BAB II

TINJAUAN PUSTAKA

Pada bab ini akan dijelaskan mengenai studi pustaka alat ukur IC *digital*, teori-teori penunjang sistem alat ukur IC *digital* baik perangkat keras (*Hardware*), maupun perangkat lunak (*Software*), serta beberapa teori penunjang lainnya.

2.1 (Integrated Circuit) IC

IC adalah komponen elektronik yang terbuat dari bahan semikonduktor, dimana IC merupakan gabungan dari komponen seperti resistor, kapasitor, dioda, dan transistor yang telah terintegrasi menjadi sebuah rangkaian berbentuk *chip*. IC adalah komponen yang dipakai sebagai otak peralatan elektronika. Berdasarkan struktur dari gerbang-gerbang yang terdapat didalam nya, IC terdiri dari :

1. IC *Dioda-Dioda Logic* (DDL)
2. IC *Transisistor-Transistor Logic* (TTL)
3. IC *Resistor-Transistor Logic* (RTL*)*
4. IC *Complementary Metal Oxide Semiconductor* (CMOS)

Namun pada pembahasan ini hanya menjelaskan jenis IC TTL dan CMOS. Kedua jenis IC rangkaian tersebut dinamakan juga IC *digital*. Adapun penjelasan dari IC ini sebagai berikut :

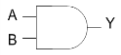
2.1.1 IC TTL

Pertama kali IC dengan teknologi TTL ini dipekenalkan oleh (*texas instrument*) TI pada tahun 1964. Kemudian keluarga IC ini dengan pesat penggunaannya karena keseimbangan antara kecepatan dan konsumsi dayanya.

Keluarga TTL telah memiliki beberapa jenis selain IC TTL pada umumnya yaitu TTL dengan daya rendah (*low power* TTL), TTL dengan kecepatan tinggi (*High-speed* TTL), TTL *schotty* dengan daya rendah, TTL *schotty* dengan kecepatan tinggi dan lain sebagainya. Semua jenis IC TTL memiliki dasar rangkaian yang sama. Dalam satu kemasan IC terdapat beberapa macam *gate* (gerbang) yang dapat melakukan berbagai macam fungsi logik seperti AND, NAND, OR, NOR, XOR serta beberapa fungsi logika lainnya seperti *shift register* dan *multiplexser*. berikut ini merupakan bagian dari gerbang logika dasar:

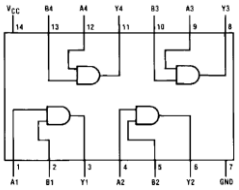
a. Gerbang AND (7408)

   Gerbang logika yang kerjanya seperti saklar seri. Gerbang AND mempunyai dua atau lebih *input* dan memiliki satu *output*. *Output* akan berlogika "1" jika semua *input* (*input* A AND B) berlogika "1". Jika salah satu *input* berlogika "0" maka *output* akan berlogika "0".



Gambar 2.1 Simbol gerbang AND

Untuk menguji gerbang AND, digunakan IC 7408. Dimana struktur dari IC ini adalah:



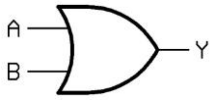
Gambar 2.2 Struktur IC 7408

Tabel 2.1 Tabel Kebenaran gerbang AND

Masukan		Keluaran
A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

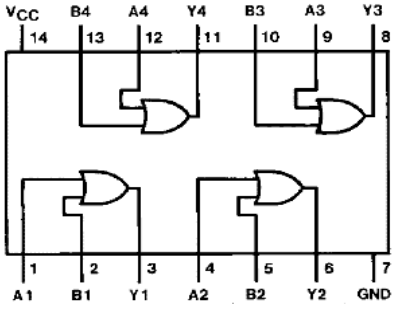
b. Gerbang OR (7432)

Gerbang OR mempunyai dua atau lebih *input* dan memiliki satu *output*. Apabila salah satu *input* berlogika "1", maka *output* akan berlogika "1". Jika semua *input* berlogika "0", maka *output* akan berlogika "0".



Gambar 2.3. Simbol gerbang OR

Untuk menguji gerbang OR, digunakan IC 7432. Dimana struktur dari IC ini adalah:



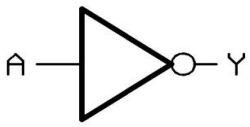
Gambar 2.4. Struktur IC 7432

Tabel 2.2. Tabel Kebenran Gerbang OR

Masukan		Keluaran
A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1

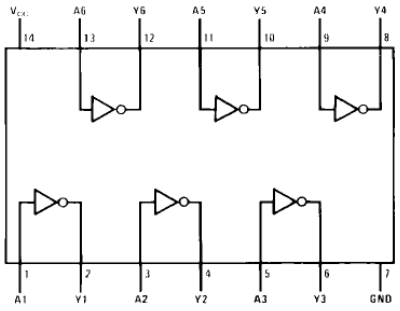
c. Gerbang NOT (7404)

Gerbang NOT hanya memiliki satu *input* dan satu *output* saja. Apabila *input* berlogika “0”, maka *output* akan berlogika “1”. Dan jika semua *input* berlogika “1”, maka *output* akan berlogika “0”.



Gambar 2.5 Simbol Gerbang NOT

Untuk menguji gerbang OR, digunakan IC 7404. Dimana struktur dari IC ini adalah:



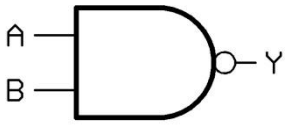
Gambar 2.6 Struktur gerbang NOT

Tabel 2.3. Tabel Kebenaran Gerbang NOT

Masukan	Keluaran
A	Y
0	1
1	0

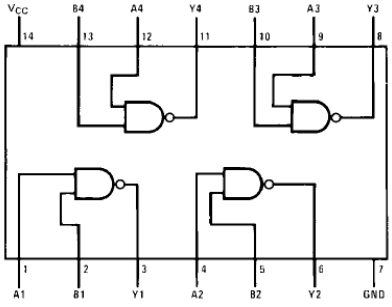
d. Gerbang NAND (7400)

Gerbang NAND merupakan kombinasi dari gerbang AND dan gerbang NOT. Sehingga keluaran dari gerbang NAND merupakan komplemen dari keluaran gerbang AND.



