

310000 10 13027

**PROTOTIPE ROBOT BERKAKI EMPAT
BERKEMAMPUAN MERAYAP
(SISTEM KENDALI)**

PROYEK AKHIR

Disusun Oleh :

I GUSTI MADE ARTHA YOGAISWARA

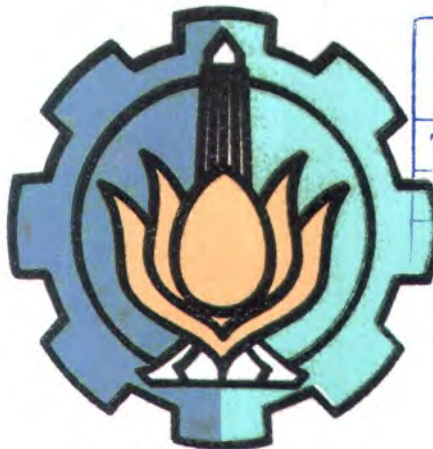
NRP. 7195 030 013

RSE
629.892

Yog

P-1

1998



**PERPUSTAKAAN
ITS**

Tgl. Terima	11-11-00
Terima Dari	It
No. Agenda Prp.	9107

**JURUSAN TEKNIK ELEKTRO
POLITEKNIK ELEKTRONIKA NEGERI SURABAYA
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA
1998**

PROTOTIPE ROBOT BERKAKI EMPAT BERKEMAMPUAN MERAYAP (SISTEM KENDALI)

PROYEK AKHIR

**Diajukan Sebagai Persyaratan
Penyelesaian Studi Program Pendidikan Diploma III
Politeknik**

**Surabaya, Agustus 1998
Mengetahui / Menyetujui**

Dosen Penguji I



DR. Ir. Mauridhi Hery P.

NIP. 131 569 364

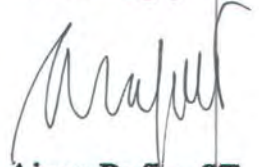
Dosen Penguji II



Afrida Helen, ST

NIP. 132 170 593

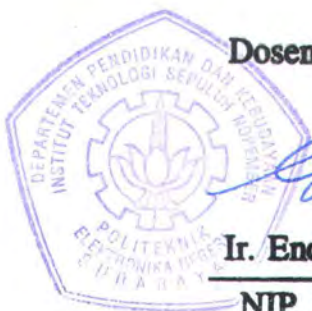
Dosen Penguji III



Ainur Rofiq, ST

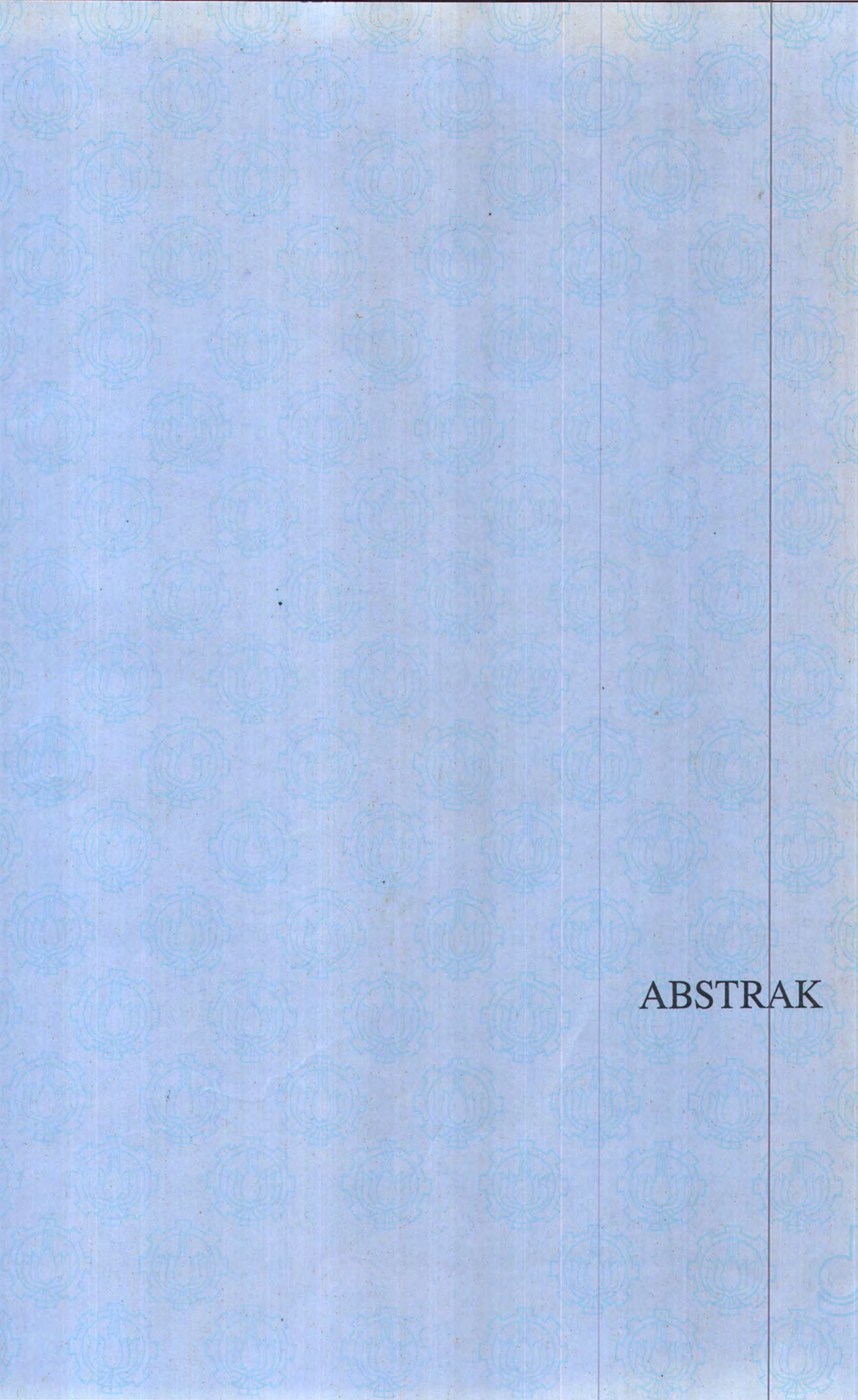
NIP. 131 859 915

Dosen Pembimbing



Ir. Endra Pitowarno

NIP. 131 651 450



ABSTRAK

ABSTRAK

Robot Berkaki Empat Berkemampuan Merayap merupakan robot generasi I yang dalam pergerakannya menggunakan kaki dan memiliki kemampuan untuk merayap baik pada bidang vertikal maupun horizontal.

Robot ini dirancang untuk bergerak dengan kaki yang dilengkapi karet penghisap agar dapat menempel di permukaan vertikal seperti kaca, whiteboard, dan acrylic atau bahkan pada dinding. Hal utama pada perancangan adalah pada bagian kaki dari robot yang mampu menempel pada bidang vertikal. Pergerakannya akan memperlihatkan kemampuan untuk mengangkat kaki secara diagonal dan merayap, pada saat kaki yang lainnya menahan beban ataupun tubuh robot. Keterbatasan dari prototype ini adalah hanya mampu merayap pada bidang yang datar dan tidak berongga.

Arah yang ingin dicapai dari robot ini adalah untuk melakukan pekerjaan yang beresiko tinggi yang berkaitan dengan ketinggian misalnya pada saat melakukan pembersihan kaca jendela pada gedung bertingkat banyak.

KATA PENGANTAR

KATA PENGANTAR

Dengan mengucapkan puji syukur kepada Sang Hyang Widhi Wasa, yang telah memberikan berkah dan tuntunannya sehingga terealisasi proyek akhir ini dengan judul :

PROTOTIPE ROBOT BERKAKI EMPAT BERKEMAMPUAN MERAYAP

Dengan Sub Judul :

SISTEM KENDALI

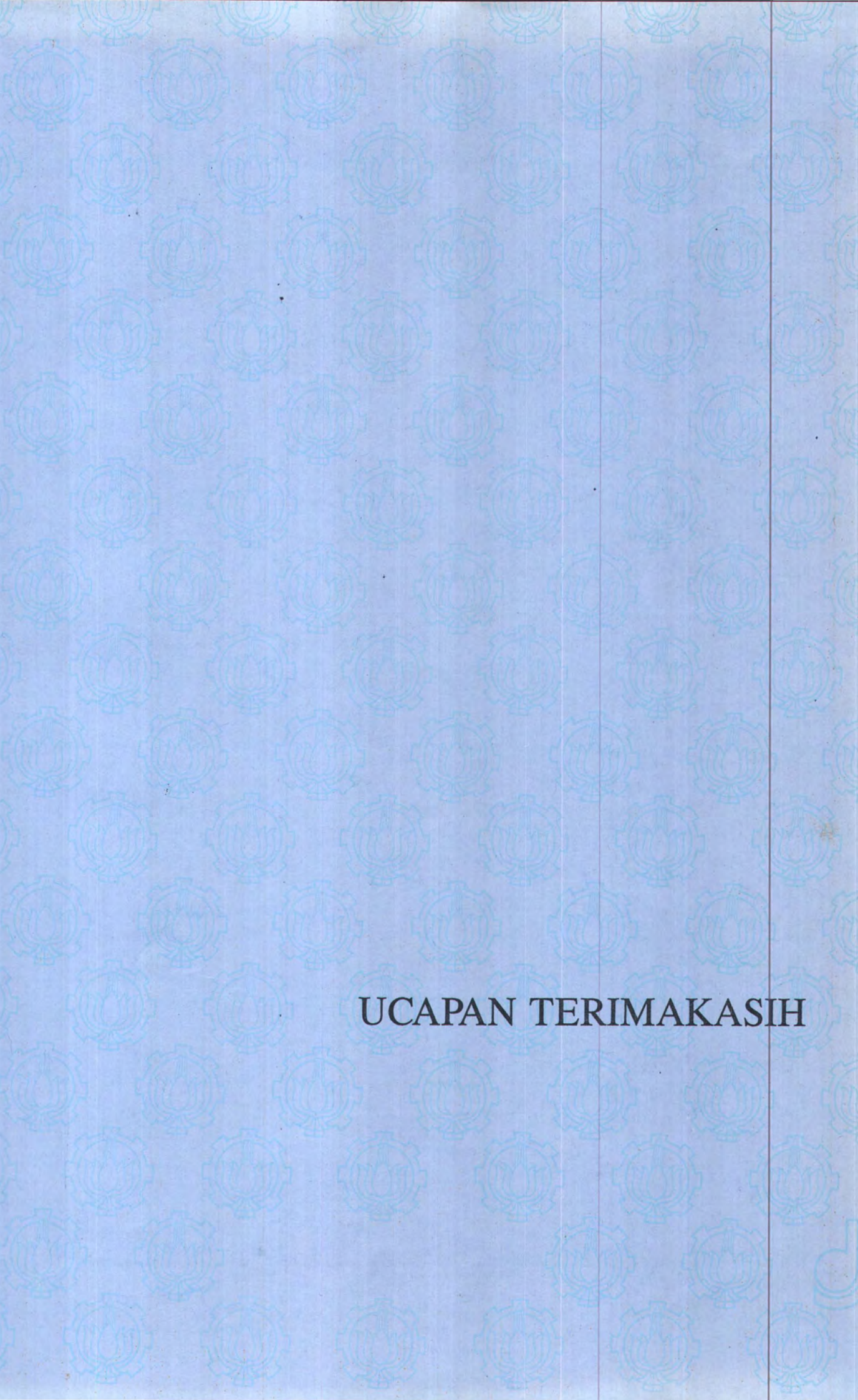
Buku laporan ini ditulis sebagai prasyarat penyelesaian studi program pendidikan Diploma III Politeknik Elektronika Surabaya. Materi yang disajikan adalah mengenai sistem kendali secara keseluruhan beserta rangkaian elektronik yang bersangkutan perihal robot berkaki empat berkemampuan merayap ini.

Dengan terealisasinya buku ini, kami berharap semoga dapat bermanfaat dan diterima sebagai sumbangan dalam rangka turut serta memikul tanggung jawab pembangunan bagi bangsa dan negara.

Penulis menyadari bahwa buku laporan ini masih jauh dari sempurna, oleh karena itu penulis membuka tangan selebar-lebarnya untuk saran dan kritik yang bersifat membangun.

Surabaya, Juli 1998

Penulis



UCAPAN TERIMAKASIH

UCAPAN TERIMA KASIH

Kami menghaturkan Puji Syukur kepada Sang Hyang Widhi Wasa, yang telah memberikan berkah dan tuntunannya hingga terselesaikannya tugas akhir tepat pada waktunya.

Keberhasilan ini juga berkat perhatian, dukungan dan bantuan yang telah diberikan dari berbagai pihak kepada kami. Maka dari itu kami tak lupa mengucapkan terima kasih kepada:

1. Bapak Dr. Ir. M.Nuh, DEA , selaku Direktur Politeknik Elektronika Negeri Surabaya ITS yang telah memberikan ijin pemakaian laboratorium dan fasilitas lain guna terselesaikannya proyek akhir ini.
2. Bapak Ir. Gigih Prabowo , selaku Kepala Jurusan Teknik Elektro.
3. Bapak Ir. Endra Pitowarno, selaku dosen pembimbing yang telah meluangkan waktunya untuk memberikan saran dan fasilitas yang kami perlukan.
4. Rekan kerja satu team, Rini Hendarto atas kerja samanya hingga terselesaikannya proyek akhir ini.
5. Seluruh rekan kerja satu lab. Atas bantuannya.
6. Rekan-rekan mahasiswa yang turut membantu terwujudnya proyek ini baik secara langsung maupun tidak langsung.
7. Seluruh staf dan karyawan Politeknik Elektronika Negeri Surabaya.
8. Ayah, Ibu ,kakak,serta Adikku yang tercinta yang selalu memberikan dorongan, semangat serta doanya.
9. Teman teristimewa yang selalu berada di sisiku pada saat sulit.

10. Semua pihak, yang dengan amat sangat penulis mohon maaf karena keterbatasan penulis untuk mengingat mereka yang telah membantu, baik secara langsung maupun tidak langsung, dalam menyelesaikan proyek akhir ini.

Surabaya, Juli 1998

Penulis

DAFTAR ISI

DAFTAR ISI

	Halaman
HALAMAN JUDUL	i
HALAMAN PENGESAHAN	ii
ABSTRAK	iii
KATA PENGANTAR	iv
UCAPAN TERIMA KASIH	v
DAFTAR ISI	vii
DAFTAR GAMBAR	xi
DAFTAR TABEL	xiii
BAB I PENDAHULUAN	1
1.1. LATAR BELAKANG	1
1.2. TUJUAN	2
1.3. PERMASALAHAN	2
1.4. PEMBATAHAN MASALAH	3
1.5. SISTEMATIKA PEMBAHASAN	3
BAB II TEORI PENUNJANG	5
2.1. PENGENALAN ROBOTIKA	5
2.2. MOBILE ROBOT	6
2.2.1. OPERATOR ORIENTED	6
2.2.2. SELF RUNNING	7

2.3. NON_MOBILE ROBOT	7
2.4. BAHASA PEMROGRAMAN ROBOT	8
2.4.1. BAHASA ASSEMBLY	8
2.4.2. PENGGUNAAN BAHASA ASSEMBLY	8
2.4.3. ASSEMBLER	9
2.4.4. PERNYATAAN BAHASA ASSEMBLY	9
2.4.4.1. DAERAH LABEL	10
2.4.4.2. DAERAH OPCODE	11
2.4.4.3. DAERAH OPERAND	11
2.4.4.4. DAERAH KOMENTAR	11
2.4.5. MACRO ASSEMBLER	12
2.4.6. MACROS	12
2.5. PENGERTIAN SISTEM MINIMUM	13
2.6. ARSITEKTUR MIKROPROSESOR Z80	16
2.6.1. REGISTER-REGISTER CPU Z80	24
2.6.2. TEKNIK-TEKNIK INTERRUPT Z80	25
2.6.2.1. INTERRUPT ENABLE_DISABLE	25
2.6.2.2. MODE-MODE INTERRUPT RESPON	26
2.7. PIRANTI MEMORI	28
2.7.1. EPROM	28
2.7.2. RAM	31
2.7.3. INTERFACING PIRANTI MEMORI DENGAN Z80	33
2.8. PIRANTI I/O (INPUT/OUTPUT)	35
2.8.1. PPI 8255	35
2.8.2. INTERFACING KE PPI 8255	40

2.9. INTERFACING PERANTI I/O KE CPU Z80	41
2.9.1. INTERFACING KE RELAY	42
2.9.2. INTERFACING KE DC MOTOR	43
2.9.3. INTERFACING KE SWITCH (on/off)	45
BAB III PERENCANAAN DAN PEMBUATAN	47
3.1. PERENCANAAN SISTEM MINIMUM Z80	48
3.1.1. RANGKAIAN OSILATOR DAN RESET	48
3.2. PERENCANAAN INTERFACING MEMORI KE CPU Z80	50
3.2.1. PERENCANAAN EPROM	52
3.2.2. PERENCANAAN RAM	53
3.3. PERENCANAAN INTERFACING DENGAN PPI 8255	54
3.4. PRINSIP KERJA SEMI OTOMATIS	55
3.5. MODE OPERATOR ORIENTED	55
3.6. OPEN LOOP CONTROLLER	57
3.7. ALGORITMA GERAKAN ROBOT	57
3.8. FLOWCHART	60
3.9. CUPLIKAN PROGRAM	62
BAB IV PENGUJIAN DAN ANALISA	66
4.1. TUJUAN	66
4.2. PENGUJIAN MODUL SISTEM MINIMUM Z80	66
4.3. PENGUJIAN SISTEM GERAK	72
4.4. PENGUJIAN PROGRAM KEMUDI	75
4.5. PENGUJIAN PEWAKTUAN	78

BAB V PENUTUP	81
5.1. KESIMPULAN	81
5.2. SARAN	82
 DAFTAR PUSTAKA	 83
LAMPIRAN A : DATA MEKANIK	84
LAMPIRAN B : LISTING PROGRAM	91
LAMPIRAN C : SISTEM KENDALI	102
LAMPIRAN D : FOTO GAMBAR	106
LAMPIRAN E : PETUNJUK MANUAL PENGOPERASIAN	109
LAMPIRAN F : DAFTAR RIWAYAT HIDUP	113

DAFTAR GAMBAR

DAFTAR GAMBAR

GAMBAR	Halaman
2.1. Blok Diagram Sistem Rangkaian Berbasis Mikroprosesor	14
2.2. Ilustrasi Blok Diagram Sistem yang dikontrol	14
2.3. Ilustrasi Blok Diagram Sistem Rangkaian Berbasis Mikroprosesor yang digabung dengan Sistem yang dikontrol	15
2.4. Arsitektur Z80	17
2.5. Konfigurasi Pin Z80 berdasarkan fungsi Input-Output	18
2.6. Konfigurasi Pin Z80 berdasarkan Tata Letak dalam DIL 40 Pin	19
2.7. Register-Register Z80	24
2.8. Kompabilitas fungsi pin berdasarkan tipe EPROM	29
2.9. Konfigurasi fungsi pin berdasarkan tipe RAM	32
2.10. Pemetaan alamat dari RAM dan EPROM	34
2.11. Konfigurasi pin PPI 8255	37
2.12. Format control word PPI 8255	39
2.13. Formasi Set/Reset PPI 8255	40
2.14. Rangkaian Dekoder dalam penentuan pengalamatan PPI 8255	41
2.15. Rangkaian Driver Relay	42
2.16. Rangkaian Driver Relay Motor DC	43
2.17. Rangkaian Driver Pembalik Arah motor DC	44
2.18. Interfacing PPI 8255 ke Switch On/Off	45
3.1. Blok Diagram dari Robot Merayap I	48

3.2. Rangkaian Osilator Kristal 4 MHz	49
3.3. Rangkaian Reset Power On	50
3.4. Pemetaan Alamat EPROM 2764 dan RAM 6264	50
3.5. Hubungan CPU Z80 dengan Sistem Memori	51
3.6. Rangkaian Dekoder untuk pemilihan alamat kerja PPI 8255	54
3.7. Rangkaian remote Control	56
3.8. Flowchart Gerakan Robot	61
3.9. Hubungan Sistem Minimum menggunakan Z80, memori 6264 & 2764, dua buah PPI serta peripheral penunjang lainnya	65
4.1. Rangkaian Pengujian Port PPI dan Sistem Minimum	67
4.2. Rangkaian pengujian Driver Relay dan Limit Switch	73

DAFTAR TABEL

DAFTAR TABEL

TABEL	Halaman
2.1. Ringkasan pengaruh instruksi terhadap IFF1 dan IFF2	26
2.2. Operasi dasar PPI 8255	37
3.1. Kapasitas memori EPROM 2764	52
3.2. Tabel kebenaran saat CPU menghubungi EPROM	53
3.3. Kapasitas memori RAM 6264	53
3.4. Tabel kebenaran saat CPU menghubungi RAM	54
4.1. Pewaktuan dari masing-masing pergerakan	73
4.2. Pengukuran arus masing-masing motor	74
4.3. Tabel perbandingan pewaktuan	80

BAB I

PENDAHULUAN

BAB I

PENDAHULUAN

1.1 LATAR BELAKANG

Dalam era globlisasi ini, ilmu pengetahuan dan teknologi semakin merupakan salah satu penunjang dalam keberhasilan suatu pembangunan. Bahkan ilmu pengetahuan dan teknologi (iptek) semakin lebih meluas lagi perkembangannya dan mencakup hampir seluruh bidang. Salah satu realisasi dari iptek adalah perkembangan dan aplikasi pada bidang robotika. Bidang robotika sangat erat hubungannya dengan pemecahan masalah yang dihadapi oleh manusia, dimana kemampuan dan kecerdasan dari robot tergantung dari sistem robot yang dibangun secara keseluruhan.

Dalam era robotika akhir ini, konsumen selalu menuntut dan menanti adanya perkembangan dalam teknologi dan aplikasi yang terkandung dalam suatu robot dalam membantu manusia dalam menyelesaikan tugasnya. Salah satu dari pekerjaan beresiko tinggi adalah pada saat melakukan pembersihan kaca pada gedung bertingkat banyak, hingga dirasa perlu untuk dikembangkan suatu robot sebagai alat bantu.

Fungsi dasar dari robot ini adalah untuk melakukan pergerakan merayap baik pada bidang vertikal maupun bidang datar atau dinding yang bersifat tidak memiliki lubang atau rongga pada permukaannya dengan pengoperasian melalui seorang operator. Beberapa aspek dari robot yang menjadi pertimbangan adalah bobot yang ringan, kemampuan dalam membawa beban yang tinggi, sistem gerak

yang fleksibel dan sederhana serta pertimbangan terhadap biaya produksi yang relatif rendah.

Realisasi dari robot ini merupakan generasi pertama dari ROBOT BERKAKI EMPAT BERKEMAMPUAN MERAYAP dengan mengaplikasikan prinsip kerja semiautomatic yang bermode operator oriented.

1.2 TUJUAN

Tujuan dari pelaksanaan dan realisasi proyek akhir ini adalah :

Untuk mengembangkan suatu teknologi baru robotika yang berorientasi pada peran pengganti manusia dalam berbagai pekerjaan yang beresiko tinggi yang berkaitan dengan ketinggian, misalnya membersihkan kaca jendela maupun pengecatan tembok pada gedung bertingkat banyak.

1.3 PERMASALAHAN

Permasalahan yang akan dibahas adalah berbagai aspek yang berkaitan dengan sistem kendali secara total dan algoritma dari robot, seperti yang akan didasarkan pada teori penunjang dalam bab pembahasan berikut.

1.4 PEMBATASAN MASALAH

Permasalahan yang berkaitan dengan realisasi dari ROBOT BERKAKI EMPAT BERKEMAMPUAN MERAYAP ini dikhususkan untuk berbagai permasalahan dan pembahasan yang terkategori dalam subjudul ini, yakni :

- * Sistem kendali secara keseluruhan dari robot
- * Algoritma sistem gerak robot

Perkecualian dari pembahasan dalam sub judul ini adalah yang menyangkut permasalahan mengenai konstruksi dan sistem gerak dari robot.

1.5 SISTEMATIKA PEMBAHASAN

Dalam penyelesaian proyek akhir ini penulis mengkategorikan pembahasan ke dalam lima bagian sebagai berikut :

BAB I PENDAHULUAN

Bab ini berisikan penguraian dari latar belakang, maksud dan tujuan, permasalahan, pembatasan masalah serta sistematika pembahasan yang berkaitan dengan perencanaan robot secara umum.

BAB II TEORI PENUNJANG

Bab ini berisi tentang teori-teori penunjang dalam perencanaan dan realisasi dengan ruang lingkup yang berhubungan dengan batasan permasalahan yang telah disebutkan di atas.

BAB III PERENCANAAN DAN PEMBUATAN

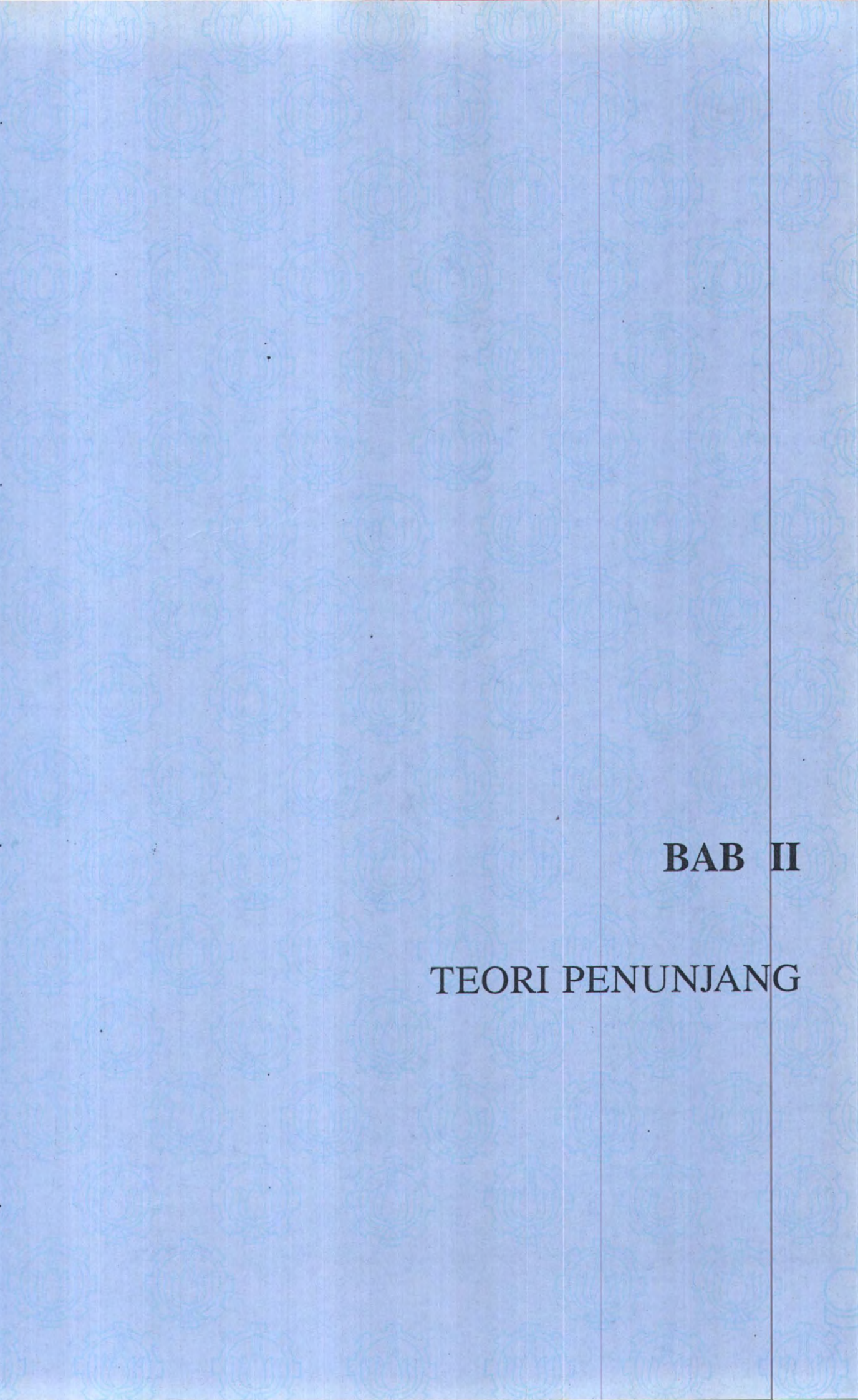
Bab ini berisikan tentang uraian lengkap dalam perencanaan dan pembuatan dari ruang lingkup permasalahan.

