PERCOBAAN 1

PEMROGRAMAN *GENERAL PURPOSE I/O* SEBAGAI OUTPUT-1 LED GESER

1.1 Tujuan Praktikum

a. Mahasiswa dapat membuat program running LED menggunakan Atmel Studio 7 IDE

b. Mahasiswa dapat menggunakan perintah-perintah geser , perintah-perintah logika, perintah membaca *input*, dan perintah menulis *output*.

1.2 Landasan Teori

1.2.1 Variable Dalam Bahasa C

Semua nama *variable* di dalam bahasa c AVR harus diberi tipe, tipe-tipe *variable* di dalam bahasa c adalah:

a. char = 8 bit yang berkisar -128 s/d 127
b. signed char = 8 bit yang berkisar -128 s/d 127
c. unsigned char = 8 bit yang berkisar 0 s/d 255
d. int = 16 bit yang berkisar -32768 s/d 32767
e. signed int = 16 bit yang berkisar -32768 s/d 32767
f. unsigned int = 16 bit yang berkisar 0 s/d 65535
g. long = 32 bit yang berkisar -2147483648 s/d 2147483647
h. signed long = 32 bit yang berkisar -2147483648 s/d 2147483647
i. unsigned long = 32 bit yang berkisar 0 s/d 4294967295
j. float = 32 bit yang berkisar 1.28E-38 s/d 3.4E38

1.2.2 Format Penulisan Program

Penulisan program di dalam bahasa c dipecah dalam fungsi-fungsi. Setiap fungsi program memiliki tugas yang berbeda, dan dipanggil lewat fungsi utama (main). Setiap fungsi memiliki tubuh fungsi yang selalu dibatasi dengan tanda: {}

```
Void nama_fungsi(void)
{
  ----------perintah_1
  ----------perintah_2
}
```

Penulisan program selalu dimulai dengan header file seperti contoh berikut:

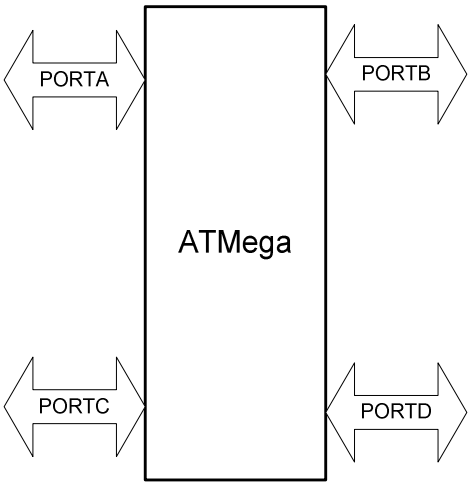
```
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
//Contoh program c AVR
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
//header file
#include <avr/io.h>
#include<util/delay.h>
//fungsi inisialisasi IO-----------------------------------------------------
Void inisialisasi_IO()
{
  -------perintah_1
  -------perintah_2
}

//fungsi utama---------------------------------------------------------------
Int main(void)
{
   Inisialisasi_IO();      //panggil fungsi inisialisasi IO
   while(1)
        {
          ----------perintah_1
          ----------perintah_2
        }
}
```

Proses pelaksanaan program selalu dimulai dari fungsi ***main***, pada awal program fungsi inisialisasi IO dipanggil untuk dilaksanakan. Setelah

kembali dari fungsi inisialisasi_IO proses pelaksanaan program kembali ke fungsi main dan masuk ke dalam loop tertutup for(;;) untuk melaksanakan perintah_1, perintah_2 secara berulang.

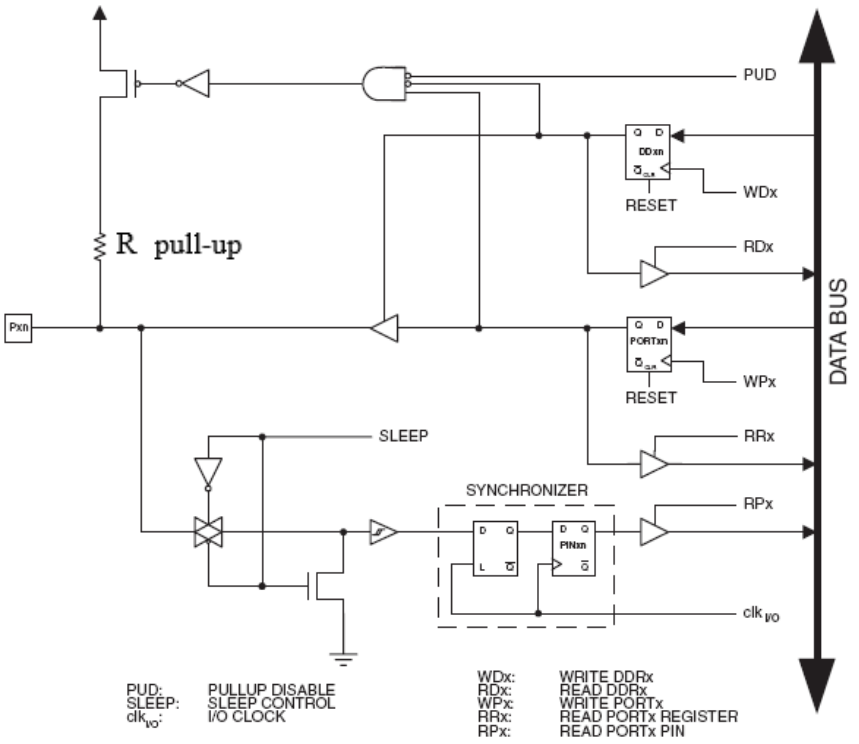
1.2.3 *General Purpose Input Output* (GPIO)

