BAB II

TINJAUAN PUSTAKA

2.1 Osiloskop

Osiloskop adalah alat ukur elektronika yang berfungsi memproyeksikan bentuk sinyal listrik agar dapat dilihat dan dipelajari. Pada awalnya osiloskop terdiri dari tabung sinar katode dan peranti pemancar elektron yang berfungsi untuk memproyeksikan sorotan elektron ke layar tabung sinar katode, sehingga sorotan elektron membekas pada layar dan dengan bantuan beberapa rangkaian khusus didalam osiloskop tersebut maka akan menyebabkan sorotan bergerak berulang-ulang dari kiri ke kanan layar. Pengulangan sorotan tersebutlah yang menyebabkan bentuk sinyal kontinyu pada layar sehingga sinyal tersebut dapat dilihat dan dipelajari.

Dalam bidang elektronika, perangkat osiloskop merupakan instrumen alat ukur yang memiliki posisi yang sangat vital mengingat sifatnya yang mampu menampilkan bentuk gelombang yang dihasilkan oleh rangkaian yang sedang diamati. Dewasa ini secara prinsip terdapat 2 (dua) tipe osiloskop, yakni osiloskop analog dan osiloskop digital. Masing-masing tipe osiloskop tersebut memiliki kelebihan dan keterbatasannya.

2.1.1 Osiloskop Analog

Osiloskop analog bekerja dengan cara menggambarkan bentuk-bentuk gelombang listrik melalui gerakan pancaran elektron (*electron beam*) dalam sebuah tabung sinar katode yang berjalan dari kiri ke kanan. Pancaran elektron

yang dipancarkan oleh bagian *electron gun* akan membentur dinding layar tabung sinar katode sehingga elektron pada lapisan fosfor layar akan ter-eksitasi dan mengakibatkan perpendaran atau nyala pada layar, hal tersebutlah yang akan menjadi gambar bentuk dasar gelombang yang di ukur oleh osiloskop.

Pada osiloskop analog, pembetuk gelombang yang akan ditampilkan pada layar diatur oleh sepasang lapisan pembelok (*deflector plate*) secara vertikal maupun secara horizontal, pembelokan pancaran elektron dilakukan oleh lapisan tersebut dimana ketika lapisan pembelok tersebut diberi sebuah tegangan tertentu maka akan mengakibatkan pancaran elektron berbelok dengan harga tertentu pula. Sebagai contoh apabila tegangan pada semua pelat tersebut adalah 0 (nol) Volt, maka pancaran elektron akan bergerak lurus membentur layar sehingga pada layar hanya akan terlihat sebuah nyala titik ditengah layar. Pengaturan tegangan pada lapisan pembelok tersebut akan berkaitan terhadap pengaturan *Time/Div* untuk lapisan pembelok horizontal, dan berkaitan terhadap pengaturan *Volt/Div* untuk lapisan pembelok vertikal.

Cahaya yang dihasilkan oleh fosfor mempunyai waktu hidup yang sangat pendek setelah pancaran elektron berlalu. Untuk fosfor yang sering digunakan pada tabung sinar katode (CRT) adalah jenis P31, dimana fosfor jenis tersebut untuk cahaya yang dihasilkan akan turun hingga ke suatu harga yang masih dapat dilihat dengan nyaman dalam ruang yang bercahaya sedang dalam waktu 38 mikrodetik. Jika laju pancaran elektron untuk meng-eksitasi ulang terjadi dibawah 1/38 mikrodetik atau sekitar 26 kHz, maka akan terjadi penurunan cahaya secara dramatis pada layar.

Kedipan (*flicker*) merupakan suatu fenomena lain yang membatasi kinerja CRT. Jika laju eksitasi ulang jatuh dibawah harga minimum tertentu, umumnya sekitar 15 sampai 20 Hz, maka akan terjadi kedipan yang mengakibatkan peragaan di layar akan tampak nyala dan padam secara bergantian.

2.1.2 Osiloskop Digital

Jika pada osiloskop analog gelombang yang ditampilkan pada layar langsung diberikan dari rangkaian lapisan pembelok pancaran elektron vertikal sehingga berkesan "*real time*", maka pada osiloskop digital gelombang yang akan ditampilkan terlebih dahulu melalui tahap *sampling* (pencuplikan sinyal) dan kemudian data hasil *sampling* tersebut diolah secara digital. Osiloskop digital menyimpan nilai-nilai tegangan hasil *sampling* tersebut bersama dengan skala waktu gelombangnya pada memory sementara sebelum kemudian ditampilkan. Pada prinsipnya osiloskop digital bekerja dengan cara mencuplik sinyal (*sampling*), menyimpan data, memproses data, kemudian menampilkan data hasil pemrosesan dan kemudian akan berulang kembali seperti itu.

Osiloskop digital mempunyai 2 (dua) cara untuk mencuplik sinyal masukan, yakni dengan cara *single shot* atau *real time sampling*. Dengan kedua teknik ini, osiloskop memperoleh semua cuplikan sinyal dengan satu *event* picu. Sayangnya laju cuplik osiloskop digital akan membatasi lebar pita (*bandwidth*) osiloskop ketika beroperasi dalam waktu nyata (*realtime*). Secara teori osiloskop digital membutuhkan masukan dengan minimal 2 (dua) cuplikan per periode sinyal (gelombang) untuk merekonstruksi suatu bentuk gelombang. Namun pada

praktiknya, 3 (tiga) atau lebih cuplikan setiap periode gelombang akan memberikan akurasi akuisisi yang lebih baik.

Apabila proses pencuplikan sinyal tidak dapat sama cepat dengan sinyal masukan yang disamplingnya, maka osiloskop tidak akan dapat mengumpulkan suatu jumlah yang cukup untuk merekonstruksi bentuk sinyal yang dicupliknya sehingga akan berakibat menghasilkan suatu peragaan yang lain dari bentuk gelombang aslinya.

Dengan menggunakan metode alternatif lain yakni menggunakan *equivalent-time sampling* osiloskop digital secara akurat dapat menangkap sinyal-sinyal yang sifatnya repetitif. Dengan menggunakan teknik ini, osiloskop digital menerima cuplikan-cuplikan pada banya *event-event* picu yang kemudian secara berangsur-angsur merekonstruksi keseluruhan bentuk gelombangnya. Namun hal ini hanya dapat diterapkan pada *bandwidth analog* pada frekuensi tertentu saja.

Pada umumnya osiloskop digital baik menggunakan teknik *realtime* maupun *equivalent time sampling* tetap akan mencuplik sinyal pada laju maksimum tanpa mengacu terhadap berapa dasar waktu (*time base*) yang digunakan.

