To toggle LED

```
int main(void)
{
	GPIO_Handle_t GPIOLed;

	  GPIOLed.pGPIOx = GPIOD;

	  //Toggle Green LED connected to PD12

	  GPIOLed.GPIO_PinConfig.GPIO_PinNumber = GPIO_PIN_NO_12;

	  GPIOLed.GPIO_PinConfig.GPIO_PinMode = GPIO_MODE_OUT;

	  GPIOLed.GPIO_PinConfig.GPIO_PinSpeed = GPIO_SPEED_FAST;

	  GPIOLed.GPIO_PinConfig.GPIO_PinOPType = GPIO_OP_TYPE_PP;

	  GPIOLed.GPIO_PinConfig.GPIO_PinPuPdCOntrol = GPIO_NO_PUPD;



	  //Enable the clock for GPIOD Peripheral

	  GPIO_PeriClockControl(GPIOD,ENABLE);



	  //Initializing the GPIO Peripheral



	  GPIO_Init(&GPIOLed);
	  /* Loop forever */

		while(1){

			GPIO_ToggleOutputPin(GPIOD,GPIO_PIN_NO_12);

			delay();

		}



}
void delay(){
```

```
        for (uint32_t i=0;i<500000;i++);

}
```

To TURN On all LEDS

```
int main(void)
{
GPIO_Handle_t GPIOLed;

GPIOLed.pGPIOx = GPIOD;

//Toggle Green LED connected to PD12


GPIOLed.GPIO_PinConfig.GPIO_PinMode = GPIO_MODE_OUT;

GPIOLed.GPIO_PinConfig.GPIO_PinSpeed = GPIO_SPEED_FAST;

GPIOLed.GPIO_PinConfig.GPIO_PinOPType = GPIO_OP_TYPE_PP;

GPIOLed.GPIO_PinConfig.GPIO_PinPuPdCOntrol = GPIO_NO_PUPD;

GPIOLed.GPIO_PinConfig.GPIO_PinNumber = GPIO_PIN_NO_12;
GPIO_Init(&GPIOLed);

GPIOLed.GPIO_PinConfig.GPIO_PinNumber = GPIO_PIN_NO_13;
GPIO_Init(&GPIOLed);

GPIOLed.GPIO_PinConfig.GPIO_PinNumber = GPIO_PIN_NO_14;
GPIO_Init(&GPIOLed);

GPIOLed.GPIO_PinConfig.GPIO_PinNumber = GPIO_PIN_NO_15;
GPIO_Init(&GPIOLed);


//Enable the clock for GPIOD Peripheral

GPIO_PeriClockControl(GPIOD,ENABLE);


/* Loop forever */

// Turn ON all LEDs using GPIO_WriteToOutputPin()
GPIO_WriteToOutputPin(GPIOD, GPIO_PIN_NO_12, GPIO_PIN_SET);
GPIO_WriteToOutputPin(GPIOD, GPIO_PIN_NO_13, GPIO_PIN_SET);
GPIO_WriteToOutputPin(GPIOD, GPIO_PIN_NO_14, GPIO_PIN_SET);
GPIO_WriteToOutputPin(GPIOD, GPIO_PIN_NO_15, GPIO_PIN_SET);

while(1);
```

```
}
```

1. Turn LEDs ON/OFF: Write a program to control 4 LEDs connected to GPIO pins. Implement a function void controlLED( ) that turns the specified LED ON (true) or OFF (false).

```
void controlLED(uint8_t LedNumber, bool state);
int main(void)
{
GPIO_Handle_t GPIOLed;

GPIOLed.pGPIOx = GPIOD;

//Toggle Green LED connected to PD12


GPIOLed.GPIO_PinConfig.GPIO_PinMode = GPIO_MODE_OUT;

GPIOLed.GPIO_PinConfig.GPIO_PinSpeed = GPIO_SPEED_FAST;

GPIOLed.GPIO_PinConfig.GPIO_PinOPType = GPIO_OP_TYPE_PP;

GPIOLed.GPIO_PinConfig.GPIO_PinPuPdCOntrol = GPIO_NO_PUPD;

GPIOLed.GPIO_PinConfig.GPIO_PinNumber = GPIO_PIN_NO_12;
GPIO_Init(&GPIOLed);

GPIOLed.GPIO_PinConfig.GPIO_PinNumber = GPIO_PIN_NO_13;
GPIO_Init(&GPIOLed);

GPIOLed.GPIO_PinConfig.GPIO_PinNumber = GPIO_PIN_NO_14;
GPIO_Init(&GPIOLed);

GPIOLed.GPIO_PinConfig.GPIO_PinNumber = GPIO_PIN_NO_15;
GPIO_Init(&GPIOLed);


//Enable the clock for GPIOD Peripheral

GPIO_PeriClockControl(GPIOD,ENABLE);

controlLED(GPIO_PIN_NO_12, true);
controlLED(GPIO_PIN_NO_13, false);
controlLED(GPIO_PIN_NO_14, true);
controlLED(GPIO_PIN_NO_15, false);
delay();
controlLED(GPIO_PIN_NO_12, false);
controlLED(GPIO_PIN_NO_13, true);
controlLED(GPIO_PIN_NO_14, false);
controlLED(GPIO_PIN_NO_15, true);
delay();
}

void controlLED(uint8_t LedNumber, bool state){
if(state){
GPIO_WriteToOutputPin(GPIOD, LedNumber, GPIO_PIN_SET);
```

```
}else{
GPIO_WriteToOutputPin(GPIOD, LedNumber, GPIO_PIN_RESET);
}
}
void delay(){
for (uint32_t i=0;i<500000;i++);

}
```

