PANDUAN 01

Praktik Mikrokontroler

Topik: Pengenalan Bahasa C dan Software CVAVR

A. Struktur Bahasa C pada CVAVR

Penggunaan mikrokontroler yang diterapkan di berbagai alat rumah tangga, otomotif, sampai dengan kendali, membuat mikrokontroler mulai masuk didunia pendidikan. Banyak varian dan type dari mikrokontroler yang dipelajari dan digunakan di dunia pendidikan. Salah satu varian yang banyak dipelajari dan digunakan adalah produk dari ATMEL dengan type keluarga AVR. Banyak software yang dapat digunakan untuk memprogram mikrokontroler keluarga AVR, dengan bahasa pemrograman masing-masing.

Salah satu bahasa pemrograman yang dikembangkan atau digunakan dunia pendidikan adalah bahasa C dengan struktur dan kemudahan yang dimilikinya. Perkembangan bahasa pemrograman yang dimulai dari bahasa tingkat rendah (bahasa assembly/bahasa mesin) sampai dengan bahasa tingkat tinggi (salah satunya bahasa C). Bagi mikrokontroler bahasa assembly merupakan bahasa yang mudah untuk diterjemahkan bagi prosesornya, sehingga dikatakan sebagai bahasa tingkat rendah. Sedangkan bahasa tinggkat tinggi merupakan bahasa yang sulit diterjemahkan oleh prosesor yang ada di didalam mikrokontroler. Pemilihan bahasa C sebagai bahasa pemrograman untuk mikrokontroler dikarenakan mudah dipahami dan diterjemahkan bagi user atau programmer.

Bahasa C memiliki struktur pemrograman yang khusus, selain itu bahasa C memiliki sifat ***case sensitive***. Artinya tersebut adalah bahwa penulisan kata/word program sangat sensitif dengan mendeteksi perbedaan kapital tidaknya huruf yang digunakan. Satu huruf yang berbeda pada satu kata yang diulang, menyebabkan software tidak akan bisa meng-compile seluruh program yang dibuat.

Setiap bahasa pemrograman memiliki type data masing-masing. Type data merupakan jangkauan suatu data yang mampu/dapat dikerjakan/diolah oleh mikroprosesor dalam program yang dibuat. Penggunaan type data ini juga harus sesuai kebutuhan dan disesuaikan dengan fungsi setiap data. Pemilihan penggunaan type data dapat mempengaruhi besarnya memory file yang dibuat. Berikut daftar type data yang dapat digunakan dalam pemrograman bahasa C;

Type	Size (Bits)	Range (jangkauan)
bit	1	0 , 1
Bool, _Bool	8	0 , 1
char	8	-128 sampai 127
unsigned char	8	0 sampai 255
signed char	8	-128 sampai 127
Int	16	-32768 sampai 32767
short int	16	-32768 sampai 32767
unsigned int	16	0 sampai 65535
signed int	16	-32768 sampai 32767
long int	32	-2147483648 sampai 2147483648
unsigned long int	32	0 sampai 4294967295
signed long int	32	-2147483648 sampai 2147483648
Float	32	±1.175e-38 sampai 3.402e38
Double	32	±1.175e-38 sampai 3.402e38

Penggunaan type data bersamaan dengan variable data yang akan digunakan. Penulisan type data sesuai struktur dapat dilihat sebagai berikut:

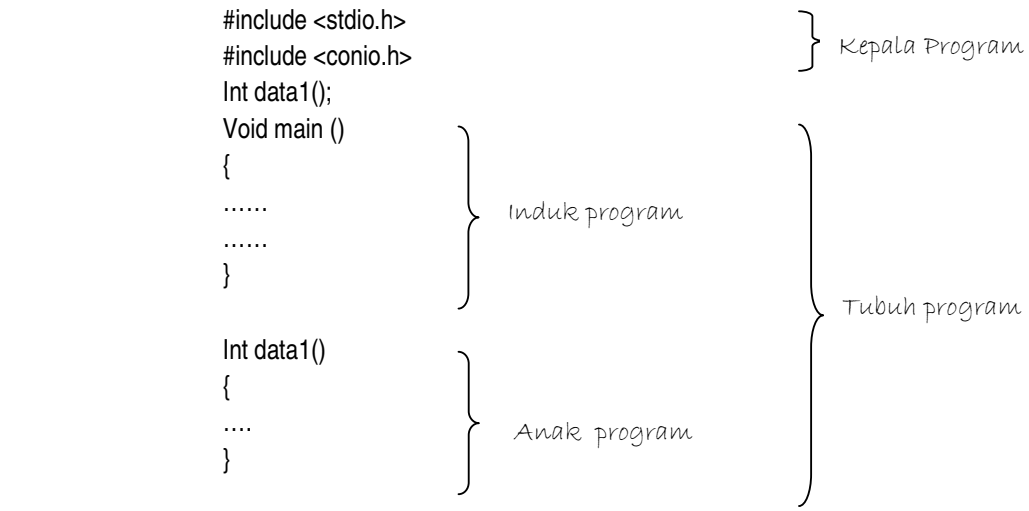
bit data_1; → terdapat variable dengan nama **data_1** dengan type data **bit**

int data_2; → terdapat variable dengan nama **data_2** dengan type data **integer**

