PERANCANGAN ALAT DAN PERANGKAT LUNAK UNTUK MEMBUAT SINTESA PROGRAM DARI SUATU *PROGRAMMABLE LOGIC DEVICE*

TUGAS AKHIR

Diajukan Sebagai Syarat Untuk Memperoleh Gelar Sarjana Pada
Jurusan Teknik Elektro Fakultas Teknologi Industri
Universitas Islam Indonesia

Disusun oleh:

Nama : **AGUS KURNIAWAN**

No. Mhs : **01524137**

JURUSAN TEKNIK ELEKTRO

FAKULTAS TEKNOLOGI INDUSTRI

UNIVERSITAS ISLAM INDONESIA

YOGYAKARTA

2007

PERANCANGAN ALAT DAN PERANGKAT LUNAK UNTUK MEMBUAT SINTESA PROGRAM DARI SUATU *PROGRAMMABLE LOGIC DEVICE*

TUGAS AKHIR

Diajukan Sebagai Syarat Untuk Memperoleh Gelar Sarjana Pada
Jurusan Teknik Elektro Fakultas Teknologi Industri
Universitas Islam Indonesia

Disusun oleh:

Nama : **AGUS KURNIAWAN**

No. Mhs : **01524137**

JURUSAN TEKNIK ELEKTRO

FAKULTAS TEKNOLOGI INDUSTRI

UNIVERSITAS ISLAM INDONESIA

YOGYAKARTA

2007

LEMBAR PENGESAHAN

TUGAS AKHIR

PERANCANGAN ALAT DAN PERANGKAT LUNAK UNTUK MEMBUAT SINTESA PROGRAM DARI SUATU PROGRAMMABLE LOGIC DEVICE

Disususun Oleh:

Nama : AGUS KURNIAWAN

No. Mhs : 015 24 137

Yogyakarta, Februari 2007

Pembimbing I,

(Ir.Hj. Budi Astuti, MT)

Pembimbing II,

(Tito Yuwono,ST, M.Sc)

LEMBAR PENGESAHAN PENGUJI
TUGAS AKHIR

PERANCANGAN ALAT DAN PERANGKAT LUNAK UNTUK MEMBUAT SINTESA PROGRAM DARI SUATU PROGRAMMABLE LOGIC DEVICE

Disusun Oleh :

Nama : **AGUS KURNIAWAN**

No. Mahasiswa : **01 524 137**

Telah dipertahankan di Depan Sidang Penguji sebagai
Salah Satu Syarat untuk Memperoleh Gelar Sarjana Teknik Elektro
Fakultas Teknologi Industri Universitas Islam Indonesia
Jogjakarta,

Tim Penguji	**Tanda tangan**
1. **Ir.Hj. Budi Astuti, MT.**	
2. **Tito Yuwono, ST, M.Sc.**	
3. **Yusuf Aziz A, ST.**	

Mengetahui,
Ketua Jurusan Teknik Elektro
Fakultas Teknologi Industri
Universitas Islam Indonesia

(**Tito Yuwono,ST, M.Sc**)

HALAMAN PERSEMBAHAN

Dengan penuh hormat, cinta dan kasih sayang skripsi ini dipersembahkan sebagai ungkapan terima kasih kepada mereka yang sangat

***Ayahnda dan Ibunda tercinta** terima kasih atas limpahan cinta dan kasih sayang, doa, dukungan serta kepercayaan yang diberikan dan tak terbalaskan oleh ananda. **Kakak** dan **Adik** tersayang terima kasih atas **doa** serta dukungan kepada ananda*

Serta

***Mutiara hidupku**, pasangan jiwa, sahabat serta **pendamping setia** **Susantiyana** yang selalu menemani dikejauhan dan menyemangati dalam doa.*

Sahabat dan saudara yang selalu setia

Sebagian terpenting dalam c

Kenangan yang terindah

Aamiin Ya Rabbal' Alamin

MOTTO

"Sungguh bersama kesukaran pasti ada kemudahan"

(QS. Al-Baqarah : 153)

"Hai orang-orang yang beriman, mintalah pertolongan dari Allah dengan kesabaran dan salat. Sungguh Allah bersama orang-orang yang sabar "

(QS.Asy Syarh : 5)

"Tiada suatu kepayahan, kesakitan, kesedihan,kesusahan, penderitaan dan kesukaran sampaipun duri yang menyakitkan itu menimpa kepada seorang mu'min melainkan dengan itu semua Allah akan menutupi dosa-dosanya "

(HR. Bukhari dan Muslim)

KATA PENGANTAR

بِسْمِ اللَّهِ الرَّحْمَٰنِ الرَّحِيمِ

Assalamu'alaikum Wr.Wb.

Alhamdulillah, puji syukur kita panjatkan kehadirat Allah SWT karena atas rahmat, taufik dan hidayah-Nya penyusun dapat menyelesaikan tugas akhir ini dengan baik. Shalawat serta salam semoga selalu tercurah kepada Nabi Junjungan kita Nabi Muhammad SAW beserta keluarga dan para sahabat.

Tugas Akhir dengan judul ***"Perancangan Alat Dan Perangkat Lunak Untuk Membuat Sintesa Program Dari Suatu Programmable Logic Device"*** sebagai salah satu syarat untuk memperoleh gelar Sarjana Teknik Elektro pada Fakultas Teknologi Industri Universitas Islam Indonesia.

Selama proses penyelesaian Tugas Akhir ini, penulis menyadari banyak ilmu dan pengalaman serta dorongan dan doa yang diberikan kepada penulis. Pada kesempatan ini, Penulis ingin mengucapkan terima kasih yang tak terhingga kepada :

1. Bapak Dekan Fakultas Teknologi Industri,
2. Bapak. Tito Yuwono, ST, MSc selaku Ketua Jurusan Teknik Elektro Universitas Islam Indonesia sekaligus Dosen Pembimbing Tugas Akhir II
3. Ibu H. Ir. Budi Astuti, MT selaku Dosen Pembimbing Tugas Akhir I, atas kepercayaan, bantuan serta bimbingannya selama Penulis mengerjakan Tugas Akhir
4. Bapak Wahyudi Budi Pramono, ST selaku Dosen Pembimbing Akademik, terima kasih atas bimbingannya.

5. Segenap Dosen Teknik Elektro dan FTI yang pernah mengajar penulis selama Penulis kuliah atas ilmu dan pengetahuan yang diberikan.
6. **Kedua orang tua** penulis **di Banda Aceh**, terima kasih atas doa, perhatian, dukungan, kasih sayang dan cintanya sehingga akhirnya penulis dapat menyelesaikan kuliah.
7. "**Bundaku**" tercinta **Susantiyana** yang selalu menemani dan memberikan dukungan disaat susah dan senang sehingga penulis menjadi sabar dan tetap bersemangat menyelesaikan tugas akhir ini.
8. Mas Tri, mas Agung, mas Heri dan seluruh asisten di laboratorium Jurusan Teknik Elektro UII. terima kasih atas bantuan dan dukungannya.
9. Teman – teman seperjuangan terima kasih atas bantuan dan kerjasamannya.
10. Terima Kasih kepada semua pihak yang telah membantu dalam pengerjaan Tugas Akhir ini, yang tidak dapat penulis sebutkan satu persatu.

Penyusun menyadari bahwa laporan ini masih banyak kekurangannya, oleh karena itu kritik dan saran yang membangun sangat penyusun harapkan. Akhirnya mudah – mudahan laporan ini dapat bermanfaat kepada penulis pada khususnya dan pembaca pada umumnya, amin.

Wassalammualikum Wr Wb.

Yogyakarta, 15 Februari 2007

Penulis

ABSTRAK

Setiap PLD yang digunakan umumnya diproteksi guna menghindari penjiplakan sistem oleh pihak lain. Maka diperlukan analisa yang cukup rumit dan membutuhkan ketelitian tinggi dalam menganalisa kombinasi biner dari masukan dan keluarannya, dengan tujuan mendapatkan *equation* yaitu suatu rumusan yang menjelaskan hubungan antara input dan output PLD. Maka dirancang suatu *software* untuk mempermudah penganalisaan *Equation* PLD, dalam hal ini dipilih software Delphi 6.0. Dirancang juga *hardware* pendukung seperti Programmer yang berfungsi untuk memasukkan *equation* yang telah berbentuk file Jedec untuk diprogramkan ke PLD, serta simulator yang berfungsi membaca masukkan dan keluaran dari PLD dan menghasilkan *equation* secara otomatis. Hal ini dapat memudahkan untuk mensintesa PLD baru yang memiliki Sistem yang sama dengan PLD yang lama. Dalam pengujian, digunakan alat penghitung pada alat wartel dan alat pengaktifan 7segmen sebagai bukti bahwa simulator PLD yang dirancang dapat bekerja dengan baik, tentunya alat ini dapat bekerja secara umum dalam arti semua PLD dengan berbagai macam sistem kerja dapat dibuat tiruannya.

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I

PENDAHULUAN

1.1 Latar Belakang

Mendesain rangkaian digital yang merupakan sebagian dari suatu sistem yang kompleks dengan baik, seringkali merupakan suatu masalah tersendiri. Untuk menghasilkan desain akhir sesuai dengan *performance* yang diinginkan, kita sering dihadapkan pada pertimbangan yang saling berbenturan /membingungkan.

Untuk merealisasikannya, pertama kali yang dilakukan adalah menentukan blok diagram dari sistem secara global/keseluruhan. Setelah itu muncul pertimbangan-pertimbangan untuk menentukan isi dari blok-blok tersebut, misalnya bagaimana bentuk data/sinyal, komponen pokok apa yang akan digunakan dan lain-lainnya. Selanjutnya ditentukan hubungan antara input-output dalam bentuk ekspresi logika. Kemudian pada saat akan mengimplementasikan ekspresi tersebut dalam perancangan pasti berhadapan dengan masalah pemilihan komponen logika apa yang akan digunakan. Ini cukup membingungkan. Bagian ini merupakan bagian yang cukup menentukan untuk menghasilkan desain yang optimal, dalam arti murah, mudah interkoneksinya, membutuhkan tempat yang sekecil-kecilnya, waktu desain yang singkat, kebutuhan daya yang kecil, dan lain-lainnya, dan yang terakhir kepastian akan bekerja dengan baik.

1.3 Batasan Masalah

Kemampuan PLD dalam suatu sistem digital cukup luas, dapat sebagai pengganti fungsi AND, OR, INVERT, XOR, REGISTER maupun CARRY.

Dalam tugas akhir ini kemampuan sistem yang dirancang dibatasi untuk memecahkan program PLD 20 pin sebagai fungsi AND, OR, INVERT dan XOR saja, sedangkan jenis PLD yang digunakan GAL 16v8 yang mempunyai sistem kerja yang dibatasi sebagai alat hitung pada wartel dan sebagai pengaktifan 7segmen

1.4 Manfaat Penelitian

Dalam kebanyakan peralatan digital seperti EPROM PROGRAMMER, perangkat wartel, mesin lift sering dipakai suatu PLD yang programnya diproteksi, sehingga tidak dapat meminta program tersebut dari pembuatnya, apalagi kalau peralatan tersebut sudah merupakan produk yang usang yang telah tidak didukung servisnya oleh produsennya, maka perlatan tersebut sudah tidak bisa diperbaiki lagi. Dengan tugas akhir ini diharapkan mampu membantu mendapatkan program sintesis dari peralatan yang memakai PLD *device*.

1.5 Tujuan Penelitian

Dalam pembuatan alat dan perangkat lunak untuk membuat sintesa program PLD, maka tugas akhir ini bertujuan :

1. Mengetahui langkah-langkah pembuatan perangkat keras, mulai perancangan rangkaian, sampai perakitan ke papan PCB.
2. Memahami cara pemecahan program dari PLD secara manual yang selanjutnya dapat menyusunnya dalam sebuah sistem perangkat lunak.

3. Membantu mempercepat kerja manusia dari cara manual menjadi sistem perangkat lunak yang cepat, efisien dan mudah penggunaannya.

1.6 Sistematika Penulisan

Laporan tugas akhir ini disusun secara sistematis menjadi lima bab, yang terdiri dari :

BAB I Pendahuluan

Bab ini berisi tentang latar belakang masalah, perumusan masalah, batasan masalah, tujuan penulisan dan sistematika penulisan.

BAB II Dasar Teori

Bab ini akan menjelaskan tentang teori – teori yang digunakan dalam perancangan dan pembuatan alat.

BAB III Perancangan Sistem

Bab ini menjelaskan perancangan sistem, komponen yang digunakan serta desain perangkat kerasnya.

BAB IV Analisis dan Pembahasan

Bab ini akan menganalisa dan menjelaskan hasil dari pengujian alat yang dibuat dan akan dibandingkan dengan teori yang digunakan.

BAB V Penutup

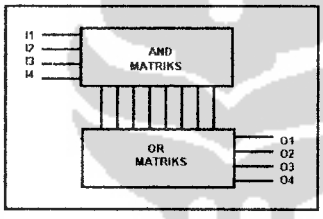
Bab ini berisi kesimpulan dari peralatan yang dibuat dan berisi saran guna pengembangan dimasa yang akan datang.

