

BAB II

TEORI PENUNJANG

Pada bab ini akan dibahas mengenai teori dan komponen penunjang yang akan digunakan dalam perancangan “Implementasi Sistem Kontrol pada Jemuran Otomatis”.

2.1 Sistem Kendali

Sistem Kendali atau *control system* terdiri dari dua kata yaitu *system* dan *control*. *System* berasal dari Bahasa Latin (*systema*) dan bahasa Yunani (*sustema*) adalah suatu kesatuan yang terdiri dari komponen atau elemen yang dihubungkan bersama untuk mencapai suatu tujuan tertentu. *Control* itu memiliki arti mengatur, mengarah dan mengendalikan. Jadi *system control* adalah hubungan antara komponen-komponen fisik yang membentuk suatu konfigurasi sistem sehingga memberikan hasil yang diharapkan atau dapat diperaktekan secara manual maupun dalam sistem yang otomatis. Hubungan antara *input* dan *output* pada sistem menunjukkan adanya hubungan sebab akibat dari sebuah proses, yang berawal dari *input* sampai bisa menghasilkan *output*. Maksud dari sistem kendali adalah menetapkan atau mendefinisikan *input* dan *output*. Jika *input* dan *output* telah ditentukan, maka memungkinkan untuk menetapkan atau mendefinisikan sifat dari komponen-komponen sistem tersebut.

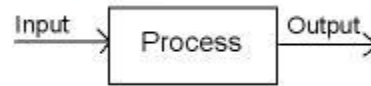
2.1.1 Jenis-Jenis Sistem Kendali

Jenis-jenis sistem kendali terdiri dari 2 macam yaitu sistem *open loop* dan sistem *close loop*.

2.1.1.1 Sistem Open Loop

Sistem kendali *open loop* adalah sebuah sistem yang tidak memiliki umpan balik (*feedback*), sehingga bila terdapat gangguan dari dalam maupun dari luar maka sistem tidak dapat melaksanakan tugas seperti yang diharapkan. Suatu

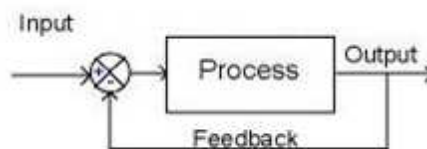
sinyal masukan diberikan ke dalam sistem kendali dimana keluarannya bertindak sebagai sinyal penggerak dimana sinyal penggerak ini yang kemudian menghasilkan proses yang akan dikendalikan untuk menghasilkan output yang diinginkan. Proses *open loop* dapat dilihat pada gambar 2.1.



Gambar 2.1 Sistem Open Loop

2.1.1.2 Sistem Close Loop

Sistem kendali *close loop* adalah sistem kontrol yang memiliki *feedback*, berbeda dengan sistem *open loop*. Pada bagian *output* dari sistem kontrol ini akan dikirim kembali untuk dibandingkan dengan *input* yang diberikan. Bila masih terdapat selisih antara *output* dan *input*, maka sistem masih memiliki *error*. *Error* merupakan selisih antara *input* dan *output* atau sistem belum mencapai hasil yang diinginkan. Proses kerja dari sistem *close loop* dapat dilihat pada gambar 2.2.

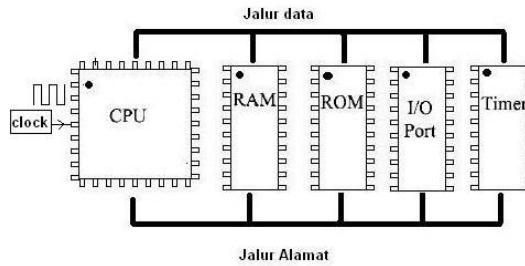


Gambar 2.2 Sistem Close Loop

2.2 Perangkat Keras

2.2.1 Mikrokontroler

Mikrokontroler merupakan rangkaian terintegrasi yang berisi komponen-komponen yang diperlukan oleh sebuah komputer seperti CPU, I/O, jalur komunikasi, memori, *timer* dan lain sebagainya. Mikrokontroler dapat diberi sebuah program yang bekerja sesuai dengan keinginan pengguna dan dapat dihapus dengan cara khusus.



Gambar 2.3 Ilustrasi mikrokontroler

Beberapa fitur yang terdapat pada mikrokontroler adalah:

1. CPU (*Central Processing Unit*)

CPU adalah suatu unit pengolahan pusat yang terdiri dari 2 bagian, yaitu unit pengendali (*Control Unit*) dan logika (*Arithmetic Logic Unit*). Disamping itu, CPU mempunyai beberapa simpanan yang berukuran kecil yang disebut dengan *register*. Adapaun fungsi utama dari unit pengendali ini adalah mengatur dan mengendalikan semua peralatan yang ada pada sistem dan juga dapat mengatur kapan alat *input* menerima data dan kapan data diolah serta ditampilkan pada alat *output*. Sedangkan unit logika berfungsi untuk melakukan semua perhitungan aritmetika yang terjadi sesuai dengan instruksi program dan dapat juga melakukan keputusan dari operasi logika atau pengambilan keputusan sesuai dengan instruksi yang diberikan.

2. Bus Alamat

Bus alamat berfungsi sebagai sejumlah lintasan saluran pengalamatan. Pengalamatan ini harus ditentukan terlebih dahulu untuk menghindari terjadinya kesalahan pengiriman sebuah instruksi dan terjadinya tabrakan antara dua buah alat yang bekerja secara bersamaan.

3. Bus data

Bus data merupakan lintasan saluran keluaran masuknya data dalam suatu mikrokontroler. Umumnya saluran data yang masuk sama dengan saluran data yang keluar.

4. Bus control

Bus kontrol atau bus kendali berfungsi untuk mengintegrasikan operasi mikrokontroler dengan operasi rangkaian luar.

5. Memori

Memori berfungsi untuk menyimpan data atau program. Ada beberapa jenis memori, diantaranya adalah ROM (*Read Only Memory*) dan RAM (*Random Access Memory*) serta berdasarkan tingkatannya, memori terbagi menjadi *register* internal, memori utama dan memori masal. *Register* internal adalah memori yang terdapat didalam ALU (*Arithmetic Logic Unit*). Memori utama adalah memori yang ada pada suatu sistem, waktu aksesnya lebih lambat dibandingkan dengan *register* internal. Sedangkan memori masal dipakai untuk menyimpan data yang berkapasitas tinggi, yang biasanya berbentuk disket, pita magnetik atau kaset.

6. RAM (Random Access Memory)

RAM adalah memori yang dapat dibaca atau ditulis. Data dalam RAM bersifat *volatile*, dimana isinya akan hilang begitu IC kehilangan catu daya. Karena bersifat yang demikian, RAM hanya digunakan untuk menyimpan data pada saat program bekerja.

7. ROM (Read Only Memory)

ROM merupakan memori yang hanya dapat dibaca, dimana isinya tidak dapat berubah apabila IC telah kehilangan catu daya. ROM dipakai untuk menyimpan program, pada saat direset maka mikrokontroler akan langsung bekerja dengan program yang terdapat didalam ROM tersebut. Ada berbagai jenis ROM antara lain ROM murni, PROM (*Programmable Read Only Memory*), EPROM (*Erasable Programmable Read Only Memory*), yang paling banyak digunakan diantara tipe-tipe diatas adalah EPROM yang dapat deprogram ulang dan dapat juga dihapus dengan sinar ultraviolet.

8. Perangkat Input/output

Perangkat *input* dan *output* mikrokontroler adalah suatu peranti yang menghubungkan proses didalam mikrokontroler dengan dunia luar (rangkaian lain), peranti ini dibutuhkan sebagai media komunikasi dengan perangkat lain atau peubah tipe sinyal. Contoh perangkat I/O antara lain:

1. USART (*Universal Synchronous Asynchronous Receiver Transmitter*) merupakan adapter yang digunakan sebagai komunikasi *serial sinkron* dan *asinkron*.
2. SPI (*Serial Peripheral Interface*) merupakan *port* komunikasi *serial sinkron*.
3. I2C *bus* (*Inter-Itergrated Circuit Bus*) merupakan antarmuka *serial bus* yang dikembangkan oleh *philihps*.
4. *Analog to Digital Conversion* (ADC) adalah rangkaian yang digunakan untuk mengubah data *analog* ke data *digital*.
5. *Digital to Analog* (DAC) adalah rangkaian untuk mengubah data digital ke data *analog*.
6. *Analog Comparator* adalah rangkaian komparator tegangan, saat ini peranti ini telah terintegrasi dalam *port* mikrokontroler.
7. USB *Converter* merupakan peranti yang memungkinkan komunikasi *serial* dengan karakteristik format komunikasi USB.

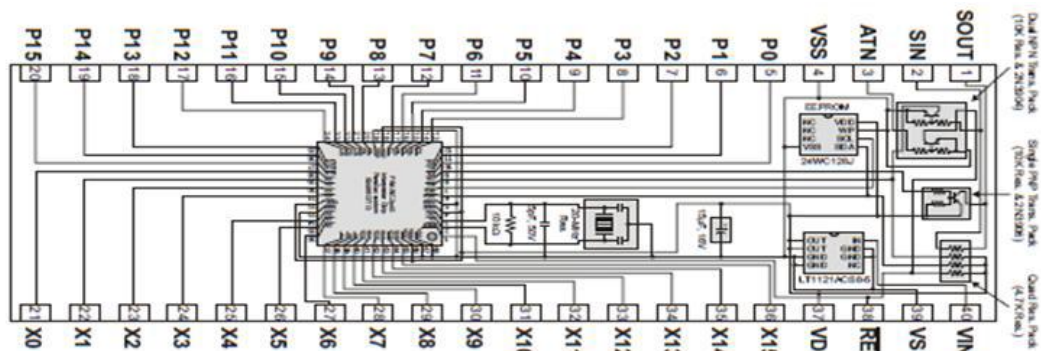
9. Clock

Clock atau pewaktu berfungsi memberikan referensi waktu dan sinkronisasi antar eleman.