General Purpose Input Output merupakan terminal input dan output digital. ATMega AVR 8535/16/32 memiliki 4 terminal input output yang terdiri atas PORTA, PORTB, PORTC, dan PORTD. Setiap port berisi 8 bit data (1 Byte), sehingga secara keseluruhan berjumlah 32 bit seperti ditunjukkan dalam Gambar 1.1.



Gambar 1.1 GPIO

Setiap PORT mempunyai tiga register DDR (Data Direction Register), PIN (Port Input Register), dan PORT (Port Output). DDR berfungsi untuk menentukan arah dari sebuah port, dibangun sebagai input atau output. PIN merupakan register untuk membaca terminal output. PORT merupakan register terminal output yang berfungsi untuk memegang data (latch). Sebuah pin terminal dapat dilihat dalam Gambar 1.2.



Gambar 1.2 Rangkaian Internal Port 1 bit

Pxn = PORTxn dimana x adalah A, B, C, atau D dan n adalah 0, 1, 2, 3, 4, 5, 6, atau 7, contoh PORTA5. Dalam Gambar 2 terlihat bahwa jika diinginkan Pxn sebagai output maka DDRxn (Data Direction Register) harus berlogika 1 untuk mengaktifkan tri-state. Sehingga data yang dipegang oleh PORTxn akan keluar ke Pxn melewati tri-state. Pada kondisi ini karena DDRxn = 1 akan menyebabkan output gerbang AND = 0 dan output gerbang NOT = 1, sehingga MOSFET OFF yang akan berakibat R pull-up tidak terhubung ke Vcc atau R pull-up tidak terpasang. Pada saat Pxn diinginkan sebagai terminal Input maka DDRxn = 0 yang menyebabkan tri-state OFF sehingga Pxn tidak terhubung ke PORTxn melainkan terhubung ke PINxn (Port Input). Selain itu kondisi DDRxn = 0 menyebabkan masukan gerbang AND = 0 dan secara default PUD = 0, sehingga untuk mengaktifkan R pull-up

ditentukan oleh PORTxn =1. Secara keseluruhan Register-register pada PORTA ditunjukkan dalam Gambar 3.3.

Bit	7	6	5	4	3	2	1	0	
	PORTA7	PORTA6	PORTA5	PORTA4	PORTA3	PORTA2	PORTA1	PORTA0	PORTA
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Bit	7	6	5	4	3	2	1	0	
	DDA7	DDA6	DDA5	DDA4	DDA3	DDA2	DDA1	DDA0	DDRA
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Bit	7	6	5	4	3	2	1	0	
	PINA7	PINA6	PINA5	PINA4	PINA3	PINA2	PINA1	PINA0	PINA
Read/Write	R	R	R	R	R	R	R	R	
Initial Value	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	

Gambar 1.3 Register-Register PORTA

1.2.4 Perintah Pada C AVR untuk mengakses PORT I/O

Untuk membentuk PORTA low sebagai input, PORTA high sebagai output, dan PORTB sebagai output maka perintahnya adalah sebagai berikut:

```
//*****************************************************
//              inisialisasi I/O
//*****************************************************
void inisiasi_IO()
{
    //PortA0 – Port A3 = input dan Port A4 – Port A7 = output
    DDRA = 0b11110000;                  //atau
    DDRA &= ~((1<<PA0)|(1<<PA1)|(1<<PA2)|(1<<PA3));       //dan
    DDRA |= (1<<PA4)|(1<<PA5)|(1<<PA6)|(1<<PA7);
    DDRB = 0xff;   //Port B sebagai output semua atau
    DDRB = 0b11111111;
    DDRA |= (1<<PORTA0);              //R pull-up terpasang pada PORTA0
}

//*****************************************************
```

```
int main(void)
{
   Inisialisasi_IO();
    PORTB = 0xff;
    While(1)
    {
        ----------
    }
    Return 0;
}
//*****************************************************
```

