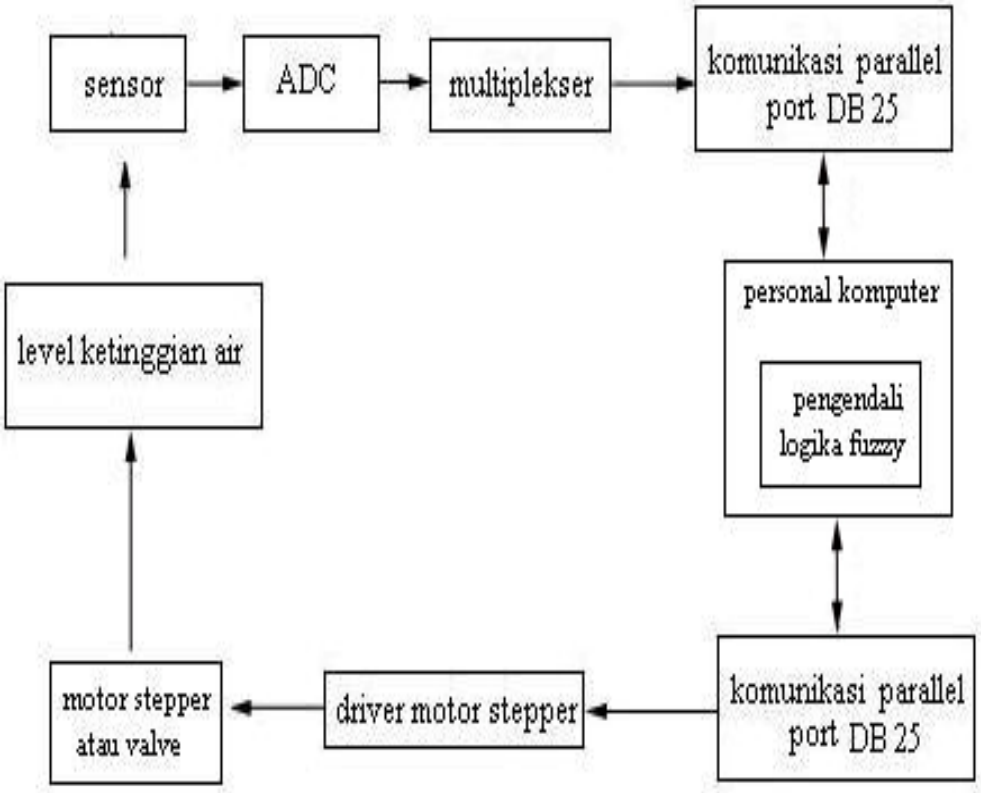
BAB III

PERANCANGAN SISTEM

3.1 Gambaran Umum

Sebelum membahas tentang perancangan perangkat keras dan perangkat lunak yang akan dirancang, maka dapat dilihat Gambaran umum dari sistem kontrol dan monitoring dari level ketinggian air, Gambar 3.1 merupakan blok diagram sistem secara keseluruhan.



Gambar 3.1 *Blok Diagram Sistem Secara Keseluruhan*

Dari Gambar 3.1, dapat dijelaskan urutan kerja sistem, yaitu :

1. Sistem kontrol

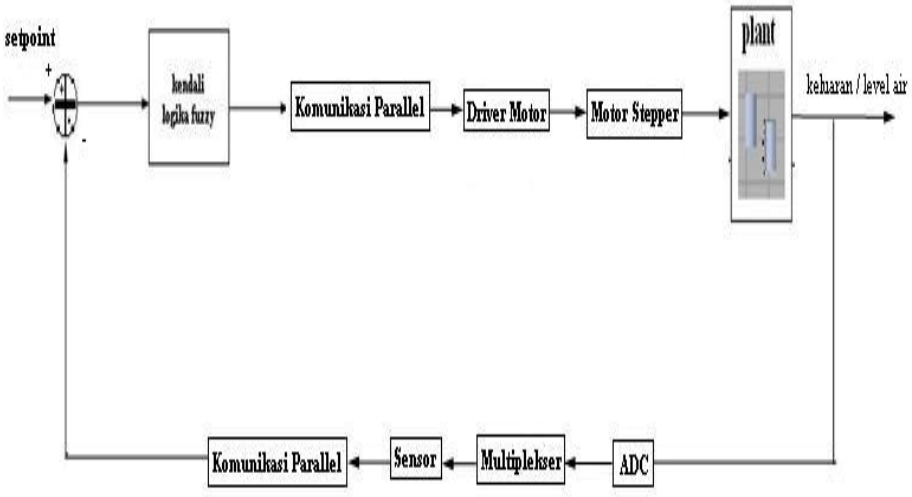
   - Sensor potensiometer menangkap informasi level ketinggian air berdasarkan perubahan tegangan dan kemudian mengirimkan data analog ke *analog to digital converter* (ADC)
   - Jika sudah terbaca ke ADC, kemudian data analog dari sensor diubah menjadi data dalam bentuk digital atau biner yang jumlah maksimal bitnya adalah 8 bit dan sebagai indikator bitnya digunakan led sebanyak 8 buah, kemudian output dari ADC akan diproses oleh rangkaian multiplekser.
   - Pada rangkaian multiplekser data dari ADC diproses dan kemudian dikeluarkan dalam satu saluran, data ini kemudian dikirim ke komputer melalui komunikasi paralel yaitu menggunakan *connector* DB 25 pada komputer.
   - Data yang telah masuk ke komputer, kemudian diolah dan diproses menggunakan *software* LabVIEW untuk mengimplementasikan kendali logika *fuzzy*, setelah itu data yang telah diproses kemudian dikeluarkan kembali pada connector DB 25, yang kemudian digunakan untuk mengendalikan motor steper untuk mengendalikan sistem buka dan tutup keran sehingga level ketinggian air dapat dikendalikan.

3.2 Perancangan Sistem

Pada bab ini dilakukan perancangan perangkat lunak dan perangkat keras yang diperlukan untuk merealisasikan proyek tugas akhir yang akan direalisasikan, yaitu pengendalian level ketinggian air dengan menggunakan kendali logika *fuzzy*. Dibawah ini adalah Gambaran umum tentang perancangan sistem yang akan dilakukan.

1. Perancangan perangkat keras yang mendukung untuk pengendalian level ketinggian air.
2. Perancangan perangkat lunak dengan menggunakan *software* labView sebagai implementasi dari kendali logika *fuzzy.*

Gambar 3.2 menunjukkan diagram blok pengendalian sistem.



Gambar 3.2 *Diagram Blok Pengendalian Sistem*

Ket :

keluaran = nilai besaran yang akan dikontrol yaitu level air

3.3 Perancangan perangkat keras (*hardware*)

Dalam perancangan perangkat keras ini akan dibuat dan digunakan beberapa perangkat keras yang mendukung untuk pengendalian level ketinggian air, yaitu meliputi :

a. sensor level ketinggian yang menggunakan potensiometer
b. rangkaian ADC 0804 digunakan untuk mengubah data analog dari sensor menjadi data digital
c. multiplekser digunakan untuk mengubah banyak data atau sinyal menjadi satu data yang akan dikeluarkan pada satu output
d. rangkaian driver motor stepper
e. penggunaan connector DB 25 sebagai interface antara perangkat keras dengan komputer
f. rangkaian catu daya sebagai supply tegangan

3.3.1 Potensiometer

Variabel resistor atau potensiometer adalah resistor yang nilai tahanannya bisa diubah-ubah sesuai dengan nilai yang di kehendaki, pada tugas akhir ini besarnya hambatan potensiometer yang digunakan yaitu maksimum 50 kΩ, sesuai dengan persamaan (*3.1*).

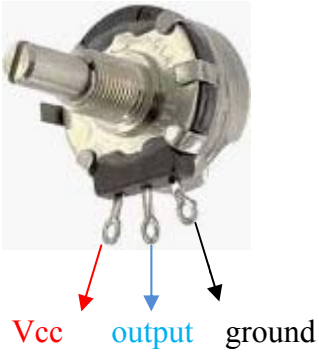
$V(output) = \frac{R1}{R2}\ V\ (input)$ ……...…….(*3.1*)

Ket :

R1 = perubahan hambatan.