BAB II

DASAR TEORI

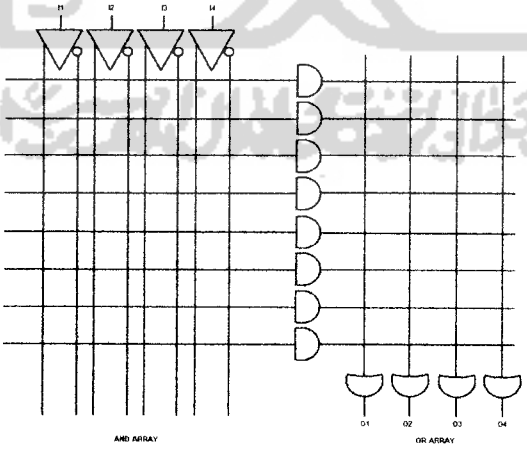
2.1 Sedikit Mengenai PLD

PLD (*Programable Logic Device*) berisi beberapa buah AND *gate* dan OR *gate* dengan titik-titik hubung input/output tiap *gate* berupa matriks yang dapat diprogram oleh pemakai. Satu chip PLD yang telah diprogram dapat diekivalenkan dengan beberapa chip TTL. Secara blok diagram, PLD berisi dua blok *gate*, yaitu blok AND-*gate* dan blok OR-*gate*.



Gambar 2.1 AND-*gate* dan OR-*gate*

Adapun susunan AND-OR *gate* secara umum dari PLD adalah sebagai berikut :



Gambar 2.2 Susunan AND-OR

Pada bagian paling depan terdapat satu inverter untuk setiap input, Blok yang dapat diprogram, pada awalnya (sebelum diprogram) terhubung melalui dioda-fuse. Pada saat memprogram, harus diputus dioda-fuse yang tidak diperlukan, sedangkan yang diperlukan dibiarkan terhubung. Sebagai tambahan, beberapa PLD mempunyai fungsi REGISTER.

Secara tipikal, PLD dengan kemasan 20 pin dapat menggantikan 4 sampai dengan 12 chip TTL standart. Sangat efektif untuk mengurangi luas PCB. PLD dibuat dengan teknologi Schottky TTL sehingga sangat cepat (>25MHz), dan mudah dikoneksikan dengan komponen lain.

2.2 Kategori PLD Berdasar *Array* Yang Dapat Diprogram

2.2.1 *Programmable* AND – *Fixed* OR

PLD jenis ini hanya memungkinkan pemrograman untuk jalur perkalian saja, yaitu masukkan untuk gerbang AND. Sedangkan masukan gerbang OR, yaitu jalur penjumlahan sudah tertentu, sudah *fixed*, tidak dapat diprogram.

2.2.2 *Fixed* AND – *Programmable* OR

PLD jenis ini mempunyai jalur masukan gerbang AND yang tidak dapat diprogram, sudah *fixed*, dan jalur masukan gerbang OR yang dapat diprogram untuk fungsi logika tertentu.

2.2.3 *Programmable* AND – *Programmable* OR Device

PLD jenis ini memungkinkan pemrograman baik jalur gerbang AND maupun jalur gerbang OR.

2.3 Kategori PLD Berdasarkan Cara Penghapusan

2.3.1 *Programmable Array Logic* (PAL)

PLD yang termasuk jenis ini merupakan jenis yang hanya dapat diprogram sekali. Setelah dioda-fuse putus pada saat diprogram, tidak dapat dikembalikan terhubung, sedangkan yang masih terhubung dapat diputuskan dengan memprogramnya kembali.

2.3.2 *Erasable Programmable Logic Device* (EPLD)

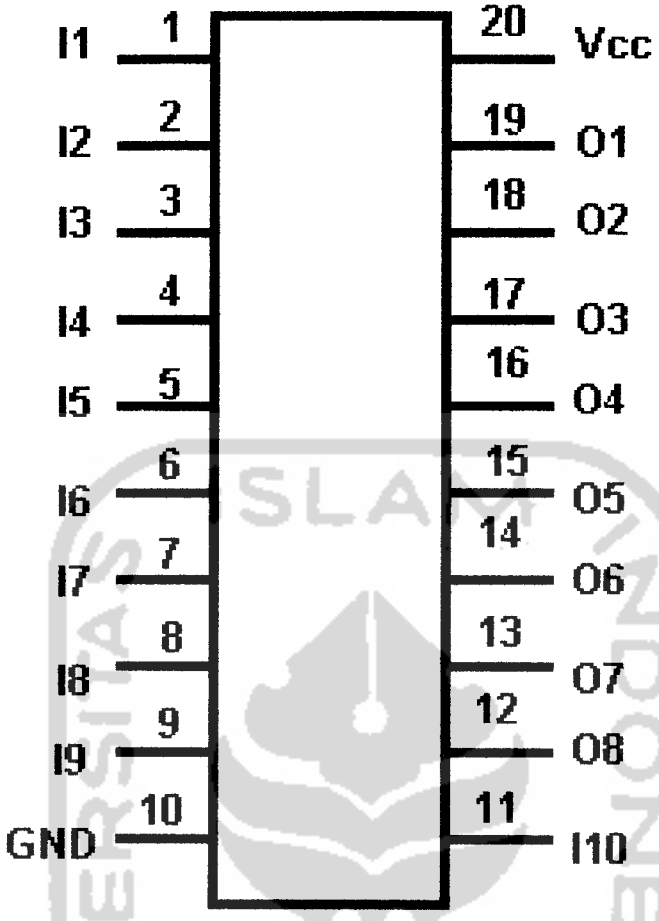
EPLD selain dapat diprogram oleh pemakai juga memungkinkan pemakai menghapus apa yang telah diprogram. Untuk menghapus apa yang telah diprogram dipergunakan cahaya ultra violet. Selanjutnya pemakai dapat memprogramnya kembali. Prinsip ini sama dengan yang diterapkan pada EPROM.

2.3.3 *Generic Array Logic* (GAL)

Generic Array Logic yang dikeluarkan oleh Lattice Semiconductor Corporation menggunakan teknologi *electrically erasable cell*. Dengan teknologi ini, memungkinkan penghapusan apa yang telah diprogram secara electric. *Electrically erasable cell* mempunyai waktu hapus yang cepat, namun operasi penulisan yang dapat dilakukan terbatas.

2.4 Diskripsi Pin-pin Pada PLD 20 Pin

Pada Gambar 2.3 ditunjukkan fungsi dari masing-masing pin pada PLD 20 pin.



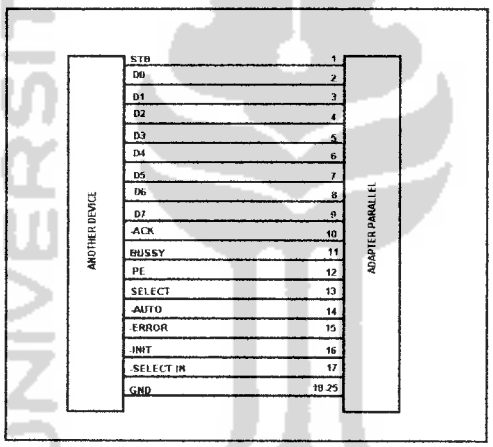
Gambar 2.3 PLD 20 pin

2.5 Adapter Port Paralel

Adapter adalah peralatan yang dapat menghubungkan komputer dengan pihak di luar komputer. Adapter paralel adalah peralatan yang menghubungkan komputer dengan printer, itulah sebabnya adapter pararel disebut juga adapter printer.

2.5.1 Informasi Fungsi Pin-pin Adapter Paralel

Pada saat adapter dirancang khusus untuk menghubungkan printer dengan komputer dan mengirim data 8 bit menuju printer. Sedangkan printer hanya memberikan kode-kode kondisi ke komputer sebanyak 5 bit saja.



Gambar 2.3 Pin-pin adapter paralel

Dari informasi fungsi pin-pin tersebut terlihat bahwa ada 12 saluran keluar tetapi hanya ada 5 pin saluran masuk.

2.5.2 Alamat Masing-masing Fungsi Pada Port Paralel

Printer adapter dapat merespon instruksi input dan output. Instruksi output mentransfer data 8 bit dari bus ke pin-pin konektor DB25.

Berikut ini deskrispsi pada port paralel :

Tabel 2.1 Output ke alamat 378H

Output ke alamat 378H							
Bit 7 Pin 9	Bit 6 Pin 8	Bit 5 Pin 7	Bit 4 Pin 6	Bit 3 Pin 5	Bit 2 Pin 4	Bit 1 Pin 3	Bit 0 Pin 2

Instruksi pada port ini meng-*capture* data dari data bus dan menghadirkan pada pin-pin DB-25. Masing-masing pin mempunyai arus 2,6 ma sampai dengan 24 ma.

Tabel 2.2 Output ke alamat 37AH

Output ke alamat 37AH							
			Bit 4 IRQ Enable	/Bit 3 Pin 17	Bit 2 Pin 16	/Bit 1 Pin 14	/Bit 0 Pin 1

Instruksi pada alamat ini menyebabkan port mengakses ke 5 *least significant* bits dari data bus, 4 dari *least significant* bits dihadirkan pada konektor, beberapa dari output di-invertkan.

Jika bit ke 4 diisi 1, card akan menginterupsi CPU pada kondisi bila pin 10 dalam keadaan transisi dari *High* ke *Low*. Pin-pin ini merupakan *driver open* kolektor yang membutuhkan resistor *pull up* ke +5v, memakai resistor sekitar 4,7K.

Tabel 2.3 Input dari alamat 379H

Input dari alamat 379H							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Pin 11	Pin 10	Pin 12	Pin 13	Pin 15	-	-	-

Untuk membaca input dari konektor pin 10, 11, 12, 13 dan 15 berada pada alamat 379H seperti terlihat pada tabel 2.2.3.

2.6 Pemrograman DELPHI

Delphi merupakan pemrograman terstruktur yang berbasis pada obyek Pascal dari Borland, bekerja pada ruang lingkup sistem operasi ***Windows.*** Struktur bahasanya dengan bahasa objek pascal ini sangat mendukung untuk pemrograman **OOP**(*Object-Oriented Programing*), maksudnya perluasan atas pemrograman terstruktur yang mengutamakan pemakaian-ulang dan *enkapsulasi* data (kombinasi data ke dalam sebuah unit tunggal) berdasarkan fungsinya, DELPHI juga mempunyai fungsi untuk memberikan fasilitas pembuatan aplikasi visual, sehingga meningkatkan produktivitas dalam pembuatan program yang meliputi kualitas pengembangan visual, kecepatan kompilasi, kekuatan bahasa pemrograman, fleksibelitas terhadap arsitektur basis data, dan pola desain dan pemakaian yang diwujudkan oleh *framework*nya. Tugas akhir ini menggunakan DELPHI versi 6.0.

IDE (*Integrated Development Environment*) merupakan lingkungan tempat semua *tools* yang diperlukan untuk desain, menjalankan, dan mengetes aplikasi yang disajikan

dan terhubung dengan baik, sehingga memudahkan pengembangan program. IDE ini akan muncul pertama kali saat membuka atau akan menjalankan program DELPHI, IDE terdiri atas *main window, component palette, toolbar, object inspector, form desainer, code editor,* dan *code explorer,* sedangkan tampilan IDE seperti berikut.

Main Window (jendela utama) mempunyai tiga bagian yaitu menu utama, *toolbar,* dan *component palette.* Menu utama merupakan bagian yang digunakan untuk mendukung jalannya pembuatan program atau aplikasi, meliputi *file, edit search, view, project, run, component, database, tools, help.* Bagian menu utama tersebut mengandung perintah-perintah yang akan digunakan untuk mendukung pembuatan program atau aplikasi tersebut walaupun di dalamnya ada sebagian yang sudah ditampilkan dalam *toolbar* atau *speedbar. Toobar* merupakan bagian menu utama yang dibuat dalam bentuk *icon* dengan tujuan untuk pemanfaatan perintah denagan cepat tanpa harus membuka menu utama, sehingga lebih mempermudah dan mempercepat dalam membuat aplikasi atau program. *Component palette* merupakan *toolbar* yang berisi *page control* dengan semua komponen tersebut yang digunakan untuk membantu dalam penghubung antara program dengan aplikasi yang diinginkan atau sebagai *interface*nya.

Form desainer merupakan jendela yang digunakan untuk membuat aplikasi atau meletakkan komponen yang akan digunakan dalam aplikasi. Penggunaan sebagai berikut dengan cara memilih komponen dari *component pallette* dan meletakkan ke dalam *form,* setelah di dalam *form* maka komponen tersebut dapat diatur posisinya atau ukuran dengan *mouse,* serta dapat mengubah tampilan dan perilaku komponen dengan menggunakan *object inspector* dan *code editor.*

Object inspector terdiri atas dua bagian, yaitu bagian *tab properties* yang berfungsi untuk membuat atau *property* yang dimiliki oleh suatu *item* sesuai dengan yang diinginkan. Bagian *tab events* berisi tentang *event* yang dapat direspon oleh suatu obyek dalam *form*. Jadi setiap obyek yang sudah ada dalam *form* dapat diatur dengan menggunakan *tab properties* serta dapat diperlakukan sesuai dengan keinginan melalui *tab events*.

