LED Blinking on FPGA – Report

1. Objective

The aim of this project is to implement an RGB LED blinking system on an FPGA using Verilog. The Red, Green, and Blue (RGB) LEDs are controlled individually to create different color patterns. The FPGA will sequentially toggle the three colors at a fixed interval.

---

2. FPGA Board and Connections

This project is implemented on the VSDSquadron Mini FPGA, which is based on the Lattice iCE40 FPGA. The RGB LED consists of three separate LEDs (Red, Green, and Blue) embedded in a single package, with each color controlled independently.

Pin Configuration

Signal	FPGA Pin	Description
clk	35	System clock input (12 MHz)
led_r	21	Red LED output
led_g	22	Green LED output
led_b	23	Blue LED output

- The RGB LED has three control signals for Red, Green, and Blue.
- Each LED is active-low, meaning 0 turns it ON and 1 turns it OFF.

---

3. Verilog Code (rgb_blink_led.v)

The RGB LED is controlled using a counter-based clock divider. Each LED toggles sequentially to create a blinking effect.

```
module rgb_blink_led (
    input clk,        // 12 MHz system clock
    output reg led_r,  // Red LED
    output reg led_g,  // Green LED
    output reg led_b   // Blue LED
);
```

```
reg [23:0] counter;  // 24-bit counter for timing
reg [1:0] state;     // State variable to switch between colors

always @(posedge clk) begin
    counter <= counter + 1;

    if (counter == 24'd6000000) begin  // ~0.5s delay
        state <= state + 1;
        counter <= 0;
    end
end

always @(*) begin
    case (state)
        2'b00: begin
            led_r = 0; led_g = 1; led_b = 1;  // Red ON
        end
        2'b01: begin
            led_r = 1; led_g = 0; led_b = 1;  // Green ON
        end
        2'b10: begin
            led_r = 1; led_g = 1; led_b = 0;  // Blue ON
        end
        default: begin
            led_r = 1; led_g = 1; led_b = 1;  // All OFF
        end
    endcase
end
```

endmodule

Code Explanation:

1. Clock (clk) – Controls the timing of color changes.
2. Counter (counter[23:0]) – Divides the 12 MHz clock to create a 0.5s delay before switching colors.
3. State Variable (state[1:0]) – Determines which LED should be ON.
4. LED Control Logic:
    - State 00 → Red ON, Green OFF, Blue OFF
    - State 01 → Red OFF, Green ON, Blue OFF
    - State 10 → Red OFF, Green OFF, Blue ON
    - Default → All OFF

---

4. FPGA Implementation Steps

1. Synthesis (Yosys)

yosys -p "read_verilog rgb_blink_led.v; synth_ice40 -json rgb_blink_led.json"

- Converts Verilog to FPGA logic gates.

2. Place & Route (NextPNR)

nextpnr-ice40 --json rgb_blink_led.json --pcf pin_constraints.pcf --asc rgb_blink_led.asc --package hx8k

- Maps logic to FPGA resources and assigns pins.

3. Generate Bitstream (IcePack)

icepack rgb_blink_led.asc rgb_blink_led.bin

- Creates the binary file for FPGA programming.

4. Flashing FPGA (IceProg)

iceprog rgb_blink_led.bin

- Uploads the program to the FPGA.
- The RGB LED should start changing colors every 0.5s.

---

5. Simulation & Debugging

Run Simulation

make test

- Simulates the Verilog design using Icarus Verilog.

View Waveform

gtkwave waveform.fst

- Ensures proper color transitions.

Debugging Tips:

- If LED does not change color, check pin assignments (pin_constraints.pcf).
- If LED flickers too fast, increase the counter delay (24'd6000000).

---

6. Conclusion

- Successfully implemented an RGB LED blinking sequence on FPGA.
- Used open-source tools (Yosys, NextPNR, IceProg).
- Verified design through simulation & waveform analysis.

This project demonstrates basic FPGA control of multi-color LEDs, useful for embedded system applications.

1.2 Block Diagram

The block diagram shown in Figure 1 shows the key components of the VSDSquadron FPGA Mini (FM) board.

