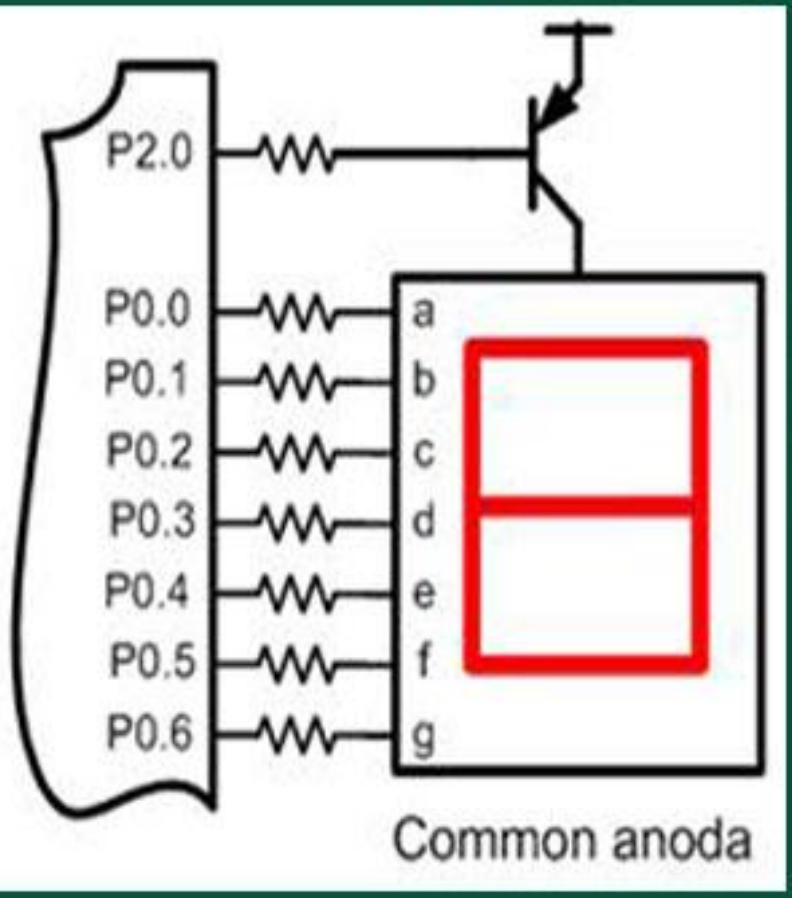
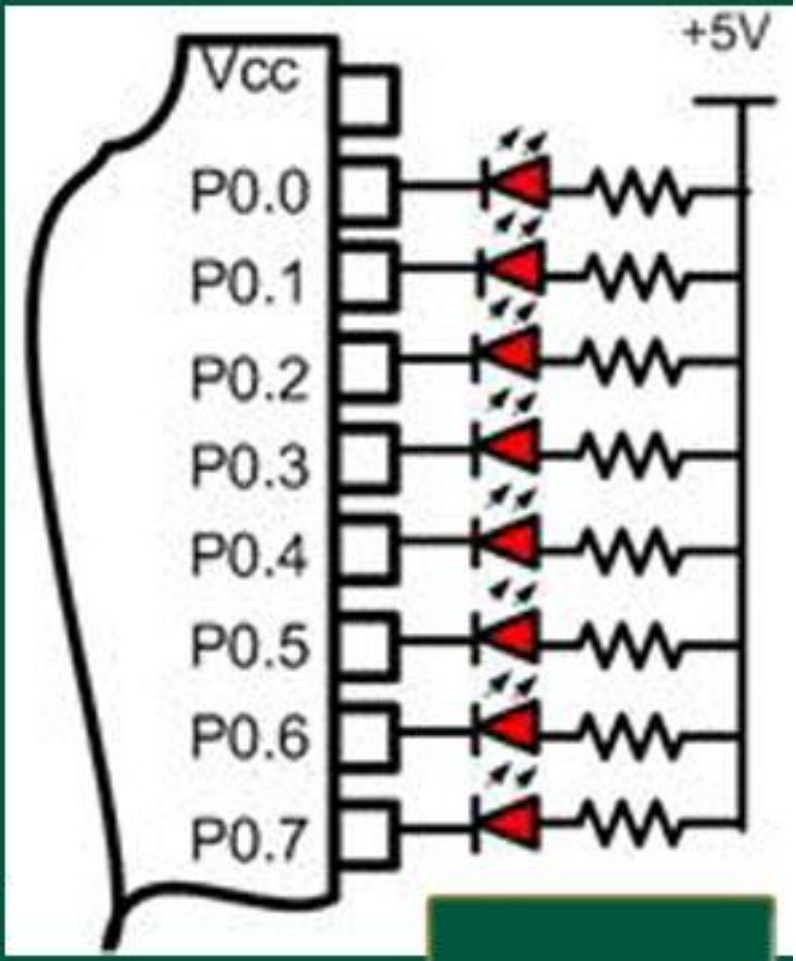
KEMENTERIAN PENDIDIKAN DAN KEBUDAYAAN
REPUBLIK INDONESIA
2013

Sistem Kontrol Terprogram





Semester 4

Kelas XI

PENULIS

KATA PENGANTAR

Puji syukur kehadirat Alloh SWT, dengan tersusunnya buku Teknik Sistem Kontrol Terprogram untuk kelas XII semester 1 ini, semoga dapat menambah khasanah referensi khususnya di bidang tekologi industri yang akhir-akhir ini berkembang begitu pesatnya di Indonesia.

Isi buku ini sengaja disajikan secara praktis dan lengkap sehingga dapat membantu para siswa Sekolah Menengah Kejuruan (SMK), guru serta para praktisi industri. Teknik Sistem Kontrol Terprogram untuk kelas XII semester 1 yang selama ini dideskripsikan secara variatif dan adaptif terhadap perkembangan serta kebutuhan berbagai kalangan praktisi industri. Penekanan dan cakupan bidang yang dibahas dalam buku ini sangat membantu dan berperan sebagai sumbangsih pemikiran dalam mendukung pemecahan permasalahan yang selalu muncul didalam mempelajari dasar-dasar instrumentasi.

Oleh karena itu, buku ini disusun secara integratif antar disiplin ilmu yaitu Teknik Sistem Kontrol Terprogram untuk kelas XII yang saling mendukung sehingga skill yang diperlukan terkait satu dengan lainnya.

Tim penulis mengucapkan terima kasih kepada berbagai pihak yang telah membantu materi naskah serta dorongan semangat dalam penyelesaian buku ini. Kami sangat berharap dan terbuka untuk masukan serta kritik konstruktif dari para pembaca sehingga dimasa datang buku ini lebih sempurna dan implementatif.

Tim Penulis

DAFTAR ISI

Contents

DAFTAR GAMBAR

DAFTAR TABEL

BAB 1

KEGIATAN BELAJAR :
MERANCANG RANGKAIAN KONTROL

Sebelum mempelajari mikrokontroler, kalian harus mengetahui terlebih dahulu tentang perkembangan mikroprosesor sebagai teknologi yang berkembang dengan pesat karena tuntutan keinginan manusia yang serba cepat dan mudah. Jika kalian mengetahui dan paham tentang teknologi mikroprosesor , maka kalian akan sangat mudah untuk mempelajari tentang teknologi mikrokontroler, maka pada pembelajaran awal dikegiatan belajar 2 ini kalian harus dapat membedakan tentang mikroprosesor dan mikrokontroler, bagaimana keuntungan dan penggunanaan keduanya dipakai.

Tugas 1

Coba kalian amati gambar dibawah ini:

Tablet

Laptop

Handphone

Mesin ATM

PLC

Ultrasonography

Mesin Cuci

AC

Rice Cooker

Pemijat Listrik

Parkir Otomatis

Mobil

1. Identifikasi perlatan manakah yang berteknologi mikroprosesor atau mikrokontroler!
2. Bandingkan mikroprosesor dengan mikrokontroler manakah yang lebih banyak kapasitas memorinya?
3. Mengapa pada mikrokontroler ada pengistilahan sistem yang tertanam (*embedded system*) ,mengapa demikian?
4. Sebutkan beberapa alasan suatu sistem kontrol harus menggunakan mikroprosesor atau harus menggunakan mikrokontroler!

Perkembangan Mikroprosesor

Untuk mengenal dan memahami mikroprosesor secara mendalam diperlukan pengetahuan yang memadai dan sumber atau referensi lengkap. Pengetahuan tersebut meliputi sejarah munculnya mikroprosesor, perkembangan mikroprosesor hingga arsitektur dasar komputer dan mikroprosesor.

Istilah mikroprosesor berasal dari *kata microprossesor*, yang secara kasar artinya pemroses mikro atau pengolah mikro. Secara fisisk dapat diartikan sebuah chip atau keping kecil yang merupakan perpaduan suatu rangkaian elektronika yang rumit yang dirancang untuk mengerjakan pekerjaan-pekerjaan yang kompleks.

Gambar 1 Komponen IC Mikroprosesor produksi Intel

Mikroprosesor lebih dikenal dengan sebutan *CPU* atau *Central Processing Unit* adalah sebuah rangkaian terintegrasi (IC) sebagai unit mesin pengolah yang bekerja melakukan fungsi pokok komputasi. Mikroprosesor biasanya dipabrikasi dalam suatu chip tunggal. Bukan mustahil saat ini disaku atau dalam tas kerja atau tas sekolah kalian ada chip mikroprosesor. Telepon genggam (HP), laptop, Tablet atau komputer telapak tangan yang dikenal dengan *Personal Digital Assistance* (PDA) dan sejenisnya pasti menggunakan teknologi mikroprosesor.

Mikroprosessor dapat dikelompokkan menurut teknologi yang dipergunakan, menurut jumlah bit data, menurut struktur atau menurut kemampuan/karakteristik mikroprosessor dan menurut fungsi dari mikroprosessor itu sendiri. Berdasarkan jumlah bit data (*Word Size*) pada waktu ini telah terdapat banyak macam mirkoprosessor, mulai dari mikroprosessor 1 bit, 4bit, 8 bit. 16 bit, 32 bit dan 64 bit.

1.1. Pekembangan Mikroprosesor Intel

1Pekembangan Mikroprosesor Intel

Perkembangan mikroprosesor dari tahun ke tahun mengalami perubahan yang sangat cepat, perkembangan mikroprosesor berdasarkan tahun pembuatan adalah sebagai berikut :

1) Mikroprosesor pertama oleh Intel (1971). Mikroprosesor pertama di dunia adalah Intel 4004 merupakan prosesor 4-bit, Kebanyakan Kalkulator masih berbasis mikroprosesor 4-bit.
2) Tahun 1971 : Intel mengeluarkan mikroprosesor 8-bit yaitu Intel 8008.
3) Tahun 1973 : Intel memperkenalkan mikroprosesor 8-bit modern pertama Intel 8080 (10x lebih cepat dari 8008), dan diikuti Motorola MC6800.
4) Tahun 1977 : Intel memperkenalkan 8085 yang merupakan mikroprosesor 8-bit terakhir yang dibuat Intel dengan frekuensi clock dan kecepatan lebih tinggi.
5) Perusahaan lain yang mampu menyaingi Intel 8085 adalah Zilog Corporation dengan Z80.
6) Tahun 1978 : Intel mengeluarkan mikroprosesor 16-bit yaitu 8086, setahun kemudian mengeluarkan 8088 dengan kecepatan eksekusi dan memori lebih besar dari 8085, serta mulai digunakannya cache memori (sistem antrian yang

mengatur pemberian instruksi sebelum menjalankannya). Intel 8086/8088 disebut juga CISC (Complex Instruction Set Komputer) karena jumlah dan kompleksitas instruksinya.

7) Tahun 1981 : IBM membuat PC menggunakan mikroprosesor 8088 untuk menjalankan aplikasi seperti spreadsheet dan pengolah kata.
8) Tahun 1983 : Intel mengeluarkan mikroprosesor 16-bit 80286, dengan kemampuan memori 16 MB.
9) Tahun. 1986 : Intel mengeluarkan mikroprosesor 32-bit pertama 80386, dengan kemampuan memori 4 GB.
10) Tahun 1989 : Intel mengeluarkan mikroprosesor 32-bit 80486, dengan kemampuan memori 4 GB + 8K Cache.
11) Tahun 1993 : Intel memperkenalkan mikroprosesor 32-bit Pentium I, Th. 1997 Pentium II, kemudian berturut-turut Pentium III dan Pentium 4 pada Th. 2000, dimana mulai digunakan teknologi memori RAMBUS menggantikan teknologi SDRAM.

Tugas 2

Dari data diatas coba kalian buatkan tabel perkembangan mikroprosesor intel sampai dengan saat ini !

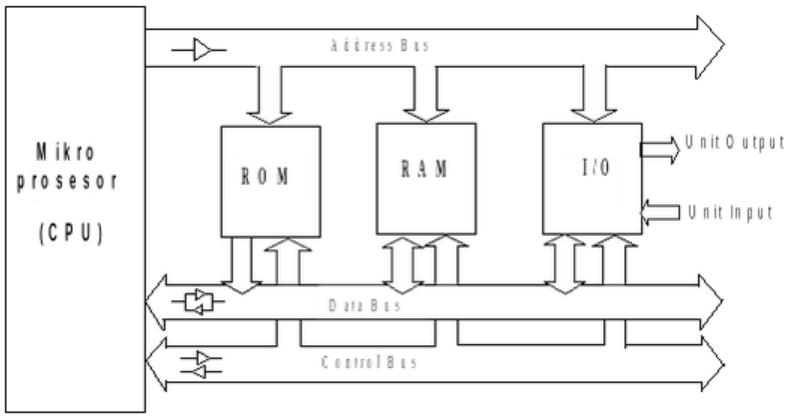
Contoh : Tabel perkembangan mikroprosesor Intel

Nama Prosesor	Tahun	Clock Speed	Lebar Data
8080	1974	5 Mhz	8 bit
8088	1979	5 MHz	16 bit

1.2. Arsitektur Mikrokomputer

2.Arsitektur Mikrokomputer

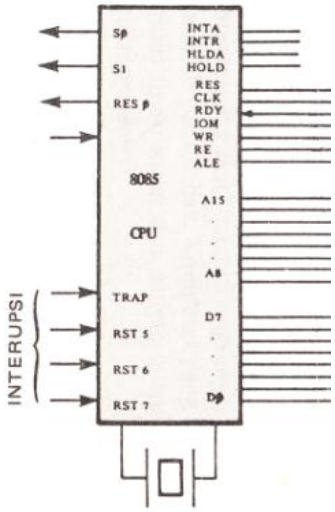
Mikroprosesor tidak dapat berdiri sendiri komponen ini dapat mengolah data, melakukan perhitungan,membutuhkan komponen yang lain, sehingga dinamakan sistem mikroprosesor ataupun mikrokomputer. Adapun arsitektur mikrokomputer dapat dilihat pada gambar dibawah ini:



Gambar 2 Arsitektur mikrokomputer

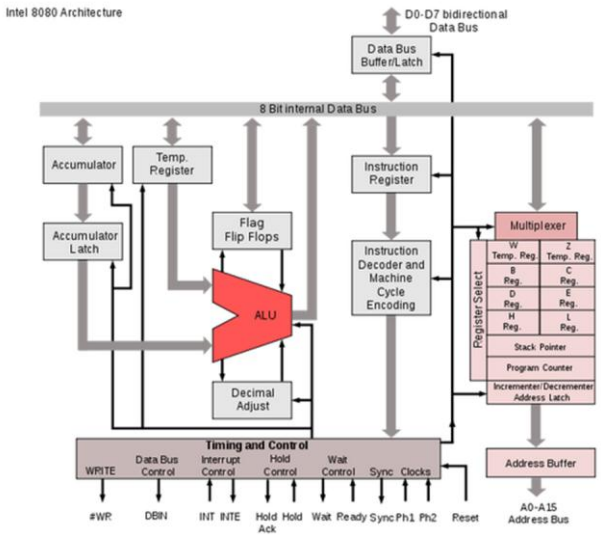
Mikroprosesor/CPU

Mikroprosesor atau CPU adalah "otak" yang merupakan pengendali utama semua operasi dalam sistem komputer. Mikroprosesor mengambil instruksi biner dari memori, menerjemahkannya menjadi serangkaian aksi dan menjalankannya. Aksi tersebut bisa berupa transfer data dari dan ke memori, operasi aritmatika dan logika, atau pembangkitan sinyal kendali. Pada system mikroprosesor terdapat jalur data, jalur alamat dan jalur pengendali *control signal* untuk mengintruksikan ke komponen lain nya. Seperti pada gambar 2.3 tampilan mikroprosesor 8085, memiliki 8 bit artinya jalur data dari D0 sampai dengan D7, memililiki 16 jalur alamat dari A0 sampai dengan A15 dan memiliki jalur pengendali.



Gambar 3 Mikroprosesor 8085

Adapun CPU atau komponen CPU mempunyai 3 komponen internal utama yaitu ALU (Aritmatik Logic Unit), CU (control Unit), dan register. Dapat dilihat pada gambar dibawah ini salah satu gambar arsitektur mikroprosesor intel 8080



Gambar 4 Arsitektur internal mikroprosesor/CPU

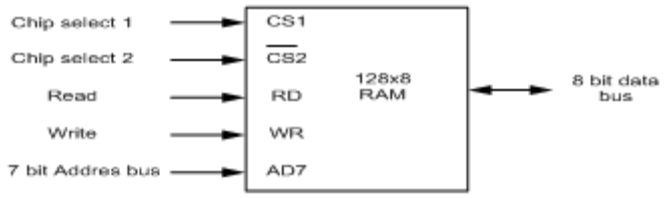
a. ALU (*Arithmetic and Logic Unit*) Unit ini berfungsi untuk menangani operasi-operasi aritmatika (penjumlahan,pengurangan, perkalian, pembagian) dan operasi logika(and,or,exor dan lain-lain).

b. CU (*Control Unit*/ unit kendali) berfungsi mengatur semua proses internal (perpindahan data, penaganan interupsi, pengendalian jalannya program, pengendalian piranti masukan dan keluaran beserta daya (*resourse*) pada mikroprosesor.

c. Register pada dasarnya adalah memori yang dapat diakses dengan sangat cepat oleh mikroprosesor . Operasi-operasi aritmatika dan logika yang kompleks membutuhkan tempat penyimpanan sementara hasil dan tahapan operasi tersebut. Proses pemindahan data dari lokasi memori ke media penyimpanan juga memerlukan tempat penampungan sementara pada register.

Memori

Memori adalah komponen yang digunakan untuk menyimpan instruksi-instruksi biner yang akan dijalankan oleh mikroprosesor, serta data yang digunakan untuk bekerja. Memori yang dapat langsung diakses oleh mikroprosesor, yaitu RAM (*random access memory*) yang dapat dibaca-tulis dan ROM (*read only memory*) yang hanya dapat dibaca saja.
RAM merupakan suatu chip di luar dari mikroprosesar yang berupa memori yang sifatnya sementara dan mempunyai kapasitas tertentu. RAM untuk mikrokomputer aplikasi separti dalam gambar 2.4 dan table fungsi seperti dalam table II-1.

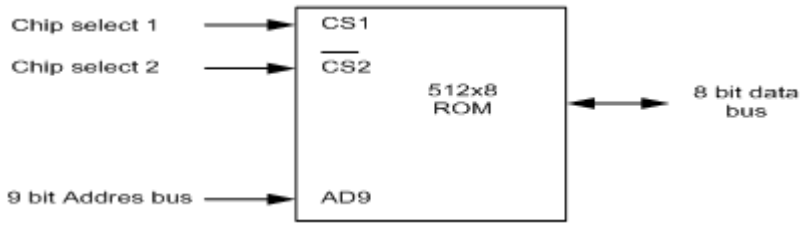


Gambar 5 Blok diagram RAM

Tabel 1 Tabel fungsi

CS1	$\overline{CS2}$	RD	WR	Memory Function	State of data bus
0	0	x	x	Inhibit	High-impendace
0	1	x	x	Inhibit	High-impendace
1	0	0	0	Inhibit	High-impendace
1	0	0	1	Write	Input data to RAM
1	0	1	x	Read	Output data to RAM
1	1	x	x	Inhibit	High-impendace

ROM adalah tempat penyimpanan data yang fungsinya sebagai operasi system kerja untuk mengintruksikan ke mikroprosesor dimana ROM adalah tempat pemnyimpanan data permanen dan mempunyai kapasitas tertentu, diagram blok untuk ROM seperti pada gambar 2.5.



Gambar 6 Blok diagram ROM

Port input/output

Port input/output adalah komponen yang menghubungkan mikroprosesor dengan perangkat luar (harddisk printer, keyboard, monitor, dll.). Jadi port disini berlaku sebagai "pintu" ke perangkat luar. Sebagaimana memori, port I/O juga bukan merupakan komponen tunggal (artinya ada banyak port di dalam sistem komputer) yang masing-masing diberi alamat tertentu. Dengan demikian mikroprosesor tahu, misalnya, ke mana untuk mengirim data ke printer, mengambil data dari mouse dsb.

Bus

Bus adalah kumpulan jalur yang menghubungkan ketiga komponen di atas. Bus dapat dianalogikan sebagai jalan umum di muka rumah kita yang dapat kita lewati jika hendak menuju rumah tetangga, kantor, dsb. Bedanya, di jalan umum pada suatu waktu bisa terdapat banyak orang atau kendaraan yang melewatinya. sedangkan untuk bus, pada suatu saat hanya bisa ada satu keadaan (biner) untuk setiap jalurnya. Dengan kata lain, ada banyak komponen yang terhubung ke bus, tapi hanya sebuah komponen yang akan mengisi bus tersebut pada suatu saat. Bus dalam sistem komputer dibagi menjadi 3 kelompok:

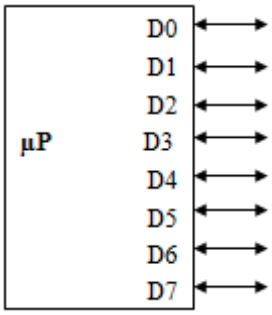
Mari kita simak ketiga bus tersebut bekerja :

a. Bus data membawa informasi ke dan dari unsur prosesor. Bus data membawa instruksi yang dimabil dari memori, masukan data dari alat masukan, data yang disimpan ke dalam memori, dan keluaran data menuju alat keluaran.

   Bus data (d*ata bus*), yang digunakan untuk lewatnya data dari dan ke masing-masing komponen di atas.Bus data mempunyai ukuran tertentu misalnya 8, 16, atau 32 jalur. Ukuran ini tidak harus sama dengan ukuran data pada setiap lokasi memori. Misalnya apabila berukuran memori adalah 8 bit, maka dengan bus data 32 bit akan dapat memindahkan 4

data (menulis/membaca 4 lokasi memori) sekaligus. Lebar bus data biasanya digunakan sebagai klasifikasi mikroprosesor. Misal mikroprosesor 8 bit, 16 bit, atau 32 bit, artinya dia memiliki lebar bus data 8, 16 atau 32.

Bus data menunjukkan kemampuan Mikroprosesor untuk menyalurkan sejumlah data sekaligus. Mikroposesor 8 bit atau 1 byte, secara hardware terdiri dari 8 buah bus ($D7_{MSB}$... $D0_{LSB}$). Besar data yang dapat disalurkan sekaligus mulai $(00000000 \text{ s/d } 11111111)_{(B)}$ atau $(00 \text{ s/d } FF)_{(H)}$.



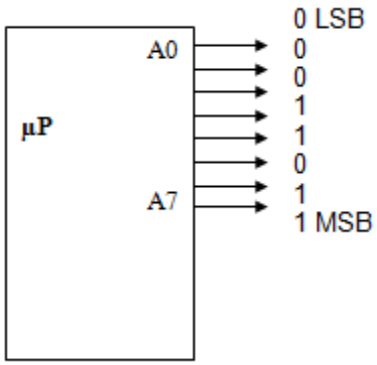
Gambar 7 Bus Data 8 Bit

b. Bus alamat dipakai untuk menetapkan ke mana perginya data atau dari mana datangnya data, Bus ini memilih sebuah lokasi dalam memori atau sebuah register alat maukan – keluaran I/O.

Bus alamat (a*ddress bus*), yang digunakan oleh mikroprosesor untuk mengirim informasi alamat memori atau port I/O yang akan dihubungi olehnya. Ukuran bus alamat menentukan berapa kapasitas memori yang ada, misalnya ukuran bus alamat 16 bit (16 jalur alamat) akan mampu mengalamati 2^{16} atau 65536 (64 kb) lokasi memori. Perhatikan arah panah ke dan dari bus alamat pada Gambar 2.7.

Mikroprosesor ideal akan memiliki memory internal tak terbatas, tetapi kondisi riil yang ada untuk menyimpan data dan program jumlahnya perlu dibatasi. Akibatnya mikroprosesor harus mampu mencapai memori luar (external memory) untuk menyimpan data (read) dan mengambil

kembali (write). Komunikasi antara mikroprosesor dengan external memory dilakukan melalui "bus alamat".



Gambar 8 Bus Alamat 8 bit

c. Bus Pengendali dipakai untuk mengendalikan urutan dan sifat dasar operasi yang sedang berjalan. Bus pengendali khususnya menunjukkkan tipe operasi seperti "baca data dari memori ke prosesor, baca dari alat masukan ke prosesor atau tulis pada alat keluaran dari prosesor, sebagai tambahan, interupsi, akses memori langsung dan fungsi pengendalian lainnya dibawa oleh saluran bus pengendali untuk mengimplementasikan penjadwalan dan penyerempakan kejadian.