BAB IV PENGUJIAN DAN ANALISA

Bab ini menguraikan tentang pengujian rangkaian, analisis sistem kendali robot secara keseluruhan dan data hasil pengujian/pengukuran.

BAB V PENUTUP

Bab ini berisikan tentang kesimpulan dan saran-saran dalam realisasi, aplikasi, serta pengembangan untuk selanjutnya dari proyek ini.



BAB II

TEORI PENUNJANG

BAB II

TEORI PENUNJANG

Pada bab ini akan dibahas mengenai dasar-dasar teori yang dapat menunjang dari perencanaan, pengerjaan dan realisasi dari proyek robot berkaki empat berkemampuan merayap.

2.1 Pengenalan Robotika

Istilah robot yang biasanya terdengar umumnya mengandung istilah suatu makhluk yang menyerupai manusia atau bahkan bertingkah laku seperti manusia, hanya saja struktur tubuhnya tidak seperti manusia melainkan terbuat dari bahan logam. Istilah tersebut dapat dikategorikan pernyataan yang benar namun masih banyak definisi dari robot yang lainnya. Pada hakekatnya definisi dari robot mengandung beberapa kesamaan unsur-unsur pendukung antara lain adalah :

- Programmable (mampu diprogram)
- Automatic (otomatis)
- Manipulator (perangkat manipulasi)
- Humanlike (memiliki kemiripan dengan manusia)

Dari unsur-unsur di atas jelaslah bahwa robot bukan hanya sekedar perkakas biasa, namun merupakan mesin khusus yang dapat dikontrol oleh manusia lewat suatu *processor* maupun *controller*. Terdapat dua definisi yang dapat diterima oleh kalangan industri mengenai robot, yakni ;

- * Robot merupakan peralatan yang mampu melakukan fungsi-fungsi yang biasanya dilakukan oleh manusia, atau peralatan yang dapat bekerja dengan kecerdasan yang mirip dengan kecerdasan manusia. Definisi ini dikembangkan oleh Computer Aided Manufacturing International.
- * Robot merupakan peralatan manipulator yang mampu diprogram dan memiliki berbagai fungsi yang dirancang untuk memindahkan suatu barang melalui berbagai gerakan yang terprogram untuk melaksanakan pekerjaan tersebut.

2.2 Mobile Robot

Kata 'mobile' mempunyai arti bergerak dimaksudkan adalah sistem robot tersebut mampu memindahkan dirinya sendiri dari posisi A ke posisi B, dimana kedua posisi tersebut berada pada jarak tertentu (keseluruhan badan robot berpindah tempat). Bisa kita bayangkan robot tersebut dapat bergerak secara dinamis.

2.2.1 Operator Oriented

Mobile robot dengan operator oriented adalah pengendalian gerakan dari robot yang membutuhkan seorang operator. Jadi seluruh gerakan dari robot untuk memindahkan tubuhnya tergantung dari instruksi yang diberikan dari seorang operator.

2.2.2 Self Running

Mobile robot dengan self running adalah pengendalian gerakan dari robot yang berdasarkan program kemudi yang diberikan, sehingga seolah-olah robot itu bergerak sendiri. Jenis ini tidak tergantung dari kemudi seorang operator dan juga biasanya ditempatkan beberapa jenis sensor untuk mendeteksi situasi sekelilingnya (untuk mengenali medan jelajah). Sensor tersebut akan memberikan informasi kepada sistem robot, kemudian oleh perangkat prosesor/kontroler informasi tersebut diolah yang nantinya dijadikan sebagai acuan dalam melakukan pergerakan selanjutnya. Dalam hal ini juga dapat dikatakan bahwa terdapat hubungan timbal balik.

Self running robot dapat dikategorikan lagi menjadi dua, jika dilihat dari tingkat kecerdasan dari robot tersebut. Yakni robot dengan kecerdasan buatan dan tanpa kecerdasan buatan. Robot dengan kecerdasan buatan memiliki maksud bahwa robot tersebut berkemampuan secara mandiri untuk merespon/bereaksi yang sesuai, di dalam kondisi yang tidak dapat ditentukan sebelumnya (*Intelligence is the ability to autonomously do something appropriate under multiple unpredictable conditions*)¹⁾ Sebaliknya robot tanpa kecerdasan buatan secara keseluruhan bergantung pada instruksi yang diberikan.

2.3 Non_Mobile Robot

Kebalikan dari pengertian mobile robot, maka non_mobile robot memiliki pengertian sistem robot yang tidak dapat memindahkan posisinya dari suatu

¹⁾ A Meystel, *Autonomous Mobile Robots Vehicle with Cognitive Control*, World Scientific Singapore, 1991

tempat ke tempat yang lain. Artinya robot tersebut hanya dapat menggerakkan tubuhnya saja. Misalnya hal tersebut terjadi pada perangkat manipulatornya yakni lengan robot, tangan, kaki dan sebagainya, sementara tubuh robot berada pada posisi yang tetap.

2.4 Bahasa Pemrograman Robot

Algoritma yang digunakan untuk mendapatkan fungsi manipulator yang berfungsi sebagai kinematic, dinamic, control trajectory, scanning ataupun vision biasanya menggunakan embedded pada pengontrolan modul perangkat lunaknya.

Halangan besar dalam menggunakan manipulator pada mesin assembly untuk penggunaan secara umum adalah kurangnya ketepatan dan efisiensi komunikasi antara pemakai dan robot sehingga pemakai dapat secara langsung mengatur manipulator yang diinginkan.

2.4.1 Bahasa Assembly

Bahasa assembly adalah merupakan salah satu bahasa tingkat rendah yang dipakai dalam pemrograman. Mengenai bahasa assembly akan dibahas lebih lanjut di bawah ini.

2.4.2 Penggunaan Bahasa Assembly

Bahasa assembly digunakan karena sangat sulit dalam memprogram suatu mikroprosesor menggunakan bahasa mesin berupa heksadesimal. Program bahasa assembly atau assembler adalah program yang melibatkan input (sumber dari

program) yang dikodekan dengan mnemonic atau bahasa simbol mesin dan dirubah ke dalam program bahasa mesin heksadesimal (program obyek).

Alasan utama bahwa perangkat lunak tidak ditulis dengan bahasa mesin heksadesimal secara langsung karena saat sebuah program dirubah dalam bahasa mesin, alamat-alamat harus direlokasi kembali. Sebagai contoh bila JP 1000H dinaikkan 2. Hal ini akan sulit dalam pekerjaan bila program memiliki 10 sampai 15 instruksi JP. Assembler melakukan kerja tersebut secara otomatis bagi user.

2.4.3 Assembler

Assembler adalah program yang merubah perangkat lunak yang ditulis dalam bahasa mesin simbolis ke bahasa mesin heksadesimal. Bahasa mesin simbolis dari perangkat lunak ini disebut program sumber atau source program dan heksadesimal disebut sebagai program obyek atau object program. Source program menyediakan assembler dengan sumber dari konversi dan output yang berupa bahasa mesin heksadesimal dari assembler adalah obyeknya.

2.4.4 Pernyataan Bahasa Assembly

Sebelum dilakukan penulisan source program bahasa assembly, pernyataan bahasa assembly dasar harus dikuasai terlebih dahulu. Pernyataan ini meliputi empat daerah yang berbeda, yakni :

- ⇒ Label
- ⇒ Opcode
- ⇒ Operand

⇒ Comment (Komentar)

yang digunakan untuk menampung beberapa tipe dari informasi. Setiap daerah harus berisi ruang atau beberapa karakter alphanumeric, dan setiap daerah harus dipisahkan dengan paling tidak satu tempat. Jika tidak data masukan yang berbentuk bebas.

2.4.4.1 Daerah Label

Daerah label meliputi alamat memori simbolis yang digunakan untuk mengacu pada pernyataan pada program. Label-label dipilih dan harus diakhiri dengan titik dua. pada beberapa assembler Z80. Label dibentuk dari karakter alphanumeric dan harus diawali dengan sebuah tulisan. Digit sisanya dapat berupa nomor dari tulisan, dan dengan beberapa assembler nomor tertentu dari karakter spesial dapat juga digunakan.

Komentar dapat juga dimasukkan pada daerah label pada assembler Z80 umumnya. Jika karakter pertama dari daerah label adalah asterik (*) atau titik koma (;), seluruh pernyataan dianggap oleh assembler sebagai komentar. Komentar tidak dirubah oleh assembler selama untuk kepentingan dokumentasi.

Yang juga sangat penting adalah beberapa assembler tidak memperbolehkan penggunaan opcode sebagai label.

2.4.4.2 Daerah Opcode

Daerah opcode harus selalu mengandung Z80 opcode yang valid atau opcode semu (pseudo). Jika tidak, assembler akan menyatakan kesalahan. Pemasukan lain pada daerah ini akan mengakibatkan kesalahan.

2.4.4.3 Daerah Operand

Daerah operand dapat berisi register-register, data atau label. Jika lebih dari satu operand yang muncul, maka harus dipisahkan dengan koma. Data harus dalam bentuk desimal, biner, oktal, heksadesimal atau ASCII. Saat menggunakan data heksadesimal hal yang perlu diingat adalah untuk menambahkan leading zero jika terdapat heksadesimal data yang dimulai dengan tulisan atau huruf, misalnya (07AH). Jika tidak ditambahkan leading zero, assembler akan mengasumsikan sebagai kesalahan. Data ASCII dapat juga digunakan sebagai operand, dan jika digunakan harus dikelilingi oleh apostroph.

Sebagai tambahan untuk menentukan sistem nomor dari data operand, assembler dapat juga melakukan operasi aritmatik pada data operand.

2.4.4.4 Daerah Komentar

Daerah komentar harus dimulai dengan titik koma atau asterik pada assembler Z80 pada umumnya dan dilanjutkan hingga akhir baris. Jika dilanjutkan pada baris berikutnya, maka baris berikutnya tersebut harus pula dimulai dengan tanda titik koma atau tanda asterik.

2.4.5 Macro Assembler

Macro assembler adalah bentuk istimewa dari assembler standar yang memperbolehkan user untuk menentukan opcode baru²⁾. sebagai contoh Z80 tidak dapat menukar isi dari register HL langsung dengan register BC. Hal ini mungkin umum dalam beberapa program untuk memakai tugas ini. Tugas dapat diprogram sebagai subrutin atau sebagai macro jika macro assembler bisa digunakan. Dengan subrutin SWITCH, instruksi call SWITCH digunakan dengan disisipkan pada main program setiap waktu yang berisi HL ditukar dengan BC. Jika tugas yang sama dikembangkan sebagai macro, program mengandung opcode SWITCH setiap waktu saat HL dan BC ditukar.

2.4.6 Macros

Untuk mengkonversi subrutin yang terdaftar menjadi macro, pertama kali macro harus didefinisikan dan semua langkah dari subrutin kecuali RET diakhiri dengan pernyataan definisi macro. Dari titik ini assembler secara otomatis akan mengikuti pengkodean macro pada program setiap waktu dimana opcode baru SWITCH ditemukan pada program. Yang perlu diperhatikan adalah assembler menyertakan langkah-langkah antara MACRO dan MEND dari macro original ke program. Assembler mendahulukan setiap langkah dari macro dengan tanda plus (+) sehingga dapat diketahui pada daftar/listing.

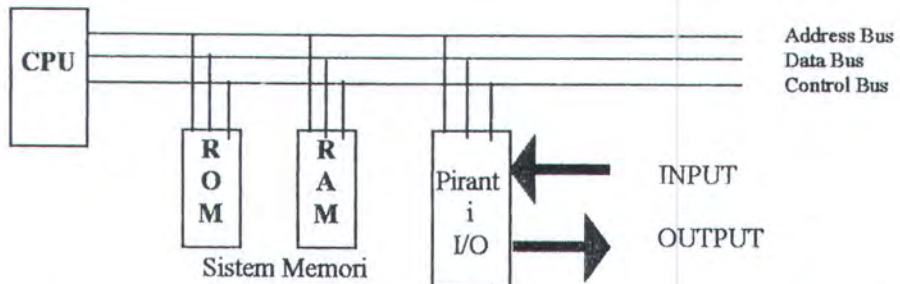
²⁾ Barry Brey, **The Z80 Microprocessor Hardware Software Programming and Interfacing**, Prentice Hall Inc, New Jersey, 1988

2.5 Pengertian Sistem Minimum

Dalam merencanakan suatu peralatan kontrol elektronik yang berbasis mikroprosesor memerlukan pemahaman yang cukup tentang aspek aspek sistem yang optimal. Pengertian optimal menghendaki bahwa aspek handal, efektif dan efisien harus dipenuhi. Aspek terakhir yang dimungkinkan adalah efisien dalam sisi biaya. Orientasi disain dapat dicapai dengan dua cara, yakni orientasi subyek dan orientasi obyek. Maksudnya orientasi subyek berangkat dari piranti kontrol yang dimiliki atau yang telah ada. Pemakai tinggal merancang sistem interface dari piranti kontrol, misalnya dari mikroprosesor ke dalam sistem yang akan dikontrol. Sedangkan orientasi subyek berangkat dari subyek yang akan dikontrol. Membangun dari awal suatu sistem rangkaian berbasis mikroprosesor sebagai sistem kontrol elektronik dapat dikategorikan sebagai langkah disain yang berorientasi pada obyek. Dari fenomena ini muncul pengertian sistem minimum, yang dapat diterjemahkan sebagai sistem rangkaian yang minim dan telah dapat digunakan sebagai rangkaian kontrol dalam menjawab permasalahan obyek yang harus diselesaikan³⁾. Pengertian ini akan lebih tepat dipakai bila piranti IC yang membentuk sistem rangkaian ini berdiri sendiri sendiri.

Sistem rangkaian berbasis mikroprosesor Z80 dalam bentuk yang paling ringkas tetapi telah memenuhi persyaratan sebagai kontrol elektronik ditunjukkan oleh diagram blok di bawah ini :

³⁾ Endra Pitowarno, **Mikroprosesor dan Interface 1**, PES-SBY, Surabaya, 1995, hal 89

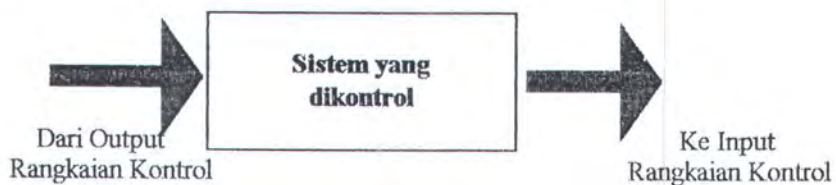


Gambar 2.1 Blok Diagram Sistem Rangkaian Berbasis Mikroprosesor⁴⁾

Sistem rangkaian pada gambar di atas terdiri dari tiga komponen pendukung utama yaitu,

1. CPU/ Mikroprosesor
2. Sistem Memori : dapat berupa EPROM dan RAM
3. Sistem piranti I/O berupa IC peripheral

Sedangkan yang disebut sebagai sistem yang di kontrol diilustrasikan pada gambar di bawah berikut :

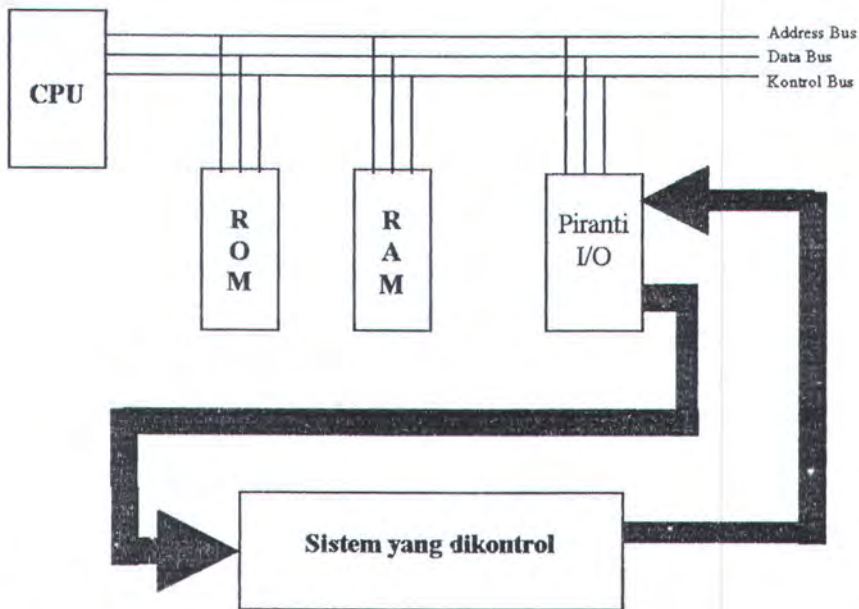


Gambar 2.2 Ilustrasi blok diagram sistem yang dikontrol⁵⁾

Blok diagram di atas dapat mewakili seluruh model permasalahan yang dapat diatur dengan kontrol elektronik.

⁴⁾ Ibid. hal. 89

⁵⁾ Ibid. hal. 90



Gambar 2.3 Ilustrasi Blok Diagram Sistem Rangkaian Berbasis Mikroprosesor yang digabung dengan sistem yang dikontrol⁶⁾

Apabila sistem pengontrol pada gambar 2.1 digabung dengan rangkaian yang dikontrol maka blok diagramnya akan menjadi seperti di atas.

Dalam prakteknya terminal input maupun terminal output sistem yang dikontrol, dapat beroperasi pada macam macam besaran listrik atau fisika. Tetapi besaran ini harus diterjemahkan ke dalam besaran logika apabila dihubungkan ke rangkaian berbasis mikroprosesor.

Piranti piranti penerjemah ini disebut sebagai interfacing ke piranti sensor maupun aktuator. Piranti sensor merupakan piranti input dari sistem yang akan dikontrol sedangkan piranti aktuator merupakan piranti outputnya.

⁶⁾ Ibid. hal. 91

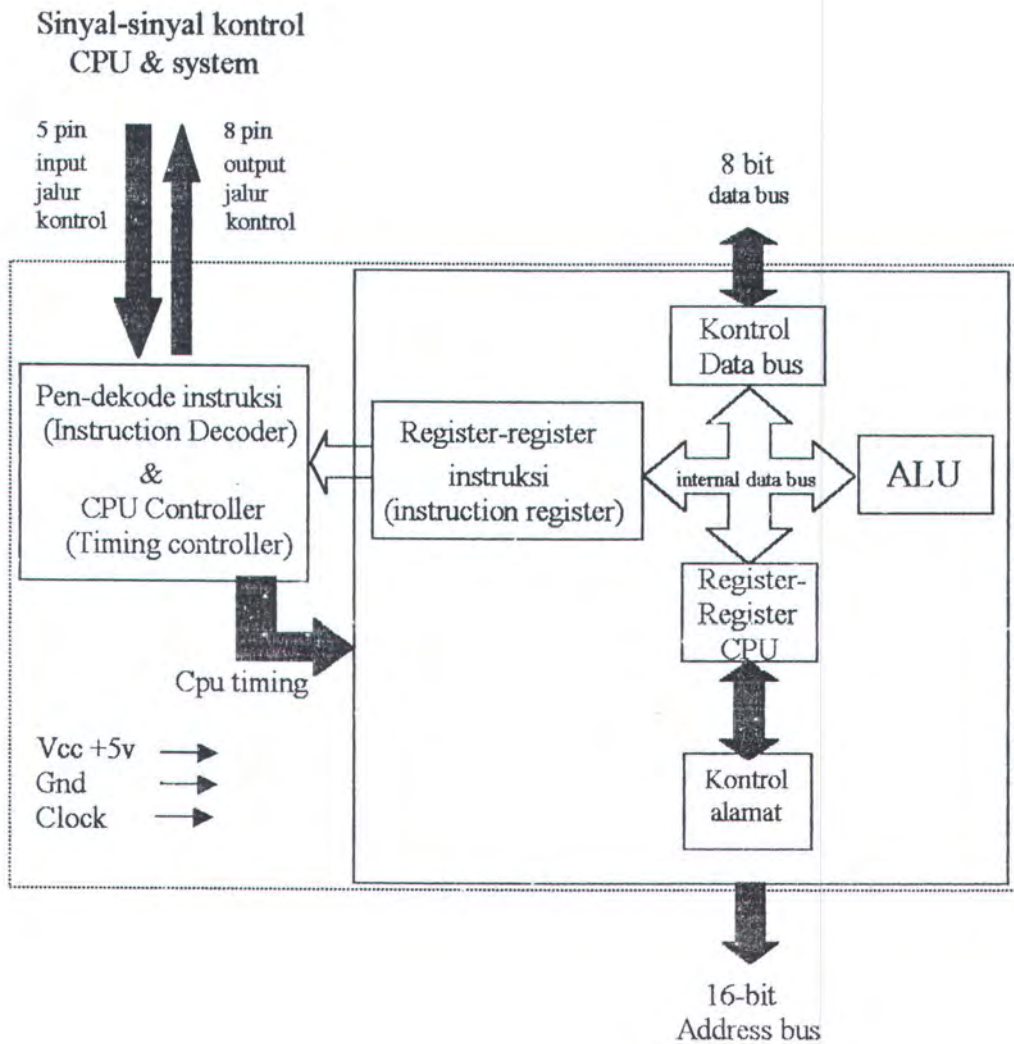
Pada bagian berikut akan dijelaskan masing masing komponen pembangun dari sistem minimum yang diaplikasikan pada proyek akhir ROBOT BERKAKI EMPAT BERKEMAMPUAN MERAYAP ini.

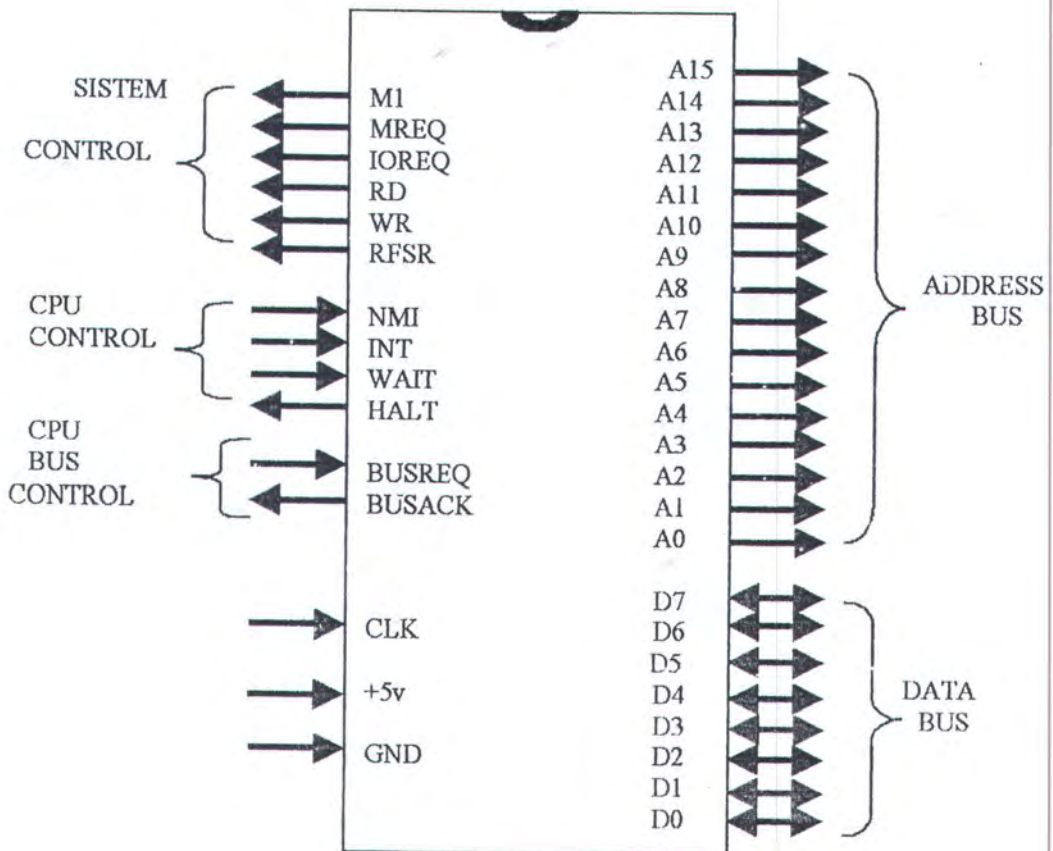
2.6 Arsitektur Mikroprosesor Z80

CPU Z80 pertama kali dibuat oleh Zilog Corporation sekitar tahun 1975-1976. Arsitektur CPU Z80 didisain dengan mengaplikasikan mode standar bus, yaitu pengelompokan jalur-jalur dalam tiga macam bus sebagai berikut:

- Address bus
- Data bus
- Control bus

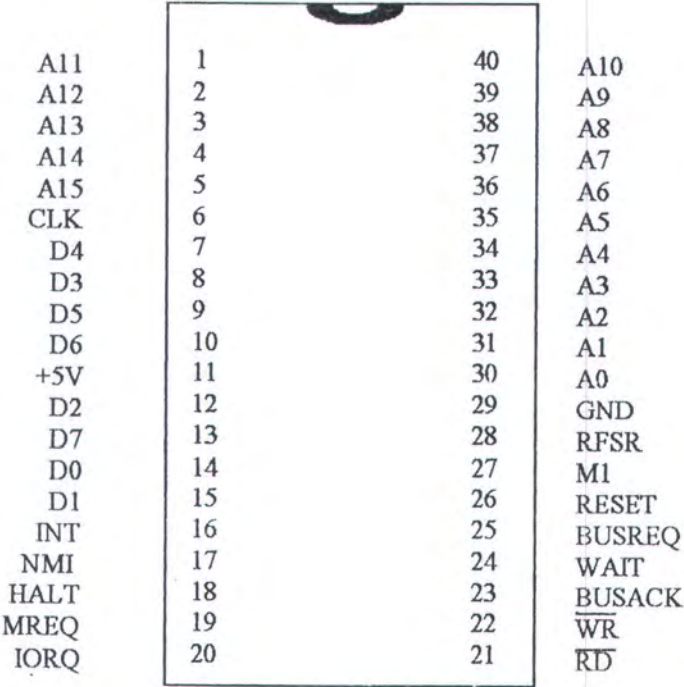
Gambar 1 menunjukkan arsitektur CPU Z80 baik untuk tipe non-CMOS maupun tipe CMOS. Kemampuan kerja Z80 ini dapat mencapai 4 MHz, sedang untuk tipe CMOS dapat mencapai 16 MHz. Z80 ini bekerja dengan tegangan tunggal +5V. Ia memiliki internal register sebanyak 208 bit, termasuk diantaranya 6 buah general purpose register yang dapat digunakan terpisah (tiap 8-bit) atau digunakan dalam bentuk register pair 16-bit. Tersedia pula instruksi exchange yang dapat digunakan untuk melipatgandakan accumulator atau general purpose register yang lain. Z80 juga mempunyai sebuah register Stack Pointer, Program Counter, dua buah Index Register, sebuah Refresh Register dan sebuah Interrupt Register.

Gambar 2.4 Arsitektur Z80⁷⁾⁷⁾ Ibid. hal. 45



Gambar 2.5 Konfigurasi pin Z80 berdasarkan fungsi Input-Output⁸⁾

⁸⁾ Ibid. hal. 46



Gambar 2.6 Konfigurasi pin Z80 berdasarkan tata letak dalam DIL 40 pin⁹⁾

⁹⁾ Ibid. hal. 46

Berikut keterangan tentang fungsi fungsi dari pin tersebut :

A₀ sampai A₁₅ Address Bus, berfungsi sebagai output , dapat memiliki tiga kondisi (high, low dan tri-state). A₀ hingga A₁₅ membentuk 16 bit address bus dengan posisi A₀ sebagai LSB dan A₁₅ sebagai MSB. Kapasitas memori dengan pengalamatan langsung adalah 64 KB.

BUSACK Bus Acknowledge berfungsi sebagai output aktif low. Bus acknowledge berfungsi untuk memberi tahu pada rangkaian tertentu yang sebelumnya meminta pada CPU untuk melakukan sesuatu bahwa CPU address bus, CPU data bus dan sinyal sinyal kontrol MREQ, IORQ, RD dan WR telah dalam keadaan *high impedance* (tri-state), untuk selanjutnya rangkaian tertentu dapat mengontrol bus bus untuk digunakan.

BUSREQ Bus Request, berfungsi sebagai input aktif low. Bila rangkaian luar menginginkan agar suatu saat CPU bus berada dalam keadaan tri-state dengan maksud menggunakan bus tersebut untuk suatu keperluan maka bus request dapat digunakan. Dengan mengaturnya dalam keadaan low maka CPU bus yaitu address bus, data bus, MREQ, IORQ, RD dan WR akan tri-state. Apabila dalam keadaan tri-state maka CPU akan memberikan sinyal BUSACK.