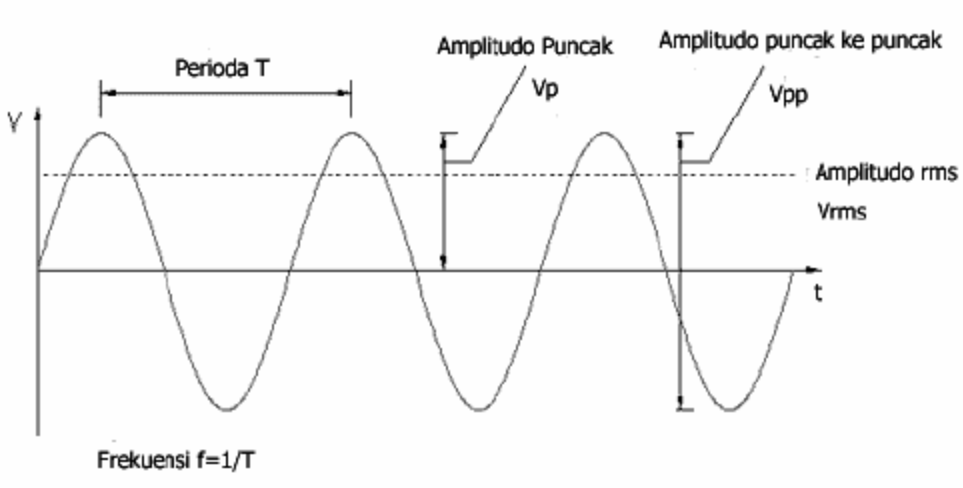
2.2 Konsep Dasar Sinyal

Sinyal merupakan besaran fisis fungsi waktu yang berisikan informasi. Sinyal biasanya rentan terhadap gangguan dan interferensi dari sinyal-sinyal lain baik dari dalam sistem maupun dari luar sistem. Dalam dunia elektronika terdapat 2 (dua) jenis sinyal yang secara umum dikenal, yaitu :

1. sinyal analog, dan
2. sinyal digital.

Kedua sinyal tersebut memiliki karakteristik masing-masing, dan dalam penggunaannya pun berbeda antara satu dengan yang lainnya, karena masing-masing sinyal tersebut memiliki kelebihan dan kekurangan sesuai dengan media transmisi dan jenis komunikasi yang digunakannya.

2.2.1 Sinyal Analog



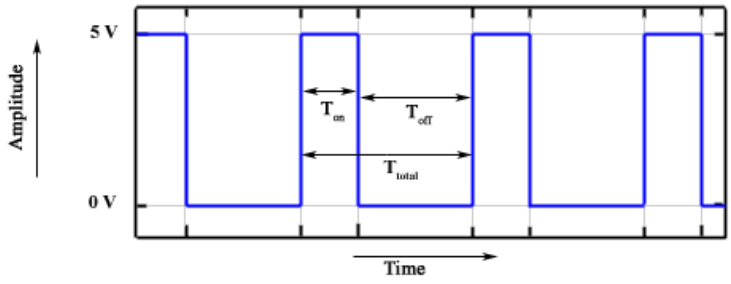
Gambar 2.1 Sinyal Analog

Sinyal analog adalah sinyal data dalam bentuk gelombang yang bersifat kontinyu, yang membawa informasi dengan mengubah karakterisitik gelombang. Dua parameter/karakteristik terpenting yang dimiliki oleh gelombang isyarat analog adalah amplitudo dan frekuensi. Isyarat analog ini biasanya dinyatakan dengan gelombang sinus, mengingat gelombang sinus merupakan dasar untuk semua bentuk isyarat analog. Gelombang pada sinyal analog umumnya berbentuk gelombang sinus yang memiliki 3 (tiga) variabel dasar, yaitu :

1. **amplitudo** merupakan ukuran tinggi rendahnya tegangan dari sinyal analog,
2. **frekuensi** adalah jumlah gelombang sinyal analog dalam satuan detik, dan
3. **fasa** adalah besar sudut sinyal analog pada saat-saat tertentu.

2.2.2 Sinyal Digital

Sinyal digital merupakan sinyal data dalam bentuk pulsa yang dapat mengalami perubahan yang tiba-tiba dan mempunyai harga "0" (logika *low*) dan "1" (logika *high*). Sinyal digital hanya memiliki 2 (dua) keadaan, yaitu "0" dan "1", sehingga tidak mudah terpengaruh oleh derau (*noise*). Pada umumnya sinyal digital disebut juga sebagai sinyal diskrit. Karena sinyal digital hanya memiliki dua keadaan saja maka nilai sinyal digital ini biasanya disebut juga dengan bit. Bit merupakan istilah khas pada sinyal digital. Kemungkinan nilai untuk sebuah bit adalah 2 buah, dan kemungkinan nilai untuk 2 (dua) bit adalah 4 buah (00, 01, 10, dan 11). Secara umum jumlah kemungkinan yang dapat terbentuk dari suatu kombinasi *n* bit adalah sebesar 2^n buah.



Gambar 2.2 Sinyal Digital

Sistem sinyal digital merupakan hasil pencuplikan dari sinyal analog. Sinyal digital pada dasarnya di kode-kan kedalam bentuk biner atau kedalam

bentuk hexadesimal. Besar nilai suatu sinyal digital dibatasi oleh lebar pita (*bandwidth*) data atau jumlah bit yang digunakannya. Semakin besar jumlah bit yang digunakan maka nilai pembacaan hasil *sampling* akan semakin akurat.

Berikut merupakan beberapa keistimewaan dari sistem sinyal digital yang tidak dimiliki oleh sistem sinyal analog yaitu :

1. mampu mengirimkan informasi dengan kecepatan tinggi,
2. penggunaan yang berulang-ulang terhadap informasi tidak akan mempengaruhi kualitas dan kuantitas informasi itu sendiri,
3. informasi dapat dengan mudah diproses dan dimodifikasi kedalam bentuk lain,
4. dapat memproses informasi dalam jumlah besar dan dapat mengirimkannya secara interaktif antar sistem, dan
5. lebih kebal terhadap noise.

Dari keistimewaan diatas maka sistem sinyal digital-lah yang biasanya digunakan pada perangkat-perangkat modern masa kini seperti : mikrokontroler, mikroprosesor, komputer, *handphone*, dan perangkat lainnya. Agar sinyal analog dapat diolah secara digital oleh perangkat-perangkat tersebut, maka besaran sinyal analog harus terlebih dahulu dikonversikan kedalam besaran sinyal digital dengan menggunakan perangkat ADC (*Analog to Digital Converter*).

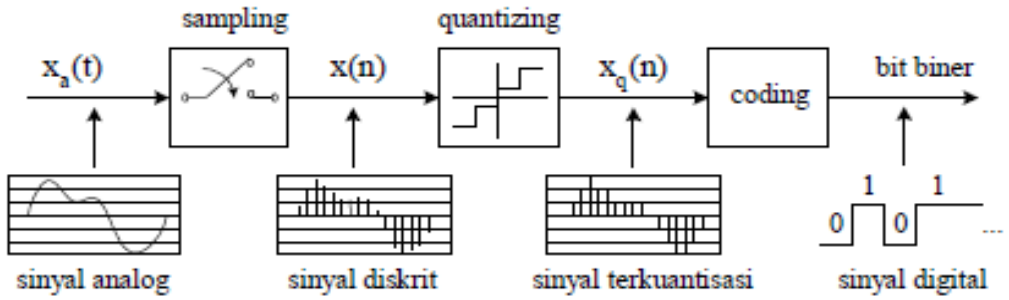
2.3 ADC (*Analog to Digital Converter*)

Analog to Digital Converter atau yang biasa disebut dengan ADC merupakan suatu rangkaian yang berfungsi untuk mengubah sinyal analog menjadi sinyal digital. Dengan menggunakan ADC, kita dapat mengamati

perubahan sinyal-sinyal seperti sinyal suara, sinyal radar, sinyal sonar, dan berbagai sinyal-sinyal lain yang merupakan sinyal analog. Hal yang paling penting dari suatu rangkaian ADC adalah resolusi, yaitu besaran analog terkecil yang dapat dikonversikan menjadi satuan digital.

Untuk memproses sinyal analog dengan perangkat digital, pertama-tama perlu mengkonversikan terlebih dahulu dari besaran analog kedalam besaran digital yaitu dengan mengkonversi menjadi suatu deret angka yang mempunyai presisi terbatas yang dilambangkan kedalam bentuk biner. Prosedur ini dinamakan konversi analog-ke-digital (*A/D converter*).

