LAB 1 Pre Work CMPE 336 / EEE 347

Hassaan Ali 2344810
Sabawoon Afzal Khattak 2328284
Ihab R. Ahmad 2328466

"The content of the report represents the work completed by the submitting team only, and no material has been borrowed in any form."

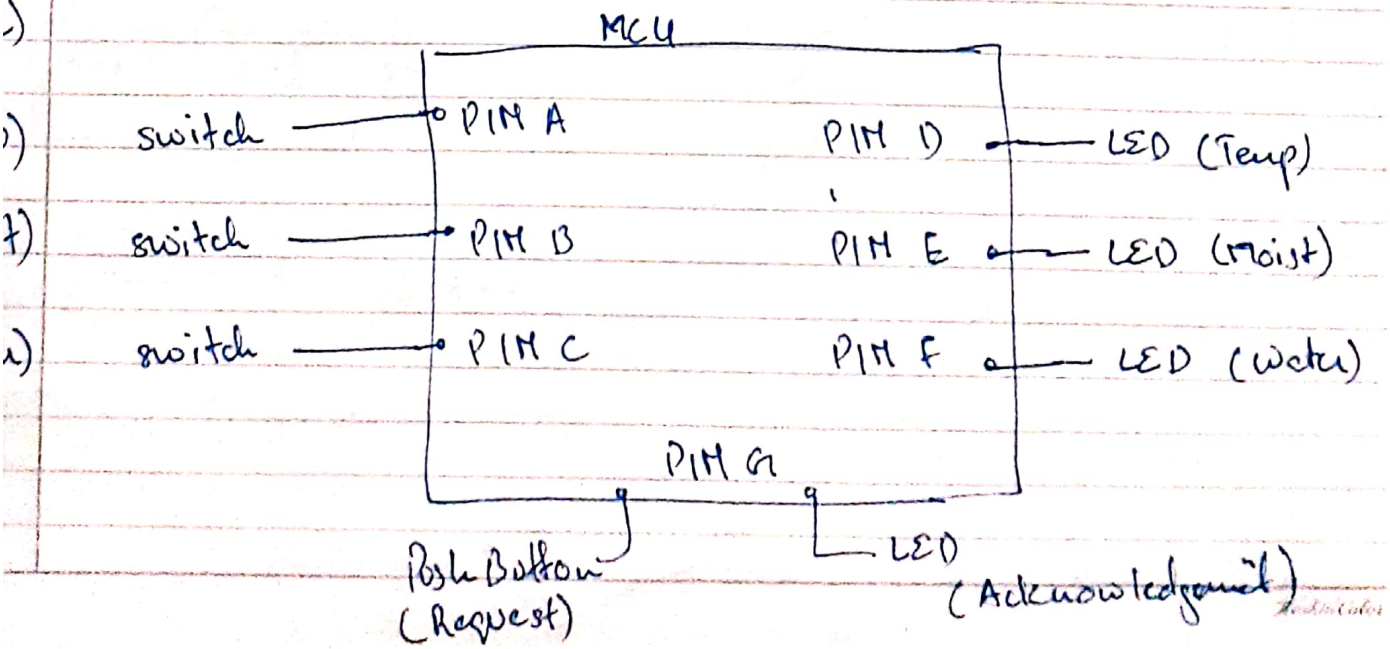
Objective: The purpose of the first laboratory exercise is getting familiar with the development environment in order to apply course learnings.

1.2.2

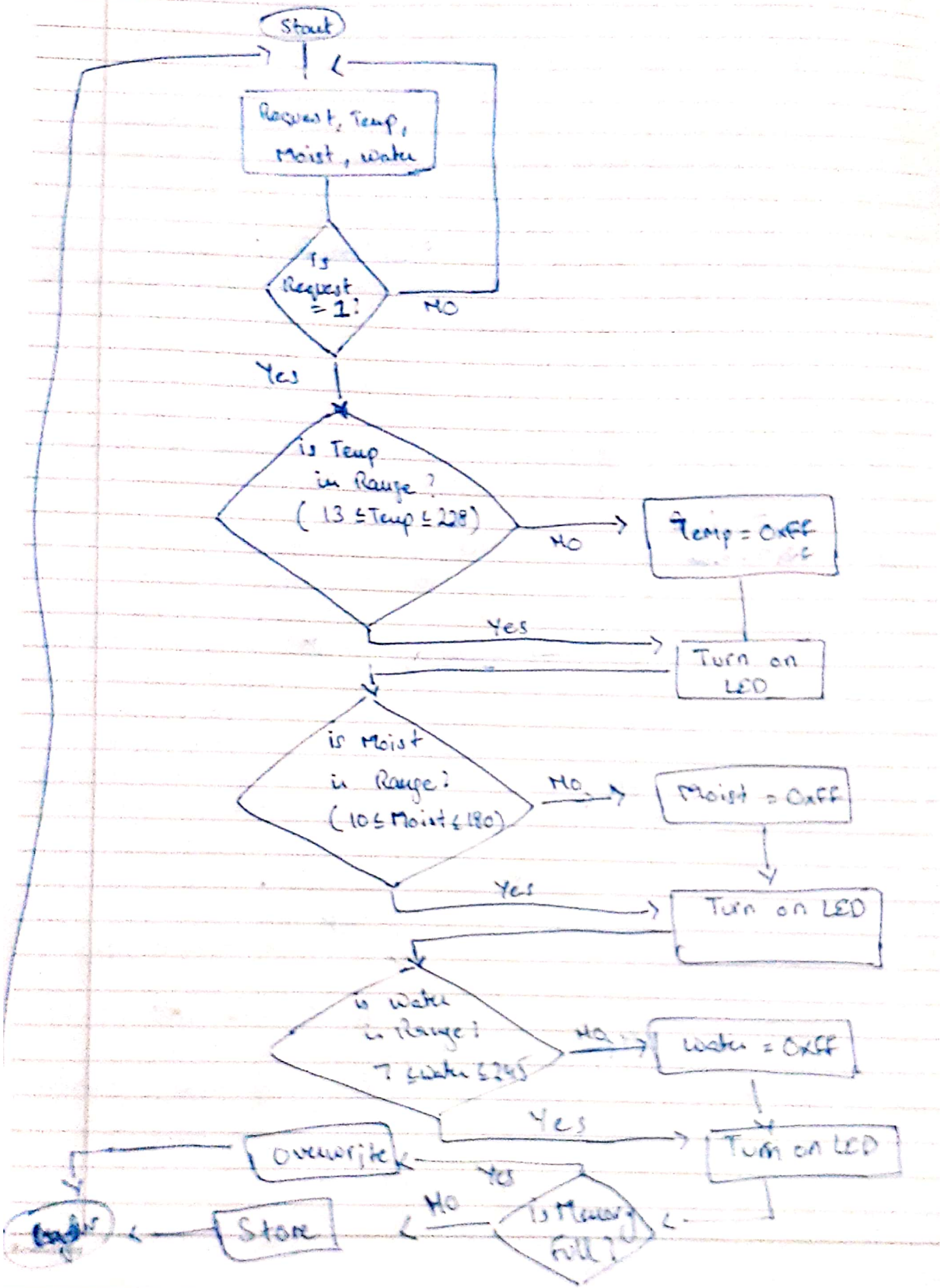
(a) After review of ATMEGA128, we have decided the following configuration of inputs/outputs to the MCU pins:

(b)

Port	Input/Output	Purpose
A	Input	Temperature Control Switch
B	Input	Moisture Control Switch
C	Input	Water Control Switch
D	Output	Temperature Display LED
E	Output	Moisture Display LED
F	Output	Water Display LED
G	Both	Request Push Button & Acknowledgment LED.



d) Flow Chart



Question : Which lines in the code do not affect the machine state at all?

Answer : Up to .ORG 0x0050, lines of code do not affect machine state at all.

1.2.4 What is the difference b/w RAM & ROM?

Ans RAM is volatile memory, means it requires power. ROM is non-volatile memory. This means that RAM is used as temporary storage, while ROM is a form of permanent storage. ROM is less slower & has less storage capacity compared to RAM.

Q Why is there a need for both a flash memory & EEPROM?

Ans The need for both is due to the necessity that we cannot run program code out of EEPROM, and because flash memory erases bytes in chunks unlike EEPROM which is capable of byte by byte erasing.

1.2.5 What is the function of the control unit?

Ans Control unit manages the ALU, memory & I/O devices.

Q How does the control unit get the instruction that it must execute?

Ans It obtains the instructions from RAM & ROM.

Q How does it know the number of bytes in an instruction?

Ans After decoding, it knows the number of bytes.

Code

```
;Lab1
;Hassaan Ali 2344810
;Ihab R. Ahmad 2328466
;Sabawun Afzal Khattak 2328284


.INCLUDE "m128def.inc"

.EQU ZEROS = 0x00
.EQU ONES = 0xFF

.EQU T_LO_LMT = 0x0D                    ; 13  Temperature Low
.EQU T_HI_LMT =0xE4                     ; 228 Temperature High

.EQU M_LO_LMT = 0x0A                    ; 10  Moisture Low
.EQU M_HI_LMT = 0xC8                    ; 180 Moisture High

.EQU W_LO_LMT = 0x07                    ; 7   Water Low
.EQU W_HI_LMT = 0xF5                    ; 245  Water High

.EQU MEM_START = 0x100
.EQU MEM_END = 0x10FF



.CSEG
```

```
LDI XL, 0x00
LDI XH, 0x01


LDI YL, 0xFF
LDI YH, 0x10

LDI R16, 0x30
 LDI R17, ZEROS
 OUT DDRA, R17                ;Pin A input
 OUT DDRB, R17                ;Pin B input
 OUT DDRC, R17                ;Pin C input
LDI R17, ONES
OUT DDRD, R17                 ;Pin D output
OUT DDRE, R17                 ;Pin E output
 STS DDRF, R17                                          ;Pin F output

LDI R18, 0b00000010             ;load R18 with 0x10
STS DDRG, R18                 ;make PG1 an output pin, PG0 INPUT
REQUEST:        LDS R18, PING               ;load portG into R18
                SBRS R18, 0                 ;skip if bit PG0(push button) is high
                RJMP REQUEST                ;check again if low

  IN R19, PINA
  IN R22, PINB
  IN R25, PINC

  LDS R23, PORTG
```

