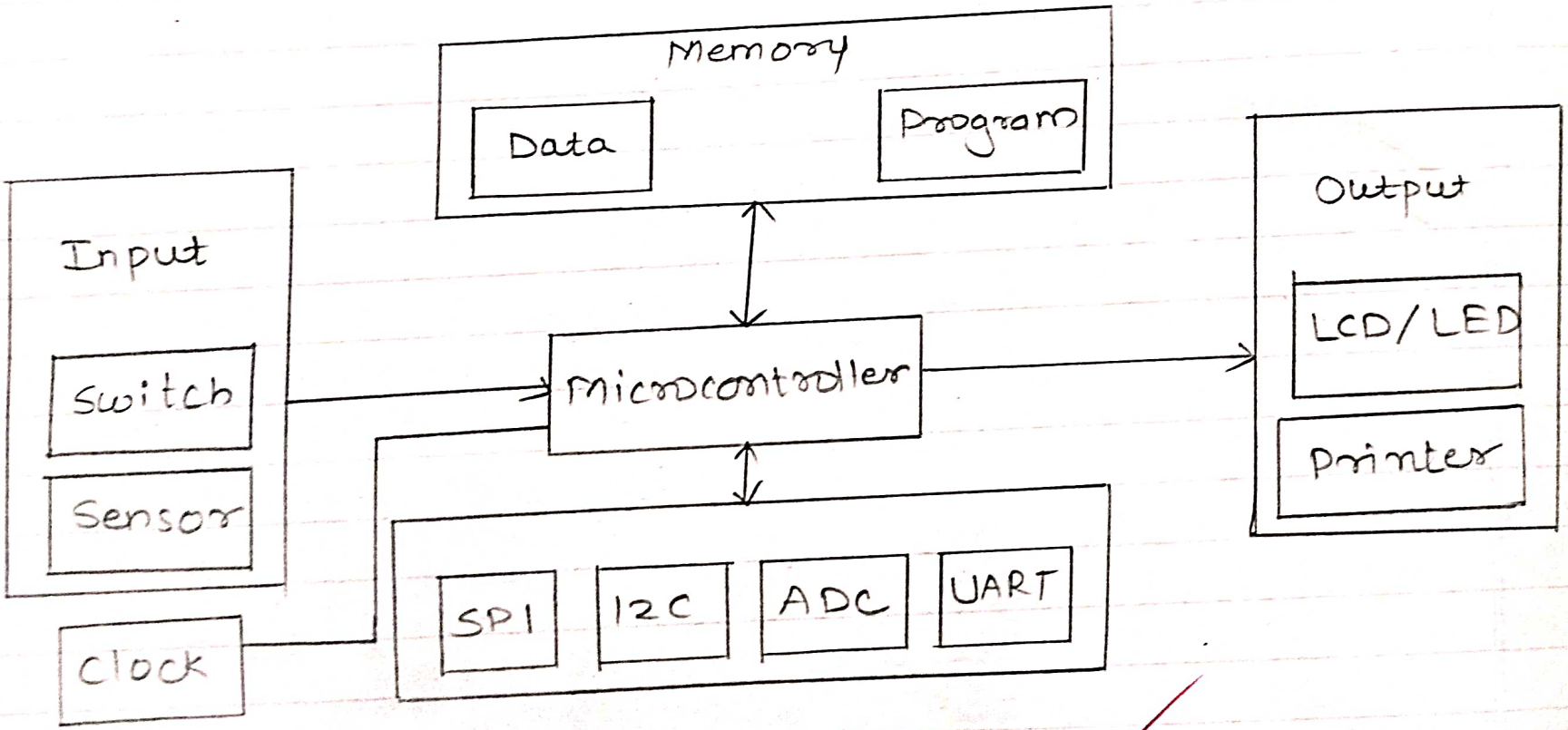
* Description :

Block Diagram of an Embedded System



EXPERIMENT : No. Practicle Number 1

PAGE No.
DATE

Design and develop a reprogrammable embedded computer using 8051 microcontrollers and to show the following aspects.

a. Programming
b. Execution
c. Debugging

→ Design and development of a Reprogrammable Embedded System (RBS) using 8051 Microcontroller (MC).

* Software Requirement : Editor for schematic drawing like Eagle.
* Hardware Requirement : Soldering Iron, Tweezer, Cutter, Multimeter, Components as per Table.
* Components of a Computer :
(a) Embedded system is its controller which is called as microprocessor unit (MPU).
(b) MPU needs more peripherals to finish a task.
(c) MPU units generally have on chip peripherals such as memory elements like ROM/RAM basic, function elements like Timers/Counters/Interrupts and special interfaces like UART/SPI/I2C/CAN etc.
* Re-programmable Embedded System (RES) :
(a) A re-programmable embedded system means there is no need to pull out MCU every time one wants to program it and hence provide flexibility in programming and operations.
(b) To execution of intended program RES also provide debugging facility and chip programming for other users.

Sundaram

Teacher's Sign.: _____

EXPERIMENT : No. _____ PAGE No. _____ DATE _____

* The Reprogrammable embedded system consists of
- Sockets for placing microcontroller - 40 pin
- DC socket for external power supply (DC 5V)
- 1 LED for power on indication and 1 push button for reset.
- 11.0592 MHz Quartz Crystal Oscillator.
- 8 LEDs for output pin state indication at port P0
- 1 DIP switch for input pin activation.
- Connector and driver for serial communication RS232
- Multiple-pin connectors for direct access to I/O ports.
- Connector for SPI programming.
- 1 Piece buzzer for audio / Frequency output.
- Additional power supply connectors.

* Selection of component for a given application:
Every application circuit is build around some components which should be selected as per the functionality of the application, availability of components, cost of entire system, procurement time for components and most importantly meeting of some critical parameters of intended application.

* Selection of Processor:
- Selection is based on architecture, availability, cost, time to prototype and market, testability and debug ability.
- Some advance controllers of 8051 architecture provide boot-loader, in system programming and in

Sundaram

Teacher's Sign.: _____

EXPERIMENT : No. _____ PAGE No. _____ DATE _____

system application programming.

- Re-programmability is achieved using ISP (In-System-Programming) feacture provided by NXP P89V51RD2 or by Atmel AT89S52.
- NXP's P89V51RD2 and Atmel's AT89S52 features include;
  - 8-bit, 40-pin controller in DIP package.
  - Operating voltage +5V
  - Operating frequency 0 to 40 MHz
  - 32-Input/Output pins
  - 3-16 bit Timers
  - 8-Interrupt levels
  - 1-UART
  - 1 KB of user RAM
  - 64 KB of Flash

* Selection of other components:

* Serial communication interface →

- UART (Universal Asynchronous Receiver Transmitter) is required for boot-loader/ISP/IAP programming and also for applications that include PC interfacing.
- Operating voltage requirement is +5V

* Oscillator →

- Oscillator is used as a clock signal generator.
- crystal oscillator are used for their frequency stability and hence should be chosen over other type of oscillator.