Sebuah sinyal mengandung informasi tentang amplitudo, frekuensi dan sudut fasa. Untuk mendapatkan informasi tersebut dari sebuah sinyal menggunakan perangkat analog adalah rumit dan kurang akurat. Oleh karena itu biasanya untuk memprosesnya digunakan metode pengolahan secara digital. Agar sinyal digital yang didapatkan cukup akurat untuk dapat dikembalikan menjadi sinyal analog maka perlu diperhatikan jumlah cuplikan (*sampling*) oleh perangkat ADC dan besarnya angka yang dipakai untuk mewakili tiap cuplikannya.



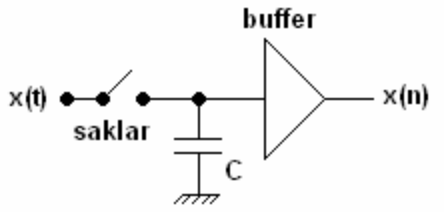
Gambar 2.3 Proses Konversi Sinyal Analog ke Digital

Secara umum proses pengkonversian sinyal terbagi menjadi 3 (tiga) langkah seperti yang digambarkan pada Gambar 2.1 yaitu :

1. pencuplikan (*sampling*),
2. kuntisasi (*quantizing*), dan
3. pengkodean (*coding*).

2.3.1 *Sampling* (Pencuplikan)

Proses pencuplikan secara sederhana ditunjukan oleh Gambar 2.4. apabila saklar ditutup sebentar kemudiian dibuka kembali maka kapasitor C akan terisi muatan yang sama besar dengan besar sinyal x(t) saat saklar ditutup. Rangkaian *buffer* ditambakan agar muatan pada kapasitor tetap terjaga ketika sedang digunakan oleh proses yang berikutnya.



Gambar 2.4 Pencuplikan Sinyal

Secara matematis proses konversi suatu sinyal analog waktu-kontinyu $x_n(t)$ menjadi sinyal waktu-diskrit yang bernilai kontinyu x(n) diperoleh dengan cara mengambil "cuplikan" sinyal waktu-kontinyu pada saat waktu diskrit. Sehingga dapat direpresentasikan kedalam persamaan :

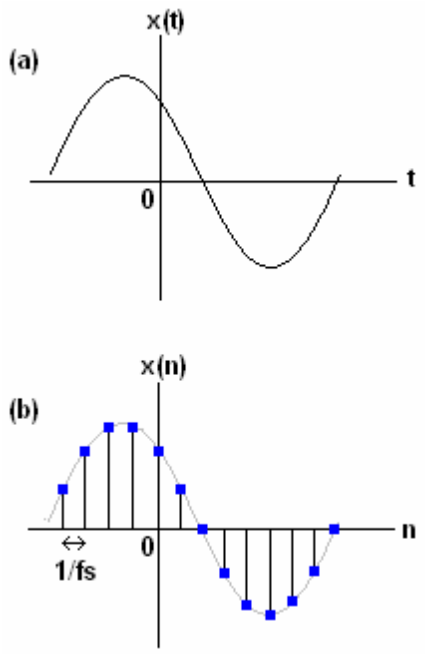
$$x_a(t) = x_a(t)|_{t=nT} = x_a(nT) = x_a\left({}^{n}/_{f_s}\right) \Rightarrow x(n) \qquad (2.1)$$

Dimana :

T = interval pencuplikan (detik)

Fs = laju pencuplikan (Hz) = 1/T

n = bilangan bulat, $-\infty < n < \infty$



Gambar 2.5 Proses Pencuplikan. (a) Sinyal Analog, (b) Hasil Pencuplikan Sinyal

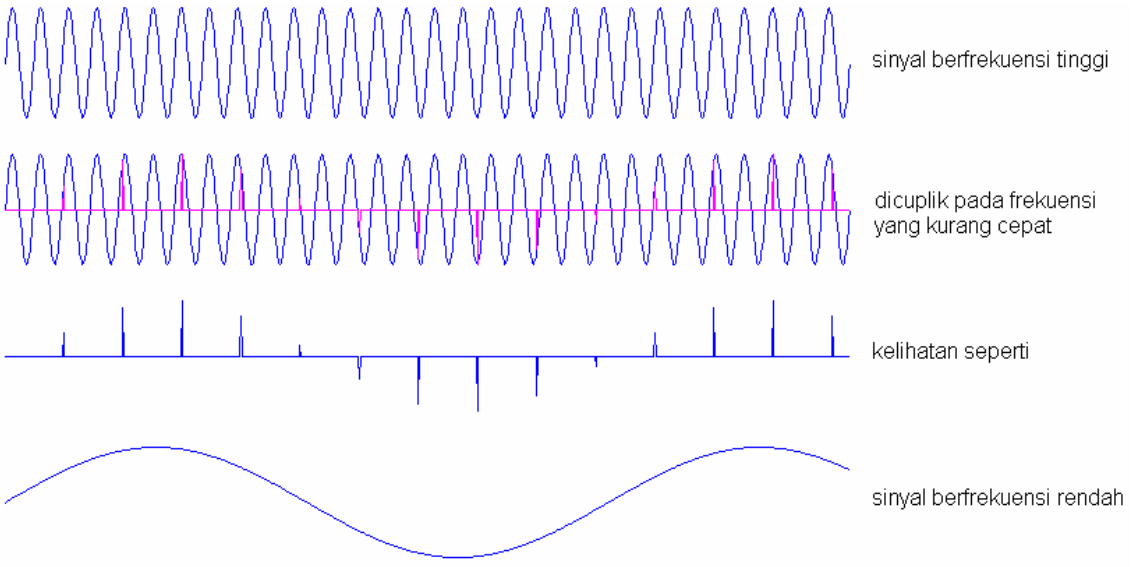
Kaidah Pencuplikan Sinyal

Kecepatan pengambilan sampel sinyal (pencuplikan) dari sinyal analog yang akan dikonversi haruslah memenuhi kriteria *Nyquist* yaitu :

$$f_s > 2f_{in\,max} \tag{2.2}$$

Dimana frekuensi *sampling* (Fs) minimum adalah 2 (dua) kali frekuensi sinyal analog maksimum yang akan dikonversikan (Finmax). Misalnya apabila sinyal analog yang akan dikonversi mempunyai frekuensi sebesar 100 Hz maka frekuensi *sampling* dari ADC minimal 200 Hz. Atau bila dibalik, apabila frekuensi *sampling* ADC sebesar 200 Hz maka sinyal analog yang akan

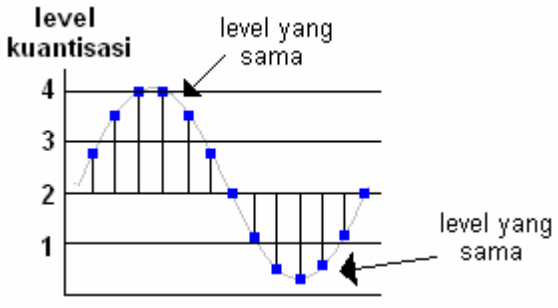
dikonversi harus mempunyai frekuensi maksimum sebesar 100 Hz. Apabila kriteria *Nyquist* ini tidak dipenuhi maka akan timbul efek yang disebut *aliasing* karena frekuensi tertentu terlihat sebagai frekuensi yang lain.



Gambar 2.6 *Aliasing* Sinyal Karena Tidak Sesuai dengan Kaidah Pencuplikan Sinyal.