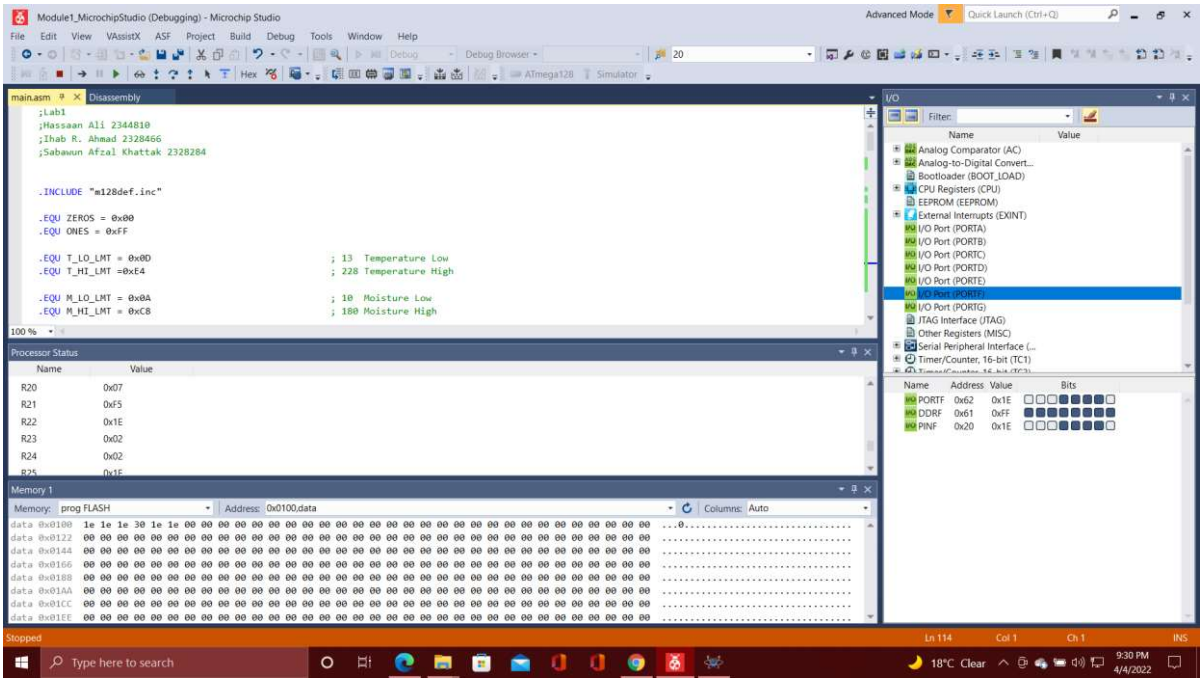
```
  LDI R24, 0b00000010          ;make the 2th bit high
           OR R23,R24          ;mask it onto PG1
       STS PORTG, R24          ;send modification to pinG

  LDI R20, T_LO_LMT
  LDI R21, T_HI_LMT
           CP R19, R20         ;compare the values
           BRSH Loop1          ;branch same or higher
           LDI R19, ONES       ;replace R19 with ones
  JMP Loop2

Loop1:     CP R21, R19         ;compare the values
           BRCC Loop2          ;branch if Lower
           LDI R19, ONES       ;replace R19 with ones

Loop2:     OUT PORTD, R19
  LDI R20, M_LO_LMT
  LDI R21, M_HI_LMT
           CP R22, R20         ;compare the values
           BRSH Loop3          ;branch if same or higher
           LDI R22, ONES       ;replace R22 with ones
  JMP Loop4

Loop3:     CP R21, R22         ;compare the values
           BRCC Loop4          ;branch if lower
           LDI R22, ONES       ;replace R22 with ones

Loop4:     OUT PORTE, R22
  LDI R20, W_LO_LMT
```

```
  LDI R21, W_HI_LMT
                    CP R25, R20              ;compare the values
                    BRSH Loop5               ;branch if same or higher
                    LDI R25, ONES            ;replace R25 with ones
  JMP Loop6

Loop5:          CP R21, R25                ;compare the values
                BRCC Loop6                 ;branch if lower
                LDI R25, ONES              ;replace R25 with ones

Loop6:          STS PORTF, R25

STORETemp:      ST X+, R19
  RJMP Check1
STOREMoist:     ST X+, R22
  RJMP Check2
StoreWater:     ST X+, R25
                RJMP Check3
StoreNull:      ST X+, R16
  RJMP Check4


Check1:         CP YL, XL
   CPC YH,XH
   BRCS OVERWRITE
  RJMP STOREMoist

Check2:         CP YL, XL
   CPC YH,XH
```