Basic Stamp adalah suatu mikrokontroler yang dikembangkan oleh *Parallax Inc* yang diprogram menggunakan bahasa pemrograman *basic* dan populer sekitar pada tahun 1990an. Mikrokontroler BS2P40 ini mampu menampung hingga 4000 instruksi (8x2 Kbyte EEPROM) dengan kecepatan prosesor 20 MHz *turbo* dimana kecepatan eksekusi program hingga 12000 instruksi per detik. Program yang dibuat di-*download* melalui port serial dengan menggunakan *converter USB to serial* untuk computer yang tidak memiliki *port*

serial, serta membutuhkan *power supply* saat men-*download* program dengan besar tegangan yang diperlukan mikrokontroler ini untuk bekerja adalah 9-12 VDC dengan tegangan output 5 VDC.

Beberapa macam versi dari Basic Stamp yaitu *Basic Stamp1*, *Basic Stamp2*, *Basic Stamp 1e*, *Basic Stamp 2P*, *Basic Stamp 2Pe* dan *Basic Stamp 2sx*.



Gambar 2.4 Alokasi Pin Basic Stamp

Diskripsi pin *Basic Stamp BS2P40*:

Table 2.1 Diskripsi Pin BS2P40

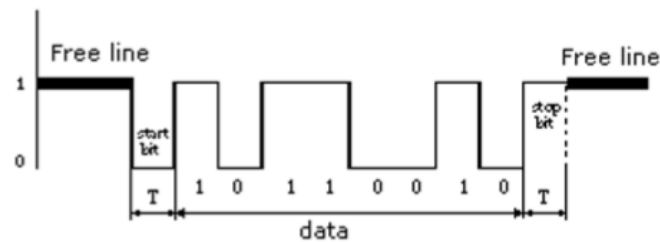
Pin	Nama	Keterangan
1	SOUT	<i>Serial out</i> untuk pemrograman yang terkoneksi ke PC pada port di PIN RX (DB9 PIN 2 / DB25 PIN 3)
2	SIN	<i>Serial input</i> untuk pemrograman yang terkoneksi ke PC pada port di PIN TX (DB9 PIN 3 / DB25 PIN 2)
3	STN	<i>Serial data</i> untuk pemrograman yang terkoneksi ke PC pada port di PIN DTR (DB9 PIN 4 / DB25 PIN 20)
4	VSS	<i>Serial data</i> untuk pemrograman yang terkoneksi ke PC pada port di PIN DTR (DB9 PIN 5 / DB25 PIN 7)
5-20	P0-P15	PIN I/O dimana logika <i>high</i> = 5V dan <i>low</i> = 0V
21-36	X0-X15	PIN I/O sekunder (<i>Auxiliary</i>) dimana logika <i>high</i> = 5V dan <i>low</i> = 0V
21 & 37	VDD	PIN <i>input</i> tegangan sebesar 5V

22 & 38	RES	PIN <i>reset</i>
23 & 39	VSS	PIN <i>ground</i> mikrokontroler
24 & 40	VIN	PIN <i>input</i> tegangan yang dilewatkan <i>regulator</i> 5V membutuhkan sumber sebesar 5,5 – 12 VDC

2.2.2 Sistem Komunikasi Serial

Port serial adalah *port* yang paling populer digunakan untuk keperluan koneksi ke piranti luar. Kata “Serial”, menggambarkan prinsip kerja *port* ini yang memberikan data. Cara kerjanya adalah diawali dengan mengambil sebuah byte data lalu kemudian mengirimkan perdelapan bit dalam *byte* tersebut satu persatu dalam satu jalur data. Keuntungannya adalah bahwa *port* ini hanya membutuhkan satu kabel untuk mengirimkan kedelapan bit tadi (dibandingkan *port paralel* yang membutuhkan delapan kabel). Keuntungan lainnya adalah efisiensi dalam biaya dan tentunya ukuran kabel yang kecil. Kerugiannya yakni bahwa *port serial* membutuhkan delapan kali lebih lama untuk mengirimkan data dibanding dengan proses pengiriman dengan delapan kabel.

Terdapat dua cara dalam komunikasi data secara *serial*, yaitu komunikasi data *serial* secara *sinkron* dan komunikasi data *serial* secara *asinkron*. Pada komunikasi data *serial sinkron*, *clock* dikirimkan bersama-sama dengan data *serial*, sedangkan pada komunikasi data *asinkron* *clock* tidak dikirimkan bersama data *serial*, tetapi dibangkitkan secara terpisah baik pada bagian pemancar maupun pada bagian penerima. Kecepatan pengiriman data dan *fase clock* pada bagian pemancar dan bagian penerima harus *sinkron*, untuk itu diperlukan *sinkronisasi* antara dua bagian tersebut. Salah satu caranya adalah dengan mengirimkan bit ‘start’ dan bit ‘stop’. Untuk bit ‘start’ adalah data *biner* 0 dan untuk bit ‘stop’ adalah data *biner* 1. Setelah pengiriman bit ‘start’ maka akan diikuti oleh data yang akan dikirim, selanjutnya diakhiri dengan bit ‘stop’. Berikut adalah contoh pengiriman karakter B2 heksa atau 10110010 *biner* tanpa bit paritas. Dapat terlihat pengiriman data diawali dengan bit ‘start’ lalu data B2 heksa dan diakhiri dengan bit ‘stop’ sebagai akhir dari pengiriman dapat dilihat pada gambar 2.5.



Gambar 2.5 Pengiriman Data Serial

Kecepatan pengiriman data (*baud rate*) bervariasi, mulai dari 110, 135, 150, 300, 600, 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200, 230400, 460800 dan 921600 (bit/detik). Pada komunikasi data serial *baud rate* dari kedua bagian harus diatur pada kecepatan yang sama. Setelah itu harus ditentukan panjang datanya, apakah 6, 7 atau 8 bit, juga apakah data disertai dengan paritas genap, paritas ganjil atau tidak menggunakan paritas. Untuk menentukan *baud rate* dapat dilihat pada persamaan di bawah ini:

Misalkan XTAL yang digunakan adalah = 11.0592 MHz

Maka:

$$\text{Machine Cycle Frequency} = \frac{11.0592 \text{ MHz}}{12} = 921.6 \text{ kHz}$$

(*Machine Cycle = 12*)

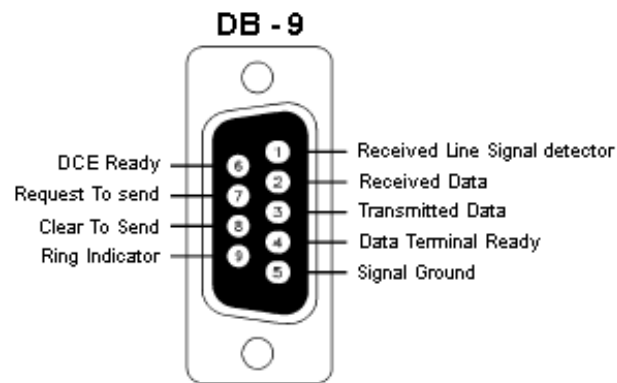
$$\text{Machine Cycle Frequency Mode 1} = \frac{921.6 \text{ kHz}}{16} = 57.600 \text{ Hz}$$

(*Bit Counter Mode 1 = 16*)

$$\text{Baud rate} = \frac{57.600 \text{ Hz}}{3} = 19.200 \text{ Baud rate}$$

($2^n = 8 \text{ bit data}$)

Konektor *port serial* atau yang biasa disebut DB-9 (COM1 dan COM2) dapat dilihat pada bagian belakang komputer (CPU) memiliki kaki sejumlah 9 pin seperti pada gambar 2.6.



Gambar 2.6 Konektor Serial DB 9

Table 2.2 konfigurasi pin dan nama sinyal konektor serial DB9

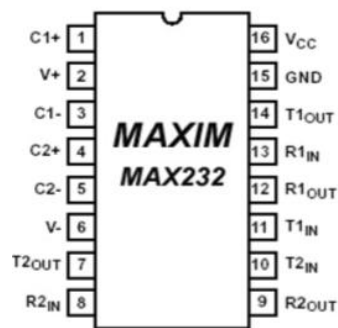
Nama Pin	Nama Sinyal	Direction	Keterangan
1	DCD	In	<i>Data Carrier Detect/Received LineSignal Detect</i>
2	RxD	In	<i>Received Data</i>
3	TxD	Out	<i>Transmite Data</i>
4	DTR	Out	<i>Data Terminal Ready</i>
5	GND	-	<i>Ground</i>
6	DSR	In	<i>Data Set Ready</i>
7	RTS	Out	<i>Request to Sent</i>
8	CTS	In	<i>Clear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

2.2.3 MAX232

Untuk dapat berhubungan dengan PC, mikrokontroler harus membutuhkan komponen tambahan baik komunikasi paralel maupun *serial*. Pada pembuatan tugas akhir ini yang digunakan adalah komunikasi *serial*. Pada mikrokontroler sendiri terdapat *buffer* yang dapat digunakan sebagai pendukung proses komunikasi tersebut. Pada saat ini banyak komponen yang dapat digunakan untuk pendukung proses komunikasi tersebut, salah satu contohnya adalah Maxim232.

Maxim232 berfungsi sebagai perantara antara mikrokontroler dengan *port serial*, karena mikrokontroler tidak dapat mengirim data begitu saja maka diperlukan Maxim232. di dalam IC terdapat *charge pump* yang akan membangkitkan +10 Volt dan -10 Volt dari sumber +5 Volt tunggal dalam IC DIP (Dual in-line Package) 16 pin (8 pin x 2 baris) ini terdapat 2 buah *transmitter* dan dua buah *receiver*. Jadi IC ini berfungsi sebagai perantara karena maxim232 hanya menerima data dari mikrokontroler untuk kemudian dikirim ke pc melalui DB9.

