TUGAS AKHIR

RANCANG BANGUN JAM DIGITAL BERBASIS FPGA

Diajukan Sebagai Salah Satu Syarat
Untuk Memperoleh Gelar SarjanaTeknik Elektro

Disusun Oleh :

Nama : Doddy Firmansyah

No.Mhs : 01 524 006

JURUSAN TEKNIK ELEKTRO
FAKULTAS TEKNOLOGI INDUSTRI
UNIVERSITAS ISLAM INDONESIA
YOGYAKARTA
2007

LEMBAR PENGESAHAN PEMBIMBING

RANCANG BANGUN JAM DIGITAL BERBASIS FPGA

TUGAS AKHIR

Disusun Oleh :

Nama : Doddy Firmansyah

No.Mhs : 01 524 006

Telah dikonsultasikan dan disetujui oleh pembimbing skripsi
Jurusan Teknik Elektro Universitas Islam Indonesia Yogyakarta

Yogyakarta, Januari 2007

Menyetujui / Mengesahkan

Dosen Pembimbing I	Dosen Pembimbing II
(Tito Yuwono, ST. M.Sc)	(Medilla Kusriyanto, ST)

LEMBAR PENGESAHAN PENGUJI

RANCANG BANGUN JAM DIGITAL BERBASIS FPGA

TUGAS AKHIR

Disusun Oleh :

Nama : Doddy Firmansyah

No.Mhs : 01 524 006

Telah Dipertahankan di Depan Sidang Sebagai Salah Satu Syarat Untuk Memperoleh Gelar Sarjana Teknik Elektro Fakultas Teknologi Industri Universitas Islam Indonesia

Yogyakarta, Maret 2007

Tim Penguji

Tito Yuwono, ST. M.Sc
Ketua

Medilla Kusriyanto, ST
Anggota I

Yusuf Aziz Amrullah, ST.
Anggota II

Mengetahui
Ketua Jurusan Teknik Elektro
Universitas Islam Indonesia

(Tito Yuwono, ST. M.Sc)

HALAMAN PERSEMBAHAN

بِسْمِ اللَّهِ الرَّحْمَٰنِ الرَّحِيمِ

DENGAN RASA SYUKUR DAN KERENDAHAN HATI
HASIL KARYA INI SAYA PERSEMBAHKAN KEPADA :

Keluarga yang aku sayangi bapak, ibu dan adik-adikku yang telah mendidik dan menyayangi dengan penuh kesabaran dan rasa kasih sayangnya yang tiada terkira dan Saudara-saudaraku yang telah mendukungku

MOTTO

- Bila ingin buah yang baik, tanamlah bibit yang baik dan harus dipupuk oleh pupuk yang dibeli dari hasil yang halal.

(Anonim)

- Sesungguhnya semua nikmat yang diberikan oleh Allah SWT senantiasa harus kita syukuri.

(Anonim)

- Aku adalah aku, perbuatanku menjadi tanggung jawabku.

(Anonim)

- Hanya kepada-Mu-lah kami menyembah dan hanya kepada-Mu-lah kami mohon pertolongan.

(Al'Fatihah)

- Teruslah berusaha dan jangan pernah menyerah .....

KATA PENGANTAR

بِسْمِ اللهِ الرَّحْمٰنِ الرَّحِيْمِ

Assalamu'alaikum Wr.Wb.

Puji syukur penulis panjatkan kehadirat ALLAH SWT yang telah melimpah ruahkan segala karuniaNya, sehingga penulis dapat menyelesaikan Tugas Akhir ini. Shalawat dan salam semoga tercurah pada junjungan Rasulullah Muhammad SAW beserta keluarga, dan pengikutnya.

Dengan dilaksanakannya penelitian dalam bentuk Tugas Akhir ini penulis dapat belajar banyak tentang VHDL, Xilinx dan FPGA. Melalui Tugas Akhir ini pula penulis dapat menerapkan ilmu-ilmu yang didapat dibangku kuliah dengan kenyataan pada saat pembuatan Tugas Akhir, sehingga memberikan pengalaman yang sangat berharga.

Pada kesempatan ini penulis ingin mengucapkan terimakasih kepada beberapa pihak, yaitu :

1. Bapak Fathul Wahid, ST, M.Sc, selaku Dekan Fakultas Teknologi Industri yang telah memberikan izin untuk melakukan Tugas Akhir ini.
2. Bapak Tito Yuwono, ST, M.Sc, selaku Ketua Jurusan Teknik Elektro yang telah berkenan membuka cakrawala keilmuan Teknik Elektro.
3. Bapak Medilla Kusriyanto, ST, dan Bapak Tito Yuwono, ST. M.Sc, selaku Dosen Pembimbing yang telah banyak memberikan bimbingan dan banyak memberikan dukungan moril dalam pelaksanaan Tugas Akhir ini.
4. Semua Dosen Jurusan Teknik Elektro yang telah membantu memberikan ide dan informasi yang diperlukan untuk penyusunan Tugas Akhir ini.

5. Mas Heri Kendali, Mas Tri Telkom dan Mas Agung TTL yang telah banyak memberikan bantuan selama kami berada di Laboratorium.
6. Teman-teman seperjuangan Sany, Amir, Whinar, Enggar, Cahyo, Mukti, Ucup, Nanda, Adi, Dana dan semua rekan-rekan elektro UII pada umumnya yang telah membantu dalam penyusunan tugas akhir ini.
7. Ayahanda-ibunda, adik-adikku, atas doa dan kasih sayangnya yang tidak pernah putus.
8. Pujaan Hatiku (Fitri Hidayah), yang sudah banyak membantu dan selalu memberikan semangat dalam penyusunan tugas akhir ini.
9. Pihak-pihak yang tidak mungkin penulis sebutkan satu persatu.

Semoga Allah SWT memberikan balasan limpahan rahmat dan karunia serta kelapangan hati atas segala kebaikan yang mereka berikan.

Penulis menyadari bahwa Tugas Akhir ini masih banyak terdapat kekurangannya, untuk itu sangat diharapkan saran dan kritik yang sekiranya dapat menambah pengetahuan serta lebih menyempurnakan Tugas Akhir ini. Semoga apa yang telah penulis ketengahkan ini dapat bermanfaat bagi kita semua.

Wasalamu'alaikum Wr.Wb

Yogyakarta, Pebruari 2007

Penulis

ABSTRAK

Pada zaman modern sekarang ini, masyarakat menuntut tersedianya kemudahan disegala bidang. Demikian halnya kemudahan dalam pembacaan waktu. Seperti halnya sebuah penunjuk waktu analog yang terdapat pada pusat kota. Karena penunjuk waktu yang digunakan masih berupa analog, maka pembacaan waktu terkadang kurang tepat. Dengan adanya perkembangan teknologi dapat dirancang sebuah jam digital dalam sebuah *IC* dengan menggunakan FPGA. Yang mana FPGA tersebut merupakan *hardware* yang berfungsi sebagai sebuah *IC*. Keluaran dan masukan dari FPGA tersebut sebagai pengendali dari sebuah jam digital. Penampil yang digunakan pada tugas akhir ini adalah *seven segment*.

Dengan menggunakan *seven segment* sebagai penampil, maka pembacaan waktu tidak dipengaruhi oleh kondisi gelap (tanpa cahaya), sehingga pada kondisi ruangan gelap tetap dapat mengetahui waktu pada saat itu. Penggunaan jam digital ini sangat fleksibel dalam artian kondisi gelap maupun terang, waktu tetap dapat dibaca. Dibandingkan jam analog, jam digital hanya membutuhkan sedikit tempat.

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I

PENDAHULUAN

1.1. Latar Belakang

Dunia telah menunjukan adanya perkembangan teknologi yang semakin meningkat dan semakin canggih terutama dalam perancangan *IC*. Dengan berkembangnya teknologi perancangan *IC*, maka semakin berkembang pula *hardware* yang dihasilkan. Seiring dengan kemajuan teknologi tersebut manusia dituntut untuk bekerja lebih disiplin dalam segi waktu. Salah satunya yaitu penggunaan jam digital. Disamping penampilan lebih mewah dibandingkan dengan jam analog, jam digital juga dapat membantu lebih tepat dalam pembacaan waktu.

Pada penelitian ini akan difokuskan pada perancangan jam digital dalam sebuah *IC* dengan menggunakan FPGA. Yang mana FPGA tersebut merupakan *hardware* yang berfungsi sebagai sebuah *IC*. Keluaran dan masukan dari FPGA tersebut sebagai pengendali dari sebuah jam digital. Penampil yang digunakan pada penelitian ini adalah *seven segment.*

Dengan menggunakan *seven segment* sebagai penampil, maka pembacaan waktu tidak dipengaruhi oleh kondisi gelap (tanpa cahaya), sehingga pada kondisi ruangan gelap tetap dapat mengetahui waktu pada saat itu. Penggunaan jam digital ini sangat fleksibel dalam artian kondisi gelap maupun terang, waktu tetap dapat dibaca. Dibandingkan jam analog, jam digital hanya membutuhkan sedikit tempat.

1.2. Rumusan Masalah

Berdasarkan latar belakang yang telah dijelaskan sebelumnya, maka dapat diambil suatu rumusan masalah: "Bagaimana merancang jam digital dalam satu buah *IC* dengan menggunakan FPGA".

1.3. Batasan Masalah

Agar permasalahan yang dibahas pada penulisan tidak menyimpang dari apa yang diteliti, maka akan membatasi sebagai berikut :

1. Penulisan hanya difokuskan pada perancangan *IC* jam digital yang menggunakan perangkat keras FPGA, perangkat lunak VHDL, dan Xilink ISE 7.
2. FPGA yang digunakan adalah pegasus yang memakai *IC* XC2S50.
3. Fitur yang terdapat pada jam digital hanya berupa alarm yang dapat diatur secara fleksibel.
4. Format waktu pada jam digital menggunakan waktu 24 jam bukan AM atau PM.

1.4. Tujuan Penelitian

Mewujudkan sebuah *IC* yang berisi jam digital.

1.5. Manfaat Penelitian

Manfaat penelitian ini adalah: diperoleh *prototype* IC jam digital.

1.6. Metodologi Penelitian

1. Studi Kepustakaan

   Studi ini dilakukan dengan cara melihat dan mencari literatur yang ada untuk memperoleh data yang berhubungan dengan alat yang dibuat.

2. Metode observasi

   Metode ini adalah melakukan penelitian dan mempelajari peralatan yang sudah ada untuk memberikan gambaran yang jelas sehingga dapat dipakai sebagai acuan dalam perencanaan dan pembuatan alat.

3. Perencanaan Rangkaian

   Perencanaan rangkaian diperlukan untuk mendapatkan hasil rangkaian yaitu dengan memodifikasi rangkaian-rangkaian yang berhubungan dengan Tugas Akhir ini.

4. Pengukuran dan Pengujian alat

   Metode ini meliputi pengujian alat sehingga diperoleh data-data dari hasil pengujian alat tersebut, dan sekaligus mendapatkan informasi yang baik juga akurat serta dapat dipertanggung jawabkan.

5. Penyusunan Laporan

   Setelah dilakukan pengujian alat, data dan informasi yang diperoleh dianalisa, kemudian disusun dalam sebuah laporan.

I.7. Sistematika Penulisan

Sistematika Tugas akhir ini terdiri dari 5 bab bagian isi laporan, dengan penjelasan bab sebagai berikut :

BAB I : PENDAHULUAN

Berisi tentang latar belakang masalah, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, metodologi penelitian dan sistematika penulisan.

BAB II : DASAR TEORI

Bab ini memuat teori-teori yang berhubungan dengan penelitian dan juga berisi dasar teori yang berhubungan dengan fungsi atau piranti yang akan digunakan.

BAB III : PERANCANGAN SISTEM

Bab ini menjelaskan seluruh metode-metode perancangan yang digunakan, cara mengimplementasikan rancangan dan pengujian sistem dengan perangkat lunak VHDL dan Xilinx ISE 7.

BAB IV : ANALISIS DAN PEMBAHASAN

Bab ini membahas tentang hasil pengujian dan analisis dari sistem yang dibuat dibandingkan dengan dasar teori sistem dan hasil pengujian akhir di level perangkat keras dengan FPGA. Selain itu juga akan dibahas batasan dan hambatan yang ditemui selama proses perancangan dan implementasi sistem.

