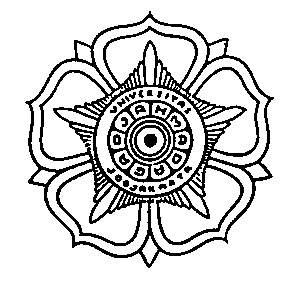
LAPORAN PROJEK

ELEKTONIKA DIGITAL LANJUT

“STOPWATCH”



Disusun oleh :

Yulvi Hidayati 15/386045/SV/09431

Zaki Isra Mustaqim 15/386046/SV/09432

PROGRAM STUDI DIPLOMA ELEKTRONIKA DAN INSTRUMENTASI

DEPARTEMEN TEKNIK ELEKTRO DAN INFORMATIKA

SEKOLAH VOKASI

UNIVERSITAS GADJAH MADA

YOGYAKARTA

2017

DAFTAR ISI

# BAB I

# PENDAHULUAN

## Latar Belakang

Seiring dengan perkembangan teknologi digital saat ini mahasiswa khususnya bidang elektronika harus dapat meningkatakan pemahaman terhadap materi perkuliahan dalam hal ini mata kuliah Elektronika Digital Lanjut untuk dapat merancang suatu sistem digital.

Stopwatch adalah alat yang digunakan untuk mengukur lamanya waktu yang diperlukan dalam suatu kegiatan (anonim) yang dapat dimatikan dan diaktifkan, stopwatch dibagi menjadi dua jenis yakni stopwatch jarum dan *stopwatch* digital. Dalam hal ini penulis ingin memaparkan kan perancangan suatu stopwatch digital dengan memanfatkan FPGA dan *software* Quartus II.

Metode yang penulis ambil dalam merancang *stopwatch*  ini menggunakan pemodelan FSM (*finite state machine)* dengan membagi nya menjadi beberapa bagian yaitu untuk detik satuan ,detik puluhan,dan menit . Total ada tiga buah *seven segment* pada FPGA yang terpakai dengan Batasan waktu yang dirancang enam menit.

***.***

# Rumusan Masalah

Bagaimana membuat rancangan sebuah *Stopwatch* digital yang dapat diimplementasikan pada FPGA?

# Tujuan dan Manfaat

1. Menerapkan teori elektronika digital lanjut dan diimplementasikan secara langsung pada FPGA
2. Dapat membuat suatu rancangan sistem digital
3. Mampu menganalisa sistem yang telah dirancang.

# BAB II

# LANDASAN TEORI

## 2.1 Dasar Teori

### 2.1.1 Field-Programmable Gate Array (FPGA)

Field-Programmable Gate Array (FPGA) merupakan sebuah IC digital yang sering digunakan untuk mengimplementasikan rangkaian digital. FPGA berbentuk komponen [elektronika](https://id.wikipedia.org/wiki/Elektronika) dan [semikonduktor](https://id.wikipedia.org/wiki/Semikonduktor) yang terdiri dari komponen gerbang terprogram ([programmable logic](https://id.wikipedia.org/w/index.php?title=Programmable_logic&action=edit&redlink=1)) dan sambungan terprogram ([interkoneksi](https://id.wikipedia.org/wiki/Interkoneksi)). Komponen gerbang terprogram yang dimiliki meliputi jenis [gerbang logika](https://id.wikipedia.org/wiki/Gerbang_logika) biasa ([AND](https://id.wikipedia.org/w/index.php?title=AND&action=edit&redlink=1), [OR](https://id.wikipedia.org/wiki/OR), [NOT](https://id.wikipedia.org/w/index.php?title=NOT&action=edit&redlink=1)) maupun jenis fungsi matematis dan kombinatorik yang lebih kompleks, seperti decoder, adder, subtractor, multiplier, dll. Blok-blok komponen di dalam FPGA bisa juga mengandung elemen memori ([register](https://id.wikipedia.org/wiki/Register)) mulai dari [flip-flop](https://id.wikipedia.org/wiki/Flip-flop) sampai pada RAM ([Random Access Memory](https://id.wikipedia.org/wiki/Random_Access_Memory)).

FPGA sangat sesuai untuk pemrosesan [komputasi](https://id.wikipedia.org/wiki/Komputasi) dari [algoritme](https://id.wikipedia.org/wiki/Algoritme) [integrasi](https://id.wikipedia.org/w/index.php?title=Integrasi&action=edit&redlink=1) [numerik](https://id.wikipedia.org/w/index.php?title=Numerik&action=edit&redlink=1). Keuntungan implementasi FPGA digunakan untuk meningkatkan efisiensi rancangan dengan cara mengurangi pemakaian pemrograman perangkat lunak ([*software*](https://id.wikipedia.org/wiki/Software)). FPGA mempunyai koreksi [*error*](https://id.wikipedia.org/w/index.php?title=Error&action=edit&redlink=1) yang kecil dan merupakan teknologi yang bebas (*t*[*echnology*](https://id.wikipedia.org/w/index.php?title=Echnology&action=edit&redlink=1)-*independent*) untuk diimplementasikan dalam berbagai [algoritme](https://id.wikipedia.org/wiki/Algoritme). Kinerja aplikasi FPGA lebih cepat dibandingkan dengan aplikasi [mikrokontroler](https://id.wikipedia.org/wiki/Mikrokontroler), karena FPGA hanya mensintesis perangkat keras ([*hardware*](https://id.wikipedia.org/wiki/Hardware)) saja, sementara mikrokontroler mengeksekusi instruksi perangkat lunak ([*software*](https://id.wikipedia.org/wiki/Software)) yang digunakan untuk mengendalikan perangkat keras ([*hardware*](https://id.wikipedia.org/wiki/Hardware)), sehingga waktu tunda yang diimplementasikan hanya memakan waktu tunda perambatan ([propagation delay](https://id.wikipedia.org/w/index.php?title=Propagation_delay&action=edit&redlink=1)) saja.