2.3.2 Kuantisasi (*Quantizing*)

Sinyal digital merupakan sebuah deretan angka hasil pencuplikan yang diwakili oleh beberapa digit dengan jumlah tertentu yang menentukan keakuratan pencuplikan sinyal. Proses melakukan konversi sinyal yang telah dicuplik menjadi sinyal digital yang diwakili oleh sebuah nilai dengan jumlah digit tertentu disebut dengan kuantisasi.



Gambar 2.7 Proses Kuantisasi

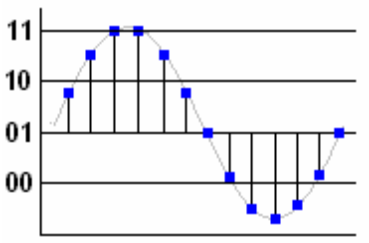
Gambar 2.7 menunjukan contoh proses kuantisasi yang menggunakan empat level. Pada gambar tersebut terdapat 4 buah sinyal yang menempati level yang sama, yang artinya keempat sinyal tersebut dikelompokkan kedalam level yang sama walaupun pada sinyal yang sebenarnya terdapat perbedaan meski perbedaan tersebut tidak terlalu jauh. Selisih antara nilai kuantisasi dengan sinyal sebenarnya disebut dengan galat kuantisasi (*quantization error*). Dimana galat tersebut dapat ditunjukkan pada persamaan (2.3).

$$e_q = x_q(n) - x(n) \tag{2.3}$$

Jarak antar level kuantisasi tersebut dinamakan resolusi. Sebagai contoh apabila suatu peranti ADC memiliki resolusi 10-bit, maka level kuantisasi dari rentang tegangan maksimum peranti ADC hingga teganan minimumnya tersebut terdapat 1023 level. Kuantisasi merupakan proses yang tidak dapat dibalik sehingga menyebabkan distorsi sinyal yang tidak dapat diperbaiki. Untuk mengurangi galat kuatisasi tersebut maka hal yang perlu dilakukan adalah dengan memilih peranti ADC yang memiliki tingkat resolusi yang tinggi sehingga akan memiliki level kuantisasi yang lebih banyak lagi.

2.3.3 Pengkodean (*Coding*)

Proses pengkodean dalam peranti ADC menetapkan bilangan biner tertentu pada setiap level kuantisasi. Bila terdapat level kuantisasi sejumlah *N*, maka bilangan biner yang diperlukan setidaknya $2^b \geq N$.



Gambar 2.8 Proses Pengkodean

Pada Gambar 2.8 terdapat 4 (empat) level kuantisasi, sehingga pengkodean yang dibutuhkan hanya 2-bit, dimana dari 2-bit tersebut akan didapatkan 4 keadaan yaitu : 00, 01, 10, 11.

2.4 Op-Amp (*Operational Amplifier*)

Operational amplifier atau yang biasa disebut sebagai op-amp merupakan sejenis IC yang didalam nya terdiri dari beberapa komponen pasif seperti transistor, resistor, dan dioda yang telah didesain sedemikian rupa sehingga menjadi sebuah komponen yang dapat digunakan untuk berbagai macam fungsi.

Beberapa aplikasi op-amp yang sering digunakan diantaranya rangkaian dasar penguat diferensial, rangkaian *buffer* sinyal, rangkaian penguat tak-membalik (*non-inverting amplifier*), rangkaian penguat membalik (*inverting amplifier*) dan rangkaian penjumlah (*adder*).

Op-amp memiliki 2 (dua) rangkaian umpan balik (*feedback*) yaitu rangkaian *feedback* positif dan rangkaian *feedback* negatif, rangkaian *feedback* negatif memiliki peranan yang sangat penting karena rangkaian tersebut dapat menghasilkan penguatan yang dapat terukur sedangkan rangkaian *feedback* positif dapat menghasilkan osilasi (sinyal yang berosilasi).

2.4.1 Op-amp Ideal

Pada dasarnya op-amp adalah sebuah penguat diferensial (*differential amplifier*) yang memiliki 2 (dua) masukan yaitu masukan membalik (*inverting input*) dan masukan tak-membalik (*non-inverting input*). Op-amp yang ideal seharusnya memiliki penguatan loop terbuka (*open loop gain*) yang tak terhingga besarnya. Penguatan yang sangat besar tersebut membuat op-amp menjadi tidak stabil, dan penguatan yang keluarnya pun mejadi tidak terukur (*infinite*). Maka diperlukanlah peran rangkaian *feedback* negatif, sehingga op-amp dapat dirangkai menjadi aplikasi dengan nilai penguatan yang terukur (*finite*).

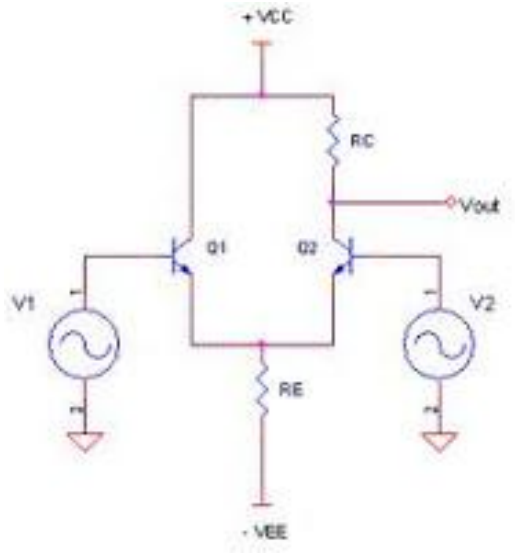
Impedansi masukan op-amp yang ideal seharusnya memiliki besar impedansi yang tak terhingga sehingga arus yang masuk kedalam rangkaian op-amp tersebut sekitar ~0 Volt. Sebagai contoh jenis op-amp LF-353 memiliki besar impedansi masukan sebesar 10^{12} Ohm. Nilai impedansi tersebut relatif sangat besar sehingga arus masukan terhadap op-amp LF-353 sangatlah kecil.

Terdapat 2 (dua) aturan penting dalam melakukan analisis terhadap rangkaian op-amp berdasarkan karakteristik op-amp ideal. Dalam beberapa literatur, aturan ini dinamakan sebagai *golden rule* yang isinya adalah sebagai berikut :

1. perbedaan tegangan antara tegangan V+ dan V- pada op-amp adalah nol (V+ - V- = 0 atau V+ = V-)
2. arus pada rangakaian masukan op-amp adalah nol (i+ = i- = 0).

2.4.2 Karakteristik Dasar Op-amp

Seperti yang telah dijelaskan sebelumnya bahwa pada dasarnya op-amp merupakan penguat diferensial (*differential amplifier*) dimana op-amp juga memiliki 2 (dua) unit masukan yaitu masukan pembalik (*inverting input*) dan masukan tak-membalik (*non-inverting input*). Rangkaian dasar dari penguat diferensial adalah sebagai berikut.

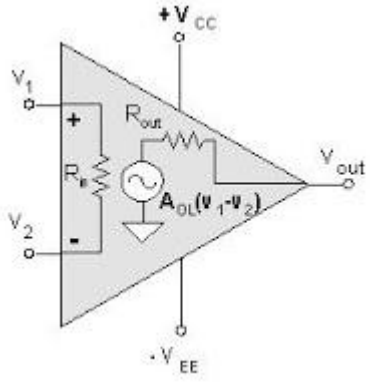


Gambar 2.9 Rangkaian Penguat Diferensial

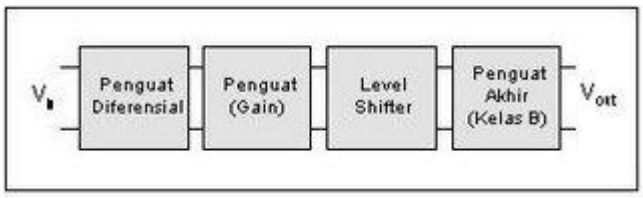
Pada Gambar 2.9 menunjukkan bahwa tegangan output dari rangkaian tersebut adalah Vout = A(V1 – V2), dengan A adalah nilai penguatan dari penguat diferensialnya. Masukan pada titik V1 dikatakan sebagai masukan tak-membalik karena fasa tegangan keluaran pada Vout sama dengan fasa tegangan masukan pada V1, sedangkan masukan pada titik V2 dikatakan sebagai masukan pembalik

karena fasa tegangan keluaran pada Vout berlawanan dengan fasa tegangan masukan pada V2.