2. Blink LEDs in Sequence: Write a program that blinks the 4 LEDs in sequence (LED1 -> LED2 -> LED3 -> LED4) with a delay between each. After LED4, the sequence should repeat.

```
void controlLED(uint8_t LedNumber, bool state);
int main(void)
{
GPIO_Handle_t GPIOLed;

GPIOLed.pGPIOx = GPIOD;

//Toggle Green LED connected to PD12


GPIOLed.GPIO_PinConfig.GPIO_PinMode = GPIO_MODE_OUT;

GPIOLed.GPIO_PinConfig.GPIO_PinSpeed = GPIO_SPEED_FAST;

GPIOLed.GPIO_PinConfig.GPIO_PinOPType = GPIO_OP_TYPE_PP;

GPIOLed.GPIO_PinConfig.GPIO_PinPuPdCOntrol = GPIO_NO_PUPD;

GPIOLed.GPIO_PinConfig.GPIO_PinNumber = GPIO_PIN_NO_12;
GPIO_Init(&GPIOLed);

GPIOLed.GPIO_PinConfig.GPIO_PinNumber = GPIO_PIN_NO_13;
GPIO_Init(&GPIOLed);

GPIOLed.GPIO_PinConfig.GPIO_PinNumber = GPIO_PIN_NO_14;
GPIO_Init(&GPIOLed);

GPIOLed.GPIO_PinConfig.GPIO_PinNumber = GPIO_PIN_NO_15;
GPIO_Init(&GPIOLed);


//Enable the clock for GPIOD Peripheral

GPIO_PeriClockControl(GPIOD,ENABLE);
while(1){
controlLED(GPIO_PIN_NO_12, true);
delay();
controlLED(GPIO_PIN_NO_12, false);
controlLED(GPIO_PIN_NO_13, true);
delay();
controlLED(GPIO_PIN_NO_13, false);
controlLED(GPIO_PIN_NO_14, true);
```

```
delay();
controlLED(GPIO_PIN_NO_14, false);
controlLED(GPIO_PIN_NO_15, true);
delay();
controlLED(GPIO_PIN_NO_15, false);
}
}

void controlLED(uint8_t LedNumber, bool state){
if(state){
GPIO_WriteToOutputPin(GPIOD, LedNumber, GPIO_PIN_SET);
}else{
GPIO_WriteToOutputPin(GPIOD,LedNumber, GPIO_PIN_RESET);
}
}
void delay(){
for (uint32_t i=0;i<500000;i++);

}
```

3. Binary Counter with LEDs: Implement a binary counter using the 4 LEDs. Starting from 0000 (all OFF), increment the count every second, displaying the binary representation of the counter on the LEDs (ON = 1, OFF = 0).

```
void blinkLED(uint8_t counter);
int main(void)
{
GPIO_Handle_t GPIOLed;

GPIOLed.pGPIOx = GPIOD;

//Toggle Green LED connected to PD12


GPIOLed.GPIO_PinConfig.GPIO_PinMode = GPIO_MODE_OUT;

GPIOLed.GPIO_PinConfig.GPIO_PinSpeed = GPIO_SPEED_FAST;

GPIOLed.GPIO_PinConfig.GPIO_PinOPType = GPIO_OP_TYPE_PP;

GPIOLed.GPIO_PinConfig.GPIO_PinPuPdCOntrol = GPIO_NO_PUPD;

GPIOLed.GPIO_PinConfig.GPIO_PinNumber = GPIO_PIN_NO_12;
GPIO_Init(&GPIOLed);

GPIOLed.GPIO_PinConfig.GPIO_PinNumber = GPIO_PIN_NO_13;
GPIO_Init(&GPIOLed);

GPIOLed.GPIO_PinConfig.GPIO_PinNumber = GPIO_PIN_NO_14;
GPIO_Init(&GPIOLed);

GPIOLed.GPIO_PinConfig.GPIO_PinNumber = GPIO_PIN_NO_15;
```

```
GPIO_Init(&GPIOLed);


//Enable the clock for GPIOD Peripheral

GPIO PeriClockControl(GPIOD,ENABLE);
while(1){
uint8_t counter = 0;
blinkLED(counter);
counter++;
if(counter == 15){
counter = 0;
}
delay();
}
}


void blinkLED(uint8_t counter){
GPIO_WriteToOutputPin(GPIOD, GPIO_PIN_NO_12, (counter &
(1<<0))?GPIO_PIN_SET:GPIO_PIN_RESET);
GPIO_WriteToOutputPin(GPIOD, GPIO_PIN_NO_12, (counter &
(1<<1))?GPIO_PIN_SET:GPIO_PIN_RESET);
GPIO_WriteToOutputPin(GPIOD, GPIO_PIN_NO_12, (counter &
(1<<2))?GPIO_PIN_SET:GPIO_PIN_RESET);
GPIO_WriteToOutputPin(GPIOD, GPIO_PIN_NO_12, (counter &
(1<<3))?GPIO_PIN_SET:GPIO_PIN_RESET);
}

void delay(){
for (uint32_t i=0;i<500000;i++);

}
```