Pemodelan FPGA membutuhkan informasi terkait dengan tingkat perbedaan abstraksi dan jenis model yang digunakan. Seorang perancang FPGA harus mampu mengambil beberapa tahapan pemodelan untuk memastikan hasil model rancangannya melalui model simulasi yang telah disediakan oleh [vendor](https://id.wikipedia.org/w/index.php?title=Vendor&action=edit&redlink=1) FPGA masing-masing.

### 2.1.2 Seven Segment

**Pengertian Seven Segment Display** – Seven Segment Display (7 Segment Display) dalam bahasa Indonesia disebut dengan Layar Tujuh Segmen adalah komponen Elektronika yang dapat menampilkan angka desimal melalui kombinasi-kombinasi segmennya. Seven Segment Display pada umumnya dipakai pada Jam Digital, Kalkulator, Penghitung atau Counter Digital, Multimeter Digital dan juga Panel Display Digital seperti pada Microwave Oven ataupun Pengatur Suhu Digital . Seven Segment Display pertama diperkenalkan dan dipatenkan pada tahun 1908 oleh Frank. W. Wood dan mulai dikenal luas pada tahun 1970-an setelah aplikasinya pada LED (Light Emitting Diode).Terdapat 2 Jenis LED 7 Segmen, diantaranya adalah “LED 7 Segmen common Cathode” dan “LED 7 Segmen common Anode”.

### 2.1.3 Rangkaian Kombinasional

Rangkaian kombinasional terdiri dari gerbang logika yang memiliki output yang selalu tergantung pada kombinasi input yang ada. Rangkaian kombinasional melakukan operasi yang dapat ditentukan secara logika dengan memakai sebuah fungsi boolean. ada prinsipnya, rangkaian kombinasional merupakan penerapan dan penerjemah langsung dari aljabar boole, yang biasanya dinyatakan sebagai fungsi logika. Operator logika yang digunakan dalam aljabar boole adalah inversi/negasi (NOT), perkalian logika (AND), penambahan logika (OR).

Contoh dari rangkaian kombinasional :

1. Multiplexer

Rangkaian logika kombinasional Multiplexer atau disingkat MUX adalah alat atau komponen elektronika yang bisa memilih input (masukan) yang akan diteruskan ke bagian output (keluaran). Pemilihan input mana yang dipilih akan ditentukan oleh signal yang ada di bagian kontrol (kendali) Select.

1. Demultiplexer

Rangkaian logika kombinasional Demultiplekser adalah Komponen yang berfungsi kebalikan dari MUX. Pada DEMUX, jumlah masukannya hanya satu, tetapi bagian keluarannya banyak. Signal pada bagian input ini akan disalurkan ke bagian output (channel) yang mana tergantung dari kendali pada bagian SELECTnya.

### 2.1.4 Dekoder BCD to 7 Segment

Rangkaian bcd to 7 segment decoder merupakan rangkaian digital yang mempunyai kemampuan untuk mengkonversi bilangan bcd menjadi bilangan desimal, yang fungsinya sebagai alat untuk mempermudah konversi bilangan. Proses pengkodean data BCD menjadi tampilan angka desimal **dilakukan secara terpisah untuk tiap ruas/segment** (ruas a sampai ruas g).

### 2.1.5 FSM (finited state machine)

Finite State Machines (**FSM**) adalah sebuah metodologi perancangan sistem kontrol yang menggambarkan tingkah laku atau prinsip kerja sistem dengan menggunakan tiga hal berikut: State (Keadaan), Event (kejadian) dan action (aksi).

# BAB III

# METODOLOGI PENELITIAN

## 3.1 Alat dan Bahan

1. Hardware FPGA

2. Personal Computer (PC)

3. Software Quartus II Altera

## 3.2 Rancangan Sistem :

Dalam merancang stopwatch digital ini penulis membaginya dalam beberapa tahapan :

