BAB II

TINJAUAN PUSTAKA

2.1 Tinjauan Pustaka

Penelitian sebelumnya dilakukan oleh Hanung Pratama N. (2011) mengenai ”Pengendali Suhu Water Heater Dengan Jaringan Syaraf Tiruan BACKPROPAGATION”. Penelitian ini menerapakan metode Jaringan Syaraf Tiruan pada *water heater* sebagai *plant*. Dimana menggunakan *neural network toolbo*x pada MATLAB 7.4 dan NI DAQ USB 6009 sebagai akuisisi data dan identifikasi data dari *water heater*. Proses pengambilan data menggunakan program *simulink* dengan memberikan tegangan kontrol secara random antara 0-5 volt, dimana input adalah nilai perubahan suhu (e) dan output adalah nilai tegangan kontrol (u). Jaringan syaraf tiruan menggunakan metode *backpropagation* untuk melatih data. Sensor yang digunakan adalah LM35. Struktur jaringan terbaik adalah diperoleh dari pelatihan, terdiri dari 10 sel neuron lapisan input, 5 sel neuron hidden *layer*, dan 1 sel neuron lapisan output. JST yang dirancang mempunyai respon yang cukup cepat dan mampu beroperasi pada rentang suhu 40 sampai 90 derajat Celcius.

Penelitian juga dilakukan oleh Sri Utami Kusumadewi (2007) mengenai ”Implementasi Jaringan Syaraf Tiruan berbasis Metode Backpropagation sebagai Pengendali Kecepatan Motor DC”. Penelitian ini menerapkan jaringan syaraf tiruan pada pengendalian motor DC dengan

metode *backpropagation*. *Interface* yang digunakan dirancang menggunakan Delphi. Untuk mengatur pengiriman dan penerimaan data dari sensor *optocoupler* ke komputer maupun sebaliknya digunakan sebuah mikrokontroler AT 89C51. Sebagai driver, juga digunakan mikrokontroller AT 89C2051 untuk membangkitkan pulsa-pulsa PWM. Hasil yang diperoleh dari menunjukkan masih terdapat error rata-rata sebesar 14,88 rpm.

Penelitian lainnya dilakukan oleh Jhon Hendra (2006) mengenai "Simulasi Jaringan Syaraf Tiruan berbasis Metode Backpropagation sebagai Pengendali Level Air". Penelitian ini juga menerapkan jaringan syaraf tiruan dengan metode *backpropagation* untuk pengendalian *plant* level air. Pertama-tama dilakukan penentuan ukuran dari tangki yang akan kita kendalikan ketinggiannya, kemudian menentukan besarnya aliran air masuk maksimum yang mengalir ke tangki, dan yang terakhir menentukan besarnya luas pipa pembuangn air. Sistem tersebut kemudian dikendalikan menggunakan sistem kendali invers berbasis jaringan syaraf tiruan. Pengujian sistem telah dilakukan untuk mengontrol level air dengan ketinggian antara 0 sampai 50 cm dengan ukuran alas tangki sebesar 50 x 60 cm. dengan menggunakan metode *backpropagation* dan fungsi *Gradient Descent Momentum* diperoleh struktur jaringan yang terbaik terdiri dari 7 sel neuron input, 5 sel neuron lapisan tersembunyi dan 1 sel neuron output, fungsi aktivasi pada setiap lapisan menggunakan fungsi sigmoid bipolar, dengan *learning rate* 0,6 dan *momentum coefficient* 0,6

menghasilkan *mean square error* (MSE) 0,00077. Persentase MSE pengujian adalah 0,0604%. Banyaknya jumlah data masukan berpengaruh terhadap banyaknya iterasi dan *mean square error* yang dihasilkan.

Selain itu penelitian juga dilakukan oleh Insan Kamil Ahmad P. (2011) mengenai ”Sistem Pengendalian Suhu Furnace berbasis PID”. Penelitian ini meliputi perancangan dan implementasi sistem kendali suhu dan masa aktif *reactor furnace* untuk proses kimia. Sistem kendali menggunakan mikrokontroler Atmega 8535 dengan control PID dan dengan nilai masa aktif dan *set point* sebagai masukan. Nilai masa aktif dan *set point* dikirim melalui keypad yang terhubung ke mikrokontroller. Nilai masa aktif akan dimulai saat *nilai present value* sama dengan nilai *set point* dan akan menghentikan sistem setelah *timer* sama dengan nilai masa aktif. Nilai *set point* akan dibandingkan dengan nilai *present value* atau nilai suhu terakhir yang dideteksi oleh sensor dan hasil perbandingan disebut *error*. *Error* tersebut akan dimanipulasi oleh kontrol PID untuk mengatur tegangan pada *furnace*. Dari perancangan sistem pengendali PID yang telah dibuat, nilai Kp=10.728, Ki=0.012364, Kd=2364. Pengujian sistem keseluruhan dilakukan dengan menggunakan nilai *set point* 600^0C dan suhu awal pada *furnace* sebesar 30^0C. Dari hasil yang diperoleh menunjukkan bahwa kontrol PID mampu mengendalikan suhu *furnace* dengan kestabilan yang baik dan tanggapan yang cepat terhadap perubahan suhu yang dikendalikan.

2.2 *Pyrolisis*

Pyrolisis dapat didefinisikan sebagai dekomposisi thermal material organik pada suasanan *inert* (tanpa kehadiran oksigen) yang akan menyebabkan terbentuknya senyawa volatile. *Pyrolisis* pada umumnya diawali pada suhu 200^{o}C dan bertahan pada suhu sekitar 450-500^{o}C (Sheth and Babu, 2006). *Pyrolisis* suatu biomassa akan menghasilkan tiga macam produk, yaitu gas, cair, dan padat (*char*). Jumlah produk gas, cair, dan *char* tergantung pada jenis prosesnya (suhu dan waktu *pyrolisis*), seperti terlihat pada tabel 2.1.

Tabel 2.1 Kandungan Produk Cair, Padat, dan Gas pada Berbagai Jenis *Pyrolisis*

Jenis *Pyrolisis*	**Komposisi**		
	Gas	**Cair**	**Padat (*Char*)**
Fast Pyrolisis - Suhu moderat (~500^{o}C) - Waktu *pyrolisis* singkat (<2 detik)	13 %	75 % (senyawa organik)	12 %
Carbonization - Suhu relatif rendah - Waktu *pyrolisis* lama	35 %	30 % (air)	35 %
Gasification - Suhu tinggi (>800^{o}C) - Waktu *Pyrolisis* lama	85 %	5 % (*thar*)	10 %

Dua cara utama dari pemanasan partikel biomassa dalam sistem *fast pyrolisis* terdiri dari :

- Transfer panas gas – padat seperti pada reaktor *entrained flow* dimana panas ditransfer dari gas panas ke partikel biomassa yang ter*pyrolisis* oleh konveksi
- Transfer panas padat – padat dengan kebanyakan transfer panas konduksi

Yield yang tinggi dari cairan *fast pyrolisis* dapat diperoleh dari kombinasi yang optimal antara lain :

- Tingkat pemanasan yang sangat tinggi
- Temperature uap air di bawah $600^{o}C$
- *Residence time* uap air yang sangat rendah dan pemunduran yang cepat dari *resultan* uap air

Meskipun tiga prosuk utama dari gas, cair dan *char* terbentuk, antara gas dan *char* diperkecil melalui kontrol temperatur dan *residence time* secara umum 15 – 20 % berat dari bahanbaku kering. Temperatur yang tinggi yang dibutuhkan untuk *pyrolisis* dapat diperoleh dalam beberapa cara, antara lain :

1. Memanaskan campuran gas – padat melalui dinding reaktor.
2. Memanaskan dengan transfer panas medium yaitu gas, misalnya *preheated recycle gas* atau cair, misalnya *molten metal* atau *molten salt.*
3. Memanaskan melalui reaksi eksotermis ke dalam reaktor, misalnya oksidasi parsial.

Dalam semua kasus, mekanisme transfer panas dan kontrol adalah langkah penting dalam desain reaktor.

Sebagai contoh dalam proses *pyrolisis* adalah proses asap cair untuk menghasilkan *bio oil*. Pada proses ini, menggunakan bahan baku organik *bio massa* berupa sekam padi. Range suhu yang digunakan untuk menghasilkan *bio oil* berada pada suhu 450^{o}C, dimana hasilnya adalah berupa *bio diesel* (solar). Perlu diketahui bahwa untuk perubahan suhu yang tinggi ataupun rendah akan menyebabkan perubahan produk yang dihasilkan.

