Laporan 5 Praktikum Sistem Digital

Sidartha Prastya. P - 13219033

*Institut Teknologi Bandung*

*Jalan Ganesha No.10, Bandung, Indonesia*

13219033@std.stei.itb.ac.id

***Abstract*—Laporan ini berisikan simulasi membuat Counter 00 hingga 99 dan sebaliknya menggunakan FPGA IVE Altera dan dibuat menggunakan Quartus. Output berupa dua digit 7-segment LED dan bertambah/berkurang dengan otomatis menggunakan clock.**

***Keywords*— 7-segment, Counter, Clock.**

1. INTRODUCTION

Laporan ini merupakan tugas untuk praktikum 5 Sistem Digital EL2102, yaitu mengenai penggunaan clock untuk membuat counter 2 desimal.

1. SPESIFIKASI

*Hardware* yang digunakan pada praktikum ini adalah satu set FPGA Altera Cyclone EP4CE6E22C8 dengan *software* yang digunakan adalah Intel Quartus. Pada simulasi ini terdapat 3 buah input mekanik berupareset, stop, dan count down.

Hasil yang diinginkan adalah membuat counter 2 desimal naik dan turun dari 00 hingga 99 dan sebaliknya, lalu Kembali ke 00 bila menyentuh 99 atau langsung ke 99 bila menyentuh 00.

1. PERANCANGAN

Pada simulasi ini terdapat 4 input yang terdiri dari 3 input mekanik dan 1 input berupa clock. Lalu, terdapat output berupa 2 buah LED 7-segment. Masing-masing LED terdapat 7 buah segment, akan tetapi karena segment masing-masing LED terhubung menjadi satu, maka pergantian angka dilakukan dengan clock yang sangat cepat sehingga tidak terdeteksi oleh penglihatan manusia.

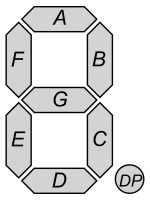


Figure 1. 7-segment LED

1. *Pembagian Segment setiap Digit*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Angka | a | b | c | d | e | f | g |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| 2 | 1 | 1 | 0 | 1 | 1 | 0 | 1 |
| 3 | 1 | 1 | 1 | 1 | 0 | 0 | 1 |
| 4 | 0 | 1 | 1 | 0 | 0 | 1 | 1 |
| 5 | 1 | 0 | 1 | 1 | 0 | 1 | 1 |
| 6 | 1 | 1 | 1 | 1 | 0 | 1 | 1 |
| 8 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 9 | 1 | 1 | 1 | 1 | 0 | 1 | 1 |

Tabel 1. Pembagian Kondisi Segment untuk Setiap Digit

1. *Entity Counter 00-99*

*Diagram, schematic

Description automatically generated*

Figure 2. Entity of Counter 00-99

Pada entity dari desain ini, terdiri dari 3 buah komponen, yaitu

1. Clock Divider

Clock divider berfungsi untuk membagi clock menjadi 2, yaitu 1 Hz dan 1 kHz. Clock 1 Hz digunakan untuk melakukan perubahan pada digit decimal dan 1 kHz digunakan untuk melakukan pergantian LED yang hidup secara bergantian.

Pada FPGA jenis ini memiliki clock sebesar 50 MHz, maka untuk menjadikan clock 1 Hz, maka dibuat variable baru yang akan berubah Ketika counter clock sebesar 25.000.000. Sehingga bila dijumlahkan akan tepat menjadi 1 Hz (naik-turun). Demikian pula untuk clock 1 kHz akan berubah Ketika counter sebesar 25.000.

Diagram

Description automatically generated

Figure 3. Clock Divider

1. Digit Up/Down/Stop

Komponen ini berfungsi untuk melakukan penambahan/pengurangan/diam pada digit yang ditampilkan. Pertambahan dilakukan setiap 1 Hz. Penambahan atau pengurangan ditentukan dengan input berupa *updown* yang bernilai antara 0 dan 1, lalu stop dilakukan apabila input stop bernilai 1.

Diagram

Description automatically generated

Figure 4. Digit Up/Down/Stop

1. Segment On

Pada komponen ini, karena setiap digit pada 7-segment memiliki segment a-g yang menyatu, maka dari itu dibuat system switch setiap 1 kHz agar LED terlihat berganti seperti normal.

Pada komponen ini juga dilakukan decoding dari integer menjadi BCD yang akan dibagi berdasarkan segment a-g.

Diagram

Description automatically generated

Figure 5. Segment on switch

1. *Desain pada FPGA*

Pada rangkaian komponen *push button* akan memberikan input ‘0’ ketika ditekan dan juga sebaliknya. Hal ini juga berlaku untuk LED. Ketika diberikan input berupa ‘0’, maka LED akan menyala, dan juga sebaliknya. Oleh karena itu, semua desain logika yang telah dibuat sebelumnya akan diberikan gerbang “not” dan juga berlaku untuk semua input yang akan diberikan.