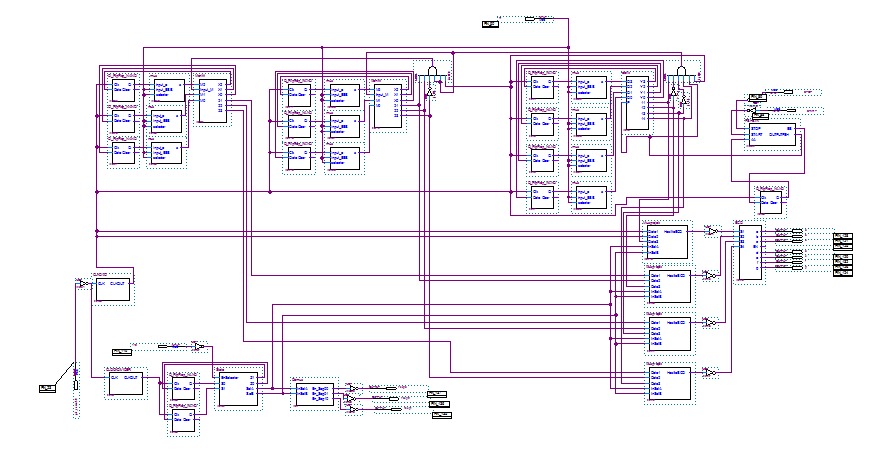
1. Grand Design

Dalam membuat suatu rancangan sistem hal utama dan yang pertama dilakukan adalah melihat gambaran besar (*grand design)* dari sistem yang akan kita bangun. Kita spesifikasi dulu apa saja yang kita mau dari sistem kita dengan batasan-batasannya. Kemudian kita menstrategikan bagaimana kita akan merealisasikan desain tersebut.

Spesisfikasi sistem *stopwatch* yang kami rancang pada projek ini terdapat *Output-*nyatiga buah seven segment dan *input*-nya tiga *push button.* Tiga tombol *input* sebagai start,stop dan reset. Proses yang dilakukan harus mampu menghitung waktu dari 0 sampai 6 menit yang di tampilkan dengan 000 sampai 600. Tombol start untk memulai proses perhitungan, stop untuk berhenti dan menahan display, dan reset untuk mengembalikan nilai pada kedaan 0.

1. Realisasi

Untuk dapat mewujudkan rancangan tersebut kami menyiapkan beberapa *library* sistem rangkaian yang harus dimiliki yang diambil dari praktikum-praktikum sebelumnya yaitu : *clock divider*, BCD *(binary code decimal)to seven segment)*, *D-flipflop*, *multiplexer* 2x1, *multiplexer* 3x1 dan *Demultiplexer* 2x3. Library yang disebutkan tersebut sudah ada, dan langkah selanjutnya membuat rangkaian kombinasional untuk *stopwatch-*nya dengan metode FSM.



### 3.2.1 Gambar desain rangkaian

### 3.2.2Flowchart Sistem

Menit&detik 0 (7-segmnet 000)

ya

RESET

RANDOM  
NUMBER

START

tidak

tidak

START

ya

Menghitung  
CounterUP

Berhenti Menghitung CounterUp

tidak

STOP

RESET

ya

ya

RESET

tidak

tidak

START

ya

### 3.2.3 VHDL (Source Code)

-- Copyright (C) 1991-2011 Altera Corporation

-- Your use of Altera Corporation's design tools, logic functions

-- and other software and tools, and its AMPP partner logic

-- functions, and any output files from any of the foregoing

-- (including device programming or simulation files), and any

-- associated documentation or information are expressly subject

-- to the terms and conditions of the Altera Program License

-- Subscription Agreement, Altera MegaCore Function License

-- Agreement, or other applicable license agreement, including,

-- without limitation, that your use is for the sole purpose of

-- programming logic devices manufactured by Altera and sold by

-- Altera or its authorized distributors. Please refer to the

-- applicable agreement for further details.

-- PROGRAM "Quartus II 64-Bit"

-- VERSION "Version 11.0 Build 157 04/27/2011 SJ Full Version"

-- CREATED "Mon Dec 04 06:23:28 2017"

LIBRARY ieee;

USE ieee.std\_logic\_1164.all;

LIBRARY work;

ENTITY DONE IS

PORT

(

119 : IN STD\_LOGIC;

R : IN STD\_LOGIC;

STOP : IN STD\_LOGIC;

START : IN STD\_LOGIC;

pin\_name39 : IN STD\_LOGIC;

7seg3 : OUT STD\_LOGIC;

7seg1 : OUT STD\_LOGIC;

7seg2 : OUT STD\_LOGIC;

a : OUT STD\_LOGIC;

b : OUT STD\_LOGIC;

c : OUT STD\_LOGIC;

d : OUT STD\_LOGIC;

e : OUT STD\_LOGIC;

f : OUT STD\_LOGIC;

g : OUT STD\_LOGIC

);

END DONE;

ARCHITECTURE bdf\_type OF DONE IS

ATTRIBUTE black\_box : BOOLEAN;

ATTRIBUTE noopt : BOOLEAN;

COMPONENT mux\_0

PORT(input\_a : IN STD\_LOGIC;

input\_BBB : IN STD\_LOGIC;

selector : IN STD\_LOGIC;

z : OUT STD\_LOGIC);

END COMPONENT;

ATTRIBUTE black\_box OF mux\_0: COMPONENT IS true;

ATTRIBUTE noopt OF mux\_0: COMPONENT IS true;

COMPONENT mux\_1

PORT(input\_a : IN STD\_LOGIC;

input\_BBB : IN STD\_LOGIC;

selector : IN STD\_LOGIC;

z : OUT STD\_LOGIC);

END COMPONENT;

ATTRIBUTE black\_box OF mux\_1: COMPONENT IS true;

ATTRIBUTE noopt OF mux\_1: COMPONENT IS true;

COMPONENT mux\_2

PORT(input\_a : IN STD\_LOGIC;

input\_BBB : IN STD\_LOGIC;

selector : IN STD\_LOGIC;

z : OUT STD\_LOGIC);