Maxim232 mempunyai 16 kaki yang terdiri untuk keperluan *port serial*, komunikasi mikrokontroler dengan maxim. Letak dari masing-masing *port* diperlihatkan pada gambar 2.7.



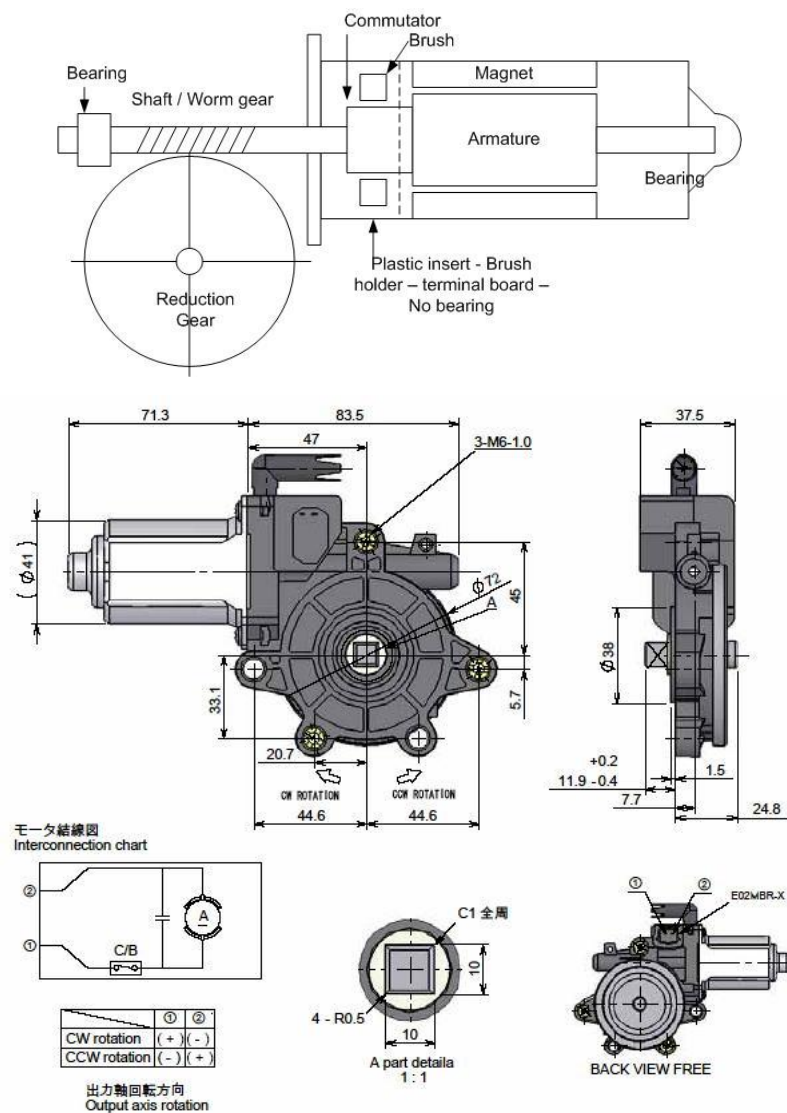
Gambar 2.7 Konfigurasi Pin MAXIM232

Adapun nama dan fungsi dari kaki-kaki pin pada Maxim232 adalah sebagai berikut:

1. VCC (pin 16) : Power Supply
2. GND (pin 15) : Ground
3. T_{1IN} dan R_{1OUT} (pin 11 dan 12) : Pin ini terhubung dengan mikrokontroler
4. R_{1IN} dan T_{1OUT} (pin 13 dan 14) : Pin ini terhubung dengan pin 2 dan 3 DB9
5. C_{1+} dan C_{1-} : Kapasitor 1
6. C_{2+} dan C_{2-} : Kapasitor 2
7. V_+ dan V_- : Tegangan referensi dari Maxim232

2.2.4 Motor DC (Power Window)

Motor DC (power window) adalah suatu motor yang mengubah energi listrik searah menjadi mekanis yang berupa tenaga penggerak torsi. Motor DC digunakan dimana *control* kecepatan dan kecepatan torsi diperlukan untuk memenuhi kebutuhan. Bagian DC yang paling penting adalah *rotor* dan *stator*. Bagian *stator* adalah badan motor, sikat-sikat dan inti kutub magnet. Bagian *rotor* adalah bagian yang berputar dari suatu motor DC. Yang termasuk *rotor* ialah lilitan jangkar, jangkar, komutator, tali, *isolator*, poros, bantalan dan kipas. Jenis motor dc yang dipergunakan dapat dilihat pada gambar 2.8.



Gambar 2.8 Motor DC (Power Window)

Motor power window banyak digunakan karena torsi tinggi dengan rating tegangan input yang rendah yaitu 12 VDC, dan dimensi motor yang relatif simple dilengkapi dengan *internal gearbox* sehingga memudahkan untuk instalasi mekanik.

Prinsip kerja motor DC power window mempunyai bagian *stator* yang berupa magnet permanen dan bagian yang bergerak *rotor* yang berupa koil atau gulungan kawat tembaga. Dimana setiap ujungnya tersambung dengan komutator ini dihubungkan dengan kutub *positif* (+) dan kutub *negative* (-) dari catu daya.

Arus listrik dari kutub *positif* akan masuk melalui komutator, kemudian berjalan mengikuti gulungan kawat sebelumnya, akhirnya masuk ke kutub *negative* dari catu daya. Karena adanya medan elektromagnetik maka motor akan berputar.

Karena putaran *rotor*, arus listrik didalam kawat akan berjalan bolak-balik karena jalanya sesuai dengan medan magnet, maka *rotor* akan selalu berputar terus menerus selama arus listrik tetap mengalir di dalam kawat.

Untuk karakteristik pada motor DC antara lain:

1. **Kutub medan.** Secara sederhana digambarkan bahwa interaksi dua kutub magnet akan menyebabkan perputaran pada motor DC. Motor DC memiliki kutub medan yang *stasioner* dan *dynamo* yang menggerakkan *bearing* pada ruang diantara kutub medan. Motor DC sederhana memiliki dua kutub medan yaitu utara dan selatan. Garis magnetik energi membesar melintasi bukaan diantara kutub-kutub dari utara ke selatan. Untuk motor yang lebih besar atau lebih kompleks terdapat satu atau lebih elektromagnet. Elektromagnet menerima listrik dari sumber daya dari luar sebagai penyedia struktur medan.
2. **Dynamo.** Bila arus masuk menuju *dynamo*, maka arus ini akan menjadi elektromagnet. *Dynamo* yang terbentuk silinder, dihubungkan ke as penggerak untuk mengerjakan beban. Untuk kasus motor DC yang kecil, *dynamo* berputar dalam medan magnet yang terbentuk oleh kutub-kutub, sampai kutub utara dan selatan magnet berganti lokasi. Jika hal ini terjadi, arusnya berbalik untuk merubah kutub-kutub utara dan selatan *dynamo*.
3. **Commutator.** Komponen ini terutama ditemukan dalam motor DC. Kegunaannya adalah untuk membalikan arah arus listrik dalam *dynamo*.

Commutator juga membantu dalam *transmisi* arus antara *dynamo* dan sumber daya. Keuntungan utama motor DC adalah sebagai pengendali kecepatan, yang tidak mempengaruhi kualitas pasokan daya. Motor ini dapat dikendalikan dengan mengatur: tegangan *dynamo* (meningkatkan tegangan *dynamo* yang dimana akan meningkatkan kecepatan), arus medan (menurunkan arus medan akan meningkatkan kecepatan).

2.2.5 Roda Gigi

Roda gigi digunakan untuk mentransmisikan daya besar dan putaran yang tepat. Roda gigi memiliki gigi di sekelilingnya, sehingga penerusan daya dilakukan oleh gigi-gigi kedua roda yang saling berkait. Roda gigi sering digunakan karena dapat meneruskan putaran dan daya yang lebih bervariasi dan lebih kompak dari pada menggunakan alat transmisi yang lainnya, selain itu roda gigi juga memiliki beberapa kelebihan jika dibandingkan dengan alat transmisi lainnya, yaitu :

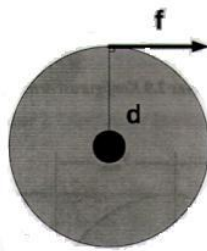
1. Sistem transmisinya lebih ringkas, putaran lebih tinggi dan daya yang besar.
2. Sistem yang kompak sehingga konstruksinya sederhana.
3. Kemampuan menerima beban lebih tinggi.
4. Efisiensi pemindahan dayanya tinggi karena faktor terjadinya slip sangat kecil.
5. Kecepatan transmisi roda gigi dapat ditentukan sehingga dapat digunakan dengan pengukuran yang kecil dan daya yang besar.

Roda gigi harus mempunyai perbandingan kecepatan sudut tetap antara dua poros. Di samping itu terdapat pula roda gigi yang perbandingan kecepatan sudutnya dapat bervariasi. Ada pula roda gigi dengan putaran yang terputus-putus.

Dalam teori, roda gigi pada umumnya dianggap sebagai benda kaku yang hampir tidak mengalami perubahan bentuk dalam jangka waktu lama.

2.2.5.1 Torsi dan Kecepatan

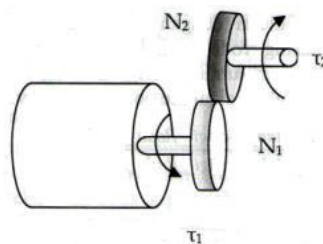
Kecepatan dan torsi adalah dua parameter dasar yang menjadi ukuran bagi suatu motor. Dua hal tersebut dapat ditemui pada *datasheet* dari motor tersebut tetapi seringkali torsi atau kecepatan yang dihasilkan oleh motor tidak memenuhi kebutuhan untuk aktuator yang dipakai. Pada permasalahan seperti inilah peran roda gigi dibutuhkan untuk mengonversi torsi dan kecepatan agar sesuai dengan kebutuhan. Torsi merupakan perkalian dari f gaya (beban) dengan d jari-jari (panjang lengan dari poros).