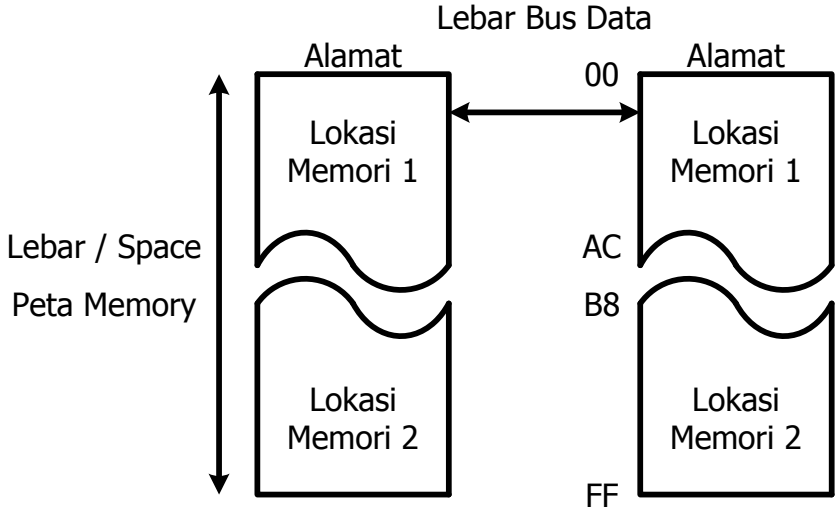
Bus kendali (*control bus*), yang berisi jalur-jalur untuk keperluan pengiriman sinyal kendali antar komponen, misalnya sinyal yang menandakan isyarat untuk membaca, atau menulis, pemilihan memori atau port, interupsi, dll. Isyarat-isyarat ini yang kemudian menentukan aksi apa yang harus dilakukan oleh masing-masing komponen.

Bus pengendali menyediakan empat fungsi yaitu:

a) Menyerempakan memori
b) Penyerempakan masukan-keluaran
c) Penjadwalan CPU/MPU – interupsi dan DMA
d) Pembantu, seperti detak dan memasang kembali/hapus (reset)

Peta Memori

Setiap data informasi disimpan dalam memori pada suatu kumpulan lokasi memori. Ukuran data pada setiap alamat memori ditentukan oleh lebar bus data. Susunan lokasi memori dinamakan peta memory (*space memory*).

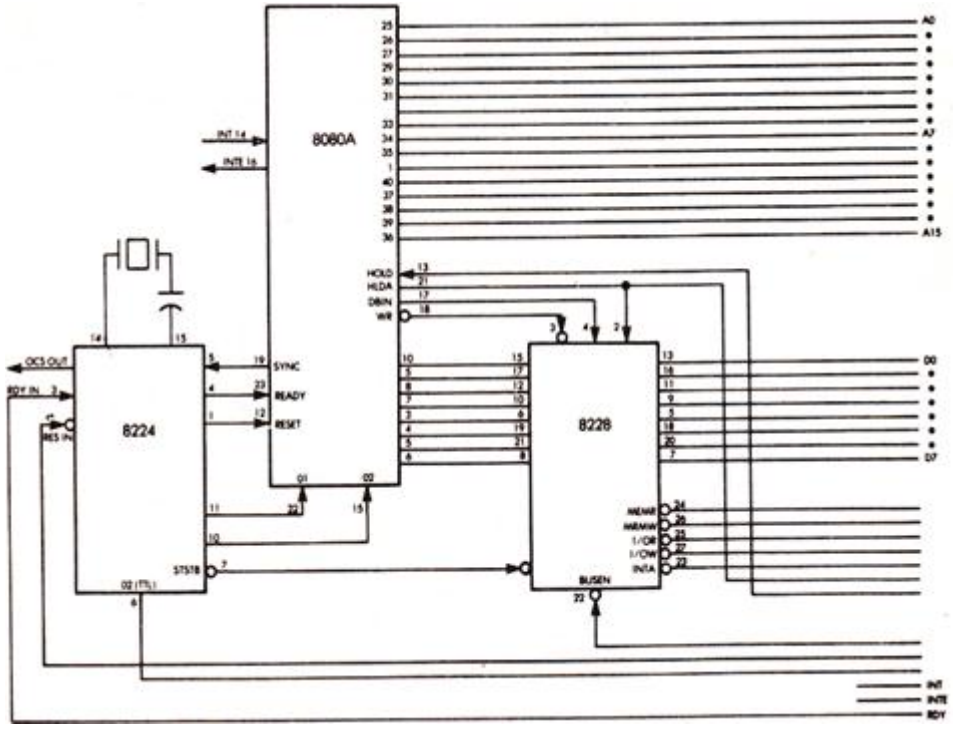


Gambar 9 Peta Memori

Pemahaman tentang peta memori sangat diperlukan, untuk merencanakan tempat menyimpanan data informasi pada saat melakukan perencanaan program.

Gambar 1.9 menunjukkan posisi/arti : lebar bus data, alamat, dan lebar peta memori dalam peta memori. Setiap ruang memori mempunyai satu alamat sendiri-sendiri yang unik, yaitu tidak ada yang kembar/sama (00, AC, B8, FF). Kumpulan alamat ruang memori disebut "lebar peta memori", sehingga jumlah data yang tersimpan dalam peta memori tergantung dari : ***lebar bus data * lebar peta memori***

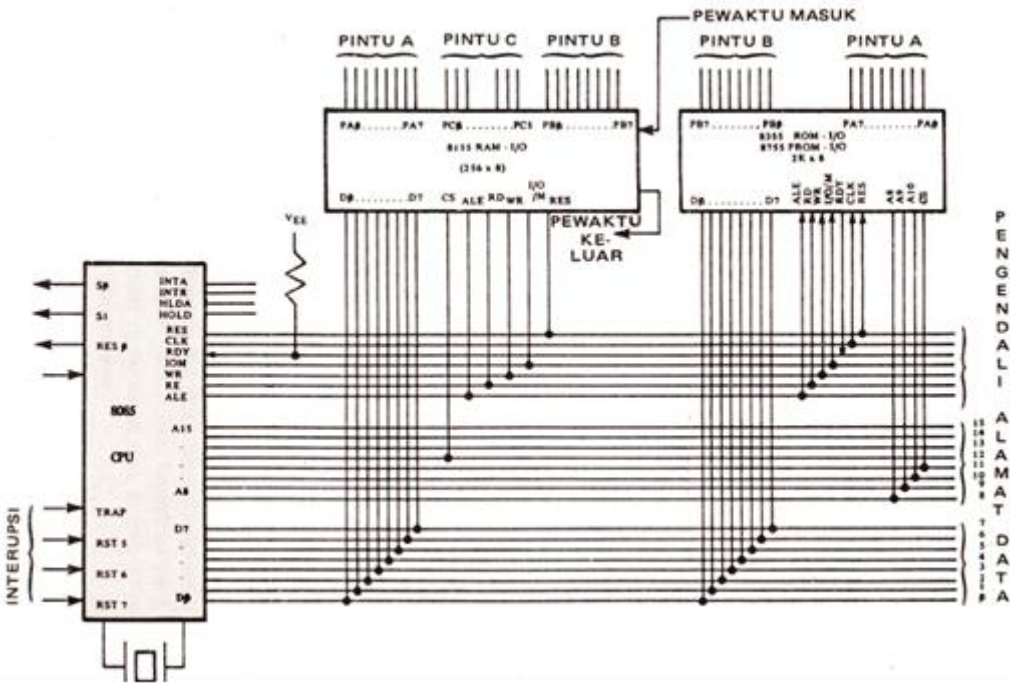
Contoh :
Coba kalian perhatikan gambar CPU 8080 yang dilengkapi komponen pendetak 8224 dan buffer 8228 di bawah ini :



1. Bus data dapat dilihat dari D0 sd D7 berarti 8 bit, artinya mikroprosesor tersebut memiliki lebar bus data 8 bit
2. Bus alamat dapat dilihat dari A0 sampai dengan A15, berarti mikrprosesor tersebut mempunyai kapasitas memori sebesar 2^{15}= 64 K byte.
3. Bus Pengendaliannya adalah INT, HOLD, HLDA, DBIN, WR, SYNC, READY, RESET, yang masing-masing mempunyai fungsi tertentu dalam mengendalikan data yang masuk atau keluar mikroprosesor.

Tugas 3

Amati gambar mikroprosesor 8085 di bawah ini, identifikasi bus data, bus alamat dan bus pengendali (*control*) nya!



Karakteristik yang harus diperhatikan pada mikroprosesor

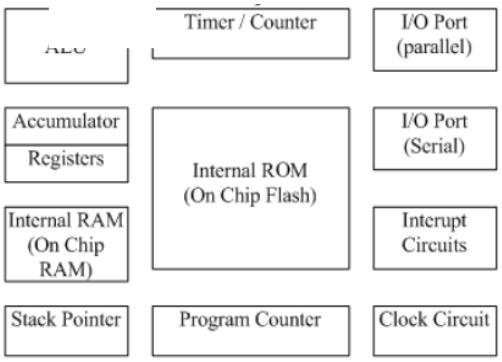
1. Ukuran bus data internal (internal data bus size) jumlah saluran yang terdapat daam mikroposesor yang menyatakan jumlah bit yang dapat ditransfer antar komponen di dalam mikroprosesor.
2. Ukuran bus data eksternal (external data bus size): Jumlah saluran yang digunakan untuk transfer data antar komponen antara mikroprosesor dan komponen-komponen di luar mikroprosesor.
3. Ukuran alamat memori (memory address size): Jumlah alamat memori yang dapat dialamati oleh mikroprosesor secara langsung.
4. Kecepatan clock (clock speed): Rate atau kecepatan clock untuk menuntun kerja mikroprosesor.
5. Fitur-fitur spesial (special features): Fitur khusus untuk mendukung aplikasi tertentu seperti fasilitas pemrosesan floating point, multimedia dan sebagainya

Perkembangan Mikrokontroler

Mikrokontroler dapat diibaratkan sebagai suatu komputer yang berada dalam sebuah chip khusus atau komputer dengan chip tunggal. Kata ' mikro' menyatakan bahwa sebuah peralatan yang kecil dan kata '*Controller* (pengontrol)' menyatakan bahwa alat ini digunakan untuk mengontrol satu atau lebih fungsi obyek, peristiwa atau proses. Ini juga disebut suatu pengontrol yang ditanamkan, oleh karena mikrokontrolers adalah sering ditanamkan di dalam sistem (emmbedded System) atau peralatan yang mereka kontrol.

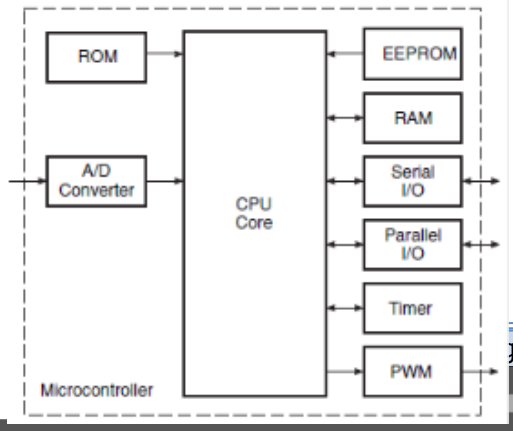
Mikrokontroler berisi suatu prosesor yang disederhanakan, beberapa memori (RAM dan ROM), Port I/O dan peralatan peripheral seperti *counters/timers*, pengubah analog-to-digital dan lain-lain yang semuanya terintegrasi pada sebuah chip tunggal. Ini adalah keunggulan dari mikrokontroler dimana prosesor dan komponen peripheral tersedia dalam sebuah chip yang membedakan dengan system yang berbasis mikroprosesor. Contoh mikrokontroller ATMEL 89c51/ AT 89S51/XX, ATMEGA 8535, ATMEGA 16 PIC, dll

Blok diagram mikrokontroler secara umum dapat dilihat pada gambar di bawah ini:

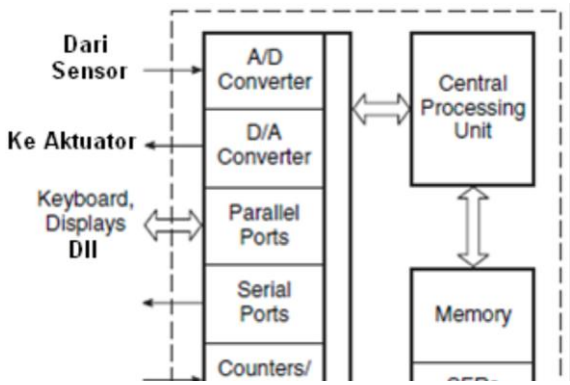


Gambar 10 Blok Diagram mikrokontroler

Sama halnya dengan mikroprosesor,teknologi mikrokontroler berkembang sangat pesat, sesuai denga keinginan dan kebutuhan manusia, dari mulai mikrokontroler yang sederhana sampai dengan yang kompleks sehingga dalam satu chip IC terdapat rangkaian ADC dan DAC. Dengan teknologi yang berkembang menggunakan CISC atau RISC seperti pada arsitektur AVR *(Alf and Vegard RISC).*



Gambar 11 Mikrokontroler dilengkapi fasilitas ADC



Gambar 12 Mikrokontroler dilengkapi fasilitas ADC dan DAC

Beberapa mikrokontroler yang beredar dipasaran merupakan keluaran beberapa pabrik yang sudah terkenal, misal:

- ✓ Intel., contoh: 8031, 89C51 dll
- ✓ Zilog, contoh: Z8, Z8F1680 dll
- ✓ Microchip, contoh: PIC16F84, PIC16F877 dll
- ✓ Motorola, contoh: 68HC11, MC68HC705V12CFN dll
- ✓ Philips Semiconductors, contoh: LPC2000, LPC900, LPC700 dll
- ✓ Atmel, contoh: Atmel AT89 series (Intel 8051/MCS51 architecture), Atmel AT91 series (ARM THUMB architecture), AT90, Tiny & Mega series – AVR.

Tugas 5 :

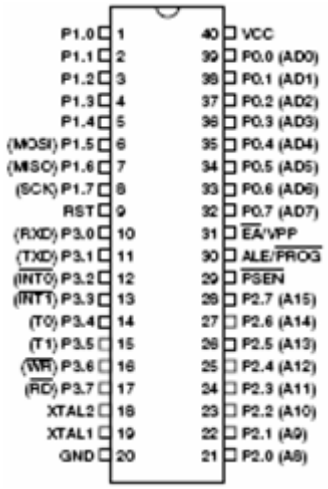
Agar kalian lebih mengenal tentang teknologi dengan arsitektur mikrokontroler CISC (*Complete Instruction Set Komputer*) dan arsitektur AVR *(Alf and Vegard RISC)*, coba kalian bandingkan kelebihan dan

kekurangan dari kedua teknologi tersebut, kemudian kalian klasifikasikan mikrokontroler apa saja yang termasuk CISC dan RISC!

Arsitektur Mikrokontroler

Pada Kegiatan belajar 2, ini kalian akan mempelajari mikrokontroler keluarga ATMEL 8051 , AT 89S51, mengapa demikian karena seri S sudah berteknologi ISP (*In System Program*) yaitu teknologi untuk dapat memasukan program (*download*) secara langsung tanpa harus melepas dan memasang IC lebih berkembang dibanding dengan seri C, tetapi untuk perkembangan jenis RISC seperti AVR ATMEGA, silahkan kalian dapat memperlajarinya setelah pengetahuan tentang AT89S51 dikuasai, karena mikrokontroler semakin berkembang semakin mudah untuk dipelajari.

AT89S51 mempunyai konsumsi daya rendah, mikrokontroller 8-bit CMOS dengan 4K byte memori *Flash ISP* (***In System Programmable***/ dapat diprogram didalam sistem). Pada gambar 2.12 ditunjukkan bentuk fisik dan konfigurasi pin dari sebuah mikrokontroler seri AT89Sxx**.**



Gambar 13 Mikrokontroler

DESKRIPSI PIN :
VCC Tegangan Supply pada pin 40
GND Ground pada pin 20
Port0 pada pin 32 sampai dengan pin 39. Port 0, merupakan port I/O 8-bit *open drain* dua arah. Sebagai sebuah port, setiap pin dapat mengendalikan 8 input TTL. Ketika logika "1" dituliskan ke port 0, maka port dapat digunakan sebagai input dengan *high*

impedansi. Port 0 dapat juga dikonfigurasikan untuk *multipleksing* dengan *address/ data bus* selama mengakses memori program atau data eksternal. Pada mode ini P0 harus mempunyai pull-up.

Port1pada pin 1 sampai dengan pin 8. Port 1 merupakan port I/0 8-bit dua arah dengan *internal pull up*. *Buffer output* port 1 dapat mengendalikan empat TTL input. Ketika logika "1" dituliskan ke port 1, maka port ini akan mendapatkan *internal pull up* dan dapat digunakan sebagai input. Port 1 juga menerima alamat byte rendah selama pemrograman dan verifikasi *Flash*.

Port Pin Fungsi Alternatif sebagai pin ISP, pin 6 sampai dengan pin 8

P1.5 MOSI (digunakan untuk *In System Programming*)
P1.6 MISO (digunakan untuk *In System Programming*)
P1.7 SCK (digunakan untuk *In System Programming*)

Port2 pin 21 sampai dengan pin 28. Port 2 merupakan *port I/O* 8-bit dua arah dengan *internal pull- up*. *Buffer output* port 2 dapat mengendalikan empat TTL input. Ketika logika "1" dituliskan ke port 2, maka port ini akan mendapatkan *internal pull up* dan dapat digunakan sebagai input.

Port3 pin 10 sampai dengan pin 17. Port 3 merupakan port I/O 8-bit dua arah dengan *internal pull up*. *Buffer output* port 3 dapat mengendalikan empat TTL input. Ketika logika "1" dituliskan ke port 3, maka port ini akan mendapatkan *internal pull up* dan dapat digunakan sebagai input. Port 3 juga melayani berbagai macam fitur khusus, sebagaimana yang ditunjukkan pada tabel berikut:

Port Pin	Fungsi Alternatif
P3.0	RXD (port serial input)
P3.1	TXD (port serial output)
P3.2	INT0 (interupsi eksternal 0)
P3.3	INT1 (interupsi eksternal 1)
P3.4	T0 (input eksternal timer 0)
P3.5	T1 (input eksternal timer 1)
P3.6	WR (write strobe memori data eksternal)

P3.7	WR (read strobe memori program eksternal)

RST pada pin 9. *Input Reset*. Logika high "1" pada pin ini untuk dua siklus mesin sementara *oscillator* bekerja maka akan me-reset devais.

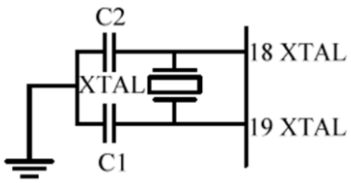
ALE/PROG pada pin 30.*Address Latch Enable (ALE)* merupakan suatu pulsa output untuk mengunci byte low dari alamat selama mengakses memori eksternal. Pin ini juga merupakan input pulsa pemrograman selama pemrograman *flash* (paralel). Pada operasi normal, ALE mengeluarkan suatu laju konstan 1/6 dari frekuensi osiilator dan dapat digunakan untuk pewaktu eksternal.

PSEN pada pin 29.*Program Store Enable* merupakan *strobe read* untuk memori program eksternal.

EA/ VPP pada pin 31. *External Access Enable*. EA harus di hubungkan ke GND untuk enable devais, untuk mengakses memori program eksternal mulai alamat 0000H s/d FFFFH. EA harus dihubungkan ke VCC untuk akses memori program internal. Pin ini juga menerima tegangan pemrogramman (VPP) selama pemrograman Flash. Ingat EA = 0 maka ekternal program, tetapi jika EA= 1 pada pin ini jika kita hanya menggunakan program internal yang ada pada mikrokontroler maka akan digunakan program pada on chip flash

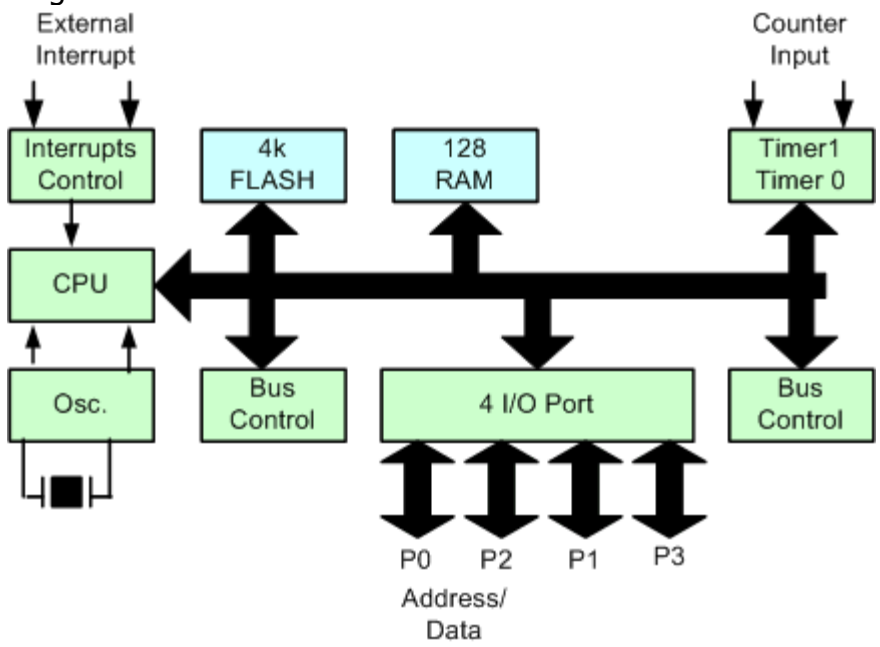
XTAL1 pada pin 19. Input untuk penguat oscilator inverting dan input untuk rangkaian internal clock

XTAL 2pada pin 18. Output dari penguat oscilator inverting.



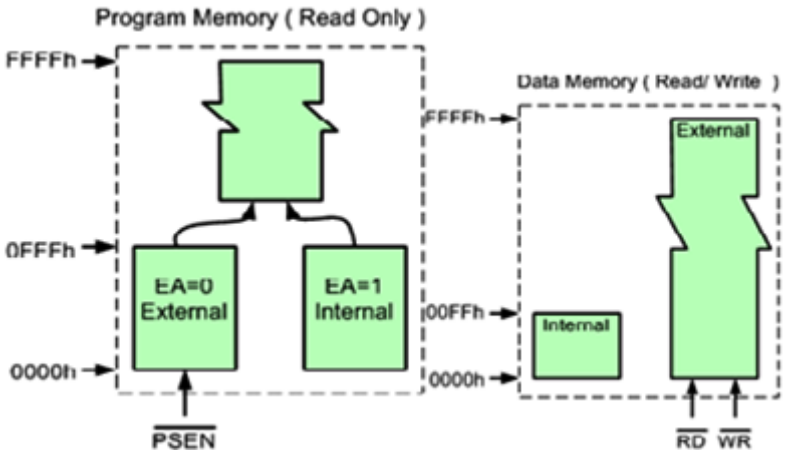
Gambar 14 Pin Osilator pada mikrokontroler

Adapun arsitektur dari mikrokontroler AT89S51 dapat dilihat sebagai berikut :



Gambar 15 Diagram blok mikrokontroler 89S51

Dan arsitektur memori mikrokontroler 8051 dapat dilihat pada gambar



Gambar 16 Arsitektur Memori Mikrokontroller 8051

Tugas 6:

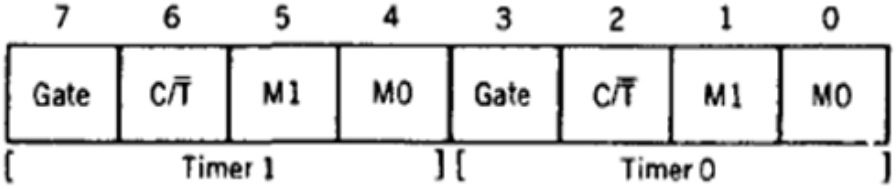
Coba kalian amati blok diagram dari arsitetur mikrokontroler dan arsitektus memori tersebut diatas

1. Identifikasi fungsi masing-masing blok tersebut?
2. Apakah yang dimaksud dengan memori flash dan memori RAM?
3. Pada Semua komponen 8051 mempunyai ruang alamat yang terpisah untuk memori program dan memori data,coba jelaskan apakah yang dimaksud dengan memori program dan memori data?
4. Berapakah kemampuan kapasitas memori program dan memori data tersebut?

Fasilitas Timer dan Counter

Banyak aplikasi mikrokontroler memerlukan penghitungan dari kejadian eksternal, seperti frekuensi dari pulsa atau pembangkitan tunda waktu internal antar komputer. Kedua contoh tersebut dapat dilakukan menggunakan teknik software, namun loop software untuk penghitungan atau pewaktuan mejadikan prosesor terbebani. Oleh karena itu untuk menghidari hal tersebut kita dapat menggunakan fasilitas yang tersedia di mikrokontroler berupa up counter 16-bit yang diberi nama T0 dan T1. Setiap counter mungkin deprogram untuk menghitung pulsa clock internal, beraksi sebagai timer atau diprogram sebagai counter untuk menghitung pulsa eksternal.

Bit	Symbol	Fungsi
0	IT0	sinyal interupsi eksternal jenis bit kontrol. Diatur ke 1 oleh program agar imterupsi eksternal 0 dapat di trigger dengan sinyal falling edge. Jika diatur ke 1 oleh program, jika sinyal low level diberikan pada interupsi eksternal 0 akan membangkitkan interupsi.



Register Fungsi Khusus Timer Mode Control (TMOD)

Bit	Symbol	Fungsi
7/3	Gate	Bit gerbang OR yang digunakan untuk mengontrol RUN/STOP dari timer. Jika diatur ke 1 oleh program berarti membolehkan timer utnuk bekerja jika bit TR1/0 pada TCON diatur ke 1 dan sinyal pada interupsi pin $\overline{INT1}/0$ adalah HIGH. Jika diatur ke 0 oleh program membolehkan timer untuk bekerja jika TR1/0 pada TCON diatur ke 1
6/2	C/$\overline{T}$	Jika diatur ke 1 oleh program membuat timer 1/0 bertindak sebagai counter dengan menghitung pulsa dari pin eksternal 3.5 (T1) atau 3.4 (T0). Jika diatur ke 0 oleh program maka membuat timer bertindak sebagai sebuah timer dengan mengitunh frekuensi internal
5/1	M1	Moda operasi timer/counter memilih bit 1. Diatur 1/0 oleh program untuk memilih moda.
4/0	M0	Moda operasi timer/counter memilih bit 0. Diatur 1/0 oleh program untuk memilih moda.

