BAB II
LANDASAN TEORI

2.1 Kalor

2.1.1 Pengertian Kalor

Kalor adalah energi yang dipindahkan akibat adanya perbedaan temperatur. Satuan kalor adalah kalori dimana, 1 kalori adalah kalor yang diperlukan untuk menaikkan temperatur 1 gr air dari 14,5 ^{o}C menjadi 15,5 ^{o}C.

Dalam sistem *British*, 1 Btu (*British Thermal Unit*) adalah kalor untuk menaikkan temperatur 1 lb air dari 63 F menjadi 64 F.

1 kal = 4,186 J

1 J = 0,2389 kal

Kapasitas kalor (C) adalah jumlah kalor yang diperlukan untuk menaikkan temperatur dari suatu sampel bahan sebesar 1 ^{o}C. [1]

$$C = \frac{\Delta Q}{\Delta T} \quad \text{..... (2.1)}$$

Kalor jenis, c adalah jumlah kalor yang diperlukan untuk menaikkan temperatur dari 1 gr massa bahan sebesar 1 ^{o}C.

$$c = \frac{\Delta Q}{m.\Delta T} \quad \text{..... (2.2)}$$

2.1.2 Asas *Black*

Azas *Blac*k menyatakan bahwa apabila dua jenis zat A dan B temperaturnya berbeda maka setelah dicampurkan, zat yang bertemperatur lebih tinggi (t_A) akan memberikan kalor (panas) pada benda yang bertemperatur lebih rendah (t_B) terus menerus sampai dicapai temperatur kesetimbangan t_C (konstan).

Dalam sebuah persamaan matematis dan dalam keadaan ideal dimana tidak ada zat lain yang terlibat dalam proses ini, maka azas black dapat dituliskan sebagai berikut:

$$Q_A = Q_B$$

$$m_A.c_A.\Delta t_A = m_B.c_B.\Delta t_B$$

$$m_A.c_A.(t_A - t_C) = m_B.c_B.(t_C - t_B)$$

Azas ini juga berlaku untuk lebih dari pencampuran dua zat, sehingga secara umum azas black dapat di tuliskan sebagai berikut:

$$Q_{lepas} = Q_{terima} \quad \text{………………………………………………………} \quad (2.3)$$

Artinya jumlah kalor yang dilepas oleh zat yang bertemperatur lebih tinggi akan seluruhnya diterima oleh zat yang bersuhu lebih rendah.

Jika zat atau benda yang menerima kalor lebih dari satu jenis, maka seberapa besar suatu zat menerima kalor di bandingkan zat lain ditentukan oleh kalor jenis benda (selain oleh massanya).

Kalor jenis benda merupakan karakter/sifat/propertis suatu benda yang unik (berbeda dari yang lain) yang menunjukan seberapa sulit zat/benda tersebut dapat menerima kalor. Sebuah benda yang memiliki kalor jenis kecil cenderung akan mudah panas di banding zat yang memiliki kalor jenis besar. Hal ini dimiliki biasanya oleh logam yang cenderung lebih mudah panas, karena memang logam pada umumnya memiliki kalor jenis kurang dari 0,5 (kalor jenis tertinggi adalah 1 dimiliki oleh air (H_2O))[2].

Berikut ini tabel kalor jenis dari berbagai bahan :

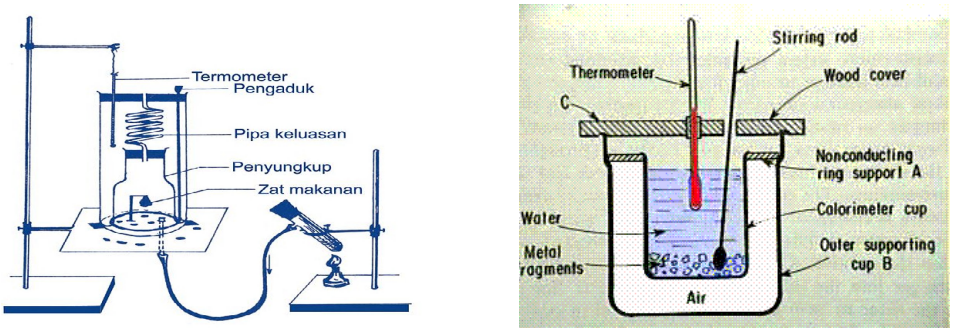
Tabel 2.1. kalor jenis berbagai bahan[1]

Substance	Specific heat c (kal/gr^{0}C)			Substance	Specific heat c (kal/gr^{0}C)	Substance	Specific heat c (kal/gr^{0}C)
Bahan				Bahan		Bahan	
Lead	0,00000	0,03050	0,03065				
Gold	0,03066	0,03080	0,03190	Emas	0,03100	Timbal	0,03100
Mercury	0,03191	0,03300	0,04400	Merkuri	0,03300		
Cadmium	0,04401	0,05500	0,05550				
Silver	0,05551	0,05600	0,06650	Perak	0,05600		
Germanium	0,06651	0,07700	0,08450				
Brass	0,08451	0,09200	0,09220	Kuningan	0,09000		
Copper	0,09221	0,09240	0,09970	Seng	0,09300	Tembaga	0,09300
Iron	0,09971	0,10700	0,13750	Besi	0,11000		
Silicon	0,13751	0,16800	0,18400	Kaca	0,16000		
Glass	0,18401	0,20000	0,20500				
Marble	0,20501	0,21000	0,21250				
Alumunium	0,21251	0,21500	0,31250				
Wood	0,31251	0,41000	0,42300				
Beryllium	0,42301	0,43600	0,45800				

Steam	0,45801	0,48000	0,49000				
Ice	0,49001	0,50000	0,54000	Minyak Tanah	0,52000		
Alcohol	0,54001	0,58000	0,79000	Alkohol	0,55000	Gliserin	0,58000
Water	0,79001	1,00000	1,00000	Air	1		

2.1.3 Kalorimeter

Kalorimeter adalah alat yang digunakan untuk mengukur jumlah kalor (nilai kalori) yang dibebaskan pada pembakaran sempurna (dalam O2 berlebih) suatu senyawa, bahan makanan, bahan bakar. Sejumlah sampel ditempatkan pada tabung beroksigen yang tercelup dalam medium penyerap kalor (kalorimeter), dan sampel akan terbakar oleh api listrik dari kawat logam terpasang dalam tabung. Contoh kalorimeter adalah kalorimeter makanan, kalorimeter larutan.[3]



Gambar 2.1. Bentuk alat kalorimeter

2.2 *Software* Visual Basic 6.0

Perancangan *software* dititik beratkan pada pembangunan sebuah program *interface* yang *user friendly* dan yang terpenting adalah *software* harus mampu berkomunikasi dengan *hardware* sehingga dapat menyampaikan informasi yang sesuai. Pada sistem ini *software* yang digunakan adalah *Visual Basic* 6.0.