END COMPONENT;

ATTRIBUTE black\_box OF mux\_2: COMPONENT IS true;

ATTRIBUTE noopt OF mux\_2: COMPONENT IS true;

COMPONENT mux\_3

PORT(input\_a : IN STD\_LOGIC;

input\_BBB : IN STD\_LOGIC;

selector : IN STD\_LOGIC;

z : OUT STD\_LOGIC);

END COMPONENT;

ATTRIBUTE black\_box OF mux\_3: COMPONENT IS true;

ATTRIBUTE noopt OF mux\_3: COMPONENT IS true;

COMPONENT mux\_4

PORT(input\_a : IN STD\_LOGIC;

input\_BBB : IN STD\_LOGIC;

selector : IN STD\_LOGIC;

z : OUT STD\_LOGIC);

END COMPONENT;

ATTRIBUTE black\_box OF mux\_4: COMPONENT IS true;

ATTRIBUTE noopt OF mux\_4: COMPONENT IS true;

COMPONENT mux\_5

PORT(input\_a : IN STD\_LOGIC;

input\_BBB : IN STD\_LOGIC;

selector : IN STD\_LOGIC;

z : OUT STD\_LOGIC);

END COMPONENT;

ATTRIBUTE black\_box OF mux\_5: COMPONENT IS true;

ATTRIBUTE noopt OF mux\_5: COMPONENT IS true;

COMPONENT mux\_6

PORT(input\_a : IN STD\_LOGIC;

input\_BBB : IN STD\_LOGIC;

selector : IN STD\_LOGIC;

z : OUT STD\_LOGIC);

END COMPONENT;

ATTRIBUTE black\_box OF mux\_6: COMPONENT IS true;

ATTRIBUTE noopt OF mux\_6: COMPONENT IS true;

COMPONENT mux\_7

PORT(input\_a : IN STD\_LOGIC;

input\_BBB : IN STD\_LOGIC;

selector : IN STD\_LOGIC;

z : OUT STD\_LOGIC);

END COMPONENT;

ATTRIBUTE black\_box OF mux\_7: COMPONENT IS true;

ATTRIBUTE noopt OF mux\_7: COMPONENT IS true;

COMPONENT mux\_8

PORT(input\_a : IN STD\_LOGIC;

input\_BBB : IN STD\_LOGIC;

selector : IN STD\_LOGIC;

z : OUT STD\_LOGIC);

END COMPONENT;

ATTRIBUTE black\_box OF mux\_8: COMPONENT IS true;

ATTRIBUTE noopt OF mux\_8: COMPONENT IS true;

COMPONENT mux\_9

PORT(input\_a : IN STD\_LOGIC;

input\_BBB : IN STD\_LOGIC;

selector : IN STD\_LOGIC;

z : OUT STD\_LOGIC);

END COMPONENT;

ATTRIBUTE black\_box OF mux\_9: COMPONENT IS true;

ATTRIBUTE noopt OF mux\_9: COMPONENT IS true;

COMPONENT demux

PORT(InSelA : IN STD\_LOGIC;

InSelB : IN STD\_LOGIC;

En\_Seg00 : OUT STD\_LOGIC;

En\_Seg01 : OUT STD\_LOGIC;

En\_Seg10 : OUT STD\_LOGIC

);

END COMPONENT;

COMPONENT menittt

PORT(M2 : IN STD\_LOGIC;

Input\_M : IN STD\_LOGIC;

M1 : IN STD\_LOGIC;

M0 : IN STD\_LOGIC;

X2 : OUT STD\_LOGIC;

X1 : OUT STD\_LOGIC;

X0 : OUT STD\_LOGIC;

21 : OUT STD\_LOGIC;

22 : OUT STD\_LOGIC;

23 : OUT STD\_LOGIC

);

END COMPONENT;

COMPONENT muxprojek

PORT(Data1 : IN STD\_LOGIC;

Data2 : IN STD\_LOGIC;

Data3 : IN STD\_LOGIC;

InSelA : IN STD\_LOGIC;

InSelB : IN STD\_LOGIC;

HasiltoBCD : OUT STD\_LOGIC

);

END COMPONENT;

COMPONENT d\_flipflop\_nand

PORT(Clk : IN STD\_LOGIC;

Data : IN STD\_LOGIC;

Q : OUT STD\_LOGIC;

Qbar : OUT STD\_LOGIC

);

END COMPONENT;

COMPONENT detik

PORT(D3 : IN STD\_LOGIC;

D2 : IN STD\_LOGIC;

D1 : IN STD\_LOGIC;

D0 : IN STD\_LOGIC;

P : IN STD\_LOGIC;

Y3 : OUT STD\_LOGIC;

Y2 : OUT STD\_LOGIC;

Y1 : OUT STD\_LOGIC;

Y0 : OUT STD\_LOGIC;

11 : OUT STD\_LOGIC;

12 : OUT STD\_LOGIC;

13 : OUT STD\_LOGIC;

14 : OUT STD\_LOGIC

);

END COMPONENT;

COMPONENT state

PORT(EnSelector : IN STD\_LOGIC;

S0 : IN STD\_LOGIC;

S1 : IN STD\_LOGIC;

Z1 : OUT STD\_LOGIC;

Z0 : OUT STD\_LOGIC;

SelA : OUT STD\_LOGIC;

