# COMPUTER AIDED DIGITAL DESIGN

# REPORT

## PROJECT TITLE :- CELSIUS TO FAHRENHEIT CONVERTER

Name : Santosh Hanamappa Mokashi
SRN: PES2UG22EC118
Section: “B”

# ABSTRACT

The  SystemVerilog code consists of a Celsius to Fahrenheit conversion module and an associated testbench . The module takes input signals for clock (`clk`), reset (`rst`), and a 4-bit Celsius value (`celsius`), producing a 7-bit Fahrenheit output (`fahrenheit`). The module employs a synchronous always block triggered on the positive edge of the clock. If the reset is active, it initializes the Fahrenheit output to zero; otherwise, it performs the Celsius to Fahrenheit conversion using a simple formula. The testbench initializes clock and reset signals, provides different Celsius input values at specified time intervals, and monitors the corresponding Fahrenheit output. The simulation concludes after a defined sequence of time steps. This setup facilitates the verification and testing of the Celsius to Fahrenheit conversion module.

# ***Design Implementation:***

The project consists of a SystemVerilog module named `ctf` responsible for converting temperatures from Celsius to Fahrenheit. It takes input signals for the clock (`clk`), reset (`rst`), and a 4-bit Celsius temperature (`celsius`). The output is a 7-bit Fahrenheit temperature (`fahrenheit`). The conversion formula is implemented in a synchronous always block that triggers on the rising edge of the clock. When the reset signal is asserted, the Fahrenheit output is set to 0. Otherwise,the Fahrenheit temperature is calculated using the formula

$F=59\times C+32$The testbench, named `ctf_tb`, initializes the necessary signals, including clock, reset, Celsius input, and Fahrenheit output. It simulates the module by toggling the clock, releasing the reset after a delay, and providing different Celsius input values. The testbench monitors and displays the Celsius and Fahrenheit values during simulation and stops after a specified time.

**DESIGN CODE :**

```
module ctf(
  input logic clk,
  input logic rst,
  input logic [3:0] celsius,
  output logic [6:0] fahrenheit
);
always_ff @(posedge clk) begin
    if (rst) begin
      fahrenheit <= 0;
    end else begin
      fahrenheit <= (celsius * 9/5) + 32;
    end
  end
endmodule
```

**TESTBENCH :**

```
module ctf_tb;
  logic clk;
  logic rst;
  logic [3:0] celsius;
  logic [6:0] fahrenheit;
 initial begin
    clk = 0;
    rst = 1;
    #5 rst = 0;
    celsius = 9;
    #10 celsius =10  ;
    #10 celsius = 0;
    #10 $stop;
  End

 always begin
    #1 clk = ~clk;
  end
ctf dut (.clk(clk), .rst(rst),   .celsius(celsius), .fahrenheit(fahrenheit) );
 initial begin
    $display("Celsius\tFahrenheit");
    $display("-------\t----------");
    $monitor("%d\t%d", celsius, fahrenheit);
  end
endmodule
```

Test Bench;
The testbench (**`ctf_tb`**) for the Celsius to Fahrenheit conversion module (**`ctf`**) initiates with the declaration and initialization of signals, including the clock (**`clk`**), reset (**`rst`**), Celsius input (**`celsius`**), and Fahrenheit output (**`fahrenheit`**). It sets the clock to 0 and the reset to 1, waits for a brief period, and then deasserts the reset to initiate the simulation. Subsequently, it assigns different Celsius values at specific time intervals. The clock is generated by toggling its value in an always block every time unit. The testbench monitors and displays the values of both Celsius and Fahrenheit in a tabular format throughout the simulation. After the specified time, the simulation concludes, providing a comprehensive verification environment for the Celsius to Fahrenheit conversion module. The testbench aims to validate the correct behavior of the conversion under varying input conditions and ensures that the Fahrenheit output corresponds accurately to the Celsius input based on the specified conversion formula.