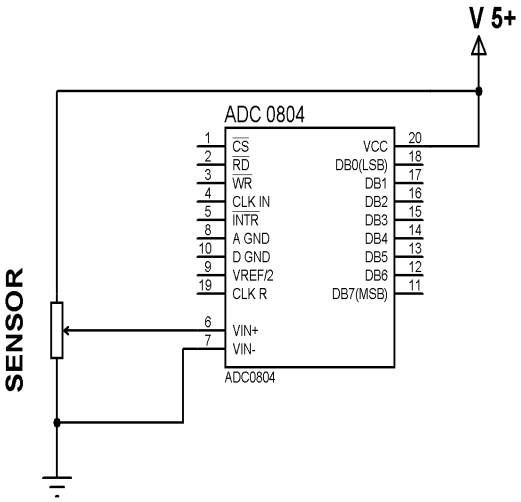
R2 = hambatan maksimum.

Dari persamaan (*3.1*) dapat diamati bahwa Bahwa tegangan *V (output)* berbanding lurus dengan Perubahan hambatan, perubahan tegangan V (*output*) inilah yang nantinya akan digunakan sebagai *indikator* perubahan level ketinggian air. Atas dasar tersebutlah maka digunakanlah potensiometer sebagai sensor pendeteksi level ketinggian air, berikut adalah Gambar 3.3 potensio meter sebagai sensor ketinggian air.



Gambar 3.3 *Potensiometer*

Gambar 3.4 menunjukkan rangkaian sensor level ketinggian air.



Gambar 3.4 *Rangkaian Sensor*

3.3.2 *Analog To Digital Converter* (ADC)

IC ADC 0804 mempunyai dua masukan analog, Vin (+) dan Vin (-), sehingga dapat menerima masukan diferensial. Masukan analog sebenarnya (Vin) sama dengan selisih antara tegangan-tegangan yang dihubungkan dengan ke dua pin masukan yaitu Vin= Vin (+) – Vin (-). Kalau masukan analog berupa tegangan tunggal, tegangan ini harus dihubungkan dengan Vin (+), sedangkan Vin (-) digroundkan. Untuk operasi normal, ADC 0804 menggunakan Vcc = +5 Volt sebagai tegangan referensi. Dalam hal ini jangkauan masukan analog mulai dari 0 Volt sampai 5 Volt (skala penuh), karena IC ini adalah SAC 8-bit, resolusinya akan sama dengan.

$$\text{Resolusi} = \frac{tegangan\ skala\ penuh}{2^n - 1} = \frac{5\ Volt}{255} = 19.6\ \text{mVolt} \quad \ldots\ldots(3.2)$$

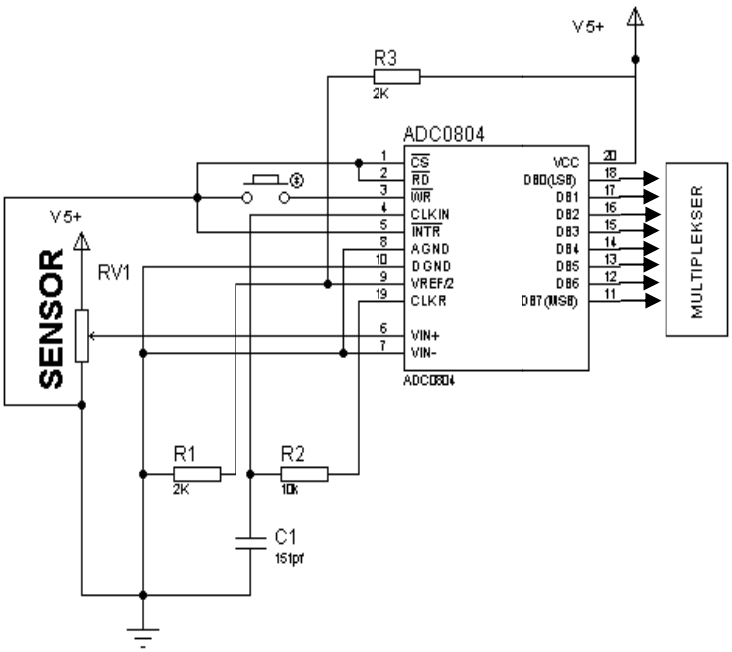
(*n menyatakan jumlah bit keluaran biner IC analog to digital converter*)

IC ADC 0804 memiliki generator *clock intenal* yang harus diaktifkan dengan menghubungkan sebuah resistor eksternal (R) antara pin CLK OUT dan CLK IN serta sebuah kapasitor eksternal (C) antara CLK IN dan ground digital. Frekuensi *clock* yang diperoleh di pin CLK OUT sama dengan :

$$f = \frac{0.91}{RC} \quad \ldots\ldots(3.3)$$

Untuk sinyal clock ini dapat juga digunakan sinyal eksternal yang dihubungkan ke pin CLK IN. ADC 0804 memiliki 8 keluaran digital sehingga dapat langsung dihubungkan dengan saluran data mikrokomputer. Masukan (chip select, aktif rendah) digunakan untuk mengaktifkan ADC 0804. Jika berlogika tinggi, ADC

0804 tidak aktif (*disable*) dan semua keluaran berada dalam keadaan impedansi tinggi. Masukan (write atau start convertion) digunakan untuk memulai proses konversi. Untuk itu harus diberi pulsa logika 0. Sedangkan keluaran (interrupt atau end of convertion) menyatakan akhir konversi. Pada saat dimulai konversi, akan berubah ke logika 1 dan di akhir konversi akan kembali ke logika 0. Dibawah ini adalah Gambar 3.5 rangkaian ADC dengan sensor.



Gambar 3.5 *Rangkaian ADC Dengan Sensor*

Nama dan fungsi dari kaki-kaki (pin) pada ADC 0804 adalah sebagai berikut :

1. chip select (CS)

   Pin ini berfungsi sebagai pin untuk mengaktifkan atau menonaktifkan ADC Masukan dari pin ini diatur oleh mikrokontroler.

2. Read (RD)

   Berfungsi membaca data analog dari sensor ketika pin ini digroundkan.

3. Write (WR)

   Berfungsi sebagai pin eksekusi penulisan perintah ketika dilakukan interupsi.

4. CLK

   Berfungsi sebagai pembangkit sinyal osilator.

5. INTR

   Pin ini dihubungkan ke mikrokontroler apabila dilakukan interupsi.

6. Vin (+)

   Sebagai pin masuan analog yang diperoleh dari sensor.

7. Vin (-)

   Sebagai ground dari Vin (+).

8. A GND

   Berfungsi sebagai pin ground data analog.

9. Vref/2

   Pin ini digunakan untuk proses adjustment, Out put dari ADC. Pin ini berfungsi juga sebagai tegangan cadangan pada ADC. Pin ini dihubungkan dengan ground apabila mode konversi yang digunakan adalah mode konversi yang digunakan adalah mode konversi secara terus menerus (Free Running).

10. D GND

    Sebagai pin ground digital

11. CLK

    Sebagai pembangkit sinyal osilator

12. D0 – D7

    Sebagai pin out atau address data bus dari ADC .

3.3.3 Multiplekser

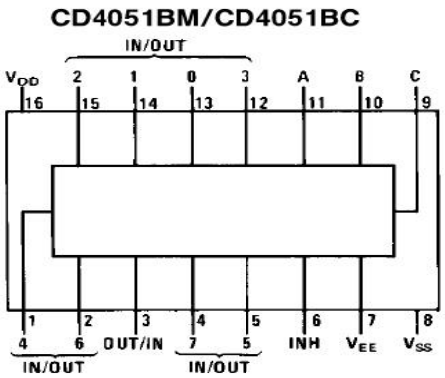
Multiplekser adalah suatu sirkuit yang berfungsi menggabungkan beberapa atau banyak sinyal elektrik menjadi satu sinyal tunggal. Biasanya input multiplekser berupa data yang terdiri dari 8 bit. Input-input tersebut akan diseleksi urutan keluarannya oleh suatu pengontrol skema dari IC multipleser CD4051 Ditunjukkan pada Gambar 3.6.