Tabel 2.2 *Fraksi Bio Oil*

No.	Produk	Suhu Kerja
1	*Gasoline*	480^{o}C
2	*Bio diesel*	450^{o}C
3	*Heavy oil*	420^{o}C

Dari Tabel di atas, dapat diketahui perubahan suhu untuk tiap produk sebesar 30^{o}C. Ketika suhu referensi 450^{o}C untuk menghasilkan *bio* diesel (solar) mengalami lonjakan melebih 30^{o}C, maka akan menghasilkan produk *gasoline*. Begitu pula dengan keadaan sebaliknya, akan menghasilkan produk *heavy oil*. Maka untuk itu, untuk mendapatkan produk yang sesuai, *overshoot* tidak boleh melebihi 30^{o}C dari suhu referensinya. Untuk mendapatkan hasil yang lebih baik, *overshoot* berada pada range 5^{o}C dari suhu referensinya.

2.2.1 *Pyrolisis* Katalitik

Pyrolisis katalitik adalah proses *pyrolisis* yang menggunakan katalisator. Katalisator disini berfungsi untuk memecah hidrokarbon rantai panjang menjadi hidrokarbon rantai pendek (C_1 – C_5). Disamping itu, katalisator mampu meningkatkan kecepatan dekomposisi dan memperbesar produk cair hasil *pyrolisis* (Scheirs and Kaminsky, 2006).

Penelitian pertama *pyrolisis* katalitik dilakukan oleh Uemichi et.al (1983) yang meneliti *pyrolisis* polietilen menggunakan katalisator Pt/silica-alumina dan Pt/alumina. Beberapa tahun kemudian Ishihara et.al (1992) melaporkan bahwa *pyrolisis* polietilen menggunakan katalisator silica-alumina dapat memperpendek rantai polimer dan meningkatkan cabang rantai. Betrame and Carniti (1989) membandingkan berbagai macam katalisator yang digunakan bahwa katalisator zeolit merupakan katalisator yang paling efektif.

Walendzie (2001) melaporkan bahwa temperatur optimum *pyrolisis* sampah polyolefin 410 – 430^{o}C sedangkan jika menggunakan *pyrolisis* katalitik maka temperatur optimum berkisar 390^{o}C. Manos (2000) meneliti *pyrolisis* katalitik dengan katalisator zeolit mendapati bahwa produk cair yang diperoleh berupa hidrokarbon dengan range C_3 – C_{15}.

2.3 Jaringan Syaraf Tiruan

Jaringan syaraf tiruan merupakan tiruan syaraf biologi, yang pertama kali diperkenalkan oleh McCulloh dan Pitts pada tahun 1943, Hebb pada tahun 1949 memperkenalkan sebuah teknik pelatihan jaringan syaraf tiruan yang dinamakan Hebbian rule. Pada tahun 1961 Rosenbaltt memperkenalkan jaringan syaraf satu lapis yang diberi nama perseptron dan digunakan pada pengenalan pola optis.

Aplikasi jaringan syaraf dibidang kendali pertama kali diperkenalkan oleh Widrow dan Smith pada tahun 1964 yaitu jaringan *Adaptive Linear Network* (ADALINE) yang digunakan untuk mengendalikan sistem pendulum terbaik. Algoritma pembelajaran *backpropagation* diperkenalkan oleh Werbos pada tahun 1974 dan Rumelhart pada tahun 1986 yang mengantarkan konsep *Multi-layer Perceptron* (MLP). Jaringan syaraf tiruan sebagai sistem kendali cerdas memiliki keunggulan sebagai berikut :

- Jaringan syaraf belajar dari pengalaman bukan diprogram
- Jaringan syaraf memiliki kemampuan untuk melakukan generalisasi data yang tersembunyi berdasarkan data pelatihan yang ada
- Jaringan syaraf memiliki kecepatan komputasi yang cepat dan dapat diimplementasikan secara *real time.*

2.3.1 Dasar-dasar Jaringan Syaraf Tiruan

Jaringan syaraf tiruan sebagai sistem pengolah informasi, mempunyai karakteristik yang sama dengan jaringan syaraf biologi. Jaringan syaraf tiruan telah dikembangkan sebagai generalisasi model matematis kognisi manusia atau syaraf biologi, dengan berdasarkan beberapa asumsi sebagai berikut.

- Pengolahan informasi terjadi dalam beberapa elemen sederhana yang disebut neuron
- Sinyal dilewatkan anatarneuron melalui penghubung
- Setiap penguhubung mempunyai bobot jaringan yang akan dikalikan dengan sinyal yang melewatinya
- Setiap neuron mempunyai fungsi aktivasi (biasanya tidak linier) untuk net input (jumlah terbobot sinyal masukan) untuk menentukan sinyal keluarannya.

Karakteristik jaringan syaraf ditentukan oleh :

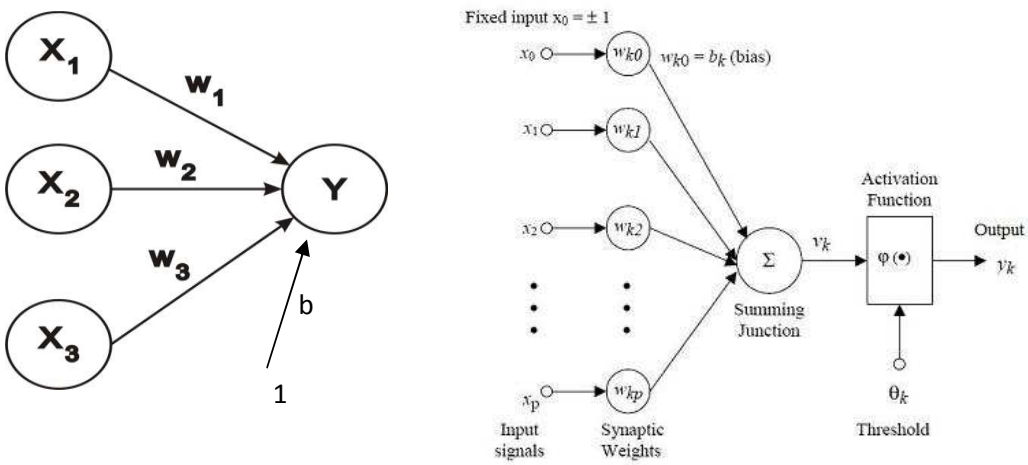
- Pola hubungan antarneuron (arsitekturnya)
- Metode penentuan bobot jaringan (*training* atau algoritma belajar)
- Fungsi aktivasinya

Jaringan syaraf terdiri atas sejumlah elemen pemroses yang disebut neuron, unit, sel, atau node. Setiap neuron dihubungkan ke neuron lainnya oleh penghubung terarah dengan bobot tertentu. Bobot tersebut merepresentasikan informasi yang akan digunakan oleh jaringan untuk menyelesaikan suatu permasalahan. Jaringan syaraf dapat diterapkan pada

berbagai permasalahan seperti penyimpanan data, pemanggilan data atau pola, klasifikasi pola, pemetaan *input-output*, pengelompokan pola yang sama, atau menemukan solusi untuk menyelesaikan masalah optimasi.

Setiap neuron mempunyai keadaan internal yang disebut aktivasi yang merupakan fungsi dari masukan yang diterimanya. Kemudian neuron akan mengirim aktivasinya ke beberapa neuron lainnya. Perlu dicatat bahwa suatu neuron hanya dapat mengirim sinyal sekali dalam suatu waktu, meskipun sinyal tersebut dipancarkan ke beberapa neuron.