4. Alternate Blinking: Create a program that makes LED1 and LED3 blink alternately with LED2 and LED4, each group toggling every second.

```
void controlLED(uint8_t LedNumber, bool state);
int main(void)
{
GPIO_Handle_t GPIOLed;

GPIOLed.pGPIOx = GPIOD;

//Toggle Green LED connected to PD12


GPIOLed.GPIO_PinConfig.GPIO_PinMode = GPIO_MODE_OUT;

GPIOLed.GPIO_PinConfig.GPIO_PinSpeed = GPIO_SPEED_FAST;

GPIOLed.GPIO_PinConfig.GPIO_PinOPType = GPIO_OP_TYPE_PP;
```

```
GPIOLed.GPIO_PinConfig.GPIO_PinPuPdCOntrol = GPIO_NO_PUPD;

GPIOLed.GPIO_PinConfig.GPIO_PinNumber = GPIO_PIN_NO_12;
GPIO_Init(&GPIOLed);

GPIOLed.GPIO_PinConfig.GPIO_PinNumber = GPIO_PIN_NO_13;
GPIO_Init(&GPIOLed);

GPIOLed.GPIO_PinConfig.GPIO_PinNumber = GPIO_PIN_NO_14;
GPIO_Init(&GPIOLed);

GPIOLed.GPIO_PinConfig.GPIO_PinNumber = GPIO_PIN_NO_15;
GPIO_Init(&GPIOLed);


//Enable the clock for GPIOD Peripheral

GPIO_PeriClockControl(GPIOD,ENABLE);

controlLED(GPIO_PIN_NO_12, true);
controlLED(GPIO_PIN_NO_13, false);
controlLED(GPIO_PIN_NO_14, true);
controlLED(GPIO_PIN_NO_15, false);
delay();
controlLED(GPIO_PIN_NO_12, false);
controlLED(GPIO_PIN_NO_13, true);
controlLED(GPIO_PIN_NO_14, false);
controlLED(GPIO_PIN_NO_15, true);
delay();
}

void controlLED(uint8_t LedNumber, bool state){
if(state){
GPIO WriteToOutputPin(GPIOD, LedNumber, GPIO_PIN_SET);
}else{
GPIO_WriteToOutputPin(GPIOD, LedNumber, GPIO_PIN_RESET);
}
}
void delay(){
for (uint32_t i=0;i<500000;i++);

}
```

