BAB II
TEORI DASAR

2.1 Sistem Kontrol

Sistem kontrol adalah proses pengaturan atau pengendalian terhadap satu atau beberapa besaran *(variabel, parameter)* sehingga berada pada suatu harga tertentu. Di dalam dunia industri, dituntut suatu proses kerja yang aman dan berefisiensi tinggi untuk menghasilkan produk dengan kualitas dan kuantitas yang baik serta dengan waktu yang telah ditentukan. Otomatisasi sangat membantu dalam hal kelancaran operasional, keamanan (investasi, lingkungan), ekonomi (biaya produksi) maupun mutu produk.

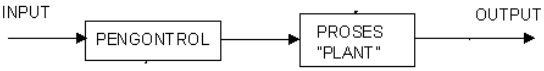
2.2 Sistem Kontrol Otomatis

Sistem kontrol otomatis dalam suatu proses kerja berfungsi mengendalikan proses tanpa adanya campur tangan manusia (otomatis). Ada dua sistem kontrol pada sistem kendali atau kontrol otomatis yaitu :

2.2.1 Open Loop

Open loop merupakan suatu sistem kontrol yang outputnya tidak berpengaruh terhadap aksi pengontrolan. Dengan demikian pada sistem kontrol ini, nilai keluaran tidak diumpan-

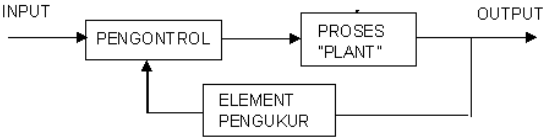
balikkan ke parameter pengendalian. Diagram Blok Sistem Pengendalian Open Loop dapat dilihat pada gambar 2.1.



Gambar 2.1 Diagram Blok Sistem Pengendali *Open Loop*

2.2.2 Close Loop

Close loop merupakan suatu sistem kontrol yang sinyal keluarannya *(output)* memiliki pengaruh langsung terhadap aksi pengendalian yang dilakukan. Sinyal *error* yang merupakan selisih dari sinyal masukan dan sinyal umpan balik *(feedback)*, lalu diumpankan pada komponen pengendalian *(controller)* untuk memperkecil kesalahan sehingga nilai keluaran sistem semakin mendekati harga yang diinginkan. Diagram blok sistem pengendalian *close loop* dapat dilihat pada Gambar 2.2.



Gambar 2.2 Diagram Blok Sistem Pengendali *Close Loop*

2.3 Pengendalian *On-Off*

Dalam suatu pengendali proses dikenal berbagai jenis cara salah satunya adalah proses pengendalian *on-off*. Pada

proses pengendalian jenis ini hanya akan terdapat dua jenis *output* yaitu bersifat *low* dan *high*. Seperti tecerminkan dari namanya, pengendalian *on-off* hanya bekerja pada dua posisi, yaitu posisi "*on*" dan posisi "*off*".

Karena karakteristik kerjanya yang hanya *on* dan *off*, *controller* jenis *on-off* juga sering disebut sebagai *two posision controller*, *gap controller* atau *snap controller*. Kata *snap* secara harfiah berarti menampar. Sebuah *controller on-off* kemudian juga lazim disebut *snap controller*. Ungkapan kata *snap action* kelak akan juga dipakai untuk kerja *controller* jenis lain yang karena besarnya *gain* menjadi bekerja secara *on-off*.

Proses pengendalian *on-off* banyak dipakai di sistem pengendali yang sederhana karena harganya yang relatif murah. Namun, tidak semua proses dapat dikendalikan secara *on-off,* karena banyak operasi proses yang tidak dapat mentolerir fluktuasi proses *variable*. Jadi, syarat utama untuk memakai pengendali *on-off* bukan untuk menghemat biaya unit *controller* melainkan karena proses memang tidak dapat mentolerir fluktuasi proses *variable* pada batas-batas kerja pengendalian *on-off*.

2.4 Teknik Instrumentasi

Instrumentasi merupakan *device* atau peralatan yang digunakan untuk menunjang sebuah sistem dalam menjalankan proses tertentu untuk tujuan tertentu pula. Setiap kegiatan proses dalam sebuah system di industri senantiasa membutuhkan

peralatan–peralatan otomatis untuk mengendalikan parameter–parameter prosesnya. Otomatisasi tidak saja diperlukan demi kelancaran operasi, keamanan, ekonomi, maupun mutu produk, tetapi lebih mengutamakan pada kepentingan penggunaan manusia *(user)* sebagai kontrol manual, kecepatan, kualitas, serta kuantitas yang dihasilkan dibandingkan dengan menggunakan kontrol manual, dalam hal ini manusia sebagai pengendali dan pelaku keputusan.