Gambar 2.7 Simbol Gerbang NAND

Untuk menguji gerbang NAND, digunakan IC 7400. Dimana struktur dari IC ini adalah:



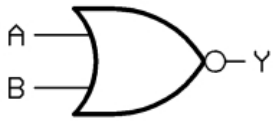
Gambar 2.8 Struktur gerbang NAND

Tabel 2.4. Tabel Kebenaran Gerbang NAND

Masukan		Keluaran
A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0

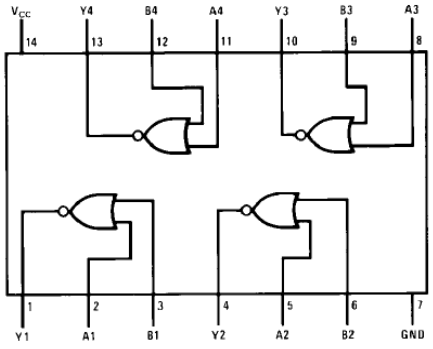
e. Gerbang NOR (7402)

Gerbang NOR merupakan kombinasi dari gerbang OR dan gerbang NOT. Sehingga keluran dari gerbang NOR merupakan komplemen dari keluaran gerbang OR.



Gambar 2.9. Simbol gerbang NOR

Untuk menguji gerbang NOR, digunakan IC 7402. Dimana struktur dari IC ini adalah:



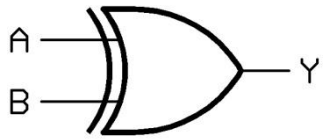
Gambar 2.10 Struktur Gerbang NOR

Tabel 2.5. Tabel Kebenaran Gerbang NOR

	Masukan	Keluaran
A	B	Y
0	0	1
0	1	0
1	0	0
1	1	0

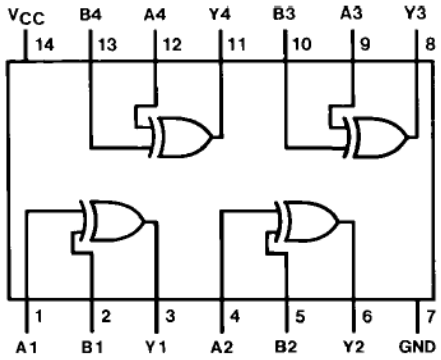
f. Gerbang XOR (7486)

Gerbang XOR merupakan kata lain dari exclusive – OR. XOR akan memberikan *output* logika "1", jika *input*nya memberikan keadaan yang berbeda. Dan jika *input*nya memberikan keadaan yang sama, maka *output*nya akan memberikan logika "0".



Gambar 2.11. Simbol gerbang XOR

Untuk menguji gerbang XOR, digunakan IC 7486. Dimana struktur dari IC ini adalah:



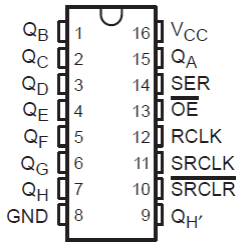
Gambar 2.12 Struktur Gerbang XOR

Tabel 2.6. Tabel Kebenaran Gerbang XOR

Masukan		Keluaran
A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0

g. *Shift Register (74595)*

Shift register merupakan kumpulan dari elemen-elemen memori yang bekerja bersama sebagai satu unit. *Register* yang paling sederhana adalah sebuah penyimpanan kata *biner*, jenis lain dari sebuah *register* adalah dapat mengubah kata yang tersimpan dengan menggeser bit-bitnya kekiri atau kekanan. Pencacah merupakan jenis khusus dari *register*, yang dirancang untuk mencacah atau menghitung jumlah pulsa-pulsa. *Register buffer* merupakan jenis *register* yang paling sederhana yang mempunyai fungsi untuk menyimpan kata *digital*. *Register* geser dapat memindahkan bit-bit yang tersimpan dari kiri ke kanan pergeseran bit itu penting dalam operasi aritmatik dan operasi logika yang dipakai dalam komputer. Berikut ini merupakan gambar struktur dari *shift register*.

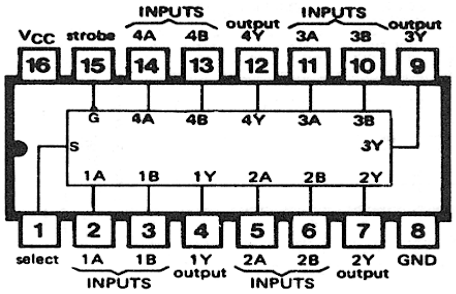


Gambar 2.13 Struktur IC 74595

h. Multiplexer (74157)

Multiplexer merupakan rangkaian kombinasional yang memiliki masukkan sejumlah 2^nbit, n *selector* dan satu *output*. Multiplexer disebut juga data *selector* karena *selector* pada rangkaian multiplexer berfungsi untuk memilih data pada *input* mana yang akan dilewatkan ke *output*. Seperti *decoder*, multiplexer juga memiliki *enable* yang bersifat *low* yang berfungsi untuk mengaktifkan atau menonaktifkan rangkaian.

Untuk menguji multiplexer pada perancangan digunakan IC tipe 74157, adapun struktur IC ini adalah sebagai berikut.



Gambar 2.14 Struktur IC 74157

2.1.2 IC (*Complementyary Metal Oxide Semiconductor*) CMOS

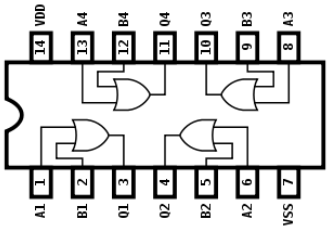
CMOS adalah sebuah jenis utama dari rangkaian terintegrasi. Teknologi CMOS digunakan di mikroprosesor, pengontrol mikro, RAM statis dan sirkuit logika *digital* lainnya. Teknologi CMOS juga digunakan dalam banyak sirkuit *analog*, seperti sensor gambar, pengubah data dan terintegrasi untuk berbagai jenis komunikasi.

CMOS juga sering disebut *complementary-symentry metal-oxidesemiconductor* atau COSMOS (semionduktor logam oksida

komplementer simetris). Kata komplementer simetris merujuk pada kenyataan bahwa biasanya desain *digital* berbasis CMOS menggunakan pasangan komplementer dan simetris dari MOSFET semikonduktor tipe-p dan semikonduktor tipe-n untuk fungsi logika.

Pada prinsipnya IC TTL dan IC CMOS mempunyai dasar pengertian yang sama. Walaupun demikian ada beberapa perbedaan, juga keuntungan dan kerugiannya. CMOS menjadi populer dan disukai karena disipasi dayanya yang rendah dan kemampuan beroperasi dalam tegangan antara yang lebar. IC CMOS difabrikasi dengan menggunakan dua metal oksida semikonduktor (MOS).