M1	M0	Moda
0	0	0
0	1	1
1	0	2
1	1	3

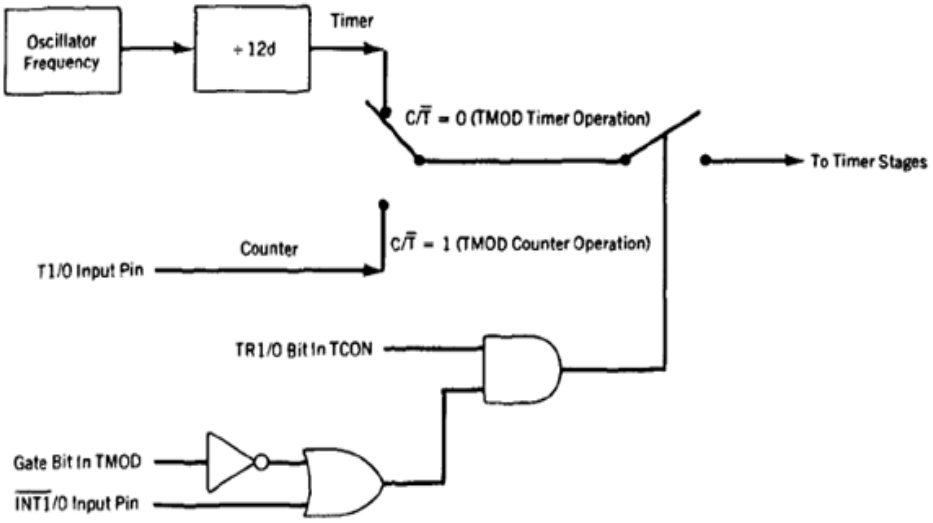
Counter dibagi menjadi dua register 8-bit yang disebut dengan timer low (TL0, TL1) dan timer high (TH0, TH1). Semua aksi counter dikontrol oleh rgister (TCON) dan instruksi program tertentu.

TMOD didedikasikan hanya untuk dua timer dan dapat dipertimbangkan menjadi dua duplikat register 4-bit, masing-masing mengontrol aksi dari satu timer. TCON mempunyai bit kontrol dan flag untuk timer pada upper nibble dan bit kontrol dan flag untuk interupsi eksternal pada lower nibble. Gamba 2.10 menunjukan bit yang ditugaskan untuk TMOD dan TCON.

Interupsi Timer Counter

Counter telah dimasukan ke dalam chip, sehigga processor dapat melakukan kerja penghitungan dan pewaktuan. Ketika program

menginginkan untuk menghitung jumlah tertentu dari pulsa internal atau kejadian eksternal, sebuah angka ditempatkan pada counter. Counter bertambah dari angka awal ke maksimum dan kemudian kembali ke nol pada pulsa akhir dan juga mengatur timer flag. Kondisi flag diuji oleh sebuah nstruksi untuk memberitahu program bahwa hitungan telah dilakukan atau flag digunakan untuk meninterupsi program.



Gambar 17 Logika Kontrol Timer/Counter

Serial Data Input/Output

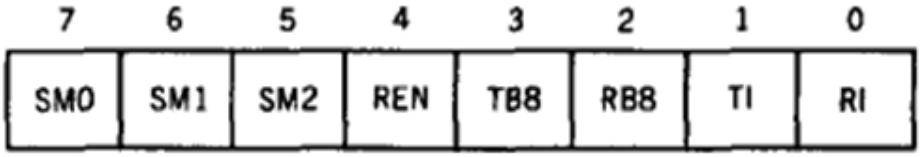
Konputer harus dapat berkomunikasi dengan komputer yang lain. Salah satu cara yang murah untuk berkomunikasi adalah mengirimkan dan menerima bit data serial. 8051 mempunyai rangkaian komunikasi data serial yang menggunakan register SBUF untuk mempertahankan data. Register SCON mengontrol komunikasi data, register PCON mengontrol rate data. Dan pin RDX (P3.0) dan TDX (P3.1) menghubungkan ke jaringan data serial.

SBUF secara fisik terdiri dari dua register. Satu adalah hanya menulis dan digunakan untuk mempertahankan data yang akan dikirimkan 8051 melalui TDX. Yang lain adalah hanya membaca dan mempertahankan data yang diterima dari sumber eksternal melalui RDX. Keduanya saling tidak terpisah menggunakan alamat 99h.

Terdapat empat moda programmable untuk komunikasi data serial yang dipilih oleh pengaturan bit SMX pada SCON. Baud rate ditentukan oleh moda yang dipilih. Gambar 1.17 menunjukan tugas bit untuk SCON dan PCON.

Interupsi Data Serial

Komunikasi data serial adalah proses komunikasi yang relative lambat, mengambil beberapa mdetik setiap byte data untuk melakukannya.



Gambar 18 Register Fungsi SCON dan PCON

Register Fungsi Khusus Serial Port Control (SCON)

Bit	Symbol	Fungsi
7	SM0	Moda port serial bit 0. Diatur ke 1 atau 0 oleh program untuk memilih moda
6	SM1	Moda port serial bit 1. Diatur ke 1 atau 0 oleh program untuk memilih moda
5	SM2	Bit komunikasi multi processor. Diatur ke 1 atau 0 oleh program untuk membolehkan multiprocessor untuk berkomunikasi pada mode 2 dan 3. Ketika diatur ke 1 sebuah interupsi dibangkitkan jika bit 9 dari data yang diterima adalah 1; tidak ada interupsi yang dibangkitkan jika bit 9 adalah 0. Jika diatur ke 1 untuk mode 1, tidak ada interupsi yang dibangkitkan kecuali bit stop yang valid diterima. Diatur ke 0 jika mode 0 digunakan
4	REN	Bit receive enable. Diatur ke 1 untuk membolehkan penerimaan. Diatur ke 0 untuk tidak membolehkan penerimaan
3	TB8	Transmitted bit 8. Diatur ke 1 atau 0 oleh program pada mode 2 dan 3

2	RB8	Receive bit 8. Bit 8 dari data yang diterima dari mode 2 dan 3; bit stop pada mode 1. Tidak digunakan pada mode 0
1	T1	Transmit interrupt flag. Diatur ke 1 pada akhir dari 7 pada mode 0, dan pada awal dari bit stop untuk mode yang lain. Harus diatur ke 0 oleh program.
0	R1	Receive interrupt flag. Diatur ke 1 pada akhir dari 7 pada mode 0, dan pada setengah jalan dari bit stop untuk mode yang lain. Harus diatur ke 0 oleh program.

SMO	SM1	MODE	Keterangan
0	0	0	Shift register; baud =f/12
0	1	1	8-bit UART; baud = variable
1	0	2	9-bit UART; baud = f/32 atau f/64
1	1	3	9-bit UART; baud = variable

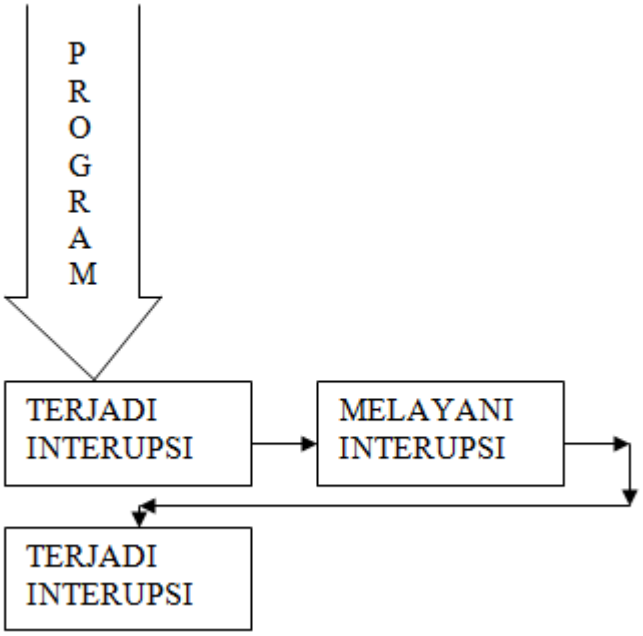
Interupsi

Interupsi adalah kejadian atau peristiwa yang menyebabkab mikrokontroler berhenti sejenak untuk melayani interupsi tersebut. Program yang dijalankan pada saat melayani interupsi di sebut *Interrupt Service Routine* (Rutin Layanan Interupsi).

Analoginya adalah sebagai berikut: seseorang sedang mengetik laporan ,mendadak telepon berdering dan menginterupsi orang tersebut sehingga menghentikan pekerjaan mengetik lalu mengangkat telepon setelah pembicaraan di telepon selesai, maka dia melanjutkan pengerjaan mengetik kembali.

Demikian pula pada sistem mikrokontroler yang sedang menjalankan programmya, saat terjadi interupsi, program berhenti sesaat, melayani interupsi tersebut dengan menjalankan program yang berada pada alamat yang ditunjuk oleh vector dari interupsi yang terjadi hingga selesai dan kembali meneruskan program yang terhenti oleh interupsi tadi. Seperti yang terlihat pada gambar 1.18.

Sebuah program yang seharusnya berjalan terus lurus, tiba-tiba terjadi interupsi dan harus melayani interupsi tersebut terlebih dahulu hingga selesai sebelum ia kembali meneruskan pekerjaannya.



Gambar 19 Alur kerja dari interupsi

Hampir sebagian besar aplikasi mikrokontroler melibatkan respon suatu kejadian yang cukup cepat untuk mengontrol lingkungan (umumnya disebut *real-time programming*). Interupsi merupakan satu-satunya cara dalam real-time programming yang bisa melakukan pengaturan dengan baik.

Lima jenis interupsi disediakan pada 8051. Tiga jenis dihasilkan secara otomatis dari operasi internal. Timer flag 0, timer flag 1, dan interrupt port serial (RI atau TI). Dua jenis interupsi dipicu oleh sinyal luar yang dihasilkan oleh rangkaian yang dihubungkan pada pin INT0 dan INT1 (pin port P3.2 dan P3.3)
Semua fungsi interupsi dibawah kontrol program. Pemrogram bisa merubah kontrol bit pada register Interrupt Enable (IE), register Interrupt Priority (IP) dan register Timer Control (TCON). Program bisa memblok semua atau kombinasi interupsi yang manapun dari aksi program dengan penyetingan dan atau peng-clear-an bit bit yang tepat pada register-register ini. Register IE dan IP ditunjukkan pada Gambar.

Setelah interupsi ditangani oleh rutin interupsi yang ditempatkan oleh pemrogram di lokasi interupsi pada program memori, program yang diinterupsi harus memulai operasi saat interupsi- interupsi

diberikan. Program dimulai kembali dengan menyimpan alamat PC yang di interupsi pada tumpukan (stack) di RAM sebelum merubah PC ke alamat interupsi dalam ROM. Alamat PC akan diambil dari stack sesudah interupsi RET1 dieksekusi pada akhir dari rutin interupsi.

7	6	5	4	3	2	1	0
EA	----	ET2	ES	ET1	EX1	ET0	EX0

Gambar 20 Register Fungsi IE dan IP

Interupsi Enable (IE)

Special Functon Register Interrupt Enable (IE)

Bit	Symbol	Fungsi
7	EA	Bit Enable Interrupt. Jika bit ini diatur ke 0 oleh program maka semua interupsi tidak dapat dilakukan; jika diatur ke 1 mengizinkan melakukan interupsi individu yang dibolehkan oleh enable bit mereka.
6	-	Tidak dipakai
5	ET2	Dicadangkan untuk penggunaan ke depan
4	ES	Enable serial port interrupt. Jika diatur ke 1 oleh program untuk membolehkan serial port interrupt; jika diatur ke 0 serial port interrupt tidak aktif
3	ET1	Enable timer 1 overflow interrupt. Jika diatur ke 1 oleh program membolehkan timer 1 overflow interrupt; jika diatur ke 0 tidak membolehkan overflow interrupt.
2	EX1	Enable external interrupt 1. Jika diatur ke 1 oleh program membolehkan interupsi $\overline{\text{INT1}}$; Dan jika diatur ke 0 tidak membolehkan interupsi $\overline{\text{INT1}}$
1	ET0	Enable timer 0 overflow interrupt. Jika diatur ke 1

Bit	Symbol	Fungsi
		oleh program membolehkan timer 0 overflow interrupt; jika diatur ke 0 tidak membolehkan overflow interrupt.
0	EX0	Enable external interrupt 0. Jika diatur ke 1 oleh program membolehkan interupsi $\overline{\text{INT0}}$; dan jika diatur ke 0 tidak membolehkan $\overline{\text{INT0}}$

Vector Interupsi

Vektor Interupsi adalah nilai yang tersimpan ke Program Counter pada saat terjadi interupsi sehingga program akan menuju ke alamt yang ditunjukkan oleh Program Counter. Pada saat program menuju ke alamt yang tunjukkan interrupt Vector , flag-flag yang set karena terjadinya interrupt akan di clearkan kecuali RI dan TI.

Kelima interupsi dan system reset dari 89S51 mempunyai vector tertera pada table di bawah ini:

INTERUPSI	FLAG	ALAMAT VEKTOR
System reset	RST	0000h
Interupsi Ekternal 0	IE0	0003h
Interupsi Timer 0	IT0	000Bh
Interupsi Ekternal 1	IE1	0013h
Interupsi Timer 1	IT1	001Bh
Interupsi Serial	RI atau TI	0023h

Piranti Keras Mikrokontroler dan antarmuka (*interfacing*) untuk masukan dan keluaran (I/O)

Menghubungkan port parallel dengan LED

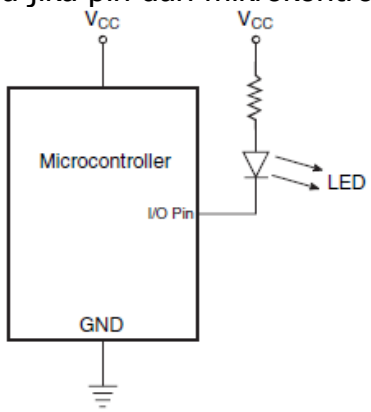
Konfigurasi secara umum untuk menghubungkan sebuah LED ke sebuah mikrokontroler adalah seperti ditunjukan pada gambar 1.21 (a) LED menyala ketika pin mikrokontroler diberi sinya LOW dan akan mati ketika pada pin mikrokontroler adalah HIGH. Tahanan yang terpasang pada angkaian digunakan untuk membatasi arus yang mengalir melalui LED.

Nilai tahanan ditentukan berdasarkan persamaan

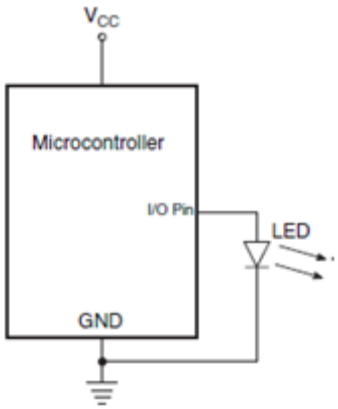
$$R = (VCC - VLED)/I$$

Dimana VLED adalah tegangan di LED dan I arus yang mengalir melalui LED.

Pada umumnya VLED dan I adalah 1,5 V dan 20 mA. Jika kemampuan sumber arus mikrokontroler cukup untuk mengendalikan secara langsung LED, maka LED dihubungkan ke mikrokontroler seperti ditunjukan pada gambar 1.21 (b). LED dalam hal ini akan menyala jika pin dari mikrokontroler adalah HIGH.



Gambar 21 (a) Aktif Low

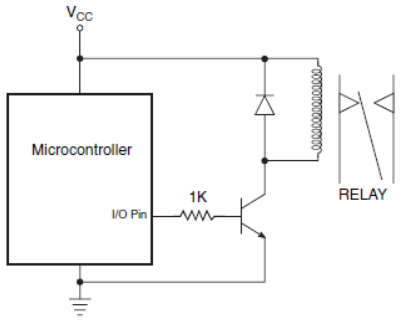


Gambar 22 (b) Aktif High

Antarmuka dengan Rele

Gambar 1.22 menunjukan diagram hubungan untuk antarmuka rele dengan mikrokontroler. Transistor NPN digunakan untuk menyediakan arus yang diinginkan ke kumparan rele karena mikrokontroler tidak dapat secara langsung dihubungkan ke pin mikrokontroler. Ketika pin mikrokontroler di set HIGH transistor akan ON, kemudian arus mengalir melalui kumparan dan kontak

akan tertutup. Ketika pin mikrokontroler adalah LOW, transistor OFF dan arus induktor sekarang mengalir melalui dioda akan turun menjadi nol dan kontak akan terbuka.



Gambar 23 Antarmuka rele dengan mikrokontroler

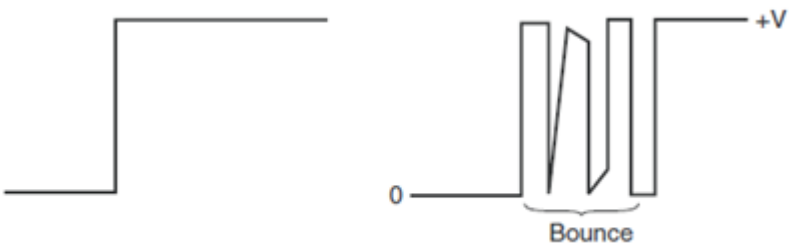
Antarmuka Keyboard

Keyboard digunakan untuk memasukkan data, nilai dan lain-lain ke dalam system mikrokontroler. Keyboard umumnya terdapat dalam tiga konfigurasi yaitu: keyboard linier, keyboard matrix dan keyboard yang dikodekan. Keyboard linier digunakan ketika jumlah masukan yang akan diberikan sedikit. Keyboard yang secara umum digunakan adalah keyboard matrix.

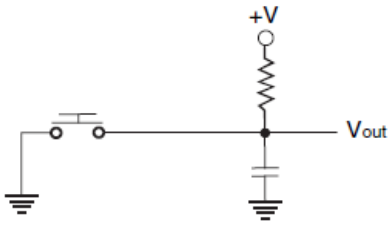
Ketika keyboard dihubungkan ke mikrokontroler, faktor-faktor berikut harus dipertimbangkan,

1. Kontak Bounce (pantulan)

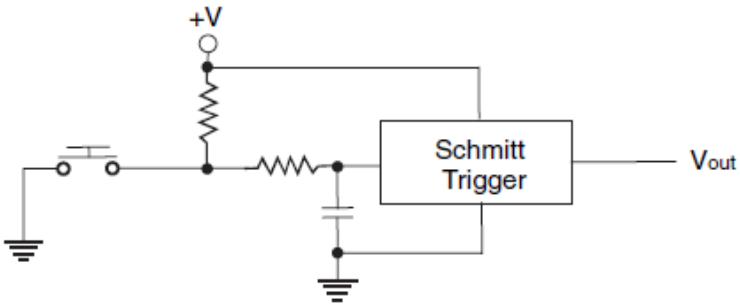
Kontak bounce mengacu pada osilasi dari kontak ketika tombol ditekan (gambar. 1.23). keyboard yang berkualitas mempunyai perioda bounce 1 – 5 mdetik.Jjika bounce diabaikan mikrokontroler merespon tombol ditekan dan dilepas beberapakali padahal kenyataannya hanya ditekan sekali. Kontak dbouncing melalui hardware atau software dilakukan untuk meghindari pengaruh beberapa kali kontakyang tidak diinginkan ketika kontak menutup, dengan demikian ini akan kita peroleh hanya beroperasi tunggal ON dan OFF. Hardware debouncing dilakukan menggunakan rangkaian RC (gambar 1.24 (a) atau sebuah rangkaian Schmitt trigger (gambar 1.24 (b). jika debouncing dilakukan dengan menggunakan software, tunda waktu yang diberikan adalah 20 -50 mdetik etelah tombol ditekan sebeum program dieksekusi.



Gambar 24 **Bounce kontak**



Gambar 25 Hardware debouncing dengan menggunakan rangkaian RC



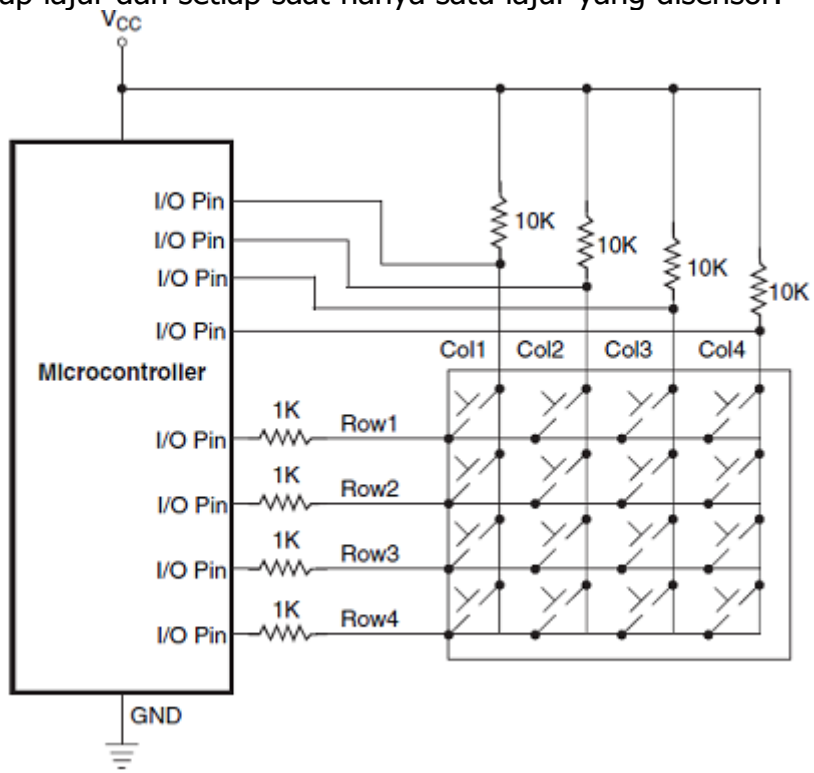
Gambar 26 Hardware debouncing dengan menggunakan Schmitt trigger

2. Multiple key. Jika lebih dari satu tombol ditekan, maka hanya tombol yang valid saja yang akan dieksekusi.

3. Key hold. Terdapat dua jenis aktuasi keyboard yaitu: two-key lock-out dan N-key rollover. Two-key lock-out hanya memperhitungkan satu tombol saja yang ditekan. Sedangkan N-

key rollover akan mengabaikan semua tombol yang ditekan sampai hanya tinggal satu saja yang ditekan.

Gambar 1.25 menunjukan hubungan keypad matrix-16 dengan mikrokontroler.Di sini, masing-masing kolom dan lajur dihubungkan ke pin mikrokontroler. Kolom biasanya pada level HIGH. Lajur dikonfigurasi sebagai output dan kolom digunakan sebagai garis scan. Tombol aktuasi disensor dengan mengirimkan sinyal LOW ke setiap lajur dan setiap saat hanya satu lajur yang disensor.



Gambar 27 Hubungan keypad matrix 16-tombol dengan mikrokontroler

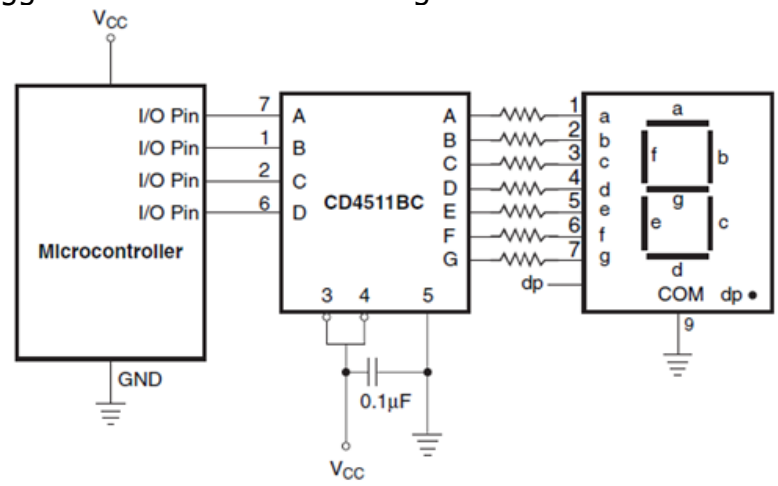
Antarmuka Seven-Segment

Seven-segment biasanya berisi LED yang disusun dengan pola angka delapan. Gambar 1.26 menunjukan konfigurasi dari antarmnuka 7-segment common cathode (CC) dengan mikrokontroler.IC CD4511 adalah IC decoder BCD ke 7-segment. Mikrokontroler memberikan kode BCD yang ekuivalen dengan digit yang akan ditampikkan 7-segment melalui ke IC 4511.

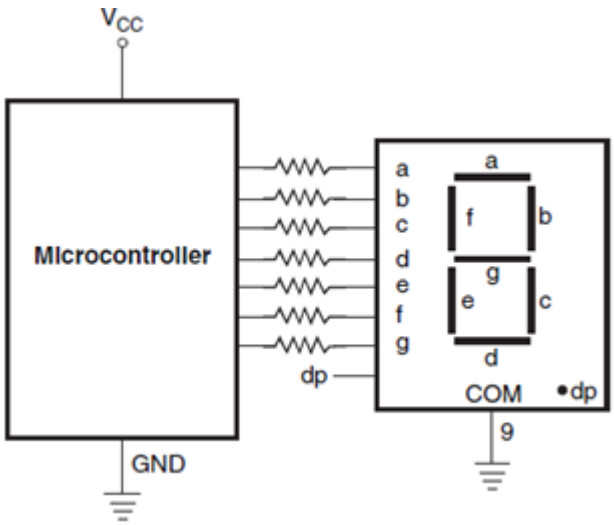
Display Seven-Segment dapat juga dihubungkan secara langsung tanpa menggunakan decoder BCD ke seven-segment. Dalam hal ini kode seven-segment dari digit dihasilkan oleh program

mikrokontroler itu sendiri. Gambar 2.21 menunjukan hubungan secara langsung untuk display common anode.