5. Traffic Light Simulation: Simulate a traffic light system using the 4 LEDs. Assign them as Red, Yellow, Green, and a Pedestrian light. Use appropriate timing sequences to mimic real-world behavior.

```
void delay(uint32 t time);
void controlTrafficLight(uint8_t red, uint8_t yellow, uint8_t green,
uint8 t pedestrian);
int main(void)
{
GPIO_Handle_t GPIOLed;
```

```
GPIOLed.pGPIOx = GPIOD;

//Toggle Green LED connected to PD12


GPIOLed.GPIO_PinConfig.GPIO_PinMode = GPIO_MODE_OUT;

GPIOLed.GPIO_PinConfig.GPIO_PinSpeed = GPIO_SPEED_FAST;

GPIOLed.GPIO_PinConfig.GPIO_PinOPType = GPIO_OP_TYPE_PP;

GPIOLed.GPIO_PinConfig.GPIO_PinPuPdCOntrol = GPIO_NO_PUPD;

GPIOLed.GPIO_PinConfig.GPIO_PinNumber = GPIO_PIN_NO_12;//Green
GPIO_Init(&GPIOLed);

GPIOLed.GPIO_PinConfig.GPIO_PinNumber = GPIO_PIN_NO_13;//Orange
GPIO_Init(&GPIOLed);

GPIOLed.GPIO_PinConfig.GPIO_PinNumber = GPIO_PIN_NO_14;//Red
GPIO_Init(&GPIOLed);

GPIOLed.GPIO_PinConfig.GPIO_PinNumber = GPIO_PIN_NO_15;//Blue
GPIO_Init(&GPIOLed);


//Enable the clock for GPIOD Peripheral

GPIO PeriClockControl(GPIOD,ENABLE);
while(1){
// Red light ON (Pedestrian light ON)
controlTrafficLight(GPIO_PIN_SET, GPIO_PIN_RESET, GPIO_PIN_RESET,
GPIO_PIN_SET);
delay(5000); // 5 seconds

// Yellow light ON (All others OFF)
controlTrafficLight(GPIO_PIN_RESET, GPIO_PIN_SET, GPIO_PIN_RESET,
GPIO_PIN_RESET);
delay(2000); // 2 seconds

// Green light ON (Pedestrian light OFF)
controlTrafficLight(GPIO_PIN_RESET, GPIO_PIN_RESET, GPIO_PIN_SET,
GPIO_PIN_RESET);
delay(5000); // 5 seconds
}
}


void controlTrafficLight(uint8_t red, uint8_t yellow, uint8_t green,
uint8_t pedestrian)
{
GPIO_WriteToOutputPin(GPIOD, GPIO_PIN_NO_14, red); // Red LED
GPIO_WriteToOutputPin(GPIOD, GPIO_PIN_NO_13, yellow); // Yellow LED
GPIO_WriteToOutputPin(GPIOD, GPIO_PIN_NO_12, green); // Green LED
GPIO_WriteToOutputPin(GPIOD, GPIO_PIN_NO_15, pedestrian); //
Pedestrian LED
```

```
}

// Simple delay function
void delay(uint32_t time)
{
for (volatile uint32_t i = 0; i < (time * 1000); i++)
; // Delay loop (time in milliseconds)
}
```

6. LED Pattern Generator: Allow the user to define custom ON/OFF patterns for the 4 LEDs via an array. For example, the input [1, 0, 1, 0] should turn LED1 and LED3 ON, and LED2 and LED4 OFF.

```
void setPattern(uint8_t pattern[]);
int main(void)
{
GPIO_Handle_t GPIOLed;

GPIOLed.pGPIOx = GPIOD;

//Toggle Green LED connected to PD12


GPIOLed.GPIO_PinConfig.GPIO_PinMode = GPIO_MODE_OUT;

GPIOLed.GPIO_PinConfig.GPIO_PinSpeed = GPIO_SPEED_FAST;

GPIOLed.GPIO_PinConfig.GPIO_PinOPType = GPIO_OP_TYPE_PP;

GPIOLed.GPIO_PinConfig.GPIO_PinPuPdCOntrol = GPIO_NO_PUPD;

GPIOLed.GPIO_PinConfig.GPIO_PinNumber = GPIO_PIN_NO_12;
GPIO_Init(&GPIOLed);

GPIOLed.GPIO_PinConfig.GPIO_PinNumber = GPIO_PIN_NO_13;
GPIO_Init(&GPIOLed);

GPIOLed.GPIO_PinConfig.GPIO_PinNumber = GPIO_PIN_NO_14;
GPIO_Init(&GPIOLed);

GPIOLed.GPIO_PinConfig.GPIO_PinNumber = GPIO_PIN_NO_15;
GPIO_Init(&GPIOLed);


//Enable the clock for GPIOD Peripheral

GPIO_PeriClockControl(GPIOD,ENABLE);
uint8_t pattern1[4] = {1, 0, 1, 0};
uint8_t pattern2[4] = {0, 1, 0, 1};
uint8_t pattern3[4] = {1, 1, 1, 1};
uint8 t pattern4[4] = {0, 0, 0, 0};
while(1){
setPattern(pattern1);
delay();
```

```
setPattern(pattern2);
delay();

setPattern(pattern3);
delay();

setPattern(pattern4);
delay();
}
}


void setPattern(uint8_t pattern[])
{
GPIO_WriteToOutputPin(GPIOD, GPIO_PIN_NO_12, pattern[0]);
GPIO_WriteToOutputPin(GPIOD, GPIO_PIN_NO_13, pattern[1]);
GPIO_WriteToOutputPin(GPIOD, GPIO_PIN_NO_14, pattern[2]);
GPIO_WriteToOutputPin(GPIOD, GPIO_PIN_NO_15, pattern[3]);
}

void delay(){
for (uint32_t i=0;i<500000;i++);

}
```