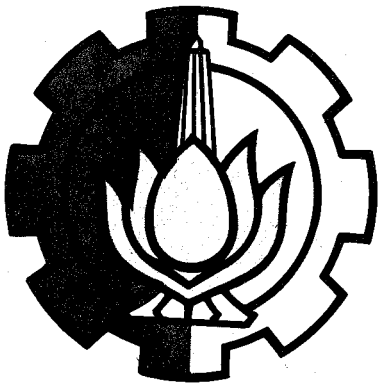
3100097009237

PERPUSTAKAAN
I T S
Tgl. Terima 22 SEP 1994
Terima Dari H
No. Agenda Prp. 2727

ALAT PENGUKUR OKSIGEN SATURASI DALAM DARAH DENGAN METODE "NON-INVASIVE"

RSE
621 391 6
Bud
a-1
1994



MILIK PERPUSTAKAAN
INSTITUT TEKNOLOGI
SEPULUH – NOPEMBER

Oleh :

Rohan Budiman

N R P : 2902201606

JURUSAN TEKNIK ELEKTRO
FAKULTAS TEKNOLOGI INDUSTRI
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
S U R A B A Y A

1994

ALAT PENGUKUR OKSIGEN SATURASI DALAM DARAH DENGAN METODE "NON-INVASIVE"

TUGAS AKHIR

Diajukan Guna Memenuhi Sebagian Persyaratan
Untuk Memperoleh Gelar
Sarjana Teknik Elektro
Pada
Jurusan Teknik Elektro
Fakultas Teknologi Industri
Institut Teknologi Sepuluh Nopember
Surabaya

Mengetahui / Menyetujui :

Dosen Pembimbing I	Dosen Pembimbing II
(Ir. Harmani Suhardjo)	(Ir. Djoko Purwanto)

SURABAYA
AGUSTUS, 1994

KATA PENGANTAR

Atas berkat Rahmat Allah Yang Maha Esa, maka penulis berhasil menyelesaikan Tugas Akhir yang berjudul :

Alat Pengukur Oksigen Saturasi Dalam Darah Dengan Metode "Non-Invasive"

Tugas akhir ini merupakan salah satu syarat yang harus ditempuh untuk meraih gelar kesarjanaan di Jurusan Teknik Elektro, Fakultas Teknologi Industri, Institut Teknologi Sepuluh Nopember Surabaya.

Dalam mengerjakan Tugas Akhir ini penulis banyak mendapat bantuan dan bimbingan dari berbagai pihak. Pada kesempatan ini penulis menyampaikan ucapan terima kasih kepada :

- Ir. Harmani Suhardjo, selaku dosen pembimbing yang telah banyak memberikan bimbingan dan pengarahan kepada penulis dalam perencanaan dan pembuatan alat serta penyusunan naskah Tugas Akhir ini.
- Ir. Djoko Purwanto, selaku dosen pembimbing yang telah memberikan dukungan dan bimbingan serta saran positif sehingga penulis dapt menyelesaikan Tugas Akhir ini.
- Ir. Soetikno, selaku Koordinator Bidang Studi Elektronika Jurusan Teknik Elektro, Fakultas Teknologi Industri ITS yang telah memberikan persetujuan penulis untuk melaksanakan Tugas Akhir ini.
- Dr.Ir. Shalehuddin, M Meng, selaku Ketua Jurusan Teknik Elektro, Fakultas Teknologi Industri ITS yang telah memberikan persetujuan

kepada penulis untuk melaksanakan Tugas Akhir.

- Seluruh staf pengajar dan admninistrasi Jurusan Teknik Elektro FTI ITS, yang telah membantu kelancaran pelaksanaan Tugas Akhir ini.
- Seluruh rekan-rekan mahasiswa Jurusan Teknik Elektro FTI ITS dan semua pihak yang telah turut membantu baik secara langsung maupun tidak langsung.

Akhir kata, penulis berharap semoga segala sesuatu yang telah dihasilkan dalam pelaksanaan Tugas Akhir ini dapat bermanfaat bagi kemajuan ilmu pengetahuan dan kesejahteraan umat manusia.

Surabaya, Agustus 1994

Penulis

ABSTRAK

Pengukuran Oksigen Saturasi adalah salah satu cara mengukur kadar oksigen dalam darah dengan jalan mengukur perbandingan hemoglobin (Hb) dan oksihemoglobin (HbO_2), yaitu dengan menggunakan dua berkas sinar yang akan diteruskan oleh hemoglobin dan oksihemoglobin sesuai dengan hukum Beer.

Mikrokontroler 8031 digunakan sebagai pengolah pusat untuk membaca hasil pengukuran dan menampilkan pada LCD dan untuk pengolahan lebih lanjut digunakan komputer IBM PC yang bisa berfungsi sebagai PC-based bedside monitor.

DAFTAR ISI

MILIK PERPUSTAKAAN
INSTITUT TEKNOLOGI
SEPULUH – NOPEMBER

DAFTAR GAMBAR

DAFTAR TABEL

BAB I

PENDAHULUAN

I.1. LATAR BELAKANG

Pengetahuan akan gas-gas dalam darah manusia sangat penting dalam dunia kedokteran. Dalam mendiagnosa dan mengobati pasien, dokter perlu mengetahui komposisi gas-gas dalam darah, terutama untuk pasien yang memerlukan pengorganisasian pernafasan.

Salah satu gas dalam darah yang penting diketahui adalah oksigen. Keadaan oksigen ini bisa diketahui dari tekanan parsialnya (PO_2) atau dari oksigen saturasi (SaO_2 atau SAT). Oksigen saturasi itu sendiri didefinisikan sebagai kemampuan darah dalam mengikat oksigen yang diperoleh dari proses respirasi.

Untuk mengetahui oksigen saturasi bisa didapat dengan cara mengambil sampel darah pasien dan kemudian dianalisa dengan fotometer medis. Tapi dengan cara ini tidak menguntungkan jika pasien dalam kondisi gawat atau pada bayi (*infant*) dan cara ini tidak menguntungkan, sehingga dikembangkan cara mengukur oksigen saturasi tanpa mengambil sampel (*non-invasive*) yang menggunakan sensor optis yang dipasang pada pasien. Dan hal ini dikenal dalam dunia kedokteran dengan istilah *Non-invasive Percutaneous Oxygen Saturasion Monitoring*.

I.2. RUANG LINGKUP

Pada tugas akhir ini akan dirancang suatu sistem berbasis mikrokontroler untuk mengukur oksigen saturasi dalam darah yang bersifat non-invasive, dengan menggunakan sensor optis.

Sistem dirancang agar dapat bekerja sendiri (stand alone) ataupun terhubung dengan komputer IBM PC sebagai pengumpul data disamping tempat tidur (PC based bedside monitoring).

Saat sistem berada pada mode stand alone sistem melakukan pengukuran sekali dan hasilnya ditampilkan pada layar peraga. Dan jika sistem dihubungkan dengan komputer IBM PC, maka pengambilan data diberikan oleh PC dan hasilnya diolah dan ditampilkan pada layar monitor.

Secara ringkas sistem yang dirancang sebagai berikut

- Berbasis mikrokontorler
- Bekerja pada mode berdiri sendiri (stand alone) atau bersama PC sebagai PC-based bedside monitor lewat saluran serial.
- Mengunakan metoda non-invasive, absorsi-spektrometri, berdasarkan penggunaan sensor optis.

I.3. SISTEMATIKA PEMBAHASAN

Dalam tugas akan pembahasan akan diuraikan dalam bab per bab sebagai berikut :

- BAB I Pendahuluan berisikan latar belakang dari alat yang akan dibuat, ruang lingkup dan sistematika pembahasan.

- BAB II Teori Penunjang akan menguraikan dasar teori pengukuran, dasar-dasar optis, teori dasar dari komponen utama yang akan digunakan, yaitu mikrokontroler 8031, ADC serta operasional amplifier.
- BAB III Perencanaan meliputi perencanaan perangkat keras, yaitu perencanaan rangkaian mikrokontroler, memori, ADC, osilator, sensor, penguat, filter, tampilan, papan kunci beserta decodernya dan perencanaan perangkat lunak baik perangkat lunak sistem mikrokontroler maupun perangkat lunak pada komputer IBM-PC.
- BAB IV Pengujian Peralatan beserta analisa hasil pengukuran.
- BAB V Penutup yang berisikan kesimpulan dan saran-saran.

BAB II

TEORI DASAR

Dalam bab ini akan dijelaskan teori-teori menjadi dasar dari pengukuran oksigen dalam darah, yang meliputi gas-gas dalam darah, dasar optik dan cara pengukuran oksigen secara invasive ataupun non-invasive. Serta teori-teori dasar dari sistem yang dirancang, diantaranya teori mikrokontroler dan komponen pendukung lainnya.

II.1. OKSIGEN DALAM DARAH

Fungsi utama darah adalah membawa oksigen (O_2) dari paru-paru menuju sel-sel tubuh dan membawa karbondioksida (CO_2) untuk dikeluarkan lewat paru-paru juga. Oksigen dibawa kejaringan oleh hemoglobin (Hb) dalam sel-sel darah merah[1]. Hemoglobin yang membawa oksigen disebut Oksihemoglobin (HbO_2) dan yang tidak membawa oksigen adalah hemoglobin (Hb).

Pengetahuan tentang konsentrasi oksigen dalam darah akan membantu dokter dalam mendiagnosa organ-organ pernapasan. Terutama tekanan parsial oksigen (PO_2) yang sering dipakai sebagai indeks untuk mengetahui keadaan pencampuran oksigen yang diperlukan untuk perawatan kehidupan. Dan pada manusia dewasa normal, misalnya, PO_2=100 mmHg maka berarti darah dalam

---

[1]Guyton, Athur C. Fisiologi Kedokteran I, Terjemahan Adji Dharma dan P.Lukmanto. EGC,Jakarta 1983 hal 40

keadaan setimbang dengan gas pada tekanan parsial oksigen 100 mmHg. Untuk itu diperlukan alat penganalisa darah untuk mengukur tekanan parsial Oksigen (PO_2), tekanan parsial karbondioksida PCO_2 dan pH darah.

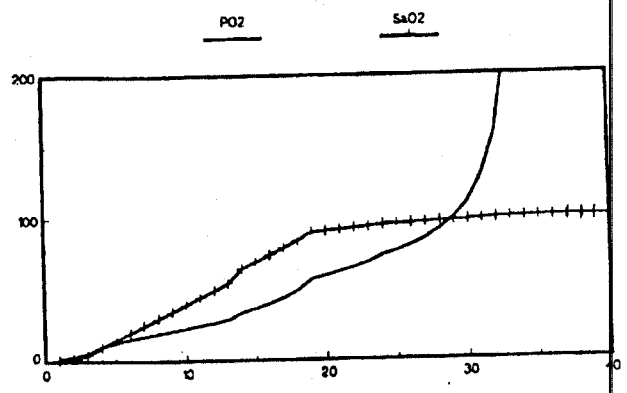
Cara menganalisa darah dengan jalan mengambil sampel dari tubuh manusia disebut dengan istilah merusak (invasive). Dalam bidang medis alat ini disebut instrumen dengan metode invasive.

Dalam keadaan tertentu, misalnya pasien dalam keadaan gawat ataupun pada bayi yang kondisi peredaran darahnya sangat payah, pengambilan sampel darah secara berkala untuk diukur akan memberatkan pasien. Lebih-lebih lagi jika menggunakan kateter (catheter) yang dimasukkan dalam arteri. Pendarahan dari letak kateter, amenia karena pengambilan darah atau lainnya akan berkembang dalam keadaan tertentu. Dan hal ini tidak menguntungkan bagi kesehatan pasien.

Keadaan ini dikembangkan untuk membuat sistem pengukuran yang tidak memerlukan pengambilan sampel darah. Dimana sistem dapat memonitor keadaan darah melewati kulit dan jaringan tubuh (percutaneous monitoring). Dan sistem ini dalam dunia medis disebut sistem non-invasive (tidak merusak).

Selain PO_2, PCO_2, dan pH dikenal juga prosentasi Oksigen Saturasi (SaO_2) yang menunjukkan prosentase oksigen yang terkombinasi dengan hemoglobin dalam darah. Baik SaO_2 maupun PO_2 sebenarnya menunjukkan atau mengukur hal yang sama, yaitu jumlah oksigen dalam darah, atau dengan kata lain jika sudah diketahui salah satunya (misal SaO_2), maka yang lainnya

tidak diperlukan lagi[2)].



Gambar 2.1.

Perbandingan SaO_2 dan PO_2

II.2. DASAR-DASAR OPTIS

Cahaya adalah gelombang elektromagnet yang mempunyai spektrum yang sangat luas, yaitu antara panjang gelombang 6.10^{-8} m sampai 10^{-7}m. Sinar atau radiasi elektromagnetik yang bersifat partikel dan gelombang. Fenomena refraksi, refleksi, interferensi konstruktif dan destruktif merupakan contoh perilaku radiasi gelombang elektromagnetik sebagai gelombang. Sedangkan pengaruh photoelektrik merupakan contoh dari radiasi elektromagnetik

---

[2)]Yutaka Usuda and Katsuo Numata Respiratory Management in RCU, Yokohama City University School of Medicine and The University of Tokyo School on Medicine, hal 400

bersifat partikel. Radiasi elektromagnetik yang bersifat gelombang akan mengikuti hukum-hukum atau poskulat dalam gelombang, yaitu hukum Herzt. Dalam hal ini hubungan antara panjang gelombang dan frekwensi dan kecepatan adalah sebagai berikut :

$$\lambda f = \frac{c}{n}$$

$$\lambda = \frac{c}{nf}$$

dimana λ = panjang gelombang (nm)

c = kecepatan cahaya (3×10^8 m/det)

f = frekwensi (Herzt)

n = indeks relatif

Jadi hubungan antara frekwensi dengan panjang gelombang berbanding terbalik. Cahaya pada daerah tampak (visible) pada setiap panjang gelombang mempunyai warna yang berbeda pula. Untuk cahaya dengan panjang gelombang tunggal mempunyai warna yang tunggal pula, dan disebut cahaya monokromatik.

Sesuai sifat gelombang elektromagnet cahaya mengandung foton-foton yang mempunyai partikel-partikel cahaya yang dipancarkan secara diskrit. Planck menyatakan bahwa setiap foton hasil radiasi dan frekwensi f mempunyai

energi sebesar[3] :

$$E = h f$$

dimana

E = Energi photon (ergs)

h = adalah konstanta planck

= $6{,}554\ 10^{-27}$ erg.detik

Pada cahaya monokromatik dianggap mempunyai pancaran foton-foton yang mempunyai energi yang sama.

Bila suatu berkas cahaya melewati suatu zat dalam larutan, akan diserap bila energi dari foton-foton dalam berkas cahaya tersebut sama dengan energi transisi yang diserap zat tadi. Maka bila cahaya dengan panjang gelombang yang berubah-ubah melalui suatu zat, penyerapan yang terjadi akan berubah-ubah pula dan akan mencapai suatu harga maksismum pada suatu panjang gelombang dimana energi foton sama dengan energi transisinya.
Jika penyerapan cahaya diukur pada suatu panjang gelombang tetap, hasil yang diperoleh akan tergantung pada tiga faktor sebagai berikut :

1. Ketebalan bahan yang dilalui cahaya.
2. Konsentrasi bahan.
3. Effisiensi bahan tersebut dalam menyerap cahaya.

---

[3]Krane, Keneth S. Modern Physics, John Wiley & Sons,1983 hal 432

MILIK PERPUS... INSTITUT TEKNOLOGI SEPULUH – NOPEMBER

Pada penyerapan cahaya monokromatis berlaku hukum Beer yang dapat didefinisikan sebagai berikut :

"Suatu zat hanya akan dapat menyerap cahaya yang mempunyai panjang gelombang tertentu. Dan pada suatu lapisan tipis cahaya yang diserap sebanding dengan intensitasnya dan sebanding pula dengan jumlah molekul penyerap".

Untuk cahaya tampak (visible) penyerapan terjadi pada larutan, sedang pada infra merah penyerapan terjadi pada solid sampel. Penyerapan yang masing-masing berbeda untuk berbagai zat sesuai dengan tiga faktor diatas memungkinkan kita untuk dapat mengetahui konsentrasi zat/sampel.

II.3. PENGUKURAN KONSENTRASI ZAT SECARA OPTIS

Berdasarkan hukum Beer dibuat suatu alat yang dapat mengukur konsentrasi suatu zat secara optis. Alat tersebut disebut spektrometer. Dengan alat ini konsentrasi suatu senyawa dalam larutan dapat ditentukan. Pengukuran dilakukan pada daerah cahaya tampak (visible) dengan panjang gelombang sekitar 380 nm -780 nm.

Pengukuran dengan spektrometer adalah sebagai berikut :

Sampel yang akan diukur dimasukkan dalam tabung reaksi. Untuk darah harus dipisahkan dulu antara plasma yang diukur dengan sel-selnya dengan cara sentrifugal. Tabung yang berisi sampel disinari dengan cahaya yang mempunyai panjang gelombang tertentu. Intensitas cahaya datang adalah I_o sedang intensitas cahaya yang diteruskan adalah I_s. Untuk menghilangkan

faktor pemantulan cahaya oleh cawan dan penyerapan cahaya oleh pelarut, maka digunakan cawan yang berisi pelarut saja sebagai referensi. Seperti pada cawan sampel, maka cawan ini juga disinari dengan cahaya yang sama, dan cahaya yang diteruskan disimbolkan I_r. Maka cahaya yang ditransmisikan oleh senyawa dalam larutan didefinisikan sebagai I_s/I_r (transmitan = T). Karena harga transmitan berbanding terbalik secara logaritmik dengan konsentrasi senyawa, maka bentuk ini dikonversikan menjadi **absorban** (A), dengan [4]:

$$A = -\log\left(\frac{I_s}{I_r}\right) = \log T = 2 - \log \%T$$

sedang

$$I_s = I_o\, 10^{-\alpha c x}$$

dimana α = absorsivitas

c = konsentrasi (mol/liter)

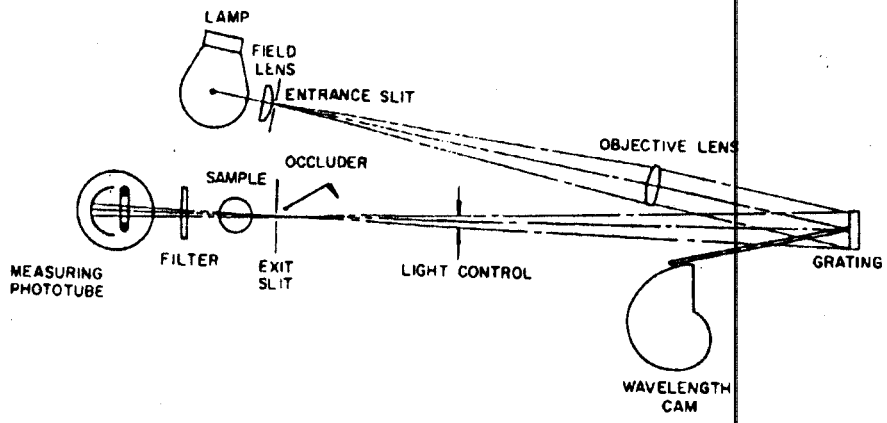
x = lebar berkas cahaya yang lewat cawan.

Alat ini menggunakan filter (fotometer filter atau kalorimeter) untuk memilih panjang gelombang tertentu. Kisi pemantul digunakan untuk memilih bandwidth yang paling sempit sehingga alat ini disebut *Spektrofotometer*. Intensitas sumber cahaya dikontrol untuk menentukan level cahaya yang sesuai untuk foto. Selain itu pada alat ini juga digunakan sebuah

---

[4]Webster, John G ed, Medical instrumentation Application and Design. Houghton Mifflin Company, 1978 hal 517

monokromator yang berfungsi untuk membuat cahaya yang menyinari sampel dengan satu panjang gelombang saja (monokromatik). Hal ini disebabkan karena Hukum Beer hanya berlaku untuk cahaya monokromatik. Sinar yang diteruskan dikonversikan menjadi sinyal listrik oleh tabung foto (Phototube).



Gambar 2.2
Spektrofotometer

Untuk mengatasi masalah kondensasi uap metalik, minyak, debu dan panas yang mempengaruhi sistem optik digunakan spektrometer dua berkas. Disini berkas cahaya dipisah menjadi dua, satu untuk menyinari sampel dan satunya lagi untuk menyinari referensi. Kemudian keluaran digunakan untuk menghitung transmitan.

Alat ini bisa dipakai untuk mengukur oksigen saturasi, tetapi mempunyai kekurangan, yaitu memerlukan sampel darah untuk diukur.

II.4. PENGUKURAN OKSIGEN SATURASI.

Prinsip dasar dari alat untuk mengukur oksigen saturasi atau oksimeter adalah untuk mengukur perbandingan antara hemoglobin (Hb) dengan oksihemoglobin (HbO_2). Untuk darah terhemolisa, menurut hukum Beer, *absorbance* diberbagai panjang gelombang adalah[5] :

$$A(\lambda) = WL\,[a_o(\lambda)C_0 + a_r(\lambda)C_r]$$

dimana

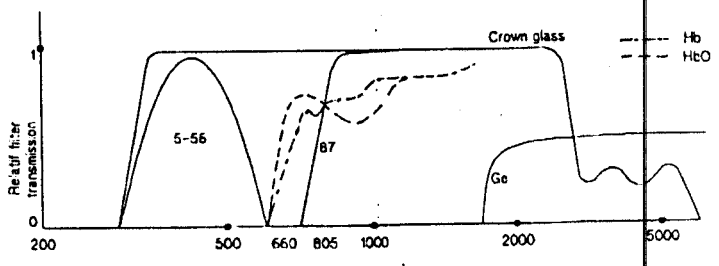
W = berat hemoglobin persatuan volume

L = lebar zat yang dilalui sinar

a_o, a_r = absortivitas HbO_2 dan Hb

C_o, C_r = konsentrasi relatif dari HbO_2 dan Hb

($C_o + C_r = 1$)



Gambar 2.3
Karakteristik Spektrum Sumber Cahaya dan Filter

[5] Ibid, hal 94

Pada gambar menunjukkan bahwa a_o dan a_r akan sama panjang gelombang 805 nm, yang dinamakan panjang gelombang isobetic. Jika panjang gelombang ini disebut λ_2, maka :

$$WL = \frac{A(\lambda_2)}{2a(\lambda_2)}$$

dimana

$$a(\lambda_2) = a_o(\lambda_2) = a_r(\lambda_2)$$

sehingga

$$A(\lambda) = \frac{A(\lambda_2)}{2a(\lambda_2)}[a_0(\lambda)C_o + a_r(\lambda)C_r]$$

Dengan diukurnya absorban dari gelombang yang kedua λ_2, Oksigen saturasi dinyatakan dengan :

$$C_o = x + y.\frac{A(\lambda_1)}{A(\lambda_2)}$$

dengan x dan y adalah konstanta yang tergantung karakteristik darah. Dalam prakteknya λ_1 dipilih dimana panjang gelombang tersebut perbedaan antara a_o dan a_r maksimum, yaitu pada panjang gelombang 660 nm.

Polanyi dan Hehir menurunkan metode empiris dari pengukuran pantulan cahaya, dengan membandingkan pantulan dari dua buah cahaya

dengan panjang gelombang yang khusus. Satu adalah sinar merah, yang kedua adalah mendekati infra merah (near infra red). Oksigen saturasi didekati dengan hubungan dibawah[6] :

$$SaO_2 = A + B\frac{R_{805}}{R_{665}}$$

dimana R_{805} dan R_{665} adalah pantulan (reflectant) sinar dengan panjang gelombang 805 nm dan 665 nm. A dan B adalah konstanta-konstanta yang tergantung dari karakteristik darah dan sensor. Metode ini belum sempurna karena adanya efek ketidaklinieran dalam penghamburan (scattering) yang muncul pada plasma dan sel darah merah (red Blood cell/RBC). Untuk pittman dan Duling mengembangkan metode gelombang ketiga pada daerah isobetic. Sedang Lubbers mengembangkan metode lain, yaitu dengan analisa multikomponen untuk mengatasi ketidaklinieran tersebut.

Setsuo Takatani et.al, mengembangkan metode lain yang mengacu pada metode-metode tersebut diatas. Metode menggunakan 2 buah sinar merah dan dekat infra merah (near infra red). Metode ini tidak terlalu rumit seperti metode Pittman dan Duling yang memerlukan gelombang ketiga, ataupun dengan metode Lubbers yang menggunakan analisa multikomponen.

Pendekatan yang digunakan Setsuo Takatani et.al dalam mengukur oksigen saturasi adalah sebagai berikut :

---

[6]Setsuo Takatani, Hiroyuki Noda, Hisateru Takatano, dan Tetsuzo Akutsu. A Miniature Hybryd Reflection Type Optical Sensor for Measurement of Hemoglobin Content and Oxygen Saturation of Whole Blood. IEE Trans. on Biomedical Engineering, vol 35, no.3, March 1988

$$SaO_2 = A + B\frac{R_{795} + C_1}{R_{665} + C_2}$$

R_{795} dan R_{665} adalah pantulan sinar dengan panjang gelombang 795 dan 665 nm. Dengan A dan B konstaanta-konstanta yang tergantung karakteristik darah dan sensor. Sedang C_1 dan C_2 juga konstanta-konstanta yang diperlukan untuk menghasilkan data yang optimal dalam rentang luas hematrocit.

II.5. KARAKTERISTIK PENGUKURAN OKSIGEN SATURASI

Beberapa keuntungan pengukuran Oksigen saturasi (SaO_2) dibandingkan dengan PO_2, sebagai berikut[7] :

a. Sifat kontinyuitas/kesinambungan

SaO_2 : Tidak memerlukan pemindahan probe, sehingga memungkinkan untuk pengukuran secara kontinyu.

PO_2 : Daerah pengukuran mesti dipindahkan setelah beberapa saat.

b. Sifat non-invasive

SaO_2 : Tidak ada bahaya terbakar/melepuh karena tidak memerlukan pemanasan. Serta tidak memerlukan *adesive* untuk menempelkan probe.

---

[7]Katsuyuki Miyasaka. Noninvasive Oximetre of Oxygen in Blood. Anesthesiological Departement National Pediatrics Hospital.

PO_2 : Bisa terbakar/melepuh karena memerlukan pemanasan, dan kulit bisa rusak terkena *adesive* yang perlu diberikan pada waktu pemasangan probe.

c. Kesederhanaan

SaO_2 : Tidak terpengaruh oleh gas anesti serta mudah penggunaannya karena tidak memerlukan kalibrasi.

PO_2 : Bisa terpengaruh gas anesti. Penanganannya rumit.