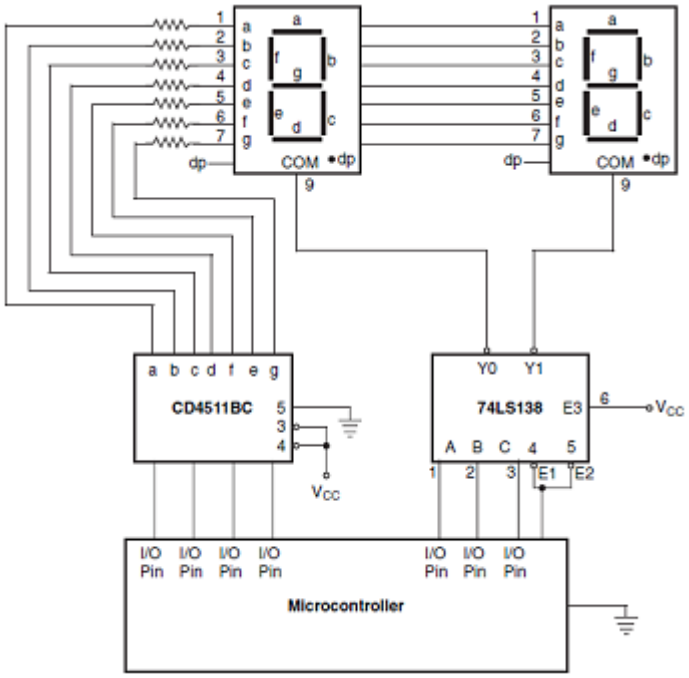
Jika akan digunakan lebih dari satu display, maka display di-multiplex-kan. Mata manusia tidak dapat menditeksi kedip display jika display berkedip setiap 10 mdetik. Waktu 10 mdetik dibagi dengan jumlah display yang digunakan untuk mendapatkan interval waktu antara perbaharuan setiap display. Gambar 1.27 menunujukan rangkaian multiplex untuk dua display common cathode. IC 7438 adalah garis 3 sampai 8 dari decoder digunakan untuk menyeleksi display. Gambar 1.28 menunjukan multiplex dari display common anode untuk hubungan langsung tanpa menggunakan driver BCD ke 7-segment.



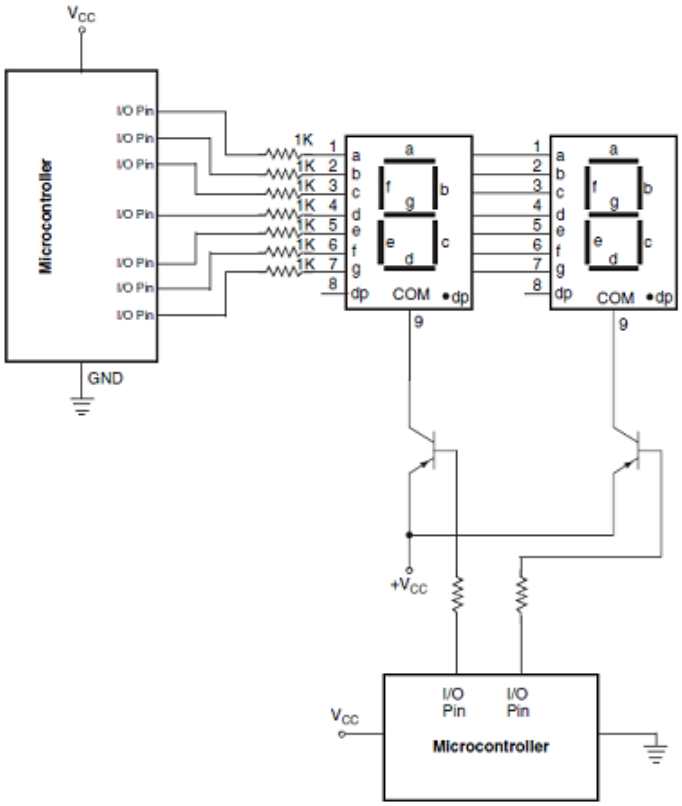
Gambar 28 Konfigurasi antarmuka display common cathode dengan mikrokontroler



Gambar 29 Rangkaian hubungan secara langsung untuk display common anode



Gambar 30 Rangkaian multiplex untuk dua display common cathode



Gambar 31 Rangkaian multiplex untuk dua display common anode

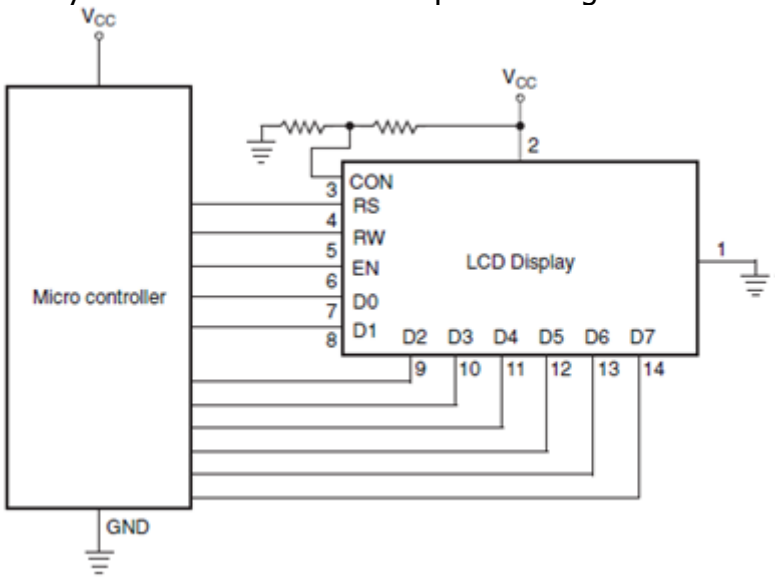
Antarmuka Display LCD

LCD (*liquid crystal display*) mempunyai antarmuka yang lebih baik jika dibandingkan dengan display LED, display LCD jauhlebih mudah untuk menampilkan pesan teks dalam display LCD. Display LCD juga mengkonsumsi daya yang lebih kecil dari pada LED. Namun demikian display LED mempunyai intensitas yang lebih baik dari pada display LCD.

Display LCD dapat diperoleh dengan format 8× 2, 16× 2, 20 × 2 atau 20× 4. 20× 2 berearti terdiri dari dua baris yang setiap barisnya teriri dari masing-masing 20 karakter.Display ini di dalamnya dilengkapi dengan pengontrol LCD. Gambar 1.30 menunjukan antarmuka dari sebuah display LCD dengan mikrokontroler. Terdapat tiga buah kontrol yang terdapat di display LCD yaitu: EN (enable), RS (register select) dan R/W (read/write). EN digunakan untuk menginstruksikan LCD bahwa mikrokontroler sedang mengirimkan data. Ketika RS adalah HIGH, data dari data teks akan ditampilkan pada LCD. Ketika RS adalah LOW, data diolah sebagai perintah atau instruksi ke modul

LCD.Ketika RW adalah LOW, instruksi pada bus data ditulis pada LCD. Ketika RW adalah HIGH, data akan dibaca dari LCD.

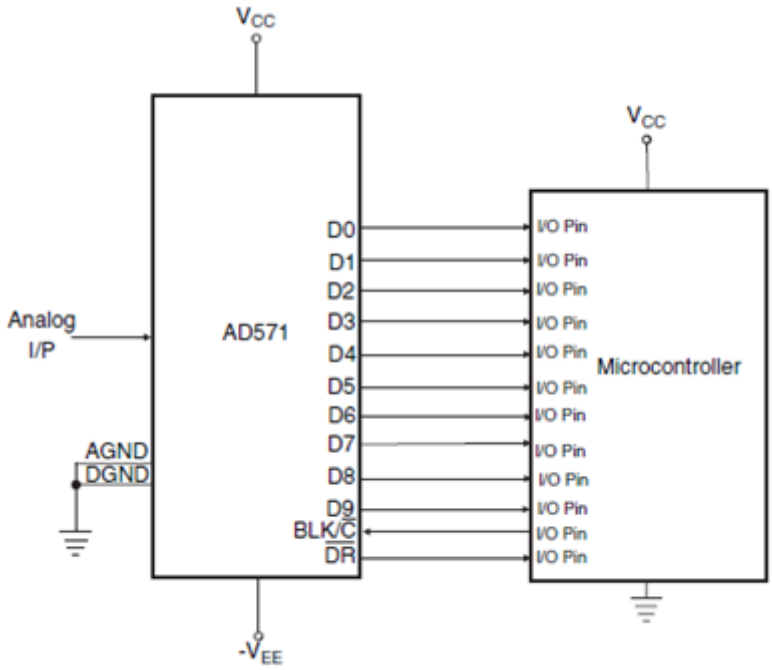
Software akan meng-inisialisasi LCD untuk pertama kalinya dengan mengatur lebar dari bus data, dengan memilih karakter, huruf, pembersian LCD, tuning pada modul LCD, tuning cursor, pengaturan pasisi cursor dan lain sebagainya. Kemudian data akan di-displaykan dikirim ke garis data dan sinyal kontrol dibuat unruk meyakinkan bahwa LCD beroperasi dengan benar.



Gambar 32 Antarmuka display LCD dengan mikrokontroler

Antarmuka Pengubah A/D

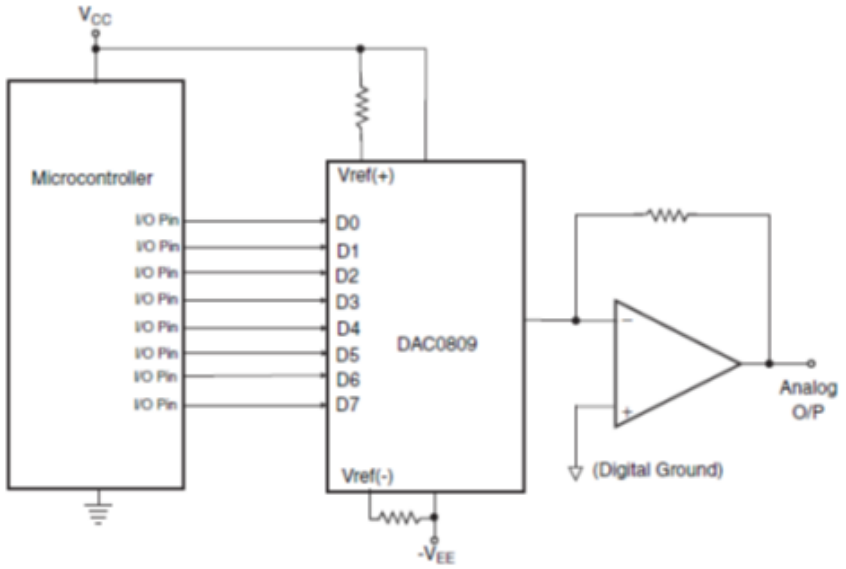
Pengubah A/D digunakan untuk menfasilitasi antarmuka mikrokontroler dengan sinyal analog.Gambar 1.31 menunjukan antarmuka pengubah A/D jenis AD571 dengan mikrokontroler.AD571 adalah pengubah A/D delapan-bit. Seperti dapat dilihat ari , bahwa output data dan kontrol dari pengubah A/D dihubungkan pin I/O mikrokontroler. Mikrokontroler mengirim perintah seperti mulai konversi, memilih kanalinput jika pengubah A/D mempunyai kanal input lebih dri satu dan lain-lain. Mikrokontroler juga mensensor sinyal dari pengubah A/D seperti akhir dari konversiutnuk menyimpan bit digital. Dalam kasus ini, mikrokontroler megirim sinyal LOW pada garis BLANK/DR untu memulai proses konversi, kemudian menunggu sinyal data ready (DR) menjadi LOW. Kemudian bit output digital diterima oleh icrocontroller dan diproses menurut sesuai dengan program yang dibuat sebelumnya.



Gambar 33 Antarmuka pengubah A/D dengan mikrokontroler

Antarmuka Pengubah D/A

Ketika antarmuka suatu pengubah D/A diberkan kepada mikrokontroler, garis data dan garis kontrol, seperti memulai konversi dan garis memilih chip (chip select) dihubungkakan ke pin mikrokontroler. Software membangkitkan sinyal yang diperlukan untuk memulai proses konversi. Gambar 1.32 menunjukan antarmuka pengubah D/A jenis DAC 809 dengan mikrokontroler. DAC-809 adalah suatu pengubah Bit D/A delapan bit. Di sini, keluaran keluarannya adalah berupa arus, denga demikian jika menginginkan keluarannya adalah tegangan, maka diperlukan pengubah arus ke tegangan pada output-nya



Gambar 34 Antarmuka pengubah D/A dengan mikrokontroler

Piranti Lunak Mikrokontroler

Pemrograman Mikrokontroler AT89S/XX

Setelah kalian memahani hal-hal dasar berkenaan dengan mikrokontroler, diantaranya arsitektur MK meliputi CPU, memori, I/O port, dan periperal tambahan, selanjutnya mengetahui berbagai istilah dan bagian umum dari pin-pin MK. Bagian penting selanjutnya adalah teknik pemrograman dan hal yang terkait dengan pemrogramana.

Perkembangan pemrograman sangat cepat seiring dengan perkembangan teknologi mikrokontroler itu sensdiri. Diawali dengan bahasa assembler samapai dengan bahasa tingkat tinggi, saat ini pengguna dapat memeilih bahasa yang akan digunakansesuai dengan compiler (penerjemah bahasa).

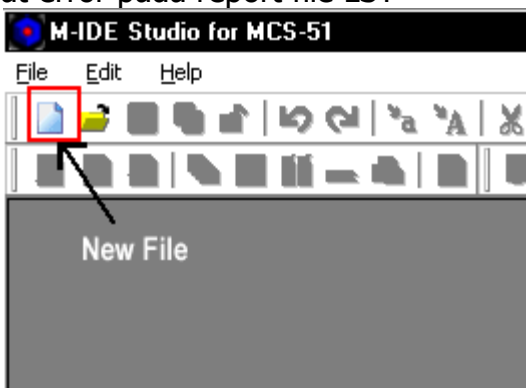
Pada kegiatan belajar ini sebagai permulaan mempelajari mikrokontroler diperkenalkan bahasa assembele, namun untuk perkembangan lebih lanjut kalian diperbolehkan mempelajari bahasa tingkat tinggi seperti bahasa basic, bahasa C, pascal, java dan berbagai hasa lainnya sesuai dengan perkembangan teknologi informasi dan komunikasi.
Kalian dapat menuliskan program assember pada note pad kemudian di compiler menggunakan ASM51 atau MIDE.

Tugas 8

Mengapa setelah kalian menuliskan program assembler dibutuhkan compiler? Berikan alasannya!

Compiler MIDE Studio

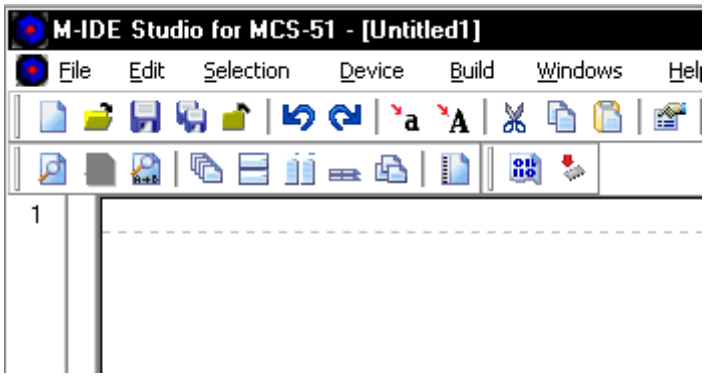
M-IDE Studio adalah salah satu cara yang digunakan untuk menjalankan kompilasi untuk komponen MCS-51. M-IDE Studio mempunyai beberapa fitur yang dapat digunakan untuk edit, compil, dan debug file.
The M-IDE Studio juga dapat digunakan untuk menulis program dalam bahasa C. Dengan menggunakan software ini, maka kita dapat melihat error pada report file LST



Gambar 35 M-IDE Studio

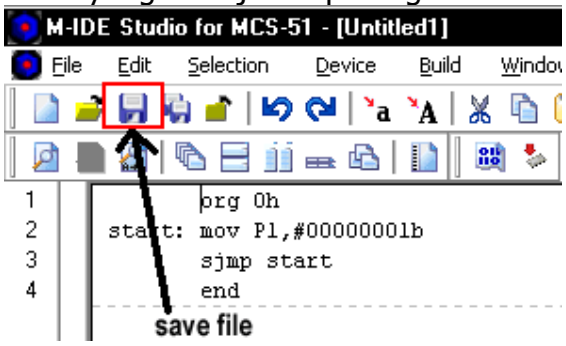
Bila anda perhatikan pada menu toolbar dan menu pilihan, tampak terlihat disable. Hal ini karena file belum dibuat. Untuk membuat sebuah file, lakukan langkah-langkah berikut:

1. Membuat File Baru
   Untuk membuat file baru, klik pada menu File atau short cut seperti yang ditunjukkan pada gambar, sehingga akan tampak halaman kosong.



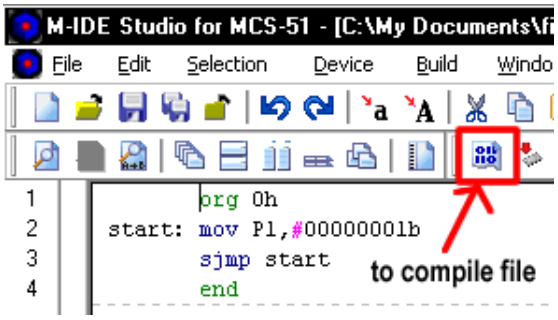
Gambar 36 File baru dengan halaman kosong

2. Menulis sebuah program Tulis program assembly pada halaman kosong, dan lakukan penyimpanan file. Bila file telah tersimpan maka akan tampak teks instruksi yang berwarna-warni. Sebagaimana yang ditunjukan pada gambar 1.35.



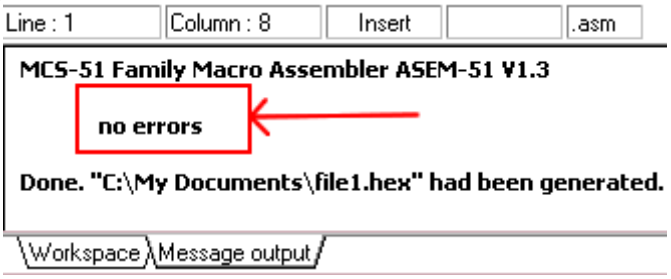
Gambar 37 Menu penyimpanan file

3. Kompilasi
   Agar file dengan ekstensi ASM tersebut dapat diloadkan ke mikrokontroller, maka perlu dilakukan kompilasi dari file ASM ke HEX.



s

4. Debug



Gambar 38 Debug file

Dengan menggunakan mikrokontroler AT89S dapat dilakukan pemrograman mikrokontroler dengan cara *In System Programming* (ISP). Artinya IC mikrokontroler tidak perlu dicabut pasang untuk pemrograman. Untuk melakukan download program dapat menggunakan ISP programmer buatan M. Asim Khan atau Downloader Usb Asp Avr & 8051.

Bahasa Assembly di Mikrokontroler

Pemrograman menggunakan bahasa assembly (bahasa tingkat rendah) memerlukan pemahaman register, termasuk didalamnya nama setiap register dari chip kontroler yang digunakan dan struktur register itu sendiri.

Secara fisik, kerja dari sebuah mikrokontroler dapat dijelaskan sebagai siklus pembacaan instruksi yang tersimpan di dalam memori. Mikrokontroler menentukan alamat dari memori program yang akan dibaca, dan melakukan proses baca data di memori. Data yang dibaca diinterprestasikan sebagai instruksi.Alamat instruksi disimpan oleh mikrokontroler di register, yang dikenal sebagai program counter. Instruksi ini misalnya program aritmatika yang melibatkan 2 register.

Sarana yang ada dalam program assembly sangat minim, tidak seperti dalam bahasa pemrograman tingkat atas (high level language programming) semuanya sudah siap pakai.Penulis program assembly harus menentukan segalanya, menentukan letak program yang ditulisnya dalam memori-program, membuat data konstan dan tablel konstan dalam memori-program, membuat variabel yang dipakai kerja dalam memori-data dan lain sebagainya.

1.2.1. Program Assembly

Program-sumber assembly (assembly source program) merupakan kumpulan dari baris-baris perintah yang ditulis dengan program penyunting-teks (text editor) sederhana, misalnya program EDIT.COM dalam DOS, atau program NOTEPAD dalam Windows atau MIDE-51. Kumpulan baris-printah tersebut biasanya disimpan ke dalam file dengan nama ekstensi *.ASM dan lain sebagainya, tergantung pada program Assembler yang akan dipakai untuk mengolah program-sumber assembly tersebut.

Setiap baris-perintah merupakan sebuah perintah yang utuh, artinya sebuah perintah tidak mungkin dipecah menjadi lebih dari satu baris. Satu baris perintah bisa terdiri atas 4 bagian, bagian pertama dikenali sebagai label atau sering juga disebut sebagai symbol, bagian kedua dikenali sebagai kode operasi, bagian ketiga adalah operand dan bagian terakhir adalah komentar.Antara bagian-bagian tersebut dipisahkan dengan sebuah spasi atau tabulator.

Label

Label dipakai untuk memberi nama pada sebuah baris-perintah, agar bisa mudah menyebitnya dalam penulisan program. Label bisa memberi nama pada baris bersangkutan.

Bagian label sering disebut juga sebagai bagian symbol, hal ini terjadi kalau label tersebut tidak dipakai untuk menandai bagian program, melainkan dipakai untuk menandai bagian data.

Bagian Kode Operasi

Kode operasi (operation code atau sering disingkat sebagai OpCode) merupakan bagian perintah yang harus dikerjakan. Dalam hal ini dikenal dua macam kode operasi, yang pertama adalah kode-operasi untuk mengatur kerja mikroprosesor / mikrokontroler. Jenis kedua dipakai untuk mengatur kerja program assembler, sering dinamakan sebagai assembler directive.Kode-operasi ditulis

dalam bentuk mnemonic, yakni bentuk singkatan-singkatan yang relatip mudah diingat, misalnya adalah MOV, ACALL, RET dan lain sebagainya. Kode-operasi ini ditentukan oleh pabrikpembuatikroprosesor/mikrokontroler.Tugas penerjemahan tersebut dilakukan oleh program yang dinamakan sebagai Program Assembler.Di luar kode-operasi yang ditentukan pabrik pembuat mikroprosesor/mikrokontroler, ada pula kode-operasi untuk mengatur kerja dari program assembler, misalnya dipakai untuk menentukan letak program dalam memori (ORG), dipakai untuk membentuk variabel (DS), membentuk tabel dan data konstan (DB, DW) dan lain sebagainya.

Bagian operand

Operand merupakan pelengkap bagian kode operasi, namun tidak semua kode operasi memerlukan operand, dengan demikian bisa terjadi sebuah baris perintah hanya terdiri dari kode operasi tanpa operand. Sebaliknya ada pula kode operasi yang perlu lebih dari satu operand, dalam hal ini antara operand satu dengan yang lain dipisahkan dengan tanda koma.Bentuk operand sangat bervariasi, bisa berupa kode-kode yang dipakai untuk menyatakan Register dalam prosesor, bisa berupa nomor-memori (alamat memori) yang dinyatakan dengan bilangan atau pun nama label, bisa berupa data yang siap di-operasi-kan.

Bagian komentar

Bagian komentar merupakan catatan-catatan penulis program, bagian ini meskipun tidak mutlak diperlukan tapi sangat membantu masalah dokumentasi. Membaca komentar-komentar pada setiap baris-perintah, dengan mudah bisa dimengerti maksud tujuan baris bersangkutan, hal ini sangat membantu orang lain yang membaca program.Pemisah bagian komentar dengan bagian sebelumnya adalah tanda spasi atau tabulator, meskipun demikian huruf pertama dari komentar sering-sering berupa tanda titik-koma, merupakan tanda pemisah khusus untuk komentar.Untuk keperluan dokumentasi yang intensip, sering-sering sebuah baris yang merupakan komentar saja, dalam hal ini huruf pertama dari baris bersangkutan adalah tanda titik-koma.AT89S51 memiliki sekumpulan instruksi yang sangat lengkap.Instruksi MOV untuk byte dikelompokkan sesuai dengan mode pengalamatan (addressing modes).Mode pengalamatan menjelaskan bagaimana operand dioperasikan.Berikut penjelasan dari berbagai mode

pengalamatan. Bentuk program assembly yang umum ialah sebagai berikut :

Label/Simbol	Opcode	Operand	Komentar
	Org	0H	
Start: Kiri: Delay: Del1: Del2:	Mov Mov Mov Call RL DEC CJNE Sjmp mov mov djnz djnz ret end	A, #11111110b R0, #7 P0, A Delay A R0 R0, #0, Kiri Start R1, #255 R2, #255 R2, del2 R1, del1	; Isi Akumulator ; Isi R0 dengan 7 ; Copy A ke P0 ; Panggil Delay

Isi memori ialah bilangan heksadesimal yang dikenal oleh mikrokontroler kita, yang merupakan representasi dari bahasa assembly yang telah kita buat. Mnemonic atau opcode ialah kode yang akan melakukan aksi terhadap operand .Operand ialah data yang diproses oleh opcode. Sebuah opcode bisa membutuhkan 1 ,2 atau lebih operand, kadang juga tidak perlu operand. Sedangkan komentar dapat kita berikan dengan menggunakan tanda titik koma (;). Berikut contoh jumlah operand yang berbeda beda dalam suatu assembly.

```
CJNE R5,#22H, aksi ;dibutuhkan 3 buah operand
MOVX @DPTR, A ;dibutuhkan 2 buah operand
RL A ;1 buah operand
NOP ; tidak memerlukan operand
```

Program yang telah selesai kita buat dapat disimpan dengan ekstension .asm. Lalu kita dapat membuat program objek dengan ekstension HEX dengan menggunakan compiler MIDE-51, yang dijelaskan sebagai berikut:

1.2.2. Struktur Pemrograman

Dalam pemograman 8051 memakai bahasa assembler yang dapat dibuat di teks editor dan diubah menjadi bahasa mesin melalui suatu proses. Penerjemahan ini melalui dua fase, fase pertama merubah bahasa assembler menjadi kode-kode simbol (mnemonik),

sedangkan fase kedua akan menerjemahkan mnemonik menjadi bahasa mesin.

A.1.Simbol
Simbol adalah alpha numeric dari konstanta numeric, alamat, makro, dan sebagainya. Karakter dalam symbol dapat berupa gabungan huruf besar (A…Z) dan huruf kecil (a…z), bilangan desimal (0…9), karakter khusus (? dan _). Symbol hanya boleh didefinisikan satu kali dengan panjang maksimal 255 karakter, selain itu dilarang memakai kata cadangan (NE, MOD, SHR, …) dan dilarang pula memakai instruksi operand (A, DPTR, …).

A.2. **Label**
Label adalah symbol khusus sebelum statemen dan berhubungan dengan alamat fisik DB dan DW, ataupun data langsung reservation DS dan DBIT.

A.3. **Kontrol**
Kontrol digunakan untuk mengontrol program assembler, seperti keberadaan file sumber sebagai tempat file objek dan format file tersebut.Kata kontrol dalam assembler selalu dimulai tanda dollar ($) dan tiada ruang di antara $ dan kata kontrol.