Sebuah keunggulan yang unik dari CMOS ini adalah tidak adanya arus yang melewatinya baik dalam keadaan status 1 atau 0. Karena itu daya hanya terdisipasi ketika IC CMOS dalam perubahan dari keadaan 1 menuju 0 dan sebaliknya. Oleh karena itu daya disispasi dari CMOS tergantung dari frekuensi perubahan keadaan. IC CMOS hanya cocok digunakan pada aplikasi kecepatan yang sedang karena disipasi dayanya yang rendah. Namun pada pembahasan ini IC CMOS yang dapat diukur yaitu memiliki gerbang. Pada gambar dibawah ini ditunjukkan gambar IC CMOS yang memiliki gerbang.



Gambar 2.15. Konfigurasi IC CMOS Gerbang

2.2 Perangkat Keras (*Hardware*)

Perangkat keras alat ukur IC yang akan diuraikan dibawah ini adalah perangkat-perangkat fisik yang terdiri dari komponen-komponen elektronik penunjang sistem kerja alat ukur IC dimana dari setiap komponen-komponen yang ada, akan dirangkai dan terintegrasi menjadi satu sistem berbasis mikrokontroler ATmega32.

2.2.1 Mikrokontroler ATmega32

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

Mikrokontroler berbeda dengan mikroprosesor, perbedaan tersebut dapat dilihat dari kecepatan proses, cara menangani tugas dan unit-unit pendukung lain yang bertugas sesuai dengan fungsinya. Pada mikrokontroler sudah terdapat RAM, ROM, dan CPU didalam IC nya, sedangkan pada mikroprosesor hanya terdapat CPU saja dan bagian RAM dan ROM nya terpisah sehingga agar mikroprosesor dapat bekerja sesuai dengan fungsinya perlu ditambahkan perangkat penunjang lain seperti rangkaian RAM dan ROM. Karena perbedaan dalam arsitekturnya maka untuk kecepatan proses

masih tetap lebih unggul mikroprosesor, namun untuk proses-proses yang tidak memiliki kompleksitas kerja yang besar mikrokontroler lebih dibutuhkan karena dalam segi biaya lebih murah mikrokokntroler dan dalam segi ergonomisnya pun lebih praktis menggunakan mikrokontroler, karena didalam mikrokontroler telah terdapat rangkaian RAM, ROM, CPU, dan unit I/O yang siap digunakan untuk keperluan apapun.

2.2.1.1 Arsitektur ATmega32

ATmega32 tergolong Mikrokontroler jenis AVR yang memiliki arsitektur RISC (*Reduced Instruction Set Computing*) 8 bit, dimana semua instruksi dikemas dalam kode 16 bit dan sebagian besar instruksi dikemas berbeda dengan instruksi MCS51 yang membutuhkan 12 siklus *clock*. Secara garis besar, arsitektur mikrokontroler ATmega32, yaitu.

a. 32 saluran I/O (Port A, Port B, Port C, dan Port D).
b. 10 bit 8 *Channel* ADC (*Analog to Digital Converter*).
c. 4 *channel* PWM.
d. 6 *Sleep Modes* : *Idle*, ADC *Noise Reduction*, *Power-save*, *Power- down,Standby and Extended Standby*.
e. 3 buah *timer/counter.*
f. *Analog comparator.*
g. *Watchdog timer* dengan osilator *internal.*
h. 512 *byte* SRAM.
i. 512 *byte* EEPROM.

j. 8 kb *Flash memory* dengan kemampuan *Read While Write*.

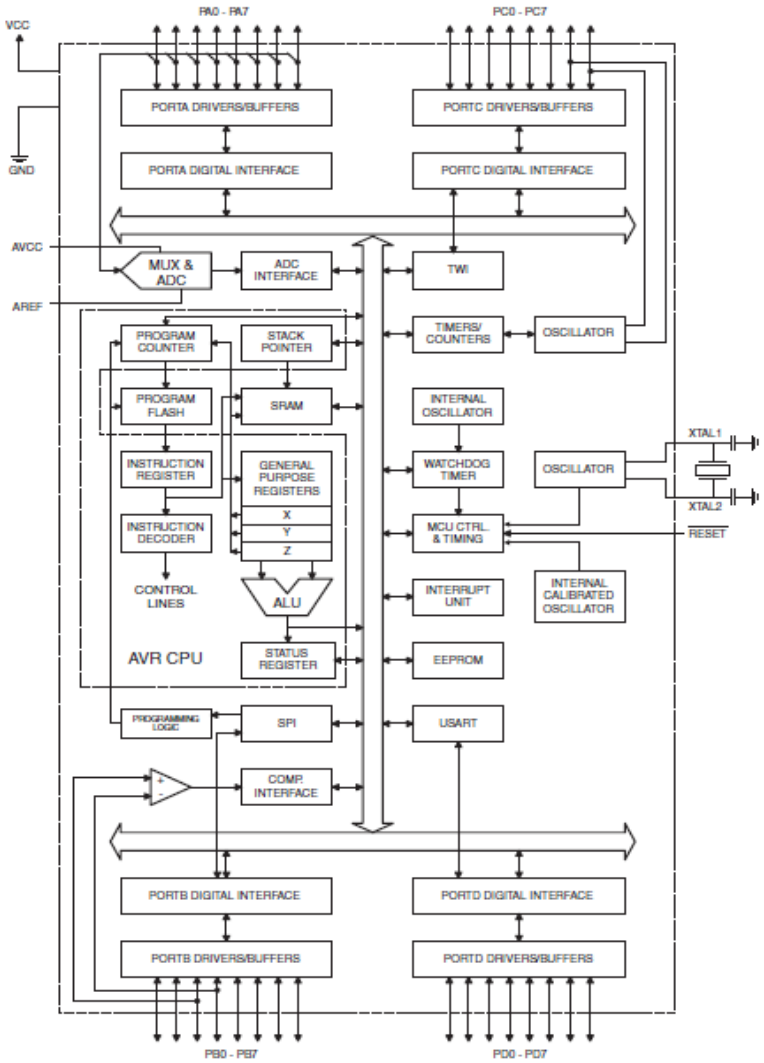
k. Unit interupsi (internal & eksternal).

l. Port antarmuka SPI32 "*memory map*".

m. Port USART untuk komunikasi serial dengan kecepatan optimum 2,5Mbps.