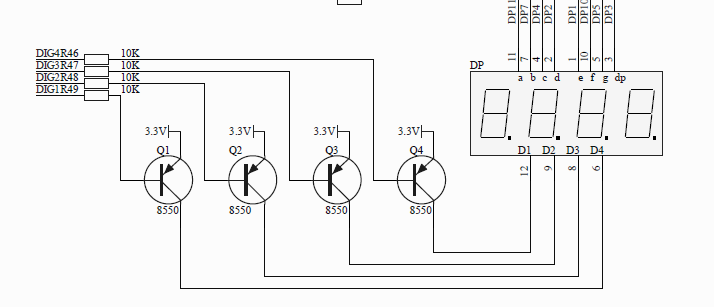


Figure 6. Rangkaian 7-segment LED FPGA

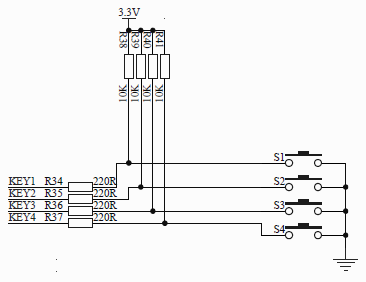


Figure 7. Push Button FPGA

1. IMPLEMENTASI

BEGIN

divclk : clock\_divider port map(

clock => clk,

reset => not reset,

clk1hz => clk1hz,

clk1khz => clk1khz

);

count : digit port map(

clock => clk1hz,

reset => not reset,

updown => up\_down,

stop => not stop,

digit1 => digit1,

digit2 => digit2

);

segment : segment\_on port map(

clk1khz => clk1khz,

digit1 => digit1,

digit2 => digit2,

a => a,

b => b,

c => c,

d => d,

e => e,

f => f,

g => g,

s1 => s1,

s2 => s2

);

end behav;

1. *Program VHDL*

Dalam implementasi meggunakan program VHDL, ditentukan terlebih dahulu entity yang terdiri dari

Input : clock, stop, updown, reset

Output : a, b, c, d, e, f, g, s1, dan s2

library ieee;

use ieee.std\_logic\_1164.all;

entity counter is

port (

clk : in std\_logic;

up\_down : in std\_logic;

reset : in std\_logic;

stop : in std\_logic;

a,b,c,d,e,f,g : out std\_logic;

s1,s2 : out std\_logic

);

end counter;

architecture behav of counter is

signal clk1hz, clk1khz: std\_logic;

signal digit1, digit2 : integer range 0 to 9;

component clock\_divider is

port(

clock : in std\_logic;

reset : in std\_logic;

clk1hz: out std\_logic;

clk1khz: out std\_logic

);

end component;

component digit is

port(

clock : in std\_logic;

reset : in std\_logic;

updown: in std\_logic;

stop : in std\_logic;

digit1: out integer range 0 to

9;

digit2: out integer range 0 to 9

);

end component;

component segment\_on is

port(

clk1khz : in std\_logic;

digit1 : in integer range 0 to 9;

digit2 : in integer range 0 to 9;

a,b,c,d,e,f,g : out std\_logic;

s1, s2 : out std\_logic

);

end component;

1. **Clock\_divider.vhd**

LIBRARY ieee;

USE ieee.std\_logic\_1164.ALL;

ENTITY clock\_divider IS

port(

clock : in std\_logic;

reset : in std\_logic;

clk1hz: out std\_logic;

clk1khz: out std\_logic

);

END clock\_divider;

ARCHITECTURE RTL OF clock\_divider IS

signal pulsehz : std\_logic;

signal pulsekhz: std\_logic;

signal counterhz : integer range 0

to 24999999 := 0;

signal counterkhz : integer range 0

to 24999 := 0;

BEGIN

Process (reset, clock)

Begin

if (reset ='1') then

pulsehz <= '0';

pulsekhz <= '0';

counterhz <= 0;

counterkhz <= 0;

1. **Digit.vhd**

else

pulse2 <= 0;

end if;

end if;

elsif updown ='0' and stop

='0' then --down

if pulse1 > 0 then

pulse1 <= pulse1 - 1;

else

pulse1 <= 9;

if pulse2 > 0 then

pulse2 <= pulse2-1;

else

pulse2 <= 9;

end if;

end if;

else --stop

pulse1 <= pulse1;

pulse2 <= pulse2;

end if;

end if;

end if;

end process;

digit1 <= pulse1;

digit2 <= pulse2;

end RTL;

LIBRARY ieee;

