PROYEK 1

PROGRAMMABLE CLOCK DIVIDER

PERANCANGAN KOMPONEN TERPROGRAM

Disusun oleh:

Nama	:	Yohanes Stefanus
NRP	:	5022211089
Github repository	:	https://github.com/yohanesstef/clock-divider.git

INSTITUT TEKNOLOGI SEPULUH NOPEMBER

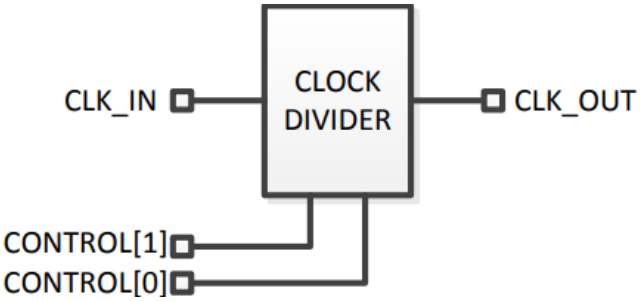
TEKNIK ELEKTRO

2023/2024

Spesifikasi:

Sebuah Programmable Clock Divider dengan frekuensi input 32MHz dan frekuensi output bervariasi sesuai dengan control input. Clock divider akan memiliki 3 input port, CLK_IN, CONTROL[0], dan CONTROL[1], dan 1 output port, CLK_OUT.

Diagram Blok:



Gambar 1. Diagram Block Clock Divider

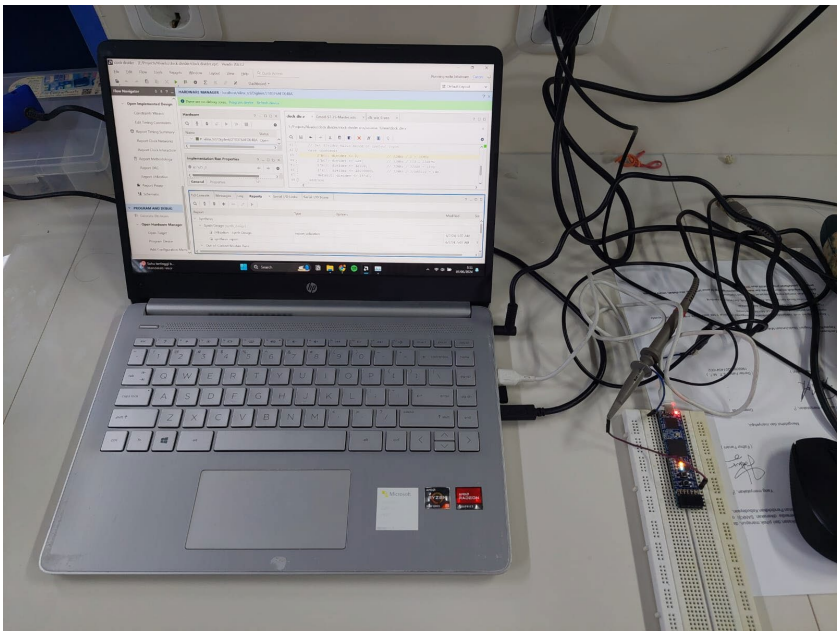
Tabel Input-Output:

CONTROL[1]	CONTROL[0]	Frekuensi CLK_OUT
0	0	16 MHz
0	1	20 kHz
1	0	1000 Hz
1	1	1 Hz