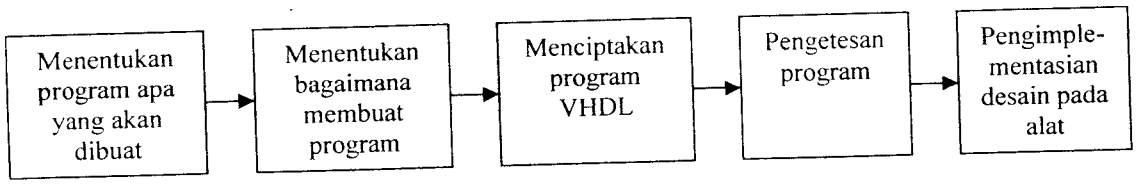
BAB II

DASAR TEORI

2.1. VHDL

VHDL merupakan salah satu jenis *hardware deskripcion language* (HDL) yang digunakan untuk memprogram PLD. V pada singkatan VHDL adalah VHSIC (*Very Hidh Speed Integrated Circuit*). VHDL adalah standar bahasa yang digunakan pada IEEE (*Institute of Elactrical and Electronic Enginers*) dan juga salah satu yang dapat digunakan untuk HDL. VHDL adalah bahasa pemrograman tingkat tinggi yang menggunakan pernyataan-pernyataan yang sama dengan bahasa Inggris. Dengan pernyataan bahasa tingkat tinggi, VHDL dapat dengan mudah dibaca dan dimengerti bahkan jika pembaca tidak pandai dalam bahasa pemrograman. Program VHDL dapat dikembangkan melalui pemrograman umum dan langkah-langkahnya dapat dilihat pada Gambar 2.1.

1. Mendefinisikan apa fungsi dari program tersebut
2. Menentukan solusi bagaimana tugas akan diselesaikan
3. Membuat program
4. Menguji dan mensimulasi program dan jika ada revisi penulisan
5. Implementasi desain



Gambar 2.1. Langkah-langkah pemrograman pada program VHDL

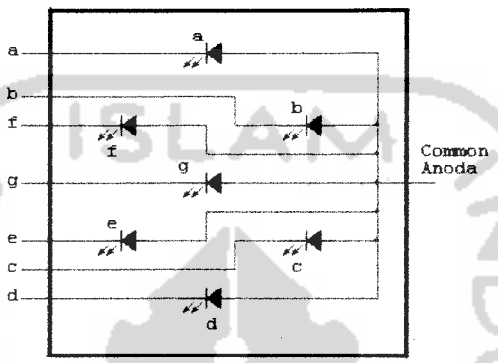
2.2. Xilinx

Xilinx *(Xilinx Foundation Series)* adalah suatu perangkat lunak yang berfungsi untuk merancang satu buah *IC*, yang nantinya hasil dari perancangan tersebut didownload ke FPGA. Dengan menggunakan Xilinx, proses simulasi rangkaian yang telah dirancang dapat diketahui apakah sudah benar atau masih mengandung kesalahan.

Proses perancangan dengan menggunakan Xilinx sama seperti merancang suatu rangkaian logika secara manual, akan tetapi dengan menggunakan simulator Xilinx dapat meminimalisasi kesalahan pada proses perancangan. Untuk proses perancangan rangkaian digital, Xilinx mempunyai tiga cara, yaitu dengan menggunakan State Diagram, HDL (*Hardware Description Language*) dan *Schematic*, yang mana pada proses perancangan bisa menggunakan salah satu cara saja atau menggabungkan ketiga cara tersebut.

2.3.1. *Seven Segment Common Anoda*

Seven segment common anoda adalah peraga *seven segment* dimana semua *anoda* digabungkan satu sama lain dan dikeluarkan sebagai hubungan tunggal (*common*). Masukan pada sebelah kiri menjadi ruas-ruas dan penampil Gambar 2.3 merupakan gambar rangkaian peraga *seven segment common anoda*.



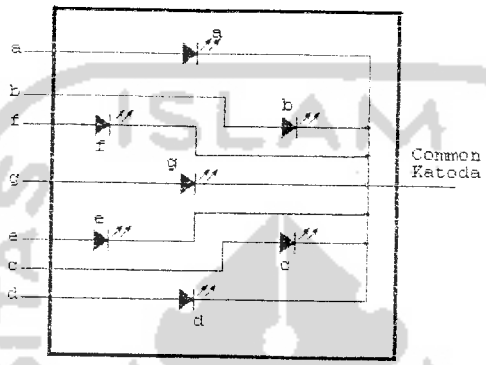
Gambar 2.3. Rangkaian *seven segment common anoda*

Tabel 2.1. Masukan *seven segment common anoda*

Tampilan	A	B	C	D	E	F	G
0	0	0	0	0	0	0	1
1	1	0	0	1	1	1	1
2	0	0	1	0	0	1	0
3	0	0	0	0	1	1	0
4	1	0	0	1	0	0	0
5	0	1	0	0	1	0	0
6	0	1	0	0	0	0	0
7	0	0	0	1	1	1	1
8	0	0	0	0	0	0	0
9	0	0	0	0	1	0	0

2.3.2. *Seven Segment Common Katoda*

Semua *katoda* pada *seven segment common katoda* digabungkan satu sama lain dan keluaran pada sisi kanan sebagai hubungan tunggal (pemakaian bersama). Masukan sisi kiri sebagai ruas-ruas dari penampil dan sisi kanan sebagai *katoda*.



Gambar 2.4. Rangkaian *seven segment common katoda*

Tabel 2.2. Masukan *seven segment common katoda*

Tampilan	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	0	1	1	1	1	1
7	1	1	1	0	0	0	0
8	1	1	1	1	1	1	1
9	1	1	1	1	0	1	1

2.4. Saklar Tombol *Push Button*

Saklar tombol *push button* adalah bentuk yang paling umum dari pemutus dan penyambung arus secara manual yang sering dijumpai di industri. Saklar tombol *push* button pada kondisi awalnya NO (*Normally Open*), maka ketika ditekan akan menyambung atau menghubungkan rangkaian. Begitu pula sebaliknya jika saklar tombol *push button* pada kondisi awalnya NC (*Normally Close*), maka ketika ditekan akan membuka atau memutus rangkaian dan kembali pada posisi tertutup ketika tombol dilepas.

Gambar 2.5. Saklar tombol *push button*

2.5. *Field Programmable Gate Array* (FPGA)

Field Programmable Gate Array merupakan sebuah piranti digital yang dapat diprogram untuk merepresentasikan sistem logika yang telah dirancang. Teknologi IC FPGA diperkenalkan pada tahun 1985 oleh perusahaan semikonduktor Xilinx. FPGA adalah sebuah konsep teknologi IC yang dapat diprogram dan dihapus seperti halnya

RAM. FPGA kemudian berkembang pesat, baik dari segi kepadatan gerbang, kecepatan dan disertai dengan penurunan harga jual.

Penemuan FPGA telah membuat peningkatan yang pesat akan pembuatan *prototipe* beberapa sistem digital. Salah satu produsen FPGA yang ada dipasaran adalah Xilinx, disamping produsen lainnya Actel dan Altera. Prinsip dasar dari pemrograman atau pengkonfigurasian FPGA Xilinx adalah pengubahan gambar rangkaian elektronik digital dari perangkat lunak Xilinx menjadi file aliran bit (*bitstream*) dan dikonfigurasi (*didownload*) ke dalam IC FPGA Xilinx tersebut sehingga IC tersebut terkonfigurasi secara perangkat keras yang dirancang dalam perangkat lunak Xilinx. FPGA produk Xilinx sudah melewati beberapa generasi antara lain XC2000, XC3000 dan XC4000. Tiap generasi memiliki sifat dan kemampuan yang berbeda. Perbedaan tersebut meliputi kecepatan, kapasitas gerbang logika, jumlah CLB dan jumlah IOB.

2.5.1. FPGA keluarga Xilinx Spartan II

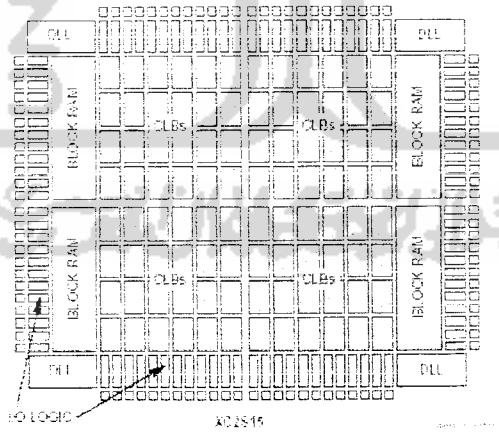
Spartan II merupakan salah satu keluarga FPGA yang dikeluarkan oleh Xilinx. Keluarga Spartan II merupakan keluarga FPGA yang memiliki 15.000 sampai 200.000 gerbang. IC Xilinx ini dapat diprogram dan dihapus dengan waktu yang tidak terbatas serta murah harganya. Kelurga Spartan ini dapat diprogram dengan mudah menggunakan Xilinx *Development System* ataupun dengan *development system* lain yang dikembangkan oleh para pengguna.

Tabel 2.3. Data Keluarga Spartan II

Device	Logic Cells	System Gates (Logic and RAM)	CLB Array (R x C)	Total CLBs	Maximum Available User I/O[1]	Total Distributed RAM Bits	Total Block RAM Bits
XC2S15	432	15,000	8 x 12	96	86	6,144	16K
XC2S30	972	30,000	12 x 18	216	132	13,824	24K
XC2S50	1,728	50,000	16 x 24	384	176	24,576	32K
XC2S100	2,700	100,000	20 x 30	600	196	38,400	40K
XC2S150	3,888	150,000	24 x 36	864	260	55,296	48K
XC2S200	5,292	200,000	28 x 42	1,176	284	75,264	56K

2.5.2. Struktur Dasar Keluarga Spartan II

Suatu piranti FPGA terdiri atas *Configurable Logic Blocks* (CLB), unit *input/output*, empat buah *Delay-Locked Loops* (DLLs), unit RAM dan unit routing yang dapat diprogram secara otomatis penuh. Susunan dan letak masing-masing bagian tersebut dapat dilihat pada Gambar 2.6.

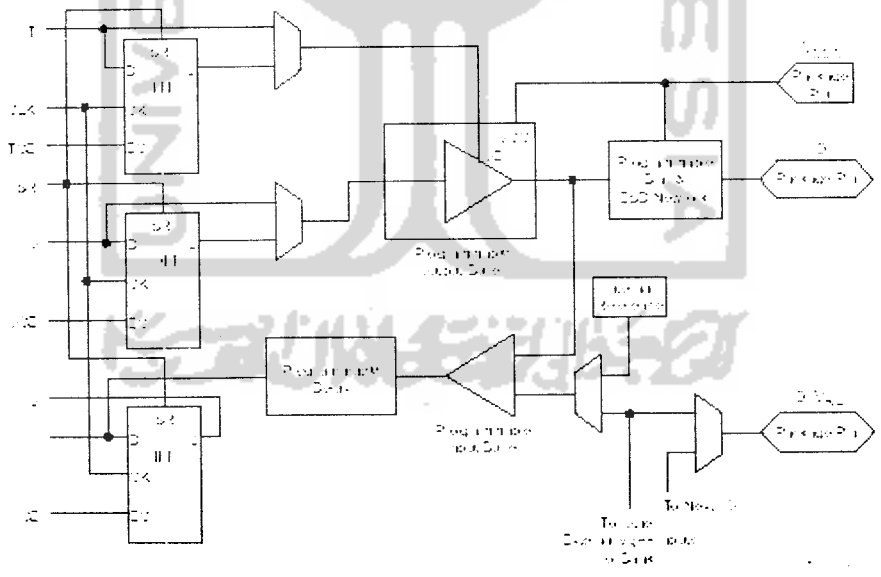


Gambar 2.6. Diagram blok dasar keluarga Spartan II

2.5.2.1 *Input/Output Blocks* (IOB)

Input/Output blocks merupakan bagian dari FPGA yang berfungsi menghubungkan FPGA dengan piranti lain yang akan terkoneksi IOB keluarga Spartan II mampu bekerja pada berbagai macam standar I/O seperti TTL, CMOS dan PCI. Kemampuan untuk menyesuaikan dengan berbagai macam I/O didukung dengan kemampuan tiap *pad* I/O untuk ditambahi *pull-up* dan *pull-down resistor*.