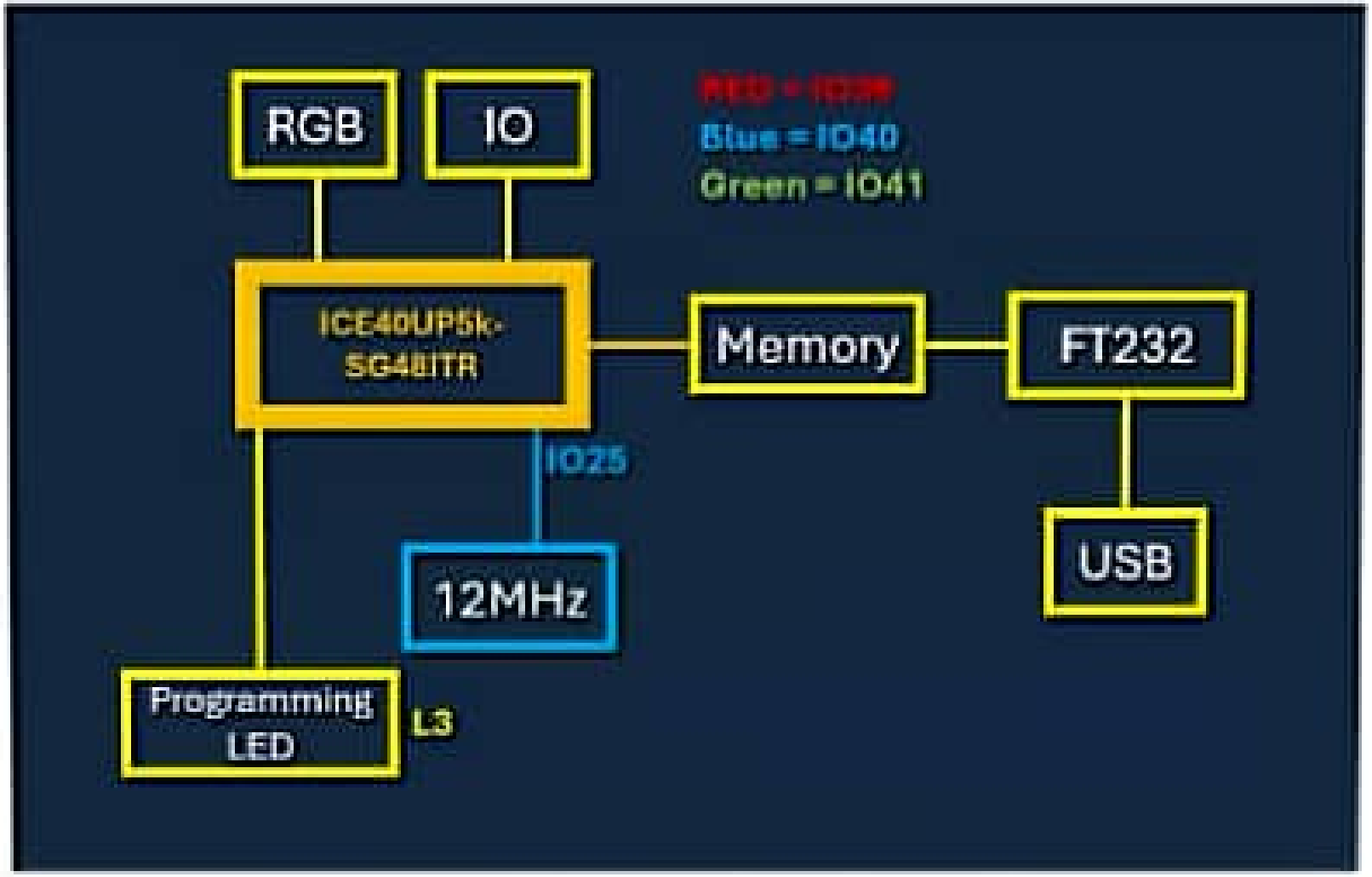

Figure 1: VSDSquadron FPGA Mini (FM) board Block Diagram

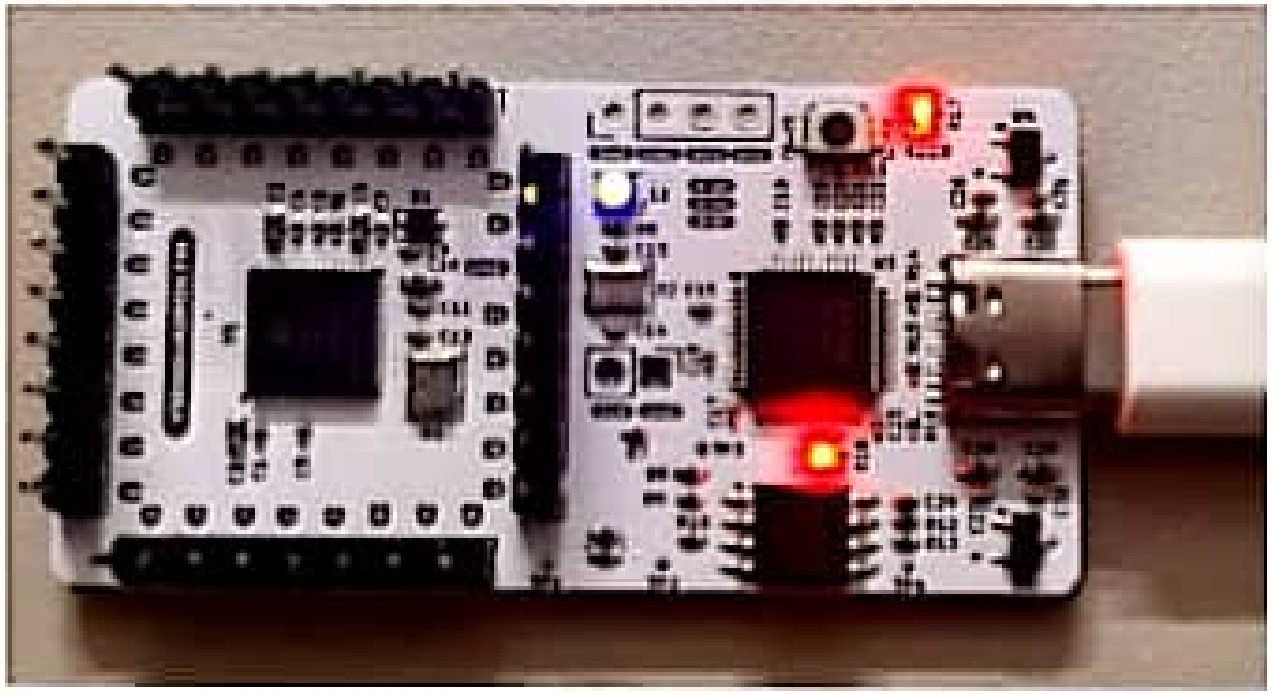

Figure 3: Micro-C end of USB cable connected to board

FNC	Pin Type	BANK	Differential Pair	Pin number
IOB_0a	PIO	2	-	46
IOB_2a	DPIO	2	TRUE_of_IOB_3b	47
IOB_3b_G0	DPIO/GBIN	2	COMP_of_IOB_2a	44
IOB_1a	DPIO	2	TRUE_of_IOB_5b	48
IOB_5b	DPIO	2	COMP_of_IOB_1a	45
IOB_6a	PIO	2	-	2
IOB_8a	DPIO	2	TRUE_of_IOB_9b	4
IOB_9b	DPIO	2	COMP_of_IOB_8a	3
IOB_10a	DPIO	1	TRUE_of_IOB_11b	-
IOB_11b_G5	DPIO/GBIN	1	COMP_of_IOB_10a	-
creset_b	CONFIG	1	-	6
IOB_12a_CI_CDONE	CONFIG/DPIO/GBIN	1	TRUE_of_IOB_13b	-
CDONE	CONFIG	1	-	7
IOB_13b	DPIO	1	COMP_of_IOB_12a	9
IOB_16a	PIO	1	-	5
IOB_18a	PIO	1	-	10
IOB_20a	PIO	1	-	11
IOB_22a	DPIO	1	TRUE_of_IOB_23b	12
IOB_23b	DPIO	1	COMP_of_IOB_22a	21
IOB_24a	DPIO	1	TRUE_of_IOB_25b	13
IOB_25b_G1	DPIO/GBIN	1	COMP_of_IOB_24a	20
IOB_29b	PIO	1	-	19
IOB_31b	PIO	1	-	18