Gambar 2.9 Torsi

$$torsi_1 \times kecepatan_1 = torsi_2 \times kecepatan_2$$

$Torsi_1$ dan $kecepatan_1$ merupakan parameter *output* dari motor sedangkan $torsi_2$ dan $kecepatan_2$ merupakan parameter *output* dari roda gigi pada poros *output* (biasanya yang terhubung ke roda). Hal yang perlu diperhatikan dalam desain mekaniknya adalah perhitungan kebutuhan torsi untuk menggerakkan sendi atau roda. Salah satu metoda yang paling umum ialah menggunakan sistem *gear* seperti tampak pada gambar di bawah.



Gambar 2.10 Transmisi Hubungan Langsung *Gear* Motor DC

Pada gambar 2.10, N_1 adalah jumlah gigi pada *gear* poros motor, N_2 ialah jumlah gigi pada poros *output*, τ_1 ialah torsi pada poros motor dan τ_2 ialah torsi pada poros *output*.

2.2.5.2 Jenis Roda Gigi

1. *Spur Gears*

Kombinasi roda gigi ini banyak dipakai karena pemasangannya yang mudah dan efisiensinya cukup tinggi. Salah satu bentuk penggunaan yang harus dihindari adalah pada beban berat karena dapat merusak geriginya. Efisiensi yang diberikan oleh kombinasi roda gigi ini berkisar 90%, tergantung *datasheet* komponen.



Gambar 2.11 *Spur Gear*

2. *Helical Gears*

Kombinasi roda gigi ini beroperasi seperti *spur gear* tetapi dengan pergerakan yang lebih lembut. Efisiensi roda gigi ini sebesar 90%, tergantung *datasheet* komponen.

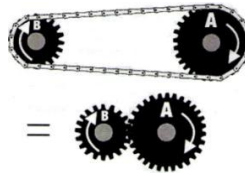


Gambar 2.12 *Helical Gear*

3. *Sprokel Gears*

Kombinasi antara dua buah roda gigi dengan menggunakan rantai atau menggunakan *belt (timing belt)* dapat dianggap sebagai kombinasi antara tiga buah roda gigi. Arah putaran dari salah satu roda gigi selalu sama seperti roda gigi lainnya karena jumlah roda gigi setara dengan tiga.

Kombinasi ini bergerak seperti *spur gear* tetapi mempunyai efisiensi yang sangat rendah karena besarnya area kontak sehingga friksi yang terjadi meningkat, efisiensinya sebesar 80%.



Gambar 2.13 *Sproket Gear*

4. *Bevel Gears*

Kombinasi *bevel* sangat bagus digunakan untuk operasi yang membutuhkan perubahan sudut rotasi hanya saja mempunyai efisiensi yang cukup buruk, yaitu sekitar 70%.



Gambar 2.14 *Bevel Gear*

5. *Rack and Pinion Gears*

Kombinasi roda gigi ini banyak ditemukan dalam sistem pengemudian. Kombinasi roda gigi ini sangat bagus untuk mengubah gerak rotasi menjadi gerak translasi. Efisiensi *gear* ini sebesar 90%.



Gambar 2.15 *Rack and Pinion Gear*

6. *Worm Gears*

Efisiensi kombinasi roda gigi ini cukup rendah, yaitu sekitar 70%. Kombinasi ini mempunyai rasio yang cukup tinggi. Keuntungan lainnya

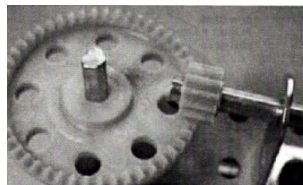
adalah tidak bisa *back-driveable* sehingga yang bisa memutar roda gigi (*worm gear*) adalah *worm* yang terpasang pada motor sehingga hal seperti gravitasi atau gaya lainnya tidak dapat memutar roda gigi. Keuntungan ini hampir mirip motor servo yang biasa digunakan untuk menahan beban pada robot tangan.

7. *Planetary Gears*

Kombinasi roda gigi ini mempunyai rasio roda gigi yang sangat tinggi (tergantung jenis produk) dan mempunyai efisiensi sekitar 80%.

8. *Crown and Pinion Gears*

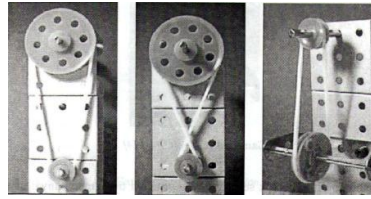
Crown and pinion adalah roda gigi berjenis *reduction gear* dengan posisi poros dari *pinion* (roda gigi yang lebih kecil) tidak searah dengan poros *crown*.



Gambar 2.16 *Crown and Pinion Gear*

9. *Roda Pulley*

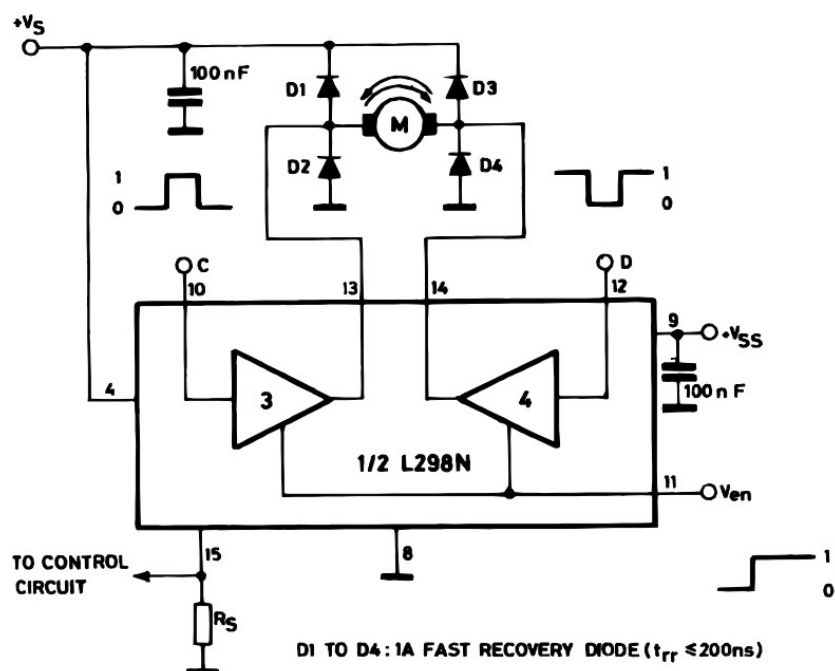
Roda puli berbentuk seperti roda gigi tetapi tidak mempunyai gerigi dengan rongga di sisi luarnya. Fungsinya adalah untuk mentransmisikan gaya pada jarak jauh dan jika diameter antara 2 buah puli berbeda maka fungsinya sama dengan rantai roda gigi. Dua buah puli terhubung satu sama lain dengan menggunakan *belt drive* yang elastis. Hal ini juga didukung dengan keelastisan dari *belt drive* yang membuat puli dapat dihubungkan pada jarak berapapun asalkan *belt drive* tidak slip atau putus. *Belt-drive* dapat terpasang terbalik dan dapat digunakan untuk menghubungkan puli yang mempunyai sudut rotasi yang berbeda sesuai rotasi poros. Kelemahan penggunaan puli yang harus diwaspadai adalah putusnya puli karena beban terlalu berat ataukah slip karena jarak antar puli terlalu dekat.



Gambar 2.17 Konfigurasi *Pulley Wheel Gear*

2.2.6 L298

L298 adalah *dual full bridge driver* yang bertegangan dan berarus tinggi yang dirancang untuk menerima logika TTL tingkat standard dan berfungsi untuk *men-drive* beban induktif seperti relay, solenoid, DC dan motor stepper. Pada IC ini terdapat dua input enable yang dapat mengaktifkan atau me-non aktifkan alat secara *independent* berdasarkan sinyal input. *Emitter* dari transistor pada setiap *bridge* terhubung bersama dan sambungan luar yang bersesuaian dapat digunakan untuk penghubung *external sensing resistor*. Input tenaga tambahan menetapkan supaya pada tegangan lebih rendah, logika tetap bekerja.



Gambar 2.18 Contoh Rangkaian untuk Sebuah Motor Menggunakan L298

Table 2.3 logika L298

Input		Output
$V_{en}=H$	C=H ; D=L	Forward
	C=L ; D=H	Reverse
	C=D	Fast Motor Stop
$V_{en}=L$	C=X ; D=X	Free Running Motor Stop

Keterangan: L = Low , H = High, X = Don't care

2.2.7 LCD (*Liquid Crystal Display*)

Display LCD sebuah *liquid crystal* atau perangkat elektronik yang dapat digunakan untuk menampilkan angka atau teks. JHD 162A yang digunakan adalah modul LCD matrix dengan konfigurasi 16 karakter dan 2 baris dengan setiap karakternya dibentuk oleh 8 baris *pixel* dan 5 kolom *pixel* (1 baris terakhir adalah kursor).

Memori LCD terdiri dari 9.920 bit CGROM, 64 byte CGRAM dan 80x8 bit DDRAM yang diatur pengalamatannya oleh *Address Counter* dan akses datanya (pembacaan maupun penulisan datanya) dilakukan melalui register data.