A.4. **Penunjukkan**
Digunakan untuk mendefinisikan symbol, ruang memori, cadangan, nilai hasil, memori program dan pemilihan ruang memori yang berbeda, juga sebagai penunjuk lokasi counter.
Contoh : CODE, END.

A.5**. Instruksi mnemonik**
Standar instruksi mnemonik :
Dalam pemakaiannya standar instruksi ditandai oleh tanda tertentu sesuai mode pengalamatan, (@) sebagai tanda mode pengalamatan tidak langsung, (#) sebagai tanda operand yang dijalankan immediate, (/) sebagai tanda dalam spesifikasi bagian bit alamat yang dikomplemenkan sebelum instruksi.

A.6**. Bit Pengalamatan**
Tanda (.) memiliki arti khusus dalam pemakaian bahasa assembler, yaitu pemakaian spesifikasi secara tepat satu bit dalam pengalamatan bit.

A.7. **Harfiah dalam ASCII**

Karakter ASCII dapat dipakai langsung sebagai immediate operand atau didefinisikan dalam memori program, dengan diikuti tanda apostrophe (‘).

A.8. **Komentar**

Komentar digunakan untuk menuliskan karakter string, tapi tidak akan diproses dalam program. Komentar dimulai tanda semi kolon (;).

A.9. **Counter Lokasi**

Bahasa assembler selalu memakai lokasi counter terhadap kelima segmennya, yaitu kode, internal data, eksternal data, indirect internal data dan bit data. Tiap lokasi counter diinisialisasi dengan nol dan bisa dimodifikasi dengan penunjukan alamat.

A.10. **Bilangan dan Operator**

Bahasa assembler dapat menggunakan salah satu dari jenis bilangan yaitu binary, octal, desimal ataupun heksadesimal.Tapi secara defaultnya bilangan yang dipakai berbasiskan desimal.

A.11. **Operasi Bergantung Waktu**

Prioritas operasi dalam bahasa assembler dengan urutan prioritas tertinggi ke prioritas terendah : (,); high,low; *, /, mod, shr, shl; +,-; EQ, LT,GT, LE,GE, NE, =, <, >, <=, >=, <>; NOT; AND; OR; XOR.

B. **SET INSTRUKSI**

Keluarga MCS-51 memiliki 111 perintah dengan 49 buah tipe perintah single-byte, 45 buah tipe perintah dua-byte, dan 17 buah tipe perintah tiga-byte. Perintah ini dapat dikelompokkan dalam empat kelompok berdasarkan fungsinya.

MCS-51 memiliki instruksi dengan jumlah yang cukup banyak. Instruksi-instruksi tersebut dapat dimasukkan ke dalam beberapa golongan . Daftar kelompok tersebut dapat dilihat pada table di bawah ini:

B.1. **Instruksi Aritmatik**

Instruksi Aritmatik mencakup instruksi-instruksi yang melakukan proses aritmatik, antara lain: penjumlahan,pengurangan,perkalian,

dan pembagian. Umumnya instruksi ini menggunakan *Accumulator* sebagai salah satu *operand*-nya

Instruksi	Penjelasan singkat
ADD A,*source*	Menambah data dengan *Accumulator*
ADD A,#data	
ADDC A,*Source*	Menambah data, *carry flag*, dan *accumulator*
ADDC A,#data	
SUBB A,***Source***	Mengurangi *Accumulator* dengan data
SUBB A,#data	
INC A	Menambah dengan 1
INC A,*Source*	
INC DPTR	
DEC A	Mengurangi dengan 1
DEC **Source**	
MUL AB	Mengalikan *Accumulator* dengan B
DIV AB	Membagi *Accumulator* dengan B
DA A	Konversi data ke desimal

Instruksi		Penjelasan singkat
ANL	A,***Source***	Operasi logika AND antar bit pada kedua data
ANL	A,#data	
ANL	*Direct*,A	
ANL	*Direct,*#data	
ORL	A,***Source***	Operasi logika OR antar bit pada kedua data
ORL	A,#data	
ORL	*Direct*,A	
ORL	*Direct,*#data	
XRL	A,***Source***	Operasi logika XOR antar bit pada kedua data
XRL	A,#data	
XRL	*Direct*,A	
XRL	*Direct,*#data	
CRL	A	Memberi nilai 00h pada *Accumulator*
CPL	A	Komplemen setiep bit pada *Accumulator*
RL	A	Merotasi *Accumulator* ke kiri
RLC	A	Merotasi *Accumulator* ke kiri melalui carry flag
RR	A	Merotasi *Accumulator* ke kanan
RRC	A	Merotasi *Accumulator* ke kanan melalui carry flag
SWAP	A	Menukar posisi 4 bit terendah (*lower nibble*) dengan 4 bit tertinggi (*upper nible*)

B.2. **Instruksi Logika**

Instruksi logika mencakup instruksi-instruksi yang melakukan proses logika terhadap register 8 bit.

Umumnya instruksi ini menggunakan *Accumulator* atau alamat (*direct*) sebagai salah satu *operand*-nya

B.3. **Instruksi Boolean**

Instruksi Boolean mencakup instruksi-instruksi yang hanya melibatkan 1 bit saja.Instruksi ini menggunakan carry flag atau register 1 bit lainnya sebagai *operand*

Instruksi		Penjelasan singkat
CLR	C	Memberi nilai '0' pada bit

CLR	Bit	
SETB	C	Memberi nilai '1' pada bit
SETB	Bit	
CPL	C	Komplemen bit
CPL	Bit	
ANL	C,Bit	Operasi logika AND antar bit
ANL	C,Bit	
ORL	C,Bit	Operasi logika OR antar bit
ORL	C,Bit	
MOV	C,Bit	Mengisi niali dari bit ke bit
MOV	Bit,C	
JC	Rel	Lompat ke alamat tertentu jika carry flag = '1'
JNC	Rel	Lompat ke alamat tertentu jika carry flag = '0'
JB	Bit ,Rel	Lompat ke alamat tertentu jika bit = '1'
JNB	Bit ,Rel	Lompat ke alamat tertentu jika bit = '0'
JBC	Bit ,Rel	Lompat ke alamat tertentu jika bit = '1' lalu mengisis bit dengan nillai 0

B.4. **Instruksi Percabangan (*Branch*)**

Instruksi Percabangan (*Branch*) mencakup instruksi-instruksi yang melakukan proses perpindahan alamat. Instruksi-instruksi tersebut antara lain: pemanggilan rutin (*Call*) dan lompat (*Jump*)

Instruksi		**Penjelasan Singkat**
ACALL	Addr11	Memanggil suatu subrutin secara absolut dengan alamat 11-bit
LCALL	Addr16	Memanggil suatu subrutin secara jauh (long) dengan alamat 16-bit
RET		Kembali dari suatu sub rutin
RETI		Kembali dari suatu rutin interupsi
AJMP	Addr11	Lompat ke alamat tertentu
LJMP	Addr16	
SJMP	Rel	

JMP	@a+DPTR	
JZ	Rel	Lompat ke alamat tertyentu jika Accumulator = 00h
JNZ	Rel	Lompat ke alamat tertyentu jika Accumulator tidak bernilai 00h
CJNE	A,direct,rel	Membadingkan ke dua operand dan melompat ke alamat tertentu jika ke dua operand tidak sama
CJNE	A,#data,rel	
CJNE	Rn,#data,rel	
CJNE	@R1,#data,rel	
DJNZ	Rn,rel	Mengurangi operand dan melompat ke alamat tertentu jika operand tidak bernilai 00h
DJNZ	Direct	
NOP		Tidak ada operasi

B.5. **Instruksi Transfer Data**

Instruksi Transfer Data mencakup instruksi-instruksi yang melakukan proses pemindahan atau pertukaran data yang melibatkan register 8 bit atau 16 bit.

Instruksi		**Penjelasan Singkat**
MOV	A,***Source***	Mengisi nilai operand kedua (***Source)*** ke dalam operand pertama (*Destination*)
MOV	A,#data	
MOV	Dest,A	
MOV	Dest , ***Source***	Mengisi nilai dari operand kedua (*source*) ke dalam operand pertama (*destination*)
MOV	Dest,#data	
MOV	DPTR,#data 16	
MOV	A,@A+DPTR	Mengisi nilai dari program

MOVC	A,@A+PC	memory ke dalam *Accumulator*
MOVX	A,@Ri	Mengisi nilai dari *external data memory*
MOVX	A,@DPTR	
MOVX	@Ri,A	
MOVX	@DPTR,A	
PUSH	Direct	Mengisi nilai ke dalam *stack*
POP	Direct	Mengambil nilai dari *stack*
XCH	A,Source	Menukar nilai kedua *operand*
XCHD	A,@Ri	Menukar 4 bit terendah dari kedua *operand*

Sistem Pengalamatan

- Pengalamatan Langsung

Immediate data

Proses pengalamatan ini terjadi pada sebuah perintah ketika nilai operand merupakan data yang akan dip roses. Biasanya operand tersebut selalu diawali dengan tanda ‘’#” seperti pada contoh di bawah ini:

```
Mov     A,#05H
Mov     A,#Data ; pada bagian di atas data telah didefinisikan sebagai bilangan
          ; tertentu (data EQU  05) contohnya bilangan 5.
```

- Pengalamatan Langsung Data (direct)

Proses pengalamat ini terjadi pada sebuah perintah ketika nilai operand merupakan alamat dari data yang akan diisi, dipindahkan atau diproses.

```
Mov     P0,A
```

Port 0 adalah salah satu I/O dari AT89C51 yang mempunyai alamat 80H. Perintah pada contoh di atas dapat di tulis juga:

```
Mov     80H,A
```

Pengalamatan Tak Langsung

Proses pengalamat ini terjadi pada sebuah perintah ketika salah satu operand merupakan register berisikan alamat dari data yang akan diisi dan dipindahkan. Pengalamatan jenis ini biasa digunakan untuk melakukan penulisan, pemindahan atau pembacaan beberapa data dalam lokasi memori yang urutan beraturan.

Jika proses ini dilakukan dengan menggunakan pengalamatan langsung , jumlah baris program yang akan diperlukan akan cukup panjang seperti contoh di bawah ini:

```
Mov     50H, #08H
Mov     51H, #08H
Mov     52H, #08H
Mov     53H, #08H
Mov     54H, #08H
Mov     55H, #08H
Mov     56H, #08H
Mov     58H, #08H
```

Dengan digunakan system pengalamatan tak langsung, dapat diubah menjadi:

```
        Mov   R0,#50H
Loop:
        Mov   @R0,#08H
        Inc   R0
        Cjne  R0,#58H,loop
```

Dalam listing ini, R0 digunakan sebagai register yang menyimpan alamat dari data yang akan dituliskan. Dengan melakukan penambahan pada isi R0 dan mengulang perintah penulisan data ke alamat yang ditunjukan R0 hingga register ini menunjukan nilai 57H+1, atau 58H.

AT89S51 mempunyai sebuah register 16 bit (DPTR) dan dua buah register 8 bit (R0 dan R1) yang dapat digunakan untuk melakukan penngalamatan tidak langsung.
Contoh-contoh perintah menggunakan system pengalamatan tak langsung:

```
Mov     @R0,A       ;R0 digunakan sebagai register penyimpan
alamat
Mov     A,@R1       ;R1 digunakan sebagai register penyimpan
alamat
Add     A,@R0       ;R0 digunakan sebagai register penyimpan
alamat
Mov     @DPTR,A     ;DPTR digunakan sebagai register
penyimpan                                          alamat
Mov     A,@A+DPTR ;DPTR digunakan sebagai register
penyimpan                                          alamat
```

- Pengalamatan Kode

Pengalamatan kode merupakan pengalamatan ketika operand merupakan alamat dari instruksi jump dan call (ACALL,JMP,LJMP

dan LCALL). Biasanya operand tersebut akan menunjukkan ke suatu alamat yang telah di beri label sebelumnya seperti contoh:

```
                Acall   delay
                ……..

        Delay:
                Mov     B,#0FFH
        LoopDelay:
                Djnz    B LoopDelay
                Ret
```

- Pengalamatan Bit

Proses pengalamatan ketika operand menunjukkan ke alamat RAM internal ataupun Register fungsi khusus yang mempunyai kemampuan pengalamatan secara bit (*bit addressable*).
Berdasarkan penulisannya, pengalamatan ini terdiri dari:

- Langsung menunjuk ke alamat bit

```
Setb  0B0H
```

Perintah ini memberikan logika 1 pada bit di alamat B0H dengan pengalamat secara bit.

- Menggunakan operator titik
- Menggunakan lambang assembler secara standard
- Menggunakan lambang assembler secara bebas

Pengalamatan bit ialah penunjukkan menggunakan simbol titik (.)yang menunjuk alamat lokasi bit, baik dalm RAM internal atau perangkat keras.

Contoh :

```
SETB pl.7       ;Set bit port 1.7 aktif
SETB TR1        ;Set TR1 (Timer 1 aktif)
SETB RXD        ;memberikan logika 1 pada kaki RXD yang
                berada di ;port 3.0
```

F. Operator

Operator digunakan untuk melakukan aksi aritmatika, logika pergeseran bit,dan lain-lain pada operand.Beberapa operator yang tersedia diantaranya adalah :

- Operator Aritmatika
  - ✓ * untuk perkalian
  - ✓ / untuk pembagian
  - ✓ + untuk pertambahan
  - ✓ - untuk pengurangan

  Contoh:

```
MOV A, #25H+3H ;  sama dengan MOV A,#28H
```

- Operator Logika
  - ✓ OR untuk operasi OR
  - ✓ AND untuk operasi AND
  - ✓ XOR untuk operasi XOR
  - ✓ EXOR untuk operasi EXOR
  - ✓ NOT untuk operasi NOT

Contoh :

```
MOV A, #20H  OR 40H          ;sama dengan MOV A, #60H
MOV A,#10H  AND 31H          ;sama dengan MOV A,  10H
```

Contoh Program

Sekarang kita akan memodifikasi program tersebut sedemikian rupa hingga bisa untuk menghidupkan lampu LED (sebanyak empat) dalam kelompok genap dan ganjil secara bergantian sebagai mana program berikut:

```
;-------------------------------------------------------------------------------------------
    --------
; lampu flip-flop ganjil dan genap pada port 1
;-------------------------------------------------------------------------------------------
ORG  00H
MULAI:      MOV   P1, #01010101B     ;led p1.4 s/d p1.7 nyala
      ACALL DELAY                ;panggil subrutin delay
      MOV   P1, #10101010B     ;led p1.0 s/d p1.3 nyala
      ACALL DELAY                ;panggil subrutin delay
      SJMP  MULAI          ;lompat kealamat dengan label mulai

;-----------------------------------
; subrutin delay
;-----------------------------------

DELAY:              MOV   R0,#5          ;isi register R0 dengan 5
DELAY1:     MOV   R1,#0FFH     ;isi register R1 dengan FF (Hex)
DELAY2:
       DJNZ R1, DELAY2           ;kurangi R1 dengan 1, bila hasil
                                        ; belum sama Dengan 0 maka lompat
                                                ;ke delay2
             DJNZ      R0, DELAY1        ;kurangi R0 dengan 1, bila hasil
                                        ; belum     sama     Dengan  0
    maka                                                ;lompat ke delay1
```

```
RET                                 ;kembali ke alamat setelah perintah
                                                ;`Acall Delay'

End
```

Program berikut ini merupakan aplikasi untuk membuat LED menyala bergantian bergeser dari P0.0 ke P0.7 kemudian kembali ke awal lagi secara berulang-ulang

```
;-------------------------------------------------------------------------------------------
; lampu nyala  bergeser pada port 0
;-------------------------------------------------------------------------------------------
$MOD51
      ORG    0H
      MOV   A, #11111110B      ;simpan data ke akumulator
MULAI:      MOV   P0,A                  ;keluarkan isi akumulator ke port0
      ACALL DELAY               ;panggil subrutin delay
      RL      A                 ;putar isi akumulator ke kiri 1 bit
      SJMP   MULAI         ;lompat kealamat dengan label mulai

;-----------------------------------
; subrutin delay
;-----------------------------------

DELAY:            MOV   R0,#5               ;isi register R0 dengan 5
DELAY1:    MOV   R1,#0FFH          ;isi register R1 dengan FF (Hex)
DELAY2:
            DJNZ  R1, DELAY2          ;kurangi  R1  dengan  1,  bila  hasil
                                            ; belum sama Dengan 0 maka
                                                  ; lompat ke delay2
                  DJNZ R0, DELAY1           ;kurangi  R0  dengan  1,  bila
   hasil                                          ;  belum  sama
   Dengan 0 maka                                   ;lompat ke delay1

RET                                 ;kembali  ke  alamat  setelah  perintah
                                                ;`Acall Delay'

END
```

Simulasi Menggunakan Software Proteus ISIS

Pada saat kita melakukan percobaan mikrokontroler terkadang kita mengalami beberapa kendala baik di programnya atau di *hardware*nya, maka akan terdapat dua kemungkinan masalah :

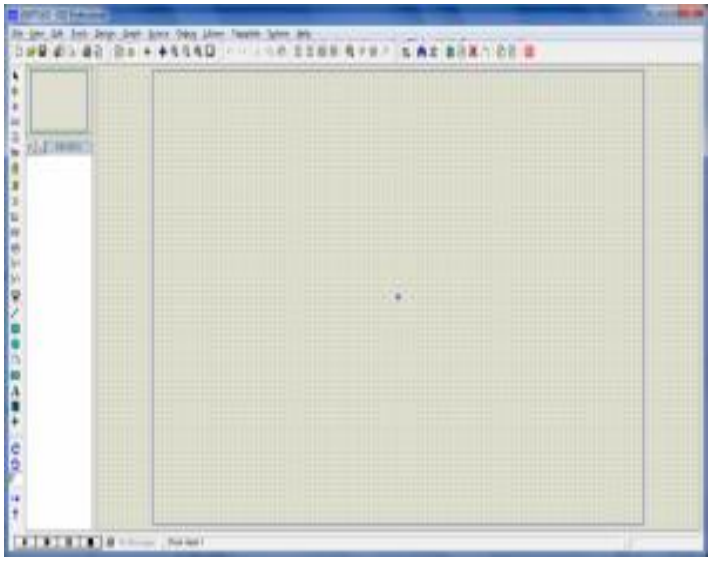
1. Hardware yang tidak berfungsi, atau
2. Program yang di buat salah

3. Perpaduan keduanya

Untuk mengatasi hal tersebut, kita harus melokalisir manakah yang salah dari sistem mikrokontroler kita apakah program ataukah perangkat kerasnya ?. Untuk melihat programnya benar atau tidak, kita sudah membuatnya di software MIDE, pada saat di *build* jika tidak ada *error* berarti program yang kita buat sudah benar, kemudian di compile ke hexa. Karena itulah sebelum kita mendownload program kedalam mikrokontroler maka alangkah baiknya kita simulasikan terlebih dahulu. Salah satu software simulasi yang akan diperkenalkan adalah software Proteus ISIS. Kelebihan software ini kita dapat menggambar rangkaian dan mensimulasikannya.

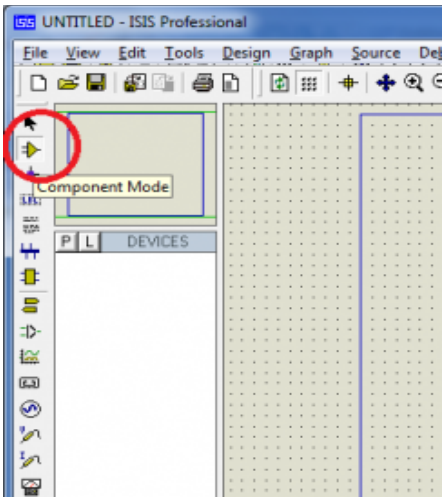
Gambar 39 tampilan ISIS 7 pada saat akan di install

❖ satu perangkat lunak simulasi yang digunakan adalah PROTEUS ISIS 7 Profesional. Berikut tampilan awalnya



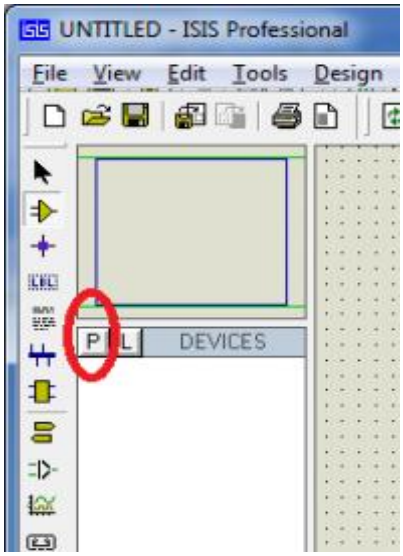
Gambar 40 tampilan awal ISIS 7 l

❖ Dapat di pilih di *Component mode* sepert gambar 1.40.



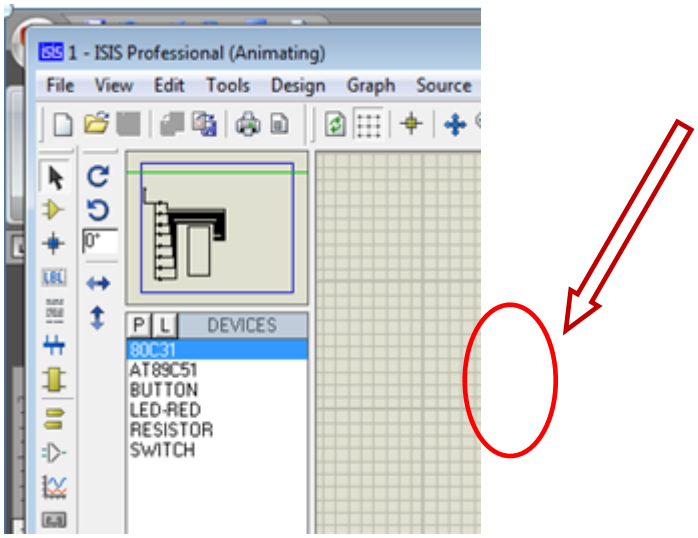
Gambar 41 tampilan Tool Box ISIS 7

❖ Kemudian pilih komponen yang akan digunakan, lihat gambar. Pada toolbox sebelah kiri, pilih *Component mode* kemudian klik tombol yang berisi huruf P Untuk mengaktifkan *Pick Device*. Pick Device adalah box dialog untuk memilih komponen yang akan kita gunakan.



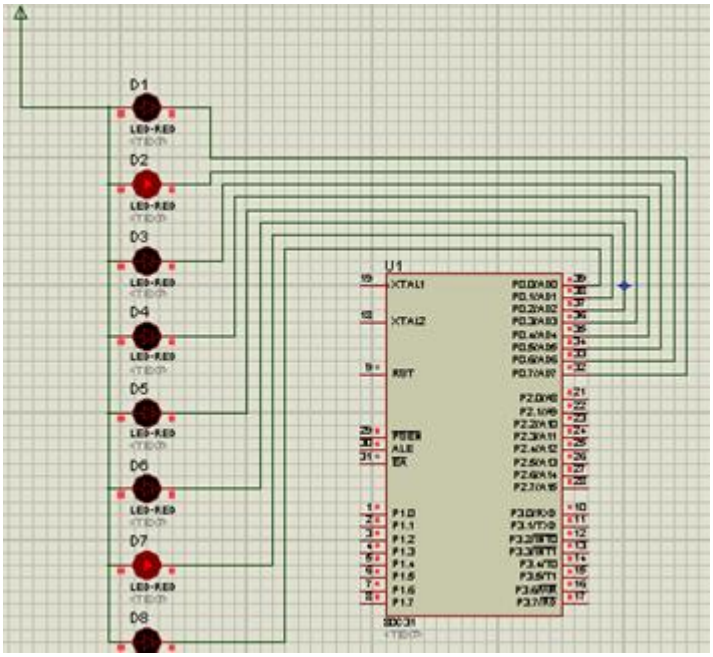
Gambar 42 Tampilan memilih komponen

❖ Contoh: akan membuat gambar rangkaian mikrokontroler dengan display led Komponen yang dipilih adalah : 8031, AT89C51, led red,resistor.



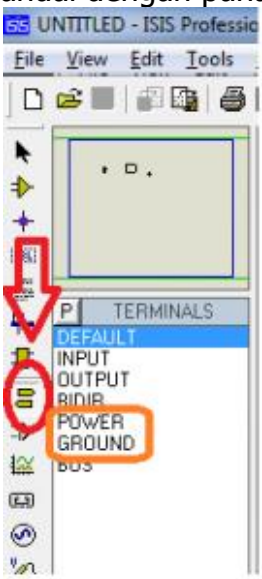
Gambar 43 Tampilan daftar komponen yang akan dirangkai.

❖ Tampilan setelah dirangkai dengan beberapa led



Gambar 44 tampilan rangkaian mikrokontroler dengan output led.

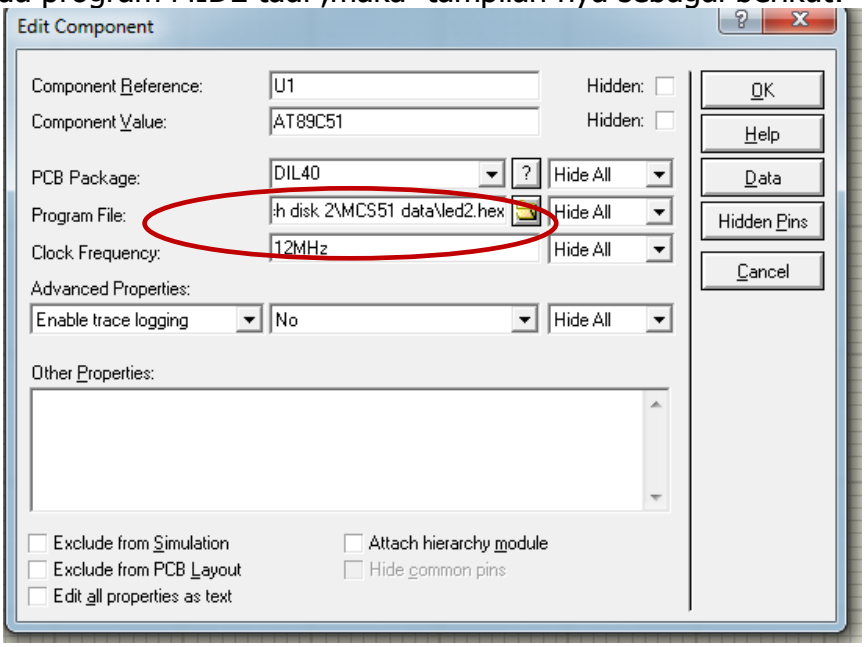
❖ Jika kita memerlukan sumber /power dan ground maka perhatikan toolbar di sebelah kiri ditandai dengan panah merah.



