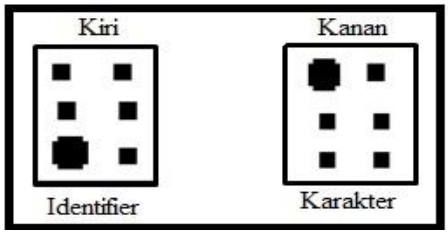
BAB II
LANDASAN TEORI

2.1 Kode Braille

Braille adalah sejenis sistem tulisan sentuh yang digunakan oleh penyadang cacat tunanetra. Sistem ini awalnya dirancang oleh Louis Braille yang berkewarganegaraan Prancis yang buta sejak kecil. Ketika usia 15 tahun, Louis Braille mengubah bentuk tulisan latin biasa menjadi tulisan sentuh yang dapat digunakan tentara untuk membaca dalam gelap. Sistem ini dinamakan sistem braille dengan tujuan mendapatkan kemudahan dalam membaca. Namun saat itu braille tidak mempunyai huruf W, tetapi sekarang braille mempunyai huruf W. ada beberapa versi tulisan braille diberbagai negara, yaitu Standar Braille, American Modified Braille, US Computer Braille, US Extended Computer Braille, Euro Computer Braille , ISO 8859-I Braille, Russian Braille, Greek Braille, Hebrew Braille, Arab Braille, Japanese Braille, Korean Braille, Chinese Braille, Braille ASCII, Unicode.

Selain standar braille, umumnya semua versi diatas memiliki perbedaan dalam hal, yaitu membedakan antara huruf besar dan huruf kecil dalam satu blok, membedakan antara huruf besar dan angka dalam satu blok, banyaknya jenis karakter yang dapat diterjemahkan kedalam kode braille dan jumlah dot yang digunakan (st*andar braille menggunakan 6 dot sedangkan versi lama braille menggunakan 8 dot*).



Gambar 2.1 Contoh Kode Braille Huruf a

2.2 Perangkat Keras

2.2.1 Mikrokontroler Basic Stamp (BS2P40)

Basic Stamp adalah suatu mikrokontroler yang dikembangkan oleh *Parallax Inc* yang diprogram menggunakan bahasa pemrograman *basic* dan populer sekitar pada tahun 1990-an. Mikrokontroler Basic Stamp membutuhkan catu daya saat men-*download* dan program di *download* melaui *port* serial.

Kode *PBasic* (Pemograman *Basic*) disimpan di dalam EEPROM serial pada *board* basic stamp. EEPROM digunakan dalam basic stamp 1 dan 2 yang dijamin menyimpan selama 40 tahun ke depan dan mampu ditulisi ulang 10.000.000 kali per lokasi memori.

Mikrokontroler Basic Stamp memiliki beberapa versi yang berbeda–beda, yaitu basic stamp 1, basic stamp 2, basic stamp 1e, basic stamp 2P, basic stamp 2Pe dan basic stamp 2sx. Basic Stamp jalan pada tegangan DC 5 sampai 15 volt. Basic Stamp yang dipakai adalah BS2P40 yang mempunyai 40 *pin I/O*. Pemilihan basic stamp ini karena membutuhkan banyak masukan atau keluaran yang dipakai dalam pengontrolan *solenoid*. Berikut ini adalah tampilan mikrokontroler BS2P40.



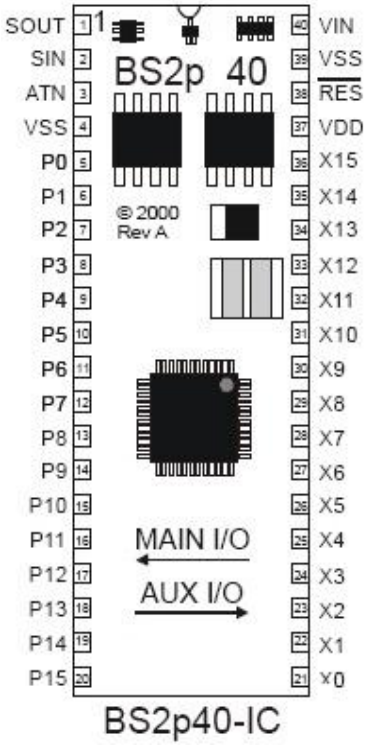
Gambar 2.2 Modul Mikrokontroler BS2P40

Modul Basic Stamp 2p40 mempunyai spesifikasi perangkat keras sebagai berikut:

1. Mikrokontroler Basic Stamp 2P40 *Interpreter Chip* (P*BASIC*48W/P40)
2. 8 x 2Kbyte EEPROM yang mampu menampung hingga 4000 perintah.

3. Kecepatan prosesor 20MHz Turbo dengan kecepatan eksekusi program hingga 12000 instruksi per detik.
4. RAM sebesar 38 *byte* (12 I/O, 26 variabel) dengan *Scratch Pad* sebesar 128 *byte*.
5. Jalur masukan/keluaran sebanyak 32 *pin*.
6. Tersedia jalur komunikasi serial UART RS-232 dengan konektor DB9.
7. Tegangan masukan 9 – 12 V_{DC} dengan tegangan keluaran 5 V_{DC}.

Berikut ini adalah alokasi *pin* yang terdapat pada mikrokontroler BS2P40.



Gambar 2.3 Alokasi *Pin* Basic Stamp

2.2.2 Pengenalan Basic Stamp Editor

Perangkat lunak merupakan faktor penting dalam tahap perancangan alat bantu komunikasi antara tunanetra dan tunarungu. Perangkat lunak ini merupakan algoritma atau *listing* program yang ditanamkan kedalam mikrokontroler. Program dapat bermacam-macam bentuk dan bahasanya sesuai dengan spesifikasi dari mikrokontroler yang digunakan.

Mikrokontroler BS2P40 menggunakan bahasa pemrograman *basic*. Perangkat lunak yang digunakan adalah basic stamp *editor*. Basic stamp *editor* adalah sebuh *editor* yang dibuat oleh Paralax-Inc untuk menulis *listing* program meng-*compile* dan men-*download*-nya ke mikrokontroler keluarga basic stamp. Program ini memungkinkan penggunanya memprogram Basic Stamp dengan bahasa *basic* yang relatif ringan dibandingkan bahasa pemrograman lainnya. Berikut ini beberapa perintah-perintah dasar yang dapat digunakan pada mikrokontroler BS2P40.

Tabel 2.1 Beberapa Perintah Dasar Basic Stamp

Instruksi	Keterangan
DO...LOOP	Perulangan
GOSUB	Memanggil prosedur
IF..THEN	Percabangan
FOR...NEXT	Perulangan
PAUSE	Waktu tunda milidetik
IF...THEN	Perbandingan
PULSIN	Menerima pulsa
GOTO	Loncat ke alamat memori tertentu
HIGH	Menset *pin* I/O menjadi 1
LOW	Menset *pin* I/O menjadi 0

2.2.2.1 Memprogram Basic stamp

Dalam membuat sebuah program secara umum, dapat dibagi menjadi 4 bagian penting, yaitu;

1. *Header*

Header ditulis paling awal *listing* program yang dibuat. Bagian ini menentukan tiga prosesor yang digunakan dan versi dari *compiler PBASIC* yang digunakan untuk meng-*compile* bahasa *basic* menjadi bahasa mesin.

' {$*STAMP* BS2}
' {$P*BASIC* 2.5}

2. *Variable*

Beberapa ketentuan untuk mendeklarasikan variabel dalam mikrokontroler

PIN : *PIN* dari mikrokontroler (0-15)

VAR : variable

PIN yang digunakan sudah ditentukan sesuai dengan konfigurasi perangkat keras yang digunakan adalah BS2P40.

3. Program utama

Pada bagian program utama bisa melakukan dua mode yaitu mode pengetikan langsung atau mode pemanggilan prosedur. Mode pengetikan langsung akan efektif jika program tidak terlalu banyak dan kasus yang sederhana, tetapi jika program sudah mulai banyak atau rumit, maka sebaiknya program utama memanggil prosedur. Pemanggilan prosedur akan mempermudah dalam pemeriksaan dan lebih terkendali.

Contoh program:

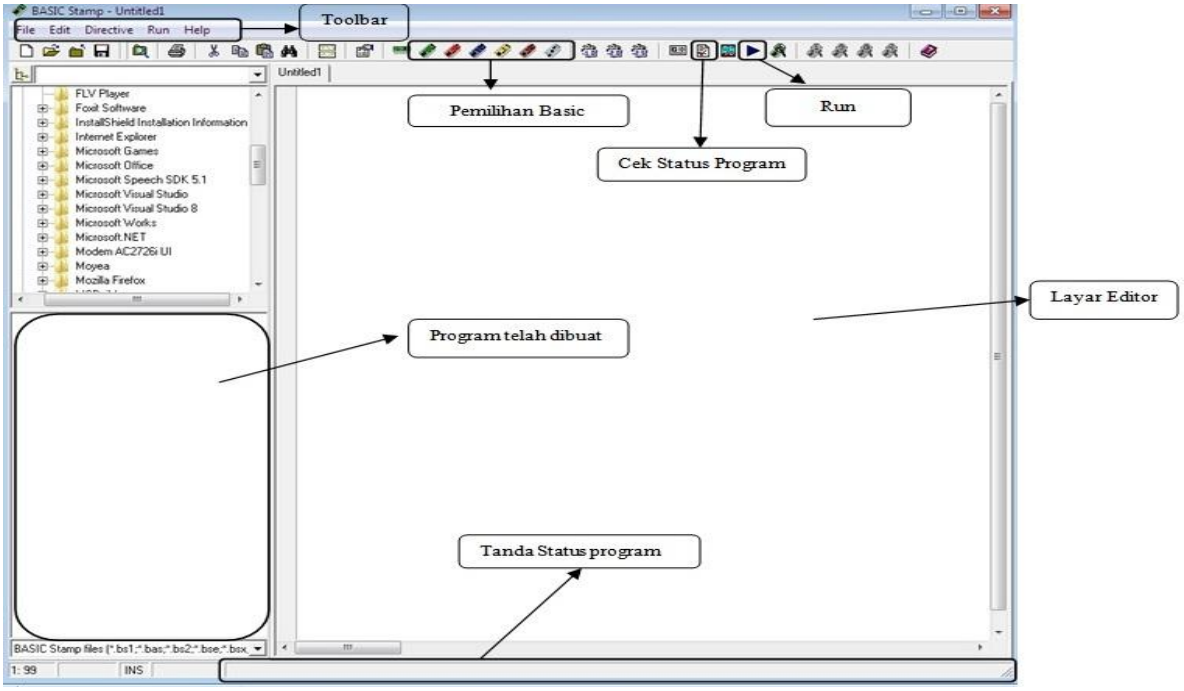
```
Main:
HIGH 8          'LED Mati
PAUSE 1000      ‘delay 1 detik
LOW 8           LED Hidup
PAUSE 5000      'delay 5 detik
GOTO Main       ‘kembali ke main
```

4. Prosedur

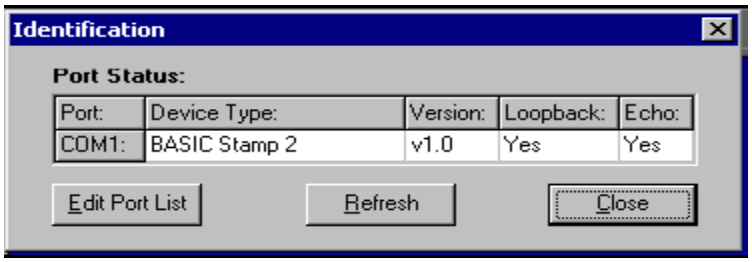
Sebuah prosedur harus mempunyai nama prosedur yang disimpan dibagian paling atas prosedur itu sendiri, serta harus diakhiri dengan *return* agar kembali lagi ke program utama dan melanjutkan kembali urutan program berikutnya.

Contoh Program:

Main:
GOSUB Hello
DEBUG "How are you?"
END
Hello:
DEBUG "Hello, my friend!", CR
RETURN



Gambar 2.4 Layar *Editor* Basic Stamp

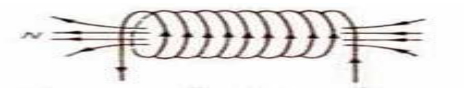


Gambar *2.5* Download *Program*

2.2.3 Solenoid

Solenoid adalah alat yang dapat meng-*konversi* sinyal elektrik atau arus listrik menjadi gerak mekanik. *Solenoid* dibuat dari kumparan dan inti besi yang dapat digerakkan yang berfungsi sebagai aktuator pada alat bantu kode braille.

Secara skematik bentuk dari *solenoid* dapat dilihat pada gambar 2.6, dimana *solenoid* terdiri dari n buah lilitan kawat berarus listrik (I), medan magnet yang dihasilkan memiliki arah seperti pada gambar 2.6, dimana kutub utara magnet mengikuti aturan tangan kanan.



Gambar 2.6 *Solenoid* dengan lilitan n



Gambar 2.7 *Solenoid*

Kuat medan magnet pada *solenoid* dengan jumlah lilitan persatuan panjang n dapat dihitung dengan menggunakan persamaan dibawah ini:

$B = \mu_0 . n . I$ ……(2.1)

dimana:

$n = \frac{N}{l}$ ……(2.2)

keterangan:

B = Medan magnet

μ_0 = Konstanta permeabilitas udara

n = jumlah lilitan persatuan panjang

N = Jumlah lilitan

l = Panjang lilitan

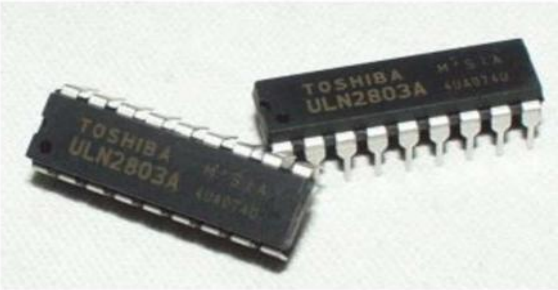
I = Arus

2.2.3.1 Cara Kerja Solenoid

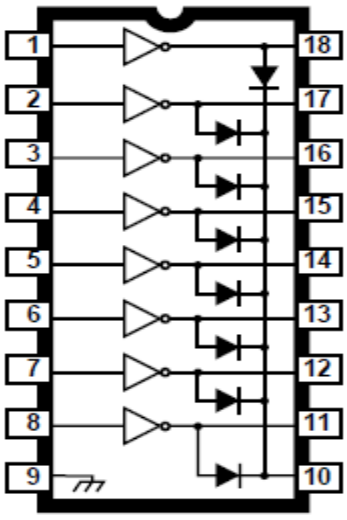
Prinsip kerja dari s*olenoid* berdasarkan pada pengantar yang membawa arus kedalam kumparan sehingga kumparan akan menimbulkan medan magnet. Medan magnet ini dibuat sedemikian rupa sehingga keadaannya selalu tolak-menolak antara medan magnet, sedangkan untuk mengubah arah putaran *solenoid* dengan cara membalikan polaritas sumber tegangannya.

2.2.3.2 Driver Solenoid IC ULN2803

IC ULN2803 merupakan *driver solenoid* dengan desain *dual in-lines packages*. IC ULN2803 memiliki 8 *channel* masukan dan 8 keluaran dan setiap *channel* mengeluarakan arus 500 mA.



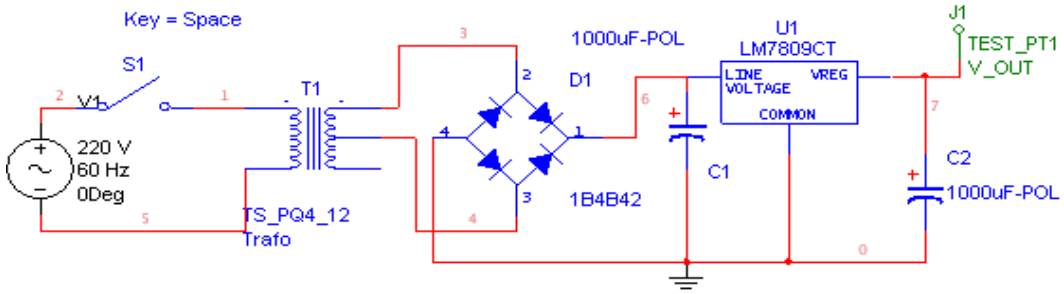
Gambar 2.8 Bentuk Fisik IC ULN2803



Gambar 2.9 Skema dalam ICULN2803

2.2.4 Catu Daya

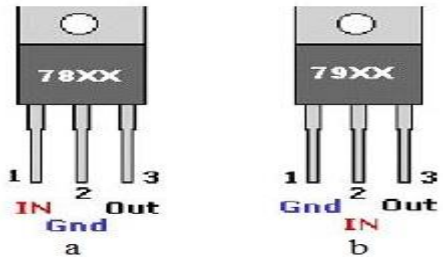
Memperoleh suatu catu daya dengan nilai keluaran yang tetap, maka digunakan IC catu daya 78xx untuk catu daya positif dan IC catu daya 79xx untuk catu daya negatif



Gambar 2.10 Rangkaian Catu Daya Positif

2.2.4.1 LM78xx (IC Catu Daya Tegangan Positif)

Pada gambar 2.11 dibawah ini terlihar bahwa IC LM78xx mempunyai tiga kaki yang digunakan sebagai komponen pendukung dari Vcc untuk menghasilkan tegangan 5, 6, 8, 9, 10, 12, 15, 18, 24 volt. Simbol "xx" menandakan besar tegangan yang dihasilkan pada gambar tersebut, misalkan *output*-nya sebesar 5 volt maka nilai untuk menandakan simbol "xx" tersebut adalah 05, jadi pada IC tersebut akan tertulis LM7805. IC Catu daya ini befungsi untuk menstabilkan tegangan. Penerapan IC ini mengharuskan $V_i > V_o$. IC Catu daya yang digunakan yaitu LM7805 untuk menghasilkan keluaran 5 volt dan LM7809 untuk menghasilkan keluaran 9 volt.



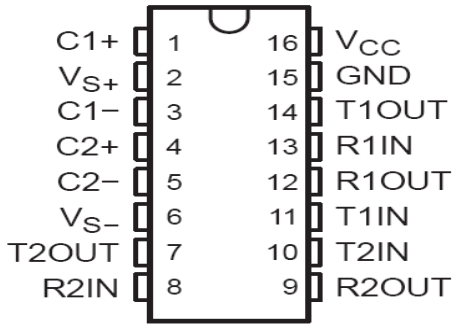
Gambar 2.11 IC Catu Daya

2.2.5 MAX232

Komunikasi yang berhubungan dengan PC tidak bisa dilakukan begitu saja tanpa ada penghubung antara PC dan mikrokontroler. Oleh karena itu mikrokontroler membutuhkan komponen tambahan baik untuk berkomunikasi. Pada pembuatan tugas akhir ini yang digunakan adalah komunikasi serial. Pada mikrokontoler sendiri terdapat *buffer* yang dapat digunakan sebagai pendukung proses komunikasi serial. Pada saat ini banyak komponen yang dapat digunakan untuk pendukung proses komunikasi ini, salah satunya adalah Max232.

Max232 berfungsi sebagai perantara antara mikrokontoler dengan *port* serial, karena mikrokontroler tidak dapat mengirim data begitu saja maka diperlukan Max232.

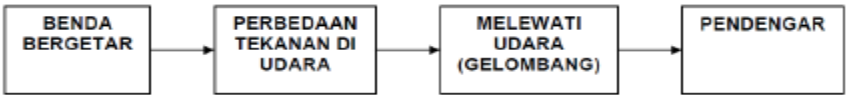
Max232 mempunyai 16 kaki yang terdiri keperluan untuk *port* serial, komunikasi mikrokontoler dengan maxim. Letak dari masing-masing port dapat dilihat pada gambar 2.12



Gambar 2.12 *Konfigurasi Pin* MAX232

2.3 Suara

Suara adalah sesuatu yang dihasilkan oleh getaran yang berasal dari benda bergerak, sesuatu yang menghasilkan bunyi, dan suara adalah sebuah tekanan gelombang udara, maka memiliki nilai kontinu terhadap waktu. Suara atau bunyi biasanya merambat melalui udara dan akan memantul apabila ada dinding, suara atau bunyi tidak dapat merambat melalui ruang hampa.



Gambar 2.13 Skema Suara

Suara berhubungan erat dengan rasa "mendengar". Suara/bunyi biasanya merambat melalui udara. Suara/bunyi tidak bisa merambat melalui ruang hampa. Gelombang suara bervariasi dalam tingkatan tekanan suara (amplitudo) dan dalam frekuensi. Jumlah waktu yang diperlukan untuk terjadinya suatu getaran atau gelombang dinamakan perioda (T), sedangkan jumlah gelombang yang terjadi setiap detik dinamakan frekuensi (f) dengan satuan m/dt (Hz).

Suara yang berada pada jarak pendengaran manusia disebut "*AUDIO*". Suara di luar jarak pedengaran manusia dapat dikatakan "NOISE" (getaran yang tidak beratur dan tidak berurutan dalam berbagai frekuensi, tidak dapat didengar manusia).

2.4 Speech Recognition

Teknologi pengenalan suara adalah teknologi yang menggunakan peralatan dengan sumber masukannya berupa suara, seperti *mikrofon* menginterpretaskan suara manusia sebagai metode alternatif interaksi dengan komputer. Teknologi pengenalan suara tidak sama dengan teknologi *voice recognition* yang hanya mengenali suara sebagai identifikasi keamanan. Sistem komersial untuk pengenalan suara telah ada sejak tahun 1990. Walaupun teknologi ini sukses, masih sedikit orang yang menggunakan sistem pengenalan suara pada komputer. Kebanyakan pengguna komputer dalam berinteraksi dengan komputernya merasa lebih nyaman dengan menggunakan peralatan-peralatan masukan konvensional yaitu keyboard dan mouse, walaupun kenyatanya dengan menggunakan teknologi pengenalan suara memungkinan pengguna untuk berbicara secara langsung dan cepat serta efisien.

2.4.1 Mode Sistem Pengenalan Pembicaraan

Terdapat dua mode pada sistem pengenalan pembicaraan yaitu:

1. Mode *Diktasi*. Pada mode ini pengguna komputer mengucapkan kata atau kalimat yang selanjutnya akan dikenali oleh komputer dan diubah menjadi data teks. Kemungkinan jumlah kata yang dapat dikenali dibatasi oleh jumlah kata yang telah terdapat pada *database*. Pengenalan mode *diktasi* merupakan *speaker dependent*. Keakuratan mode ini tergantung pada pola suara dan akses pembicara serta pelatihan yang telah dilakukan
2. Mode *command and control*. Pada mode ini pengguna komputer mengucapkan kata atau kalimat yang sudah terdefinisi terlebih dahulu pada *database* dan selanjunya digunakan untuk menjalankan perintah tertentu pada aplikasi komputer, jumlah perintah yang dikenali tergantung dari aplikasi yang telah didefinisikan terlebih dahulu pada *database*. Mode ini merupakan *speaker independent* karena jumlah kata yang dikenali biasanya terbatas sekali dan ada kemungkinan pembicara tidak perlu melakukan pelatihan pada sistem sebelumnya.

Terdapat empat proses utama pada sistem pengenalan pembicaraan, baik pada mode *diktasi* ataupun *command and control* yaitu: pemisahan kata, ketergantungan terhadap pengguna, pencocokan kata dan perbendaharaan kata.

2.4.2 Pemisahan Kata

Pemisahan kata adalah proses untuk memisahkan suara yang diucapkan oleh pengguna menjadi beberapa bagian. Masing-masing bagian dapat berupa kalimat atau satu kata. Terdapat tiga macam metode yang digunakan pada proses pemisahan kata ini yaitu: *discrete speech*, *word spotting* dan *continuous speech*.

1. Pada *discrete speech*, pengguna diharuskan mengucapkan kalimat secara terpenggal dengan adanya jeda sejenak diantara kata. Jeda tersebut digunakan oleh sistem untuk mendeteksi awal dan akhir sebuah kata. *Discrete speech* ini mempunyai kelebihan yaitu sedikit *resource* yang digunakan oleh sistem untuk mendeteksi suara, tetapi mempunyai kelemahan yaitu ketidaknyamanan pengguna dalam mengucapkan sebuah kata atau kalimat.
2. Pada *word spotting*, dalam sebuah kalimat yang diucapkan pengguna, sistem hanya mendeteksi kata yang terdapat di dalam perbendaharaan yang dimilikinya,

dan mengabaikan kata lain yang tidak dimilikinya. Sehingga walau pengguna mengucapkan kalimat yang berbeda, bila di dalam kalimat tersebut terdapat sebuah kata yang sama dan terdapat di perbedaharaan sistem, maka hasil pengenalan akan sama. Kelemahan metode ini adalah besar kemungkinan sistem melakukan kesalahan arti pengenalan dalam bentuk kalimat. Tetapi metode ini mempunyai kelebihan yaitu pengguna dapat mengucapkan kalimat secara normal tanpa harus berhenti diantara kata.