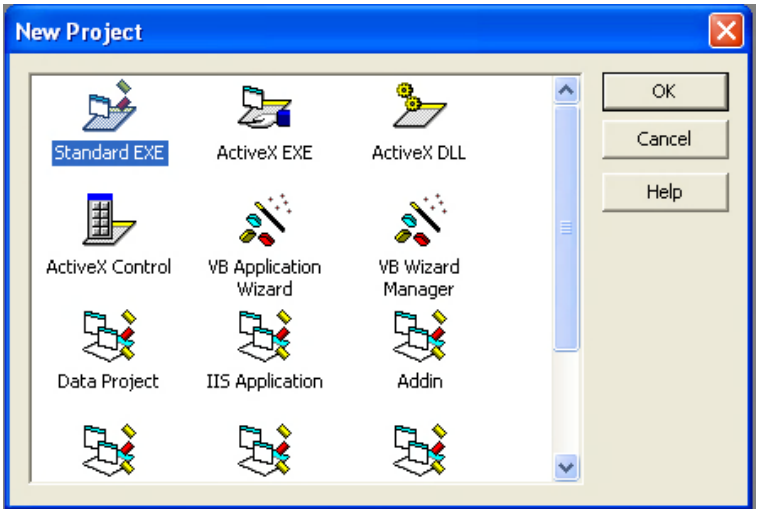
Bahasa pemrograman adalah bahasa yang dimengerti oleh *object* untuk melakukan tugas-tugas tertentu, salah satu contoh bahasa *Visual Basic*. Bahasa pemrograman *Visual Basic* yang dikembangkan oleh *Microsoft* sejak tahun 1991 merupakan pengembangan dari pendahulunya yaitu bahasa pemrograman BASIC

(*Baginners All-purpose Symbolic Instruction Code*) yang dikembangkan pada era 1950-an. *Visual Basic* merupakan salah satu *Development Tool* yaitu alat bantu untuk membuat berbagai macam program *object*, khususnya yang menggunakan sistem operasi *Windows*, juga salah satu bahasa pemrograman *object* yang mendukung *object* (*Object Oriented Programming* = OOP). Dalam pemrograman berbasis obyek (OOP) kita perlu mengenal istilah *object*, *property, method* dan *event.* Berikut adalah keterangan mengenai hal tersebut diatas :

1. *Object* adalah komponen didalam sebuah program
2. *Property* adalah karakteristik yang dimiliki oleh *object*.
3. *Method* adalah aksi yang dapat dilakukan oleh *object*.
4. *Event* adalah kejadian yang dapat dialami oelh *object*.

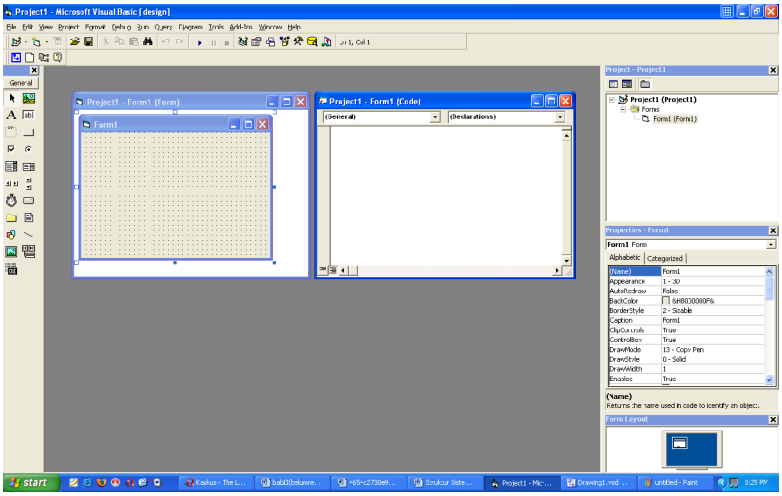
Seperti program berbasis *Windows* lainnya, *Visual Basic* terdiri dari banyak jendela (*windows*) ketika kita akan melalui *Visual Basic* sekumpulan *windows* yang saling berkaitan inilah yang disebut dengan *Integrated Development Environment* (IDE). Program yang berbasis *windows* bersifat *Event-Driven*, artinya program bekerja berdasarkan *event* yang terjadi pada *object* di dalam program tersebut, misalnya jika seorang *user* mengklik sebuah tombol maka program akan memberikan "reaksi" terhadap *event* klik tersebut. Program akan memberikan reaksi sesuia dengan kode-kode program yang dibuat untuk suatu *event* pada *object* tertentu. Pada waktu memulai *Visual Basic* beberapa *windows* kecil berada di dalam sebuah *windows* besar (*windows* induk), bentuk inilah yang dikenal dengan format *Multiple Document Interface* (MDI).

Pada gambar 2.2 memperlihatkan tampilan awal saat akan membuat sebuah **New Project** pada Visual Basic 6.0.



Gambar 2.2. Tampilan awal Visual Basic

Pada gambar 2.3 memperlihatkan contoh tampilan *Integrated Development Environment* (IDE) pada sebuah *project Visual Basic* dengan sebuah *form* dan sebuah *Command Button*.



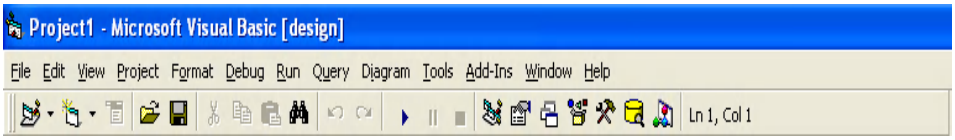
Gambar 2.3. Tampilan IDE Visual Basic

Menu pilihan pada *Visual Basic*

1. Menu Bar/Toolbar

   Menu Bar *Visual Basic* berisi semua perintah *Visual Basic* yang dapat dipilih untuk melakukan tugas tertentu, isi dari menu ini sebagaian hampir sama dengan program-program *windows* pada umumnya.

Toolbar adalah tombol-tombol (*shortcut*) yang mewakili suatu perintah tertentu pada *Toolbar*. Ini dapat dilihat pada gambar 2.4.



Gambar 2.4. Menu Bar/*Toolbar*

2. Toolbox

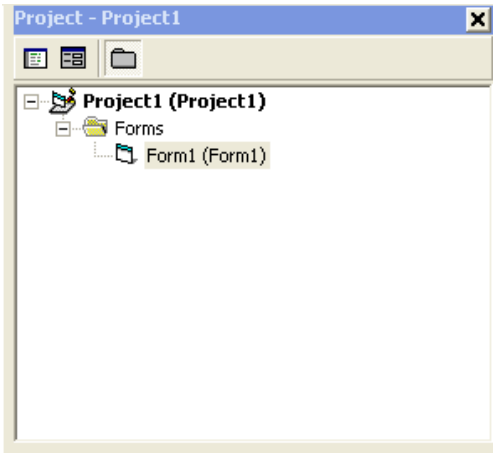
Toolbox adalah sebuah "kotak piranti" yang mengandung semua objek atau "*control*" yang dibutuhkan untuk membentuk suatu program aplikasi. Kontrol adalah suatu objek yang akan menjadi penghubung antara program aplikasi dan *user*nya yang kesemuanya harus diletakkan di dalam jendela *form*. *Toolbox* dapat disembunyikan untuk memberikan ruangan bagi element pada *Intergrated Development Environment* (IDE) lainnya. Sehingga lebih mempermudah desain maupun penulisan program. Ini dapat dilihat pada gambar 2.5.



