 **B.N.M. Institute of Technology**

**An Autonomous Institution under VTU, Approved by AICTE**

**Department of Information Science and Engineering**

**LED Toggle Using Button**

**Course Name: Microcontroller and Embedded System**

**Course Code: 23ISE142**

In embedded systems, one common task is to control LEDs using user inputs. This project explores how to control multiple LEDs using a button on the LPC2148 microcontroller. Specifically, it demonstrates how to toggle multiple LEDs simultaneously with a button press, where the LEDs can be turned on in random combinations.

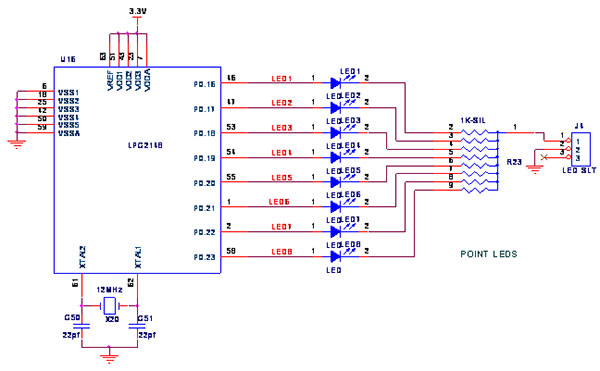
The project involves configuring GPIO pins on the LPC2148 to manage both the input (button) and output (LEDs). When the button is pressed, the system will randomly turn on a combination of LEDs connected to specific pins on the microcontroller. This demonstrates fundamental concepts of GPIO pin configuration, button reading, and random number generation in embedded programming.

**Working**

The LED Toggle Using Button works as follows:

1. Hardware Setup:
   * LEDs (P0.10 to P0.15): These pins are configured as output pins and are connected to LEDs.
   * Button (P0.14): This pin is configured as an input and is connected to a push button. The button is assumed to be active-low (i.e., when pressed, it connects the pin to ground).
2. GPIO Initialization:
   * The LED pins (P0.10 to P0.15) are set as outputs to control the LEDs.
   * The button pin (P0.14) is set as an input to read the button's state.
   * The GPIO direction registers (IODIR0) are used to configure the pins.
3. Button Press Detection:
   * The program continuously checks if the button is pressed. Since the button is active-low, the state of the button pin is read using the IOPIN0 register. If the pin value is low, it indicates that the button is pressed.
4. Random LED control:
   * When the button is pressed, a simple pseudo-random number generator is used to generate a random pattern for the LEDs.
   * The random number generator outputs a value between 0 and 63, which corresponds to a 6-bit value, with each bit representing the state of an LED (ON or OFF).
   * This value is used to determine which LEDs (P0.10 to P0.15) should be turned on. For example, if the random value is 0x3F, all 6 LEDs will be turned on; if the random value is 0x15, LEDs at positions P0.10, P0.12, and P0.14 will be on.
5. LED update:
   * The LED pins are updated based on the random value generated. The IOSET0 and IOCLR0 registers are used to set and clear individual bits (LEDs), turning them on or off. This ensures that the LEDs are controlled in the random combination as determined by the generated number.
6. Debouncing:
   * A simple debounce mechanism is implemented using a delay function to avoid multiple triggers from a single button press due to mechanical bouncing of the button. This ensures that only one random LED pattern is generated per press.
7. Button Release:
   * After a random LED combination is set, the program waits for the button to be released. This ensures that a new random pattern is generated only after the button is released and pressed again.

**Circuit Diagram of LED toggle using button with LPC2148**

****

The circuit for the LED Toggle Using Button project connects six LEDs to P0.10 to P0.15 of the LPC2148 microcontroller, with each LED controlled by the microcontroller's output pins through current-limiting resistors. A push button is connected to P0.14 as an input, using a pull-down configuration to detect button presses. When the button is pressed, the microcontroller generates a random pattern to turn on a combination of LEDs, and the LEDs are turned on or off based on this random pattern. A simple debounce mechanism ensures reliable detection of button presses.

**Program code:**

#include <lpc214x.h>

#define BUTTON\_PIN 14

#define LED\_MASK (0x3F << 10) // P0.10 to P0.15

void delay\_ms(unsigned int ms) {

for (unsigned int i = 0; i < ms \* 1000; i++) {

\_\_asm volatile ("NOP");

}

}

void init\_gpio() {

IODIR0 |= LED\_MASK; // Set P0.10 to P0.15 as output

IODIR0 &= ~(1 << BUTTON\_PIN); // Button as input

}

// Simple pseudo-random generator (linear congruential)

unsigned int rand\_seed = 0xACE1;

unsigned int random\_bits() {

rand\_seed = (rand\_seed \* 1103515245 + 12345) & 0xFFFFFFFF;

return (rand\_seed >> 10) & 0x3F; // Random bits for 6 LEDs

}

int is\_button\_pressed() {

return (IOPIN0 & (1 << BUTTON\_PIN)) == 0;

}

int main() {

init\_gpio();

int prevButtonState = 1;

while (1) {

int currentButtonState = is\_button\_pressed();

if (prevButtonState == 1 && currentButtonState == 0) {

delay\_ms(50); // Debounce

if (is\_button\_pressed()) {

unsigned int led\_pattern = random\_bits();

IOCLR0 = LED\_MASK; // Turn off all LEDs

IOSET0 = (led\_pattern << 10); // Set random LED pattern

}

while (is\_button\_pressed()); // Wait for button release

delay\_ms(50); // Debounce

}

prevButtonState = currentButtonState;

}

return 0;

}

**Output:** On each press of the button connected to P0.14, a random combination of LEDs connected to P0.10–P0.15 lights up .



**Student Name:** Ankitha S A

**USN:** 1BG23IS007

**Student Name:** C M Kavana

**USN:** 1BG23IS010

**Date:**