Tabel 1. Hubungan Input Ouput

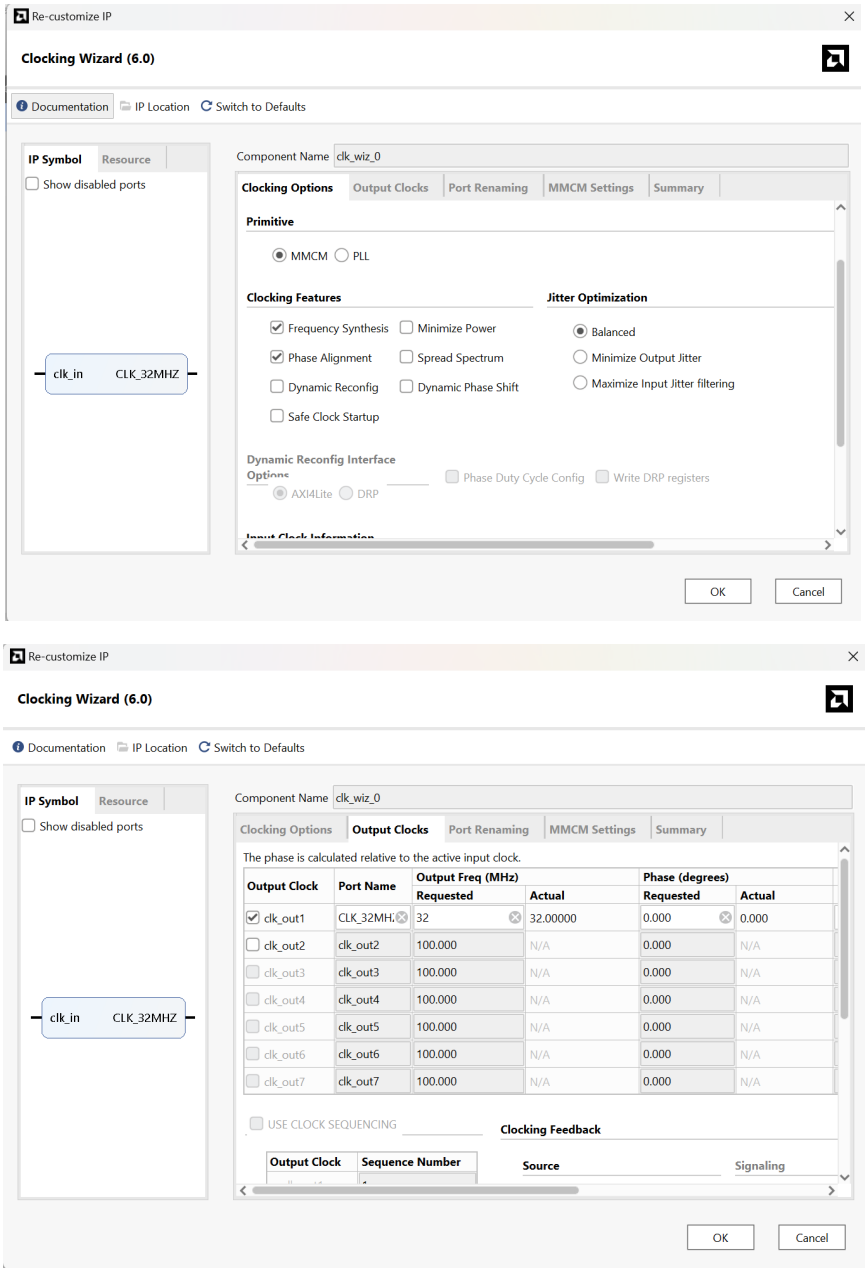
Pembahasan:

Pada proyek kali ini, saya menggunakan FPGA Xilinx Spartan-7 (XC7S25-1CSGA225C) dan software Vivado sebagai IDE-nya.



Gambar 2. Software Vivado dan FPGA Xilinx Spartan-7

Program verilog yang saya buat untuk clock divider kali ini menggunakan suatu variable pembagi untuk menghasilkan clock yang diinginkan. Board CMOD S7 yang saya gunakan memiliki oscilator external 12MHz. Untuk memenuhi spesifikasi clock 16 MHz, saya menggunakan IP (intellectual property) Clocking Wizard.