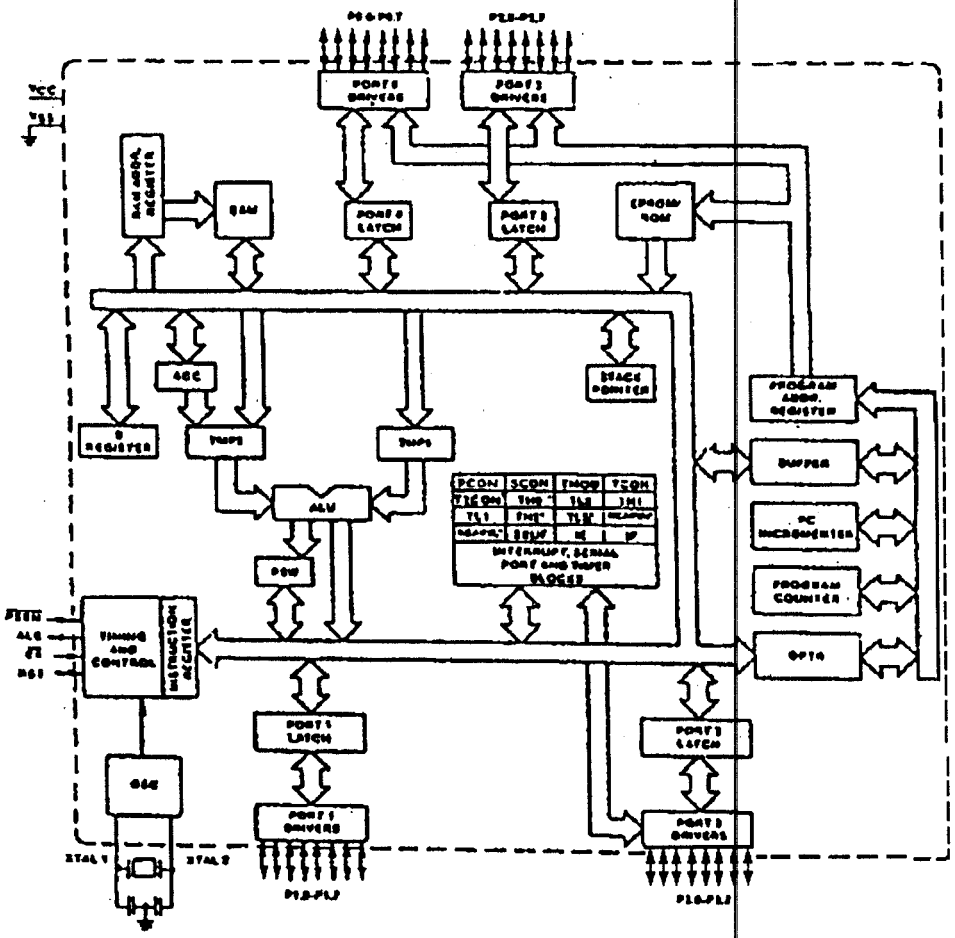
d. Kehandalan

SaO_2 : Terpengaruh oleh perpindahan. Tidak memerlukan kalibrasi. Gelombang pulsa dapat diukur secara simultan dengan kehandalan yang tinggi

PO_2 : Lebih tidak terpengaruh oleh perpindahan. Memerlukan kalibrasi. Kurang handal dalam pengukuran karena terpengaruh oleh sirkulasi, ketebalan kulit dll.

II.6. MIKROKONTROLER 8031

Mikrokontroler 8031 termasuk salah satu keluarga MCS-51 yang merupakan chip tunggal 8 bit, yang diproduksi oleh Intel. Pada gambar 2.4 memperlihatkan arsitektur diagram blok dari keluarga MCS-51.



Gambar 2.4[8)]
Arsitektur Diagram Blok MCS-51

Spesifikasi umum yang dimiliki oleh mikrokontroler 8031 adalah sebagai berikut :

- 8-bit Unit Pemproses Pusat (CPU)
- Rangkaian clock oscillator
- Kemampuan yang luas untuk proses boolean

[8)]Intel, Embededed Controller Handbook, Santa Clara, 1988, hal 10-2

- 32 jalur input ouput
- 128 byte data RAM
- 2 buah pewaktu/pencacah 16-bit
- Port seial fullduplex
- 5 sumber struktur interupsi dengan 2 tingkat prioritas
- 64k alamat untuk memori program eksternal
- 64K alamat untuk memori data eksternal

II.6.1. FUNGSI PIN-PIN PADA MIKROKONTROLER 8031

Pada gambar 2.5 memperlihatkan konfigurasi pin-pin pada mikrokontroler 8031. Fungsi pin-pin pada 8031 dapat dikelompokkan menjadi pin sumber tegangan, pin kristal, pin kontrol, pin input-output dan pin interupsi.

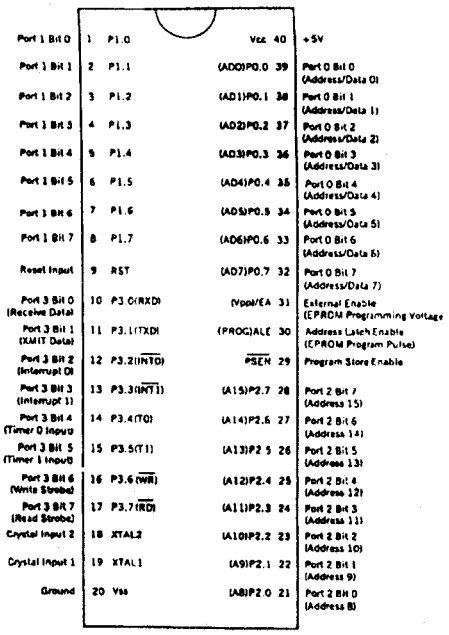
Fungsi dari tiap pin pada mikrokontroler 8031 adalah sebagai berikut :

a. Vcc pin positif sumber tegangan = 5 Volt DC.

b. Vgg pin ground sumber tegangan.

c. Port 0

Port 0 merupakan port input output 8-bit dua arah. Port ini dapat digunakan sebagai multipleks bus alamat rendah dan bus data selama adanya akses kememori program eksternal atau ke memori data eksternal. Port 0 ini mampu menggerakkan 8 buah IC TTL jenis LS.



Gambar 2.5[9)]
Konfigurasi Pin Pada Mikrokontroler 8031

d. Port 1

Port1 merupakan port input output 8-bit dua arah. Setiap pin dapat digunakan sebagai masukan atau keluaran tanpa tergantung dari pin lainnya. Port ini mampu menggerakkan 4 buah IC TTL jenis LS.

e. Port 2

Port2 merupakan port input output 8-bit dua arah. Port ini dapat digunakan sebagai bus alamat tinggi selama adanya akses ke memori program eksternal atau memory data eksternal dan mampu menggerakkan 4 buah IC TTL jenis LS.

[9)]Ibid, hal 10-3

f. Port 3

Port 3 merupakan port input output 8-bit dua arah. port ini dapat digunakan juga untuk berfungsi sebagai pin-pin istimewa bagi 8031 seperti sebagai berikut :

- P3.0 (RxD) : Masukan penerimaan data serial.
- P3.1 (TxD) : Keluaran pengiriman data serial.
- P3.2 (INT0) : Interupsi 0 eksternal.
- P3.3 (INT1) : Interupsi 1 eksternal.
- P3.4 (T0) : Masukan eksternal pewaktu/pencacah 0.
- P3.5 (T1) : Masukan eskternal pewaktu/pencacah 1.
- P3.6 (WR) : Strobe penulisan memori data eksternal.
- P3.7 (RD) : Strobe pembacaan memori data eksternal.

g. RST/Vpd

Pin ini berfungsi untuk mereset sistem 8031. Perubahan taraf tegangan dari rendah ketinggi akan mereset mikrokontroler.

h. ALE/PROG

ALE (address Latch Enable) digunakan untuk mengunci alamat rendah pada saat pengaksesan memori program eksternal. Pin ini mampu menggerakkan 8 buah IC TTL jenis LS.

i. PSEN

PSEN (Program Store Enable) adalah sinyal kendali untuk strobe memory program eksternal.

j. EA/Vpp

Untuk pengoperasian 8031, pin ini harus dihubungkan dengan ground agar dapat menjalankan instruksi dari memori program eksternal.

k. XTAL1

Pin ini merupakan masukan ke penguat osilator berpenguatan tinggi. Pin ini dihubungkan dengan kristal atau sumber osilator eksternal.

l. XTAL2

Pin ini merupakan keluaran dari penguat osilator. Pin ini dihubungkan dengan kristal atau ground jika menggunakan sumber osilator eksternal.

II.6.2. PERANGKAT KERAS UNIT PEMPROSES PUSAT (CPU)

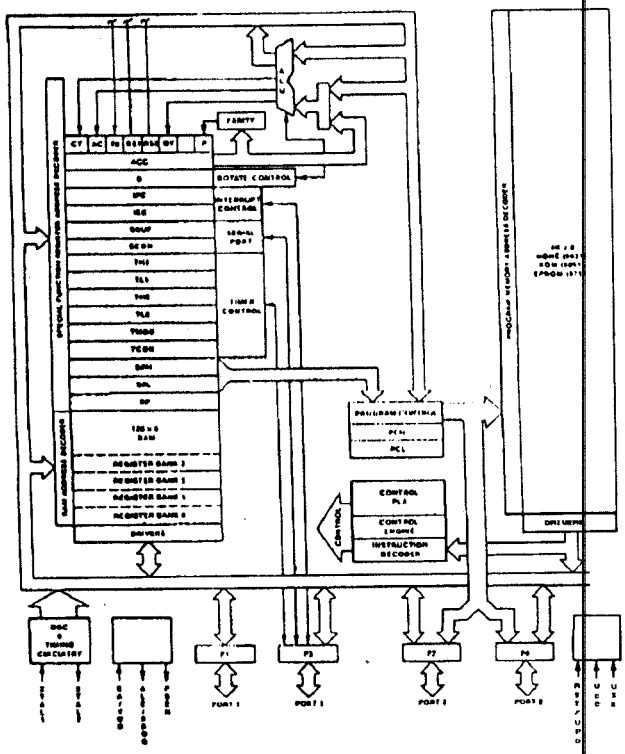
Pada gambar memperlihatkan diagram blok dari mikrokontroler 8031. Fungsi dari masing-masing blok pada gambar 2.6 tersebut adalah sebagai berikut :

a. Program Counter (PC).

PC merupakan register 16 bit yang digunakan untuk mengatur urutan pengambilan instruksi yang akan dijalankan.

b. Dekoder Instruksi.

Bagian ini menterjemahkan setiap instruksi program dan membangkitkan sinyal yang akan mengatur fungsi dari setiap bagian pada CPU.



Gambar 2.6[10)]
Diagram Blok Mikrokontroller 8031

c. RAM Data Internal.

RAM data internal yang dimiliki 8031 sebesar 128 byte yang terdiri dari :

- Register Bank

  Terdapat empat buah register bank, setiap bank terdiri dari 8 buah register R_0 - R_7.

- 128 addressable bits.

  Bagian ini berlokasi pada alamat 20H sampai 2FH.

[10)]Intel, MCS-51 Family of Single Chip Microcomputer User's Manual, Santa Clara, 1981, hal 2-2

- Stack

  Stack dapat ditempatkan pada RAM data internal dengan jumlah stack sebesar RAM data internal tersebut.

- Register B :

  Register ini digunakan bersama resgister A untuk instruksi perkalian dan pembagian.

- Register Program Status Word (PSW) :

  Register ini meliputi bit-bit carry (CY), auxilary carry (AC), flag 0 (F0), pemilih register bank (RS0 dan RS1), over-flow (OV) dan parity flag (P).

  Flag CY,AC dan OV pada umumnya menandakan keadaan dari operasi aritmatika yang terakhir. Flag P merupakan parity dari register A. Flag carry juga digunakan sebagai akumulator Boolean untuk operasi bit. Dua bit pemilihan register bank RS0 dan RS1 memungkinkan pengaksesan pada register bank yang perlainan, yakni :

  - (0,0) ——————————> Bank 0 (00H - 07H)
  - (0,1) ——————————> Bank 1 (08H - 0FH)
  - (1,0) ——————————> Bank 2 (10H - 17H)
  - (1,1) ——————————> bank 3 (18H - 1FH)

- Penunjuk Stack (Stack Pointer / SP) :

  SP adalah register 8 bit yang menunjukkan (push) ke stack, juga

sebagai alamat dari data selanjutnya yng dikeluarkan (pop). Nilai SP akan bertambah selama push.

- Penunjuk Data (Data Pointer / DTPR) :

  DTPR besarnya 16 bit yang terdiri dari Penunjuk Data Tinggi (Data Pointer High /DPH) sebesar 8 bit dan Penunjuk Data Rendah (Data Pointer Low / DPL) sebesar 8 bit. DTPR digunakan untuk pengalamatan register tak langsung untuk memindahkan konstanta-konstanta memori program eksternal, pemindahan data pada memori data eksternal dan untuk percabangan (branch) program sampia 64 k byte.

- Port 0, 1, 2 dan 3 :

  Ke empat buah port tersebut menghasilkan 32 jalur input output untuk berhubungan ke luar. Seluruh port dapat dialamati secara byte atau bit. Port 0 (P0) dan port 2 (P2) dapat digunakan untuk menambah jumlah memori luar. Port 3 (P3) berisi sinyal kontrol khusus seperti sinyal baca dan tulis. Port 1 (P1) digunakan hanya untuk input output.

- Register Prioritas Interupsi (Interrupt Priority Register / IP) :

  IP berisi bit-bit kontrol untuk mengaktifkan interupsi pada taraf yang diinginkan.

- Register Pemungkin Interupsi (Interupt Enable Register / IE) :

  IE menyimpan bit-bit untuk mengaktifkan ke lima sumber interupsi yang berisikan bit untuk menghidupkan/mematikan (enable/disable)

setiap sumber interupsi.

- Register Mode Pewaktu/Pencacah (Timer/Counter Mode Register/TMOD) :

  Bit-bit pada register TMOD digunakan untuk memilih pewaktu/pencacah yang akan bekerja. Setiap pewaktu/pencacah memiliki register tersendiri untuk menyimpan harga hitungannya. Untuk fungsi pewaktu, isi register akan ditambah satu untuk setiap siklus mesin (machine cycle). Setiap siklus mesin terdiri dari 12 periode osilator, sehingga keccepatan hitungnya sama dengan 1/12 frekwensi osilator. Untuk fungsi pencacah isi register akan ditambah satu setiap perubahan dari taraf tegangan rendah ketaraf tinggi yang dikenakan pada pin T0 untuk pencacah 0 dan T1 untuk pencacah 1.

- Register Kontrol Pewaktu/pencacah (Timer/Counter Control Register / TCON) :

  Semua pewaktu/pencacah dikontrol oleh bit-bit dari register TCON. Bit-bit mulai/berhenti (start/stop) untuk semua pewaktu/pencacah, flag-flag over-flow dan permintaan interupsi disimpan dalam TCON.

- Register Pewaktu/Pencacah Tinggi 0 dan 1 (TH0 dan TH1), Register Pewaktu/Pencacah Rendah 0 dan 1 (TL0 dan TH1) :

  Untuk dua buah pewaktu/pencacah (0 dan 1) 16 bit mempunyai empat lokasi register, register-register ini dapat dibaca dan ditulisi TH0 dan Th1 digunakan untuk 8 bit tinggi dari pewaktu/pencacah 0 dan 1 serta TL0 dan TL1 digunakan untuk 8 bit rendah dari

pewaktu/pencacah 0 dan 1.

- Register Kontrol Serial (Serial Control Register / SCON) :

  SCON mempunyai bit-bit enable untuk penerimaan port serial. Pemilihan mode operasi dari port serial juga dilakukan dengan bit-bit pada register tersebut.

- Penyangga Data Serial (Serial Data Buffer / SBUFF) :

  SBUFF digunakan untuk menampung data masukan atau keluaran dari port serial yang tergantung pada kerja dari port serial menerima atau mengirimkan data.

Pada tabel 2.1 diperlihatkan bentuk tabel dari register-register fungsi khusus (SFR) yang terdapat mikrokontroler 8031.

e. Bagian Aritmatika.

Bagian Aritmatika dari prosesor membentuk bebrapa fungsi manipulasi data yang dilaksanakan oleh ALU (Arithmatic Logic Unit)., Register A, Register B dan PSW.

f. Rangkaian Osilator.

Rangkaian yang terdapat di dalam serpih adalah rangkaian paralel anti-resonansi dengan batas frekwensi mulai dari 1,2 sampai dengan 12 MHz.

g. Prosesor Boolean.

Prosesor Boolean adalah prosesor bit yang berdiri sendiri, yang memiliki pasangan instruksi tersendiri, akumulator. Bit Ram yang dapat dialamati (Bit Addressable RAM) dan terminal masukan/keluaran.

fungsi lainnya seperti rotate, clear, complement dan lain-lain. ALU juga dapat membuat keputusan kondisi suatu percabangan (conditional branching dicisions), dan memberikan data path dan register-register sementara yang digunakan untuk transfer data dalam sistem. Instruksi-instruksi lainnya dibuat dari fungsi-fungsi dasar ini.

Operasi-operasi dasar digabungkan dan dikombinasikan dengan logika yang diperlukan untuk membuat instruksi-instruksi kompleks seperti meng-increment register terpisah 16 bit. Sebagai contoh untuk mengeksekusi satu bentuk instruksi compare, 8031 meng-increment program counter tiga kali, membaca tiga byte dari memori program, menghitung alamat register dengan operasi logika, dua kali membaca memori data internal, membuat perbandingan artimatika dari dua buah variable, menghitung 16 bit alamat tujuan dan memutuskan apakah melakukan percabangan atau tidak yang semuanya itu hanya dilakukan dalam 24 periode osilator

ALU juga dapat memanipulasi satu bit data sama baiknya dengan delapan bit data. Bit-bit tunggal dapat di set, cleared, complement, dipindahkan, di-test dan digunakan dalam komputasi logika.

ALU dengan kemampuan yang tinggi ini menyebabkan 8031 dapat melakukan operasi kontrol secara real-time dan algoritma data yang intensif. Operasi-operasi terpisah sebanyak 51 buah memindahkan dan memanipulasi tiga tipe data yaitu boolean (1-bit), byte (8 bit) dan alamat (16 bit). Ada sebelas mode pengalamatan, yaitu tujuh untuk data, empat untuk kontrol urutan program. Operasi-operasi umumnya membolehkan beberapa mode

pengalamatan.

II.6.4. ORGANISASI MEMORI

Mikrokontroler 8031 memiliki pengalamatan yang membedakan alamat memori program dengan alamat memori data. Alamat memori program yang dapat digunakan adalah 64 Kbyte. Memori data dapat diperluas mencapai 64 Kbyte pada RAM, sebagai tambahan dari 128 byte yang sudah ada dalam chip. Ke 128 byte data RAM internal dapat dibagi menjadi tiga segmen, yaitu segmen Register Bank, segmen Bit Addressable dan segmen Scratch Pad. RAM tersebut dapat diakses dengan cara pengalamatan langsung atau pengalamatan tak langsung, seperti yang diperlihatkan pada gambar 2.7 .

- Daerah Register Bank 0 -3 :

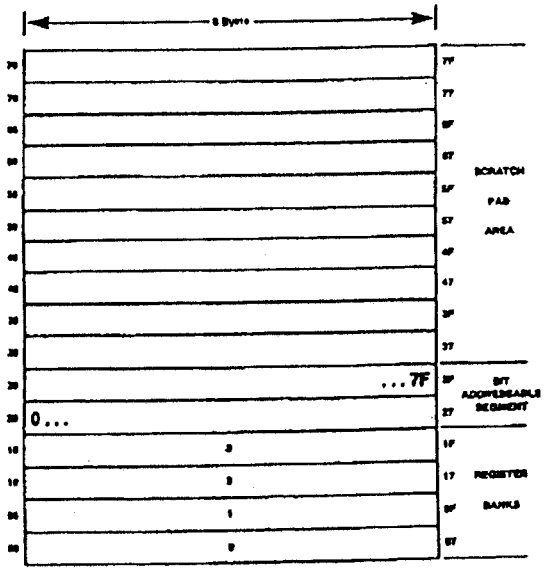
  Register Bank 0 - 3 menempati alamat 0 sampai 1FH (32 byte), yang masing-masing register bank terdiri 8 byte register dari 0 sampai 7.

Reset akan menginisialisasi Stack pointer ke alamat 07H dan akan mulai pada alamat 08H yang merupkan alamat register pertama (R0) dari Register Bank 1. Sehingga untuk penggunaan lebih dari satu bank register, maka SP harus mendapat inisialisasi ke alamat yang berbeda pada RAM internal yang tidak digunakan untuk penyimpanan data.

- Daerah Bit Addressable :

  Pada segmen ini terdapat 16 byte dengan alamat 20H - 2FH. Masing-masing ke 128 bit dari segment ini dapat dialamatkan secara langsung (0 - 7FH).

Pengalamatan bit-bit ini dapat dilakukan dengan dua cara. Cara pertama secara langsung (0 - 7FH). Cara kedua dengan menggunakan sesuai dengan byte dari 20H - 2FH, maka bit 0 - 7 dapat dialamatkan sebagai bit 220.0 - 20.7 dadn bit 8 - FH dengan 21.0 - 21.7 dan seterusnya.



Gambar 2.7.[12)]
128 byte Data Dengan Pengalamatan Langsung Dan Tak Langsung

Setiap dari 16 byte pada segment ini juga dapat dialamatkan seperti sebuah byte.

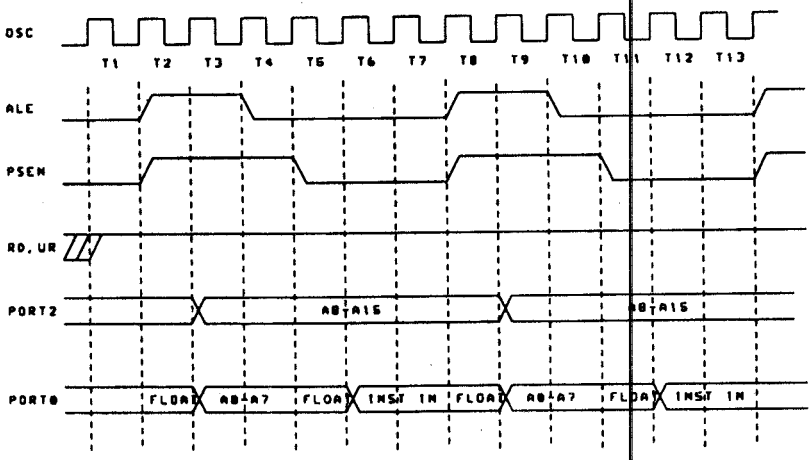
- Daerah Scratch Pad :

  Byte 30H sampai 7FH yang merupakan daerah Scraccth Pad yang tersedia untuk digunakan sebagi tempat data RAM.

[12)]Op.cit., Intel, 1988, hal 9-6

II.6.5. PENGAKSESAN MEMORI EKSTERNAL.

Untuk dapat mengakses memori program eksternal, pin EA mikrokontroler 8031 harus dihubungkan pada taraf tegangan rendah. Saat mengakses memori luar mikrokontroler menggunakan Port 0 untuk bus alamat rendah yang dimultipleks dengan bus data sedangkan Port 2 untuk bus alamat tinggi.



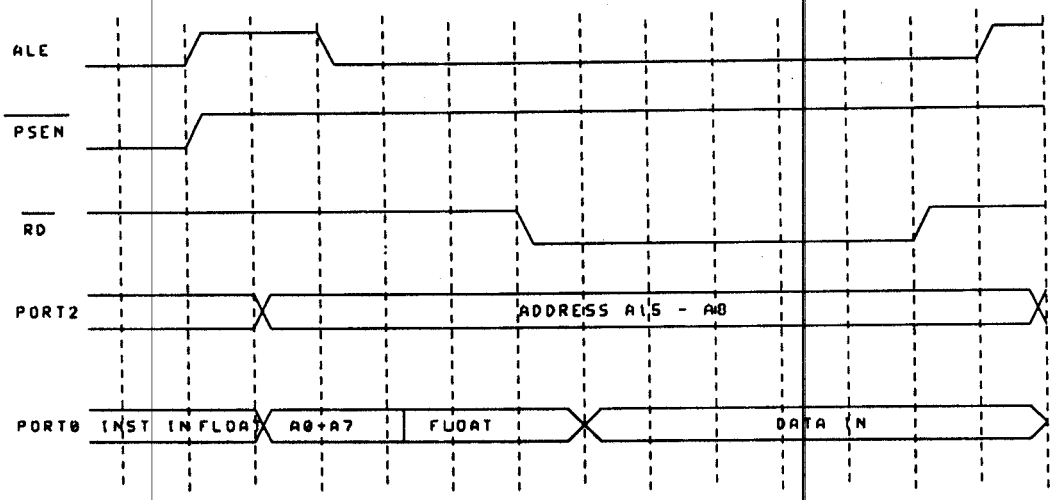
Gambar 2.8.[13)]
Siklus Baca Memori Program Eksterna

Sinyal ALE digunakan untuk mengerendel bus alamat rendah dari port 0 ke memori luar, sehingga antara bus alamat rendah dengan bus data dapat dipisahkan. Pada saat membaca memori program eksternal maka sinyal PSEN aktif, sedangkan sinyal WR dan sinyal RD tidak dalam keadaan aktif pada siklus pembacaan memori program eksternal ini. Setiap siklus baca dari memori program eksternal memerlukan 6 periode osilator. Pada gambar

---

[13)]Ibid., hal 10-21

diperlihatkan siklus baca memori program eksternal tersebut.



Gambar 2.9.[14)]
Siklus Baca Memori Data Eksternal

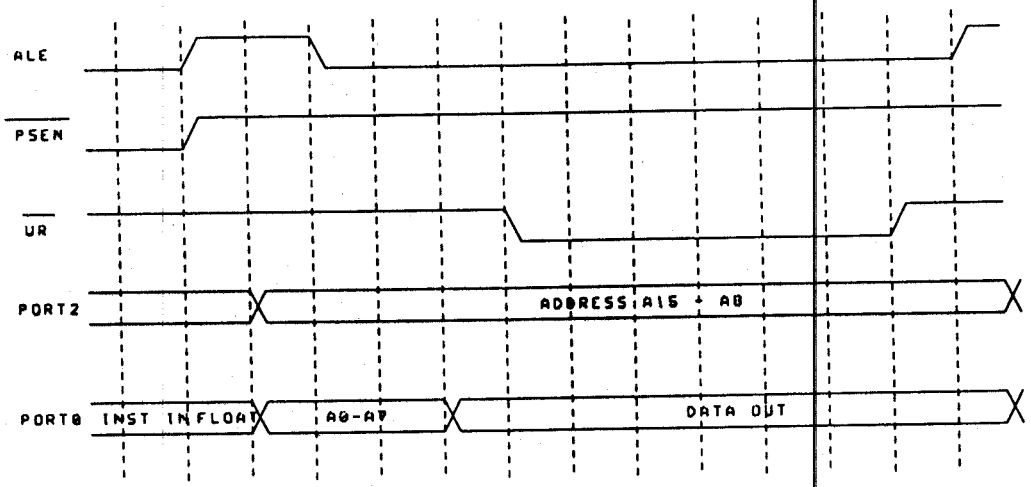
Untuk pembacaan memori data eksternal maka sinyal RD akan aktif, sedangkan sinyal PSEN dan sinyal WR tidak dalam keadaan aktif pada siklus pembacaan memori data eksternal ini. Setiap siklus baca memori data eksternal memerlukan 12 periode osilator. Pada gambar diperlihatkan siklus baca memori data eksternal.

Apabila pada saat penulisan memori data eksternal maka sinyal WR akan aktif, sedangkan sinyal PSEN dan sinyal RD tidak dalam keadaan aktif pada siklus penulisan memori data eksternal ini. Setiap siklus tulis memori data eksternal memerlukan 12 periode osilator. Pada gambar diperlihatkan

---

[14)]Ibid., hal 10-21

siklus tulis memori data eksternal.