Secara umum, sistem instrumentasi terdiri dari empat peralatan, yaitu peralatan masukan *(input),* pengkondisi sinyal *(sinyal conditioning)*, sistem pengolah, dan peralatan pencatat. Peralatan masukan (*input*) merupakan peralatan pertama yang menerima besaran yang akan diukur. *Output* yang dihasilkan dari peralatan masukan berupa sinyal-sinyal listrik.

2.5 Mikrokontroller ATMega 8535

Mikrokontroller ATMega 8535 merupakan salah satu IC yang dapat digunakan sebagai media pengolah sinyal analog menjadi sinyal digital. Mikrokontroller ATMega8535 ini memiliki 8 jalur ADC *(Analog to Digital Converter)* dengan resolusi 10 bit. Jalur-jalur ADC tersebut terdapat pada PORTA.0 sampai dengan PORTA.7 (kaki 33 sampai dengan kaki 40). Bentuk dan skematis ATMega8535 dapat dilihat pada gambar 2.3.

Gambar 2.3 Bentuk dan skematis ATMega8535

Selain kaki-kaki tersebut, mikrokontroller ATMega8535 juga memiliki kaki-kaki yang lain yaitu VCC, GND, RESET, XATL1, XATL2, AVCC, dan AREF. Fungsi kaki-kaki pada ATMega8535 tersebut adalah :

1. VCC merupakan positif sumber tegangan,
2. GND merupakan ground sumber tegangan,
3. AVCC**,** merupakan catu daya yang digunakan untuk memasukan analog ADC yang terhubung ke Port A,
4. *RESET,* merupakan pin reset yang akan bekerja bila diberi pulsa rendah (*aktif low*) selama minimal 1.5 us,
5. XTAL2, merupakan input ke penguat osilator pembalik dan input ke internal clock,
6. XTAL1, merupakan output dari penguat osilator pembalik,
7. AREF, merupakan tegangan reeferensi analog untuk ADC.

Untuk melakukan pemrograman dalam mikrokontroller keluarga *AVR*, Atmel telah menyediakan *software* khusus yang dapat diunduh dari *website* resmi Atmel. *Software* tersebut adalah AVRStudio. *Software* ini menggunakan bahasa *assembly* sebagai bahasa perantaranya. Selain AVRStudio, ada beberapa *software* lain yang dapat digunakan untuk membuat program pada mikrokontroller keluarga AVR. *Software* selain AVRStudio ini menggunakan bahasa seperti bahasa C, Java, atau *Basic*. Untuk melakukan pemindahan pemrograman tingkat tinggi dari komputer ke dalam chip, dapat digunakan beberapa cara seperti menggunakan kabel JTAG maupun menggunakan STK buatan Atmel.

2.6 CodeVision AVR

CodeVisionAVR merupakan sebuah *cross-compiler C*, *Integrated Development Environtment (IDE)*, dan *Automatic Program Generator* yang didesain untuk mikrokontroller buatan Atmel seri AVR. *CodeVisionAVR* dapat dijalankan pada system operasi Windows 95, 98, Me, NT4, 2000, XP, Vista dan 7. *Crosscompiler C* mampu menerjemahkan hampir semua perintah dari bahasa *ANSI C*, sejauh yang diijinkan oleh arsitektur dari AVR. *IDE* mempunyai fasilitas internal berupa software *AVR Chip In- System Programmer. AVR Chip In-System Programmer* digunakan untuk melakukan transfer program kedalam chip mikrokontroller. *Software In-System Programmer* didesain untuk

bekerja dengan Atmel *STK500/AVRISP/AVRProg*, *Kanda Systems STK200+/300, Dontronics DT006, Vogel Elektronik VTEC-ISP, Futurlec JRAVR* maupun *MicroTronics ATCPU/ Mega2000 programmers/development boards*. Tampilan *IDE* perangkat lunak *CodeVision AVR* dapat dilihat pada gambar 2.11. *CodeVisionAVR* juga mempunyai *Automatic Program Generator* bernama *CodeWizardAVR*. *CodeWizardAVR* berfungsi untuk menulis semua instruksi yang diperlukan untuk membuat fungsi-fungsi berikut:

- Set-up akses memori eksternal,
- Identifikasi sumber reset untuk chip,
- Inisialisasi port input/output,
- Inisialisasi interupsi eksternal,
- Inisialisasi Timer/Counter,
- Inisialisasi Watchdog-Timer,
- Inisialisasi UART (USART),
- Inisialisasi Pembanding Analog,
- Inisialisasi ADC,
- Inisialisasi Antarmuka SPI Inisialisasi Antarmuka Two-Wire,
- Inisialisasi Antarmuka CAN,
- Inisialisasi Bus I2C, Sensor Suhu LM75, Thermometer atau Thermostat DS1621, Real-Time Clock PCF8563, PCF8583, DS1302, dan DS1307,
- Inisialisasi Bus 1-Wire dan Sensor Suhu DS1820, DS18S20,
- Inisialisasi modul LCD.

2.7 Visual Basic

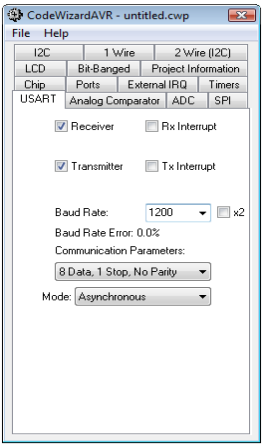
Visual basic adalah salah satu alat bantu untuk mengembangkan perangkat lunak berbasis Microsoft Windows secara mudah dan cepat (*Rapid Development, RAD*). Kata "*Visual*" merujuk pada cara Visual Basic dalam membuat tampilan antar muka (*graphical user interface, gui*) secara *visual*. Kata "*Basic*" merujuk kepada bahasa pemrograman BASIC (*Beginners All-Purpose Symbolic instruction Code).* Program aplikasi Visual Basic tersusun atas objek-objek, seperti *button, list box, text box, form* dan sebagainya. Tiap-tiap objek ini mempunyai *properties, events*, dan *methods.*

Properties adalah atribut suatu objek yang menggambarkan sifat-sifat objek tersebut, seperti tinggi, lebar, warna latar belakang, nama dan sebagainya. *Events* adalah aktivitas yang dikenali oleh objek, seperti penekanan tombol *mouse* kiri dua kali, penekanan *mouse* kanan, penekanan *botton* pada *worksheet,* dan sebagainya. *Event*s dapat terjadi atau dibangkitkan oleh pemakai program, kode program atau sistem. *Method* adalah rutin yang bekerja pada objek.

Dalam teknik pengukuran, visual basic sangat membantu dan mempermudah pekerjaan. Nilai suatu pengukuran kadang-kadang perlu ditampilkan pada monitor komputer. Agar hal ini dapat dilakukan, diperlukan suatu komunikasi antara mikrokontroller dengan komputer. Salah satu jenis komunikasi yang sering digunakan antara mikrokontroller dengan komputer

adalah komunikasi serial secara asinkron. Untuk memulai komunikasi serial, terlebih dahulu harus dilakukan pembuatan program pada mikrokontroller dan komputer. Program yang dimasukkan ke mikrokontroller dibuat dengan menggunakan CodevisionAVR. Program yang ditulis di komputer dibuat dengan menggunakan visual basic. Hal pertama yang harus dilakukan untuk membuka jalur komunikasi serial adalah *set-up* komunikasi serial. *Set-up* komunikasi serial dilakukan pada CodevisionAVR dan Visual Basic.

Set-up pada CodevisionAVR dilakukan dengan me-klik Tab USART, kemudian *chek box Receiver* dan *chek box Transmiter* di-*cheklist.* Kemudian *baudrate* diset pada suatu harga tertentu. *Set-up* komunikasi serial dapat dilihat pada gambar 2.4.



Gambar 2.4 *Set-up* komunikasi serial

Contoh program utama untuk komunikasi serial pada codevisionAVR dapat dilihat pada tabel 2.1. Pada baris pertama program utama nilai variabel i ditambah satu. Pada baris berikutnya nilai variabel i diperiksa. Bila nilai variabel i lebih dari 9, maka nilai variabel i diubah menjadi nol.