2.4.3 Blok Diagram Op-amp



(a)



(b)

Gambar 2.10 (a) Diagram Skematik Op-Amp, (b) Blok Diagram Op-Amp

Didalam op-amp terdapat beberapa bagian, pertama adalah penguat diferensial, lalu bagian penguatan (*gain*), kemudian rangkaian penggeser level (*level shifter*) dan yang terakhir adalah penguatan akhir yang biasanya dibuat menggunakan metode *push-pull amplifier* kelas b.

Pada Gambar 2.10 (a) dapat dilihat terdapat 2 (dua) buah masukan yaitu masukan tak-membalik (+) dan masukan pembalik (-). Umumnya op-amp bekerja dengan menggunakan catu daya simetrik (+Vcc dan –Vcc) namun ada juga beberapa jenis op-amp yang bekerja hanya menggunakan catu daya tunggal (+Vcc

dan *Ground*). R_{in} adalah resistansi masukan yang nilai idealnya tak berhingga, sedangkan R_{out} adalah resistansi keluaran yang besarnya ~ 0 Ohm. A_{OL} adalah nilai penguatan loop terbuka dari op-amp tersebut yang biasanya besar penguatannya tak berhingga.

2.5 Mikrokontroler

Mikrokontroler adalah suatu terobosan teknologi prosesor yang hadir untuk memenuhi kebutuhan akan perkembangan teknologi yang begitu pesat di masa kini. Mikrokontroler dibuat dengan teknologi semikonduktor dimana mikrokontroler tersebut dibangun oleh transistor-transistor dengan jumlah yang sangat banyak dan dirangkai sedemikian rupa sehingga dapat bekerja sebagai unit pemroses (kontroler).

Mikrokontroler berbeda dengan mikroprosesor, perbedaan tersebut dapat dilihat dari kecepatan proses, cara menangani tugas, dan unit-unit pendukung lain yang bertugas sesuai dengan fungsinya. Pada mikrokontroler sudah terdapat RAM, ROM, dan CPU didalam IC-nya, sedangkan pada mikroprosesor hanya terdapat CPU saja dan bagian RAM dan ROM nya terpisah sehingga agar mikroprosesor dapat bekerja sesuai dengan fungsinya perlu ditambahkan perangkat penunjang lain seperti rangkaian RAM dan ROM. Karena perbedaan dalam arsitekturnya maka untuk kecepatan proses masih tetap lebih unggul mikroprosesor, namun untuk proses-proses yang tidak memiliki kompleksitas kerja yang besar mikrokontroler lebih dibutuhkan karena dalam segi biaya lebih murah mikrokokntroler dan dalam segi ergonomisnya pun lebih praktis menggunakan mikrokontroler, karena didalam mikrokontroler telah

terdapat rangkaian RAM, ROM, CPU, dan unit I/O yang siap digunakan untuk keperluan apapun.

2.6 Komunikasi USB

USB (*Universal Serial Bus*) adalah sebuah standard komunikasi serial yang digunakan untuk komunikasi antar perangkat. Pada awalnya sistem USB didesain dari perkembangan sebuah antarmuka untuk berkomunikasi dengan bermacam-macam tipe periferal tanpa batasan dan kesulitan dalam penggunaanya, tidak seperti pada perangkat antarmuka sebelumnya. Perangkat antarmuka USB memiliki banyak kelebihna dibanding perangkat antarmuka sebelumnya, seperti :

- mudah untuk digunakan, sehingga tidak perlu lagi mengotak-atik konfigurasi-konfigurasi dan *setup* yang rumit
- cepat, sehingga tidak akan terjadi kemacetan komunikasi pada peranti antarmukanya
- dapat dipercaya, karena tingka kesalahan komunikasi (galat) jarang terjadi, karena menggunakan metode *automatic retries* (pengulang otomatis) ketika terjadi kesalahan
- serbaguna, banyak macam perangkat periferal yang dapat menggunakan peranti antarmuka ini
- biaya yang minim, sehingga dalam pembuatan peralatannya tidak memerlukan dana yang banyak
- daya rendah, artinya dapat menghemat penggunaan daya pada *portable computer* (laptop)

- didukung oleh sistem operasi Windows dan sistem operasi yang lain, sehingga dapat mempermudah pengembang untuk mengembangkan perangkat antarmuka yang diinginkannya.

Pada setiap komputer masa kini telah terdapat *port* USB yang dapat digunakan untuk menghubungkan perangkat lain (periferal) seperti *keyboard*, *mouse*, *scanner*, *digital camera*, *printer* dan peralatan lain sebagai perangkat tambahan dengan masing-masing kegunaanya.

USB merupakan solusi komunikasi antara komputer dengan perangkat lain yang dibutuhkan oleh sistem komputer tersebut, karena sistem antarmukanya cocok untuk semua tipe perangkat yang standard. Suatu sistem USB pada umumnya terdiri dari beberapa bagian diantaranya :

- *host controller*, pada sistem USB terdapat beberapa *host* yang bertanggung jawab pada keseluruhan protokol sistem USB. *Host controller* bertugas mengendalikan penggunaan jalur *bus* data, sehingga tidak ada satu pun peralatan USB yang dapat menggunakan jalur *bus* data kecuali mendapat persetujuan dari *host controller*
- *hub*, seperti halnya hub untuk jaringan komputer, USB hub menyediakan titik interkoneksi yang dapat memungkinkan banyak peralatan USB untuk dapat terhubung terhadap *host controller*. Topologi jaringan yang digunakan oleh sistem USB adalah topologi *star*, semua perangkat USB secara logika terhubung langsung dengan *host controller*. Hub terhubung dengan USB *host controller* secara *upstream* (data mengalir menuju ke host) dan

terhubung dengan peralatan USB secara *downstream* (data mengalir dari *host* ke perangkat USB). Fungsi utama dari hub adalah bertanggungjawab untuk mendeteksi pada pemasangan dan pelepasan peralatan USB dengan *port* USB

- *peralatan USB*, semua hal pada sistem USB selain *host controller* merupakan peralatan USB. Dalam kecepatan transfer datanya peralatan USB dikelompokan menjadi 3 (tiga) yaitu : *low speed* (kecepatan transfer hingga 1,5 Mbps), *full speed* (kecepatan transfer hingga 12 Mbps), dan *high speed* (kecepatan transfer data hingga 480 Mbps).

2.6.1 Evolusi Sistem Antarmuka

Alasan utama mengapa suatu sistem antarmuka yang baru tidak begitu sering muncul, hal tersebut dikarenakan sistem antarmuka yang ada telah banyak menarik perhatian pengguna untuk tidak mengambil hal yang rumit untuk membuat sistem antarmuka yang baru. Menggunakan sistem antarmuka yang telah ada membuat lebih hemat biaya dan waktu dalam pembuatan desain sistem antarmuka peralatannya. Hal inilah yang membuat IBM PC memilih kompatibilitas sistem antarmukanya menggunakan *centronics parallel interface* dan *RS-232 serial port interface* yang sudah ada untuk menghubungkan perangkat pengguna seperti *printer* dan *modem* yang ada di pasaran.