1.3 Peralatan yang Dipergunakan

a. Personal Computer (PC) atau Laptop

b. Program Atmel Studio 7

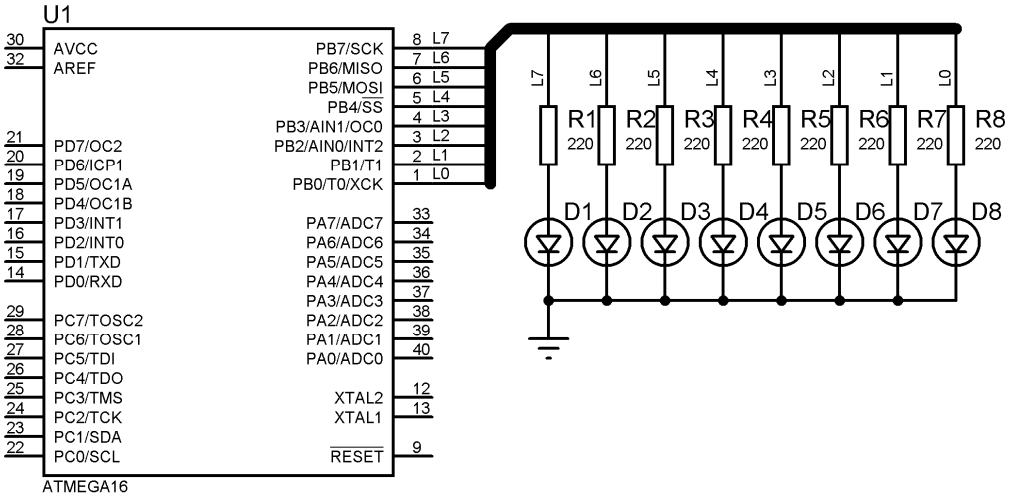
c. Program Proteus

1.4 Tugas 1

a. Buatlah rangkaian seperrti Gambar 1.4. dengan PROTEUS

b. Buat flowchart untuk running LED sedemikian rupa sehingga jika program di jalankan nyala satu (1) LED berjalan dari PORTB0 ke PORTB7 dan kembali lagi ke PORTB0, demikian seterusnya dengan waktu penundaan (delay) 100ms.

c. Buat program dalam Atmel setudio

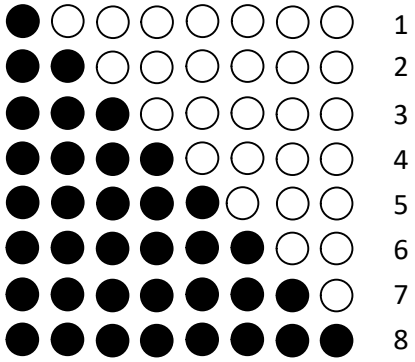


Gambar 1.4 LED aktif tinggi

d. Jika program telah selesai ditulis, lakukan kompilasi sehingga menghasilkan file hex.
e. Masukkan file hex ke dalam Mikrokontroler secara virtual dengan melakukan cick 2x pada Gambar Atmega16
f. Jalankan program dengan menekan tombol play dibagian kiri bawah.
g. Jika Program telah berjalan sesuai dengan yang diinginkan, rekam simulasi tersebut dalam bentuk video.
h. Salin program dengan tulisan tangan pada lembar yang telah disediakan.
i. Buatlah laporan pada lembar yang telah disediakan
j. Upload video ke Google drive, dan sertakan linknya dalam laporan

1.5 Tugas 2

a. Gambar rangkaian tetap seperti Gambar 1.4
b. Buat flochart untuk running LED sedemikian rupa sehingga jika program di jalankan pola nyala LED sperti Gambar 1.5. LED akan menyalaya dengan urutan 1 sd 7 dan kemudian dari 7-1 dan seterusnya.



Gambar 1.5 Pola nyala LED

c. Buat program dalam Atmel Studio.
d. Jika program telah selesai ditulis, lakukan kompilasi sehingga menghasilkan file hex.
e. Masukkan file hex ke dalam Mikrokontroler secara virtual dengan melakukan cick 2x pada Gambar Atmega16

f. Jalankan program dengan menekan tombol play dibagian kiri bawah.
g. Jika Program telah berjalan sesuai dengan yang diinginkan, rekam simulasi tersebut dalam bentuk video.
h. Salin program dengan tulisan tangan pada lembar yang telah disediakan.
i. Buatlah laporan pada lembar yang telah disediakan
j. Upload video ke Google drive, dan sertakan linknya dalam laporan

1.6 Pertanyaan

a. Apa fungsi Register DDR pada AVR?
b. Jelaskan fungsi R pull-up pada Gambar 1.2
c. Jelaskan prosedur pada saat sebuah PORT diinisiasi sebagai PORT input