EXPERIMENT : No. _____ PAGE No. DATE

* Connector :→

- DB9 female PCB Mount : 3 pins of DB9 connector (pin 2-RD, pin 3-TXD and pin 5-GND) are used for connections between PC and UART IC i.e. MAX232.
- connectors for direct access to Ports : In order to enable microcontroller ports to be directly connected to additional components, each of them is connected to 8 pin, on-board connector.

* Input selection →

- 8-DIP switches are provided on board here for interfacing with any of input port.
- Inputs from sensors / ADC / PC may also be connected through port connectors.

* Output Selection →

- LED : 8-LEDs are connected at port0 with 1KΩ resistor network RN1.
- LCD : 16x2, LCD may be connected using I/O port connectors.
- Output at PC / DAC / Motors (through drivers) is also supported.

* Power Supply →

- There is a connector on the development board enabling connection to external power supply source (DC-5V).
- Voltage is necessary for device operation can also be obtained from PC via USB

cable at connector J7/J8.

* selection of tools →

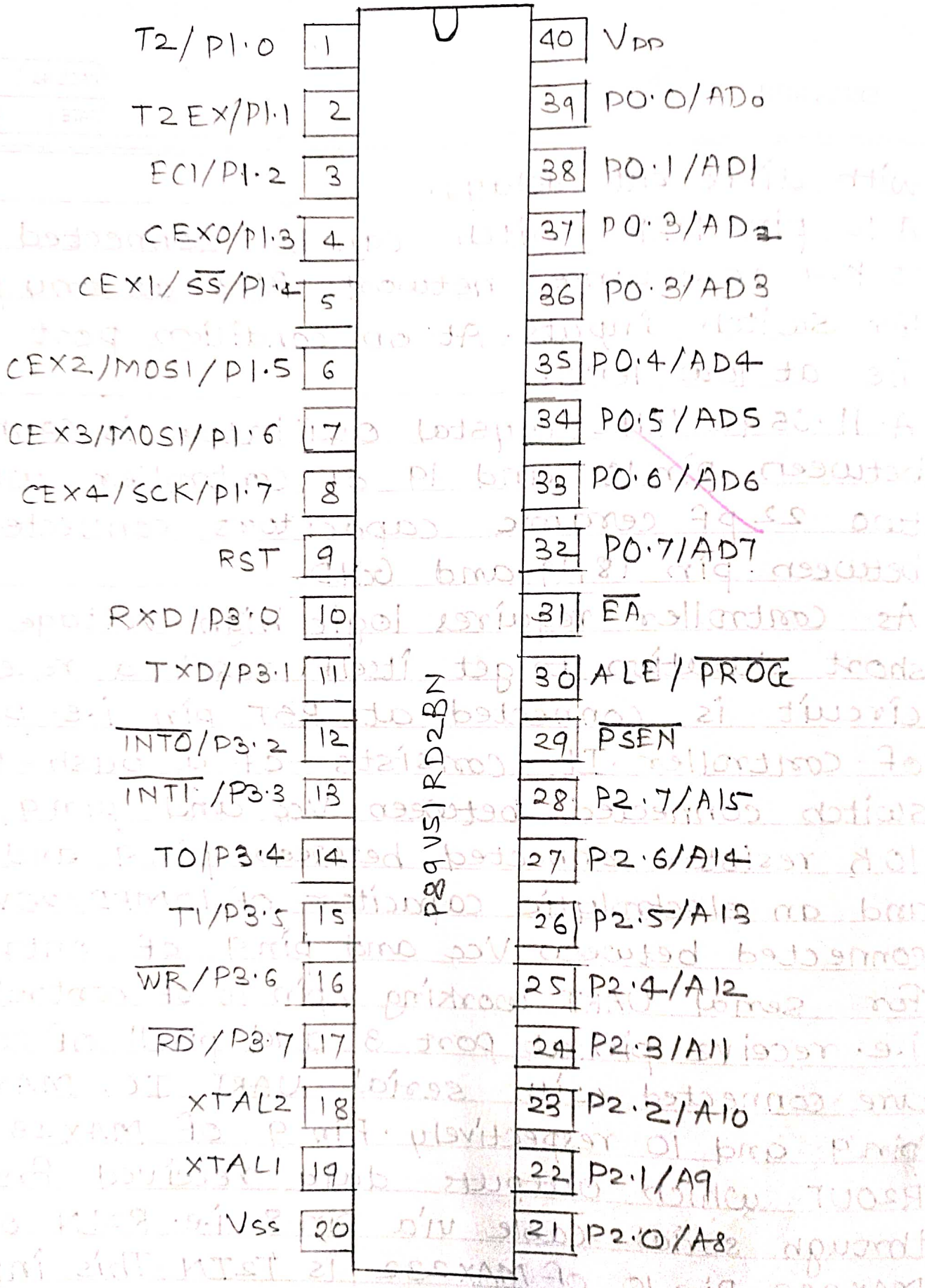
- Some tools and editors are required to prepare assembly language program and it's compiling i.e. hex file generation and writing this hex file to flash memory.
- Free downlabel Keil u vision version 4, editor is used for writing assembly language program and its compiling.
- Free downloaded Flash Magic or USB programmable is used for Flash programming.
- Hyper terminal available with windows is used for debugging purpose.

* Schematic Diagram :

1. Microcontroller 89V51RD2 is biased with +5V power supply connected at pin 40, GND to suppress supply spikes.
2. Enable Access (EA), pin 31 and PSEN pin 29 are all connected with Vcc. PSEN bar is connected to high logic as only internal flash memory is in use.
3. Cathode of all 8 LEDs are connected at different pins of port 0 i.e. from pin 32 to 39 of controller. LED anode will be connected to Vcc through 1K-Ω resistance network RN1. These LEDs will be used in program to view outputs or to check proper functioning by blinking them

with different delays.

4. A 16 pin DIP switch can be connected through 10 KΩ resistance network RN3 at any port for switch inputs. At on condition port will be at low level.
5. A 11.0592 MHz crystal oscillator is connected between pin 18 and 19 of controller, with two 22 pf ceramic capacitors connected between pin 18,19 and GND.
6. As controller requires logic high voltage for short duration to get itself reset, a reset circuit is connected at RST pin i.e. pin 9 of controller. It consists of a push-to-on switch connected between Vcc and pin 9, a 10K resistor connected between pin 9 and GND and an electrolytic capacitor of 10MFD/25V, connected between Vcc and pin 9 of controller.
7. For serial UART working, pin 10 of controller i.e. receive pin at port 3 and pin 11 of controller are connected with serial UART IC, MAX232 pin 9 and 10 respectively. Pin 9 of MAX232 is R2OUT which outputs data received from PC through serial cable via pin 8 i.e. R2IN of MAX232. Pin 10 of MAX232 is T2IN. This input data is then sent to PC through serial cable via pin 7 i.e. T2OUT of MAX232.
8. IC MAX232 is biased with +5V supply at pin 16, GND at pin 15. Rest of its biassing is done as per recommended circuitry. Four



Pin diagram of P89V51RD2/AT89S52

EXPERIMENT : No.