SelB : OUT STD\_LOGIC

);

END COMPONENT;

COMPONENT bcd

PORT(S1 : IN STD\_LOGIC;

S2 : IN STD\_LOGIC;

S3 : IN STD\_LOGIC;

S4 : IN STD\_LOGIC;

a : OUT STD\_LOGIC;

b : OUT STD\_LOGIC;

c : OUT STD\_LOGIC;

EN : OUT STD\_LOGIC;

d : OUT STD\_LOGIC;

e : OUT STD\_LOGIC;

f : OUT STD\_LOGIC;

g : OUT STD\_LOGIC

);

END COMPONENT;

COMPONENT pshbtn

PORT(STOP : IN STD\_LOGIC;

START : IN STD\_LOGIC;

AA : IN STD\_LOGIC;

BB : OUT STD\_LOGIC;

OUTPUTPSH : OUT STD\_LOGIC

);

END COMPONENT;

COMPONENT clkdvd

PORT(CLK : IN STD\_LOGIC;

CLKOUT : OUT STD\_LOGIC

);

END COMPONENT;

COMPONENT clockdivider

PORT(CLK : IN STD\_LOGIC;

CLKOUT : OUT STD\_LOGIC

);

END COMPONENT;

SIGNAL SYNTHESIZED\_WIRE\_105 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_106 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_2 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_107 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_4 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_5 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_108 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_109 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_12 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_14 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_16 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_17 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_19 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_110 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_111 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_112 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_113 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_25 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_114 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_30 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_31 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_32 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_33 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_35 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_37 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_39 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_40 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_41 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_42 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_43 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_115 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_46 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_47 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_48 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_116 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_50 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_52 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_53 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_54 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_56 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_57 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_58 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_117 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_63 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_68 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_69 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_70 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_71 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_72 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_73 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_74 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_75 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_77 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_79 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_80 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_81 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_82 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_118 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_85 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_86 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_88 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_89 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_90 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_91 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_92 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_93 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_94 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_95 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_96 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_98 : STD\_LOGIC;

SIGNAL SYNTHESIZED\_WIRE\_99 : STD\_LOGIC;

BEGIN

b2v\_inst : demux

PORT MAP(InSelA => SYNTHESIZED\_WIRE\_105,

InSelB => SYNTHESIZED\_WIRE\_106,

En\_Seg00 => SYNTHESIZED\_WIRE\_53,

En\_Seg01 => SYNTHESIZED\_WIRE\_54,

En\_Seg10 => SYNTHESIZED\_WIRE\_57);

b2v\_inst12 : menittt

PORT MAP(M2 => SYNTHESIZED\_WIRE\_2,

Input\_M => SYNTHESIZED\_WIRE\_107,

M1 => SYNTHESIZED\_WIRE\_4,

M0 => SYNTHESIZED\_WIRE\_5,

X2 => SYNTHESIZED\_WIRE\_12,

X1 => SYNTHESIZED\_WIRE\_14,

X0 => SYNTHESIZED\_WIRE\_16,

21 => SYNTHESIZED\_WIRE\_114,

22 => SYNTHESIZED\_WIRE\_117,

23 => SYNTHESIZED\_WIRE\_115);

b2v\_inst13 : muxprojek

PORT MAP(Data1 => SYNTHESIZED\_WIRE\_108,

Data2 => SYNTHESIZED\_WIRE\_108,

Data3 => SYNTHESIZED\_WIRE\_109,

InSelA => SYNTHESIZED\_WIRE\_105,

InSelB => SYNTHESIZED\_WIRE\_106,

HasiltoBCD => SYNTHESIZED\_WIRE\_72);

b2v\_inst17 : d\_flipflop\_nand

PORT MAP(Clk => SYNTHESIZED\_WIRE\_108,

Data => SYNTHESIZED\_WIRE\_12,

Q => SYNTHESIZED\_WIRE\_94);

b2v\_inst18 : d\_flipflop\_nand

PORT MAP(Clk => SYNTHESIZED\_WIRE\_108,

Data => SYNTHESIZED\_WIRE\_14,

Q => SYNTHESIZED\_WIRE\_95);

b2v\_inst19 : d\_flipflop\_nand

PORT MAP(Clk => SYNTHESIZED\_WIRE\_108,

Data => SYNTHESIZED\_WIRE\_16,

Q => SYNTHESIZED\_WIRE\_96);

SYNTHESIZED\_WIRE\_107 <= SYNTHESIZED\_WIRE\_17 AND SYNTHESIZED\_WIRE\_109 AND SYNTHESIZED\_WIRE\_19 AND SYNTHESIZED\_WIRE\_110 AND SYNTHESIZED\_WIRE\_111 AND SYNTHESIZED\_WIRE\_108;

SYNTHESIZED\_WIRE\_19 <= NOT(SYNTHESIZED\_WIRE\_112);

