**Report**

**(Day 3)**

Reported by: P. PRUDVI REDDY (BU21EECE0100361)

Mail Id: [ppoli@gitam.in](mailto:ppoli@gitam.in)

**Objective:**

* To design and simulate a system using the AT89C51 microcontroller to control the sequential blinking of LEDs, implemented and tested in Proteus. The project involves creating hex files using normal, loop, and array implementation methods for LED control.

Inputs:

* Where the AT89C51 pins from the P1.0 to P1.7 act as the inputs for the design where having the hidden power supply and Hex files created from keil uvision from the n normal, loop, and array c program inserted to the AT89C51

Outputs:

* Where the LED’s connected to the pins from P1.0 to P1.7 of the Microcontroller with a common ground to the LED’s act as the output

**Logic:**

Normal:

 The main function continuously loops to maintain the LED blinking sequence.

 Sequentially, each LED connected to port P1 is turned on for a short duration.

 The LEDs are turned on one by one in a sequence: P1.0, P1.1, P1.2, ..., P1.7.

 Each LED is illuminated for 50 milliseconds, controlled by the delay function.

 The delay function creates a delay of approximately 50 milliseconds by executing nested loops.

 The value assigned to P1 corresponds to the binary representation of the LEDs being turned on.

 Once all LEDs have been illuminated, the sequence restarts from the first LED.

For loop:

 The main function initializes a variable k to 0x01, representing the first LED.

 Inside an infinite loop, a for loop iterates through the values from 0 to 7.

 At each iteration, the value of k is assigned to port P1, turning on the corresponding LED.

 A delay of 50 milliseconds is introduced to keep the LED illuminated briefly.

 After each iteration, k is left-shifted by one bit to move to the next LED.

 Once all LEDs are illuminated, k is reset to 0x01, restarting the sequence.

Array:

 The main function initializes an array A with hexadecimal values representing the LEDs connected to port P1.

 Inside an infinite loop, a for loop iterates through the elements of array A.

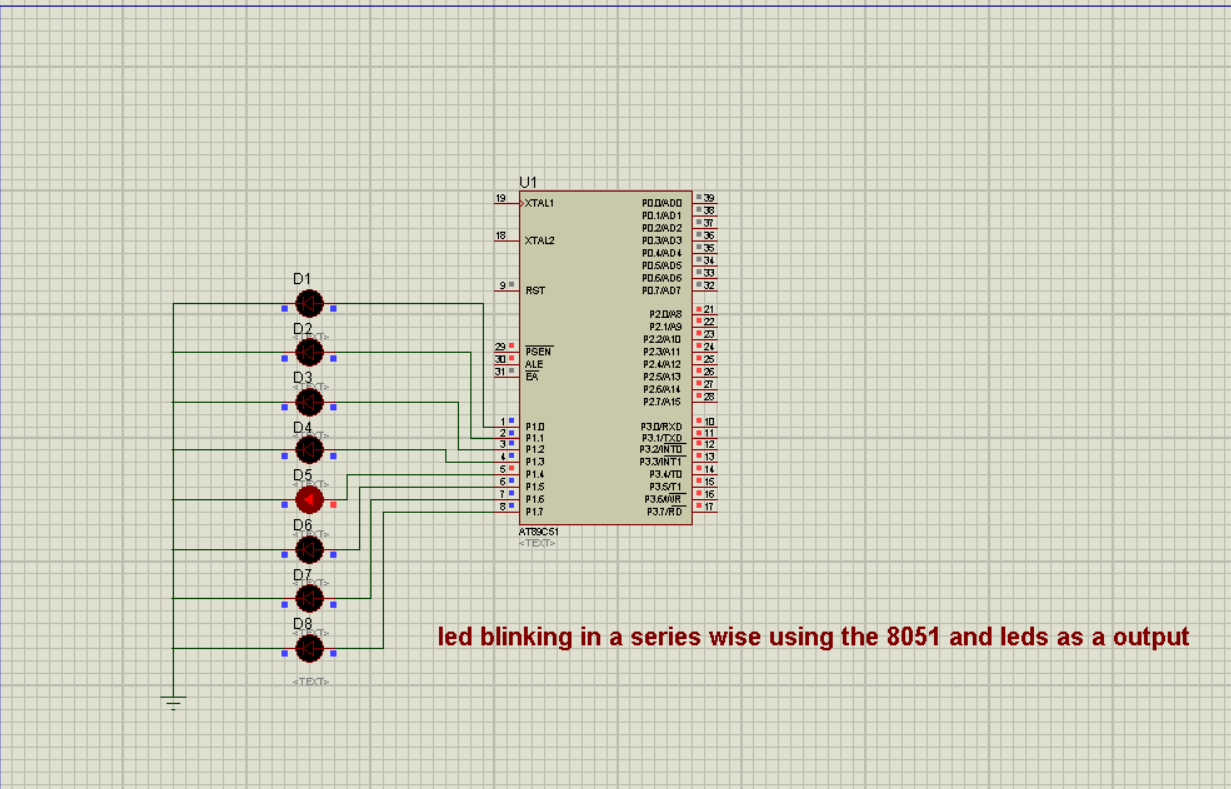
 At each iteration, the value of the current element A[k] is assigned to port P1, turning on the corresponding LED.