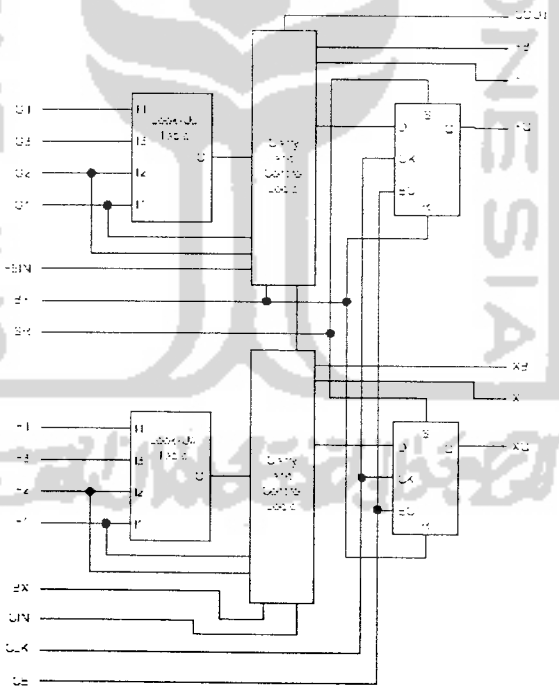
Bagian *buffer* pada Spartan II IOB *input path* akan menghubungkan sinyal *input* yang masuk secara langsung dengan logika internal atau secara tidak langsung melalui *input flip-flop optional*. Sedangkan bagian *output path* termasuk buffer 3 keadaan akan men-*drive* sinyal *output* menuju *pad output*. Sama dengan sinyal *input*, sinyal *output* ini dapat dihubungkan dengan *pad output* melalui logika internal maupun melalui *output flip-flop optional*.



Gambar 2.7. Blok IOB Spartan II

2.5.2.2 *Configurable Logic Blocks* (CLB)

CLB merupakan bagian dari FPGA yang berfungsi merubah logika-logika terprogram yang dimasukkan menjadi fungsi-fungsi yang dipahami oleh FPGA dan dapat bekerja sesuai dengan program yang diinginkan. CLB Spartan II terdiri atas *Logic Cell (LC)* sebagai bangunan utama. Sebuah LC terdiri atas 4 buah *input* yang akan membangkitkan fungsi logika yang diinginkan, *carry logic* dan elemen penyimpan. Keluaran setiap LC akan men-*drive* keluaran CLB dan *input* pada D *flip-flop*. Setiap CLB pada Spartan II terdiri atas empat LC yang tersusun dalam dua *slices* yang identik. Tiap CLB ini juga memuat logika yang akan mengkombinasi generator pembangkit fungsi logika untuk 4 sampai 6 input.



Gambar 2.8. CLB pada Spartan II

Generator pembangkit fungsi diimplementasikan dalam sebuah *look-up tables* (LUT) 4 input. LUT ini juga dapat membangkitkan sebuah RAM 16 x 1 sinkron serta membangkitkan fungsi *shift register* 16 bit. Fungsi RAM yang dibangkitkan generator ini akan melengkapi *block RAM* yang dimiliki oleh Spartan II sehingga mampu menghasilkan unit penyimpan data yang handal.

2.5.2.3 *Programmable Routing Matrix*

Programmable Routing Matrix merupakan cara sebuah FPGA melakukan *routing* menghubungkan CLB-CLB dan IOB-IOB yang digunakan dalam desain menjadi satu kesatuan sistem. *Routing* ini dilakukan secara otomatis penuh. Namun untuk keperluan tertentu, optimasi jalur yang paling pendek dapat dilakukan *routing* manual.

Dalam keluarga Spartan II ada beberapa macam routing yang bisa digunakan yaitu:

1. *Local Routing*

   Local Routing digunakan untuk mengimplementasikan hubungan antara LUT, *flip-flop* dan *General Routing Matrix* (GRM), antara jalur umpan balik internal CLB dengan LUT lain pada CLB yang sama untuk koneksi *high speed,* serta antara jalur-jalur langsung yang bisa dibuat untuk memperkecil *delay*.

Gambar 2.9. Struktur *local routing*

2. *General Purpose Routing*

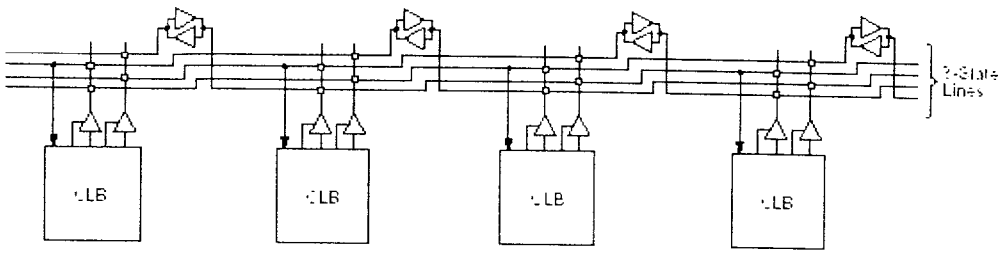
   General Purpose Routing digunakan untuk melakukan koneksi vertikal dan horizontal antar kolom dan baris CLB yang digunakan.

3. *I/O Routing*

   I/O Routing khusus digunakan untuk menghubungkan *array* pada CLB dengan IOB.

4. *Dedicated Routing*

   Dedicated Routing digunakan untuk menguhubungkan beberapa CLB, IOB ataupun LUT yang memerlukan perlakuan khusus untuk memaksimalkan performa.



Gambar 2.10. Koneksi BUFT untuk *dedicated Horizontal Bus Line*

5. *Global Routing*

   Global Routing digunakan untuk menghubungkan *clock* dengan bagian yang membutuhkan serta sinyal-sinyal dengan *fan-out* yang tinggi ke bagian lain.

2.5.3. Pemrograman FPGA

Keluarga Spartan II yang dikeluarkan oleh Xilinx dapat dengan mudah diprogram menggunakan *development system software* keluaran Xilinx yaitu Xilinx *Foundation Series*. Saat ini versi yang terbaru adalah versi 7.1. *Software* memiliki kemampuan *place and route tools* (PAR) yang memungkinkan penggunanya menyusun IOB, LUT dan CLB menjadi sebuah kesatuan sistem.

FPGA dapat diprogram menggunakan metode *schematic* ataupun menggunakan *Hardware Description Language*, baik Verilog maupun VHDL. Masing-masing metode memiliki kelebihan dan kelemahannya sendiri.

2.5.4. Mode Operasi

Keluarga Spartan II dapat dioperasikan dalam 4 mode yaitu:

1. *Slave Serial mode*
2. *Master Serial mode*
3. *Slave Parallel mode*
4. *Boundary Scan mode*

Mode yang paling mudah digunakan adalah *boudary scan mode* dimana tidak diperlukan koneksi-koneksi khusus, cukup menggunakan kabel paralel yang dikoneksikan menggunakan JTAG, maka desain dapat dengan mudah diimplementasikan ke dalam FPGA.

Tabel 2.4. Konfigurasi Pin yang terdapat pada Modul Pegasus

Pegasus Expansion Connector Pinout								
Connector B1			Connector A1			Connector A2		
Pin	signal	B1	Pin	signal	A1	Pin	signal	A2
39	TDO	TDO	39	TDO	TDO	39	GCK0	GCK0
40	TDI	TDI	40	TDI	TDI	40	GND	GND
37	TMS	TMS	37	TMS	TMS	37	n/c	n/c
38	TCK	TCK	38	TCK	TCK	38	n/c	n/c
35	MB1-INIT	90	35	MA1-INIT	189	35	MA2-INT	138
36	GND	GND	36	GND	GND	36	Not used	n/c
33	MB1-WAIT	95	33	MA1-WAIT	192	33	MA2-WAIT	140
34	M1-RST	94	34	M1-RST	191	34	MA2-RST	139
31	MB1-DSTB	97	31	MA1-DSTB	194	31	MA2-DSTB	142
32	MB1-WRIT	96	32	MA1-WRIT	193	32	MA2-WRIT	141
29	MB1-DB7	99	29	MA1-DB7	199	29	MA2-DB7	147
30	MB1-ASTB	98	30	MA1-ASTB	195	30	MA2-ASTB	146
27	MB1-DB5	101	27	MA1-DB5	201	27	MA2-DB5	149
28	MB1-DB6	100	28	MA1-DB6	200	28	MA2-DB6	148
25	MB1-DB3	108	25	MA1-DB3	203	25	MA2-DB3	151
26	MB1-DB4	102	26	MA1-DB4	202	26	MA2-DB4	150
23	MB1-DB1	110	23	MA1-DB1	205	23	MA2-DB1	160
24	MB1-DB2	109	24	MA1-DB2	204	24	MA2-DB2	152
21	P-LS6CLK	112	21	LS6CLK	3	21	P-IO18	162
22	MB1-DB0	111	22	MA1-DB0	206	22	MA2-DB0	161
19	P1-D57	114	19	D57	5	19	P-IO16	164
20	P-CSA	113	20	CSA	4	20	P-IO17	163
17	P-DB6	119	17	DB6	7	17	P-IO14	166
18	P-OE	115	18	OE	6	18	P-IO15	165
15	P-DB5	121	15	DB5	9	15	P-IO12	168
16	P-WE	120	16	WE	8	16	P-IO13	167
13	P-DB4	123	13	DB4	14	13	P-IO10	173
14	P-ADR5	122	14	ADR5	10	14	P-IO11	172
11	P-DB3	126	11	DB3	16	11	P-IO8	175
12	P-ADR4	125	12	ADR4	15	12	P-IO9	174
9	P-DB2	129	9	DB2	18	9	P-IO6	178
10	P-ADR3	127	10	ADR3	17	10	P-IO7	176
7	P-DB1	133	7	DB1	21	7	P-IO4	180
8	P-ADR2	132	8	ADR2	20	8	P-IO5	179
5	P-DB0	133	5	DB0	23	5	P-IO2	187
6	P-ADR1	134	6	ADR1	22	6	P-IO3	181
3	VCCC	VCCO	3	VCCC	VCCO	3	VCCC	VCCO
4	P-ADR0	136	4	ADR0	24	4	P-IO1	188
1	GND	GND	1	GND	GND	1	GND	GND
2	VU	VU	2	VU	VU	2	VU	VU

BAB III

PERANCANGAN SISTEM

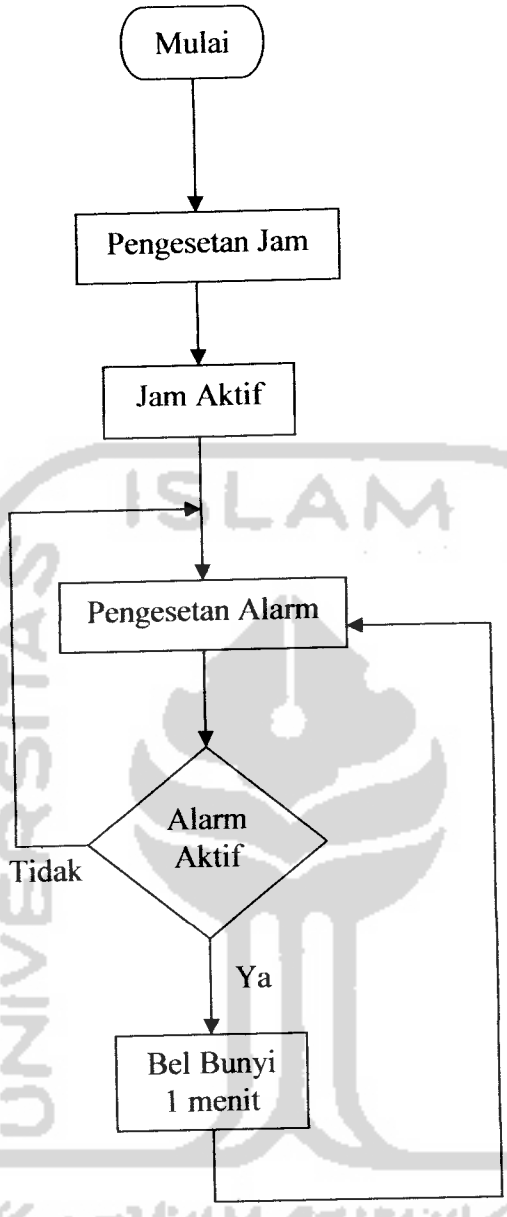
3.1. Skema Alat

Perancangan jam digital yang menggunakan FPGA dapat dilihat pada blok diagram sebagaimana pada Gambar 3.1.