n. 4.5 sampai 5.5V *operation*, 0 sampai 16M

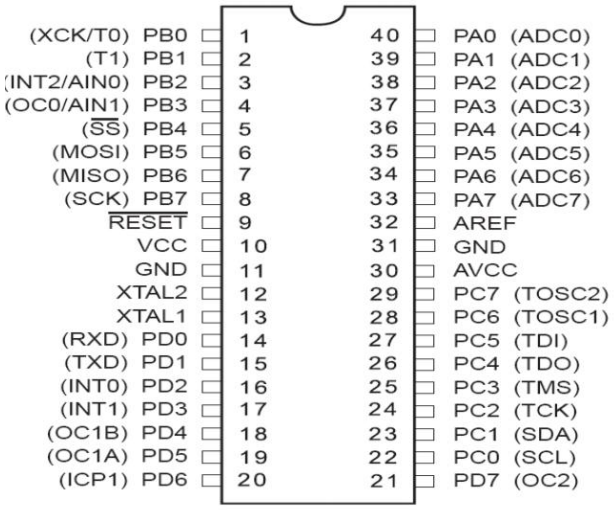
2.2.1.2 Blok Diagram ATmega32



Gambar 2.16 Blok diagram ATmega32

2.2.1.3 Konfigurasi Pin ATmega32

Mikrokontroler ATmega32 mempunyai jumlah pin sebanyak 40 buah, dimana 32 pin digunakan untuk keperluan port I/O yang dapat menjadi pin *input*/*output* sesuai konfigurasi. Berikut ini merupakan konfigurasi pin dari ATmega32.



Gambar 2.17 Konfigurasi Pin ATmega32

Berikut penjelasan dari fungsi-fungsi konfigurasi pin ATmega32.

a. VCC, merupakan pin yang berfungsi sebagai pin masukkan catu daya.
b. GND (*Ground*), merupakan pin ground.
c. PORTA (PORTA0-7), merupakan pin I/O dua arah dan berfungsi khusus sebagai pin masukan ADC.
d. PORTB (PORTB0-7), merupakan pin I/O dua arah dan fungsi khusus sebagai pin *timer/counter*, komparator analog dan SPI.

e. PORTC (PORTC0-7), merupakan pin I/O dua arah dan fungsi khusus yaitu TWI, komparator *analog*, dan *timer oscilator*.

f. PORTD (PORTD0-7), merupakan pin I/O dua arah dan fungsi khusus yaitu komparator *analog*, interupsi eksternal dan komunikasi serial USART.

g. *RESET*, merupakan pin untuk mereset mikrokontroler.

h. XTAL1 dan XTAL2 merupakan pin untuk eksternal *clock*.

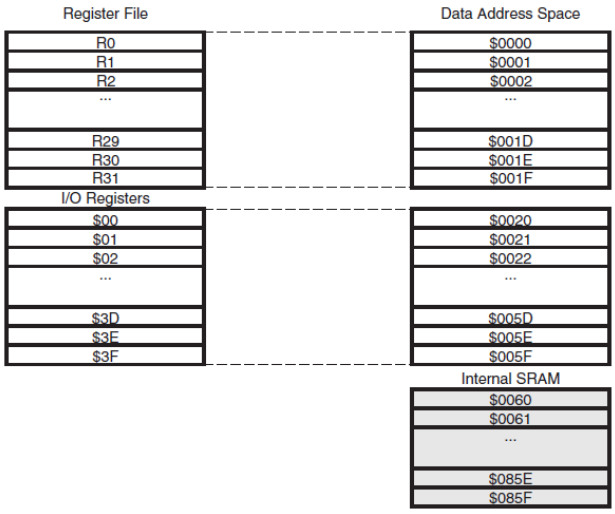
i. AVCC merupakan pin masukan untuk tegangan ADC.

j. AREF merupakan pin masukan untuk tegangan referens ADC.

2.2.1.4 Peta Memory ATmega32

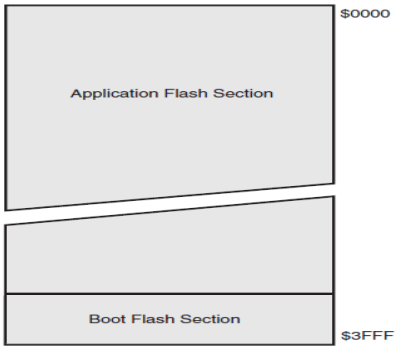
ATmega32 memiliki ruang pengalamatan memori data dan memori program yang terpisah. Memori data terbagi menjadi 3 bagian yaitu 32 buah *register* umum, 64 buah *register* I/O, dan 512 *byte* SRAM internal. *Register* untuk keperluan umum menempati *space* data pada alamat terbawah yaitu $00 sampai $1F. Sementara itu *register* khusus untuk menangani I/O dan kontrol terhadap mikrokontroler menempati 64 alamat berikutnya yaitu mulai dari $20 sampai $5F. *Register* tersebut merupakan *register* yang khusus digunakan untuk mengatur fungsi terhadap berbagai peripheral mikrokontroler seperti kontrol *register*, *timer/counter*, fungsi I/O, dan sebagainya. *Register* khusus alamat memori secara lengkap dapat dilihat pada gambar 2.18 Alamat memori berikutnya digunakan untuk SRAM 512 *byte*, yaitu pada lokasi $60 sampai dengan $25F. ATmega32 memiliki 608 alamat memori data yang

terbagi menjadi 3 bagian, yaitu 32 buah *register* umum, 64 buah *register* I/O, dan 512 *byte internal* SRAM.



Gambar 2.18 Skema *Memori Data* ATmega32

Memori program yang terletak pada *flash perom* tersusun dalam *word* atau 2 byte karena setiap instruksi memiliki lebar 16-bit atau 32bit. AVR ATmega32 memiliki 4KByte x 16 Bit *flash perom* dengan alamat mulai dari $000 sampai $FFF. AVR tersebut memiliki 12 bit *program counter* (PC) sehingga mampu mengalamati isi *flash*.



Gambar 2.19 Skema Memori Program ATmega32

2.2.1.5 *Status Register*

Status register adalah *register* berisi status yang dihasilkan pada setiap operasi yang dilakukan ketika suatu instruksi dieksekusi. SREG merupakan bagian dari inti CPU mikrokontroler.