Gambar 2.10[15)]
Siklus Tulis Memori Data Eksternal

Pada pengaksesan memori program eksternal selalu menggunakan pengalamatan 16 bit. Untuk pengasesan memori data eksternal digunakan istruksi MOVX. Instruksi MOVX ini dapat dilakukan dengan cara pengalamatan 16 bit jika menggunakan DTPR dengan jangkauan 64 Kbyte atau pengalamatan 8 bit dengan menggunakan register R0 dan R1 dengan jangkauan hanya 256 byte.

II.6.6. PEWAKTU/PENCACAH

Mikrokontroler 8031 mempunyai dua buah register pewaktu/pencacah 16 bit. yaitu Pewaktu/Pencacah 0 terdiri dari TH0 dan TL0 dan yang lainnya

---

[15)]Ibid,. hal 10-22

Pewaktu/Pencacah 1 terdiri dari TH1 dan TL1.

Dalam fungsinya sebagai "Pewaktu", isi regsiter akan bertambah satu setiap siklus mesin. Ini berarti sebagai penghitung siklus mesin. Setiap siklus mesin terdiri dari 12 periode osilator, sehingga kecepatan penghitungan adalah 1/12 dari frekwensi osilator.

Tabel 2.2.[16] Register Kontrol Mode Pewaktu/Pencacah (TMOD)

BIT	KETERANGAN
GATE	Jika Gate =1, Pewaktu/Pencacah 1 akan aktif jika pint INT1 = 1. Jika Gate = 0, Pewaktu/Pencacah 1 akan aktif jika TRx = 1
C/T	Pemilih Pewaktu atau Pencacah, jika C/T = 1 akan berfungsi sebagai pencacah dan jika C/T = 0 akan berfungsi sebagai pewaktu
M1	Pemilih mode sesuai nilai binernya
M0	Pemilih mode sesuai nilai binernya
GATE	Jika Gate =1, Pewaktu/Pencacah 1 akan aktif jika pint INT1 = 1. Jika Gate = 0, Pewaktu/Pencacah 1 akan aktif jika TRx = 1
C/T	Pemilih Pewaktu atau Pencacah, jika C/T = 1 akan berfungsi sebagai pencacah dan jika C/T = 0 akan berfungsi sebagai pewaktu
M1	Pemilih mode sesuai nilai binernya
M0	Pemilih mode sesuai nilai binernya

Dalam fungsinya sebagai "Pencacah", isi register akan bertambah satu setiap mendapatkan sebuah perubahan transisi taraf tegangan tinggi (1) ke taraf tegangan rendah (0) pada pin input T0 dan T1. Untuk mengetahui

---

[16]Ibid, hal 6-7

perubahan dari transisi 1 ke 0 diperlukan dua siklus mesin, sehingga maksimum kecepatan penghitungan adalah 1/24 dari frekwensi osilator.

Pewaktu atau Pencacah mempunyai empat macam mode operasi. pemilihan fungsi sebagai Pewaktu atau sebagai Pencacah ditentukan dengan bit kontrol C/T yang terdapat pada register fungsi khusus TMOD. pada gambar diperlihatkan Register Kontrol Mode Pewaktu/Pencacah (TMOD)

II.6.6.1. MODE 0.

Pewaktu/Pencacah digunakan sebagai penghitung 8 bit dengan 32 prescaler. Gambar memperlihatkan Pewaktu/Pencacah 1 pada Mode 0. Pada mode ini register Pewaktu/Pencacah merupakan register 13 bit dengan TL1 sebesar 5 bit dan TH1 sebesar 8 bit. Jika pemutaran perhitungan melebihi dari keseluruhan 1 menjadi keseluruhan 0, akan mengaktifkan flag interup TF1. Perhitungan input dapat dilakukan jika TR1 = 1 dan Gate = 0 atau INT1 = 1. TR1 adalah bit kontrol yang terdapat pada register fungsi khusus TCON.

II.6.6.2. MODE 1.

Penggunaan Pewaktu/Pencacah pada mode 1 serupa dengan mode 0 seperti yang telah dijelaskan di atas. Perbedaannya adalah pada mode ini register Pewaktu/Pencacah dijalankan dengan 16 bit.

II.6.6.3. MODE 2.

Pada mode ini register Pewaktu/Pencacah 1 berfungsi sebagai

penghitung 8 bit (TL1) dengan pengisian kembali otomatis. Overflow dari TL1 tidak hanya mengaktifkan TF1 tetapi juga melakukan pengisian kembali TL1 dengan isi yang ada pada TH1, yang dilakukan secara perangkat lunak. Pengisian kembali ini tidak akan merubah isi TH. Penghitungan input dapat dilakukan jika TR1 = 1 dan Gate = 0 atau INT1 = 1. TR1 adalah bit kontrol yang terdapat pada register fungsi khusus TCON.

II.6.6.4. MODE 3.

Pada mode ini Pewaktu/Pencacah 1 tidak aktif dan sedangkan isi registernya tetap. Keadaan serupa dengan jika TR1 berlogika 0. Pewaktu/Pencacah 0 berbentuk dengan register TL0 dan TH0 sebagai dua penghitung yang terpisah. Gambar memperlihatkan penggunaan pewaktu/pencacah 0 pada mode 3. TL0 menggunakan kontrol bit dari Pewaktu/Pencacah 0 : C/T, GATE, TR0, INT0 dan TF0. Sedangkan TH0 hanya tidak digunakan untuk pewaktu (penghitung siklus mesin) dan menggunakan TR1 dan TF1 dari Pewaktu/Pecacah 1. Sehingga sekarang TH0 adalah Pengontrol Pewaktu 1.

Mode 3 ini digunakan untuk aplikasi yang membutuhkan tambahan pewaktu atau pencacah 8 bit. Pewaktu/Pecacah 1 tetap dapat digunakan oleh serial port sebagai pembangkit baud rate atau penggunaan lainnya yang tidak memerlukan interupsi.

Bentuk dan susunan bit-bit pada register fungsi khusus Kontrol Pewaktu/Pencacah (TCON) seperti pada tabel 2.3.

Tabel 2.3.[17] Register Kontrol Pewaktu/Pencacah (TCON)

BIT	KETERANGAN
TF1 (TCON.7)	Saat Pewaktu/Pencacah 1 over flow, maka TF1 = 1, Saat vektor CPU melakukan interupsi, maka TF1 = 0
TR1 (TCON.6)	Jika TR1 = 1, maka Pewaktu/Pecacah akan aktif. Jika TR1 = 0, Pewaktu/Pencacah akan non-aktif
TF0 (TCON.5)	Saat Pewaktu/Pencacah 1 over flow, maka TF0 = 1, Saat vektor CPU melakukan interupsi, maka TF0 = 0
TR0 (TCON.4)	Jika TR0 = 1, maka Pewaktu/Pecacah akan aktif. Jika TR0 = 0, Pewaktu/Pencacah akan non-aktif
IE1 (TCON.3)	Saat interupsi eksternal diterima, maka IE1 = 1. Saat interupsi dilakukan (diproses), maka IE1 = 0
IT1 (TCON.2)	Jika IT1 = 1, maka interupsi eksternal akan terdeteksi saat sinyal pada
IE0 (TCON.1)	Saat interupsi eksternal diterima, maka IE0 = 1. Saat interupsi dilakukan (diproses), maka IE0 = 0
IT0 (TCON.0)	Jika IT0 = 1, maka interupsi eksternal akan terdeteksi saat sinyal pada

II.6.7. INTERUPSI.

Mikrokontroler 8031 mempunyai lima buah sumber interupsi yang dapat membangkitkan permintaan interupsi (interupsi request), yaitu :

- INT0 : Permintaan interupsi eksternal dari pin P3.2
- INT1 : Permintaan interupsi eksternal dari pin P3.3
- Pewaktu/Pencacah 0

---

[17]Ibid, hal 6-8

- Pewaktu/Pencacah 1
- Port Serial : Pengiriman atau penerimaan data telah lengkap

Setelah terjadi suatu interupsi maka CPU akan mengaktifkan LCALL secara perangkat keras. LCALL yang dilakukan akan mengisi program counter PC dengan alamat vektro interupsi sesuai dengan sumber datangnya interupsi. Pada alamat vektor interupsi tersebut berisikan program subroutine yang harus dilakukan jika interupsi tersebut terjadi. Setelah program subroutine tersebut selesai dikerjakan, CPU akan melanjutkan urutan instruksi berikutnya dari program yang semula dikerjakan. Pada tabel alamat vektro interupsi sesuai dengan sumber interupsinya.

Tabel 2.4.[18] Alamat Vektor Interupsi.

SUMBER INTERUPSI		ALAMAT VEKTOR
Interupsi Eksternal 0	(IE0)	0003H
Pewaktu/Pencacah 0	(TF0)	000BH
Interupsi Eksternal 1	(IE1)	0013H
Pewaktu/Pencacah	(TF1)	001BH
Port Serial		0023H

Seluruh sumber interupsi dapat diaktifkan atau non-aktifkan dengan mengatur bit-bit tertentu pada register fungsi khusus Pemungkin Interupsi (IE). bentuk dan susunan bit-bit pada register IE tersebut dapat dilihat pada tabel 2.5.

---

[18]Ibid, hal 9-11

MILIK PERPUSTAKAAN
INSTITUT TEKNOLOGI
SEPULUH – NOPEMBER

Tabel 2.5.[19] Register Pemungkin Interupsi (IE)

BIT	KETERANGAN
EA (IE.7)	Jika EA = 0, maka semua interupsi akan dinonaktifkan, status dari IE.0 - IE.4
X	
X	
ES (IE.4)	Untuk interupsi dari Port Serial
ET1 (IE.3)	Untuk interupsi dari Pewaktu/Pencacah 1
EX1 (IE.2)	Untuk interupsi Eksternal 1
ET0 (IE.1)	Untuk interupsi dari Pewaktu/Pencacah 0
EX0 (IE.0)	Untuk interupsi Kesternal 0

Pada 8031 tersedia dua tingkat prioritas interupsi yang dapat digunakan oleh semua sumber interupsi. Pemilihan dari prioritas ini dengan cara me-reset atau set bit-bit pada register fungsi khusus prioritas Interupsi (IP). Interupsi dengan tingkat yang rendah dapat mengalami interupsi dari yang tingkatnya lebih tinggi. tetapi interupsi dengan level yang tinggi tidak dapat mengalami interupsi dari yang tingkatnya lebih rendah atau sama. Jika dua atau lebih interupsi yang tingkatnya sama terjadi bersamaan, maka prioritasnya berturut-turut dari tinggi ke rendah sesuai dengan interupsi standar 8031 adalah :

- Interupsi Eksternal 0
- Interupsi Timer 0

---

[19]Ibid, hal 9-11

- Interupsi Eksternal 1
- Interupsi Timer 1
- Interupsi Serial

Bentuk dan susunan bit-bit pada Register IP tersebut dapat dilihat pada tabel .2.6.

Tabel 2.6.[20] Register Prioritas Interupsi (IP)

BIT	KETERANGAN
X	
X	
X	
PS (IP.4)	Untuk pengaturan prioritas interupsi Port Serial
PT1 (IP.3)	Untuk pengaturan interupsi dari Pewaktu/Pencacah 1
PX1 (IP.2)	Untuk pengaturan interupsi dari interupsi Eksternal 1
PT0 (IP.1)	Untuk pengaturan interupsi Pewaktu/Pencacah 0
PX0 (IP.0)	Untuk pengaturan interupsi dari interupsi Eksternal 0

II.6.8. PORT SERIAL

Port serial pada 8031 dapat mengirim dan menerima data secara bersamaan (full duplex). Pada port ini juga terdapat penyangga (buffer), sehingga penerimaan dapat dilakukan ketika terjadi proses pembacaan dari register penerima. Register pengirim dan penerima keduanya diakses dari register fungsi khusus Penyangga Data Serial (SBUF). Penulisan ke register SBUF akan mengisi register pengiriman dan membaca SBUF akan mengakses

---

[20]Ibid, hal 9-12

register penerima yang berbeda.

Tabel 2.7.[21] Register Kontrol Serial (SCON)

BIT	KETERANGAN
SM0 (SCON.7)	Pemilih mode serial port sesuai nilai binernya
SM1 (SCON.6)	Pemilih mode serial port sesuai nilai binernya
SM2 (SCON.5)	Keaktifan dari komunikasi multiprosesor pada mode 2 dan 3
REN (SCON.4)	Keaktifan penerima yang diset software
TB8 (SCON.3)	Data ke 9 yang dikirim pada mode 2 dan 3
RB8 (SCON.2)	Data ke 9 yang dikirim pada mode 2 dan 3, stop bit pada mode 1 dan tidak digunakan pada mode 0
TI (SCON.1)	Flag interupsi pada pengiriman
RI (SCON.0)	Flag interupsi pada penerimaan

Register yang berisi status dan kontrol dari port serial adalah register fungsi khusus Kontrol Serial (SCON). Bentuk dan susunan register ini dapat dilihat pada tabel 2.7. Selain bit status dan kontrol, juga mengandung bit ke 9 pada pengiriman atau penerimaan data secara serial.

Port serial 8031 dapat digunakan pada empat macam mode. Isi dari bit SM0 dan SM1 pada register SCON menentukan mode yang digunakan seperti yang akan dijelaskan di bawah sebagai berikut.

II.6.8.1. MODE 0

Pada mode ini, data serial akan masuk dan keluar melalui RxD. TxD mengeluarkan clock pergeseran 8 bit data dikirim dan diterima. Bit yang

---

[21]Ibid, hal 9-18

pertama yang dikirim atau diterima adalah LSB. baud rate mode 0 ini adalah 1/12 dari frekwensi osilator.

II.6.8.2. MODE 1

Penerimaan data pada RxD dan pengiriman data pada TxD. Data yang dikirim atau diterima adalah data 10 bit, yaitu bit start yang berlogika 0, 8 bit data dan bit stop yang berlogika 1. pada penerimaan bit stop akan menempati RB8 pada register SCON. Baud rate pada mode 1 ini dapat diatur (variable).

II.6.8.3. MODE 2

Data yang diterima atau dikirim adalah data 11 bit, yaitu bit start, 8 bit data, bit ke 10 yang dapat diprogram dan bit stop. Pada pengiriman bit ke 10 ditempatkan pada TB8 dari register SCON. Baud rate pada mode 2 ini dapat dipilih i/32 atau 1/64 dari frekwensi osilator.

II.6.8.4. MODE 3

Mode 3 ini sama dengan pada mode 2 seperti yang telah dijelaskan diatas, kecuali pada baud rate mode 3 ini dapat diatur (variable).

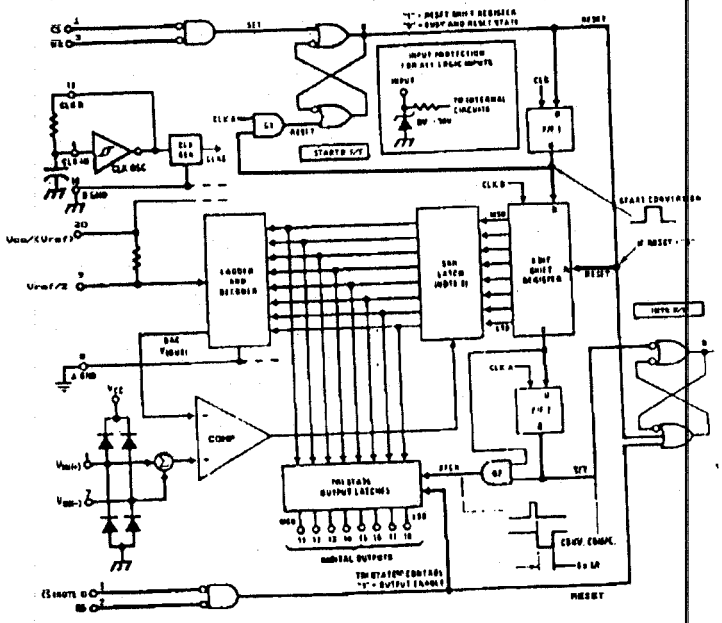
II.7. PENGUBAH ANALOG KE DIGITAL (ADC)

Rangkaian analog to digital converter diperlukan untuk mengubah tegangan analog menjadi logika digital dalam bentuk bit-bit agar dapat diproses secara digital baik oleh mikrokontroler maupun oleh komputer.

Dalam menggunakan ADC harus diperhatikan beberapa hal, antara lain : kecepatan konversi (conversion time) keakuratan (accuracy), stabilitas(stability), maupun faktor biaya (cost).

Secara umum ADC dapat dibedakan menjadi dua golongan, yaitu golongan open-loop, misalnya : tipe *flash*-ADC, *Slope converter, dual slope converter, dan close loop,* misalnya: *single counter ADC, successive approximation ADC.*

Dalam pembahasan ini akan dijelaskan mengenai IC ADC 0804 yang merupakan jenis *successive approximation* karena dalam perencanaan peralatan menggunakan tipe ADC tipe ini.



Gambar 2.11.[22)]
Blok Diagram ADC 0804

[22)]...., linear Databook 2, National Semiconductor Corporation, hal 3-29

II.7.1. SUCCESSIVE APROXIMATION ADC.

ADC jenis ini didasarkan penggunaan DAC (*Digitas to Analog COnverter*) dengan sistem logika yang mengemudikan DAC sampai output yang diinginkan sesuai.

Pada dasarnya rangkaian sistem ini terdiri dari tiga bagian pokok seperti ditunjukkan dalam gambar 2.16. Ketiga bagian pokok tersebut, yaitu :

1. Successive Approximation Register (SAR).
2. Digital to Analog Converter (DAC).
3. Comparator

Proses konversi diawali dengan memberikan sinyal WR pada SAR yang kemudian SAR akan terkunci dan *shift register stage* di *reset*. Konverter dimulai bila pin CS dan WR low, register yang diisi oleh perangkat keras dengan byte data pada saat bersamaan dengan aktifnya RI. Kondisi ini akan men-*set start filip-flop (F/F)* dan level '1' yang dihasilkan akan me-*reset shift register* 8 bit dan me-*reset* INTR F/F sekaligus memberikan logika '1' pada D flop F/F1.

Konversi akan berakhir dengan ditandai aktifnya pin INTR dan kemudian data siap dibaca dengan mengaktifkan pin CS dan RD secara bersamaan yang menyebabkan INTR F/F di reset TRI-STATE *output latches* akan di-*enable*-kan agar dapat mengeluarkan data 8 bit.

II.8. OPERASIONAL AMPLIFIER.

Dalam pemakaian umum Operational Amplifier sering dingkat demgam menggunakan Op-Amp. Secara umum karakteristik Op-Amp yang terpenting

adalah :

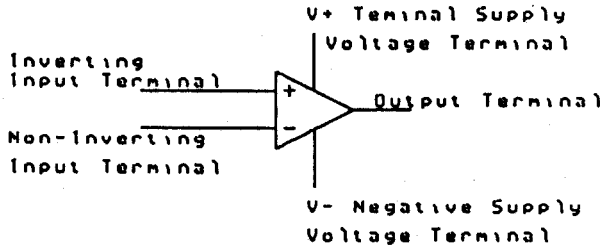
a. Inpedansi masukan amat tinggi, sehingga arus masukan praktis dapat diabaikan.

b. Penguatan loop terbuka sangat tinggi.

c. Impedansi keluaran amat rendah, sehingga keluaran penguat tidak terpengaruh oleh pembebanan.

Dalam penerapannya pada rangkaian-rangkaian elektronika, Op-Amp digambarkan dengan simbol khusus seperti terlihat pada gambar 2.12.

Teminal input *inverting* diberi tanda (-). Hal ini berarti tegangan output akan berbeda 180^0 terhadap tegangan input *inverting* . Sedangkan input *non-inverting* diberi tanda (+), dimana output yang dihasilkan akan sephasa dengan tegangan inputnya.



Gambar 2.12.[23)]
Simbol Skematik Op-Amp

Tegangan terminal catu daya dan terminal lain untuk kompensasi frekuensi atau penyetelan *null* digambarkan pada kedua sisi miring segitiga.

[23)]Jung, Walter G., IC Op-Amp Cookbook, Howard W.Sams & Co. Inc., Indiana, 1985, hal 26

Terminal-terminal yang disebutkan terakhir tersebut diatas, tidak selalu diperlihatkan dalam simbol.

Dalam praktek, umumnya Op-Amp selalau memerlukan komponen luar baik komponen pasif maupun aktif. Dan dalam prakteknya Op.Amp banyak digunakan sebagai :

a. Differensial Amplifier

b. Integrator

c. Filter

d. Rectifier

e. Limiter / Clipper

f. dan lainnya

Pada dasarnya penggunaan Op-Amp dalam penerapannya seperti diatas selalu didasarkan pada salah satu dari dua konfigurasi dasar atau perpaduan dari dua konfigurasi dasar Op-Amp seperti dibawah ini, yaitu :

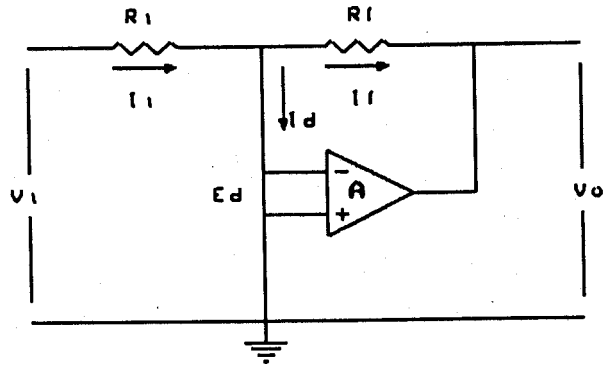
a. Inverting Amplifier

b. Non-Inverting Amplifier

II.8.1. INVERTING AMPLIFIER.

Hubungan dasar dari rangkaian dengan konfigurasi inverting amplifier diperlihatkan pada gambar 2.13. Beberapa macam *feedback* umumnya sering digunakan dalam penerapan rangkaian yang menggunakan Op-Amp. *Feedback* negatif membuat kemudahan dalam merencanakan besarnya penguatan yang dikehendaki.

Resistor R_f menyebabkan sebagian dari sinyal output V_o yang berlawanan phasa dengan input inverting diumpankan kembali ke terminal input inverting. Dengan tipe *feedback* ini penguatan tegangan A_{cl} efektif dari rangkaian akan lebih rendah daripada penguatan open loop A_{ol}.



Gambar 2.13.
Inverting Amplifier

Penguatan dari rangkaian ini ditentukan oleh perbandingan R_i dan R_f yang dinyatakan oleh persamaan :

$E_d = 0$;

$$I = \frac{V_i}{R_i}$$

$$V_o = -I \,.\, R = -\frac{R_f \,.\, V_i}{R_i}$$

$$A_{cl} = \frac{V_o}{V_i} = -\frac{R_f}{R_i}$$

Tanda minus (-) menunjukkan bahwa antara input inverting dan outputnya berbeda phasa 180°.

II.8.2. NON-INVERTING AMPLIFIER.

Gambar 2.14. menunjukkan rangkaian Op-Amp dengan konfigurasi non-inverting amplifier. Dari gambar tersebut terlihat bahwa sinyal output diumpankan ke input inverting oleh R_f yang membentuk konfigurasi feedback negatif, maka pada R_f akan mengalir arus I yang besarnya sama dengan yang mengalir pada R_i. Dimana besarnya arus I yang mengalir adalah :

$V_i = V;$

$$V_o = V_i + I.R_f$$

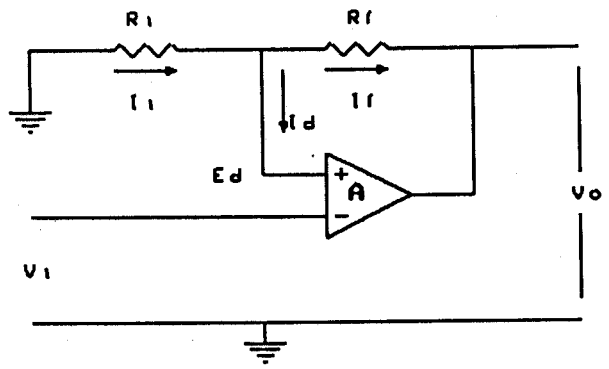
$$V_o = V_i + \frac{R_f \cdot V_i}{R_i}$$

Maka penguatan tegangan A_{cl} untuk konfigurasi non-inverting adalah sebesar :

$$A_{cl} = \frac{V_o}{V_i} = \frac{V_i}{V_i} + \frac{R_f \cdot V_i}{V_i \cdot R_i}$$

$$A_{cl} = 1 + \frac{R_f}{R_i}$$

Terlihat bahwa dengan menggunakan konfigurasi non-inverting amplifier, output yang dihasilkan akan sephasa dengan input non-inverting.



Gambar 2.14
Non-Inverting Amplifier

BAB III

PERENCANAAN

Perencanaan pembuatan tugas akhir ini terbagi dalam dua bagian utama, yaitu perencanaan perangkat keras dan perencanaan perangkat lunak. Perencanaan perangkat keras dimulai dari blok diagram rangkaian dan prinsip kerja secara umum. Sedangkan perencanaan perangkat lunak dibagi dalam dua bagian lagi, yaitu prencanaan perangkat lunak untuk menjalankan mikrokontroller dan perangkat lunak pada komputer. Perencanaannya meliputi diagram alir (flow chart) dan uraian cara kerjanya.

III.1. PERENCANAN PERANGKAT KERAS

Dalam merencanakan perangkat keras pada tugas akhir ini, diusahakan menggunakan komponen-komponen yang mudah didapat.

Komponen utama perangkat keras adalah mikrokontroller 8031 yang berfungsi mengontrol proses penerima data dari ADC, proses penyimpanan data, mengolahnya dan menampilkan pada tampilan, serta mengirimkan secara serial pada komputer.

III.1.1. BLOK DIAGRAM

Dalam pengoperasiannya, mikrokontroller dirangkai dengan komponen-komponen penting lainnya. Bagian-bagian dari perangkat keras yang dirancang

terdiri dari beberapa unit, antara lain :

- unit pemproses utama
- unit ocillator
- unit sensor, penguat dan filter
- unit Analog to Digital Converter
- unit memori
- unit tampilan
- unit keyboard
- unit dekoder
- unit serial

Unit pemproses utama merupakan bagian utama dari sistem yang dirancang, terdiri dari mikrokontroller 8031, IC 74LS373 sebagai *latch*, IC 74LS245 sebagai buffers dan kristal.

Unit oscillator berfungsi untuk menyalakan LED sebagai sumber sinar serta pembangkit sinyal baca dan tulis pada unit ADC.

Bagian sensor berfungsi mengubah energi cahaya menjadi energi listrik, yang kemudian dikuatkan oleh amplifier.

Unit Analog to Digital Convertion berfungsi mengubah sinyal analog dari bagian penguat untuk diubah menjadi sinyal digital, agar bisa dibaca mikrokontroller.

Rangkaian memori terdiri memori program, yaitu EPROM dan memori data yaitu RAM. Fungsi EPROM adalah untuk menyimpan perangkat lunak mikrokonroller, sedangkan RAM digunakan untuk menyimpan data hasil

pembacaan transduser serta menyimpan hasil olahan mikrokontroller.