D_0 sampai D_7 Data Bus, dapat berfungsi sebagai input maupun output, bisa dalam tiga kondisi (high, low dan tri-state). D_0 sampai D_7 adalah 8 bit *bidirectional* data bus digunakan untuk pertukaran antara CPU dan memori atau rangkaian I/O.

$\overline{\text{HALT}}$

Halt state, berfungsi output aktif low. Sinyal halt yang muncul menunjukkan bahwa CPU telah mendapat perintah dari program untuk menghentikan proses kerjanya sementara, sampai ia mendapat sinyal interrupt baik INT ataupun NMI. Selama menunggu sinyal interrupt, CPU menjalankan instruksi NOP secara berulang untuk menjaga agar memori tetap valid.

$\overline{\text{INT}}$

Interrupt Request, berfungsi sebagai input aktif low. Apabila peralatan I/O membutuhkan fasilitas agar suatu saat dapat melakukan proses interrupt pada CPU (interupsi terhadap proses yang sedang dikerjakan). Maka ia dapat menggunakan kontrol INT. Bila INT diaktifkan maka CPU akan melakukan interupsi terhadap program yang sedang dijalankan dan akan menuju ke suatu program yang telah didefinisikan sesuai dengan pemberian intruksi Int. Alamat awal dari program ini disebut sebagai interrupt vector.

$\overline{\text{IORQ}}$

Input/Output Request, berfungsi sebagai output aktif low dan dapat memiliki tiga kondisi (high, low dan tri-state). Adanya sinyal aktif IORQ menunjukkan bahwa A_7 hingga A_0 dari address bus sedang berisi alamat I/O yang valid untuk

melakukan operasi read atau write pada piranti I/O tersebut. Sinyal IORQ juga dihasilkan secara bersamaan oleh sinyal M1 pada saat suatu permintaan interrupt telah diterima oleh CPU untuk menunjukkan bahwa operasi berikut yang merupakan interrupt vector boleh ditempatkan di data bus.

$\overline{M1}$

Machine Cycle One, berfungsi sebagai output dan aktif low. M1 bersama sama IORQ menunjukkan bahwa siklus mesin (operasi) yang sedang berjalan pada saat itu adalah *opcode fetch cycle* (siklus pengambilan opcode) dari eksekusi sebuah instruksi.

$\overline{MREQ}$

Memory Request, berfungsi sebagai output aktif low dan dapat memiliki tiga kondisi (high, low dan tri-state). Adanya sinyal aktif MREQ menunjukkan bahwa address bus sedang berisi alamat yang valid untuk suatu proses write atau read memori.

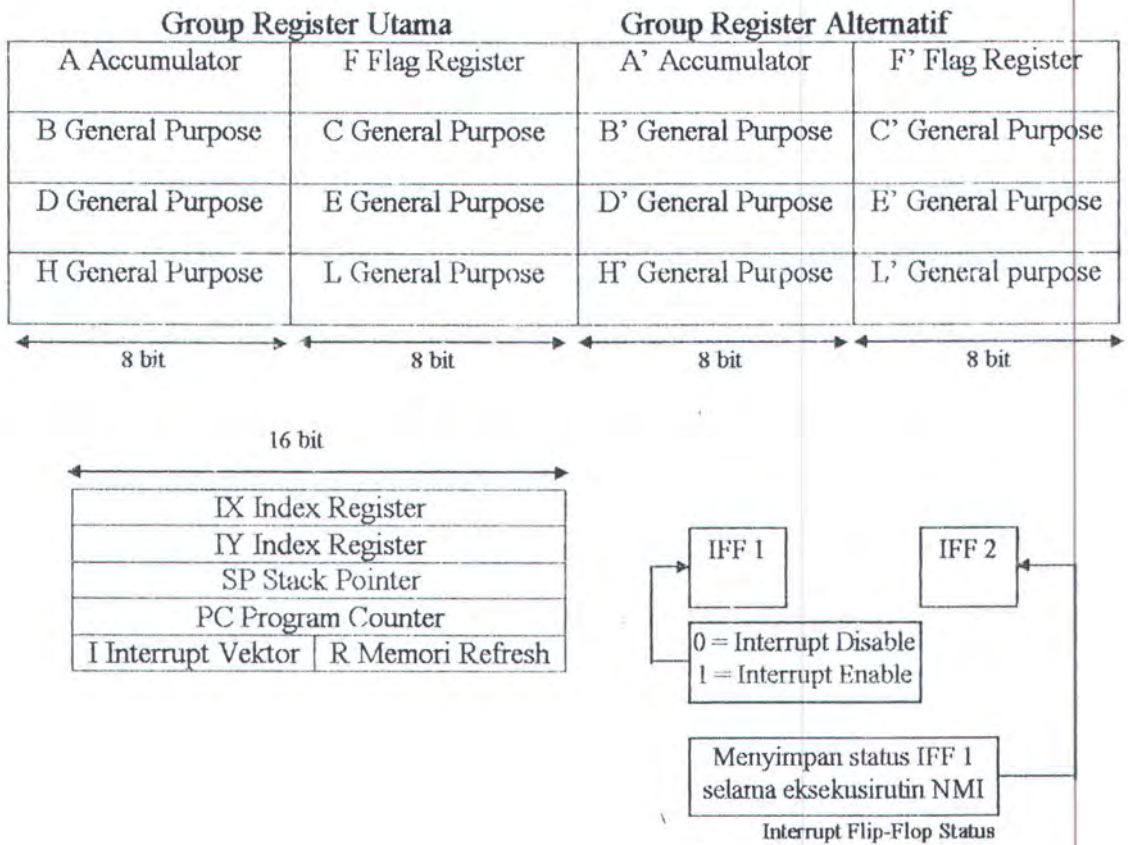
$\overline{NMI}$

Non Maskable Interrupt, berfungsi sebagai input, *negative edge triggered*. NMI memiliki prioritas yang lebih tinggi daripada INT. Maksudnya, bila pertama diberi INT kemudian NMI maka CPU akan memperhatikan NMI meskipun proses yang dikehendaki oleh INT belum selesai. Tetapi apabila diberikan NMI terlebih dahulu kemudian INT maka INT harus menunggu proses yang dikehendaki NMI selesai terlebih dahulu sehingga INT akan diperhatikan. Pada setiap pemberian sinyal NMI, CPU akan melakukan proses jump ke alamat 0066H.

$\overline{\text{RD}}$	Read, berfungsi sebagai output aktif low dan dapat memiliki tiga kondisi (low, high, tri-state). Adanya sinyal RD yang aktif menunjukkan bahwa CPU menginginkan proses pembacaan data dari memori atau peralatan input.
$\overline{\text{RESET}}$	Reset berfungsi sebagai input aktif low. Reset berarti menginisialisasi CPU kembali ke posisi : PC diset 0000H, semua register termasuk I dan R diclearkan dan interrupt status diset pada mode 0. Selama reset terjadi, address bus dan data bus berada dalam kondisi <i>high impedance</i> (tri-state), sedangkan semua sinyal output berada dalam kondisi tidak aktif (inactive). Untuk mendapatkan hasil sinyal reset yang lengkap harus diberikan minimal selama tiga siklus clock secara penuh.
$\overline{\text{RSFH}}$	Refresh, berfungsi sebagai output aktif low. RSFH dan MREQ bersama sama menunjukkan bahwa konfigurasi A7 sampai A0 pada address bus dapat digunakan sebagai refresh address pada sistem memori dinamik (dynamic memories).
$\overline{\text{WR}}$	Write, berfungsi output aktif low dan dapat memiliki tiga kondisi (low, high dan tri-state). WR menunjukkan bahwa CPU menginginkan suatu proses penulisan data ke memori atau peralatan output.
CLK	Clock, berfungsi sebagai input. Pada jalur ini harus diberikan pulsa clock dari rangkaian osilator agar rangkaian dapat bekerja.

2.6.1 Register-register CPU Z80

CPU Z80 memiliki Read/Write memori yang disebut sebagai register sebanyak 208 bit atau 26x8 bit. Memori di dalam CPU ini sangat berguna bagi user dalam proses programming. Gambar di bawah menunjukkan bagaimana interval memori dibagi dalam 18 buah register 18 bit dan 4 buah register 16 bit.



Gambar 2.7 Register-register Z80¹⁰⁾

¹⁰⁾ Ibid. hal. 50

2.6.2 Teknik-teknik Interrupt Z80

Dalam praktek, penggunaan dari suatu interrupt adalah untuk mengijinkan suatu peralatan peripheral input/output port atau suatu sistem rangkaian yang terhubung ke sistem CPU yang utama untuk menunda operasi yang sedang dijalankan oleh CPU sewaktu-waktu pada saat proses sedang berjalan, dengan maksud meminta atau memerintahkan CPU agar mengeksekusi program-program rutin dari peripheral tersebut sesuai vektor dari interrupt yang telah didefinisikan sebelumnya. Biasanya service rutin ini berisi program yang meliputi pertukaran data atau status dan informasi kontrol, antara CPU dan peripheral. Bila service rutin ini telah dieksekusi, CPU akan kembali melakukan proses yang sebelumnya.

2.6.2.1 Interrupt Enable-Disable

CPU Z80 memiliki 2 buah instruksi untuk operasi interrupt. Yang pertama adalah berupa *software maskable interrupt* (INT) dan yang kedua adalah *non maskable interrupt* (NMI). NMI memiliki status yang tertinggi setelah reset yang artinya NMI tidak dapat dielakkan oleh user, bilamana sinyal NMI datang maka secara otomatis proses yang sedang berlangsung akan berhenti dan pengalamatan dari program secara otomatis melakukan jump ke alamat 0066H.

INT umumnya disediakan untuk fungsi yang sangat penting yang dapat di'*enable*' atau di'*disable*' sesuai keinginan programmer. Di dalam CPU Z80 terdapat sebuah instruksi Disable Interrupt (DI) dan sebuah instruksi Enable Interrupt, serta ada dua buah flip-flop yang bekerja mengaktifkan interrupt, yakni IFF1 dan IFF2. Status IFF1 difungsikan untuk menghalangi interrupt-interrupt,

sedang IFF2 sebagai lokasi penyimpanan sementara dari IFF1. Instruksi enable interrupt (EI) akan mengeset IFF1 dan IFF2 menuju status enable. Jika suatu maskable interrupt INT ditangkap oleh CPU, IFF1 dan IFF2 secara otomatis direset, menghalangi perintah INT berikutnya sampai diberikan lagi instruksi EI yang baru. Perlu diingat bahwa untuk semua kasus yang diterangkan di atas, IFF1 dan IFF2 adalah selalu sama. Tabel di bawah ini menunjukkan ringkasan pengaruh dari instruksi-instruksi berbeda terhadap dua buah flip-flop IFF1 dan IFF2.

Tabel 2.1 Ringkasan pengaruh instruksi terhadap IFF1 dan IFF2¹¹⁾

Aksi yang dilakukan	IFF1	IFF2	Keterangan
RESET CPU	0	0	Status INT disabled
Eksekusi instruksi DI	0	0	Status INT disabled
Eksekusi instruksi EI	1	1	Status INT disabled
Eksekusi LD A,I	x	x	IFF2 dicopy ke parity flag
Eksekusi LD A,R	x	x	IFF2 dicopy ke parity flag
Menerima NMI	0	x	Status INT disabled
Eksekusi RETN	IFF2	x	Setelah return IFF2 dicopy kembali ke IFF1

catatan : tanda x menunjukkan bahwa statusnya tidak berubah

2.6.2.2 Mode-mode Interrupt Respons

CPU Z80 dapat diprogram untuk memberikan respon terhadap maskable interrupt INT dalam salah satu dari tiga mode yang tersedia¹²⁾.

Mode 0 Dengan mode ini, piranti penginterrupt dapat memberikan bermacam-macam instruksi dengan bebas untuk dieksekusi oleh

¹¹⁾ Ibid. hal. 58

¹²⁾ Ibid. hal. 59

CPU. Jumlah clock cycle yang dibutuhkan untuk pelaksanaan instruksi ini secara otomatis bertambah dua clock cycle, berbeda bila dilaksanakan tanpa kemudi interrupt (instruksi dijalankan normal). Hal ini terjadi karena secara otomatis CPU menambahkan dua buah wait states terhadap adanya respon interrupt untuk memberikan waktu yang cukup terhadap suatu proses daisy chain eksternal yang merupakan pengatur/kontrol terhadap prioritas proses, yang biasanya metode daisy chain selalu digunakan untuk mengikuti aplikasi interrupt. Setelah sinyal reset dikenakan pada suatu CPU, secara otomatis respon interrupt diset pada mode 0.

- Mode 1 Bila mode ini dipilih oleh programmer, CPU akan memberikan respon terhadap suatu interrupt dengan mengeksekusi suatu perintah restart ke alamat 0038H. Jadi pada prinsipnya adalah identik dengan pengaktifan non-maskable interrupt hanya saja pemanggilannya tidak ditujukan pada alamat 0066H melainkan ke alamat 0038H. Jumlah siklus yang dibutuhkan untuk menyelesaikan instruksi restart ini juga ditambah dengan dua wait states, atau lebih dua dari keadaan normal.
- Mode 2 Di antara mode-mode yang lain, mode ini adalah yang bervariasi terhadap hal penempatan *vector interrupt*-nya. Dengan hanya memberikan satu byte ke data bus, suatu panggilan tak langsung (indirect call) dapat dilakukan ke setiap lokasi memori. Dengan mode ini programmer dapat memilih mode interrupt di mana saja di seluruh daerah memori yang mungkin untuk dialamati.

2.7 Piranti memori

Sampai dengan dekade 90-an ini telah banyak dikenal piranti memori yang digunakan dalam sistem rangkaian elektronik. Untuk memori berupa chip IC yang dipasang tetap dalam rangkaian, telah dikenal mulai dari IC RAM (Random Access Memory), PLD (Programmable Logic Device) dan PAL (Programmable Array Logic). Tipe IC RAM masih dibagi-bagi lagi menjadi static RAM dan dinamic RAM. Belakangan ini muncul istilah Flash RAM karena sifat accessnya yang begitu cepat. Demikian pula dengan ROM, yang juga dibagi lagi menjadi programmable ROM (PROM), Ultraviolet Erasable Programmable ROM (UVROM), Electrical EPROM (EEPROM), dan beberapa tipe EPROM khusus lainnya.

2.7.1 Erasable Programmable Read Only Memory (EPROM)

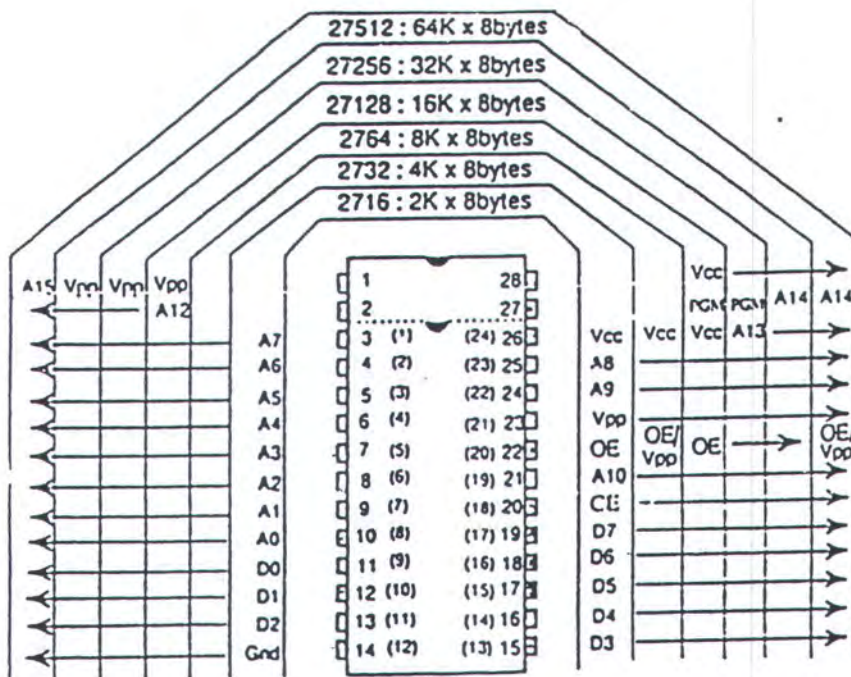
EPROM merupakan suatu IC memori yang bersifat hanya dapat dibaca pada saat IC tersebut sudah terintegrasi dengan rangkaian, atau yang biasanya disebut Read Only Memory (ROM). Artinya seluruh isi memori yang ada hanya untuk dibaca saja datanya. Proses pengisian data pada EPROM dilakukan secara khusus sebelum IC ini dipasang dalam rangkaian. Cara pengisian dan penghapusan tergantung dari spesifikasi atau tipe dari IC memori tersebut.

Cara penghapusan data isi memori pada ROM dapat dilakukan dengan dua cara, yakni cara pertama untuk tipe UVROM ataupun EPROM saja dapat dihapus isi memorinya dengan cara memberikan sinar ultraviolet dengan intensitas tertentu serta dalam jangka waktu tertentu, sesuai dengan spesifikasinya

masing-masing. Sedangkan cara kedua yakni untuk tipe EEPROM dapat dilakukan penghapusan dengan cara memberikan tegangan tertentu pada pin atau jalur yang telah disediakan.

Berdasarkan pembuatannya terdapat dua macam ROM yaitu pertama dengan tipe CMOS dan yang kedua bertipe Non CMOS. Tipe yang pertama biasanya ditandai dengan kode C di tengah-tengah IC tersebut.

Kompabilitas berbagai macam tipe EPROM berdasarkan fungsi pin dapat dilihat dalam gambar di bawah ini.



Gambar 2.8 Kompabilitas fungsi pin berdasar tipe EPROM¹³⁾

¹³⁾ Ibid. hal. 40

Keterangan fungsi pin¹⁴⁾ :

VPP	Pin ini harus diberi tegangan tertentu, yang biasanya lebih besar dari tegangan Vcc, selama pemrograman EPROM. Misalnya untuk tipe 2764 (Non CMOS), tegangan VPP adalah 21 Volt. Sedangkan untuk tipe 27C64 (CMOS) atau tipe 2764A, tegangan VPP adalah 12,5 Volt. Untuk tipe 27256, tegangan VPPnya adalah 21 Volt, sedangkan untuk tipe 27C256 besarnya VPP adalah 12,75 Volt.
PGM	Adalah merupakan jalur kontrol untuk pemrograman. PGM harus diberi logika 0 bila dilakukan proses penulisan data ke EPROM. PGM dan VPP digunakan pada saat EPROM diprogram melalui EPROM Writer atau EPROM Programmer.
OE	Output Enable. OE harus diberi logika 0 agar data yang berada di lokasi tertentu di EPROM dapat dibaca.
CE	Chip Enable. CE harus diberi logika 0 agar chip EPROM aktif (dapat dihubungi).
D7-D0	Merupakan Data Bus. Dihubungkan langsung ke data bus dari CPU.
A(?) - A0	Merupakan Address Bus, dimana jumlah pin-nya tergantung dari kapasitas memori EPROM. Pin address ini dihubungkan langsung ke address bus dari CPU sesuai dengan nomor pin.

¹⁴⁾ Ibid. hal. 41

Dengan memperhatikan persamaan dan perbedaan fungsi kaki-kaki EPROM mulai dari tipe 2716 hingga tipe 27256, dapat dikembangkan suatu perancangan dari rangkaian sistem memori yang sesuai.

2.7.2 Random Access Memory (RAM)

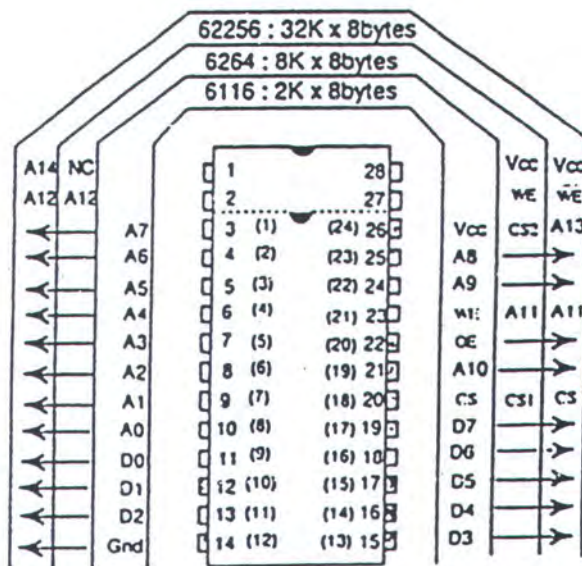
RAM merupakan suatu unit memori yang bersifat read dan write. Maksudnya adalah dalam sistem rangkaian yang sedang bekerja, isi data dari RAM ini selain dapat dibaca juga dapat dipanggil atau diubah dengan cara mengisinya dengan data yang baru. Pembacaan dan pengisian datanya dapat dilakukan pada saat sistem rangkaian bekerja.

Menurut cara menyimpan data, dikenal ada dua macam RAM, yakni static RAM dan dinamic RAM. Static RAM dapat menyimpan data yang telah dituliskan kepadanya, selama catu daya tetap diberikan, atau selama tidak ada proses penulisan data baru ke alamat yang sama. Sehingga hampir tidak diperlukan rangkaian tambahan untuk mengoperasikan static RAM ini di dalam rangkaian.

Sedangkan dinamic RAM bekerja menyimpan data seperti sifat kapasitor menyimpan muatan yang dihubungkan dengan suatu beban. Makin lama, muatan atau data akan hilang sejalan dengan mengalirnya arus ke beban. Untuk dapat mempertahankan datanya maka dinamic RAM harus dirangkai dengan suatu rangkaian yang disebut *Memory Refreshing Circuit*. Rangkaian ini berfungsi untuk menyegarkan kembali ingatan atau data yang disimpan dengan cara memberikan pulsa refresh secara kontinu dengan selang waktu tertentu.

Keterangan fungsi pin pada RAM :

- D7-D0 Data bus. Dihubungkan langsung ke data bus CPU.
- A(?) - A0 Merupakan address bus, dimana jumlah pin tergantung dari kapasitas memori RAM. Dihubungkan langsung ke address bus dari CPU sesuai dengan nomor pin.
- WE Write Enable. Pin ini harus diberilogika 0 agar data pada RAM dapat dibaca oleh CPU.
- CS Chip Select. Berlaku seperti CE pada EPROM. CS harus diberi logika 0 agar chip RAM dapat aktif dan dapat dihubungkan dengan CPU.



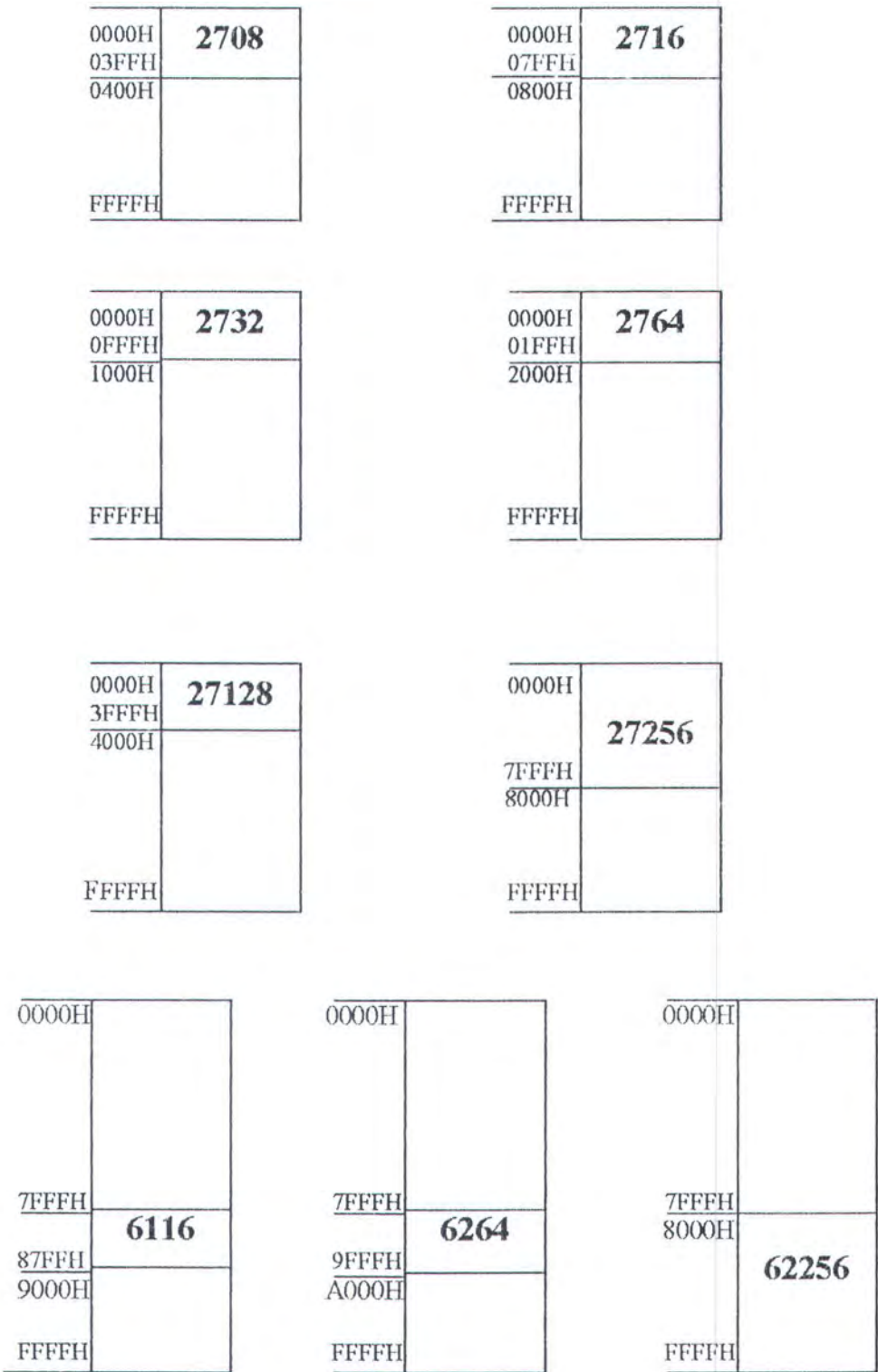
Gambar 2.9 Konfigurasi fungsi pin berdasarkan tipe RAM¹⁵⁾

¹⁵⁾ Ibid. hal. 42

2.7.3 Interfacing Piranti Memori dengan Z80

Pemilihan suatu RAM ataupun EPROM tidak hanya berdasarkan bagaimana membuat sistem rangkaian memori selengkap mungkin, tetapi juga perlu mempertimbangkan seberapa besar sebenarnya kapasitas memori yang kita butuhkan dalam pemecahan suatu permasalahan berkaitan dengan kontrol programming. Hal yang terpenting dalam pemilihan tipe memori adalah pemetaan memori dari masing-masing RAM atau EPROM. Pada umumnya letak keduanya dapat dipertukarkan, tetapi dalam prakteknya, alamat RAM dimulai dari alamat 0000h dan EPROM dimulai dari alamat 8000h (untuk kapasitas memori masing-masing 256 KB). Dapat pula ditambahkan dengan adanya kemudahan dalam pemilihan IC dekoder dalam pengendalian memori tersebut. Dengan adanya konfigurasi alamat seperti di atas maka hanya diperlukan satu gate inverter pada pin A15 yang akan membedakan apakah IC tersebut sebagai RAM atau EPROM yang dihubungi oleh CPU. Dengan adanya kemudahan ini kita tidak perlu untuk merancang dengan menggunakan IC dekoder dalam penentuan fungsi dari IC tersebut.

Pemetaan dari RAM dan EPROM untuk CPU dapat dilihat pada gambar berikut ini :



Gambar 2.10 Pemetaan alamat dari RAM dan EPROM¹⁶⁾

¹⁶⁾ Ibid. hal. 98-100

2.8 Piranti I/O (Input/Output)

Dalam membangun piranti I/O dengan menggunakan IC TTL seluruhnya akan membutuhkan jumlah komponen yang relatif banyak. Dalam banyak aplikasi, perancangan dapat mengaplikasikan suatu IC khusus yang dapat menghubungkan atau meng-*interface*-kan peralatan-peralatan luar agar dapat berkomunikasi dengan CPU. Salah satunya adalah IC PPI 8255 atau Programmable Peripheral Interfacing. Selain itu juga terdapat Z80PIO yang khusus digunakan dalam lingkup perancangan sistem berbasis CPUZ80.