Code editor merupakan jendela penyunting yang digunakan untuk menuliskan program DELPHI. Editor Delphi mempunyai fasilitas *highlight* untuk memudahkan menemukan kesalahan, fasilitas kerangka program sehingga dalam membuat program tidak perlu menuliskan program seluruhnya.

Code explorer digunakan untuk mempermudah melakukan navigasi terhadap *file unit*. *Code explorer* berisi pohon yang menampilkan semua *type, class, property, method, variable global* dan *rutin global* yang didefinisikan dalam unit serta menampilkan semua unit yang ada di *clausa uses*.

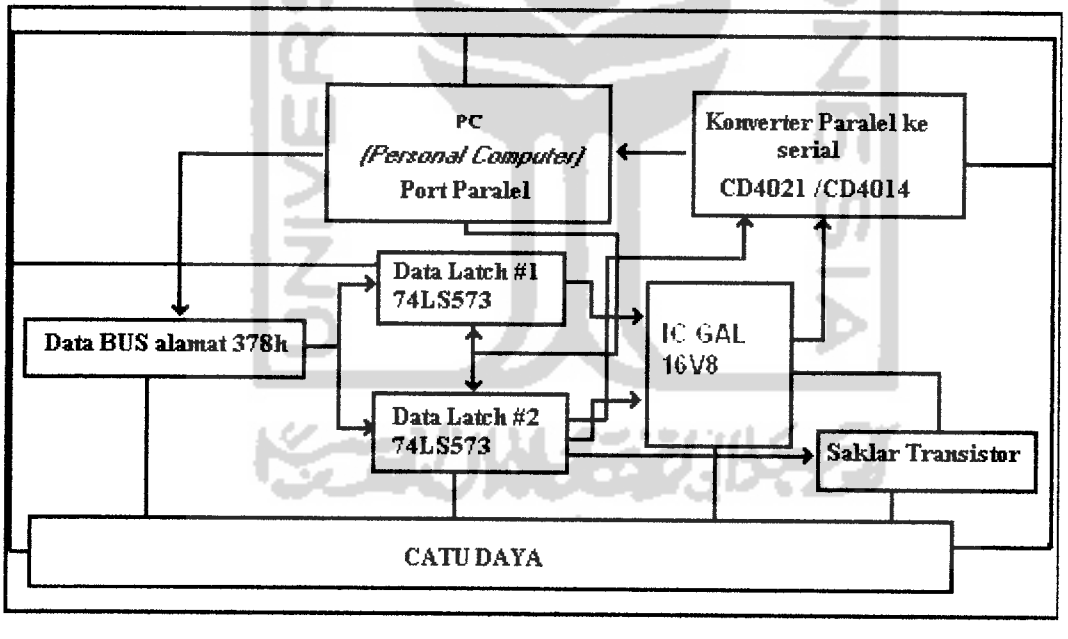
BAB III

PERANCANGAN SISTEM

Pada bab ini akan dibahas mengenai perancangan sistem yang di dalamnya terdapat perancangan rangkaian elektronika dan sistem *software* dengan Delphi dari simulator PAL, berdasar teori yang dibahas pada bab-bab terdahulu.

3.1 Blok Diagram.

Blok diagram dari sistem diperlihatkan seperti pada gambar blok diagram berikut ini.



Gambar 3.1 Blok diagram simulator PLD

Penjelasan dari diagram blok di atas adalah sebagai berikut :

PC (*Personal Computer*)

Blok ini terdiri dari sebuah komputer dengan memanfaatkan paralel port sebagai masukan dari data yang dikirimkan oleh rangkaian PLD simulator. Analisa dari data-data yang didapat dilakukan oleh perangkat lunak komputer dengan memakai bahasa pemrograman delphi.

Data Latch.

Blok ini terdiri dari dua buah IC (*Integrated Circuit*) tipe 74LS573 yang dipakai untuk mengembangkan 8 bit data yang dikirim dari PC pada *port* paralel di alamat 378h menjadi 10 bit data sebagai kombinasi masukkan di PLD, serta 1 bit sebagai saklar yang menghubungkan PLD dengan power suplai.

Power Switch

Blok ini dibuat dengan sebuah transistor PNP yang dipergunakan untuk memberi tegangan PLD, sebagai indikator bahwa PLD dalam keadaan terhubung dengan power suplai dipasang LED, indikator ini harus benar-benar diperhatikan statusnya karena bila dalam keadaan PLD mendapat tegangan maka sama sekali tidak diijinkan melepas atau memasangnya di soket simulator karena hal ini akan merusakkan IC PLD tersebut.

Konverter Paralel ke serial

Blok ini menggunakan IC CD4021 yang diperlukan untuk mengubah data paralel dari keluaran IC PLD ke data dengan format data serial. Konversi dilakukan mengingat

kemampuan port paralel PC yang hanya mampu membaca masukkan sebanyak 5 bit data, di alamat 379Ah.

Power suplai

Adalah blok sebagai pemberi tegangan ke rangkaian PLD simulator maupun IC PLD yang dianalisa.

3.2 Perancangan Perangkat Keras

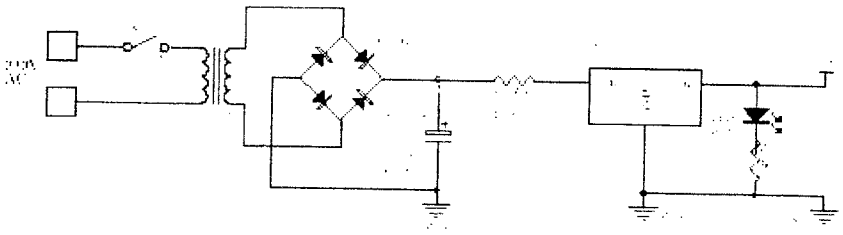
Pada penelitian ini perangkat keras dari sistem terdiri dari IC PLD yang dianalisa, data *latch*, sistem *switch*, konverter paralel ke seri dan rangkaian catu daya.

3.2.1 Rangkaian Catu Daya

Rangkaian catu daya merupakan bagian yang sangat penting karena tanpa catu daya alat ini tidak dapat bekerja. Alat ini memerlukan catu daya yang dapat memberikan tegangan sebesar 5VDC, yang digunakan untuk meberikan tegangan kerja dari rangkaian PLD simulator maupun ke IC PLD yang akan dianalisa.

Mempertahankan suatu level tegangan yang konstan sangat diperlukan dalam rangkaian catu daya ini, dengan demikian rangkaian catu daya pada tugas akhir ini menggunakan regulator tegangan (*voltage regulator*) yang berbentuk IC (*integrated circuit*) yang mengandung sejumlah rangkaian untuk tegangan referensi, sistem pengontrol, penguat komparator, dan pelindung tegangan berlebih (*over voltage*).

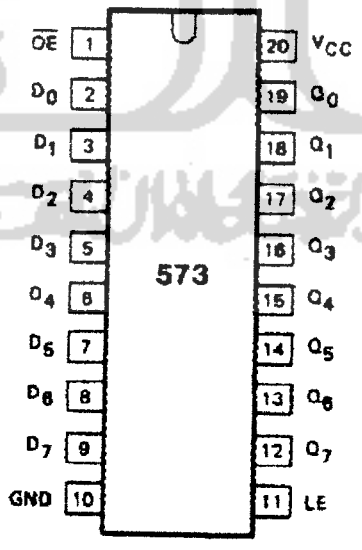
Pada alat ini pelindung tegangan berlebih menggunakan regulator jenis positif regulator dengan tipe LM7805 untuk penstabil tegangan 5VDC.



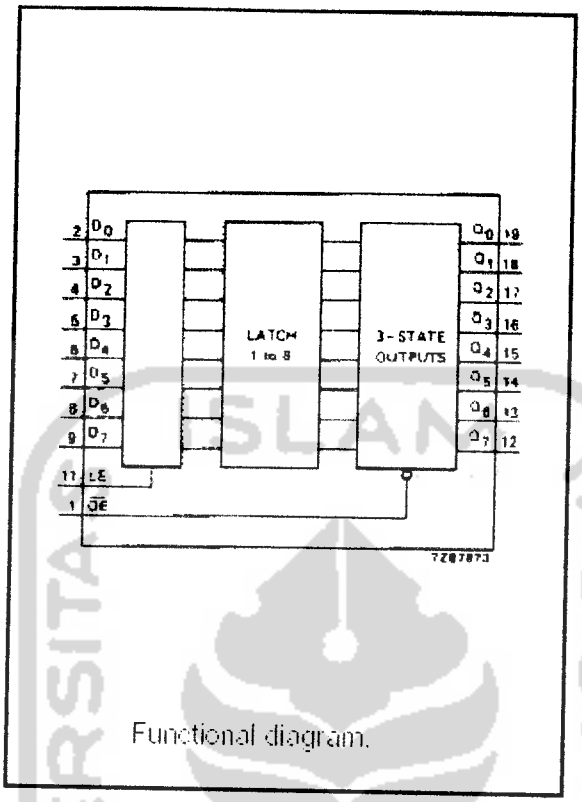
Gambar 3.2 Rangkaian catu daya

3.2.2 Data Latch

Pada sistem PLD simulator untuk menerima data dari *port* paralel PC yang berupa 8 bit data untuk dikembangkan menjadi 10 bit sebagai kombinasi masukkan ke IC PLD, serta untuk mengatur *switch* transitor untuk memberikan/memutus tegangan dari rangkaian catu daya ke IC PLD.

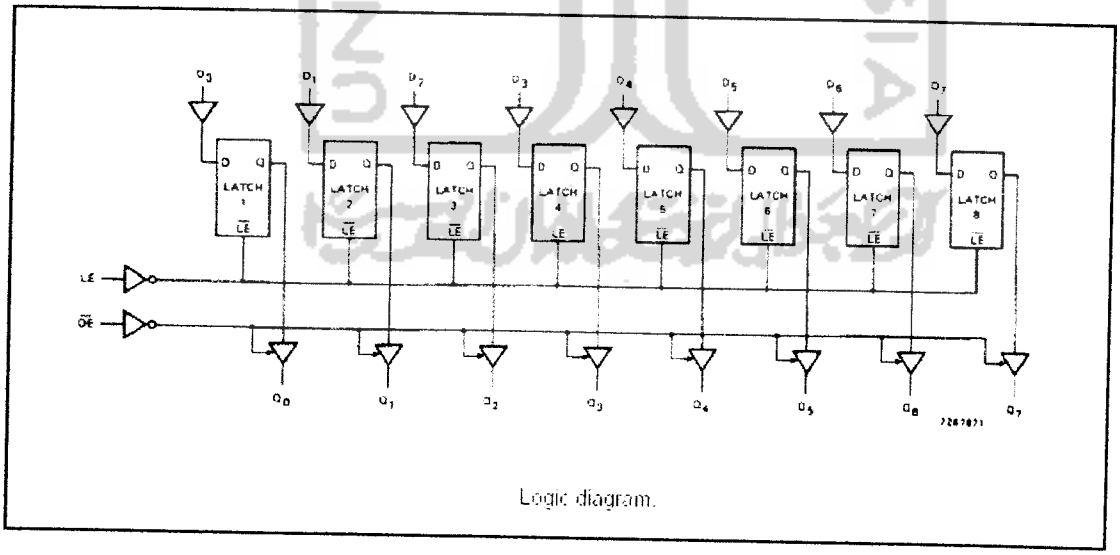


Gambar 3.3 Susunan pin-pin IC 74LS573



Functional diagram.

Gambar 3.4 Diagram fungsi dari IC 74LS573



Logic diagram.

Gambar 3.5 Diagram logika 74LS573

Tabel 3.1 Tabel kebenaran IC 74LS573

$\overline{\text{OUTPUT ENABLE}}$	$\overline{\text{LATCH}}$ ENABLE	DATA	OUTPUT
L	H	H	H
L	H	L	L
L	L	l	L
L	L	h	H
H	X	X	Z

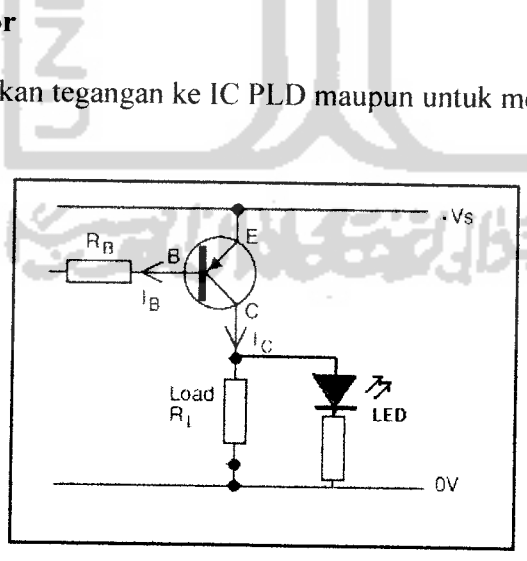
NOTE: H = High voltage Level, L = Low voltage Level, X = Don't Care, Z = High Impedance State, l = Low voltage level one set-up time prior to the high to low latch enable transition, h = High voltage level one set-up time prior to the high to low latch enable transition.