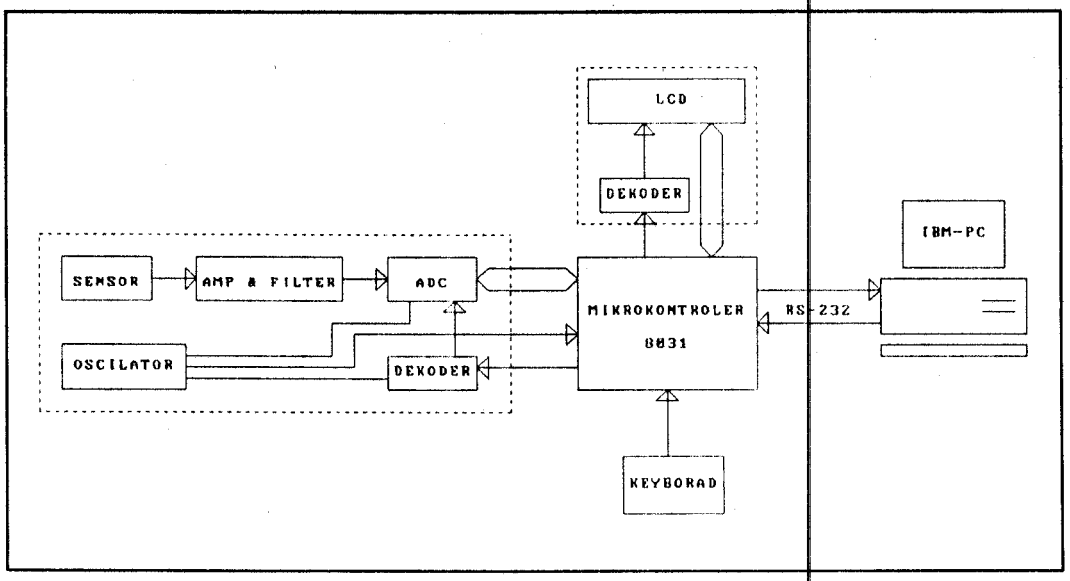
Unit tampilan berfungsi untuk menampilkan hasil pengukuran transduser yang dikonversi ke digital oleh ADC sehingga bisa dibaca.

Keyboard berfungsi untuk memberikan perintah atau mengoperasikan sistem yang dirancang.

Untuk menghubungkan bagian-bagian tersebut agar tidak terjadi kekacauan dalam mengakses data diperlukan unit dekoder, yang berfungsi mengatur alamat dari masing-masing unit.

Unit serial berfungsi untuk menangani pengiriman data secara serial dari/ke komputer.

Hubungan dari masing-masing unit digambarkan dalam bentuk blok diagram, seperti gambar 3.1.

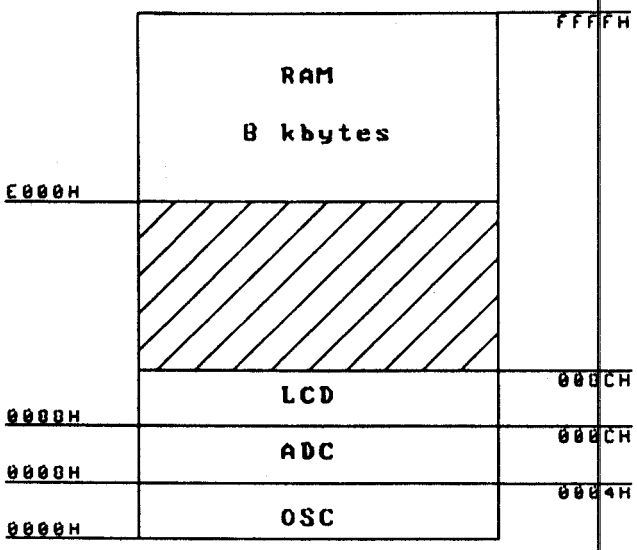


Gambar 3.1.
Blok Diagram Rangkaian

III.1.2. PETA MEMORI

Agar tidak terjadi kekacauan dalam pembacaan/penulisan data pada unit-unit yang terhubung ke sistem mikrokontroller, harus direncanakan terlebih dahulu pengalamatan dari masing-masing unit tersebut dalam peta memori.

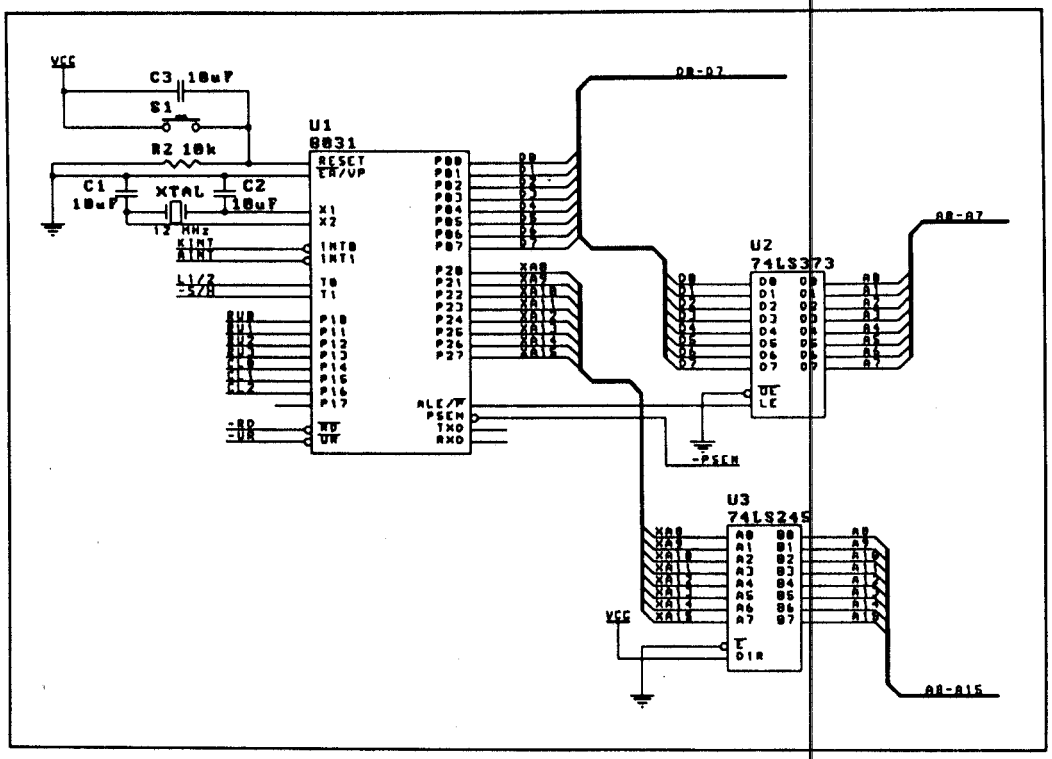
Pada sistem yang dirancang, direncanakan sistem pengalamat unit-unit sebagai berikut :



Gambar 3.2.
Peta memori

III.1.3. RANGKAIAN MIKROKONTROLLER

Mikrokontroller yang digunakan adalah 8031 yang merupakan keluarga MCS-51. Rangkaian yang dirancang ditunjukkan pada gambar 3.3.



Gambar 3.3.
Rangkaian Mikrokontroler 8031

Port 0 mikrokontroler dihubungkan dengan IC 74LS373 yang berfungsi sebagai *latch* untuk data address, dan port 2 dihubungkan dengan 74LS245 sebagai *buffers* data address.

Pin ALE dihubungkan dengan pini LE IC 74LS373, sehingga saat sinyal ALE aktif (aktif high) sinyal address akan diteruskan ke IC latch. Dan saat ALE kembali rendah yang berakhirnya proses pengiriman sinyal address IC 74LS373 akan didisable. Hal ini menandakan bahwa port 0 berisi sinyal data dari/ke memori atau unit lain.

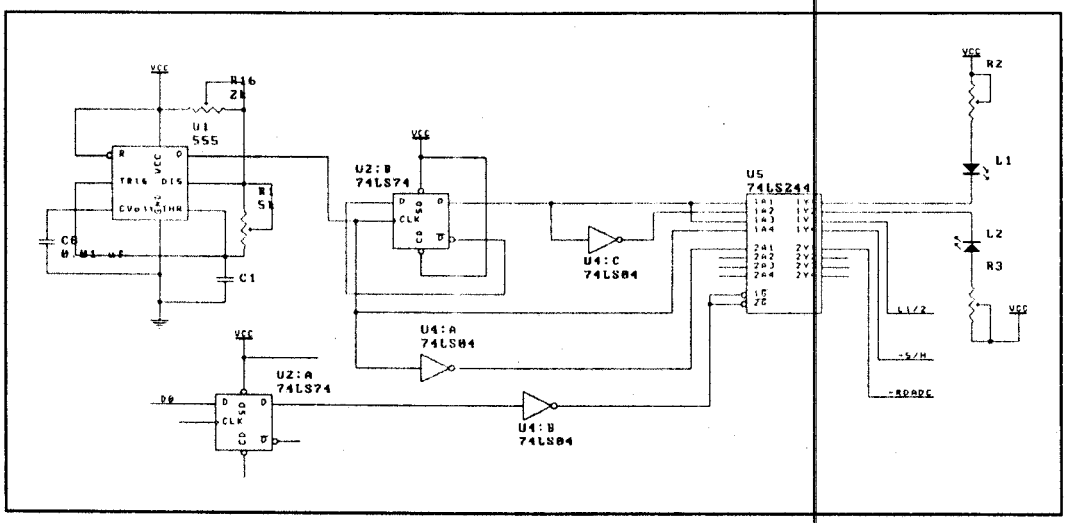
Pin PSEN dihubungkan dengan ke pin CS EPROM. Sebab sinyal

PSEN rendah berarti mikrokontroler sedang melakukan pembacaan memori program luar. Serta pin RD dan WR akan high.

Mikrokontroler juga dihubungkan dengan kristal 11.059 MHz pada pin X1 dan X2. Osilator untuk kristal tersebut sudah terdapat pada 8031 (on-chip oscilator). Konfigurasi krital ini dapat juga dilakukan secara eksternal osilator, yaitu lewat pin X1 dan X2 dihubungkan dengan ground.

III.1.4. RANGKAIAN OSILATOR

Osilator digunakan untuk menyalakan LED secara bergantian. Selain itu juga menghasilkan sinyal *strobe* untuk ADC, yaitu *strobe* untuk memulai konversi (-S/H) dan strobe untuk pembacaan ADC(-RDADC)



Gambar 3.4.
Rangkaian Osilator

Komponen yang digunakan untuk osilator adalah IC 555 yang dirangkai sebagai *astable multivibrator* dengan *duty cylce* 50 % dengan frekuensi 4 KHz,

dan kemudian dibagi 2 dengan menggunakan D FF menjadi 2 KHz untuk menyalakan LED. Sedangkan yang 4 KHz digunakan untuk memberi *strobe* pada ADC untuk memulai sampling.

IC *buffer* 74LS244 digunakan sebagai gerbang untuk mengontrol *on/off* sensor. Untuk menyalakan LED diberikan data 1 (D0=1) pada alamat memori osilator, sehingga *buffer* transparan dan osilator akan menyalakan LED. Dan untuk mematikan kembali, diberikan data 0 (D0=0) pada alamat memori osilator, sehingga *buffer* akan *high impedance*. D FF digunakan sebagai *latch* yang akan mempertahan keadaan *buffer* transparan atau *high impedance*.

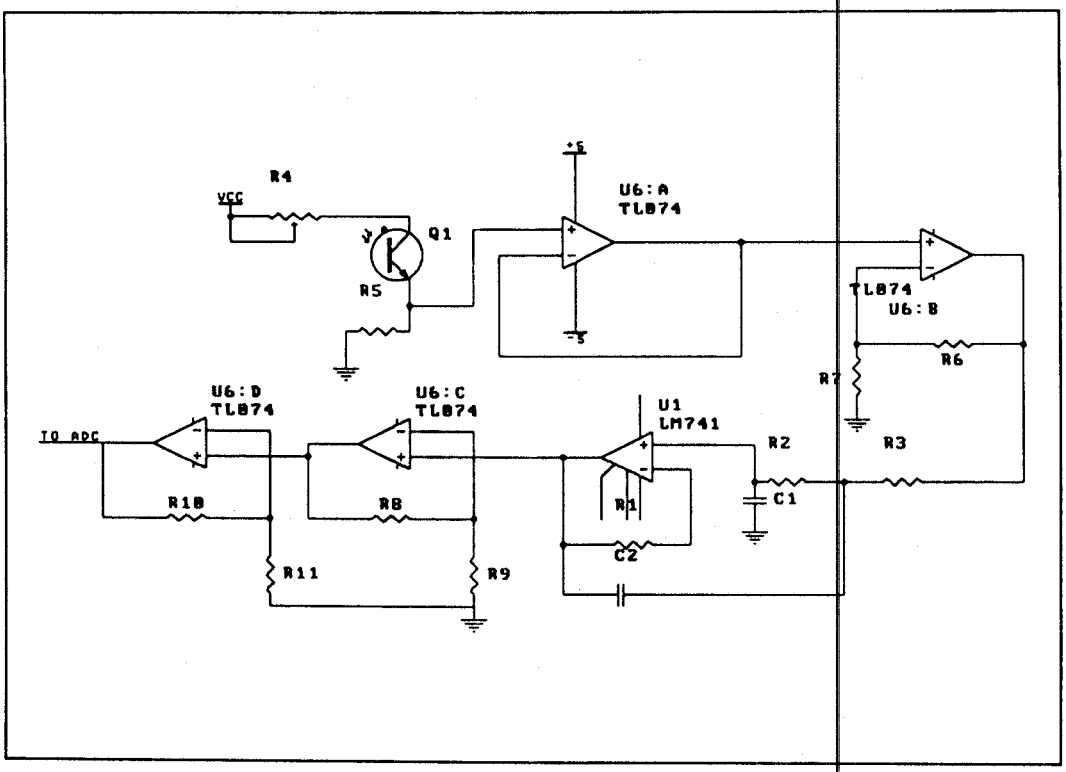
III.1.5. RANGKAIAN SENSOR, PENGUAT DAN FILTER

Sensor penerima cahaya menggunakan fototransistor MRD 300 dari motorola yang memberikan respon relatif 100% pada panjang gelombang 800 nm dan 60 % pada 650 nm. resistor variable pada kolektro digunakan agar transistor tidak mengalami saturasi. Dan nilai resistor diatur agar pada intensitas cahaya yang paling besar transistor tidak saturasi.

Karena respon penerimaan cahaya yang berlainan, maka intensitas dari kedua LED diatur sehingga pada pantulan yang sempurna, keluaran transistor menunjukkan level yang sama.

Selanjutnya hasil keluaran dari fototransistor dikuatkan dengan penguat operasional (op-amp) TL 074. Penguatan dibutuhkan agar level tegangan yang dihasilkan oleh fototransistor bisa diproses oleh ADC. Pada tugas akhir ini

dilakukan penguatan sebesar 500 kali. Dengan penguatan masing-masing op-amp 10 kali dan 5 kali.



Gambar 3.5.
Rangkaian Sensor, Penguat dan Filter

Filter dibutuhkan untuk menghilangkan derau dari frekuensi luar sistem serta efek *ringing* akibat *switching* tegangan dari masing-masing LED yang bisa mengganggu konversi sinyal analog ke sinyal digital.

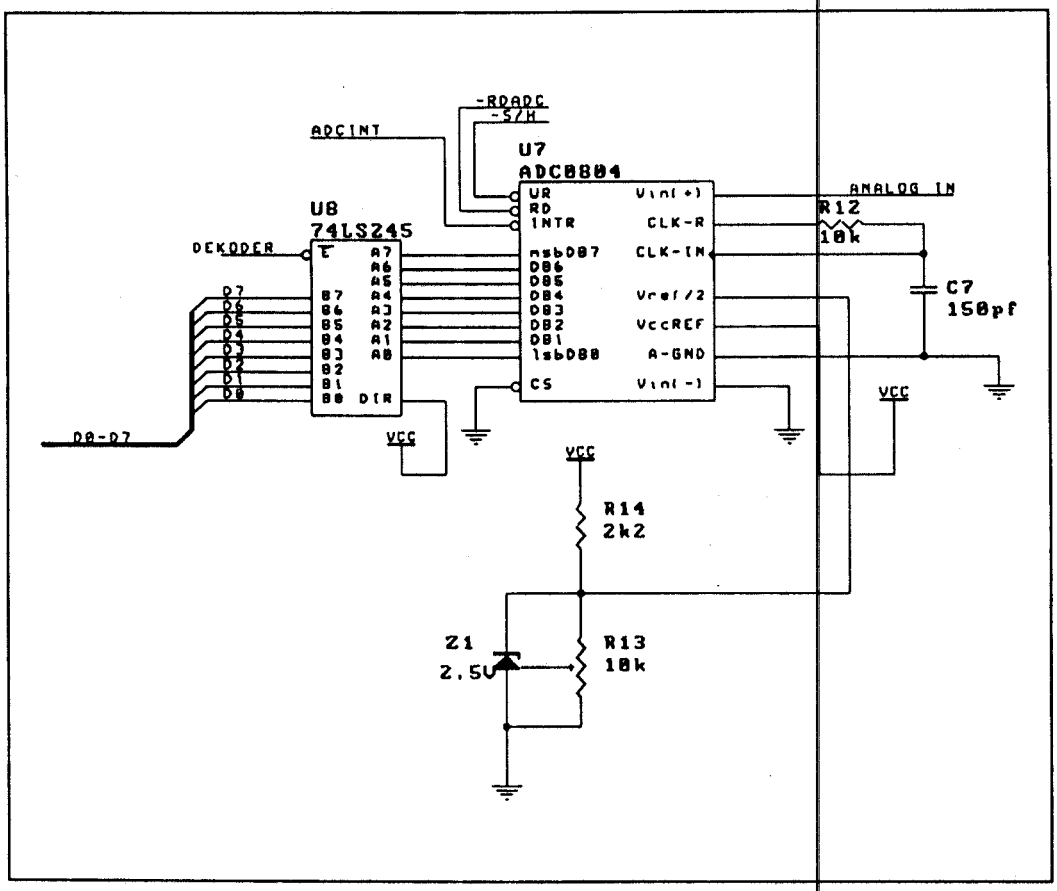
Penguat dan filter terdapat dalam satu rangkaian yang sederhana, seperti pada gambar 3.5.

III.1.6. RANGKAIAN KONVERSI SINYAL ANALOG KE DIGITAL

Untuk konversi sinyal analog menjadi sinyal digital diperlukan unit konverter, dan dalam tugas akhir ini digunakan IC dari jenis *semi flash* ADC,

yaitu ADC0804. ADC ini mempunyai input 1 chanel dan 8 bit output, sehingga resolusinya adalah 1/256.

IC ADC ini mempunyai *acess time* 130 ns dan difungsikan dengan *voltage reference* 2.5 volt untuk menghasilkan *reange* input sebesar 5 volt (0-5 volt). IC ini juga tidak memerlukan *zero adjust* dengan waktu konversi 100 us. Rangkaian ADC yang direncanakan ditunjukkan pada gambar 3.6.



Gambar 3.6.
Rangkaian Analog to Digital Convertion

Untuk memulai ADC bekerja diberikan sinyal *start sampling* (-S/H) dari osilator. Setelah konversi selesai ADC akan menginterupsi mikrokontroler, dan selanjutnya mikrokontroler mengambil data dari ADC.

Untuk menjaga agar tidak terjadi tabrakan sinyal pada bus data, antara bus data ADC dengan bus data sistem mikrokontroler diberi komponen *buffer* 74LS245. Sehingga saat mkrokontroler membaca meori data eksternal 0008H, buffer akan transparan dan data ADC bisa dibaca mikrokontroler. Selain keadaan itu *buffer* akan berada dalam keadaan *high impedance* dan bus data ADC akan terisolasi dengan bus data mikrokontroler.

III.1.7. RANGKAIAN MEMORI

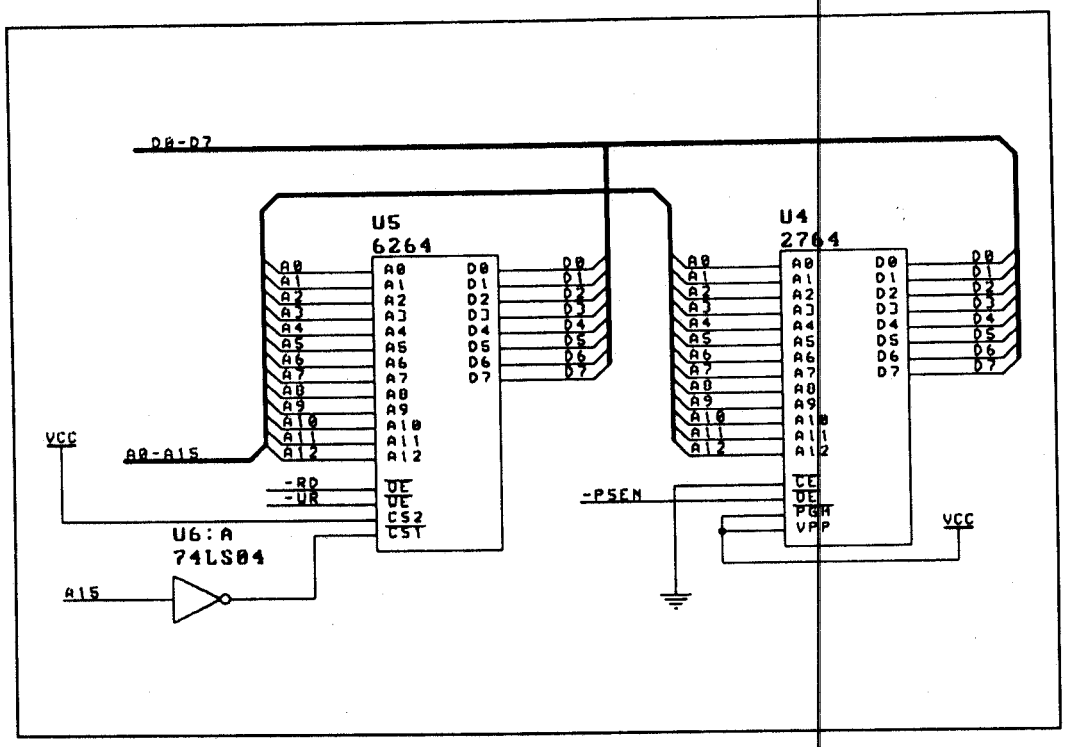
Memori dibutuhkan untuk menyimpan program (EPROM) dan untuk menyimpan data pengukuran dan data hasil yang diproses dari hasil pengukuran (RAM).

Untuk menyimpan program digunakan IC EPROM 2784 dengan kapasitas penyimpanan memori sebesar 8 kbyte. Dan untuk memori data menggunakan IC RAM statis 6264 yang mampu menampung data sebesar 4 kbyte.

Pin CE dari EPROM dihubungkan dengan *ground* sedangkan kaki OE/VPP dihubungkan dengan pin PSEN dari mikrokontroler . Sehingga jika sinyal PSEN aktif *(aktif low)* berarti mikrokontroler melakukan pembacaan instruksi dari program yang disimpan pada EPROM.

Pin OE dari RAM dihubungkan dengan pid RD mikrokontroler dan pin WE dihungungkan dengan pin WR mikrokontroler. Pin ini digunakan untuk memberikan sinyal baca/tulis data pada eksternal memori. Pin CS2 dihubungkan dengan Vcc, sedangkan pin Cs1 dihubungkan dengan gerbang

NOT dan selanjutnya dihubungkan dengan pin A15 mikrokontroler. Sehingga jika terjadi pembacaan/penulisan memori eksternal yang mengaktifkan A15 berarti pembacaan atau penulisan pada RAM.



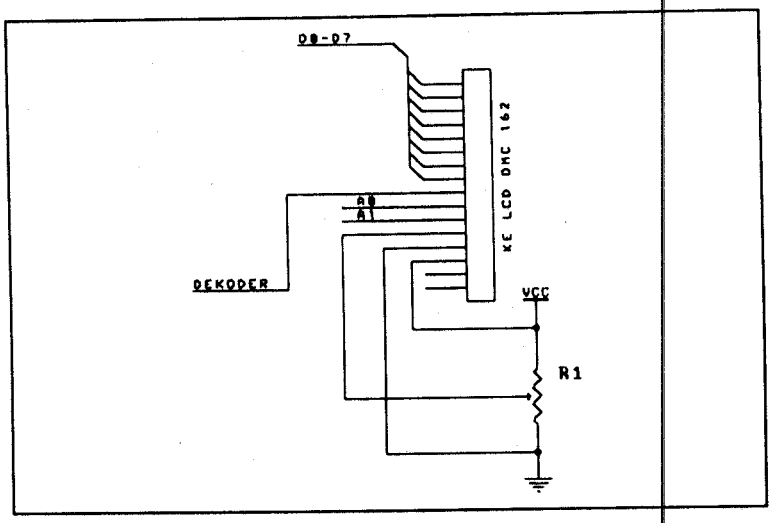
Gambar 3.7.
Rangkaian Memory

III.1.8. RANGKAIAN TAMPILAN

Rangkaian tampilan menggunakan *Liquid Crytal Display* (LCD) tipe DMC162 buatan Seiko Instruments Inc. Keuntungan menggunakan LCD ini antara lain catu daya

yang dibutuhkan kecil, sehingga cocok untuk sistem yang portable, dapat menampilkan karakter-karakter yang cukup bervariasi tidak hanya alphanumerik saja dan mudah dalam pembuatan rangkaiannya serta

pemogramaannya.



Gambar 3.8.
Rangkaian Display

Karakteristik dari LCD DMC162 adalah sebagai berikut :

- mempunyai 2 register 8 bit, yaitu register intruksi (IR) dan register data (DR)
- 2 baris x 16 karakter
- memori lokal yaitu : RAM 80x8 bit dan ROM sebagai generator karakter

III.1.9. RANGKAIAN PAPAN KUNCI

Rangkaian papan kunci (*keyboard)* dibuat dengan saklar mekanis yang dihubungkan secara matriks dalam bentuk baris dan kolom 4 x 3.

Dalam menangani papan kunci hal pokok yang perlu diperhatikan adalah :

- mendeteksi kunci yang ditekan

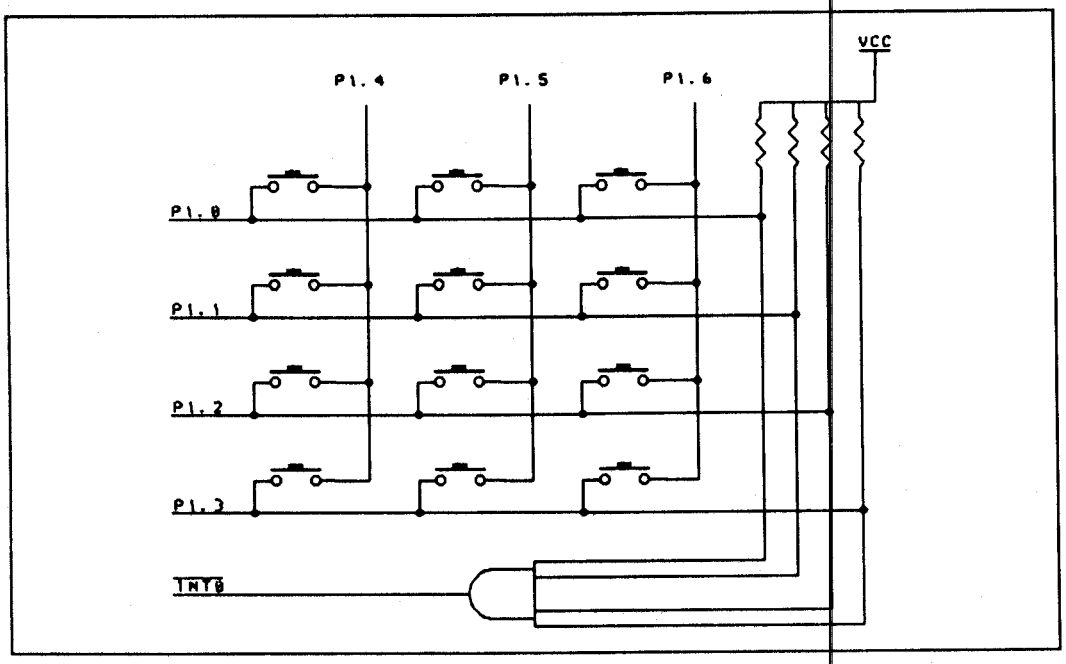
- melakukan *debouncing* pada kunci yang ditekan
- pengkodean kunci yang ditekan

Hal diatas dapat dilakukan secara perangkat lunak.