PAGE No.

DATE

number 10 MFD/63V electrolytic capacitors are connected as recommended.

9. DB9 connector is connected between MAX 232 and PC.

4/2/19

good.

EXPERIMENT : No. Practicle NO. 2

PAGE No. DATE

Demonstrate timer control registers of 8051 and developement program to generate give time delay.

→ * STEPS :

1. Open project tab.
2. Click New uVision project.
3. Save your project with name.
4. Search AT89C51 which is 8051 microcontroller.
5. Select AT89C51 → OK.
6. One dialog box will open → Click NO.
7. Open file tab → new file.
8. Type program
9. Save your program with .c extension.
10. + ☐ Target open.
11. Rigut Click on source group.
12. Click on Add Existing file to------.
13. Choose your .c file.
14. Click on add → Close
15. Click on Translate To debugging
16. Project → Build target → for compiling
17. Rigut click target
18. Options for Target
19. Change frequency 11.0592
20. Click on checkbox ☑ Use on-chip ROM
21. Select output tab → Click on ☑ create HEX file → OK
22. select Translate
23. Select Re-Build
24. Open proteus 8 professional.

Sundaram

Teacher's Sign.: _____

EXPERIMENT : No. _____ PAGE No. _____ DATE _____

25. Create new project.
26. Set project name.
27. Next → Next → Next → till finish.
28. Click on isis [isis].
29. Select [P] → Pick device.
30. Search AT89C51
31. 8051 Microcontroller. → OK
32. [P] select
33. Select LED-RED Light → OK
34. put microcontroller on graphical page.
35. put LED on graphical page.
36. Go to [⊟] → GROUND.
37. Put on graphical page.
38. Join one corner of LED to P2.0/A8.
39. Join second corner of LED to ground.
40. Double click on microcontroller.
41. Select program file (.hex file)
42. Change clock frequency to 11.0592 Hzs. → OK
43. Run [▷]

* Components Used :

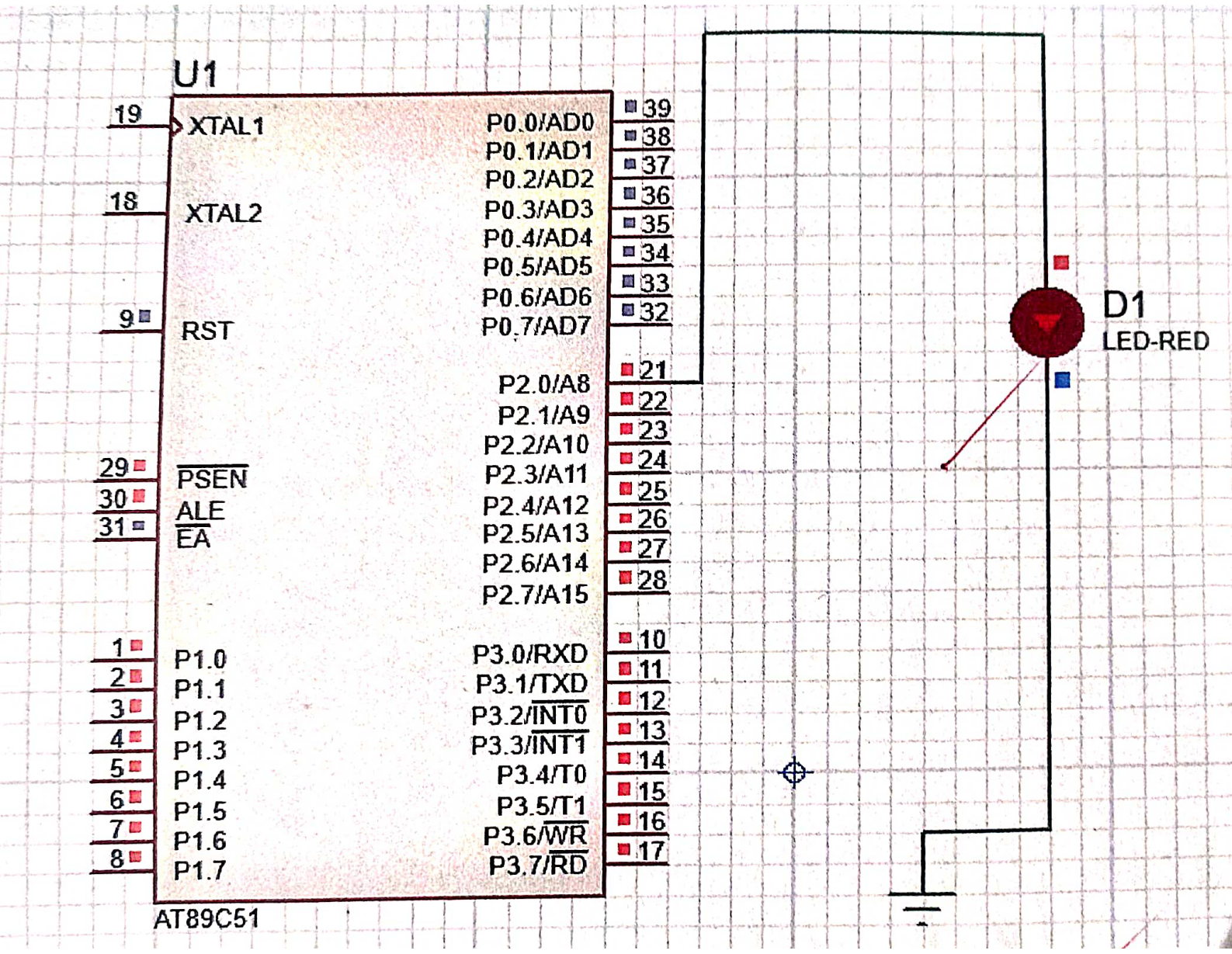
1. AT89C51
2. LED-RED

* Code :

```
#include <reg51.h>
void delay()
{
int i;
```

Sundaram

Teacher's Sign.: _____


U1
19
XTAL1
18
XTAL2
9
RST
29
PSEN
30
ALE
31
EA
1
P1.0
2
P1.1
3
P1.2
4
P1.3
5
P1.4
6
P1.5
7
P1.6
8
P1.7
AT89C51
P0.0/AD0
P0.1/AD1
P0.2/AD2
P0.3/AD3
P0.4/AD4
P0.5/AD5
P0.6/AD6
P0.7/AD7
39
38
37
36
35
34
33
32
P2.0/A8
P2.1/A9
P2.2/A10
P2.3/A11
P2.4/A12
P2.5/A13
P2.6/A14
P2.7/A15
21
22
23
24
25
26
27
28
P3.0/RXD
P3.1/TXD
P3.2/INT0
P3.3/INT1
P3.4/T0
P3.5/T1
P3.6/WR
P3.7/RD
10
11
12
13
14
15
16
17
D1
LED-RED

```
# include <reg51.h>
void delay()
{
int i;
for (i=0; i<20; i++)
{
TMOD =0X01;
TL0 =0XB0;
TH0 = 0X3C;
TR0 =1;
while (TF0==0)
TF0=0;
TR0=0;
}
}
void main()
{
while(1)
{
P2=0X00;
delay();
P2=0XFF;
delay();
}
}
```

o alphabet

EXPERIMENT:

No.