IOB_32a_SPI_SO	DPIO/CONFIG_SPI	1	-	14
IOB_33b_SPI_SI	DPIO/CONFIG_SPI	1	-	17
IOB_34a_SPI_SCK	DPIO/CONFIG_SPI	1	-	15
IOB_35b_SPI_SS	DPIO/CONFIG_SPI	1	-	16
VCCPLL	VCCPLL	-	-	29
IOT_36b	DPIO/TC	0	COMP_of_IOT_37a	25
IOT_37a	DPIO/TC	0	TRUE_of_IOT_36b	23
IOT_38b	DPIO	0	COMP_of_IOT_39a	27
IOT_39a	DPIO	0	TRUE_of_IOT_38b	26
IOT_41a	PIO	0	-	28
IOT_42b	DPIO	0	COMP_of_IOT_43a	31
IOT_43a	DPIO	0	TRUE_of_IOT_42b	32
IOT_44b	DPIO	0	COMP_of_IOT_45a	34
IOT_45a_G1	DPIO/GBIN	0	TRUE_of_IOT_44b	37
IOT_46b_G0	DPIO/GBIN	0	-	35
IOT_47a	PIO	0	-	-
IOT_48b	DPIO	0	COMP_of_IOT_49a	36
IOT_49a	DPIO	0	TRUE_of_IOT_48b	43
IOT_50b	DPIO	0	COMP_of_IOT_51a	38
IOT_51a	DPIO	0	TRUE_of_IOT_50b	42
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
GND	GND	GND	-	Paddle
GND	GND	GND	-	Paddle
GND	GND	GND	-	Paddle
VCC	VCC	VCC	-	8
VCC	VCC	VCC	-	30
VCCIO_0	VCCIO	0	-	33
SPI_Vccio1	VCCIO	1	-	22
VCCIO_2	VCCIO	2	-	1
VPP_2V5	VPP	VPP	-	24

Table 2: ICE40UP5K-SG48ITR FPGA device IO Bank Assignment