Rangkaian data *latch* terdiri dari dua buah IC jenis 74LS573. Kerja dari IC ini adalah sebagai berikut :

- mula-mula kombinasi data pararel yang dibutuhkan diaplikasikan di data bus.
- Untuk memasukkan data yang sudah ada di data bus ke register IC dengan memberikan 1 (satu) denyut postif di pin LE dari IC.
- Kombinasi data yang ada di dalam register selanjutnya dapat ditampilkan ke keluaran IC dengan memberikan logika negatif pada pin /OE (*output enable*).

3.2.3 Saklar Transistor

Untuk memberikan tegangan ke IC PLD maupun untuk memutuskan IC PLD



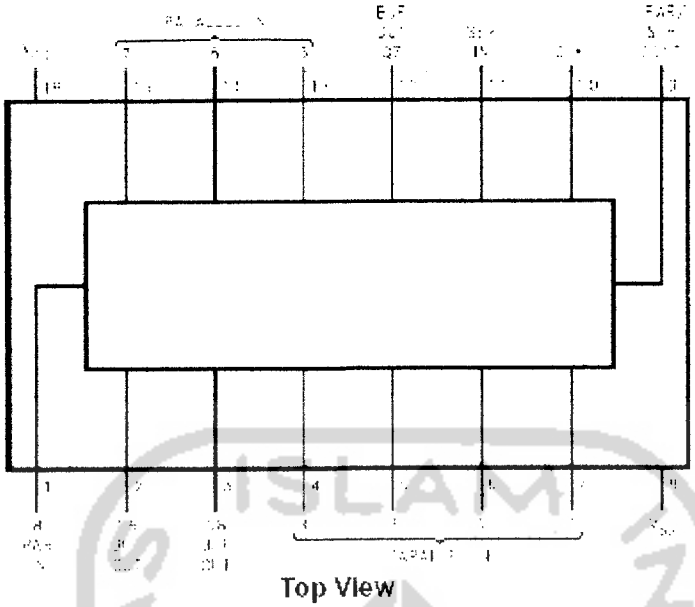
Gambar 3.6 Gambar rangkaian transistor sebagai saklar

dengan sistem catu daya dipergunakan sebuah transistor jenis PNP dalam hal ini digunakan tipe BC557. Sebuah tahanan sebesar 2k2 dipasang antara basis transistor dengan keluaran IC data *latch* dimaksudkan untuk memberikan tegangan bias transistor.

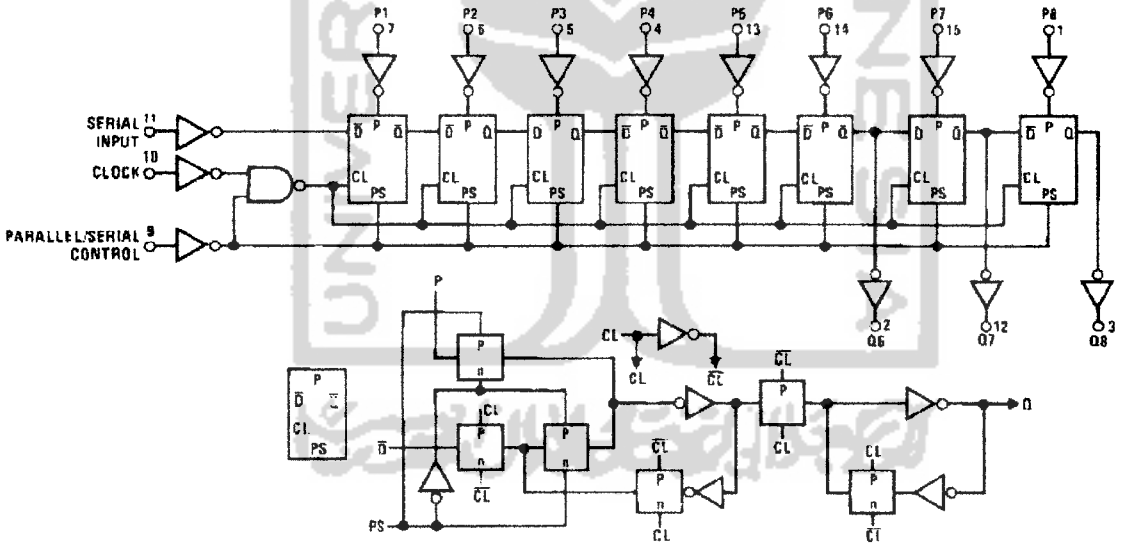
Bila logika positif diaplikasikan di basis transistor maka transistor dalam keadaan ON, jika logika rendah diberikan maka transistor menjadi OFF. Sebuah LED dipasang untuk memberi informasi apakah IC PLD sedang terhubung dengan catu daya atau tidak, indikator ini penting karena IC PLD akan rusak bila dilepas atau dipasang dalam keadaan mendata tegangan.

3.3.4 Rangkaian Konverter Paralel ke Serial.

Agar keluaran dari IC PLD sebanyak 8 bit data dapat diterima oleh komputer, mengingat kemampuan *port* paralel komputer hanya mampu menerima 5 bit data di alamat 379, maka diperlukan sebuah rangkaian konversi data paralel ke data serial. Rangkaian konverter ini dipergunakan IC jenis CD4021 atau CD4014 yang berfungsi sebagai 8-*stage static shift register.*



Gambar 3.7 Diagram koneksi CD4042/4014



Gambar 3.8 Diagram logika CD4021/CD4014

Tabel 3.2 Tabel kebenaran CD4021/CD4014

C_L (Note 1)	Serial Input	Parallel/ Serial Control	PI 1	PI n	Q1 (Internal)	Q_n (Note 2)
X	X	1	0	0	0	0
X	X	1	0	1	0	1
X	X	1	1	0	1	0
X	X	1	1	1	1	1
⌐	0	0	X	X	0	$Q_{n\ 1}$
⌐	1	0	X	X	1	$Q_{n\ 1}$
¬	X	0	X	X	Q1	Q_n

X Don't care case

Note 1: Level change

Note 2: No change

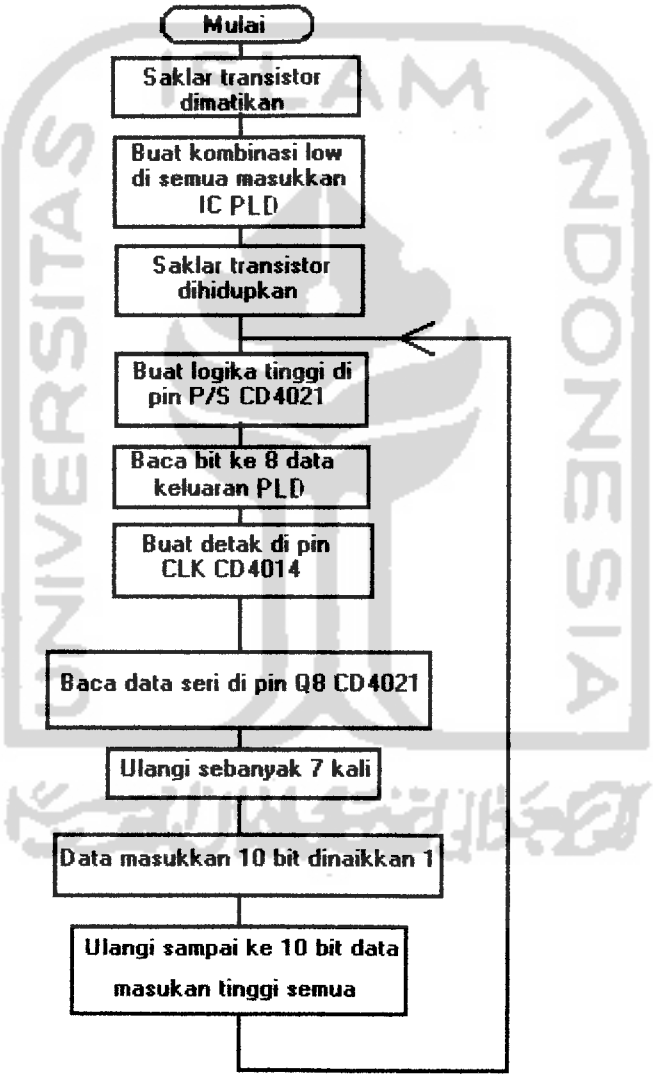
Dari tabel kebenaran dapat digunakan untuk membuat alur logika untuk mengoperasikan IC CD4021/CD4014, sebagai berikut :

- Data keluaran yang akan dibaca dikoneksikan ke pin P1 s/d P8
- Dengan membuat logika tinggi di pin paralel/serial kontrol maka register internal akan menerima data paralel dari P1 s/d P8. Pada saat ini dapat dibaca data keluaran di port Q8 yang merupakan bit ke 8 dari data paralel.
- Selanjutnya data berikutnya dapat dibaca dengan cara membuat logika rendah pin paralel/serial kontrol, setiap detak postif akan diterima 1 bit data di pin Q8.
- Data yang diterima adalah berurutan dari MSB (*Most Significant Bit*) ke LSB (*Least Significant Bit*).

3.3 Perancangan perangkat lunak

3.3.1 Diagram Alir

Untuk membuat perangkat lunak dipakai bahasa pemrograman delphi, diagram alir dari sistem PLD simulator dapat dilihat pada Gambar 3.9.



Gambar 3.9 Diagram alir perangkat lunak

3.3.2 Program untuk mengirim data ke IC 74LS573

Procedure ini dipakai untuk membuat *low* pin LE (*Latch Enable*) 74LS573 Kaki LE terhubung di bit 4 data bus, pada alamat 37Ah sehingga :

- untuk membuat *high* kombinasinya XXXX1011 = 11 desimal
- untuk membuat *low* kombinasinya XXXX0011 = 3 desimal
- bit ke 3 low karena rangkaian internal *port* paralel alamat 37Ah terdapat gerbang NOT
- bit ke 5 s/d ke 8 tidak dapat diatur dari luar.

```
procedure latch;
var
   dlatch:byte;
begin
   dlatch:=11;
   PortOut(890,dlatch);
   dlatch:=3;
   PortOut(890,dlatch);
   dlatch:=11;
   PortOut(890,dlatch);
end;
```

Procedure untuk mengirimkan kombinasi data di bus data ke register 74LS573, pin input 74LS573

dihubungkan ke alamat 378h dari *port* paralel}

```
procedure data_out;
begin
{  kombinasi data di bus data dipersiapkan}
   d573:=data_bus
   PortOut(888,d573);
{  Dengan membuat pin LE (Latch) low maka data masuk ke
regiter
   74LS573}
   latch;
{  Bebaskan kembali bus data}
   PortOut(888,0);
end;
```

3.3.3 Program untuk membaca data di port paralel alamat 379H dari IC CD4021

```
procedure TForm1.Baca_Data(Sender: TObject);
var
   dstring:string;
   t1:string;
   i,j:integer;
begin
   dstring:='';
```

```
{  Membuat pin P/S CD4021 berlogika tinggi untuk memsukkan
data ke
   register CD4021. Pin P/S dikoneksikan pada Q7 IC 74LS573
ke 2, maka
   bit ke 7 harus dibuat high}
   d573_2:=d573_2 or 64;
   PortOut(888,d573_2);
{  Data di bus di latch supaya masuk di register 74LS573 ke
2}
   latch2;
{  Membuat pin P/S CD4021 low}
   d573_2:=d573_2 xor 64;
{  Baca 8 bit data secara serial}
   for i:=1 to 8 do
   begin
      PortOut(888,d573_2);
      latch2;
      PortOut(890,11);
      InValue := PortIn(889);
      if invalue=56 then
         dstring:=dstring+'0'
      else
         dstring:=dstring+'1';
{  Buat 1 detak postif}
```

```
        PortOut(890,11);
        PortOut(890,15);
    end;
end;
```

3.3.4 Cara menghidupkan/mematikan saklar transistor

```
procedure TForm1.Switch_on_off(Sender: TObject);
begin
{   basis transistor dihubungkan di Q2 keluaran 74LS573
    untuk menghidupkan/mematikan saklar transistor dengan
mengatur data
    bit ke 2}

{   Data dipersiapkan di bus data}
    dout:=2;

{   akan dikirim ke 74LS573 yang ke 2}
    d573_2:=d573_2 xor dout;
    PortOut(888,d573_2);

{   membuat pin LE 74LS573 yang ke 2 low untuk membuat data
tersimpan di
    register internal 74LS573}
```

```
    latch2;
    PortOut(888,0);
end;
```

BAB IV

PENGUJIAN ALAT

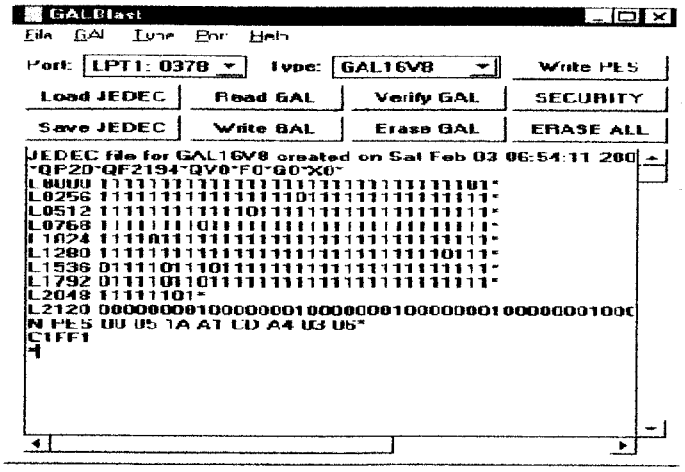
Dalam bab ini akan diperlihatkan hasil pengujian dari sistem yang dibuat, pengujian yang dilakukan meliputi :

- Pembacaan PLD yang tidak diproteksi
- Pembacaan PLD yang diproteksi
- Pembacaan dari PLD kosong
- Hasil pembacaan PLD kosong dengan alat
- PLD yang diproteksi dibaca dengan simulator
- Pembuatan persamaan oleh simulator
- Proses kompilasi persamaan dengan kompiler PLD
- Proses pemograman PLD
- Pembacaan kembali PLD hasil sintesa
- Pengujian PLD hasil dengan alat wartel

4.1 Pembacaan PLD dengan programmer

4.1.1 Pembacaan PLD yang tidak diproteksi

Pada pengujian ini dilakukan dengan membaca PLD yang tidak mengalami proteksi dengan bantuan programmer.