Gambar 3. Panel Clocking Wizard

Program:

Berikut adalah program verilognya:

```
`timescale 1ns / 1ps
```

```
//////////////////////////////////////////////////////////////////////////////////
////////
// Company: Institute Technology Sepuluh Nopember
// Engineer: Yohanes Stefanus
//
// Create Date: 07.06.2024 04:21:55
// Design Name:
// Module Name: clock_div
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////
////////


module clock_div(
    input clk_in,
    input [1:0] control,
    output reg clock_out
);

// Instantiate clock wizard
wire CLK_32MHZ;

clk_wiz_0 clk_wizard_inst (
    .CLK_32MHZ(CLK_32MHZ), // output clock 32MHz
    .clk_in(clk_in)        // input clock 12MHz
);

reg[24:0] counter = 25'd0;
reg[24:0] divider = 25'd1;

always @(posedge CLK_32MHZ) begin
    // Set divider value based on control input
    case (control)
        2'b00: divider <= 2;             // 32MHz / 2 = 16MHz
        2'b01: divider <= 128;           // 32MHz / 128 = 250kHz
        2'b10: divider <= 32000;         // 32MHz / 32000 = 1kHz
        2'b11: divider <= 32000000;      // 32MHz / 32000000 = 1Hz
```

```
        default: divider <= 25'd1;
    endcase

    counter <= counter + 25'd1;
    if(counter >= (divider - 1)) begin
        counter <= 25'd0;
    end

    clock_out <= (counter < (divider / 2)) ? 1'b1 : 1'b0;
end

endmodule
```

Pembahasan Program:

- Inisialisasi modul

```
module clock_div(
    input clk_in,
    input [1:0] control,
    output reg clock_out
);
```

Deklarasi modul dalam Verilog tersebut saya berinama clock_div. Pada modul tersebut terdapat inpu clk_in, input control (2-bit), dan 1 output register clock_out.

- Inisialisasi clock wizard

```
wire CLK_32MHZ;

clk_wiz_0 clk_wizard_inst (
    .CLK_32MHZ(CLK_32MHZ), // output clock 32MHz
    .clk_in(clk_in)        // input clock 12MHz
);
```

Untuk dapat menggunakan IP clock wizard ke dalam program Verilog, sebuah instance dari clock wizard harus dideklarasikan. Saya membuat wire CLK_32MHZ untuk menyimpan hasil dari clock yang sudah dimultiplikasi dari 12MHz ke 32MHz.

- Register

```
reg[24:0] counter = 25'd0;
reg[24:0] divider = 25'd1;
```

Register counter digunakan untuk menghitung jumlah pulsa posedge dari clock 32 MHz, sedangkan register divider untuk menyimpan nilai pembagi. Pemilihan ukuran register 25 bit karena 2 pangkat 25 memiliki nilai maksimum 33.554.432 sehingga cukup untuk menyimpan nilai 32.000.000 Hz.

- Loop Clock

- Switch case

```
case (control)
        2'b00: divider <= 2;              // 32MHz / 2 = 16MHz
        2'b01: divider <= 128;            // 32MHz / 128 = 250kHz
        2'b10: divider <= 32000;          // 32MHz / 32000 = 1kHz
        2'b11: divider <= 32000000;       // 32MHz / 32000000 = 1Hz
        default: divider <= 25'd1;
    endcase
```

Switch case tersebut merepresentasikan logika yang sesuai dengan tabel 1. Pada switch case tersebut, nilai register divider disesuaikan agar pembagian clock sesuai dengan spesifikasi yang diinginkan.