Tabel 2.1 Program utama komunikasi serial

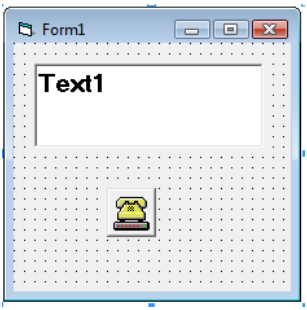
```
while (1)
      {
      // Place your code here

          i++;
          if (i>9) i=0;
          putchar (i+48);
          delay_us(5);
          putchar ('A');
          delay_ms(200);

      };
```

Pada baris ketiga program utama, nilai variabel i dikirim ke komputer dengan menggunakan perintah putchar(i+48). Nilai variabel i perlu ditambah dengan 48 karena karakter ASCII ke 48 sampai dengan karakter ASCII ke 57 berupa karakter 0 sampai dengan karakter 9. Setelah karakter i dikirimkan ke komputer, proses ditahan selama 5 mikrosekon. Hal ini dilakukan supaya komputer sempat melakukan aktivitas yang lain sebelum komputer menerima data yang lain.

Pada baris berikutnya mikrokontroller mengirim karakter A ke komputer. Pengiriman karakter ini dimaksudkan sebagai tanda bagi komputer bahwa data berikutnya yang diterima adalah data angka yang baru. Setelah mikrokontroller mengirimkan karakter A,

proses ditahan selama 250 milidetik. Hal ini dilakukan supaya angka yang nantinya ditampilkan di komputer dapat diamati perubahannya. Supaya komputer dapat menerima data yang dikirimkan oleh mikrokontroller, perlu disiapkan program untuk menerima data tersebut. Program yang digunakan untuk menerima data dari mikrokontroller ditulis dengan menggunakan bahasa pemrograman Visual Basic 6.0.

Form aplikasi program untuk menerima data dari mikrokontroller dapat dilihat pada gambar 2.5. Pada *form* tersebut terdapat objek *Text Box* dan objek *MSComm* (dilambangkan dengan gambar telepon). Objek *Text Box* digunakan untuk menampilkan data yang diterima oleh komputer. Objek *MSComm* digunakan untuk komunikasi secara serial.



Gambar 2.5 *Form load* aplikasi komunikasi serial

Supaya objek *MSComm* dapat digunakan untuk komunikasi serial, beberapa properti objek tersebut harus di-set

ulang sesuai dengan parameter komunikasi serial. Beberapa properti yang harus di-set ulang dapat dilihat pada tabel 2.2.

Tabel 2.2 Properti objek MSComm yang harus di-set

Properti	Harga
Name	MSComm 1
RThreshold	1
SThreshold	1
Settings	1200,n,8,1

Selain me-set ulang beberapa properti yang ada pada tabel 2.2, jalur komunikasi serial harus dibuka terlebih dahulu. Jalur komunikasi serial dibuka dengan cara me-set properti *PortOpen* dengan nilai *True*. *Setting* properti *PortOpen* biasanya dilakukan di prosedur *Form_load().* Objek *MSComm* mempunyai *event Oncomm. Event Oncomm* akan dipicu jika komputer telah selesai mengirim atau menerima data secara lengkap. Bila *event Oncomm* dipicu karena komputer telah selesai mengirim data, maka properti *comEvent* akan sama dengan *comeEvsend*. Bila *event OnComm* dipicu karena komputer telah selesai menerima data, maka *ComEvent* akan sama dengan *ComEvReceive.*

Pada prosedur yang dilayani event OnComm dituliskan beberapa baris perintah yang digunakan untuk menangani penerimaan data secara serial. *List* program secara lengkap untuk

menangani penerimaan data secara serial dapat dilihat pada tabel 2.3.

Tabel 2.3
***List* program komunikasi serial dalam bahasa Visual Basic 6**

```
Dim kata As string

Private Sub Form_Load()
        MSComm1.PortOpen = True
End Sub

Private Sub Form_Unload(Cancel As Integer)
        MSComm1.PortOpen = False
End Sub

Private Sub MSComm1_OnComm()
        If (MSComm1.CommEvent = comEvReceive) Then
                buffer = MSComm1.Input
                kata = kata & buffer
                angka = InStr(kata, "A")
                If (angka <> 0) Then
                    Text1.Text = kata
                    kata = ""
                End If
        End If
End Sub
```