Sebagai contoh diilustrasikan oleh gambar 2.1. Neuron Y menerima masukan dari neuron X_1, X_2, dan X_3 dengan aktivasi masing-masing x_1, x_2, dan x_3. Bobot jaringan dari X_1, X_2, dan X_3 ke Y masing-masing w_1, w_2, dan w_3. Sedangkan bobot yang dikalikan dengan 1 sebagai masukan ke dalam neuron Y dinamakan bias (b).



a) Struktur Neuron b) Proses Komputasi Neuron

Gambar 2.1 Jaringan Syaraf Tiruan Satu Lapis

Input jaringan ke neuron Y adalah jumlah terbobot dari sinyal neuron X_1, X_2, dan X_3 dengan persamaan :

$y\_in = x_1w_1 + x_2w_2 + x_3w_3 + b$ (2.1)

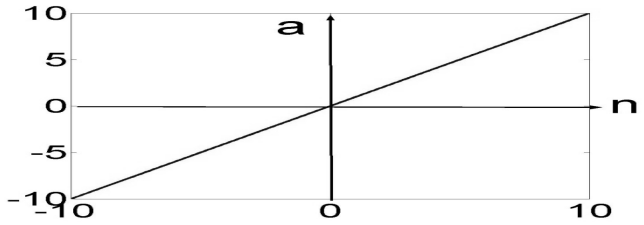
atau

$y\_in =$ $+ b$ (2.2)

Aktivasi neuron Y ditentukan oleh suatu fungsi terhadap input neuron, *y = f(y_in)*. Fungsi aktivasi yang sering digunakan dalam jaringan syaraf adalah sebagai berikut :

- Fungsi identitas atau fungsi linier, ditunjukkan pada gambar 2.2 yang memiliki persamaan :

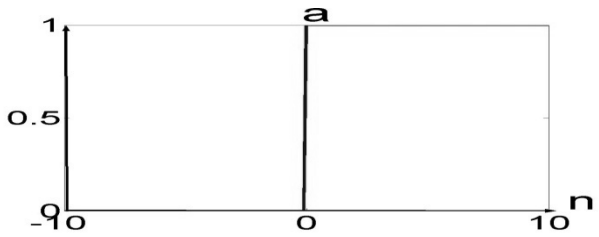
  f(s) = s + C, untuk semua s (2.3)



Gambar 2.2 Fungsi Aktivasi Linier

- Fungsi undak biner atau *hard limiter*, ditunjukkan pada gambar 2.3 yang memiliki persamaan :

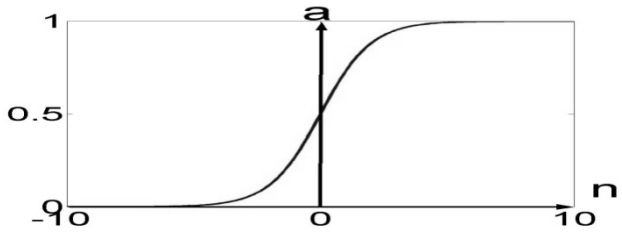
  f(s) = (2.4)



Gambar 2.3 Fungsi Aktivasi Undak

- Fungsi *sigmoid*, ditunjukkan pada gambar 2.4 dan memiliki persamaan :

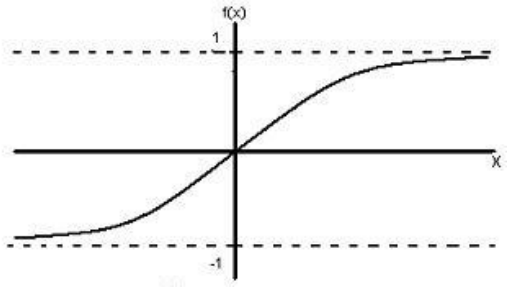
  f(s) = ——— (2.5)



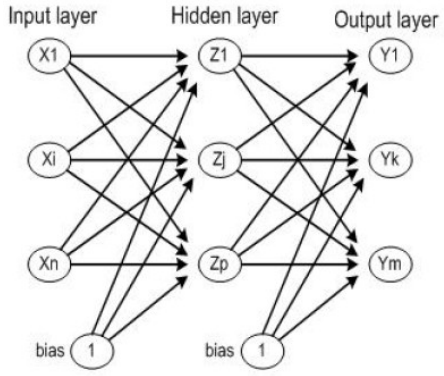
Gambar 2.4 Fungsi Aktivasi *Sigmoid*

- Fungsi *bipolar sigmoid*, ditunjukkan pada gambar 2.5 dan memiliki persamaan :

  f(s) = ——— (2.6)



Gambar 2.5 Fungsi Aktivasi *Bipolar Sigmoid*



Gambar 2.6 Jaringan Syaraf Tiruan Multi Lapis

Jaringan syaraf digambarkan ke dalam lapisan-lapisan neuron. Umumnya neuron dalam satu lapis mempunyai perilaku yang sama. Kunci utama yang menentukan perilaku neuron adalah fungsi aktivasi dan pola hubungan terbobot pengiriman dan penerimaan sinyal. Susunan neuron dalam bentuk lapisan-lapisan dan pola hubungan di dalam dan antar lapisan disebut arsitektur jaringan. Berdasarkan arsitekturnya tersebut dikenal beberapa jaringan sebagai berikut.

- Jaringan satu lapis

  Mempunyai satu lapis hubungan terbobot dengan neuron masukan yang menerima sinyal dari dunia luar dan neuron keluaran yang menghasilkan respon jaringan seperti ditunjukkan oleh gambar 2.1. Untuk jaringan satu lapis ini bobot untuk suatu neuron keluaran tidak akan mempengaruhi bobot untuk neuron keluaran lainnya.

- Jaringan multi lapis

  Jaringan dengan satu atau lebih lapisan tersembunyi yang terdapat diantara neuron masukan dan neuron keluaran. Contoh jaringan dengan satu lapisan tersembunyi ditunjukkan pada gambar 2.6. Jika *input* adalah x_i dan bobot adalah w_{ij}, dan bias adalah maka perhitungan *input* ke sebuah neuron pada lapisan ke-j dirumuskan

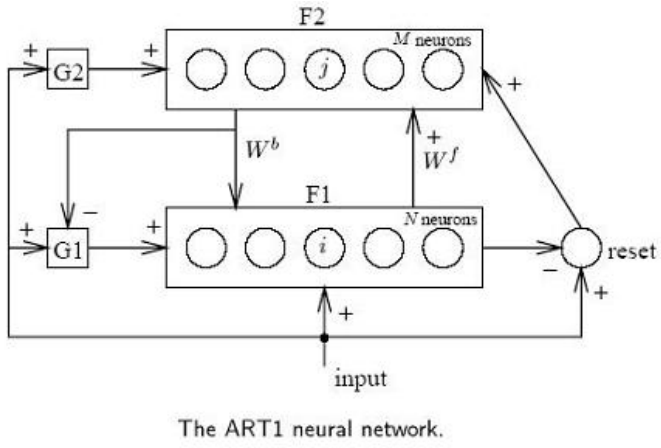
  $S_j(t) = \sum_{i=1}^{n} w_{ij} x_i(t) + b_i$ (2.7)

  Atau dalam bentuk matriks

  $S_i(t) = w_i x + b_i$ (2.8)

- Jaringan kompetitif

  Dalam lapisan kompetitif ini hanya ada satu neuron dengan aktivasi terbesar yang akan keluar sebagai pemenang. Lapisan kompetitif ini merupakan bagian terbesar dari jaringan syaraf. Contoh jaringan kompetitif adalah jaringan ART (*Adaptive Resonance Theory*) seperti pada gambar 2.7.



Gambar 2.7 Struktur Jaringan ART

2.3.2 Metode Pembelajaran

Belajar atau *training* bagi jaringan syaraf merupakan proses penentuan nilai bobot jaringan. Ada dua macam metode belajar yaitu metode terbimbing (*supervised training*) dan metode tak terbimbing (*unsupervised training*).

- Metode terbimbing

  Pada metode ini, *training* dilakukan dengan memberikan sederetan vektor *training* atau pola-pola sesuai dengan vektor target keluaran. Aplikasi dari metode belajar ini misalnya pada masalah klasifikasi pola, penggabungan pola yang dikenal sebagai *associative memory*,

seta pemetaan nonlinier dari n dimensi vektor masukan ke m dimensi vektor keluaran.

- Metode tak terbimbing

  Pada metode ini hanya diperlukan sederetan vektor masukan tanpa vektor targetnya. Jaringan akan memodifikasi bobotnya sehingga vektor *input* yang mempunyai kesamaan paling banyak akan dikelompokkan ke dalam neuron keluaran yang sama (*cluster*). Jaringan ini sering disebut sebagai *self-organizing neural nets*.

2.3.3 Jaringan Syaraf Tiruan *Backpropagation*

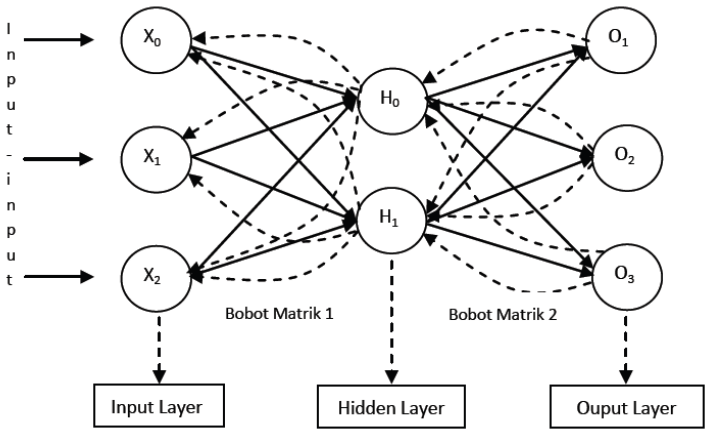
Metode pembelajaran jaringan syaraf rambat balik (*Backpropagation Neural Network*) menggunakan ide perambatan balik nilai *error* atau lebih dikenal sebagai *generalized delta rule*. *Delta rule* merupakan metode *gradient descent* untuk meminimalkan jumlah kuadrat *error* keluaran yang dihasilkan oleh jaringan.