Input pada multiplekser merupakan output dari ADC yang kemudian diberikan pada kaki 13, 14, 15, 12, 1, 5, 2 dan 4, hal ini ditunjukkan pada Gambar 3.4 rangkaian multiplekser. Output akan diperoleh dari kaki 3 dan sinyal yang dikeluarkan pada output ditentukan oleh input A, B, C pada kaki 11, 10 dan 9. Proses pengontrolan output tersebut sesuai Tabel 3.1 *Pengontrol Output Multiplekser*

Tabel 3.1 *Pengontrol Output Multiplekser*

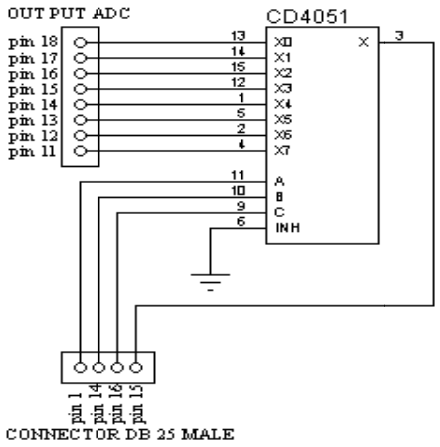
INPUT STATES				**"ON" Channels**
INHIBIT	**C**	**B**	**A**	
0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6
0	1	1	1	7
1	*	*	*	None

Jika pada input C = 0, B = 0, A = 0 maka output pin 3 akan bernilai sama dengan input pin 13. Jika C = 0, B = 0 dan A = 1 maka output pin 3 akan bernilai sama dengan input pin 15. Dan demikian seterusnya.



Gambar 3.6 *Skema IC Multiplekser CD4051*

Di bawah ini adalah Gambar 3.7 Rangkaian Multiplekser.

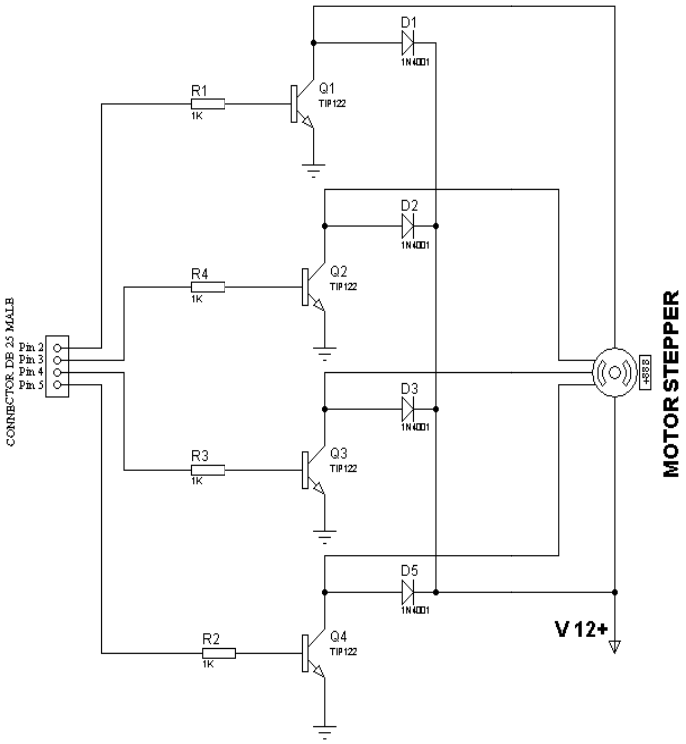


Gambar 3.7 *Rangkaian Multiplekser*

3.3.4 Perancangan Driver Motor Stepper

Pada proyek tugas akhir ini jenis motor stepper yang digunakan yaitu jenis motor stepper unipolar, dikarenakan jenis motor stepper ini mudah dalam pengontrolannya dan memiliki rangkaian driver yang sederhana dibandingkan

dengan jenis motor stepper bipolar. Pada proyek tugas akhir ini, untuk menggerakkan motor stepper digunakan port DB 25 pada komputer, pada kenyataannya arus yang berasal dari komputer kecil yaitu berkisar 2.6 mA dan untuk mengatasi arus induksi yang berasal dari motor stepper, maka digunakan suatu rangkaian driver yang mampu untuk mengatasi permasalahan tersebut, maka pada proyek tugas akhir ini digunakan rangkaian darlington dengan menggunakan transistor TIP 122 serta diode 1N4001 yang berfungsi untuk mengatasi masalah arus induksi yang berasal dari lilitan motor stepper, Gambar 3.8 menunjukkan rangkaian driver motor stepper.



Gambar 3.8 *Rangkaian Driver Motor Stepper*

3.3.5 Komunikasi Data parallel

Pada komputer dilengkapi dengan parallel port, pada dasarnya selain digunakan untuk koneksi ke printer, juga bisa digunakan sebagai input output sistem. *Input* dan *output* tersebut dapat bekerja apabila diprogram. Pada keadaan normal (tidak aktif) tegangan pada pin-pin ini adalah 0 volt, namun bila kita beri *high*, maka tegangannya akan berubah menjadi 5 volt**.**

Port parallel atau port printer terdiri dari tiga bagian yaitu :

1. Data port

   Data port digunakan untuk mengirimkan data ke sistem, pada proyek tugas akhir ini data port digunakan untuk mengirimkan data-data yang nantinya akan mengontrol putaran motor stepper.

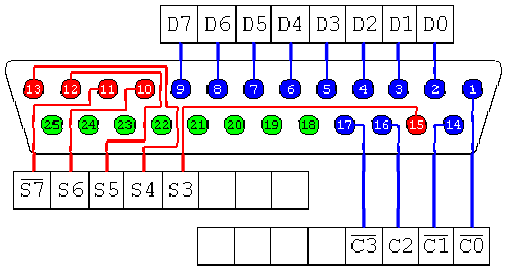
2. Port *control*

   Port *control* berfungsi untuk mengirimkan kode-kode kontrol dari komputer ke sistem yang akan dikontrol .

3. Port status

   Port status berfungsi untuk mengirimkan kode-kode dari sistem ke komputer.

Di bawah ini adalah Gambar 3.9 konfigurasi pin DB 25 *male*.



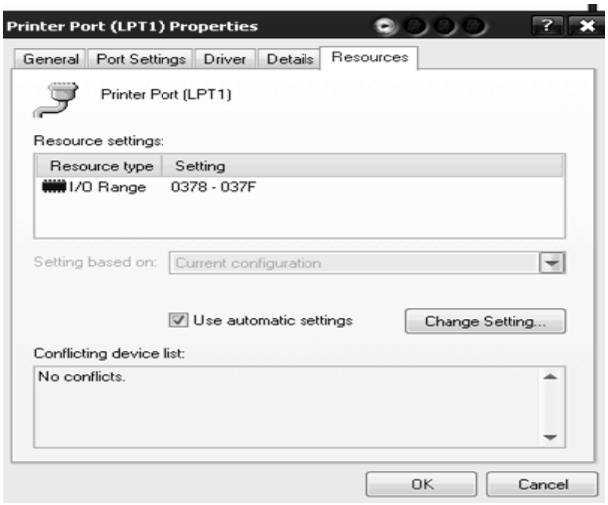
Gambar 3.9 *konfigurasi pin DB 25 Male*

Data line, *control line*, dan *status line* terhubungkan dengan suatu alamat pada komputer, maka dengan melakukan pengalamatn yang tepat, data dapat dengan mudah dikirim dan dibaca melalui parallel port. Untuk mengidentifikasi base address dari parallel port, dapat diketahui dari *windows devices manager*.

Untuk windows 98 :

1. Pada desktop, klik kanan pada mykomputer dan pilih propertis
2. Klik pada tab *device manager* dan cari LPT1
3. Setelah memilih LPT1, klik tombol propertis
4. Lalu pilih tab *resources* dan address selanjutnya akan terlihat pada *Input* atau *Output Range*

Di bawah ini adalah Gambar 3.10 Propertis LPT1 pada windows *devices manager*



Gambar 3.10 *Propertis LPT1 Pada Windows Devices Manager*

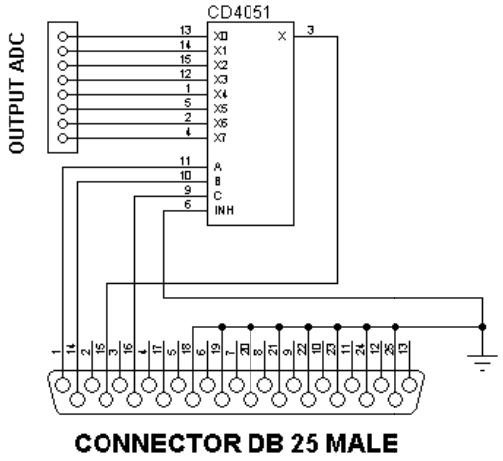
Base address biasanya dalam bentuk hexadesimal, seperti 278, 378, atau 3BC untuk kasus seperti Gambar 3.10, maka dapat disimpulkan pada Tabel 3.2 *Base Address.*

Tabel 3.2 *Base Address*

Register	LPT1
Data register (*base address* + 0)	0*378
Status register (*base address* + 1)	0*379
Control register (*base address* + 2)	0*37a

Untuk mengkomunikasikan sistem dengan computer, maka perlu dirancang beberapa rangkaian elektronika dengan menggabungkan port parallel DB 25 sebagai *interface* nya, diantaranya:

1. Rangkaian multiplekser dengan port parallel DB 25, seperti terlihat pada Gambar 3.11 rangkaian multiplekser dengan port parallel DB 25.