Bit	7	6	5	4	3	2	1	0	
	JTD	ISC2	–	JTRF	WDRF	BORF	EXTRF	PORF	MCUCSR
Read/Write	R/W	R/W	R	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	See Bit Description					

Gambar 2.20 *Status Register* ATmega32

Status register ATmega32 dapat dilihat pada gambar diatas. Dari gambar tersebut dapat dijelaskan *register* dari ATmega32, sebagai berikut:

a. Bit7 → I (*global Iinterrupt enable*), Bit harus di set untuk mengaktifkan *(enable)* semua jenis interupsi.
b. Bit6 → T (*bit copy storage*), Instruksi BLD dan BST menggunakan bit T sebagai sumber atau tujuan dalam operasi bit. Suatu bit dalam sebuah *register* GPR dapat disalin ke bit T menggunakan instruksi BST, dan sebaliknya bit T dapat disalin kembali kesuatu bit dalam *register* GPR dengan menggunakan instruksi BLD.
c. Bit5 → H *(half cary flag).*
d. Bit4 → S (*sign bit*) merupakan hasil operasi EOR antara *flag* -N (negatif) dan *flag* V (komplemen dua *overflow*)

e. Bit3 → V (*two's component overflow flag*) bit ini berfungsi untuk mendukung operasi matematis.

f. Bit2 → N (*negative flag*) flag N akan menjadi set, jika suatu operasi matematis menghasilkan bilangan negatif.

g. Bit1 → Z (*zero flag*) bit ini akan menjadi set apabila hasil operasi matematis menghasilkan bilangan 0.

h. Bit0 → C (*cary flag*) bit ini akan menjadi set apabila suatu operasi menghasilkan *carry*.

2.2.2 Komunikasi USB

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

a. mudah untuk digunakan, sehingga tidak perlu lagi mengotak-atik konfigurasi-konfigurasi dan *setup* yang rumit,

b. cepat, sehingga tidak akan terjadi kemacetan komunikasi pada peranti antarmukanya,

c. dapat dipercaya, karena tingkat kesalahan komunikasi (galat) jarang terjadi, karena menggunakan *metode automatic retries* (pengulang otomatis) ketika terjadi kesalahan,

d. serbaguna, banyak macam perangkat periferal yang dapat menggunakan peranti antarmuka ini,

e. biaya yang minim, sehingga dalam pembuatan peralatannya tidak memerlukan dana yang banyak,

f. daya rendah, artinya dapat menghemat penggunaan daya pada *portable computer* (laptop)

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

Pada setiap komputer masa kini telah terdapat port USB yang dapat digunakan untuk menghubungkan perangkat lain (periferal) seperti *keyboard*, *mouse, scanner, digital camera, printer* dan peralatan lain sebagai perangkat tambahan dengan masing-masing kegunaanya. USB merupakan solusi komunikasi antara komputer dengan perangkat lain yang dibutuhkan oleh sistem komputer tersebut, karena sistem antarmukanya cocok untuk semua tipe perangkat yang standard. Suatu sistem USB pada umumnya terdiri dari beberapa bagian diantaranya :

a. *host controller*, pada sistem USB terdapat beberapa *host* yang bertanggung jawab pada keseluruhan protokol sistem USB. *Host controller* bertugas mengendalikan penggunaan jalur *bus data*, sehingga tidak ada satu pun peralatan USB yang dapat menggunakan jalur *bus data* kecuali mendapat persetujuan dari *host controller*,

b. *hub*, seperti halnya hub untuk jaringan komputer, USB hub menyediakan titik interkoneksi yang dapat memungkinkan banyak peralatan USB untuk dapat terhubung terhadap *host controller*. Topologi jaringan yang digunakan oleh sistem USB adalah topologi star, semua perangkat USB secara logika terhubung langsung dengan *host controller*. Hub terhubung dengan USB host controller secara *upstream* (*data* mengalir menuju ke *host*),

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

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

2.2.2.1 Evolusi Sistem Antarmuka

Alasan utama mengapa suatu sistem antarmuka yang baru tidak begitu sering muncul, hal tersebut dikarenakan sistem antarmuka yang ada telah banyak menarik perhatian pengguna untuk tidak mengambil hal yang rumit untuk membuat sistem antarmuka yang baru. Menggunakan sistem antarmuka yang telah ada membuat lebih hemat biaya dan waktu

dalam pembuatan desain sistem antarmuka peralatannya. Hal inilah yang membuat IBM PC memilih kompatibilitas sistem antarmukanya menggunakan *centronics parallel interface* dan RS-232 *serial port interface* yang sudah ada untuk menghubungkan perangkat pengguna seperti *printer* dan modem yang ada di pasaran.

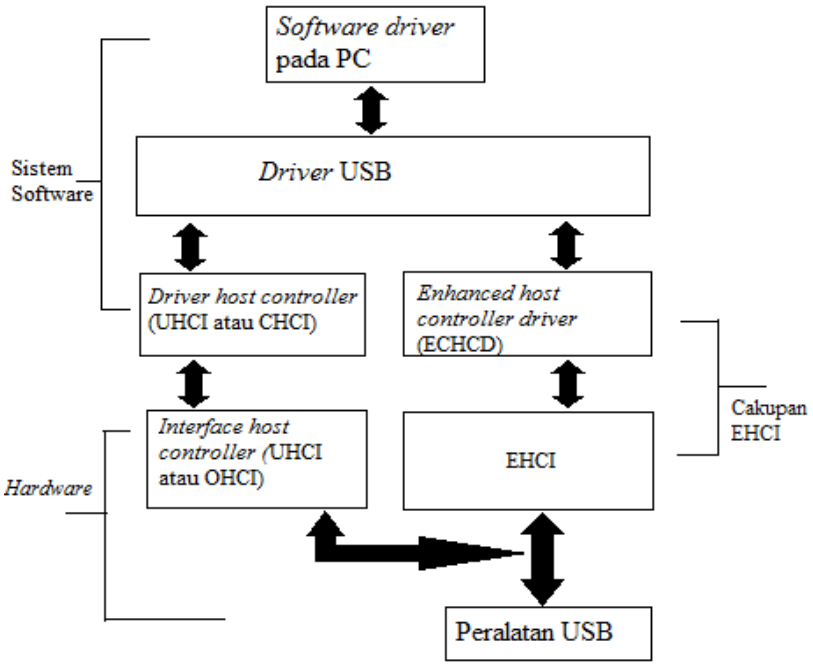
Sistem antarmuka standard yang digunakan IBM PC telah membuktikan kehandalannya dalam 2 (dua) dekade kebelakang. Namun semakin canggih dan hebatnya komputer-komputer masa kini membuat semakin meningkatnya jumlah perangkat-perangkat *(periferal)* pendukung komputer, sehingga sistem antarmuka yang terdahulu sudah tidak dapat menanganinya lagi dikarenakan kecepatan komunikasinya yang terbatas dan memiliki ekspansi antarmuka yang terbatas juga. Hal inilah yang kemudian merujuk kepada pengembangan sistem antarmuka USB.