Gambar 2.5. *Toolbox*

3. Project Window

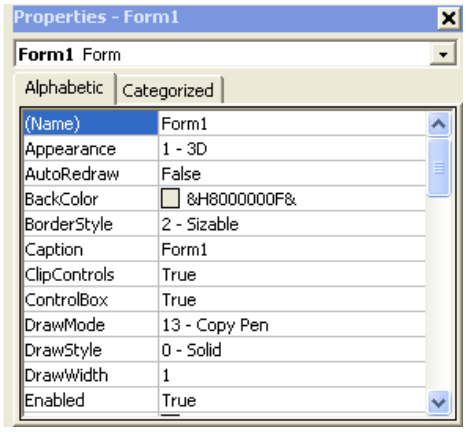
Window ini menampilkan seluruh *form*, *class*, *class module* dan komponen lainnya yang ada pada sebuah *project*. Ini dapat dilihat pada gambar 2.6.



Gambar 2.6. *Project Window*

4. Property Window

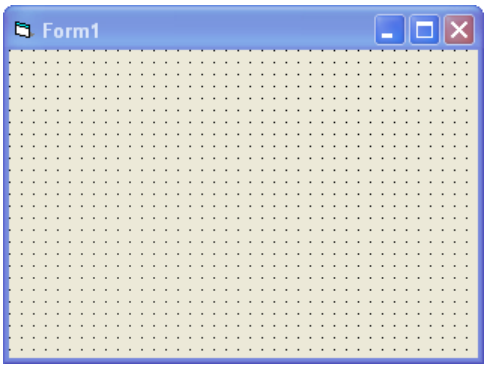
Window ini berisi seluruh *property* dari masing-masing *object* pada sebuah *project* yang meliputi *property form* dan kontrol-kontrol yang ada pada *form* tersebut. Beberapa *property* dapat diisikan pada tahap desain dan adapula *property* yang harus diisikan dengan menuliskan kode selama program dijalankan (*runtime*). Ini dapat dilihat pada gambar 2.7.



Gambar 2.7. *Property Window*

5. Form

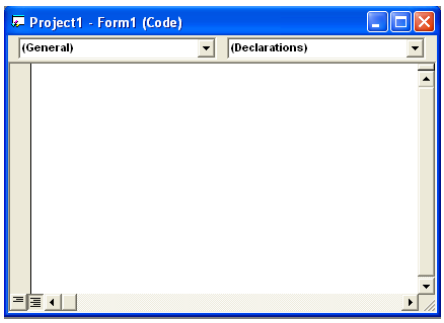
   Form adalah sebuah atau beberapa *window* untuk pembuatan program aplikasi. *Form* ini dapat memuat berbagai macam *control* (tombol-tombol maupun teks) yang diperlukan dalam desain program yang sesuai dengan kebutuhan program. Ini dapat dilihat pada gambar 2.8.



Gambar 2.8. *Form*

6. Code Window

   Pada *window* inilah semua kode/perintah tentang program dituliskan dengan memperhatikan *event* apa saja yang diperlukan. Ini dapat dilihat pada gambar 2.9.[4]



Gambar 2.9. *Code Window*

2.2.1 Pengaksesan port serial dengan Visual Basic 6.0

Pada *port* serial komputer dengan VB 6.0 dapat diakses dengan menggunakan MSComm. *Library* untuk pengaksesan *port* serial melalui kontrol MSComm yaitu Mscomm32.ocx . Kontrol MSComm pada jendela *toolbox*

didapat dari komponen kontrol Microsoft Comm Control 6.0. kontrol MSComm menyediakan fasilitas komunikasi program aplikasi dengan *port* serial untuk mengirim dan menerima data melalui *port* serial. Setiap MSComm hanya menangani satu *port* serial sehingga jika ingin menggunakan lebih dari satu *port* serial, maka harus menggunakan MSComm sebanyak *port* serial yang dipakai.

Jumlah properti pada MSComm sangat banyak sehingga tidak akan dibahas secara keseluruhan. Namun hanya membahas beberapa properti yang cukup sesuai dengan kebutuhan saja properti-properti yang sering dipakai adalah sebagai berikut :

- *CommPort* : digunakan untuk menentukan nomor *port* serial yang akan dipakai.
- *Setting* : digunakan untuk menset nilai *baud rate*, *pariti*, jumlah bit data, dan jumlah *bit stop*.
- *PortOpen* : digunakan untuk membuka ataupun menutup *port* serial yang dihubungkan dengan MSComm ini.
- *Input* : digunakan untuk mengambil data string yang ada pada *buffer* penerima.
- *Output* : Digunakan untuk menulis data *string* pada *buffer* kirim.

Berikut merupakan properti untuk mengirim satu karakter dengan MSComm pada Commport 1 :

```
Private Sub Form_Load ()
MSComm1.CommPort = 1
MSComm1.Settings = "9600,N,8,1"
MSComm1.PortOpen = True
MSComm1.Output =  “A”
MSComm1.PortOpen = False
End Sub
```

Kode-kode program pada prosedur di atas akan melakukan aksi sebagai berikut:

- *Port* serial yang digunakan adalah Comm 1
- *Setting* MSComm dengan *baud rate* 9600. tanpa bit paritas, jumlah data 8 bit dan jumlah *stop bit* adalah1
- Membuka *port* serial Comm 1
- Mengirim satu karakter (“A”)
- Menutup kembali com serial yang dipakai

Even pada MSComm hanya mempunyai satu even saja yaitu even OnComm saja. Even OnComm dibangkitkan jika nilai properti dari CommEvent berubah yang mengindikasikan telah terjadi even pada *port* serial baik even komunikasi maupun even *error* berikut ini merupakan properti CommEvent, nilai-nilai properti ini tidak tersedia pada saat *design time*, tetapi hanya dapat di baca pada saat *run time.*

- comEventFrame : *Hardware* mendeteksi adanya kesalahan framing.
- comEventRxParity : *Hardware* mendeteksi adanya kesalahan parity.
- comEventRxOver : *Buffer* penerima mengalami over flow, tidak ada ruang kosong lagi pada *buffer* penerima.
- comEventTxFull : *Buffer* pengiriman telah penuh.
- comEventOverrun : *Port* mengalami *overrun*
- comEventBreak : Sinyal *Break* dikirim
- comEventDCB : Mendapatkan kembali *Device Control Clock* (DCB) dari *port* serial.

Berikut adalah nilai–nilai properti dari even komunikasi pada CommEvent yaitu :

- commEvSend : Jumlah karakter pada *Buffer* kirim lebih sedikit dari pada nilai properti Sthreshold. Even ini akan dibangkitkan jika nilai properti sthreshold tidak diisi dengan “0”.
- comEvReceive : Telah diterima karakter sebanyak nilai properti Rthreshold. Even ini akan dibangkitkan terus menerus sampai data diambil dari *Buffer* penerima menggunakan perintah *Input*. Even ini akan dibangkitkan jika nilai pada properti Rthreshold tidak diisi “0”.