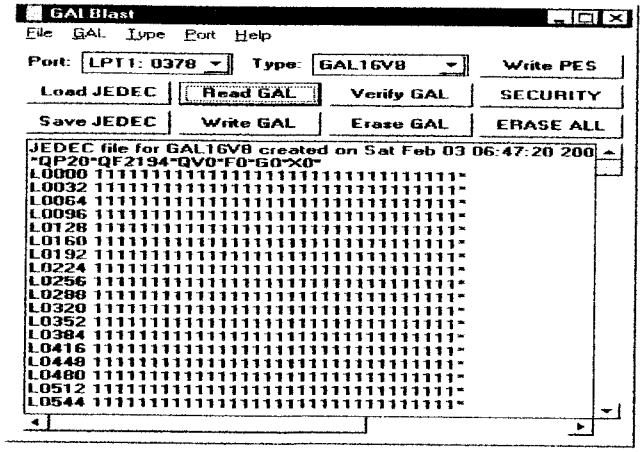


Gambar 4.1 Hasil pembacaan PLD yang tidak diproteksi dengan programmer

Gambar 4.1 menunjukkan bahwa data biner yang terprogram di dalam PLD dapat diketahui dengan baik oleh programmer, dalam hal ini kita tidak mengalami kesulitan untuk melakukan pengkopian dari PLD tersebut, simulator dalam hal ini tidak diperlukan. Kenyataan di lapangan menunjukkan bahwa hal semacam ini jarang terjadi, seorang pembuat sistem pada umumnya melakukan proteksi terhadap proses penanggulangan pembajakan sistem yang dibuatnya, salah satunya dengan memproteksi program yang dibuat, pekerjaan ini juga dilakukan dalam pembuatan program dalam PLD.

4.1.2 Pembacaan PLD yang diproteksi

Gambar 4.2 memperlihatkan hasil pembacaan PAL yang telah mengalami proses proteksi, data biner yang didapat dari pembacaan tidak dapat diketahui dari pembacaan tersebut.

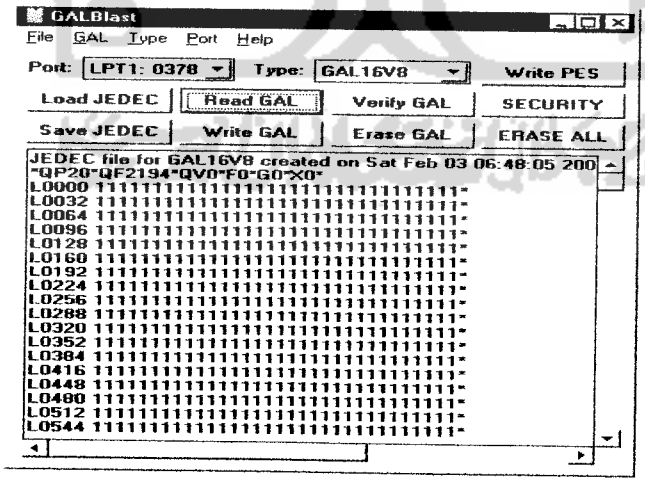


Gambar 4.2 Data biner yang diambil dari PLD yang di proteksi

4.1.3 Pembacaan PLD kosong

Sebuah PLD yang masih kosong dicoba dilakukan pembacaan dengan programmer, hasil yang didapatkan seperti terlihat dalam gambar 4.3. Dari gambar 4.2 dan 4.3 dapat dilihat bahwa hasil pembacaan dari kedua PLD menunjukkan hasil yang sama, padahal yang satu masih kosong sedang yang lain berisi program tetapi diproteksi.

Dalam hal ini sebuah cara lain untuk mendapatkan data biner dari PLD yang diproteksi harus didapat dengan cara lain, salah satunya dengan bantuan simulator.

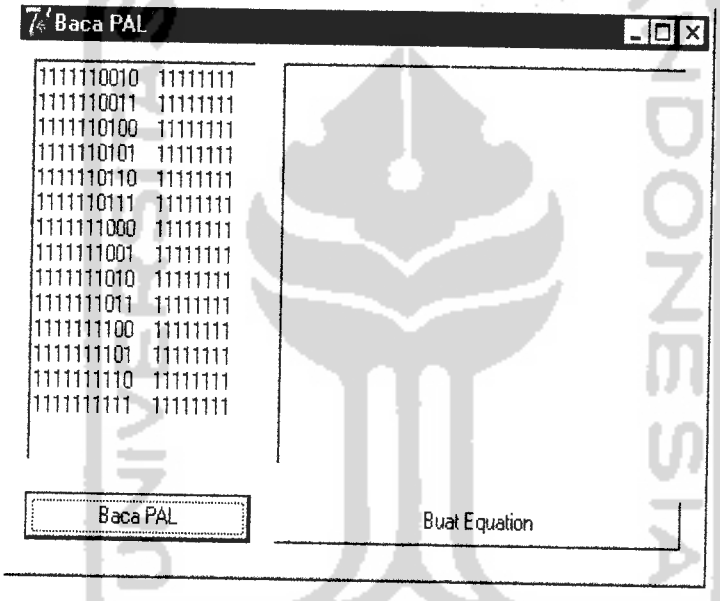


Gambar 4.3 Data dari PLD kosong

4.2 Pemrosesan PLD dengan simulator.

4.2.1 Pembacaan PLD kosong dengan simulator

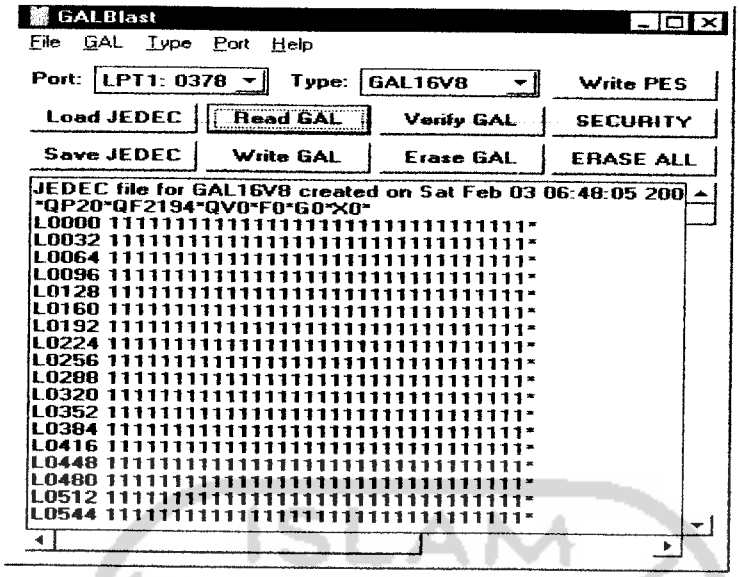
Untuk mengetahui data biner yang bisa diambil dari PLD menggunakan simulator terlebih dahulu akan dilakukan proses pembacaan dari PLD kosong, hasil dari pembacaan ini merupakan suatu dasar pembanding antara PLD yang berisi data biner maupun yang belum. Hasil pengujian ini ditunjukkan dalam gambar 4.4.



Gambar 4.4 Hasil pembacaan PLD kosong dengan simulator

4.2.2 Pembacaan PLD yang terproteksi dengan simulator

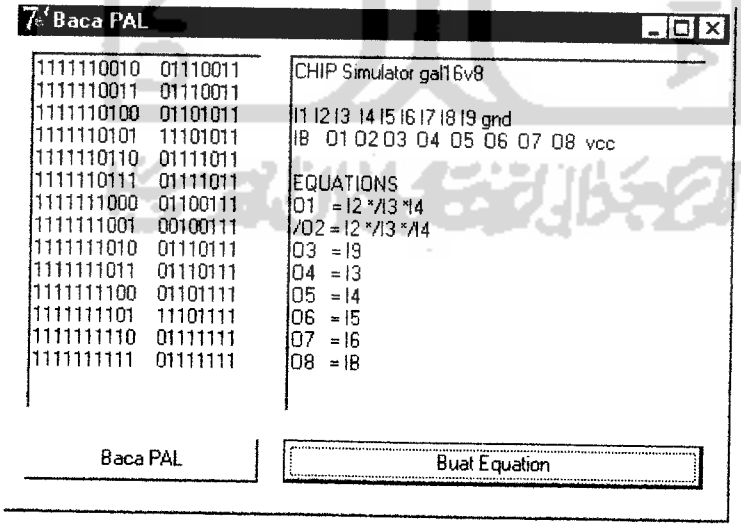
Gambar 4.5 diperlihatkan bahwa suatu PLD yang telah diprogram dan diproteksi dapat diambil data-data biner yang ada di dalamnya



Gambar 4.5 Data biner yang didapat simulator dari PLD terproteksi

4.2.3 Pembuatan persamaan PLD dengan simulator

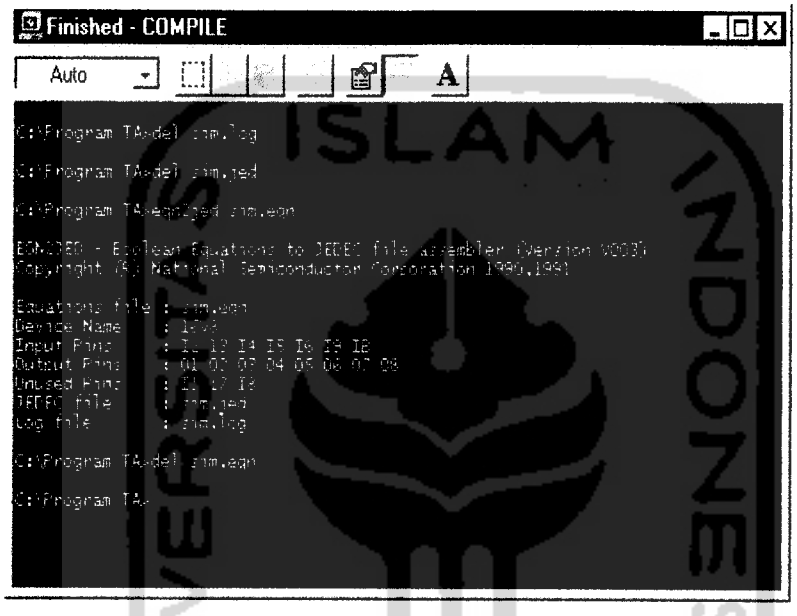
Setelah didapat data biner dari PLD seperti Gambar 4.5, proses selanjutnya dilakukan pembuatan persamaan hubungan antara masukkan dan keluaran dari PLD, hasil yang didapat seperti pada Gambar 4.6.



Gambar 4.6 Pembuatan persamaan PLD dengan simulator

4.2.4 Kompilasi persamaan PLD dengan kompiler PLD

File yang berisi data persamaan PLD selanjutnya dilakukan proses kompilasi dengan menggunakan kompiler PLD dari National semiconductor yang bernama OPAL, hasil dari proses kompilasi tersebut dapat dilihat pada Gambar 4.7 berikut:

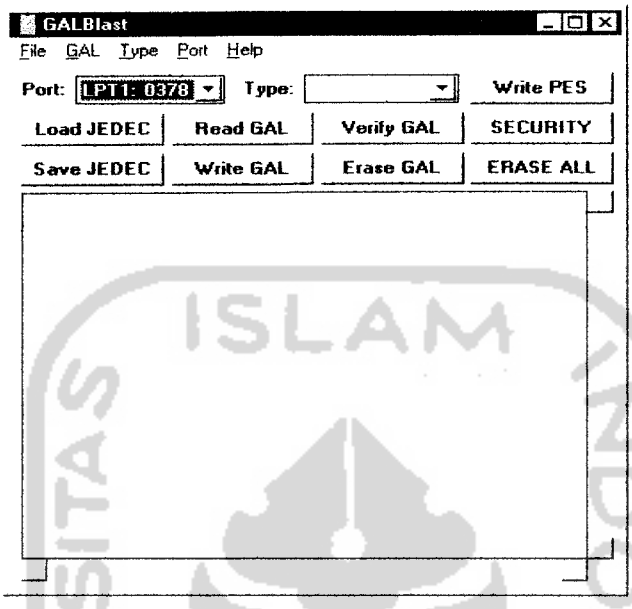


Gambar 4.7 Kompilasi persamaan PLD

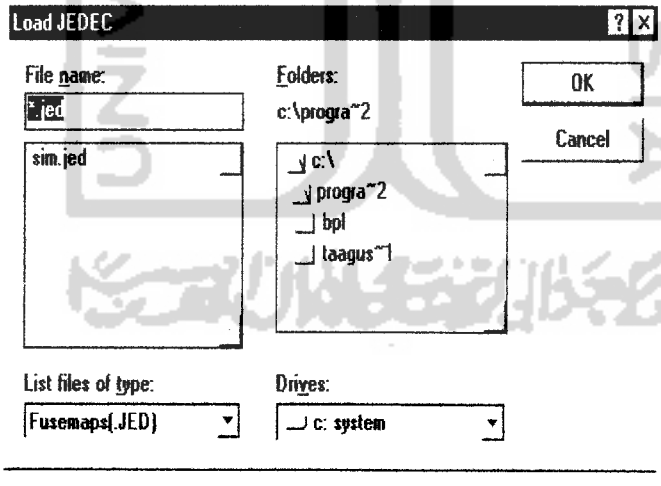
4.2.5 Proses pemrograman PLD

Agar PLD dapat dipakai terlebih dahulu dilakukan proses pemrograman dengan suatu programmer, dalam tugas akhir ini digunakan programmer yang sudah jadi, sehingga dalam hal ini tidak akan diulas bagaimana kerja dari sistem programmer PLD.

