MODUL 5

SISTEM PENGENDALIAN BERBASIS MIKROKONTROLER

I. KISI-KISI

1. Sistem Mikrokontroler
2. Arsitektur Mikrokontroler ATMEL AT89S51
3. Organisasi Memori AT89S51
4. Set intruksi AT89S51
5. Kemasan Fisik AT89S51
6. Bahasa Assembly AT89S51
7. Papan Pemrogram AT89S51
8. Simulator AT89S51 dengan TS Control Emulator

II. SISTEM MIKROKONTROLER

Mikrokontroler berkembang bersamaan dengan perkembangan mikroprosesor itu sendiri. Tetapi mikroprosesor sepertinya lebih akrab bagi kita karena telah digunakan sehari-hari sebagai otak dari sebuah sistem komputer. Namun demikian sebenarnya mikrokontroler juga telah akrab dengan kehidupan kita hanya kita tidak menyadari keberadaannya. Perlu diketahui bahwa mikrokontroler adalah merupakan otak dari segala peralatan otomatis yang dekat dengan kita. Misalnya, mikrontroler ada dalam peralatan VCD player, minicompo, remote kontrol, televisi, mesin cuci, pendingin udara, timbangan elektronik, mesin fotocopy, mesin wartel, printer, scanner dll. Bahkan dalam satu komputer bisa terdapat lebih dari 10 buah mikrokontroler.

Mikrokontroler juga terdapat banyak versi mulai dari yang untuk keperluan khusus hingga yang untuk keperluan umum. Beberapa jenis keluarga mikrokontroler yang terkenal adalah dari keluarga Intel, Motorola, Texas Instrument, Fujitsu, Microchip dll. Produk mikrokontroler Intel yang terkenal adalah 80C51 atau sering dikenal dengan keluarga MCS51. Pada produk ini, Intel menggunakan ROM sebagai media penyimpan program atau instruksi. Kelemahan produk ini adalah, program hanya bisa dituliskan sekali ke dalam ROM dan selanjutnya tidak bisa dihapus lagi. Hal ini tentu menyulitkan bagi kalangan yang akan belajar mikrokontroler karena jika kita membuat program tetapi belum sempurna, kita tidak bisa mengganti program dalam ROM itu.

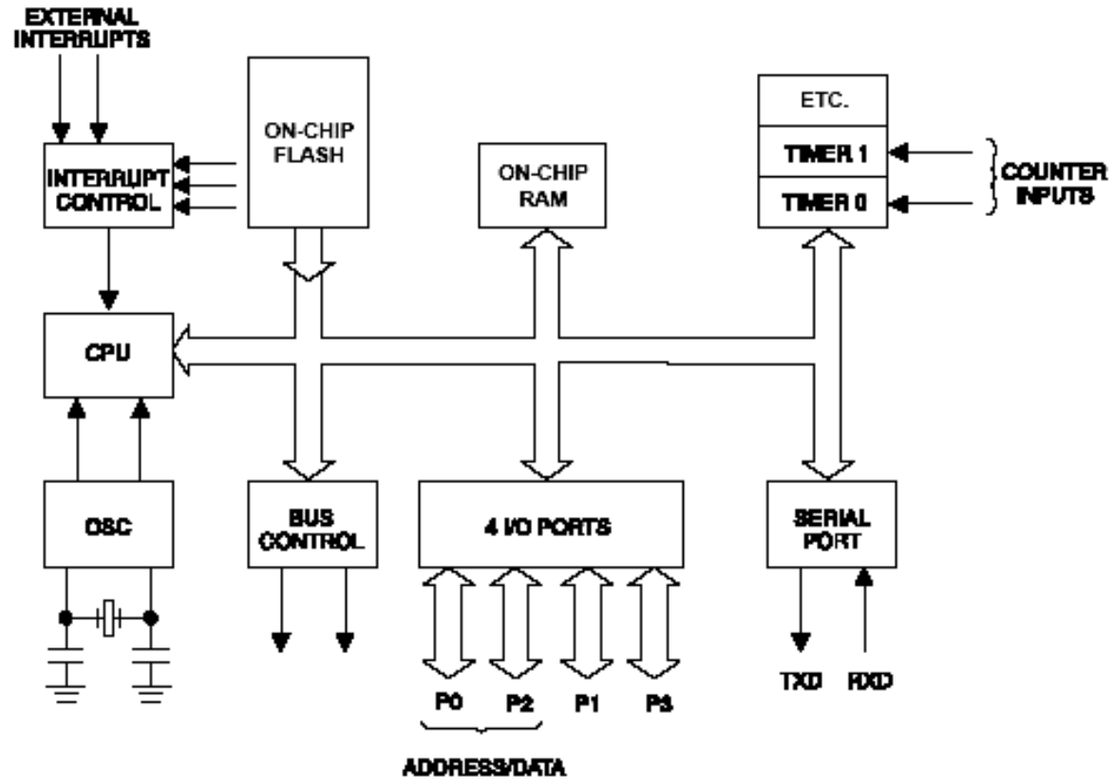
Tahun 1993, **Atmel** mengeluarkan seri mikrokontroler berdasarkan arsitektur MCS51 dari Intel dengan memakai teknologi *flash ROM* sebagai pengganti ROM. Dengan flash rom ini, program yang telah dituliskan bisa dihapus berulang kali sampai dirasakan program telah berjalan sesuai dengan keinginan. Hal ini sangat menghemat biaya dalam pembuatan program mikrokontroler. Seri mikrokontroler Atmel adalah AT89S51 dengan beberapa variannya kemudian menjadi terkenal sekali karena kemampuannya, teknologi flash rom, sudah menjadi standard industri dan harganya yang murah sekali. Dengan alasan itu, maka dalam praktikum ini juga akan dipelajari mikrokontroler Atmel AT89S51.

III. ARSITEKTUR MIKROKONTROLER ATMEL AT89S51

AT89S51 adalah mikrokontroler dari Atmel yang kompatibel (bisa dipertukarkan) dengan MCS51 buatan Intel. Perbedaannya hanyalah program disimpan dalam flash rom. Diagram blok AT89S51 terlihat seperti gambar di bawah.

CPU (Central Processing Unit) adalah otak dari mikrokontroler. CPU ini yang menerjemahkan instruksi sekaligus melaksanakan instruksi atau program dalam flash rom. Di dalam CPU terdapat ALU (Arithmatic Logic Unit), register akumulator (**A**) dan register tambahan (**B**). ALU adalah ibarat kalkulatornya sedang register dipakai untuk menyimpan data atau angka yang akan dihitung. Hasil operasi hitungan akan tersimpan di akumulator lagi.

Block Diagram



FLASH (ROM) digunakan untuk menyimpan program. Program adalah urut-urutan instruksi bagi mikrokontroler untuk melaksanakan fungsi yang kita inginkan. Ukuran flash rom AT89S51 adalah 4kB (4096 byte). Kehebatan flash rom ini adalah dapat dihapus diprogram ulang hingga 10.000 kali. Program dalam flash tidak akan hilang walaupun daya listrik dimatikan. Untuk menghapus dan memprogram flash lagi digunakan cara ISP (*In System Programming*). Dengan ISP ini, AT89S51 bisa diprogram ulang saat mikrokontroler berada dalam rangkaian tanpa harus melepas dan cukup dihubungkan dengan kabel ISP yang tersedia.

RAM (*Random Access Memory*) digunakan untuk menyimpan data sementara. Data yang tersimpan di RAM dengan mudah diubah dengan instruksi program tetapi data ini akan hilang jika daya listrik dimatikan. Ukuran RAM AT89S51 adalah 128B (baca:128 byte). Dilihat dari ukuran flash rom dan ram adalah sangat kecil dibanding pada komputer. Namun untuk keperluan pengendalian yang cukup komplek pun, ukuran itu telah mencukupi.

OSC (Oscilator) adalah pembangkit clock atau detak. Pulsa clock ini digunakan sebagai penanda dan sinkronisasi pelaksanaan instruksi dalam program. Kecepatan CPU dalam melaksanakan instruksi tergantung dari kecepatan pulsa clock ini. Kecepatan pulsa clock ditentukan oleh komponen kristal (XTAL) yang dipasang diluar. Frekuensi kristal yang bisa dipakai adalah dari 0Hz hingga 24MHz (24 juta Hz).

I/O PORT (*Input/Output Port*) adalah merupakan pintu untuk keluar masuk data dari mikrokontroler. Terdapat 4 port (P0, P1, P2, P3) yang masing-masing port berukuran 8 bit atau 1 byte jadi total terdapat 4 x 8bit = 32 bit atau 32 pintu data. Selain itu beberapa port berfungsi ganda seperti yang akan dijelaskan nanti.

SERIAL PORT adalah salah satu kelebihan mikrokontroler ini. Dengan serial port dapat dilakukan komunikasi data serial dengan standard USART (Universal Synchronous/ Asynchronous Receiver Transmiter) yaitu standard komunikasi serial yang bersifat full duplex.

TIMER. AT89S51 juga dilengkapi 2 buah timer/counter. Masing-masing timer bisa digunakan secara independen. Adanya timer/counter membuat mikrokontroler ini sangat baik untuk keperluan pengendalian. Ukuran tiap-tiap timer/counter adalah 16 bit.

INTERUPT adalah fasilitas untuk mengubah urutan pelaksanaan instruksi karena adanya kondisi yang dipantau. Terdapat paling tidak 7 buah sumber interupsi dalam AT89S51.

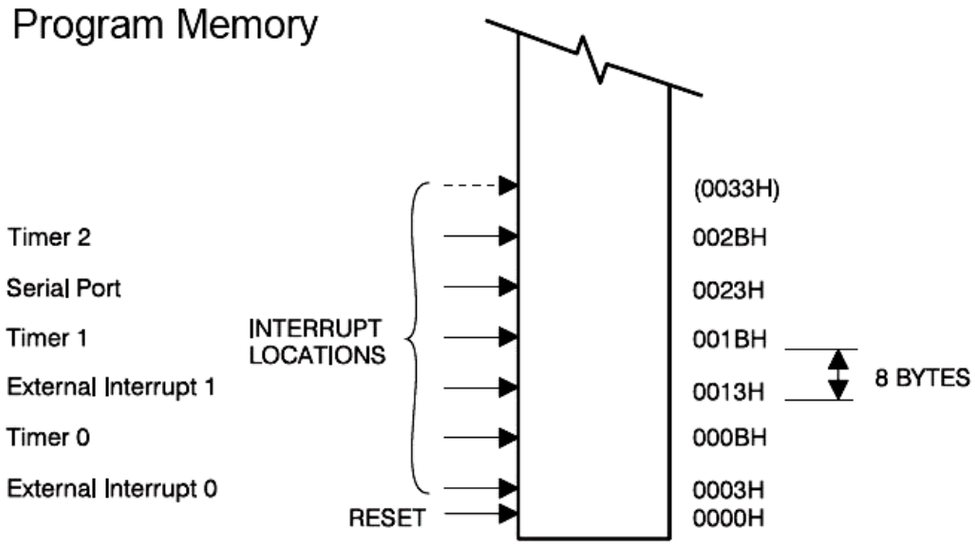
BUS CONTROL digunakan untuk mengendalikan memori tambahan. Sering dalam praktek diinginkan untuk menambah memori program (ROM) dan/atau menambah memori data (RAM) dengan memori eksternal. Dengan bus control ini, masing-masing memori bisa ditambahi hingga sebesar 64kB. Tetapi dengan cara ini perlu dikorbankan Port0 untuk saluran data/alamat yang dimultipleks dan sebagian atau semua Port2 untuk saluaran alamat.

IV. ORGANISASI MEMORI AT89S51

AT89S51 memisahkan antara memori untuk program dan untuk data dalam FLASH dan RAM. Metode ini membuat CPU 8 bit dapat dengat sangat efisien mengakses kedua memori. Dengan alamat 8 bit hanya akan dapat mengakses 256 lokasi memori, tetapi dengan manipulasi bit yang efisien akan didapat alamat 11bit (2048 lokasi memori) dan 16bit (64kB lokasi memori).

PROGRAM MEMORI

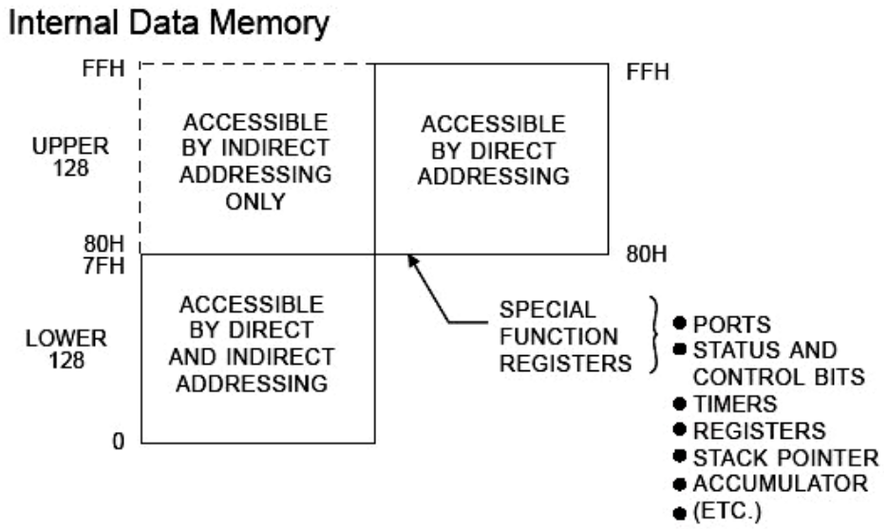
AT89S51 hanya terdapat flash sebesar 4kB untuk menyimpan program. Jika program yang dibuat lebih besar dari 4kB (0000H – 0FFFH) tentu ini membutuhkan tambahan memori. Tambahan untuk ROM bisa dipasang secara eksternal dan totalnya AT89S51 dapat mengakses program memori sebesar 64kB (0000H – FFFFH). Ketika pertama kali mikrokontroler dihidupkan (atau di-reset), secara otomatis CPU akan melaksanakan perintah/instruksi pada memori program di alamat 0000H (memori paling bawah).



Sedangkan ketika terjadi interupsi, maka program akan loncat ke alamat instruksi di lokasi 0003H jika sumber interupsi 0 dari luar, 000B jika Timer 0 habis dst.

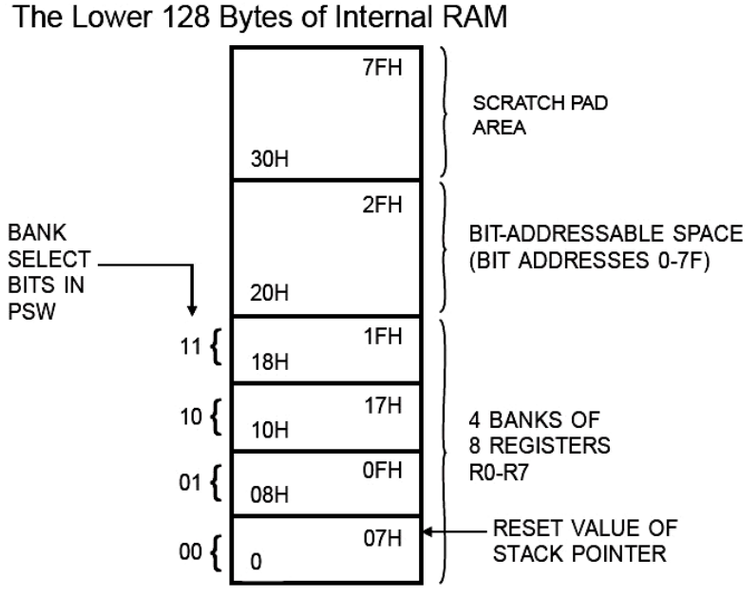
MEMORI DATA

Memori data internal terlihat seperti gambar berikut. Ruang memori data biasanya dikelompokkan ke dalam 3 yaitu ruang memori **128 atas**, ruang memori **128 bawah** dan **SFR** (*Special Function Register*). Sebenarnya AT89S51 hanya bisa mengakses memori data sebesar 256 alamat, tetapi dalam kenyataannya dengan cara pengalamatan yang berbeda dapat diakses 384 alamat yaitu yang dikelompokkan dalam 3 ruang memori di atas.



Ruang 128 Bawah

Memori data internal bagian 128 bawah terlihat seperti gambar berikut. Lokasi ini bisa dialamati dengan cara *direct* (langsung) dan *indirect* (tak-langsung).



Lokasi 00H – 1FH digunakan untuk 8 buah register keperluan umum yaitu R0, R1, R3, R4, R5, R6, dan R7. Sebenarnya register ini hanya menempati 8 alamat, tetapi dengan PSW (Program Status Word) akan dapat dipilih satu diantara 4 bank yang aktif untuk lokasi register R0 – R7. Mulai alamat 20H – 2F adalah RAM yang bisa dialamati bit per bit. Dan alamat 30H – 7FH adalah area bebas yang bisa dipakai untuk keperluan menyimpan data apa saja.

Ruang 128 Atas

Ruang 128 atas hanya dipunyai oleh mikrokontroler yang mempunyai RAM sebesar 256B atau lebih misalnya seri AT89C52, AT89C55, AT89S8252 atau AT89S53. Sedang pada seri AT89S51 yang akan dipakai disini ukuran RAM hanya 128B sehingga tidak mempunyai ruang 128 atas.

SFR (Special Function Register)

SFR atau register fungsi khusus menempati ruang alamat seperti pada ruang 128 atas hanya saja cara akses ke SFR hanya bisa dilakukan dengan cara *direct addressing*. SFR terdiri dari banyak register untuk fungsi khusus seperti terlihat pada gambar di bawah. Termasuk dalam SFR adalah Port (P0, P1, P2, P3), register dalam CPU yaitu akumulator (A) dan register B (B), Timer (TH0, TH1, TL0, TL1, TCON, TMOD), interupsi (IE, IP, PCON), komunikasi serial (SBUF, SCON), Pointer (DPH, DPL), PSW (Program Status Word), SP (Stack Pointer).

SFR Space

Byte address	Bit address								
FF									
F0	F7	F6	F5	F4	F3	F2	F1	F0	B
E0	E7	E6	E5	E4	E3	E2	E1	E0	ACC
D0	D7	D6	D5	D4	D3	D2	—	D0	PSW
B8	—	—	—	BC	BB	BA	B9	B8	IP
B0	B7	B6	B5	B4	B3	B2	B1	B0	P3
A8	AF	—	—	AC	AB	AA	A9	A8	IE
A0	A7	A6	A5	A4	A3	A2	A1	A0	P2
99	not bit addressable								SBUF
98	9F	9E	9D	9C	9B	9A	99	98	SCON
90	97	96	95	94	93	92	91	90	P1
8D	not bit addressable								TH1
8C	not bit addressable								TH0
8B	not bit addressable								TL1
8A	not bit addressable								TL0
89	not bit addressable								TMOD
88	8F	8E	8D	8C	8B	8A	89	88	TCON
87	not bit addressable								PCON
83	not bit addressable								DPH
82	not bit addressable								DPL
81	not bit addressable								SP
80	87	86	85	84	83	82	81	80	P0

Special Function Registers

Perhatikan pula bahwa register B, A, PSW, IP, P3, IE, P2, SCON, P1, TCON, P0 adalah register yang dapat dialamati bit per bit. Sedang register lain seperti SBUF, TH1, TH0, TL1, TL0, TMOD, PCON, DPH, DPL dan SP adalah register yang tidak dapat dialamati bit per bit.

Pada ruang SFR ada beberapa alamat yang tidak dipakai (masih kosong) yang mungkin akan terpakai pada seri Atmel yang akan datang.

PENGALAMATAN RAM

Ada beberapa cara mengalamati RAM yaitu:
1. Direct Addressing (pengalamatan langsung)
2. Indirect Addressing
3. Register Addressing
4. Immediate Constant

1. **Direct Addressing**

   ADD A, 7FH

   artinya tambahkan data di akumulator (A) dengan isi memori data di alamat 7FH dan hasilnya disimpan lagi di A.

2. **Indirect Addressing**

   ADD A, @R0

   artinya tambahkan data di akumulator (A) dengan isi memori yang alamatnya ada pada register R0 dan hasilnya disimpan lagi di A.

3. **Register Addressing**

   ADD A, R7

   artinya tambahkan isi A dengan isi R7 dan hasilnya disimpan lagi di A atau A = A + R7

4. **Immediate Constant**

   ADD A, #127

   tambahkan isi A dengan data 127 (desimal) dan hasilnya disimpan lagi di A (A = A + 127).

V. SET INSTRUKSI AT89S51

Set instruksi adalah kumpulan instruksi/perintah yang ada pada mikrokontroler untuk membuat program. Set instruksi tersebut kemudian dituliskan secara urut untuk membentuk logika perintah tertentu sedemikian rupa mikrokontroler melakukan perintah seperti yang diinginkan. Kumpulan set instruksi dengan logika tertentu kemudian sering kita sebut program. Dalam hal ini karena program berada di level yang dekat dengan bahasa mesin sering disebut sebagai bahasa assembly. AT89S51 mempunyai format penulisan program adalah sebagai berikut:

[Label:] Mnemonic [operand1] [, operand2] [, operand3] [;komentar]

Suatu perintah selalu diawali oleh mnemonic (kode instruksi) dan boleh diikuti operand atau tidak sama sekali. Jika mnemonic diikuti operand maka harus dipisahkan oleh spasi. Jika operand yang dipakai lebih dari satu maka dipisahkan oleh koma. Untuk memberi komentar

didahului tanda titik koma. Suatu baris perintah bisa diawali dengan “label”. Label ini digunakan untuk menandai pengulangan atau instruksi yang loncat dst.
Contoh:
1. Instruksi yang tidak mempunyai operand
   RET NOP RETI
2. Instruksi dengan satu operand
   INC A INC R0 DEC A
3. Instruksi dengan 2 operand
   ADD A, #10
   MUL A, B
4. Instruksi dengan 3 operand
   CJNE A, 45H, lompat

Tabel 1. Instruksi Aritmatika Mikrokontroler Atmel.

Mnemonic	Operasi	Mode Pengalamatan				Siklus
		Dir	Ind	Reg	Imm	
ADD A, <byte>	A = A + <byte>	√	√	√	√	12 clock
ADDC A, <byte>	A = A + <byte> + C	√	√	√	√	12 clock
SUBB A, <byte>	A = A - <byte> - C	√	√	√	√	12 clock
INC A	A = A + 1	Akumulator saja				12 clock
INC <byte>	<byte> = <byte> + 1	√	√	√		12 clock
INC DPTR	DPTR = DPTR + 1	Data pointer saja				24 clock
DEC A	A = A - 1	Akumulator saja				12 clock
DEC <byte>	<byte> = <byte> - 1	√	√	√		12 clock
MUL AB	B:A = B x A	Reg A dan B saja				48 clock
DIV AB	A = Bulat[A/B] B = Sisa[A/B]	Reg A dan B saja				48 clock
DA A	Decimal Adjust	Akumulator saja				12 clock

Tabel 2. Instruksi Logika Mikrokontroler Atmel.

Mnemonic	Operasi	Mode Pengalamatan				Siklus
		Dir	Ind	Reg	Imm	
ANL A, <byte>	A = A .AND. <byte>	√	√	√	√	12 clock
ANL <byte>, A	<byte> = <byte> .AND. A	√				12 clock
ANL <byte>, #data	<byte> = <byte> .AND. #data	√				24 clock
ORL A, <byte>	A = A .OR. <byte>	√	√	√	√	12 clock
ORL <byte>, A	<byte> = <byte> .OR. A	√				12 clock
ORL <byte>, #data	<byte> = <byte> .OR. #data	√				24 clock
XRL A, <byte>	A = A .XOR. <byte>	√	√	√	√	12 clock
XRL <byte>, A	<byte> = <byte> .XOR. A	√				12 clock
XRL <byte>, #data	<byte> = <byte> .XOR. #data	√				24 clock
CLR A	A = 00H	Akumulator saja				12 clock
CPL A	A = .NOT. A	Akumulator saja				12 clock
RL A	Putar kiri 1 bit Reg A	Akumulator saja				12 clock
RLC A	Putar kiri 1 bit Reg A lewat C	Akumulator saja				12 clock
RR A	Putar kanan Reg A	Akumulator saja				12 clock
RRC A	Putar kanan 1 bit Reg A lewat C	Akumulator saja				12 clock
SWAP A	Tukar 4bit rendah dan tinggi Reg A	Akumulator saja				12 clock

Tabel 3. Instruksi Transfer Data Pada Mikrokontroler Atmel.

Mnemonic	Operasi	Mode Pengalamatan				Siklus
		Dir	Ind	Reg	Imm	
MOV A, <sumber>	A = <sumber>	√	√	√	√	12 clock
MOV <tujuan>, A	<tujuan> = A	√	√	√		12 clock
MOV <tujn>, <smbr>	<tujuan> = <sumber>	√	√	√	√	24 clock
MOV DPTR, #data16	DPTR = data langsung 16 bit				√	24 clock
PUSH <sumber>	INC SP ; MOV @SP, <sumber>	√				24 clock
POP <tujuan>	MOV <tujuan>, @SP ; DEC SP	√				24 clock
XCH A, <byte>	Tukar antara reg A dan <byte>	√	√	√		12 clock
XCHD A, @Ri	Tukar 4bit rendah antara reg A dan isi memori dengan alamat di R0/R1		√			12 clock

Tabel 4. Instruksi Transfer Data Yang Mengakses Memori Data Eksternal.

Mnemonic	Operasi	Siklus
MOVX A, @Ri	A = baca memori eksternal dgn alamat pada reg R0/R1	24 clock
MOVX @Ri, A	tulis reg A ke memori eksternal dgn alamat pada reg R0/R1	24 clock
MOVX A, @DPTR	A = baca memori eksternal dgn alamat pada reg DPTR	24 clock
MOVX @DPTR, A	tulis reg A ke memori eksternal dgn alamat pada reg DPTR	24 clock

Tabel 5. Instruksi Baca Data Dari Memori Program.

Mnemonic	Operasi	Siklus
MOVC A, @A + DPTR	A = baca memori program dgn alamat pada reg (A + DPTR)	24 clock
MOVC A, @A + PC	A = baca memori program dgn alamat pada reg (A + PC)	24 clock

Tabel 6. Instruksi Boolean.

Mnemonic	Operasi	Siklus
ANL C, bit	C = C .AND. bit	24 clock
ANL C, /bit	C = C .AND. .NOT. bit	24 clock
ORL C, bit	C = C .OR. bit	24 clock
ORL C, /bit	C = C .OR. .NOT. bit	24 clock
MOV C, bit	C = bit	12 clock
MOV bit, C	bit = C	12 clock
CLR C	C = 0	12 clock
CLR bit	bit = 0	12 clock
SETB C	C = 1	12 clock
SETB bit	bit = 1	12 clock
CPL C	C = .NOT. C	12 clock
CPL bit	bit = .NOT. bit	12 clock
JC rel	loncat ke alamat rel (label) jika C = 1	24 clock
JNC rel	loncat ke alamat rel (label) jika C = 0	24 clock
JB bit, rel	loncat ke alamat rel (label) jika bit = 1	24 clock
JNB bit, rel	loncat ke alamat rel (label) jika bit = 0	24 clock
JBC bit, rel	loncat ke alamat rel (label) jika bit = 1 dan kemudian buat bit = 0	24 clock

Tabel 7. Instruksi Loncat Tidak Bersyarat.

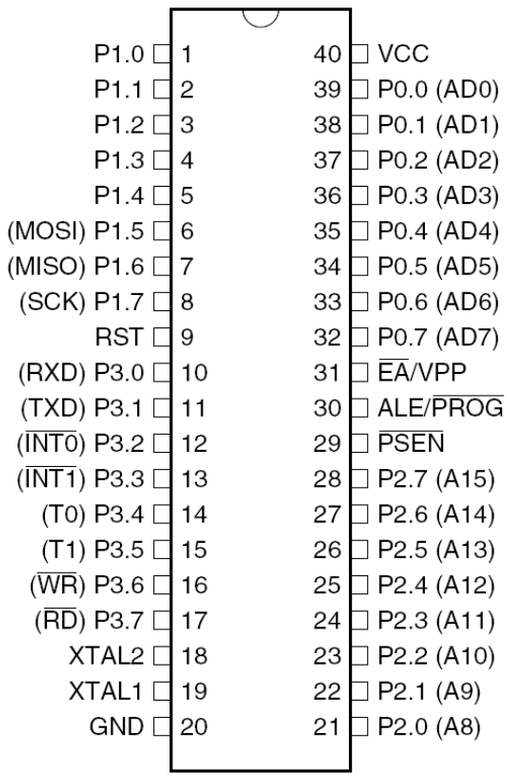
Mnemonic	Operasi	Siklus
JMP rel	Loncat ke alamat rel (label)	24 clock
JMP @A + DPTR	Loncat ke alamat yang ditunjuk (A + DPTR)	24 clock
CALL rel	Panggil subrutin pada alamat rel (label)	24 clock
RET	Kembali dari subrutin	24 clock
RETI	kembali dari interupsi	24 clock

Tabel 8. Instruksi Loncat Dengan Syarat.

Mnemonic	Operasi	Mode Pengalamatan				Siklus
		Dir	Ind	Reg	Imm	
JZ rel	Loncat jika A = 0	Akumulator saja				24 clock
JNZ rel	Loncat jika A ≠ 0	Akumulator saja				24 clock
DJNZ <byte>, rel	<byte> dikurangi 1 dan loncat ke alamat rel (label) jika belum = 0	√		√		24 clock
CJNE A, <byte>, rel	Loncat ke alamat rel (label) jika A ≠ <byte>	√			√	24 clock
CJNE <byte>, #data, rel	Loncat ke rel jika <byte> ≠ #data		√	√		24 clock

VI. KEMASAN FISIK AT89S51

Bentuk fisik AT89S51 adalah seperti gambar di bawah.



AT89S51 tersedia dalam kemasan IC (*Integrated Circuit*) 40 pin (kaki). Pin 40 (VCC) dihubungkan ke sumber tegangan stabil 5V dan pin 20 (GND) ke ground.

Port 0

Port 0 adalah pintu dua arah untuk input atau output 8 bit. Jika logika 0 dikeluarkan di port 0 maka setiap pin dapat menarik arus hingga 25mA yang cukup untuk menggerakkan 8 input IC TTL. Jika logika 1 dikeluarkan di port 0 maka setiap pin ada pada impedansi tinggi. Port 0 juga dipakai untuk saluran data dan alamat yang dimultipleks jika ada tambahan memori eksternal.

Port 1, Port 2 dan Port 3

Port 1, 2 dan 3 adalah port dua arah juga untuk input atau output data masing-masing 8 bit. Setiap pin pada port ini telah dilengkapi resistor pull-up internal sehingga jika logika 1 dikeluarkan ke port, maka akan muncul tegangan 5V pada output (bukan impedansi tinggi seperti pada port 0).

Saat output logika 1 pada setiap pin hanya dapat mengeluarkan arus maksimum 600µA. Sedang saat output logika 0 pada setiap pin dapat menarik arus maksimum 15mA. Port 2 juga dipakai sebagai saluran alamat jika digunakan tambahan memori eksternal. Port 3 bit 0 (P3.0) juga berfungsi sebagai input data pada komunikasi serial (RXD) dan P3.1 sebagai output data serial pada komunikasi data serial (TXD). P3.2 dan P3.3 dipakai sebagai input interupsi dari luar

($\overline{\text{INT0}}$ dan $\overline{\text{INT1}}$) jika sistem interupsi diaktifkan. Sedang P3.4 dan P3.5 dipakai sebagai input untuk timer/counter (T0 dan T1) jika sumber pulsa dari luar saat timer/counter diaktifkan.

XTAL2 dan XTAL1
Pin-pin ini digunakan untuk menyambungkan resonator kristal untuk pembangkit pulsa clock bagi CPU. Kristal yang bisa dipakai adalah berfrekuensi 0Hz hingga 33MHz.

VII. BAHASA ASSEMBLY AT89S51

Penulisan bahasa assembly untuk mikrokontroler Atmel AT89S51 dapat digunakan sembarang program editor. Yang perlu diperhatikan bahwa file program dalam bahasa assembly harus disimpan dalam format teks (*text*). Untuk menuliskan bahasa assembler haruslah dipahami beberapa istilah sbb:

1. **Label dan Simbol**.
   Label mewakili suatu alamat dari instruksi atau data. Penulisan label diakhiri titik dua (**:**). Simbol adalah seperti label hanya tidak diakhiri tanda titik dua. Label dan simbol harus diawali dengan huruf.

```
PAR      EQU  500          ;"PAR" adalah suatu simbol dari nilai 500
START:   MOV  A,#0FFH      ;"START" adalah label yang menunjuk lokasi intruksi tsb.
```

2. **Penulisan konstanta langsung**.
   Konstanta langsung yang dipakai diakhir bilangan ditulis dengan simbol "**B"** untuk biner, "**D**" atau " " untuk desimal, "**H**" untuk heksadesimal. Penulisan konstanta langsung harus diawali dengan tanda "#" dan diikuti angka tidak boleh huruf.

```
MOV  A, #255
MOV  A, #11111111B
MOV  A, #0FFH
MOV  A, #255D
```

   Semua instruksi di atas adalah sama yaitu mengisi A dengan data 255.

3. **Penulisan simbol assembler lain**

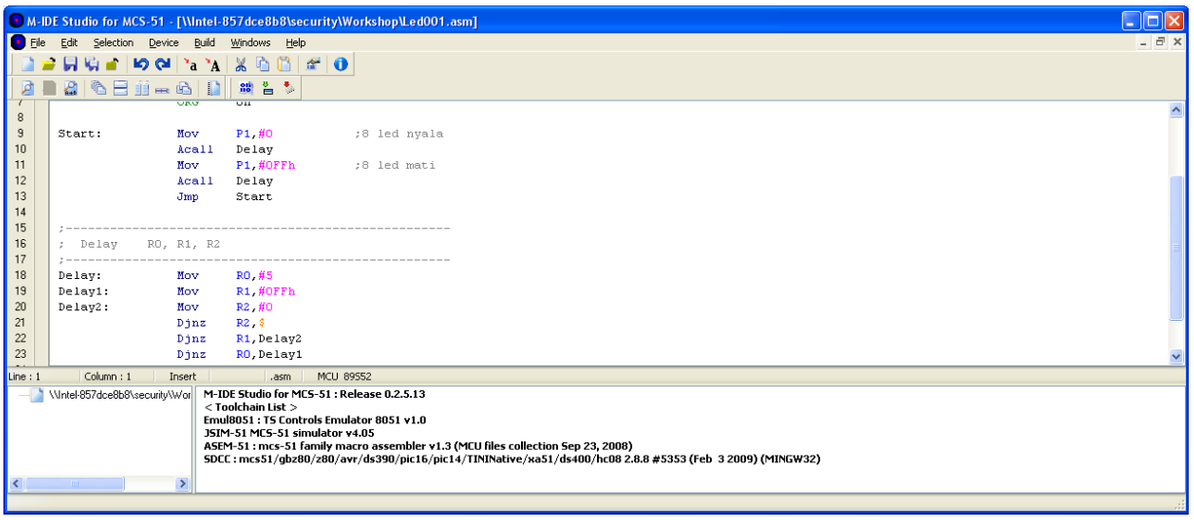
```
ULANG:  JNB  TI, ULANG
        JNB  TI, $          kedua instruksi itu adalah sama.
```

4. Pengalamatan tak langsung (*Indirect Addressing*) dilakukan dengan tanda "at"/ @ bersama dengan R0, R1, PC atau DPTR.

```
ADD     A, @R0
MOVC    A, @A+PC
```

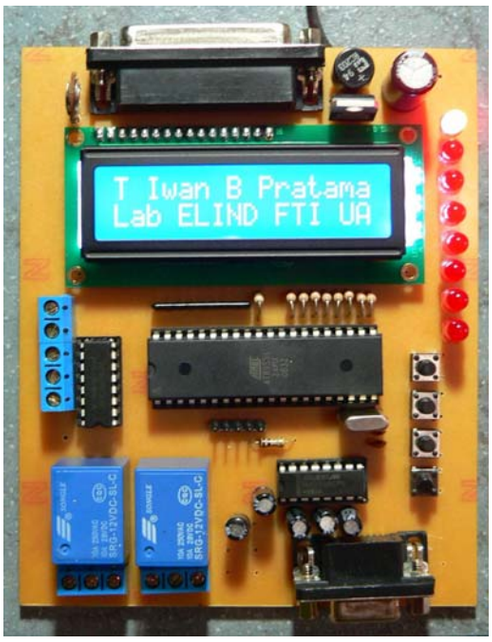
5. Program untuk penulisan program assembly bisa digunakan **M-IDE studio for MCS51**. Tampilan program tampak seperti gambar di bawah. Program *assembler* yang ditulis kemudian disimpan dengan nama ekstensi file **.ASM**. Program M-IDE telah dilengkapi dengan program simulator dengan TS Control Emulator, kompiler bahasa asembler dengan ASEM-51 dan juga kompiler untuk penulisan program dengan bahasa C yaitu dengan SDCC.

   Jika program telah dituliskan, maka untuk mengkompile program ke dalam bahasa mesin tinggal klik "BUILD" atau F9. Hasil kompilasi akan terbentuk dua file baru dengan nama sama tetapi dengan ekstensi .LST dan .HEX. File ekstensi .LST digunakan untuk melacak program apabila terjadi kesalahan penulisan sintak yaitu dengan membuka file LST ini dan mencari petunjuk kesalahan yang harus dibetulkan. Sedang file dengan ekstensi HEX ini nantinya yang akan diprogramkan/diisikan ke mikrokontroler.



VIII. MODUL PERCOBAAN

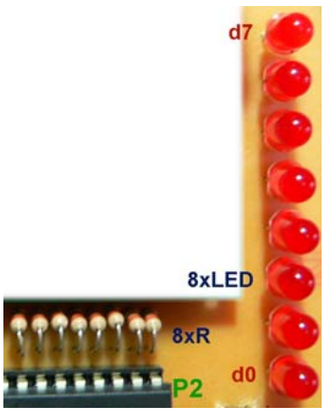
Modul percobaan terdiri dari papan percobaan AT89S51 seperti terlihat di gambar bawah, power supply unit, kabel ISP untuk memprogram mikrokontroler, komputer PC dan kabel serial komunikasi.



Papan percobaan AT89S51 terdiri dari mikrokontroler AT89S51, 8 led merah, LCD 16x2 karakter, driver IC ULN2003 untuk menggerakkan motor DC atau motor stepper, 2 relay untuk menggerakkan peralatan DC maupun AC, 4 push button untuk memberikan input data atau interupsi ke mikrokontroler, interface RS232 untuk komunikasi serial dengan komputer dan konektor DB25 untuk komunikasi paralel dengan komputer atau printer.

Untuk menghidupkan papan percobaan disediakan power supply atau dengan power supply AC atau DC dengan tegangan antara 7V hingga 15V.

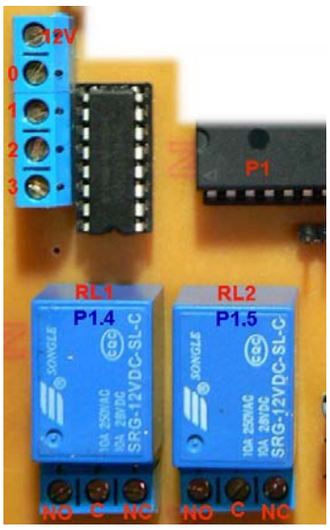
Keterangan Papan Percobaan:



8 LED merah dihubungkan ke P2 dengan urutan dari atas ke bawah yaitu P2.7 hingga P2.0 yang aktif logika rendah (0). Masing-masing led dihubungkan seri dengan resistor 330 ohm untuk menurunkan arus agar tidak terlalu membebani output port 2 mikrokontroler.

Logika pengendalian LED seperti tabel berikut:

No	Logika Port 2	Keadaan LED
1	0	Nyala
2	1	Mati



IC penggerak ULN2003 digunakan untuk menggerakkan peralatan DC seperti kipas DC 12V. Tersedia 4 terminal 0, 1, 2, 3 yang dihubungkan ke Port P1.0, P1.1, P1.2 dan P1.3 lewat IC ULN2003. Output ini aktif logika tinggi.

Contohnya: hubungkan motor/kipas DC 12V ke terminal +12V dan terminal 0, maka logika pengendalian motor adalah:

No	Logika P1.0	Keadaan Kipas
1	0	Mati
2	1	Nyala

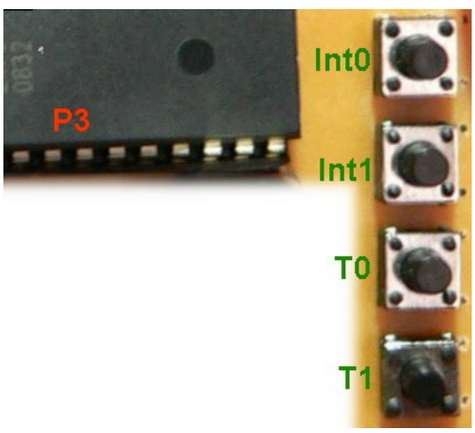
Contoh lain: hubungkan motor stepper tipe unipolar ke terminal 12V dan terimnal 0, 1, 2 dan 3. Maka logika pengendalian motor stepper adalah:

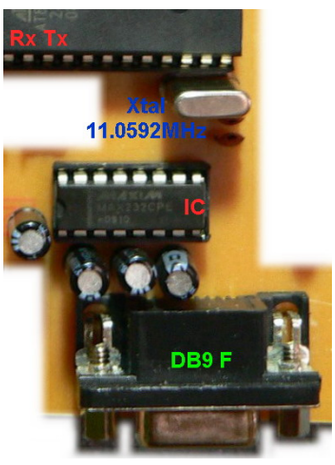
No		P0.0 hingga P0.3		P0.0 hingga P0.3
1	**Putar Kiri**	0011	**Putar Kanan**	0011
2		1001		0110
3		1100		1100
4		0110		1001

P1.4 dan P1.5 lewat IC ULN2003 juga dihubungkan ke relay elektromekanik. Dengan keluaran relay RL1 dan RL2 bisa digunakan untuk menggerakkan alat DC maupun AC. Relay bekerja seperti switch mekanik dengan logika pengendaliaannya dalah sbb:

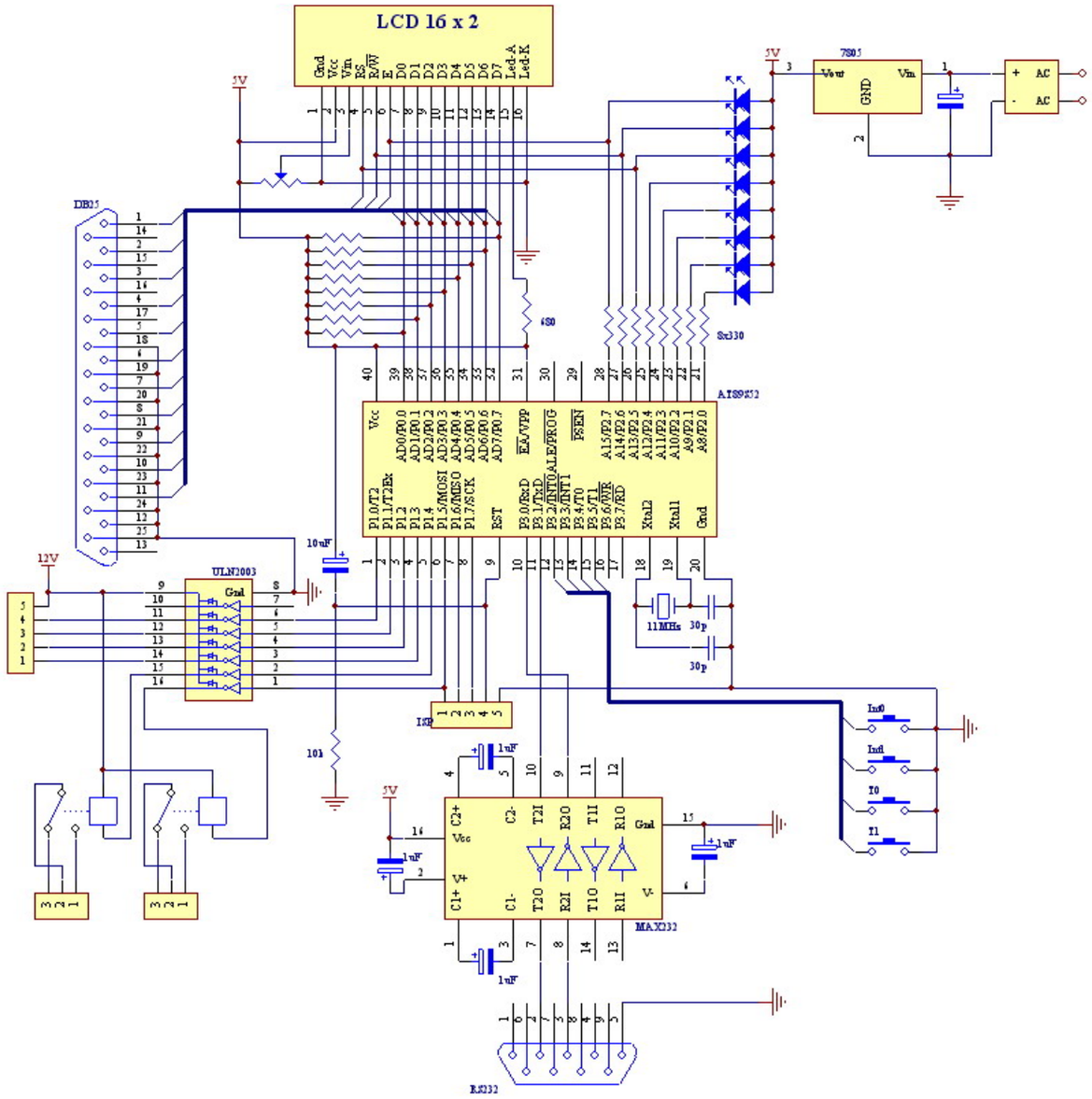
No	P1.4 (P1.5)	Kondisi Relay	Terminal yang terhubung
1	0	Mati	C terhubung ke NC
2	1	Aktif	C terhubung ke NO

Kit LCD (*Liquid Crystal Display*) MM1602 adalah submodul untuk menampilkan karakter sebanyak 16 huruf dalam 2 baris. MM1602 mempunyai mikrokontroler sehingga sebagian tugas untuk menampilkan tulisan ke LCD sudah ditangani oleh mikrokontroler tersebut. Untuk menampil-kan tulisan LCD diperlukan 8 bit data dan 2 bit kontrol R/S dan E. Data dihubungkan ke port 0 sedang dua bit kontrol dihubungkan ke P2.7 dan P2.5.





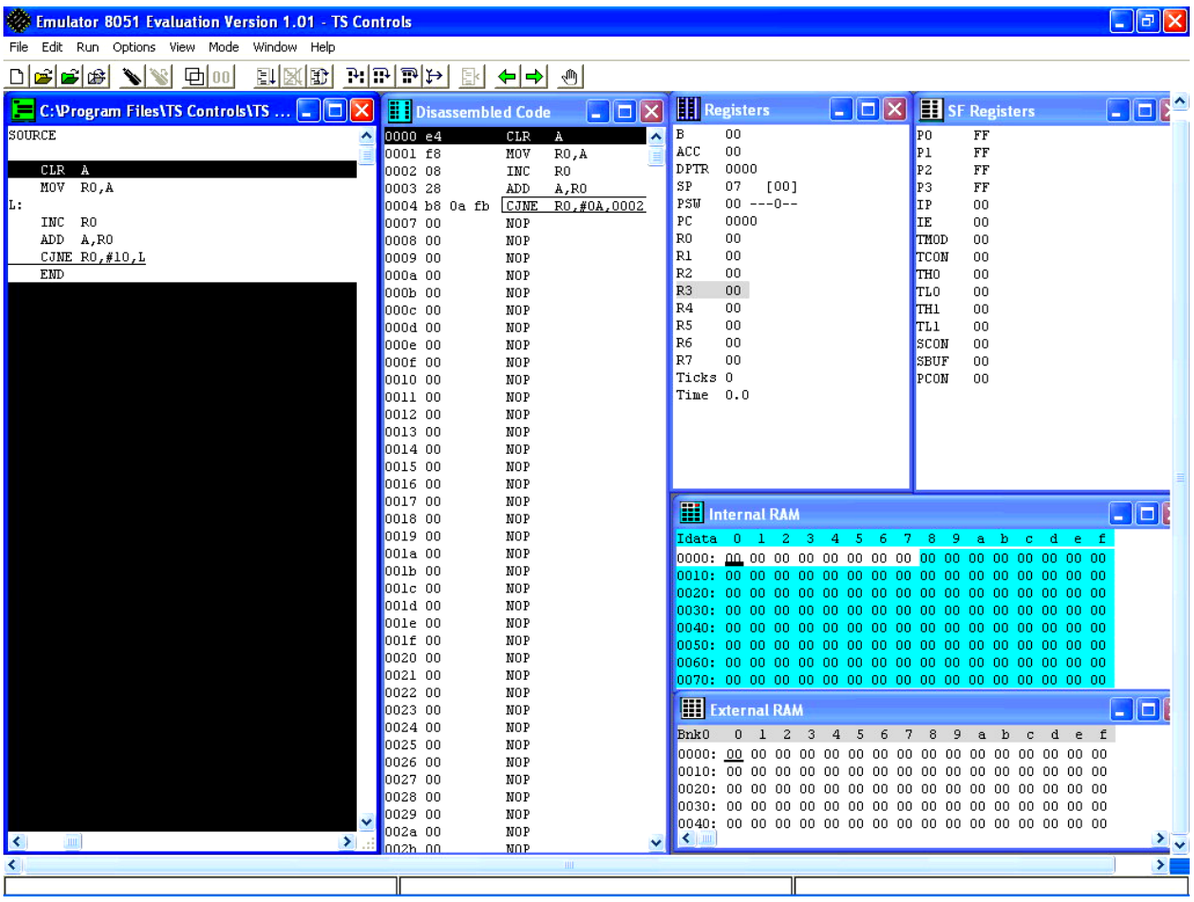
Tombol input dan komunikasi serial RS232.



IX. SIMULASI AT89S51 DENGAN TS CONTROL EMULATOR 8051

Program AT89S51 yang telah ditulis kemudian di-*compile* dengan program ASM51.EXE dan dikonversi dengan program OH.EXE sehingga akan terbentuk file program dengan ekstensi LST dan HEX. Kedua program ini yang akan kita pakai. Program dengan ekstensi LST dapat dibuka dengan program editor (CRIMSON) untuk melihat error dalam penulisan program. Sedang program dengan ekstensi HEX dapat kita simulasikan dengan TS CONTROL untuk melihat logika kerja program. Tampilan ketika menjalankan TS CONTROL adalah seperti gambar berikut.

Window yang muncul adalah sebagai berikut: Layar SOURCE CODE digunakan untuk membuka program sumber (berekstensi LST). Layar DISASSEMBLED CODE digunakan untuk membuka program konversi (berekstensi HEX). Layar lain yaitu REGISTERS dan SF REGISTERS (Special Function Register) adalah hasil simulasi nilai-nilai register ketika program disimulasikan. Selain itu ada tampilan nilai INTERNAL RAM dan EXTERNAL RAM ketika mensimulasi program.Jalankan program dengan klik RUN dan lihat perubahan nilai pada register dan RAM. Untuk menjalankan program step per step (satu instruksi demi satu instruksi) klik RUN terus klik STEP atau tekan F11.



XI. TATA LAKSANA PERCOBAAN

A. Bagian Penulisan Program Assembler dan Simulasi

1.a. PENULISAN PROGRAM ASSEMBLY MIKROKONTROLER AT89S51

PERINTAH!

[ ] Buka program M-IDE Studio.

[ ] Tuliskan program assembler berikut.

```
;---------------------------------------------------------------
; Program kendalikan 8 LED di Port 2
;---------------------------------------------------------------

                ORG     0H
Start:          Mov     P2,#0           ;8 led nyala
                Mov     P2,#0FFh        ;8 led mati
                Jmp     Start
                END
```

[ ] Simpan dengan nama MOD501.ASM

[ ] Compile dengan perintah: BUILD atau F9

1.b. SIMULASI PROGRAM MENGENDALIKAN 8 LED DI PORT 1

PERINTAH!

[ ] Buka program TS Control

[ ] Buka file MOD501.LST dan MOD501.HEX

[ ] Tuliskan disassembled code (kode heksadesimal untuk program di atas)

[ ] Tuliskan juga terjemahan tiap-tiap instruksi di program ini

2. PENGAMATAN HASIL SIMULASI

PERINTAH!

[ ] Catat nilai awal SFR Port 1 (P2) dan Program Counter (PC) di layar Register

[ ] Jalankan program dengan tekan F11 dan catat lagi nilai P1 dan PC
[ ] Tekan F11 dan catat P2 dan PC

3a. PENGENDALIAN 8 LED MENYALA SECARA BERGESER
PERINTAH!
[ ] Tulislan program pengendalian 8 led di port 2 yang berjalan secara bergeser
[ ] Simpan program dengan nama MOD502.ASM

```
;-------------------------------------------------------------
; Program kendalikan 8 LED di Port 2 yang menyala bergeser
;-------------------------------------------------------------

                ORG     0H
Start:          Mov     P2,#01111111b           ;led d7 nyala
                Mov     P2,#10111111b           ;led d6 nyala
                Mov     P2,#11011111b           ;led d5 nyala
                Mov     P2,#11101111b           ;led d4 nyala
                Mov     P2,#11110111b           ;led d3 nyala
                Mov     P2,#11111011b           ;led d2 nyala
                Mov     P2,#11111101b           ;led d1 nyala
                Mov     P2,#11111110b           ;led d0 nyala
                Jmp     Start
                END
```

[ ] Compile dengan BUILD atau F9.
[ ] Buka program TS Control
[ ] Buka file MOD502.LST dan MOD502.HEX
[ ] Tuliskan disassembled code (kode heksadesimal untuk program di atas)
[ ] Tuliskan juga terjemahan tiap-tiap instruksi di program ini

3b. PENGAMATAN HASIL SIMULASI
PERINTAH!
[ ] Catat nilai awal SFR Port 1 (P2) dan Program Counter (PC) di layar Register
[ ] Jalankan program dengan tekan F11 dan catat lagi nilai P2 dan PC
[ ] Tekan F11 dan catat P2 dan PC

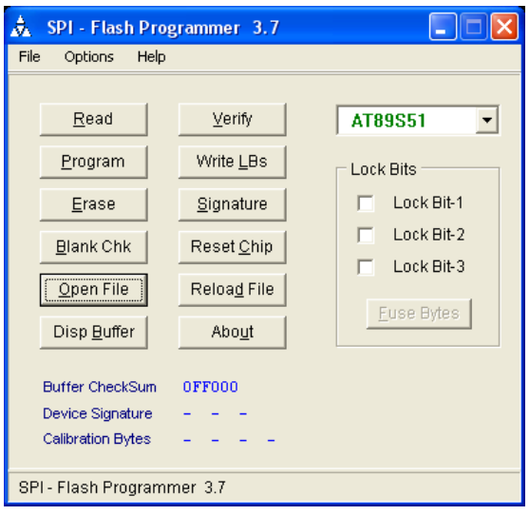
B. Bagian Menjalankan Program Di Modul Percobaan

4. PROGRAM MENGENDALIKAN 8 LED DI PORT 2
PERINTAH!
[ ] Ubah program MOD501.ASM menjadi seperti berikut ini

```
;-----------------------------------------------------------------------------------------------
;       Program Kendalikan 8 LED di Port 2
;       File:   Mod503.asm
;-----------------------------------------------------------------------------------------------

                Org     0H

Start:          Mov     P2,#00H             ;8 LED nyala semua
                Acall   Delay               ;Tunda sebentar
                Mov     P2,#0FFH            ;8 LED mati semua
                Acall   Delay               ;Tunda sebentar
                Ajmp    Start               ;Kembali ke start lagi

;-----------------------------------------------------------------------------------------------
;       Prosedure Delay
;-----------------------------------------------------------------------------------------------
Delay:          Mov     R1,#5
Delay1:         Mov     R2,#0
Delay2:         Mov     R3,#0
```

```
        Djnz    R3,$
        Djnz    R2,Delay2
        Djnz    R1,Delay1
        Ret

        End
```

[ ] Simpan dengan nama MOD503.ASM
[ ] Compile file.
[ ] Sambungkan kabel ISP ke Printer port komputer.
[ ] Jalankan program ISP-Flash Programmer 3.0 atau SPI-Flash Programmer 3.7 untuk Mengisikan program ke mikrokontroler.



Tampilan programmer seperti gambar di samping.
Buka file MOD501.HEX dengan klik “Open File”
Pilih tipe mikrokontroler yang sesuai (AT89S51)
Klik “Signature” untuk mengetes mikrokontroler yang terhubung.
Jika tidak terjadi kesalahan tinggal klik “Program”.

[ ] Amati hasil jalannya program di atas!

[ ] Ganti nilai di R0 dengan 20, amati hasilnya

[ ] Ganti nilai R3 dengan 20, amati hasilnya

5. PROGRAM 1 LED MENYALA SECARA BERGESER DI PORT 1

PERINTAH!
[ ] Buka file MOD502.ASM dan ubah menjadi seperti gambar dibawah berikut
[ ] Simpan file dengan nama MOD504.ASM

```
;-------------------------------------------------------------------------------------------
;       Program Kendalikan LED di Port 2 menyala bergeser
;       File:   Mod504.asm
;-------------------------------------------------------------------------------------------

        Org     0H

Start:  Mov     P2,#01111111b       ;LED ke-1 nyala
        Acall   Delay               ;Tunda sebentar
        Mov     P2,#10111111b       ;LED ke-2 nyala
        Acall   Delay               ;Tunda sebentar
        Mov     P2,#11011111b       ;LED ke-3 nyala
        Acall   Delay               ;Tunda sebentar
        Mov     P2,#11101111b       ;LED ke-4 nyala
        Acall   Delay               ;Tunda sebentar
        Mov     P2,#11110111b       ;LED ke-5 nyala
        Acall   Delay               ;Tunda sebentar
        Mov     P2,#11111011b       ;LED ke-6 nyala
        Acall   Delay               ;Tunda sebentar
        Mov     P2,#11111101b       ;LED ke-7 nyala
        Acall   Delay               ;Tunda sebentar
        Mov     P2,#11111110b       ;LED ke-8 nyala
        Acall   Delay               ;Tunda sebentar
        Ajmp    Start               ;Kembali ke start lagi

;-------------------------------------------------------------------------------------------
;       Prosedure Delay
;-------------------------------------------------------------------------------------------
Delay:  Mov     R1,#5
Delay1: Mov     R2,#0
```

```
Delay2:     Mov     R3,#0
            Djnz    R3,$
            Djnz    R2,Delay2
            Djnz    R1,Delay1
            Ret

            End
```

[ ] Buat sendiri program yang menghasilkan gerakan lampu led yang beragam.
[ ] Tunjukkan pada asisten praktikum.
[ ] Catat listing program yang anda buat.

6. PROGRAM ASSEMBLER MOD505.ASM

PERINTAH!

[ ] Praktekkan program Mod505.asm untuk mengendalikan motor kipas DC 12V yang dihubungkan di terminal +12V dan terminal 0
[ ] Amati dan catat hasil percobaan anda
[ ] Praktekkan program Mod505.asm untuk mengendalikan bohlam AC 220V yang dihubungkan ke Relay terminal C dan NO.
[ ] Amati dan catat hasil percobaan anda

```
;-------------------------------------------------------------------------------------------
;       Program Kendalikan Motor DC dan Lampu AC
;       File:   Mod505.asm
;-------------------------------------------------------------------------------------------

            Org     0H

Start:      Mov     P1,#00H         ;8 LED nyala semua
            Acall   Delay           ;Tunda sebentar
            Mov     P1,#0FFH        ;8 LED mati semua
            Acall   Delay           ;Tunda sebentar
            Ajmp    Start           ;Kembali ke start lagi

;-------------------------------------------------------------------------------------------
;       Prosedure Delay
;-------------------------------------------------------------------------------------------
Delay:      Mov     R1,#5
Delay1:     Mov     R2,#0
Delay2:     Mov     R3,#0
            Djnz    R3,$
            Djnz    R2,Delay2
            Djnz    R1,Delay1
            Ret

            End
```

7. PROGRAM ASSEMBLER MOD506.ASM

PERINTAH!

[ ] Praktekkan Timer di program Mod506.asm untuk menunda tepat 1 detik
[ ] Amati dan catat hasil percobaan anda

```
;-----------------------------------------------------------------------------
;  Program Kendali 8 LED di Port 2 dengan waktu tunda 1 detik
;       File :  Mod506.asm
;-----------------------------------------------------------------------------
            ORG     0H
Cacah       EQU     20              ;cacah pengulangan
Isi         EQU     -50000          ;isi Timer 0 yang mencacah dari -50 000
            MOV     TMOD, #01H      ;Timer 0 mode 1 -> GATE = 0, C/T = 0, M1 = 0, M0 = 1
Start:      MOV     P2, #0FH        ;Hidupkan 4 LED kanan
            CALL    Tunda           ;Tunda 1 detik
            MOV     P2, #0F0H       ;Hidupkan 4 LED kiri
```

```
                CALL    Tunda               ;Tunda 1 detik
                SJMP    Start               ;Ulang ke start lagi

;-------------------------------------------------------------------------------
;  Prosedur Tunda 1 detik tepat
;-------------------------------------------------------------------------------
Tunda:          MOV     R0, #Cacah          ;Register R0 untuk pengulangan 20x
Ulang:          MOV     TH0, #HIGH Isi      ;Register TH0 diisi byte tinggi isi counter -50000
                MOV     TL0, #LOW Isi       ;Register TL0 diisi byte rendah isi counter -50000
                SETB    TR0                 ;Hidupkan Timer 0
                JNB     TF0, $              ;Menunggu hingga Timer 0 overflow
                CLR     TF0                 ;Clear bit TF0 agar bisa dicek lagi nanti kalau overflow
                CLR     TR0                 ;Matikan dulu Timer 0
                DJNZ    R0, Ulang           ;Cek R0 = 0 ? untuk pengulangan 20x
                RET                         ;Kembali dari pemanggilan prosedur

                END
```

8. PROGRAM ASSEMBLER MOD507.ASM

PERINTAH!

[ ] Praktekkan Interupsi dengan switch button INT0 di program Mod507.asm

[ ] Amati dan catat hasil percobaan anda

```
;-------------------------------------------------------------------------------------------
;       Program Interupsi INT1 untuk mengubah tampilan led
;       File :  Mod507.asm
;-------------------------------------------------------------------------------------------
                Org     0H
                Jmp     ProgUtama

                Org     13H
                Jmp     LayIntEks1

                Org     30H
ProgUtama:      Mov     A,#01111111b
                mov     IE,#10000100b       ;aktifkan interupsi eksternal 1

Ulang:          Mov     P2,A
                Call    delay
                RR      A
                Jmp     Ulang
                end

;-------------------------------------------------------------------------------------------
;       Subrutin delay
;-------------------------------------------------------------------------------------------
delay:          Mov     R0,#3
delay1:         Mov     R1,#0
delay2:         Mov     R2,#0
                Djnz    R2,$
                Djnz    R1,Delay2
                Djnz    R0,Delay1
                Ret

;-------------------------------------------------------------------------------------------
;       Layanan interupsi INT1
;-------------------------------------------------------------------------------------------
LayIntEks1:     Mov     P2,#11111110b
                Call    delay
                Mov     P2,#11111101b
                call    delay
                mov     P2,#11111011b
                call    delay
                mov     P2,#11110111b
                call    delay
                mov     P2,#11101111b
```

```
        call    delay
        mov     P2,#11011111b
        call    delay
        mov     P2,#10111111b
        call    delay
        mov     P2,#01111111b
        call    delay
        reti
```

9. PROGRAM ASSEMBLER MOD508.ASM

PERINTAH!

[ ] Praktekkan program Mod508.asm untuk menampilkan tulisan di LCD

[ ] Amati dan catat hasil percobaan anda

```
;-------------------------------------------------------------------------------------------
;       Program Tulisan di LCD
;       Data        P0
;       E           P2.7
;       RS          P2.5
;       File  :     Mod508.asm
;-------------------------------------------------------------------------------------------

        Org     0H
E       Equ     P2.7
RS      Equ     P2.5

;-------------------------------------------
; Inisialisasi LCD
;-------------------------------------------
        Mov     A,#38H              ;set function
        Call    Wrins
        Mov     A,#38H              ;set function
        Call    Wrins
        Mov     A,#0CH              ;display on, cursor off, blink off
        Call    Wrins
        Mov     A,#01H              ;clear display
        Call    Wrins
        Mov     A,#06H              ;Entry mode
        Call    Wrins

        Mov     P1,#0
;-------------------------------------------
; Tulis Karakter ke LCD
;-------------------------------------------
Ulang:
        Mov     R3,#0
        Mov     DPTR,#Kalimat1

Kirim1: Mov     A,#0
        Movc    A,@A+DPTR
        Call    WrData
        Inc     R3
        Inc     DPTR
        Cjne    R3,#16,Kirim1

        Mov     A,#0C0H             ;baris ke 2
        Call    Wrins

        Mov     R3,#0
        Mov     DPTR,#Kalimat2

Kirim2: Mov     A,#0
        Movc    A,@A+DPTR
        Call    WrData
        Inc     R3
        Inc     DPTR
```

```
        Cjne    R3,#16,Kirim2

        Call    Delay
        Call    Delay

        Mov     A,#01H              ;clear
        Call    Wrins
        Mov     A,#02H              ;Home
        Call    Wrins

        Call    Delay
        Call    Delay
        Call    Delay

        Jmp     Ulang

;-------------------------------------------
; Delay 5 s
;-------------------------------------------
Delay:  Mov     R5,#15
Delay1: Mov     R6,#0
Delay2: Mov     R7,#0
        Djnz    R7,$
        Djnz    R6,Delay2
        Djnz    R5,Delay1
        Ret

;-------------------------------------------
; Write Instruksi
;-------------------------------------------
Wrins:  Clr     RS
        Setb    E
        Mov     P0,A
        Clr     E
        Mov     R0,#5
Balik0: Mov     R1,#0
        Djnz    R1,$
        Djnz    R0,Balik0
        Ret

;-------------------------------------------
; Write Data
;-------------------------------------------
Wrdata:Setb     RS
        Setb    E
        Mov     P0,A
        Clr     E
        Mov     R0,#5
Balik1: Mov     R1,#0
        Djnz    R1,$
        Djnz    R0,Balik1
        Ret

;-------------------------------------------
; Data Kalimat1 dan Kalimat2
;-------------------------------------------
Kalimat1:       DB      'Lab Elektronika '
Kalimat2:       DB      '> TI FTI UAJY <'
        End
```

10. PROGRAM ASSEMBLER MOD509.ASM

PERINTAH!

[ ] Sambungkan kabel serial RS232 dan praktekkan program Mod509.asm

[ ] Jalankan program Hyperterminal di komputer anda dan hubungkan pada baudrate yang sama.

[ ] Catat hasil percobaan anda.

```
;-------------------------------------------------------------------------------
```

```
;          Program terima dan kirim data serial lewat RS232
;          Tekan Keyboard tombol A, B, 1, 2, atau yang lain
;          File:     Mod509.asm
;-------------------------------------------------------------------------------
                ORG   0H

                MOV   SCON, #50H    ;Port serial mode 1 (8 bit UART) baudrate dg Timer 1
                MOV   TMOD, #20H    ;Timer 1 mode 2 (8 bit isi ulang)
                MOV   TH1, #0FDH    ;nilai isi ulang TH1 = 0F4H untuk baudrate 9600 bps
                SETB  TR1           ;aktifkan timer 1 untuk menghasilkan baudrate

Ulang:          JNB   RI, $         ;Tunggu terimaan 1 byte data (jika ada) sebelumnya tlh selesai
                MOV   A, SBUF       ;Pindahkan data yang diterima ke akumulator A
                CLR   RI            ;Reset RI untuk memantau penerimaan data serial berikutnya

                CJNE  A, #'s', Led4kiri ;Jika yang dikirim komputer tombol bukan S
                MOV   P2, #0FFH     ;8 led mati terus kirim pesan

                MOV   DPTR, #Kalimat;Alamat penunjuk Kalimat
                MOV   R0, #65               ;Jumlah huruf yang dikirim

Kirim:          CLR   A
                CLR   TI
                MOVC  A, @A + DPTR
                MOV   SBUF, A
                JNB   TI, $
                INC   DPTR
                DJNZ  R0, Kirim
                MOV   P1,#0                 ;matikan semua I/O di port 1
                AJMP  Ulang

Led4kiri:       CJNE  A, #'a', Led4kanan
                MOV   P2,#0FH
                AJMP  Ulang

Led4kanan:      CJNE  A, #'b', Relay1
                MOV   P2, #0F0H
                AJMP  Ulang

Relay1:         CJNE  A,#'d',Relay2
                SETB  P1.5
                JMP   Ulang

Relay2:         CJNE  A,#'e',Led8nyala
                SETB  P1.4
                JMP   Ulang

Led8nyala:      MOV   P2, #00H
                AJMP  Ulang

;-------------------------------------------------------------------------------
;          Data pesan
;-------------------------------------------------------------------------------
Kalimat:        DB    'Lab Elektronika Industri Prodi Teknik Industri FTI UAJY -- 2009', 13, 10
                END
```