Konfigurasi pin LCD untuk keperluan antar muka suatu komponen elektronika dengan mikrokontroler, perlu diketahui fungsi dari setiap kaki yang ada pada komponen tersebut.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
VSS	VCC	VEE	RS	R/W	E	DB0	DB1	DB2	DB3	DB4	DB5	DB6	DB7	LED+	LED-

Gambar 2.19 Blok Pin LCD

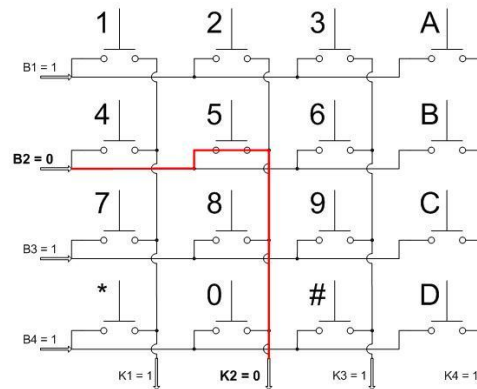
1. Kaki 1 (GND) : Kaki ini berhubungan dengan tegangan 0 volt (*Ground*).
2. Kaki 2 (VCC) : Kaki ini berhubungan dengan tegangan +5 Volt yang merupakan tegangan untuk sumber daya..
3. Kaki 3 (VEE/VLCD) : Tegangan pengatur kontras LCD, kaki ini terhubung pada cermet. Kontras mencapai nilai maksimum pada saat kondisi kaki ini pada tegangan 0 volt.

4. Kaki 4 (RS) : *Register Select*, kaki pemilih register yang akan diakses. Untuk akses ke Register Data, logika dari kaki ini adalah 1 dan untuk akses ke Register Perintah, logika dari kaki ini adalah 0.
5. Kaki 5 (R/W) : Logika 1 pada kaki ini menunjukkan bahwa modul LCD sedang pada mode pembacaan dan logika 0 menunjukkan bahwa modul LCD sedang pada mode penulisan. Untuk aplikasi yang tidak memerlukan pembacaan data pada modul LCD, kaki ini dapat dihubungkan langsung ke *ground*.
6. Kaki 6 (E) : *Enable Clock LCD*, kaki mengaktifkan *clock* LCD. Logika 1 pada kaki ini diberikan pada saat penulisan atau membacaan data.
7. Kaki 7 – 14 (D0 – D7) : Data bus, kedelapan kaki LCD ini adalah bagian di mana aliran data sebanyak 4 bit ataupun 8 bit mengalir saat proses penulisan maupun pembacaan data.
8. Kaki 15 (Anoda) : Berfungsi untuk tegangan positif dari *backlight* LCD sekitar 4,5 volt (hanya terdapat untuk LCD yang memiliki *backlight*).
9. Kaki 16 (Katoda) : Tegangan negatif *backlight* LCD sebesar 0 volt (hanya terdapat pada LCD yang memiliki *backlight*).

2.2.8 Keypad

Keypad matriks adalah tombol-tombol yang disusun secara matriks baris dikali kolom, sehingga dapat mengurangi penggunaan pin *input*. Sebagai contoh keypad matriks 4x4 cukup menggunakan 8 pin untuk 16 tombol.

Proses pengecekan dari tombol yang dirangkai secara matriks adalah teknik *scanning*, yaitu proses pengecekan yang dilakukan dengan cara memberi umpan-data pada satu bagian dengan mengecek *feedback* pada bagian yang lain. Dalam hal ini, pemberian umpan data dilakukan pada bagian baris dan pengecekan *feedback* pada bagian kolom.



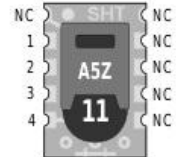
Gambar 2.20 Rangkaian Dasar Keypad 4x4

Cara kerja pada gambar 2.20, tombol yang di tekan adalah tombol “5”.Maka B2 bernilai nol, sedangkan B1, B3, dan B4 adalah satu. Kemudian dengan mengetahui asal data dari B2, dan *feedback* terdeteksi pada k2, maka dapat disimpulkan bahwa tombol yang di tekan adalah tombol “5”.

2.2.9 Sensor SHT11.

SHT11 merupakan multi sensor untuk kelembaban dan suhu secara digital. Sensor tipe SHT11 menggunakan teknologi CMOS yang telah dipatenkan sehingga menjamin kesetabilan dan *reliability* yang tinggi. Dalam chip ini terdiri dari *capacitive palymer sensing element* untuk *relative humidity* sensor dan suhu sensor. Keduanya dihubungkan pada 14 bit ADC (*Analog to digital conversion*) dan *interface serial*, di dalam chip itu sendiri. Output yang dihasilkan berupa kualitas sinyal yang superior, waktu respon yang cepat, tidak sensitif terhadap *external disturbace*. Sensor dapat dilihat pada gambar 2.21.

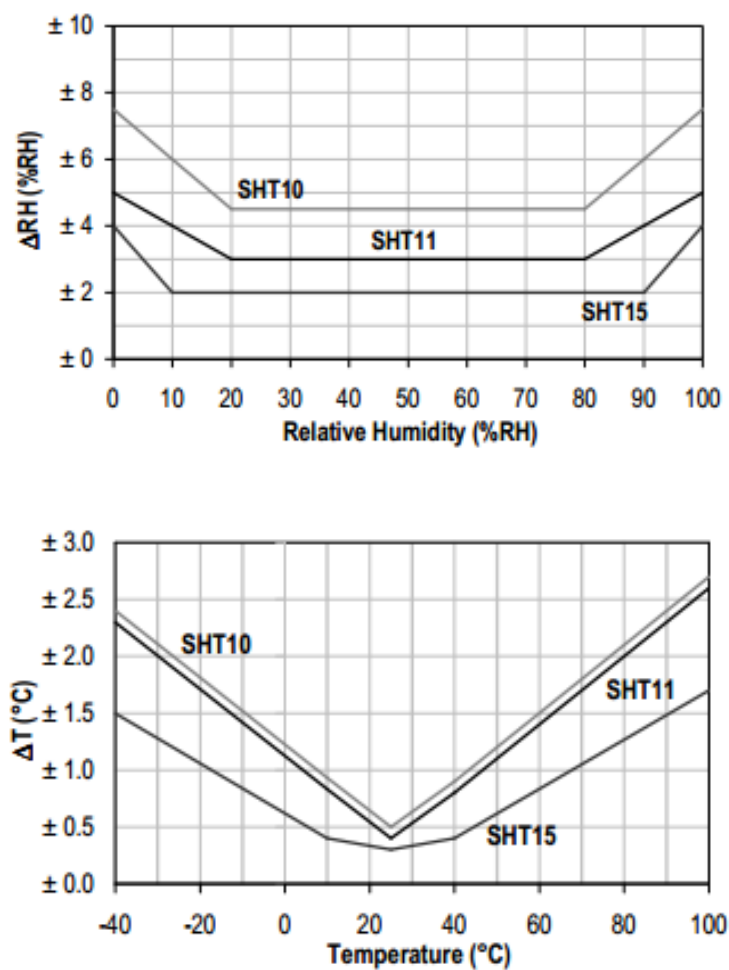
Pin	Name	Comment
1	GND	Ground
2	DATA	Serial Data, bidirectional
3	SCK	Serial Clock, input only
4	VDD	Source Voltage
NC	NC	Must be left unconnected



Gambar 2.21 Konfigurasi pin SHT11

Antarmuka *2-wire serial interface* dan *internal voltage regulation* membuat sistem integrasi yang mudah dan cepat. Juga karena bentuknya yang kecil dan konsumsi powernya yang hemat, sensor ini merupakan pilihan yang terbaik. Sensor ini tersedia dalam tipe bentuk yaitu *surface-mountable LCC (Lealess Chip Carrier)* dan *pluggable 4-pin single-in-line*.

Sensor ini terdiri dari elemen *polimer kapasitif* (digunakan untuk mengukur kelembaban), sensor suhu, 14 bit ADC (*Analog to Digital Converter*), dan *interface serial 2* kabel. Didalamnya juga terdapat *memory* kalibrasi yang digunakan untuk menyimpan koefisien kalibrasi hasil pengukuran sensor. Data hasil pengukuran dari SHT11 ini berupa *digital logic* yang diakses secara *serial*.



Gambar 2.22 Grafik Akurasi suhu dan Kelembaban

Pada gambar 2.22 di atas terlihat akurasi sensor SHT11 untuk pengukuran suhu dan kelembaban.

Akurasi pengukuran suhu tipe SHT11:

Untuk $-40^{\circ}C < T < 28^{\circ}C$ error pengukuran $\pm 2.25^{\circ}C$

Untuk $28^{\circ}C < T < 123.8^{\circ}C$ error pengukuran $\pm 3^{\circ}C$

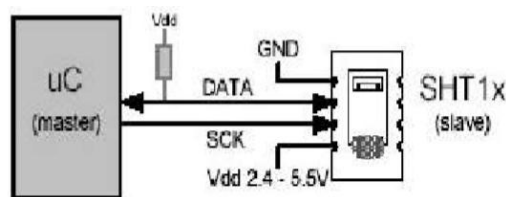
Akurasi pengukuran kelembaban tipe SHT11:

Untuk $0\% RH < H < 23\%RH$ error pengukuran $\pm 5\%RH$

Untuk $23\% RH < H < 85\%RH$ error pengukuran $\pm 4\%RH$

Untuk $85\% RH < H < 100\%RH$ error pengukuran $\pm 5\%RH$

SHT11 mempunyai karakteristik antara lain range kelembaban $0\% RH - 100\%RH$, ketelitian $\pm 4\%RH$ untuk range kerja $23\% RH - 84\%RH$ sedangkan untuk $RH < 23\%$ atau $RH < 84\%$ ketelitian $\pm 5\%$, range suhu $-40^{\circ}C - 123.8^{\circ}C$. untuk SHT11 memerlukan supplay antara $2,4V - 5,5V$. setiap *power-up*, *chip* ini memerlukan 11ms untuk mencapai keadaan stabil (untuk jelasnya karakteristik dan akurasi kelembaban dan suhu dapat dilihat pada gambar 2.22), waktu respon normalnya 4s, untuk jelasnya respon waktu dilihat pada tabel 2.4.



Gambar 2.23 Sepesifikasi Interface Aplikasi Sirkuit SHT11

Serial Interface dari SHT11 dioptimalkan untuk pembacaan sensor dan konsumsi power, tidak kompatibel dengan *12C interface*.