Gambar 3.1. Blok diagram jam digital

3.2. Diagram Alir Jam Digital

Dari digram alir dari Gambar 3.2, hal pertama yang dilakukan adalah pengesetan jam. Pengesetan jam di sini meliputi pengesetan waktu (jam dan menit) serta pengaktifan saklar ON/OFF jam, setelah pengesetan jam maka jam akan aktif. Setelah jam aktif, kemudian dilakukan pengesetan alarm sesuai pada jam berapa alarm akan aktif. Setelah pengesetan alarm, apabila waktu pada jam sama dengan waktu pada alarm dan saklar alarm pada kondisi ON maka alarm akan aktif selama 1 menit, bunyi atau tidaknya bel pada alarm tergantung dari saklar ON/OFF alarm. Aktif atau tidaknya alarm, jam akan tetap aktif. Sehingga pada diagram alir jam digital ini tidak ada pengakhiran (END).

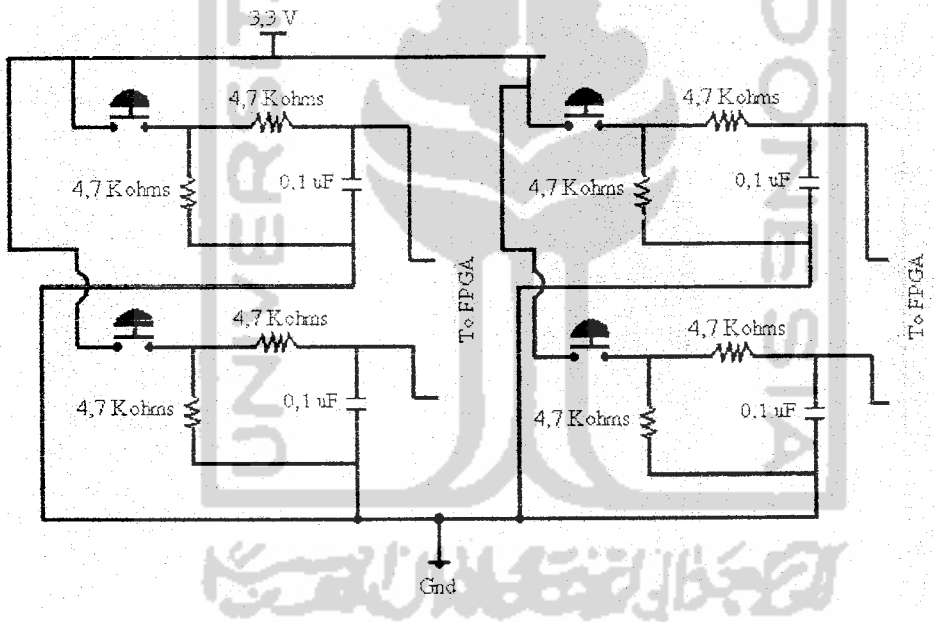


Gambar 3.2. Diagram alir jam digital

3.3. Perancangan Perangkat Keras

3.3.1. Rangkaian Saklar Tombol *Push Button*

Rangkaian saklar tombol *push button* pada tugas akhir ini disusun sedemikian rupa sehingga membentuk baris yang masing-masing dihubungkan dengan *port input*. Rangkaian saklar tombol *push button* berfungsi sebagai masukan. Pada perancangan ini unit rangkaian saklar tombol *push button* terdiri dari 4 buah tombol yang digunakan untuk masukan nilai yang akan menjadi *setpoint*, dan sebagai tombol dalam pengesetan jam dan pengesetan alarm.



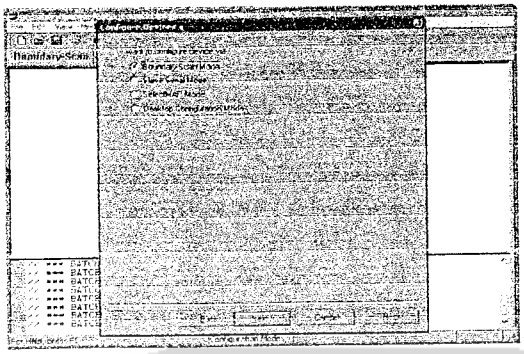
Gambar 3.3. Rangkaian saklar tombol *push button*

3.3.2. FPGA (*Field programmable Gate Array*)

Pada perancangan ini, FPGA digunakan sebagai perangkat utama untuk memproses program yang sebelumnya sudah dibuat dalam perangkat lunak VHDL dan Xilinx ISE 7. FPGA yang digunakan pada perancangan ini menggunakan rangkaian pegasus dengan IC XC2S50. Pada rangkaian Pegasus terdapat 140 pin I/O yang dapat diprogram dan berfingsi sebagai penghubung dengan piranti lainnya. Untuk mengaktifkan pin-pin mana yang akan dipakai, maka harus dilakukan pengesetan pin terlebih dahulu di dalam *assign package pins.*

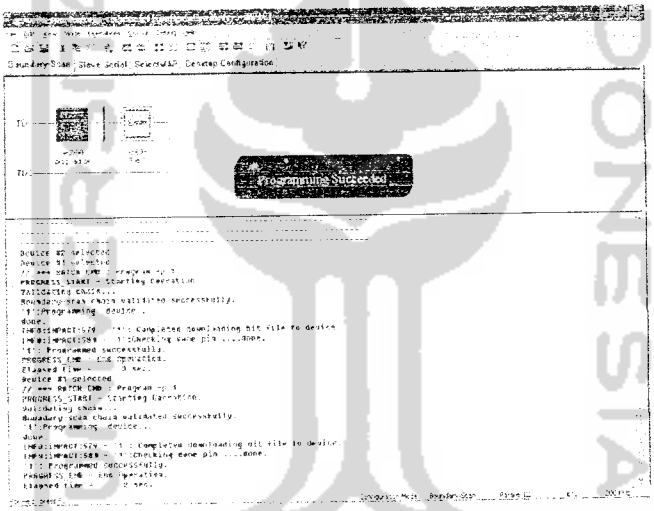
Gambar 3.4. Tampilan *Assign package pins* yang digunakan

Kemudian menjalankan perintah *configure device* untuk men*download* program yang telah dibuat ke dalam FPGA. Akan muncul tampilan seperti pada Gambar 3.5, kemudian dipilih *boundary scan mode* untuk men*download* rancangan melalui *parallel port.*



Gambar 3.5. Tampilan Mode untuk Men*download* Program

Adapun tampilan pada waktu proses men*download* sukses adalah seperti pada Gambar 3.6.

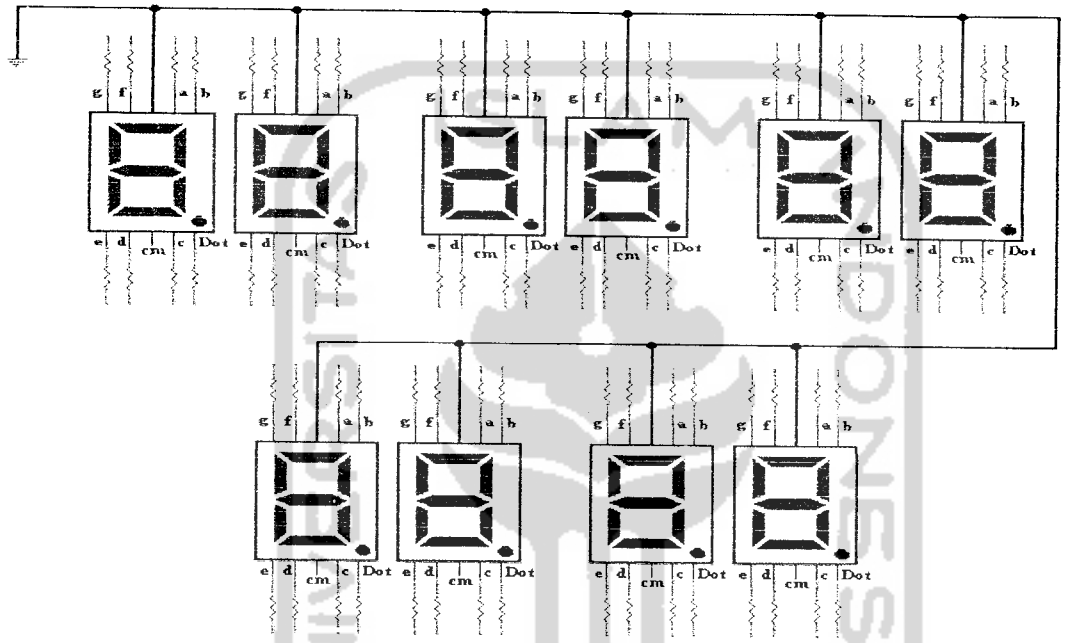


Gambar 3.6. Tampilan Proses *Download* Sukses

3.3.3. Rangkaian *Seven Segment*

Rangkaian *seven segment* pada tugas akhir ini merupakan *output* dari FPGA. Rangkaian *seven segment* yang digunakan berisi 10 buah penampil yang terdiri dari: 6 buah penampil yang berfungsi sebagai penampil jam dan 4 buah penampil yang berfungsi sebagai penampil alarm. Hambatan (R) yang digunakan sebesar 100ohm

yang masing-masing *output* dihubungkan pada FPGA dan sebagai penghubung dengan FPGA menggunakan kabel data yang biasanya dipakai pada komputer. Adapun gambar rangkaian *seven segment* dapat dilihat pada Gambar 3.7.



Gambar 3.7. Rangkaian *seven segment*

3.3.4. Rangkaian Alarm

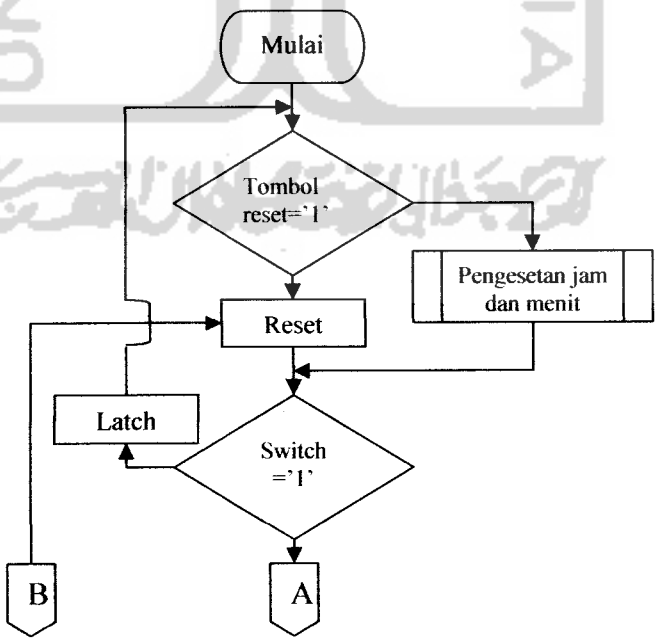
Rangkaian alarm pada tugas akhir ini merupakan *output* dari FPGA. Pada perancangan ini rangkaian berfungsi sebagai penanda atau pengingat waktu yang diinginkan. Prinsip kerja dari rangkaian alarm ini adalah apabila tampilan jam sama dengan tampilan alarm, maka rangkaian alarm akan aktif sehingga speaker akan berbunyi. Adapun gambar rangkaian alarm dapat dilihat pada Gambar 3.8.