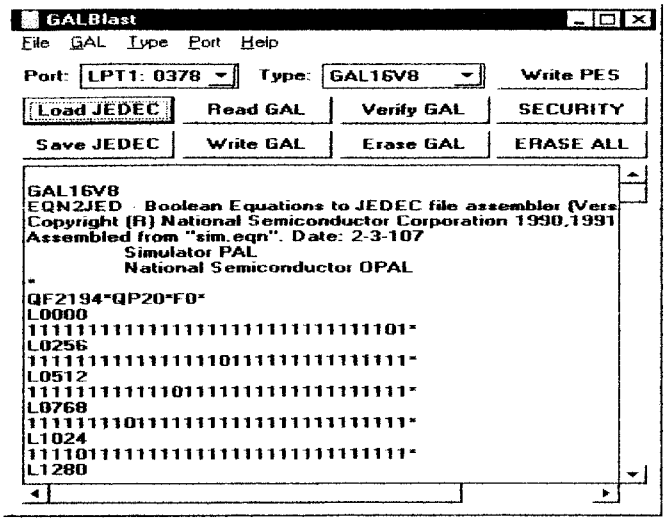
Hasil pemrograman PLD dapat dilihat dari gambar-gambar berikut :



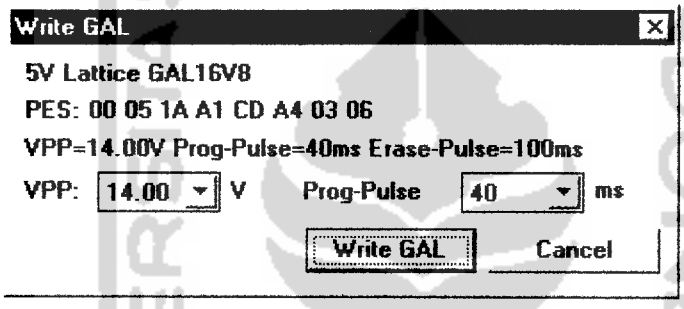
Gambar. 4.8 Programmer PLD yang dijalankan dalam keadaan mula-mula.



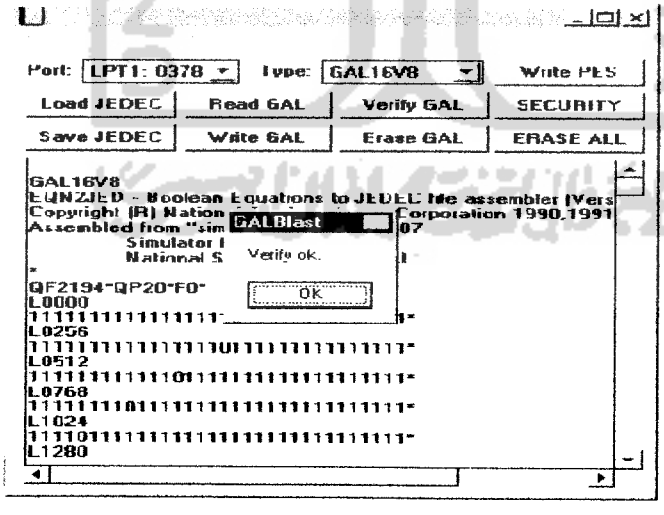
Gambar. 4.9 Proses pemilihan file JEDEC yang akan dimasukkan ke PLD dari programmer



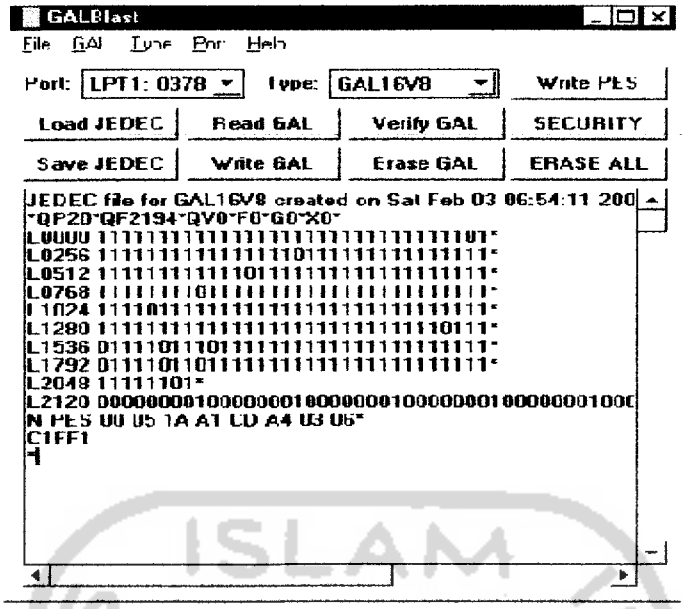
Gambar. 4.10 File JEDEC yang telah diload ke buffer programmer



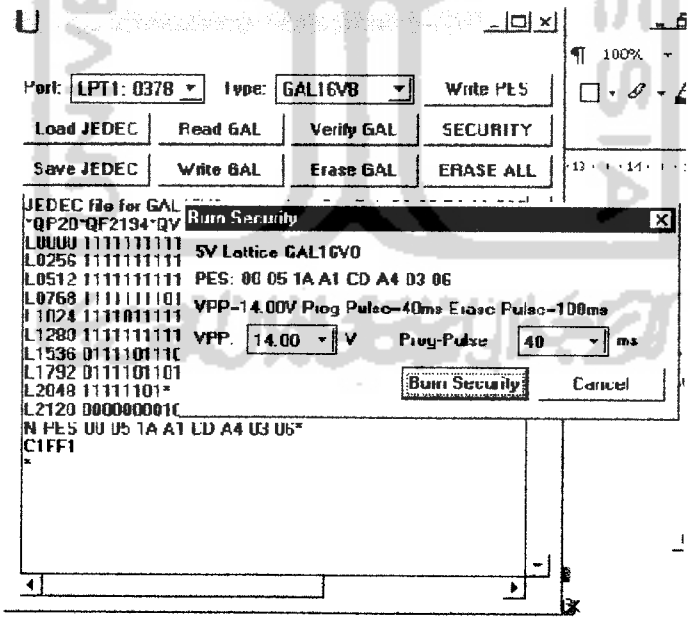
Gambar. 4.11 Proses penulisan PLD



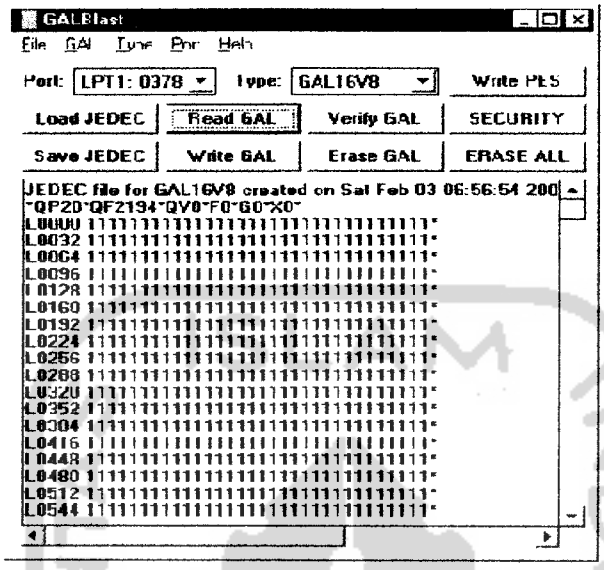
Gambar 4.12 Validasi dari penulisan



Gambar 4.13 Pembacaan kembali PLD yang sudah diprogram

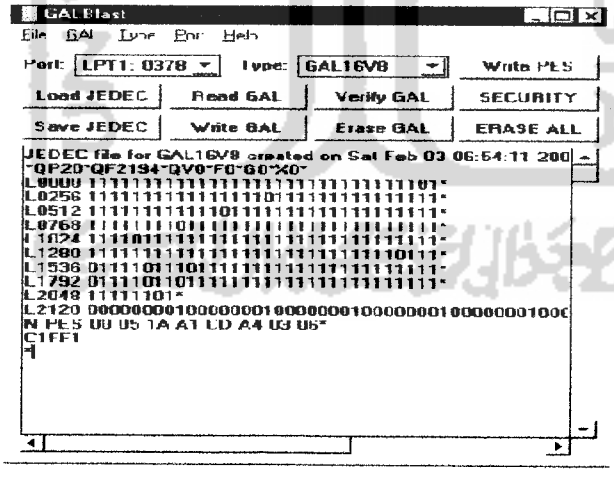


Gambar 4.14 Proses proteksi PLD



Gambar 4.15 Pembacaan kembali PLD setelah mengalami proteksi

4.2.6 Pembacaan kembali PLD hasil sintesa



Gambar 4.16 Pembacaan kembali PLD hasil simulasi

Gambar 4.16 memperlihatkan hasil pembacaan kembali PLD, hasil yang ditunjukkan memperlihatkan kesamaan terhadap PLD aslinya.

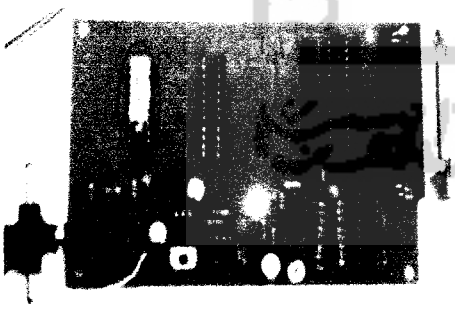
4.3 Pengujian dengan alat wartel

Gambar. 4.17 diperlihatkan bahwa PAL yang didapat dari proses simulasi dapat bekerja dengan benar pada alat.

Gambar 4.17 Pengujian dengan alat wartel

4.4 Analisa

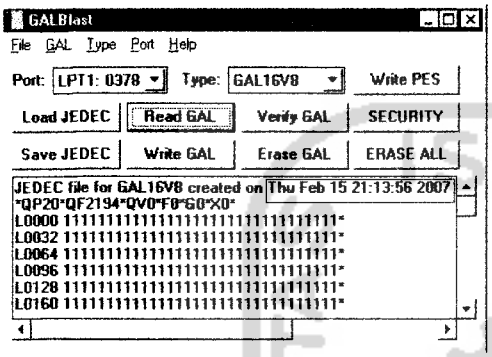
Setelah dilakukan pengujian maka dilakukan penganalisaan berdasarkan gambar-gambar berikut :



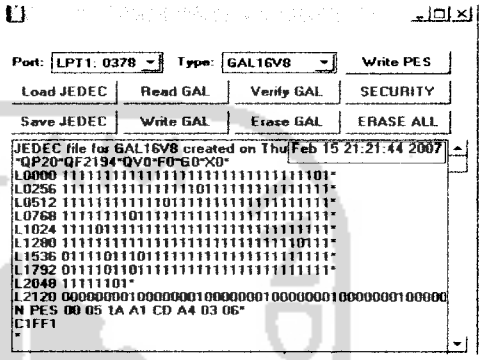
Gambar 4.18 PLD terprogram di Programmer

Gambar 4.19 PLD kosong yang sudah disintesa

Berdasarkan Gambar 4.18 dijelaskan bahwa PLD terprogram terpasang di progammer dan pada Gambar 4.19 PLD yang telah disintesakan akan dibaca kembali di progammer, maka hasilnya dapat dilihat pada Gambar 4.20 dan Gambar 4.21.