Pembelajaran jaringan menggunakan metode *backpropagation* meliputi tiga langkah yaitu :

1. Perambatan maju (*feedforward*) polam masukan input
2. Perhitungan dan perambatan balik (*backpropagation*) nilai *error*, serta
3. Penyesuain nilai bobot jaringan berdasarkan besarnya *error* tersebut

Setelah pembelajaran, aplikasi dari jaringan terlatih merupakan fase perhitungan arah maju saja.

Arsitektur jaringan syaraf rambat balik dengan satu lapisan tersembunyi ditunjukkan oleh gambar 2.8.



Gambar 2.8 Jaringan Syaraf Rambat Balik dengan Satu Lapisan Tersembunyi

Selama perhitungan arah maju, setiap neuron *input* (Xi) menerima sinyal masukan dan memancarkannya ke setiap neuron tersembunyi $Z_i,\ldots,Z_p$. Setiap neuron tersembunyi kemudian menghitung aktivasinya dan mengirim sinyal aktivasi tersebut ke setiap neuron keluaran. Setiap neuron keluaran Y_k menghitung aktivasinya untuk menyatakan respon jaringan terhadap pola masukan yang diberikan.

Selama proses pembelajaran, setiap neuron keluaran membandingkan aktivasinya y_k dengan nilai target t_k untuk menentukan besarnya *error*. Berdasarkan besarnya *error* ini, kemudian dihitung factor δ_k (k = 1,…,m). δ_k digunakan untuk mendistribusikan nilai *error* pada neuron keluaran Y_k kembali ke semua neuron pada lapisan sebelumnya. Nilai ini juga digunakan untuk memperbarui nilai bobot antara neuron keluaran dan neuron – neuron tersembunyi. Dengan cara yang sama, factor δ_j (j = 1,…,p) dihitung untuk setiap neuron tersembunyi Z_j. Nilai δ_j tidak

dirambatkan kembali ke neuron masukan, namun digunakan untuk memperbarui nilai bobot antara neuron – neuron tersembunyi dengan neuron masukan.

Setelah semua faktor δ dihitung, bobot untuk semua lapisan diperbarui secara simultan. Algoritma yang digunakan dalam pembelajaran jaringan syaraf rambat balik ini adalah sebagai berikut.

Langkah 0 Inisialisasi bobot

Langkah 1 Sementara kondisi berhenti salah, lakukan langkah 2-9

Langkah 2 Untuk setiap pasang data belajar, lakukan langkah 3-8

Arah maju :

Langkah 3 Setiap neuron masukan (X_i = 1,…,n) menerima sinyal masukan x_i dan meneruskan sinyal tersebut ke lapisan di atasnya (lapisan tersembunyi).

Langkah 4 Setiap neuron tersembunyi (Z_j = 1,…,p) menghitung sinyal masukan terbobot :

$$Z\_in_j = v_{0j} + \sum_{i=1}^{n} x_i v_{ij} \quad (2.9)$$

dan menerapkan fungsi aktivasinya untuk menghitung sinyal keluaran :

$$z_j = f(z\_in_j) \quad (2.10)$$

dan mengirim sinyal ini ke semua neuron di atasnya (lapisan keluaran)

Langkah 5 Setiap neuron keluaran (Y_k, k=1,…,m) menghitung sinyal masukan terbobot :

$$Y\_in_k = w_{0k} + \sum_{j=1}^{p} z_j w_{ij} \quad (2.11)$$

dan menerapkan fungsi aktivasinya untuk menghitung sinyal keluaran :

$$y_k = f(y\_in_k) \quad (2.12)$$

Perambatan balik nilai error :

Langkah 6 Setiap neuron keluaran (Y_k, k=1,…,m) menerima pola target sesuai dengan pola masukan, menghitung nilai *error* :

$$\delta_k = (t_k - y_k) f'(y\_in_k) \quad (2.13)$$

Jika fungsi aktivasi yang digunakan adalah fungsi sigmoid maka :

$$F'(y\_in_k) = f(y\_in_k)(1-f(y\_in_k)) = y_k(1-y_k) \quad (2.14)$$

Sehingga persamaan 2.13 dapat ditulis sebagai :

$$\delta_k = (t_k - y_k) y_k (1-y_k) \quad (2.15)$$

Menghitung nilai koreksi bobot yang digunakan untuk memperbarui nilai bobot $w_j k$:

$$\Delta w_{jk} = \alpha \delta_k \quad (2.16)$$

Menghitung nilai koreksi bias yang digunakan untuk memperbarui nilai w_{0k} :

$$\Delta w_{0k} = \alpha \delta_k \quad (2.17)$$

Dan mengirim nilai *error* δ_k ke lapisan bawahnya

Langkah 7 Setiap neuron tersembunyi (Z_j, j=1,…,m) menghitung masukan delta dari neuron di atasnya :

$$\delta\_in_j = \sum_{k=1}^{m} \delta_k w_{jk} \quad (2.18)$$

dikalikan dengan turunan dari fungsi aktivasinya untuk menghasilkan nilai *error* :

$\delta_j = \delta\_in_j f'(z\_in_j)$ (2.19)

menghitung nilai koreksi bobot yang digunakan untuk memperbarui nilai bobot $v_j k$:

$\Delta w_{ij} = \alpha \delta_j x_i$ (2.20)

Menghitung nilai koreksi bias yang digunakan untuk memperbarui nilai $v_0 j$

$\Delta w_{0j} = \alpha \delta_j$ (2.21)

Memperbarui nilai bobot dan bias :

Langkah 8 Setiap neuron keluaran (Y_k, k=1,…,m) memperbarui nilai bias dan bobot (j=0,…,p) :

$W_{jk}(new) = w_{jk}(old) + \text{“}w_{jk}$ (2.22)

Setiap neuron tersembunyi (Z_j, j=1,…,m) memperbarui nilai bias dan bobot (i=0,…,n)

$V_{ij}(new) = v_{ij}(old) + \text{“}v_{ij}$ (2.23)

Langkah 9 Menguji kondisi berhenti

2.4 *Microcontroller* AVR ATMEGA 32

AVR merupakan seri mikrokontroler CMOS 8-bit buatan Atmel, berbasis arsitektur RISC (*Reduced Instruction Set Computer*). Hampir semua instruksi dieksekusi dalam satu siklus *clock*. AVR mempunyai 32 *register general-purpose*, *timer/counter fleksibel* dengan mode *compare*, *interrupt internal* dan *eksternal*, serial UART, *programmable watchdog timer*, dan *mode power saving*, ADC, dan PWM *internal*. AVR juga mempunyai *in-system programmable flash on-chip* yang mengijinkan memori program untuk diprogram ulang dalam sistem menggunakan hubungan serial SPI.

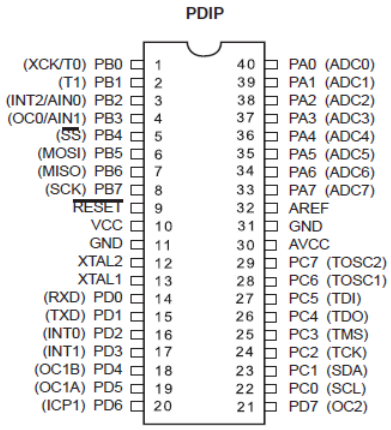
2.4.1 Fitur ATMEGA 32

Fitur-fitur yang dimiliki ATMEGA 32 sebagai berikut :

1. Mikrokontroler AVR 8 bit yang memiliki kemampuan tinggi dengan daya yang rendah.
2. Arsitektur RISC dengan *troughput* mencapai 16 MIPS pada frekuensi 16 MHz.
3. Memiliki kapasitas *flash* memori 32 Kbyte, EEPROM 1024 Byte, dan SRAM 2 Kbyte.
4. Saluran I/O sebanyak 32 buah, yaitu *Port* A, *Port* B, *Port* C, dan *Port* D.
5. CPU yang terdiri dari 32 buah *register*.
6. Unit interupsi internal dan eksternal.