PAGE No.

DATE

18/2/19

good

EXPERIMENT : No. Practical NO. 3

PAGE No. DATE

A] Code I/O use one of the four codes of 8051 for output interface to 8 LED simulate binary counter on LED.

AND

B] To interface 8 LED at I/O code and create different pattern.

→ * Components Used :

1. AT89C51
2. LED - GREEN

* Code :

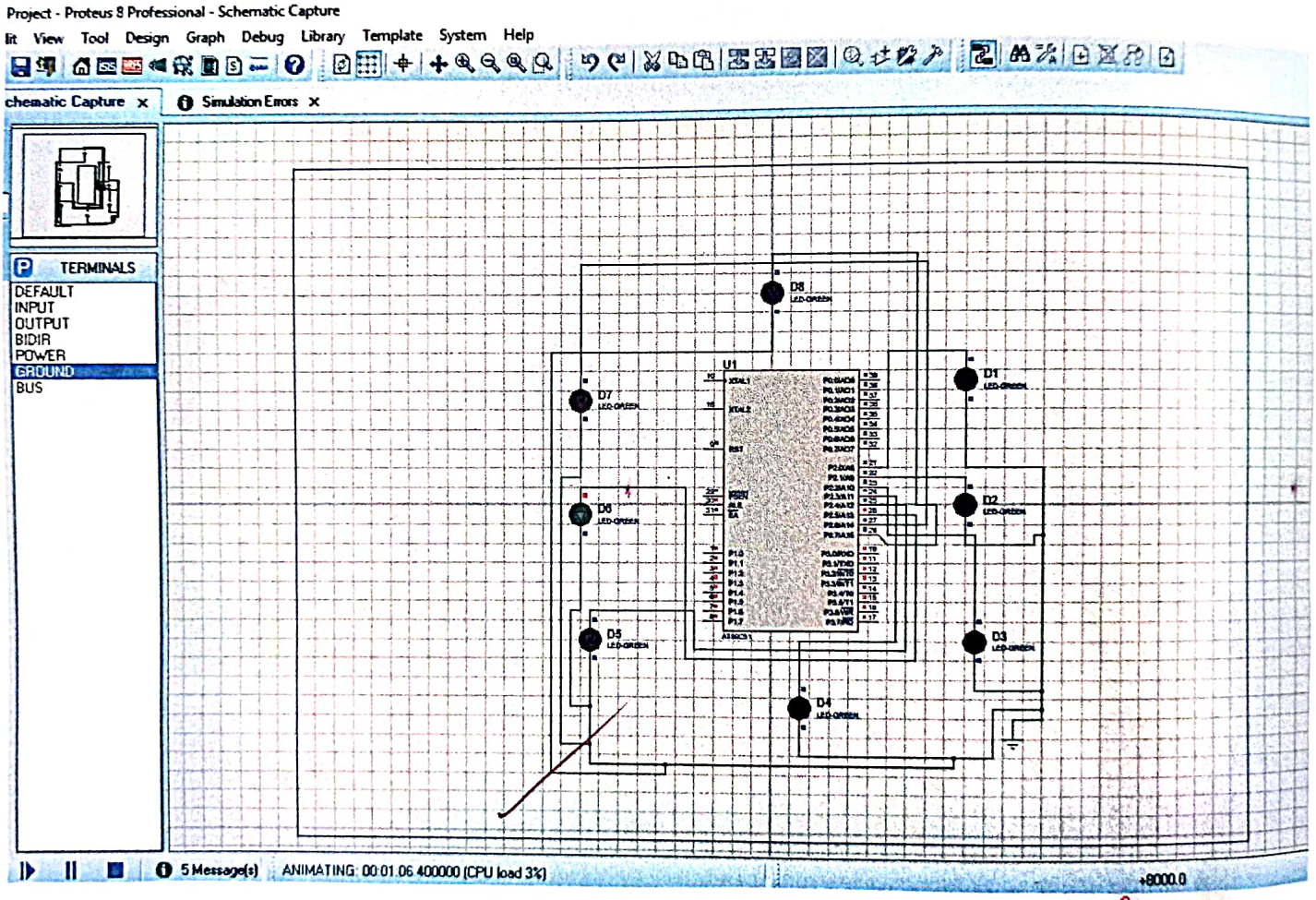
```
#include <reg51.h>
void delay()
{
int i;
for (i=0 ; i<30000 ; i++);
}
void main()
{
P2 = 0X01;
delay();
P2 = 0X02;
delay();
P2 = 0X04;
delay();
P2 = 0X08;
delay();
P2 = 0X80;
```

Sundaram

Teacher's Sign.: _____

practical 3 A] B]