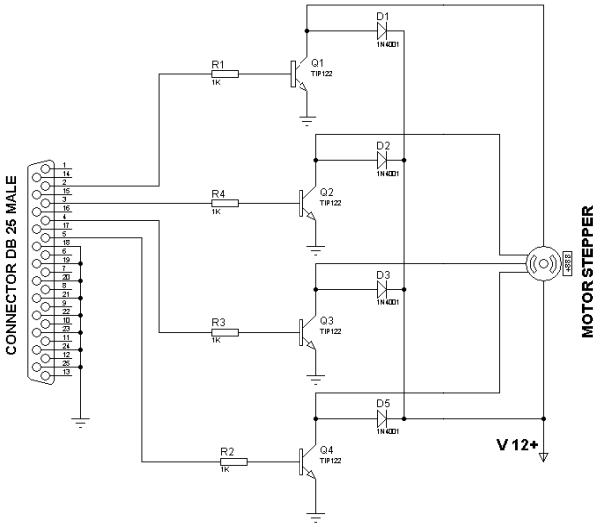


Gambar 3.11 *Rangkaian Multiplekser Dengan Port* Parallel *DB 25*

Dari Gambar diatas dapat dijelaskan cara kerja rangkaian, yaitu :

- Data output ADC masuk sebagai input pada pin 13, 14, 15, 12, 1, 5, 2, 4, pada CD4051, kemudian data ini dikontrol oleh port 11, 10, 9 sebagai pengaktif kemudian data tersebut dikeluarkan pada satu out put, yaitu pin 3.

2. Rangakaian driver motor stepper dengan port parallel DB 25, seperti terlihat pada Gambar 3.12.



Gamabar 3.12 *Rangkaian Driver Motor Dengan Port* Parallel *DB 25*

Dari Gambar 3.12 dapat dijelaskan cara kerja rangkaian, yaitu :

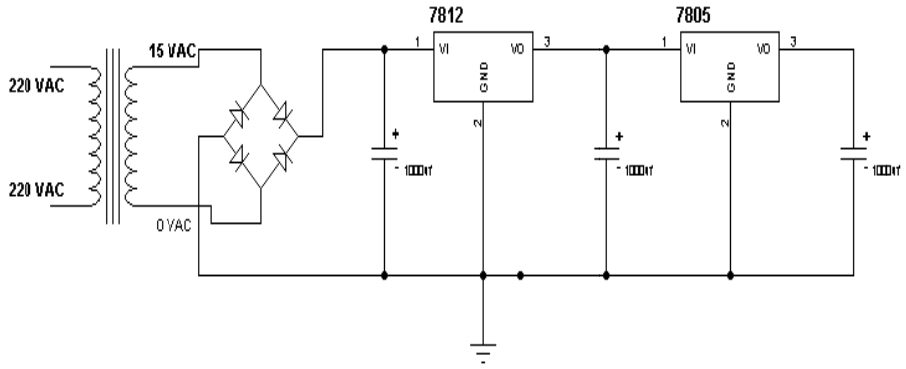
- Data yang berfungsi untuk mengontrol motor stepper berasal dari pin 2, 3, 4, 5 pada port parallel DB 25, data ini kemudian dimasukkan ke kaki-kaki TIP 122, TIP 122 berfungsi sebagai penguatan arus, kemudian diode 1N4001 berfungsi untuk menjaga arus balik dari arus induksi yang berasal dari kumparan-kumparan motor stepper.

3.3.6 Rangkaian Catu Daya

Rangkaian catu daya memberikan tegangan pada alat pengedali, rangkaian catu daya memanfaatkan tegangan dari PLN sebesar 220 VAC, tegangan 220 VAC ini kemudian diturunkan oleh trafo penurun tegangan sehingga menjadi 15 VAC, tegangan 15 VAC kemudian disearahkan oleh diode bridge menjadi tegangan DC,

keluaran dari diode bridge kemudian masuk ke IC regulator yang fungsinya adalah untuk menstabilkan tegangan, pada proyek tugas akhir ini IC regulator yang digunakan IC L7805 dan L78012 yang menghasilkan tegangan 5 VDC dan 12 VDC. Karena tegangan yang diperlukan pada setiap rangkaian tidak sama, rangkaian catu daya catu daya ini mempunyai dua buah keluaran yaitu 5 VDC dan 12 VDC yang berfungsi untuk memberikan pasokan tegangan pada setiap rangkaian. Kapasitor 2200 µf berfungsi untuk membuang noise pada tegangan DC.

Tegangan 5 VDC digunakan sebagai tegangan masukan pada rangkaian ADC dan multiplekser sedangkan 12 VDC digunakan sebagai tegangan masukan pada rangkaian driver motor. Gambar rangkaian catu daya ditunjukkan pada Gambar 3.13 di bawah ini.



Gambar 3.13 *Rangkaian Catu Daya*

3.4 Perancangan Perangkat Lunak (*software*)

Pada perancangan perangkat lunak ini akan dirancang suatu pengendali logika *fuzzy* dalam perangkat lunak (*software*) LabVIEW untuk mengendalikan level

ketinggian air, oleh sebab itu dilakukanlah perancangan perangkat lunak yang meliputi :

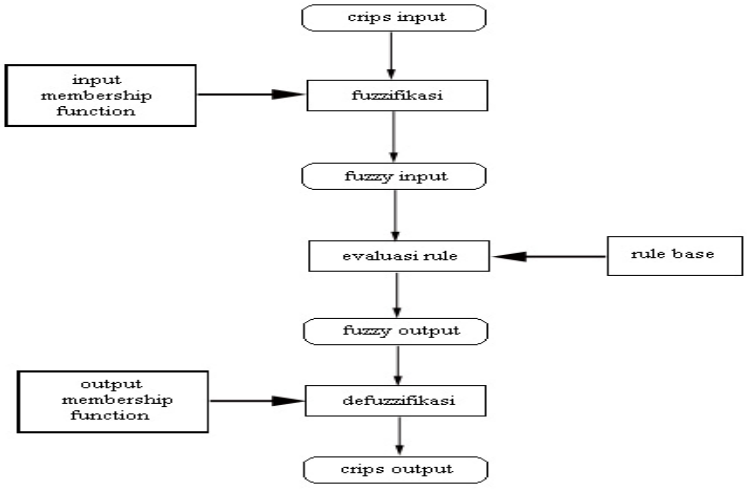
a. Perancangan kendali logika *fuzzy*

b. Perancangan program dalam perangkat lunak (*software*) LabVIEW

3.4.1 Perancangan Kendali Logika *Fuzzy*

Pada proyek tugas akhir ini, perancangan kendali logika *fuzzy* digunakan sebagai pengendali untuk mengendalikan level ketinggian air , Gambar 3.14 menunjukkan diagram blok pengendali logika *fuzzy* , dalam implementasinya kendali logika *fuzzy* memiliki empat komponen utama sebagai konfigurasi dasar sistem kendali logika *fuzzy* yaitu :

1. Perangkat *fuzzifikasi*
2. Basis kaidah (*rule base*)
3. Mekanisme inferensi (*inference mechanism*)
4. Perangkat *defuzzifikasi*



Gambar 3.14 *Diagram Blok Pengendali Logika Fuzzy*

1. Perangkat *Fuzzifikasi*

   Fuzzifikasi berfungsi untuk menstransformasikan variabel masukan berupa sinyal masukan *(input)* yang bersifat bukan *fuzzy* (variabel numerik) untuk dikonversi atau diubah menjadi variabel *fuzzy* (variabel linguistik) dengan menggunakan operator *fuzzifikasi (fuzzifier)*, *fuzzifikasi* adalah pemetaan *input* ke semesta himpunan *fuzzy* dan secara simbolis pemetaan ini dinyatakan pada persamaan *3.4* dibawah ini.

   x = fuzzifier (xo)…...…….(*3.4*)

   Dimana *xo* adalah masukan *crisp* dari proses, *x* adalah himpunan *fuzzy* dan *fuzzifier* adalah operator *fuzzifikasi*.