Selain tipe data, bahasa C memiliki struktur penulisan akan simbol-simbol operasi aritmatik. Setiap penggunaan simbol-simbol aritmatik memiliki fungsi masing-masing. Berikut table simbol-simbol aritmatik yang digunakan dalam bahasa C;

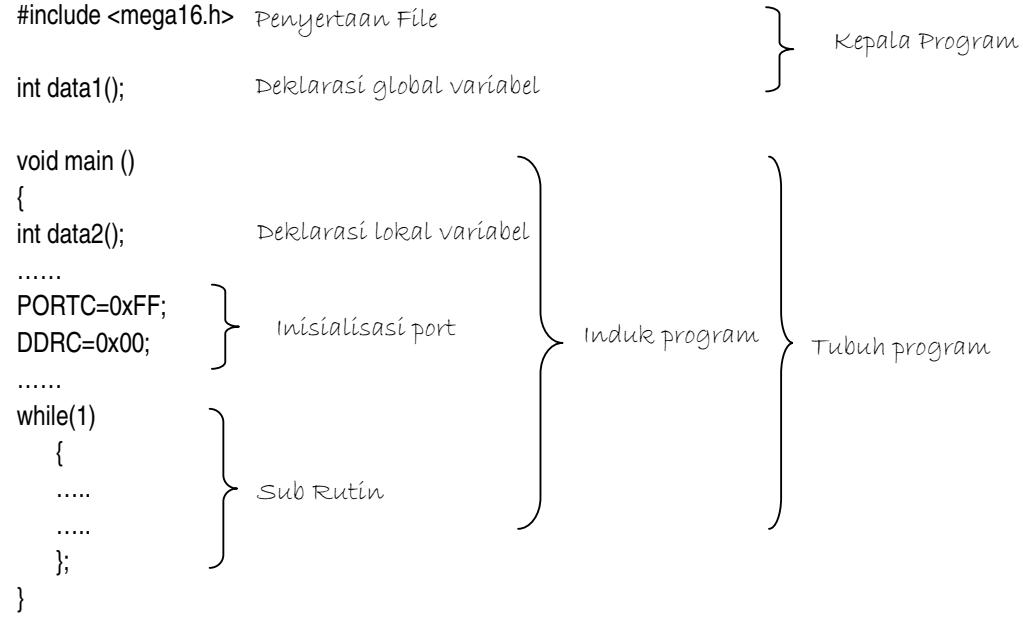
Operator	Keterangan	Operator	Keterangan
+	Penjumlahan	-	Pengurangan
*	Perkalian	/	Pembagian
%	Modulus	++	Penjumalahan berkelanjutan
--	Pengurangan berkelanjutan	=	Sama dengan/memberikan nilai
==	Nilainya sama dengan	~	
!		!=	Hasil tidak sama dengan
<	Lebih kecil	>	Lebih besar
<=	Hasil lebih kecil sama dengan	>=	Hasil lebih besar samadengan
&	Dan/AND	&&	AND (dua kondisi)
|	OR	||	OR (dua kondisi)
^	Faktor pangkat	?:	
<<	Geser bit kekiri	>>	Geser bit kekanan
-=	Hasil pengurangan sama dengan	+=	Hasil penjumlahan sama dengan
/=	Hasil bagi sama dengan	%=	Hasil modulus sama dengan
&=	Hasil peng-AND-an sama dengan	*=	Hasil perkalian sama dengan
^=	Hasil pangkat sama dengan	|=	Hasil peng-OR-an sam dengan
>>=	Hasil penggeseran bit kekanan sama dengan	<<=	Hasil penggeseran bit kekiri sama dengan

Instruksi-instruksi bahasa pemrograman yang ada pada bahasa C tidak semuanya digunakan dalam pemrograman mikrokontroler. Struktur dan urutan penulisan program hampir sama untuk keduanya. Struktur bahasa C memiliki kepala program, dan tubuh program, sedangkan tubuh program bisa terdiri dari induk program dan anak program. Berikut struktur sederhana dari pemrograman bahasa C.



Penggunaan struktur penulisan bahasa pemrograman bahasa C dapat terusun dari sebuah tubuh program yang dapat terdiri dari sebuah induk program dan satu atau lebih anak program. Anak program memiliki fungsi untuk mengerjakan satu blok program yang sering digunakan secara berulang-ulang. Anak program akan diakses oleh induk program sesuai dengan kebutuhan akan sub bagian program tersebut. Sedangkan kepala program berfungsi untuk menyertakan file acuan/library guna mengolah (Compile/Build) program yang telah dibuat.

Penulisan struktur bahasa C didalam CVAVR dapat dilihat seperti dibawah ini:



Deklarasi sebuah variable dapat digolongkan menjadi dua, yaitu local variable dan global variable. *Local variable* dipakai dan hanya dapat diakses pada sub program tempat mendeklarasikannya, sedangkan *global variable* dipakai dan dapat diakses seluruh bagian program. *Inisialisasi PORT* digunakan untuk memfungsikan PORT yang dituju sebagai masukan/keluaran serta nilai defaultnya. Sedangkan bagian *sub rutin* adalah blok program yang akan selalu dikerjakan terus-menerus oleh mikroprosesor selama mikrokontroler hidup.