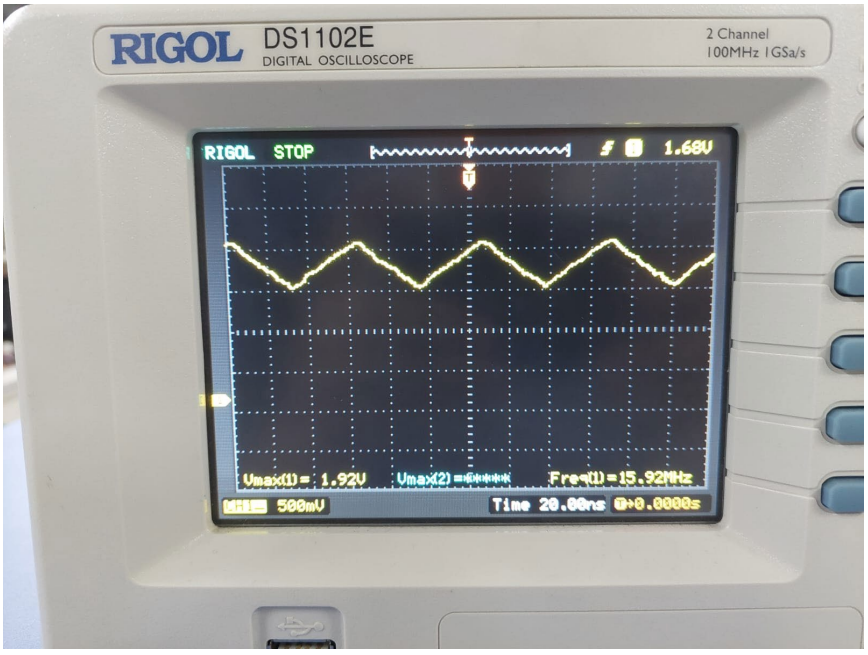
- Perhitungan clock

```
    counter <= counter + 25'd1;
    if(counter >= (divider - 1)) begin
        counter <= 25'd0;
    end

    clock_out <= (counter < (divider / 2)) ? 1'b1 : 1'b0;
```

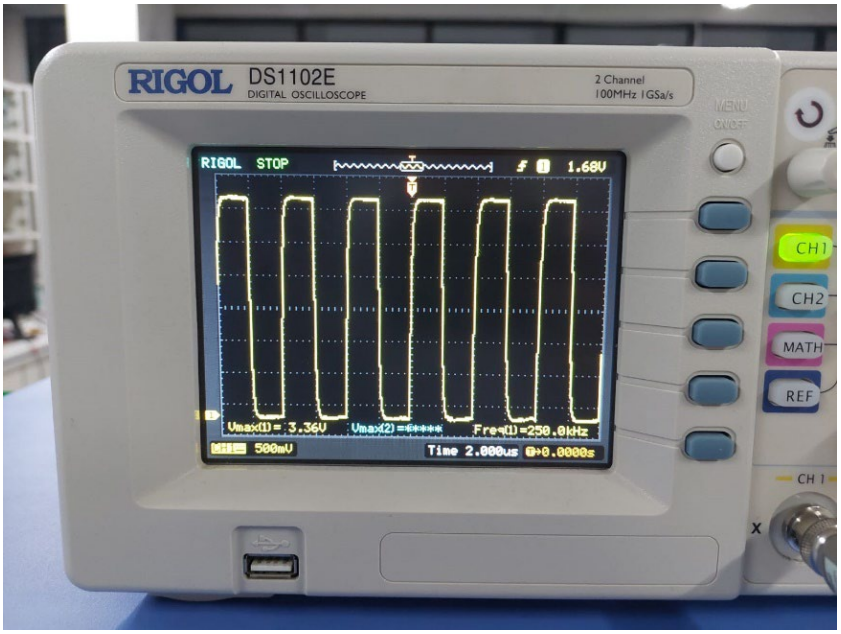
Pada program tersebut, nilai counter terus diincrement setiap positif edge dari CLK_32MHZ. Lalu, nilai counter akan direset menjadi 0 ketika jumlah perhitungan pada counter sudah sama dengan nilai divider. Selanjutnya, clock_out akan bernilai 1 untuk setengah nilai awal dari counter terhadap divider dan akan bernilai 0 untuk sisanya. Hal ini memungkinkan sinyal yang dihasilkan berada pada 50% duty cycle.

Hasil Pengukuran:



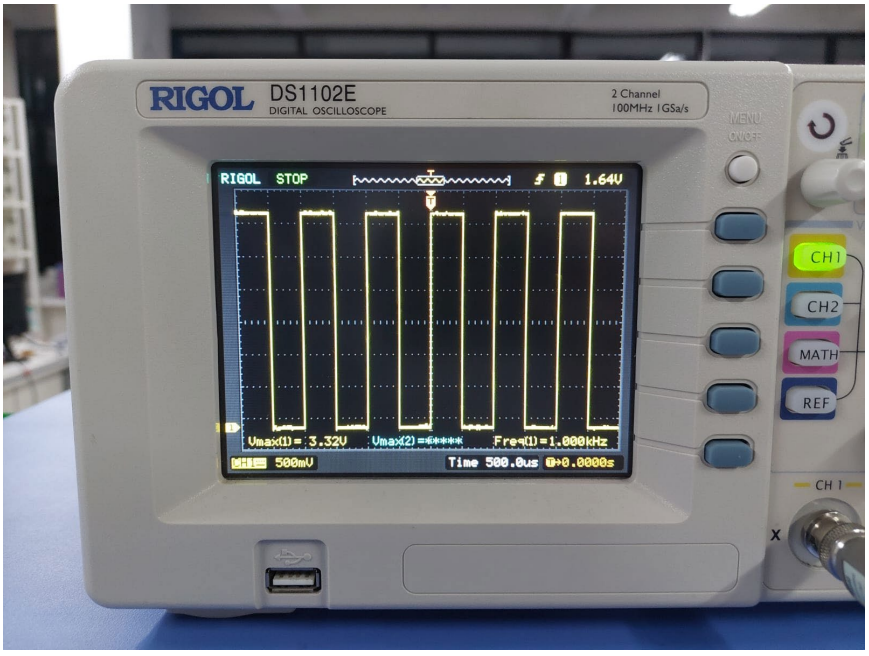
Gambar 4. Clock 16 MHz

Pada clock 16 MHz, sinyal yang dihasilakan tidak sempurna kotak. Sinyal yang dihasilkan berbentuk segitiga dengan frekuensi yang bervariasi sekitar 15,5 MHz hingga 16,5 MHz.



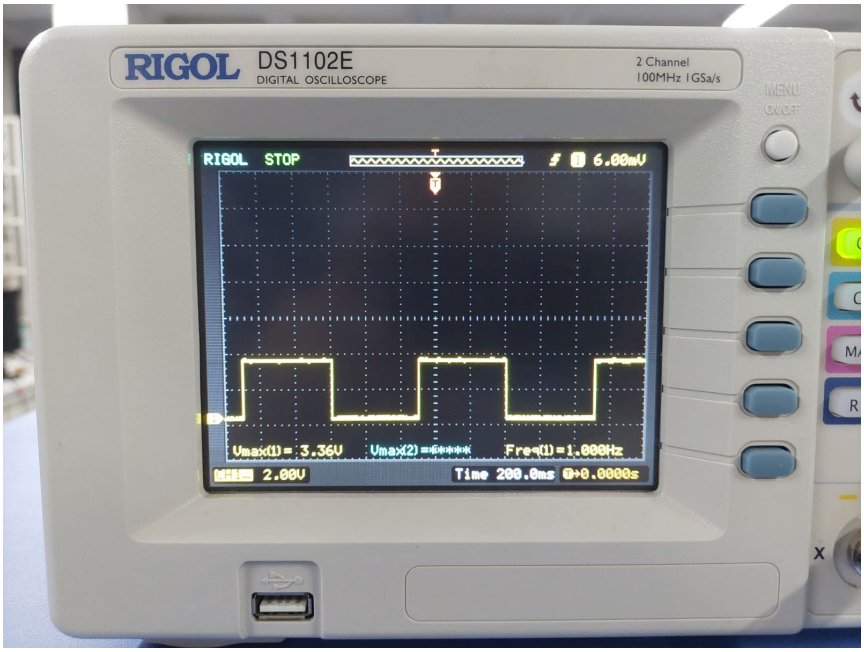
Gambar 5. Clock 250 kHz

Pada clock 250 kHz, frekuensi yang terukur stabil di 250 kHz dengan bentuk gelombang yang masih belum kotak sempurna.



Gambar 6. Clock 1 kHz

Pada clock 1 kHz, bentuk sinyal sudah kotak sempurna dengan frekuensi yang stabil di 1 kHz.



Gambar 7. Clock 1 Hz

Pada output clock 1 Hz, sinyal berbentuk kotak sempurna dengan frekuensi yang stabil di 1 Hz.

Repository:

https://github.com/yohanesstef/clock-divider.git