Sistem antarmuka standard yang digunakan IBM PC telah membuktikan kehandalannya dalam 2 (dua) dekade kebelakang. Namun semakin canggih dan hebatnya komputer-komputer masa kini membuat semakin

meningkatnya jumlah perangkat-perangkat (periferal) pendukung komputer, sehingga sistem antarmuka yang terdahulu sudah tidak dapat menanganinya lagi dikarenakan kecepatan komunikasinya yang terbatas dan memiliki ekspansi antarmuka yang terbatas juga. Hal inilah yang kemudian merujuk kepada pengembangan sistem antarmuka USB.

2.6.2 Tugas Komputer sebagai *Host*

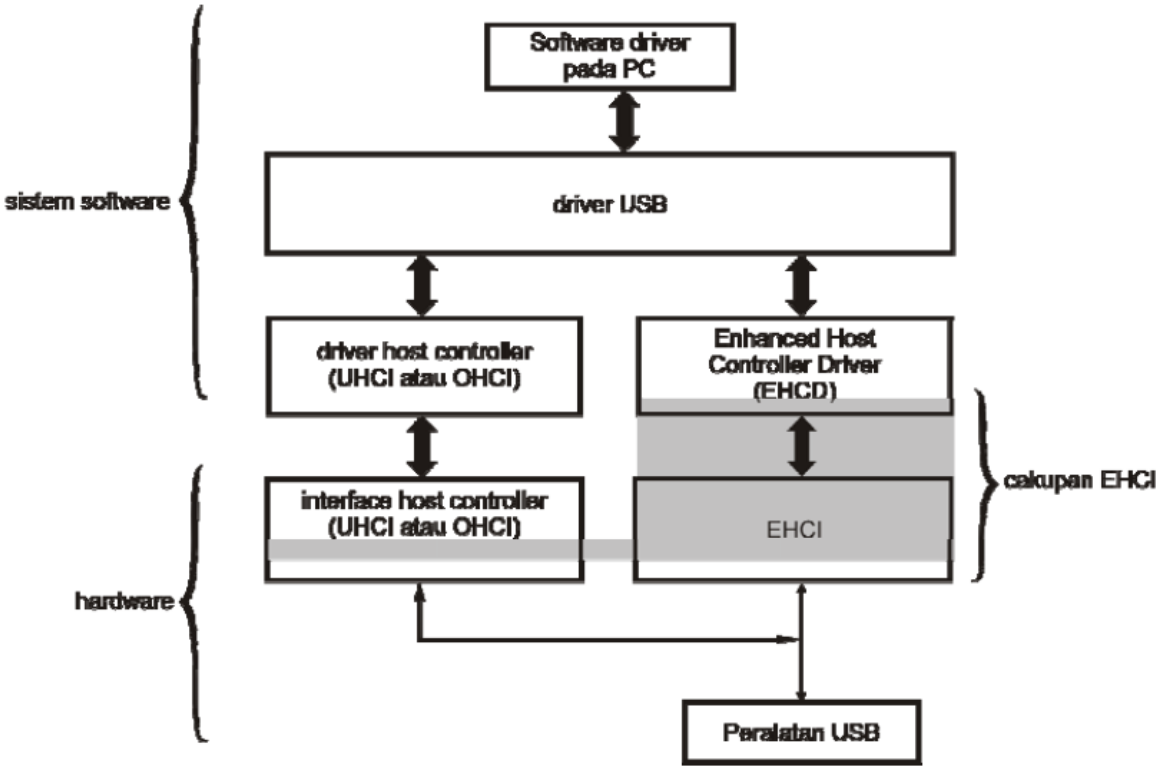
Dalam hal ini komputer akan bekerja sebagai penyedia (*host*) sedangkan perangkat akan bekerja sebagai periferal yang memiliki peranan yang telah ditentukan masing-masing sesuai dengan tugasnya. Untuk dapat berkomunikasi dengan perangkat USB, sebuah komputer memerlukan suatu perangkat keras (*hardware*) dan perangkat lunak (*software*) yang dapat membuat komputer bekerja sebagai penyedia (*host*) USB.

Perangkat keras (*hardware*) yang dibutuhkan oleh bagian penyedia (*host*) diantaranya adalah USB *host controller* dan USB hub yang dapat membuat *port* USB menjadi lebih dari 1 (satu). Sedangkan perangkat lunak (*software*) yang dibutuhkan oleh bagian penyedia (*host*) adalah *host controller driver*, dan USB *driver*-nya masing-masing sesuai dengan perangkat USB yang dihubungkan dengan komputer.

Pada awalnya USB *host controller* didesain untuk menangani satu peralatan USB saja, namun dalam perkembangannya muncullah *Universal/Open Host Controller Intreface* (UHCI/OHCI) yang terdiri dari 2 (dua) bagian yaitu : *host controller driver* , dan *host controller*. *Host controller driver* adalah suatu program perangkat lunak komputer yang bertanggung jawab dalam penjadwalan

lalu lintas data pada jalur *bus* data dengan cara menempatakan dan menjaga transaksi data dalam sistem memori, sementara *host controller* bekerja memindahkan data dari sistem memori ke peralatan USB dengan cara memproses struktur data.

Selanjutnua untuk dapat memenuhi kebutuhan transfer data dengan kecepatan tinggi, maka dikembangkan *Enhanced Host Controller Interface* (EHCI). EHCI mampu mendukung peralatan USB *high speed* karena memang didesain untuk efisiensi dalam penggunaan memori pada peranti *host controller*.



Gambar 2.11 Blok Diagram Sistem USB

Sistem USB *host controller* terdiri atas sejumlah lapisan perangkat keras (*hardware*) dan perangkat lunak (*software*) yang ditunjukkan pada Gambar 2.13. Secara umum penjelasan singkat dari setiap lapisan-lapisan tersebut adalah :

- ***software driver pada PC***, bertugas mengeksekusi perintah dari *host controller* yang bersesuaian dengan peralatan USB tertentu sesuai fungsinya. *Driver* peralatan USB secara khusus merupakan bagian dari sistem operasi atau yang disertakan dengan peralatan USB yang kemudian di-*install* pada PC sehingga peralatan USB tersebut dapat digunakan sesuai fungsinya
- ***driver USB***, adalah suatu perangkat lunak *bus driver* yang memisahkan detil *host controller* tertentu untuk sistem operasi tertentu
- ***host controller driver***, berfungsi menyediakan *driver* perangkat lunak diantara lapisan perangkat keras *host controller* dengan pealatan USB. Detil dari *host controller driver* tergantung kepada sistem antarmuka perangkat keras USB *host controller*
- ***host controller***, adalah lapisan implementasi perangkat keras yang spesifik. Terdapat 1 (satu) spesifikasi *host controller* yang digunakan untuk peralatan USB *high speed* yaitu *Enhanced Host Controller Interface* (EHCI), dan terdapat 2 (dua) spesifikasi *host controller* yang digunakan untuk peralatan USB *full speed* dan *low speed* yaitu *Universal Host Controller Interface* (UHCI) dan *Open Host Controller Interface* (OHCI)
- ***peralatan USB***, adalah suatu perangkat keras yang akan melaksanakan fungsinya dengan pengguna. Interaksi antara pengguna dengan peralatan USB mengalir dari aplikasi melalui

lapisan-lapisan perangkat lunak dan perangkat keras sesuai dengan diagram pada Gambar 2.13.