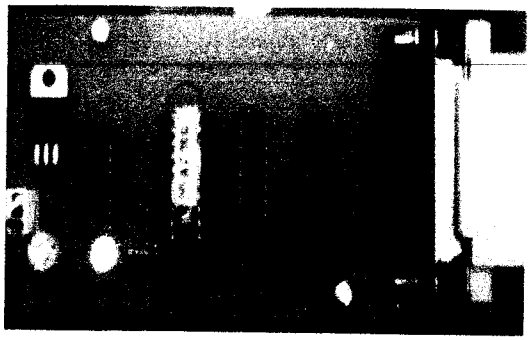
Gambar 4.20 pembacaan PLD terprogram Pada Programmer



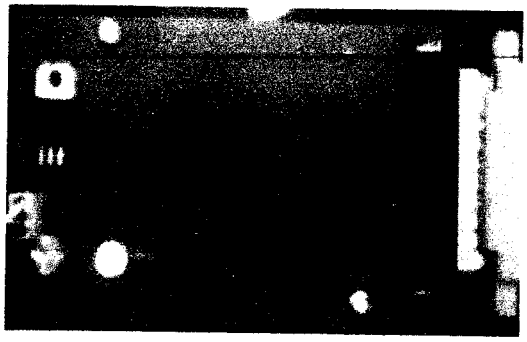
Gambar 4.21 pembacaan kembali PLD yang telah disintesa

Dapat dilihat pada Gambar 4.20 bahwa ternyata PLD yang terprogram pada saat dibaca pada programmer hanya mengeluarkan kombinasi biner 1, dikarenakan PLD tersebut telah terproteksi sehingga hanya mengeluarkan kombinasi biner 1, sedangkan pada Gambar 4.21 PLD yang telah disintesakan mengeluarkan kombinasi biner yang berbeda dengan gambar 4.20, hal itu disebabkan PLD tersebut belum diproteksi.

Pada bagian ini juga dijelaskan pembacaan PLD terprogram dan PLD yang telah disintesa pada alat Simulator, dapat dilihat pada Gambar 4.22 dan 4.23.

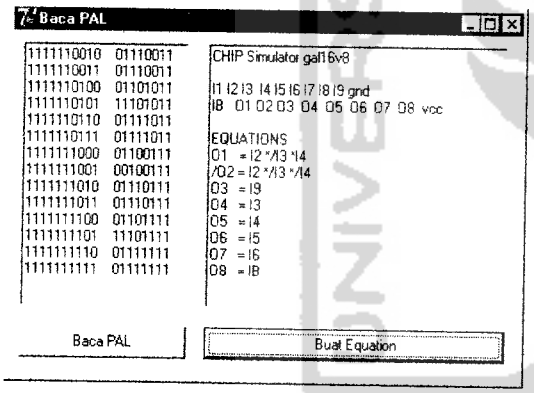


Gambar 4.22 PLD terprogram pada Simulator

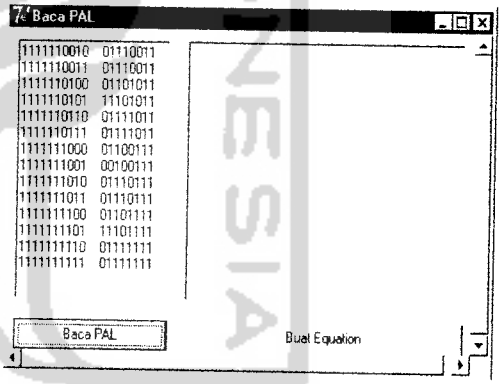


Gambar 4.23 PLD yang telah disintesa pada Simulator

Pada gambar 4.22 dan gambar 4.23 PLD yang telah terprogram maupun yang telah disintesa terpasang di Simulator, maka dapat dilihat hasil pembacaannya pada Gambar 4.24 serta Gambar 4.25.



Gambar 4.24 Hasil pembacaan PLD Terprogram di Simulator



Gambar 4.25 hasil pembacaan kembali PLD yang telah disintesa di Simulator

Dari Gambar 4.24 hasil analisa dan *Equation* telah didapatkan dari PLD yang terprogram, sedangkan pada Gambar 4.25 PLD hasil sintesa dibaca kembali dan didapatkan hasil yang sama dengan PLD yang terprogram dan siap untuk didapatkan *Equation*nya.

BAB V

PENUTUP

5.1 KESIMPULAN

Setelah selesainya uraian pada tahap-tahap pembuatan, pengujian dan pembahasan sistem pada tugas akhir ini, maka dapat dibuat kesimpulan sebagai berikut:

1. Penggunaan PLD memiliki beberapa keuntungan diantaranya harganya tidak terlalu mahal dan fleksibilitasnya tinggi, karena dapat diprogram, sehingga dapat memudahkan desain, selain itu program yang sudah ditanam dapat diproteksi sehingga orang lain tidak dapat menyalinnya.
2. Proses sintesa dilakukan dengan memberikan kombinasi biner 10 bit pada kaki - kaki input PLD mulai dari kondisi *Low* sampai *High*, kemudian dilihat pengaruh output terhadap input. Setelah didapat datanya kemudian disusun kembali menjadi *Equation*.
3. Setelah dilakukan pengujian ternyata didapatkan bahwa PLD yang belum terpogram (kosong) dan PLD yang telah diprogram tetapi di *secure* menunjukkan kombinasi biner yang sama yaitu FF. Hal ini disebabkan karena standarisasi PLD yang belum terprogram adalah FF dan yang telah terprogram tetapi di *secure* akan terlihat diprogramer seperti kosong guna melindungi *Equation* didalamnya.
4. Digunakan 2 buah alat uji untuk membuktikan bahwa alat Simulator dapat mensintesa program PLD dengan berbagai sistem kerja yang berbeda.

5.2 Saran

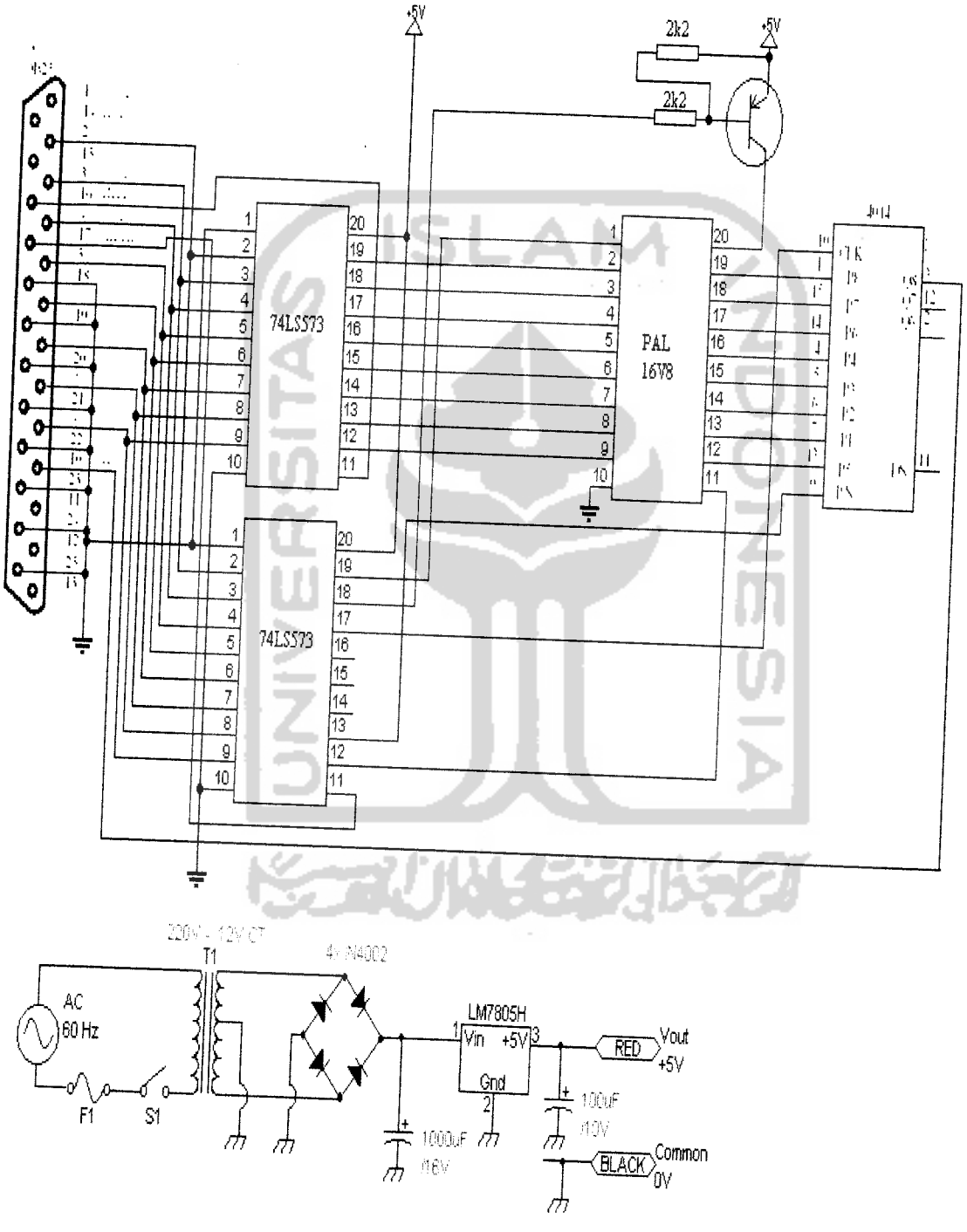
Untuk pengembangan-pengembangan selanjutnya penulis menyarankan:

1. Dalam merancang *hardware*,alat ukur dan perangkat lain yang digunakan harus dipilih yang mempunyai kualitas bagus untuk memperkecil *error*.
2. Pada perancangan selanjutnya alat simulator dan programmer dapat disatukan dalam satu PCB untuk penghematan biaya.
3. Diharapkan dalam pengembangan selanjutnya tidak lagi menggunakan pararel *port* melainkan menggunakan USB *port*.

DAFTAR PUSTAKA

1. A.E Fitzgerald, David E. Higginbotham, Arvin Grabel, 1981. *Dasar-dasar Rangkaian Elektronika*. Alih Bahasa: Pantur Silaban. Jakarta : Erlangga.
2. DigiWare, *Pengetahuan Komponen Pasif II*,(On-line) at http:// www.DigiWare.com
3. *Edminister. A joseph,* Rangkaian Listrik. Jakarta : seri buku Schaum.
4. Harris Semiconductor, CMOS 8-*Stage Static Shift Register*, CD4014B, CD4021B *Types* (On-line) at http:// www.google.com/Data Sheet CD4014B, CD4021B.
5. Kadir,Abdul, 2001. *Dasar Pemograman Delphi 6.0.* yogyakarta : Andi.
6. Lattice Semikonduktor Corporation, *Gal 16v8 High Performance CMOS PLD (on-line) Available at http://* www.Google.com/Data Sheet GAL 16v8.
7. Muchlas, 2005. *Rangkaian Digital*. Yogyakarta : Gaya Media.
8. Muhsin Muhammad, 2004. *Elektronika Digital*. Yogyakarta: Andi.
9. National Semikonduktor, *LM7805 Series Voltage Regulator*, http:// www.google.com/National Semiconductor/Data Sheet Regulator LM7805.
10. Suryatmo F, 1997, *Teknik Digital.* Jakarta : Bumi Aksara.
11. Wikipedia, FreeEncyclopedia, *Transistor*,http:// www.ThefreeEncyclopedia.com.