2. Basis Kaidah (*rule base*)

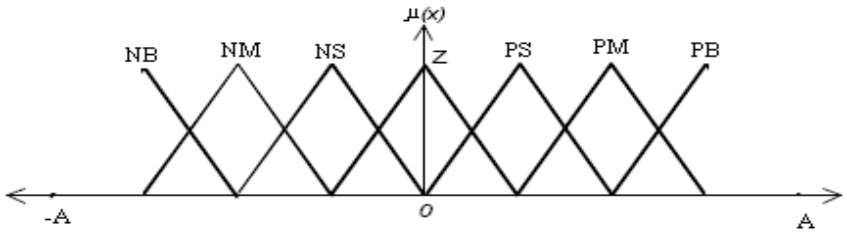
   Basis Kaidah (*rule base*) pada kontrol *fuzzy* merupakan suatu bentuk aturan relasi atau implikasi seperti "Jika-Maka" atau "*If-Then*" seperti pernyataan dibawah ini :

   "JIKA "X=A DAN "JIKA"Y=B "MAKA" Z=C

   Dalam mendefinisikan himpunan *fuzzy* biasanya dilakukan secara subyektif dan didasarkan pada pengalaman dan pertimbangan yang menyangkut kerekayasaan, hal ini dilakukan untuk mendefinisikan variabel-variabel linguistik ke dalam bentuk himpunan *fuzzy* yang mampu menterjemahkan keadaan objek nyata kebentuk *fuzzy*.

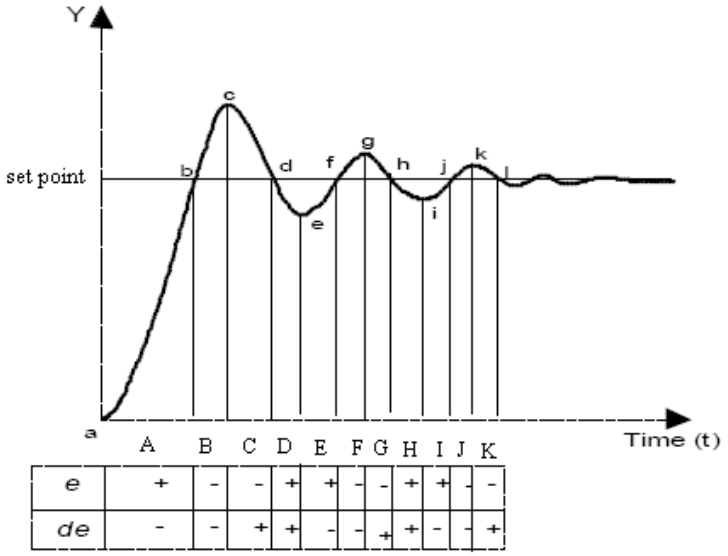
   Sebagai contoh memilih fungsi segitiga, himpuan *fuzzy* didefinisikan dalam semesta pembicaraan, yaitu dengan membagi semesta pembicaraan menjadi beberapa ruang segment, misalnya pembagian ruang dilakukan menjadi 7

variabel linguistik yaitu : positif besar (PB), positif menengah (PM), positif kecil (PS), nol (Z), negatif besar (NB), negatif menengah (NM), negatif kecil (NS). Seperti terlihat pada Gambar 3.15 dibawah ini.



Gambar 3.15 *Pendefinisian Dengan Fungsi Segitiga*

Untuk mendapatkan kaidah kendali yang baik, diperlukan ketepatan penggabungan antara intuisi dan pengalaman terhadap karakteristik proses yang diamati, misalnya tanggapan peralihan dari proses, dalam proyek tugas akhir ini kaidah kendali *fuzzy* mengacu pada trayektori bidang fasa (*e* dan Δe) dan diagram fungsi waktu respon sistem lingkar tertutup (*close loop*) masukan tangga seperti Gambar 3.15 dengan mengambil lintasan trayektorinya, akan membuat keadaan proses secepat mungkin mengarah pada titik singular, hal ini berarti keluaran proses secepat mungkin mencapai keadaan tunak sama dengan acuan. Gambar 3.16 menunjukkan tanggapan (*step respon*) sistem loop tertutup.



Gambar 3.16 *Tanggapan (step respon) Sistem Loop Tertutup*

Dengan mendesain suatu himpunan kaidah *fuzzy* yang dapat mengurangi lewatan maksimum (*overshoot*) dan waktu naik (*rise time*) maka masing-masing peubah masukan atau keluaran dibagi menjadi tujuh keanggotaan *fuzzy* yaitu negatif besar (NB), negatif menengah (NM), negatif kecil (NS), nol (Z), positif kecil (PS), positif menengah (PM) dan positif besar (PB). Nilai kualitatif ditentukan pada setiap titik yang bertanda huruf (A-I), seperti ditunjukkan pada Tabel 3.3 di bawah ini.

Tabel 3.3 *Kaidah Kendali Fuzzy*

Kaidah	***e***	**Δ*e***	***U***	**Referensi**	**Fungsi**
1	PB	Z	PB	Titik a	Perpendek *rise time*
2	PM	Z	PM	Titik e	Perpendek *rise time*
3	PS	Z	PS	Titik l	Perpendek *rise time*
4	Z	NB	NB	Titik b	Kurangi *overshoot*
5	Z	NM	NM	Titik f	Kurangi *overshoot*
6	Z	NS	NS	Titik j	Kurangi *overshoot*
7	NB	Z	NB	Titik c	Kurangi *overshoot*
8	NM	Z	NM	Titik g	Kurangi *overshoot*
9	NS	Z	NS	Titik k	Kurangi *overshoot*
10	Z	PB	PB	Titik d	Perkecil osilasi

11	Z	PM	PM	Titik h	Perkecil osilasi
12	Z	PS	PS	Titk i	Perkecil osilasi
13	Z	Z	Z	Titik set	

Agar pengendali *fuzzy* yang dirancang berlaku untuk seluruh keadaan sistem, maka kaidah-kaidah kendali yang disusun pada Tabel 3.3 dilengkapi dengan kaidah-kaidah kendali hingga jumlah kaidah kendali sama dengan jumlah kombinasi dari variabel linguistik *error* (*e*) dan turunan error (Δ*e*) yang tersedia, seperti pada Tabel 3.4 dibawah ini.

Tabel 3.4 *Kaidah-Kaidah Kendali Fuzzy Dengan Tujuh MF*

Δe / e	NB	NM	NS	Z	PS	PM	PB
PB	Z	PS	PM	PB	PB	PB	PB
PM	NS	Z	PS	PM	PB	PB	PB
PS	NM	NS	Z	PS	PM	PB	PB
Z	NB	NM	NS	Z	PS	PM	PB
NS	NB	NB	NM	NS	Z	PS	PM
NM	NB	NB	NB	NM	NS	Z	PS
NB	NB	NB	NB	NB	NM	NS	Z

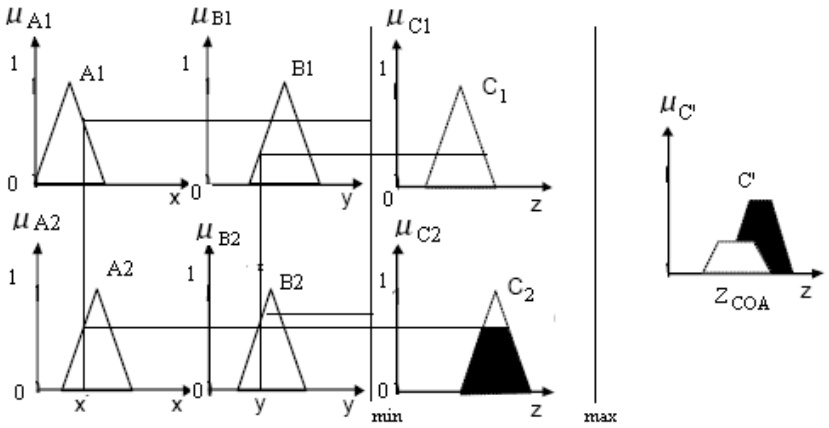
3. Mekanisme Inferensi (*inference mechanism*)

Mekanisme inferensi atau pengambilan keputusan dalam sistem *kendali logika fuzzy* dilakukan dengan penalaran pendekatan (*approximate reasoning*), misalkan masukan pengendali logika *fuzzy* adalah *fuzzy* tunggal yakni A, B sehingga penalaran *fuzzy* dapat dilakukan dengan kaidah operasi minimum Mamadani.