Roll No.12



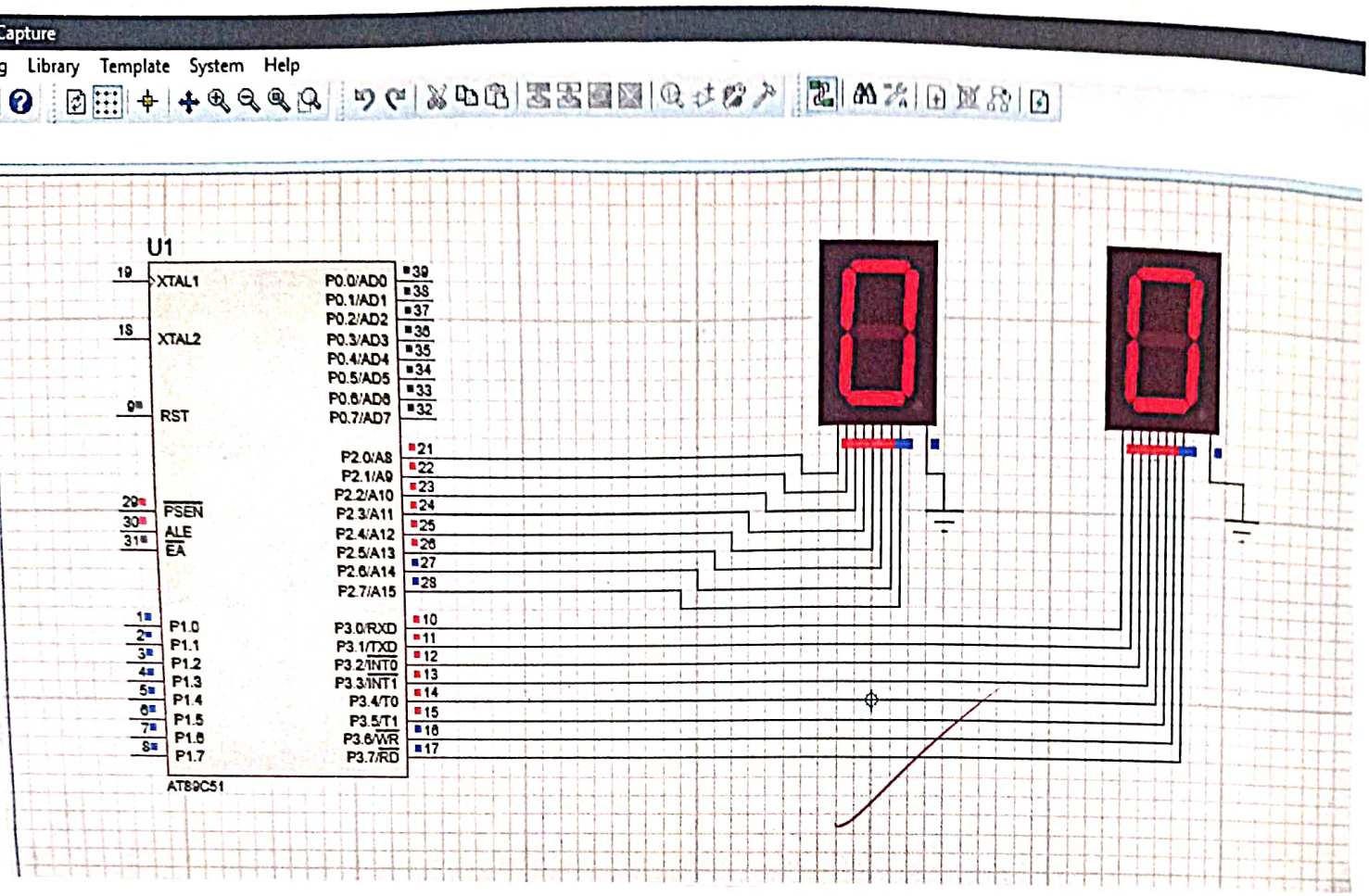
good.

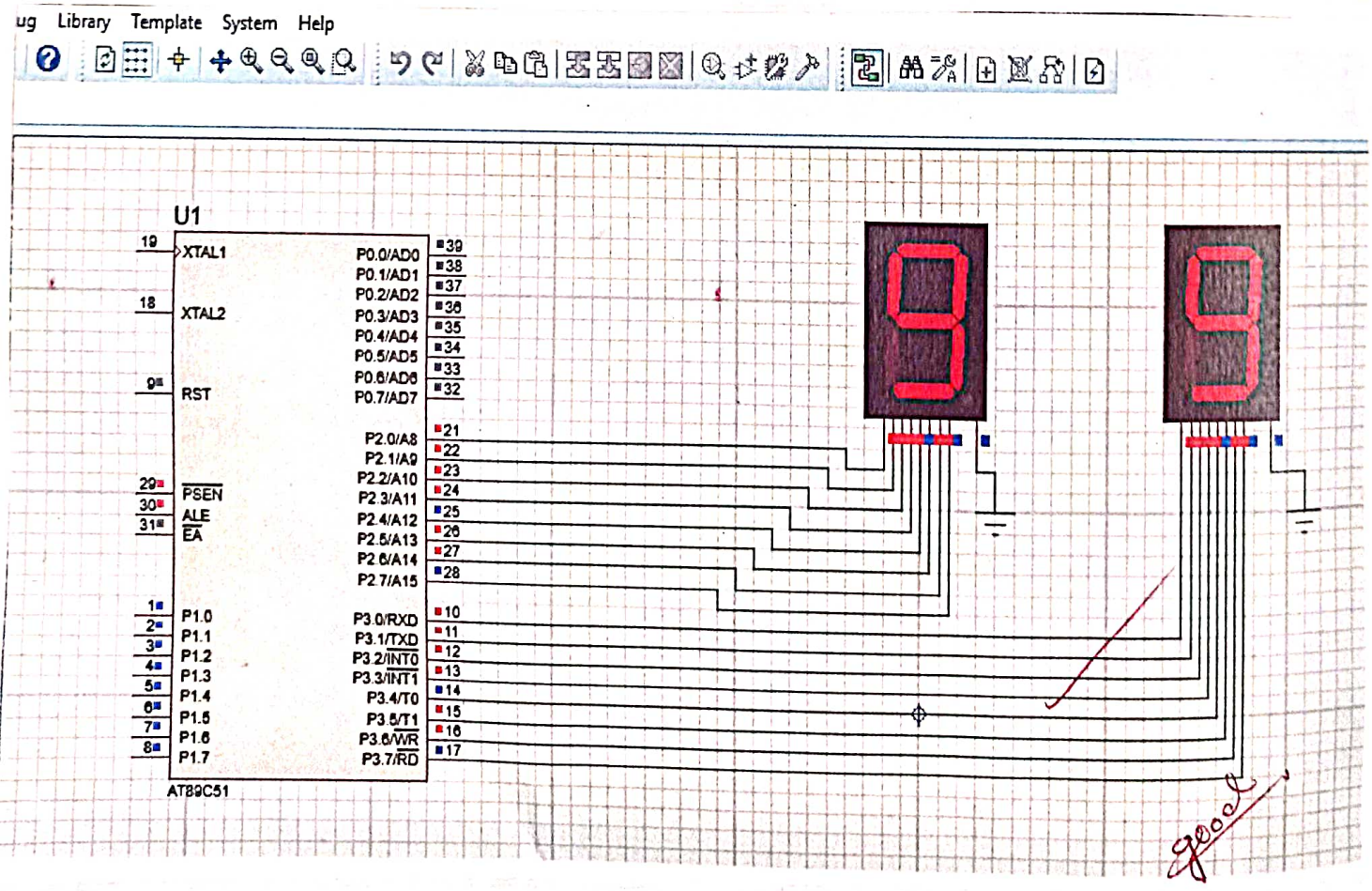
EXPERIMENT : No.

```
delay();
P2 = 0X40;
delay();
P2 = 0X20;
delay();
P2 = 0X10;
delay();
}
```

22/2/19

practical 4B]





EXPERIMENT: No. Practical NO. 4 PAGE No. DATE

4B] To demonstrate interfacing of 7 segment LED display and generate counting from 0 to 99 with fixed time delay.