- commEvCTS : Terjadi perubahan pada saluran *Clear to Send.*
- commEvDSR : Terjadi perubahan pada saluran *Data Set Ready*.
- commEvRing : Terjadi perubahan pada saluran *Carrier Detect.*
- comEvRing : Terdeteksi adanya sinyal *Ring.*
- comEvEOF : Karakter *End of File* diterima.

Berikut adalah contoh program pada penggunaan Even OnComm untuk berkomunikasi. Namun yang hanya dibaca hanya even comEvReceive saja, yang lainya diabaikan

```
Private Static Sub MsComm1_OnCom ()
Dim Buffer as variant
Select Case MSComm1.CommEvent
        Case comEvReceive
                If  MSComm1,InbufferCount = 1 then
                Buffer = CStr(MSComm1.Input)
                Text1.Text  = asc(Midbuffer, 1,1))
                End If
        End If
End Select
End Sub
```

Kode-kode program pada prosedur di atas akan melakukan aksi dengan mendeteksi even comEvReceive, kemudian menentukan apakah sudah diterima 1 karakter pada *buffer* penerima.[4]

2.2.2 Pengaksesan secara langsung melalui register UART

Saluran yang digunakan UART untuk berkomunikasi serial yaitu TXD dan RXD serta saluran-saluran untuk kontrol, yaitu DCD, DSR, RTS, CTS, DTR, dan RI. Saluran ini ada yang berfungsi sebagai *output* dan data yang sebagai *input.* Terkecuali saluran RXD, saluran-saluran ini dapat diakses melalui register UART. Berikut adalah tabel dan lokasi bit saluran tersebut pada UART.

Tabel 2.2. Alamat dan Lokal bit pada register UART

Nama Pin	Nomor pin DB9	Com1	Com2	Bit	Arah
RXD	2	3FBH	2FBH	2	Input
TXD	3	3FBH	2FBH	6	Output
DTR	4	3FCH	2FCH	0	Output
RTS	7	3FCH	2FCH	1	Output
CTS	8	3FEH	2FEH	4	Input
DSR	6	3FEH	2FEH	5	Input
RI	9	3FEH	2FEH	6	Input
DCD	1	3FEH	2FEH	7	Input

Untuk dapat mengaksesnya dapat menggunakan fungsi *Port_Out* dan *Port_In*, namun pada VB 6.0 tidak disediakan secara langsung, harus ada program tambahan tersendiri untuk pengaksesan UART ini.[4]

2.2.3 Pengaksesan port serial dengan kontrol MSComm

Kontrol MSComm menyediakan fasilitas komunikasi program aplikasi dengan *port* serial untuk mengirim dan menerima data melalui *port* serial. Setiap MSComm hanya menangani satu *port* serial sehingga jika ingin menggunakan lebih dari satu *port* serial, maka harus menggunakan MSComm sebanyak *port* serial yang dipakai.[4]

Jumlah properti pada MSComm sangat banyak sehingga tidak akan dibahas secara keseluruhan. Namun hanya membahas beberapa properti yang cukup sesuai dengan kebutuhan saja properti-properti yang sering dipakai adalah sebagai berikut :

- *CommPort* : digunakan untuk menentukan nomor *port* serial yang akan dipakai.
- *Setting* : digunakan untuk menset nilai *baudrate*, *pariti*, jumlah bit data, dan jumlah *bit stop*.
- *PortOpen* : digunakan untuk membuka ataupun menutup *port* serial yang dihubungkan dengan MSComm ini.

- *Input* : digunakan untuk mengambil data string yang ada pada *buffer* penerima.
- *Output* : Digunakan untuk menulis data *string* pada *buffer* kirim.

Berikut merupakan properti untuk mengirim satu karakter dengan MSComm pada port Comm 1 :

```
Private Sub Form_Load ()
MSComm1.CommPort = 1
MSComm1.Settings = "9600,N,8,1"
MSComm1.PortOpen = True
MSComm1.Output =  “A”
MSComm1.PortOpen = False
End Sub
```

Kode-kode program pada prosedur di atas akan melakukan aksi sebagai berikut:

- *Port* serial yang digunakan adalah Comm 1
- *Setting* MSComm dengan *baud rate* 9600. tanpa bit paritas, jumlah data 8 bit dan jumlah *stop bit* adalah1
- Membuka *port* serial Comm 1
- Mengirim satu karakter (“A”)
- Menutup kembali com serial yang dipakai

2.3 Mikrokontroler AT89S51

Mikrokontroler AT89S51 merupakan salah satu IC mikrokontroler yang termasuk dalam *MCS-51*^{TM} dari Intel. Sebuah mikrokontroler tidak dapat bekerja apabila tidak diisi oleh suatu program. Salah satu keunggulan dari AT89S51 adalah dapat diisi ulang dengan program yang berbeda sebanyak 1000 kali pengisian. Instruktur-instruktur perangkat lunak berbeda untuk masing-masing jenis mikrokontroler. Instruksi-instruksi hanya dapat dipahami oleh jenis mikrokontroler yang bersangkutan. Instruksi-instruksi tersebut dikenal sebagai bahasa pemrograman sistem mikrokontroler.

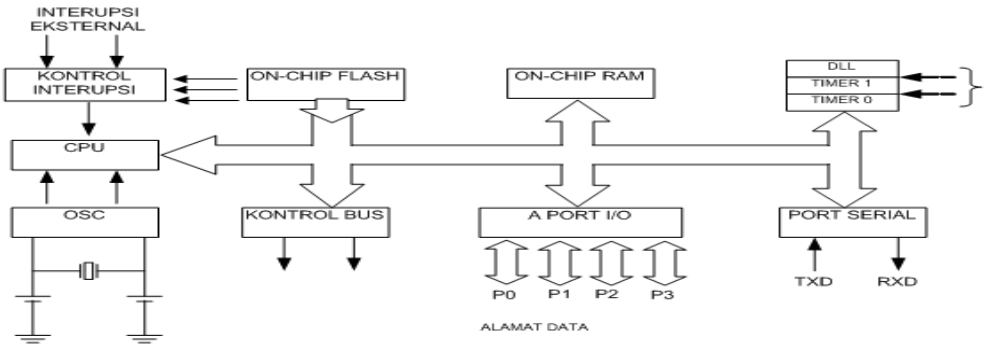
Sebuah mikrokontroler tidak dapat memahami instruksi-instruksi yang berlaku pada mikrokontroler lain. Sebagai contoh, mikrokontroler buatan Intel dengan mikrokontroler buatan Motorolla memiliki perangkat instruksi yang berbeda.