Serial Clock Input (SCK):

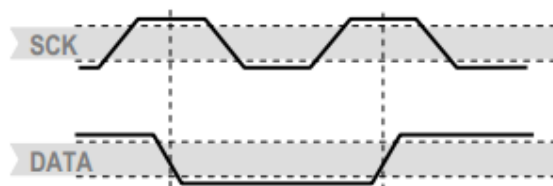
Digunakan untuk men-sinkronisasi komunikasi antara mikrokontroller dan SHTxx. Karena *interface* ini terdiri dari *static logic* sepenuhnya maka tidak ada batasan *frekwensi* minimum dari SCK.

Serial Data (DATA):

Data pin merupakan *tri-state* pin yang digunakan untuk *transfer data in* dan *data out*. DATA berubah setelah transisi turun, dan *valid* pada transisi naik dari *serial clock* SCK. Selama transmisi, DATA line harus stabil selama SCK *high*. Untuk menghindari adanya *signal contetion*, *microcontroller* hanya diperbolehkan men-*drive* DATA dengan *low*. Eksternal *pull-up* resistor (10K) diperlukan untuk memastikan *logic high*.

2.2.9.1 Mengirim Command

Untuk memulai transmisi dikirimkan “Transmisi Start” yang berupa men-*drive low* DATA line ketika SCK *high*, diikuti pulsa *low* pada SCK dan men-*drive high* DATA line ketika SCK masih ber-*logic high* tampak pada gambar 2.24.



Gambar 2.24 Transmisi Start SHT11

Selanjutnya bagian “Command” terdiri dari 3 bit *addres* (yang mendukung hanya 000) dan 5 bit “Command”. SHT11 mengindikasikan penerimaan “Command” dengan men-*drive* DATA *low* (ACK bit) setelah transisi *low* ke-8 dari *clock* SCK. Kontrol DATA line dilepas (sehingga menjadi *high* dikarenakan *pull-up*) setelah transisi turun ke-9 *clock* SCK.

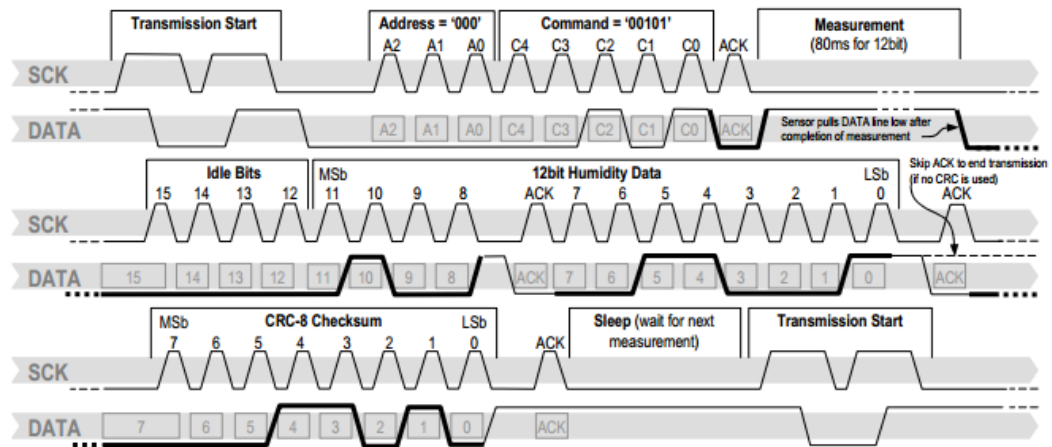
Tabel 2.4 Command SHT11

Command	Code
Reserved	0000x
Measure temperature	00011
Measure Humidity	00101
Read Status Register	00111
Write Status registerReserved	00110
Reserved	0101x-1110x
Soft reset, resets the interface, clears, clears the status register to default values wait minimum 11 ms before next command	11110

2.2.9.2 Pengukuran Sensor

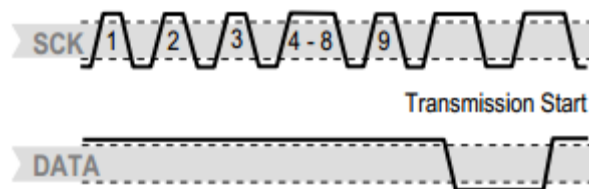
Setelah mengirimkan “Command Measure”, mikrokontroller menunggu proses pengukuran selesai untuk pengukuran 8,12,14 bit. Waktu sesungguhnya bervariasi sampai $\pm 15\%$ kecepatan dari internal oscillator. Untuk menandakan bahwa pengukuran selesai, SHT11 *men-drive low DATA line*. Microcontroller harus menunggu tanda ini sebelum menjalankan *clock* SCK lagi.

Kemudian 2 byte hasil pengukuran dan 1 byte CRC ditransmisikan, *microcontroller* harus memberi *signal acknowledge* untuk tiap *byte* dengan *men-drive DATA line low*. Semua nilai output dimulai dengan MSB atau *right justified*, (misal: SCK ke-5 adalah MSB untuk output 12 bit; sedangkan untuk output 8 bit, byte pertama tidak digunakan). Komunikasi berhenti setelah *acknowledge bit* dari CRC output. Bila CRC tidak diperlukan, maka *mikrokontroller* dapat menghentikan komunikasi setelah output pengukuran LSB (dengan membiarkan *anknowladge high*). *Chip* otomatis masuk dalam mode “sleep” setelah pengukuran dan komunikasi berakhir. Beda waktu antara pengukuran sekitar 1 detik. Bila komunikasi dengan *chip* hilang maka diperlukan *reset serial interface* (dengan menjalankan *clock*)

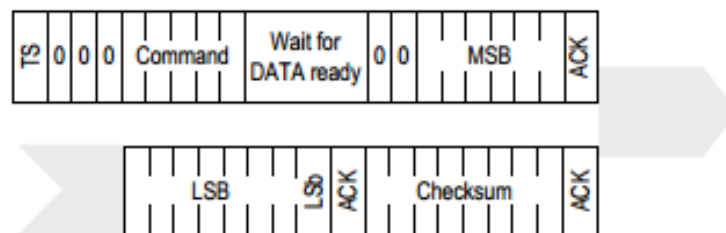


Gambar 2.25 Timing Diagram Pengukuran

SCK lebih dari 9 kali dengan menjaga DATA tetap *high* yang kemudian diikuti dengan “Transmission Start” dan “Command”. *Reset* ini tidak berpengaruh pada isi *status register* dapat dilihat pada gambar 2.26.

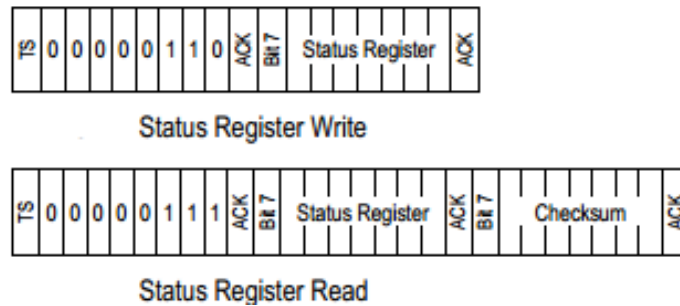


Gambar 2.26 Reset



Gambar 2.27 Skematik Pengukuran

Status *register* beberapa fungsi dari SHT11 terdapat pada “Status Register”, dibawah ini akan dideskripsikan lebih lanjut.



Gambar 2.28 Skematik Status Register

Tabel 2.5 Status Register SHT11

Bit	Type	Description	Default	
7		Reserved	0	
6	R	End of Battery (low voltage detection) '0' for Void > 2.47 '1' for Void < 2.47	X	No default value bit is only updated after a measurement
5		Reserved	0	
4		Reserved	0	
3		For testing only do not use	0	
2	R/W	Heater	0	Off
1	R/W	No reload from OTP	0	Reload
0	R/W	'1' = 8 bit RH/11 bit Temperature resolution '1' = 8 bit RH/11 bit Temperature resolution	0	12 bit RH 14 bit Temp

Default resolusi pengukuran adalah 14 bit (suhu) dan 12 bit (RH). Resolusi ini dapat diubah menjadi 12 bit (suhu) dan 8 bit (RH) untuk kegunaan *transfer data* kecepatan tinggi dan *low power application*.

Fungsi dari *End of Battery* untuk mendeteksi VDD dibawah 2,47 V. Tingkat akurasi $\pm 0,05V$. Bit ini hanya di-update bila terjadi pengukuran.

Heater dalam *chip* SHT11 terdapat elemen heater yang dapat dinyalakan. *Heater* ini bila dinyalakan akan meningkatkan suhu dari sensor $\pm 5^{\circ}C$ ($9^{\circ}F$). Dengan membandingkan hasil pengukuran suhu dan RH sebelum

dan sesudah penggunaan *Heater* maka akan diketahui berfungsi tidaknya sensor tersebut. Dalam lingkungan dengan kelembaban tinggi ($RH > 95\%$), penggunaan *Heater* akan menghambat terjadinya kondensasi, meningkatkan waktu respon dan tingkat akurasi. Bila SHT11 mengalami panas, pengukuran menunjukkan hasil pengukuran suhu yang lebih tinggi dan RH yang lebih rendah dibanding pengukuran pada kondisi normal.