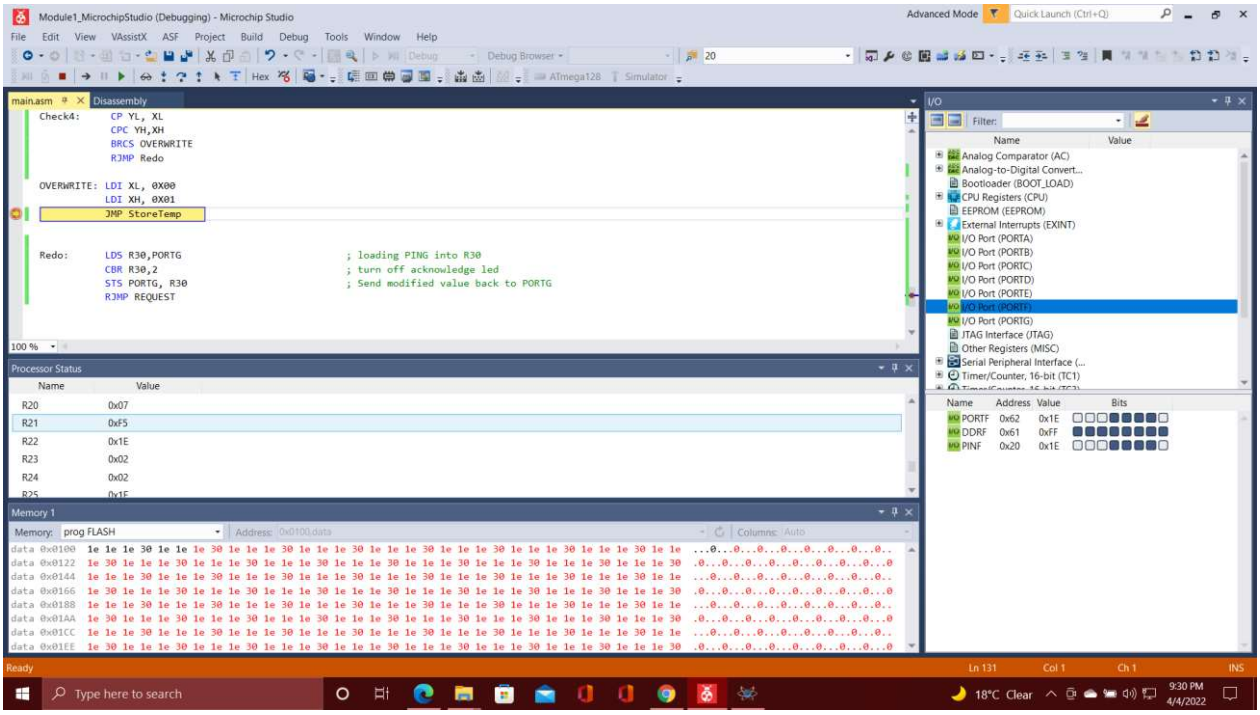
```
    BRCS OVERWRITE
   RJMP StoreWater

Check3:         CP YL, XL
    CPC YH,XH
    BRCS OVERWRITE
    RJMP StoreNull

Check4:         CP YL, XL
    CPC YH,XH
    BRCS OVERWRITE
   RJMP Redo

OVERWRITE:      LDI XL, 0X00
   LDI XH, 0X01
   JMP StoreTemp


Redo:           LDS R30,PORTG                   ; loading PING into R30
                CBR R30,2                       ; turn off acknowledge led
   STS PORTG, R30                  ; Send modified value back to PORTG
   RJMP REQUEST
```