→ * Components Used:

1. AT89C51
2. 7SEG-MPX1-CC

* Code:

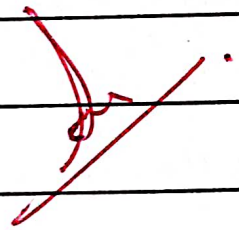
```
#include <reg51.h>
void delay(unsigned int ms)
{
unsigned int i,j;
for(i=0;i<ms;i++)
for(j=0;j<1275;j++);
}
void main(void)
{
char number[10]={0x3F, 0x06, 0x5B, 0x4f,
0x66, 0x6D, 0x7D, 0x07, 0x7F, 0x6F};
int i,j;
P2=0x00;
P3=0x00;
while(1)
{
for(i=0;i<=9;i++)
{
P2=number[i];
for(j=0;j<=9;j++)
{
```

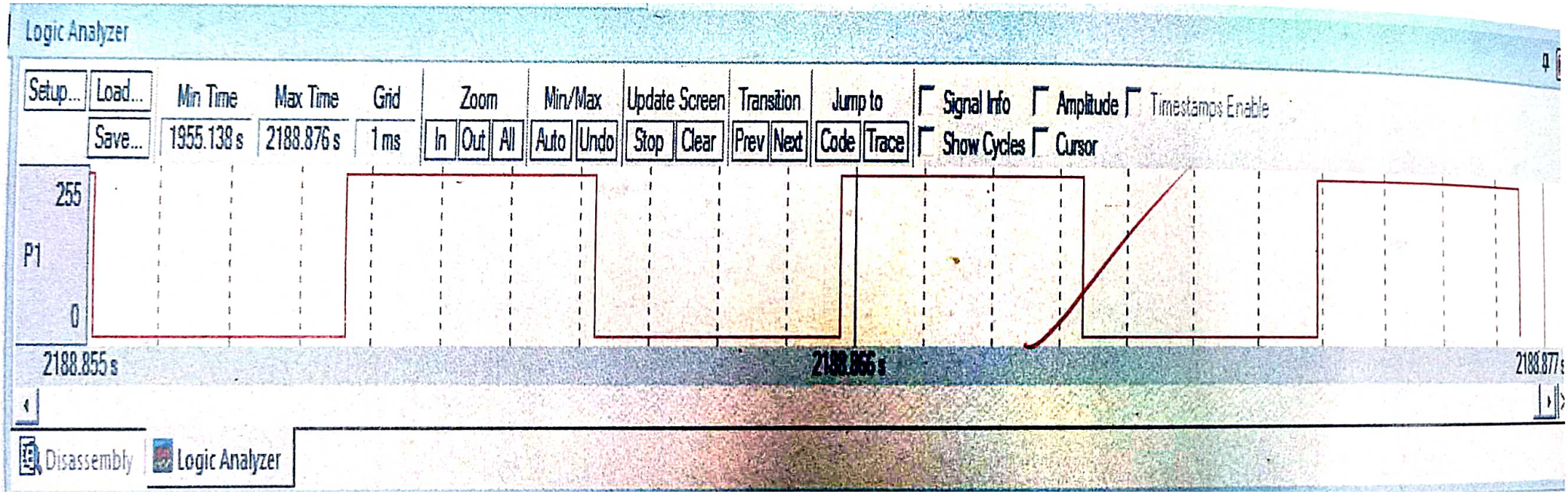
Sundaram

Teacher's Sign.:

EXPERIMENT : No.

```
P3 = number [j];
delay (50);
}
}
}
}
```




Logic Analyzer
Setup...
Load...
Save...
Min Time
1955.138 s
Max Time
2188.876 s
Grid
1 ms
Zoom
In
Out
All
Min/Max
Auto
Undo
Update Screen
Stop
Clear
Transition
Prev
Next
Jump to
Code
Trace
Signal Info
Amplitude
Timestamps Enable
Show Cycles
Cursor
255
P1
0
2188.855 s
2188.877 s
Disassembly
Logic Analyzer

EXPERIMENT : No. _____ PAGE No. _____ DATE _____

4C] Interface 8051 with D/A converter and generate square wave of given frequency.

→ * STEPS:

1. Open project tab
2. Click New µVision project
3. Save your project with name.
4. Search AT89C51 which is 8051 microcontroller.
5. Select AT89C51 → OK.
6. One dialog box will open → Click NO.
7. Open file tab → new file.
8. Type program
9. Save your program with .c extension.
10. + Target open
11. Right click on source group
12. Click on Add Existing file to...
13. choose your .c file
14. Click on add → close
15. Click on Translate
16. Project → Build target → for compiling
17. Right click target
18. Option for target
19. Change Frequency 11.0592
20. Click on checkbox ☑ Use on chip ROM
21. Select output tab → click on ☑ Create Hex file → OK
22. Select Translate
23. Select build
24. Click on debugger
25. Click logic Analyser

Sundaram

Teacher's Sign.: _____

EXPERIMENT : No.

PAGE No.
DATE

26. setup in Logic Analyser [setup]
27. Write P1 and select P1 then close.
28. Click on debug.
29. Select RUN

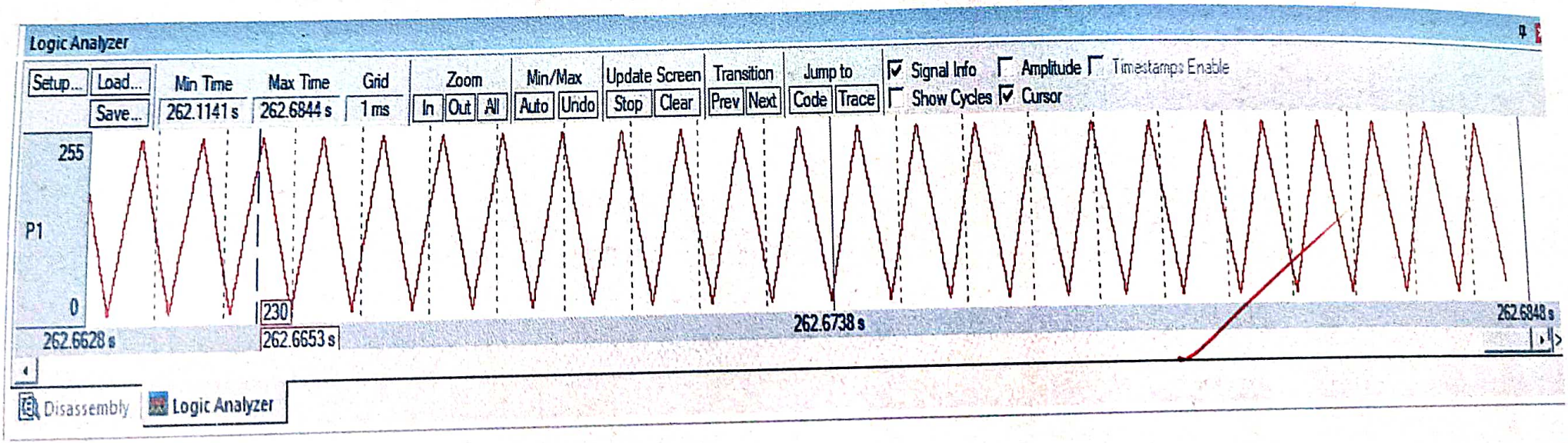
* Code :

```
#include <reg51.h>
void delay();
void main()
{
while(1)
{
P1 = 0xff;
delay();
P1 = 0x00;
delay();
}
}
void delay()
{
unsigned int i, j, k;
for (i=0; i<10; i++);
for (j=0; j<200 ; j++);
for (k=0; k<300; k++);
}
```