Beberapa Instruksi-instruksi dalam bahasa C yang sering digunakan dapat ditulis sebagai berikut:

No	Fungsi	Bahasa Pemrograman
1	Syarat	***if (kondisi)*** *{* *……(aksi yang dikerjakan)* *};*
2.	Percabangan	***if (kodisi)*** *{* *……(aksi yang dikerjakan)* *}* ***else if (****kondisi****)*** *{* *…..(aksi yang dikerjakan)* *}* …..

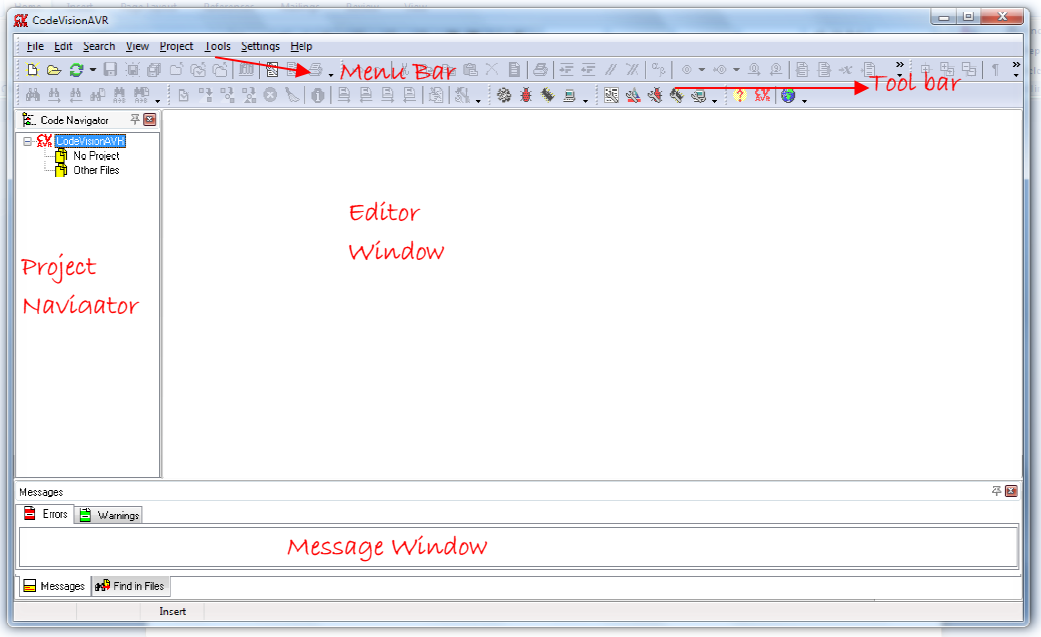
		..... **else** { .....(aksi yang dikerjakan) };
3.	Percabangan	**switch (**variable**)** { **case** nilai_variabel_ke-1**:** { ..... (aksi yang dikerjakan) } **case** nilai_variabel_ke-2**:** { ..... (aksi yang dikerjakan) } .... **default:** { .... (aksi yang dikerjakan) } }
4.	Melompat	**goto** alamat_tujuan; .... alamat_tujuan: ....
5.	Melompat keluar dari perulangan	**break;**
6.	Perulangan	**while (**kondisi**)** { .....(aksi yang dikerjakan) }
7.	Perulangan	**Do** { .....(aksi yang dikerjakan) } **While (**syarat**);**
8.	Perulangan	**for (**nilai_awal,syarat,operasi++/--**)** { .....(aksi yang dikerjakan) **};**

dan masih banyak instruksi program lainnya *(ref: Pemrograman Bahasa C, Abdul Kadir)*.

B. Pengenalan Software Code Vision AVR (CVAVR)

CodeVisionAVR merupakan sebuah cross-compiler C, Integrated Development Environtment (IDE), dan Automatic Program Generator yang didesain untuk mikrokontroler buatan Atmel seri AVR. Cross-compiler C mampu menerjemahkan hampir semua perintah dari bahasa ANSI C, sejauh yang diijinkan oleh arsitektur dari AVR, dengan tambahan beberapa fitur untuk mengambil kelebihan khusus dari arsitektur AVR dan kebutuhan pada sistem embedded.