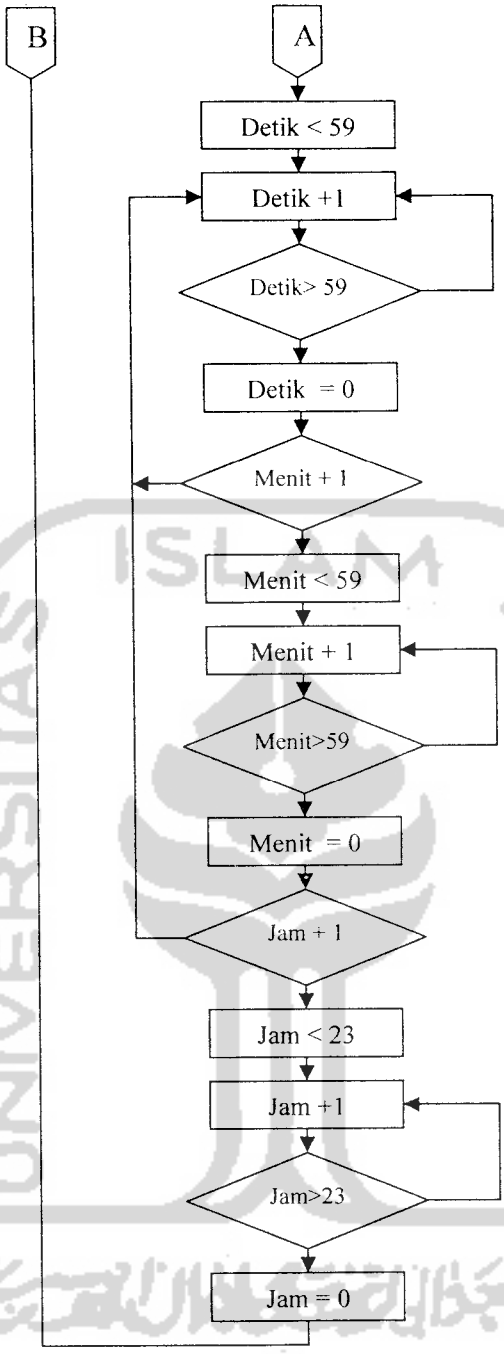
Contoh 1: program untuk jam digital, pengesetan jam, dan pengesetan menit

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity coba_counter is
port ( clk_out:in std_logic;
        reset,reset_alrm:in std_logic;
        switch,switch_alrm:in std_logic;
        set_jam,set_jam_alrm:in std_logic;
        set_menit,set_menit_alrm:in std_logic;
        bel:out std_logic;
        M_coba,J_coba:out bit_vector (0 to 13);
        D,M,J:out bit_vector (0 to 13));

end coba_counter;

architecture Behavioral of coba_counter is
 signal detik,menit,jam:integer range 0 to 59;
signal menit_coba,jam_coba:integer range 0 to 59;
begin

u1:process (reset,clk_out,set_menit,set_jam)
begin
if reset='1'then
detik <= 0;
menit <= 0;
jam <= 0;
        elsif clk_out='1'and clk_out'event then
        if switch ='1' then
                if detik < 59 then
                detik <= detik + 1;
                else
                detik <=0;
                end if;
        end if;

        if detik = 59 then
                if menit < 59 then
                menit <= menit + 1;
                 else
                menit <= 0;
```

```
                end if;