28/2/19

Teacher's Sign.:

practical 5A]



EXPERIMENT : No. Practical NO.5

PAGE No.
DATE

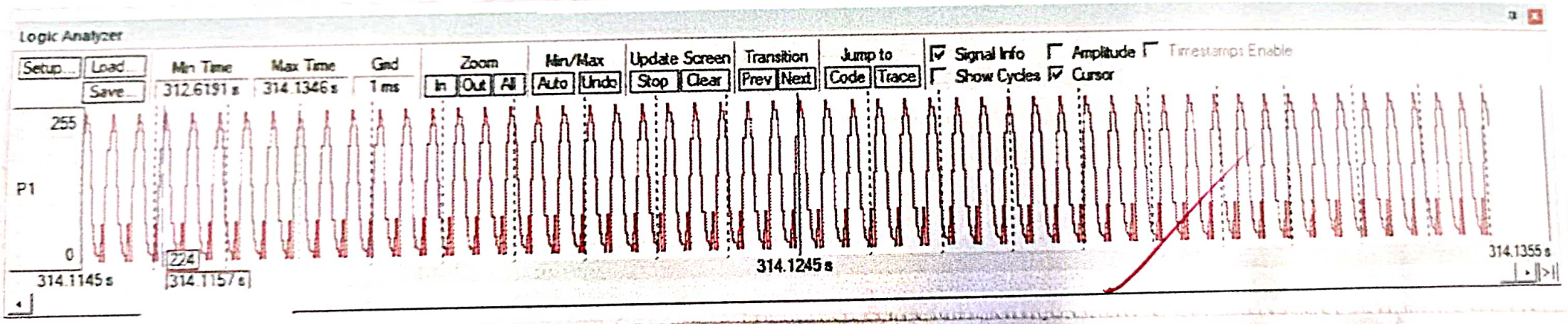
5A] Interface 8051 with D/A Converter and generate triangular wave of given frequency on oscilloscope.

→ * Code :

```
#include <reg51.h>
void main (void)
{
P1 = 0x00;
while (1)
{
P1 += 0x05;
}
while (P1 < 0xFF);
do
{
P1 -= 0x05;
}
while (P1 > 0x00);
}
}
```

28/2/19
good

practical 5B]



EXPERIMENT : No.

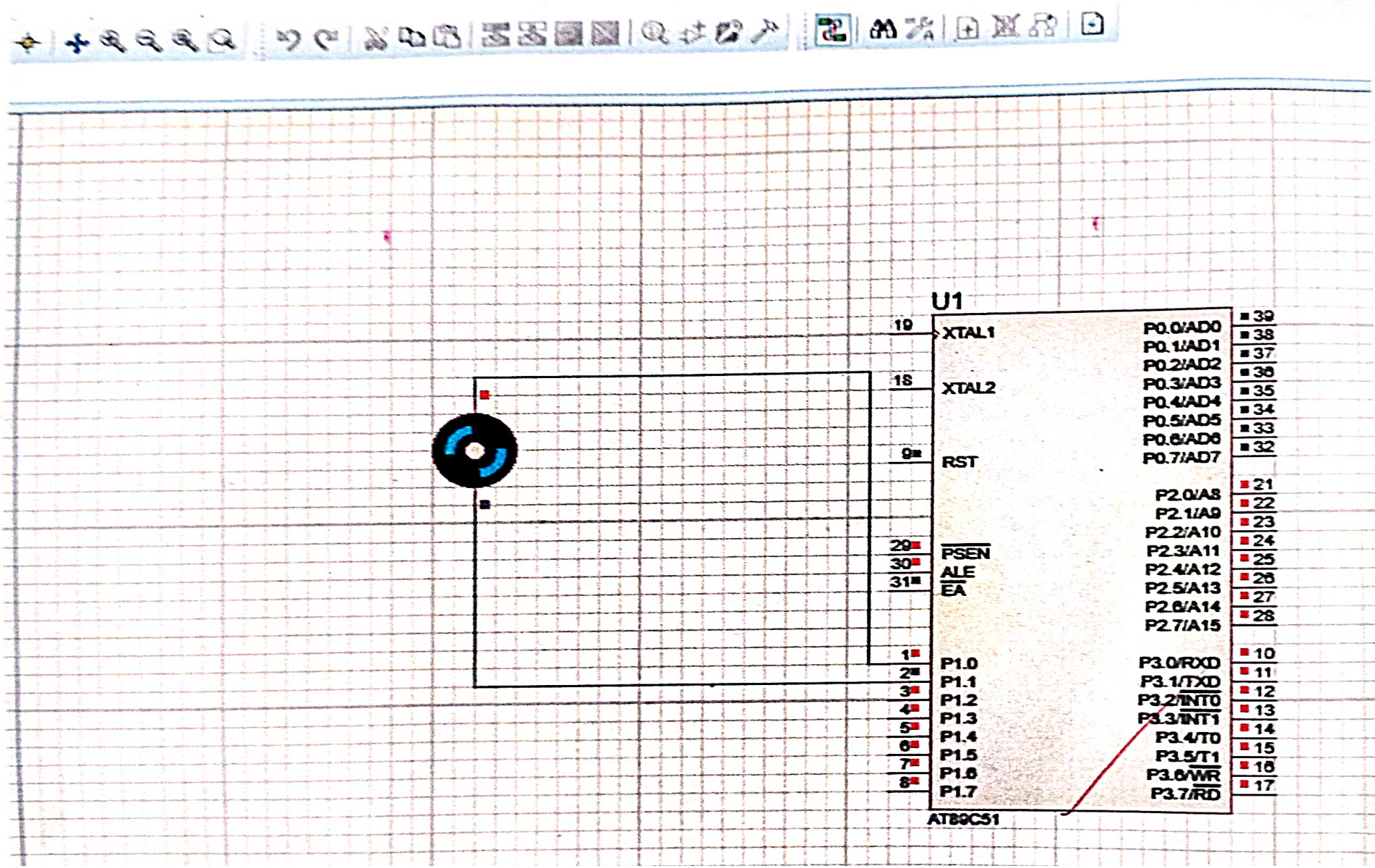
PAGE No.
DATE

5B] Using D/A converter generate sine wave on oscilus scope with help of look up table stored in data area of 8051

→ * Code:

```
#include <regs1.h>
#include <intrins.h>
void main()
{
unsigned i;
unsigned int wave_value[16] = {128, 192, 224, 255,
240, 224, 192, 128, 64, 32, 16, 10, 0, 32, 64};
while(1)
{
for(i=0; i<16; i++)
{
P1 = wave_value[i];
_nop_();
_nop_();
_nop_();
_nop_();
_nop_();
_nop_();
}
}
}
```

practical 6


U1
19
XTAL1
18
XTAL2
9
RST
29
PSEN
30
ALE
31
EA
1
P1.0
2
P1.1
3
P1.2
4
P1.3
5
P1.4
6
P1.5
7
P1.6
8
P1.7
P0.0/AD0
P0.1/AD1
P0.2/AD2
P0.3/AD3
P0.4/AD4
P0.5/AD5
P0.6/AD6
P0.7/AD7
39
38
37
36
35
34
33
32
P2.0/A8
P2.1/A9
P2.2/A10
P2.3/A11
P2.4/A12
P2.5/A13
P2.6/A14
P2.7/A15
21
22
23
24
25
26
27
28
P3.0/RXD
P3.1/TXD
P3.2/INT0
P3.3/INT1
P3.4/T0
P3.5/T1
P3.6/WR
P3.7/RD
10
11
12
13
14
15
16
17
AT89C51

EXPERIMENT : No.