Minimum : $\mu_{c'}(z) = \bigcup_{i=1}^{n} \alpha_i \wedge \mu_{c_i}(z)$ ……..(3.5)

Dimana α_i adalah fakror bobot yang menyatakan ukuran kesetaraan kaidah ke-I pada aksi kendali *fuzzy*. Dalam proyek tugas akhir ini mekanisme

inferensi *fuzzy* yang digunakan adalah mekanisme inferensi *fuzzy* mamdani, seperti terlihat pada Gambar 3.17 dibawah ini.



Gambar 3.17 *Sistem Inferensi Fuzzy Mamdani*

Hasil keluaran sistem inferensi mamdani masih berupa *fuzzy*. Untuk aksi kendali nyata yang akan diumpankan pada proses diperoleh melalui proses *defuzzifikasi*. Gambar 3.17 menunjukkan sistem inferensi *fuzzy* mamdani dengan dua kaidah *fuzzy* menggunakan *min* dan *max* untuk operator *T-norm* dan *T-conorm*.

4. *Defuzzifikas*i

Defuzzifikasi merupakan pemetaan himpunan *fuzzy* ke himpunan tegas (*crips*), yang mana proses ini merupakan kebalikan dari proses *fuzzifikasi* proses defuzzifikasi dijabarkan pada persamaan 3.6 di bawah ini.

Z_0 = defuzzier (Z)……(3.6)

Dalam proyek tugas akhir metode yang digunakan untuk proses defuzzifikasi adalah metode titik tengah (*center of area*). Persamaan untuk

menentukan nilai output aksi kontrol Z_0 kasus diskrit ditunjukkan pada persamaan 3.7 di bawah ini.

$$z_{0(coa)} = \frac{\sum_{i=1}^{n} z_i \mu(z_i)}{\sum_{i=1}^{n} \mu(z_i)} \quad \text{.....(3.7)}$$

3.4.2 Perancangan Program Dalam Perangkat Lunak (*software*) LabVIEW

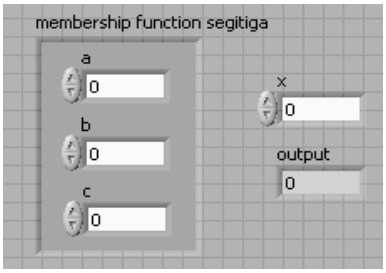
Dalam proyek tugas akhir ini dalam mengendalikan level ketinggian air diperlukan perancangan beberapa program sehingga menjadi satu-kesatuan program *pengontrol fuzzy*, beberapa program yang perlu dirancang diantaranya.

1. Perancangan program *Triangel* MF.Vi
2. Perancangan program *Trapezium*.Vi
3. Perancangan program *Input array*.Vi
4. Perancangan program Pengaman.Vi
5. Perancangan program Tangki.Vi
6. Perancangan program multiplekser.Vi
7. Perancangan Program Motor Stepper.Vi
8. Perancangan program *Fuzzy*.Vi
9. Perancangan program Pengontrol *fuzzy*.Vi

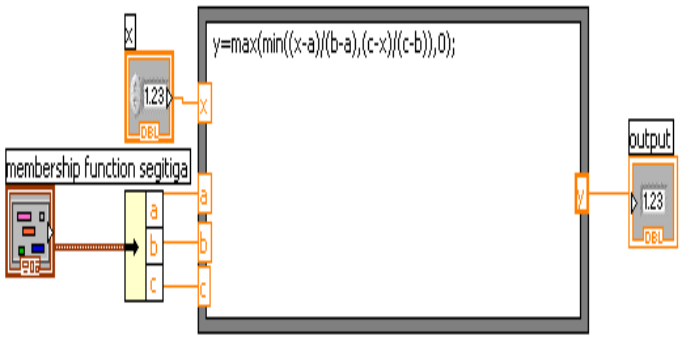
1. Perancangan Program *Triangel* MF.Vi

   Perancangan program *Triangel* MF.Vi dilakukan untuk mendefinisikan himpunan *fuzzy* dalam semesta pembicaraan sehingga memenuhi basis kaidah (*rule base*) sesuai dengan fungsi keanggotaan segitiga (*triangel*).

Gambar 3.18 dan 3.19 menunjukkan *front panel* dan *blok diagram* dari Program Triangel.Vi.



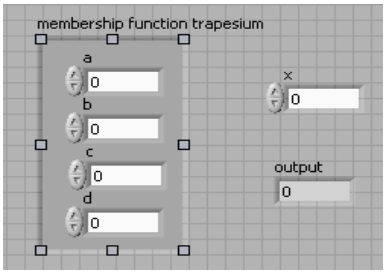
Gambar 3.18 *Front Panel Program Triangel.Vi*



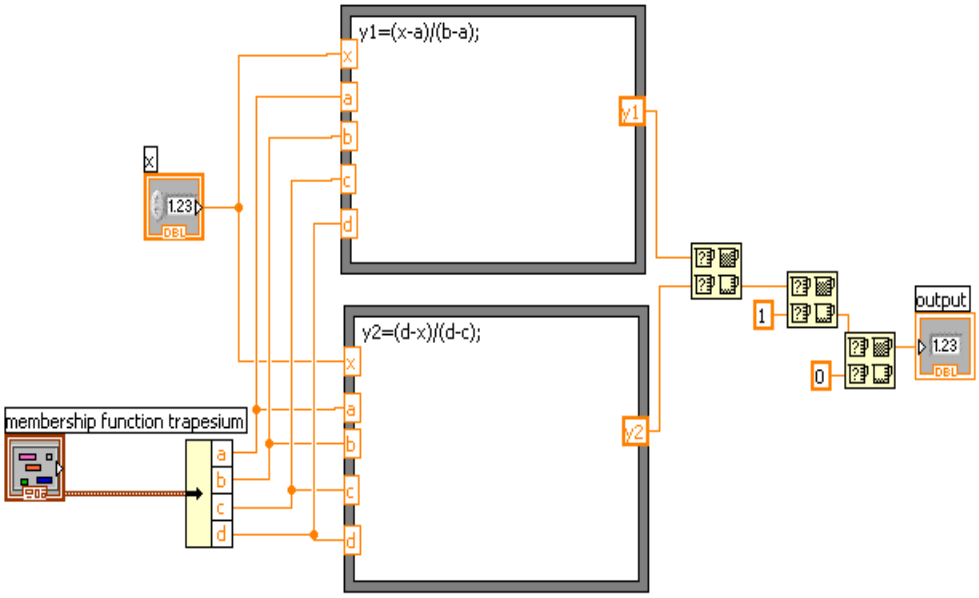
Gambar 3.19 *Blok Diagram Program Triangel.Vi*

2. Perancangan Program *Trapezium*.Vi

Perancangan program *trapezium* MF.Vi dilakukan untuk mendefinisikan himpunan *fuzzy* dalam semesta pembicaraan sehingga memenuhi basis kaidah (*rule base*) sesuai dengan fungsi keanggotaan trapesium (*trapezoid*). Gambar 3.20 dan 3.21 menunjukkan *front panel* dan *blok diagram* dari Program trapezium.VI.