2.3.1 Karakteristik Mikrokontroler AT89S51

Mikrokontroler AT89S51 memiliki beberapa fasilitas, diantaranya sebagai berikut :

1. Sebuah CPU (*Central Prossesing Unit*) 8 bit yang termasuk keluarga Osilator *internal* dan rangkaian *timer*.
2. Flash memori 4 Kbyte.
3. RAM internal 128 byte.
4. Empat buah *programmable* port I/O, masing-masing terdiri atas 8 buah jalur I/O.
5. Lima buah jalur interupsi (2 buah interupsi eksternal dan 3 buah internal).
6. Sebuah port serial dengan kontrol *serial full duplex* UART.
7. Kemampuan melaksanakan operasi perkalian, pembagian dan operasi boolean (bit).
8. Kecepatan pelaksanaan interuksi per siklus 1 mikrodetik pada frekuensi clock 1 MHz.

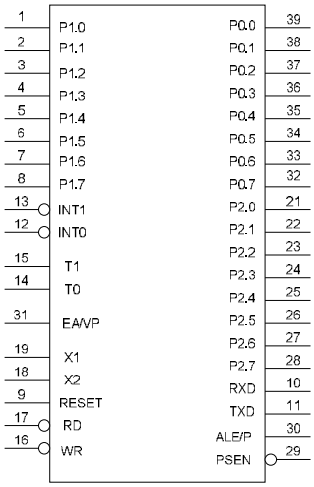
Dengan fasilitas seperti diatas, pembuatan alat menggunakan AT89S51 menjadi lebih sederhana dan tidak memerlukan IC pendukung yang banyak. Agar lebih jelasnya dapat dilihat pada gambar 2.10 dibawah ini.



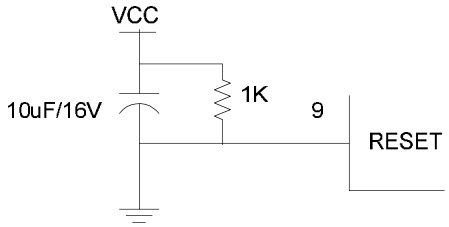
Gambar 2.10. Diagram Blok Mikrokontroler AT89S51

2.3.2 Deskripsi Pin-Pin AT89S51

Susunan pin-pin mikrokontroler AT89S51 memperlihatkan pada gambar 2.11 dibawah ini. Penjelasan dari masing-masing pin adalah sebagai berikut :



Gambar 2.11. Diagram Pin Mikrokontroler AT89S51



Gambar 2.12. Rangkaian Reset

1. Pin 1 sampai 8 (Port 1) merupakan port pararel 8 bit dua arah (*output-input*) yang dapat digunakan untuk berbagai keperluan (*general purpose*).
2. Pin 9 (Reset) adalah input reset (aktif tinggi). Pulsa transisi dari rendah ke tinggi akan me-reset AT89S51. Pin ini dihubungkan dengan rangkaian *power on reset* seperti ditunjukan pada gambar 2.12 diatas.
3. Pin 10 sampai 17 (Port 3) adalah port pararel 8 bit dua arah (output-input) yang memiliki fungsi pengganti. Fungsi pengganti meliputi TxD (*Transmision Data*), RxD (*Received Data*), Int 0 (*Interupsi 0*), Int 1 (*Interupsi 1*), T0 (*Timer 0*), T1 (*Timer 1*), WR (*Write*) dan RD (*Read*). Bila fungsi

pengganti tidak dipakai, pin-pin ini dapat digunakan sebagai port pararel 8 bit serba guna.

4. Pin 18 dan 19 (XTAL1 dan XTAL2) adalah pin input kristal, yang merupakan input *clock* bagi rangkaian osilator internal.
5. Pin 20 (*Ground*) dihubungkan ke Vss atau Ground.
6. Pin 21 sampai 28 (Port 2) adalah port pararel 2 selebar 8 bit dua arah. Port 2 ini mengirimkan byte alamat bila dilakukan pengaksesan memori eksternal.
7. Pin 29 adalah pin PSEN (*Program Strobe Enable*) yang merupakan sinyal pengontrol yang membolehkan program memori eksternal masuk ke dalam bus selama proses pemberian/pengambilan instruksi.
8. Pin 30 adalah pin output ALE (*Address Latch Enable*) yang digunakan untuk menahan alamat memori eksternal selama pelaksanaan instruksi.
9. Pin 31 (EA). Bila pin ini diberi logika tinggi, maka mikrokontroler akan melaksanakan instruksi dari ROM ketika isi program counter kurang dari 4096. Bila diberi logika rendah, maka mikrokontroler akan melaksanakan seluruh instruksi dari memori program diluar.
10. Pin 32 sampai 39 (*Port 0*) merupakan port pararel 8 bit *open drain* dua arah. Bila diberi logika rendah, mikrokontroler akan melaksanakan seluruh instruksi dari memori program luar.
11. Pin 40 (Vcc) dihubungkan ke +5 Volt.

2.3.3 Organisasi Memori

Semua mikrokontroler dalam keluarga MCS-51 memiliki pembagian ruang alamat (*address space*) untuk program dan data. Pemisahan memori program dan memori data membolehkan memori data untuk diakses oleh alamat 8 bit. Meskipun demikain, alamat data memori 16 bit dapat dihasilkan melalui register DPTR (Data Pointer Register).

Memori program hanya dapat dibaca tidak bisa ditulis, karena disimpan dalam *Flash Memori*. Memori program sebesar 64 Kbyte dapat dimasukkan dalam EPROM eksternal, seperti tampak pada gambar 2.13.

Sinyal yang membolehkan pembacaan dari memori program eksternal adalah pin PSEN. Pada gambar 2.14 memperlihatkan memori data yang terletak pada ruang alamat terpisah dari memori program. RAM ekternal 64 Kbyte dapat dialamati dalam ruang memori data eksternal. CPU menghasilkan sinyal *read* dan *write* selama menghubungi memor data eksternal.

Mikrokontroler AT89S51 memiliki 5 buah ruang alamat, yaitu :

(a) Ruang alamat kode (*Code Address Space*) sebanyak 64 Kbyte, yang seluruhnya merupakan ruang alamat kode eksternal.

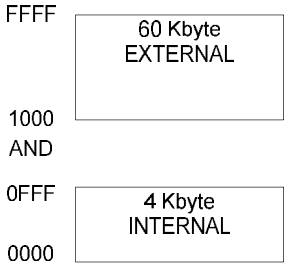
(b) Ruang alamat memori data internal yang dapat dialamati secara langsung, yang terdiri atas :

1. RAM sebanyak 128 byte.
2. Hardware register sebanyak 128 byte.

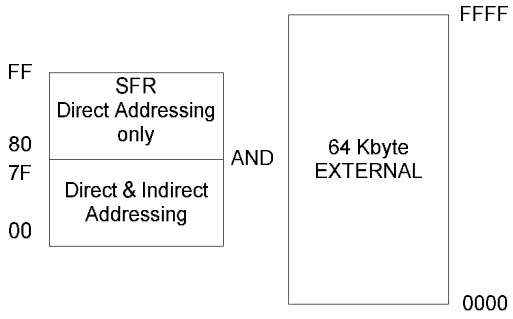
(c) Ruang alamat memori data internal yang dialamati secara tidak langsung sebanyak 128 byte, seluruhnya diakses dengan pengalamatan tidak langsung.

(d) Ruang alamat memori data eksternal sebanyak 64 Kbyte yang dapat ditambahkan oleh pemakai.

(e) Ruang alamat bit, yang dapat diakses dengan pengalamatan langsung.



Gambar 2.13. Struktur Program Memori AT89S51



Gambar 2.14. Struktur Data Memori AT89S51

2.3.4 Special Function Register (SFR)

SFR berisi register-register dengan fungsi tertentu yang disediakan oleh mikrokontroler seperti timer dan lain-lainnya. AT89S51 memiliki 21 SFR yang terletak pada memori 80H - FFH. Masing-masing ditunjukkan pada tabel 2.3 yang meliputi simbol, nama dan alamatnya.

Tabel 2.3. *Spesial Function Register*

Simbol	Nama	Alamat
ACC	Akumulator	E0H
B	*B register*	F0H
PSW	*Program Status Word*	D0H
SP	*Stack Pointer*	81H
DPTR	*Data Pointer* 16 bit *DPL byte* rendah *DPH byte* tinggi	 82H 83H
P0	*Port 0*	80H
P1	*Port 1*	90H
P2	*Port 2*	A0H
P3	*Port 3*	B0H
IP	Interupt Priority Control	B8H
IE	Interupt Enable Control	A8H
TMOD	Timer/Counter Mode Control	89H
TCON	Timer/Counter Control	88H
TH 0	*Timer/Counter High Low byte*	8CH
TL 0	*Timer/Counter Low byte*	8AH
TH 1	*Timer/Counter High byte*	8DH
TH 1	*Timer/Counter Low byte*	8BH
SCON	*Serial Control*	98H
SBUF	*Serial Data Buffer*	99H
PCON	*Power Control*	87H

2.3.5 Timer AT89S51

Mikrokontroler AT89S51 mempunyai dua buah timer, yaitu Timer 0 dan Timer 1, setiap timer terdiri dari 16 bit timer yang tersimpan dalam dua buah register yaitu THx untuk *Timer High Byte* dan TLx untuk *Timer Low Byte* yang

keduanya dapat berfungsi sebagai *counter* maupun sebagai timer. Secara fisik timer juga merupakan rangkaian flip-flop yang dapat diaktifkan dan dinonaktifkan setiap saat. Perbedaan keduanya terletak pada sumber *clock* dan aplikasinya.

Timer mempunyai sumber *clock* dengan frekuensi tertentu yang sudah pasti sedangkan *counter* mendapat sumber *clock* dari pulsa yang hendak dihitung jumlahnya. Aplikasi dari timer atau pewaktu biasa digunakan untuk aplikasi menghitung lamanya suatu kejadian yang terjadi sedangkan *counter* atau penghitung biasa digunakan untuk aplikasi menghitung jumlah kejadian yang terjadi dalam periode tertentu. Perilaku dari register THx dan TLx diatur oleh register TMOD dan TCON. Timer dapat diaktifkan melalui perangkat keras maupun perangkat lunak.

Periode waktu timer/counter dapat dihitung menggunakan rumus 2.4 dan 2.5 sebagai berikut.

Sebagai timer/counter 8 bit

$$T = (255 - TLx) * \frac{12}{frekuensiXTAL}\,\mu s \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (2.4)$$

Sebagai timer/counter 16 bit

$$T = (65535 - THxTLx) * \frac{12}{frekuensiXTAL}\,\mu s \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (2.5)$$

Di mana : THx = isi register TH0 atau TH1 dan TLx = isi register TL0 atau TL1.

Dapat diakses secara bit

	Register Timer				Register Interupsi			
88H	TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0
	TCON.7	TCON.6	TCON.5	TCON.4	TCON.3	TCON.2	TCON.1	TCON.0

Tidak dapat diakses secara bit

	Timer 1				Timer 0			
89H	GATE	C/T	M1	M0	GATE	C/T	M1	M0

Gambar 2.15. Register TCON dan TMOD

Pengontrolan kerja *timer/counter* diatur oleh register TCON. Register ini bersifat *bit addresable* sehingga bit TF1 dapat disebut TCON.7 dan seterusnya hingga bit IT0 sebagai TCON.0. Register ini hanya mempunyai 4 bit saja yang berhubungan dengan timer seperti diperlihatkan gambar 2.15 dan dijelaskan pada tabel 2.4.

Tabel 2.4. Fungsi bit register TCON yang berhubungan dengan *timer*

Nama Bit	Fungsi
TF1	*Timer* 1 *overflow flag* yang akan diset jika *timer overflow*
TR1	Membuat timer 1 aktif (*set*) dan nonaktif (*clear*)
TF0	*Timer* 0 *overflow flag* yang akan diset jika *timer overflow*
TR0	Membuat timer 0 aktif (*set)* dan nonaktif (*clear*)

Register TMOD berfungsi untuk pemilihan mode operasi *timer/counter* dengan fungsi setiap bitnya adalah sebagai berikut :

Gate : Pada saat TRx = 1, *timer* akan berjalan tanpa memperlihatkan nilai pada Gate (timer dikontrol software).

C/T : Pemilihan fungsi timer (0) atau counter (1).

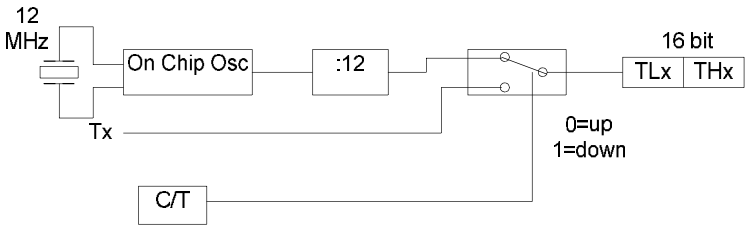
M1 & M0 : Untuk memilih mode timer dengan variasi seperti pada tabel 2.5.

Tabel 2.5. *Mode Timer*

M1	M0	Mode	Operasi
0	0	0	Timer 13 bit
0	1	1	Timer/Counter 16 bit
1	0	2	Timer 8 bit di mana nilai timer tersimpan pada TLx. Register THx berisi nilai isi ulang yang akan dikirim ke TLx setiap *overflow*.
1	1	3	Pada mode ini, AT89S51 bagaikan memiliki 3 buah timer. Timer 0 terpisah menjadi 2 buah timer 8 bit (TL0-TF0 dan TH0-TF1) dan timer tetap 16 bit.

Prinsip Kerja Timer

Pada gambar 2.16 *Timer* mempunyai dua buah sumber *clock* untuk beroperasi, yaitu sumber *clock* internal dan sumber *clock* eksternal. Jika *timer* menggunakan sumber *clock* eksternal, maka bit C/T harus di-set atau berkondisi *high*, saklar akan menghubungkan sumber *clock timer* ke pin Tx (T0 untuk timer 0, T1 untuk timer 1). Apabila sumber *clock* internal digunakan, *input clock* berasal dari osilator yang telah dibagi 12. Maka bit C/T harus di-clear atau berkondisi *low* sehingga saklar akan menghubungkan sumber *clock timer* ke osilator yang telah dibagi 12.[8]



Gambar 2.16. Operasi *Timer*

2.4 Port serial/RS-232

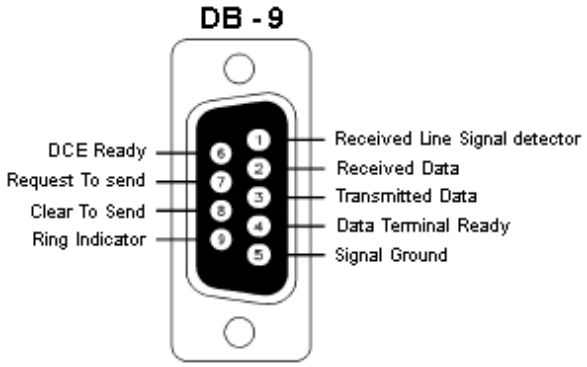
Port serial lebih sulit ditangani daripada *port* paralel karena peralatan yang dihubungkan ke *port* serial harus berkomunikasi dengan menggunakan transmisi serial, sedangkan data di komputer diolah secara paralel. Sehingga, data dari/ke *port* serial harus dikonversikan ke/dari bentuk paralel untuk bisa digunakan secara *hardware* hal ini bisa digunakan oleh UART *(Universal Asynchronus Receiver Transmitter).*

Adapun keunggulan menggunakan port serial dari pada port paralel sebagai transfer data yaitu :

1. Kabel port serial bisa lebih panjang dibandingkan kabel port paralel. Hal ini karena port serial mengirimkan logika 1 sebagai –3 Volt hingga –25 Volt dan logika 0 sebagai +3 Volt hingga +25 Volt, sedangkan port paralel menggunakan TTL, yakni hanya 0 Volt untuk logika 0 dan +5 Volt untuk logika 1. ini berarti port serial memiliki rentang kerja 50 Volt sehingga

kehilangan daya karena panjang kabel bukan merupakan masalah serius jika dibandingkan dengan port paralel.

2. Transmisi serial memerlukan lebih sedikit kabel dibandingkan dengan transmisi paralel.
3. *Port* serial memungkinkan untuk berkomunikasi dengan menggunakan *Infra Red.*



Gambar 2.17. Konfigurasi *port* serial *male*

EIA *(Electronic Industry Association)* mengeluarkan spesifikasi listrik untuk standar RS-232 yaitu :

1. *Space* (logika 0) antara +3 sampai +15 Volt.
2. *Mark* (logika 1) antara –3 sampai –15 Volt.
3. Daerah antara +3 Volt dan –3 Volt tidak ditetapkan.
4. Tegangan rangkaian terbuka tidak boleh melebihi 25 Volt (terhadap *Ground*).
5. Arus pada rangkaian tertutup *(Short Circuit)* atau hubung singkat tidak boleh melebihi 500 mA.

Tabel 2.6. Konfigurasi pin dan nama sinyal konektor serial DB9

Nama Pin	Nama Sinyal	Direction	Keterangan
1	DCD	In	Data Carrier Detect/Received Line Signal Detect
2	RxD	In	Received Data
3	TxD	Out	Transmite Data
4	DTR	Out	Data Terminal Ready
5	GND	-	Ground
6	DSR	In	Data Set Ready
7	RTS	Out	Request to Send
8	CTS	In	Clear to Send
9	RI	In	Ring Indicator

Keterangan mengenai saluran RS-232 pada konektor DB9 adalah sebagai berikut :

1. *Received Line Signal Detect*, dengan saluran ini DCE memberitahukan ke DTE bahwa terminal masukan ada data masukan.
2. *Received Data*, digunakan DTE menerima data dari DCE.
3. *Transmite Data*, digunakan DTE mengirimkan data ke DCE.
4. *Data Terminal Ready*, pada saluran ini DTE memberitahukan kesiapan sinyalnya.
5. *Signal Ground*, saluran Ground.
6. *Ring Indicator,* pada saluran ini DCE memberitahukan ke DTE bahwa stasiun menghendaki hubungan dengannya.
7. *Clear to Send,* dengan saluran ini DCE memberitahukan ke DTE boleh mengirimkan data.
8. *Request to Send*, dengan saluran ini DCE diminta mengirimkan data oleh DTE.
9. *DCE Ready,* sinyal aktif pada saluran ini menunjukan bahwa DCE sudah siap.[4]

2.5 Pengaturan Baudrate

Untuk transmisi data antara pengiriman dan penerimaan harus menggunakan lau kecepatan (*Baudrate*) yang sama agar terjadi kesinkronan antara data yang dikirim dan data yang diterima. *Timer* 1 dapat digunakan sebagai pewaktu untuk mengatur *baudrate* pada komunikasi serial. Lama pengiriman tiap bit data = *timer* 1 *overflow* x 32.[9]

Rumus frekuensi osilator dengan *baudrate* :

$$\frac{12x(FFh - Th1)}{F_{OSC}} = \frac{1}{Baudratex32} \quad \text{..... (2.6)}$$

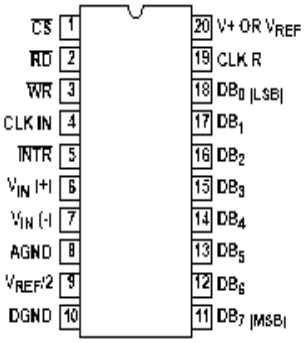
Rumus nilai register TH1:

$$TH1 = 256 = \frac{F_{OSC}}{12xBaudratex32} \quad \text{..... (2.7)}$$

2.6 Analog to Digital Converter (ADC)

Salah satu komponen penting dalam sistem akuisisi data adalah pengubah besaran *analog* ke *digital* atau disebut juga ADC (*Analog to Digital Converter*). Pengubah ini akan mengubah besaran-besaran analog menjadi bilangan-bilangan *digital* sehingga dapat diproses oleh mikrokontroler maupun oleh PC. Peranan pengubah ini menjadi semakin penting sebagai dampak dari perkembangan sistem perangkat keras digital.

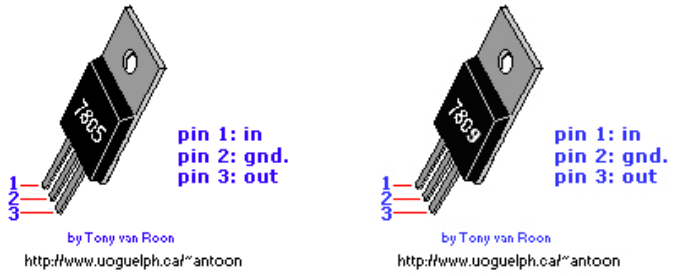
Contoh aplikasi ADC ini bisa dilihat misalnya pada *voltmeter digital, sampling* suara dengan komputer sehingga suara dapat disimpan secara *digital* dalam disket dan lain sebagainya. Pada ADC terdapat beberapa metode konversi *analog* ke *digital*. Beberapa metode tersebut diantaranya adalah *ladder network*, s*uccessive approximation*, dan *comparator*. Namun pada umumnya, yaitu untuk seluruh seri ADC080X, menggunakan metode konversi *successive aproximation*. Pada gambar 2.18 dapat dilihat salah satu contoh ADC yaitu ADC0804, yang merupakan ADC dengan satu *input*.[13]