Practical NO. 6

PAGE No.
DATE

Interface stepper motor with 8051 and write a program to move the motor through a given angle in clockwise or counter clockwise direction

→ * Component :

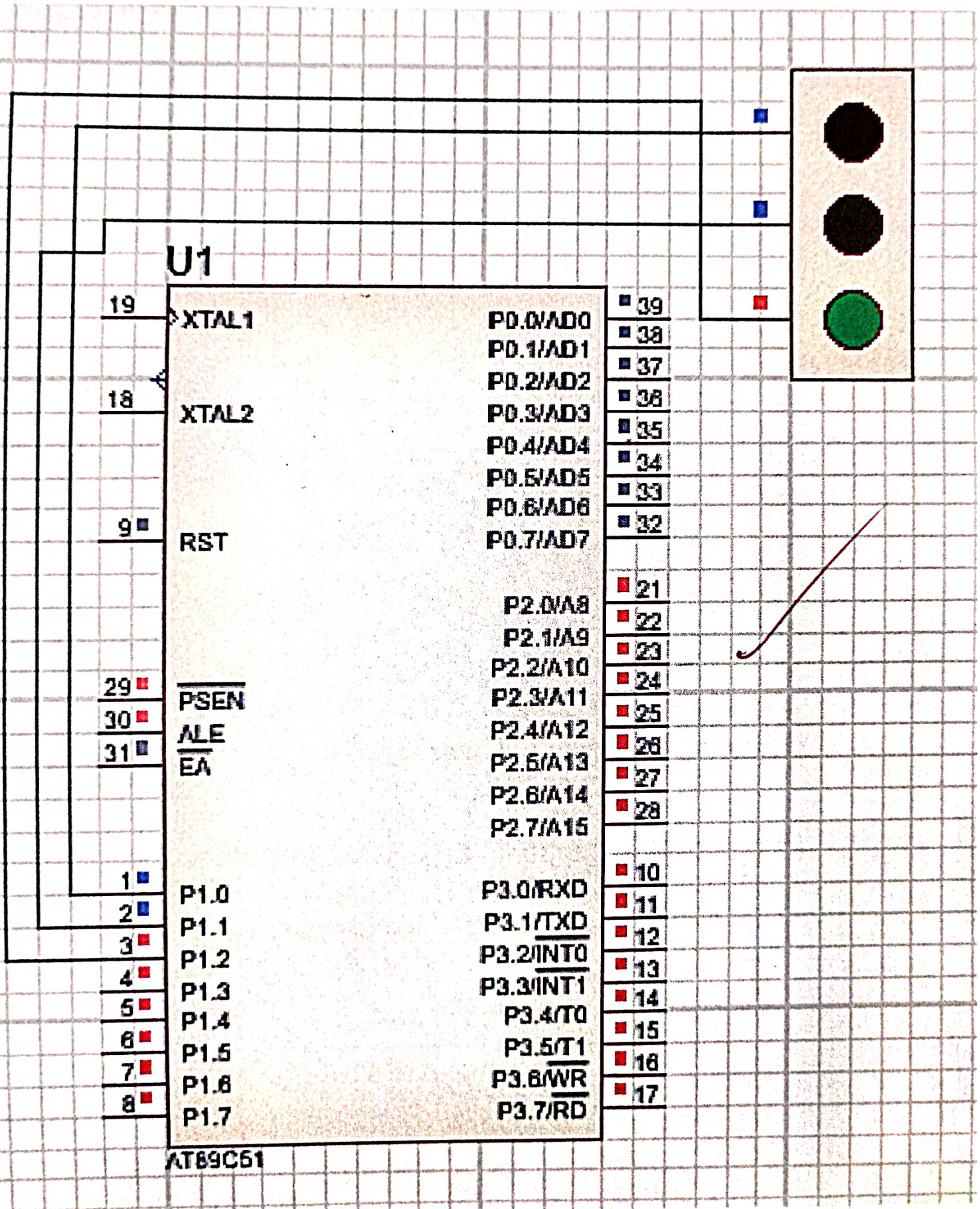
1. AT89C51
2. DC Motor Active simple DC motor

* code :

```
#include <reg51.h>
sbit motp = P1^0;
sbit motn = P1^1;
void main()
{
unsigned int i;
motp = motn =0;
while (1)
{
motp=1;
motn=0;
for(i=0;i<60000;i++);
motp=0;
motn=1;
for(i=0; i<60000 ; i++);
}
}
```

28/2/19

practical 7


U1
19
XTAL1
18
XTAL2
9
RST
29
PSEN
30
ALE
31
EA
1
P1.0
2
P1.1
3
P1.2
4
P1.3
5
P1.4
6
P1.5
7
P1.6
8
P1.7
AT89C51
P0.0/AD0
P0.1/AD1
P0.2/AD2
P0.3/AD3
P0.4/AD4
P0.5/AD5
P0.6/AD6
P0.7/AD7
39
38
37
36
35
34
33
32
P2.0/A8
P2.1/A9
P2.2/A10
P2.3/A11
P2.4/A12
P2.5/A13
P2.6/A14
P2.7/A15
21
22
23
24
25
26
27
28
P3.0/RXD
P3.1/TXD
P3.2/INT0
P3.3/INT1
P3.4/T0
P3.5/T1
P3.6/WR
P3.7/RD
10
11
12
13
14
15
16
17

EXPERIMENT : No.

Practical NO·7

PAGE No.
DATE

Generate traffic signal.

→ * component :

1. AT89C51
2. Traffic Signal

* code :

```
#include <reg51.h>
Sbit red = P1^0;
Sbit yellow = P1^1;
sbit green = P1^2;
 void main()
{
unsigned int i;
red = yellow = green = 0;
while(1)
{
red = 1;
for (i=0; i<60000; i++);
red = 0;
yellow = 1;
for( i=0; i<60000; i++);
yellow = 0;    green = 1;
for (i=0; i<60000; i++);
for(i=0; i<60000; i++);
green = 0;
}
}
```

28/2/19
V. good.

Teacher's Sign.: