BAB II

DASAR TEORI

Dasar teori yang dibutuhkan untuk mendukung pembuatan tugas akhir ini adalah teori tentang *device* atau komponen yang digunakan, antara lain teori tentang.

1. *Short Message Service* (SMS)
2. Komunikasi Serial RS232
3. Mikrokontroler AT89C52
4. Bahasa *Assembley*
5. Pemecah Sandi Pada *Seven Segment*

2.1 *Short Message Service*

Short Message Service (SMS) adalah salah satu tipe *Instant Messaging* (IM) yang memungkinkan user untuk bertukar pesan singkat kapanpun, walaupun user sedang melakukan *call* data atau suara. SMS dihantarkan pada *channel signal Global System for Mobile Communication* (GSM). SMS juga digunakan pada teknnologi GPRS dan CDMA. SMS menjamin pengiriman pesan oleh jaringan, jika terjadi kegagalan pesan akan disimpan dahulu di jaringan, pengiriman paket SMS bersifat *out of band* dan menggunakan *bandwidth* rendah.

2.1.1 Elemen-elemen SMS

1. *Short Message Entity* (SME)

   Short Message Entity (SME) adalah elemen yang dapat mengirim atau

menerima pesan singkat. SME dapat berupa *software* aplikasi pada *mobile handset*, dapat juga berupa perangkat *facsimile*, perangkat telex, *remote internet server*, dll.

Sebuah SME dapat berupa server yang terkoneksi dengan SMS centre secara langsung atau melalui *gateway*. Dikenal juga *External SME* (ESME) yang merepresentasikan sebuah WAP proxy/server, *Email Gateway* atau *Voice Mail Server*

2. *Short Message Service Center*

SMS Service Centre (SMSC) memegang peran kunci dalam arsitektur SMS. Fungsi utama SMSC adalah menyampaikan pesan singkat antara SME dengan MS, juga menyimpan dan meneruskan pesan singkat (menyimpan pesan jika penerima SME tidak tersedia). SMSC dapat terintegrasi sebagai bagian dari *mobile network* (contoh: terintegrasi dengan MSC) atau sebagai entitas network independen. *SMSC* mentransfer pesan dalam format *point to point* pada sistem yang melayani.

3. *SMS-Gateway* dan *SMS_interworkingmobile switching Center*

SMS *Gateway Mobile Switching Center* (SMS-GMSC) adalah sebuah aplikasi MSC yang mampu menerima pesan singkat dari SMSC, menginterogasi *Home Location Register* (HLR) untuk informasi *routing*, dan mengirimkan pesan pendek tersebut ke MSC dan piranti bergerak yang dituju.

4. *Home Location Register*

Home Location Register (HLR) adalah elemen jaringan yang berisi data detil dari setiap *subscriber*. Sebuah HLR biasanya mampu mengatur ratusan bahkan ribuan *subscriber*. Ketika diinterogasi oleh SMSC, HLR memberikan informasi *routing* mengenai pelanggan yang ingin dituju.

5. *Mobile Switching Center*

Mobile Switching Centre (MSC) melakukan fungsi registrasi, authentikasi, update lokasi user, *billing service* dan sebagai *interface* dengan jaringan lain. Selain itu MSC juga bertanggung jawab untuk *call set-up*, *release* dan *routing*. Melalui MSC, *mobile network* terhubung dengan jaringan lain seperti *Public Switched Telephone Network* (PSTN), *Integrated Service Digital Network* (ISDN), *Circuit Switched Public Data Network* (CSPDN) dan *Packet Switched Public Data Network* (PSPDN).

6. *Vision Location Register*

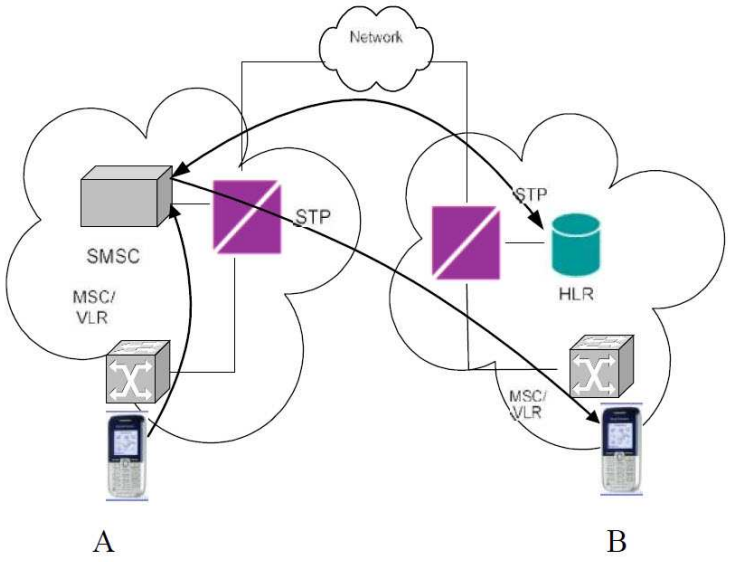
V*isitor Location Register* (VLR) berisi informasi dinamis tentang user yang terkoneksi dengan *mobile network* termasuk lokasi user tersebut HRL yang *roaming* ke HRL lainnya. VLR biasanya terintegrasi dengan MSC. Informasi ini dibutuhkan oleh MSC untuk melayani pelanggan yang berkunjung.

7. *Base Station System*

Semua fungsi yang tekait dengan transmisi sinyal radio elektromagnetis antara MSC dan piranti bergerak di lakukan di *Base Station System* (BSS).

BBS terdiri dari *Base Station Controllers* (BSCs), juga dikenal sebagai wilayah sel. BSC dapat mengendalikan satu atau lebih BTS dan bertanggung jawab dalam pemberian sumber data yang semestinya ketika pelanggan bergerak dari satu sektor suatu BTS ke sektor lain, terlepas dari apakah sektor berikutnya tersebut berada dalam BTS yang sama atau berbeda.

2.1.2 Arsitektur SMS



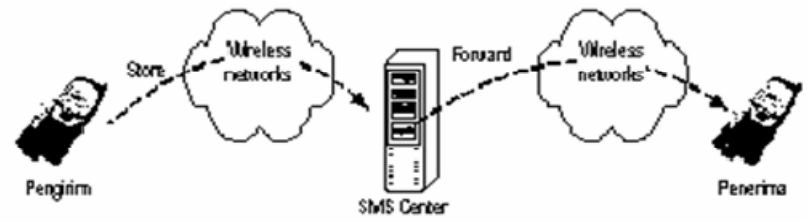
Gambar 2.1 *Susunan Jaringan dan Aliran Pesan*

- SMS dikirim melalui MSC/VLR ke SMSC pesan MAP "*forward SM*"
- SMSC harus merouting informasi dari HLR B. SMSC mengirim MAP "*send routing info for SM*" dengan nomor MSISDN B.
- HLR mengirim kembali IMSI dari end user B dan VLR nya
- SMSC mengirim SMS sebagai MAP message melaluiMSC/VLR ke end user B.

2.1.3 Mekanisme *Store And Forward* Pada SMS

SMS adalah data tipe *asynchoronous message* yang pengiriman datanya dilakukan dengan mekanisme protokol *store and forward*. Hal ini berarti bahwa pengirim dan penerima SMS tidak perlu berada dalam status berhubungan (*connected/online*) satu sama lain ketika akan saling bertukar pesan SMS. Pengiriman pesan SMS secara *store and forward* berarti pengirim pesan SMS menuliskan pesan dan nomor telepon tujuan dan kemudian mengirimkannya (*store*) ke server SMS (*SMS-Center*) yang kemudian bertanggung jawab untuk mengirimkan pesan tersebut (*forward*) ke nomor telepon tujuan.

Keuntungan mekanisme *store and forward* pada SMS adalah, penerima tidak perlu dalam status *online* ketika ada pengirim yang bermaksud mengirimkan pesan kepadanya, karena pesan akan dikirim oleh pengirim ke SMSC yang kemudian dapat menunggu untuk meneruskan pesan tersebut ke penerima ketika ia siap dan dalam status *online* di lain waktu. Ketika pesan SMS telah terkirim dan diterima oleh SMSC, pengirim akan menerima pesan singkat (konfirmasi) bahwa pesan telah terkirim (*message sent*). Hal-hal inilah yang menjadi kelebihan SMS dan populer sebagai layanan praktis dari sistem telekomunikasi bergerak.



Gambar 2.2 *Mekanisme Store And Forward*

2.1.4 *Basic Feature SMS*

SMS mempunyai beberapa *basic feature*, seperti.

1. *Message Submission and Delivery*

   Terdiri dari *message sending* dan *message delivery*. Pada *message sending*, pesan dikirm dari MS ke SMSC, dialamatkan ke SME lain sebagai *mobile user* lain atau *host internet*. *Originator* (asal) SME menentukan *validity period* dari pesan tersebut, pesan yang sudah tidak valid lagi akan dihapus oleh sepanjang pengiriman pesan. Fitur ini dikenal sebagai *Short Message-Mobile Originated* (SM-MO).

   Pada *message delivery*, pesan disampaikan oleh SMSC ke MS. Dikenal sebagai *Short Message Mobile Terminated* (SM-MT). SM-MO dan SM-MT dapat dikirim/diterima saat *voice call* atau koneksi data sedang berlangsung.

2. *Status Report*

   SME asal (*originator*) meminta *status report* pada pengiriman pesan singkat ke SME penerima (*recipient*). *Status report* memberikan indikasi pada user asal apakah pesan terkirim dengan sukses atau tidak kepada SME penerima.

3. *Reply Path*

   Replay Path dapat diatur oleh SME asal (atau SMSC *serving*) untuk mengindikasikan bahwa SMSC serving dan mampu untuk meng-*handle* secara langsung *reply* dari SME penerima.

4. *Validity Period*

   Pesan originator dimungkinkan untuk menentukan *validity period* sebuah pesan. *Validity period* ini menentukan batas waktu sebuah pesan harus dikirim sebelum akhirnya dihilangkan oleh jaringan.

2.1.5 Cara kerja SMS

Sekali pesan dikirim, pesan tersebut akan diterima dahulu oleh SMSC yang kemudian disampaikan pada nomor tujuan. Untuk melakukan ini SMSC mengirimkan sebuah SMS *request* ke HLR melalui *Signal Transfer Point* (STP) untuk menemukan pelanggan tujuan. Saat HLR menerima pesan tersebut maka HLR akan merespon ke SMSC dengan status pelanggan berupa.

1. *Inactive* atau *Active*
2. Letak pelangan yang dimaksud (pelanggan tujuan).

Jika tidak aktif maka SMSC akan meng-*hold* pesan tersebut sampai pada periode tertentu. Saat pelanggan menyalakan *handset* maka akan terjadi *update location* pada HLR dan HLR akan mengirim status terhadap pesan yang belum terkirim. SMSC mentransfer pesan dalam format *point to point*. Jika aktif akan segera terkirim. SMSC menerima verifikasi jika pesan tersebut sudah diterima oleh

nomor yang dituju dan mengkategorikan pesan tersebut sebagai sebuah ”pesan terkirim” dan tidak akan melakukan percobaan pengiriman pesan lagi.

Prinsip dasarnya adalah bahwa hanya ada satu *Short Massage Service Center* yang menerjemahkan pesan untuk dikirimkan pada sebuah jaringan GSM atau CDMA. SMS dapat dikirimkan dan diterima bersamaan dengan *voice*, data dan *fax* menggunakan *channel* yang berbeda dengan SMS. Oleh karena itu pengguna SMS jarang atau tidak pernah mendapatkan signal sibuk pada saat jaringan *voice* sedang sibuk, kecuali memang SMS Center tersebut memang sibuk. Berbeda kondisinya apabila jaringan sedang sibuk.

2.1.6 *AT command*

Dibalik tampilan menu message pada ponsel sebenarnya adalah *AT Command* yang bertugas mengirim atau menerima data ke atau dari *SMS-Center*. *AT Command* tiap-tiap SMS device bisa berbeda-beda, tetapi pada dasarnya sama. Beberapa *AT Command* yang penting untuk SMS yaitu.

- AT+CMGS : untuk mengirim SMS
- AT+CMGL : untuk memeriksa SMS
- AT+CMGD : untuk menghapus SMS

AT Command untuk SMS, biasanya diikuti oleh data I/O yang diwakili oleh unit-unit PDU.

2.1.7 PDU Sebagai Bahasa SMS dan Bagian-bagiannya

SMS terdiri dari *Protocol Data Unit* (PDU). Didalam PDU terdapat beberapa header yang sesuai standar GSM 07.05. Data yang mengalir ke/dari SMS-Center

harus berbentuk PDU. PDU berisi bilangan *Heksadesimal* yang mencerminkan bahasa I/O. PDU terdiri atas beberapa *header*. *Header* untuk kirim SMS ke SMS Center berbeda dengan SMS yang diterima dari SMS Center.

0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F.

Misal : Untuk angka *Decimal* 1000, bilangan *Heksadesimal* adalah 3E8

Cara mengonversikannya :

1000 : 16 = 62 → 8
62 : 16 = 3 sisa → E
3 : 0 = 0 sisa 3 → 3

2.1.8 PDU untuk mengirim SMS terdiri dari 8 header:

1. Nomor SMS-Center

   Header pertama ini terbagi atas tiga *subheader*, yaitu.

   - Jumlah pasangan *heksadesimal* SMS-Center dalam bilangan heksa.
   - *National*/*International Code*

     a. Untuk *National*, kode *subheader*-nya yaitu 81

     b. Untuk *International*, kode *subheader*-nya yaitu 91.

   - No SMS-Centernya sendiri, dalam pasangan *heksa* dibalik-balik.

   Jika tertinggal satu angka heksa yang tidak memiliki pasangan, angka tersebut akan dipasangkan dengan huruf F didepannya.

Tabel 2.1 *Nomor SMS-Center Operator Selular Indonesia*

No	Operator Selullar	SMSC	Kode PDU
1	Telkomsel	62811000000	07912618010000F0
2	Satelindo	62816125	59126181652
3	Exelcom	62818445009	07912618485400F9
4	Indosat-IM3	62855000000	07912658050000F0

2. Tipe SMS

   Untuk SEND tipe SMS = 1. Jadi bilangan heksanya adalah 01

3. Nomor Referensi SMS

   Nomor referensi ini dibiarkan dulu 0, jadi bilangan heksanya adalah 00. Nanti akan diberikan sebuah nomor referensi otomatis oleh ponsel/alat *SMS-gateway*.

4. Nomor Ponsel Penerima

   Sama seperti cara menulis *PDU Header* untuk *SMS-Center*, header ini juga terbagi atas tiga bagian, sebagai berikut.

   - Jumlah bilangan desimal nomor ponsel yang dituju dalam bilangan heksa.
   - National/International Code.
   - Untuk Nasional, kode subheader-nya 81
   - Untuk Internasional, kode subheader-nya 91
   - Nomor ponsel yang dituju, dalam pasangan heksa dibalik-balik. Jika tertinggal satu angka heksa yang tidak memiliki pasangan, angka tersebut dipasangkan dengan huruf F didepannya.

Contoh :

Untuk nomor ponsel yang dituju = 628129573337 dapat ditulis dengan cara sebagai berikut.

Cara : 628129573337 diubah menjadi :

1. 0C : ada 12 angka
2. 91
3. 26-18-92-75-33-73

Digabung menjadi : 0C91261892753373

5. Bentuk SMS, antara lain :
   - 00 : dikirim sebagai SMS
   - 01 : dikirim sebagai telex
   - 02 : dikirim sebagai fax

   Dalam hal ini, untuk mengirim dalam bentuk SMS tentu saja dipakai 00

6. Skema Encoding Data I/O

   Ada dua skema, yaitu.

   1. Skema 7 bit : ditandai dengan angka 00
   2. Skema 8 bit : ditandai dengan angka lebih besar dari 0

   Kebanyakan ponsel/SMS *Gateway* yang ada dipasaran sekarang menggunakan skema 7 bit sehingga digunakan 00.

   a. Jangka Waktu Sebelum *SMS Expired*

Tabel 2.2 *Batas Waktu Validitas SMS*

Integer	Jangka Waktu Validitas SMS
0-143	(INT+1) x 5 menit (berarti: 5 menit s/d
144-167	12 jam + ((INT-143)x30 menit)
168-196	(INT-166) x 1 hari
197-255	(INT-192) x 1 minggu

Agar SMS pasti terkirim sampai ke ponsel penerima, sebaiknya tidak diberi batasan waktu validnya.

b. Isi SMS

Header ini terdiri atas dua subheader, yaitu :

- Panjang isi (jumlah huruf dari isi)

  Misalnya untuk kata “hello” : ada 5 huruf : 05

- Isi berupa pasangan bilangan heksa

  Untuk ponsel/SMS Gateway berskema encoding 7 bit, jika mengetikan suatu huruf dari keypad-nya, berarti kita telah membuat 7 angka I/O berturutan.

Ada dua langkah untuk mengkonversikan isi SMS, yaitu

- Langkah pertama: mengubahnya menjadi kode
- Langkah kedua mengubah kode 7 bit menjadi 8 bit diwakili oleh pasangan heksa.

Contoh: Untuk kata “hello”

Langkah pertama:

Bit 7 0
H 1101000
E 1100101
L 1101100
L 1101100
O 1101111

H	1110 1000	E8
E	0011 0010 1	32
L	1001 1011 00	9B
L	1111 1101 100	FD
O	0000 0110 1111	06

Oleh karena total 7 bit x 5 huruf = 35 bit, sedangkan yang kita perlukan adalah 8 bit x 5 huruf = 40 bit, maka diperlukan 5 bit dummy yang diisi dengan bilangan 0.

Setiap 8 bit mewakili suatu pasangan heksa. Tiap 4 bit mewakili satu angka heksa, tentu saja karena secara logika 2 = 16.

Dengan demikian kata "*hello*" hasil konversinya menjadi E8329BFD06.

2.1.9 Menggabungkan Header

Secara lengkap PDU untuk kirim SMS adalah sebagai berikut. Untuk mengirimkan kata hello ke ponsel nomor 628129573337 lewat *SMS-Center*, tanpa membatasi jangka waktu valid, maka header PDU lengkapnya :

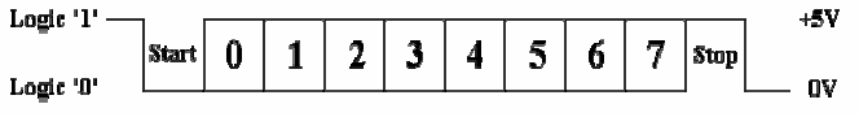
07912618485400F901000C9126189275337300000 5E8329BFD06

2.2 Komunikasi Serial RS232

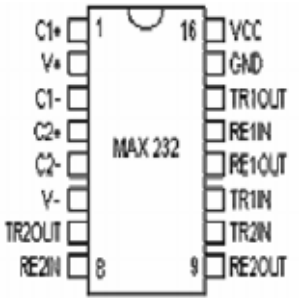
Komunikasi data serial sangat berbeda dengan format pemindahan data pararel. Disini, pengiriman bit-bit tidak dilakukan sekaligus melalui saluran pararel, tetapi setiap bit dikirimkan satu persatu melalui saluran tunggal (perhatikan ***Gambar 2.3)***. Dalam pengiriman data secara serial harus ada sinkronisasi atau penyesuaian antara pengirim dan penerima agar data yang dikirimkan dapat diterima dengan tepat dan benar oleh penerima. Salah satu mode transmisi dalam komunikasi serial adalah *mode asynchronous*. Transmisi serial mode ini digunakan apabila pengiriman data dilakukan satu karakter tiap pengiriman. Antara satu karakter dengan yang lainnya tidak ada waktu antara yang tetap.Karakter dapat dikirimkan sekaligus ataupun beberapa karakter kemudian berhenti untuk waktu yang tidak tentu, kemudian dikirimkan sisanya. Dengan demikian bit-bit data ini dikirimkan dengan periode yang acak sehingga pada sisi penerima data akan diterima kapan saja. Adapun sinkronisasi yang terjadi pada mode transmisi ini adalah dengan memberikan bit-bit penanda awal dari data dan penanda akhir dari data pada sisi pengirim maupun dari sisi penerima.

Format data komunikasi serial terdiri dari parameter-parameter yang dipakai untuk menentukan bentuk data serial yang dikomunikasikan, dimana elemen-elemennya terdiri dari.

1. Kecepatan mobilisasi data per bit (*baud rate*)
2. Jumlah bit data per karakter (*data length*)
3. Parity yang digunakan
4. Jumlah stop bit dan start bit



Gambar 2.3 *Format Pengiriman Data Serial*



Gambar 2.4 *IC Serial MAX 232*

RS232 sebagai komunikasi serial mempunyai 9 pin yang memiliki fungsi masing-masing. Pin yang biasa digunakan adalah pin 2 sebagai *received* data, pin 3 sebagai *transmited* data, dan pin 5 sebagai *ground signal*. Karakteristik elektrik dari RS232 adalah sebagai berikut:

- Space (logic 0) mempunyai level tegangan sebesar +3s/d+25Volt.
- Mark (logic 1) mempunyai level tegangan sebesar -3 s/d -25 Volt.
- Level tegangan antara +3 s/d -3 Volt tidak terdefinisikan.
- Arus yang melalui rangkaian tidak boleh melebihi dari 500 mA., ini dibutuhkan agar sistem yang dibangun bekerja dengan akurat.

2.3 Mikrokontroller AT89C52

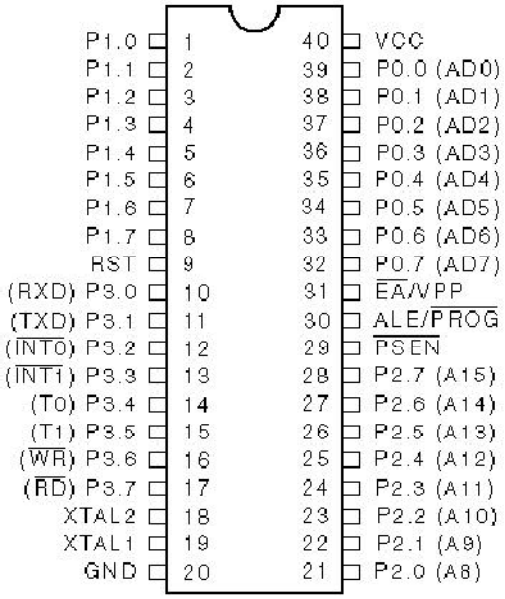
Mikrokontroler AT89C52 merupakan salah satu keluarga dari MCS-51 keluaran Atmel dengan 4 Kbyte Flash *Programmable and Erasable Read Only Memory* (PEROM). Flash PEROM merupakan memori dengan teknologi non-volatile memory, isi memory tersebut dapat diisi ulang ataupun dihapus berkali-kali.

Memori ini biasa digunakan untuk menyimpan instruksi (perintah) berstandar MCS-51 code. Jenis mikrokontroller ini pada dasarnya dapat digunakan untuk mengolah data per bit maupun data 8 bit secara bersamaan. Sebuah mikrokontroler dapat bekerja apabila dalam mikrokontroller tersebut terdapat sebuah program yang berisi instruksi-instruksi yang akan digunakan untuk menjalankan sistem mikrokontroller tersebut. Mikrokontroler AT89C51 ini merupakan mikrokontroler satu chip yang mempunyai konfigurasi sebagai berikut.

- CPU (*Central Processing Unit*) 8 bit yang termasuk keluarga dari MCS51.
- Kemampuan boolean processor (logika 1 bit).
- 32 jalur port I/O bidirectional, dapat dialamati tiap port.
- Program memori internal berupa Flash ROM dengan kapasitas 4KB.
- RAM internal sebesar 128 byte.
- Dua buah timer internal 16 bit.
- Satu buah port komunikasi serial.
- Dua buah interupsi eksternal.

Susunan pin-pin pada Mikrokontroller AT89C51 diperlihatkan pada ***Gambar 2.5***.

- Port 0 (1 byte) terletak pada pin 32 hingga 39.
- Port 1 (1 byte) terletak pada pin 1 hingga 8.
- Port 2 (1 byte) terletak pada pin 21 hingga 28.
- Port 3 (1 byte) terletak pada pin 10 hingga 17.
- Kristal dipasang pada pin 18 dan 19.
- Reset terletak pada pin 9.
- Pin catudaya positif 5V pada kaki 40, ground pada kaki 20.



Gambar 2.5 *Konfigurasi Pin AT89C52*

2.4 Bahasa *Assembly*

Assembler adalah program komputer yang men-*translitrasi* program dari bahasa *assembly* ke bahasa mesin. Sedangkan bahasa *assembly* adalah ekuivalensi bahasa mesin dalam bentuk alpanumerik. *Mnemonics* alpanumerik digunakan sebagai alat bantu bagi programer untuk memprogram mesin komputer daripada menggunakan serangkaian 0 dan 1 (bahasa mesin) yang panjang dan rumit.

2.4.1 Konstruksi Program *Assembly*

Program sumber *assembly* terdiri dari kumpulan baris-baris perintah dan biasanya di simpan dengan extension .ASM dengan 1 baris untuk satu perintah, setiap baris perintah tersebut bisa terdiri atas beberapa bagian, yakni bagian label, bagian *nmemonic*, bagian operand yang bisa lebih dari satu dan bagian terakhir bagian komentar. Program sumber (*source code*) dibuat dengan program Pinnacle 52.

Hasil kerja program yang telah di kompile dalam bahasa *assembler* ini adalah "*assembly listing*". Dan juga "file yang ektensi HEX". File dengan ektensi HEX inilah yang akan diisikan kedalam *Chip Mikrokontroler*. Ketentuan penulisan *source code* adalah sebagai berikut:

1. Masing-masing bagian dipisahkan dengan spasi atau TAB, khusus untuk *operand* yang lebih dari satu masing-masing operand dipisahkan dengan tanda koma.
2. Bagian-bagian tersebut tidak harus semuanya ada dalam sebuah baris, jika ada satu bagian yang tidak ada maka spasi atau TAB sebagai pemisah bagian tetap harus ditulis.
3. Bagian label ditulis mulai huruf pertama dari baris, jika baris bersangkutan tidak mengandung label maka label tersebut digantikan dengan spasi atau TAB, yakni sebagai tanda pemisah antara bagian label dan bagian *mnemonic*.

2.4.2 Instruksi Yang Digunakan

Beberapa instruksi yang digunakan dalam penyusunan program adalah sebagai berikut :

1. EQU

   Konstanta data yang dideklarasikan dengan nilai pada variabel tertentu.
2. BIT

   Konstanta bit yang di deklarasikan dengan 1 bit data variabel tertentu.
3. ORG

   Digunakan untuk menunjukkan lokasi memori tempat instruksi atau perintah yang ada dibawahnya disimpan.

4. CLR

   Memberikan nilai “nol” pada bit tertentu.

5. SETB

   Memberikan nilai “satu” pada bit tertentu.

6. CALL

   Instruksi melakukan lompatan dengan area sebesar 2 Kbyte.

7. LCALL

   Sama seperti instruksi CALL, hanya instruksi ini digunakan jika label yang dipanggil letaknya lebih jauh dari 2Kbyte.

8. MOV

   Instruksi ini melakukan peindahan data variabel pada kode operasi kedua dan disimpan di variabel pada kode operasi pertama.

9. MOVC

   Instruksi ini melakukan pemindahan data variabel pada kode operasi kedua ditambah *carry* dan disimpan di variabel pada kode operasi pertama.

10. JMP

    Melakukan lompatan dan menjalankan program yang berada di alamat yang di tentukan oleh label tertentu.

11. SJMP

    Melakukan lompatan untuk jarak yang pendek (*Short JMP*).

12. LJMP

    Melakukan lompatan untuk jarak yang jauh (*Long JMP*).

13. JZ

    Lompat jika akumulator bernilai “nol”. Jika semua bit pada akumulator bernilai “nol”, maka lompat ke alamat yang telah ditentukan dan di lanjutkan ke interuksi berikutnya.

14. JB

    Lompat jika bit tertentu bernilai “satu”, artinya jika bit yang telah ditentukan bernilai “satu”, maka lompat ke alamat yang telah ditentukan dan dilanjutkan ke instruksi berikutnya.

15. JNB

    Kebalikan dari JB, JNB merupakan instruksi untuk beralih ke alamat tertentu jika bit tertentu bernilai “nol”.

16. DJNZ

    Melakukan pengurangan pada Rn (R0…R7) dengan 1 dan lompat ke alamat yang ditetukan apabila hasilnya bukan 00. Apabila hasilnya telah mencapai 00, maka program akan terus menjalankan instruksi dibawahnya.

17. CJNE

    Instruksi ini melakukan perbandingan antara data tujuan dan data sumber serta melakukan lompatan ke alamat yang ditentukan jika hasil perbandingan tidak sama.

18. RET

    Instruksi ini melakukan lompatan ke alamat yang disimpan dalam SP dan SP-1. Instruksi isi biasa digunakan pada saat kembali dari subrutin yang di panggil dengan ACALL datau LCALL.

19. SWAP

    Baca 4 bit (*nibble*) data pertama LSB dari akumulator. Perintah SWAP memungkinkan untuk menukar 4 bit rendah menjadi 4 bit tinggi dari akumulator (berlaku sebaliknya).

20. PUSH

    Melakukan penyimpanan data dari suatu register atau memori kedalam stack. Lokasi data dalam stack tersebut dtunjuk oleh *stuck pointer*. Pada saat instuksi ini dijalankan, nilai dari *stuck pointer* akan bertambah satu dan register atau memori yang di PUSH akan masuk kedalam alamat yang ditunjuk oleh *stuck pointer* tersebut.

21. POP

    Melakukan pengambilan data dari dalam stuck ke suatu register atau memori. Lokasi data dalam stack tersebut ditunjuk oleh stack pointer. Pada saat instruksi ini dijalankan, data di alamat yang ditunjuk oleh stuck pointer akan dipindah ke register atau memori dilanjutkan dengan pengurangan nilai stuck pointer dengan 1.

22. ANL

    Logika AND untuk variabel tertentu. Perintah ANL bekerja dengan melakukan operasi AND antara variabel yang ditentukan dengan nilai tertentu untuk kemudian hasilnya disimpan pada alamat yang ditunjuk oleh variabel yang ditentukan.

23. ORL

    Logika OR untuk variabel tertentu. Perintah ORL bekerja dengan melakukan operasi OR antara variabel yang ditentukan dengan nilai

tertentu untuk kemudian hasilnya disimpan pada alamat yang ditunjuk oleh variabel yang ditentukan.