2.6.3 Lapisan Komunikasi USB

Komunikasi data pada sistem USB terbagi menjadi 3 (tiga) lapisan yaitu : *physical layer*, *protocol engine layer*, dan *application layer*. Logika komunikasi antara perangkat lunak *driver host controller* dengan fungsi peralatan USB menggunakan sebuah jalur komunikasi pipa (*pipes*). Jalur pipa ini diasosiasikan sebagai jalur antara *endpoint* pada peralatan USB dengan perangkat lunak *host controller* yang bersesuaian.

Endpoint adalah sumber atau tujuan dari data yang ditransmisikan melalui antarmuka USB, karena pada sistem antarmuka USB terdidri dari seperangkat *endpoint* yang terkelompok. Aliran komunikasi data pada *bus* data yang dapat dilakukan adalah dua arah yaitu : *IN* dimana data mengalir dari peralatan USB ke bagian penyedia (*host*) dan *OUT* dimana data mengalir dari bagian penyedia (*host*) ke peralatan USB. Berikut merupakan penjelasan singkat dari masing-masing lapisan komunikasi USB.

2.6.3.1 *Physical Layer*

Physical layer adalah lapisan yang meliputi antarmuka fisik ke kabel USB, *Serial Interface Engine* (SIE), dan *Host Controller* (HC). Fungsi utama dari bagian *physical layer* ini adalah mentransmisikan dan menerima data USB. Sistem kabel dalam USB adalah menggunakan 4 (empat) kawat *twisted pair*, yang terdiri dari V*bus*, D+, D-, dan *Ground*. Persinyalan komunikasi data dilakukan

oleh sepasang kabel yaitu D+ dan D-, jika logika data yang akan ditransmisikan adalah data 0 (*low*) maka persinyalan D+ adalah *low* dan D- adalah *high*. Apabila data yang akan ditransmisikan adalah data 1 (*high*) maka persinyalan D+ adalah *high* dan D- adalah *low*. Sedangkan kabel V*bus* dan *Ground* digunakan sebagai sumber catu daya perangkat USB dari bagian penyedia (*host*) dalam hal ini komputer.

Serial Interface Engine (SIE) bertanggung jawab dalam sistem *decoding* dan *encoding* data pada transmisi USB. Selain itu bagian ini juga memberikan sinyal CRC untuk setiap data USB yang akan ditransmisikan dan memeriksa sinyal CRC untuk data USB yang masuk kedalam *bus host controller*. Bagian SIE juga bertugas mendeteksi persinyalan SOP (*Start of Packets*), EOP (*End of Packets*), *RESET* dan *RESUME* pada *bus* USB.

Host Controller (HC) merupakan pengendali dalam sistem USB, menerima semua transaksi, mengendalikan akses jalur *bus* USB, dan merupakan mesin utama dalam aliran protokol komunikasi USB. Fungsi-fungsi dari bagian *host controller* itu sendiri adalah : memproses data, penanganan galat komunikasi, menagani tingkat protokol komunikasi, dan pembentukan *frame*.

2.6.3.2 *Protocol Engine Layer*

Protocol Engine Layer adalah lapisan komunikasi yang berfungsi mentranslasikan data antara lapisan aplikasi dengan protokol transaksi USB. Didalam lapisan ini terdapat bagian yang bertanggung jawab transfer data berlangsung dengan sukses tanpa terjadi galat, hal tersebut dilakukan dengan beberapa metode komunikasi diantaranya adalah :

- ***handshaking***, adalah suatu metoda kontrol komunikasi dengan menggunakan informasi status dan kontrol sehingga dapat mengetahui aliran data yang sedang berlangsung mengindikasikan kesuksesan atau kegagalan
- ***Ack* (*Acknowledge*)**, mengindikasikan bahwa komputer atau perangkat telah menerima data tanpa kesalahan (galat). Perangkat harus memberikan sinyal Ack dalam paket *handshake* pada transaksi *IN*, dan bagian penyedia pun harus memberikan sinyal Ack dalam paket *handshake* pada transaksi *OUT*
- ***Nak* (*Negative Acknowledge*)**, mengindikasikan bahwa komputer atau perangkat sedang berada dalam keadaan sibuk atau tidak ada data yang harus diberikan. Apabila komputer mengirimkan data pada waktu perangkat sedang sangat sibuk untuk menerima data, perangkat harus memberikan sinyal *Nak* didalam paket *handshake*.
- ***STALL***, memiliki 3 (tiga) pengertian yaitu : permintaan kontrol yang tidak didukung (*unsupported control request*), kegagalan permintaan kontrol (*control request failed*) atau kegagalan *endpoint* (*endpoint failed*). Ketika perangkat menerima permintaan kendali transfer yang tidak didukung oleh bagian penyedia (*host*) maka perangkat harus memberikan sinyal *Stall* kepada komputer
- ***NYET***, hanya peranti USB *high speed* yang menggunakan *NYET*, yang berarti *stands for not yet*
- ***ERR***, komunikasi jenis ini mengindikasikan perangkat tidak memberikan *handshake* yang diharapkan dalam suatu transaksi komunikasi

- ***No Response***, mengindikasikan status yang terjadi ketika perangkat dan komputer menerima *handshake* namun data yang diterimanya adalah *dummy* (kosong / bukan data yang diharapkan).

2.6.3.3 *Application Layer*

Lapisan aplikai (*application layer*) ini berada pada 2 (dua) tempat yang berbeda yaitu pada bagian *host controller* dan pada peralatan USB. Pada USB *host controller* lapisan ini disebut dengan USB *System Software* sedangkan pada peralatan USB disebut sebagai *firmware* USB *device*.

USB *System Software* berfungsi dalam pengalokasian lebar jalur data (*bandwidth*) dan manajemen daya pada *bus* agar tegangan dan daya pada jalur juga komunikasi pada jalur *bus* data tetap terkendali.

2.6.4 Macam-macam Jenis Transfer USB

USB didesain untuk menangani berbagai jenis tipe perangkat dengan bermacam-macam jenis kecepatan transfer data, respon waktu, dan pengkoreksi galat. 4 (empat) jenis data transfer yang dapat menangani kebutuhan yang berbeda-beda dan dapat menyokong tipe transfer yang sesuai dengan kebutuhan dari perangkat. Pada Tabel 2.1 memperlihatkan macam-macam fitur dan kegunaan dari masing-masing tipe transfer.

Tabel 2.1 Macam dan Kegunaan Tipe Transfer USB

Transfer Type	**Control**	**Bulk**	**Interrupt**	**Isochronous**
Typical Use	Identification and Configuration	Printer, scanner, drive	Mouse, keyboard	Streaming audio, video
Required?	Yes	No	No	No
Low speed allowed?	Yes	No	Yes	No
Data bytes/millisecond per transfer, maximum possible per pipe (high speed)*	15,872 (thirty-one 64-byte transaction / microframe)	53,248 (thiteen 512-byte transaction / microframe)	24,576 (three 1024-byte transaction / microframe)	24,576 (three 1024-byte transaction / microframe)
Data bytes/millisecond per transfer, maximum possible per pipe (full speed)*	832 (thirten 64-byte transaction / frame)	1216 (nineteen 64-byte transaction / frame)	64 (one 64-byte transaction / frame)	1023 (one 1023-byte transaction / frame)
Data bytes/millisecond per transfer, maximum possible per pipe (low speed)*	24 (three 8-byte transaction	Not allowed	0.8 (8 bytes per 10 milliseconds)	Not allowed
Direction of data flow	IN and OUT	IN or OUT	IN or OUT (USB 1.0 support IN only)	IN or OUT
Reserved bandwidth for all transfer of the type (percent)	10 at low/full speed, 20 at high speeed (minimum)	None	90 at low/full speed, 80 at high speed (isochronous & interrupt combine maximum)	
Error correction	Yes	Yes	Yes	No
Message or Stream data?	Message	Stream	Stream	Stream
Guaranteed delivery rate?	No	No	No	Yes
Guaranteed latency (maximum time between tranfers)?	No	No	Yes	Yes
*Assumes transfer use maximum packet size.				