7. *Port* USART untuk komunikasi serial.
8. Fitur *peripheral*
    - Tiga buah *timer*/*counter* dengan kemampuan pembandingan.
        - 2 buah *timer/counter* 8 bit dengan *prescaller* terpisah dan *mode compare*
        - 1 buah *timer/counter* 16 bit dengan *prescaller* terpisah, mode *compare*, dan mode *capture*
    - *Real time counteri* dengan *oscillator* tersendiri
    - 4 *channel* PWM
    - 8 *channel*, 10 bit ADC
        - 8 *single ended channel*
        - 7 *differential channel* hanya pada kemasan TQFP
        - 2 *differential channel* dengan *programmable gain* 1x, 10x, atau 200x
    - *Byte oriented two wire serial interface*
    - *Programmable serial* UART
    - Antarmuka SPI
    - *Watchdog timer* dengan *oscillator internal*
    - *On-chip analog comparator*

2.4.2 Konfigurasi Pin AVR ATMEGA 32



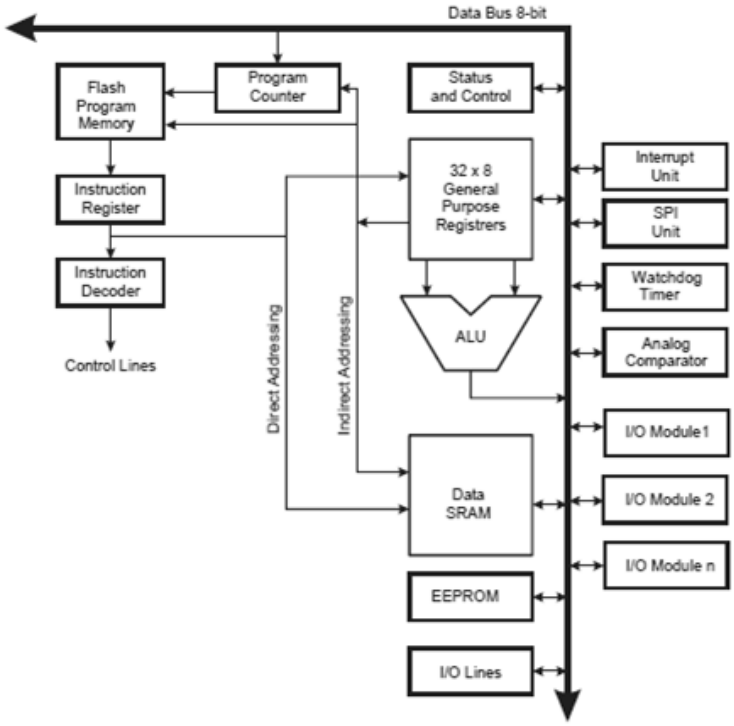
Gambar 2.9 Konfigurasi Pin ATMEGA 32

Konfigurasi pin ATMEGA 32 dengan kemasan 40 pin DIP (*Dual Inline Package*) dapat dilihat pada gambar 2.9. dari gambar diatas dapat dijelaskan fungsi dari masing-masing pin ATMEGA 32 sebagai berikut :

1. *Vcc* merupakan pin yang berfungsi sebagai catu daya
2. *GND* merupakan pin *ground*
3. *Port A (PA0…PA7)* merupakan pin *input/output* dua arah dan pin masukan ADC
4. *Port B (PB0…PB7)* merupakan pin *input/output* dua arah dan pin fungsi khusus
5. *Port C (PC0…PC7)* merupakan pin *input/output* dua arah dan pin fungsi khusus
6. *Port D (PD0…PD7)* merupakan pin *input/output* dua arah dan pin fungsi khusus
7. *RESET* merupakan pin yang digunakan untuk me-*reset* mikrokontroler
8. *XTAL1* dan *XTAL2* merupakan pin masukan *clock eksternal*

9. *AVCC* merupakan pin masukan tegangan untuk ADC
10. *AREF* merupakan pin masukan tegangan referensi ADC

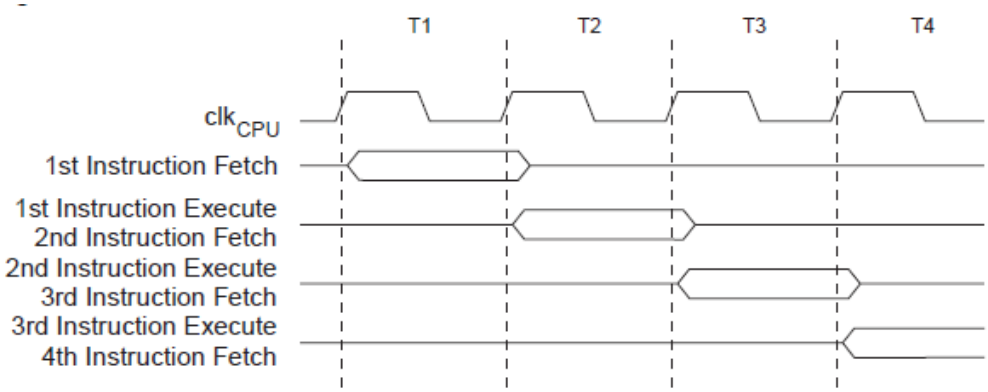
2.4.3 Arsitektur AVR RISC



Gambar 2.10 Arsitektur Mikrokontroler AVR RISC

Dari gambar diatas, AVR menggunakan arsitektur Harvard dengan memisahkan antara memori dan *bus* untuk program dan data untuk memaksimalkan kemampuan dan kecepatan. Instruksi dalam memori program dieksekusi dengan *pipelining single level*. Dimana ketika satu instruksi dieksekusi, instruksi berikutnya diambil dari memori program. Konsep ini mengakibatkan instruksi dieksekusi setiap *clock cycle*. CPU terdiri dari 32x8-bit *general purpose register* yang dapat diakses dengan cepat dalam satu *clock cycle*, yang mengakibatkan operasi *Arithmetic*

Logic Unit (ALU) dapat dilakukan dalam satu *cycle*. Pada operasi ALU, dua *operand* berasal dari *register*, kemudian operasi dieksekusi dan hasilnya disimpan kembali pada *register* dalam satu *clock cycle*. Operasi aritmatik dan *logic* pada ALU akan mengubah bit-bit yang terdapat pada *status register* (SREG). Proses pengambilan instruksi dan pengeksekusian instruksi berjalan secara *parallel*, dapat dilihat pada gambar di bawah ini.



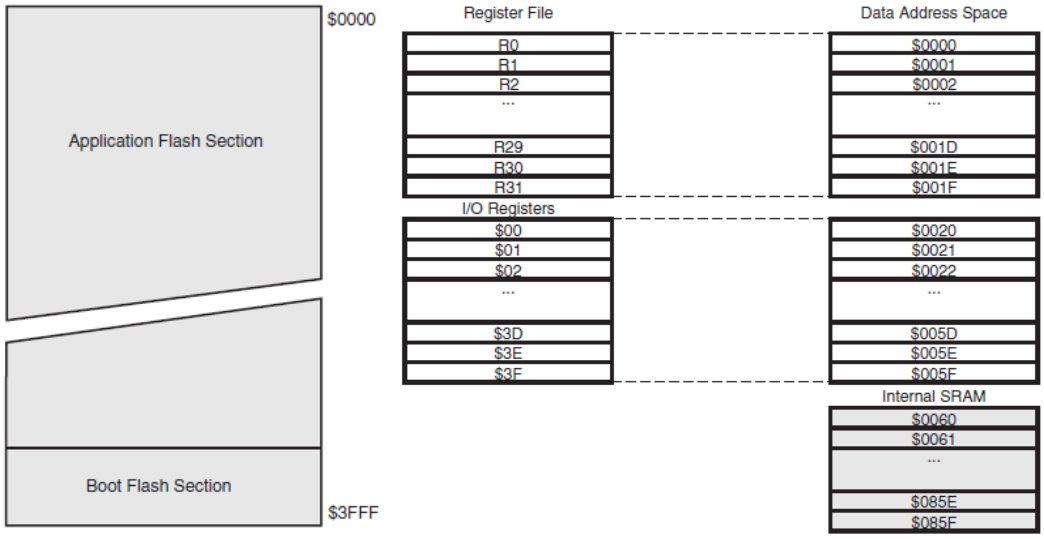
Gambar 2.11 Proses Pengambilan dan Pengeksekusian Instruksi Secara Paralel

2.4.4 Struktur Memori

Arsitektur AVR mempunyai dua memori utama, yaitu memori data dan memori program. Selain itu, ATMEGA 32 memiliki EEPROM untuk menyimpan data. ATMEGA 32 memiliki 32 Kbyte *on-chip in system reprogrammable flash memory* untuk menyimpan data memori dengan alamat 000H – FFFH.

