



שפת תכנון חומרה –

Verilog

מטלה מספר 3

מגיש :

חיים עוזר - 316063569

LVL TOP ויזאול'י תוסף-Ex3

❖ Code:

```
module ex3 (
    input wire clk,reset,
    input wire [1:0] div,           // input
    output wire div_clk);          // Output divided clock

    ClockDivider ClkDiv
    (.clk(clk),.reset(reset),.in(div[1:0]),.out(div_clk));

endmodule
```

Clock Divider

❖ Code:

```
module ClockDivider (
    input wire clk, reset,
    input wire [1:0] in,
    output wire out
);

    reg [23:0] counter;
    reg div_clk_reg;

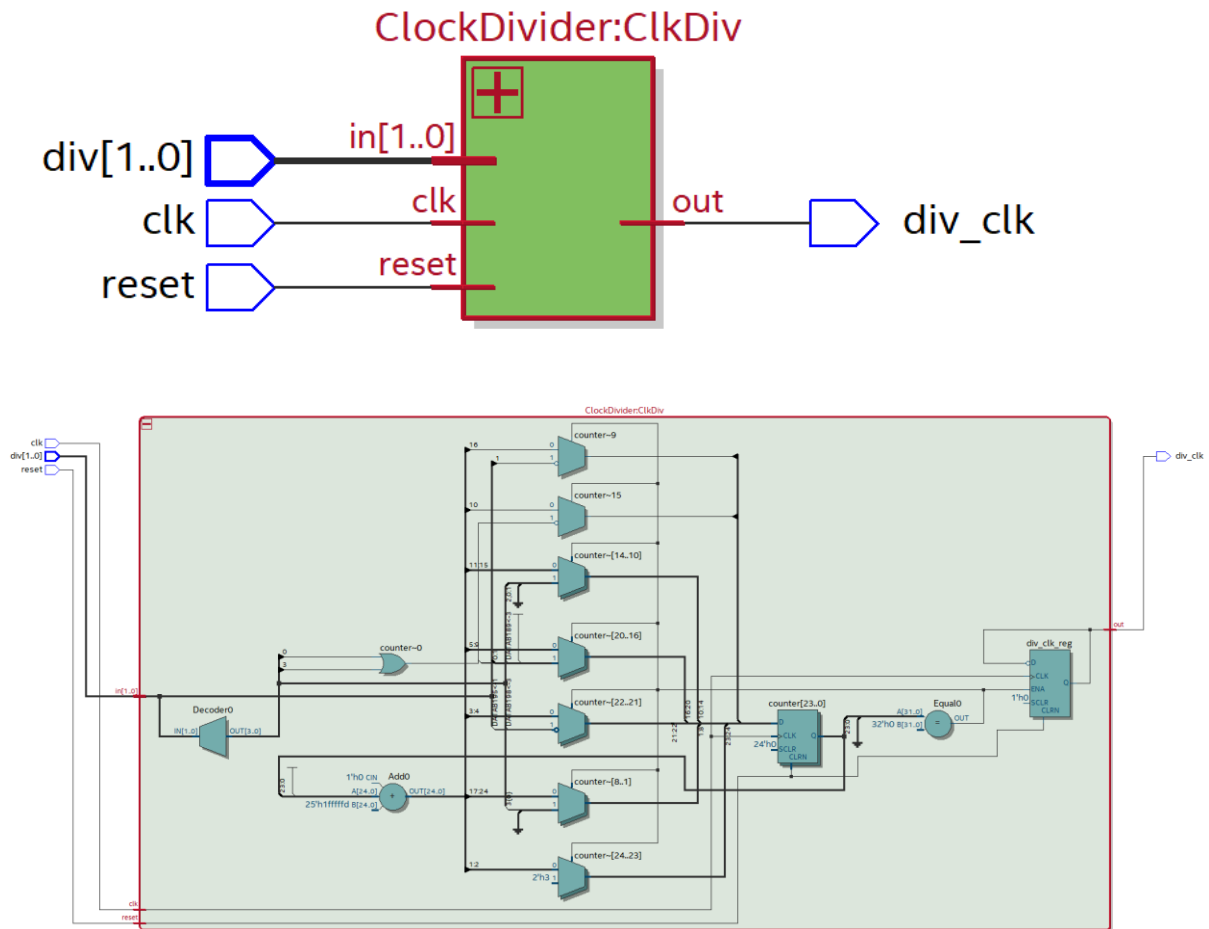
    initial begin
        counter <= 0; // Use non-blocking assignment
        div_clk_reg <= 1'b1; // Use non-blocking assignment
    end

    always @(posedge clk or posedge reset) begin
        if (reset) begin // asynchronous
            counter <= 0;
        end
        else begin
            if (counter == 0) begin
                case (in)
                    2'b00: counter <= 249_999; // 100 Hz
                    2'b01: counter <= 24_999; // 1 kHz
                    2'b10: counter <= 2_499; // 10 kHz
                    2'b11: counter <= 249; // 100 kHz
                    default: counter <= 0;
                endcase
                div_clk_reg <= ~div_clk_reg;
            end else if (counter > 0) begin // synchronous and counter > 0
                counter <= counter - 1'b1; // Decrement counter
            end
        end
    end

    assign out = div_clk_reg;

endmodule
```

## ❖ RTL:



## ❖ Compilation:

Quartus Prime Lite Edition - C:/intelFPGA\_lite/18.0/EX3/ex3 - ex3

File Edit View Project Assignments Processing Tools Window Help

Search altera.com

Project Navigator Files

Files

- ClockDivider.v
- ex3.v

Tasks

Compilation

Task

- Compile Design
- Analysis & Synthesis
- Fitter (Place & Route)
- Assembler (Generate program)
- Timing Analysis
- EDA Netlist Writer

Flow Summary

Table of Contents

Flow Status: Successful - Wed Feb 21 19:42:22 2024

Quartus Prime Version: 18.0.0 Build 614 04/24/2018 SJ Lite Edition

Revision Name: ex3

Top-level Entity Name: ex3

Family: Cyclone V

Device: 5CGXFC7C7F23C8

Timing Models: Final

Logic utilization (in ALMs): N/A

Total registers: 25

Total pins: 5

Total virtual pins: 0

Total block memory bits: 0

Total DSP Blocks: 0

Total HSSI RX PCSs: 0

Total HSSI PMA RX Deserializers: 0

Total HSSI TX PCSs: 0

Total HSSI PMA TX Serializers: 0

Total PLLs: 0

Total DLLs: 0

IP Catalog

Installed IP

- Project Directory
  - No Selection Available
- Library
  - Basic Functions
  - DSP
  - Interface Protocols
  - Memory Interfaces and Controllers
  - Processors and Peripherals
  - University Program
- Search for Partner IP

Messages

System Processing (14)

21057 Implemented 56 device resources after synthesis - the final resource count might be different

Quartus Prime Analysis & Synthesis was successful. 0 errors, 2 warnings

**הסבר הקוד במילים שלי:**

המונה הוא כמו טיימר שסופר לאחור מערך שנקבע על ידי האות div. כאשר הוא מגיע לאפס, הוא מפעיל את עדכון השעון המחולק. האיפוס האסינכרוני RESET מאפס באופן מיידי את המונה ואת השעון המחולק כאשר הוא נטען. חלוקת השעון הסינכרוני מבטיחה שתדר השעון המחולק נשלט על ידי אות ה-div.

**שיטת החלוקה להמרת התדר תתבצע בצורה הבא:**

לדוגמא עבור  $DIV = 00$  המונה מאותחל לחלוקה של 249,999 ערך זה נקבע על ידי התחשבות בתדר השעון של 50 מגה-הרץ (שעון חיצוני) ותדר המוצא הרצוי של 100 הרץ.

$$\text{Counter Value} = \frac{\text{Clock Frequency}}{\text{Desired Output Frequency}} - 1$$

**במקרה שלנו:**

$$\frac{50,000,000}{100} - 1 = 499,999$$

המונה מוגדר ל-499,999 אך בשביל לקבל עליה וירידה בזמן מחזור זה נחלק ב-2 ונקבל 249,999 כדי להשיג תדר מוצא של 100 הרץ כאשר DIV הוא "00".

המונה סופר לאחור מהערך הראשוני הזה, וכאשר הוא מגיע לאפס, הוא מפעיל את עדכון אות השעון המחולק, ויוצר פלט של 100 הרץ.

**Test Bench:**

## ❖ Code:

```

`timescale 1ns/1ns

module tb_ClockDivider;

    reg clk;
    reg reset;
    reg [1:0] div;
    wire div_clk;

    ClockDivider CD (.clk(clk), .reset(reset), .in(div), .out(div_clk));

    // Continuous clock generation
    always #10 clk = ~clk; // clock every 10 ns

    initial begin
        clk = 0;
        reset = 1; // Initial asynchronous reset
        #10_000 reset = 0; // Release reset after 10 ns
    end

    initial begin
        #20_000 // Wait for a while before changing div values

        div = 2'b11; #40_000 // Set DIV to 11 (100 kHz)
        div = 2'b10; #400_000 // Set DIV to 10 (10 kHz)
        div = 2'b01; #2_000_000 // Set DIV to 01 (1 kHz)
        div = 2'b00; #10_000_000; // Set DIV to 00 (100 Hz)

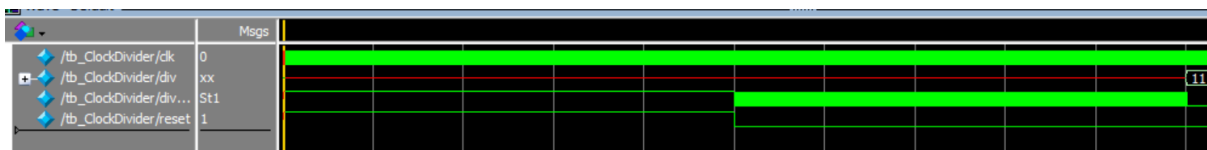
        //sum time to run in modlesim 12.47 mSEC
    end

end
endmodule

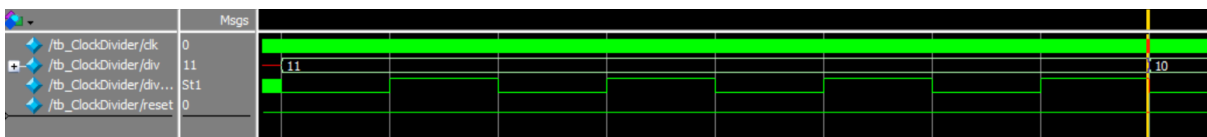
```

## ❖ Wave – Model sim:

Reset check:

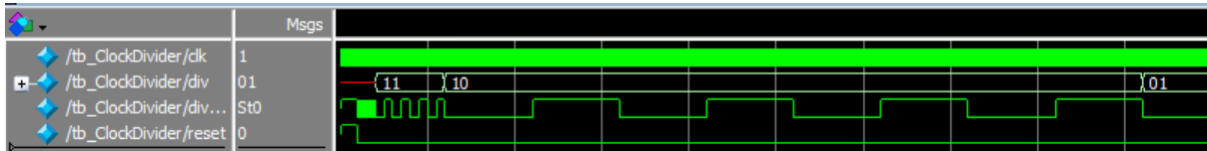


Div 11 check:



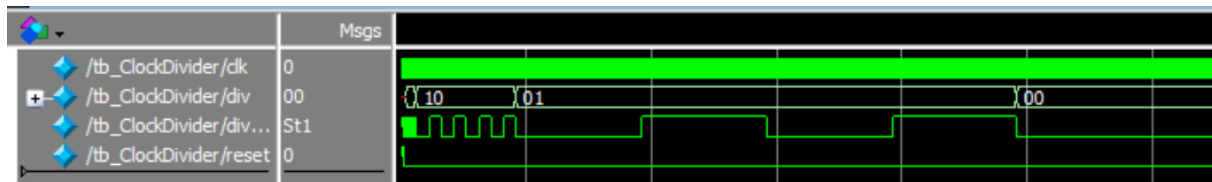
ניתן להבחין כי בזמן הבדיקה של 40KnsSec נקבל 4 עליות שעון ו4 ירידות שעון,  
כלומר זמן מחזור יהיה 10KnsSec ואז התדר שמתקבל יהיה 100Khz.

## Div 10 check:



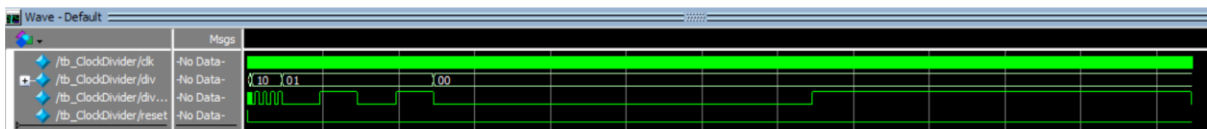
ניתן להבחין כי בזמן הבדיקה של 400KnSec נקבל 4 עליות שעון, כלומר זמן מחזור יהיה 100KnSec ואז התדר שמתקבל יהיה 10Khz.

## DIV 01 check:



ניתן להבחין כי בזמן הבדיקה של 2M nSec נקבל 2 עליות שעון, כלומר זמן מחזור יהיה 1MnSec ואז התדר שמתקבל יהיה 1Khz.

## DIV 00 check:



ניתן להבחין כי בזמן הבדיקה של 10M nSec נקבל עליית שעון בודדת וירידת שעון אחת, כלומר זמן מחזור יהיה גם 10MnSec ואז התדר שמתקבל יהיה 100 Hz.