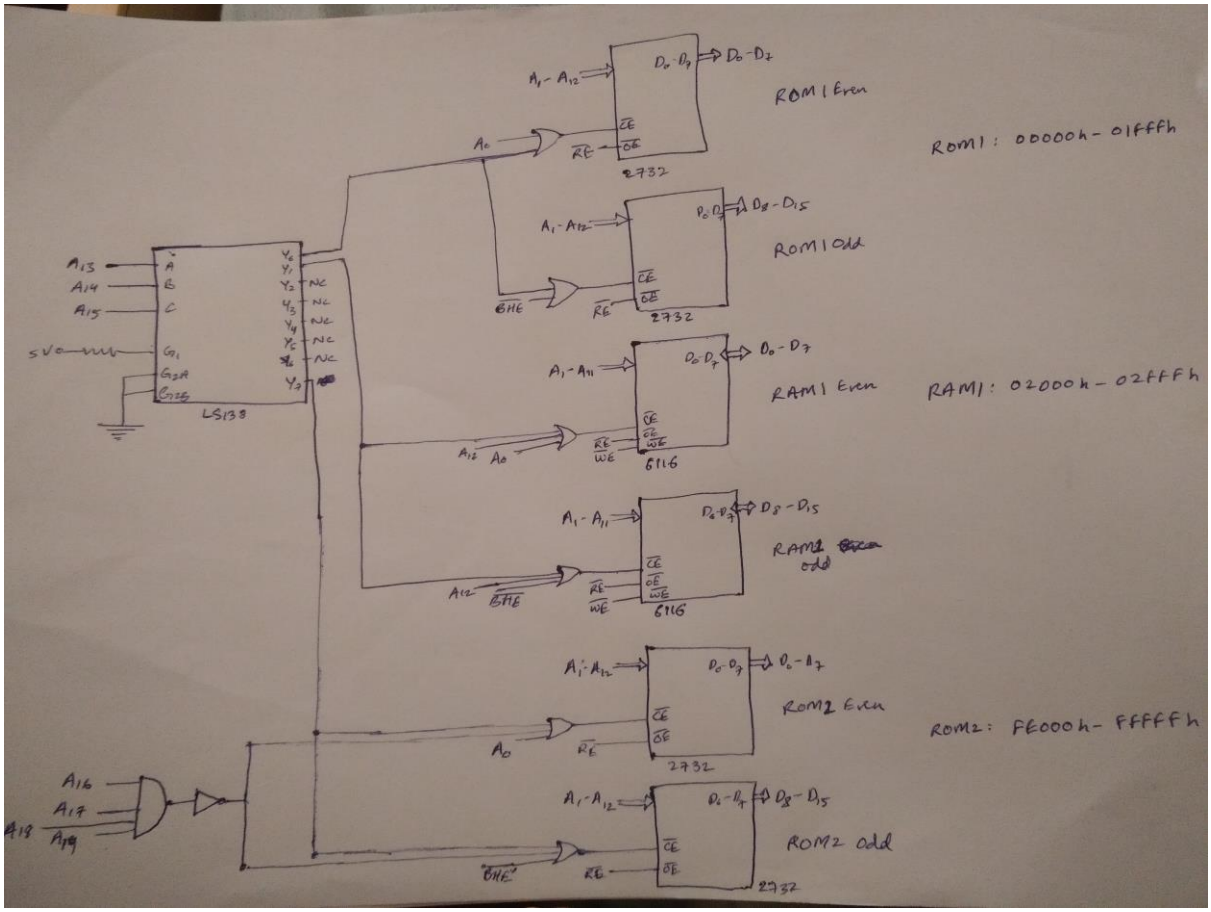
- hFE is in the range 0-255 (can be expressed in 8 bits.). If it is greater than 255, we display 255.
- We cannot do floating-point calculations in 8086, we are using reasonable approximations to deal with remainder of a division operation for hFE calculation.
- The design already includes a transistor which is placed in the TUT slot.
- The switch provided turns the sounder (alarm) off, and allows the execution to continue.
- We cannot generate 1-10 µA exact resolutions of 1 µA because the resolution of DAC 0808 doesn't allow it, but we generate reasonable approximations as best allowed by the resolution of DAC0808.
- All calculations are started with the approximation which arises from the resolution of ADC0808 being 19.53mV when we have Vref+=5V and Vref-=0V.