Memori data terdiri dari 3 bagian :

- 32 buah *general purpose register* (GPR) / register umum
- 64 buah register I/O
- 2 Kbyte SRAM internal



(a) (b)

Gambar 2.12 (a) Memori Program, (b) Memori Data

2.4.5 Register I/O

Mikrokontroler ATMEGA 32 mempunyai 64 register I/O. Register Input Output ini berfungsi untuk mengatur dan melakukan kontrol *peripheral* mikrokontroler, contoh dari register I/O :

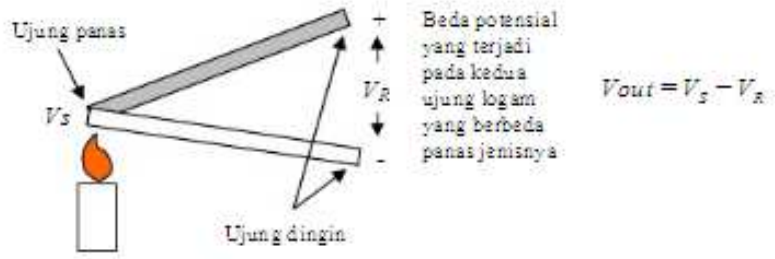
Tabel 2.3 Tabel Register I/O

Address	Nama	Fungsi
$3F	SREG	Status Register
$3E	SPH	Stack Pointer High
$3D	SPL	Stack Pointer Low
$3C	OCR0	Output Compare Register (0)
$03	TWDR	TW1 Data Register
$02	TWAR	TW1 Address Register
$01	TWSR	TW1 Status Register
$00	TWBR	TW1 Bit Rate Register

2.5 Termokopel

Termokopel merupakan suatu sensor suhu dimana terbentuk dari jenis logam yang berbeda disatukan salah satu ujungnya dan ujung tersebut dipanaskan, sehingga menimbulkan beda potensial pada ujung-ujung yang lain. Hal ini diakibatkan oleh kecepatan gerak elektron dari dua material yang berbeda daya hantar panas sehingga mengakibatkan beda potensial.

Termokopel dibangun berdasarkan asas *Seeback* dimana bila dua jenis logam yang berlainan disambungkan akan menjadi rangkaian tertutup sehingga perbedaan temperatur pada sambungan akan menimbulkan beda potensial listrik pada kedua logam tersebut.



Gambar 2.12 Beda Potensial Pada Termokopel

Terdapat beberapa tipe termokopel berdasarkan penggunaannya, antara lain sebagai berikut :

1. Tipe K (*Chromel (*Ni-Cr alloy)) / *Alumel* (Ni-Al alloy)

   Merupakan termokopel untuk tujuan umum, lebih murah, dan memiliki range suhu -200^{o}-1200^{o}C.

2. Tipe E (*Chromel* / *Constantan* (Cu-Ni alloy))

   Memiliki output yang besar (68 uV/^{o}C) membuatnya cocok digunakan pada temperatur rendah, dan merupakan tipe non magnetik.

3. Tipe J (*Iron* / *Constantan*)

   Rentangnya terbatas (-40^{o}-750^{o}C) membuatnya kurang popular dibandingkan dengan tipe K. Memiliki sensitivitas sekitar ~52 uV/^{o}C.

4. Tipe N (*Nicrosil* (Ni-Cr-Si alloy) / *Nisil* (Ni-Si alloy))

   Stabil dan tahanan yang tinggi terhadap oksidasi membuat tipe N cocok untuk pengukuran suhu yang tinggi tanpa platinum. Dapat mengukur suhu di atas 1200^{o}C. Sensitifitasnya sekitar 39 uV/^{o}C pada 900^{o}C, sedikit dibawah tipe K dan merupakan perbaikan dari tipe K.

5. Tipe B (*Platinum-Rhodium* / Pt-Rh)

   Memiliki sensitivitas sekitar 10 uV/^{o}C. Tipe B member output yang sama pada suhu 0^{o}C hingga 42^{o}C.

6. Tipe R (*Platinum* / *Platinum with* 7% *Rhodium*)

   Dapat mengukur suhu diatas 1600^{o}C, sesitivitasnya sekitar 10 uV/^{o}C. dan biayanya sangat tinggi.

7. Tipe S (*Platinum* / *Platinum with* 10% *Rhodium*)

   Memiliki kesamaan dengan tipe R. Stabillitasnya yang tinggi, digunakan untuk standar pengukuran titik leleh emas.

8. Tipe T (*Copper* / *Constantan*)

   Cocok untuk pengukuran antara -200^{o} - 350^{o}C. Sering dipakai sebagai alat pengukur alternative sejak penelitian kawat tembaga. Memiliki sensitivitas ~43 uV/^{o}C.

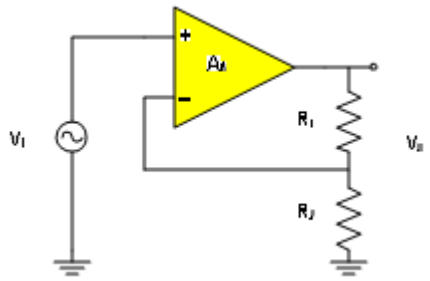
Gambar 2.13 Termokopel Seri-K

2.6 *Operational Amplifier* LM 358

Op-Amp adalah piranti yang berfungsi mengindera dan memperkuat sinyal masukan baik DC maupun AC. Op-Amp terdiri atas tiga rangkaian dasar, yaitu penguat diferensial impedansi masukan tinggi, penguat tegangan penguatan tinggi, dan penguat keluaran impedansi rendah. Karakteristik Op-Amp yang terpenting adalah :

1. Impedansi masukan sangat tinggi
2. Penguatan loop terbuka sangat tinggi
3. Impedansi keluaran amat rendah, sehingga keluaran penguat tidak terpengaruh pembebanan

Op-Amp fungsi umumnya adalah sebagai penguat, tetapi dengan penambahan beberapa komponen pasif eksternal dapat digunakan untuk berbagai macam aplikasi. Salah satu aplikasi Op-Amp sebagai penguat adalah penguat *non-inverting.* Rangkaian penguat *non-inverting* ditunjukkan pada gambar berikut ini.



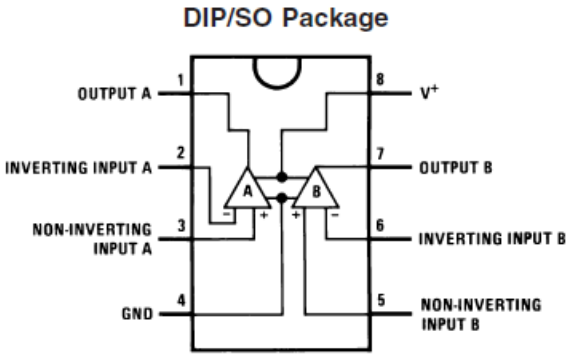
Gambar 2.14 Panguat *Non-inverting*

Hubungan tegangan masukan dan keluaran diperlihatkan persamaan di bawah ini :

Vo = A·Vi

$$Vo = \left(1 + \frac{R_2}{R_1}\right) \cdot Vi \qquad (2.23)$$

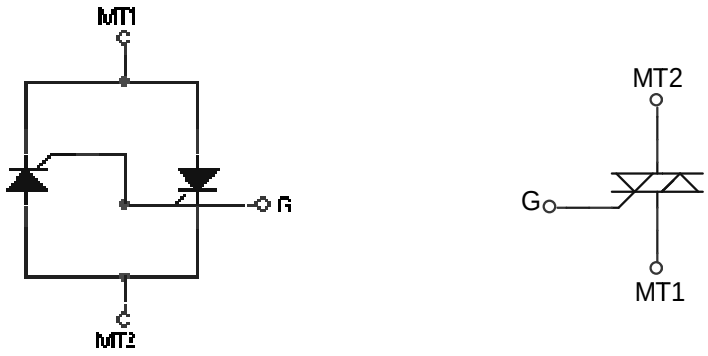
LM 358 pada kasus ini, akan digunakan sebagai penguat *non-inverting*. Pemilihan pada IC ini, dikarenakan memiliki keunggulan dalam fungsinya sebagai penguat, murah, dan sangat mudah ditemukan di pasaran. Berikut adalah konfigurasi pin dari IC LM 385.



Gambar 2.15 Konfigurasi Pin LM 358

2.7 TRIAC BT 139

SCR (*Sillicon Controlled Rectifier*) dan TRIAC (*Triode Alternating Current*) adalah *thyristor* yang paling sering digunakan. TRIAC dapat bersifat kondiktif dalam dua arah. Dalam hal ini dapat dianggap sebagai dua buah *thyristor* yang terhubung *invers parallel* dengan kondisi gerbang seperti ditunjukkan pada gambar 2. (a). TRIAC mempunyai tiga terminal; terminal utama 2 (MT2), terminal utama 1 atau (MT1) dan gerbang (G). Gambar 2. (b) menunjukkan simbol TRIAC.