2.2.2.2 Tugas Komputer sebagai *Host*

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

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

Pada awalnya USB *host controller* didesain untuk menangani satu peralatan USB saja, namun dalam perkembangannya muncullah *Universal/Open Host Controller Intreface* (UHCI/OHCI) yang terdiri dari 2 (dua) bagian yaitu: *host controller driver* dan *host controller*. *Host controller driver* adalah suatu program perangkat lunak komputer yang bertanggung jawab dalam penjadwalan lalu lintas data pada jalur *bus data* dengan cara menempatakan dan menjaga transaksi data dalam sistem memori, sementara *host controller* bekerja memindahkan data dari sistem memori ke peralatan USB dengan cara memproses struktur data. Selanjutnya untuk dapat memenuhi kebutuhan *transfer data* dengan kecepatan tinggi, maka dikembangkan *Enhanced Host Controller Interface* (EHCI). EHCI mampu mendukung peralatan USB *high speed* karena memang didesain untuk efisiensi dalam penggunaan memori pada peranti *host controller*.



Gambar 2.21 Blok Diagram Sistem USB

Sistem USB *host controller* terdiri atas sejumlah lapisan perangkat keras (*hardware*) dan perangkat lunak (*software*) yang ditunjukkan pada Gambar 2.21. Secara umum penjelasan singkat dari setiap lapisan-lapisan tersebut adalah.

a. ***Software driver*** **pada PC**, bertugas mengeksekusi perintah dari *host controller* yang bersesuaian dengan peralatan USB tertentu sesuai fungsinya. *Driver* peralatan USB secara khusus merupakan bagian dari sistem operasi atau yang disertakan dengan peralatan USB yang kemudian di-*install* pada PC sehingga peralatan USB tersebut dapat digunakan sesuai fungsinya,

b. ***Driver* USB**, adalah suatu perangkat lunak *bus driver* yang memisahkan detil *host controller* tertentu untuk sistem operasi tertentu,

c. ***Host controller driver***, berfungsi menyediakan *driver* perangkat lunak diantara lapisan perangkat keras *host controller* dengan pealatan USB. Detail dari *host controller driver* tergantung kepada sistem antarmuka perangkat keras USB *host controller*,

d. ***Host controller***, adalah lapisan implementasi perangkat keras yang spesifik. Terdapat 1 (satu) spesifikasi *host controller* yang digunakan untuk peralatan USB *high speed* yaitu *Enhanced Host Controller Interface* (EHCI), dan terdapat 2 (dua) spesifikasi *host controller* yang digunakan untuk peralatan USB *full speed* dan *low speed* yaitu *Universal Host Controller Interface* (UHCI) dan *Open Host Controller Interface* (OHCI),

e. **peralatan USB**, adalah suatu perangkat keras yang akan melaksanakan fungsinya dengan pengguna. Interaksi antara pengguna dengan peralatan USB mengalir dari aplikasi melalui lapisan-lapisan perangkat lunak dan perangkat keras sesuai dengan diagram pada Gambar 2.21.

2.2.2.3 Lapisan Komunikasi USB

Komunikasi data pada sistem USB terbagi menjadi 3 (tiga) lapisan yaitu : *physical layer*, *protocol engine layer*, dan *application layer*. Logika komunikasi antara perangkat lunak *driver host controller* dengan

fungsi peralatan USB menggunakan sebuah jalur komunikasi pipa (*pipes*). Jalur pipa ini diasosiasikan sebagai jalur antara *endpoint* pada peralatan USB dengan perangkat lunak *host controller* yang bersesuaian.

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

2.2.2.4 USB – HID *Communication Class*

Human interface device (HID) class adalah salah satu kelas komunikasi USB yang didukung oleh hampir seluruh sistem operasi baik itu sistem operasi Windows, mulai dari Windows ’98 hingga yang terbaru, juga sistem operasi yang lain. Biasanya *driver* untuk kelas komunikasi HID ini sudah terangkum menjadi satu kesatuan didalam sistem operasi sehingga tidak perlu lagi untuk meng-*install driver*. Oleh karena itu banyak *vendor-vendor* perangkat USB yang menggunakan kominikasi kelas HID dalam pembuatan produknya.

Pemberian nama HID (*Human Interface Device*) diambil dari perangkat antarmuka yang berhubungan langsung dengan manusia (*human interface*). Sebagai contoh adalah sebuah *mouse* dapat

mendeteksi ketika adanya penekanan tombol ataupun gerakan yang dilakukan oleh manusia sehingga memberikan nilai masukan yang berbeda-beda terhadap komputer untuk melakukan tugasnya.

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

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

b. sebuah antarmuka HID harus memiliki sebuah *IN interrupt* untuk dapat mengirimkan permintaan *report* kepada *host,*

c. sebuah antarmuka HID dapat memiliki setidaknya 1 (satu) buah *endpoint* interupsi *IN* dan 1 (satu) buah *endpoint* interupsi *OUT,*

d. *endpoint* interupsi *IN* memperbolehkan HID untuk mengirimkan informasi kepada komputer (*host*) dalam waktu yang tidak dapat diprediksi. Sebagai contoh tidak ada cara lain sebuah komputer untuk mengetahui ketika pengguna menekan salah satu tombol

pada *keyboard* selain menggunakan transaksi interupsi, sehingga *driver* pada komputer (*host*) menggunakan transaksi interupsi untuk menanyai secara periodik terhadap perangkat apabila terdapat data yang baru yang akan diproses oleh komputer,

e. kecepatan *transfer* data pada perangkat dibatasi, terutama pada USB *low speed* dan USB *full speed*. Sebuah komputer dapat memberikan batasan *endpoint* interupsi untuk kecepatan rendah tidak lebih dari 800 *byte*/detik, dan untuk kecepatan penuh, maksimal *endpoint* interupsinya adalah 64 kByte/detik. Sedangkan untuk kecepatan tinggi, maksimal *endpoint* interupsinya adalah 25 Mbyte/detik apabila komputer (*host*) mendukung *high-bandwidth* bila tidak maka hanya berkisar di 8 Mbyte/detik.