USE ieee.std\_logic\_1164.ALL;

entity digit is

port(

clock : in std\_logic;

reset : in std\_logic;

updown: in std\_logic;

stop : in std\_logic;

digit1: out integer range 0 to 9;

digit2: out integer range 0 to 9

);

end digit;

architecture RTL of digit is

signal pulse1, pulse2 : integer range 0 to 9;

begin

process (reset, clock)

begin

if reset = '1' then

pulse1 <= 0;

pulse2 <= 0;

else

if clock ='1' and clock'event then

if updown ='1' and stop ='0' then

--up

if pulse1 < 9 then

pulse1 <= pulse1 + 1;

else

pulse1 <= 0;

if pulse2 < 9 then

pulse2 <=

pulse2+1;

elsif rising\_edge (clock) then

if counterhz = 24999999 then

pulsehz <= not pulsehz;

counterhz <= 0;

else

counterhz <= counterhz + 1;

end if;

if counterkhz = 24999 then

pulsekhz<= not pulsekhz;

counterkhz <= 0;

else

counterkhz <= counterkhz + 1;

end if;

end if;

end process;

clk1hz <= pulsehz;

clk1khz <= pulsekhz;

END RTL;

1. **Segment\_on.vhd**

LIBRARY ieee;

USE ieee.std\_logic\_1164.ALL;

entity segment\_on is

port(

clk1khz : in std\_logic;

digit1 : in integer range 0 to 9;

digit2 : in integer range 0 to 9;

a,b,c,d,e,f,g : out std\_logic;

s1, s2 : out std\_logic

);

end segment\_on;

architecture RTL of segment\_on is

signal switch : std\_logic := '0';

signal BCD1, BCD2 : std\_logic\_vector (0 to 6);

begin

process(digit1, digit2)

begin

case (digit2) is

when 0 => BCD1 <= "0000001";

when 1 => BCD1 <= "1001111";

when 2 => BCD1 <= "0010010";

when 3 => BCD1 <= "0000110";

when 4 => BCD1 <= "1001100";

when 5 => BCD1 <= "0100100";

when 6 => BCD1 <= "0100000";

when 7 => BCD1 <= "0001111";

when 8 => BCD1 <= "0000000";

when 9 => BCD1 <= "0000100";

when others => BCD1 <= "1111111";

end case;

case (digit1) is

when 0 => BCD2 <= "0000001";

when 1 => BCD2 <= "1001111";

when 2 => BCD2 <= "0010010";

when 3 => BCD2 <= "0000110";

when 4 => BCD2 <= "1001100";

when 5 => BCD2 <= "0100100";

when 6 => BCD2 <= "0100000";

when 7 => BCD2 <= "0001111";

when 8 => BCD2 <= "0000000";

when 9 => BCD2 <= "0000100";

when others => BCD2 <= "1111111";

end case;

end process;

process (clk1khz)

begin

if clk1khz ='1' and clk1khz'event then

if switch ='0' then

s1 <= '0';

s2 <= '1';

a <= BCD1(0);

b <= BCD1(1);

c <= BCD1(2);

d <= BCD1(3);

e <= BCD1(4);

f <= BCD1(5);

g <= BCD1(6);

switch <= not switch;

else

s1 <= '1';

s2 <= '0';

a <= BCD2(0);

b <= BCD2(1);

c <= BCD2(2);

d <= BCD2(3);

e <= BCD2(4);

f <= BCD2(5);

g <= BCD2(6);

switch <= not switch;

end if;

end if;

end process;

end architecture;

*B. Pin Planner*

Gambar 12. Pengujian FPGA dengan Hasil Salah/Lebih dari 9

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *Node Name* | *Direction* | | *Location* | | |
| Clock | Input | | PIN\_23 | | |
| S1 | Output | | PIN\_135 | | |
| S2 | Output | | PIN\_133 | | |
| a | Output | | PIN\_128 | | |
| b | Output | | PIN\_121 | | |
| c | Output | | PIN\_125 | | |
| d | Output | | PIN\_129 | | |
| e | Output | | PIN\_132 | | |
| f | | Output | | PIN\_126 |
| g | | Output | | PIN\_124 |
| Reset | | Input | | PIN\_88 |
| Stop | | Input | | PIN\_89 |
| Updown | | Input | | PIN\_90 |

Tabel di atas adalah penempatan pin pada simulasi VHDL pada FPGA yang digunakan.

V. PENGUJIAN

Berikut adalah gambar pengujian dari beberapa macam counter . Untuk simulasi berupa video, akan disertakan di dalam zip pengumpulan.

A picture containing graphical user interface

Description automatically generatedA picture containing schematic

Description automatically generated

Figure 8. Simulasi pada FPGA