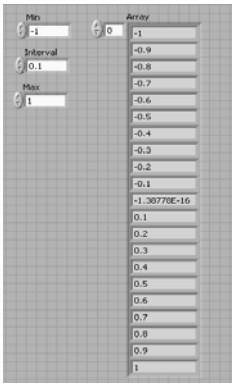
Gambar 3.20 *Front Panel Program Trapezium.Vi*



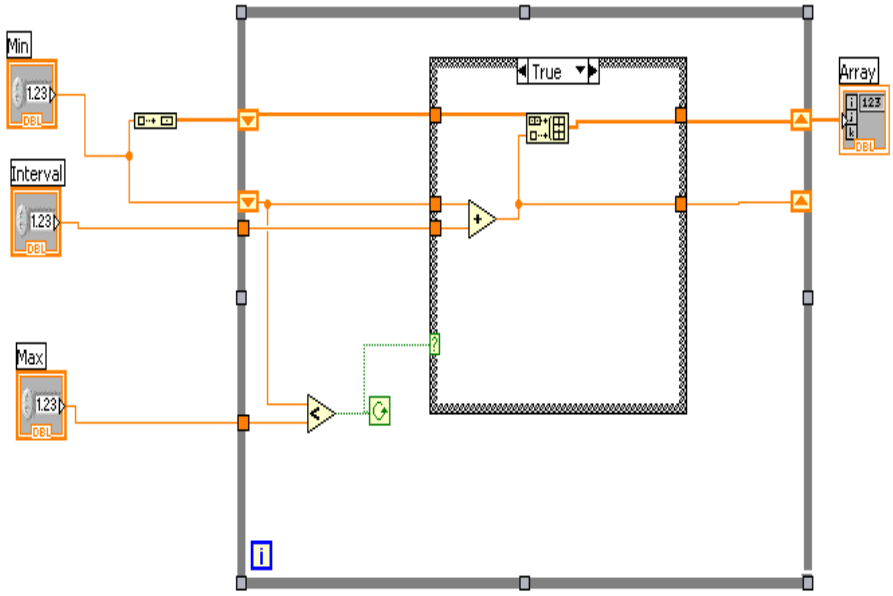
Gambar 3.21 *Blok Diagram Program Trapezium.Vi*

3. Perancangan *Input Array*.Vi

Perancangan program Input Array.Vi digunakan untuk membatasi besarnya nilai dan interval inputan untuk pendefinisian semesta pembicaraan pada fungsi keanggotaan segitiga (*triangel*) dan tarpesium (*trapezoid*). Gambar 3.22 dan 3.23 menunjukkan Perancangan program *front panel* dan *blok diagram input array*.



Gambar 3.22 *Front Panel Program Input Array.Vi*



Gambar 3.23 *Blok Diagram Program Input Array.Vi*

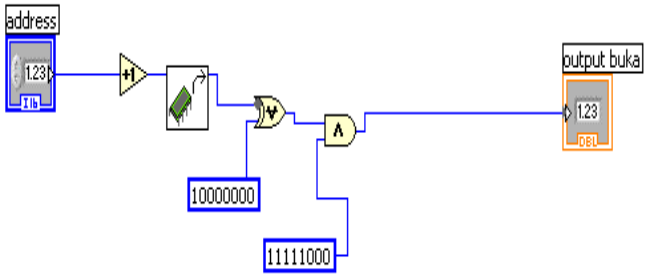
4. Perancangan Program Pengaman.Vi

Perancangan progam pengaman.Vi digunakan sebagai sakelar, prinsip tersebut memanfaatkan status 4 dan status 5 pada pin port parallel DB 25, yaitu pin 13 dan pin 12, jika status 4 aktif maka putaran keran akan mati hal ini menandakan bahwa keran telah membuka penuh sedangkan jika status 5 aktif maka putaran keran akan mati hal ini menandakan keran telah menutup

penuh, Gambar 3.24 dan 3.25 menunjukkan *front panel* dan *blok diagram* program pengaman.Vi



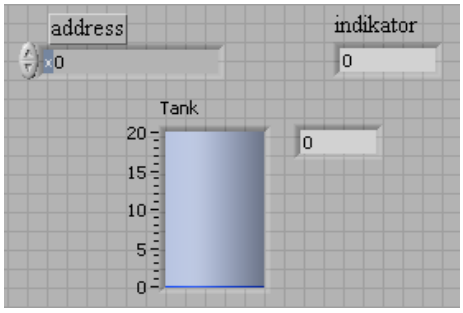
Gambar 3.24 *Front Panel Program Pengaman.Vi*



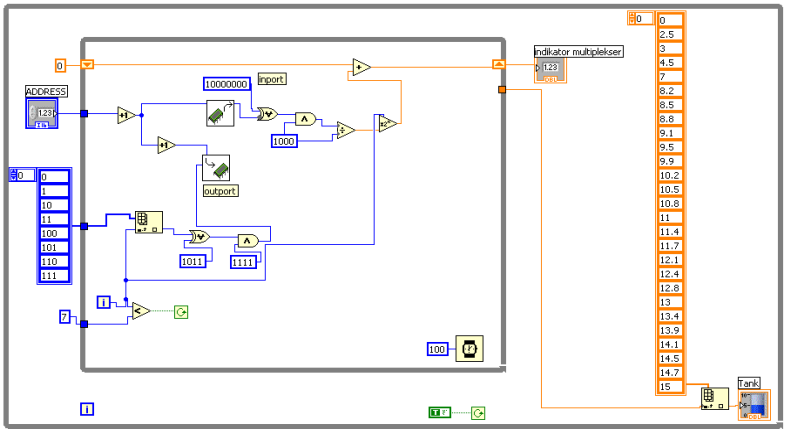
Gambar 3.25 *Blok Diagram Program Pengaman.Vi*

5. Perancangan program Tangki.Vi

Perancangan program tangki.Vi digunakan untuk mengamati perubahan level ketinggian air minimum 0 cm dan maksimum 15 cm, pertama kali dilakukan pengujian perubahan data digital dari output ADC terhadap perubahan level ketinggian air pada tangki proses, selanjutnya data-data tersebut diolah pada program tangki.Vi untuk menampilakn level ketinggian air pada tangki proses. Gambar 3.26 dan Gambar 3.27 menunjukkan *front panel* dan *blok diagram* program Tangki.Vi



Gambar 3.26 *Front Panel Program Tangki.Vi*



Gambar 3.27 *Blok Diagram Program Tangki.Vi*

6. Perancangan Program Multiplekser.Vi

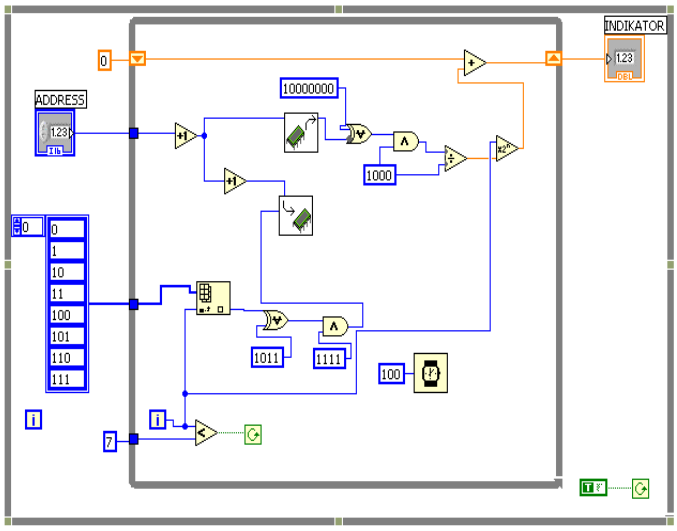
Perancangan program Multiplekser.Vi digunakan untuk mengetahui nilai keluaran dari rangkaian multiplekser, apakah sesuai dengan nilai dari keluaran rangkaian ADC (*analog to digital converter*).

front panel program Multiplekser.VI ditunjukkan pada Gambar 3.28 di bawah ini.