Konfigurasi papan kunci dari sistem yang dirancang dibuat dengan sususan sebagai berikut :

- baris dengan jumlah 4 buah dihubungkan dengan port P1.0 sampai dengan P1.3 secara berurutan
- kolom sebanyak 3 buah dihubungkan dengan P1.4 sampai dengan P1.6

Sebagai *scan lines* adalah kolom dan baris sebagai *return lines*. Perangkat lunak menuliskan logika 0 pada *scan lines* dan logika 1 pada *return lines*. Bila salah satu kunci



Gambar 3.9.
Rangkaian Papan Kunci

ditekan maka salah satu *scan line akan* terhubung dengan *return lines*, sehingga *return lines* tersebut akan berubah ke logika 0. Keempat *return lines* dihubungkan dengan gerbang NAND, dan keluaran dari gerbang NAND dihubungkan dengan INT0 dari mikrokontroler. Jadi jika ada kunci yang ditekan akan menghasilkan sinyal interupsi pada mikrokontroler. Dan selanjutnya rutin interupsi menterjemahkan kode kunci yang ditekan pada kode *ASCII* sehingga bisa di tampilkan pada rangkaian display. Sebelum kunci dikodekan harus dilakukan *debouncing*. Hal ini perlu dilakukan karena kontak mekanis akan berosilasi sebelum stabil.

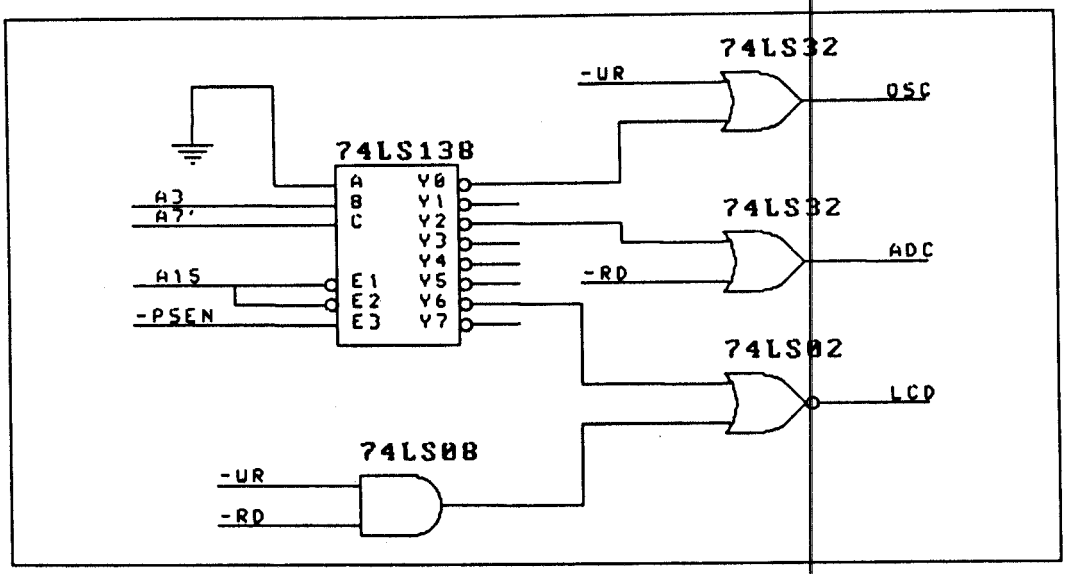
III.1.10. RANGKAIAN DEKODER

Rangkaian dekoder digunakan untuk memilih peralatan yang akan diakses dengan alamat memori program eksternal, yaitu osilator, tampilan dan ADC. Rangkaian dekoder pada tugas akhir ini menggunakan IC 74LS138 dengan beberapa gerbang logika lainnya.

Perencanaan rangkaian dekoder harus memperhatikan peta memori yang direncanakan seperti pada tabel 3.1. berikut :

Tabel 3.1. Alamat Input/Output

Alamat	Device
0088H - 008CH	Tampilan
0008H - 000CH	ADC
0000H - 0004H	Osilator

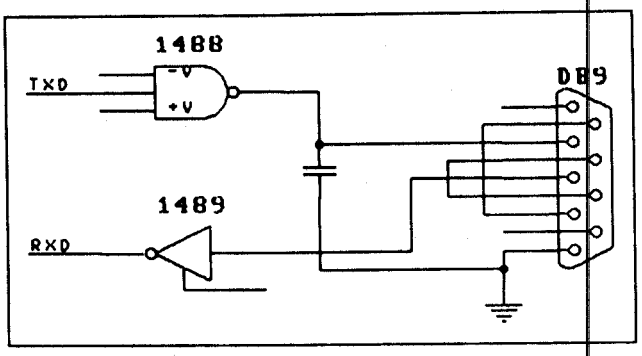


Gambar 3.10.
Rangkaian Dekoder

III.1.11. RANGKAIAN SERIAL

Rangkaian serial digunakan untuk mengubah level TTL ke level RS-232C atau sebaliknya. Kedua level ini sinyal ini berbeda dalam pendefinisian level logikanya. Pada level TTL logika '0' didefinisikan pada level 0 - o.4V, logika '1' didefinisikan pada level 2.4V - 5V. Pada standar level RS-232C, logika '0' didefinisikan pada tegangan +3V -+15V, sedangkan logika '1' pada tegangan (-3V) - (-15V). Pada sistem yang direncanakan, data serial dikirimkan melalui kabel trasmisi pada level standar RS-232C, sedangkan level yang berhubungan dengan mikrokontroler ata peripheralnya adalah standar level TTL. Oleh karena itu, sebelum data serial diterima oleh mikrokontroler atau peripheralnya, data tersebut harus diubah terlebih dahulu ke level RS-232C. Untuk mengubah level RS-232C ke level TTL digunakan IC MC1489, sedangkan untuk mengubah level TTL ke level RS-232C digunakan MC1488.

Rangkaian pengubah level ini bisa dilihat pada gambar 3.11.



Gambar 3.11.
Rangkaian Konverter Tegangan Pada Saluran Serial.

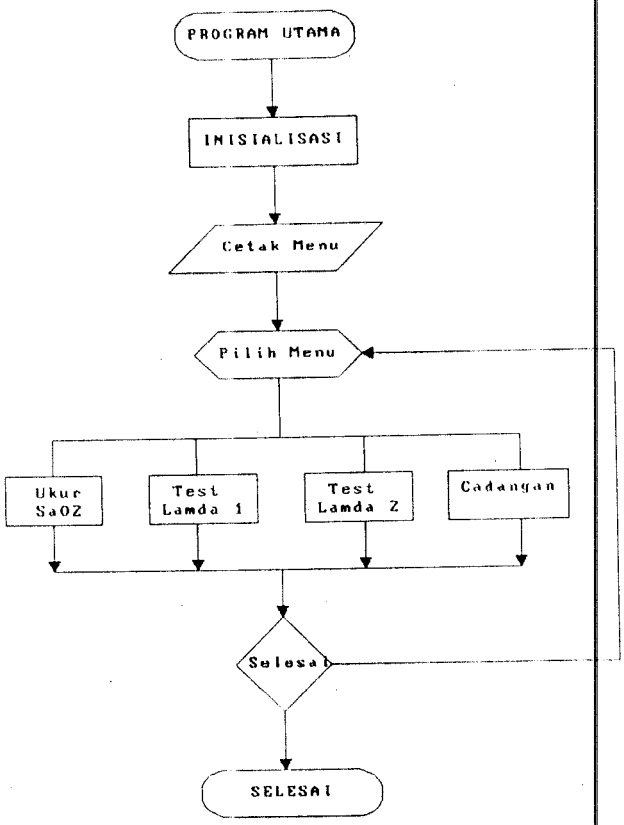
III.2. PERENCANAAN PERANGKAT LUNAK

Perangkat lunak terdiri dari dua bagian, yaitu perangkat lunak pada sistem mikrkontroler dan perangkat lunak pada komputer. Program mikrokontroler ditulis dalam bahasa assembly 8031 dan kemudian dengan bantuan perangkat lunak MCS-51 *cross assembler* diubah kedalam bahasa mesin dan dimasukkan ke EPROM, sedangkan program pada komputer menggunakan bahasa Turbo C.

III.2.1. PROGRAM MIKROKONTROLER

Pada program mikrokontroler ini dilakukan proses inisialisasi 8031 dan menyiapkan kondisi awal dari peralatan agar siap menerima data.

Setelah dilakukan reset, akan dialakukan inisialisasi, yang meliputi isi register stack pointer (SP), menginisialisasii yang akan digunakan beserta prioritasnya dan inisialisasi baud rate untuk data serial. Selain itu juga menginisialisasi LCD yang digunakan.



Gambar 3.12.
Flow Chart Program Utama Pada Mikrokontroler

Selanjutnya akan ditampilkan menu pilihan, yang dapat dipilih sesuai keinginan pemakai, dengan menekan papan kunci. Dan selanjutnya akan mengaktifkan subrutin yang sesuai.

Subrutin-subrutin yang digunakan adalah sebagai berikut :

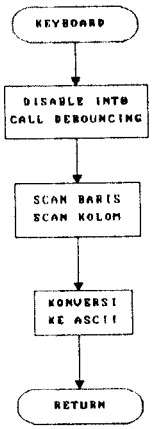
- subrutin yang menangani papan kunci
- subrutin untuk tampilan
- subrutin untuk menangani pengiriman data serial

- subrutin pengambilan dan pengolahan data

III.2.1.1. PAPAN KUNCI

Papan kunci yang digunakan seperti pada gambar perangkat keras, yaitu menggunakan tombol tekan mekanis yang dihubungkan berbentuk matrik 4 x 3. Kemudian baris dihubungkan dengan P1.0 - P1.3 dan kolom dengan P1.4 - P1.7.

Kemudian dikirimkan data 0FH ke port 1, sehingga kolom sesalu dalam keadaan low dan baris dalam keadaan high. Jika terjadi penekanan pada salah satu tombol, pada



Gambar 3.13.
Flow Chart Matrik Keyboard

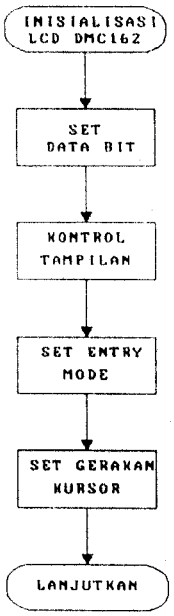
salah satu baris akan berlogika '0'. Selanjutnya gerbang OR akan meneruskan logika '0' pada pin INT0, sehingga program akan menuju alamat vektor 03h. Pada alamat vektor ini akan ditujukan pada subrutin papan kunci. Pada subrutin ini akan dicek, pada baris

berapa yang berlogika '0' , kemudian dikirim data 70H sehingga kolom akan

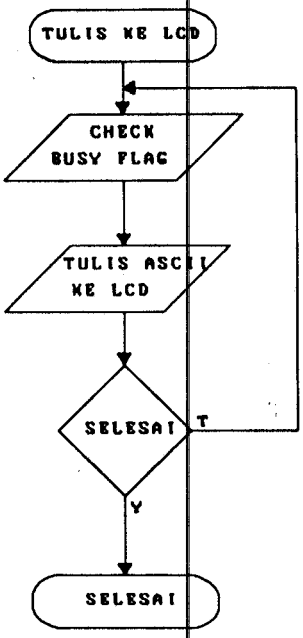
MILIK PERPUSTAKAAN
INSTITUT TEKNOLOGI
SEPULUH - NOPEMBER

berlogika '1' dan baris berlogika '0'. Karena ada tombol yang ditekan maka salah satu kolom akan berlogika '0'. Selanjutnya hasil pembacaan baris dan kolom papan kunci akan dikonversikan ke kode ASCII, sehingga bisa dipakai untuk keperluan selanjutnya.

III.2.1.2. TAMPILAN



Gambar 3.14
Flow Chart Inisialisai LCD

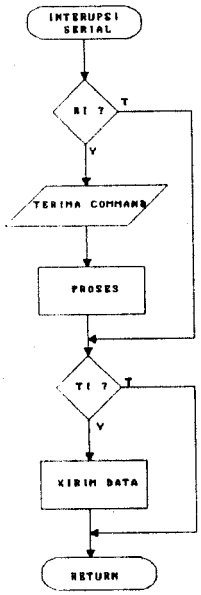


Gambar 3.15.
Flow Chart Tulis Ke LCD

Proses penanganan tampilan tidak terlalu rumit, hal ini dikarenakan adanya kemudahan yang terdapat pada LCD DMC162, antara lain adanya karakter generator. Sehingga setelah dilakukan inisialisasi pada program untuk LCD dan akanmelakukan penulisan karakter tertentu pada LCD, cukup dengan mengirimkan kode ASCII-nya saja.

Hal penting yang harus dilakukan adalah pada waktu membaca/menulis ke LCD. Misalkan akan melakukan penulisan, maka pin WR dari LCD DMC 162 harus diaktifkan, karena pada rangkaian tampilan mempunyai alamat 88H-8CH dan pin A0 mikrokontroler dihubungkan dengan pin WR LCD, maka DPTR cukup diisi alamat 89H dan kode karakter yang dikirim ke alamat tersebut akan ditampilkan.

III.2.1.3. KOMUNIKASI SERIAL



Gambar 3.16.
Flow Chart Komunikasi Serial

Komunikasi serial dilakukan dengan IBM-PC. Proses pengiriman dimulai dengan PC menginterupsi mikrokontroler sehingga melaksanakan rutin serial. Selanjutnya PC mengirimkan kode permintaan data , mikrokontroler setelah menerima kode tersebut mengirimkan data yang

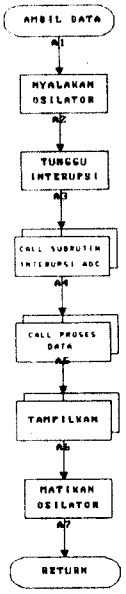
diminta kembali PC. Proses komunikasi disini dilakukan dengan kecepatan 4800 bps.

III.2.1.4. PENGAMBILAN DAN PENGOLAHAN DATA

Proses pengambilan data dari sensor dilakukan dalam beberapa tahap sebagai berikut :

- program akan mengirimkan data 1h pada alamat 0h sehingga buffer 74LS244 akan transparan sehingga LED1 dan LED2 akan menyala bergantian.
- setengah waktu dari menyalanya tiap LED akan memberikan sinyal start konversi pada ADC.
- setelah ADC selesai membaca dan mengkonversi data, ADC akan memberikan sinyal interup yang dihubungkan ke pin INT1
- selanjutnya, setelah mikrokontroler menerima interup dari ADC, program akan menuju alamat vektor 13H dan data pada ADC akan dibaca oleh mikrokontroler.
- setelah rutin pembacaan ADC selesai, osilator dimatikan kembali
- kemudian data hasil pembacaan ADC dari masing-masing LED akan diolah dan ditampilkan pada LCD

Proses pembacaan data agar tidak salah selalu mengecek pada pin P3.4. Jika P3.4 kondisinya *high* berarti LED1 yang yang aktif dan bila kondisinya *low* maka berarti LED2 yang aktif.

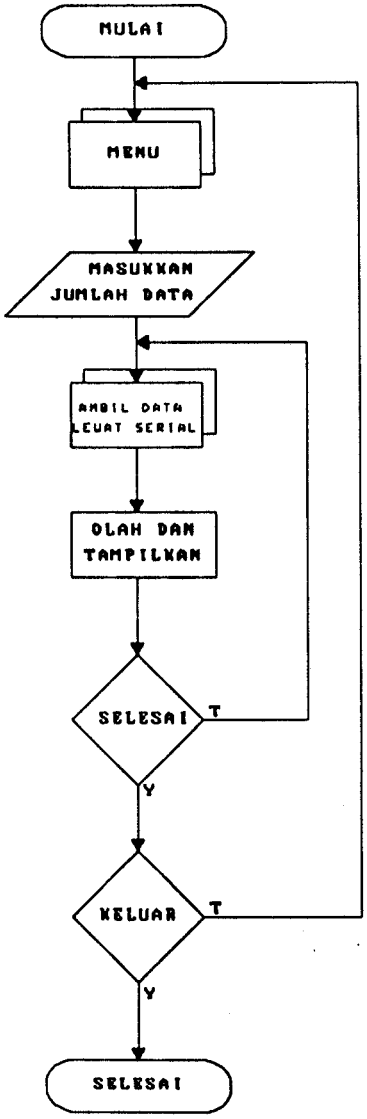


Gambar 3.17.
Flow Chart Pengambilan Dan Pengolahan Data

III.2.2. PERANGKAT LUNAK PADA KOMPUTER

Perangkat lunak pada PC menggunakan bahasa C, dengan menggunakan kompiler Turbo C, buatan Borland. Pada program komputer data hasil pengukuran dapat disimpan pada magnetik disk dalam sistem database sesuai data pasien. Selain itu data hasil pengukuran dapat ditampilkan dalam bentuk grafik.

Hal yang ditangani dalam perangkat lunak pada komputer adalah penanganan komunikasi serial dan pengolahan data.



Gambar 3.18.
Flow Chart Pengolahan Data Pada Komputer

BAB IV

PENGUJIAN DAN PENGUKURAN

Setelah alat yang dirancang dirakit dan perangkat lunak sudah siap, maka tahap selanjutnya adalah melakukan pengujian dan pengukuran.

Pengujian peralatan dilakukan dengan mengamati sinyal-sinyal yang dihasilkan pada titik tertentu dari rangkaian. Sedangkan pengukuran dilakukan pada beberapa sampel.

IV.1. PENGAMATAN SINYAL

Pengamatan sinyal dilakukan pada IC 555 yang berfungsi sebagai osilator, IC 74LS74 sebagai pembagi frekuensi.

IC 555 yang berfungsi sebagai oasilator pembangkit denyut/gelombang persegi diamati pada pin 3 terhadap *ground* yang berfungsi sebagai output multivibrator astable. Pengamatan yang dilakukan dengan osiloskop didapat sinyal persegi dengan *duty cycle* 50 % dan frekuensi sinyal sebesar 4 KHz

Pengamatan pada pin 9 dari IC 74LS74 yang berfungsi sebagai pembagi frekwensi. Dari hasil pengamatan didapat sinyal persegi dengan frekuensi 2 KHz dan *duty cycle* 50 %.

Dengan demikian osilator berfungsi sesuai yang direncanakan, yaitu menyalakan LED sebagai sumber sinyal secara bergantian dengan frekuensi 2 KHz dan separuh dari nyalanya LED dilakukan pengukuran yang dimulai

saat IC 555 memberikan sinyal start konversi setiap 4 KHz.

IV.2. PENGAMATAN SALURAN KOMUNIKASI SERIAL

Pengamatan komunikasi serial dilakukan dengan memberikan sinyal secara software dari komputer PC ke mikrokontroler 8031. dan memerintahkan mikrokontroler melakukan proses pengukuran.

Pengamatan komunikasi serial dilakukan dengan kecepatan transfer data dari komputer ke mikrokontroler sebesar 4800 bps.

Dari hasil pengamatan tidak ditemukan kesalah pengiriman, maupun kesalahan pembacaan, sehingga komunikasi serial dianggap memenuhi syarat yang diinginkan.

IV.3. PENGUKURAN

Pengukuran dilakukan dengan mengambil data oksigen saturasi dari sejumlah sampel dari laboratorium Krimatologi Klinik RSUD Dr. Soetomo, serta pengambilan data R_{795} dan R_{665} dari alat yang dibuat. Pengambilan sampel di rumah sakit dan pengambilan data dilakukan pada saat yang berurutan dan pada orang yang sama. Pengambilan data ini dilakukan untuk menentukan konstanta-konstanta yang diperlukan dan menguji sejauh mana sensor yang digunakan mendekati alat yang telah ada.

Darah yang diambil adalah darah vena, hal ini karena untuk pengambilan darah arteri lebih sulit dan memerlukan prosedur khusus. Pengukuran dilakukan pada 5 sampel, hasil pengukuran ditampilkan pada

tabel dibawah ini :

Tabel 4.1. Hasil Pengukuran

SaO_2	R_{795}	R_{665}	
65 %	217	185	
70 %	189	176	
67 %	224	210	
70 %	232	200	
71 %	219	195	
93 %	239	70	
95 %	250	50	

Dari hasil pengukuran bisa didapat konstanta-konstanta A dan B, untuk mendapatkan harga A dan B, maka data dianggap linier, sehingga bisa dilakukan pendekatan analisis garis lurus terbaik dengan metode kuadrat terkecil.

$$SaO_2 = A + B\frac{R_{795}}{R_{665}}$$

dengan menggunakan regresi linier sederhana persamaan ini bisa didekati :

$$\bar{y} = a + b.\bar{x}$$

dengan

y = SaO_2

a = A

b = B

$x = R_{795}/R_{665}$

$$b = \frac{n\sum_{i=1}^{n} x_i y_i - (\sum_{i=1}^{n} x_i)(\sum_{i=1}^{n} y_i)}{n\sum_{i=1}^{n} x_i^2 - (\sum_{i=1}^{n} x_i)^2}$$

$$a = \bar{y} - b\bar{x}$$

dari persamaan diatas didapat harga konstanta sebagai berikut :

a = 58,74

b = 8,95

Selanjutnya dilakukan analisa variansi untuk menguji nilai keterandalan dengan persamaan :

$$r = \frac{J_{xy}}{\sqrt{J_{xx} J_{yy}}}$$

dimana :

$$J_{xx} = \sum_{i=1}^{n} x_i^2 - \frac{[\sum_{i=1}^{n} x_i]^2}{n}$$

$$J_{yy} = \sum_{i=1}^{n} y^2{}_i - \frac{[\sum_{i=1}^{n} y_i]^2}{n}$$

$$J_{xy} = \sum_{i=1}^{n} x_i y_i - \frac{[\sum_{i=1}^{n} x_i] [\sum_{i=1}^{n} y_i]}{n}$$

didapat r = 0,593 dan r^2 = 0.352

berarti koefisien korelasi dari alat tehadap dua variabel acak adalah 35,2%

BAB V

PENUTUP

V.I. KESIMPULAN

Setelah dilakukan pengujian peralatan beserta komponen pendukungnya berdasarkan teori-teori dari literatur, maka didapat kesimpulan sebagai berikut :

- Sensor optis tidak memenuhi spesifikasi yang diharapkan
- Hasil pengukuran yang ditunjukan belum stabil
- Dari hasil analisa kehandalan alat ukur masih rendah
- Spesifikasi alat pengukur oksigen saturasi dalam darah secara *non-invasive* mempunyai spesifikasi sebagai berikut :
  - Menggunakan metode *absorbsi-spektrometri*
  - Menampilkan Osigen Saturasi (%), yaitu perbandingan antara HbO_2 dengan Hb yang berfungsi
  - Menggunakan sensor optis tipe *finger tip reflectant*
  - Mempunyai resolusi pengukuran 8 bit, resolusi ADC 256 pada skala penuh
  - Menggunakan LCD matrik 2 baris x 16 kolom dengan kecerahan dapat diatur

- Dapat bekerja dalam mode *Stand Alone* ataupun sebagai *PC-based bedside monitor*
- Kecepatan komunikasi secara serial dengan PC sebesar 4800 bps (dapat ditingkatkan lagi)
- Program pada PC dapat menampilkan grafik antara SaO2 dan waktu pengambilan.

V.2. SARAN-SARAN

Agar didapat pengukuran yang lebih baik, perlu dilakukan perbaikan performansinya yang meliputi :

- Sensor diukur pada rentang data yang lebih lebar untuk bisa mengetahui tingkat keberhasilannya, misalnya dengan pengambilan sampel pada pasien yang mempunyai persentasi oksigen yang tinggi (olah ragawan) dan pada pasien yang mempunyai persentasi oksigen yang rendah (pasien yang mempunyai kesulitan dalam pernafasan).
- Sensor yang digunakan diganti dengan sensor yang telah diproduksi, misalnya DURASENSOR DS-100A yang digunakan oleh Siemens pada peralatan serupa, yaitu Sirecust 732
- Menggunakan ADC dengan resolusi yang lebih tinggi
- Sistem PC-based bedside monitor dapat dikembangkan lebih lanjut untuk masukan yang lebih banyak, misalnya ECG, Blood Presuare, Suhu, Heart Rate dan lain sebagainya, yang dapat dihubungkan pada PC secara serial yang tergabung menjadi satu.

DAFTAR PUSTAKA

1. Intel, MCS-51 Family of Single Chip Microprocomputer User's Manual, Intel Corp, Santa Clara U.S.A.,1981
2. Coughlin, Robert F; Driscoll, Faderick F. & Soemitro, Herman W., Penguat Operasional dan Rangkaian Terpadu Linear, Erlangga, Jakarta 1985
3. Webster JG ed Medical Intrumentation Aplication and Design. Houghton Mifflin Company, 1978
4. Walpole, Ronald E. Ilmu Peluang dan Statistika untuk Insinyur dan Ilmuwan Penerbit ITB, 1986
5. Guyton, Athur C. Fisiologi Kedokteran I dan II, Terjemahan Adji Dharma dan P. Lukmanto, EGC, Jakarta, 19983
6. Motorola, Optoelectonics Device Data, Motorola Inc,USA, 1989
7. Katysuyuki Miyasaka, Noninvasive Oximetre of Oxygen in Blood, Anesthesiological Departement National Pediatrics Hospital
8. Krane, Keneth S. Modern Physics, John Wiley & Sons, 1983
9. Setsuo Takatani, Hiroyuki Noda, Hisateru Takatano, dan Tessuzo Akutsu. A miniature Hybrid Reflection Type Optical Sensor fo Measurement of Hemoglobin Content and Oxygen Saturation of Whole Blood. IEE Trans. on Biomedical Engineering, vol 35, no.3,

March 1988.