                if set_jam_alrm ='1'then
                        if jam_coba < 23 then
                        jam_coba <= jam_coba + 1;
                        else
                        jam_coba <= 0;
                        end if;
                end if;
                if switch_alrm ='1' then
                        if menit_coba <= menit and  jam_coba <= jam and menit <=
                        menit_coba and jam <= jam_coba then
                                bel <='1';
                        else
                                bel <='0';
                        end if;
                end if;
                if switch_alrm ='0' then
                        bel <='0';
                end if;
        end if;
end process;
```

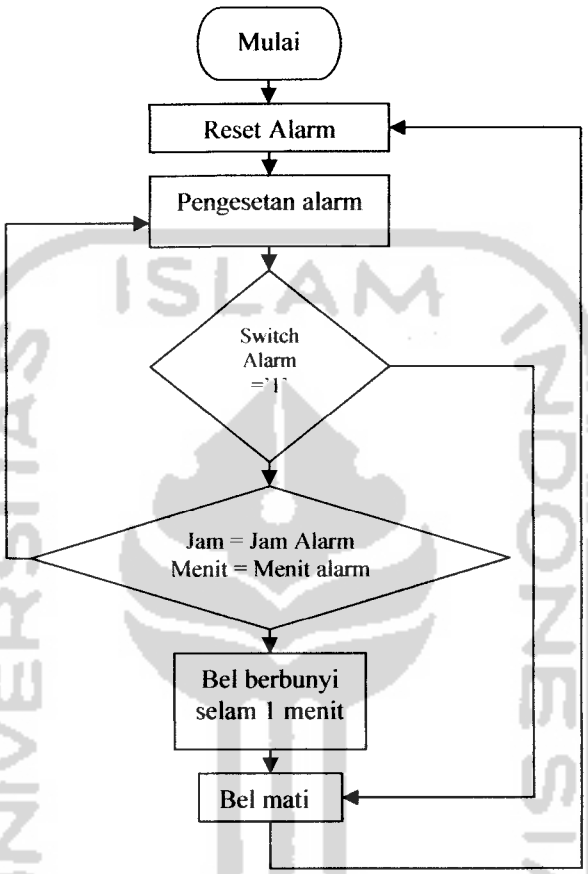
Dari contoh 1 dan contoh 2 dapat dibuat diagram alirnya yang dapat dilihat pada Gambar 3.9, Gambar 3.10.a, Gambar 3.10.b, dan Gambar 3.11.





Gambar 3.9. Diagram alir jam

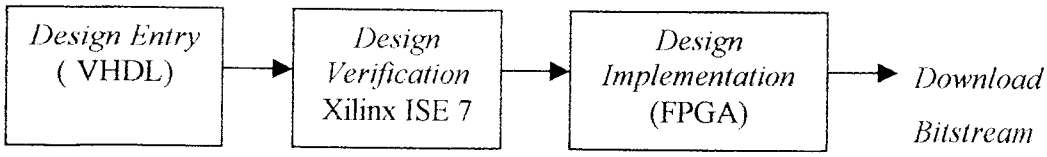
untuk alarm, switch alarm harus diaktifkan. Diagram alir untuk alarm ditunjukkan pada Gambar 3.11.



Gambar 3.11. Diagram alir *alarm*

3.4.2. Xilinx

Proses perancangan menggunakan Xilinx dapat dilihat pada Gambar 3.9 dimana proses perancangan dapat dibagi menjadi 3 bagian yaitu pertama perancangan rangkaian, kemudian verifikasi hasil rancangan dan proses yang ketiga implementasi rancangan.



Gambar 3.12. Proses perancangan menggunakan Xilinx

Implementasi rancangan pada Xilinx dengan meng-*compile file* rancangan menjadi *file* konfigurasi yang telah dioptimalisasi dari penggunaan gerbang logika dan interkoneksi (*wiring*) antar komponen. *Download bitstream* dapat dilakukan dengan mudah dari PC ke FPGA menggunakan *demoboard* Xilinx XC2S50. Kedua peralatan tersebut juga dapat diprogram dalam sistem dengan menghubungkan JTAG ke peralatan pin yang telah di program.

BAB IV

HASIL PENGAMATAN DAN ANALISA

4.1. Pengujian Perangkat Keras (*Hardware*)

4.1.1. Analisis Rangkaian Saklar Tombol *Push Button*

Pada perancangan jam digital ini, digunakan 4 saklar tombol *push button*. Dimana masing-masing saklar atau tombol memiliki *output* sendiri-sendiri. Dibawah ini adalah hasil pengujian saklar tombol *push button* :

Tabel 4.1. Hasil pengujian saklar tombol *push button*

NO	Data Tombol	Masukan ke FPGA	Keterangan
1	Set_jam	1	3,3 volt
2	Set_menit	1	3,3 volt
3	Set_jam_alrm	1	3,3 volt
4	Set_menit_alrm	1	3,3 volt

4.1.2. Analisis Rangkaian *Seven Segment*

Pada perancangan ini Rangkaian *seven segment* berfungsi untuk menampilkan waktu dan berfungsi untuk menampilkan pengesetan alarm.

Tabel 4.2. Hasil Pengujian Rangkaian *Seven segment*

No.	Keluaran Program	Tampilan Seven Segment
1.	"0111111 0111111"	00
2.	"0111111 0000110"	01
3.	"0111111 1011011"	02
4.	"0111111 1001111"	03
5.	"0111111 1100110"	04
6.	"0111111 1101101"	05
7.	"0111111 1111101"	06
8.	"0111111 0000111"	07
9.	"0111111 1111111"	08
10.	"0111111 1101111"	09
11.	"0000110 0111111"	10
12.	"0000110 0000110"	11
13.	"0000110 1011011"	12
14.	"0000110 1001111"	13
15.	"0000110 1100110"	14
16.	"0000110 1101101"	15
17.	"0000110 1111101"	16
18.	"0000110 0000111"	17
19.	"0000110 1111111"	18
20.	"0000110 1101111"	19
21.	"1011011 0111111"	20
22.	"1011011 0000110"	21
23.	"1011011 1011011"	22
24.	"1011011 1001111"	23
25.	"1011011 1100110"	24
26.	"1011011 1101101"	25
27.	"1011011 1111101"	26
28.	"1011011 0000111"	27
29.	"1011011 1111111"	28
30.	"1011011 1101111"	29
31.	"1001111 0111111"	30
32.	"1001111 0000110"	31
33.	"1001111 1011011"	32
34.	"1001111 1001111"	33
35.	"1001111 1100110"	34
36.	"1001111 1101101"	35

37.	"1001111 1111101"	36
38.	"1001111 0000111"	37
39.	"1001111 1111111"	38
40.	"1001111 1101111"	39
41.	"1100110 0111111"	40
42.	"1100110 0000110"	41
43.	"1100110 1011011"	42
44.	"1100110 1001111"	43
45.	"1100110 1100110"	44
46.	"1100110 1101101"	45
47.	"1100110 1111101"	46
48.	"1100110 0000111"	47
49.	"1100110 1111111"	48
50.	"1100110 1101111"	49
51.	"1101101 0111111"	50
52.	"1101101 0000110"	51
53.	"1101101 1011011"	52
54.	"1101101 1001111"	53
55.	"1101101 1100110"	54
56.	"1101101 1101101"	55
57.	"1101101 1111101"	56
58.	"1101101 0000111"	57
59.	"1101101 1111111"	58
60.	"1101101 1101111"	59

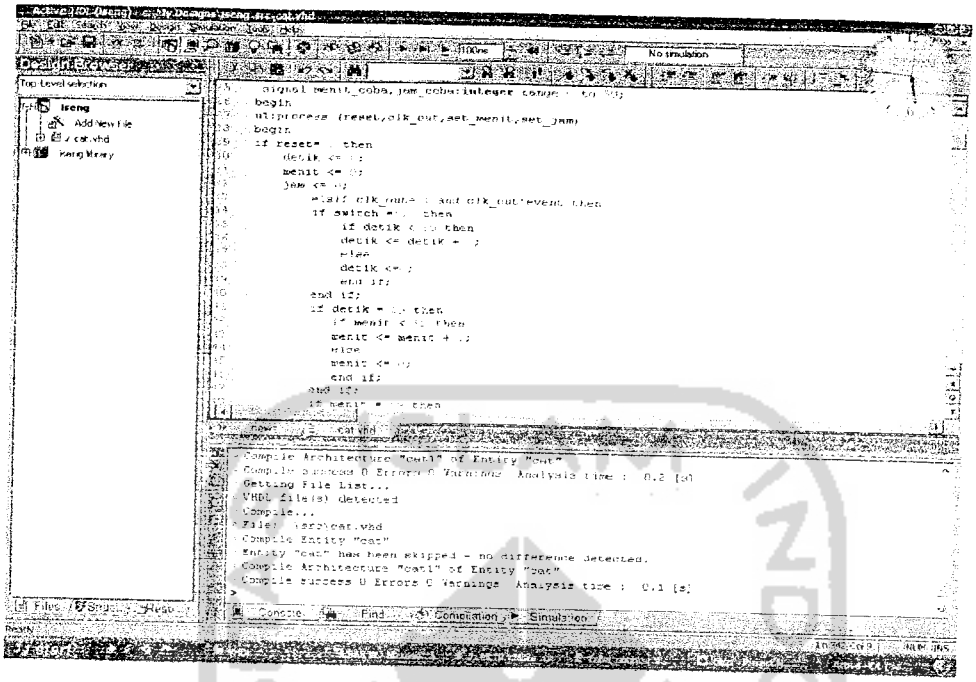
4.2. Analisis Perangkat Lunak (*Software*)

4.2.1 Analisis VHDL

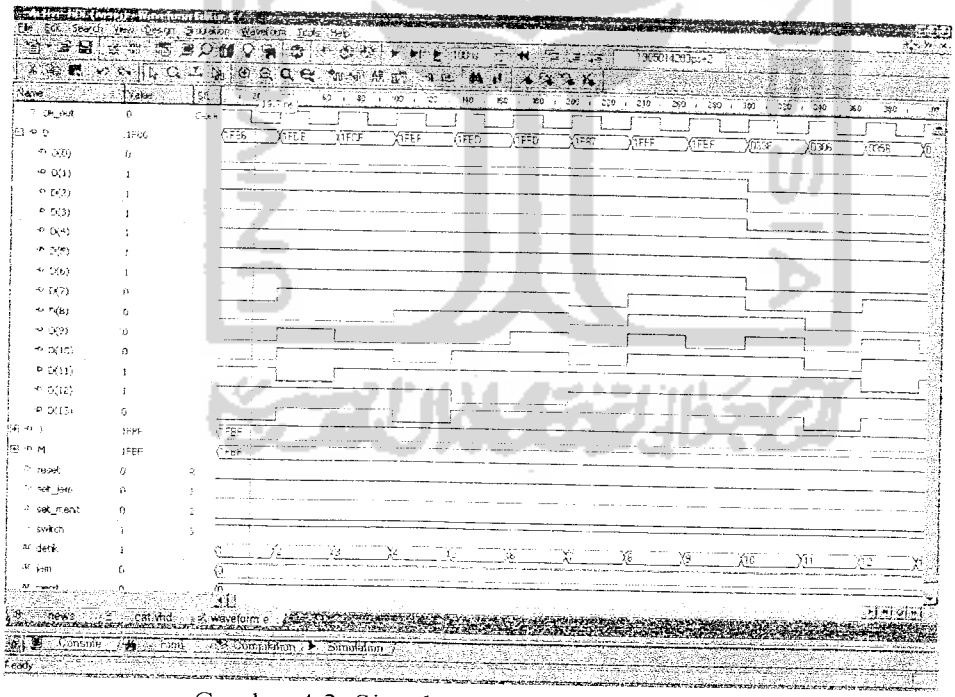
Pada perancangan perangakat lunak ini program yang dibuat tidak langsung menggunakan program Xilinx tetapi program terlebih dahulu dibuat menggunakan program VHDL, hal ini dilakukan karena pada program VHDL proses meng-*comfile* nya sangat cepat sehingga dapat diketahui ada tidaknya kesalahan penulisan pada pembuatan program secara cepat. Apabila program langsung dibuat pada program Xilinx, maka pada proses *Syntesize-XST* untuk mengetahui ada tidaknya kesalahan

penulisan dalam pembuatan program sangat lama dan terkadang proses *Syntesize-XST* tidak dapat bekerja sebagaimana mestinya, sehingga harus mengulang-ulang dalam proses *Syntesize-XST* untuk mengetahui ada tidaknya kesalahan dalam membuat program.

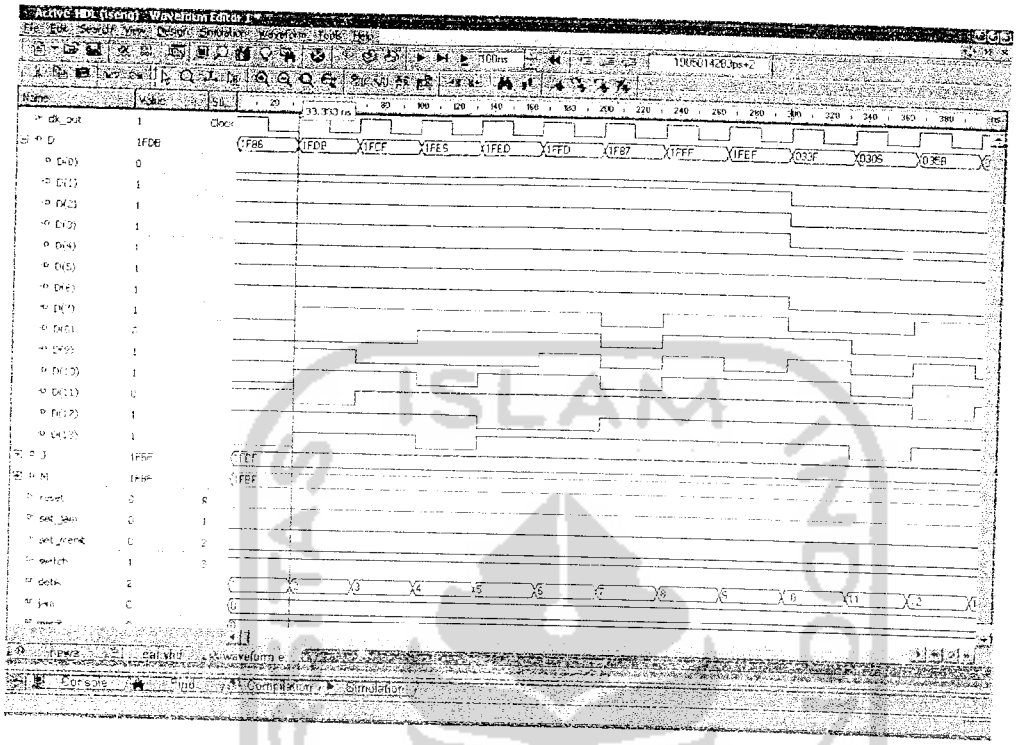
Pada perangkat lunak VHDL dibuat program jam dan alarm yang menggunakan keluaran *seven segment*. Pada perancangan ini sinyal keluaran dari program juga ditampilkan yang dimaksudkan untuk mengetahui proses kinerja dari program-program tersebut. Tampilan program yang menggunakan keluaran *seven segment* yang sukses atau tidak ada *error* setelah melalui proses *comfile* adalah seperti ditunjukkan pada Gambar 4.1. Pada program VHDL juga dapat mengetahui bentuk sinyal keluaran dari program *seven segment*. Pada perancangan ini clock yang digunakan untuk menampilkan bentuk sinyal keluaran sebesar 40MHz, disaat clock = 1 maka penampil detik akan bertambah satu kemudian disaat detik menunjukkan angka 00 maka penampil menit akan bertambah satu, disaat menit menunjukkan angka 00 maka penampil jam akan bertambah satu. Hal ini terjadi terus berulang-ulang, program yang dibuat berupa *counter*.



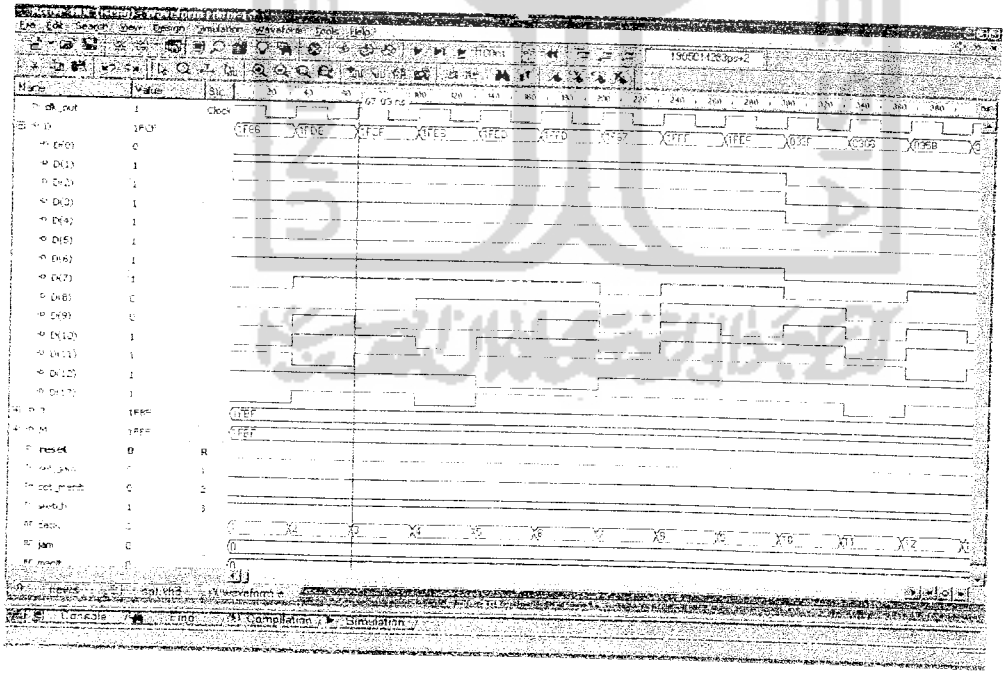
Gambar 4.1. Tampilan program untuk keluaran *seven segment*



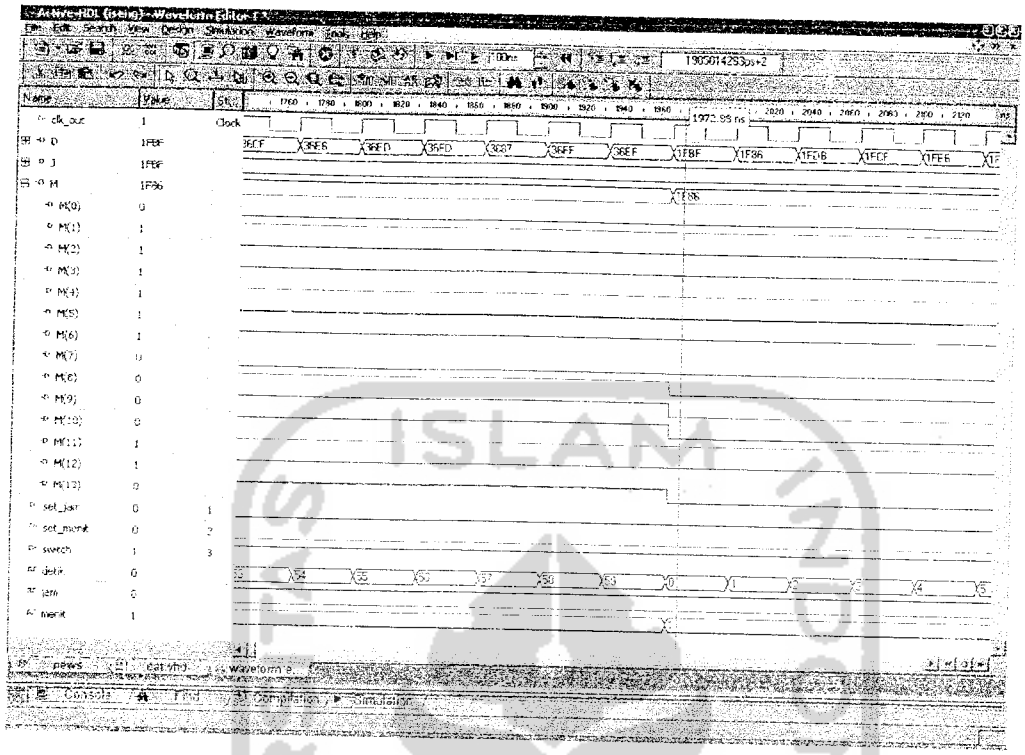
Gambar 4.2. Sinyal untuk keluaran detik ke-1



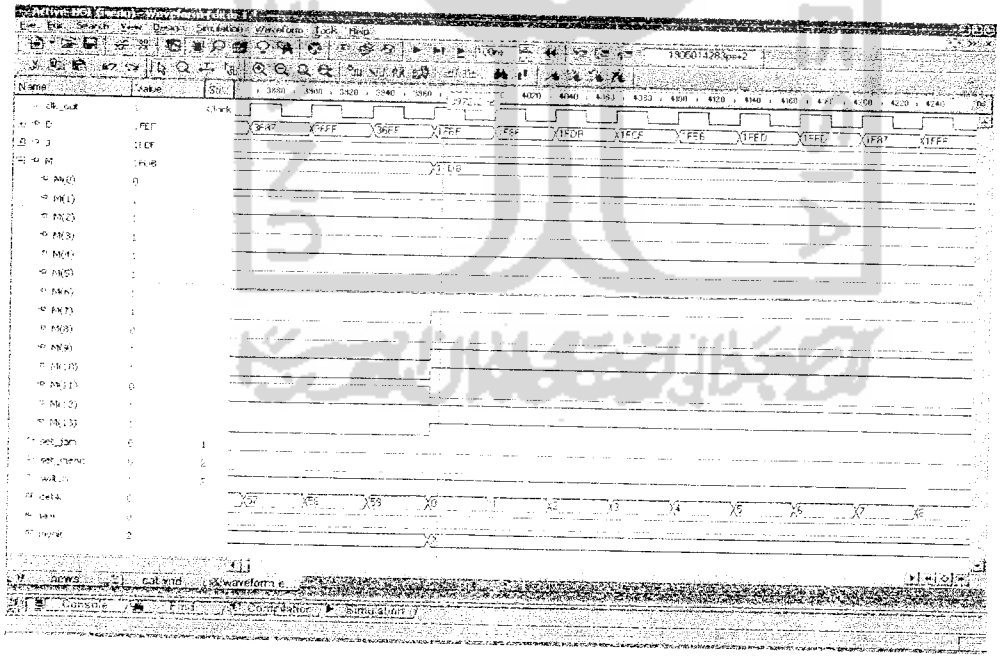
Gambar 4.3. Sinyal untuk keluaran detik ke-2



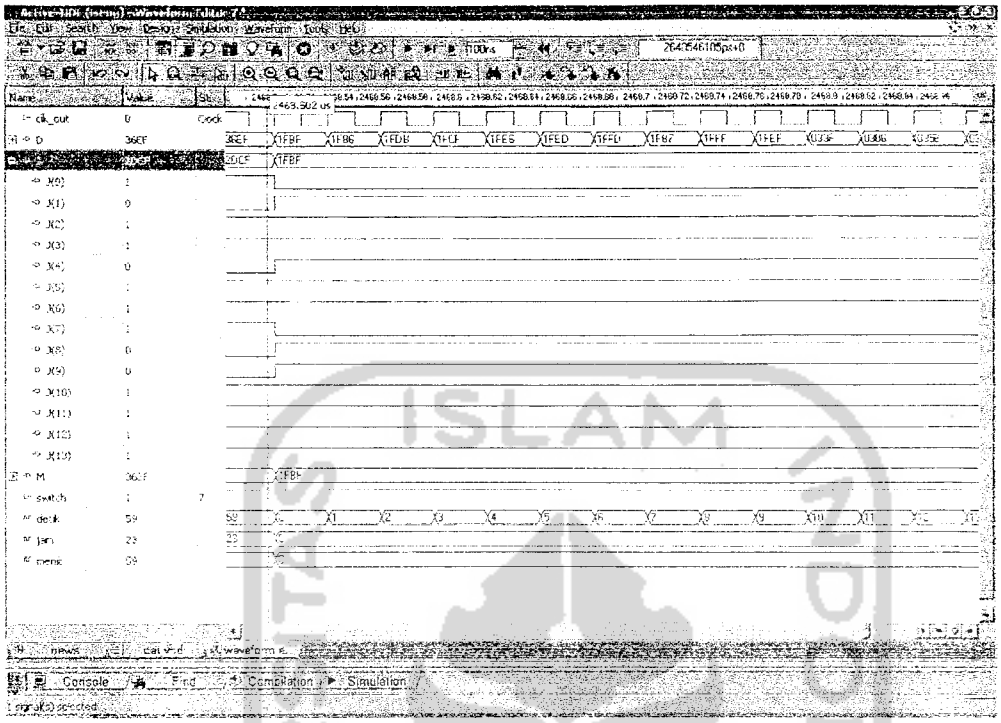
Gambar 4.4. Sinyal untuk keluaran detik ke-3



Gambar 4.5. Sinyal keluaran untuk menit ke-1



Gambar 4.6. Sinyal keluaran untuk menit ke-2



Gambar 4.7. Sinyal keluaran untuk jam ke-0

Gambar 4.8. Sinyal keluaran untuk jam ke-0 pada saat detik ke-1

DAFTAR PUSTAKA

Ibrahim K F, Ir.P.Insap Santosa, M.Sc., *TEKNIK DIGITAL*, Andi Yogyakarta, 1996

.

Owen B., *Dasar-dasar Elektronika*, Erlangga 2004.

Pellerin D, Douglas Taylor., *VHDL Made Easy*, Prentice Hall, PTR, 1997.

Sjoholmand Lindh S., *VHDL FOR DESIGNERS*, Prentice Hall, Europe, 1997.

Van den Bout D., *The Practical Xilinx Designers Lab Book* 1.5, Prentice Hall, Upper Saddle River, 1999.