Pada baris pertama *list* program pada tabel 2.3, terdapat sintaks yang bertujuan untuk mendeklarasikan suatu variabel dengan nama kata bertipe string. Variabel kata tersebut perlu dideklarasikan sebagai variabel global karena variabel ini yang akan digunakan untuk menampung semua data yang telah diterima oleh komputer. Tiga baris program berikutnya merupakan prosedur *Form_Load()*. Prosedur ini akan dikerjakan oleh program aplikasi ketika aplikasi akan ditutup. Prosedur ini memuat perintah untuk menutup jalur komunikasi secara serial. Baris-baris

berikutnya merupakan prosedur *MSComm1.Oncomm().* Prosedur ini memuat beberapa perintah yang digunakan untuk menangani penerimaan data secara serial. Pada baris pertama prosedur tersebut properti *ComEvent* diperiksa. Bila properti *ComEvent* sama dengan *ComEvReceive,* maka prosedur *MSComm1.Oncomm()* diaktivasi karena komputer selesai menerima data. Jika prosedur *MSComm1.OnComm()* diaktivasi karena komputer selesai menerima data, maka semua perintah sebelum *end if* akan dikerjakan.

Data yang diterima oleh komputer disimpan didalam properti *input*. Data tersebut, untuk memudahkan pengolahan, dipindahkan ke variabel *buffer (buffer=MSComm1.Input).* data yang sudah disimpan di variabel *buffer* digabungkan dengan data yang telah diterima sebelumnya *(kata=kata & buffer).* Setelah data yang baru digabungkan dengan data yang lama, data tersebut diperiksa apakah dalam data yang telah digabung terdapat karakter A atau tidak (angka=InStr(kata,"A"). Jika didalam variabel kata terdapat karakter A, maka nilai angka tidak sama dengan nol. Jika nilai angka tidak sama dengan nol, maka variabel kata ditampilkan melalui *text box (Text1.text*=kata*)*. Setelah variabel kata ditampilkan melalui *text box*, variabel kata dikosongkan *(kata="").*

2.8 Ultrasonik

Ultrasonik, sebutan untuk jenis suara diatas batas suara yang bisa didengar manusia. Seperti diketahui, telinga manusia hanya bisa mendengar suara dengan frekuensi 20 Hz sampai 20 KHz. Lebih dari itu hanya beberapa jenis binatang yang mampu mendengarnya, seperti kelelawar dan lumba-lumba. Lumba-lumba bahkan memanfaatkan ultrasonik untuk mengindera benda-benda di laut. Prinsip ini kemudian ditiru oleh sistem pengindera kapal selam. Dengan cara mengirimkan sebuah suara dan mengitung lamanya pantulan suara tersebut maka dapat diketahui jarak kapal selam dengan benda tersebut. Mula-mula suara dibunyikan, kemudian dihitung lama waktu sampai terdengar suara pantulan. Jarak dapat dihitung dengan mengalikan kecepatan suara dengan waktu pantulan. Kemudian hasilnya dibagi 2. Misalnya lama waktu pantulan adalah 1 detik, maka jaraknya adalah (344,424m/detik x 1 detik)/2 = 172 m.

Sensor ultrasonik terdiri dari dua unit, yaitu unit pemancar dan unit penerima. Struktur unit pemancar dan penerima sangatlah sederhana, sebuah kristal *piezoelectric* dihubungkan dengan mekanik jangkar dan hanya dihubungkan dengan diafragma penggetar. Tegangan bolak-balik yang memiliki frekuensi kerja 40 KHz – 400 KHz diberikan pada plat logam. Struktur atom dari kristal *piezoelectric* akan berkontraksi (mengikat), mengembang atau menyusut terhadap polaritas tegangan yang diberikan, dan ini disebut dengan efek *piezoelectric*. Kontraksi yang terjadi diteruskan ke diafragma

penggetar sehingga pantulan gelombang ultrasonik akan terjadi bila ada objek tertentu, dan pantulan gelombang ultrasonik akan diterima kembali oleh unit sensor penerima. Selanjutnya unit sensor penerima akan menyebabkan diafragma penggetar bergetar dan efek *piezoelectric* menghasilkan sebuah tegangan bolak-balik dengan frekuensi yang sama. *Ping))) Paralax Ultrasonic Range Finder* dapat dilihat pada gambar 2.6.