10. Ayala, Kenneth J., The 8051 Microcontroller Architecture, Programming and Applications, Wstern Carolina University, 1991

11. Ratno Dwi Santoso, MS,Ir., Mustadjab Hary Kusnadi, MS, Ir. Analisis Rgresi, Andi Offset, Yogyakarta, 1992

```
; alamat I/O
lcd     equ  88h   ; alamat LCD
lcd1    equ  89h
lcd2    equ  8ah
osc     equ  00h   ; alamat oscilator
adc     equ  08h   ; alamat ADC

; ram

nokey      equ  20h.0
lmd1       equ  20h.1
lmd2       equ  20h.2
flag       equ  20h.3
noadc      equ  20h.4
line       equ  21h
kodekey    equ  22h
asc0       equ  23h
asc1       equ  24h
asc2       equ  25h
biner      equ  26h


bcddat     equ  29h
dataadc    equ  2ah
rdata      equ  2bh
tdata      equ  2ch
tempo      equ  2dh
sisa       equ  2eh

konstA1    equ  41h
konstA2    equ  42h
konstA3    equ  43h
konstA4    equ  44h

konstB1    equ  45h
konstB2    equ  46h
konstB3    equ  47h
konstB4    equ  48h

data1      equ  49h        ; data R795
data2      equ  4ah        ; data R665

x1         equ  4bh        ; x
x2         equ  4ch        ; di depan
x3         equ  4dh        ; koma

x4         equ  4eh        ; x dibelakang
x5         equ  4fh        ; koma

h1         equ  50h        ;
h2         equ  51h        ; hasil didepan koma
h3         equ  52h        ;
h4         equ  53h        ;

k1         equ  54h        ;
k2         equ  55h        ; hasil dibelakang koma
k3         equ  56h        ;
k4         equ  57h        ;



        org  0h
        ljmp  mulai

        org  03h
        ljmp  keypad

        org  013h
        ljmp  intadc

        org  023h
        ljmp  serial

        org  150h
rohan   db   'Rohan Budiman   ';10h
        db   '290 220 1606    ';20h
        db   'FTI Elektro ITS ';30h
        db   'S u r a b a y a ';40h
        db   'Alat Pengukur   ';50h
        db   'Oksigen Saturasi';60h
        db   'Dalam Darah     ';70h
        db   'scr NON INVASIVE';80h

        org   200h

menupil db   'Pilih menu      ';10h
        db   'Tekan [*]       ';20h
        db   '1. Ukur Oksigen ';30h
        db   '2. Test lamda 1 ';40h
        db   '3. Test lamda 2 ';50h
        db   '4. Cadangan ..  ';60h

        org   300h

pilmenu db   'Oksigen Saturasi';10h
        db   'Hasil Pengukuran';20h
        db   'Lamda 1 = ';2ah
        db   'Hasil Pengukuran';3ah
        db   'Lamda 2 = ';44h
        db   'Utk Pengembangan';54h
        db   'Lebih Lanjut    ';64h
        db   ' Test Oscilator ';74h
        db   '----   ***   ----';84h


        org  400h
mulai:
        mov  sp,#6ah
        mov  ie,#10010001b   ; (                )
        mov  ip,#00000101b   ; (prioritas interupsi-->
Serial yang lain sesuai
                             ; urutan EX0,ET0,EX1,ET1)
        mov  tmod,#00100000b
        mov  tcon,#01000000b
        mov  scon,#50h
        mov  87h,#0
        mov  th1,#0fah       ; baud rates 4800
        mov  tl1,th1
        setb nokey
        setb noadc
        mov  p1,#0fh
        setb ea
        setb p3.4
        setb p3.5
        mov  p3,#0ffh
        clr  ea
        call oscoff
        call nolbuffer

        call delay
        call delay
        call delay

initdsp:
```

```
        mov  dptr,#88h
        mov  a,#38h       ; function set 8 bits data
        movx @dptr,a

        mov  a,#01h       ; display clear
        call chkbusy
        movx @dptr,a

        mov  a,#0eh       ; display control (disp on,
cursor on, blink off)
        call chkbusy
        movx @dptr,a

        mov  a,#06h       ; entry mode set
        call chkbusy
        movx @dptr,a

        mov  a,#14h       ; cursor shift right
        call chkbusy
        movx @dptr,a

start:
        mov  dptr,#rohan
        mov  r2,#0h
        call dspclr
        call crshome

next:
        call introdsp
        cjne r2,#80h,next
        setb ea
next1:
        setb  ex0
        call  waitkey
        mov   r2,#0h
        mov   p3,#0ffh

menu:
        setb ie.4
        mov  scon,#50h
        mov  rdata,#0
        mov  tdata,#0
        setb  ex0
        call  nolbuffer
        mov   dptr,#menupil
        call  dspclr
        call  crshome
        call  display
        setb  ea
        setb  ex0
        setb ie.4
        call  waitkey
        call  delay
        setb  ea
        mov    a,kodekey
        cjne  a,#'*',turun

menu1:
        setb ie.4
        mov   dptr,#menupil
        call  dspclr
        call  crshome
        call  display
        call  nolbuffer
        setb  ea
        setb  ex0
        call  waitkey
        mov   a,kodekey
        cjne  a,#'1',menu2
        call  kosongkan
        call  proses
        call  olah
        call  SaO2
smenu11:
        call  waitkey
        mov   a,kodekey
        cjne  a,#'5',smenu12
        mov   biner,data1
        call  view
        jmp   smenu11

smenu12:
        cjne  a,#'6',smenu13
        mov   biner,data2
        call  view
        jmp   smenu11
smenu13:
        cjne  a,#'7',balik1
        call  SaO2
        jmp   smenu11
naik:
        jmp   menu1
turun:
        jmp   balik
menu2:
        cjne  a,#'2',menu3
        call  proses
        call  tamlamd1
        jmp   balik
menu3:
        cjne  a,#'3',menu4
        call  proses
        call  tamlamd2
        jmp   balik
menu4:
        cjne  a,#'4',menu9
        call  proses
        call  cadangan
        jmp   balik
menu9:
        cjne  a,#'9',menutest
        call  testosc
        jmp   balik

menutest:
        cjne  a,#0d4h,menuterus
        call  proses
        call  lihattest
        jmp   balik
menuterus:
        cjne  r2,#60h,naik

balik:
        setb   ie.0
        call   waitkey
balik1:
        call   delay
        call   nolbuffer
        mov    r2,#0h
        jmp    menu

exit:
        mov  a,#00h
        mov  r0,1
```

```
        mov  r1,a
        mov  r2,a
        mov  r3,a
        mov  r4,a
        mov  r5,a
        mov  r6,a
        mov  r7,a
        setb p3.4
        setb p3.5

        ljmp  mulai

nolbuffer:
        push  psw
        push  acc
        push  dph
        push  dpl
        setb ie.4
        mov   a,#20h
        mov    kodekey,a
        pop   dpl
        pop   dph
        pop   acc
        pop   psw
        ret


display:
        call  dspclr
        call  crshome
        call  tulis
        push  dpl
        push  dph
        mov   dptr,#lcd
        mov   a,#0c0h
        call  chkbusy
        movx  @dptr,a
        pop   dph
        pop   dpl
        call  tulis
        ret

tulis:
        mov   r1,#10h

nextchar:
        mov   a,r2
        movc  a,@a+dptr
        push  dpl
        push  dph
        mov   dptr,#lcd1
        call  chkbusy
        movx  @dptr,a
        pop   dph
        pop   dpl
        inc   r2
        djnz  r1,nextchar
        ret

tulis1:
        mov   r1,#0ah

nextchar1:
        mov   a,r2
        movc  a,@a+dptr
        push  dpl
        push  dph
        mov   dptr,#lcd1
        call  chkbusy
        movx  @dptr,a
        pop   dph
        pop   dpl
        inc   r2
        djnz  r1,nextchar1
        ret


oscon:
        push  psw
        push  acc
        push  dph
        push  dpl

        mov   dptr,#osc
        mov   a,#01h
        movx  @dptr,a

        pop   dpl
        pop   dph
        pop   acc
        pop   psw
        ret

oscoff:
        push  psw
        push  acc
        push  dph
        push  dpl

        mov   dptr,#osc
        mov   a,#00h
        movx  @dptr,a

        pop   dpl
        pop   dph
        pop   acc
        pop   psw
        ret
baris1:
       push dpl
       push dph
       push acc
       mov  dptr,#lcd
       mov  a,#80h
       call chkbusy
       movx @dptr,a
       pop  acc
       pop  dph
       pop  dpl
       ret

baris2:
       push dpl
       push dph
       push acc
       push psw
       mov  dptr,#lcd
       mov  a,#0c0h
       call chkbusy
       movx @dptr,a
       pop  psw
       pop  acc
       pop  dph
```

```
        pop  dpl
        ret



introdsp:
        call dspclr
        call crshome
        call introwrt
        push dpl
        push dph
        mov  dptr,#lcd
        mov  a,#0c0h
        call chkbusy
        movx @dptr,a
        pop  dph
        pop  dpl
        call introwrt
        ret

introwrt:
        mov  r1,#10h

itrnext:
        mov  a,r2
        movc a,@a+dptr
        push dpl
        push dph
        mov  dptr,#lcd1
        call chkbusy
        movx @dptr,a
        pop  dph
        pop  dpl
        inc  r2
;       call delay
;       call delay
        djnz r1,itrnext
        ret

delay:
        push  psw
        mov  r4,#7fh

loop1:
        mov  r7,#0ffh
        djnz r7,$
        djnz r4,loop1
        pop  psw
        ret



dbcing:
        push  psw
        mov   r0,#0fh

loop:
        mov   r1,#0efh
        djnz  r1,$
        djnz  r0,loop
        pop   psw
        ret

dbcing2:
        push  psw
        mov   r0,#07fh

loop2:
        mov   r1,#0ffh
        djnz  r1,$
        djnz  r0,loop2
        pop   psw
        ret

crshome:
        push dpl
        push dph
        push acc
        mov  dptr,#88h
        mov  a,#02h         ; cursor home
        call chkbusy
        movx @dptr,a
        mov  a,#80h         ; set DDRAM address 00h
        call chkbusy
        movx @dptr,a
        pop  acc
        pop  dph
        pop  dpl
        ret


dspclr:
        push dpl
        push dph
        mov  dptr,#lcd
        mov  a,#01h         ; display clear
        call chkbusy
        movx @dptr,a
        pop  dph
        pop  dpl
        ret


chkbusy:
        push acc
        push dpl
        push dph
        mov  dptr,#8ah

again:
        movx a,@dptr
        anl  a,#80h
        jnz  again
        pop  dph
        pop  dpl
        pop  acc
        ret

waitkey:
        setb ea
        setb ie.4
        setb ie.0
        jb   nokey,$
        setb ie.4
        setb nokey
        setb ea
        ret

keypad:
        push   psw
        push   acc
        push   dpl
        push   dph
```

```
        clr   ie.0
        clr   nokey

        call  dbcing
        call  dbcing
        call  dbcing
        call  dbcing

        call  nolbuffer
        mov   p1,#0fh
scan:
        mov   a,p1
        mov   r6,a
        mov   p1,#70h
        mov   a,p1
        orl   a,r6

adatekan:
        cjne  a,#06eh,dua
        mov   a,#'1'
        ljmp  akhir

dua:    cjne  a,#05eh,tiga
        mov   a,#'2'
        ljmp  akhir

tiga:   cjne  a,#03eh,empat
        mov   a,#'3'
        ljmp  akhir

empat:  cjne  a,#06dh,lima
        mov   a,#'4'
        ljmp  akhir

lima:   cjne  a,#05dh,enam
        mov   a,#'5'
        ljmp  akhir

enam:   cjne  a,#03dh,tujuh
        mov   a,#'6'
        ljmp  akhir

tujuh:  cjne  a,#06bh,lapan
        mov   a,#'7'
        ljmp  akhir

lapan:  cjne  a,#05bh,bilan
        mov   a,#'8'
        ljmp  akhir

bilan:  cjne  a,#03bh,nol
        mov   a,#'9'
        ljmp  akhir

nol:    cjne  a,#057h,bintang
        mov   a,#'0'
        ljmp  akhir

bintang:
        cjne  a,#067h,pagar
        mov   a,#'*'
        ljmp  akhir

pagar:
        cjne  a,#037h,salah
        mov   a,#0d4h
        jmp   akhir

salah:
        mov   a,#'X'
akhir:
        mov   kodekey,a

keypil:
        mov   dptr,#lcd
        mov   a,#8fh
        call  chkbusy
        movx  @dptr,a
        mov   a,kodekey
        mov   dptr,#lcd1
        call  chkbusy
        movx  @dptr,a

        call  dbcing2
        call  dbcing2
        call  dbcing2
        mov   p1,#0fh
        pop   dph
        pop   dpl
        pop   acc
        pop   psw

        reti

view:
        push  psw
        push  acc
        push  dph
        push  dpl
        call  dspclr
        call  crshome
        call  binbcd
        call  bcd2asc
        mov   dptr,#lcd1
        mov   a,asc0
        call  chkbusy
        movx  @dptr,a
        mov   a,asc1
        call  chkbusy
        movx  @dptr,a
        mov   a,asc2
        call  chkbusy
        movx  @dptr,a
        pop   dpl
        pop   dph
        pop   acc
        pop   psw
        ret

cadangan:
        call  dspclr
        call  crshome
        mov   r2,#44h
        mov   dptr,#pilmenu
        call  tulis
        call  baris2
        call  tulis
        ret

testosc:
        call  dspclr
        call  crshome
        mov   r2,#64h
        mov   dptr,#pilmenu
        call  tulis
```

```
        call   baris2
        call   tulis
        call   oscon
        call   delay
        call   waitkey
        call   delay
        call   oscoff
        ret

SaO2:
        push   dpl
        push   dph
        push   acc
        push   psw

        call   dspclr
        call   crshome
        mov    r2,#0h
        mov    dptr,#pilmenu
        call   tulis
        call   baris2

hasil:
        mov    a,h1
        cjne   a,#0,hasil1
        jmp    trs2
hasil1:
        call   cetak
trs2:
        mov    a,h2
        cjne   a,#0,hasil2
        jmp    trs3
hasil2:
        call   cetak
trs3:
        mov    a,h3
        cjne   a,#0,hasil3
        jmp    trs4
hasil3:
        call   cetak
trs4:
        mov    a,h4
        call   cetak


        mov    a,#','
        mov    dptr,#lcd1
        call   chkbusy
        movx   @dptr,a

        mov    a,k1
        call   cetak
        mov    a,k2
        call   cetak
        mov    a,k3
        call   cetak
        mov    a,k4
        call   cetak

        mov    dptr,#lcd1
        mov    a,#' '
        call   chkbusy
        movx   @dptr,a
        mov    a,#'%'
        call   chkbusy
        movx   @dptr,a

        pop    psw
        pop    acc
        pop    dph
        pop    dpl
        ret

cetak:
        add    a,#30h
        mov    dptr,#lcd1
        call   chkbusy
        movx   @dptr,a
        ret

tamlamd1:
        call   dspclr
        call   crshome
        mov    dptr,#pilmenu
        mov    r2,#10h
        call   tulis
        call   baris2
        call   tulis1

        mov    a,data1
        mov    biner,#0
        mov    biner,a
        call   binbcd
        call   bcd2asc

        mov    dptr,#lcd1
        mov    a,asc0
        call   chkbusy
        movx   @dptr,a
        mov    a,asc1
        call   chkbusy
        movx   @dptr,a
        mov    a,asc2
        call   chkbusy
        movx   @dptr,a
        ret

tamlamd2:
        call   dspclr
        call   crshome
        mov    dptr,#pilmenu
        mov    r2,#2ah
        call   tulis
        call   baris2
        call   tulis1

        mov    biner,#0
        mov    biner,data2
        call   binbcd
        call   bcd2asc

        mov    dptr,#lcd1

        mov    a,asc0
        call   chkbusy
        movx   @dptr,a
        mov    a,asc1
        call   chkbusy
        movx   @dptr,a
        mov    a,asc2
        call   chkbusy
        movx   @dptr,a
        ret
```

```
lihattest:
      call   dspclr
      call   crshome
      mov    dptr,#pilmenu
      mov    r2,#20h
      call   tulis1

      mov    biner,#0
      mov    biner,data1
      call   binbcd
      call   bcd2asc

      mov    dptr,#lcd1

      mov    a,asc0
      call   chkbusy
      movx   @dptr,a
      mov    a,asc1
      call   chkbusy
      movx   @dptr,a
      mov    a,asc2
      call   chkbusy
      movx   @dptr,a

      mov    dptr,#pilmenu
      call   baris2
      mov    r2,#3ah
      call   tulis1

      mov    biner,#0
      mov    biner,data2
      call   binbcd
      call   bcd2asc

      mov    dptr,#lcd1

      mov    a,asc0
      call   chkbusy
      movx   @dptr,a
      mov    a,asc1
      call   chkbusy
      movx   @dptr,a
      mov    a,asc2
      call   chkbusy
      movx   @dptr,a
      ret


proses:
      clr    ie.0
      setb   lmd1
      setb   lmd2

      call   oscon
      call   delay
      call   ambiladc
      setb   lmd1
      setb   lmd2
      call   oscoff
      ret

; modul konversi binery/hex ke bcd
; input data biner pada alamat 71h
; output data bcd pada alamat 59h,5ah

binbcd:     ; awal program
      push psw
      push acc

      mov  a,#0
      mov  r6,#03h
      mov  r1,#bcddat     ; alamat untuk menyimpan
data bcd

cler:         ; clear ram addres untuk bcd data
      mov  @r1,a
      dec  r1
      djnz r6,cler

      mov  r7,#08h    ; jumlah bit data = 8

lopbin:
      mov  r1,#biner   ; alamat data biner

shiftl:
      mov  a,@r1
      rlc  a
      mov  @r1,a
      dec  r1
                  ; add carry & double bcd data
      mov  r1,#bcddat    ; alamat data bcd
      mov  r6,#02h

bcdadj:
      mov  a,@r1
      addc a,@r1
      da   a
      mov  @r1,a
      dec  r1
      djnz r6,bcdadj
      djnz r7,lopbin

      pop  acc
      pop  psw
      ret

; soubroutin bcd to ascii0 pada 5fh
; ascii1 pada 60h
; ascii2 pada 61h

bcd2asc:
      push psw
      push acc

      mov  r1,#bcddat  ;bcd data location
      mov  a,@r1       ;get bcd data
      anl  a,#0f0h
      swap a
      orl  a,#'0'      ;convert to ascii
      mov  r1,#asc1    ;bank data to save ascii 1
      mov  @r1,a       ;saver 1st ascii data

      mov  r1,#bcddat  ;bcd data location
      mov  a,@r1       ;get bcd data

      anl  a,#0fh
      orl  a,#'0'      ;convert to ascii
      mov  r1,#asc2    ;bank data to save ascii 2
      mov  @r1,a       ;save 2nd ascii data

      mov  r1,#bcddat  ;bcd data location
      dec  r1
      mov  a,@r1       ;get bcd data
```

```
        mov     a,x5
        mov     b,konstB3
        mul     ab
        add     a,tempo
        mov     b,#10
        div     ab
        mov     tempo,a
        mov     k3,b            ; k3 baru diisi

        mov     a,x5
        mov     b,konstB2
        mul     ab
        add     a,tempo
        mov     b,#10
        div     ab
        mov     tempo,a
        mov     k2,b            ; k2 baru diisi

        mov     a,x5
        mov     b,konstB1
        mul     ab
        add     a,tempo
        mov     b,#10
        div     ab
        mov     tempo,a
        mov     k1,b            ; k1 baru diisi

;..... 4

        mov     a,x4
        mov     b,konstB4
        mul     ab
        mov     b,#10
        div     ab
        mov     tempo,a
        mov     a,b
        add     a,k3
        mov     b,#10
        div     ab
        mov     k3,b            ; k3 ditambah
        add     a,k2
        add     a,tempo
        mov     b,#10
        div     ab
        mov     k2,b            ; k2 ditambah
        add     a,k1
        mov     b,#10
        div     ab
        mov     k1,b            ; k1 ditambah
        mov     h4,a            ; h4 baru diisi

        mov     a,x4
        mov     b,konstB3
        mul     ab
        mov     b,#10
        div     ab
        mov     tempo,a
        mov     a,b
        add     a,k2
        mov     b,#10
        div     ab
        mov     k2,b            ; k2 ditambah
        add     a,k1
        add     a,tempo
        mov     b,#10
        div     ab
        mov     k1,b            ; k1 ditambah
        add     a,h4
        mov     b,#10
        div     ab
        mov     h4,b            ; h4 ditambah
        mov     h3,a            ; h3 baru diisi

        mov     a,x4
        mov     b,konstB2
        mul     ab
        mov     b,#10
        div     ab
        mov     tempo,a
        mov     a,b
        add     a,k1
        mov     b,#10
        div     ab
        mov     k1,b            ; k1 ditambah
        add     a,h4
        add     a,tempo
        mov     b,#10
        div     ab
        mov     h4,b            ; h4 ditambah
        add     a,h3
        mov     b,#10
        div     ab
        mov     h3,b            ; h3 ditambah
        mov     h2,a            ; h2 baru diisi

        mov     a,x4
        mov     b,konstB1
        mul     ab
        mov     b,#10
        div     ab
        mov     tempo,a
        mov     a,b
        add     a,h4
        mov     b,#10
        div     ab
        mov     h4,b            ; h4 ditambah
        add     a,h3
        add     a,tempo
        mov     b,#10
        div     ab
        mov     h3,b            ; h3 ditambah
        add     a,h2
        mov     b,#10
        div     ab
        mov     h2,b            ; h2 ditambah
        mov     h1,a            ; h1 baru diisi

;..... 3

        mov     a,x3
        mov     b,konstB4
        mul     ab
        mov     b,#10
        div     ab
        mov     tempo,a
        mov     a,b
        add     a,k2
        mov     b,#10
        div     ab
        mov     k2,b            ; k2 ditambah
        add     a,k1
        add     a,tempo
```

```
mov   b,#10
div   ab
mov   k1,b          ; k1 ditambah
add   a,h4
mov   b,#10
div   ab
mov   h4,b          ; h4 ditambah
add   a,h3
mov   b,#10
div   ab
mov   h3,b          ; h3 ditambah
add   a,h2
mov   b,#10
div   ab
mov   h2,b          ; h2 ditambah
add   a,h1
mov   h1,a          ; h1 ditambah

mov   a,x3
mov   b,konstB3
mul   ab
mov   b,#10
div   ab
mov   tempo,a
mov   a,b
add   a,k1
mov   b,#10
div   ab
mov   k1,b          ; k1 ditambah
add   a,h4
add   a,tempo
mov   b,#10
div   ab
mov   h4,b          ; h4 ditambah
add   a,h3
mov   b,#10
div   ab
mov   h3,b          ; h3 ditambah
add   a,h2
mov   b,#10
div   ab
mov   h2,b          ; h2 ditambah
add   a,h1
mov   h1,a          ; h1 ditambah

mov   a,x3
mov   b,konstB2
mul   ab
mov   b,#10
div   ab
mov   tempo,a
mov   a,b
add   a,h4
mov   b,#10
div   ab
mov   h4,b          ; h4 ditambah
add   a,h3
add   a,tempo
mov   b,#10
div   ab            ; h3 ditambah
mov   h3,b
mov   b,#10
add   a,h2
div   ab
mov   h2,b          ; h2 ditambah
add   a,h1
mov   h1,a          ; h1 ditambah

mov   a,x2
mov   b,konstB1
mul   ab
mov   b,#10
div   ab
mov   tempo,a
mov   a,b
add   a,h3
mov   b,#10
div   ab
mov   h3,b          ; h3 ditambah
add   a,h2
add   a,tempo
mov   b,#10
div   ab
mov   h2,b          ; h2 ditambah
add   a,h1
mov   h1,a          ; h1 ditambah

;..... 2

mov   a,x2
mov   b,konstB4
mul   ab
mov   b,#10
div   ab
mov   tempo,a
mov   a,b
add   a,k1
mov   b,#10
div   ab
mov   k1,b          ; k1 ditambah
add   a,h4
add   a,tempo
mov   b,#10
div   ab
mov   h4,b          ; h4 ditambah
add   a,h3
mov   b,#10
div   ab
mov   h3,b          ; h3 ditambah
add   a,h2
mov   b,#10
div   ab
mov   h2,b          ; h2 ditambah
add   a,h1
mov   h1,a          ; h1 ditambah

mov   a,x2
mov   b,konstB3
mul   ab
mov   b,#10
div   ab
mov   tempo,a
mov   a,b
add   a,h4
mov   b,#10
div   ab
mov   h4,b          ; h4 ditambah
add   a,h3
add   a,tempo
mov   b,#10
div   ab
mov   h3,b          ; h3 ditambah
add   a,h2
```

```
mov   b,#10
div   ab
mov   h2,b          ; h2 ditambah
add   a,h1
mov   h1,a          ; h1 ditambah

mov   a,x2
mov   b,konstB2
mul   ab
mov   b,#10
div   ab
mov   tempo,a
mov   a,b
add   a,h3
mov   b,#10
div   ab
mov   h3,b          ; h3 ditambah
add   a,h2
add   a,tempo
mov   b,#10
div   ab
mov   h2,b          ; h2 ditambah
add   a,h1
mov   h1,a

mov   a,x2
mov   b,konstB1
mul   ab
mov   b,#10
div   ab
mov   tempo,a
mov   a,b
add   a,h2
mov   b,#10
div   ab
mov   h2,b          ; h2 ditambah
add   a,h1
add   a,tempo
mov   h1,a          ; h1 ditambah

;..... 1

mov   a,x1
mov   b,konstB4
mul   ab
mov   b,#10
div   ab
mov   tempo,a
mov   a,b
add   a,h4
mov   b,#10
div   ab
mov   h4,b          ; h4 ditambah
add   a,h3
add   a,tempo
mov   b,#10
div   ab
mov   h3,b          ; h3 ditambah
add   a,h2
mov   b,#10
div   ab
mov   h2,b          ; h2 ditambah
add   a,h1
mov   h1,a          ; h1 ditambah

mov   a,x1
mov   b,konstB3
mul   ab
mov   b,#10
div   ab
mov   tempo,a
mov   a,b
add   a,h3
mov   b,#10
div   ab
mov   h3,b          ; h3 ditambah
add   a,h2
add   a,tempo
mov   b,#10
div   ab
mov   h2,b          ; h2 ditambah
add   a,h1
mov   h1,a          ; h1 ditambah

mov   a,x1
mov   b,konstB2
mul   ab
mov   b,#10
div   ab
mov   tempo,a
mov   a,b
add   a,h2
mov   b,#10
div   ab
mov   h2,b          ; h2 ditambah
add   a,h1
add   a,tempo
mov   h1,a          ; h1 ditambah

mov   a,x1
mov   b,konstB1
mul   ab
add   a,h1
mov   h1,a

;.....  Y = A + (B x X)

mov   a,konstA4
add   a,k2
mov   b,#10
div   ab
mov   k2,b
add   a,k1
mov   b,#10
div   ab
mov   k1,b
add   a,h4
mov   b,#10
div   ab
mov   h4,b
add   a,h3
mov   b,#10
div   ab
mov   h3,b
add   a,h2
mov   b,#10
div   ab
mov   h2,b
add   a,h1
mov   h1,a

mov   a,konstA3
add   a,k1
mov   b,#10
div   ab
```

```
        mov     k1,b
        add     a,h4
        mov     b,#10
        div     ab
        mov     h4,b
        add     a,h3
        mov     h3,a

        mov     a,konstA2
        add     a,h4
        mov     b,#10
        div     ab
        mov     h4,b
        add     a,h3
        mov     b,#10
        div     ab
        mov     h3,b
        add     a,h2
        mov     b,#10
        div     ab
        mov     h2,b
        add     a,h1
        mov     h1,a

        mov     a,konstA1
        add     a,h3
        mov     b,#10
        div     ab
        mov     h3,b
        add     a,h2
        mov     b,#10
        div     ab
        mov     h2,b
        add     a,h1
        mov     h1,b
osai:
        pop     psw
        pop     acc
        pop     dpl
        pop     dph
        ret

kosongkan:
        push    acc
        mov     a,#0
        mov     k1,a
        mov     k2,a
        mov     k3,a
        mov     k4,a
        mov     h1,a
        mov     h2,a
        mov     h3,a
        mov     h4,a
        mov     sisa,a
        mov     tempo,a
        pop     acc
        ret

dariserial:
        push    dph
        push    dpl
        push    acc
        push    psw
        setb    lmd1
        setb    lmd2

        call    oscon
        call    delay
        call    ambiladc
        call    oscoff
        pop     psw
        pop     acc
        pop     dpl
        pop     dph
        ret

ambiladc:
        push    dph
        push    dpl
        push    acc
        push    psw

        setb    lmd1
        setb    lmd2
        mov     data1,#0
        mov     data2,#0
        mov     dptr,#adc
        jb      p3.4,$

lamd1:
        jnb     lmd1,ambil2
        jmp     hitung1

ambil2:
        jnb     lmd2,selesaiadc
        jmp     lamd2

hitung1:
        jnb     p3.4,lamd2
        setb    noadc
        jb      p3.5,$      ; tunggu ADC baca data
        setb    ea
        setb    ie.2        ; enable ADC interup
        jb      noadc,$     ; tunggu ADC interup
        clr     lmd1
        setb    noadc
        mov     data1,dataadc
        jnb     p3.4,$

lamd2:
        jnb     lmd2,ambil1
        jmp     hitung2

ambil1:
        jnb     lmd1,selesaiadc
        jmp     lamd1
        call    tundaadc

hitung2:
        jb      p3.4,lamd1
        setb    noadc
        jb      p3.5,$      ; tunggu ADC baca data
        setb    ea
        setb    ie.2        ; enable ADC interup
        jb      noadc,$     ; tunggu ADC interup
        setb    noadc
        clr     lmd2
        mov     data2,dataadc
        mov     dataadc,#0
        call    tundaadc
        jb      p3.4,$
        jb      lmd1,lamd1

selesaiadc:
```

```
        pop   psw                                          end   mulai
        pop   acc
        pop   dpl
        pop   dph
        ret


tundaadc:
        push  psw
        mov   r4,#04h
kurang:
        mov   r7,#07fh
        djnz  r7,$
        djnz  r4,kurang
        pop   psw
        ret


intadc:
        clr   ie.2
        mov   dptr,#adc
        call  tundaadc
        movx  a,@dptr
        mov   dataadc,a
        clr   noadc
        reti

serial:
        push    psw
        push    acc
        clr     ie.4
        clr     ea

        jbc     scon.0,receive
        jbc     scon.1,transmit
        sjmp    serisai

receive:
        mov     a,sbuf

        cjne    a,#'1',ceklagi
        mov     rdata,a
        call    dariserial
        mov     tdata,data1
        jmp     kirdata
ceklagi:
        cjne    a,#'2',serisai
        mov     rdata,a
        call    dariserial
        mov     tdata,data2
kirdata:
        call    oscoff

        setb    ti
        jbc     scon.1,transmit
        sjmp    serisai

transmit:
        mov     sbuf,tdata

serisai:
        pop   acc
        pop   psw
        mov   tdata,#0
        setb  ea
        setb  ie.4
        reti
```

```
#include <graphics.h>
#include <stdlib.h>
#include <stdio.h>
#include <conio.h>
#include <alloc.h>
#include <dos.h>
#include <bios.h>
#include "mouse.h"
#include "mouse.c"

#define CLIP_ON 1              // menghidupkan clipping viewport
#define CLIP_OFF 0                                          // mematikan clipping viewport
#define BESARDATA sizeof(Datapasien)
#define SALAH 0
#define BENAR 1
#define BESARDATA1 sizeof(Oksigen);

#define COM1        0
#define DATA_READY 0x100
#define TRUE        1
#define FALSE       0

#define SETTINGS ( 0x0c0 | 0x03 | 0x00 | 0x00)

char judul2[]=" PC-BEDSIDE MONITOR ";

int Maxx,Maxy;
int mousex,mousey;              // posisi kursor mouse
char metu='T';

#include "bntmouse.h";

struct record
{
    char Nomer[11];
    char Nama[31];
    char Umur[5];
    char JenisKelamin[14];
    char Alamat[31];
    char Kota[21];
    char Oksigen[6];
};

struct record Datapasien;

struct ukur
{
    char Nomer[11];
    char Waktu[11];
    char Hasil[5];
};

struct ukur Oksigen;
struct date tanggal;

struct POSCUR {                 // struktur kursor grafik
    int x1,y1,x2,y2;
} poscur;


union inkey{
    char ch[2];
    int tbl;
}crh;

void LayarUtama(int,int,int,int,char*,int);
void CetakInformasiViewport(void);
void Utama(void);
void RohanBudiman();
void LogoRohan(void);
void Timbul(int,int,int,int,int,int);
void Tenggelam(int,int,int,int,int,int);
```

```
void TombolF(void);
void KlikMouseUtama(void);
void Bunyi(void);
void EksekusiMenu1(void);
void EksekusiMenu2(void);
void EksekusiMenu3(void);
void EksekusiMenu4(void);
void EksekusiMenu5(void);
void Pmenu1(void);
void Pmenu2(void);
void Pmenu3(void);
void IsiData(void);
void AmbilData(void);
void LihatData(void);
void SadapKetik(int *brs,int panjang,char*xx);
void PergiKe(int kolom,int baris);
void BalikKursor(int kolom,int baris);
void BuatBlockdiag(int bny,char*oksi);
void BuatGrafik(int bny,char*oksi);
void HapusCursor();
void Blank(int kolom,int baris,int banyak);
double AmbilRS();
int RSambil(int perintah);

void TampilkanGambar(char*,int,int);


char menu1[]="F1 Edit   ";
char menu2[]="F2 Measure";
char menu3[]="F3 View   ";
char menu4[]="F4 Print  ";
char menu5[]="ESC Exit";


int main(void)
{
    // permintaan untuk deteksi otomatis
    int gdriver=DETECT,gmode,errorcode;

    // inisialisasi grafik dan variabel lokal
    initgraph(&gdriver,&gmode,"");


    errorcode=graphresult();          // baca hasil initialisasi
    if (errorcode!=grOk)              // terjadi kesalahan ?
    {
        printf("Graphics error :%s\n",grapherrormsg(errorcode));
        printf("Tekan sembarang kunci untuk tunda:");
        getch();
        exit(1);
    }
    Maxx=getmaxx();
    Maxy=getmaxy();
    setcolor(getmaxcolor());      // atur warna pengambaran

    Utama();
    cleardevice();                    // bersihkan layar grafik
    closegraph();                     // tutup layar grafik
    return 0;
}

// membuat gambar Utama

void Utama(void)
{

    int x1,y1,x2,y2;


    x1=Maxx/127;y1=Maxy/100;
    x2=Maxx-x1;y2=Maxy-y1;

   LayarUtama(x1,y1,x2,y2,judul2,CLIP_ON);
```

```
    panggil();

    clearviewport();

}
void LayarUtama(int x1,int y1,int x2,int y2,                    // posisi
                char*judul,                                     // judul
                int clip)                                       // modus yang diinginkan
{
    int posx,posy,i;
    char tgl[10],bln[2],thn[4];


    Timbul(x1-(Maxx/106.6667),y1-(Maxy/80),x2+(Maxx/106.6667),y2+(Maxy/80),LIGHTGRAY,3);
    Tenggelam(x1,y1,x2,y2,LIGHTGRAY,3);
    posx=x1+((x2-x1)/2);
    posy=y1+(Maxy/24);
    Timbul(posx-(Maxx/7.1111),posy-(Maxy/48),posx+(Maxx/7.1111),posy+(Maxy/48),LIGHTBLUE,3);
    setcolor(YELLOW);
    settextstyle(DEFAULT_FONT,HORIZ_DIR,1);
    settextjustify(CENTER_TEXT,CENTER_TEXT);
    outtextxy(posx,posy+1,judul);

    Timbul(Maxx/71.1111,Maxy/11.1628,Maxx/1.0175,Maxy/9.0566,7,1);
    Timbul(Maxx/4.4138,Maxy/8.8889,Maxx/4.1290,Maxy/1.0235,7,1);
    Timbul(Maxx/80,Maxy/9.0566,Maxx/4.4138,Maxy/6.2338,7,3);
    Timbul(Maxx/80,Maxy/5.5172,Maxx/4.4138,Maxy/4.3243,7,3);
    Timbul(Maxx/80,Maxy/3.9669,Maxx/4.4138,Maxy/3.3103,7,3);
    Timbul(Maxx/80,Maxy/3.0968,Maxx/4.4138,Maxy/2.6816,7,3);
    Timbul(Maxx/80,Maxy/2.5397,Maxx/4.4138,Maxy/2.2430,7,3);

    settextstyle(DEFAULT_FONT,HORIZ_DIR,1);
    settextjustify(LEFT_TEXT,TOP_TEXT);
    setcolor(BLACK);
    outtextxy(Maxx/16,Maxy/7.7419,menu1);
    outtextxy(Maxx/16,Maxy/5,menu2);
    outtextxy(Maxx/16,Maxy/3.6923,menu3);
    outtextxy(Maxx/16,Maxy/2.9268,menu4);
    outtextxy(Maxx/16,Maxy/2.4242,menu5);

    getdate(&tanggal);
    itoa(tanggal.da_day,tgl,10);
   strcat(tgl,"-");
    itoa(tanggal.da_mon,bln,10);
    strcat(tgl,bln);
    strcat(tgl,"-");
    itoa(tanggal.da_year,thn,10);
    strcat(tgl,thn);
    setcolor(BLACK);
    outtextxy(12,16,tgl);


    RohanBudiman();

    LogoRohan();

}

void RohanBudiman()
{
    setcolor(DARKGRAY);
    setlinestyle(SOLID_LINE,1,THICK_WIDTH);
    rectangle(Maxx/35.5556,Maxy/1.1268,Maxx/4.7761,Maxy/1.0235);
    setfillstyle(SOLID_FILL,LIGHTGRAY);
    bar(Maxx/32,Maxy/1.1215,Maxx/4.8385,Maxy/1.0278);
    settextstyle(GOTHIC_FONT,HORIZ_DIR,1);
    settextjustify(LEFT_TEXT,TOP_TEXT);

    setusercharsize(2,4,1,3);
    setcolor(BLACK);
    outtextxy(Maxx/25.61,Maxy/1.11,"Rohan Budiman");

    settextstyle(SMALL_FONT,HORIZ_DIR,1);
    setusercharsize(3,2,3,2);
```

```
    outtextxy(Maxx/18.2857,Maxy/1.0838,"2902201606");
    settextstyle(SMALL_FONT,HORIZ_DIR,2);
    outtextxy((Maxx/14.8421),(Maxy/1.0526),"Copywrite 1993,1994");
}
void LogoRohan(void)
{
    char Rohan[]=" Tugas Akhir ";
    int i;

    Tenggelam(156,54,630,468,LIGHTGRAY,1);

    Timbul(Maxx/3.2,Maxy/3.2,Maxx/1.0667,Maxy/1.6,LIGHTBLUE,3);
    Tenggelam(Maxx/3.0476,Maxy/3,Maxx/1.0847,Maxy/1.6552,LIGHTBLUE,3);

    settextstyle(DEFAULT_FONT,HORIZ_DIR,3);
    settextjustify(LEFT_TEXT,TOP_TEXT);
    for (i=0;i<=2;i++){
        setcolor(WHITE);
        outtextxy((Maxx/2.56)+i,(Maxy/2.4)+i,Rohan);
    }

    for (i=0;i<=2;i++){
        setcolor(DARKGRAY);
        outtextxy((Maxx/2.5197)+i,(Maxy/2.3529)+i,Rohan);
    }

    for (i=0;i<=2;i++){
        setcolor(LIGHTGRAY);
        outtextxy((Maxx/2.5397)+i,(Maxy/2.3762)+i,Rohan);
    }

}



panggil()
{

    ResetMouse();
    MoMinMax(Maxx/127,Maxy/100,Maxx-(Maxx/127),Maxy-(Maxy/100));

    BentukCursorMouse(arrow,1,0);
    StatusMouse(MUNCUL);

    for(;;){
        crh.tbl=bioskey(1);      // baca kunci tanpa berhenti
        if (bioskey(1)!=0) TombolF();
        if (tbKiriDitekan()){
            while(tbKiriDitekan()){
                delay(50);
                mousex=MouseXGrafik();
                mousey=MouseYGrafik();

                KlikMouseUtama();
                break;
            }

        }
        if (tbKananDitekan()){
            Bunyi();
            EksekusiMenu5();
            break;
        }


    }
}

void TombolF(void)
{

    crh.tbl=bioskey(0);      // baca kunci dengan berhenti
    StatusMouse(HILANG);
    if (crh.ch[0]){
```

```
        switch(crh.ch[0]){
            case 27:                                        // Escape
                Bunyi();
                EksekusiMenu5();
                break;

            default:
                cprintf("%c",crh.ch[0]);
        }
    }
    if (crh.ch[1]){

        setlinestyle(SOLID_LINE,0,3);
        setfillstyle(SOLID_FILL,BLACK);

        switch(crh.ch[1]){
            case 59:                                        // F1
                Bunyi();
                EksekusiMenu1();
                break;

            case 60:                    // F2
                Bunyi();
                EksekusiMenu2();
                break;
                Bunyi();
            case 61:                    // F3
                Bunyi();
                EksekusiMenu3();
                break;
            case 62:                    // F4
                Bunyi();
                EksekusiMenu4();
                break;
        }
    }


    StatusMouse(MUNCUL);

}

void KlikMouseUtama(void)
{
    StatusMouse(HILANG);


    if ((mousex>(Maxx/58.1818)) && (mousex<(Maxx/4.507)))
    {
        Bunyi();

        if ((mousey>(Maxy/8.5714)) && (mousey<(Maxy/6.4865)))
            EksekusiMenu1();

        if ((mousey>(Maxy/5.3333)) && (mousey<(Maxy/4.4444)))
            EksekusiMenu2();

        if ((mousey>(Maxy/3.8710)) && (mousey<(Maxy/3.3803)))
            EksekusiMenu3();

        if ((mousey>(Maxy/3.0380)) && (mousey<(Maxy/2.7273)))
            EksekusiMenu4();

        if ((mousey>(Maxy/2.5)) && (mousey<(Maxy/2.2749)))
            EksekusiMenu5();
    }
    StatusMouse(MUNCUL);

}

void Bunyi(void)
{
    int i;
```

```
    sound(700);
    delay(70);

    sound(400);
    delay(30);

    nosound();
}


void EksekusiMenu1(void)
{

    Tenggelam(Maxx/80,Maxy/9.0566,Maxx/4.4138,Maxy/6.2338,LIGHTGRAY,3);



    settextstyle(DEFAULT_FONT,HORIZ_DIR,1);
    settextjustify(LEFT_TEXT,TOP_TEXT);
    setcolor(BLACK);
    outtextxy(Maxx/16.8421,Maxy/8.1356,menu1);


    StatusMouse(HILANG);
                    // disini panggil prosedur yang diperlukan
    Pmenu1();

    StatusMouse(MUNCUL);
    getch();

    Timbul(Maxx/80,Maxy/9.0566,Maxx/4.4138,Maxy/6.2338,LIGHTGRAY,3);
    setcolor(BLACK);
    outtextxy(Maxx/16,Maxy/7.7419,menu1);
   LogoRohan();

}


void Pmenu1()
{
    Timbul(160,60,625,465,LIGHTGRAY,1);

    settextstyle(DEFAULT_FONT,HORIZ_DIR,2);
    settextjustify(LEFT_TEXT,TOP_TEXT);
    setcolor(BLACK);

    outtextxy(300,70,"DATA PASIEN");
    setcolor(BLUE);
    settextstyle(DEFAULT_FONT,HORIZ_DIR,1);

    outtextxy(180,110 ,"Data Pribadi");
    line(180,120,273,120);
    outtextxy(180,140,"Nomer          :");
    outtextxy(180,160,"Nama           :");
    outtextxy(180,180,"Umur           :");
    outtextxy(385,180,"th.");
    outtextxy(180,200,"Jenis Kelamin  :");
    outtextxy(180,220,"Alamat         :");
    outtextxy(180,240,"Kota           :");
    Tenggelam(340,135,430,150,CYAN,2);                          //nomer
    Tenggelam(340,155,590,170,CYAN,2);                          //nama
    Tenggelam(340,175,380,190,CYAN,2);                          //umur
    Tenggelam(340,195,450,210,CYAN,2);  //jenis kelamin
    Tenggelam(340,215,590,230,CYAN,2);  //alamat
    Tenggelam(340,235,510,250,CYAN,2);  //kota

    IsiData();

}


// Untuk memasukkan Data isian
```

```
void IsiData(void)
{
    int baris=1;
    int pfield[6]={10,30,4,13,30,20};
    FILE *fp1;
    register int a;
    int banrec,ketemu=0;
    char notemp[10];
    char *namafile1=NULL;
    char *transf,*ptr_str;

    if ((namafile1=(char*)calloc(20,sizeof(char)))==NULL){
        outtextxy(430,400,"Memory Penuh");
        getch();
        return;
    }
    strcpy(namafile1,"pasien.rhn");


    if ((transf=(char*)calloc(30,sizeof(char)))==NULL){
        outtextxy(430,400,"Memory Penuh");
        getch();
        return;
    }
    settextjustify(LEFT_TEXT,BOTTOM_TEXT);
    SadapKetik(&baris,pfield[baris-1],transf);
    settextjustify(LEFT_TEXT,TOP_TEXT);

    strncpy(notemp,transf,10);

    if (strtod(transf,&ptr_str)==0){
        free(transf);
        free(namafile1);
        return;
    }

    if ((fp1=fopen(namafile1,"r+t"))==NULL)
        if ((fp1=fopen(namafile1,"at"))==NULL) return;

    banrec=panjangfile(fp1);

    if (banrec!=0){
        for (a=0;a<=banrec;a++){
            bacarecord(fp1,a);
            if (stricmp(notemp,Datapasien.Nomer)==0){
                ketemu++;
                settextjustify(LEFT_TEXT,BOTTOM_TEXT);
                PergiKe(1,2);
                outtext(Datapasien.Nama);
                HapusCursor();
                PergiKe(1,3);
                outtext(Datapasien.Umur);
                HapusCursor();
                PergiKe(1,4);
                outtext(Datapasien.JenisKelamin);
                HapusCursor();
                PergiKe(1,5);
                outtext(Datapasien.Alamat);
                HapusCursor();
                PergiKe(1,6);
                outtext(Datapasien.Kota);
                HapusCursor();
                settextjustify(LEFT_TEXT,TOP_TEXT);
                break;

            }
        }
    }
    if (ketemu==0) Initrecord();

    strncpy(Datapasien.Nomer,transf,10);
    do {

        if(ketemu==0){
```

```
            fclose(fp1);
            if ((fp1=fopen(namafile1,"at"))==NULL) return;
        }
        else {

            switch (baris){
                case 1:
                    strncpy(transf,Datapasien.Nomer,10);
                    break;
                case 2:
                    strncpy(transf,Datapasien.Nama,20);
                    break;
                case 3:
                    strncpy(transf,Datapasien.Umur,4);
                    break;
                case 4:
                    strncpy(transf,Datapasien.JenisKelamin,13);
                    break;
                case 5:
                    strncpy(transf,Datapasien.Alamat,30);
                    break;
                case 6:
                    strncpy(transf,Datapasien.Kota,20);
                    break;
            }
        }

        settextjustify(LEFT_TEXT,BOTTOM_TEXT);
        SadapKetik(&baris,pfield[baris-1],transf);
        settextjustify(LEFT_TEXT,TOP_TEXT);
        switch (baris-1){
            case 1:
                strncpy(Datapasien.Nomer,transf,10);
                break;
            case 2:
                strncpy(Datapasien.Nama,transf,20);
                break;
            case 3:
                strncpy(Datapasien.Umur,transf,4);
                break;
            case 4:
                strncpy(Datapasien.JenisKelamin,transf,13);
                break;
            case 5:
                strncpy(Datapasien.Alamat,transf,30);
                break;
            case 6:
                strncpy(Datapasien.Kota,transf,20);
                break;
        }

    } while (baris<7);

    free(transf);
    if(ketemu!=0)
        Tulisrecord(fp1,a);
    else {
        banrec=banrec+1;
        fwrite(&Datapasien,BESARDATA,1,fp1);

        if (banrec>1){
            fclose(fp1);
            if((fp1=fopen(namafile1,"r+t"))==NULL) return;
          banrec=panjangfile(fp1);
            Sortdata(fp1,banrec);
        }
    }

    fclose(fp1);

    free(namafile1);
}

Initrecord()
```

```
{
    Datapasien.Nomer[0]=NULL;
    Datapasien.Nama[0]=NULL;
    Datapasien.Umur[0]=NULL;
    Datapasien.JenisKelamin[0]=NULL;
    Datapasien.Alamat[0]=NULL;
    Datapasien.Kota[0]=NULL;
}


Sortdata(fp1,banrec)
FILE *fp1;
int banrec;
{
    register int a,b;
    char nomertemp[11];

    for (a=0;a<banrec-1;a++)
        for (b=a+1;b<banrec;b++){
            bacarecord(fp1,a);
            strcpy(nomertemp,Datapasien.Nomer);
            bacarecord(fp1,b);
            if (strcmp(nomertemp,Datapasien.Nomer)>0)
                swaprecord(fp1,b,a);
        }
}
swaprecord(fp1,i,j)
FILE *fp1;
int i,j;
{
    struct record temp;

    bacarecord(fp1,i);
    temp=Datapasien;

    bacarecord(fp1,j);
    Tulisrecord(fp1,i);

    Datapasien=temp;
    Tulisrecord(fp1,j);
}



panjangfile(FILE *fp1)
{
    long panjang;
    fseek(fp1,0,SEEK_END);
    panjang=ftell(fp1);
    return(panjang/BESARDATA);
}

bacarecord(fp1,nomrecord)
FILE *fp1;
int nomrecord;
{
    long posrec=nomrecord*BESARDATA;
    if (fseek(fp1,posrec,SEEK_SET)!=0){
        return SALAH;
    }
    else{
        fread(&Datapasien,BESARDATA,1,fp1);
        return BENAR;
    }
}

Tulisrecord(fp1,nomrecord)
FILE *fp1;
int nomrecord;
{
    long posrec=nomrecord*BESARDATA;
    fseek(fp1,posrec,SEEK_SET);
    fwrite(&Datapasien,BESARDATA,1,fp1);
}
```

```
void SadapKetik(int *brs,int panjang,char*temporer)
{
    int klm=1;
    char pindah='T';
    char ch[5];

    PergiKe(klm,*brs);

    for(klm=1;klm<=panjang;){

        crh.tbl=bioskey(1);      // baca kunci tanpa berhenti
        if (bioskey(1)!=0)
        {

            crh.tbl=bioskey(0);

            if (crh.ch[0]){
                switch(crh.ch[0]){
                    case 13:                                    // ENTER
                        HapusCursor();
                        if(klm==1){
                            switch (*brs){
                                case 2:
                                    strncpy(temporer,Datapasien.Nama,20);
                                    break;
                                case 3:
                                    strncpy(temporer,Datapasien.Umur,4);
                                    break;
                                case 4:
                                    strncpy(temporer,Datapasien.JenisKelamin,13);
                                    break;
                                case 5:
                                    strncpy(temporer,Datapasien.Alamat,30);
                                    break;
                                case 6:
                                    strncpy(temporer,Datapasien.Kota,20);
                                    break;
                            }
                        }
                        klm=1;
                        pindah='Y';
                        *brs=*brs+1;
                        break;
                    case 8:
                        if(klm!=1){
                            klm--;
                            temporer--;
                            BalikKursor(klm,*brs);
                        }
                        break;
                    case 27:            // ESCAPE
                        if(*brs==1){
                            HapusCursor();
                            pindah='Y';
                        }
                        break;
                    default:
                        sprintf(ch,"%c",crh.ch[0]);
                        if ((*brs==1)||(*brs==3)){
                            if (strcmp(ch,"0")<0)break;
                            if (strcmp(ch,"9")>0)break;
                        }
                        Blank(klm,*brs,panjang-klm);
                        setcolor(BLUE);
                        outtext(ch);
                        strcpy(temporer,ch);
                        temporer++;
                        klm++;
                        PergiKe(klm,*brs);
                }
```