3. Pada metode *continous speech*, sistem akan mengenali dan memproses setiap kata yang diucapkan. Metode ini cukup akurat dalam mengenali ucapan pengguna. Tetapi di sam*pin*g itu metode ini memerlukan *resource* yang besar dalam prosesnya. Pada metode ini, sistem harus dapat mendeteksi awal dan akhir setiap kata dalam kalimat tanpa adanya jeda diantara kata tersebut, dan setelah berhasil memisahkan kata, langkah selanjutnya adalah mencocokkan dengan perbendaharaan kata yang dimilikinya.

2.4.3 Sifat Pengenalan Pembicaran

Sistem pengenalan pembicaraan mempunyai beberapa sifat dilihat dari ketergantungan tehadap pengguna yaitu *speaker dependent*, *speaker independent* dan *speaker adaptive.*

1. *Speaker Dependent*, sistem membutuhkan pelatihan untuk setiap pengguna yang akan menggunakan sistem tersebut. Sistem tidak akan dapat mengenali pengguna yang belum pernah melakukan pelatihan.
2. *Speaker Independent*, pengguna tidak perlu melakukan pelatihan sebelum dapat menggunakan sistem, karena sistem mampu mengenali suara semua pengguna tidak tergantung warna suara dan dialek yang digunakan.
3. *Speaker Adaptive* merupakan perpaduan dari *speaker dependent* dan *speaker independent*, dimana pengguna tidak perlu melakukan pelatihan dan keakuratan pengenalan sistem akan makin meningkat jika pengguna yang sama bekerja terus menerus selama beberapa waktu tertentu.

2.4.4 Pencocokan Kata

Pencocokan kata adalah proses untuk mencocokkan kata ucapan yang berhasil di-identifikasidengan *database* yang dimiliki oleh sistem. Terdapat dua metode yang dapat dipakai pada proses pencocokan kata ini, yaitu: *whole-word matching* dan *phoneme matching*.

1. Pada *whole-word matching*, sistem akan mencari di dalam *database* kata yang tepat atau persis dengan hasil ucapan pengguna
2. *Phoneme Matching*, sistem memiliki kamus *fonem*.

   (*fonem* adalah bagian tekecil dan unik dari suara yang membentuk sebuah kata)

2.4.5 Perbendaharaan Kata

Perbendaharaan kata adalah bagian terakhir dalam sebuah pengenalan pembicaraan. Terdapat dua hal yang perlu diperhatikan pada perbendaharaan kata, yaitu ukuran dan keakuratan. Jika perbendaharaan kata berjumlah banyak maka sebuah sistem akan mudah dalam melakukan pencocokan kata, tetapi dengan makin meningkatnya jumlah perbendaharaan kata, maka jumlah kata yang menpunyai ucapan hampir sama juga meningkat, hal ini menurunkan keakuratan pengenalan. Dan sebaliknya, jika sebuah sistem mempunyai perbendaharaan kata sedikit, maka keakuratan pengenalan akan tinggi karena sedikitnya kata yang hampir sama, tetapi akan semakin banyak kata yang tidak akan dikenali. Untuk sistem pengenalan pembicaraan dengan mode *command and control*, akan lebih baik jika menggunakan jumlah perbendaharaan kata sedikit (kurang dari 100 kata), tetapi untuk mode *diktasi* akan membutuhkan jumlah perbendaharan kata yang banyak.

2.4.6 Perangkat yang Dibutuhkan untuk Pengenalan Suara Menggunakan SAPI 5.1

Sistem suara membutuhkan sumber daya yang intensif. *Engine* pengenalan suara membutuhkan RAM(*Random Access Memory*) yang cukup dan *disk space* untuk merespon permintaan pengguna dengan cepat. Oleh karena itu, sangat penting untuk menetapkan perangkat keras dan perangkat lunak ketika membuat model dan

mengimplementasikannya. Tidak semua komputer memiliki *memory, disk space* dan perangkat keras yang dibutuhkan untuk mengimplementasikan pengenalan suara dan *text-to-speech*. Dibawah ini ada tiga kategori umum dari sumber daya yang dibutuhkan:

1. Perangkat keras umum seperti kecepatan processor dan RAM

   Sistem suara membutuhkan sumber suara processor dan RAM. *Speech Recognitio*n (SR) *services* membutuhkan variasi tingkatan sumber daya yang tergantung pada jenis *engine* pengenalan suara yang sudah terinstal dan tingkatan dari *services* yang diimplementasikan, sedangkan kebutuhan *text-to -speech* (TTS) *engine* lebih stabil karena hanya tergantung pada *engine-text-to- speech* yang diinstal. SR dan TTS *engine* yang sekarang tersedia untuk SAPI 5.1 yang biasanya dapat berhasil diimplementasikan menggunkan 486/33 processor *chip* dan tambahan 1MB RAM. Tetapi *konfigurasi* seperti diatas direkomendasikan. Disarankan processor yang digunakan adalah Pentium Processor (P60 atau lebih) dan sedikitnya memiliki 16 MB dari total RAM. Sistem yang mendukung *diktasi* pengenalan suara membutuhkan sumber daya yang lebih besar. Pada umumnya yang menerapkan *command and control* (CnC) *services* hanya membutuhkan 1MB dari RAM. *Dictation sevices* akan membutuhkan sedikitnya 8 MB dari RAM. Tipe dari sam*pin*g suara, analisa dan ukuran dari pengenalan kosakata semuanya mempengaruhi sumber daya yang dibutuhkan.

2. Perangkat lunak seperti sistem operasi dan *engine* pengenalan suara dan *enginetext-to-speech.*

   Microsoft speech API hanya dapat diterapkan pada system operasi *Windows* 32 *bit*. Oleh karena itu kita membutuhkan *Windows 95, Windows NT* atau versi yang lebih besar lagi pada workstasion. Perangkat lunak yang sangat penting untuk mengimplementasikan *speech services* adalah SR *engine* dan TTS *engine*. SR dan TTS *engine* adalah *back-end processing module* dari SAPI 5.1 model. Selain perangkat lunak diatas, perangkat lunak yang harus dimiliki adalah sebuah produk dari *Microsoft Speech SDK 5.1*.

3. Perangkat keras khusus seperti *sound cards*, mikrofon, *speaker* dan *headphones*.

2.4.7 Batasan Pengenalan Suara

Terdapat tiga batasan yang sangat vital dari teknologi pengenalan suara, yaitu:

1. Identifikasi pembicaran

   Kesulitan dalam membedakan pembicara dan juga pada sistem *diktasi engine* pengenalan suara tidak dapat membedakan jika ada beberapa pembicara yang berbicara secara bersamaan.

2. Pengenalan input

   Engine pengenalan suara tidak dapat mengenali bahasa alami karena *engine* pengenalan suara hanya dapat mengenali kata yang terdapat pada kosakata yang dimilikinya lalu memprosesnya berdasarkan aturan *grammar* yang *speech engine* diketahui.

3. Akurasi pengenalan

   Akurasi pengenalan dapat dipengaruhi oleh dialek, kualitas dari mikrofon dan *level noise* yang ada selama sesi pengucapan. Sama seperti pada masalah pengenalan suara, variasi dialek dapat mengurangi keberhasilan dari *engine* pengenalan suara.

2.5 Diktasi

Diktasi merupakan salah satu jenis pengenalan suara di mana suatu mesin mendengarkan apa yang diucapkan dan menterjemahkan ke dalam bentuk teks.

Semua itu terjadi di dalam *speech engine*. Kebanyakan dari *dictation engines* yang modern menggunakan suatu langkah, di mana *engine* tersebut mendengarkan kata yang diucapkan dan memecahkan menjadi suatu rangkaian hipotesis kata. Setiap hipotesis kata berisi sebuah daftar kata-kata yang mungkin dengan memberikan beberapa kemungkinan dan menjadi tepat.

Setiap hipotesis kata kemudian disimpan dalam bentuk “context”. Jadi setiap kata dipertimbangkan hubungannya dengan kata-kata sebelum dan sesudahnya. Berdasarkan aturan-aturan pada konteks *speech engine*, maka hasil akhir yang didapat adalah yang terbaik dari kata yang telah diucapkan. Ketepatan dari diktasi tergantung dari kecepatan CPU dan sistem memori yang tersedia. Semakin banyak sumber daya semakin banyak juga konteks yang dapat dipertimbangkan pada suatu waktu dan dapat menghasilkan pengenalan yang akurat.

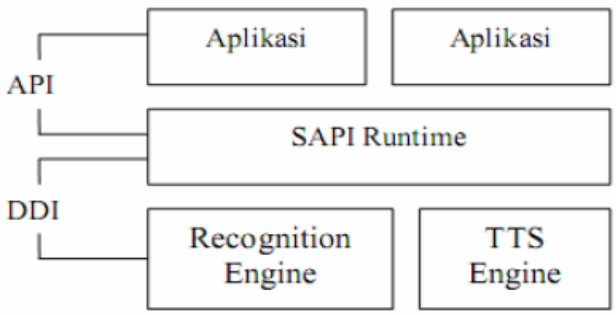
Yang penting untuk keakuratan dari pengenalan diktasi adalah *engine* dapat mengerti suara setiap pembicara. *Speech engines* dikhususkan untuk suatu bahasa tertentu bahkan mungkin hanya untuk suatu daerah. Inilah alasan mengapa ada *English Engine, French Engine, Chinese Engine* dan lain-lain. Pada setiap bahasa tersebut terdapat perbedaan-perbedaan (kadang perbedaan yang ekstrim). Seorang anak perempuan berumur 5 tahun di komputer suaranya sangat berbeda dengan seorang laki-laki berumur 47 tahun. Inilah alasanya mengapa kebanyakan mesin diktasi memerlukan perlatihan suara karena pengenalan mode diktasi merupakan *speaker dependent*, yaitu keakuratan pengenalan mode ini tergantung pada pola suara dan akses pembicara serta pelatihan yang telah dilakukan. Semakin banyak sesi pelatihan yang diselesaikan maka keakuratan pengenalan diktasi yang didapat akan semakin baik.

2.6 Perangkat Lunak

2.6.1 Microsoft Speech Application Programming Interface

SAPI 5.1 (*Speech Application Programming Interface*) merupakan API(*Application Programming Interface)* yang dikembangkan oleh *Microsoft* untuk digunakan pada *speech recognition* dan *speech synthesis* dengan sistem operasi berbasis Windows. Dengan menggunakan API tersebut dimungkinkan bagi pihak ketiga 3rd *party company* atau individu untuk mengembangkan perangkat *Speech Recognition* yang dapat berkerjasama dengan SAPI 5.1 tersebut.

Diperkenalkan oleh *Microsoft* pada tahun 1995, SAPI 5.1 memungkinkan sistem untuk mengenali suara manusia sebagai masukan dan keluaran berupa suara manusia dari *text*. Fasilitas ini menyediakan hubungan interaksi baru antara manusia dengan komputer, SAPI 5.1 merupakan bagian dari *Windows Open Services Architecture Model*. Pada SR (*Speech Recognition*) dan TTS (*Text to Speech)* terdapat beberapa *modules* yang disebut *engines*, yang telah tersedia pada semua sistem operasi Windows XP. Berikut merupakan arsitektur SAPI 5.1:



Gambar 2.14 Arsitektur SAPI 5.1

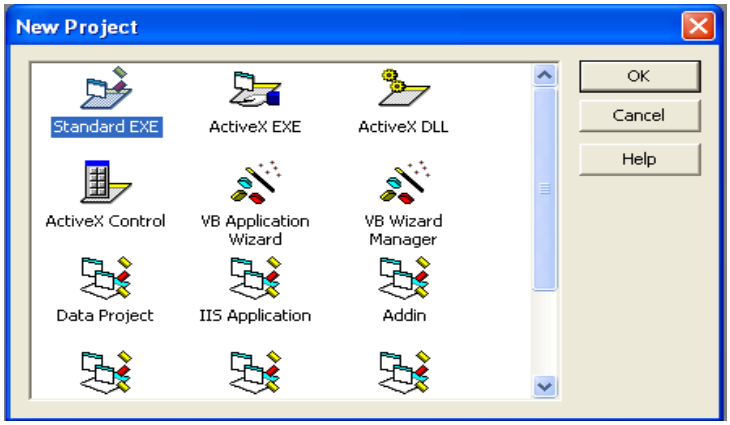
SAPI 5.1 terdiri dari dua *interface* yaitu API dan DDI (*Device Driver Interface*). SAPI 5.1 menyediakan sebuah *high-level interface* antara aplikasi dengan *speech engine*. SAPI 5.1 juga mengimplementasikan semua detail *level* bawah yang dibutuhkan untuk mengontrol dan mengatur operasi *real-time* dari berbagai *speech engines.*

2.6.2 Visual Basic 6.0

Visual basic 6.0 selain disebut sebagai bahasa pemograman, juga sering disebut sebagai sarana untuk menghasilkan program-program aplikasi berbasiskan *windows*. *Visual basic* pada dasarnya adalah bahasa pemograman komputer. Bahasa pemograman adalah perintah-perintah atau instruksi-instruksi yang dimengerti oleh komputer untuk melakukan tugas-tugas tertentu. *Visual basic* yang dikembangkan oleh *Microsoft* sejak tahun 1991 merupakan pengembangan dari pendahulunya yaitu bahasa pemograman BASIC (*Baginners All-purpose Symbolic Instruction Code*)

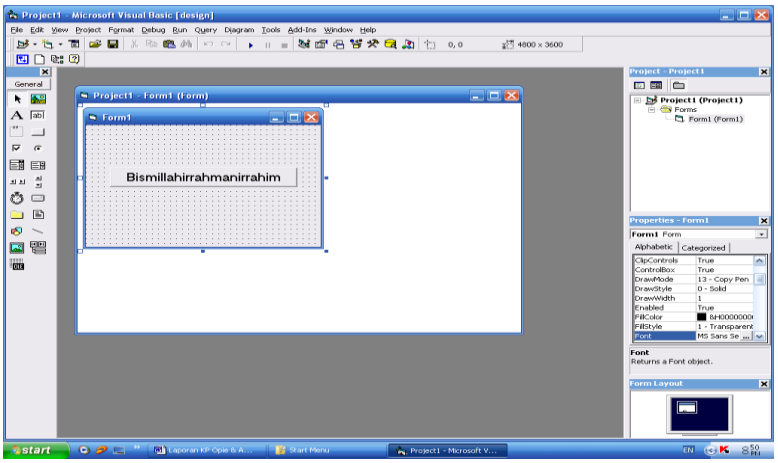
yang dikembangkan pada era 1950-an. Beberapa kemampuan atau manfaat dari *visual basic* diantaranya :

1. Untuk membuat program aplikasi seperti *windows.*
2. Untuk membuat objek-objek pembantu program seperti misalnya : kontrol *active*X, *file help*, aplikasi internet, dan sebagainya.
3. Menguji program (*debugging*) dan menghasilkan program EXE yang bersifat *executable*, atau dapat langsung dijalankan.



Gambar 2.15 Tampilan *New Project* pada *Visual Basic*

Tampilan *Itegrated Development Environment* (IDE) pada sebuah *project visual basic* dengan sebuah *form*, label dan *command button* terlihat pada gambar 2.16 ini :



Gambar 2.16 Tampilan IDE *Visual Basic*