(a) (b)

Gambar 2.16 (a) Rangkaian Ekivalen TRIAC, (b) Simbol TRIAC

TRIAC BT 139 yang digunakan memiliki karakteristik sebagai berikut :

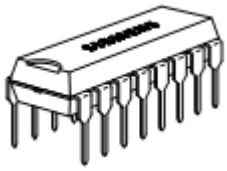
Tabel 2.4 Tabel Karakteristik BT 139

SYMBOL	PARAMETER	MAX.	MAX.	MAX.	UNIT
	BT139- BT139- BT139-	**500 500F 500G**	**600 600F 600G**	**800 800F 800G**	
V_{DRM}	Repetitive peak off-state voltages	500	600	800	V
$I_{T(RMS)}$	RMS on-state current	16	16	16	A
I_{TSM}	Non-repetitive peak on-state current	140	140	140	A

Dari tabel 2.3 dapat diketahui bahwa kemampuan TRIAC BT 139 dapat mengontrol hingga teganga 800 Vac pada V_{DRM}, sedangkan arus maksimum yang bias dilewatkan sebesar 16 Ampere pada $I_{T(RMS)}$.

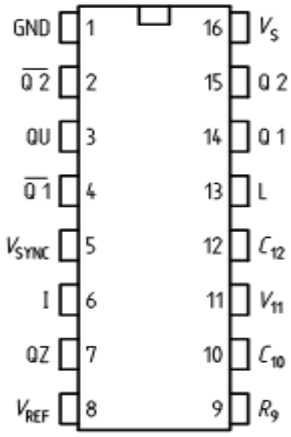
2.8 IC TCA 785

IC TCA 785 merupakan produk dari *Siemen Semiconductor* yang dibuat untuk menghasilkan pulsa pemicu (*trigger pulse*) untuk mengontrol fasa pada SCR, triac, dan transistor, antara 0^o – 180^o pada sumber tegangan ac, sedangkan jika sumber tegangannya dc maka diperlukan komutasi khusus.



Gambar 2.17 Bentuk Fisik IC TCA 785

Gambar diatas, adalah bentuk fisik IC TCA 785. IC ini dapat diaplikasikan pada kontrol tegangan ac terkontrol (*ac-ac converter*) satu fasa dan tiga fasa, penyearah terkontrol (*control rectifier*) satu fasa maupun tiga fasa dan kontrol tegangan dc terkontrol (*dc chopper*). IC ini memiliki kaki (pin) sejumlah 16 buah, seperti yang terlihat pada gambar berikut.



Gambar 2.18 Konfigurasi pin IC TCA 785

Berikut adalah fungsi dari tiap pin IC TCA 785 :

Tabel 2.5 Deskripsi Fungsi pin IC TCA 785

Pin	Symbol	Function
1	GND	Ground
2	$\overline{Q2}$	Output 2 inverted
3	Q U	Output U
4	$\overline{Q2}$	Output 1 inverted
5	V_{SYNC}	Synchronous voltage
6	I	Inhibit
7	Q Z	Output Z
8	V_{REF}	Stabilized voltage
9	R_9	Ramp resistance
10	C_{10}	Ramp capacitance
11	V_{11}	Control voltage
12	C_{12}	Pulse extension
13	L	Long pulse
14	Q 1	Output 1
15	Q 2	Output 2
16	V_S	Supply voltage

IC TCA 785 memerlukan sumber tegangan antara 8 volt hingga 18 volt, frekuensi kerja 10 Hz hingga 500 Hz, serta temperature kerja antara -250^o hingga 85^{o}C. Prinsip kerja IC ini adalah sinyal sinkronisasi dari tegangan sumber dihubungkan pada kaki nomor 5 (V_{SYNC}) melalui resistor

berhambatan tinggi. Peraba nol (*zero detector*) akan menentukan letak titik nol dan disimpan dalam memori sinkron. *Detector* ini kemudian akan mengendalikan generator gelombang tegangan gigi gergaji yang sesuai dengan frekuensi sumber tegangan. Kapasitor C_{10} dan resistor R_9 akan menentukan kemiringan dari gelombang gigi gergaji yang dihasilkan. Nilai kapasitansi kapasitor antara 500 pF sampai 1 uF dan R_9 dapat diperoleh dari resistor dengan resistansi 3 kΩ sampai 100 kΩ. Gelombang ini kemudian dibandingkan dengan tegangan referensi V_{11} oleh *comparator* (pembanding). Sinyal *output* dari *comparator* ini kemudian akan diteruskan ke rangkaian logika. Bila tegangan referensi pada kaki 11 (V_{11}) pada posisi terendah, maka sudut penyulutan akan menunjukkan α = 0°. Sehingga untuk mengatur pemicuan dapat dilakukan memvariasi tegangan yang masuk pada kaki 11 (tegangan kontrol). Persamaan yang dapat digunakan untuk mendapatkan sudut pemicuan adalah :

$$\alpha = \frac{V_{CONTROL}}{V_{SAT}} x\, 180° \qquad (2.24)$$

dimana :

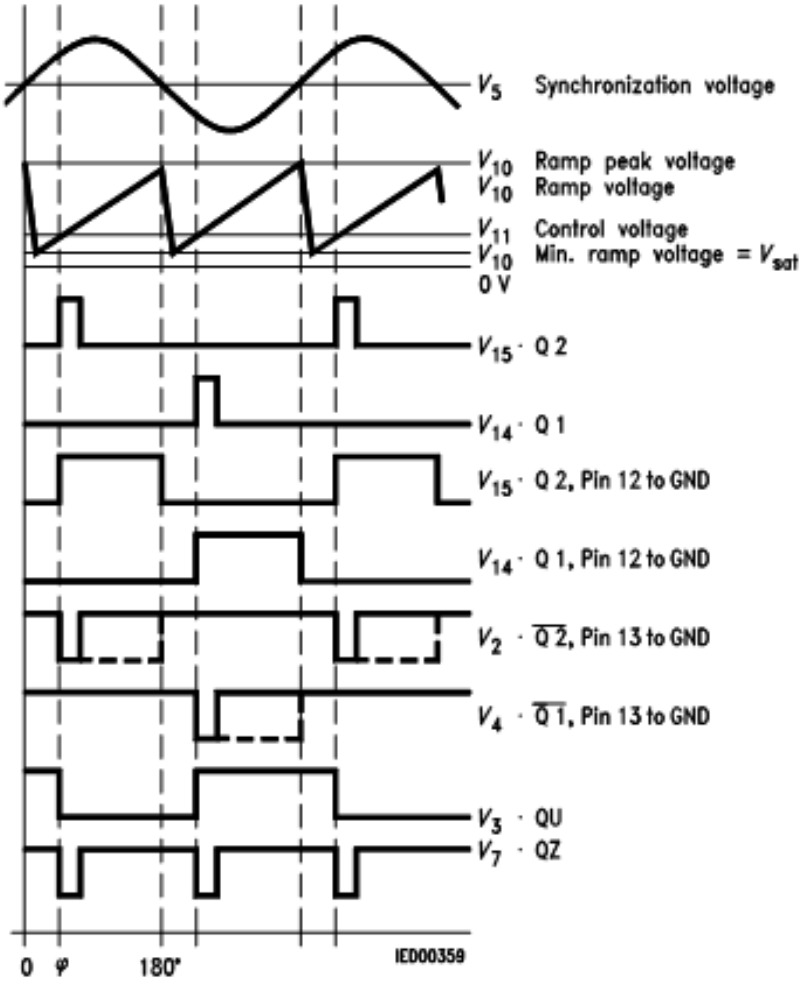
α = Sudut pemicuan (^{o})

$V_{CONTROL}$ = Tegangan kontrol, V_{11} (Volt)

V_{SAT} = Tegangan pada pin 10, V_{10} (Volt)

V_{sat} (V_{10}), biasanya bernilai 10 V, namun lebih tepatnya pengaturan sudut picu dapat dilakukan dengan menggunakan *oscilloscope,* sehingga dapat diperoleh pengukuran yang lebih baik. Pada IC ini, jika V_{sat} (V_{10}) melebihi tegangan kontrol V_{11}, makan sebuah sinyal akan diproses dalam

logika. Proses ini dapat dilakukan berdasarkan besarnya tegangan kontrol V_{11} sehingga didapatkan sudut α antara 0^o – 180^o. keluaran dari IC ini adalah pada kaki 14 (negatif) dan 15 (positif). Diagram pulsa dari proses pembangkitan sinyal ini dapat dilihat pada gambar berikut.