Components Used

Component	Quantity
8086 Microprocessor	1
8255A - Programmable Peripheral Interface	2
ADC 0808 – Analog to Digital Convertor	1
DAC 0808 –Digital to Analog Convertor	1
74LS447 – BCD to 7 segment Decoder	3
6116 – RAM	2
2732 - ROM	4
74LS373 - Latch	4
74LS245 – Octal Bus Transceiver (Buffer)	2
LF 351 – Op Amp	1
3 input OR Gate	3
74LS138 – 3:8 Decoder	2
Device DI	1
Switch	1
Sounder	1
2N2369 – NPN Transistor	1
Resistor – 5k ohms	4
Resistor – 1k ohms	1
0.1 µF – Capacitor	1
Power and Clock Generator	-
7 Segment Display	3

Memory Interfacing

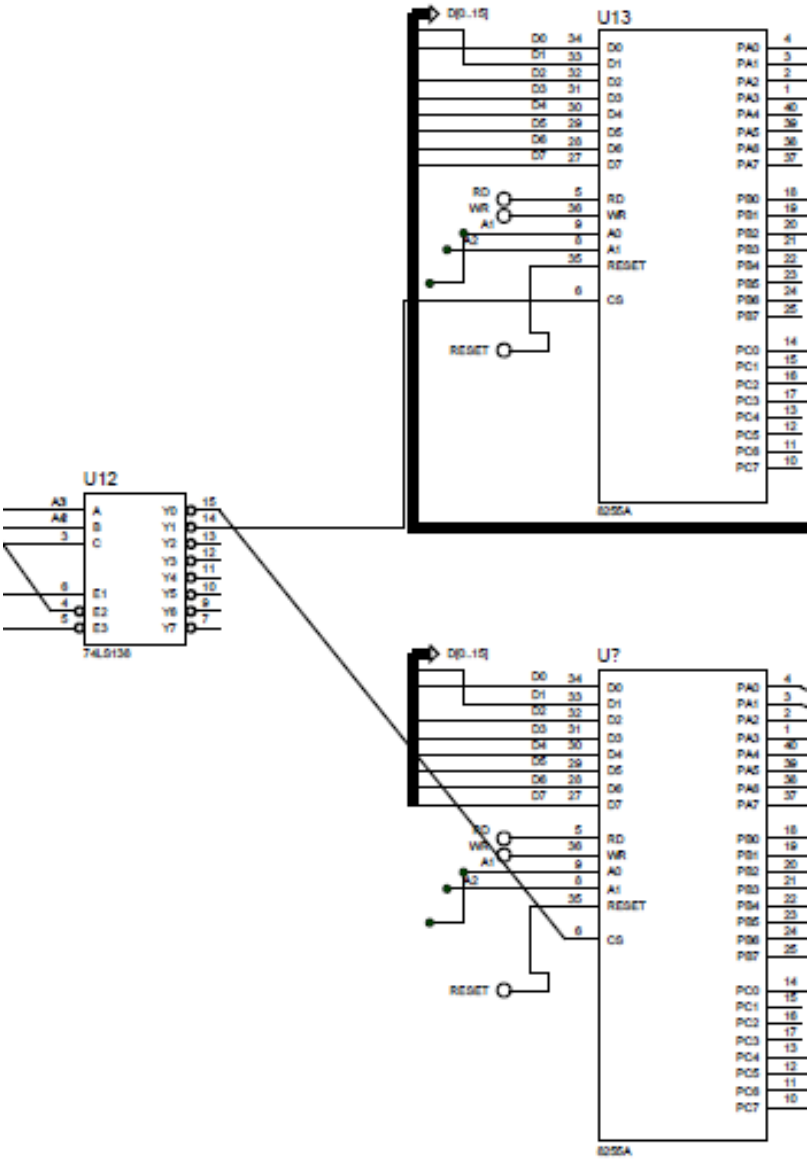
Memory Component	Start Address	End Address
ROM1	00000h	01FFFh
ROM2	FE000h	FFFFFh
RAM1	02000h	02FFFh