2.2.9.3 Konversi Output SHT11 ke Nilai Fisik

Sensor kelembaban tidak dipengaruhi secara signifikan oleh besarnya *voltage*. Untuk kompensasi ke-tidak linear-an dari sensor kelembaban dan untuk memperoleh akurasi yang tinggi, maka untuk suhu 25°C disarankan menggunakan persamaan 2.1, sedangkan untuk suhu selain 25°C , kompensasi RH menggunakan persamaan 2.2.

$$RH_{linear} = C_1 + C_2 * SO_{RH} + C_3 * SO_{RH}^2 \dots\dots\dots (2.1)$$

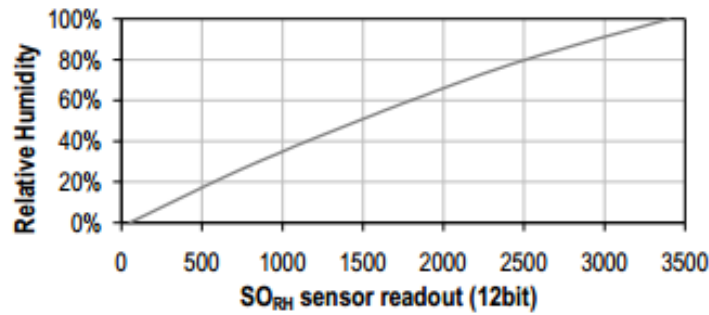
Tabel 2.6 Koefisien Konversi Kelembaban

SO_{RH}	C_1	C_2	C_3
12 bit	-4	0.0405	$-2.8 * 10^6$
8 bit	-4	0.548	$-7.2 * 10^4$

$$RH_{true} = (T_{oc} - 25) * (t_1 + t_2 * SO_{RH}) + RH_{linear} \dots\dots\dots (2.2)$$

Tabel 2.7 Koefisien Konversi oleh Suhu

SO_{RH}	t_1	t_2
12 bit	0.01	0.00008
8 bit	0.01	0.00128



Gambar 2.29 Konversi Output Sensor Kelembaban Terhadap RH

Sensor suhu PPAT (*Proportional To Absolute temperature*) merupakan sensor yang linear, konversi digital output menggunakan persamaan 2.3.

$$\text{Temperature} = d_1 + d_2 * SO_T \dots\dots\dots (2.3)$$

Tabel 2.8 Koefisien Konversi Suhu

VDD	$d_1[^{\circ}C]$	$d_1[^{\circ}F]$
5V	-40.00	-40.00
4V	-39.75	-39.50
3.5V	-39.66	-39.35
3V	-39.60	-39.28
2.5V	-39.55	-39.23

SO_T	$d_2[^{\circ}C]$	$d_2[^{\circ}F]$
14bit	0.01	0.018
12bit	0.04	0.072

$$\text{Error} = \frac{\text{Suhu sensor pembanding} - \text{Suhu sensor SHT11}}{\text{Suhu sensor pembanding}} \times 100\% \dots\dots\dots (2.4)$$

$$\text{Error} = \frac{\text{Kelembaban sensor pembanding} - \text{Kelembaban sensor SHT11}}{\text{Kelembaban sensor pembanding}} \times 100\% \dots\dots (2.5)$$

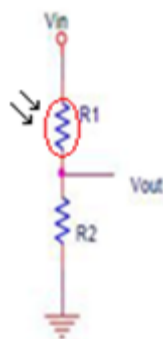
2.2.10 Sensor Cahaya

Sensor LDR (*Light Dependent Resistor*) merupakan salah satu komponen elektronika yang digunakan untuk pembuatan sensor cahaya yang pada dasarnya LDR mempunyai sifat yang sama dengan resistor, hanya saja nilai resistansi dari LDR berubah-ubah sesuai dengan tingkat intensitas cahaya yang diterimanya. LDR merupakan sensor yang berkerja apabila terkena cahaya. LDR memiliki hambatan yang sangat tinggi jika tidak terkena cahaya dan memiliki hambatan yang sangat kecil jika terkena cahaya.



Gambar 2.30 Sensor LDR Ukuran Besar dan Kecil

Sensor LDR tergolong sensor yang pasif sehingga memerlukan suatu rangkaian khusus untuk dapat dipergunakan pada suatu aplikasi tertentu seperti rangkaian pembagi tegangan menggunakan resistor. Pada rangkaian pembagi tegangan tersebut sensor LDR dipasang seperti resistor yang di gunakan untuk pembagi tegangan, dan dapat dilihat pada gambar 2.31.



Gambar 2.31 Rangkaian Dasar Sensor Cahaya

Untuk Mencari nilai V_{out} dengan persamaan:

$$V_{out} = \frac{R_2}{R_2 + R_1} V_{in} \dots\dots\dots (2.6)$$

$$V_{out} (R_2 + R_1) = R * V_{in} \dots\dots\dots (2.7)$$

Untuk Mencari nilai resitansinya dengan persamaan:

$$R_1 = \frac{R_2 * V_{in} - R_2 * V_{out}}{V_{out}} \dots\dots\dots (2.8)$$

Milsa Cara kerja gambar 2.31:

Dik: $R_1 = 0.50 \text{ K}\Omega$ $R_2 = 1 \text{ K}\Omega$ $V_{in} = 5 \text{ V}$

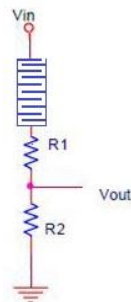
Dit: $V_{out} = V_{in} \times R_2 / (R_2 + R_1)$

$$V_{out} = 5 \text{ V} \times 1 \text{ K}\Omega / (1 \text{ K}\Omega + 0.50 \text{ K}\Omega)$$

$$V_{out} = 3.33 \text{ Volt (Data yang akan di proses mikrokontroler)}$$

2.2.11 Sensor Hujan

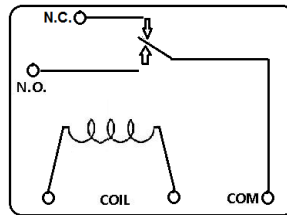
Sensor hujan berfungsi untuk memberikan nilai masukan pada tingkat elektrolisasi air hujan dimana air hujan akan menyentuh ke panel sensor hujan, dari segi cara kerja tidak jauh dari cara kerja sensor cahaya yang memanfaatkan nilai V_{out} dari hasil pembagi tegangan antara kedua resistor. Hanya yang berbeda adalah V_{in} yang akan diatur masuknya sesuai dengan tersentuhnya panel sensor oleh air hujan. Untuk cara kerja sensor hujan sama dengan cara kerja gambar 2.31 dengan menggunakan persamaan 2.6.



Gambar 2.32 Rangkaian Dasar Sensor Hujan

2.2.12 Relay

Relay merupakan saklar elektronik yang dapat dikendalikan dari rangkaian elektronik lainnya. Yang dimana terdiri dari 3 bagian utama, yaitu:



Gambar 2.33 Rangkaian Relay

1. Koil : Lilitan dari relay
2. Common : Bagian yang tersambung dengan NC
3. Kontak : Terdiri dari NC (*Normally Closed*) dan NO (*Normally Open*)

2.2.13 Catu Daya

Catu daya memegang peranan yang sangat penting dalam hal perancangan sebuah alat. Tanpa adanya masukan daya maka perangkat tidak dapat berfungsi. Begitu juga apabila pemilihan catu daya tidak tepat, maka perangkat tidak dapat bekerja dengan baik. Penentuan sistem catu daya yang akan digunakan ditentukan oleh beberapa faktor, diantaranya :

1. Tegangan

Setiap aktuator tidak memiliki tegangan yang sama. Hal ini akan berpengaruh terhadap desain catu daya. Tegangan tertinggi dari salah satu aktuator akan menentukan nilai tegangan catu daya.

2. Arus

Arus memiliki satuan Ah (*Ampere hour*). Semakin besar Ah, semakin lama daya tahan baterai bila digunakan pada beban yang sama.

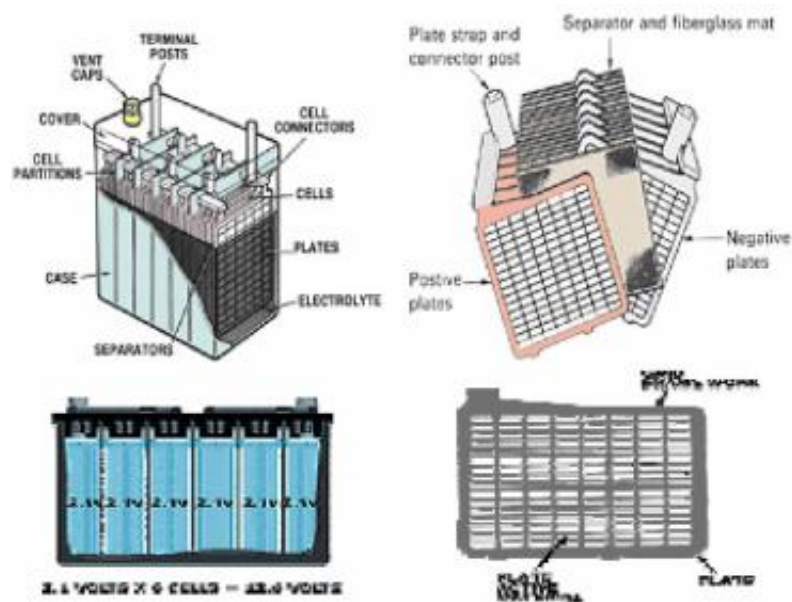
3. Teknologi Baterai

Baterai isi ulang ada yang dapat diisi hanya apabila benar-benar kosong dan ada pula yang dapat diisi ulang kapan saja tanpa harus menunggu baterai tersebut benar-benar kosong.

Catu daya yang akan digunakan pada perancangan alat ini adalah Baterai (ACCU), transformator dan regulator.

2.2.13.1 Baterai (ACCU)

Baterai merupakan sumber energi listrik yang digunakan dan Baterai ada dua tipe yaitu baterai kering dan baterai basah. Serta baterai terdiri dari beberapa komponen antara lain : Kotak baterai, terminal baterai, elektrolit baterai, lubang elektrolit baterai, tutup baterai dan sel baterai. Dalam satu baterai terdiri dari beberapa sel baterai, tiap sel menghasilkan tegangan 2 - 2,2 V. Baterai 6 V terdiri dari 3 sel, dan baterai 12 V mempunyai 6 sel baterai yang dirangkai secara seri. Tiap sel baterai mempunyai lubang untuk mengisi elektrolit baterai, lubang tersebut ditutup dengan tutup baterai, pada tutup terdapat lubang ventilasi yang digunakan untuk mengalirkan uap dari elektrolit baterai. Tiap sel baterai terdapat plat positif, saparator dan plat negatif, plat positif berwarna coklat gelap (*dark brown*) dan plat negatif berwarna abu-abu metalik (*metallic gray*).