Gambar 2.6 *Ping))) Paralax Ultrasonic Range Finder*

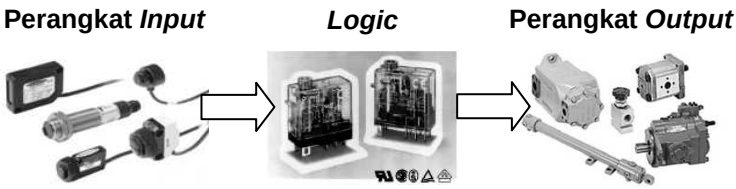
Besar amplitudo sinyal elekrik yang dihasilkan unit sensor penerima tergantung dari jauh dekatnya objek yang dideteksi serta kualitas dari sensor pemancar dan sensor penerima. Proses *sensing* yuang dilakukan pada sensor ini menggunakan metode pantulan untuk menghitung jarak antara sensor dengan obyek sasaran. Jarak antara sensor tersebut dihitung dengan cara mengalikan setengah waktu yang digunakan oleh sinyal ultrasonik dalam perjalanannya dari rangkaian Tx sampai diterima oleh rangkaian Rx, dengan kecepatan rambat dari sinyal ultrasonik tersebut pada media rambat yang digunakannya, yaitu udara. Waktu di hitung ketika pemencar aktif dan sampai ada input dari

rangkaian penerima dan bila pada melebihi batas waktu tertentu rangkaian penerima tidak ada sinyal input maka dianggap tidak ada halangan didepannya.

2.9 Relay

Relay ladder logic merupakan alur berfikir untuk menyusun rangkaian sistem kontrol. Alur berfikir tersebut dinyatakan dalam diagram tangga. *Relay ladder logic* disebut juga sebagai bahasa pemrograman untuk rangkaian sistem kontrol. *Relay ladder logic* terbagi menjadi tiga bagian utama, yaitu:

1. Pemberi informasi (*input*),
2. Pengambil keputusan (*logic*), dan
3. Usaha yang dilakukan (*output*).



Gambar 2.7 Sistem kontrol dengan menggunakan *relay*

Sistem kontrol berbasis *relay ladder logic* yang menggambarkan penjelasan di atas dapat dilihat pada gambar 2.7. Dari gambar tersebut dapat dilihat bahwa sistem kontrol dengan menggunakan *relay* mempunyai beberapa perangkat

input, dan berbagai macam perangkat *output.* Perangkat *input* di antaranya sensor sedangkan perangkat *output* dari sistem kontrol dapat berupa pompa, lampu, katup *solenoid* atau perangkat elektronik yang lainnya.

2.10 Pompa

Pompa adalah suatu alat atau mesin yang digunakan untuk memindahkan cairan dari suatu tempat ke tempat yang lain melalui suatu media perpipaan dengan cara menambahkan energi pada cairan yang dipindahkan dan berlangsung secara terus menerus. Pompa beroperasi dengan prinsip membuat perbedaan tekanan antara bagian masuk *(suction)* dengan bagian keluar *(discharge).* Dengan kata lain, pompa berfungsi mengubah tenaga mekanik dari suatu sumber tenaga (penggerak) menjadi tenaga kinetik (kecepatan), dimana tenaga ini berguna untuk mengalirkan cairan dan mengatasi hambatan yang ada sepanjang pengaliran.

Gambar 2.8 Pompa

Pada sistem pengendali ketinggian air, pompa berfungsi untuk memindahkan air di dalam suatu wadah penampung kedalam wadah penampung utama. Pompa yang digunakan pada aplikasi ini dilihat pada gambar 2.8.

Head (H) sebuah pompa adalah pemanfaatan energi mekanik yang dihasilkan pompa dalam menangani fluida yang mengalami hambatan seperti ketinggian, gesekan laju air dan tekanan. *Head* terbagi menjadi 3 antara lain :

1. *Head* statik

   Head statik adalah *head* yang terjadi penjumlahan *head elevasi* dan *head* tekanan.

2. *Head* kecepatan

   Head kecepatan adalah *head* yang terjadi akibat dari perbedaan kecepatan pada fluida.

3. *Head loss*

   Head kerugian *(loss)* yaitu *head* untuk mengatasi kerugian kerugian yang terdiri dari kerugian gesek aliran di dalam perpipaan, *head* kerugian di dalam belokan-belokan *(elbow),* percabangan, dan perkatupan *(valve).*