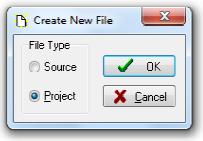
CodeVisionAVR juga mempunyai Automatic Program Generator bernama CodeWizardAVR, yang mengizinkan Anda untuk menulis, dalam hitungan menit, semua instruksi yang diperlukan untuk membuat beberapa fungsi-fungsi tertentu. Dengan fasilitas ini mempermudah para programmer pemula untuk belajar pemrograman mikrokontroler menggunakan CVAVR. Secara garis besar bagian-bagian CVAVR dapat diuraikan seperti gambar berikut ini;



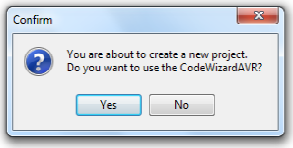
Gb.1 Tampilan window utama CVAVR

Untuk memulai menulis program didalam software CVAVR terlebih dahulu melakukan langkah-langkah sebagai berikut:

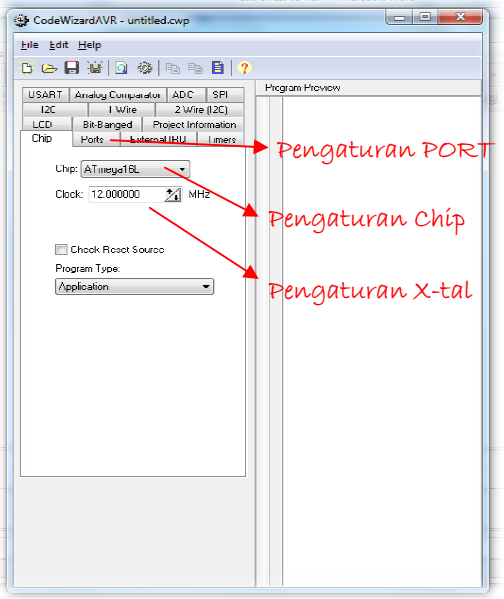
1. File → New → Pilih Project



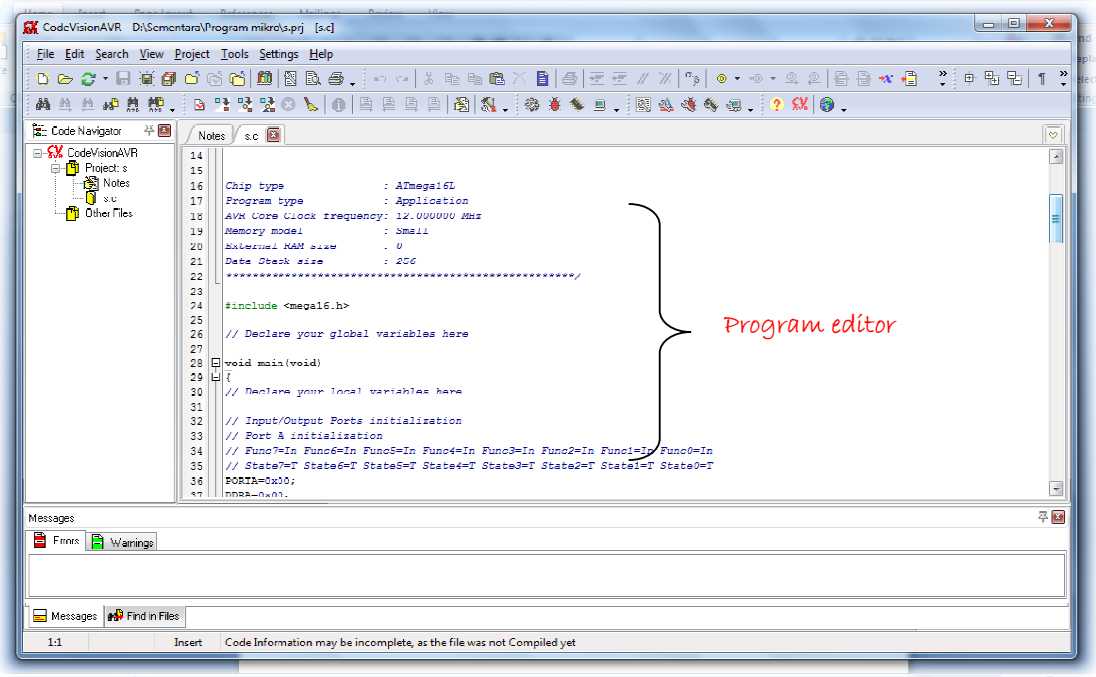
2. Selanjutnya akan muncul window konfirmasi menggunakan AGP CodeWizardAVR → Yes



3. Window CodeWizardAVR digunakan untuk pengaturan PORT dan fasilitas sesuai dengan fungsi yang diinginkan



4. Setelah selesai dengan pengaturan pada CodeWizardAVR pilih File → Generate, Save and exit *(catatan: pemeberian nama file sebanyak 3x; dengan nama file yang sama; hindari kalimat yang panjang, capital dan spasi)*
5. Selesai pemberian nama file, akan muncul window utama editor program seperti berikut



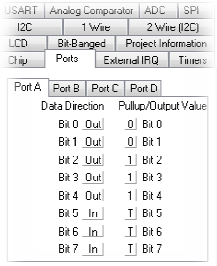
PANDUAN 02

Praktik Mikrokontroller

Topik: Output

A. Kajian Teori

Pheriperal mikrokontroller keluarga AVR (ATMega16/8535) memungkinkan untuk diset sebagai keluaran dan masukan. Pengaturan tersebut dapat dilakukan dengan bantuan Code Wizard AVR pada salah satu port yang diinginkan. Penggunaan program secara langsung juga dapat dilakukan untuk megatur fungsi dari pada setiap port pada mikrokontroller. Berikut gambaran secara umum;



Penulisan secara program:
PORTA=0xFF;
DDRA=0xFF;

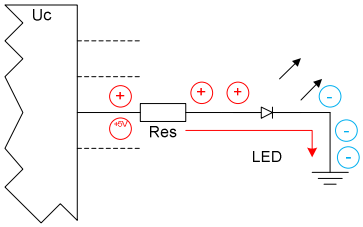
Gb.1a. Pengaturan Port Mikrokontroller CodeWizard 1b. Pengaturan PORT Secara Program

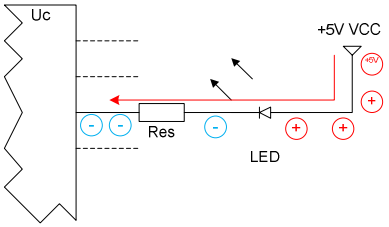
Sebagai contoh pengaturan port-A pada gambar diatas (1a), menunjukkan pada data direction sebagai Output memiliki nilai keluaran dua buah, yaitu 0 (low=0) dan 1 (high=+5V). Nilai keluaran pengaturan port mikro menentukan nilai default awal dari keluarannya. Sedangkan pengaturan port secara program (1b) seperti penulisan diatas, memiliki fungsi pada setiap instruksi sebagai berikur;

PORTA=0xXX; → pengaturan terhadap nilai keluaran Port –A
- *0xFF* → nilai keluaran Port-A pada setiap Bit = Tinggi (*0b11111111*)
- *0x00* → nilai keluaran Port-A pada setiap Bit = Rendah(*0b0000000*)
- *0x0F* → nilai keluaran Port-A pada 4 bit LSB = Tinggi dan 4 bit MSB = Rendah (*00001111*)

DDRA=0xXX; → pengaturan terhadap fungsi port-A
- *0xFF* → Nilai pengaturan port-A pada semua bit sebagai keluaran/output (*0b11111111*)
- *0x00* → Nilai pengaturan port-A pada semua bit sebagai masukan/input (*0b00000000*)
- *0x0F* → Nilai pengaturan port-A pada 4 bil LSB sebagai keluaran dan 4 bit MSB sebagai masukan (*0b00001111*)

Kembali sebagai fungsi keluaran, dapat mempengaruhi kerja dari pada hardware atau rangkaian yang nantinya akan diakses. Ada beberapa tipe kerja rangkaian untuk mengaksesnya, yaitu Aktif LOW dan Aktif High. Aktif LOW merupakan kerja rangkaian yang dapat dioperasikan/di –ON –kan dengan diberi logika rendah (*"0"/0*). Sedangkan Aktif HIGH merupakan kerja rangkaian yang dapat dioperasikan/di-ON-kan dengan diberi logika tinggi ("1"/+5V). Berdasarkan skematik dari kerja rangkaian diatas dapat digambarkan sebagai berikut;





Gb. 2a. Rangkaian dengan kerja Aktif High 2b. Rangkaian dengan kerja Aktif Low

Pengaturan nilai keluaran setiap port disesuaikan dengan prinsip kerja rangkaian yang akan dioperasikan. Secara logika untuk pengaturan nilai keluaran pada setiap port harus berkebaliakn dengan logika untuk menghidupkan/mengoperasikan rangkaian tersebut. Misalkan, rangkaian LED aktif low, maka nilai keluaran pada CodeWizard harus diatur dengan nilai 1/Tinggi. Sedangkan sebaliknya, untuk rangkaian LED aktif high, maka nilai keluaran diatur dengan nilai 0/Rendah. Modul Led yang digunakan dalam praktik memiliki kerja aktif low, sehingga nilai keluaran port-A harus diatur menjadi Tinggi. Pengaturan tersebut dengan tujuan untuk mematikan rangkaian saat pertama kali dihidupkan, atau bisa dikatakan tidak langsung bekerja.

Instruksi yang digunakan dalam CVAVR untuk meng-akses atau mengeluarkan data (output) ke salah satu Port sudah baku. Ada dua macam peng-akses-an port, yaitu secara bersamaan dan secara satu-persatu pin/bit. Sebagai contohnya adalah berikut ini (Akses ke-PORTA);

Instruksi CVAVR Secara bersamaan:

PORTA=0x0F; → pada 8 bit data PORTA akan mengeluarkan data 00001111

atau

PORTA=0b00001111; → pada 8 bit data PORTA akan mengeluarkan data 00001111

Instruksi CVAVR Secara per-bit:

PORTA.0=0; → Pada bit ke-0 PORTA akan mengeluarkan data 0 (low/0)

PORTA.3=0; → Pada bit ke-3 PORTA akan mengeluarkan data 0 (low/0)

PORTA.4=1; → Pada bit ke-4 PORTA akan mengeluarkan data 1 (high/+5V)

PORTA.7=1; → Pada bit ke-4 PORTA akan mengeluarkan data 1 (high/+5V)

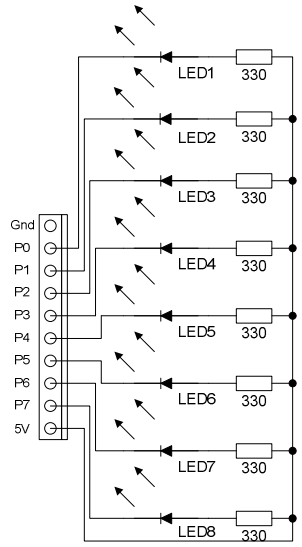
Dst.

Instruksi diatas dapat di ilustrasikan sebagai berikut:

hexa	*0x*	*0*				*F*			
biner	*0b*	*0*	*0*	*0*	*0*	*1*	*1*	*1*	*1*
logika		*low*	*low*	*low*	*low*	*high*	*high*	*high*	*high*
V out		*0*	*0*	*0*	*0*	*5V*	*5V*	*5V*	*5V*
		Bit ke-7	*Bit ke-6*	*Bit ke-5*	*Bit ke-4*	*Bit ke-3*	*Bit ke-2*	*Bit ke-1*	*Bit ke-0*
		MSB bit							*LSB bit*

B. Gambar Rangkaian Hardware

Pada labsheet kali ini akan menggunakan modul tambahan (selaian system minimum) sebagai berikut;



Gb. 3. Skematik Modul LED

C. Contoh program

C.1. Program LED menyala semua secara bersama

```
#include <mega16.h>

Void main(void)
{
……..
……..
While (1)
        {
        PORTA=0x00;
        };
}
```

C.2. Program LED Led-1 On, Led-2 Off, Led-3 On, Led-4 Off, Led-5 On, Led-6 Off, Led-7 On, Led-8 Off

```
#include <mega16.h>

Void main(void)
{
……..
……..
While (1)
        {
        PORTA.0=0;
        PORTA.1=1;
```

```
        PORTA.2=0;
        PORTA.3=1;
        PORTA.4=0;
        PORTA.5=1;
        PORTA.6=0;
        PORTA.7=1;
        };
}
```

C.3. Program LED berkedip bersamaan

```
#include <mega16.h>
#include <delay.h>

Void main(void)
{
……..
……..
While (1)
        {
        PORTA=0xFF;
        delay_ms(1000);
        PORTA=0x00;
        delay_ms(1000);
        };
}
```

C.4. Program LED geser bergantian ke-kanan

```
#include <mega16.h>
#include <delay.h>

Void main(void)
{
……..
……..
While (1)
        {
        PORTA=0b11111111;
        delay_ms(1000);
        PORTA=0b11111110;
        delay_ms(1000);
        PORTA=0b11111101;
        delay_ms(1000);
        PORTA=0b11111011;
        delay_ms(1000);
        PORTA=0b11110111;
        delay_ms(1000);
        PORTA=0b11101111;
```

```
        delay_ms(1000);
        PORTA=0b11011111;
        delay_ms(1000);
        PORTA=0b10111111;
        delay_ms(1000);
        PORTA=0b01111111;
        delay_ms(1000);
        };
}
```

D. Latihan Mandiri

D.1. Buatlah program untuk menyalakan LED geser bergantian kekiri!

D.2. Buatlah program untuk menyalakan LED geser bergantian kekanan-kekiri!

D.3. Aplikatif:

Buatlah program untuk menjalankan 3 buah motor nyala berurutan, dengan jeda waktu penyalaan antara motor satu dengan yang lain mendekati 5 detik (setelah ketiga motor nyala, tetap dipertahankan posisi tersebut/tidak berulang)!

D.4. Aplikatif:

Buatlah program untuk menjalankan 3 buah motor nyala bergantian, dengan jeda waktu penyalaan antara motor satu dengan yang lain mendekati 5 detik (setelah motor ketiga nyala, kembali penyalaan pada motor kesatu)!

D.5. Pakem:

Terdapat akses ke 8 buah lampu/LED. Buatlah program sesuai apa yang anda pikirkan (selain D.1 –D.4)!

PANDUAN 03

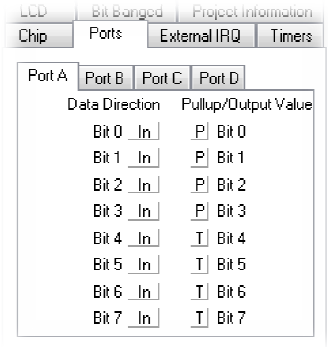
Praktik Mikrokontroller

Topik: INPUT - OUTPUT

A. Kajian Teori

Pada topic sebelumnya sudah dikenalkan cara meng-akses port sebagai keluaran, sehingga selanjutnya pada topic ini akan digabung dengan masukan atau input. Masukan untuk mikrokontroler bisa dari saklar, sinyal logika, atau rangkaian lain yang memiliki keluaran. Sebagai dasar mempelajari masukan pada mikrokontroler, pada topic ini akan digunakan saklar/button sebagai masukannya.

Pengaturan inisialisasi port pada mikrokontroler dapat dilakukan dengan dua cara, secara menggunakan CodeWizardAVR, atau secara penulisan program. Sedangkan sebagai kondisi port sebagai masukan terdapat dua karakter yaitu 'P' dan 'T'. 'P' merupakan kependekan dari Pull Up, sedangkan 'T' merupakan kependekan dari Toggle. Berikut contoh pengaturan port mikro secara CodeWizardAVR atau tertulis;



Secara penulisan Program adalah sebagai berikut:

```
PORTA=0xF0;
DDRA=0x00;
```

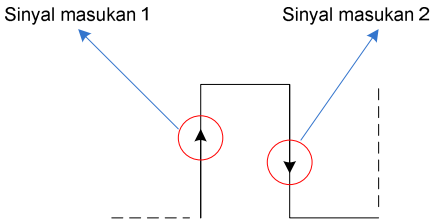
Gb.1. Pengaturan Port sebagai masukan secara CodeWizardAVR

Seperti yang telah dijelaskan pada topic sebelumnya dalam pengaturan secara tertulis inisialisasi port masukan memiliki fungsi sebagai derikut;

PORTA =0x00; → Kondisi 8 bit pada PORTA semuanya Toggle ('T')
=0xFF; → Kondisi 8 bit pada PORTA semuanya Pull up ('P')
=0xF0; → Kondisi 4 bit LSB PORTA berfungsi sebagai Toggle ('T'), sedangkan 4 bit MSB PORTA berfungsi sebagai Pull up ('P').

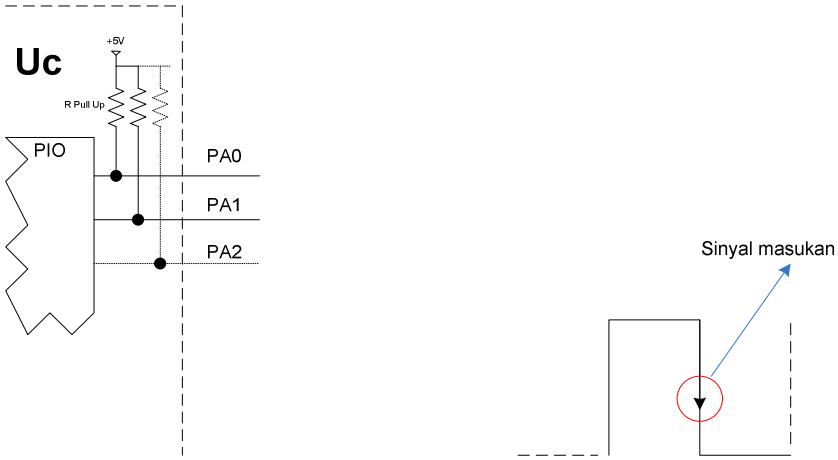
DDRA =0x00; → Semua 8 bit pada PORTA berfungsi sebagai masukan.

Fungsi pada kondisi Toggle masukan mikrokontroler akan membaca sinyal setiap ada perubahan logika. Perubahan itu bisa dari logika tinggi (1) menuju rendah (0) dikatakan sebagai kondisi *falling edge*, atau sebaliknya dari logika rendah (0) ke tinggi (1) dikatakan sebagai kondisi *rising edge*. Prinsip tersebut mengakibatkan dalam pembacaan satu gelombang sinyal terdapat dua kali sinyal masukan ke mikrokontroler. Berikut secara ilustrasi pembacaannya;



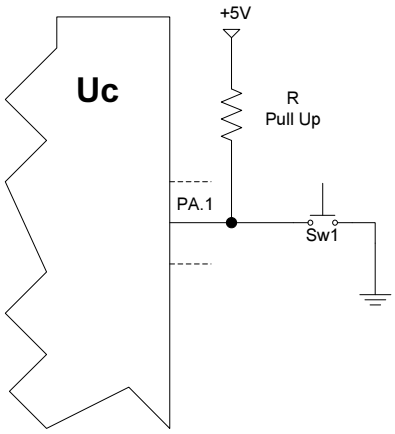
Gb.2. Pembacaan sinyal masukan pada fungsi Toggle

Kondisi pengaturan port masukan pada Pull up ('P') mendeteksi/membaca masukan hanya satu kali dalam satu gelombang masukan. Pembacaan tersebut pada saat gelombang pada kondisi dari logika tinggi (1) ke logika rendah (0) dikatakan sebagai kondidi *falling edge*. Selain itu bahwa pada pengaturan kondisi pull up mengeset pin masukan didalam mikro terhubung dengan VCC (5V) melalui resistor. Resistor yang memiliki prinsip seperti tersebut dinamakan sebagai *resistor pull up*. Resistor ini menjaga agar pada pin masukan yang telah diatur berlogika tinggi, dan menunggu sinyal masukan dengan logika rendah untuk meng-aktif-kannya. Berikut secara ilustrasi prinsip kerja masukan pada kondisi pull up;



Gb.3a. Kondisi didalam mikrokontroller pada posisi Pull Up
Gb.3b. Pembacaan sinyal pada kondisi Pull Up

Kebanyakan rangkaian masukan ke mikrokontroller mengambil prinsip *falling edge* sebagai sinyal tanda aktif, atau bisa dikatakan memiliki logika aktif jika sinyal masukannya rendah (low). Apabila terhubung dengan sebuah masukan dari saklar/button, maka saklar saat tertutup terhubung dengan ground (Gnd). Sebaliknya, apabila saklar dalam kondisi terbuka akan mempertahankan logika tinggi (high) pada masukan, dikarenakan terdapat resistor pull up yang menjaga jalur data masukan dalam kondisi tinggi. Walaupun dalam pengaturan kondisi masukan sudah di pull up, akan tetapi untuk mengamankan kondisi datanya , maka akan dipasang resistor pull up lagi di luar pada system minimum. Berikut ilustrasi skematiknya;