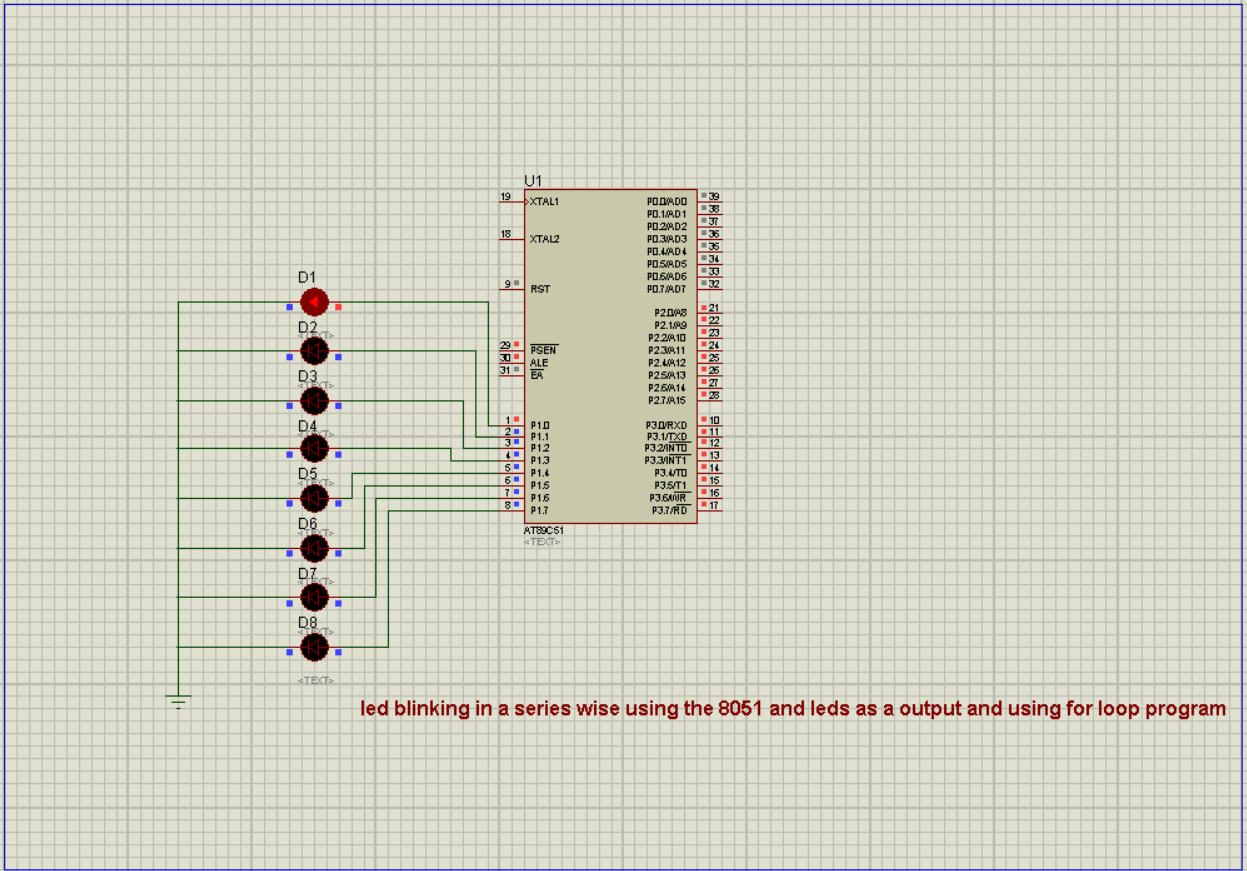
 A delay of 50 milliseconds is introduced to keep the LED illuminated briefly.

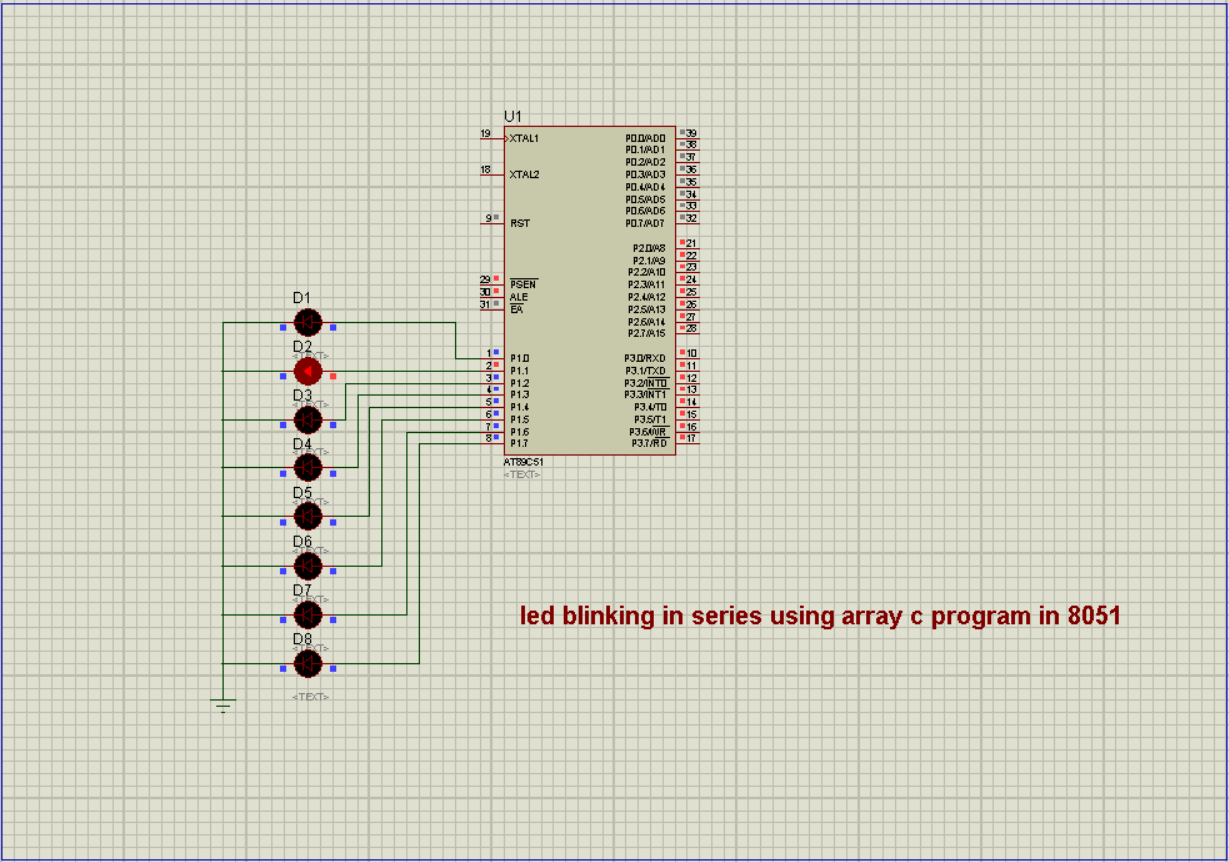
 After each iteration, the next element in the array is accessed to control the next LED.

 Once all LEDs are illuminated, the loop restarts from the beginning of the array.

Results:







Code:

* Normal:

#include <reg51.h>

void delay(unsigned int);

void main(void)

{

while(1)

{

P1=0x01;

delay(50);

P1=0x02;

delay(50);

P1=0x04;

delay(50);

P1=0x08;

delay(50);

P1=0x10;

delay(50);

P1=0x20;

delay(50);

P1=0x40;

delay(50);

P1=0x80;

delay(50);

}

}

void delay(unsigned int t)

{

unsigned int i,j;

for(i=0;i<t;i++)

for(j=0;j<1275;j++); // For 1ms, for loop need to iterate 1275 times.

}

* For loop:

#include<reg51.h>

void delay(unsigned int);

void main(void)

{

unsigned char k = 0x01;

unsigned int n;

while(1)

{

for( n=0;n<=7;n++){

P1 = k;

delay(50);

k=k<<1;

}

if(k==0){

k=0x01;

}

}

}

void delay(unsigned int t)

{

unsigned int i,j;

for(i=0;i<t;i++)

for(j=0;j<1275;j++);//1275 8051 will compute 1275 tiems in 1millisecond

}

* Array:

#include <reg51.h>

#include <stdio.h>

void delay(unsigned int);

void main(void)

{

unsigned int A[8] = {0x01,0x02,0x04,0x08,0x10,0x20,0x40,0x80};

int k=0;

while(1)

{

for(k=0; k<=7; k++)

{

P1=A[k];

delay(50);

}

}

}

void delay(unsigned int t)

{

unsigned int i,j;

for(i=0;i<t;i++)

for(j=0;j<1275;j++); // For 1ms, for loop need to iterate 1275 times.

}

Common mistakes How do I overcome:

* the common mistakes I made during this design that is encountered during the LED blinking series design process is difficulty in locating components and connecting wires within Proteus. This can be particularly challenging for those who are new to using Proteus for circuit design
* the solution that I overcome of this problem is explore tutorials or guides on the basics of Proteus. These resources can provide insights into how components are connected and how to navigate the software effectively. By familiarizing yourself with Proteus's interface and functionalities, you'll become more adept at finding components and establishing connections between them.