```
1   ----------------------------------------------------------------------------------
2   -- Company:
3   -- Engineer:
4   --
5   -- Create Date:
6   -- Design Name:
7   -- Module Name:
8   -- Project Name:
9   -- Target Devices:
10  -- Tool versions:
11  -- Description:
12  --
13  -- Dependencies:
14  --
15  -- Revision:
16  -- Revision 0.01 - File Created
17  -- Additional Comments:
18  --
19  ----------------------------------------------------------------------------------
20  library IEEE;
21  use IEEE.STD_LOGIC_1164.ALL;
22  use IEEE.STD_LOGIC_ARITH.ALL;
23  use IEEE.STD_LOGIC_UNSIGNED.ALL;
24
25  ---- Uncomment the following library declaration if using
26  ---- arithmetic functions with Signed or Unsigned values
27  --library UNISIM;
28  --use UNISIM.VComponents.all;
29
30  entity prog_jadi is
31  port (
32        clk,reset,reset_alrm: in          ;
33        switch,switch_alrm :in            ;
34        set_menit,set_menit_alrm:in       ;
35        set_jam,set_jam_alrm:in           ;
36        D,M,J:out bit_vector (0 to 13);
37        M_coba,J_coba:out bit_vector (0 to 13);
38        bel:out                 );
39  end prog_jadi;
40
41  architecture Behavioral of prog_jadi is
42   component clk_div is
43     port (clk,reset: in                ;
44           clk_out: out                 );
45   end component;
46
47   component coba_counter is
48     port ( clk_out:in std_logic;
49           reset,reset_alrm:in std_logic;
50          switch,switch_alrm:in std_logic;
51          set_jam,set_jam_alrm:in std_logic;
52          set_menit,set_menit_alrm:in std_logic;
53          bel:out std_logic;
54          M_coba,J_coba:out bit_vector (0 to 13);
55          D,M,J:out bit_vector (0 to 13));
56   end component;
57
58  signal clk_out:std_logic;
59
60
```

```
61    begin
62      U1:clk_div port map(clk,reset,clk_out);
63      U2:coba_counter port map(clk_out,reset,reset_alrm,switch,switch_alrm,set_jam,set_j
      am_alrm,set_menit,set_menit_alrm,bel,M_coba,J_coba,D,M,J);
64
65
66    end Behavioral;
67
```

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date:    [illegible]
-- Design Name:
-- Module Name:    [illegible]
-- [illegible]:
-- [illegible]:
-- [illegible]:
-- [illegible]:
--
-- [illegible]:
--
-- [illegible]:
-- [illegible]
-- [illegible]:
--
----------------------------------------------------------------------------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

---- [illegible]
---- [illegible]
--[illegible];
--use [illegible]

entity clk_div is
port (
      clk,reset: in [illegible];
      clk_out: out [illegible]);
end clk_div;

architecture Behavioral of clk_div is
signal clk_int: [illegible];
constant data: [illegible](24 downto 0):="1011111010111100001000000"; -- [illegible]
[illegible]--
signal sig1: [illegible];
signal sig3: std_logic_vector(24 downto 0);

begin
  u1:process(clk,sig3,reset)
begin
   if reset='1' then
      sig3<=data;
   elsif clk='1' and clk'event then
         if sig3="0000000000000000000000000" then
         sig3<=data;            --[illegible]--
      else
         sig3<=(sig3 - 1);
      end if;
    end if;
end process;

u2:process(sig3,clk,reset)
begin
   if reset='1' then
      sig1<='1';
```

```
60       elsif clk='1' and clk'event then
61          if sig3="00000000000000000000000000" then
62             sig1<='1';
63          else
64             sig1<='0';
65          end if;
66       end if;
67    end process;
68
69    u3:process(sig1,clk,clk_int,reset)
70    begin
71       if reset='1' then
72          clk_int<='0';
73       elsif            (sig1) then
74          clk_int<=not(clk_int);
75       else
76          clk_int<=clk_int;
77       end if;
78          clk_out<=clk_int;
79
80    end process;
81
82    end Behavioral;
83
```

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20  library IEEE;
21  use IEEE.STD_LOGIC_1164.ALL;
22  use IEEE.STD_LOGIC_ARITH.ALL;
23  use IEEE.STD_LOGIC_UNSIGNED.ALL;
24
25
26
27
28
29
30  entity coba_counter is
31  port ( clk_out:in          ;
32         reset,reset_alrm:in          ;
33         switch,switch_alrm:in          ;
34         set_jam,set_jam_alrm:in          ;
35         set_menit,set_menit_alrm:in          ;
36         bel:out          ;
37         M_coba,J_coba:out bit_vector (0 to 13);
38         D,M,J:out bit_vector (0 to 13));
39
40  end coba_counter;
41
42  architecture Behavioral of coba_counter is
43   signal detik,menit,jam:integer range 0 to 59;
44   signal menit_coba,jam_coba:integer range 0 to 59;
45  begin
46
47  u1:process (reset,clk_out,set_menit,set_jam)
48  begin
49  if reset='1'then
50     detik <= 0;
51     menit <= 0;
52     jam <= 0;
53        elsif clk_out='1'and clk_out'event then
54        if switch ='1' then
55           if detik < 59 then
56           detik <= detik + 1;
57           else
58           detik <=0;
59           end if;
60        end if;
```

```
61          if detik = 59 then
62             if menit < 59 then
63             menit <= menit + 1;
64             else
65             menit <= 0;
66             end if;
67          end if;
68          if menit = 59 then
69             if jam < 23 then
70             jam <=jam +1;
71             else
72             jam <= 0;
73             end if;
74          end if;
75         if set_menit ='1'then
76             if menit < 59 then
77             menit <= menit + 1;
78             else
79             menit <= 0;
80             end if;
81          end if;
82         if set_jam ='1'then
83             if jam < 23 then
84             jam <= jam + 1;
85             else
86             jam <= 0;
87             end if;
88          end if;
89
90       end if;
91    end process;
92
93    u2:process (clk_out,reset_alrm,set_jam_alrm,set_menit_alrm,switch_alrm,switch_alrm)
94    begin
95
96       if reset_alrm ='1'then
97       menit_coba <= 0;
98       jam_coba <= 0;
99          elsif clk_out='1'and clk_out'event then
100         if set_menit_alrm ='1'then
101            if menit_coba < 59 then
102            menit_coba <= menit_coba + 1;
103            else
104            menit_coba <= 0;
105            end if;
106         end if;
107         if set_jam_alrm ='1'then
108            if jam_coba < 23 then
109            jam_coba <= jam_coba + 1;
110            else
111            jam_coba <= 0;
112            end if;
113         end if;
114
115         if switch_alrm ='1' then
116            if menit_coba <= menit and jam_coba <= jam and menit <= menit_coba and jam
       <= jam_coba then
117               bel <='1';
118            else
119               bel <='0';
```

```
120             end if;
121          end if;
122          if switch_alrm ='0' then
123             bel <='0';
124          end if;
125       end if;
126    end process;
127
128    D <= "0111111O111111" when detik = 0  else
129             "01111110000110" when detik= 1  else
130             "01111111011011" when detik= 2  else
131             "01111111001111" when detik= 3  else
132             "01111111100110" when detik= 4  else
133             "01111111101101" when detik= 5  else
134             "01111111111101" when detik= 6  else
135             "01111110000111" when detik= 7  else
136             "01111111111111" when detik= 8  else
137             "01111111101111" when detik= 9  else
138             "00001100111111" when detik= 10 else
139             "00001100000110" when detik= 11 else
140             "00001101011011" when detik= 12 else
141             "00001101001111" when detik= 13 else
142             "00001101100110" when detik= 14 else
143             "00001101101101" when detik= 15 else
144             "00001101111101" when detik= 16 else
145             "00001100000111" when detik= 17 else
146             "00001101111111" when detik= 18 else
147             "00001101101111" when detik= 19 else
148             "10110110111111" when detik= 20 else
149             "10110110000110" when detik= 21 else
150             "10110111011011" when detik= 22 else
151             "10110111001111" when detik= 23 else
152             "10110111100110" when detik= 24 else
153             "10110111101101" when detik= 25 else
154             "10110111111101" when detik= 26 else
155             "10110110000111" when detik= 27 else
156             "10110111111111" when detik= 28 else
157             "10110111101111" when detik= 29 else
158             "10011110111111" when detik= 30 else
159             "10011110000110" when detik= 31 else
160             "10011111011011" when detik= 32 else
161             "10011111001111" when detik= 33 else
162             "10011111100110" when detik= 34 else
163             "10011111101101" when detik= 35 else
164             "10011111111101" when detik= 36 else
165             "10011110000111" when detik= 37 else
166             "10011111111111" when detik= 38 else
167             "10011111101111" when detik= 39 else
168             "11001100111111" when detik= 40 else
169             "11001100000110" when detik= 41 else
170             "11001101011011" when detik= 42 else
171             "11001101001111" when detik= 43 else
172             "11001101100110" when detik= 44 else
173             "11001101101101" when detik= 45 else
174             "11001101111101" when detik= 46 else
175             "11001100000111" when detik= 47 else
176             "11001101111111" when detik= 48 else
177             "11001101101111" when detik= 49 else
178             "11011010111111" when detik= 50 else
179             "11011010000110" when detik= 51 else
```