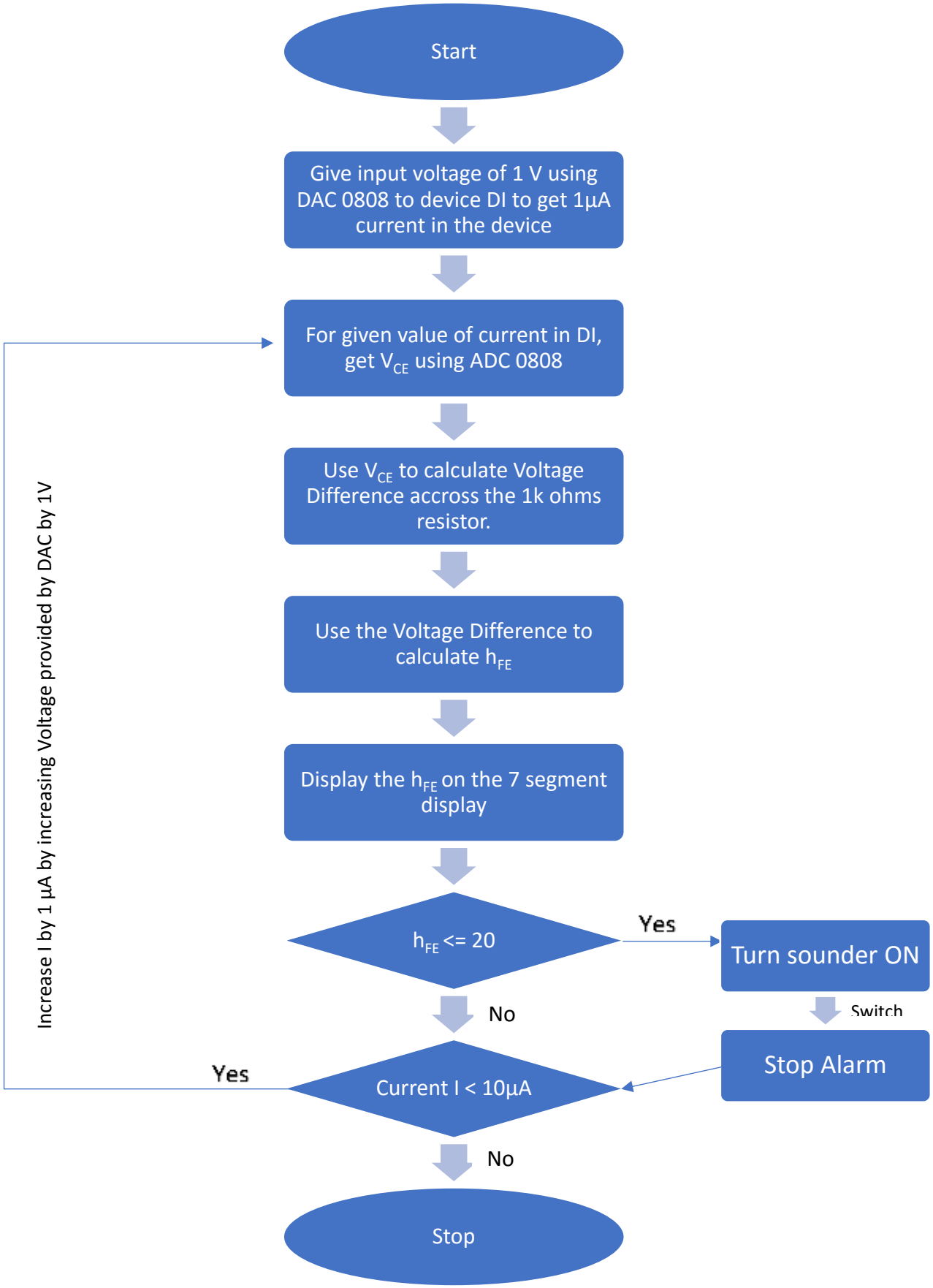
I/O Interfacing

Using two 8255A:

- The 8255A used to connect the DAC 0808, ADC 0808 and switch has starting address of 00h.
- The 8255A used to connect to 7 segment display has starting address of 08h.