Gambar 2.18. *Konfigurasi pin ADC0804*

2.7 IC LM7805 dan LM7809

Kedua IC ini mempunyai tiga kaki yang digunakan sebagai komponen pendukung dari Vcc untuk menghasilkan tegangan 5V dan 9V. IC *regulator* ini berfungsi untuk menstabilkan tegangan dan dapat bekerja dengan baik jika tegangan *input* (Vin) lebih besar dari pada tegangan *output* (Vout). Biasanya perbedaan tegangan *input* dengan *output* yang direkomendasikan tertera pada *datasheet* komponen tersebut. Contoh LM7805 dan LM7809 diperlihatkan pada gambar 2.19.[14]



(a) (b)

Gambar 2.19. *Konfigurasi pin IC LM7805 dan IC LM7809*

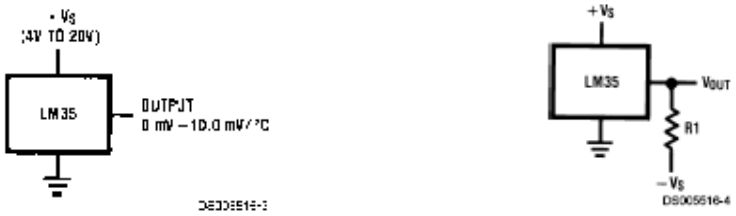
2.8 Sensor Suhu (LM35)

LM35 adalah jenis sensor suhu dengan rangkaian ketepatan yang terintegrasi dan tegangan outputnya secara linier proporsional terhadap ^{0}C. Keuntungannya

melebihi sensor suhu yang dikalibrasi oleh satuan *Kelvin* serta *output impedansi* dengan catu daya Vin sebesar 5 V.[10]

Keuntungan menggunakan sensor suhu LM35 :

- Dikalibrasi secara langsung pada satuan derajat celcius.
- Skala faktor linier ± 10,0 mV/ ^{0}C.
- Bekerja antara suhu 0 ^{0}C sampai dengan +100 ^{0}C.
- Arus kurang dari 2 mA.
- Pemanasan rendah 0,08 ^{0}C.



(a) *suhu +2 ^{0}C sampai +150 ^{0}C* **(b)** *full range (-55 ^{0}C sampai +150 ^{0}C)*

Gambar 2.20. *Konfigurasi pin LM35*

2.9 *Inverting Amplifier*

Inverting amplifier (pembalik tegangan) adalah sebuah rangkaian yang merupakan salah satu aplikasi dari *operational amplifier* (*op-amp*). Dengan menggunakan rangkaian ini tegangan masukan akan diperkuat x kali lipat. Pada rangkaian ini tegangan masukan dihubungkan ke kaki masukan inverting pada IC *op-amp*, sehingga tegangan keluaran yang dihasilkan memiliki polaritas yang berbeda dengan tegangan masukan. Perbedaan polaritas tersebut menjadikan rangkaian ini selalu menghasilkan tegangan yang terbalik dengan tegangan awal dinamakan inverting amplifier.

Rangkaian ini menggunakan dua buah resistor sebagai penentu nilai penguatan yang diinginkan. Rumus penguatannya adalah sebagai berikut :

$$Vo = \left(\frac{Rf}{Ri}\right) Vi \quad \text{.....(2.8)}$$

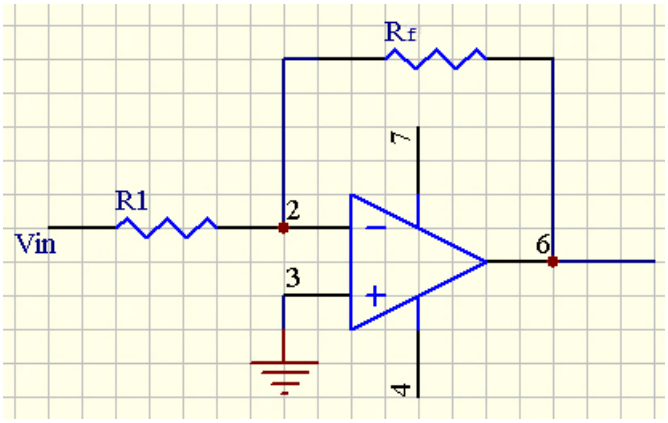
Dimana : Vo = Tegangan output

Vi = Tegangan Input

Rf = Hambatan Referensi

Ri = Hambatan Input

Tegangan keluaran dari rangkaian *inverting amplifier* ini akan selalu bernilai terbalik dengan keadaan awal.[11]

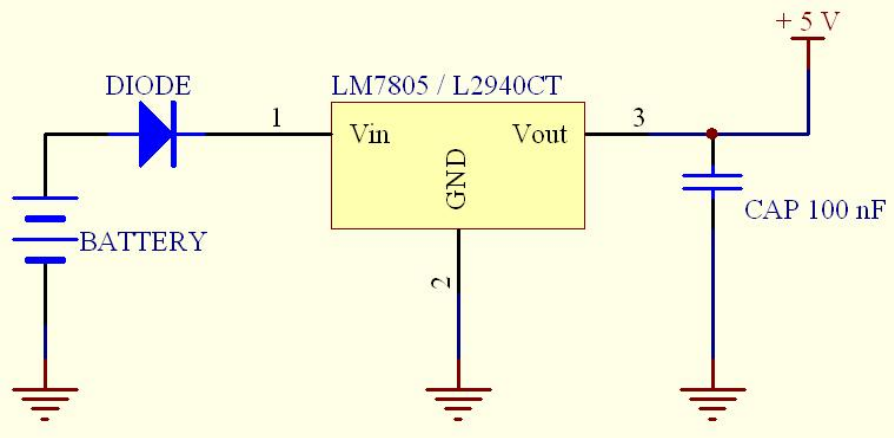


Gambar 2.21. Rangkaian Inverting Amplifier

2.10 Catu Daya

Perangkat elektronik seharusnya menggunakan catu daya arus searah DC yang stabil agar dapat bekerja dengan baik. Baterai atau accu adalah sumber catu daya DC yang paling baik. Namun untuk aplikasi yang membutuhkan catu daya yang lebih besar, sumber dari baterai dan accu tidak mencukupi. Oleh karena itu, diperlukan sumber daya yang lebih besar yaitu sumber bolak-balik AC dari pembangkit listrik. Untuk itu diperlukan suatu catu daya yang digunakan untuk mengubah arus AC menjadi DC. *Transformator* diperlukan untuk menurunkan tegangan AC dari jala-jala listrik pada kumparan primer menjadi tegangan AC yang lebih kecil pada kumparan sekunder. Untuk mendapatkan arus yang searah diperlukan dioda, dioda berperan hanya untuk meneruskan tegangan positif ke

regulator. Regulator berfungsi sebagai komponen aktif yang dapat meregulasi tegangan keluar agar stabil.[11]



Gambar 2.22. Catu daya