```
180             "110110110110111" when detik= 52 else
181             "110110110011111" when detik= 53 else
182             "110110111001110" when detik= 54 else
183             "110110111011101" when detik= 55 else
184             "110110111111101" when detik= 56 else
185             "110110100001111" when detik= 57 else
186             "110110111111111" when detik= 58 else
187             "110110111011111" when detik= 59 ;
188
189     M <= "011111110111111" when menit = 0  else
190             "011111110000110" when menit= 1  else
191             "011111110110111" when menit= 2  else
192             "011111110011111" when menit= 3  else
193             "011111111001110" when menit= 4  else
194             "011111111011101" when menit= 5  else
195             "011111111111101" when menit= 6  else
196             "011111100001111" when menit= 7  else
197             "011111111111111" when menit= 8  else
198             "011111111011111" when menit= 9  else
199             "000011001111111" when menit= 10 else
200             "000011000001110" when menit= 11 else
201             "000011010110111" when menit= 12 else
202             "000011010011111" when menit= 13 else
203             "000011011001110" when menit= 14 else
204             "000011011011101" when menit= 15 else
205             "000011011111101" when menit= 16 else
206             "000011000001111" when menit= 17 else
207             "000011011111111" when menit= 18 else
208             "000011011011111" when menit= 19 else
209             "101101101111111" when menit= 20 else
210             "101101100001110" when menit= 21 else
211             "101101110110111" when menit= 22 else
212             "101101110011111" when menit= 23 else
213             "101101111001110" when menit= 24 else
214             "101101111011101" when menit= 25 else
215             "101101111111101" when menit= 26 else
216             "101101100001111" when menit= 27 else
217             "101101111111111" when menit= 28 else
218             "101101111011111" when menit= 29 else
219             "100111101111111" when menit= 30 else
220             "100111100001110" when menit= 31 else
221             "100111110110111" when menit= 32 else
222             "100111110011111" when menit= 33 else
223             "100111111001110" when menit= 34 else
224             "100111111011101" when menit= 35 else
225             "100111111111101" when menit= 36 else
226             "100111100001111" when menit= 37 else
227             "100111111111111" when menit= 38 else
228             "100111111011111" when menit= 39 else
229             "110011001111111" when menit= 40 else
230             "110011000001110" when menit= 41 else
231             "110011010110111" when menit= 42 else
232             "110011010011111" when menit= 43 else
233             "110011011001110" when menit= 44 else
234             "110011011011101" when menit= 45 else
235             "110011011111101" when menit= 46 else
236             "110011000001111" when menit= 47 else
237             "110011011111111" when menit= 48 else
238             "110011011011111" when menit= 49 else
239             "110110101111111" when menit= 50 else
```

```
240             "110110100000110" when menit= 51 else
241             "110110110011011" when menit= 52 else
242             "110110110011111" when menit= 53 else
243             "110110111100110" when menit= 54 else
244             "110110111101101" when menit= 55 else
245             "110110111111101" when menit= 56 else
246             "110110100000111" when menit= 57 else
247             "110110111111111" when menit= 58 else
248             "110110111101111" when menit= 59 ;
249
250     J <= "011111110111111" when jam = 0  else
251             "011111110000110" when jam= 1  else
252             "011111111011011" when jam= 2  else
253             "011111111001111" when jam= 3  else
254             "011111111100110" when jam= 4  else
255             "011111111101101" when jam= 5  else
256             "011111111111101" when jam= 6  else
257             "011111110000111" when jam= 7  else
258             "011111111111111" when jam= 8  else
259             "011111111101111" when jam= 9  else
260             "000011001111111" when jam= 10 else
261             "000011000000110" when jam= 11 else
262             "000011010011011" when jam= 12 else
263             "000011010011111" when jam= 13 else
264             "000011011100110" when jam= 14 else
265             "000011011101101" when jam= 15 else
266             "000011011111101" when jam= 16 else
267             "000011000000111" when jam= 17 else
268             "000011011111111" when jam= 18 else
269             "000011011101111" when jam= 19 else
270             "101101101111111" when jam= 20 else
271             "101101100000110" when jam= 21 else
272             "101101110110111" when jam= 22 else
273             "101101110011111" when jam= 23 ;
274
275     M_coba <= "011111110111111" when menit_coba = 0  else
276             "011111100000110" when menit_coba= 1  else
277             "011111110110111" when menit_coba= 2  else
278             "011111110011111" when menit_coba= 3  else
279             "011111111001100" when menit_coba= 4  else
280             "011111111011011" when menit_coba= 5  else
281             "011111111111101" when menit_coba= 6  else
282             "011111100000111" when menit_coba= 7  else
283             "011111111111111" when menit_coba= 8  else
284             "011111111011111" when menit_coba= 9  else
285             "000011001111111" when menit_coba= 10 else
286             "000011000000110" when menit_coba= 11 else
287             "000011010011011" when menit_coba= 12 else
288             "000011010011111" when menit_coba= 13 else
289             "000011011100110" when menit_coba= 14 else
290             "000011011101101" when menit_coba= 15 else
291             "000011011111101" when menit_coba= 16 else
292             "000011000000111" when menit_coba= 17 else
293             "000011011111111" when menit_coba= 18 else
294             "000011011101111" when menit_coba= 19 else
295             "101101101111111" when menit_coba= 20 else
296             "101101100000110" when menit_coba= 21 else
297             "101101110110111" when menit_coba= 22 else
298             "101101110011111" when menit_coba= 23 else
299             "101101111100110" when menit_coba= 24 else
```

```
300             "10110111101101" when menit_coba= 25 else
301             "10110111111101" when menit_coba= 26 else
302             "10110110000111" when menit_coba= 27 else
303             "10110111111111" when menit_coba= 28 else
304             "10110111101111" when menit_coba= 29 else
305             "10011110111111" when menit_coba= 30 else
306             "10011110000110" when menit_coba= 31 else
307             "10011111011011" when menit_coba= 32 else
308             "10011111001111" when menit_coba= 33 else
309             "10011111100110" when menit_coba= 34 else
310             "10011111101101" when menit_coba= 35 else
311             "10011111111101" when menit_coba= 36 else
312             "10011110000111" when menit_coba= 37 else
313             "10011111111111" when menit_coba= 38 else
314             "10011111101111" when menit_coba= 39 else
315             "11001100111111" when menit_coba= 40 else
316             "11001100000110" when menit_coba= 41 else
317             "11001101011011" when menit_coba= 42 else
318             "11001101001111" when menit_coba= 43 else
319             "11001101100110" when menit_coba= 44 else
320             "11001101101101" when menit_coba= 45 else
321             "11001101111101" when menit_coba= 46 else
322             "11001100000111" when menit_coba= 47 else
323             "11001101111111" when menit_coba= 48 else
324             "11001101101111" when menit_coba= 49 else
325             "11011010111111" when menit_coba= 50 else
326             "11011010000110" when menit_coba= 51 else
327             "11011011011011" when menit_coba= 52 else
328             "11011011001111" when menit_coba= 53 else
329             "11011011100110" when menit_coba= 54 else
330             "11011011101101" when menit_coba= 55 else
331             "11011011111101" when menit_coba= 56 else
332             "11011010000111" when menit_coba= 57 else
333             "11011011111111" when menit_coba= 58 else
334             "11011011101111" when menit_coba= 59 ;
335
336     J_coba <= "01111110111111" when jam_coba = 0  else
337             "01111110000110" when jam_coba= 1  else
338             "01111111011011" when jam_coba= 2  else
339             "01111111001111" when jam_coba= 3  else
340             "01111111100110" when jam_coba= 4  else
341             "01111111101101" when jam_coba= 5  else
342             "01111111111101" when jam_coba= 6  else
343             "01111110000111" when jam_coba= 7  else
344             "01111111111111" when jam_coba= 8  else
345             "01111111101111" when jam_coba= 9  else
346             "00001100111111" when jam_coba= 10 else
347             "00001100000110" when jam_coba= 11 else
348             "00001101011011" when jam_coba= 12 else
349             "00001101001111" when jam_coba= 13 else
350             "00001101100110" when jam_coba= 14 else
351             "00001101101101" when jam_coba= 15 else
352             "00001101111101" when jam_coba= 16 else
353             "00001100000111" when jam_coba= 17 else
354             "00001101111111" when jam_coba= 18 else
355             "00001101101111" when jam_coba= 19 else
 56             "10110110111111" when jam_coba= 20 else
 57             "10110110000110" when jam_coba= 21 else
 58             "10110111011011" when jam_coba= 22 else
 59             "10110111001111" when jam_coba= 23 ;
```

ALAMAT PIN YANG DIGUNAKAN PADA FPGA

clk	Input	**p77**	
reset	Input	**p154**	
reset_alrm	Input	**p58**	
switch	Input	**p81**	
switch_alrm	Input	**p82**	
set_jam	Input	**p151**	
bel	Input	**p138**	
set_jam_alrm	Input	**p147**	**A2**
set_menit_alrm	Input	**p142**	
set_menit	Input	**p149**	

J<0>	Output	**p7**	
J<1>	Output	**p9**	
J<2>	Output	**p14**	
J<3>	Output	**p16**	
J<4>	Output	**p18**	
J<5>	Output	**p21**	
J<6>	Output	**p23**	
J<7>	Output	**p194**	
J<8>	Output	**p199**	
J<9>	Output	**p201**	
J<10>	Output	**p203**	
J<11>	Output	**p205**	
J<12>	Output	**p3**	
J<13>	Output	**p5**	

M<0>	Output	**p15**	**A1**
M<1>	Output	**p17**	
M<2>	Output	**p20**	
M<3>	Output	**p22**	
M<4>	Output	**p24**	
M<5>	Output	**p189**	
M<6>	Output	**p192**	
M<7>	Output	**p202**	
M<8>	Output	**p204**	
M<9>	Output	**p206**	
M<10>	Output	**p4**	
M<11>	Output	**p6**	
M<12>	Output	**p8**	
M<13>	Output	**p10**	
D<3>	Output	**p191**	
D<4>	Output	**p193**	
D<5>	Output	**p195**	
D<6>	Output	**p200**	

D<0>	Output	**p178**	
D<1>	Output	**p180**	
D<2>	Output	**p187**	
D<7>	Output	**p160**	
D<8>	Output	**p162**	
D<9>	Output	**p164**	**A2**
D<10>	Output	**p166**	
D<11>	Output	**p168**	
D<12>	Output	**p173**	
D<13>	Output	**p175**	

M_coba<0>	Output	**p125**
M_coba<1>	Output	**p127**
M_coba<2>	Output	**p132**
M_coba<3>	Output	**p134**
M_coba<4>	Output	**p136**
M_coba<5>	Output	**p90**
M_coba<6>	Output	**p95**
M_coba<7>	Output	**p102**
M_coba<8>	Output	**p109**
M_coba<9>	Output	**p111**
M_coba<10>	Output	**p113**
M_coba<11>	Output	**p115**
M_coba<12>	Output	**p120**
M_coba<13>	Output	p122

B2

J_coba<0>	Output	p119
J_coba<1>	Output	p121
J_coba<2>	Output	p123
J_coba<3>	Output	p126
J_coba<4>	Output	p129
J_coba<5>	Output	p133
J_coba<6>	Output	p135
J_coba<7>	Output	p97
J_coba<8>	Output	p99
J_coba<9>	Output	p101
J_coba<10>	Output	p108
J_coba<11>	Output	p110
J_coba<12>	Output	p112
J_coba<13>	Output	p114

(DODDY FIRMANSYAH 01524006)