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

2.2.2.5 USB – CDC *Communication Class*

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

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

Dalam bentuk lain, komunikasi USB-CDC ini digunakan untuk menggantikan peranti antarmuka RS-232 yang sudah hampir tak pernah ada di perangkat komputer masa kini, sehingga solusi untuk komunikasi tersebut adalah dengan membuat *virtual comm-port* dimana hal tersebut juga memanfaatkan kelas komunikasi USB-CDC karena kelas komunikasi ini dapat dibentuk sesuai dengan kebutuhan pembuatnya, hanya saja untuk dapat berkomunikasi antara perangkat dengan komputer dibutuhkan *driver* khusus yang dapat menangani komunikasi antara perangakat dengan komputer. Pada perancangan alat ukur IC yang dirancang menggunakan *modul* komunikasi USB *to* TTL. Adapun *modul* USB *to* TTL dapat dilihat pada gambar berikut ini.



Gambar 2.22 *Modul USB to TTL*

2.3 Perangkat Lunak (*Software*)

Perangkat lunak yang digunakan untuk mengontrol kinerja perangkat keras dari aplikasi alat ukur IC ini terbagi atas 2 bagian yaitu pemograman bahasa C pada mikrokontroler dan *visual studio* C# (*sharf*) untuk aplikasi dikomputer. Berikut merupakan uraian tentang masing-masing perangkat lunak tersebut.

2.3.1 Program Bahasa C

Bahasa C pertama kali digunakan di komputer *digital equipment corporation* PDP-11 yang menggunakan sistem opersi UNIX C adalah bahasa yang standar, artinya suatu program yang ditulis dengan bahasa C tertentu akan dapat dikonversi dengan bahasa C yang lain dengan sedikit modifikasi. Standar bahasa C yang asli adalah standar dari UNIX. Deskripsi C dari Kerninghan dan Ritchie ini kemudian dikenal secara umum sebagai "K dan R C".

2.3.1.1 Penulisan Program Bahasa C

Program Bahasa C tidak mengenal aturan penulisan di kolom tertentu, jadi bisa dimulai dari kolom manapun. Namun demikian, untuk mempermudah pembacaan program dan untuk keperluan dokumentasi, sebaiknya penulisan bahasa C diatur sedemikian rupa sehingga mudah dibaca dan dipahami. Berikut contoh penulisan program bahasa C:

```
#include <mega8535.h>
#include <delay.h>
main ()
{
……
……
}
```

Program dalam bahasa C selalu berbentuk fungsi seperti ditunjukkan dalam main (). Program yang dijalankan berada di dalam tubuh program yang dimulai dengan tanda kurung buka **{** dan diakhiri dengan tanda kurung tutup **}**. Semua yang tertulis di dalam tubuh program ini disebut dengan blok.

Tanda () digunakan untuk mengapit argumen suatu fungsi. Argumen adalah suatu nilai yang akan digunakan dalam fungsi tersebut. Dalam fungsi main diatas tidak ada argumen, sehingga tak ada data dalam (). Dalam tubuh fungsi antara tanda { dan tanda } ada sejumlah pernyataan yang merupakan perintah yang harus dikerjakan oleh prosesor. Setiap pernyataan diakhiri dengan tanda titik koma ;.

Baris pertama #include <…> bukanlah pernyataan, sehingga tak diakhiri dengan tanda titik koma (;). Baris tersebut meminta kompiler untuk menyertakan file yang namanya ada di antara tanda <…> dalam proses kompilasi. File-file ini (berekstensi .h) berisi deklarasi fungsi ataupun variable. File ini disebut *header*. File ini digunakan semacam perpustakaan bagi pernyataan yang ada di tubuh program.

Penulisan #include merupakan salah satu jenis pengarah praprosesor (*preprocessor directive*). Pengarah praprosesor ini dipakai untuk membaca file yang di antaranya berisi deklarasi fungsi, definisi konstanta dan beberapa file judul disediakan dalam C. File-file ini mempunyai ciri yaitu namanya diakhiri dengan ekstensi .h (*hexa*). Misalnya pada program #include <stdio.h> menyatakan pada kompiler

agar membaca file bernama stdio.h saat pelaksanaan kompilasi.Bentuk umum dari penulisan #include pada program adalah.

#include “namafile”

Bentuk pertama (#include <namafile>) mengisyaratkan bahwa pencarian file dilakukan pada direktori khusus, yaitu direktori file include. Sedangkan bentuk kedua (#include “namafile”) menyatakan bahwa pencarian file dilakukan pertama kali pada direktori aktif tempat program sumber dan seandainya tidak ditemukan pencarian akan dilanjutkan pada direktori lainnya yang sesuai dengan perintah pada sistem operasi.

2.3.1.2 Tipe Data

Tipe data merupakan bagian program yang paling penting karena tipe data mempengaruhi setiap instruksi yang akan dilaksanakan oleh computer. Misalnya saja 5 dibagi 2 bisa saja menghasilkan hasil yang berbeda tergantung tipe datanya. Jika 5 dan 2 bertipe integer maka akan menghasilkan nilai 2, namun jika keduanya bertipe float maka akan menghasilkan nilai 2.5000000. Pemilihan tipe data yang tepat akan membuat proses operasi data menjadi lebih efisien dan efektif.

2.3.1.3 Konstanta

Konstanta merupakan suatu nilai yang tidak dapat diubah selama proses program berlangsung. Konstanta harus didefinisikan terlebih dahulu di awal program. Konstanta dapat bernilai integer, pecahan,

karakter dan string. Contoh konstanta : 50; 13; 3.14; 4.50005; ‘A’; ‘Bahasa C’.

2.3.1.4 Variabel

Variabel adalah suatu pengenal (*identifier*) yang digunakan untuk mewakili suatu nilai tertentu di dalam proses program. Berbeda dengan konstanta yang nilainya selalu tetap, nilai dari suatu variable bisa diubah-ubah sesuai kebutuhan. Nama dari suatu variabel dapat ditentukan sendiri oleh pemrogram dengan aturan sebagai berikut :

a. terdiri dari gabungan huruf dan angka dengan karakter pertama harus berupa huruf. Bahasa C bersifat *case-sensitive* artinya huruf besar dan kecil dianggap berbeda,
b. tidak boleh mengandung spasi,
c. tidak boleh mengandung simbol-simbol khusus, kecuali garis bawah (*underscore*),
d. panjangnya bebas, tetapi hanya 32 karakter pertama yang terpakai.