Gambar 3.28 *Front Panel Multiplekser.Vi*

Gambar 3.29 menunjukkan blok diagram multiplekser.Vi



Gambar 3.29 *Blok Diagram Program Multiplekser.Vi*

7. Perancangan Program Motor Stepper.Vi

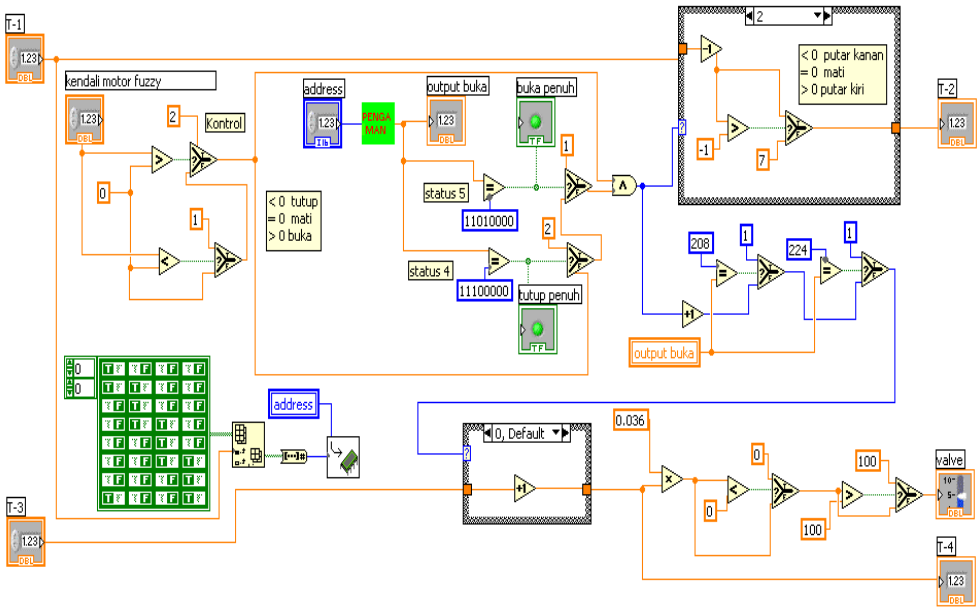
Perancangan program motor stepper.Vi digunakan untuk mengendalikan putaran motor stepper supaya dapat memutar ke kanan, ke kiri dan berhenti.

Gambar 3.30 menunjukkan front panel program motor stepper.Vi



Gambar 3.30 *Front Panel Program Motor Stepper.Vi*

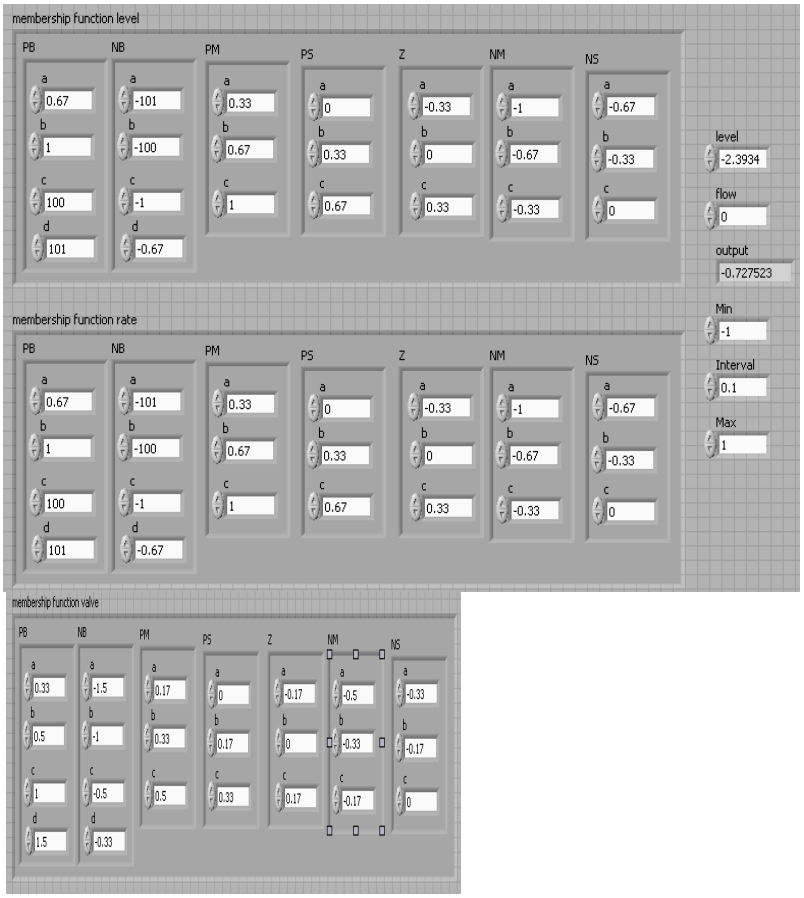
Gambar 3.31 menunjukkan blok diagram motor stepper.Vi



Gambar 3.31 *Blok Diagram Program Motor Stepper.Vi*

8. Perancangan Program *Fuzzy*.Vi

Perancangan Program *Fuzzy*.Vi dilakukan dengan menggabungkan dan merangkai Program *Triangel*.Vi, *Trapezium*.Vi dan *Input Array*.Vi sehingga dapat merealisasikan kendali logika *fuzzy* dalam perangkat lunak (*software*), yang berisi proses *fuzzifikasi*, basis kaidah (*rule base*), mekanisme inferensi (*inference mechanism*) dan *defuzzifikasi*. Gambar 3.28 menunjukkan *front panel* program *fuzzy*.Vi



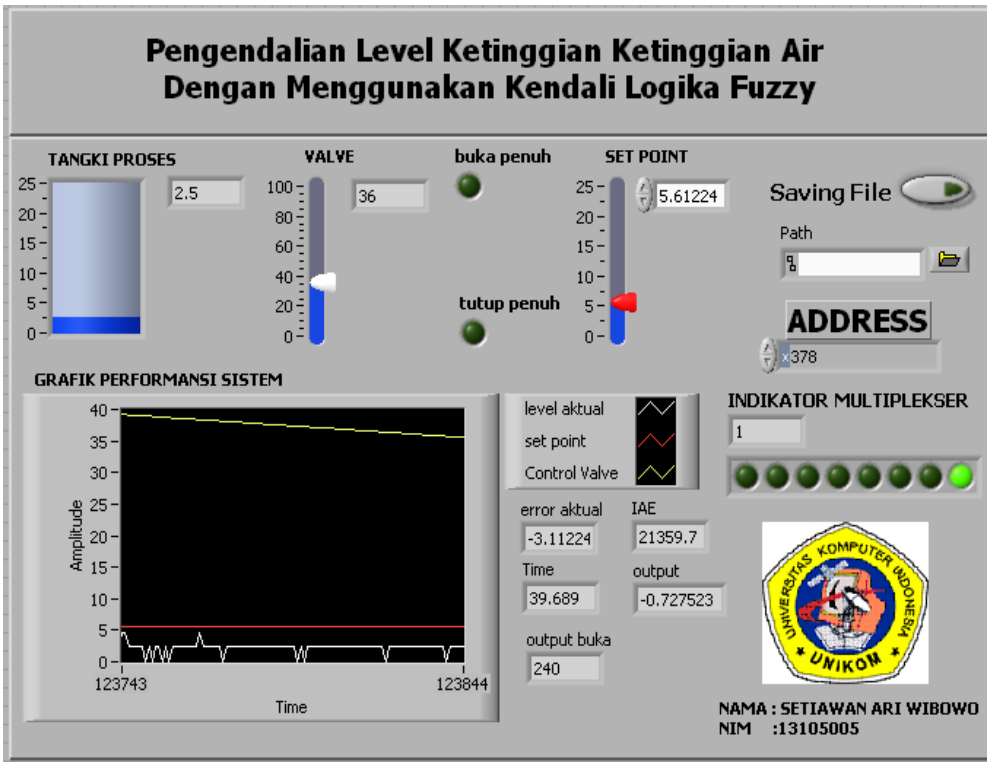
Gambar 3.32 *Front Panel Program Fuzzy.Vi*

Gambar dari blok diagram program *fuzzy*.Vi dapat dilihat pada lampiran.

9. Perancangan Program Pengontrol *Fuzzy*.Vi

Perancangan program pengontrol *fuzzy*.Vi dilakukan untuk mengendalikan level ketinggian air dari plant sehingga level ketinggian air tersebut sesuai dengan level ketinggian yang diharapkan (*setpoint*), perancangan program dilakukan dengan menggabungkan beberapa sub Vi, yaitu triangel.Vi, trapezium.Vi, input array.Vi, motor stepper.Vi, tangki.Vi, multiplekser.Vi dan *fuzzy*.Vi sehingga menjadi suatu kesatuan program pengontrol *fuzzy*.Vi yang akan mengendalikan plant.

Gambar 3.33 menunjukkan front panel program pengontrol *fuzzy*.Vi



Gambar 3.33 *Front Panel Program Pengontrol Fuzzy.Vi*

Gambar dari blok diagram pengontrol program *fuzzy*.Vi dapat dilihat pada lampiran