## TRUTH TABLE :

| F7 | F6 | F5 | F4 | F3 | F1 | F0 |
|---|---|---|---|---|---|---|
| 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 1 |
| 0 | 1 | 0 | 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 0 | 1 | 0 | 1 |
| 0 | 1 | 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 | 1 | 0 | 0 |
| 0 | 1 | 0 | 1 | 1 | 1 | 0 |
| 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 0 | 1 | 1 |
| 0 | 1 | 1 | 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 | 1 | 1 | 1 |
| 0 | 1 | 1 | 1 | 0 | 0 | 1 |
| 0 | 1 | 1 | 1 | 0 | 1 | 1 |

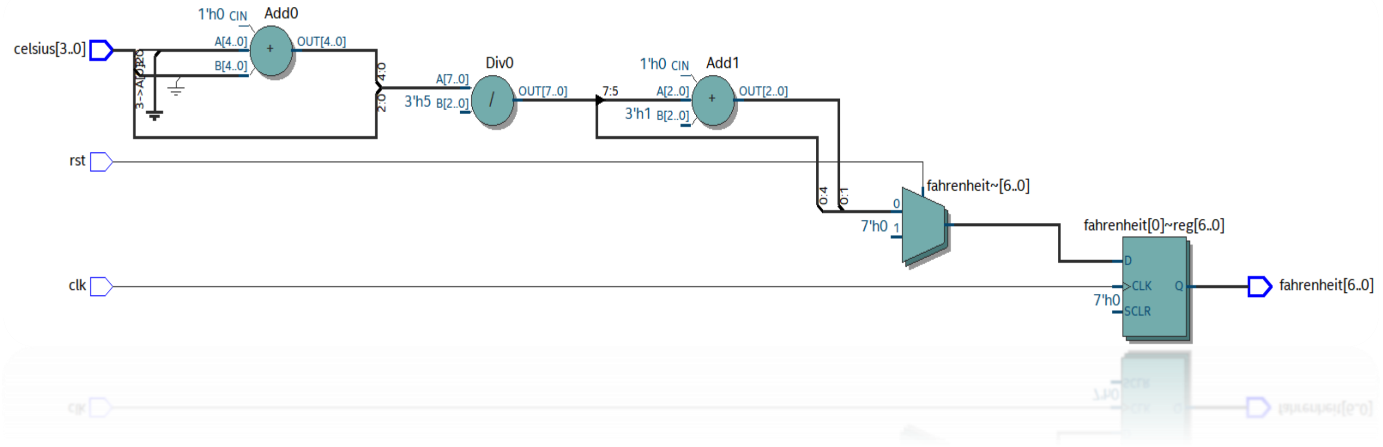
| C3 | C2 | C1 | C0 |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 0 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 0 |
| 1 | 1 | 1 | 1 |

| CLK | RST |
|---|---|
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |

## CONVERSION TABLE:

| CELSIUS | FAHRENHEIT |
|---|---|
| 0 | 32 |
| 1 | 33 |
| 2 | 35 |
| 3 | 37 |
| 4 | 39 |
| 5 | 41 |
| 6 | 42 |
| 7 | 44 |
| 8 | 46 |
| 9 | 48 |
| 10 | 50 |
| 11 | 51 |
| 12 | 53 |
| 13 | 55 |
| 14 | 57 |
| 15 | 59 |

# SIMULATION RESULT: **RTL VIEWER :**



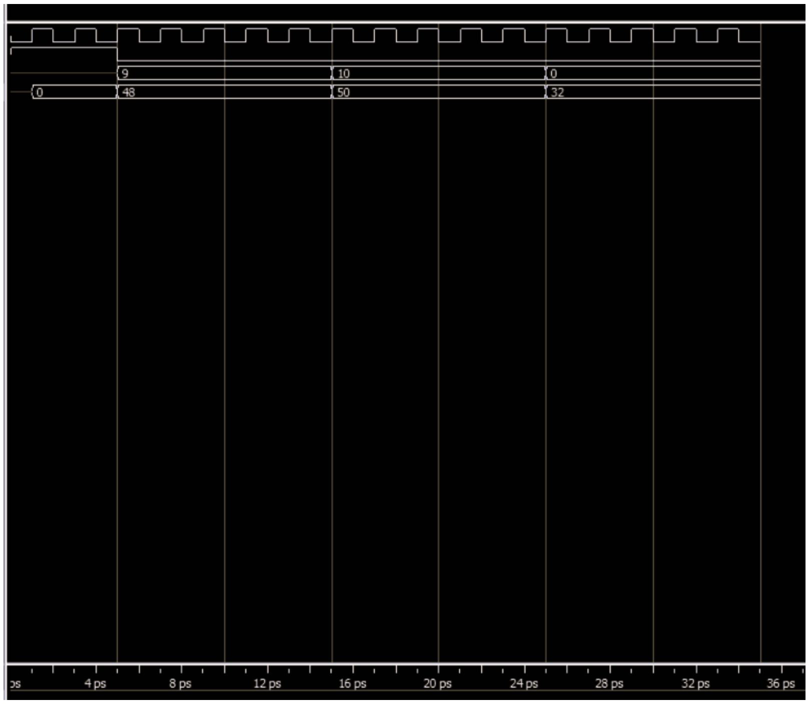
- Celsius is given as input and is carried to adder block adds the input number 9 times. it's
- Then moves to the divider block divides by 5 further proceeds to adder block that adds 32

  to the resulting.
- The resulting is forced to the mux . reset is select line & when it is 0 the data is carried

  futher or else if reset is 1 then it restarts the procedure .
- The result of mux enters input 'd' to register and it has clk as input. the register on

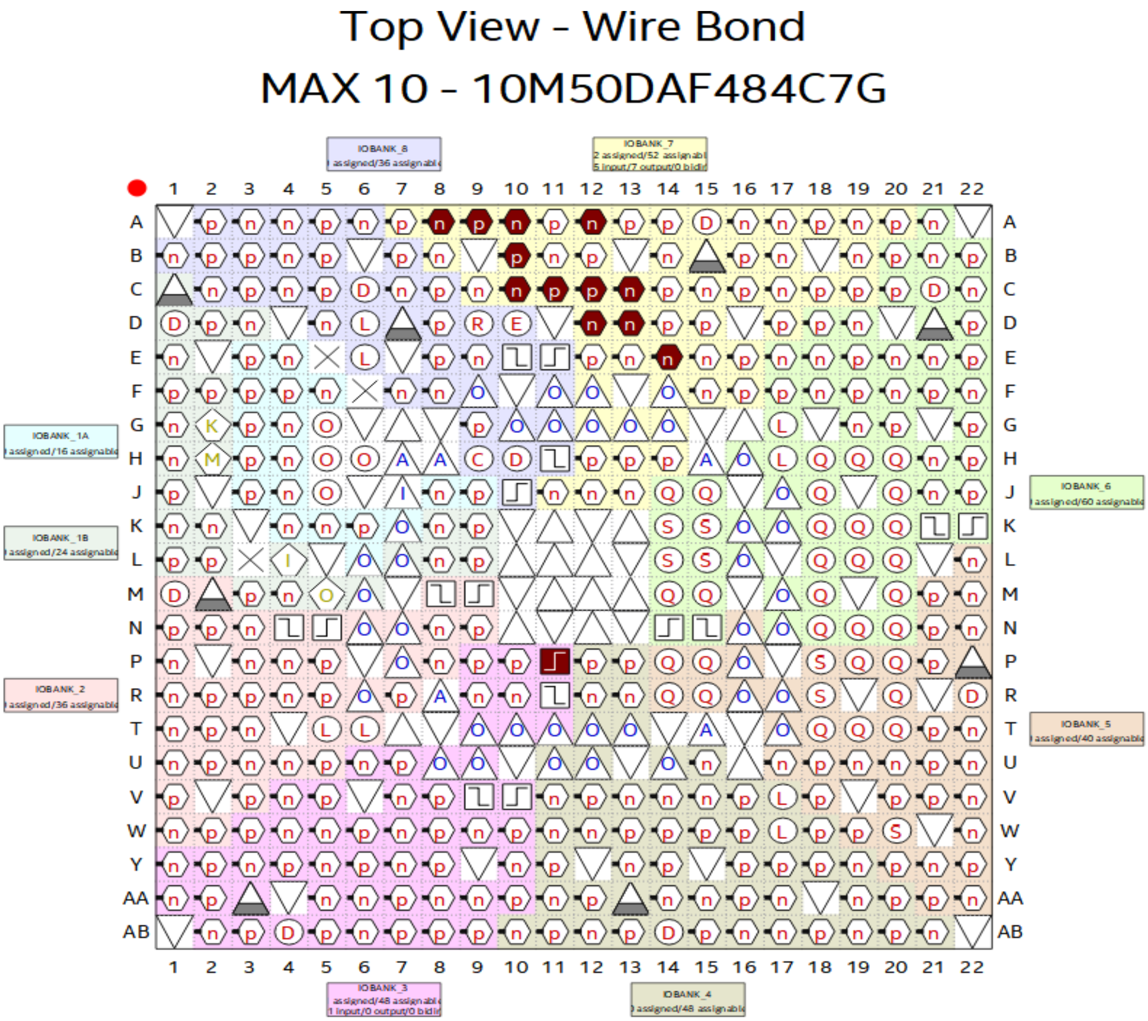
  matching clk and input of 'd' finally gives the output as degree fahrenheit.

## **Output wave form**:

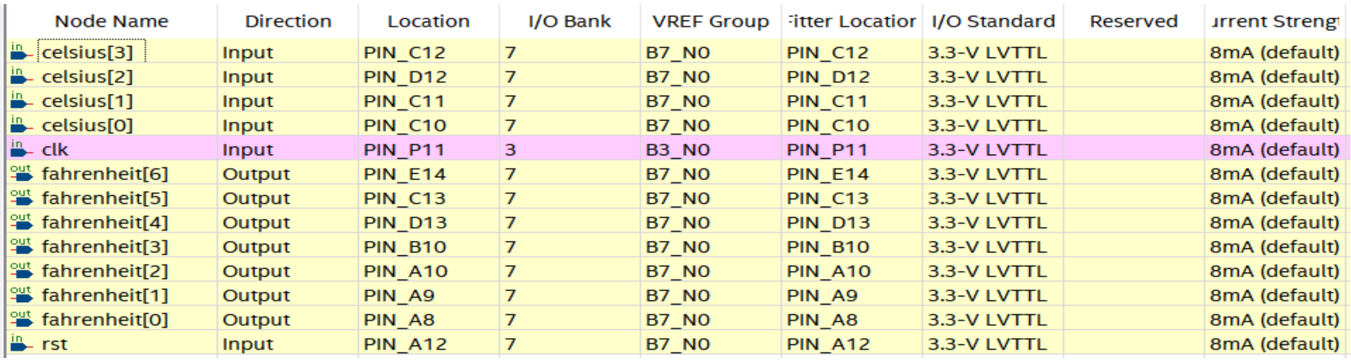


- The graph indicates waveform of clk, reset, input celsius,output fahrenheit.
- As per the testbench for every 10 psec delay the input changes so the output also changes.
- For input 9 degree celsius the reset becomes 0 and corresponding output is 48 fahrenheit
- For input 10 degree celsius the reset remains 0 and corresponding output is 50 fahrenheit
- For input 0 degree celsius the reset remains 0 and corresponding output is 32 fahrenheit

## Pin Planner :

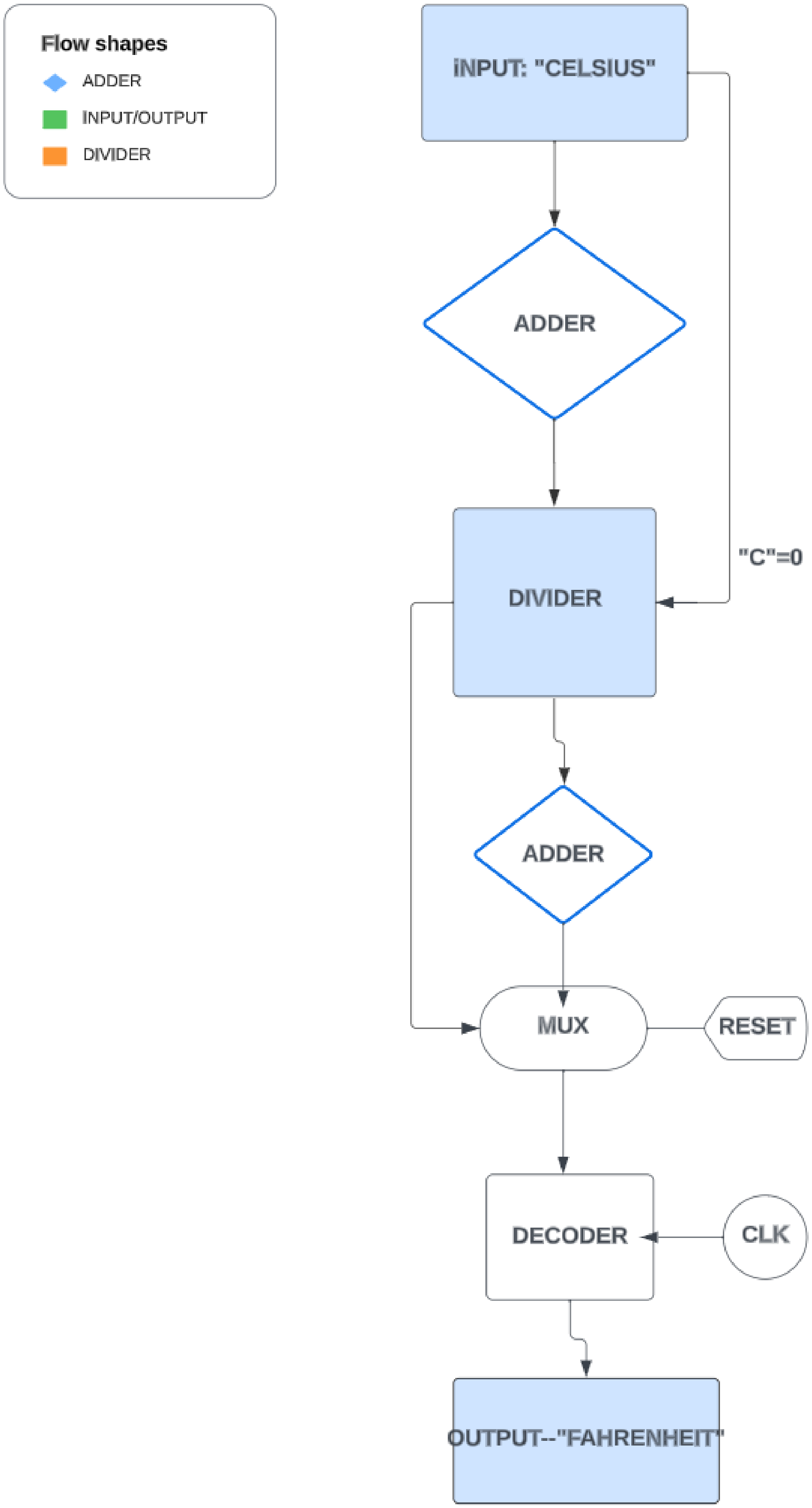


- The pin planner shows the assigned ports to inputs and outputs including clk and reset.
- Inputs are denoted by switches on the board and the output are denoted by led's.



| Node Name | Direction | Location | I/O Bank | VREF Group | itter Locatior | I/O Standard | Reserved | urrent Streng |
|---|---|---|---|---|---|---|---|---|
| celsius[3] | Input | PIN_C12 | 7 | B7_N0 | PIN_C12 | 3.3-V LVTTL | | 8mA (default) |
| celsius[2] | Input | PIN_D12 | 7 | B7_N0 | PIN_D12 | 3.3-V LVTTL | | 8mA (default) |
| celsius[1] | Input | PIN_C11 | 7 | B7_N0 | PIN_C11 | 3.3-V LVTTL | | 8mA (default) |
| celsius[0] | Input | PIN_C10 | 7 | B7_N0 | PIN_C10 | 3.3-V LVTTL | | 8mA (default) |
| clk | Input | PIN_P11 | 3 | B3_N0 | PIN_P11 | 3.3-V LVTTL | | 8mA (default) |
| fahrenheit[6] | Output | PIN_E14 | 7 | B7_N0 | PIN_E14 | 3.3-V LVTTL | | 8mA (default) |
| fahrenheit[5] | Output | PIN_C13 | 7 | B7_N0 | PIN_C13 | 3.3-V LVTTL | | 8mA (default) |
| fahrenheit[4] | Output | PIN_D13 | 7 | B7_N0 | PIN_D13 | 3.3-V LVTTL | | 8mA (default) |
| fahrenheit[3] | Output | PIN_B10 | 7 | B7_N0 | PIN_B10 | 3.3-V LVTTL | | 8mA (default) |
| fahrenheit[2] | Output | PIN_A10 | 7 | B7_N0 | PIN_A10 | 3.3-V LVTTL | | 8mA (default) |
| fahrenheit[1] | Output | PIN_A9 | 7 | B7_N0 | PIN_A9 | 3.3-V LVTTL | | 8mA (default) |
| fahrenheit[0] | Output | PIN_A8 | 7 | B7_N0 | PIN_A8 | 3.3-V LVTTL | | 8mA (default) |
| rst | Input | PIN_A12 | 7 | B7_N0 | PIN_A12 | 3.3-V LVTTL | | 8mA (default) |

## FLOWCHART:


Flow shapes
ADDER
INPUT/OUTPUT
DIVIDER
INPUT: "CELSIUS"
ADDER
DIVIDER
"C"=0
ADDER
MUX
RESET
DECODER
CLK
OUTPUT--"FAHRENHEIT"

## *Observations on Results:*

The simulation results align with the expected values, indicating the successful implementation of the Celsius to Fahrenheit conversion. The module responds correctly to changes in the input Celsius temperature and produces accurate Fahrenheit output.

## *Conclusions:*

The Celsius to Fahrenheit conversion module has been successfully designed and tested. The project demonstrates the correct operation of the conversion formula within the specified SystemVerilog framework. The testbench effectively validates the module's functionality across different input scenarios. Future improvements could include additional test cases and more comprehensive verification. The project serves as a robust foundation for temperature conversion applications.