24. CPL

    Tiap bit dari akumulator yang secara logika dikomplemenkan satu. Misalnya nilai yang asalnya 1, setelah di komplemenkan akan berubah menjadi 0. Dari faktor tersebut, CPL dapat diartikan sebagai instruksi untuk menukar nilai 0 menjadi 1 dan sebaliknya.

25. INC

    Menambahkan nilai variabel yang ditunjuk dengan 1 dan hasilnya disimpan di variabel tersebut.

26. SUBB

    Mengurangkan nilai variabel yang ditunjuk dengan nilai lain dan hasilnya disimpan divariabel tersebut.

27. ADD

    Menjumlahkan nilai variabel yang ditunjuk dengan nilai lain dan hasilnya disimpan di variabel tersebut.

28. ADDC

    Menjumlahkan nilai variabel yang ditunjuk dengan nilai dan ditambah dengan nilai *carry* yang kemudian hasilnya di simpan di variabel yang di tunjuk tersebut.

29. NOP

    No Operation, yang berarti tidak melakukan apa-apa. Tujuan dari instruksi ini adalah hanya untuk menambahkan siklus pulsa pewaktu sebesar 1 siklus.

30. DW

    Define Word, yang artinya mengacu pada data word (data 16bit) tertentu.

31. DB

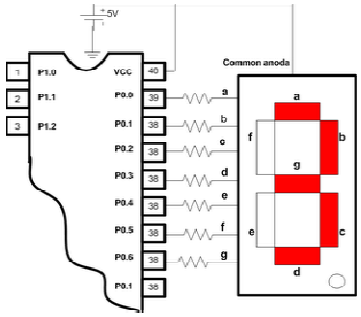
    Define Byte, yang artinya mengacu pada byte (data 8bit) tertentu.

32. END

    END Biasanya diletakkan di akhir baris file program sumber *assembler* sebagai tanda akhir pernyataan (*statement*) bagi program *assembler* dalam melakukan proses *assembly*.

2.5 Pemecah Sandi pada *Seven Segment*

Pemecah sandi merupakan suatu rangkaian logika terintegrasi yang berfungsi untuk menampilkan kode-kode *biner* menjadi tanda-tanda yang dapat di tanggapi secara *visual*. Sesuai dengan ragam cara penyandian, maka dapat kita jumpai beragam tipe *decoder*, yang salah satunya *decoder* BCD. Decoder ini mempunyai 4 saluran masuk dan 7 saluran keluar. Sinyal keluaran 0 (nol) dari decoder ini mengaktifkan (menyalakan) salah satu ruas led pada peraga tujuh ruas.



Gambar 2.6 *Pencacah Sandi BCD ke 7 Ruas*

Untuk menyatakan bilangan desimal dalam peraga tujuh ruas, maka tabel kebenarannya seperti di tunjukan pada ***Tabel 2.3***. Berdasarkan tabel tersebut dapat

di dapat di tentukan fungsi logika dari masing-masing ruas. Fungsi tersebut dikembangkan sehingga kita hanya memerlukan gerbang AND, OR dan NOR masing-masing satu IC untuk menyusun rangkaian logika tiap ruas.

Tabel 2.3 *Tabel Kebenaran Sevent Segment*

Desimal	Hexa Desimal	D	C	B	A	a	b	c	d	e	f	g	7-LED
0	0	0	0	0	0	0	0	0	0	0	0	1	
1	1	0	0	0	1	1	0	0	1	1	1	1	
2	2	0	0	1	0	0	0	1	0	0	1	0	
3	3	0	0	1	1	0	0	0	0	1	1	0	
4	4	0	1	0	0	1	0	0	1	1	0	0	
5	5	0	1	0	1	0	1	0	0	1	0	0	
6	6	0	1	1	0	1	1	0	0	0	0	0	
7	7	0	1	1	1	0	0	0	1	1	1	1	
8	8	1	0	0	0	0	0	0	0	0	0	0	
9	9	1	0	0	1	0	0	0	1	1	0	0	
10	A	1	0	1	0	1	1	1	0	0	1	0	
11	B	1	0	1	1	1	1	0	0	1	1	0	
12	C	1	1	0	0	1	0	1	1	1	0	0	
13	D	1	1	0	1	0	1	1	0	1	0	0	
14	E	1	1	1	0	1	1	1	0	0	0	0	
15	F	1	1	1	1	1	1	1	1	1	1	1	