Flow Chart


Start
Give input voltage of 1 V using DAC 0808 to device DI to get 1µA current in the device
For given value of current in DI, get V_{CE} using ADC 0808
Use V_{CE} to calculate Voltage Difference accross the 1k ohms resistor.
Use the Voltage Difference to calculate h_{FE}
Display the h_{FE} on the 7 segment display
h_{FE} <= 20
Yes
Turn sounder ON
Switch
Stop Alarm
No
Current I < 10µA
Yes
Increase I by 1 µA by increasing Voltage provided by DAC by 1V
No
Stop

ALP

```
#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#



; add your code here


         jmp     st2
         nop
         dw      0000
         dw      0000
         dw      ad_isr
         dw      0000

           db     1012 dup(0)
;main program



st2:      mov ax, 0
          mov [0ah], ax
          cli
; intialize ds, es,ss to start of RAM
          mov       ax,0200h
          mov       ds,ax
```

```
          mov       es,ax
          mov       ss,ax
          mov       sp,0FFFEH

;intialise porta  as input & b& c as output
          mov       al,91h
          out           06h,al

         mov          al,80h
         out          0eh,al

          mov al, 17
          mov [02], al

          mov          al,00001110b
         out          06h,al

            mov       al, 91h
            out 06h, al

;select ch0
st1:      mov al, [02]
            out 2, al

;give ale
          mov       al,00100000b
            out       04h,al

;give soc
          mov          al,00110000b
            out          04h,al

            nop
            nop
            nop
            nop
;make soc 0
            mov       al,00010000b
            out       04h,al
;make ale 0

            mov       al,00000000b
            out       04h,al

          mov [04], 0
aa:       cmp [04], 0
          jz aa
```

```
            mov cx, 1000
xx1:       nop
           loop xx1

            mov al, [02]
           add al, 17
           mov [02], al

           mov ax, 170
           cmp [02], al
           jbe st1

           mov ax, 10
           mov bx, [0ah]
           mov cx, 0
             mov dx, 0

x23:     add cx, 1
          add dx, ax
          cmp dx, bx
          jle x23

          dec cx

           mov al, cl
           mov ah,0
           mov bl, 10
           mov bh, 0
           div bl
           mov cl, al
           mov al, ah
           out 0ch, al

           mov al, cl

           mov ah,0
             mov bl, 10
             div bl
             mov cl, al
             mov al, ah
             out 0ah, al


           mov al, cl

           mov ah,0
             mov bl, 10
             div bl
             mov cl, al
```

```
            mov al, ah
            out 08h, al

inf:      jmp inf

ad_isr:   mov          al,01000000b
            out          04h,al
            in        al,00h

            mov [04], 1

            mov [06], al

           mov cl, 255
           sub cl, al
           inc cl
           mov al, 33
           mov ah, 0
           mul cl
           mov bx, ax



           mov al, [02]
           mov ah, 0


             mov cx, 0
             mov dx, 0



xxxx:    add cx, 1
         add dx, ax
         cmp dx, bx
         jle xxxx

            dec cx

            mov ax, cx
            mov cx, 10
            mul cx

            mov [08], ax


            cmp ax, 256
            jle xx2
            mov ax, 255
```

```
xx2:      mov al, al
          mov ah, 0
          mov bx,[0ah]
          add bx,ax
          mov [0ah], bx
          mov cl, al

            mov ah,0
            mov bl, 10
            div bl
            mov cl, al
            mov al, ah
            out 0ch, al

          mov al, cl

          mov ah,0
            mov bl, 10
            div bl
            mov cl, al
            mov al, ah
            out 0ah, al


          mov al, cl

          mov ah,0
            mov bl, 10
            div bl
            mov cl, al
            mov al, ah
            out 08h, al
         mov ax, 20
            cmp [08], ax
         jg rett

buzz:     mov          al,00001111b
            out          06h,al

            mov       al, 91h
            out 06h, al

            in al, 04h
            and al, 01h
            jz buzz
          mov          al,00001110b
            out          06h,al
```

```
            mov       al, 91h
            out 06h, al

rett:
          iret
```