Gb.4 Pemasangan Resistor Pull Up External

Pengambilan data atau mendeteksi sinyal masukan dari luar dilakukan mikroprosesor dengan instruksi program yang telah ditentukan. Instruksi pemrograman dalam bahasa C pada Code Vision AVR yaitu "PINx". Berikut penjabaran penulisan program untuk membaca sinyal data dari luar;

PINA==0b11111101; → pada PORTA bit 1 berlogika rendah (terdapat sinyal masukan), bit 0 dan bit 2-7 berlogika 1 (tidak terdapat sinyal masukan)

Atau,

PINA.1==0; → Pada PORTA bit 0 berlogika rendah yang menunjukkan terdapat sinyal masukan (saklar tertutup)

Instruksi program masukan PIN biasanya digunakan bersamaan dengan dengan intruksi syarat pada bahasa C. Salah satunya yaitu penggunaanya bersama instruksi "IF", berikut contohnya;

```
if(PINA.1==0)
  {
  …….. (aksi yang dilakukan)
  };
```

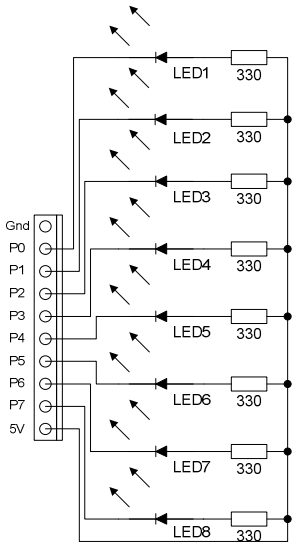
Atau pada perulangan "while";

```
While(PINA.1==0)
  {
  …….. (aksi yang dilakukan berulang-ulang)
  };
```

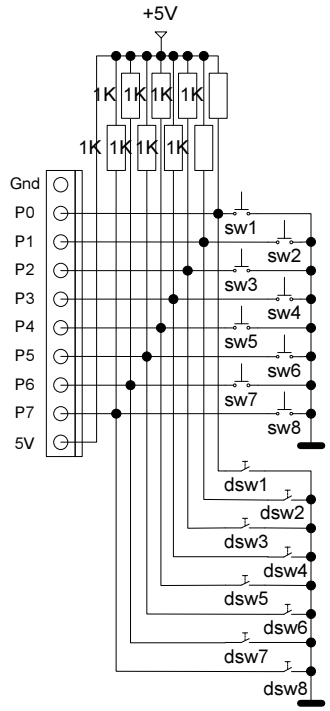
Penggunaan symbol "==" (sama dengan dua kali), mempunyai fungsi sebagai pertanyaan kondisi pada PIN yang dituju. Apakah kondisi PIN masukan dalam kondisi rendah atau pada kondisi tinggi. Sedangkan untuk mengetahui hasil dari pembacaan masukan program masukan (INPUT) digabung dengan program keluaran (OUTPUT).

B. Gambar Rangkaian Hardware

Pada labsheet kali ini akan menggunakan modul tambahan (selaian system minimum) sebagai berikut;



Gb. 5. Skematik Modul LED



Gb.5. Skematik Modul Saklar & Push Button

C. Contoh program

C.1. Jika saklar SW0 ditekan (tertutup) LED0 akan menyala, dan sebaliknya.

```
.......
......
while(1)
        {
        if(PINA.0==0)
                {
                PORTD.0=0;              //LED bit 0 ON
                }
        else
                {
                PORTD.0=1;              //LED bit 0 OFF
                };
        };
```

C.2. Saklar Sw0 untuk menghidupkan LED, Saklar Sw1 untuk mematikan LED

```
.........
.........
while(1)
        {
        if(PINA.0==0)
                {
                PORTD.0=0;              //LED 0 ON
                }
        if(PINA.1==0)
                {
                PORTD.0=1;              //LED 0 OFF
                };
        }
```

D. Latihan Mandiri

D.1. Buatlah program apabila ditekan Sw1 ditekan nyala LED berjalan bergantian ke kiri, apabila ditekan Sw2 nyala LED berjalan bergantian ke kanan.

D.2. Aplikatif

Buatlah program untuk menjalankan dua buah motor nyala bergantian terus menerus dengan jeda pindah mendekati 2 detik.

Sw1 = sebagai START (mulai menjalankan motor bergantian)

Sw2 = sebagai STOP (menghentikan bergantinya jalan motor/berhenti pada salah satu motor)

Sw3 = sebagai RESET (menghentikan dan mematikan semua motor)

D.3. Aplikatif

Buatlah program apabila Sw1 ditekan pertama LED menyala dan akan tetap menyala saat saklar dilepas, pada penekanan Sw1 yang ke dua akan mematikan LED, dan seterusnya.