SYNTHESIZED\_WIRE\_17 <= NOT(SYNTHESIZED\_WIRE\_113);

b2v\_inst23 : muxprojek

PORT MAP(Data1 => SYNTHESIZED\_WIRE\_25,

Data2 => SYNTHESIZED\_WIRE\_114,

Data3 => SYNTHESIZED\_WIRE\_112,

InSelA => SYNTHESIZED\_WIRE\_105,

InSelB => SYNTHESIZED\_WIRE\_106,

HasiltoBCD => SYNTHESIZED\_WIRE\_73);

b2v\_inst24 : menittt

PORT MAP(M2 => SYNTHESIZED\_WIRE\_30,

Input\_M => SYNTHESIZED\_WIRE\_31,

M1 => SYNTHESIZED\_WIRE\_32,

M0 => SYNTHESIZED\_WIRE\_33,

X2 => SYNTHESIZED\_WIRE\_35,

X1 => SYNTHESIZED\_WIRE\_37,

X0 => SYNTHESIZED\_WIRE\_39,

21 => SYNTHESIZED\_WIRE\_25,

22 => SYNTHESIZED\_WIRE\_58,

23 => SYNTHESIZED\_WIRE\_63);

b2v\_inst25 : d\_flipflop\_nand

PORT MAP(Clk => SYNTHESIZED\_WIRE\_108,

Data => SYNTHESIZED\_WIRE\_35,

Q => SYNTHESIZED\_WIRE\_85);

b2v\_inst26 : d\_flipflop\_nand

PORT MAP(Clk => SYNTHESIZED\_WIRE\_108,

Data => SYNTHESIZED\_WIRE\_37,

Q => SYNTHESIZED\_WIRE\_86);

b2v\_inst27 : d\_flipflop\_nand

PORT MAP(Clk => SYNTHESIZED\_WIRE\_108,

Data => SYNTHESIZED\_WIRE\_39,

Q => SYNTHESIZED\_WIRE\_89);

b2v\_inst3 : detik

PORT MAP(D3 => SYNTHESIZED\_WIRE\_40,

D2 => SYNTHESIZED\_WIRE\_41,

D1 => SYNTHESIZED\_WIRE\_42,

D0 => SYNTHESIZED\_WIRE\_43,

P => SYNTHESIZED\_WIRE\_111,

Y3 => SYNTHESIZED\_WIRE\_56,

Y2 => SYNTHESIZED\_WIRE\_77,

Y1 => SYNTHESIZED\_WIRE\_88,

Y0 => SYNTHESIZED\_WIRE\_98,

11 => SYNTHESIZED\_WIRE\_109,

12 => SYNTHESIZED\_WIRE\_112,

13 => SYNTHESIZED\_WIRE\_113,

14 => SYNTHESIZED\_WIRE\_110);

SYNTHESIZED\_WIRE\_99 <= NOT(SYNTHESIZED\_WIRE\_115);

b2v\_inst34 : state

PORT MAP(EnSelector => SYNTHESIZED\_WIRE\_46,

S0 => SYNTHESIZED\_WIRE\_47,

S1 => SYNTHESIZED\_WIRE\_48,

Z1 => SYNTHESIZED\_WIRE\_52,

Z0 => SYNTHESIZED\_WIRE\_50,

SelA => SYNTHESIZED\_WIRE\_105,

SelB => SYNTHESIZED\_WIRE\_106);

b2v\_inst35 : d\_flipflop\_nand

PORT MAP(Clk => SYNTHESIZED\_WIRE\_116,

Data => SYNTHESIZED\_WIRE\_50,

Q => SYNTHESIZED\_WIRE\_47);

b2v\_inst36 : d\_flipflop\_nand

PORT MAP(Clk => SYNTHESIZED\_WIRE\_116,

Data => SYNTHESIZED\_WIRE\_52,

Q => SYNTHESIZED\_WIRE\_48);

SYNTHESIZED\_WIRE\_46 <= NOT(119);

7seg3 <= NOT(SYNTHESIZED\_WIRE\_53);

7seg2 <= NOT(SYNTHESIZED\_WIRE\_54);

b2v\_inst4 : d\_flipflop\_nand

PORT MAP(Clk => SYNTHESIZED\_WIRE\_108,

Data => SYNTHESIZED\_WIRE\_56,

Q => SYNTHESIZED\_WIRE\_90);

7seg1 <= NOT(SYNTHESIZED\_WIRE\_57);

b2v\_inst41 : muxprojek

PORT MAP(Data1 => SYNTHESIZED\_WIRE\_58,

Data2 => SYNTHESIZED\_WIRE\_117,

Data3 => SYNTHESIZED\_WIRE\_113,

InSelA => SYNTHESIZED\_WIRE\_105,

InSelB => SYNTHESIZED\_WIRE\_106,

HasiltoBCD => SYNTHESIZED\_WIRE\_74);

b2v\_inst42 : muxprojek

PORT MAP(Data1 => SYNTHESIZED\_WIRE\_63,

Data2 => SYNTHESIZED\_WIRE\_115,

Data3 => SYNTHESIZED\_WIRE\_110,

InSelA => SYNTHESIZED\_WIRE\_105,

InSelB => SYNTHESIZED\_WIRE\_106,

HasiltoBCD => SYNTHESIZED\_WIRE\_75);

b2v\_inst43 : bcd

PORT MAP(S1 => SYNTHESIZED\_WIRE\_68,

S2 => SYNTHESIZED\_WIRE\_69,

S3 => SYNTHESIZED\_WIRE\_70,

S4 => SYNTHESIZED\_WIRE\_71,

a => a,

b => b,

c => c,

d => d,

e => e,

f => f,

g => g);