2.8.1 Programmable Peripheral Interface (PPI) 8255

PPI 8255 adalah merupakan suatu piranti paralel Input/Output dalam suatu chip serbaguna yang dapat diprogram fungsi Input/Outputnya. Awalnya PPI 8255 dibuat oleh Intel yang digunakan bersama-sama dengan mikroprosesor buatan Intel. Tetapi karena komponen ini berbasis standar bus dan relatif mudah dalam pengaplikasiannya, maka kebanyakan tipe mikroprosesor lainnya juga turut memanfaatkannya.

PPI 8255 mempunyai 24 pin I/O yang terdiri dari 3 port, yakni Port A (8 bit/pin), Port B (8 pin) dan Port C (juga 8 pin). Port A dan port C pada bit D7-D4 merupakan grup kontrol A, sedangkan bit-bit pada port sisanya tergabung dalam grup kontrol B. PPI 8255 juga dapat dioperasikan dalam 3 mode. Selengkapnya mengenai PPI 8255 dapat dilihat pada diskripsi dan konfigurasi di bawah ini :

Diskripsi fungsi pin 8255¹⁷⁾ :

PA7-PA0 terminal I/O port A

PB7-PB0 terminal I/O port B

PC7-PC0 terminal I/O port C

CS Chip Select (input, aktif low)

Bila CS diset "0" menunjukkan bahwa CPU sedang membaca data dari PPI 8255.

WR Write (input, aktif low)

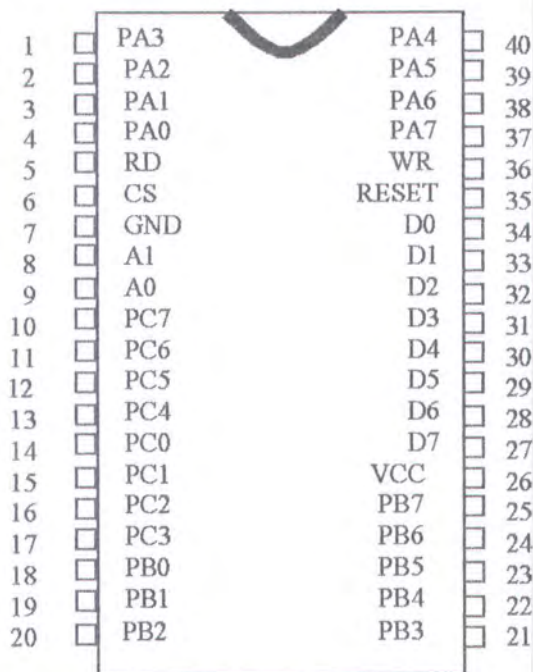
WR diset berlogika 0 menandakan bahwa CPU sedang menuliskan data ke PPI 8255.

A0 dan A1 Port select 0 dan port select 1 (input)

Kombinasi A0 dan A1 digunakan untuk memilih port mana yang bekerja.

+5V dan GND Terminal tegangan supply dan ground.

¹⁷⁾ Ibid. hal. 103



Gambar 2.11 Konfigurasi pin PPI 8255¹⁸⁾

Tabel 2.2 Operasi dasar PPI 8255¹⁹⁾

A1	A0	R	W	CS	Operasi yang dibentuk
0	0	0	1	0	READ : Port A ke Data Bus
0	1	0	1	0	READ : Port B ke Data Bus
1	0	0	1	0	READ : Port C ke Data Bus
0	0	1	0	0	WRITE : Data Bus ke Port A
0	1	1	0	0	WRITE :Data Bus ke Port B
1	0	1	0	0	WRITE : Data Bus ke Port C
1	1	1	0	0	WRITE : Data Bus ke Control Word Register
x	x	x	x	x	Data Bus 8255 berkondisi tri-state
1	1	0	1	0	Kondisi illegal
x	x	1	1	0	Data Bus 8255 berkondisi tri-state

¹⁸⁾ Ibid. hal. 103

¹⁹⁾ Ibid. hal. 104

PPI 8255 mengenal tiga mode operasi yakni :

- ♦ Mode 0 → Basic I/O
- ♦ Mode 1 → Strobed I/O
- ♦ Mode 2 → Bi-Directional Bus

Mode-mode operasi ini dapat dipilih dengan memberikan control word pada saat inisialisasi.

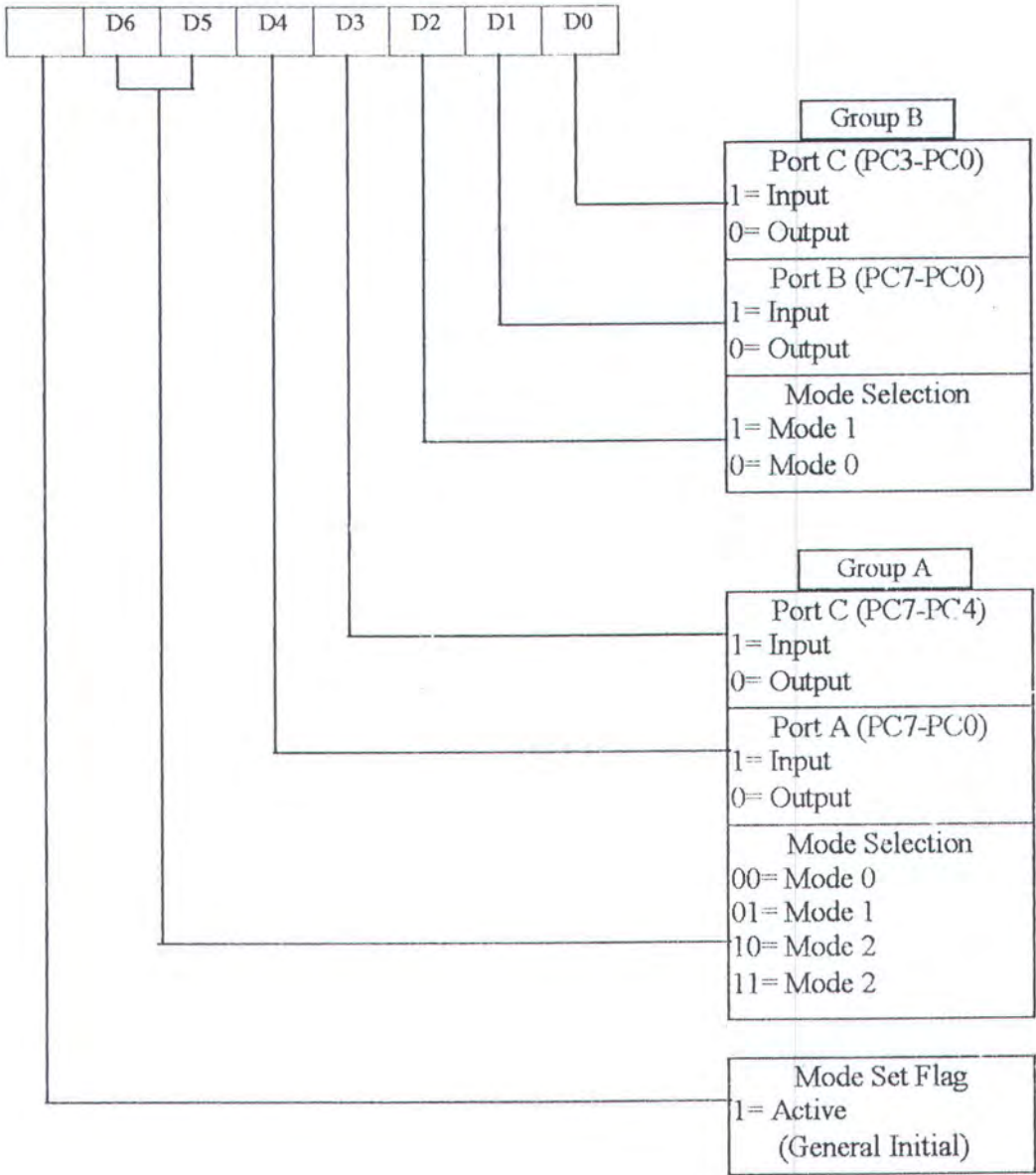
Tabel 2.3 Formasi sinyal untuk inisialisasi²⁰⁾

A1	A0	R	W	CS	Operasi yang dibentuk
1	1	1	0	0	WRITE : Data Bus ke Control Word Register

Control word yang diberikan cukup sekali pada saat awal PPI ini akan diaktifkan. Selain itu selama program sedang berjalan, PPI sewaktu-waktu dapat diubah mode pengoperasiannya dengan pemberian control word yang lain. Tersedianya fasilitas ini memungkinkan sebuah PPI untuk digunakan berbagai keperluan interfacing dengan program-program subrutin yang berbeda-beda.

Pada PPI 8255 bila input “reset” diaktifkan (1), semua port akan diset pada Input Mode (24 pin port menjadi high impedance). Setelah reset, otomatis PPI akan menggunakan semua port sebagai input port, tanpa perlu diinisialisasi (diberikan control word lagi).

²⁰⁾ Ibid. hal. 104

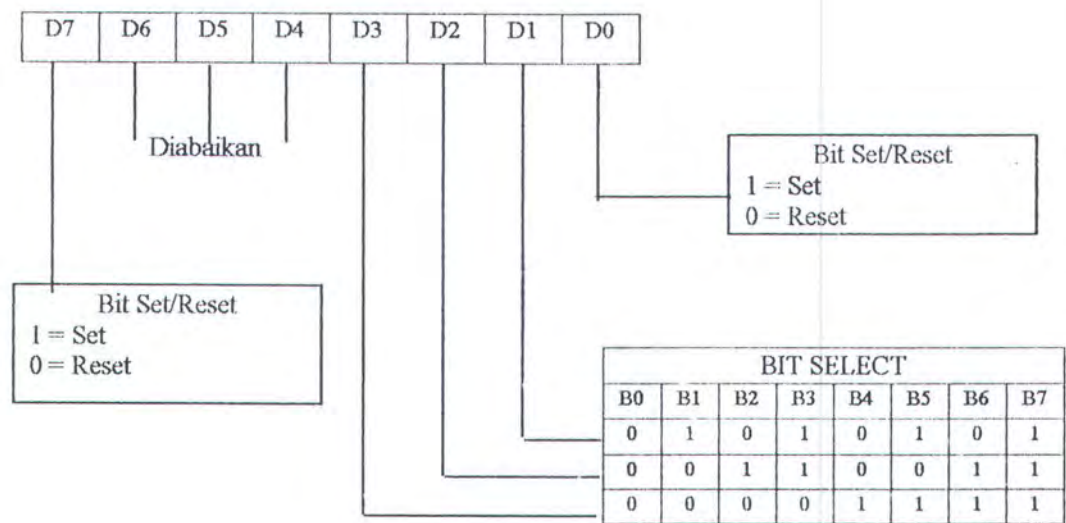


Gambar 2.12 Format Control Word PPI 8255²¹⁾

Bit D7 berfungsi untuk menunjukkan mode set dari control word. Bila D7 = “1” maka control word yang diberikan adalah untuk mengeset FLAG. Artinya set untuk inisialisasi terminal-terminal PPI 8255. Sedangkan bila berlogika “0”

²¹⁾ Ibid. hal. 105

maka control word yang diberikan adalah untuk langkah set/reset port C dengan format seperti pada gambar berikut ini :



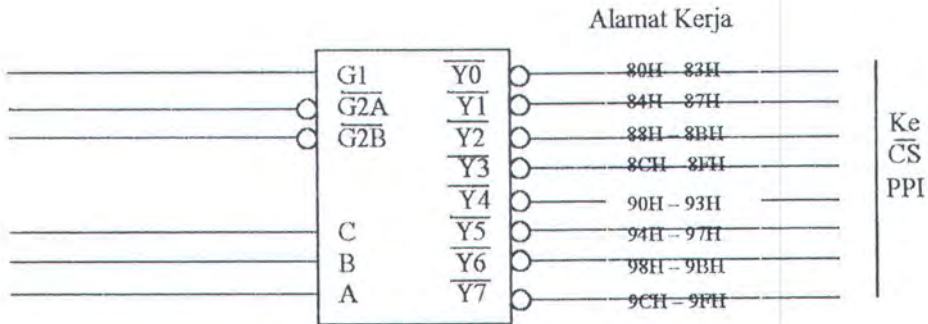
Gambar 2.13 Formasi Set/Reset PPI 8255²²⁾

Bit set/reset ini dapat sewaktu-waktu diberikan di dalam program agar dapat mengubah mode operasi port C sesuai dengan keinginan dari user.

2.8.2 Interfacing ke PPI 8255

Untuk memilih alamat kerja dari PPI 8255 dapat dilakukan dengan merangkai sebuah dekoder seperti di bawah ini, sehingga pengalamatan akan sesuai dengan pengaplikasian masing-masing portnya.

²²⁾ Ibid. hal. 106



Gambar 2.14 Rangkaian Dekoder dalam penentuan pengalaman PPI 8255²³⁾

2.9 Interfacing Piranti Input/Output ke CPU Z80

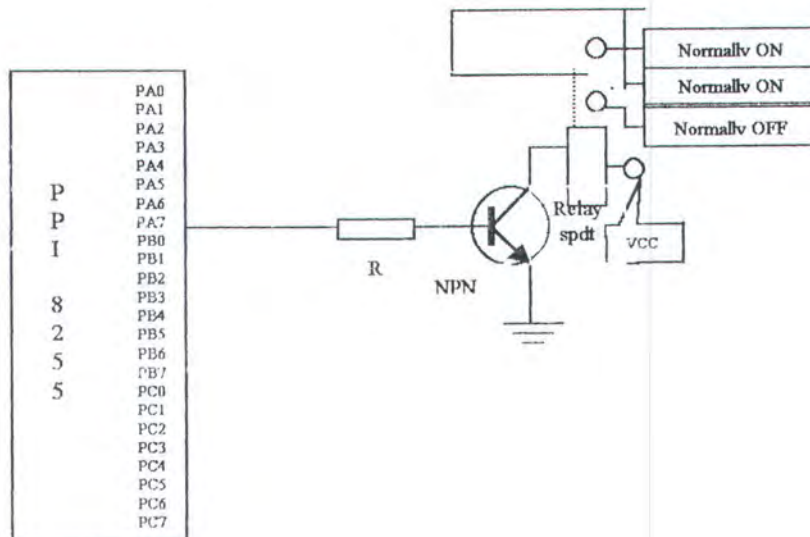
Piranti I/O dasar diasumsikan sebagai piranti rangkaian yang dihubungkan langsung ke sistem minimum dengan orientasi hubungan dan operasi tiap bit. Artinya tiap bit dalam susunan data input atau output 8 bit diasumsikan dapat berdiri sendiri atau berlaku *independent* terhadap bit-bit lainnya. Oleh karena itu piranti I/O dalam tinjauan orientasi tiap bit ini dapat dipelajari dan diaplikasikan dengan mendefinisikan dua keadaan yang dijadikan dasar perintah atau informasi yang juga identik dengan logika 0 dan logika 1.

²³⁾ Ibid. hal. 107

2.9.1 Interfacing ke Relay

Dalam mengemudikan arus yang besar dari suatu output port dapat diaplikasikan suatu cara yang efektif yakni dengan menggunakan driver relay.

Driver relay ditunjukkan pada gambar berikut ini :



Gambar 2.15 Rangkaian Driver Relay²⁴⁾

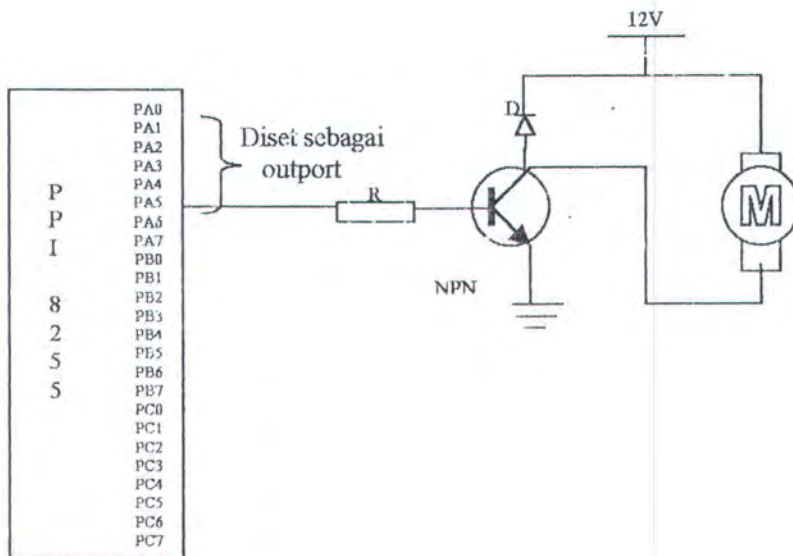
Dari gambar di atas terlihat bahwa dari sisi *driving* memerlukan arus yang relatif jauh lebih kecil dibandingkan arus yang melewati kontaktor relay. Sehingga dapat dilakukan pengontrolan arus-arus yang besar meskipun salah satu kelemahannya adalah timbulnya bunga api pada kontaktornya.

²⁴⁾ Endra Pitowarno, **Mikroprosesor dan Interface 2**, PES-SBY, Surabaya, 1995, hal 74

²⁵⁾ Ibid. hal. 76

2.9.2 Interfacing ke DC Motor

DC motor dengan *brush* dan lilitan pada sisi rotor serta menggunakan magnet tetap pada sisi statornya pada dasarnya merupakan beban yang harus dikontrol. Pada rangkaian pengontrol switching dari motor diaplikasikan pengaturan dari nilai nominal dari arus yang dibutuhkan oleh motor, switching dalam hal ini menggunakan transistor. Rangkaian driver relay untuk motor DC dapat dilihat pada gambar berikut ini :



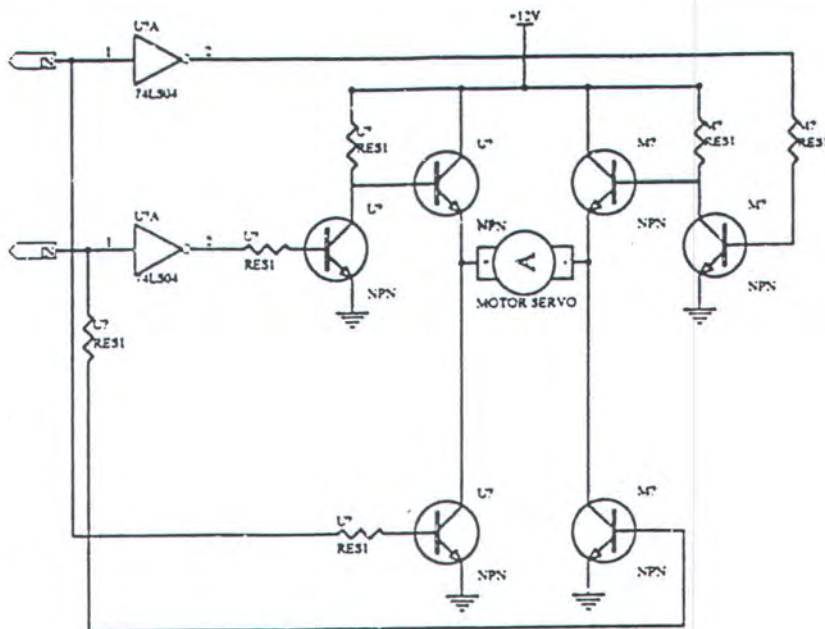
Gambar 2.16 Rangkaian Driver Relay Motor DC²⁵⁾

Pada permasalahan tertentu diinginkan untuk membalik arah putaran dari motor tersebut, salah satu pemecahan masalah tersebut adalah dengan menggunakan dua rangkaian relay (switching) untuk mengontrol sebuah motor. Sehingga dalam hal ini polaritas dari motor DC dapat diubah-ubah hanya dengan cara memberikan

²⁵⁾Ibid. hal. 76

arus trigger pada input rangkaian driver. Formasi yang digunakan untuk menjalankan motor adalah kombinasi bit 01 arah tertentu dan 10 untuk membalikkan arah dari bit sebelumnya.

Rangkaian dari driver relay pembalik arah putaran motor DC dapat dilihat pada gambar di bawah ini :



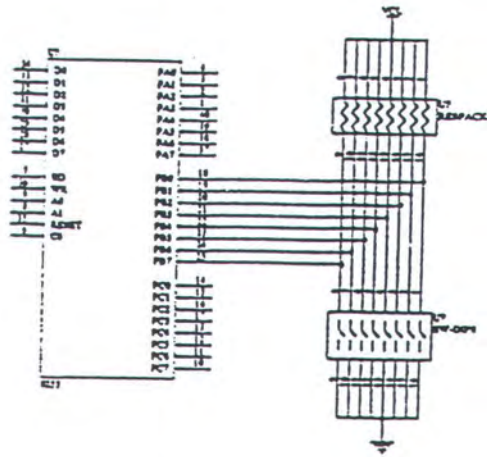
Gambar 2.17 Rangkaian driver pembalik arah motor DC²⁶⁾

²⁶⁾ Ibid. hal. 77

2.9.3 Interfacing ke Switch (on/off)

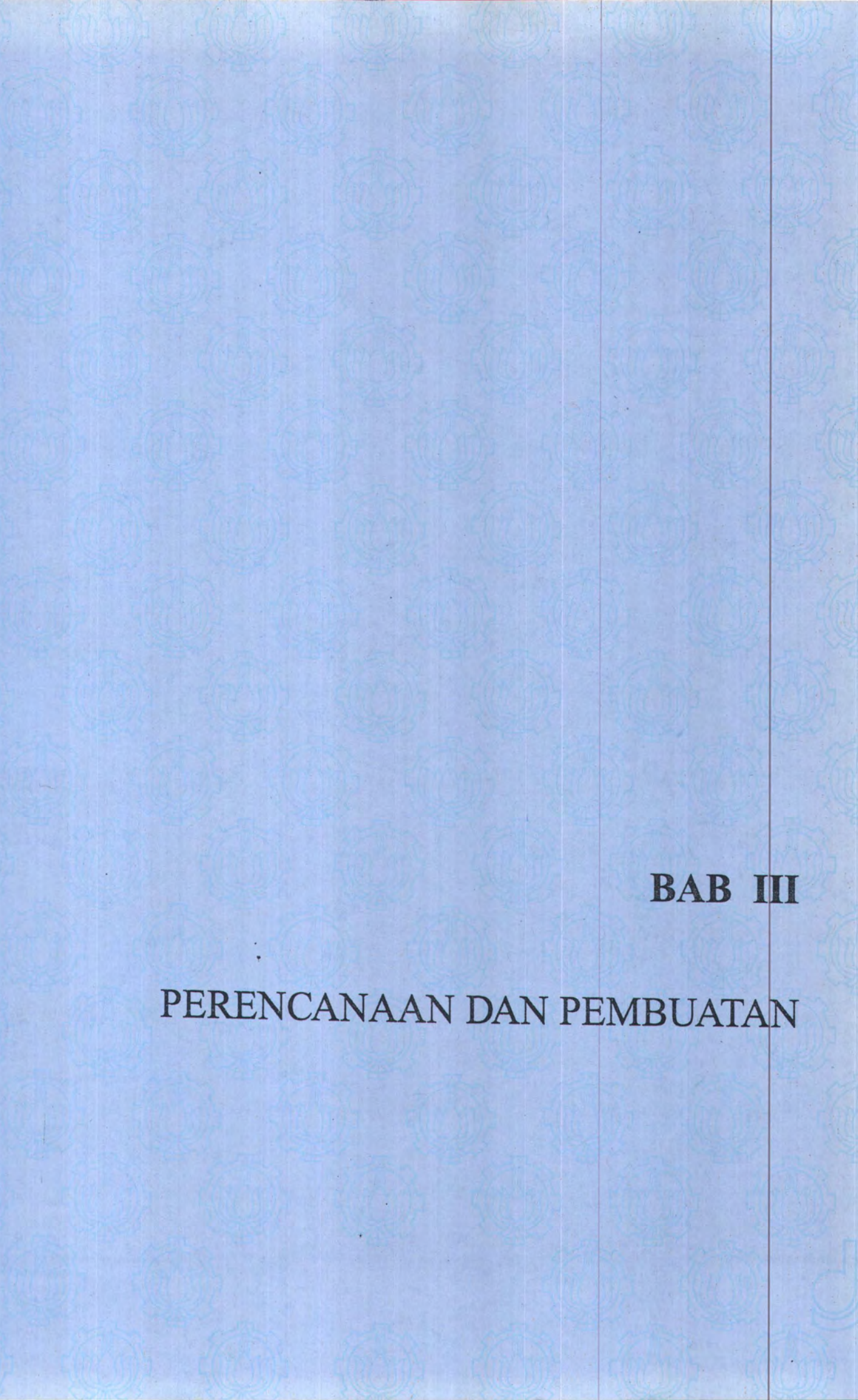
Komponen ini adalah peralatan input yang paling sederhana dan paling mendasar dalam rangkaian berbasis mikroprosesor, atau sistem digital non mikroprosesor yang lainnya. Dua keadaan yang dimilikinya dapat mudah didefinisikan sebagai dua keadaan dalam logika biner/digital. Push button switch, limit switch dan electronic switch (menggunakan solid state transistor atau IC) adalah termasuk dalam keluarga switch ini.

Sebagai contoh interfacing dasarnya diilustrasikan dalam gambar 2.17 di bawah ini. Tampak sebuah port input (port B) dari PPI 8255 digunakan sebagai terminal input. Sebuah *switch array* yang berjumlah delapan buah dihubungkan padanya. Dalam keadaan switch terbuka, logika inpunya adalah "1" (normally high). Dalam keadaan aplikasinya switch on/off ini digunakan secara meluas dengan manipulasi-manipulasi, baik dalam bentuk fisik maupun dalam sistem kerja switch itu sendiri, maupun teknik-teknik yang digunakan dalam pembacaan/scanning data input switch tersebut. Salah satu aplikasinya adalah penggunaan susunan switch button secara array atau matriks sebagai peralatan masukan data keyboard.



Gambar 2.18 Interfacing PPI 8255 ke Switch On/Off²⁷⁾

²⁷⁾ Ibid. hal. 63



BAB III

PERENCANAAN DAN PEMBUATAN

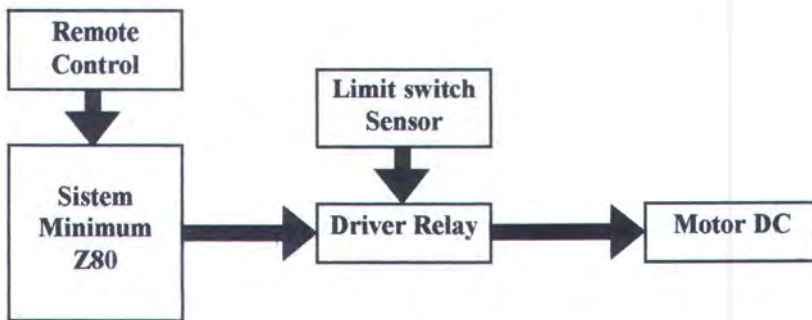
BAB III

PERENCANAAN DAN PEMBUATAN

Pada bab ini akan dibahas mengenai beberapa hal tentang perencanaan dan pembuatan proyek akhir ini. Dimana dalam perencanaan dan pembuatan ini akan yang akan menjadi acuan dalam pengujian yakni :

- Perencanaan sistem minimum Z80
- Perencanaan EPROM
- Perencanaan RAM
- Perencanaan Interfacing dengan PPI 8255
- Prinsip kerja semiotomatis dari sistem robot
- Mode kerja operator oriented
- Open loop controller
- Algoritma gerak dari robot
- Flowchart
- Cuplikan dari program

Secara garis besar, sistem secara keseluruhan dari robot dapat diilustrasikan dalam suatu diagram blok seperti gambar dibawah ini.



Gambar 3.1 Blok diagram dari Robot Merayap I

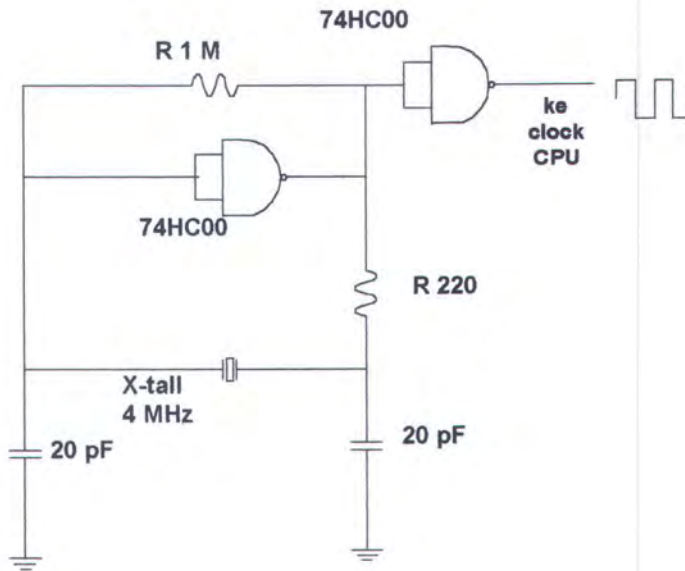
Sebagai komponen utama dari sistem CPU Z80 yakni EPROM, RAM dan 2 buah PPI 8255 yang membentuk suatu sistem minimum yang digunakan untuk mengontrol secara keseluruhan dari sistem robot merayap ini. Untuk gambar lebih lengkapnya dapat diamati pada lampiran C untuk rangkaian sistem kendali.