MILIK ... TAKAAN
... NOLOGI
SE... EMBER

```
            }


        }
        if (pindah=='Y'){
            break;
        }
    }

    if (pindah=='T'){
        HapusCursor();
        *brs=*brs+1;
    }
}


void EksekusiMenu2(void)
{
    Tenggelam(Maxx/80,Maxy/5.5172,Maxx/4.4138,Maxy/4.3243,LIGHTGRAY,3);

    settextstyle(DEFAULT_FONT,HORIZ_DIR,1);
    settextjustify(LEFT_TEXT,TOP_TEXT);
    setcolor(BLACK);
    outtextxy(Maxx/16.8421,Maxy/5.1613,menu2);

                    // disini panggil prosedur yang diperlukan

    StatusMouse(HILANG);
    Pmenu2();
    StatusMouse(MUNCUL);

    getch();

    settextstyle(DEFAULT_FONT,HORIZ_DIR,1);
    settextjustify(LEFT_TEXT,TOP_TEXT);
    setcolor(BLACK);
    Timbul(Maxx/80,Maxy/5.5172,Maxx/4.4138,Maxy/4.3243,LIGHTGRAY,3);
    setcolor(BLACK);
    outtextxy(Maxx/16,Maxy/5,menu2);
    LogoRohan();
}

void Pmenu2()
{
    Timbul(160,60,625,220,LIGHTGRAY,1);

    settextstyle(DEFAULT_FONT,HORIZ_DIR,2);
    settextjustify(LEFT_TEXT,TOP_TEXT);
    setcolor(BLACK);

    outtextxy(300,70,"DATA PASIEN");
    setcolor(BLUE);
    settextstyle(DEFAULT_FONT,HORIZ_DIR,1);

    outtextxy(180,110 ,"Data Pribadi");
    line(180,120,273,120);
    outtextxy(180,140,"Nomer          :");
    outtextxy(180,160,"Nama           :");
    outtextxy(180,180,"Umur           :");
    outtextxy(385,180,"th.");
    outtextxy(180,200,"Jenis Kelamin  :");
    Tenggelam(340,135,430,150,CYAN,2);                    //nomer
    Timbul(340,155,590,170,CYAN,2);                       //nama
    Timbul(340,175,380,190,CYAN,2);                       //umur
    Timbul(340,195,450,210,CYAN,2);  //jenis kelamin

    AmbilData();

}
void AmbilData(void)
{
    int baris=1;
    int pfield[6]={10,30,4,13,30,20};
```

```
FILE *fp1;
register int a;
int banrec,ketemu=0;
char notemp[10];
char *namafile1=NULL;
char *transf,*ptr_str;
char *banyak;
long int  bny;

if ((namafile1=(char*)calloc(20,sizeof(char)))==NULL){
     outtextxy(430,400,"Memory Penuh");
     getch();
     return;
}
strcpy(namafile1,"pasien.rhn");


if ((transf=(char*)calloc(30,sizeof(char)))==NULL){
     outtextxy(430,400,"Memory Penuh");
     getch();
     return;
}
settextjustify(LEFT_TEXT,BOTTOM_TEXT);
SadapKetik(&baris,pfield[baris-1],transf);
settextjustify(LEFT_TEXT,TOP_TEXT);

strncpy(notemp,transf,10);

if (strtod(transf,&ptr_str)==0){
     free(transf);
     free(namafile1);
     return;
}

if ((fp1=fopen(namafile1,"r+t"))==NULL)
     if ((fp1=fopen(namafile1,"at"))==NULL) return;

banrec=panjangfile(fp1);

if (banrec!=0){
     for (a=0;a<=banrec;a++){
          bacarecord(fp1,a);
          if (stricmp(notemp,Datapasien.Nomer)==0){
               ketemu++;
               settextjustify(LEFT_TEXT,BOTTOM_TEXT);
               PergiKe(1,2);
               outtext(Datapasien.Nama);
               HapusCursor();
               PergiKe(1,3);
               outtext(Datapasien.Umur);
               HapusCursor();
               PergiKe(1,4);
               outtext(Datapasien.JenisKelamin);
               HapusCursor();
               settextjustify(LEFT_TEXT,TOP_TEXT);
               break;

          }
     }
}
if (ketemu==0){
     Timbul(300,300,500,400,RED,2);
     if (banrec==0)
          outtextxy(350,350,"Record Kosong ");
     else
          outtextxy(350,350,"Data Tidak Ada");

     getch();
     fclose(fp1);
     free(namafile1);
     return;
}
```

```
    Timbul(160,222,625,465,LIGHTGRAY,1);

    Tenggelam(340,255,380,270,CYAN,2);
    outtextxy(180,260,"Banyak Pengukuran :");
    baris=baris+5;

    settextjustify(LEFT_TEXT,BOTTOM_TEXT);
    SadapKetik(&baris,4,transf);
    settextjustify(LEFT_TEXT,TOP_TEXT);
    strncpy(banyak,transf,5);

    bny=strtol(banyak,&ptr_str,10);
    if (bny==0)      return;


    BuatBlockdiag(bny,transf);
    strncpy(Datapasien.Oksigen,transf,5);
    Tulisrecord(fp1,a);
    free(transf);
    fclose(fp1);
    free(namafile1);


}
void BuatBlockdiag(int bny,char*oksi)
{
    int i;
    double jarakX;
    char koor[5];

    Timbul(160,222,625,465,LIGHTGRAY,1);
    Tenggelam(190,240,610,445,LIGHTGRAY,1);
    jarakX=420/bny;
    setwritemode(COPY_PUT);
    setcolor(DARKGRAY);
    setlinestyle(DASHED_LINE,0,2);


    for (i=1;i<=bny;i++)   {
        setcolor(DARKGRAY);
        line(190+(jarakX*i),240,190+(jarakX*i),445);
        setcolor(BLACK);

        outtextxy(187+(jarakX*i),450,"^");
    }
    for (i=0;i<=5;i++){
        setcolor(DARKGRAY);
        line(190,240+(i*41),610,240+(41*i));
        itoa(100-(i*20),koor,10);
        outtextxy(190-(strlen(koor)*8),238+(i*41),koor);

    }
    BuatGrafik(bny,oksi);

}
void BuatGrafik(int bny,char*oksi)
{
    int i=1,jarakX,sig=4;
    int koorY,nilY,nilY1;
    double total,rata;



    jarakX=420/bny;
    setwritemode(COPY_PUT);
    setcolor(LIGHTBLUE);
    setlinestyle(SOLID_LINE,0,3);
    moveto(190,445);
    settextjustify(CENTER_TEXT,CENTER_TEXT);
    for (i=1;i<=bny;i++){
        nilY1=AmbilRS();
        if (metu=='Y') {
            metu='T';
            return;
```

```
        }
        total=total+nilY1;
        koorY=445-((275/100)*nilY1);
        setcolor(LIGHTBLUE);
        lineto(190+(jarakX*i),koorY);
    }
    rata=total/bny;
    gcvt(rata,sig,oksi);
    setcolor(BLACK);
    outtextxy(320,460,"Rata-rata = ");
    outtextxy(385,460,oksi);


}



double AmbilRS()

{
    double a=96.12,b=0.19;
    double SaO2;
    int  R795,R665;

    R795=RSambil(1);
    if (metu=='Y')
        return(0);
    else{
        R665=RSambil(2);
    }
    SaO2=a+(b*(R795/R665));

    return (SaO2);

}


int RSambil(int perintah)
{
    int in, out, status, DONE = FALSE;

    bioscom(0, SETTINGS, COM1);
    if (perintah==1)in=0x31;
    else in=0x32;
    bioscom(1,in,COM1);

    while (!DONE)
    {
        bioscom(1,in,COM1);
        status=bioscom(3, 0, COM1);

        if (status & DATA_READY)
            if ((out=bioscom(2, 0, COM1) & 0xFF) != 0)
                DONE=TRUE;
        if (kbhit())
            if (getch()=='\x1B'){
                DONE=TRUE;
                metu='Y';
            }
    }
    return (out);
}



void EksekusiMenu3()
{
    Tenggelam(Maxx/80,Maxy/3.9669,Maxx/4.4138,Maxy/3.3103,LIGHTGRAY,3);


    settextstyle(DEFAULT_FONT,HORIZ_DIR,1);
    settextjustify(LEFT_TEXT,TOP_TEXT);
    setcolor(BLACK);
    outtextxy(Maxx/16.8421,Maxy/3.7795,menu3);
```

```
                    // disini panggil prosedur yang diperlukan

    StatusMouse(HILANG);
    Pmenu3();
    StatusMouse(MUNCUL);
    getch();

    Timbul(Maxx/80,Maxy/3.9669,Maxx/4.4138,Maxy/3.3103,LIGHTGRAY,3);
    setcolor(BLACK);
    outtextxy(Maxx/16,Maxy/3.6923,menu3);
    LogoRohan();
}

void Pmenu3()
{
    Timbul(160,60,625,465,LIGHTGRAY,1);

    settextstyle(DEFAULT_FONT,HORIZ_DIR,2);
    settextjustify(LEFT_TEXT,TOP_TEXT);
    setcolor(BLACK);

    outtextxy(300,70,"DATA PASIEN");
    setcolor(BLUE);
    settextstyle(DEFAULT_FONT,HORIZ_DIR,1);

    outtextxy(180,110 ,"Data Pribadi");
    line(180,120,273,120);
    outtextxy(180,140,"Nomer          :");
    outtextxy(180,160,"Nama           :");
    outtextxy(180,180,"Umur           :");
    outtextxy(385,180,"th.");
    outtextxy(180,200,"Jenis Kelamin  :");
    outtextxy(180,220,"Alamat         :");
    outtextxy(180,240,"Kota           :");

    outtextxy(180,300,"Kadar Oksigen  :");

    Tenggelam(340,135,430,150,CYAN,2);                    //nomer
    Timbul(340,155,590,170,CYAN,2);                       //nama
    Timbul(340,175,380,190,CYAN,2);                       //umur
    Timbul(340,195,450,210,CYAN,2);  //jenis kelamin
    Timbul(340,215,590,230,CYAN,2);  //alamat
    Timbul(340,235,510,250,CYAN,2);  //kota

    Timbul(340,295,385,310,CYAN,2);   //kadar oksigen

    LihatData();

}
void LihatData(void)
{
    int baris=1;
    int pfield[7]={10,30,4,13,30,20,3};
    FILE *fp1;
    register int a;
    int banrec,ketemu=0;
    char notemp[10];
    char *namafile1=NULL;
    char *transf,*ptr_str;

    if ((namafile1=(char*)calloc(20,sizeof(char)))==NULL){
        outtextxy(430,400,"Memory Penuh");
        getch();
        return;
    }
    strcpy(namafile1,"pasien.rhn");


    if ((transf=(char*)calloc(30,sizeof(char)))==NULL){
        outtextxy(430,400,"Memory Penuh");
        getch();
        return;
    }
    settextjustify(LEFT_TEXT,BOTTOM_TEXT);
```

```
    SadapKetik(&baris,pfield[baris-1],transf);
    settextjustify(LEFT_TEXT,TOP_TEXT);

    strncpy(notemp,transf,10);

    if (strtod(transf,&ptr_str)==0){
        free(transf);
        free(namafile1);
        return;
    }

    if ((fp1=fopen(namafile1,"r+t"))==NULL)
        if ((fp1=fopen(namafile1,"at"))==NULL) return;

    banrec=panjangfile(fp1);

    if (banrec!=0){
        for (a=0;a<=banrec;a++){
            bacarecord(fp1,a);
            if (stricmp(notemp,Datapasien.Nomer)==0){
                ketemu++;
                settextjustify(LEFT_TEXT,BOTTOM_TEXT);
                PergiKe(1,2);
                outtext(Datapasien.Nama);
                HapusCursor();
                PergiKe(1,3);
                outtext(Datapasien.Umur);
                HapusCursor();
                PergiKe(1,4);
                outtext(Datapasien.JenisKelamin);
                HapusCursor();
                PergiKe(1,5);
                outtext(Datapasien.Alamat);
                HapusCursor();
                PergiKe(1,6);
                outtext(Datapasien.Kota);
                HapusCursor();
                PergiKe(1,9);
                outtext(Datapasien.Oksigen);
                HapusCursor();
                settextjustify(LEFT_TEXT,TOP_TEXT);
                break;

            }
        }
    }
    if (ketemu==0){
        Timbul(300,300,500,400,RED,2);
        if (banrec==0)
            outtextxy(350,350,"Record Kosong ");
        else
            outtextxy(350,350,"Data Tidak Ada");

        getch();
        fclose(fp1);
        free(namafile1);
        return;
    }

    fclose(fp1);
    free(namafile1);

}


void EksekusiMenu4()
{

    Tenggelam(Maxx/80,Maxy/3.0968,Maxx/4.4138,Maxy/2.6816,LIGHTGRAY,3);

    settextstyle(DEFAULT_FONT,HORIZ_DIR,1);
    settextjustify(LEFT_TEXT,TOP_TEXT);
    setcolor(BLACK);
    outtextxy(Maxx/16.8421,Maxy/2.9814,menu4);
```

```
                    // disini panggil prosedur yang diperlukan

    StatusMouse(MUNCUL);
    getch();
    StatusMouse(HILANG);
    Timbul(Maxx/80,Maxy/3.0968,Maxx/4.4138,Maxy/2.6816,LIGHTGRAY,3);
    setcolor(BLACK);
    outtextxy(Maxx/16,Maxy/2.9091,menu4);
    LogoRohan();
}

void EksekusiMenu5()
{
    BentukCursorMouse(hourglass,1,0);
    StatusMouse(MUNCUL);
    StatusMouse(HILANG);

    MoMinMax(0,0,getmaxx(),getmaxy());
    Tenggelam(Maxx/80,Maxy/2.5397,Maxx/4.4138,Maxy/2.243,LIGHTGRAY,3);

    settextstyle(DEFAULT_FONT,HORIZ_DIR,1);
    settextjustify(LEFT_TEXT,TOP_TEXT);
    setcolor(BLACK);
    outtextxy(Maxx/16.8421,Maxy/2.4615,menu5);

    StatusMouse(MUNCUL);
    delay(1500);
    StatusMouse(HILANG);
    Timbul(Maxx/80,Maxy/2.5397,Maxx/4.4138,Maxy/2.243,LIGHTGRAY,3);
    setcolor(BLACK);
    outtextxy(Maxx/16,Maxy/2.4242,menu5);

    StatusMouse(MUNCUL);

    delay(750);

    cleardevice();          // bersihkan layar grafik
    closegraph();           // tutup layar grafik
    exit(0);
}

void Timbul(int x1,int y1,int x2,int y2,int warna,int tebal)
{
    int i;
    setcolor(BLACK);
    setlinestyle(SOLID_LINE,0,1);
    setfillstyle(SOLID_FILL,BLACK);
    bar(x1,y1,x2,y2);
    setcolor(WHITE);
    setlinestyle(SOLID_LINE,0,tebal);
    setfillstyle(SOLID_FILL,WHITE);
    bar(x1+1,y1+1,x2-1,y2-1);
    setfillstyle(SOLID_FILL,warna);
    bar(x1+tebal+1,y1+tebal+1,x2-(tebal+1),y2-(tebal+1));
    setlinestyle(SOLID_LINE,0,1);
    setfillstyle(SOLID_FILL,DARKGRAY);
    for (i=1;i<=tebal;i++){
        bar(x1+i,y2-i,x2-1,y2-i);
        bar(x2-i,y1+i,x2-i,y2-1);
    }
}

void Tenggelam(int x1,int y1,int x2,int y2,int warna,int tebal)
{
    int i;
    setcolor(BLACK);
    setlinestyle(SOLID_LINE,0,1);
    setfillstyle(SOLID_FILL,BLACK);
    bar(x1,y1,x2,y2);
    setlinestyle(SOLID_LINE,0,tebal);
    setfillstyle(SOLID_FILL,DARKGRAY);
    bar(x1+1,y1+1,x2-1,y2-1);
    setfillstyle(SOLID_FILL,warna);
```

```
    bar(x1+(tebal+1),y1+(tebal+1),x2-(tebal+1),y2-(tebal+1));
    setlinestyle(SOLID_LINE,0,1);
    setfillstyle(SOLID_FILL,WHITE);
    for (i=1;i<=tebal;i++){
        bar(x1+i,y2-i,x2-1,y2-i);
        bar(x2-i,y1+i,x2-i,y2-1);
    }
}

// fungsi untuk memindahkan kursor grafik
void PergiKe(int kolom,int baris)
{
    int LebarKarakter,SpasiBaris,SpasiKolom;
    int TinggiKarakter;
    int x1,y1,x2,y2;

    LebarKarakter=textwidth("W");
    SpasiKolom=LebarKarakter;
    TinggiKarakter=textheight("H");
    SpasiBaris=TinggiKarakter+12;
    moveto((kolom*SpasiKolom)+336,(SpasiBaris*baris)+127);

    // gambar cursor pada posisi bawah karakter
    setfillstyle(SOLID_FILL,GREEN);
    setlinestyle(0,0,3);
    setcolor(BLUE);
    setwritemode(XOR_PUT);
    x1=kolom*SpasiKolom+336;
    y1=baris*SpasiBaris+1+127;
    x2=kolom*SpasiKolom+SpasiKolom+336;
    y2=baris*SpasiBaris+1+127;

    poscur.x1=x1;       // simpan posisi kursor, dimana
    poscur.y1=y1;       // nilai ini akan digunakan
    poscur.x2=x2;       // oleh fungsi hapus kursor
    poscur.y2=y2;

    setcolor(CYAN);
    line (x1,y1,x2,y2);     // gambar kursor

    if (kolom>1)
    {
        x1=(kolom-1)*SpasiKolom+336;
        y1=baris*SpasiBaris+1+127;
        x2=(kolom-1)*SpasiKolom+SpasiKolom+336;
        y2=baris*SpasiBaris+1+127;

        line (x1,y1,x2,y2);;    // hapus kursor sebelumnya
    }
   setcolor(BLUE);
}


// balik kursor
void BalikKursor(int kolom,int baris)
{
    int LebarKarakter,SpasiBaris,SpasiKolom;
    int TinggiKarakter;
    int x1,y1,x2,y2;

    LebarKarakter=textwidth("W");
    SpasiKolom=LebarKarakter;
    TinggiKarakter=textheight("H");
    SpasiBaris=TinggiKarakter+12;
    moveto((kolom*SpasiKolom)+336,(SpasiBaris*baris)+127);

    // gambar cursor pada posisi bawah karakter
    setfillstyle(SOLID_FILL,GREEN);
    setlinestyle(0,0,3);
    setcolor(BLUE);
    setwritemode(XOR_PUT);

    HapusCursor();
    x1=kolom*SpasiKolom+336;
```

```
    y1=baris*SpasiBaris+1+127;
    x2=kolom*SpasiKolom+SpasiKolom+336;
    y2=baris*SpasiBaris+1+127;

    poscur.x1=x1;       // simpan posisi kursor, dimana
    poscur.y1=y1;       // nilai ini akan digunakan
    poscur.x2=x2;       // oleh fungsi hapus kursor
    poscur.y2=y2;

    setfillstyle(SOLID_FILL,CYAN);
    bar(x1,y1-1,x2,y2-TinggiKarakter-1);
    setfillstyle(SOLID_FILL,GREEN);
    setlinestyle(0,0,3);
    setcolor(CYAN);
    setwritemode(XOR_PUT);
    line (x1,y1,x2,y2);     // gambar kursor
}

void Blank(int kolom,int baris,int banyak)
{
    int LebarKarakter,SpasiBaris,SpasiKolom;
    int TinggiKarakter;
    int x1,y1,x2,y2;

    LebarKarakter=textwidth("W");
    SpasiKolom=LebarKarakter;
    TinggiKarakter=textheight("H");
    SpasiBaris=TinggiKarakter+12;

    x1=(kolom)*SpasiKolom+336;
    y1=baris*SpasiBaris+1+127;
    x2=(kolom+banyak)*SpasiKolom+SpasiKolom+336;
    y2=baris*SpasiBaris+1+127;

    setfillstyle(SOLID_FILL,CYAN);
    bar(x1,y1-1,x2-1,y2-TinggiKarakter-1);
}
// hapus kursor
void HapusCursor()
{
    setcolor(CYAN);
    line (poscur.x1,poscur.y1,poscur.x2,poscur.y2);
}
```