GAMBAR RANGKAIAN SCHEMATIC SIMULATOR



LISTING PROGRAM DELPHI

```
unit pal_sim;

interface

uses
  Windows, Messages,
  SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls;

type
  TForm1 = class(TForm)
    Button11: TButton;
    Memo1: TMemo;
    Button1: TButton;
    Memo2: TMemo;
    procedure Button1Click(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    procedure Button11Click(Sender: TObject);
  private
    { Private declarations }
  public
```

```
    { Public declarations }
  end;

var
  Form1: TForm1;

implementation

{$R *.dfm}
procedure PortOut(Port : Word; Data : Byte); stdcall;
external 'io.dll';
function PortIn(Port:Word):Byte; stdcall; external
'io.dll';
var
  dout:integer;
  d37A:integer;
  InValue : Byte;
  d573_1,d573_2:byte;
  dstring:string;
  kombinasi:string;
  t1:string;
  k:integer;
  SomeTxtFile : TextFile;
  FileEqn : TextFile;
```

```
      a1:string[1];
      buffer,biner,biner1,biner2:string;
      i:integer;
      c1:array[1..8]of integer;
      c0:array[1..8]of integer;
      max1:array[1..8]of integer;
      max0:array[1..8]of integer;
      temp:integer;


 procedure latch1;
 var
    dlatch:byte;
 begin
    dlatch:=11;
    PortOut(890,dlatch);
    dlatch:=10;
    PortOut(890,dlatch);
    dlatch:=11;
    PortOut(890,dlatch);
end;

procedure latch2;
var
```

```
    dlatch:byte;
begin
    dlatch:=11;
    PortOut(890,dlatch);
    dlatch:=3;
    PortOut(890,dlatch);
    dlatch:=11;
    PortOut(890,dlatch);
end;

procedure data_out;
begin
   d573_1:=dout;
   PortOut(888,d573_1);
   latch1;
   PortOut(888,0);
end;
procedure Max0_Max1;
begin
   if a1='0' then
   begin
      c0[i]:=c0[i]+1;
      if c1[i]>=max1[i] then max1[i]:=c1[i];
      c1[i]:=0;
```

```
      end
      else
      begin
         c1[i]:=c1[i]+1;
         if c0[i]>=max0[i] then max0[i]:=c0[i];
         c0[i]:=0;
      end;
   end;

procedure Max0_Max2;
var
   selisih,jumlah,besar,kecil:integer;
begin
{    besar:=max0[i];
   kecil:=max1[i];
   if max1[i]>besar then
   begin
      besar:=max1[i];
      kecil:=max0[i];
  end;
  selisih:=besar-kecil;
  if selisih<2 then
  begin
     max0[i]:=besar;
```

```
   max1[i]:=besar;
   if max0[i]<4 then
      max0[i]:=4
   else
   if max0[i]<=8 then
      max0[i]:=8
   else
   if max0[i]<=16 then
      max0[i]:=16
   else
   if max0[i]<=32 then
      max0[i]:=32
   else
   if max0[i]<=64 then
      max0[i]:=64
   else
   if max0[i]<=128 then
      max0[i]:=128
  else
  if max0[i]<=256 then
     max0[i]:=256
 else
 if max0[i]<=512 then
    max0[i]:=512
```

```
        else
        if max0[i]<=1024 then
           max0[i]:=1024;
        max1[i]:=max0[i];
    end;}
end;

Procedure Max0_eq_Max1;
begin
    if temp=1 then
       buffer:='I2'
    else
       if temp=2 then
          buffer:='I3'
       else
          if temp=4 then
             buffer:='I4'
          else
             if temp=8 then
                buffer:='I5'
             else
                if temp=16 then
                   buffer:='I6'
                else
```

```
                    if temp=32 then
                       buffer:='I7'
                    else
                       if temp=64 then
                          buffer:='I8'
                       else
                          if temp=128 then
                             buffer:='I9'
                          else
                             if temp=256 then
                                buffer:='I1'
                             else
                                if temp=512 then
                                   buffer:='IB';
end;


procedure banding;
var
   j5,i5:integer;
   a3,a4:string[1];
   a5:string[3];
begin
   buffer:='';
   j5:=0;
```

```
for i5:=1 to 10 do
begin
   a3:=biner1[i5];
   a4:=biner2[i5];
   if ((a3='1') or (a4='1')) then
   begin
      if j5=0 then
         j5:=1
      else
         buffer:=buffer+'*';
      if i5=1 then
         a5:='I2 '
      else
      if i5=2 then
         a5:='I3 '
      else
      if i5=3 then
         a5:='I4 '
      else
      if i5=4 then
         a5:='I5 '
      else
      if i5=5 then
         a5:='I6 '
```

```
            else
            if i5=6 then
               a5:='I7 '
            else
            if i5=7 then
               a5:='I8 '
            else
            if i5=8 then
               a5:='I9 '
            else
            if i5=9 then
               a5:='IB'
            else
            if i5=10 then
               a5:='I1 ';
            if a3=a4 then
               buffer:=buffer+a5
            else
               buffer:=buffer+'/'+a5;
      end;
   end;
end;

procedure dec_to_bin;
```

```
var
   bagi,sisa,terus,i3,i4:integer;
   binerx,a2:string;
begin
   biner:='';
   terus:=1;
   i3:=0;
   while (terus=1) do
   begin
      sisa:=temp mod 2;
      bagi:=temp div 2;
      i3:=i3+1;
      biner:=biner+inttostr(sisa);
      if bagi>=2 then
      begin
         temp:=bagi;
         terus:=1;
      end
      else
         terus:=0;
   end;
   i3:=i3+1;
   biner:=biner+inttostr(bagi);
   for i4:=i3 to 9 do
```

```
        biner:=biner+'0';
end;
procedure max0_ne_max1;
var
   i2,awal:integer;
   harga:string[1];
begin
   awal:=0;
   i2:=0;
   Reset(SomeTxtFile) ;
   ReadLn(SomeTxtFile, buffer) ;
   while not EOF(SomeTxtFile) do
   begin
      ReadLn(SomeTxtFile, buffer) ;
      i2:=i2+1;
      awal:=awal+1;
      if i=1 then
         a1:=buffer[15]
      else
      if i=2 then
         a1:=buffer[16]
      else
      if i=3 then
         a1:=buffer[17]
```

```
    else
    if i=4 then
       a1:=buffer[18]
    else
    if i=5 then
       a1:=buffer[19]
    else
    if i=6 then
       a1:=buffer[20]
    else
    if i=7 then
       a1:=buffer[21]
    else
    if i=8 then
       a1:=buffer[22];
    if i2=1 then
       harga:=a1
    else
       if a1<>harga then
          break;
end;
awal:=awal-1;
temp:=awal;
dec_to_bin;
```

```
    biner1:=biner;
    if max0[i]>max1[i] then
       temp:=max0[i]
    else
       temp:=max1[i];
    dec_to_bin;
    biner2:=biner;
    banding;
end;
procedure TForm1.Button1Click(Sender: TObject);
var
   ket:array[1..8]of string[4];
   i1:integer;
begin
   AssignFile(FileEqn, 'Simulator.eqn') ;
   Rewrite(FileEqn) ;
   memo2.Lines.Clear;
   memo2.Lines.Add('     Simulator PAL');
   writeln(FileEqn,'     Simulator PAL');
   memo2.Lines.Add('     National Semiconductor OPAL');
   writeln(FileEqn,'     National Semiconductor OPAL');
   memo2.Lines.Add('');
   writeln(FileEqn,'');
   memo2.Lines.Add('CHIP Simulator gal16v8');
```

```
    writeln(FileEqn,'CHIP Simulator gal16v8');
    memo2.Lines.Add('');
    writeln(FileEqn,'');
    memo2.Lines.Add('I1 I2 I3  I4 I5 I6 I7 I8 I9 gnd');
    writeln(FileEqn,'I1 I2 I3  I4 I5 I6 I7 I8 I9 gnd');
    memo2.Lines.Add('IB   O1 O2 O3  O4  O5  O6  O7  O8
vcc');
    writeln(FileEqn,'IB   O1 O2 O3  O4  O5  O6  O7  O8
vcc');
    memo2.Lines.Add('');
    writeln(FileEqn,'');
    memo2.Lines.Add('EQUATIONS');
    writeln(FileEqn,'EQUATIONS');
    for i1:=1 to 8 do
   begin
       ket[i1]:='O'+inttostr(i1);
       c1[i1]:=0;
       c0[i1]:=0;
       max0[i1]:=0;
       max1[i1]:=0;

  end;
  AssignFile(SomeTxtFile, 'Simulator.dat') ;
  Reset(SomeTxtFile) ;
```

```
ReadLn(SomeTxtFile, buffer) ;
while not EOF(SomeTxtFile) do
begin
   ReadLn(SomeTxtFile, buffer) ;
   a1:=buffer[15];
   i:=1;
   max0_max1;
   a1:=buffer[16];
   i:=2;
   max0_max1;
   a1:=buffer[17];
   i:=3;
   max0_max1;
   a1:=buffer[18];
   i:=4;
   max0_max1;
   a1:=buffer[19];
   i:=5;
   max0_max1;
   a1:=buffer[20];
   i:=6;
   max0_max1;
   a1:=buffer[21];
   i:=7;
```

```
        max0_max1;
        a1:=buffer[22];
        i:=8;
        max0_max1;
    end;
    for i1:=1 to 8 do
    begin
        if max0[i1]=0 then
            if max1[i1]<1024 then
                max0[i1]:=max1[i1];
        if max1[i1]=0 then
            if max0[i1]<1024 then
                max1[i1]:=max0[i1];
end;
for i1:=1 to 8 do
begin
    i:=i1;
    max0_max2;
    if max0[i1]=max1[i1] then
    begin
        temp:=max0[i1];
        Max0_eq_Max1;
        buffer:=ket[i1]+'    = '+buffer;
        form1.memo2.lines.add(buffer);
```

```
            writeln(FileEqn,buffer);
          end
          else
          begin
             max0_ne_max1;
             if max0[i1]<max1[i1] then
                ket[i1]:='/'+ket[i1]
             else
                ket[i1]:=ket[i1]+'  ';
             buffer:=ket[i1]+' = '+buffer;
             memo2.lines.add(buffer);
             writeln(FileEqn,buffer);
          end;
       end;
       CloseFile(SomeTxtFile) ;
       CloseFile(FileEqn) ;
    end;

    procedure baca_data;
    var
       t1:string;
       i,j:integer;
    begin
    {    TForm1.label2.Caption:=inttostr(d573_1);}
```

```
    dstring:='';
    d573_2:=d573_2 or 64;
    PortOut(888,d573_2);
    latch2;
    PortOut(890,11);
    PortOut(890,15);
    d573_2:=d573_2 xor 64;
    for i:=1 to 8 do
    begin
       PortOut(888,d573_2);
       latch2;
       PortOut(890,11);
       InValue := PortIn(889);
{       label2.caption:=inttostr(invalue);}
       if invalue=63 then
          dstring:=dstring+'0'
       else
          dstring:=dstring+'1';
       PortOut(890,11);
       PortOut(890,15);
  end;
  t1:='';
  for i:=1 to 3 do
      t1:=t1+dstring[i];
```

```
      for i:=5 to 8 do
         t1:=t1+dstring[i];
      t1:=t1+dstring[4];
      dstring:='';
      for i:=1 to 8 do
      begin
         j:=9-i;
         dstring:=dstring+t1[j];
     end;
     PortOut(888,0);
{    label1.Caption := dstring;}
end;
procedure TForm1.FormCreate(Sender: TObject);
begin
     memo1.Lines.Clear;
     memo2.Lines.Clear;
     d37A:=11;
     PortOut(890,d37A);
     InValue := PortIn(889);
     d573_1:=0;
     PortOut(888,d573_1);
     latch1;
     d573_2:=2;
     PortOut(888,d573_2);
```

```
      latch2;
      dout:=0;
      PortOut(888,dout);
 end;

 procedure dec2bin;
 var
    ii : integer;
    bagi:integer;
    sisa:integer;
    hasil:integer;
    tt1:string;
    jj:integer;
begin
   t1:='';
   bagi:=k;
   for ii:=1 to 8 do
   begin
      sisa :=bagi mod 2;
      if sisa=0 then t1:=t1+'0' else t1:=t1+'1';
      hasil:=bagi div 2;
      bagi:=hasil;
  end;
  tt1:='';
```

```
      for ii:=1 to 8 do
      begin
          jj:=8-ii+1;
          tt1:=tt1+t1[jj];
      end;
      t1:=tt1;
  end;

  procedure ulangan;
  var i:integer;
      j:integer;
      dstring:string;
  begin
     for k:=0 to 255 do
     begin
       dout:= k;
       data_out;
       for j:=1 to 1000 do
       begin
       end;
{       label2.Caption:=inttostr(d573_1);}
      dstring:='';
      d573_2:=d573_2 or 64;
      PortOut(888,d573_2);
```

```
        latch2;
        PortOut(890,11);
        PortOut(890,15);
        d573_2:=d573_2 xor 64;
        for i:=1 to 8 do
        begin
          PortOut(888,d573_2);
          latch2;
          PortOut(890,11);
          InValue := PortIn(889);
{          label2.caption:=inttostr(invalue);}
         if invalue=63 then
           dstring:=dstring+'0';
         if invalue=127 then
           dstring:=dstring+'1';
         PortOut(890,11);
         PortOut(890,15);
    end;
    t1:='';
    for i:=1 to 3 do
       t1:=t1+dstring[i];
    for i:=5 to 8 do
      t1:=t1+dstring[i];
    t1:=t1+dstring[4];
```

```
      dstring:='';
      for i:=1 to 8 do
      begin
        j:=9-i;
        dstring:=dstring+t1[j];
      end;
  {    label1.Caption := dstring;}
  {    t1:=inttostr(k);}
     dec2bin;
     dstring:=kombinasi+t1+'    '+dstring;
     form1.memo1.Lines.add(dstring);
     writeLn(SomeTxtFile, dstring) ;
   end;
end;


procedure TForm1.Button11Click(Sender: TObject);
begin
   AssignFile(SomeTxtFile, 'Simulator.dat') ;
   Rewrite(SomeTxtFile) ;
   dout:=2;
   d573_2:=d573_2 xor dout;
   PortOut(888,d573_2);
   latch2;
```

```
PortOut(888,0);
memo1.Lines.Clear;
memo1.lines.Add('      Input                    Output ');
memo1.lines.Add('1B98765432    12345678');
buffer:='1B98765432';
writeLn(SomeTxtFile, buffer) ;
kombinasi:='00';
d573_2:=d573_2 and 126;
PortOut(888,d573_2);
latch2;
ulangan;
kombinasi:='01';
d573_2:=d573_2 and 126;
PortOut(888,d573_2);
latch2;
d573_2:=d573_2 or 1;
PortOut(888,d573_2);
latch2;
ulangan;
kombinasi:='10';
d573_2:=d573_2 and 126;
PortOut(888,d573_2);
latch2;
d573_2:=d573_2 or 128;
```

```
      PortOut(888,d573_2);
      latch2;
      ulangan;
      kombinasi:='11';
      d573_2:=d573_2 or 1;
      PortOut(888,d573_2);
      latch2;
      ulangan;
      dout:=2;
      d573_2:=d573_2 xor dout;
      PortOut(888,d573_2);
      latch2;
      PortOut(888,0);
      CloseFile(SomeTxtFile) ;
      button1.Enabled:=true;
end;
```