3.1 Perencanaan sistem minimum Z80

Sistem minimum yang digunakan terdiri dari piranti-piranti pendukung antara lain adalah Z80 CPU, EPROM 2764, RAM 6264, dua buah PPI 8255 dan beberapa komponen pendukung lainnya.

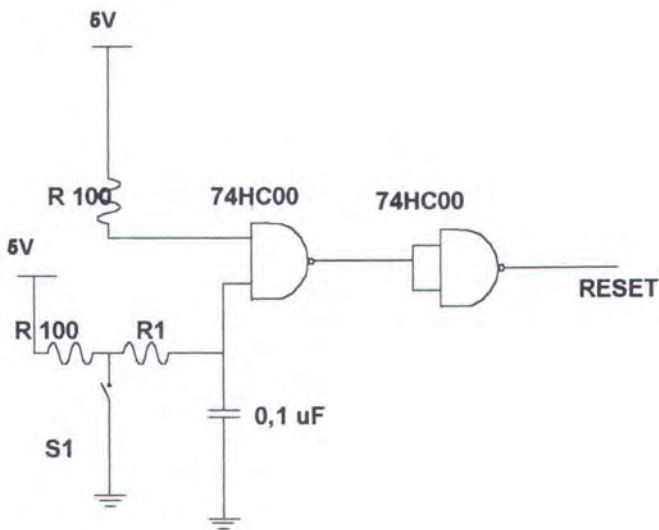
3.1.1 Rangkaian Osilator dan Reset

Rangkaian osilator digunakan untuk membangkitkan pulsa-pulsa persegi agar sistem minimum dapat bekerja. CPU Z80 membutuhkan clock dengan frekuensi 4 Mhz. Pulsa clock dihasilkan oleh rangkaian osilator seperti di bawah ini.



Gambar 3.2 Rangkaian Osilator Kristal 4 Mhz

Sedangkan rangkaian reset difungsikan agar running program dimulai lagi dari alamat awal, yakni 0000H. Rangkaian reset dirancang untuk bekerja secara reset power on. Hal ini dimaksudkan agar CPU diberikan waktu beberapa mikrodetik sebelum instruksi menuju ke alamat 0000H dilaksanakan, untuk memastikan bahwa semua bagian rangkaian sudah sampai pada kondisi pada saat CPU mulai bekerja. Bentuk rangkaian reset ditunjukkan pada rangkaian di bawah ini.



Gambar 3.3 Rangkaian Reset Power On

3.2 Perencanaan Interfacing Memori Ke CPU Z80

Pin-pin data pada CPU dihubungkan langsung ke pin-pin data pada RAM dan EPROM sesuai dengan urutannya yakni dari D0-D7. Kemudian pin alamat dari IC memori dihubungkan ke pin pada CPU sesuai dengan kapasitas memori masing-masing.

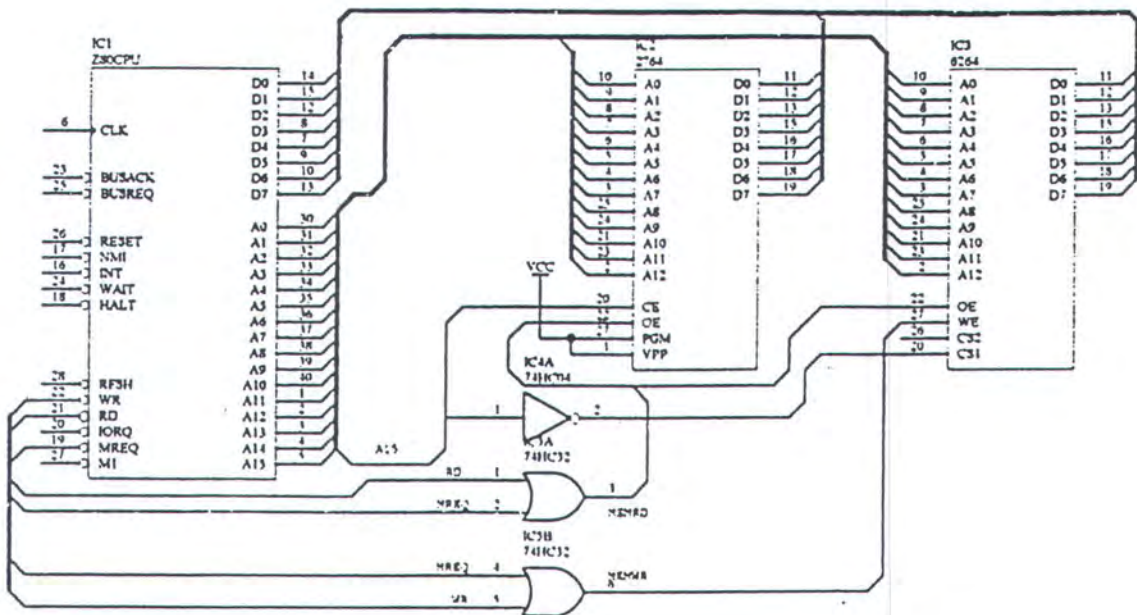
Untuk pengaktifan dari RAM dan EPROM maka terlebih dahulu kita melihat pengalamatan atau perencanaan dari memori mapping. Untuk EPROM 2764 dan RAM 6264 yang masing-masing berkapasitas 8 kilo byte, pemetaan alamat memorinya adalah sebagai berikut.

EPROM 2764	0000H
	1FFFH
RAM 6264	8000H
	9FFFH

Gambar 3.4 Pemetaan alamat EPROM 2764 dan RAM 6264

Jadi kemampuan prosesor Z80 untuk melakukan direct addressing (pengalamatan langsung) berkaitan dengan pengaktifan langsung dari pin alamat bit ke15 atau A15 yang secara langsung menentukan apakah itu pemetaan dari alamat RAM atau EPROM. Pin A15 dari CPU ini dihubungkan ke chip select atau chip enable dari masing-masing pin alamat dari IC memori.

Sehingga hubungan antara CPU Z80 dengan memori dapat ditunjukkan seperti pada gambar di bawah ini.



Gambar 3.5 Hubungan CPU Z80 dengan Sistem Memori

3.2.1 Perencanaan EPROM

Dari pemetaan di atas tampak bahwa EPROM harus aktif bila CPU melakukan pembacaan pada alamat antara 0000H sampai 1FFFH. oleh karena itu harus ada sinyal elektronik yang memberitahukan EPROM bahwa alamat yang dikeluarkan CPU mempunyai harga-harga dalam batas-batas tersebut. Untuk mengaktifkan EPROM maka pin CE (chip enable) harus diberi sinyal aktif low.

Data dari EPROM akan terbaca bila pin A15 berlogika '1' atau '0' setelah melewati gate not/inverter (IC 74HC04) guna pengidentifikasian bahwa alamat yang dituju memiliki bit ke15 bernilai 0, sedangkan pin alamat yang digunakan adalah sebanyak 12 jalur atau hingga bit ke 12, agar lebih jelas dapat diperhatikan keterangan dalam bentuk tabel di bawah ini.

Tabel 3.1 Tabel kapasitas memori EPROM 2764

Tipe EPROM	A12	A11-A0	Kapasitas
2764	0	000000000000	64 Kbit
	1	111111111111	8 KByte

Operasi pembacaan memori pada EPROM diindikasikan oleh adanya sinyal MREQ dan RD. Oleh karena itu kedua sinyal tersebut di-OR-kan dan digunakan sebagai masukan bagi pin 22 (CE) dari 2764, sehingga CE akan aktif dalam kondisi low. Berikut ini merupakan tabel kebenaran saat CPU menghubungi EPROM.

Tabel 3.2 Tabel kebenaran saat CPU menghubungi EPROM

MREQ	RD	A15	Keterangan
0	0	0	CPU membaca EPROM
1	X	X	Not Valid
X	1	X	Not Valid
X	X	1	Not Valid

3.2.2 Perencanaan RAM

Serupa dengan dengan perencanaan pada EPROM, pemetaan pada RAM juga harus aktif bila CPU melakukan pembacaan pada alamat antara 8000H sampai 9FFFH. oleh karena itu harus ada sinyal elektronik yang memberitahukan RAM bahwa alamat yang dikeluarkan CPU mempunyai harga-harga dalam batas-batas tersebut. Untuk mengaktifkan RAM maka pin CS (chip select) harus diberi sinyal aktif low.

Data dari RAM akan terbaca bila pin A15 berlogika 1 tanpa melewati gate not/inverter (IC 74HC04) yang merupakan kebalikan dari cara pembacaan pada EPROM, juga guna pengidentifikasian bahwa alamat yang dituju memiliki bit ke15 bernilai 1, sedangkan pin alamat yang digunakan adalah juga sebanyak 12 jalur atau hingga bit ke 12, agar lebih jelas dapat diperhatikan keterangan dalam bentuk tabel di bawah ini.

Tabel 3.3 Tabel kapasitas memori RAM 6264

Tipe RAM	A12	A11-A0	Kapasitas
6264	0	000000000000	64 Kbit
	1	111111111111	8 KByte

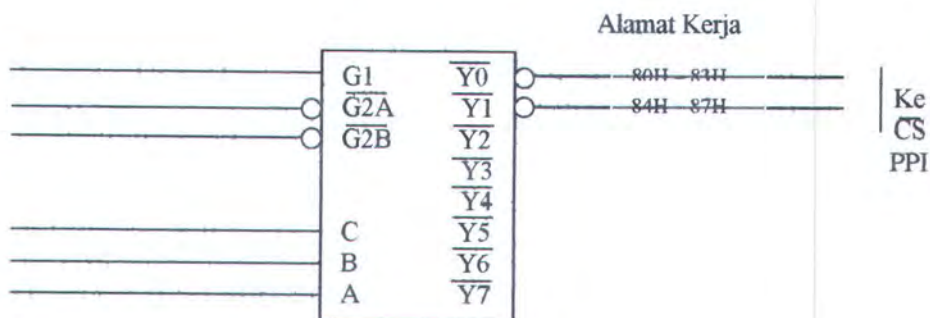
Operasi pembacaan dan penulisan memori pada RAM diindikasikan oleh adanya sinyal MREQ, RD atau WR yang aktif. Berikut ini merupakan tabel kebenaran saat CPU menghubungi RAM.

Tabel 3.4 Tabel kebenaran saat CPU menghubungi RAM

REQ	RD	WR	A15	Keterangan
0	0	1	1	CPU membaca RAM
0	1	0	1	CPU membaca RAM
1	X	X	X	Not Valid
X	X	X	1	Not Valid

3.3 Perencanaan Interfacing dengan PPI 8255

Untuk memilih alamat kerja yang sesuai dengan yang diinginkan, dapat dilakukan dengan merangkaikan sebuah dekoder. Dalam proyek robot merayap jenis dekoder yang digunakan adalah IC 74HC138 dan alamat dari PPI yang digunakan adalah pada output Y0 (80H-83H) untuk PPI1 dan output Y1 (84H-87H) untuk PPI2. Rangkaian dekoder dapat dilihat pada gambar di bawah ini.



Gambar 3.6 Rangkaian dekoder untuk pemilihan alamat kerja PPI 8255

Setiap port dari PPI1 (24 bit) pada proyek robot merayap ini diaplikasikan sebagai output ke 12 buah motor DC, sehingga jika masing-masing motor memerlukan 2 bit (untuk putar balik arah) maka akan tepat untuk penerapan interfasing pada motor DC tersebut. Sedangkan pada PPI2 port A digunakan untuk interfacing ke rangkaian remote control 8 bit dan 3 bit lagi untuk limit switch sebagai sensor posisi awal robot.

3.4 Prinsip kerja Semiotomatis

Prinsip kerja semiotomatis, dimaksudkan bahwa robot akan bekerja sebagai mana mestinya bila dipandu oleh seorang operator melalui suatu remote control, namun dalam sistem robot, setelah menerima data yang dikirim melalui remote maka prosesor akan mengolah data tersebut menjadi suatu gerakan otomatis. Dalam proyek robot ini kita ambil contoh pada bit 1 yang mengidentifikasi suatu gerakan maju, dimana dalam pergerakan maju tersebut robot secara otomatis melakukan pergerakan mengayunkan lengan atas ke kiri dan kanan secara bergantian dan diagonal, menggerakkan lengan bawah untuk setting posisi pelekatan dengan bidang permukaan serta proses pergerakan penghisapan pada bidang yang sedang dipijak oleh kaki robot. Jadi disini proses manual dijadikan sebagai acuan dalam melakukan proses otomatis.

3.5 Mode Operator Oriented

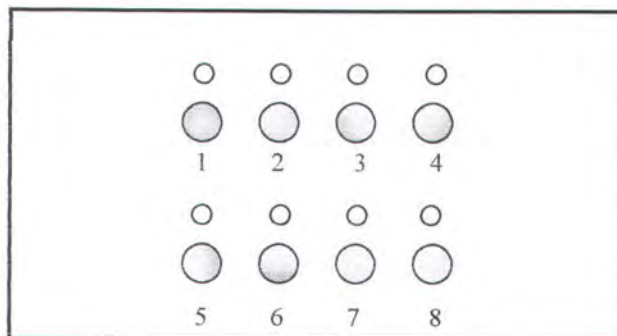
Karena mengaplikasikan mode kerja yang bersifat operator oriented, maka pada realisasinya diperlukan seorang operator yang berorientasi terhadap kinerja

dari robot. Tentu saja dalam prosesnya operator memerlukan suatu instrumen yang berfungsi sebagai penghubung dari acuan atau orientasinya terhadap sistem yang akan dikendalikannya. Dalam hal ini pada proyek robot merayap akan diaplikasikan suatu remote control dengan menggunakan media kabel sebagai perantara antara operator terhadap sistem robot secara keseluruhan.

Pada struktur remote 8 bit seperti pada lampiran D yang akan digunakan sebagai acuan terhadap beberapa gerakan utama dari robot memiliki keterangan mengenai bit-bit/tombol-tombol sebagai berikut :

1. Gerak awal
2. Gerakan maju
3. Gerakan mundur
4. Geser kanan
5. Geser kiri
6. Berhenti
7. Gerak lepaskan diri

Sedangkan rangkaiannya dapat dilihat dibawah ini :



Gambar 3.7 Rangkaian remote control

3.6 Open loop Controller

Kontroler yang digunakan bersifat openloop artinya bahwa dalam aplikasi dari program kontrol ditiadakan adanya feedback atau sensor-sensor. Namun dalam perencanaan pada proyek robot ini akan diaplikasikan sensor-sensor pembatas dari gerakan, baik itu gerakan mengayunkan lengan atas, menekan lengan bawah ataupun pada lengan penghisap. Rangkaian limit switch dihubungkan langsung pada rangkaian driver motor, dalam arti bahwa jika pergerakan dari robot telah memenuhi syarat maka limit switch akan tertekan dan secara otomatis rangkaian driver motor akan terputus dan motor tak akan bekerja. Kontroler semacam ini akan diterapkan pada seluruh rangkaian motor. Untuk lebih jelasnya dapat dilihat rangkaian driver motor pada lampiran C.

Sehingga akan dibahas lebih lanjut dalam bab perencanaan ini tentang bagaimana pengaturan dari pewaktuan dari pergerakan robot hingga mencapai limit switch dengan pewaktuan pada program sehingga terdapat sinkronisasi atau ketepatan waktu yang dibutuhkan untuk proses pergerakan tersebut.

3.7 Algoritma Gerakan Robot

Algoritma merupakan dasar dari suatu pemrograman dimana algoritma tersebut akan mengarahkan bagaimana pembuatan dari program. Untuk itu pada sistem robot merayap dengan kaki empat ini juga dibutuhkan suatu algoritma. Algoritma yang direncanakan adalah gerakan maju, mundur, geser kanan, geser kiri, berhenti, secara lebih khusus lagi terdapat gerakan seperti menekan

permukaan, menghisap permukaan dan mengangkat lengan. Untuk lebih jelasnya algoritma tersebut adalah sebagai berikut ;

♦ Gerakan Awal

Merupakan gerakan pertama kali sebelum robot melakukan pergerakan merayap, yakni berupa pergerakan menekan dari lengan bawah terhadap permukaan kemudian diikuti oleh proses penghisapan oleh lengan hisap pada permukaan yang dipijak. Keseluruhan proses ini dilakukan oleh keempat kaki dari robot.

♦ Gerakan Maju

Pada hakekatnya hanya terdiri dari tiga tiga macam gerakan utama, tetapi realisasi gerakan secara keseluruhan maka akan dilakukan berturut-turut langkah gerakan seperti berikut ini :

- I. Gerakan mengnonaktifkan lengan hisap pada kaki 1 dan 4.
- II. Gerakan mengangkat oleh lengan bawah pada kaki 1 dan 4.
- III. Gerakan rotasi $+30^\circ$ pada kaki 1 dan -30° pada kaki 4 dengan cara mengaktifkan motor pada kaki 2 dan 3 (arah putar motor 2 berlawanan dengan motor 3).
- IV. Gerakan menekan oleh lengan bawah pada kaki 1 dan 4.
- V. Gerakan menghisap oleh lengan hisap pada kaki 1 dan 4 pada permukaan yang dipijak.
- VI. Gerakan mengnonaktifkan lengan hisap pada kaki 2 dan 3.
- VII. Gerakan mengangkat oleh lengan bawah pada kaki 2 dan 3.

- viii. Gerakan rotasi $+30^\circ$ pada kaki 2 dan -30° pada kaki 3 dengan cara mengaktifkan motor pada kaki 1 dan 4 (arah putar motor 1 berlawanan dengan motor 4).
- ix. Gerakan menekan oleh lengan bawah pada kaki 2 dan 3.
- x. Gerakan menghisap oleh lengan hisap pada kaki 2 dan 3 pada permukaan yang dipijak.

♦ Gerakan Mundur

Sama halnya dengan prosedur gerakan maju seperti di atas hanya saja terjadi perubahan pada gerakan lengan atas, dimana pergerakan rotasi lengan atas bersifat berbalikan arah, sehingga akan didapatkan langkah atau pergerakan mundur.

♦ Gerakan Geser kanan dan kiri

Pada pergerakan menggeser pergerakan dari lengan atas tidak terjadi secara diagonal antara dua kaki seperti pada pergerakan maju dan mundur di atas disebutkan bahwa secara bergantian yang aktif adalah antara kaki 1 dan 4 dengan kaki 2 dan 3, melainkan terjadi pergerakan antara kaki yang sejajar (vertikal) yakni kaki 1 dan 3 secara bergantian dengan kaki 2 dan 4. Lalu kemudian untuk proses pergeseran pada lengan bawah dan lengan hisap adalah sama seperti pada pergerakan maju dan mundur. Hanya saja dalam pergerakan baik itu geser kanan maupun kiri terdapat gerakan rotasi oleh lengan atas sebesar 15° , dalam permasalahan ini akan dipecahkan dengan pewaktuan yang bernilai setengah dari pewaktuan dalam proses

pergerakan rotasi 30° dan juga berdasarkan dengan asumsi bahwa kecepatan dan konversi gear pada gearbox adalah sama untuk keempat kaki tersebut..

◆ Berhenti (tidak melakukan pergerakan)

Dalam proses pemberhentian pergerakan dari robot kapanpun juga dapat dilakukan. Cukup dengan memberikan masukan dari remote yang menandakan untuk berhenti maka lewat mikroprosesor akan mengeluarkan output berlogika '0' pada seluruh output peripheralnya yang berkaitan dengan pergerakan dari motor.

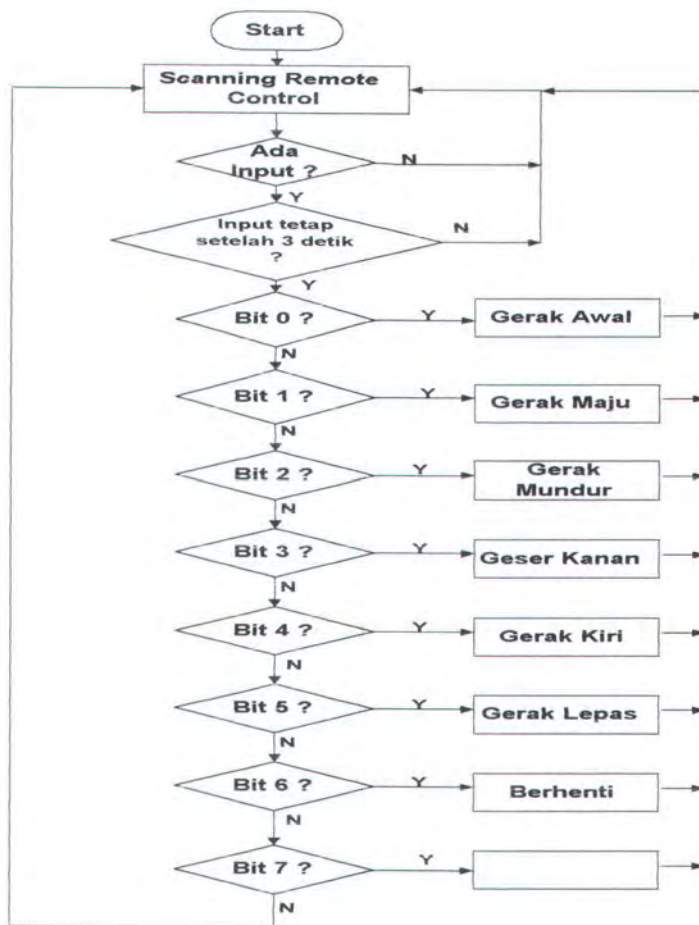
◆ Gerakan Melepas

Pada pergerakan melepas dimaksudkan bahwa keseluruhan dari proses merayap dari robot adalah sudah selesai, dimana setelah operator memberikan masukan untuk lepas, maka otomatis seluruh proses pergerakan dari lengan bawah dan lengan atas akan berhenti, tetapi perkecualian pada lengan hisap. Lengan hisap untuk keempat kaki akan melakukan pergerakan atau proses melepaskan hisapan pada permukaan yang dipijak, sehingga setelah itu robot akan dalam posisi tidak merayap lagi.

3.8 Flowchart

Flowchart atau alur kerja yang dibuat adalah berdasarkan dari algoritma diatas, sehingga dalam pembuatan program akan terarahkan dengan baik. Hanya saja dalam program nantinya tidak hanya menyinggung soal mengenai algoritma

pergerakan dari robot, tetapi juga akan diamati bagaimana alur dari proses pembacaan masukan, ataupun bagaimana proses pengecekan apakah masukan itu berubah selama proses untuk pergerakan dari acuan sebelumnya masih berlangsung. Maka dari itu untuk lebih jelasnya, selebihnya akan tergambar dalam alur kerja atau flowchart berikut ini :



Gambar 3.8 Flowchart Gerakan Robot

Keterangan :

Konfigurasi dari input port PPIP_B_2 adalah sebagai berikut:

Bit 0 adalah masukan untuk acuan dalam bergerak awal.

Bit 1 adalah masukan untuk acuan dalam bergerak maju.

Bit 2 adalah masukan untuk acuan dalam bergerak mundur.

Bit 3 adalah masukan untuk acuan dalam bergeser kanan.

Bit 4 adalah masukan untuk acuan dalam bergeser kiri.

Bit 5 adalah masukan untuk acuan dalam berhenti.

Bit 6 belum terpakai.

Bit 7 adalah masukan untuk acuan dalam melakukan gerakan lepas.

3.9 Cuplikan Program

Berikut ini adalah cuplikan program dari program utama pada listing program utama sistem kendali robot yang dapat dilihat pada lampiran B. Listing program berikut merupakan salah satu dari beberapa rutin yang ada, yakni khusus untuk rutin pergerakan maju dari robot.

```

RUTIN_MAJU :
    push af
    ld a, NONAKTIF_LH_1_4
    out(PPIP_C_1), a
    call DELAY_LH
    ld a, ANGKAT_LB_1_4
    out(PPIPB_1), a
    call DELAY_LB
    ld a, MAJU_LA_1_4
    out(PPIPA_1), a
    call DELAY_LA
    ld a, TEKAN_LB_1_4
    out(PPIPB_1), a
    call DELAY_LB

```

```

ld a,AKTIF_LH_1_4
out(PPIPC_1),a
call DELAY_LH
ld a,NONAKTIF_LH_2_3
out(PPIPC_1),a
call DELAY_LH
ld a,ANGKAT_LB_2_3
out(PPIPB_1),a
call DELAY_LB
ld a,MAJU_LA_2_3
out(PPIPA_1),a
call DELAY_LA
ld a,TEKAN_LB_2_3
out(PPIPB_1),a
call DELAY_LB
ld a,AKTIF_LH_2_3
out(PPIPC_1),a
call DELAY_LH
pop af
ret

```

Dimana keterangan perihal listing program di atas adalah :

- LH = Lengan Hisap (penghisap)
- LB = Lengan Bawah
- LA = Lengan Atas
- DELAY_* = Delay proses untuk pergerakan lengan yang bersangkutan
- Konfigurasi dari port bit pada PPI_1 :

Port A adalah untuk aplikasi dari motor pada Lengan Atas, dengan konfigurasi bit ;

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
Kaki 1		Kaki 2		Kaki 3		Kaki 4	

Begitu pula untuk port B dan port C dari PPI_1, dimana port B diaplikasikan pada motor untuk lengan bawah (LB) dan port C diaplikasikan pada motor

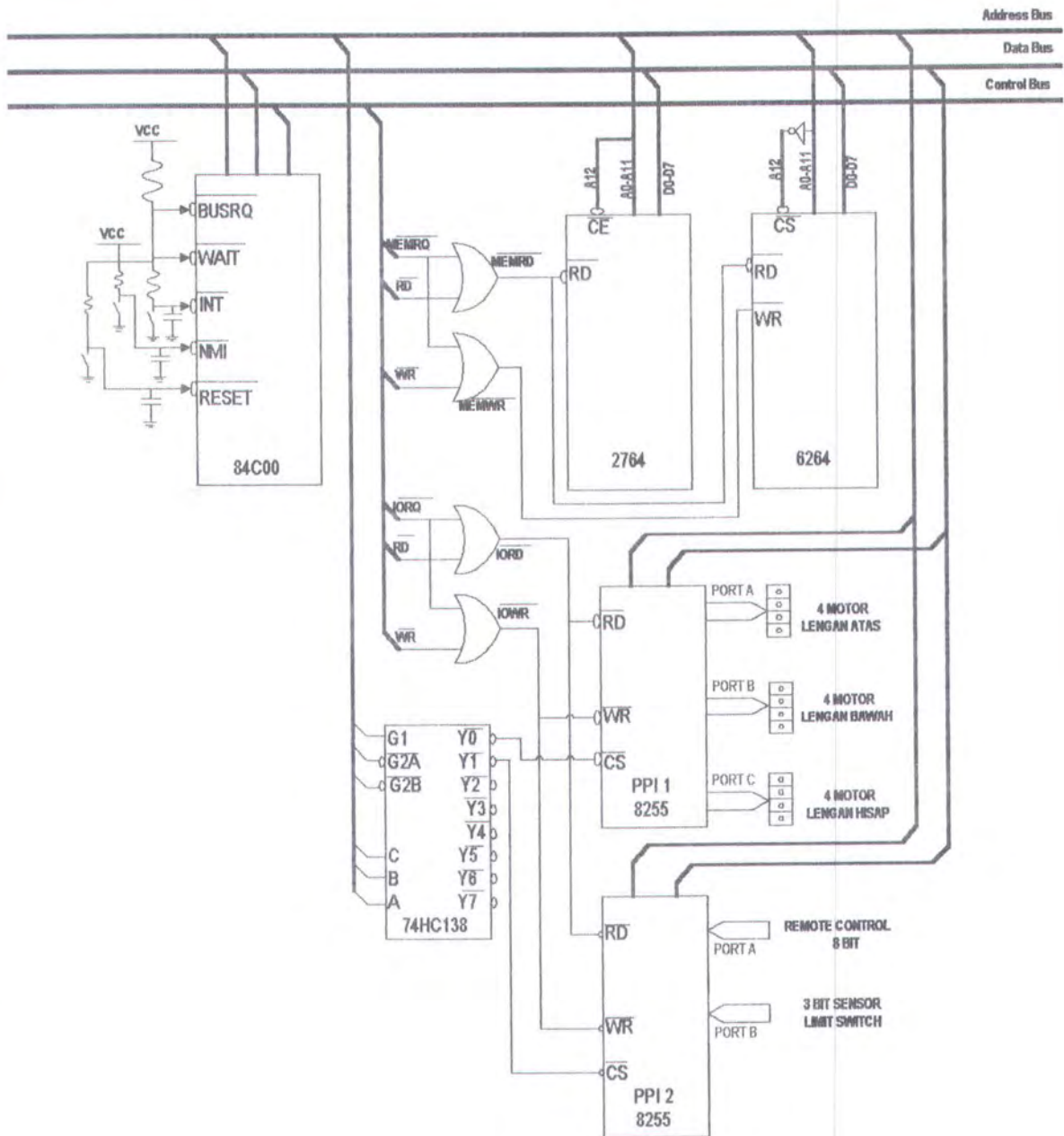
untuk lengan hisap (LH) dengan konfigurasi port bit yang sama dengan konfigurasi port bit seperti pada port A di atas.