SYNTHESIZED\_WIRE\_68 <= NOT(SYNTHESIZED\_WIRE\_72);

SYNTHESIZED\_WIRE\_69 <= NOT(SYNTHESIZED\_WIRE\_73);

SYNTHESIZED\_WIRE\_70 <= NOT(SYNTHESIZED\_WIRE\_74);

SYNTHESIZED\_WIRE\_71 <= NOT(SYNTHESIZED\_WIRE\_75);

SYNTHESIZED\_WIRE\_81 <= NOT(START);

SYNTHESIZED\_WIRE\_80 <= NOT(STOP);

b2v\_inst5 : d\_flipflop\_nand

PORT MAP(Clk => SYNTHESIZED\_WIRE\_108,

Data => SYNTHESIZED\_WIRE\_77,

Q => SYNTHESIZED\_WIRE\_91);

b2v\_inst50 : d\_flipflop\_nand

PORT MAP(Clk => SYNTHESIZED\_WIRE\_108,

Data => SYNTHESIZED\_WIRE\_79,

Q => SYNTHESIZED\_WIRE\_82);

SYNTHESIZED\_WIRE\_118 <= NOT(pin\_name39);

b2v\_inst52 : pshbtn

PORT MAP(STOP => SYNTHESIZED\_WIRE\_80,

START => SYNTHESIZED\_WIRE\_81,

AA => SYNTHESIZED\_WIRE\_82,

BB => SYNTHESIZED\_WIRE\_79,

OUTPUTPSH => SYNTHESIZED\_WIRE\_111);

b2v\_inst56 : clkdvd

PORT MAP(CLK => SYNTHESIZED\_WIRE\_118,

CLKOUT => SYNTHESIZED\_WIRE\_108);

b2v\_inst57 : clockdivider

PORT MAP(CLK => SYNTHESIZED\_WIRE\_118,

CLKOUT => SYNTHESIZED\_WIRE\_116);

b2v\_inst58 : mux\_0

PORT MAP(input\_a => SYNTHESIZED\_WIRE\_85,

input\_BBB => R,

selector => R,

z => SYNTHESIZED\_WIRE\_30);

b2v\_inst59 : mux\_1

PORT MAP(input\_a => SYNTHESIZED\_WIRE\_86,

input\_BBB => R,

selector => R,

z => SYNTHESIZED\_WIRE\_32);

b2v\_inst6 : d\_flipflop\_nand

PORT MAP(Clk => SYNTHESIZED\_WIRE\_108,

Data => SYNTHESIZED\_WIRE\_88,

Q => SYNTHESIZED\_WIRE\_92);

b2v\_inst60 : mux\_2

PORT MAP(input\_a => SYNTHESIZED\_WIRE\_89,

input\_BBB => R,

selector => R,

z => SYNTHESIZED\_WIRE\_33);

b2v\_inst62 : mux\_3

PORT MAP(input\_a => SYNTHESIZED\_WIRE\_90,

input\_BBB => R,

selector => R,

z => SYNTHESIZED\_WIRE\_40);

b2v\_inst63 : mux\_4

PORT MAP(input\_a => SYNTHESIZED\_WIRE\_91,

input\_BBB => R,

selector => R,

z => SYNTHESIZED\_WIRE\_41);

b2v\_inst64 : mux\_5

PORT MAP(input\_a => SYNTHESIZED\_WIRE\_92,

input\_BBB => R,

selector => R,

z => SYNTHESIZED\_WIRE\_42);

b2v\_inst65 : mux\_6

PORT MAP(input\_a => SYNTHESIZED\_WIRE\_93,

input\_BBB => R,

selector => R,

z => SYNTHESIZED\_WIRE\_43);

b2v\_inst66 : mux\_7

PORT MAP(input\_a => SYNTHESIZED\_WIRE\_94,

input\_BBB => R,

selector => R,

z => SYNTHESIZED\_WIRE\_2);

b2v\_inst67 : mux\_8

PORT MAP(input\_a => SYNTHESIZED\_WIRE\_95,

input\_BBB => R,

selector => R,

z => SYNTHESIZED\_WIRE\_4);

b2v\_inst68 : mux\_9

PORT MAP(input\_a => SYNTHESIZED\_WIRE\_96,

input\_BBB => R,

selector => R,

z => SYNTHESIZED\_WIRE\_5);

b2v\_inst7 : d\_flipflop\_nand

PORT MAP(Clk => SYNTHESIZED\_WIRE\_108,

Data => SYNTHESIZED\_WIRE\_98,

Q => SYNTHESIZED\_WIRE\_93);

SYNTHESIZED\_WIRE\_31 <= SYNTHESIZED\_WIRE\_99 AND SYNTHESIZED\_WIRE\_114 AND SYNTHESIZED\_WIRE\_117 AND SYNTHESIZED\_WIRE\_107 AND SYNTHESIZED\_WIRE\_107 AND SYNTHESIZED\_WIRE\_108;

END bdf\_type;

## 3.3 Pembagian Pengerjaan Project

Tabel 1. Pembagian Tugas

|  |  |  |
| --- | --- | --- |
| NO | Nama Mahasiswa | Deskripsi Pekerjaan |
| 1 | Yulvi Hidayati | 1. Menuangkan ide dan gambaran *grand design* sistem  2. Membuat library-library sistem yang diperlukan  3. Menyelesaikan laporan projek |
| 2 | Zaki Isra Mustaqim | 1. mengintegrasikan perancang sistem secara keseluruhan  2. Membuat rangkaian kombinasional stopwatch menit, detik puluhan dan detik satuan dengan metode FSM, tabel kebenaran, dan fungsi rangkaian. |