Gambar 45 tampilan daftar power dan ground

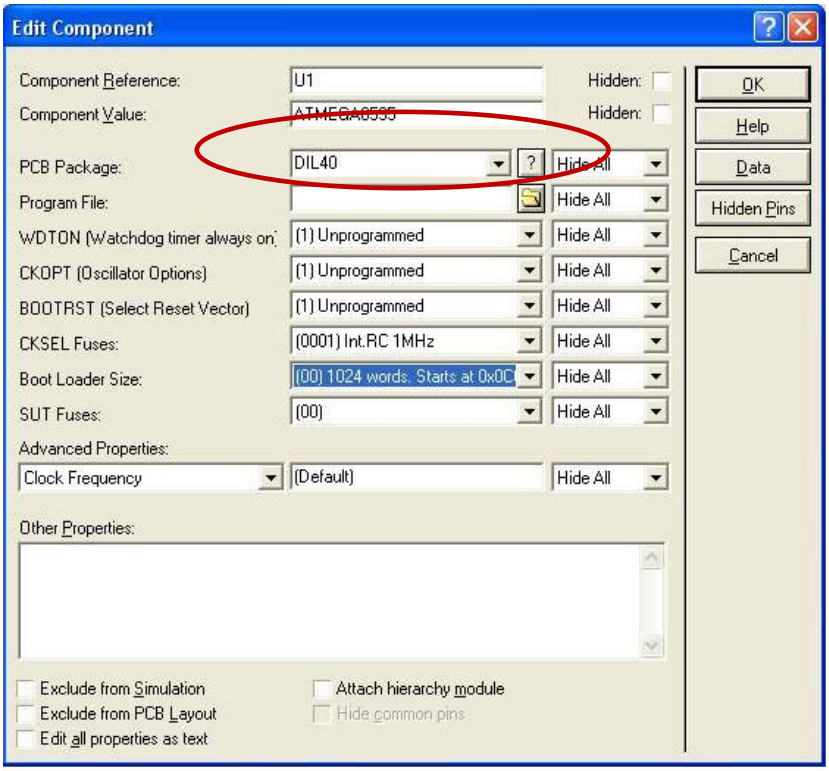
❖ Jika kita akan mensimulasikan program yang telah dibuat pada software MIDE dan di *compile* menjadi file hex. Kalian harus

mengingatnya di simpan dimana file tersebut. Untuk dapat mendownload program tadi pada gambar rangkaian di proteus isis maka klik 2 kali atau *double klik* pada gambar mikrokontrolernya, setelah tampil seperti gambar 2.43. Tempatkan cursor di posisi program file, *browse* file latihan yang sudah di *compile* menjadi hex pada program MIDE tadi ,maka tampilan nya sebagai berikut:



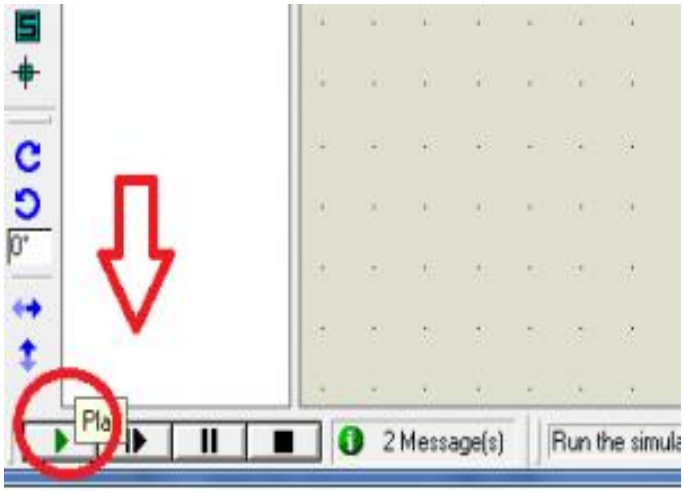
Gambar 46 Tampilan eksekusi program file yang telah menjadi hex

❖ Namun jika mikrokontroler yang dipilih adalah ATMEGA 8535, setelah kita meng-compile latihan menjadi hex maka tampilannya adalah sebagai berikut :



Gambar 47 tampilan jika menggunakan mikrokontroler ATMEGA 8535

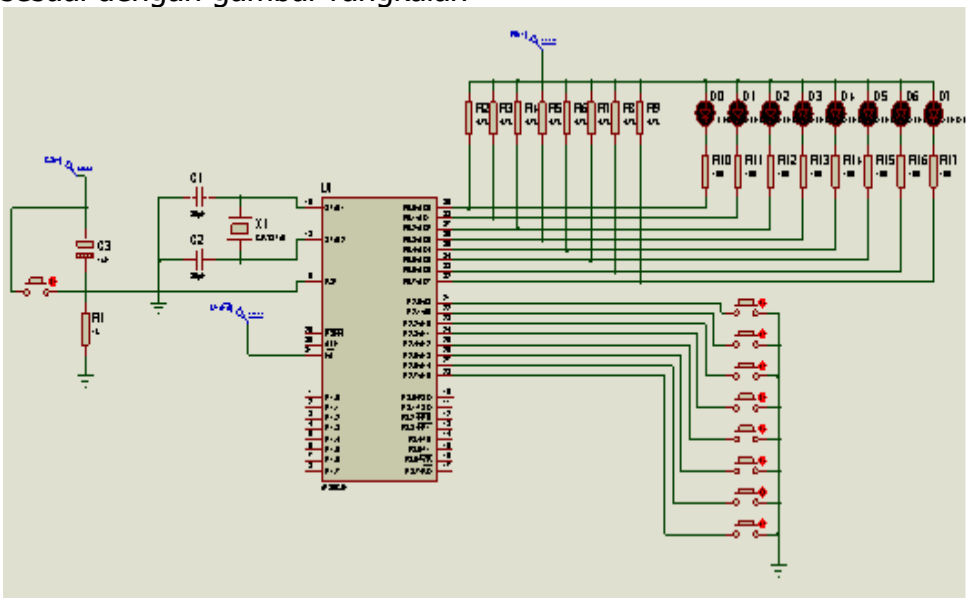
❖ Simulasi akan berjalan setelah kita menekan tombol **PLAY** di pojok kiri bawah :



Gambar 48 tampilan menjalankan simulasi

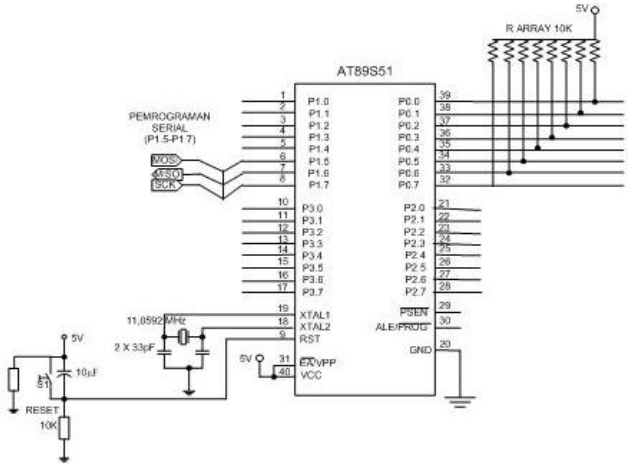
Tugas 7

Buatlah gambar rangkaian seperti di bawah ini,kemudian buat program sesuai dengan gambar rangkaian



Aplikasi Kontrol Berbasis Mikrokontroler

Pada kegiatan belajar ini, kalian diharapkan dapat mempertajam proses analisa rangkaian dan program. Untuk lebih memahaminya dapat kalian gambar menggunakan software ISIS. Sebagai contoh amati gambar di bawah ini:



Gambar 49 Gambar rangkaian minimum.

Menunjukkan rangkain minimum sebuah mikrokontroler, berikut empat bagian yang perlu diamati agar selalu ada dan terdapat dalam sebuah rangkaian minimum (minimum System).

1. Rangkaian reset pada pin 9
2. Rangkaian clock pada pin 18 dan 19
3. Adanya sumber tegangan Vcc pada pin 40dan gnd di pin 20.
4. Adanya hubungan pin EA ada pada pin 31 ke VCC, karena yang digunakan saat ini adalah program internal, jika yang digunakan eksternal maka EA di hubungkan dengan gnd.

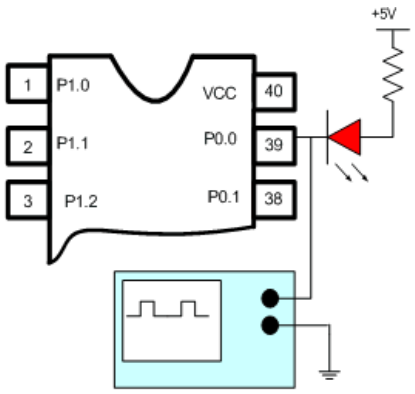
Ada dua cara pemasangan LED, dikenal dengan istilah aktif high dan aktif low Pernyataan aktif high, led akan menyala jika ada arus dari mikrokontroler menuju led, R2 dan akhirnya menuju gnd, artinya program yang diberikan harus diberi logik 1, dengan demikian logik 1 = menyala. Pernyataan aktif low, led menyala pada saat program diberi logic 0

Tugas 8:

Coba kalian gambarkan kembali rangkaian untuk aktif low! Buatkan programnya apakah logik 1 menyala, ataukah di logik 0 yang menyala led nya!

Aplikasi output pada Mikrokontroler

6.1.1 Output Led



Gambar 50 Aplikasi Output ke Led

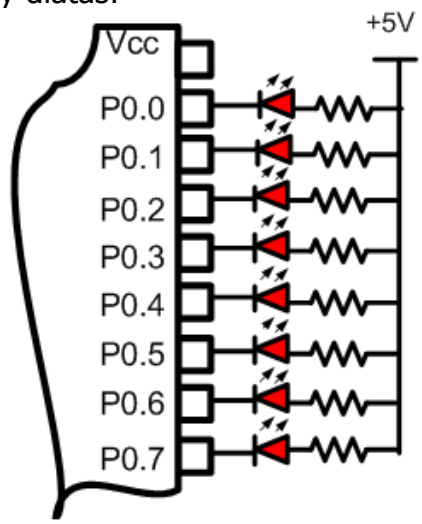
```
org 0h
start: Clr P0.0  ; kirim '0' to P0.0
       call delay; panggil call delay time
       Setb P0.0 ; kirim '1' to P0.0
       call delay; panggil call delay time
       sjmp start; lompat ke start
```

```
;=============================================
=
;subroutine delay created to rise delay time
;=============================================
=
delay: mov R1,#255
del1:  mov R2,#255
del2:  djnz R2,del2
       djnz R1,del1
       ret
end
```

Tugas 10:

Amati gambar di bawah ini !

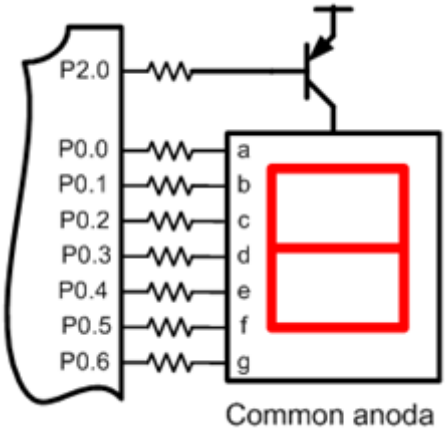
1. Buatkan program agar semua port 0,berkedip,dengan delay yang sama seperti program delay diatas!



Gambar 51 Aplikasi output ke LED

2. Buatkan program running led,geser kiri dan geser kanan,dengan delay yang sama dengan delay diatas.

6.1.2 Output Seven segmen

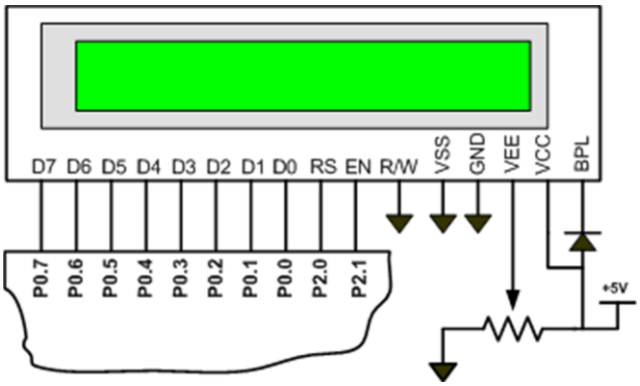


Gambar 52 Aplikasi output ke seven segmen

```
org 0h
    start: mov P2,#11111001b
      clr P0.0
          call delay
          mov P2,#10111011b
      clr P2.0
          call delay
      mov P2,#01110000b
      clr P2.0
      call delay
          sjmp start
;=============================================
    ;subroutine delay created to rise delay time
    ;=============================================
    ==
    delay: mov R1,#255
    del1:  mov R2,#255
    del2:  djnz R2,del2
           djnz R1,del1
           ret
      end
```

Tugas 11 :
Amati angka yang tertera pada seven segmen, buatlah program agar angka yang tampil dari bilangan 0 sampai dengan 9.

6.1.3 Output LCD



Pada aplikasi LCD,perlu diperhatikan adalah inisialisasi antara perangkat keras dan perangkat lunaknya, yaitu hubungan antara pin-pin yang ada di LCd dengan mikrokontroler, juga tampilan display LCD yang dapat diatur melalui program yang dibuat.

Inisialisasi LCD

```
Init_lcd:
   mov r1,#00000001b ;Display clear
   acall write_inst  ;
   mov r1,#00111000b ;Function set,
                     ;Data 8 bit,2 line font 5x7
   acall write_inst  ;
   mov r1,#00001100b ;Display on,
                        ;cursor off,cursor blink off
   acall write_inst
   mov r1,#00000110b ;Entry mode, Set increment
   acall write_inst
   ret

    Write_inst:
       clr P2.0  ; RS = P2.0 = 0, write mode instruction
       mov P0,R1 ; D7 s/d D0 = P0 = R1
       setb P2.1 ; EN = 1 = P2.1
       call delay; call delay time
       clr P2.1  ; EN = 0 = P2.1
       ret

    Write_data:
       setb P2.0 ; RS = P2.0 = 1, write mode data
       mov P0,R1 ; D7 s/d D0 = P0 = R1
```

```
        setb P2.1 ; EN = 1 = P2.1
        call delay; call delay time
        clr p2.1  ; EN = 0 = P2.1
        ret
```

Contoh :

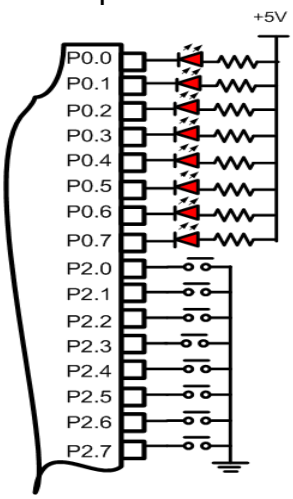
Misalkan kita akan menulis " WELCOME " di LCD .

```
 org 0h
    start: call init_LCD
           mov R1,#80h
           call write_inst
           mov R1,#'w'
           call write_data
           mov R1,#'e'
           call write_data
           mov R1,#'l'
           call write_data
           mov R1,#'c'
           call write_data
           mov R1,#'o'
           call write_data
           mov R1,#'m '
           call write_data
           mov R1,#'e'
           call write_data
    Char:
           sjmp Char
Init_lcd:
   mov r1,#00000001b ;Display clear
   acall write_inst  ;
   mov r1,#00111000b ;Function set,
                     ;Data 8 bit,2 line font 5x7
   acall write_inst  ;
   mov r1,#00001100b ;Display on,
                        ;cursor off,cursor blink off
   acall write_inst
   mov r1,#00000110b ;Entry mode, Set increment
   acall write_inst
   ret
    ;
    Write_inst:
       clr P2.0  ; RS = P2.0 = 0, write mode instruction
       mov P0,R1 ; D7 s/d D0 = P0 = R1
```

```
        setb P2.1 ; EN = 1 = P2.1
        call delay; call delay time
        clr P2.1  ; EN = 0 = P2.1
        ret
;
Write_data:
        setb P2.0 ; RS = P2.0 = 1, write mode data
        mov P0,R1 ; D7 s/d D0 = P0 = R1
        setb P2.1 ; EN = 1 = P2.1
        call delay; call delay time
        clr p2.1  ; EN = 0 = P2.1
        ret
;
delay: mov R0,#0
    delay1:mov R7,#0fh
           djnz R7,$
           djnz R0,delay1
           ret
;
end
```

Aplikasi Input

6.1.4 Input Switch output LED



Gambar 53 Aplikasi Input dan Output

```
org 0h
    CekP20: JB P2.0,CekP21
            call geser kiri
            sjmp CekP20
```

```
CekP21: JB P2.1,CekP20
        call geser kanan
        sjmp CekP20
;=============================================
====
;this subroutine geser kiri
;=============================================
=====
RLeft:  mov A,#11111110b
RLeft1: mov P0,A
        call delay
        JB P2.0,RLeft2
        sjmp EndRLeft
RLeft2: RL A
        sjmp RLeft1
EndRLeft:
        ret
;
;=============================================
======
;this subroutine geser kanan
;=============================================
======
RRight: mov A,#01111111b
RRight1:mov P0,A
        call delay
        JB P2.0,RRight2
        sjmp EndRRight
RRight2:RL A
        sjmp RRight1
EndRRight:
       ret
;=============================================
==
;subroutine delay
;=============================================
==
delay: mov R1,#255
del1:  mov R2,#255
del2:  djnz R2,del2
       djnz R1,del1
       ret
       end
```

Tugas 12

Buatlah program interupsi , program utama Led "**Blink**" di port 0, program interupsi adalah running Led di port 2, Interupsi dari koneksi luar pi port 3.2.

BAB 2

Rangkuman

Mikroprosesor dan komputer memerlukan tempat penyimpanan data dalam biner 1 atau 0, untuk itu diperlukan rangkaian digital yang dapat melakukan tugas tersebut yaitu memori. Mikroprosesor tidak dapat berdiri sendiri tetapi memerlukan komponen pendukung seperti penyimpan data pada rangkaian memori,rangkaian antarmuka yang menghubungkan piranti input dan output, mikroprosesor memerlukan program untuk melakukan pengendaliannya.
Mikrokontroler adalah piranti dimana memori dan komponen input/output telah di kemas menjadi satu sehingga lebih sederhana dan murah.Penggunaan mikroprosesor dan mikrokontroler tergantung dari sistem kendali yang akan dibuat oleh pengguna.

BAB 3

Evaluasi

1. Apa perbedaan fundamental antara mikroprosesor dangan mikrokontroler?
2. Komponen *hardware* apa saja yang dapat ditemui di dalam mikrokontroler pada umumnya, jelaskan fungsi setiap komponen tersebut!
3. Apa fungsi dari pemetaan memori pada mikroprosesor?
4. Ada berapa jenis antarmuka serial yang biasa kita temukan pada mikrokontroler dan jelaskan dimana umumnya digunakan.
5. Jelaskan keunggulan mikrokontroler keluarga 80C51 delapan-bit dibandingkan dengan mikrokontroler lain yang sejenis!
6. Dengan mengacu pada arsitektur internal dari mikrokontroler, bagaimana anda cara membedakan mikrokontroler 8-bit, 16-bit dan 32-bit?
7. Apa perbedaan dasar antara:
   A. Keluarga 80C51 dan keluarga mikrokontrolers 89C51 dengan 89S51;
   B. Keluarga 68HC1 1 dan keluarga mikrokontroler yang lainnya;
   C. Keluarga 80C51 dan keluarga mikrokontrolers 16C84.
8. Jelaskan perbedaan antara antarmuka jenis display LED dangan display LCD yang akan dihubungkan ke mikrokontroler!
9. Antarmuka apa yang diperlukan pada bagian mikrokontroler jika akan di antarmukakan dengan:
   A. keypad;
   B. LED;
   C. Mikrokontroler yang lain.
10. Sebutkan persyaratan apa saja jika mikrokontroler membutuhkan memori ekternal!

BAB 4

Tugas Praktikum

PERCOBAAN 1

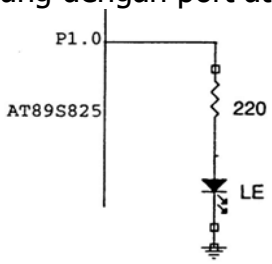
MENGHUBUNGKAN PORT PARALLEL DENGAN DISPLAY LED

A. TUJUAN :

1. Mengetahui rangkaian mikrokontroller untuk menghidupkan dan mematikan LED.
2. Mengetahui program assembly untuk menghidupkan dan mematikan LED.
3. Mengetahui beberapa instruksi assembly dasar, MOV, Setb, Clr, RL dan RR.
4. Mengetahui pembuatan instruksi waktu tunda.

B. INFORMASI

LED (Light Emitting Diode) adalah dioda yang dapat memancarkan sinar jika diberikan tegangan maju, LED ini banyak digunakan sebagai indicator/status /kondisi logika.Untuk menghubungkan LED dengan port pararlel pada mikrokontroler adalah sangat mudah. LED dapat dihubungkan langsung dengan port atau melalui resistor.



Gambar 54 Rangkaian Display LED

Perhatikan pada gambar 1.1 tersebut. Delapan buah LED terhubung ke port 1, yang difungsikan sebagai output. Pada konfigurasi tersebut LED akan nyala bila diberi logika LOW '0' melalui port 0, dan LED akan padam bila diberi logika HIGH '1' melalui port 1.

Latihan 1.1. Instruksi MOV

C. LANGKAH KERJA :

Pada percobaan 1.1 ini LED akan dihidupkan atau dimatikan dengan mengirimkan data tertentu pada port 1.
Untuk melakukan percobaan ini lakukan beberapa langkah sebagai berikut:

1. Hubungkan jumper pada Unit LED Board, untuk mengaktifkan 8 buah LED.
2. Hidupkan board MCS51.
3. Hubungkan Jumper pada ISP unit MCS-51 Board dengan rangkaian programmer.
4. Buka Program M-IDE Studio for MCS-51, sebagai editor dan compiler program.
5. Ketik program berikut ini :

```
Org 0h
Start:    MOV P1,#11100B        ; ISI P1 DENGAN 11100
SJMP start                      ;  lompat ke start
End
```

6. Simpanlah program yang anda ketik dan beri nama : latled1.asm
7. Pada program MIDE tersebut pilih Build /F9 atau untuk melakukan kompilasi program dari *.asm ke *.hex.
8. Lakukan pemrograman mikrokontroller dengan menggunakan Program Progisp 168 Software (Lihatcara mendownload program)

Latihan 1.2 (Led bergeser dengan adanya delay/waktu tunda)

Untuk melakukan percobaan ini lakukan beberapa langkah 1 sampai 8 seperti latihan 1.1

```
  org 000h
main:
  mov   p1,#11111110b     ; port 1 sebagai output dan diisi 0 pada
  bit 0
  lcall   delay500ms
  mov   p1,#11111101b     ; port 1 sebagai output dan diisi 0 pada
  bit 1   lcall   delay500ms
  mov   p1,#11111011b     ; port 1 sebagai output dan diisi 0 pada
  bit 2   lcall   delay500ms
          mov   p1,#11110111b     ; port 1 sebagai output dan diisi 0
  pada bit 3    lcall   delay500ms
```

```
    mov   p1,#11101111b     ; port 1 sebagai output dan diisi 0 pada
    bit 4 lcall   delay500ms
    sjmp  main              ; kembali ke main

;---------------
; delay 500 ms
;---------------
delay500ms:
push   acc
push      00h
mov    a,#032h          ; 500 milli second
x10ms: call   delay10ms
djnz   acc,x10ms
pop       00h
pop    acc
ret

;-------------
; delay 10 ms
;-------------
delay10ms:
push   acc
push   00h

mov    00h,#050h
d10ms1: mov    a,#0c8h
djnz   acc,$
djnz   r0,d10ms1

pop    00h
pop    acc
ret

END
```

D. EVALUASI :

1. Buatlah program dengan kondisi *Output* LED berada pada kondisi:

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Nyala	Nyala	Nyala	Nyala	Padam	Nyala	Padam	Nyala

2. Buatlah program dengan kondisi *Output* LED berada pada kondisi:

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Padam	Padam	Padam	Padam	Nyala	Nyala	Nyala	Nyala

3. Buatlah program dengan kondisi *Output* LED berada pada kondisi:

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Nyala	Padam	Nyala	Padam	Padam	Padam	Nyala	Nyala

4. Buatlah program led yang dapat bergeser ke kanan atau ke kiri !

5. Buatlah program running led !

PERCOBAAN 2

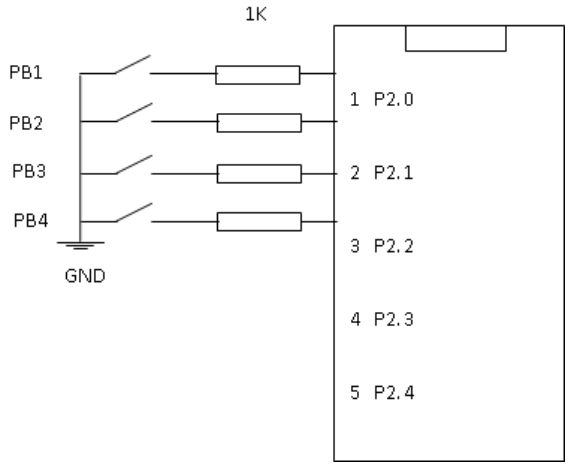
MENGHUBUNGKAN PORT PARALLEL DENGAN SAKLAR PUSH BUTTON

A. TUJUAN :

1. Mengetahui rangkaian mikrokontroller dengan interface ke saklar.
2. Mengetahui program assembly untuk mengambil data saklar dan mengeluarkan data ke LED.
3. Mengetahui beberapa instruksi assembly dasar, MOV, Setb, Clr, RL dan RR.

B. INFORMASI

Menghubungkan port parallel dengan switch (saklar).
Karena port parallel dapat bersifat bi-directional, karena itu dapat digunakan sebagai port masukan atau sebagai port keluaran. Secara Hardware menghubungkan port parallel masukan dengan port parallel keluaran adalah sama, yang membedakannya pada program (software).