➤ Berikut adalah urutan dari pergerakan maju :

1. Dari posisi awal (semua kaki sejajar) atau dari posisi digonal (kaki tak sejajar/memotong), akan dimulai dengan pengnonaktifan dari lengan hisap dari kaki 1 dan kaki 4 (kaki yang akan melakukan pergerakan) dengan pewaktuan sebesar yang telah ditentukan untuk proses tersebut (DELAY_LH).
2. Setelah itu lengan bawah pada kaki 1 dan kaki 4 akan melakukan pergerakan ke atas guna mengangkat lengan hisap (agar pada saat merayap, lengan hisap tidak terseret pada bidang permukaan pijakan) dengan pewaktuan sebesar yang telah ditentukan untuk proses tersebut (DELAY_LB).
3. Berikutnya lengan atas pada kaki 2 dan kaki 3 akan melakukan pergerakan rotasi guna memutar atau memindahkan kaki 1 dan 4 sekitar 30 derajat dengan pewaktuan yang telah ditentukan untuk proses tersebut (DELAY_LA).
4. Proses berikutnya adalah dengan menempelkan kembali lengan bawah pada kaki 1 dan kaki 4 (dengan membalik polaritas atau konfigurasi bit).
5. Terakhir adalah melakukan proses penghisapan oleh lengan hisap pada kaki 1 dan kaki 4.
6. Robot telah melakukan pergerakan maju dengan cara melangkahkan kaki secara diagonal seperti halnya pada binatang merayap lainnya.

3.10 Hubungan Sistem Minimum secara keseluruhan

Sistem minimum yang telah melibatkan piranti pendukung beserta seluruh interfacingnya dengan semua peripheralnya dapat dilihat pada gambar berikut



Gambar 3.9 Hubungan Sistem Minimum menggunakan Z80, memori 6264 & 2764, dua buah PPI serta peripheral penunjang lainnya

BAB IV

PENGUJIAN DAN ANALISA

BAB IV

PENGUJIAN DAN ANALISA

4.1 Tujuan

Tujuan dari diadakannya pengujian peralatan ini adalah :

- Mendapatkan data dari alat/sistem yang telah kita buat sehingga kita mengetahui apakah sistem kita tersebut sudah sesuai atau belum dengan perencanaan.
- Membandingkan hasil yang telah diperoleh dalam pengujian dengan yang terdapat dalam teori dan sekaligus mencari alternatif pemecahan masalah apabila data tersebut tidak sesuai.

4.2 Pengujian Modul Sistem Minimum Z80

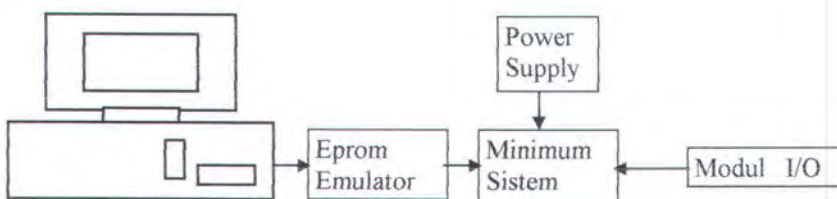
Pengujian sistem minimum ini adalah untuk mengetahui apakah modul ini dapat bekerja dengan baik sesuai yang dikehendaki. Untuk itu digunakan beberapa port pengujian antara lain pengujian port-port pada modul sistem minimum dan port pada PPI 8255 yang terdiri dari port A, B dan port C.

Untuk itu maka bisa dibuat program kemudi yang sederhana untuk penyalan LED pada output port secara rotasi dengan nyala sebuah LED pada setiap saat menurut selang waktu tertentu. Rotasi ini dapat diatur dengan menggunakan perintah RRCA ataupun RLCA untuk rotasi ke kanan atau ke kiri.

Seperangkat alat yang digunakan sebagai penunjang dalam pengujian ini adalah sebagai berikut :

- Seperangkat komputer
- EPROM Emulator
- Kabel Centronix
- PPI Card
- DC Regulated Power Supply +5V
- FZ80 I/O
- Rangkaian output LED
- Kabel secukupnya

Pemasukan program ke sistem minimum dapat dilakukan dengan menggunakan EPROM EMULATOR yang sengaja digunakan untuk melakukan pengujian program ke sistem minimum secara tepat dan lebih menghemat lifetime dari EPROM. Untuk mendapat gambaran dari sistem minimum dapat dilihat pada lampiran D. Dari program yang dibuat, apabila program dijalankan semua port akan menghasilkan lampu LED yang mengalami rotasi ke kanan, berarti port-port PPI dalam keadaan baik.



Gambar 4.1 Rangkaian Pengujian Port PPI dan Sistem Minimum

Berikut adalah prosedur dari pengujian :

- ❑ Menghubungkan kabel centronix dengan port PPI yang telah ditancapkan pada slot komputer.
- ❑ Menghubungkan soket EPROM Emulator pada soket EPROM sistem minimum Z80, menghubungkan pula power supply dari sistem minimum dan port dengan modul FZ80 I/O.
- ❑ Memasang dan memeriksa kelengkapan komponen dari sistem minimum.
- ❑ Men'download'kan program kemudi dari komputer menuju EPROM Emulator.
- ❑ Menghidupkan DC Power Supply.
- ❑ Me'reset' lalu running program dari sistem minimum.

Berikut ini adalah program pengujian yang digunakan :

```

PPIPA_1      EQU 80H      ;alamat kerja PPI
PPIPB_1      EQU 81H
PPIPC_1      EQU 82H
PPICW_1      EQU 83H
PPIIN_1      EQU 80H      ;PA-out,PB-out,PC-out
PPIPA_2      EQU 84H
PPIPB_2      EQU 85H
PPIPC_2      EQU 86H
PPICW_2      EQU 87H
PPIIN_2      EQU 83H      ;PA-in,PB-in,PC-out
;
org 0000h
;
        ld sp,0a000h
        call PPIINIT
        jp START
;
org 0038h
reti
;

```



```
org 0066h
```

```
retn
```

```
;
```

```
PPIINIT :
```

```
ld b,0
```

```
INITLOOP1 : ld a,PPIIN_1
```

```
out (PPICW_1),a
```

```
djnz INITLOOP1
```

```
ld b,0
```

```
;
```

```
INITLOOP2 : ld a,PPIIN_2
```

```
out (PPICW_2),a
```

```
djnz INITLOOP2
```

```
;
```

```
org 0100h
```

```
START : ld a,00000001B
```

```
ULANG : out (PPIPA_1),a
```

```
out (PPIPB_1),a
```

```
out (PPIPC_1),a
```

```
call TIMER
```

```
rlca
```

```
jr ULANG
```

```
;
```

```
;
```

```
TIMER :
```

```
push de
```

```
push af
```

```
ld d,00h
```

```
loop_0: ld e,06h
```

```
loop_1: ld a,00h
```

```
loop_2: dec a
```

```
jr nz,loop_2
```

```
dec e
```

```
jr nz,loop_1
```

```
dec d
```

```
jr nz,loop_0
```

```
pop af
```

```
pop de
```

```
ret
```

```
;
```

Analisa

Dari program kemudi di atas kita dapat melihat bahwa pada sistem minimum yang digunakan PPI yang pertama seluruhnya diset sebagai output port sedangkan pada PPI yang kedua port A dan port B adalah sebagai input port lalu port C nya adalah untuk output port. Konfigurasi dari inisialisasi ini akan diaplikasikan pada program sistem kendali dari robot merayap. Output dari nyala LED pada modul I/O pertama kali pada LED 00 seterusnya bergerak ke kiri dan hal ini dapat dilihat pada port A, port B maupun pada port C pada PPI_1.

Pada dasarnya output port pada PPI_1 tersebut akan diaplikasikan pada keluaran untuk driver dari motor DC, yang nantinya akan digunakan sebagai aktuator dari pergerakan robot.

Sekarang akan diuji kompatibilitas dari PPI_2 yang nantinya akan diaplikasikan sebagai input port terhadap masukan dari remote control sebagai acuan pergerakan dari robot. Pada penerapannya hanya digunakan 8 bit dari port A. Dengan inisialisasi halaman nol yang sama maka berikut adalah main program dari program uji yang dimaksud :

PPIPA_1	EQU 80H	;alamat kerja PPI
PPIPB_1	EQU 81H	
PPIPC_1	EQU 82H	
PPICW_1	EQU 83H	
PPIIN_1	EQU 80H	;PA-out,PB-out,PC-out
PPIPA_2	EQU 84H	
PPIPB_2	EQU 85H	
PPIPC_2	EQU 86H	
PPICW_2	EQU 87H	
PPIIN_2	EQU 83H	;PA-in,PB-in,PC-out

;

```

org 0000h
;
        ld sp,0a000h
        call PPIINIT
        jp START
;
org 0038h
reti
;
org 0066h
retn
;
PPIINIT :
        ld b,0
        INITLOOP1 : ld a,PPICW_1
                    out (PPIIN_1),a
                    djnz INITLOOP1

        ld b,0
        ;
        INITLOOP2 : ld a,PPICW_2
                    out (PPIIN_2),a
                    djnz INITLOOP2
;
org 0100h
ULANG :   in a,(PPIPA_2)
        out (PPIPA_1),a
        out (PPIPB_1),a
        out (PPIPC_1),a
        jr ULANG
;

```

Pada program di atas akan menghidupkan setiap LED pada FZ80 I/O baik itu pada port A, port B maupun port C dari PPI_1 hanya dan bila ada masukan dari input port yakni port A dari PPI_2 tentunya dengan konfigurasi bit yang bersesuaian.

4.3 Pengujian Sistem Gerak

Tujuan :

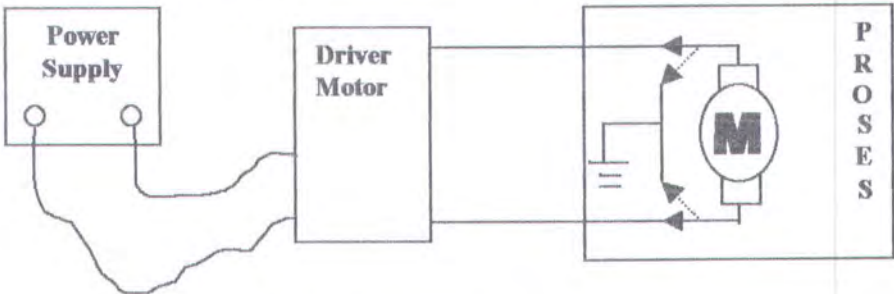
Menguji kebenaran dari pemikiran terhadap pergerakan robot yang realisasikan oleh kaki-kaki robot. Sekaligus menguji pewaktuan terhadap setiap proses yang dilakukan oleh masing-masing lengan dari kaki 1 hingga kaki 4 dari robot tersebut.

Dikarenakan sensor limit switch yang digunakan tidak diaplikasikan sebagai masukan pada port PPI dengan kekhawatiran bahwa jika saja terjadi bahwa salah satu dari sensor limit switch yang tidak berfungsi, maka akan dapat mengacaukan sekuensial dari program kemudi ataupun dapat merusak sistem mekanik untuk pergerakan yang bersangkutan. Oleh karena itu rangkaian sensor limit switch tersebut dibuat sedemikian rupa sehingga dapat dikatakan bersifat open loop terhadap rangkaian driver motor.

Peralatan :

- ☐ DC Regulator Power Supply
- ☐ Rangkaian Driver Motor DC
- ☐ Limit Switch
- ☐ Stopwatch
- ☐ Kabel secukupnya

Rangkaian Pengujian :



Gambar 4.2 Rangkaian Pengujian Driver Relay dan Limit Switch

Hasil Pengujian Pewaktuan

Tabel 4.1 Pewaktuan dari masing-masing proses pergerakan

Driver Motor		PROSES PENGHISAPAN	PROSES LENGAN BAWAH	PROSES LENGAN ATAS
0	0	-	-	-
0*	1*	32 detik (hisap)	7 detik	25 detik
1	0	30 detik (lepas)	7 detik	25 detik
1	1	-	-	-

Catatan *, transistor triggered pada driver motor
**, transistor nontriggered pada driver motor

Keterangan :

Pada hasil pengujian di atas dapat dilihat bahwa pada proses penghisapan maupun pelepasan (tidak penghisapan) dengan menggunakan motor 24 Volt DC dari keempat kaki robot diperoleh data bahwa proses tersebut memerlukan waktu sekitar 32 detik, sedangkan untuk proses pergerakan dari lengan bawah

memerlukan waktu sekitar 7 detik, lalu untuk proses pergerakan dari lengan atas memerlukan waktu sekitar 25 detik (proses pergerakan dibatasi oleh limit switch-limit switch agar dapat berlangsung dengan penuh).

Jadi dari data ini dapat dihubungkan ke software atau program kemudi dari sistem gerak, dimana dikarenakan sistem sensor limit switch bersifat openloop terhadap driver motor (tidak ada feedback ke prosesor) maka dengan pewaktuan yang presisi untuk masing-masing proses pergerakan yang bersangkutan akan didapatkan sinkronisasi antara pewaktuan secara software dengan pewaktuan pada proses-proses pergerakan tersebut. Pada program kemudi perlu diamati bahwa pewaktuan yang direncanakan pada subrutin DELAY (baik itu untuk lengan atas DELAY_LA, untuk lengan bawah DELAY_LB ataupun untuk lengan hisap DELAY_LH) minimal harus sama dengan pewaktuan pada proses pergerakan, dengan toleransi dilebihkan beberapa detik agar tetap dapat dijaga kesempurnaan dari proses pergerakan.

Hasil Pengukuran Arus

Tabel 4.2 Pengukuran arus masing-masing motor

MOTOR	Arus rata-rata (A) Saat Stabil
Lengan Atas **	1,2
Lengan Bawah *	0,7
Lengan Hisap *	0,7

Catatan *, Motor DC (wiper), 12 Volt

**, Motor DC 24 volt

Analisa :

Dari data yang diperoleh merupakan rata-rata pengukuran dari keempat kaki dari robot dengan beban yang terhubung dengan motor-motor tersebut. Nilai arus tersebut bersifat konstan untuk beban yang berbeda, yakni beban berupa penarikan lengan hisap pada *suction cup* maupun pada pengangkatan lengan bawah.

4.4 Pengujian Program Kemudi

Tujuan :

Dalam hal ini pengujian ditujukan pada pembuktian kebenaran dari respon program kemudi yang direalisasikan pada pergerakan aktuator-aktuator.

Sesuai dengan program kemudi pada lampiran listing program serta juga telah dijelaskan pada bab sebelumnya, maka akan dilakukan pengujian terhadap program kemudi tersebut, dimana disini respon dari program akan ditunjukkan pada output berupa rangkaian LED.

Port A dari PPI1 adalah untuk aplikasi dari motor pada Lengan Atas, dengan konfigurasi bit ;

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
Kaki 1		Kaki 2		Kaki 3		Kaki 4	

Begitu pula untuk port B yang diaplikasikan pada Lengan Bawah serta port C yang diaplikasikan untuk Lengan Hisap.

Berikut ini adalah urutan dari proses pembelajaran dari program :

1. Program akan melakukan proses scanning terhadap adanya masukan. Pada pengujian diambil contoh dengan memasukkan input pada bit 1, yakni untuk melakukan pergerakan maju.
2. Konfigurasi pertama adalah untuk menonaktifkan Lengan Hisap (port C) pada kaki1 dan kaki 4 untuk selang waktu sebesar waktu tunda untuk lengan hisap (DELAY_LH).

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
Kaki 1		Kaki 2		Kaki 3		Kaki 4	
0	1	0	0	0	0	0	1

3. Konfigurasi kedua adalah untuk mengangkat Lengan Bawah kaki1 dan kaki 4 untuk selang waktu sebesar waktu tunda untuk lengan bawah (DELAY_LB).

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
Kaki 1		Kaki 2		Kaki 3		Kaki 4	
0	1	0	0	0	0	0	1

4. Konfigurasi ketiga adalah gerak maju Lengan Atas kaki 2 dan kaki 3 diikuti pula dengan gerak mundur Lengan Atas kaki 1 dan kaki 4 untuk selang waktu sebesar waktu tunda untuk lengan atas (DELAY_LA).

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
Kaki 1		Kaki 2		Kaki 3		Kaki 4	
0	1	1	0	1	0	0	1

5. Konfigurasi keempat adalah gerak menekan Lengan Bawah kaki 1 dan kaki 4 untuk selang waktu sebesar waktu tunda untuk lengan bawah (DELAY_LB).

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
Kaki 1		Kaki 2		Kaki 3		Kaki 4	
1	0	0	0	0	0	1	0

6. Konfigurasi kelima adalah gerak menghisap Lengan Hisap kaki 1 dan kaki 4 untuk selang waktu sebesar waktu tunda untuk lengan hisap (DELAY_LH).

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
Kaki 1		Kaki 2		Kaki 3		Kaki 4	
1	0	0	0	0	0	1	0

7. Untuk konfigurasi berikutnya adalah berturut-turut sama dengan konfigurasi satu hingga konfigurasi kelima diatas, hanya saja pergerakan utama dilakukan oleh lengan-lengan pada kaki 2 dan kaki 3.

Analisa

Jika diamati, maka konfigurasi di atas dapat diterangkan sebagai berikut :

1. Dari posisi awal (semua kaki sejajar) atau dari posisi digonal (kaki tak sejajar/memotong), akan dimulai dengan pengnonaktifan dari lengan hisap dari kaki 1 dan kaki 4 (kaki yang akan melakukan pergerakan) dengan pewaktuan sebesar yang telah ditentukan untuk proses tersebut (DELAY_LH).
2. Setelah itu lengan bawah pada kaki 1 dan kaki 4 akan melakukan pergerakan ke atas guna mengangkat lengan hisap (agar pada saat merayap, lengan hisap tidak terseret pada bidang permukaan pijakan)

dengan pewaktuan sebesar yang telah ditentukan untuk proses tersebut (DELAY_LB).

3. Berikutnya lengan atas pada kaki 2 dan kaki 3 akan melakukan pergerakan rotasi guna memutar atau memindahkan kaki 1 dan 4 sekitar 30 derajat dengan pewaktuan yang telah ditentukan untuk proses tersebut (DELAY_LA).
4. Proses berikutnya adalah dengan menempelkan kembali lengan bawah pada kaki 1 dan kaki 4 (dengan membalik polaritas atau konfigurasi bit) selama DELAY_LB.
5. Terakhir adalah melakukan proses penghisapan oleh lengan hisap pada kaki 1 dan kaki 4 selama DELAY_LH.
6. Robot telah melakukan pergerakan maju dengan cara melangkahkan kaki secara diagonal seperti halnya pada binatang merayap lainnya.
7. Dengan urutan-urutan yang sama akan dilakukan langkah yang identik, hanya saja dilakukan oleh lengan-lengan pada kedua kaki yang berbeda yakni kaki 2 dan kaki 3.

4.5 Pengujian Pewaktuan

Untuk memperoleh pewaktuan dari masing-masing pergerakan lengan robot dengan tujuan agar proses pergerakan tersebut terpenuhi sesuai dengan waktu yang dibutuhkannya, maka pertama-tama perlu diadakan perbandingan pewaktuan secara *software* dengan secara praktik. Kemudian setelah mendapatkan waktu proses pergerakan yang tepat maka dilakukan

perhitungan terhadap *execution time* dari rutin DELAY pada program. Contoh listing program yang diaplikasikan pada pergerakan robot adalah sebagai berikut :

```

DELAY_LA :      ld b,03H
                ulang3: push de
                  push af
                  ld d,0FCH
                loop_G: ld e,0FEH
                loop_H: ld a,28H
                loop_I: dec a
                  jr nz,loop_I
                  dec e
                  jr nz,loop_H
                  dec d
                  jr nz,loop_G
                  pop af
                  pop de
                  djnz ulang3
                  ret
    
```

Dengan mengamati cuplikan program untuk penundaan pewaktuan di atas pada proses pergerakan dari lengan atas maka dengan menggunakan bantuan tabel *mnemonic* (kode mesin) dapat dilihat waktu eksekusi dari masing-masing instruksi serta dengan pengamatan alur program maka dapat dihitung jumlah waktu tunda yang terjadi. Misalnya untuk cuplikan program waktu tunda untuk pergerakan lengan atas di atas dan dengan mengetahui bahwa frekuensi rangkaian *clock* pada CPU Z80 yang bernilai 4 MHz, terhitung bahwa rutin waktu tunda tersebut bernilai kurang lebih 30 detik.

Untuk hasil penghitungan waktu eksekusi pada proses pergerakan lainnya dapat diamati pada tabel di bawah ini. Dalam hal ini waktu tunda secara perhitungan melalui program sengaja diberi toleransi sekitar 5 detik guna menjaga agar proses pergerakan dari masing-masing lengan dapat benar-benar terpenuhi.

Tabel 4.3. Tabel perbandingan pewaktuan

WAKTU EKSEKUSI PROSES PENGHISAPAN		WAKTU EKSEKUSI PROSES LENGAN BAWAH		WAKTU EKSEKUSI PROSES LENGAN ATAS	
Praktik	Teori	Praktik	Teori	Praktik	Teori
± 32 detik (hisap)	± 38 detik	± 7 detik	± 14 detik	± 25 detik	± 30 detik
± 30 detik (lepas)	± 37 detik	± 7 detik	± 14 detik	± 25 detik	± 30 detik

BAB V

PENUTUP

BAB V

PENUTUP

5.1 Kesimpulan

Beberapa hal yang dapat disimpulkan dalam proyek akhir ini, terutama yang berkenaan dengan sistem kendali secara keseluruhan serta algoritma program kemudi dari robot, yang dalam hal ini adalah Robot Berkaki Empat Berkemampuan Merayap yakni :

1. Sistem kendali yang diaplikasikan bersifat semiotomatis dengan mode kerja operator oriented yang ditujukan sebagai acuan terhadap proses pengendalian mikroprosesor Z80 dari robot yang dalam hal ini output berupa pergerakan dari aktuator-aktuator yang memiliki loop terbuka terhadap rangkaian sensor-sensornya.
2. Interfacing dengan piranti luar dilakukan dengan rangkaian remote control, driver motor serta sensor-sensor limit switch.
3. Pemrograman bersifat sekuensial dengan menerapkan berbagai macam konfigurasi pergerakan dari keempat kaki dari robot.
4. Mampu merayap pada bidang vertikal ataupun datar yang tidak berongga.

5.2 Saran-Saran

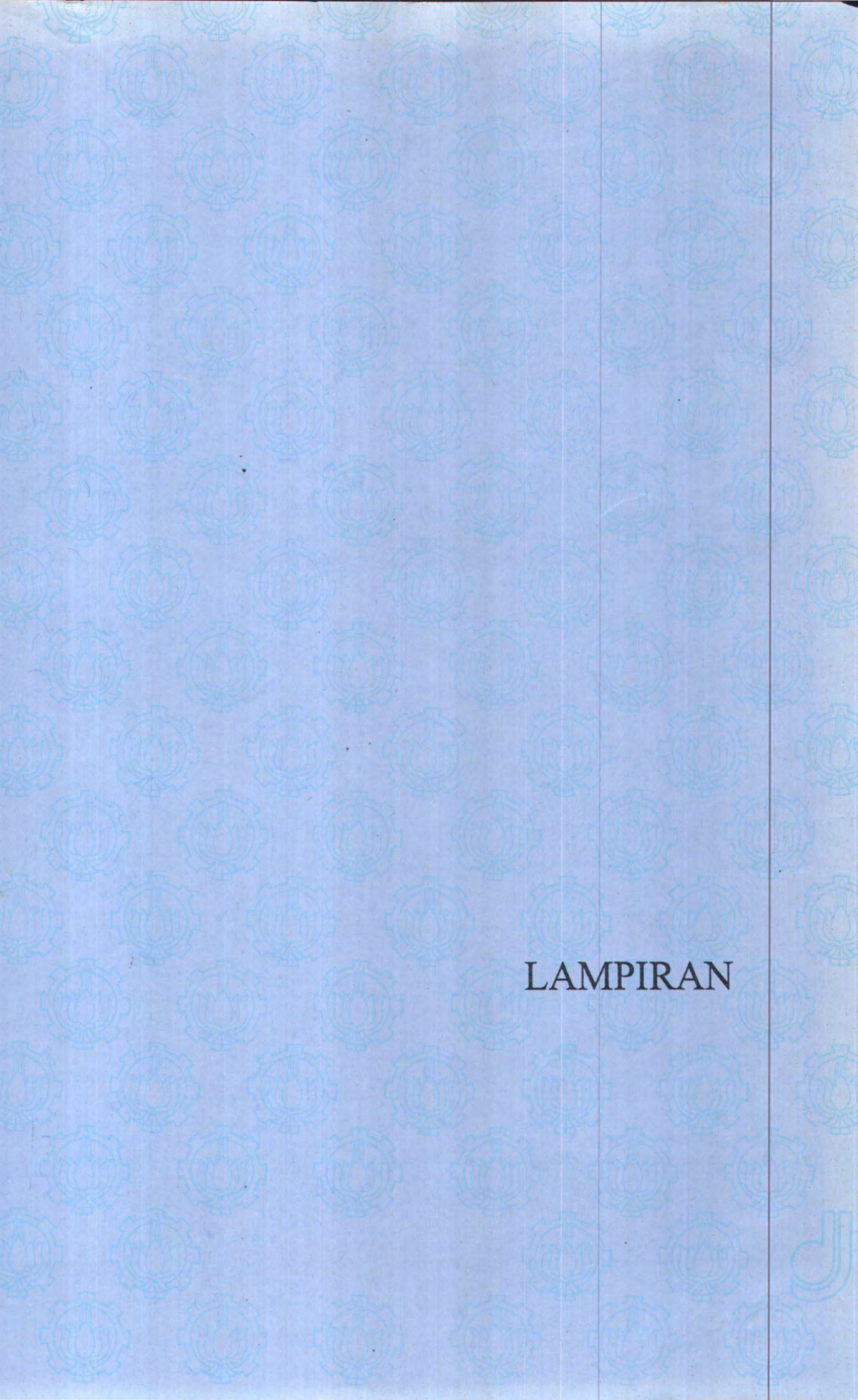
Untuk lebih menyempurnakan fungsi yang sebenarnya dari sistem robot ini maka terdapat beberapa saran yang bersifat membangun.

1. Rangkaian sensor dapat lebih disempurnakan dengan mengaplikasikan sensor perpindahan yang lainnya agar pergerakan dari robot lebih mudah divariasikan dan untuk selebihnya dapat diterapkan sistem kendali yang bersifat otomatis.
2. Untuk meningkatkan unjuk kerja dari sistem minimum ini hendaknya diperhatikan program kendali untuk robot ini terlalu banyak mengaplikasikan perintah DELAY yang menyebabkan CPU menunggu dan tidak mengerjakan perintah apapun. Waktu ini seharusnya dapat digunakan untuk melakukan task yang lainnya, misalnya penyempurnaan fungsi dengan melakukan pembersihan kaca sehingga tercapai pengoperasian multitasking dari Z80 CPU.