# BAB IV

# HASIL DAN PEMBAHASAN

## Tabel Kebenaran dan Penjelasan

1. Tabel kebenaran tombol



Hasil persamaan rangkaian:

N = P+AO’

Output = A

1. Tabel kebenaran detik satuan

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| INPUT | | | | | OUTPUT | | | | | | | | |
| A | B | C | D | E | | N3 | N2 | N1` | N0 | 1 | 2 | 3 | 4 |
| 0 | 0 | 0 | 0 | 0 | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 1 | | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 | | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 |
| 0 | 0 | 0 | 1 | 1 | | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 | 0 | | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 | 1 | | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 1 | 0 | | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 |
| 0 | 0 | 1 | 1 | 1 | | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 0 | 0 | | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 |
| 0 | 1 | 0 | 0 | 1 | | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 |
| 0 | 1 | 0 | 1 | 0 | | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 |
| 0 | 1 | 0 | 1 | 1 | | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 | 0 | | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 1 | 0 | 1 | | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 0 |
| 0 | 1 | 1 | 1 | 0 | | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 |
| 0 | 1 | 1 | 1 | 1 | | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 | 0 | | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 1 | | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 0 | | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 |

|  |  |
| --- | --- |
| N3=AD' + AE' + BCDE | |
| N2=BC' + BD' + BE' + B'CDE | |
| N1=CD' + CE' + A'C'DE | |
| N0=D'E + DE' |  |

Hasil persamaan rangkaian:

OUTPUT BCD :

* 1. A
  2. B
  3. C
  4. D

1. Tabel kebenaran detik puluhan

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| A | B | C | D | N2 | N1 | N0 | 1 | 2 | 3 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 |
| 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 1 |
| 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 |
| 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |
| 1 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 |
| 1 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 1 |

Hasil persamaan rangkaian:

N2= AC' + AD' + BCD

N1= BC' + BD' + A'B'CD

N0= C'D + CD

OUTPUT

1= A

2= B

3= C

1. Tabel kebenaran menit

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| A | B | C | D | N2 | N1 | N0` | 1 | 2 | 3 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 |
| 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 1 |
| 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 |
| 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |
| 1 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 |
| 1 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 1 |
| 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 |

Hasil persamaan rangkaian:

|  |
| --- |
| N2= AB' + AD' + BCD |
| N1= BD' + B'CD + A'BC' |
| N0= CD' + A'C'D + B'C'D |
|  |
| OUTPUT   1. A 2. B 3. C |

Penjelasan Sistem

Rangkaian disusun dengan menggabungkan semua FSM yang telah dibuat dari Tabel tombol dengan 2 *input* dimana ketika P ditekan sekali maka akan menghasilkan O bernilai 1. Sedangkan jika S ditekan sekali maka nilai O akan berubah bernilai 0. *Output* dari rangkaian FSM ini akan menjadi *input* E untuk rangkaian FSM dari Tabel detik satuan. Sehingga ketika nilai O bernilai 1 maka rangkaian detik satuan akan berjalan counter up dari 0 sampai 9. Ketika *counter up* hasil rangkaian detik satuan menghasilkan keluaran 9 (1001) dan CLK bernilai 1 maka akan menghasilkan nilai 1 sebagai input rangkaian FSM dari table detik puluhan. Sehingga rangkaian detik puluhan akan *counter* *up* hanya ketika hasil rangkaian detik satuan bernilai 9 dan CLK bernilai 1. Begitu juga pada rangkaian FSM dari tabel menit. Rangkaian menit akan counter up hanya ketika rangkaian detik puluhan bernilai 5 (101) dan CLK bernilai 1. Sedangkan untuk *RESET* dibutuhkan *multiplexer* dengan fungsi ketika nilai *selector* 1 maka nilai *Next State* pada semua FSM akan bernilai 0.

## Pengujian Sistem

Pengujian mendapatkan hasil yang sesuai dengan apa yang diharapkan. Simulasi pada FPGA berjalan dengan lancar dengan hasil 3 *input* tombol *SET*,*STOP* dan *RESET*. Tetapi nilai awal yang tampil pada FPGA ketika pertama kali di jalankan menghasilkan angka yang acak pada penampil *7-segment*. Sehingga diharuskan melakukan *RESET* pada langkah awal setelah dinyalakan. Karena terjadinya kelalaian serta kesalahan komunikasi antar tim maka hasil gambar FPGA tidak terdokumentasi dan tidak dapat kami sertakan.

# BAB V

# PENUTUP

Kesimpulan

1. Mahaiswa telah mampu menerapkan dasar teori praktikum untuk membuat suatu sistem rancangan design digital

2. Untuk merancang design stopwatch diperlukan beberapa komponen sistem pendukung yaitu clock divider, multiplexer, rangkaian kombinasional stopwatch dibat berdasarkan metode FSM

3. Stopwatch digital yang diimplementasikan pada FPGA dengan metode FSM berhasil berjalan sesuai rancangan dan teori dasar.