Gambar 55 Rangkaian Interface Push Button

Pada gambar 2.tersebut tampak rangkaian push button, bila saklar ditekan maka port sesuai dengan bit tersebut akan mendapat logika low '0' dan sebaliknya bila saklar tidak ditekan maka port tersebut akan mendapat logika high '1'.

Latihan 2.1. Ambil Data Saklar

C. LANGKAH KERJA:

Pada percobaan ini, LED akan nyala bila saklar ditekan sesuai dengan bit tersebut.

Untuk melakukan percobaan ini lakukan beberapa langkah sebagai berikut:

1. Hidupkan board MCS51
2. Hubungkan Jumper P2 dengan Switch
3. Hubungkan Jumper pada ISP unit MCS-51 Board dengan rangkaian programmer.
4. Buka Program M-IDE Studio for MCS-51, sebagai editor dan compiler program.
5. Ketik program berikut ini !

```
        Org 0h
Start: Mov A, P2 ;Ambil data dari P0 dan Simpan ke A
        Mov P1, A ;Kirim data A ke P1
sjmp start
end
```

6. Simpanlah program yang anda ketik dan beri nama : latswitch1.asm
7. Pada program MIDE tersebut pilih Build /F9 atau untuk melakukan kompilasi program dari *.asm ke *.hex.
8. Lakukan pemrograman mikrokontroller dengan menggunakan Program ISP Software (Lihat cara mendownload program)
9. Lakukan pengamatan pada LED.

Saklar	Kondisi LED yang Nyala (D1-D2-D3-D4-D5-D6-D7-D8)
PB1	
PB2	
PB3	
PB4	

Latihan 2.2.

Untuk melakukan percobaan ini lakukan beberapa langkah 1 sampai 9 seperti latihan 2.1.

Contoh program untuk mengetahui posisi saklar secara terus menerus dan keadaan saklar ditandai dengan led yang menyala di mulai PB1 samapai dengan PB4

```
    ORG 00h
Mulai:      Mov A, P2       ; baca P1 dan masukkan
                            ; isinya ke register A
```

```
            cjne A, #01, terus1          ; bandingkan  A dengan 01
            mov P1, #01h          ; menyalakan led 1
terus1:     cjneA,#02,terus2             ; bandingkan A dengan 02
                  mov P2, #02h           ; menyalakan led2
terus2:     …….                          ; bandingkan A dengan 04
    dst
    sjmp mulai
    End
```

D. EVALUASI

A. Buatlah program untuk menampilkan LED di *Port*1 dengan PB di *Port*2 dengan syarat:
Jika *PB*"Bit 0" berlogika '0', semua LED padam
Jika *PB*"Bit 1" berlogika '1', semua LED menyala

B. Buat program kombinasi penggunaan port input dan output dengandeskripsi sbb :
terdapat dua buah saklar S1 dan S2 yang masing-masing dipasang pada P3.0 dan P3.1terdapat satu buah mesin disimulasikan dengan led yang terpasang pada P1.0 .jika S1 ditekan (ON) maka mesin hidup, sedangkan jika S2 ditekan (ON) mesin mati

PERCOBAAN 3

MENGHUBUNGKAN PORT PARALLEL DENGAN RELAY

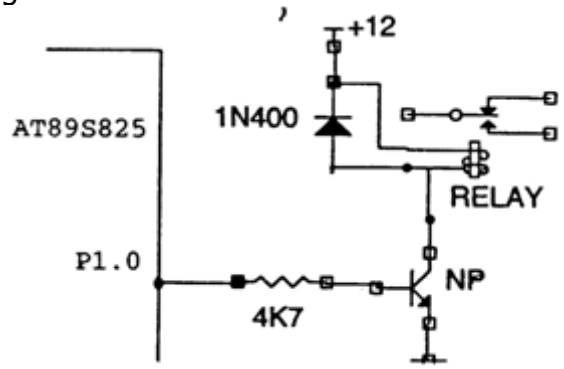
A. TUJUAN :

1. Mengetahui rangkaian mikrokontroller dengan interface ke relay.
2. Mengetahui program assembly mengeluarkan data ke relay.
3. Mengetahui beberapa instruksi assembly dasar, MOV, Setb, Clr, RL dan RR.

B. INFORMASI

Relay adalah peralatan elektronika yang sering digunakan. Hal ini karena relay dapat melakukan pensaklaran dengan arus/tegangan yang besar. Sebagai contoh jika anda ingin menyalakan bohlam 220V dirumah anda dengan kontrol dari mikrokontroler, maka salah satu jalan termudah adalah dengan menggunakan relay.

Cara menghubungkan rangkaian relay dengan port mikrokontroler di jelaskan pada gambar berikut :



Relay dengan LED dalam hal ini adalah sama yaitu sebagai alat keluaran (output device). Karena hal itu maka cara untuk mengaktifkan relay akan sama dengan cara mengaktifkan LED.

Latihan 3.1. Ambil Data Saklar di keluarkan melalui relay

C. LANGKAH KERJA

Pada percobaan ini, relayakanbekerja bila saklar ditekan sesuai dengan bit tersebut.

Untuk melakukan percobaan ini lakukan beberapa langkah sebagai berikut :

1. Hubungkan Unit MCS-51 Board dengan power supply.
2. Hubungkan Unit relay Board dengan MCS 51 di Port 2.1 dan hubungkan unit switch board di port 2.0.

3. Hubungkan Jumper pada ISP unit MCS-51 Board dengan rangkaian programmer.
4. Buka Program M-IDE Studio for MCS-51, sebagai editor dan compiler program.
5. Ketik program berikut ini :

```
        Org 0h
Start:
   Setb p2.0 ;persiapkan port  P2 sebagai masukan
   Clr P2.1 ;matikan relay
Cekport:
   Jb P2.0,Cekport;apakah  P2.0 rendah?
   Setb P2.1 ;aktifkan relay
  End
```

6. Simpanlah program yang anda ketik dan beri nama :relay1.asm
7. Pada program MIDE tersebut pilih Build /F9 atau untuk melakukan kompilasi program dari *.asm ke *.hex.
8. Lakukan pemrograman mikrokontroller dengan menggunakan Program ISP Software. (Lihat cara mendownload program)
9. Lakukan pengamatan pada relay.

D. EVALUASI :

Buatlah program relay aktif di port 3 dan input switch di port 0,dengan ketentuan semua PB t dapat mengaktifkan output relay.

PERCOBAAN 4

MENGHUBUNGKAN PORT PARALLEL DENGAN DISPLAY 7 SEGMEN

A. TUJUAN :

1. Mengetahui rangkaian interface mikrokontroller dengan 7 segmen.
2. Mengetahui program assembly untuk menampilkan data ke 7 segment.
3. Mengetahui beberapa instruksi assembly dasar, MOV, Setb, Clr, dan waktu tunda.

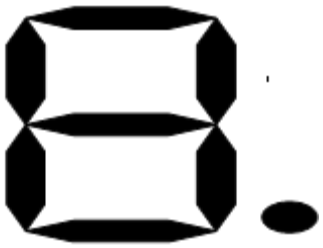
B. INFORMASI

Menghubungkan port parallel dengan seven Segment Rangkaian Seven Segment adalah rangkaian yang menampilkan tampilan numeric ataupun alphabet.LED berfungsi mengubah arus listrik menjadi cahaya, sehingga untuk menyinari ssalah satu segmen dari tampilan, arus harus diarahkan ke diode dari segment yang dimaksud.Untuk aplikasi seven segment ini digunakan konfigurasi seven segment Common Anode.Setiap tampilan Seven Segment membentuk satu digit dari tampilan banyak dogit yang lengkap.Dengan demikian, setiap digit mempunyai delapan terminal, satu untuk setiap segmen dan satu untuk sambungan bersama.Dalam beberapa aplikasi, sering ditambahkan titik desimal, sehingga terdapat sembilan terminal.

P0.6	P0.5	P0.4	P0.3	P0.2	P0.1	P0.0	Display
g	f	e	d	c	b	a	
1	0	0	0	0	0	0	0
1	1	1	1	0	0	1	1
0	1	0	0	1	0	0	2
0	1	1	0	0	0	0	3
:	:	:	:	:	:	:	:
0	0	0	1	0	0	0	A
0	0	0	0	0	1	1	b

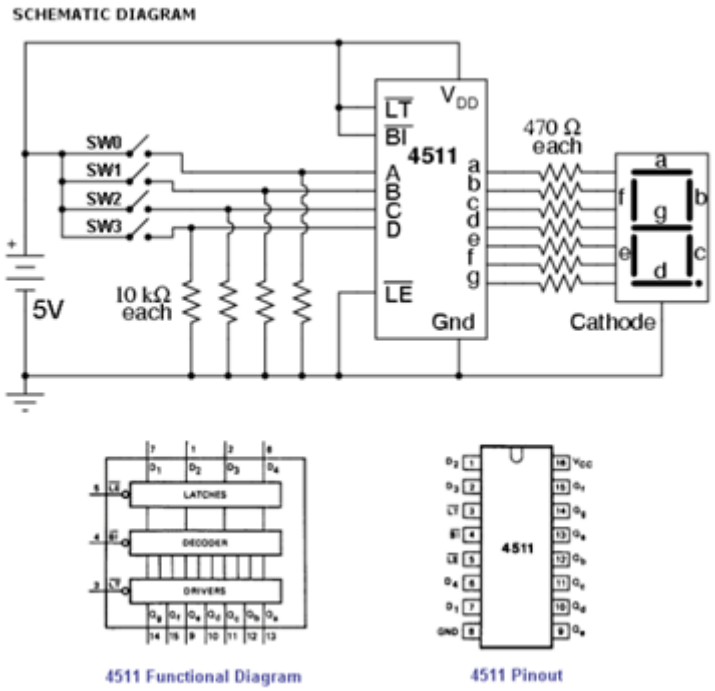
Tabel 2 Data Display 7 Segmen

Pada tabel tersebut tampak bahwa untuk menghidupkan sebuah segmen, harus dikirimkan data logika low ”0” dan sebaliknya untuk mematikan segmen, harus dikirimkan data logika high ”1”.



Gambar 56 Tampilan Seven Segmen

Pada trainer mikrokontroler MCS 51 ini,seven segmen dudah dihubungkan dengan IC decoder CD 4511,sehingga inputan ke seven segmen hanya 4 bit,yaitu A,B, C dan D. Untuk lebih jelas dapat dilihat pada gambar di bawah ini :



Gambar 57 Skematik Decoder CD4511 dengan Seven Segmen

Dari gambar diatas SW0,SW1,SW2 dan SW3 adalah output dari Decoder yang diberi inisialisai A,B,C dan D yang dihubungkan dengan port mikrokontroler.

Percobaan 4.1.Menampilkan angka pada 7 Segmen

C. LANGKAH KERJA

Pada percobaan ini, akan menampilkan angka dari mulai angka 0 sampai dengan 9 pada Seven Segmen Display
Untuk melakukan percobaan ini lakukan beberapa langkah sebagai berikut:

1. Hubungkan Unit MCS-51 Board dengan power supply.
2. Hubungkan Unit Seven segment Board dengan MCS 51 di Port 0.
3. Hubungkan Jumper pada ISP unit MCS-51 Board dengan rangkaian programmer.
4. Buka Program M-IDE Studio for MCS-51, sebagai editor dan compiler program.
5. Ketik program berikut ini:

```
;----------------------------------------
; Program 7 Segmen menggunakan IC BCD 4511
;----------------------------------------
    Org  000h
Main:
MOV       P0,#00000000B              ; Port 0 sebagai Output yang
    dipakai
                              Untuk koneksi A,B,C dan D
    LCALL DELAY500MS
    MOV   P0,#00000001B
    LCALL DELAY500MS
    MOV   P0,#00000010B
    LCALL DELAY500MS
    MOV   P0,#00000011B
    LCALL DELAY500MS
    MOV   P0,#00000100B
    LCALL DELAY500MS
    MOV   P0,#00000101B
    LCALL DELAY500MS
    MOV   P0,#00000110B
    LCALL DELAY500MS
    MOV   P0,#00000111B
    LCALL DELAY500MS
    MOV   P0,#00001000B
    LCALL DELAY500MS
    MOV   P0,#00001001B
    LCALL DELAY500MS
      SJMP        Main                 ; Kembali ke Main

;---------------
; delay 500 ms
```

```
;--------------
DELAY500MS:
        PUSH    ACC
        PUSH        00H
        MOV     A,#010H          ; 500 milli second

X10MS:  CALL    DELAY10MS
        DJNZ    ACC,X10MS

        POP 00H
        POP     ACC
        RET

;-------------
; DELAY 10 ms
;-------------
DELAY10MS:
        PUSH    ACC
        PUSH    00H

        MOV     00H,#050H
D10MS1: MOV     A,#0C8H
        DJNZ    ACC,$
        DJNZ    R0,D10MS1

        POP     00H
        POP     ACC
        RET

END
```

D. EVALUASI

Buatlah program dengan menggunakan Switchdi port0 dan seven segment di port 2. Pada saat pertama kali program dijalankan, maka seven segment akan manampilkan angka "0".

Jika switch pertama ditekan, maka seven segment akan menampilkan angka "1"

Jika switch kedua ditekan, maka seven segment akan menampilkan angka "2"

Jika switch ketiga ditekan, maka seven segment akan menampilkan angka "3"

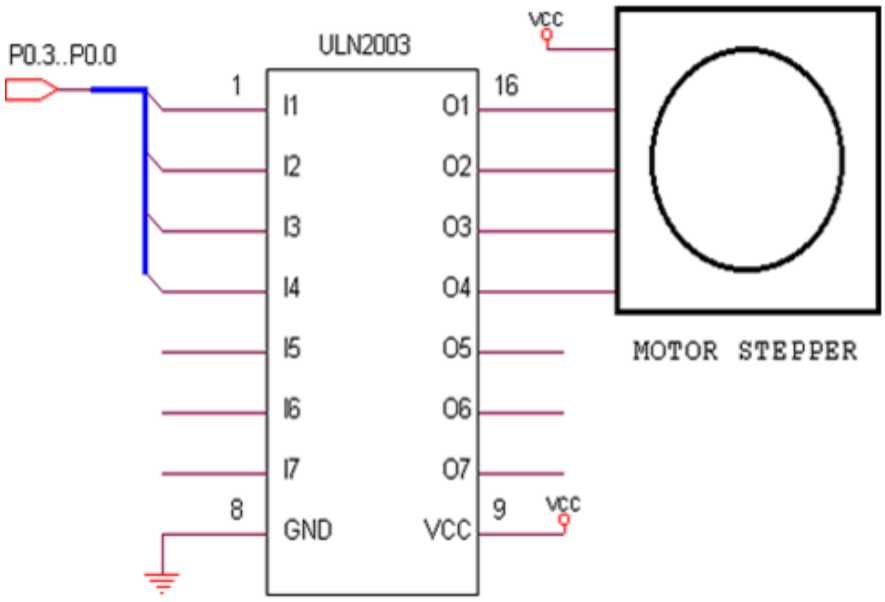
Jika switch keempat ditekan, maka seven segment akan menampilkan angka "4".

PERCOBAAN 5

MENGHUBUNGKAN PORT PARALLEL DENGAN MOTOR STEPPER

A. TUJUAN :

1. Memahami rangkaian interface mikrokontroller dengan motor stepper.
2. Mengetahui rangkaian driver motor stepper ULN2003.
3. Mengetahui bahasa assembly untuk mengatur arah putaran motor stepper.
4. Mengetahui bahasa assembly untuk mengatur arah putaran motor stepper dengan menggunakan saklar.

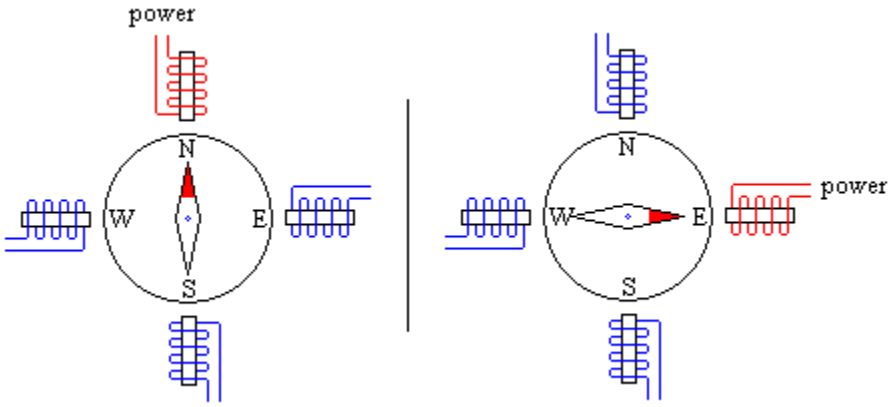


Gambar57 Rangkaian interface motor stepper dengan driver ULN2003

B. INFORMASI

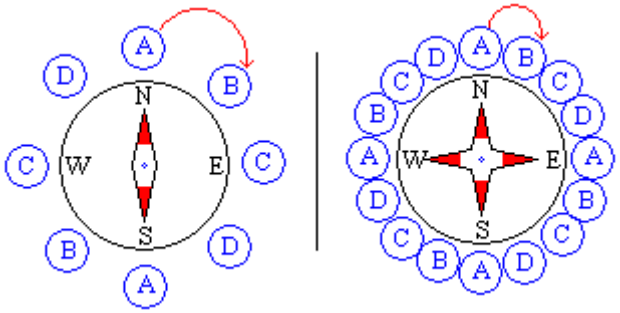
Pada Motor DC biasa, akan berputar dan berputar terus selama power supply ada. Tidak ada rangkaian cerdas tertentu yang diperlukan untuk mengendalikan motor tersebut, kecuali hanya memperlambat putaran atau membalik putaran, dengan menerapkan polaritas balik. Motor stepper adalah sangat berbeda. Jika anda memberikan power pada motor ini, maka motor ini akan berada dalam keadaan diam, agar motor dapat berputar, anda harus merubah sinyal yang masuk ke motor. Sebagai ilustrasi, dapat dibayangkan sebuah kompas dengan elektromagnet disekitarnya.

Sebagaimana digambarkan pada gambar 5.1., apabila power yang diberikan pada elektromagnet diganti, maka akan merubah posisi jarum dari kompas.



Gambar 58 Ilustrasi sebuah kompas dengan elektromagnet

Dengan empat buah elektromagnet maka gerakan akan melompat secara kasar. Sekarang bayangkan susunan yang sama dengan 100 elektromagnet yang mengitari kompas. Dengan mangatur energi yang mengalir pada setiap elektromagnet dalam berurutan, maka jarum akan memerlukan sebanyak 100 langkah untuk melakukan satu kali putaran. Tetapi dengan pengaturan 100 elektromagnet secara individu, akan memerlukan elektronika yang kompleks.



Gambar 59 Ilustrasi motor stepper dengan jarum kompas dengan elektromagnet

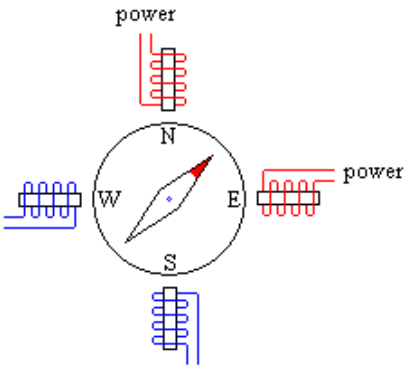
Pada ilustrasi tersebut, huruf-huruf yang melingkar mewakili elektromagnet. Semua magnet dengan huruf yang sama berada dalam keadaan koneksi. Ketika anda memberi arus pada rangkaian tersebut, maka semua elektromagnet dengan huruf yang sama akan

on pada saat itu, untuk menggerakkan kompas, maka elektromaget berikutnya harus dialiri arus, sehingga akan menimbulkan gerakan.

Tabel 3 Full Step Mode

A	B	C	D	KOMENTAR
1	0	0	0	Take a step clock wise
0	1	0	0	another step clock wise
0	0	1	0	another step clock wise
0	0	0	1	another step clock wise
0	0	0	1	No step take
0	0	1	0	Take a step back

Mode Half Steps
Dengan menghidupkan dua koil pada waktu yang bersamaan maka motor akan berada dalam posisi diantaranya.



Gambar 60 Half step mode

Tabel 4 Half Step Mode

A	B	C	D	KOMENTAR
1	0	0	0	Take a step clock wise
1	1	0	0	Half a step clock wise

0	1	0	0	The complete full step clock wise
0	1	1	0	another half step clock wise
0	0	1	0	The complete full step clock wise
0	0	1	1	Another half step clock wise
0	0	0	1	The complete full step clock wise
1	0	0	1	another half step clock wise
1	0	0	0	Start position

Gambar 61 Bentuk fisik motor stepper disk drive 1,2”

Percobaan 5.1. Penggerak Motor Putar Searah Jarum Jam

Pada percobaan ini, motor stepper akan berputar searah jarum jam, terus menerus.
Untuk melakukan percobaan ini lakukan beberapa langkah sebagai berikut:

1. Hubungkan kabel parallel antara P0 dengan motor stepper.
2. Hubungkan modul Mikrokontroler Trainer dengan power supply.
3. Hubungkan modul Mikrokontroler Trainer dengan rangkaian programmer.
4. Buka Program M-IDE Studio for MCS-51, sebagai editor dan compiler program.
5. Ketik program berikut ini :

```
org 0h
start: call StepCW
sjmp start
StepCW:
mov P0,#11101111b ; Turn on driver 1
call delay ; call delay time
mov P0,#11011111b ; Turn on driver 2
call delay ; call delay time
mov P0,#10111111b ; Turn on driver 3
call delay ; call delay time
mov P0,#01111111b ; Turn on driver 4
call delay ; call delay time
ret
StepCCW:
mov P0,#01111111b ; Turn on driver 1
call delay ; call delay time
mov P0,#10111111b ; Turn on driver 2
call delay ; call delay time
mov P0,#11011111b ; Turn on driver 3
call delay ; call delay time
mov P0,#11101111b ; Turn on driver 4
call delay ; call delay time
ret
        ;
delay: mov R0,#255
 delay1:mov R2,#255
djnz R2,$
djnz R0,delay1
ret
end
```

6. Simpanlah program yang anda ketik dan beri nama : prog81a.asm
7. Pada program MIDE tersebut pilih Build /F9 atau untuk melakukan kompilasi program dari *.asm ke *.hex.
8. Lakukan pemrograman mikrokontroller dengan menggunakan Program ISP Software (Lihat cara mendownload program).
9. Lakukan pengamatan arah putaran motor stepper apakah sudah sesuai ?

PERCOBAAN 6

I N T E R U P S I

A. TUJUAN :

1. Mengetahui sistem interupsi pada mikrokontroller.
2. Dapat menerapkan sistem interupsi pada pembuatan jam digital.
3. Dapat Mengetahui penggunakan bahasa assembly untuk penggunakan sistem interupsi.

B. INFORMASI

Pada mikrokontroller menyediakan 5 buah sumeber interupsi, 2 buah interupsi eksternal, 2 buah interupsi timer, dan 1 buah interupsi serial. Agar interupsi dapat dilayani maka instruksi assembly harus ditempatkan pada alamat vektor berikut, sesuai dengan sumber interupsi yang akan digunakan.

Source	Vector Address
IE0	0003H
TF0	000BH
IE1	0013H
TF1	001BH
RI + TI	0023H

```
Org 0000h
        Ljmp Start
        Org 000bh
        Ljmp Timer0Interupt
Start: ; Instruksi Rutinitas
        ; Instruksi Rutinitas
        ; Instruksi Rutinitas
        Sjmp Start; { Looping Forever }
        ;
Timer0Interupt:
        :
        :
```

```
Reti
End
```

Pada contoh instruksi pemrograman tersebut tampak, apabila tidak ada interupsi maka program akan menuju ke start dan menjalankan rutinitas-rutinitas secara terus menerus, tetapi apabila suatu interupsi yang dibangkitkan oleh overflow timer 0 terjadi, maka program yang semula bersarang pada rutinitas akan melompat pada alamat vektor 0bh (alamat interupsi timer 0) dan melompat ke subrutine interupsi Timer0Interupt.

Pada percobaan ini, untuk menunjukkan cara kerja sela ini maka berikut adalah program sederhana dimana ada tiga buah LED. LED1 akan menyala yang menunjukkan mikrokontroler bekerja secara biasa (normal task). LED2 akan menyala sesaat jika INT0 sedang aktif yaitu jika saklar di tekan (Int0 task). LED3 akan menyala sesaat jika INT1 sedang aktif yaitu jika saklar INT1 ditekan (Int1 task).
Untuk melakukan percobaan ini lakukan beberapa langkah sebagai berikut:

1. Hubungkan kabel parallel antara P3.2 dengan Switch sebagai INT0, P3.3 sebagai INT1.
2. Hubungkan modul Mikrokontroler Trainer dengan power supply +5V.
3. Hubungkan modul Mikrokontroler Trainer dengan rangkaian programmer.
4. Buka Program M-IDE Studio for MCS-51, sebagai editor dan compiler program.
5. Ketik program berikut ini:

```
;Program untuk menunjukkan kerja sela INT0 dan INT1
;LED1 (p1.0) akan menyala untuk normal task
;LED2 (p1.1) akan menyala jika ada interupsi INT0
:LED3 (p1.2) akan menyala jika ada interupsi INT1
;=====================================
Org 0h
Ajmp lanjut
Org 03h
Ajmp int0
Org 13h
Ajmp int1

 Lanjut:
Clr p1.1            ;matikan dulu led2
Clr p1.2            ;matikan dulu led3
Setb  ea            ;aktifkan bit global interupsi
```

```
Setb  ex0
Setb  ex1
Setb p1.0

    Int0:
Setb p1.1           ;aktifkan led2
Call delay
Clr p1.1            ;matikan led2
Reti                ;keluar dari interupsi

    Int1:
Setb p1.2           ;aktifkan led2
Call delay
Clr p1.2            ;matikan led2
Reti                ;keluar dari interupsi

  Delay:
Clr  rs0
Clrs rs1
Mov r7,#0h
 Ulang2:      mov r6,#0h
 Ulang1:      mov r5,#0h
 Ulang:
Inc     r5
Cjne    r5,#30h,ulang
Inch    r6
Cjne r6,#30h,ulang1
Inch  r7
Cjne r7,#30h,ulang2
Ret
END
```

6. Simpanlah program yang anda ketik dan beri nama : latsegmen.asm
7. Pada program MIDE tersebut pilih Build /F9 atau untuk melakukan kompilasi program dari *.asm ke *.hex.
8. Lakukan pemrograman mikrokontroller dengan menggunakan Program ISP Software (Lihat cara mendownload program).
9. Lakukan pengamatan pada LED.