DAFTAR PUSTAKA

DAFTAR PUSTAKA

1. Endra Pitowarno, **Mikroprosesor & Interface I,II**, EEPIS Surabaya, 1990
2. Endra Pitowarno & Matsumo Sutomo, **Pratikum Mikroprosesor & Interface I, II**, EEPIS Surabaya, 1990
3. Barry B Brey, **The Z80 Microprocessor, hardware, sotware, Programming and Interfacing**, Prentice Hall International Inc. United State of America, 1988
4. A Meystel, **Autonomous Mobile Robots Vihicle with Cognate Control**, World Scientific Singapore, 1991



LAMPIRAN

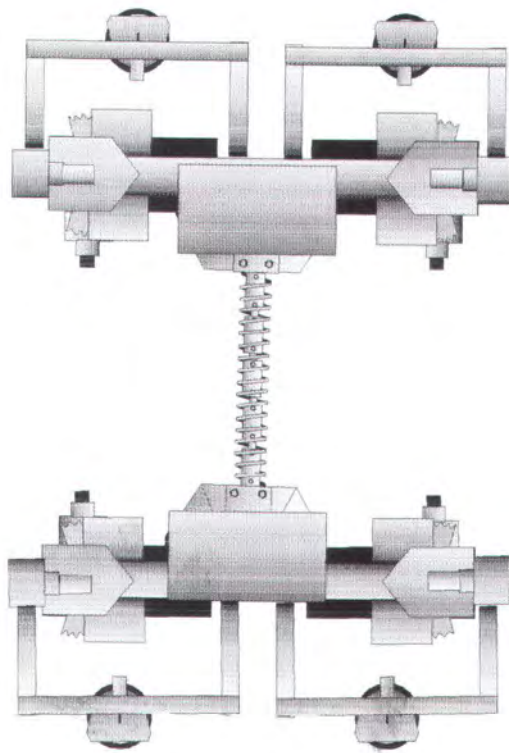
DATA MEKANIK

CR-01

ROBOT BERKAKI EMPAT

BERKEMAMPUAN

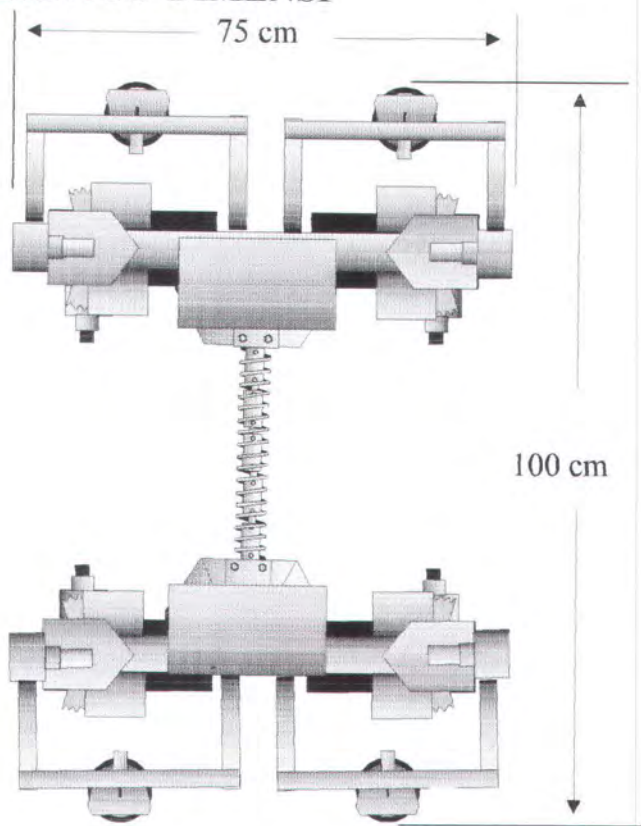
MERAYAP



POLITEKNIK ELEKTRONIKA NEGERI SURABAYA
INSTITUT TEKNOLOGI SEPULUH NOVEMBER
SURABAYA

DATA MEKANIS

• KARAKTERISTIK DIMENSI



Tabel 1. Ukuran Main Bodi Robot

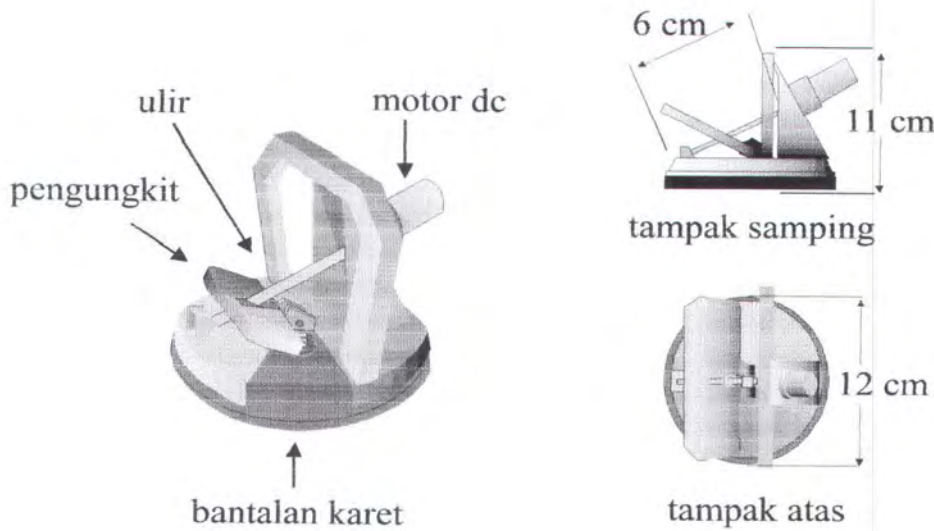
No	Keterangan	Ukuran
1.	Tinggi Robot	25 cm
2.	Lebar Robot	75 cm
3.	Panjang Robot	100 cm
4.	Jumlah motor	12 buah

No	Keterangan	Ukuran
5.	Berat Robot	15 kg

Keterangan :

- Bahan : Aluminium ketebalan 1mm
- Aktuator : Motor DC 24 V 8 buah
Motor DC 12 V 4 buah

• KAKI HISAP



Tabel 2. Ukuran Kaki Hisap

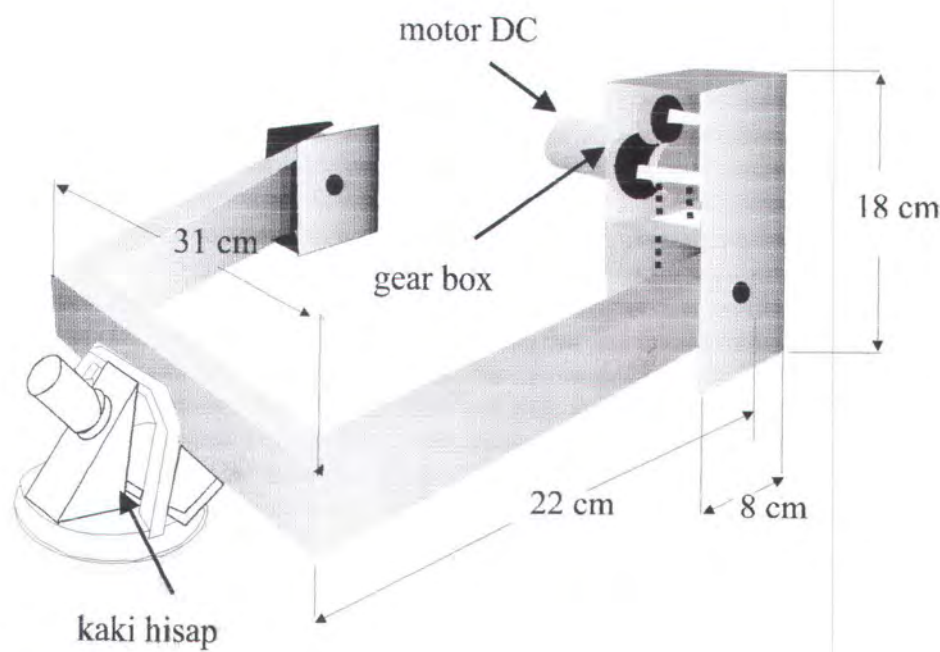
No	Keterangan	Ukuran
1.	Tinggi Kaki Hisap	11 cm
2.	Diameter Kaki Hisap	12 cm
3.	Tebal Bantalan Karet	1 cm

No	Keterangan	Ukuran
4.	Panjang ulir	6 cm
5.	Jumlaah motor penggerak	1 buah
6.	Berat kaki hisap	0,6 kg

Keterangan :

- Bahan : Aluminium plat ketebalan 1mm
Plastik sebagai penutup
Bantalan karet dengan penguat plat besi
Ulir dengan diameter 6 mm
- Aktuator : Motor DC 24 V 1 buah

• LENGAN BAWAH



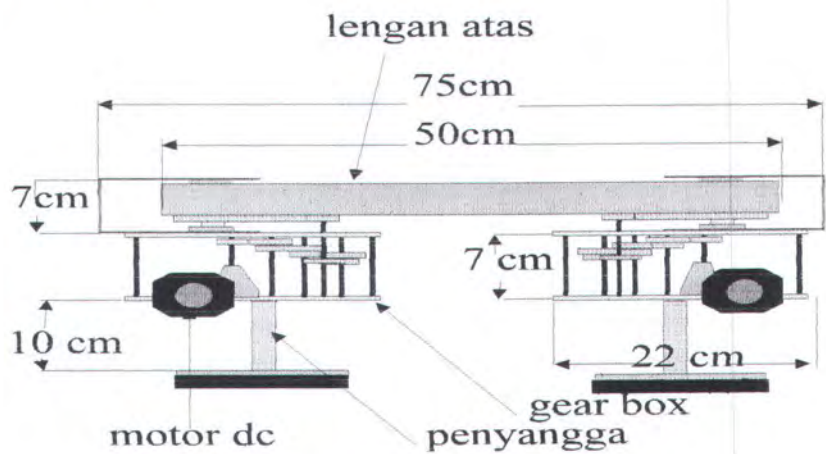
Tabel 3. Ukuran Lengan bawah

No	Keterangan	Ukuran
1.	Panjang lengan	22 cm
2.	Lebar lengan	31 cm
3.	Rasio gear box	6,01
4.	Jumlah motor penggerak	1 buah

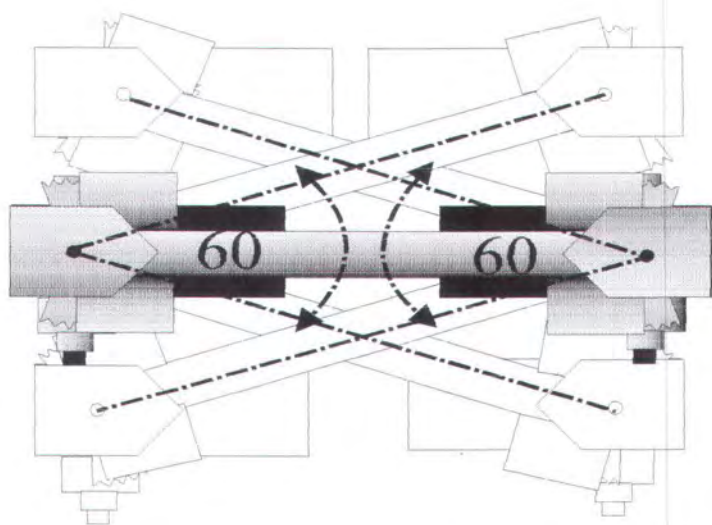
Keterangan :

- Bahan : Aluminium plat ketebalan 1mm
Aluminium box 50 × 25 mm
Gear plastik (resin)
- Aktuator : Motor DC 24 V 1 buah

• LENGAN ATAS



a. Tampak Samping



b. Tampak Atas

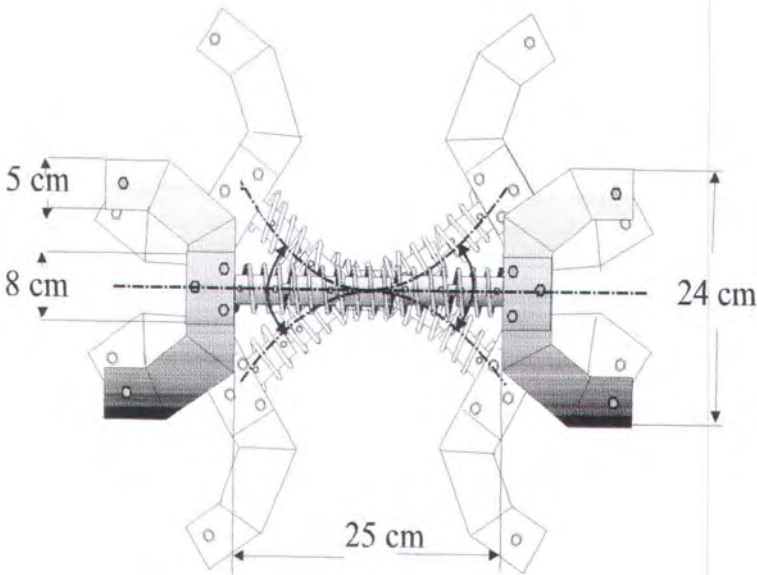
Tabel 4. Ukuran Lengan atas

No	Keterangan	Ukuran
1.	Panjang lengan	50 cm
2.	Lebar lengan	17 cm
3.	Rasio gear box	17,45
4.	Jumlah motor penggerak	1 buah

Keterangan :

- Bahan : Aluminium plat ketebalan 1mm
Aluminium box 50 × 25 mm
Gear plastik (resin)
- Aktuator : Motor DC 12 V 1 buah (tiap kaki)

• RANGKA PENYANGGA



Tabel 5. Ukuran Rangka penyangga

No	Keterangan	Ukuran
1.	Panjang rangka per	25 cm
2.	Lebar rangka	24 cm
3.	diameter per	5 cm

Keterangan :

- Bahan : Aluminium plat ketebalan 1mm
Aluminium box 50 × 25 mm
Per besi cor

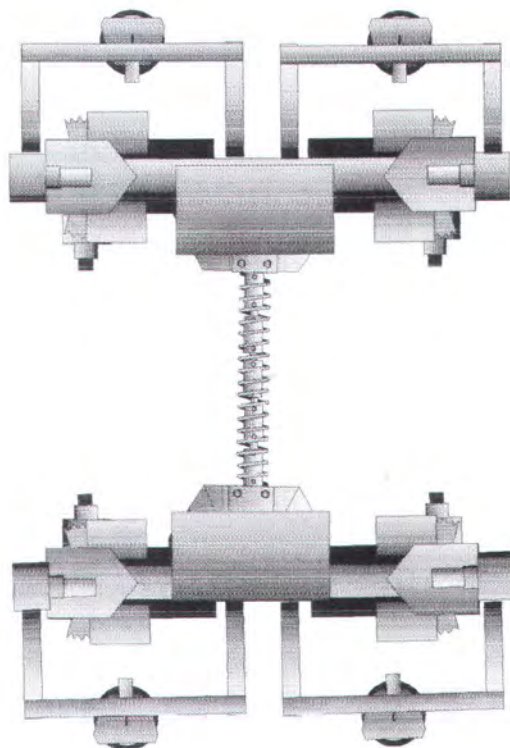
LISTING PROGRAM

CR-01

ROBOT BERKAKI EMPAT

BERKEMAMPUAN

MERAYAP



POLITEKNIK ELEKTRONIKA NEGERI SURABAYA
INSTITUT TEKNOLOGI SEPULUH NOVEMBER
SURABAYA

LAMPIRAN B

PROYEK : PROTOTYPE ROBOT BERKAKI EMPAT
BERKEMAMPUAN MERAYAP

JUDUL : PROGRAM SEMI_AUTOMATIC RUNNING
TEAM ROBOT MERAYAP
(I.G.M. ARTHA Y, RINO HENDARTO)

DOSEN : Ir. ENDRA PITOWARNO

Tgl : 15 Mei 1998

PPIPA_1	EQU 80H	;alamat kerja PPI
PPIPB_1	EQU 81H	
PPIPC_1	EQU 82H	
PPICW_1	EQU 83H	
PPIIN_1	EQU 80H	;PA-out,PB-out,PC-out
PPIPA_2	EQU 84H	
PPIPB_2	EQU 85H	
PPIPC_2	EQU 86H	
PPICW_2	EQU 87H	
PPIIN_2	EQU 83H	;PA-in,PB-in,PC-out

KETERANGAN :

LB = LENGAN BAWAH, LA = LENGAN ATAS, LH = LENGAN HISAP

PPI_1 SEBAGAI OUTPUT PORT DENGAN KONFIGURASI BIT :

* PA0 - PA7 UNTUK MOTOR LA
(berturut-turut kaki 1,2,3 dan 4)

* PB0 - PB7 UNTUK MOTOR LB

* PC0 - PC7 UNTUK MOTOR LH

PPI_2 SEBAGAI INPUT PORT DENGAN KONFIGURASI BIT :

* PA0 - PA7 UNTUK REMOTE CONTROL

* PB0 UNTUK LIMIT SWITCH SBG SENSOR POSISI

* PB1 _ PB2 UNTUK LIMIT SWITCH SBG IDENTIFIKASI PERGESERAN

BERHENTI	EQU 00000000B
AWAL_LB	EQU 10101010B
AWAL_LH	EQU 10101010B
NONAKTIF_LH_1_4	EQU 01000001B
ANGKAT_LB_1_4	EQU 01000001B
MAJU_LA_2_3	EQU 01101001B
TEKAN_LB_1_4	EQU 10000010B
AKTIF_LH_1_4	EQU 10000010B

NONAKTIF_LH_2_3	EQU 00010100B
ANGKAT_LB_2_3	EQU 00010100B
MAJU_LA_1_4	EQU 10010110B
TEKAN_LB_2_3	EQU 00101000B
AKTIF_LH_2_3	EQU 00101000B
MUNDUR_LA_2_3	EQU 10010110B
MUNDUR_LA_1_4	EQU 01101001B
LEPAS_LH	EQU 01010101B
AKTIF_LH_1_3	EQU 10001000B
AKTIF_LH_2_4	EQU 00100010B
NONAKTIF_LH_1_3	EQU 01000100B
NONAKTIF_LH_2_4	EQU 00010001B
ANGKAT_LB_1_3	EQU 01000100B
ANGKAT_LB_2_4	EQU 00010001B
TEKAN_LB_1_3	EQU 10001000B
TEKAN_LB_2_4	EQU 00100010B
KANAN_LA_1_3	EQU 10011001B
KANAN_LA_2_4	EQU 01100110B
KIRI_LA_1_3	EQU 10011001B
KIRI_LA_2_4	EQU 01100110B
TURUN_LA_1_3	EQU 01100110B
TURUN_LA_2_4	EQU 10011001B

MAIN PROGRAM

org 0000H

AWAL :

ld sp,0a000H
call PPIINIT

org 0038H

reti

org 0066H

retn

PPIINIT :

ld b,00H

PPIINLOOP1:

ld a,PPIIN_1
out(PPICW_1),a
djnz PPIINLOOP1

ld b,00H

PPIINLOOP2:

ld a,PPIIN_2
out(PPICW_2),a
djnz PPIINLOOP2

ret

org 0100H

START :

```

in a,(PIPA_2)
and 11111111B
cp 10000000B
call z,GERAK_AWAL
cp 01000000B
call z,GERAK_MAJU
cp 00100000B
call z,GERAK_MUNDUR
cp 00010000B
call z,KANAN
cp 00001000B
call z,KIRI
cp 00000100B
call z,GERAK_LEPAS
cp 00000010B
call z,STOP
jp START

```

GERAK_AWAL :

```

push af
push bc
call DELAY_3DETIK
in a,(PIPA_2)
and 11111111B
cp 10000000B
jp nz START
call RUTIN_AWAL
pop bc
pop af
ret

```

GERAK_MAJU :

```

push af
push bc
call DELAY_3DETIK
in a,(PIPA_2)
and 11111111B
cp 01000000B
jp nz, START
call RUTIN_MAJU

```

GERAK_MUNDUR :

```

push af
push bc
call DELAY_3DETIK
in a,(PIPA_2)
and 11111111B
cp 00100000B
jp nz, START
call RUTIN_MUNDUR

```

KANAN :

```

push af
push bc
call DELAY_3DETIK
in a,(PIPA_2)
and 11111111B
cp 00010000B
jp nz, START
call TENGAH
pop bc
pop af
ret

```

KIRI :

```

push af
push bc
call DELAY_3DETIK
in a,(PIPA_2)
and 11111111B
cp 00001000B
jp nz, START
call TENGAH
pop bc
pop af
ret

```

GERAK_LEPAS :

```

push af
push bc
call DELAY_3DETIK
in a,(PIPA_2)
and 11111111B
cp 00000010B
jp nz, START
call RUTIN_LEPAS
pop bc
pop af
ret

```

STOP :

```

push af
push bc
call DELAY_3DETIK
in a,(PIPA_2)
and 11111111B
cp 00000000B
jp nz, START
call RUTIN_STOP
pop bc

```



```
pop af
ret
```

```
RUTIN_AWAL :
```

```
push af
ld a,AWAL_LB
out(PPIPB_1),a
call DELAY_LB
ld a,AWAL_LH
out(PPIPC_1),a
call DELAY_LH_A
pop af
ret
```

```
RUTIN_MAJU :
```

```
push af
ld a,NONAKTIF_LH_1_4
out(PPIPC_1),a
call DELAY_LH_N
ld a,ANGKAT_LB_1_4
out(PPIPB_1),a
call DELAY_LB
ld a,MAJU_LA_1_4
out(PPIPA_1),a
call DELAY_LA
ld a,TEKAN_LB_1_4
out(PPIPB_1),a
call DELAY_LB
ld a,AKTIF_LH_1_4
out(PPIPC_1),a
call DELAY_LH_A
ld a,NONAKTIF_LH_2_3
out(PPIPC_1),a
call DELAY_LH_N
ld a,ANGKAT_LB_2_3
out(PPIPB_1),a
call DELAY_LB
ld a,MAJU_LA_2_3
out(PPIPA_1),a
call DELAY_LA
ld a,TEKAN_LB_2_3
out(PPIPB_1),a
call DELAY_LB
ld a,AKTIF_LH_2_3
out(PPIPC_1),a
call DELAY_LH_A
pop af
ret
```

```
RUTIN_MUNDUR :
```

```
push af
ld a,NONAKTIF_LH_2_3
```

```

out(PPIPC_1),a
call DELAY_LH_N
ld a,ANGKAT_LB_2_3
out(PPIPB_1),a
call DELAY_LB
ld a,MUNDUR_LA_2_3
out(PPIPA_1),a
call DELAY_LA
ld a,TEKAN_LB_2_3
out(PPIPB_1),a
call DELAY_LB
ld a,AKTIF_LH_2_3
out(PPIPC_1),a
call DELAY_LH_A
ld a,NONAKTIF_LH_1_4
out(PPIPC_1),a
call DELAY_LH_N
ld a,ANGKAT_LB_1_4
out(PPIPB_1),a
call DELAY_LB
ld a,MUNDUR_LA_1_4
out(PPIPA_1),a
call DELAY_LA
ld a,TEKAN_LB_1_4
out(PPIPB_1),a
call DELAY_LB
ld a,AKTIF_LH_1_4
out(PPIPC_1),a
call DELAY_LH_A
pop af
ret

```

TENGAH :

```

push af
push bc
ld b,11111100B
ld c,11111010B
in a,(PPIPB_2)
and 11111111B
cp b
call z, TURUN_KIRI
cp 11111010B
call z, TURUN_KANAN
jp nz,START
pop bc
pop af
ret

```

```

TURUN_KIRI: push af
ld a,NONAKTIF_LH_1_3
out(PPIPC_1),a
call DELAY_LH_N
ld a,ANGKAT_LB_1_3
out(PPIPB_1),a
call DELAY_LB

```

```

TERUS_1:ld a,TURUN_LA_2_4
        out(PPIPA_1),a
        in a,(PPIPB_2)
        and 11111111B
        cp b
        jp nz,TERUS_1
        ld a,(TEKAN_LB_1_3)
        out (PPIPB_1),a
        call DELAY_LB
        ld a,(AKTIF_LH_1_3)
        out(PPIPC_1),a
        call DELAY_LH_A
        call MULAI_GESER
        pop af
        ret

```

```

TURUN_KANAN:push af
            ld a,NONAKTIF_LH_2_4
            out(PPIPC_1),a
            call DELAY_LH_N
            ld a,ANGKAT_LB_2_4
            out(PPIPB_1),a
            call DELAY_LB

```

```

TERUS_2:ld a,TURUN_LA_1_3
        out(PPIPA_1),a
        in a,(PPIPB_2)
        and 11111111B
        cp c
        jp nz,TERUS_2
        ld a,(TEKAN_LB_2_4)
        out(PPIPB_1),a
        call DELAY_LB
        ld a,(AKTIF_LH_2_4)
        out(PPIPC_1),a
        call DELAY_LH_A
        call MULAI_GESER
        pop af
        ret

```

```

MULAI_GESER:push af
            in a,(PPIPA_2)
            and 11111111B
            cp 00001000B
            call z, GESER_KANAN
            cp 00000100B
            call z,GESER_KIRI
            pop af
            ret

```

```

GESER_KANAN:push af
            ld a,NONAKTIF_LH_1_3

```



```

out (PIIPC_1),a
call DELAY_LH_N
ld a,ANGKAT_LB_1_3
out (PIIPB_1),a
call DELAY_LB
ld a,KANAN_LA_2_4
out (PIIPA_1),a
call DELAY_LA
ld a,NONAKTIF_LH_2_4
out (PIIPC_1),a
call DELAY_LH_N
ld a,ANGKAT_LB_2_4
out (PIIPB_1),a
call DELAY_LB
ld a,KANAN_LA_1_3
out (PIIPA_1),a
call DELAY_LA
pop af
ret

```

```

GESER_KIRI :push af
ld a,NONAKTIF_LH_2_4
out (PIIPC_1),a
call DELAY_LH_N
ld a,ANGKAT_LB_2_4
out (PIIPB_1),a
call DELAY_LB
ld a,KIRI_LA_1_3
out (PIIPA_1),a
call DELAY_LA
ld a,NONAKTIF_LH_1_3
out (PIIPC_1),a
call DELAY_LH_N
ld a,ANGKAT_LB_1_3
out (PIIPB_1),a
call DELAY_LB
ld a,KIRI_LA_2_4
out (PIIPA_1),a
call DELAY_LA
pop af
ret

```

```

RUTIN_STOP :
push af
ld a,BERHENTI
out(PIIPA_1),a
out(PIIPB_1),a
out(PIIPC_1),a
pop af
ret

```

```

RUTIN_LEPAS :
push af

```

```

ld a,LEPAS_LH
out(PPIPC_1),a
call DELAY_LH_N
pop af
ret

```

```

DELAY_LH_A :
ld b,03H
ulang1: push de
push af
ld d,0FFH
loop_A: ld e,0FFH
loop_B: ld a,05H
loop_C: dec a
jr nz,loop_C
dec e
jr nz,loop_B
dec d
jr nz,loop_A
pop af
pop de
djnz ulang1
ret

```

```

DELAY_LH_N :
ld b,02H
ulang2: push de
push af
ld d,00H
loop_D: ld e,0FEH
loop_E: ld a,50H
loop_F: dec a
jr nz,loop_F
dec e
jr nz,loop_E
dec d
jr nz,loop_D
pop af
pop de
djnz ulang2
ret

```

```

DELAY_LB :
ld b,02H
ulang3: push de
push af
ld d,00H
loop_G: ld e,0F0H
loop_H: ld a,25H
loop_I: dec a

```

```

    jr nz,loop_I
    dec e
    jr nz,loop_H
    dec d
    jr nz,loop_G
    pop af
    pop de
    djnz ulang3
    ret

```

DELAY_LA :

```

    ld b,03H
ulang4: push de
        push af
        ld d,0FCH
loop_J: ld e,0FEH
loop_K: ld a,28H
loop_L: dec a
        jr nz,loop_L
        dec e
        jr nz,loop_K
        dec d
        jr nz,loop_J
        pop af
        pop de
        djnz ulang4
    ret