MICROCHIP

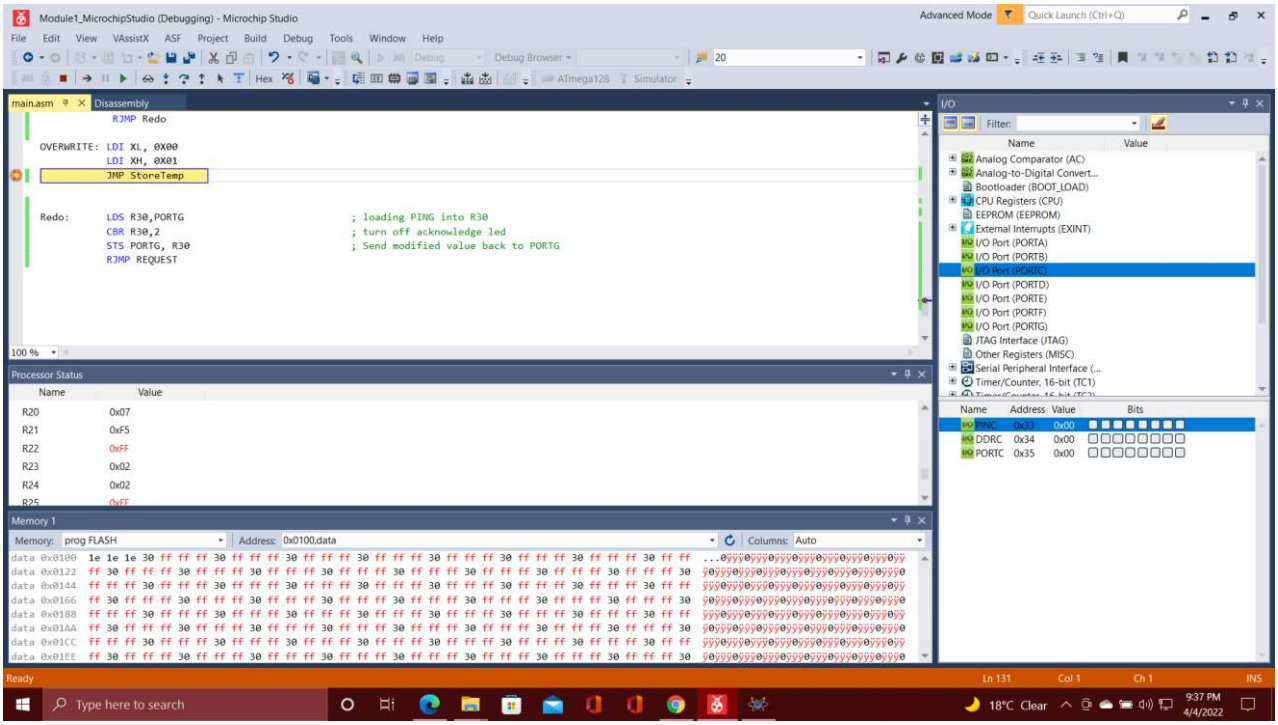
Normal Case Everything working Fine (In-Range Values)



Overwriting Case (Overwriting Working Fine)