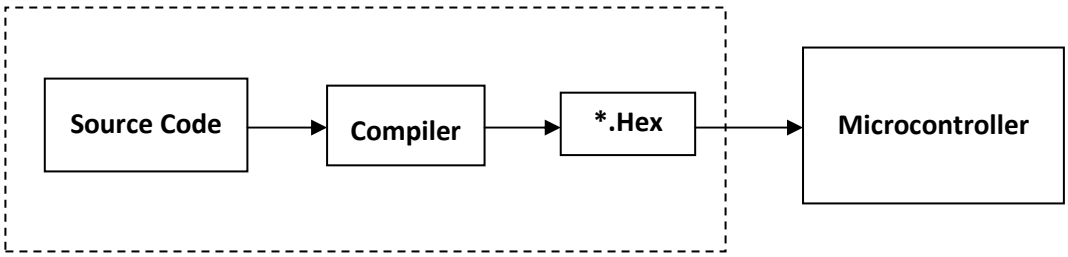


Gambar 2.19 Diagram Pulsa IC TCA 785

2.9 BASCOM AVR 1.11.9.5

Secara umum, bahasa yang digunakan untuk pemrograman mikrokontroler adalah bahasa tingkat rendah, yaitu bahasa *assembly*. Setiap mikrokontroler memiliki bahasa-bahasa pemrograman yang berbeda. Karena banyaknya hambatan dalam penggunaan bahasa assembly, banyak berkembang compiler atau penerjemah untuk bahasa tingkat tinggi. Bahasa tingkat tinggi yang banyak dikembangkan antara lain BASIC, PASCAL, dan bahasa C.

Proses pemrograman mikrokontroler diawali dengan menulis program sumber (*source code*) baik dalam bahasa *assembly*, C maupun *basic*. *Source code* kemudian di-*compile* dan akan menghasilkan kode-kode yang dapat dimengerti oleh mikrokontroler (*format *.hex*).



Gambar 2.20 Proses Pemrograman Mikrokontroler

2.9.1 Tipe Data

Tipe data berhubungan dengan variabel atau konstanta yang akan menunjukkan daya tamping dari variabel atau konstanta tersebut. Tipe data di BASCOM adalah sebagai berikut :

Tabel 2.6 Tipe Data Bascom AVR

Tipe Data	Ukuran (Byte)	Jangkauan
Bit	1/8	0 atau 1
Byte	1	0 s/d 255
Integer	2	-32.768 s/d 32.767
Word	2	0 s/d 65535
Long	4	-2147483648 s/d 2147483647
Single	4	$1.5x10^{-45}$ s/d $3.4x10^{38}$
Double	8	$5x10^{-324}$ s/d $1.7x10^{308}$
String	s/d 254	

2.9.2 Variabel

Variabel digunakan untuk menyimpan data sementara. Variabel diberi nama dan dideklarasikan terlebih dahulu sebelum digunakan. Aturan pemberian nama variabel sebagai berikut :

1. Nama variabel maksimum terdiri atas 32 karakter
2. Karakter biasa berupa angka atau huruf
3. Nama variabel harus dimulai dengan huruf
4. Variabel tidak boleh menggunakan kata-kata yang digunakan oleh BASCOM sebagai perintah, pernyataan, internal register, dan nama operator (AND, OR, DIM, dan lain-lain)

Deklarasi sangat diperlukan bila akan menggunakan pengenal (identifier) dalam suatu program. Variabel dapat dideklarasikan dengan cara :

Dim <Nama Variabel> As <Tipe Data>

Contoh :

Dim angka **As** Integer ‘angka sebagai variabel dengan tipe data integer

2.9.3 Konstanta

Berbeda dengan variabel, sebuah konstanta akan bernilai tetap. Sebelum digunakan, konstanta dideklarasikan terlebih dahulu dengan cara:

Dim nama_konstanta **As const** nilai_konstanta

Atau

Const nama_konstanta = nilai_konstanta

Contoh :

Dim pembagi **as const** 23

Const pembagi = 23

2.9.4 Penulisan Bilangan

Pada BASCOM AVR, bilangan ditulis dalam 3 bentuk :

1. Desimal ditulis biasa, contoh : 16
2. Biner diawali dengan &B, contoh : &B10001111
3. Heksadesimal diawali dengan &H, contoh : &H8F

2.9.5 Alias

Untuk mempermudah pemrograman, biasanya nama register dalam mikrokontroler dibuatkan nama yang identik dengan *hardware* yang dibuat. Contoh :

LED_1 **alias** PORTC.0 ‘nama lain dari PORTC.0 adalah LED_1

2.9.6 Array

Array atau larik merupakan sekumpulan variabel dengan nama dan tipe data yang sama, yang berbeda indeks keanggotaannya. Cara mendeklarasikan array sebagai berikut :

Dim nama_array (jumlah_anggota) **as** tipe_data

Contoh :

Dim voltage (5) **as** byte ‘variabel voltage dengan tipe data byte mempunyai 5 anggota

2.9.7 Operator

BASCOM AVR menyediakan beberapa operator untuk pengolahan data.

1. Operator aritmatik

Tabel 2.7 Operator Aritmatik di Bascom AVR

Operator	Keterangan
+	Operasi penjumlahan
-	Operasi pengurangan
*	Operasi perkalian
/	Operasi pembagian
%	Operasi sisa pembagian

2. Operator relasional

Tabel 2.8 Operator Relasional di Bascom AVR

Operator	Keterangan	Contoh
AND	Operasi AND	&B110 And &B101 hasilnya &B100
OR	Operasi OR	&B11001 Or &B10111 hasilnya &B11111
NOT	Operasi NOT	NOT &HFF hasilnya &H0
XOR	Operasi XOR	&B Xor &B111 hasilnya &B1110

2.9.8 Struktur pemilihan

1. **IF-THEN**

   Merupakan pernyataan untuk menguji apakah kondisi bernilai benar atau salah untuk melakukan sebuah instruksi. Syntax penulisannya sebagai berikut :

   If <kondisi> **Then** <Perintah>
   If <kondisi> **Then**
   <perintah1>
   <perintah2>
   ……..
   End If

2. **IF-THEN-ELSE**

   Untuk keadaan dimana kedua kondisi (benar atau salah) tetap dikenal perintah. Syntax penulisannya sebagai berikut :

   If <kondisi> **Then**
   <perintah1>
   Else
   <perintah2>
   End if

3. **IF-THEN-ELSEIF**

   Digunakan ketika terdapat lebih dari satu pengujian kondisi. Syntax penulisannya sebagai berikut :

   If <kondisi> **Then**
   <perintah1>
   Else if <kondisi2> **Then**
   <perintah2>
   Else if <kondisi3> **Then**
   <perintah3>
   ……
   ……
   End if

4. SELECT-CASE

Untuk menangani pengujian kondisi yang banyak, maka akan lebih sederhana menggunakan Select-Case. Cara penulisannya sebagai berikut :

Select case <variabel>
Case 1 : <perintah1>
Case 2 : <perintah2>
…….
End select

2.9.9 Struktur perulangan

1. FOR-NEXT

Perintah ini digunakan untuk melaksanakan perintah secara berulang sesuai dengan jumlah dan tingkat perulangannya, syntax penulisannya:

For <variabel=nilai awal> **To** <nilai akhir> <step pertambahan>

<pernyataannya>

Next

2. DO-LOOP

Pernyataan ini untuk melakukan perulangan selama kondisi yang diinginkan terpenuhi, cara penulisannya :

Do

<pernyataan>

Loop <kondisi>

Jika perulangan yang dilakukan terbatas, sesuai kondisi yang diinginkan, maka :

Do
    <pernyataan>
Loop until <kondisi>

3. **WHILE-WEND**

   Bentuk perulangan ini akan melakukan perulangan jika sebuah syarat kondisi terpenuhi. Syntax penulisannya :

   While <kondisi>
       <perintah>
   Wend

2.9.10 Struktur lompatan

1. **GOSUB**

   Perintah ini akan melakukan lompatan ke label yang ditunjuk, biasanya untuk mengerjakan sebuah rutin perintah, kemudian kembali lagi setelah rutin perintah tersebut selesai dikerjakan. Rutin yang harus dituliskan perintah **Return** pada akhir pernyataan.

2. **GOTO**

   Perintah ini untuk melakukan lompatan ke label untuk melakukan instruksi tanpa kembali lagi, sehingga tidak perlu lagi **Return**.

3. **EXIT**

   Untuk keluar secara langsung dari perulangan Do-Loop, For-Next, While-Wend. Syntax penulisannya :

   EXIT FOR
   EXIT WHILE
   EXIT FUNCTION