```

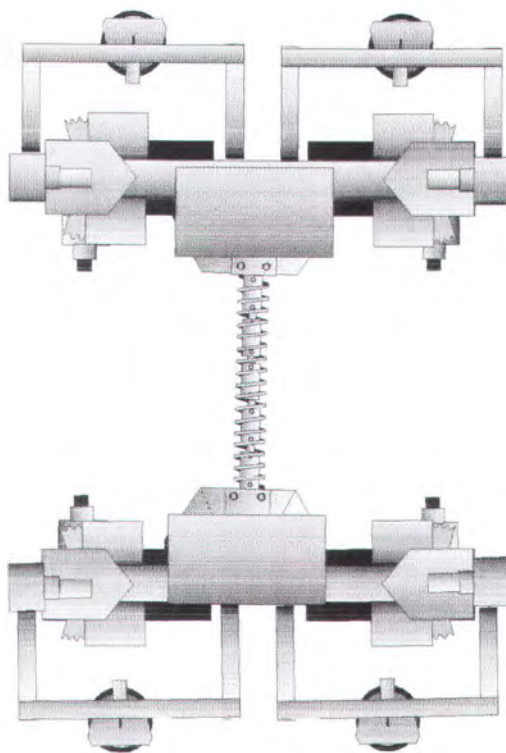
DELAY_3DETIK :

```

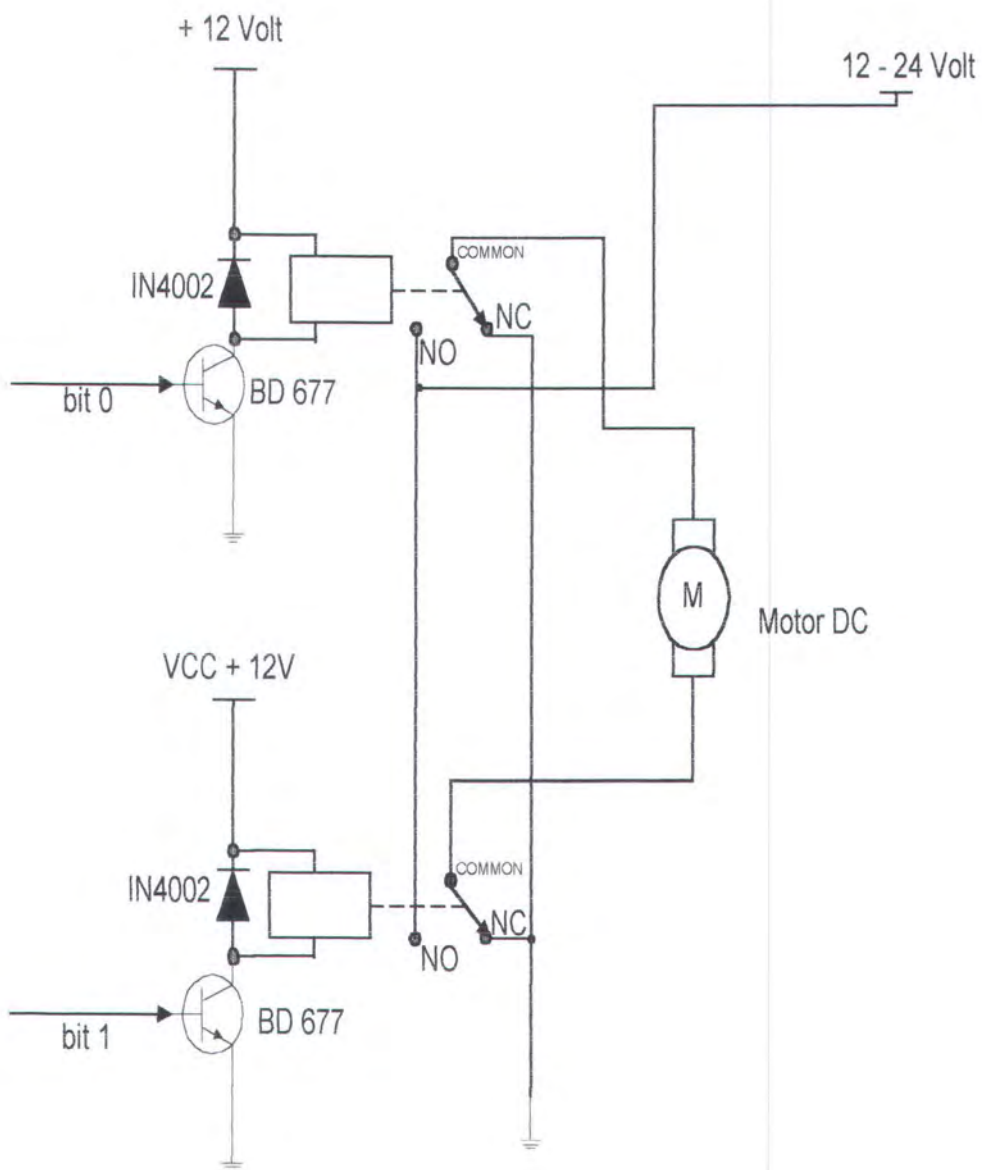
    ld b,20H
ulang5: push de
        push af
        ld d,0F1H
loop_M: ld e,03H
loop_N: ld a,0DH
loop_O: dec a
        jr nz,loop_O
        dec e
        jr nz,loop_N
    dec d
        jr nz,loop_M
        pop af
        pop de
        djnz ulang5
    ret

```

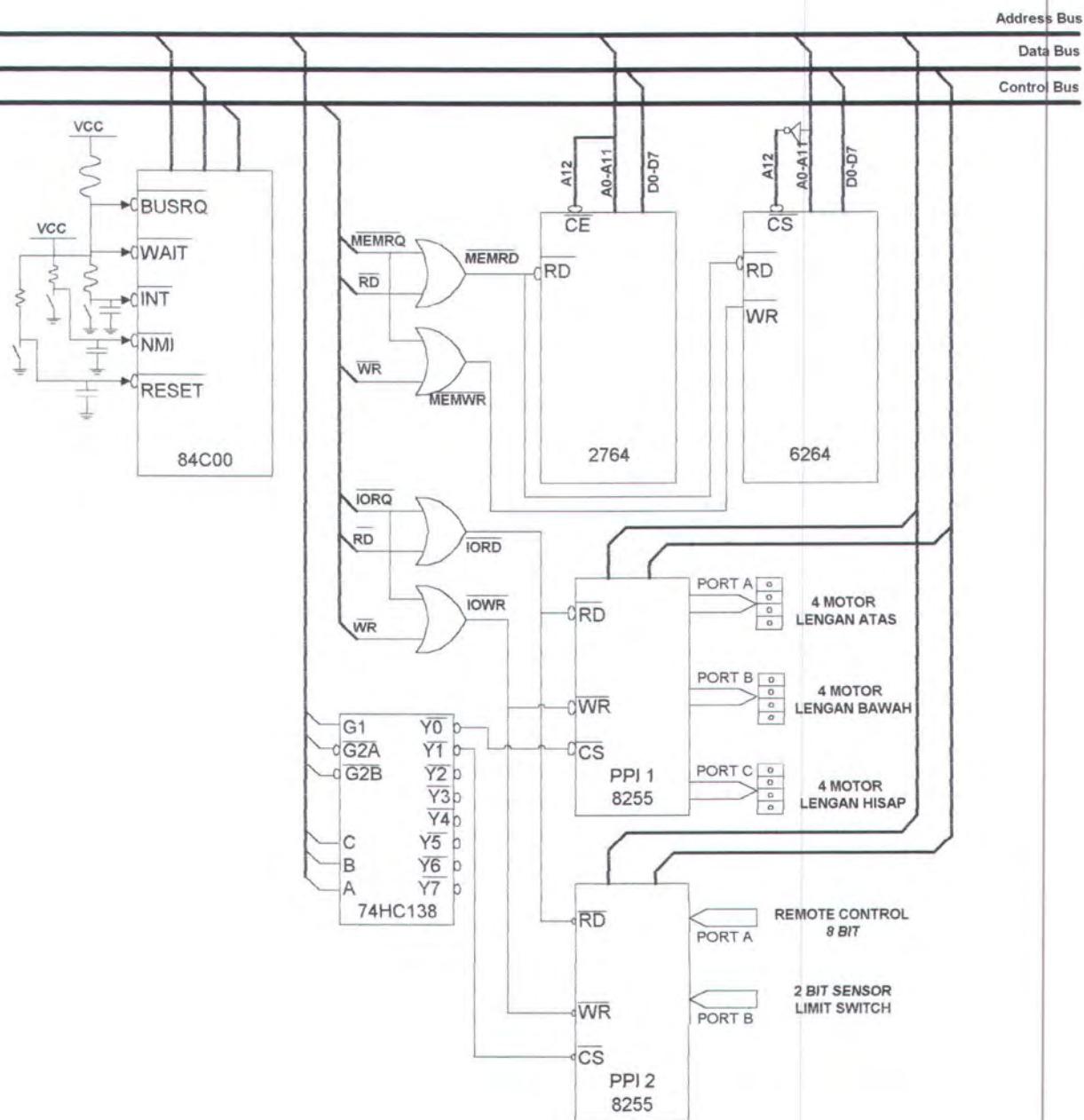

SISTEM KENDALI CR-01 ROBOT BERKAKI EMPAT BERKEMAMPUAN MERAYAP



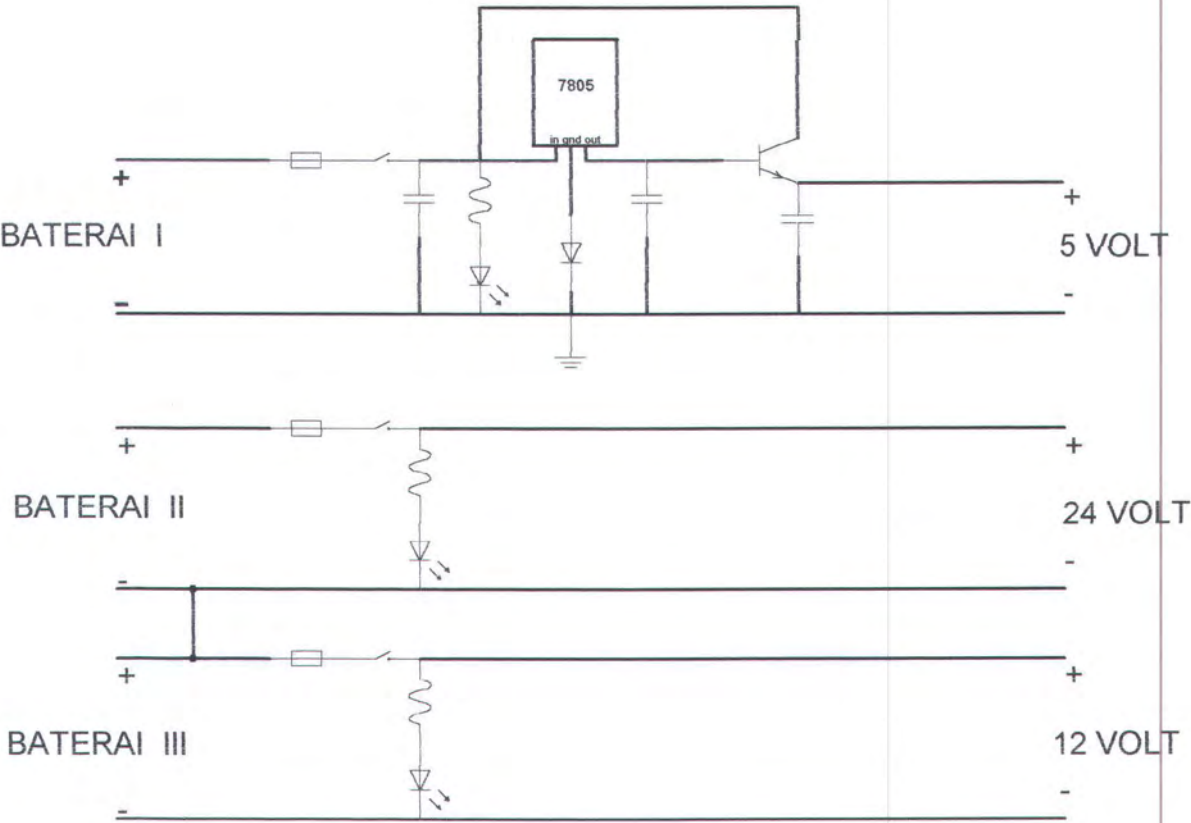
**POLITEKNIK ELEKTRONIKA NEGERI SURABAYA
INSTITUT TEKNOLOGI SEPULUH NOVENBER
SURABAYA**



Rangkaian Driver Relay Motor DC 12 dan 24 Volt



Modul Sistem Minimum Z80 beserta Perangkat Interfacenya



Rangkaian Regulator Power Supply

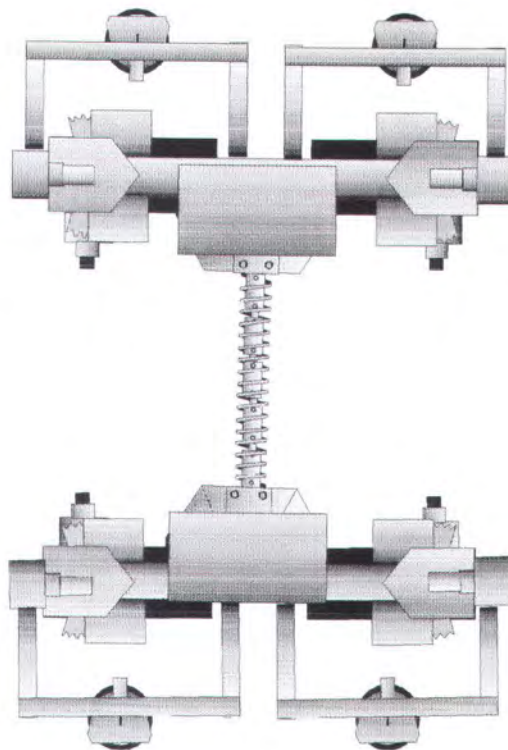
FOTO GAMBAR

CR-01

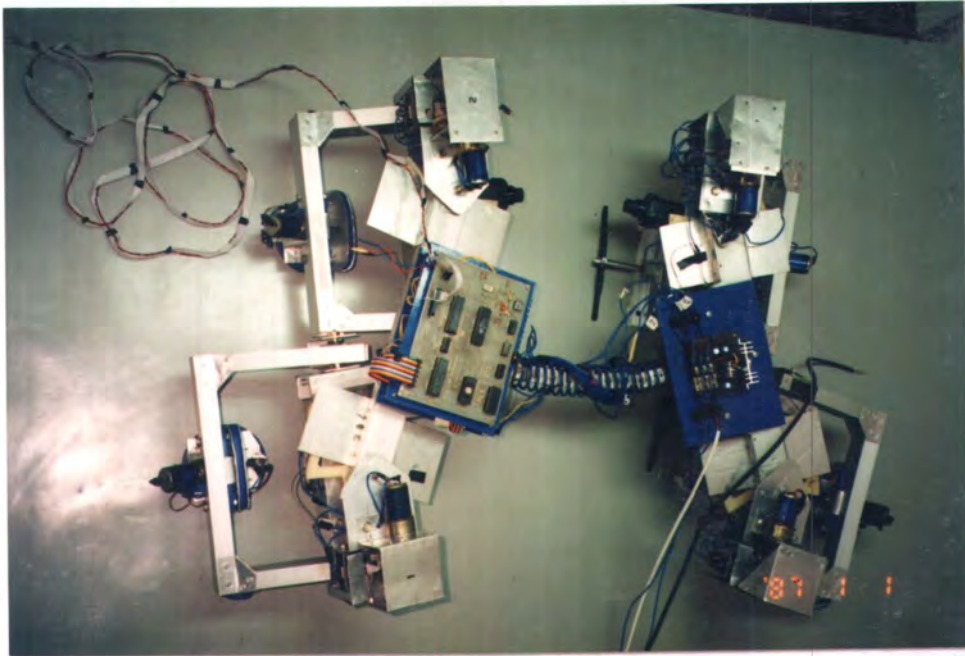
ROBOT BERKAKI EMPAT

BERKEMAMPUAN

MERAYAP



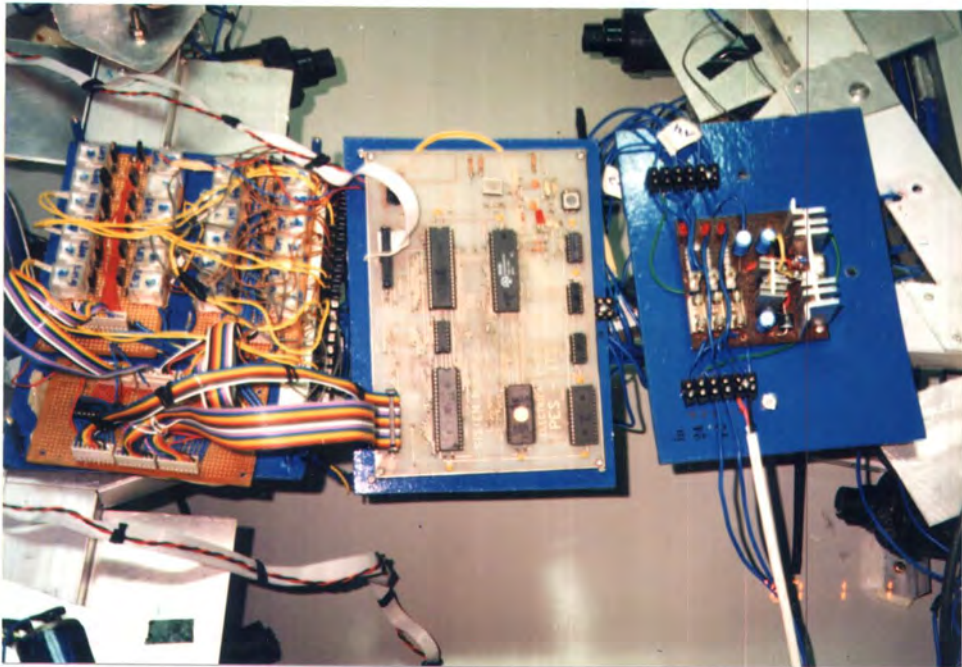
POLITEKNIK ELEKTRONIKA NEGERI SURABAYA
INSTITUT TEKNOLOGI SEPULUH NOVEMBER
SURABAYA



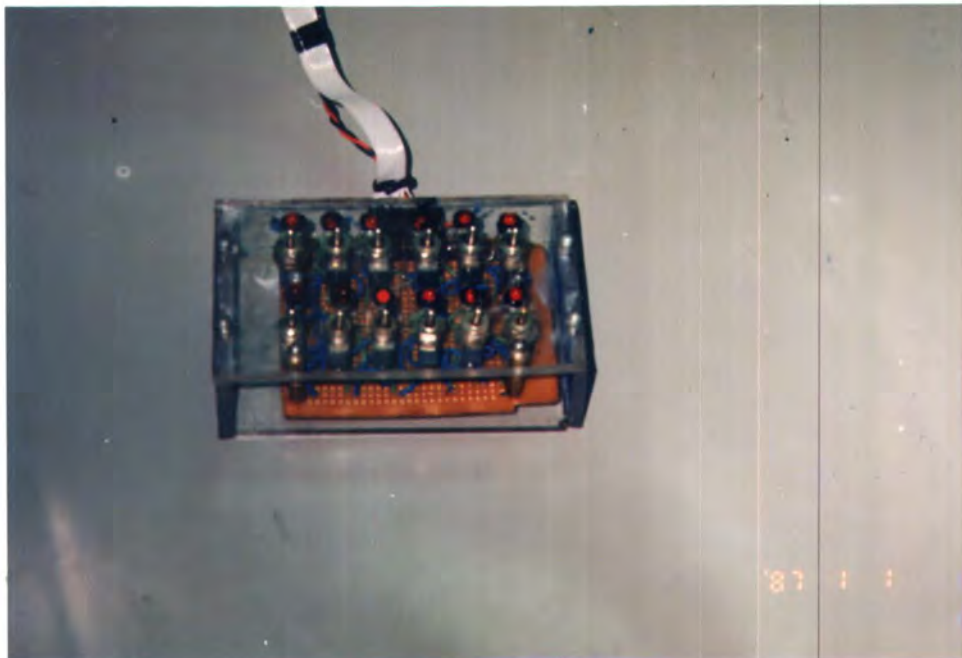
**Konstruksi Robot Berkaki Empat Berkemampuan Merayap
Tampak atas**



**Tampak depan Robot Berkaki Empat Berkemampuan
Merayap**



**Modul Sistem Minimum Z80, Driver Relay Motor DC, Interfacing buffer
Dan Regulator Tegangan**



Interfacing Peripheral berupa Remote Control

PETUNJUK MANUAL PENGOPERASIAN

Konfigurasi dari bit-bit atau tombol-tombol adalah sebagai berikut:

Bit 0 adalah masukan untuk acuan dalam bergerak awal.

Bit 1 adalah masukan untuk acuan dalam bergerak maju.

Bit 2 adalah masukan untuk acuan dalam bergerak mundur.

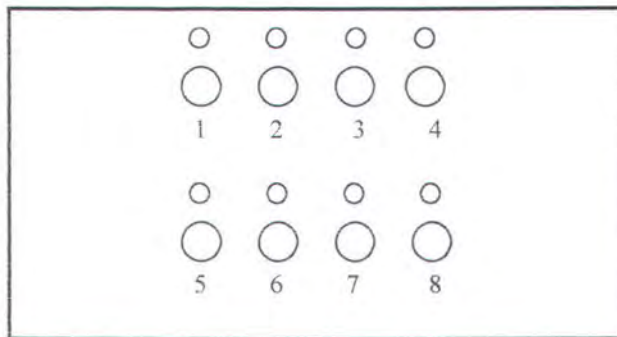
Bit 3 adalah masukan untuk acuan dalam bergeser kanan.

Bit 4 adalah masukan untuk acuan dalam bergeser kiri.

Bit 5 adalah masukan untuk acuan dalam berhenti.

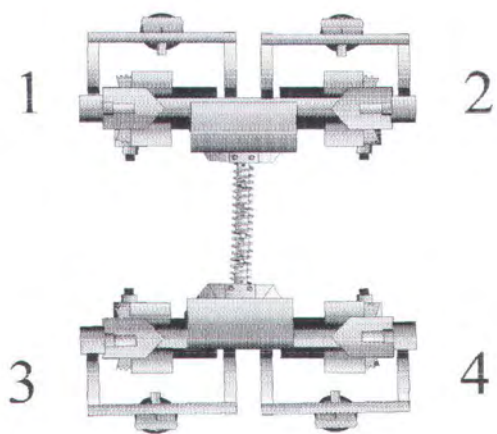
Bit 6 adalah untuk acuan dalam melakukan gerakan lepas.

Bit 7 belum terpakai.

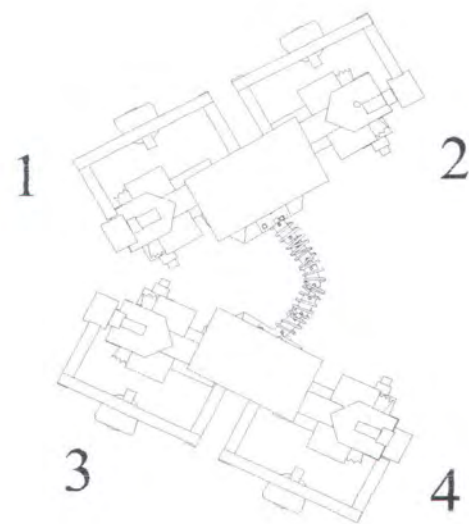


- Untuk gerakan ke posisi awal dapat dilihat pada gambar A, dimana dalam hal ini terjadi proses pergerakan mensejajarkan lengan atas, lengan bawah menekan ke permukaan bidang pijakan dan pergerakan atau proses menghisap dari lengan hisap dengan tujuan agar keseluruhan lengan dapat menempel pada bidang pijakan.

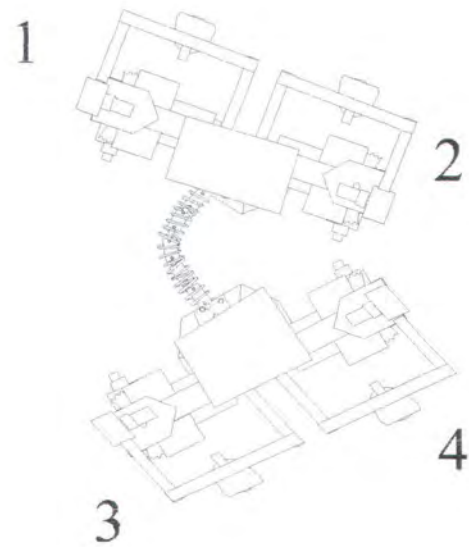
- Untuk gerakan maju dapat dilihat pada pergerakan dari posisi awal A ke posisi B lalu ke posisi C kemudian kembali lagi ke posisi B dan begitu seterusnya. Proses pergerakan berturut-turut melibatkan proses pergerakan rotasi dari lengan atas, penekanan dan pengangkatan dari lengan bawah serta penghisapan dan pelepasan dari lengan hisap.
- Untuk pergerakan mundur sama halnya dengan gerakan mundur hanya saja dari posisi awal A diteruskan ke posisi C berlanjut ke posisi B, kembali ke posisi C dan begitu seterusnya.
- Untuk pergerakan menggeser ke kanan, robot pertama-tama harus berada pada posisi awal (A) lalu mengadakan proses pergeseran seperti pada gambar D, lalu setelah itu akan dilakukan pergerakan maju.
- Untuk pergeseran ke kiri sama halnya dengan pergerakan sebelumnya, hanya saja dengan arah yang berlawanan seperti gambar E.
- Untuk proses pemberhentian akan mengakibatkan seluruh pergerakan berhenti.
- Sedangkan acuan pergerakan terakhir adalah untuk proses pelepasan diri dari bidang yang dirayap, yakni hanya dengan melakukan gerakan melepaskan keseluruhan lengan hisap secara bersamaan.



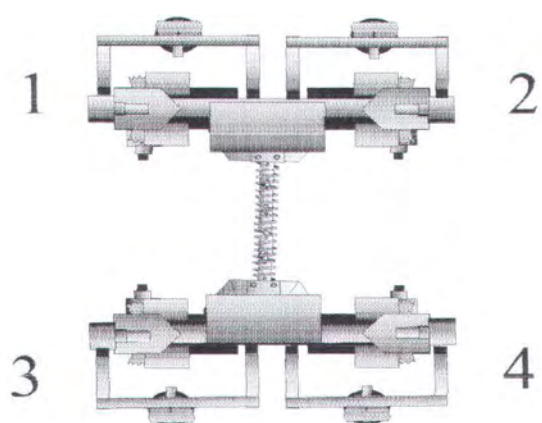
A.



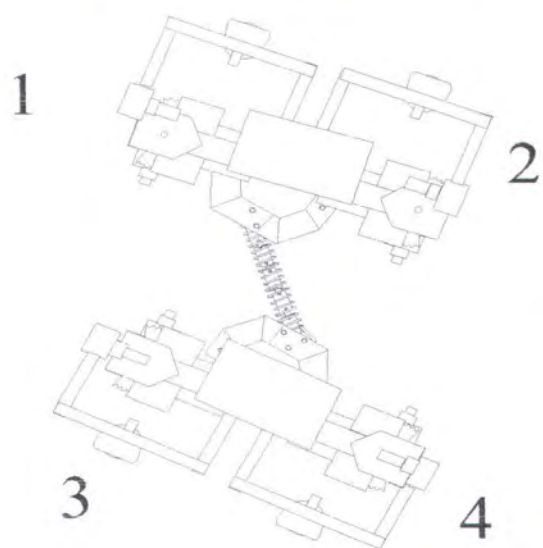
B.



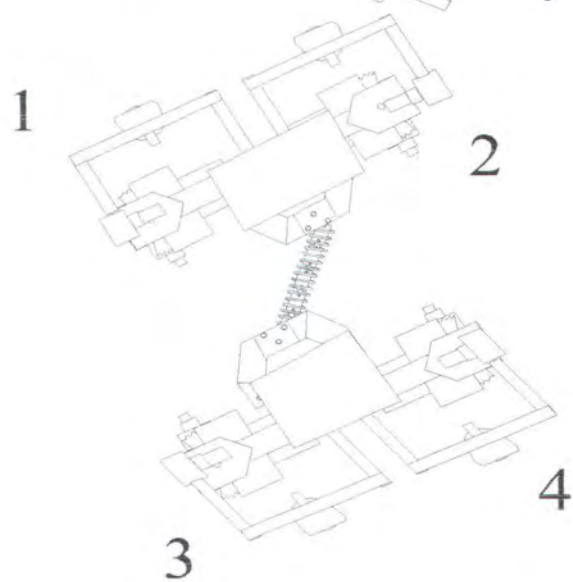
C.



A.



D.



E.

DAFTAR RIWAYAT HIDUP



Nama : I Gusti Made Artha Yogaiswara
Tempat/tanggal lahir : Ujung Pandang/ 26 April 1976
Agama : Hindu
Nama Ayah : Drh. IGK Paridjata Westra M.Agr.Sc
Nama Ibu : Titik Rosalinda
Alamat : Taman Sutorejo Timur 19
Telepon : (031) 5933543
Penulis merupakan putra kedua dari 3 orang bersaudara.

Riwayat Pendidikan :

- ☐ SDN Pucang Jajar II, Surabaya (1982 – 1984)
- ☐ SDN Kalisari II, Surabaya (1984 – 1987)
- ☐ Brunswick East Primary School, Melbourne, Australia (1987 – 1988)
- ☐ Brunswick High School, Melbourne, Australia (1988 – 1989)
- ☐ SMPN VI, Surabaya (1989 – 1992)
- ☐ SMAN II, Surabaya (1992 – 1995)
- ☐ Diterima di Politeknik Elektronika Negeri Surabaya-ITS melalui Ujian Masuk pada tahun 1995 dan mengambil jurusan teknik elektronika.