C. EVALUASI

Buatlah program untuk menampilkan LED di Port 1 dengan *toggle switch* di Port 3.2 untuk kondisi:

Pada kondisi awal, LED "Bit 7"," Bit 6", " Bit 1", dan "Bit 0" menyala.
Interrupt trigger bersifat *falling edge*.
Jika terjadi interrupt, LED "Bit 5"," Bit 4", " Bit 3", dan "Bit2" menyala sejenak.
Setelah *interrupt* selesai, LED kembali pada kondisi awal.

PERCOBAAN 7
T I M E R / COUNTER

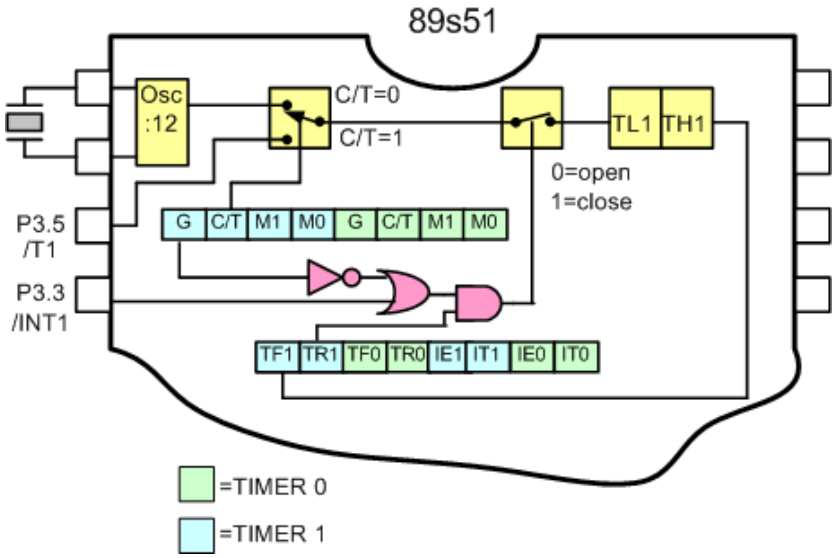
A. TUJUAN

1. Mengetahui fungsi timer dan counter pada mikrokontroller.
2. Mengetahui rangkaian interface untuk aplikasi timer dan counter.
3. Dapat memanfaatkan fungsi counter untuk mencacah pulsa.
4. Dapat memanfaatkan fungsi timer untuk membangkitkan clock dengan periode tertentu.

B. INFORMASI

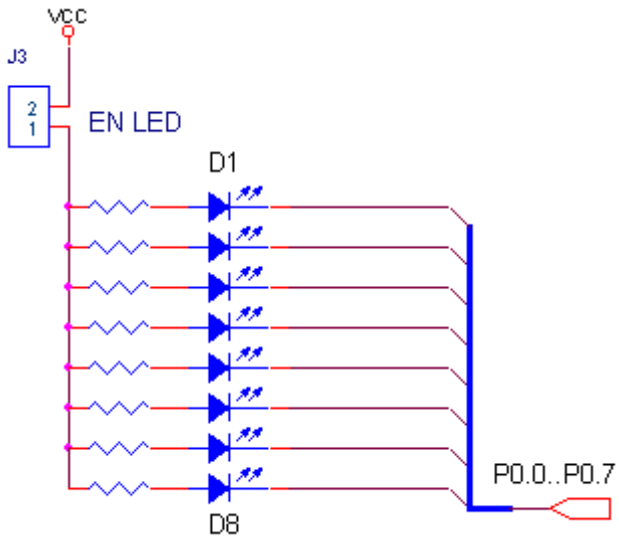
FUNGSI COUNTER

Pada keluarga MCS-51 terdapat dua buah timer/ counter 16 bit, yang dapat dikonfigurasikan sebagai timer atau counter, 8 bit, 13 bit atau 16 bit sesuai dengan mode yang dipilih. Gambar berikut merupakan contoh aplikasi Counter 8 bit dengan menggunakan mode 3



Gambar 62 Aplikasi Mode 3 sebagai counter 8 bit dengan output LED

Pada percobaan ini, pulsa diambil melalui clock generator yang dibangkitkan oleh IC 555 yang mempunyai frekuensi 1 Hz, hasil cacahan biner pada register counter akan ditampilkan pada LED.



Lakukan beberapa langkah percobaan sebagai berikut :

1. Hubungkan 1 buah kabel antara P3.5 dengan output clock IC 555.
2. Hubungkan jumper konektor LED_EN.
3. Hubungkan modul Mikrokontroler Trainer dengan power supply.
4. Hubungkan modul Mikrokontroler Trainer dengan rangkaian programmer.
5. Buka Program M-IDE Studio for MCS-51, sebagai editor dan compiler program.
6. Ketik program berikut ini :

```
org                                    0h
Start: Mov TMOD,#00000111b      ; mode 3 counter 8 bit
timer                                  0
       Setb TR0                 ; TR1 = 0, start
counting
Get:   Mov A, TL0               ; A = TL0
       CPL A                    ; A = NOT A
       Mov P0, A                ; P0 = A
       Sjmp Get                 ; Looping Forever
       End
```

7. Simpanlah program yang anda ketik dan beri nama : prog91a.asm
8. Pada program MIDE tersebut pilih Build /F9 atau untuk melakukan kompilasi program dari *.asm ke *.hex.
9. Lakukan pemrograman mikrokontroller dengan menggunakan Program ISP Software (Lihat cara mendownload program)
10. Lakukan pengamatan pada LED ?dan lengkapi tabel berikut.

INPUT Clock	DISPLAY LED							
	D1	D2	D3	D4	D5	D6	D7	D8
1								
2								
3								
4								
:								
20								

C. EVALUASI

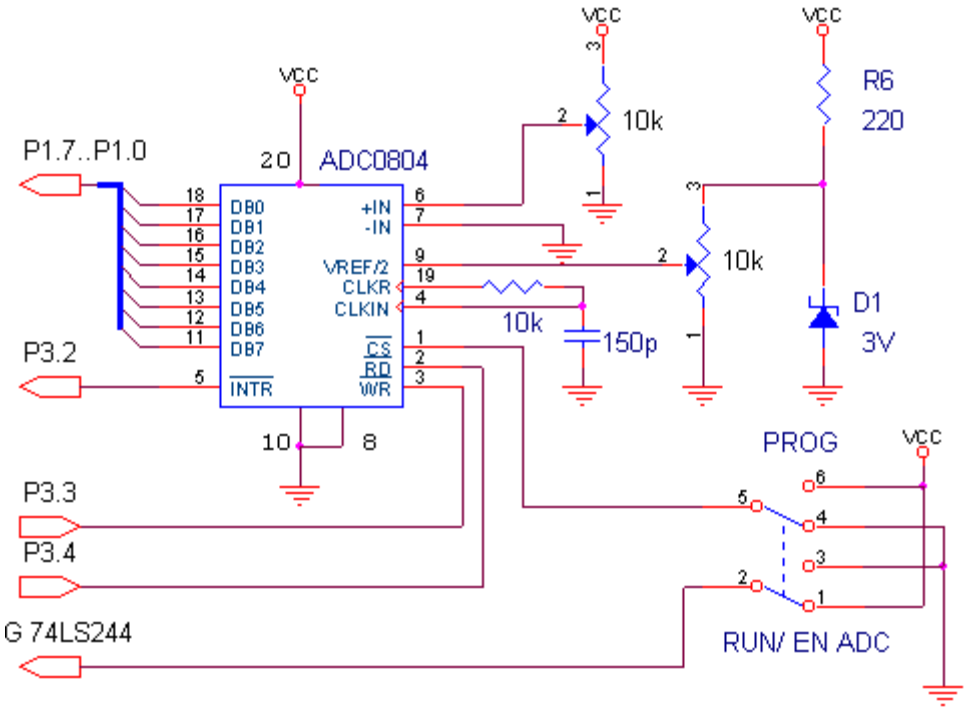
1. Buatlah program untuk menampilkan LED di *Port*1 dengan syarat :
   Semua LED dimulai dari keadaan padam.
   Gunakan *Timer 0 Sebagai timer mode* 0, dimulai dari '0000h.
   Setiap kali terjadi *interrupt*, LED akan menampilkan bilangan biner yang bertambah (mulai dari '00000000b', '00000001', '000000010b', 00000011b', …,'11111101b', '11111110b', '111111111b'. lalu kembali lagi ke '000000000b').
   Masing-masing *interrupt* hanya akan memapilkan satu nlai(*interrupt* pertama menampilkan '000000000b', *interrupt* kedua menampilkan '000000001' dst).

2. Buatlah program untuk menampilkan LED di *Port*1 dengan syarat:
   Semua LED dimulai dari keadaan padam.
   Gunakan *timer*1*sebagai timer mode* 0, dimulai dari '0000h.
   Jika di tekan switch INT0 maka LED akan berkedip 1 detik

PERCOBAAN 8
ANALOG TO DIGITAL CONVERTER (ADC)
PERCOBAAN 8
ANALOG TO DIGITAL CONVERTER (ADC)

A. TUJUAN :

1. Mengetahui rangkaian interface mikrokontroller dengan ADC 0804.
2. Mengetahui setting tegangan referensi Vref ADC0804.
3. Mengetahui perhitungan tegangan resolusi ADC0804.
4. Dapat Mengetahui program assembly untuk menampilkan data ADC ke 7 Segmen.
5. Dapat Mengetahui program assembly untuk menampilkan data ADC ke LCD Karakter 2 x 16



Gambar 63 Rangkaian ADC0804

B. DASAR TEORI

Konverter A/D tersedia secara komersial sebagai rangkaian terpadu dengan resolusi 8 bit sampai dengan 16 bit. Pada percobaan ini akan memperkenalkan ADC0801, yaitu sebagai sebuah konverter A/D 8 bit yang mudah diinterfacekan dengan sistem mikrokontroller. A/D ini menggunakan metode approksimasi berturut-turut untuk mengkonversikan masukan analog (0-5V) menjadi data digital 8 bit yang ekivalen.ADC0801 mempunyai pembangkit clock internal dan memerlukan catu daya +5V dan mempunyai waktu konversi optimum sekitar 100us.

Diagram konfigurasi pin ADC0804 ditunjukkan pada gambar 5.2. Pin 11 sampai 18 (keluaran digital) adalah keluaran tiga keadaan, yang dapat dihubungkan langsung dengan bus data bilamana diperlukan. Apabila CS (pin 1) atau RD (pin2) dalam keadaan high ("1"), pin 11 sampai 18 akan mengambang (high impedanze), apabila CS dan RD rendah keduanya, keluaran digital akan muncul pada saluran keluaran.
Sinyal mulai konversi pada WR (pin 3).Untuk memulai suatu konversi, CS harus rendah.Bilamana WR menjadi rendah, konverter akam mengalami reset, dan ketika WR kembali kepada keadaan high, konversi segera dimulai.

Konversi detak konverter harus terletak dalam daereh frekuensi 100 sampai 800kHz. CLK IN (pin 4) dapat diturunkan dari detak mikrokontroller, sebagai kemungkinan lain, kita dapat mempergunakan pembangkit clock internal dengan memasang rangkaian RC antara CLN IN (pin 4) dan CLK R (pin 19). Pin 5 adalah saluran yang digunakan untuk INTR, sinyal selesai konversi.

INTR akan menjadi tinggi pada saat memulai konversi, dan akan aktiv rendah bila konversi telah selesai. Tepi turun sinyal INTR dapat dipergunakan untuk menginterupsi sistem mikrokontroller, supaya mikrokontroller melakukan pencabangan ke subrutine pelayanan yang memproses keluaran konverter.Pin 6 dan 7 adalah masukan diferensial bagi sinyal analog. A/D ini mempunyai dua ground, A GND (pin 8) dan D GND (pin10). Kedua pin ini harus dihubungkan dengan ground. Pin 20 harus dihubungkan dengan catu daya +5VA/D ini mempunyai dua buah ground, A GND (pin 8) dan D GND (pin 10). Keduanya harus dihubungkan dengan catu daya, sebesar +5V.Pada A/D 0804 merupakan tegangan referensi yang digunakan untuk offset suatu keluaran digital maksimum. Dengan persamaan sebagai berikut:

$$V_{REF} = \frac{1}{2} V_{IN} maks$$

$$V_{RESOLUSI} = \frac{V_{IN} MAKS}{255}$$

Misalnya anda menginginkan masuk analog maksimum sebesar 4 V, maka :

Vref=0.5 x 4 = 2 volt

Resolusi ini mempunyai arti sebagai berikut:

Vin (volt)	Data Digital (biner)	Data Digital (desimal)
0,000	0000 0000	
0,0156	0000 0001	
0,0313	0000 0010	
4	1111 1111	255

A/D ini dapat dirangkai untuk menghasilkan konversi secara kontinu.Untuk melaksanakannya, kita harus menghubungkan CS, dan RD ke ground dan menyambungkan WR dengan INTR seperti pada gambar dibawah ini. Maka dengan ini keluaran digital yang kontinu akan muncul, karena sinyal INTR menggerakkan masukan WR. Pada akhir konversi INTR berubah menjadi low, sehingga keadaan ini akan mereset konverter dan mulai konversi.

Tabel 5 Koneksi Interface ADC ke Mikrokontroller

ADC	Port Mikrokontroller
/INTR	P3.2
/WR	P3.3
/RD	P3.4
D0 s/d D7	P1.0 s/d P1.7

TABEL 6 Instruksi logika pada pin kontrol A/D 0804

INPUT			OUTPUT	KEGIATAN
/WR	/RD	/INTR	DO S/D D7	Hi-Z (High Impedansi)
1	1	1	-	-

0	1	1	Hi-Z	Reset
1	1	1	Hi-Z	-
1	1	0	Hi-Z	Konversi Selesai
1	0	1	Data Out	Data Ready

Percobaan 8.1. ADC0804 dan Display ke 7 Segmen

```
;=========================================
   ==========
; PROGRAM INI DIBUAT UNTUK LATIHAN MENGUNAKAN
;   ADC 8 BIT& KONVERSI DALAM BENTUK BINER
;=========================================
   =========

;----------------------
;    BUFFER DATA
;----------------------
DATA_ADC   EQU   75H

;=======================
;      MAIN PROGRAM
;=======================
      ORG    0000H
      LJMP   START
      ORG    0050H
START:

      MOV    SP,#30H
      MOV    DATA_ADC,#00H

EN:

      LCALL  READ_ADC       ; AMBIL DATA ADC
      MOV    A,DATA_ADC
   MOV  P2,A           ; KONVERSI DATA DARI ADC TO BINER
      SJMP   EN
;---------------------------
; PROSEDUR PEMBACAAN DATA ADC
;---------------------------
; P3.1 = WR
; P3.0 = RD
; P3.2 = INTR

READ_ADC:
         PUSH     ACC
```

```
        PUSH    PSW

        SETB    P3.1        ; WR = 1
        SETB    P3.0        ; RD = 1

        CLR     P3.1        ; WR = O
        SETB    P3.0        ; RD = 1

        SETB    P3.1        ; WR = 1
        SETB    P3.0        ; RD = 1

        MOV     P0,#0FFH            ; MAKE THE P0 HIGH


WAITINTR: MOV   C,P3.2      ; TUNGGU INTR
        JC      WAITINTR
        SETB    P3.1        ; WR = 1
        CLR     P3.0        ; RD = 0

        MOV     A,P0
        MOV     DATA_ADC,A      ; AMBIL DATA ADC
        SETB    P3.1        ; WR = 1
        SETB    P3.0        ; RD = 1

        POP     PSW
        POP     ACC
        RET
```

Percobaan ini lakukan beberapa langkah sebagai berikut:

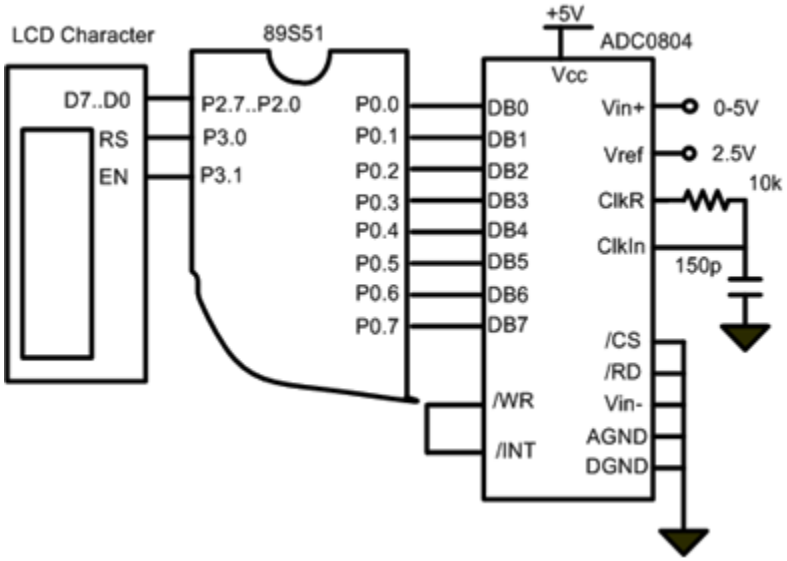
1. Pada saat langkah pemrograman posisikan saklar togle ke posisi PROG.
2. Posisikan saklar togle ke RUN untuk mengaktifkan ADC0804 CS=0
3. Hubungkan modul Mikrokontroler Trainer dengan power supply +5V.
4. Hubungkan modul Mikrokontroler Trainer dengan rangkaian programmer.
5. Buka Program M-IDE Studio for MCS-51, sebagai editor dan compiler program.
6. Ketik program berikut ini :
7. Simpanlah program yang anda ketik dan beri nama : progADC.asm
8. Pada program MIDE tersebut pilih Build /F9 atau untuk melakukan kompilasi program dari *.asm ke *.hex.

9. Lakukan download mikrokontroller dengan menggunakan Program Progisp 168 Software (Lihat Petunjuk Penggunaan)
10. Lakukan modifikasi pada program tersebut dengan manambahkan kata SUHU, pada Display1, 2, 3 dan 4 diikuti dengan data ADC.

BAB 5

Proyek

1. Rancanglah oleh kalian rangkaian aplikasi mikrokontrer dengan Analog To digital converter (ADC) dan tampilan di LCD, dapat diberikan input potensiometer.



Selesaikan Proyek tersebut dengan prosedur pekerjaan sebagai berikut:

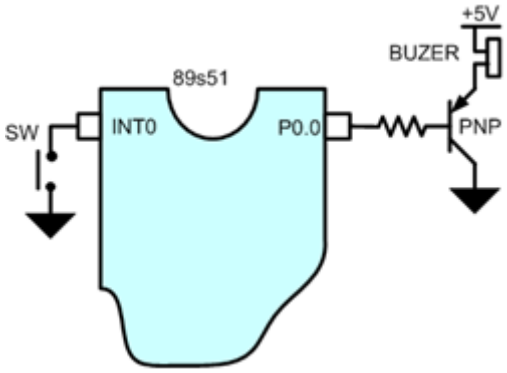
a. Persiapan
   - Menyiapkan alat dan bahan

b. Proses (Sistematika & Cara Kerja)
   - Membuat lay out/skema rangkaian sesuai dengan soal penugasan
   - Membuat program dengan bahasa assembler
   - Men *dowload* program dengan benar
   - Memasang komponen sesuai lay out /skema rangkaian yang telah dibuat
   - Melakukan penyolderan sesuai dengan lay out/skema rangkaian yang telah dibuat
   - Melakukan pengukuran rangkaian dengan menggunakan AVO meter

c. Hasil Kerja/Unjuk Kerja
   - Melakukan uji coba rangkaian

d. Sikap Kerja
   - Penggunaan alat sesuai dengan fungsinya

2. Rancanglah oleh kalian rangkaian aplikasi interupsi ekternal, dengan program utama running led di port2, output interupsi di port 0.0 untuk membunyikan buzzer, ingat posisi input interupsi ekternal berada di port 3.2 .



Selesaikan Proyek tersebut dengan prosedur pekerjaan sebagai berikut :

a. Persiapan
   - Menyiapkan alat dan bahan

b. Proses (Sistematika & Cara Kerja)
   - Membuat lay out/skema rangkaian sesuai dengan soal penugasan
   - Membuat program dengan bahasa assembler
   - Men *dowload* program dengan benar
   - Memasang komponen sesuai lay out /skema rangkaian yang telah dibuat
   - Melakukan penyolderan sesuai dengan lay out/skema rangkaian yang telah dibuat
   - Melakukan pengukuran rangkaian dengan menggunakan AVO meter

c. Hasil Kerja/Unjuk Kerja
   - Melakukan uji coba rangkaian

d. Sikap Kerja
   - Penggunaan alat sesuai dengan fungsinya

3. Rancanglah oleh kalian suatu system kendali untuk traffic light, dengan kondisi jalan simpang 4, settinglah waktu pada masing-masing lampu merah,kuning dan hijau.

Selesaikan Proyek tersebut dengan prosedur pekerjaan sebagai berikut:

a. Persiapan
   - Menyiapkan alat dan bahan

b. Proses (Sistematika & Cara Kerja)
   - Membuat lay out/skema rangkaian sesuai dengan soal penugasan
   - Membuat program dengan bahasa assembler
   - Men *dowload* program dengan benar
   - Memasang komponen sesuai lay out /skema rangkaian yang telah dibuat

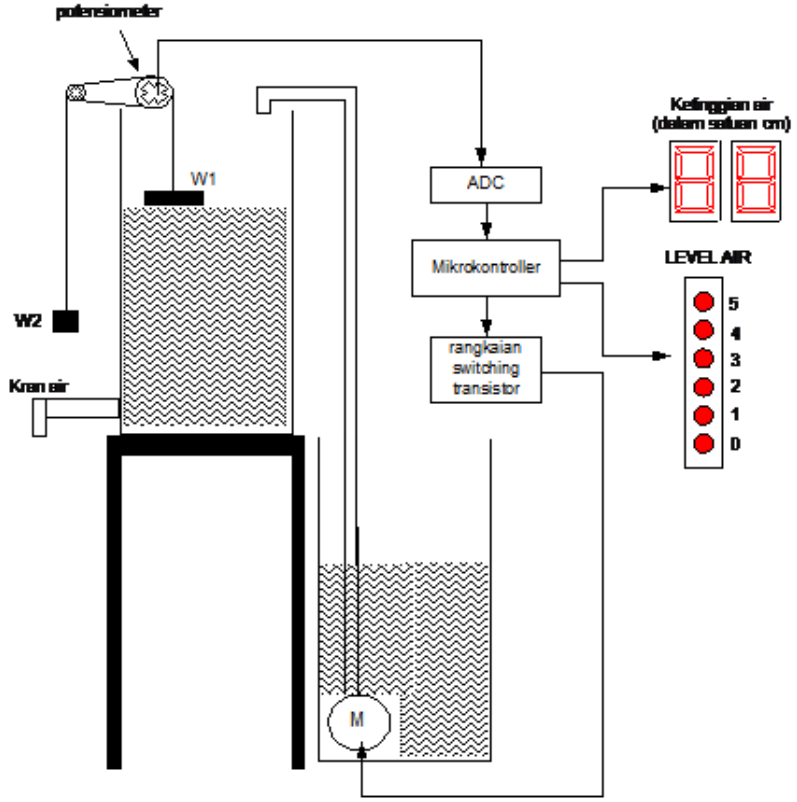
- Melakukan penyolderan sesuai dengan lay out/skema rangkaian yang telah dibuat
- Melakukan pengukuran rangkaian dengan menggunakan AVO meter

c. Hasil Kerja/Unjuk Kerja
- Melakukan uji coba rangkaian

d. Sikap Kerja
- Penggunaan alat sesuai dengan fungsinya

4. Rancanglah oleh kalian suatu system kendali ketnggiian Zat cair

Desain Plant



Selesaikan Proyek tersebut dengan prosedur pekerjaan sebagai berikut:

a. Persiapan
- Menyiapkan alat dan bahan

b. Proses (Sistematika & Cara Kerja)
- Membuat lay out/skema rangkaian sesuai dengan soal penugasan
- Membuat program dengan bahasa assembler
- Men *dowload* program dengan benar
- Memasang komponen sesuai lay out /skema rangkaian yang telah dibuat
- Melakukan penyolderan sesuai dengan lay out/skema rangkaian yang telah dibuat

- Melakukan pengukuran rangkaian dengan menggunakan AVO meter

c. Hasil Kerja/Unjuk Kerja
   - Melakukan uji coba rangkaian

d. Sikap Kerja
   - Penggunaan alat sesuai dengan fungsinya.

Daftar Pustaka

- Putra Eko Agfianto,"Belajar Mikrokontroler AT89C51/52/55 Teori dan Aplikasi",Gava Media,2004
- Djadja Hadi Maulana, Modul diktat Mikrokontroler , 2011
- http://www.mytutorialcafe.com
- Lukas Willa. (2010). Teknik Digital, mikroprosesor dan mikrokomputer, Bandung: Informatika.
- Syahban Rangkuti. (2011). Mikrokontroller Atmel AVR, Bandung: Informatika
- Widodo Budiharto. (2005). Perancangan Sistem dan Aplikasi Mikrokontroler. Jakarta: Elek Media Komputindo.
- Roudnay zaks,Austin lesea, sofyan H, Nasution. Teknik Perantaraan Mikroprosesor ,Penerbit erlangga 1993.
- Henry S.V Simanjuntak " dasar-dasar Mikroprosesor, penerbit kanisius, 2001
- Romy budhi Widodo,"Embedded System menggunakan Mikrokontroler dan Pemrograman C"penerbit ANDI Yogyakarta,2009
- Reni Nuraeni "modul pelatihan mikrokontroler untuk GMI Malaysia, PPPG Teknologi bandung, 2006
- Deni Anwar,Modul pelatihan SMKN1 Cimahi,2012
- *Dr. Putu Sudira MP., Sistem Mikroprosesor dan Mikrokontroler. Diknik Elektronika FT UNY*