Table 3 Correspondence between character codes and character patterns

Upper bit 4 bit / Lower bit 4 bit	0000	0010	0011	0100	0101	0110	0111	1010	1011	1100	1101	1110	1111
××××0000	CG RAM (1)												
××××0001	(2)												
××××0010	(3)												
××××0011	(4)												
××××0100	(5)												
××××0101	(6)												
××××0110	(7)												
××××0111	(8)												
××××1000	(1)												
××××1001	(2)												
××××1010	(3)												
××××1011	(4)												
××××1100	(5)												
××××1101	(6)												
××××1110	(7)												
××××1111	(8)												

2.3 Timing Characteristics

2.3.1 Write timing characteristics

V_{DD} = 5.0 V ± 5%, V_{SS} = 0 V, T_A = 0°C to 50°C

Item		Symbol	Standard		Unit
			Min.	Max.	
Enable cycle time		$t_{CYC}E$	1000	–	ns
Enable pulse width	High level	PW_{EH}	450	–	ns
Enable rise and fall time		t_{Er},t_{Ef}	–	25	ns
Setup time	RS, R/$\overline{W}$ — E	t_{AS}	140	–	ns
Address hold time		t_{AH}	10	–	ns
Data setup time		t_{DSW}	195	–	ns
Data hold time		t_H	10	–	ns

Write operation

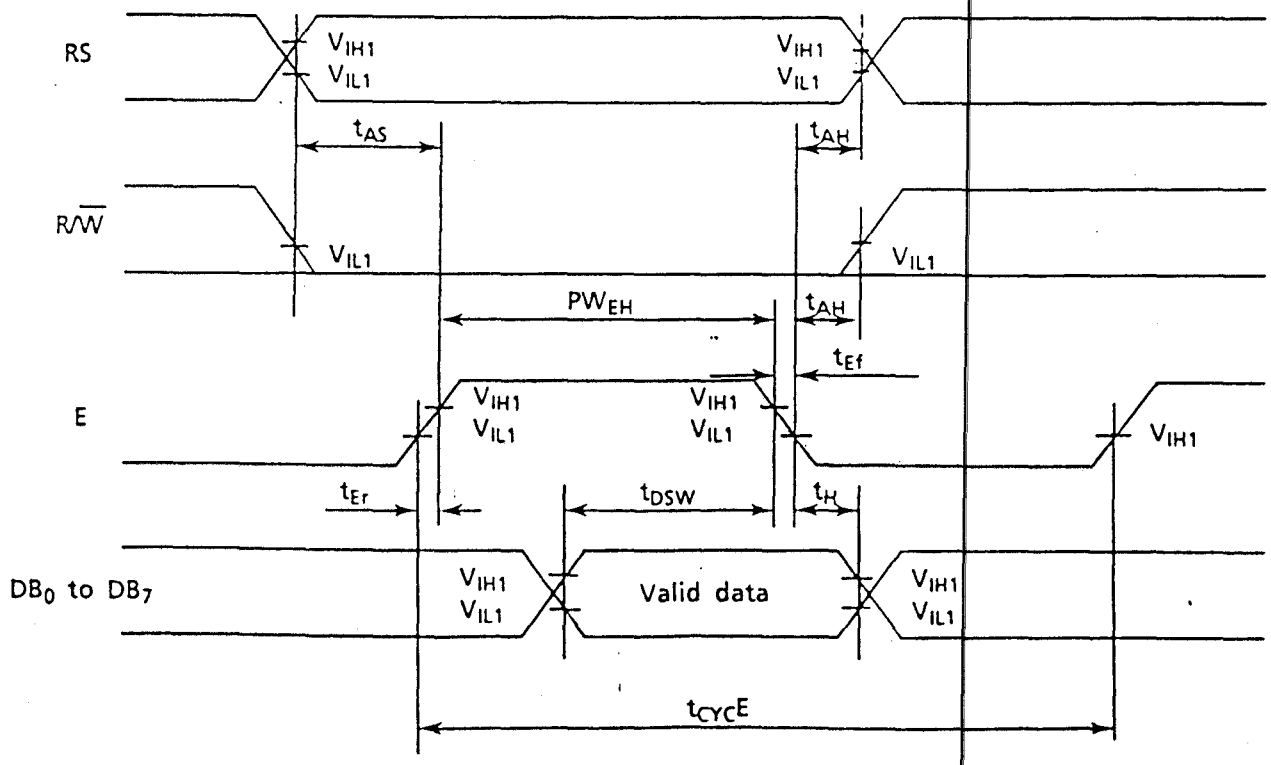


Figure 3 Data write from MPU to module

2.3.2 Read timing characteristics

V_{DD} = 5.0 V ± 5%, V_{SS} = 0 V, T_A = 0°C to 50°C

Item		Symbol	Standard		Unit
			Min.	Max.	
Enable cycle time		$t_{CYC}E$	1000	–	ns
Enable pulse width	High level	PW_{EH}	450	–	ns
Enable rise and fall time		t_{Er},t_{Ef}	–	25	ns
Setup time	RS, R/$\overline{W}$→E	t_{AS}	140	–	ns
Address hold time		t_{AH}	10	–	ns
Data delay time		t_{DDR}	–	320	ns
Data hold time		t_H	20	–	ns

Read operation

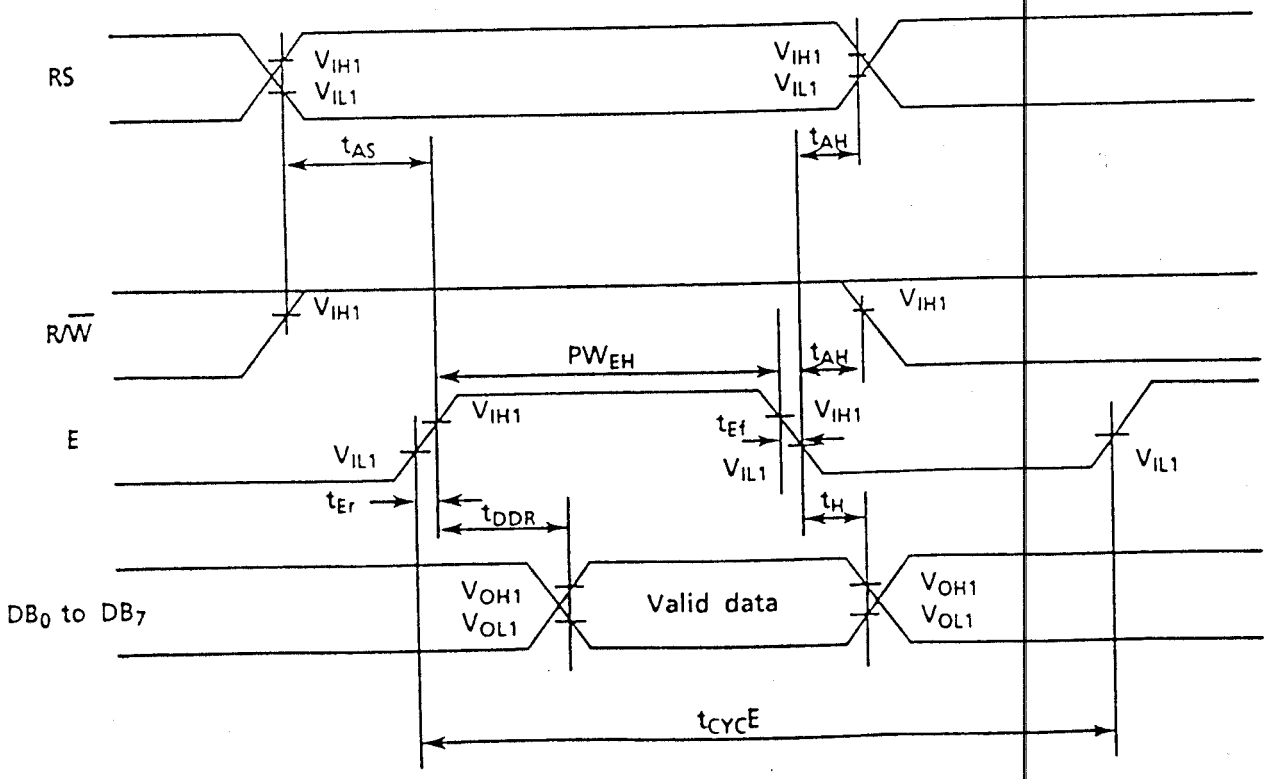


Figure 4 Data read from module to MPU

2.4 Instruction Outline

Table 5 List of instructions

Instruction	Code										Function	Execution time
	RS	R/W	DB_7	DB_6	DB_5	DB_4	DB_3	DB_2	DB_1	DB_0		
(1) Display clear	0	0	0	0	0	0	0	0	0	1	Clears all display and returns cursor to home position (address 0)	1.64 ms
(2) Cursor Home	0	0	0	0	0	0	0	0	1	•	Returns cursor to home position. Shifted display returns to home position and DD RAM contents do not change.	1.64 ms
(3) Entry Mode Set	0	0	0	0	0	0	0	1	I/D	S	Sets direction of cursor movement and whether display will be shifted when data is written or read	40 µs
(4) Display ON / OFF control	0	0	0	0	0	0	1	D	C	B	Turns ON/OFF total display (D) and cursor (C), and makes cursor position column start blinking (B)	40 µs
(5) Cursor/Display Shift	0	0	0	0	0	1	S/C	R/L	•	•	Moves cursor and shifts display without changing DD RAM contents	40 µs
(6) Function Set	0	0	0	0	1	DL	1	•	•	•	Sets interface data length (DL)	40 µs
(7) CG RAM Address Set	0	0	0	1	A_{CG}						Sets CG RAM address to start transmitting or receiving CG RAM data	40 µs
(8) DD RAM Address Set	0	0	1	A_{DD}							Sets DD RAM address to start transmitting or receiving DD RAM data	40 µs
(9) BF/Address Read	0	1	BF	AC							Reads BF indicating module in internal operation and AC contents (used for both CG RAM and DD RAM)	0 µs
(10) Data Write to CG RAM or DD RAM	1	0	Write Data								Writes data into DD RAM or CG RAM	40 µs
(11) Data Read from CG RAM or DD RAM	1	1	Read Data								Reads data from DD RAM or CG RAM	40 µs

• : Invalid bit
A_{CG} : CG RAM address
A_{DD} : DD RAM address

I/D = 1 : Increment
I/D = 0 : Decrement

S = 1 : Display shift
S = 0 : No display shift

D = 1 : Display ON
D = 0 : Display OFF

C = 1 : Cursor ON
C = 0 : Cursor OFF

B = 1 : Blink ON
B = 0 : Blink OFF

S/C = 1 : Display shift
S/C = 0 : Cursor movement

R/L = 1 : Right shift
R/L = 0 : Left shift

DL = 1 : 8 bits
DL = 0 : 4 bits

BF = 1 : Internal operation in progress
BF = 0 : Instruction can be accepted