Control transfer adalah tipe transfer yang memiliki fungsi sesuai dengan spesifikasi USB yang telah ditentukan. *Control transfer* memperbolehkan komputer untuk membaca informasi perangkat, mengatur pengalamatan perangkat, dan memilih konfigurasi dan seting lainnya yang diperlukan. Semua perangkat USB harus dapat menyokong tipe transfer ini.

Bulk transfer dimaksudkan untuk sebuah situasi dimana rata-rata transfer tidak begitu kritis, seperti mengirimkan data kepada *printer*, menerima

data dari *scanner*, atau mengakses file-file dari *harddrive*. Untuk aplikasi ini, kecepatan transfer jenis ini sangat baik digunakan dan data dapat menunggu apabila diperlukan. Apabila *bus* data dalam keadaan sibuk, *bulk transfer* akan ditunda hingga *bus* data dalam keadaan siap, sebaliknya apabila *bus* data dalam keadaan menganggur, *bulk transfer* akan berlangsung dengan sangat cepat. Hanya dalam USB *full speed* dan *high speed* yang dapat melakukan komunikasi data dengan jenis *bulk transfer*.

Interrupt transfer adalah jenis transfer data untuk perangkat, dimana perangkat harus menerima permintaan dari komputer atau dari *driver* perangkat dari sisi penyedia secara periodik. Berbeda dengan *control transfer*, *interrupt transfer* merupakan cara lain untu perangkat USB *low speed* agar dapat melakukan transfer data dengan baik.

Isochronous transfer dapat menjamin waktu pengiriman datanya namun tidak memiliki pengkoreksi galat. Data yang dikirimkan menggunakan *isochronous transfer* biasanya adalah data audio atau video yang harus dimainkan secara langsung (reatime) / *streaming*. Jenis komunikasi *isochronous transfer* adalah salah satu tipe transfer yang tidak mendukung pengulangan pengiriman data secara otomatis ketika terjadi galat ketika pengiriman datanya, sehingga terkadang kesalahan harus diterima. Hanya untuk perangkat USB *full speed* dan *high speed* saja yang dapat menggunakan tipe komunikasi *isochronous transfer*.

2.7 USB – HID *Communication Class*

Human Interface Device (HID) *class* adalah salah satu kelas komunikasi USB yang didukung oleh hampir seluruh sistem operasi baik itu

sistem operasi Windows, mulai dari Windows ’98 hingga kini, juga sistem operasi yang lain. Biasanya *driver* untuk kelas komunikasi HID ini sudah terangkum menjadi satu kesatuan didalam sistem operasi sehingga tidak perlu lagi untuk meng-*install driver*. Oleh karena itu banyak *vendor-vendor* perangkat USB yang menggunakan kominikasi kelas HID dalam pembuatan produknya.

Pemberian nama HID (*Human Interface Device*) diambil dari perangkat antarmuka yang berhubungan langsung dengan manusia (*human interface*). Sebagai contoh adalah sebuah *mouse* dapat mendeteksi ketika adanya penekanan tombol ataupun gerakan yang dilakukan oleh manusia sehingga memberikan nilai masukan yang berbeda-beda terhadap komputer untuk melakukan tugasnya.

Namun sebenarnya kelas komunikasi HID tidak memiliki antarmuka manusia yang sebenarnya, perangkat harus diatur fungsinya untuk dapat memenuhi spesifikasi dari kelas komunikasi HID agar dapat bekerja seperti yang seharusnya. Ada beberapa kemampuan dan keterbatasan dari perangkat dengan kelas komunikasi HID diantarnya adalah :

- semua data yang akan diproses berada dalam sebuah struktur yang disebut *reports*. Komputer (*host*) akan mengirim dan menerima data dengan mengirimkan dan meminta *reports* dalam *interrupt transfer* atau *control transfer*. Format dari *report* sangat fleksibel dan dapat ditangani oleh berbagai macam tipe data, namun untuk beberapa *report* harus memiliki besar data yang sudah pasti
- sebuah antarmuka HID harus memiliki sebuah *IN interrupt* untuk dapa mengirimkan permintaan *report* kepada *host*

- sebuah antarmuka HID dapat memiliki setidaknya 1 (satu) buah *endpoint* interupsi *IN* dan 1 (satu) buah *endpoint* interupsi *OUT*
- *Endpoint* interupsi *IN* memperbolehkan HID untuk mengirimkan informasi kepada komputer (*host*) dalam waktu yang tidak dapat diprediksi. Sebagai contoh tidak ada cara lain sebuah komputer untuk mengetahui ketika si pengguna menekan salah satu tombol pada keyboard selain menggunakan transaksi interupsi, sehingga *driver* pada komputer (*host*) menggunakan transaksi interupsi untuk menanyai secara periodik terhadap perangkat apabila terdapat data yang baru yang akan diproses oleh komputer
- Kecepatan transfer data pada perangkat dibatasi, terutama pada USB *low speed* dan USB *full speed*. Sebuah komputer dapat memberikan batasan *endpoint* interupsi untuk kecepatan rendah tidak lebih dari 800 byte/detik, dan untuk kecepatan penuh, maksimal *endpoint* interupsinya adalah 64 kByte/detik. Sedangkan untuk kecepatan tinggi, maksimal *endpoint* interupsinya adalah 25 Mbyte/detik apabila komputer (*host*) mendukung *high-bandwidth* bila tidak maka hanya berkisar di 8 Mbyte/detik.

Jenis komunikasi HID merupakan pengimplementasian antarmuka USB yang sederhana. Beberapa dari peralatan ini cukup mengirimkan data ke komputer dan juga menerima beberapa permintaan data dari komputer untuk konfigurasi dari peralatan tersebut.

2.8 USB – CDC *Communication Class*

USB *Communication Device Class* (USB – CDC) adalah suatu kelas perangkat USB yang dimana didalamnya dapat terdiri dari beberapa kelas komunikasi. Kelas USB-CDC ini dapat terdiri dari *custom control interface*, *data interface*, *audio* atau bahkan *mass storage interface* dalam 1 perangkatnya.

Pada umumnya kelas komunikasi ini digunakan untuk *modem*, sehingga perangkat dapat melakukan beberapa tugas berbeda dalam waktu yang bersamaan. Seperti halnya modem dapat melakukan komunikasi data antara modem dengan PC atau komputer, namun diwaktu yang bersamaan juga dapat digunakan untuk komunikasi panggilan suara. Hal tersebut dapat dilakukan dengan menggunakan kelas komunikasi USB-CDC.

Dalam bentuk lain, komunikasi USB-CDC ini digunakan untuk menggantikan peranti antarmuka RS-232 yang sudah hampir tak pernah ada di perangkat komputer masa kini, sehingga solusi untuk komunikasi tersebut adalah dengan membuat *virtual comm-port* dimana hal tersebut juga memanfaatkan kelas komunikasi USB-CDC karena kelas komunikasi ini dapat dibentuk sesuai dengan kebutuhan pembuatnya, hanya saja untuk dapat berkomunikasi antara perangkat dengan komputer dibutuhkan *driver* khusus yang dapat menangani komunikasi antara perangakat dengan komputer.