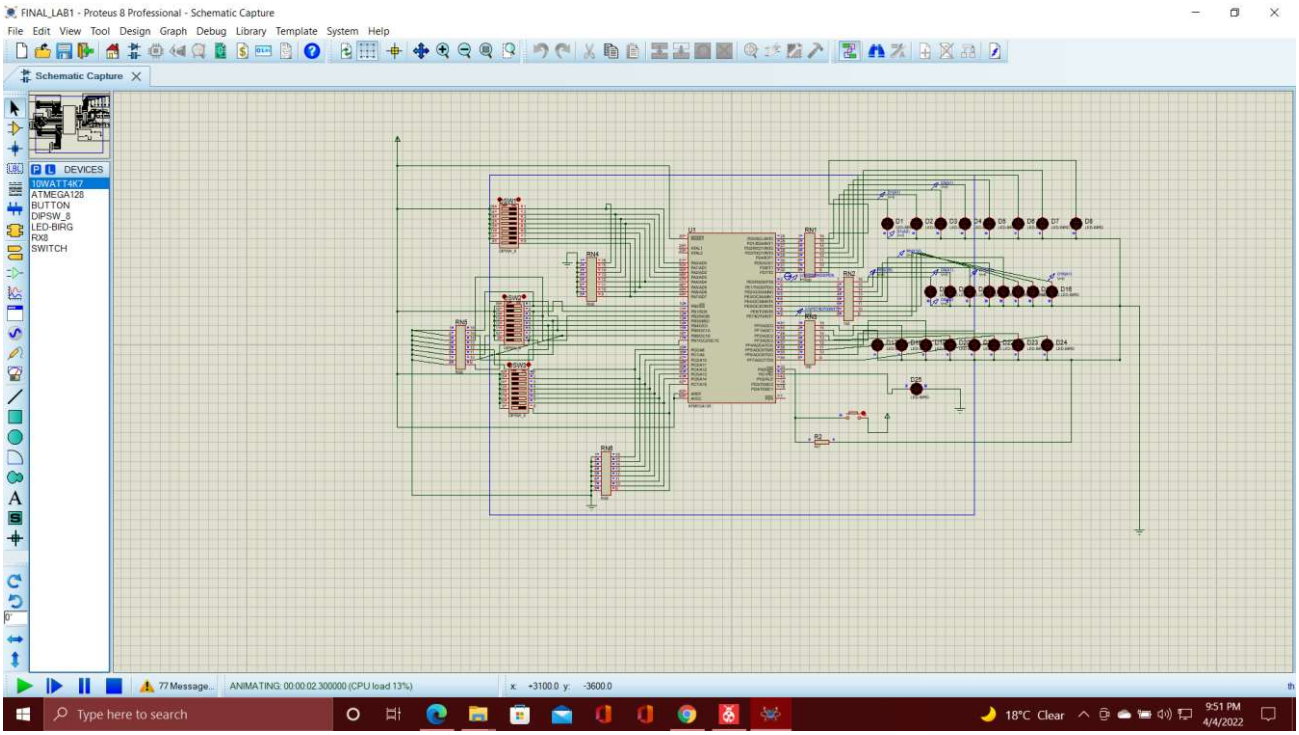


Overwriting further showed with out of range values corrected to OxFF by code

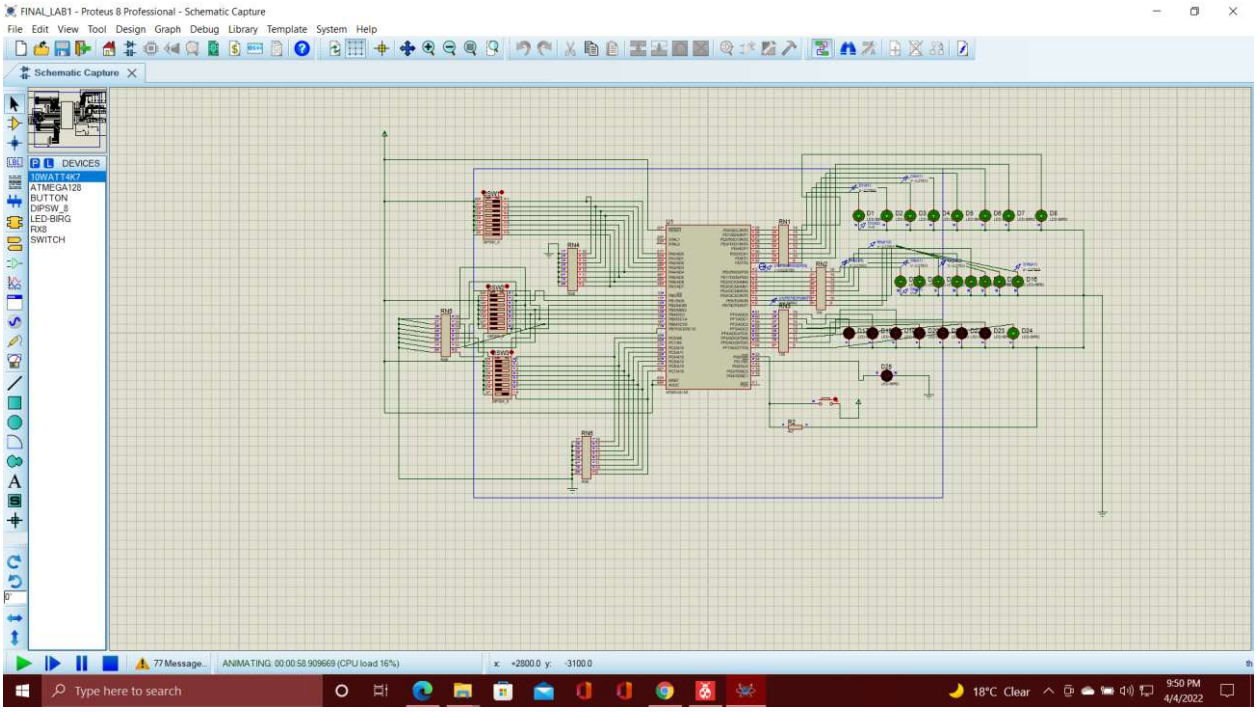


PROTEUS

Push Button Off



Push Button On (Two Invalid and One Valuid Ranges)



PUSH Button On (Valid Ranges For All i.e 128)