2.3.1.5 Deklarasi

Deklarasi diperlukan bila kita akan menggunakan pengenal (*identifier*) dalam program. *Identifier* dapat berupa variabel, konstanta dan fungsi. Berikut penjelasan dari deklarasi pada pemograman mikrokontroler menggunakan bahasa C.

a. Deklarasi Variabel

Bentuk umum pendeklarasian suatu variable adalah: nama_tipe nama_variabel; . Adapun contoh penggunaannya adalah sebagai berikut ini.

int x; // Deklarasi x bertipe integer

b. Deklarasi Konstanta

Dalam bahasa C konstanta dideklarasikan menggunakan *preprocessor* #define, contoh penulisannya.

#define PHI 3.14 #define nim “0111500382”.

c. Deklarasi Fungsi

Fungsi merupakan bagian yang terpisah dari program dan dapat diaktifkan atau dipanggil di manapun di dalam program. Fungsi dalam bahasa C ada yang sudah disediakan sebagai fungsi pustaka seperti printf (), scanf (), getch () dan untuk menggunakannya tidak perlu dideklarasikan.

Fungsi yang perlu dideklarasikan terlebih dahulu adalah fungsi yang dibuat oleh *programmer*. Bentuk umum deklarasi sebuah fungsi adalah :

Tipe_fungsi nama_fungsi(parameter_fungsi);

Contohnya :

float luas_lingkaran(int jari); void tampil(); int tambah(int x, int y);.

2.3.2 Pemograman Visual Studio Bahasa C# (*Sharp*)

C# adalah salah satu bahasa pemprograman *intermediate* yang digunakan oleh programmer untuk membuat *executable programs*/ program yang dapat di eksekusi. C# dapat memisahkan jarak antara aplikasi yang *powerfull* tapi membingungkan seperti C++ dan sangat mudah sekali untuk digunakan , C# memiliki *extension* .CS .

Sebelum menjalankan aplikasi yang dibuat dengan C# , kita memerlukan *common language runtime* (CLR) untuk meng-eksekusi program yang di *generate* dengan C#. Ini kita dapat dari .Net Framework yang dapat kita *download* langsung dari situs Microsoft secara *free* atau otomatis terinstall ke dalam komputer ketika menginstall aplikasi *visual studio* sebagai suatu syarat standar pada prosedur penginstalan.

Berikut ini beberapa kelebihan C#, diantaranya adalah :

a. *flexible*: C# program dapat di eksekusi di mesin komputer sendiri atau di transmiskan melalui web dan di eksekusi di komputer lainnya,
b. *powerful*: C# memiliki sekumpulan perintah yang sama dengan C++ yang kaya akan fitur yang lengkap tetapi dengan gaya bahasa yang lebih diperhalus sehingga memudahkan penggunanya,
c. *easier to use*: C# memodifikasi perintah yang sepenuhnya sama dengan C++ dan memberitahu dimana letak kesalahan kita bila ada kesalahan

dalam aplikasi, hal ini dapat mengurangi waktu kita dalam mencari *error*,

d. *visually oriented*: The .NET *library code* yang digunakan oleh C# menyediakan bantuan yang dibutuhkan untuk membuat tampilan yang *complicated dengan frames, dropdown, tabbed windows, group button, scroll bar* dan *background image*,

e. *secure*: Semua bahasa pemprograman yang digunakan untuk kebutuhan internet mesti memiliki *security* yang benar-benar aman untuk menghindari aksi kejahatan dari pihak lain seperti *hacker*, C# memiliki segudang fitur untuk menanganinya.

2.4 Metode Pengukuran

Dalam metode pengukuran IC yang digunakan pada perangkat yang akan dirancang menggunakan metode *exhaustive test*. *Exhaustive test* adalah teori pengujian IC dengan memakai semua kombinasi masukan dicoba satu persatu sesuai dengan tabel kebenaran. Jadi jika sebuah gerbang yang memiliki N masukkan, maka IC akan diuji sebanyak 2^n kombinasi masukkan. Metode ini tentunya lebih akurat, namun dengan menggunakan semua masukan akan memakan waktu pengujian yang lebih lama dan pemakaian kapasitas memori yang besar.

2.5 Pemilihan Komponen

Pemilihan jenis komponen dalam perancangan dan pembuatan suatu perangkat elektronik mutlak dilakukan karena berdampak langsung pada tingkat efisiensi dan efektifitas perangkat yang dibuat. Beberapa hal yang perlu diperhatikan diantaranya kualitas bahan, tingkat kecepatan dan keakuratan saat komponen bekerja,

bentuk serta ukuran dimensi komponen, sampai *budget*/pengeluaran dana yang digunakan. Sehingga pada perancangan dan pembuatan alat ukur IC ini, pemilihan jenis komponen yang digunakan harus diperhatikan, agar hasil kinerja baik dari sisi kualitas kinerja perangkat sampai jumlah pengeluaran dana yang diperuntukkan bagi pembuatan perangkat alat ukur IC ini dapat ditekan seminimal mungkin.

2.5.1 Latar Belakang Pemilihan Komponen

Latar belakang perbandingan dan pemilihan jenis komponen yang diuraikan pada bab ini dilakukan dengan cara membandingkan komponen-komponen yang digunakan pada rangkaian sistem kontrol elektronik alat ukur IC dengan beberapa komponen dari jenis yang sama namun berbeda dari sisi spesifikasinya dan harga yang pada dasarnya mempengaruhi kinerja dari komponen tersebut terhadap aplikasi alat ukur IC yang dibuat.

2.5.2 Pemilihan Jenis Mikrokontroler

Jenis mikrokontroler yang digunakan pada perangkat alat ukur IC ini adalah mikrokotroler tipe ATmega32 dari keluarga AVR. Uraian mengenai perbandingan jenis mikrokontroler ATmega32 bila dibandingkan dengan beberapa jenis mikrokontroler lainya, dapat dilihat pada Tabel 2.7 dibawah ini.

Tabel 2.7 Uraian Perbandingan Jenis Mikrokontroler

Spesifikasi	**Jenis Komponen**		
	ATmega8535	**ATmega16**	**ATmega32**
Flash	8 KB	16 KB	32 KB
RAM	0,5KB	1KB	2KB
I/O	32	32	32
PIN	40	40	40
Harga	Rp 55.000	Rp 60.000	Rp 60.000

Berdasarkan uraian pada tabel 2.7 diatas, dapat ditarik kesimpulan bahwa jenis mikrokontroler ATmega32 lebih cocok digunakan, karena memiliki kapasitas memori yang cukup sesuai kebutuhan, serta memiliki RAM yang lebih besar sehingga mempengaruhi kecepatan akses data sementara pada mikrokontroler.