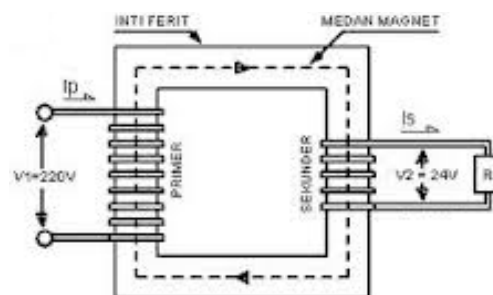
Gambar 2.34 Kontruksi Baterai

2.2.13.2 Transformator

Transformator merupakan suatu peralatan listrik elektromagnetik statis yang berfungsi untuk memindahkan dan mengubah daya listrik dari suatu

rangkaian listrik ke rangkaian listrik lainnya dengan frekuensi yang sama dan perbandingan transformasi tertentu melalui suatu gandingan magnet dan bekerja berdasarkan prinsip induksi *elektromagnetis*, dimana perbandingan tegangan antara sisi primer dan sisi sekunder berbanding lurus dengan perbandingan jumlah lilitan dan berbanding terbalik dengan perbandingan arusnya.

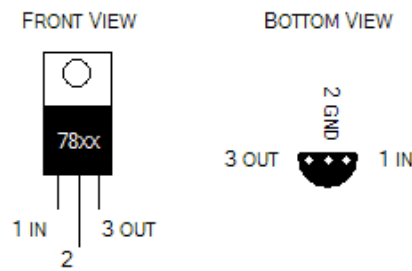
Transformator terdiri atas dua buah kumparan (primer dan sekunder) yang bersifat induktif. Kedua kumparan ini terpisah secara elektris namun berhubungan secara magnetis melalui jalur yang memiliki reluktansi rendah. Apabila kumparan primer dihubungkan dengan sumber tegangan bolak-balik maka fluks bolak-balik akan muncul di dalam inti yang dilaminasi, karena kumparan tersebut membentuk jaringan tertutup maka mengalirlah arus primer. Akibat adanya fluks di kumparan primer maka di kumparan primer terjadi induksi (*self induction*) dan terjadi pula induksi di kumparan sekunder karena pengaruh induksi dari kumparan primer atau disebut sebagai induksi bersama (*mutual induction*) yang menyebabkan timbulnya fluks magnet di kumparan sekunder, maka mengalirlah arus sekunder jika rangkaian sekunder di bebani, sehingga energi listrik dapat ditransfer keseluruhan (secara magnetisasi).



Gambar 2.35 Transformator

2.2.13.3 Regulator

Regulator tegangan menyediakan *output* tegangan DC yang konstan dan secara terus-menerus menahan tegangan *output* pada nilai yang diinginkan. Regulator hanya dapat bekerja jika tegangan *input* (V_{in}) lebih besar daripada tegangan *output* (V_{out}). Biasanya perbedaan tegangan *input* dengan *output* yang direkomendasikan tertera pada *datasheet* komponen tersebut.

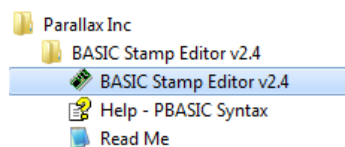
Gambar 2.36 *Pin-out Regulator*

2.3 Perangkat Lunak

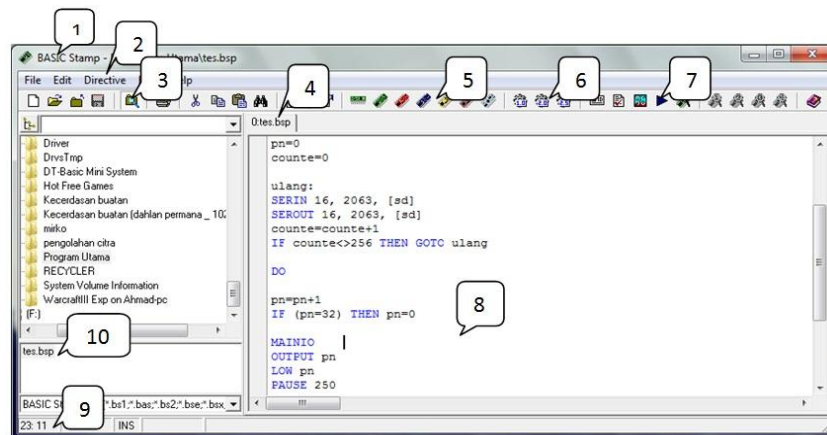
2.3.1 Pengenalan Basic Stamp Editor

Basic Stamp editor adalah sebuah *editor* yang di buat oleh *Parallax Inc* untuk menulis program, meng*compile* dan mendownloadnya ke mikrokontroler keluarga *Basic Stamp*. Bahasa pemrograman yang digunakan adalah bahasa *Basic*. Langkah-langkah untuk memulai menggunakan *editor* tersebut adalah sebagai berikut :

- **Klik *Start, Program, Parallax Inc, Basic Stamp Editor V 2.4***
- Selain itu bisa juga meng**klik icon** shortcut ***Basic Stamp Editor*** di desktop

Gambar 2.37 Menjalankan *Basic Stamp Editor*

Setelah memulai untuk menjalankan editor tersebut, maka sekarang *editor BASIC Stamp* sudah jalan dan siap digunakan. Tampilan utamanya adalah sebagai berikut.



Gambar 2.38 Tampilan Utama *Basic Stamp* Editor

Keterangan gambar 2.38:

1. Nama *editor*, nama *folder* dan nama *file* yang sedang dibuka atau dikerjakan
2. Menu utama *editor*
3. *Shortcut* untuk menyimpan, *cut*, *copy*, *paste*, *print* dan lain-lain
4. Nama *file* yang sedang dikerjakan
5. Pemilihan jenis mikrokontroler yang digunakan
6. Pemilihan versi *compiler* PBASIC
7. Menjalankan program (Program RUN)
8. Area utama pengetikan program
9. Status posisi kursor berada (baris-kolom)
10. *List file-file* yang ada di *folder* kerja
11. *Folder* utama yang digunakan untuk menyimpan *file-file* kerja

Perangkat lunak merupakan faktor penting dalam tahap perancangan suatu alat yang berisi algoritma gerak dan tugas perangkat dalam bentuk *listing* program yang ditanamkan ke dalam mikrokontroler. Program dapat bermacam-macam bentuk versi dan bahasa pemrogramannya sesuai dengan spesifikasi dari mikrokontroler yang digunakan.

Mikrokontroler *basic stamp* (BS2P40) menggunakan bahasa pemrograman *basic*. *Software* yang digunakan adalah *basic stamp editor*. Program ini memungkinkan penggunaannya memprogram *basic stamp* dengan bahasa *basic*

yang relatif ringan dibandingkan bahasa pemrograman lainnya. Berikut ini beberapa instruksi-instruksi dasar yang dapat digunakan pada mikrokontroler *basic stamp*.

Tabel 2.9 Beberapa Instruksi Dasar *Basic Stamp*

INSTRUKSI	KETERANGAN
DO...LOOP	Perulangan
GOSUB	Memanggil prosedur
IF..THEN	Percabangan
FOR ...NEXT	Perulangan
PAUSE	Waktu tunda milidetik
IF...THEN	Perbandingan
PULSOUT	Pembangkit pulsa
PULSIN	Menerima pulsa
GOTO	Loncat ke alamat memori tertentu
HIGH	Menset pin I/O menjadi 1
LOW	Menset pin I/O menjadi 0
PWM	Konversi suatu nilai digital ke keluaran analog lewat <i>pulse width modulasi</i>

2.3.2 Membuat Program

2.3.2.1 Directive

Directive ditulis di awal program. Bagian ini menentukan tipe prosesor yang digunakan dan versi dari kompilr PBASIC yang digunakan untuk mengkompilr bahasa basic menjadi bahasa mesin. Tampilannya adalah seperti gambar berikut:



Gambar 2.39 *Icon tipe Basic Stamp dan Compiler*

```
' {$STAMP BS2p}      ' tipe mikrokontroler yang digunakan
' {$PBASIC 2.5}      ' bahasa PBASIV kompiller yang digunakan
```

Gambar 2.40 Tampilan Bagian Directive

2.3.2.2 Menentukan Variabel

Menentukan pin mikrokontroler yang digunakan serta membuat variabel. Ada beberapa ketentuan untuk mendeklarasikan variable yaitu:

- 1). PIN : PIN dari mikrokontroler
- 2). VAR : Variabel
- 3). CON : Konstanta

PIN yang digunakan sudah ditentukan sesuai dengan konfigurasi *hardware/mainboard* yang digunakan adalah BS2P40. Selain itu dapat membuat variabel bebas yang nantinya dapat digunakan untuk keperluan perulangan atau yang lainnya.

Setelah menentukan variabel dan pin yang digunakan, selanjutnya membuat program utama. Pada bagian program utama bisa melakukan dua mode program, yaitu program dengan pengetikan langsung atau program dengan pemanggilan prosedur. Program pengetikan lebih efektif jika program tidak terlalu banyak dan hanya untuk menangani kasus yang sederhana. Sedangkan untuk program yang banyak, rumit dan lebih dari satu slot, maka sebaiknya menggunakan program pemanggilan prosedur.

2.3.2.3 Memeriksa Sintak Program

Memeriksa sintak program lakukan untuk memastikan semua sintak sudah benar. Untuk memeriksa sintak ini bias pilih menu *Run, Cek* sintak atau kombinasi tombol CTRL+T. Tampilan jika listing program yang kita buat sudah benar.

Gambar 2.41 Hasil Pemeriksaan Sintak yang Sukses (*tokenize sussessful*)

2.3.2.4 Menjalankan Program

Setelah program selesai, program siap di *download* ke modul *basic stamp*. Cara untuk menjalankan program dapat memilih menu *Run* atau kombinasi tombol CTRL+R. Tampilan jika *download* program sukses.



Gambar 2.42 Download Program Sukses