Syntax

Aturan penulisan program dalam Bahasa C pada ATMEL Studio

Oleh : Agus Pracoyo

Pre Proccessor

Merupakan bagian dari program dalam bahasa c yang selalu dijalankan pertama kali. Bagian ini juga melakukan proses tertentu. Banyak sekali syntax dalam pre-processor. Namun setidaknya ada dua syntax yang akan sering kita gunakan dalam latihan programming dasar, yaitu syntax **#include** dan **#define.**

#include: proses yang pertama kali dijalankan untuk memanggil library/prototype fungsi yang ada di dalam header file.

#include <avr/io.h>

Menyertakan file avr/io.h agar compiler mengenali instruksi-instruksi yang dapat digunakan pada pemrogramman **INPUT/OUTPUT** mikrokontroler seperti perintah DDRA, PORTA, PORTB, PINA, register-registernya dan lain-lain.

#include <util/delay.h>

Menyertakan file util/delay.h agar compiler mengenali perintah pemanggilan fungsi penundaan **_delay_ms()**

Dan masih banyak bentuk header file yang lain.

Bentuk Preprocessor lain

#define : proses yang pertama kali dijalankan untuk mendefinisikan konstanta dan macro. Nilai dari konstanta tidak akan berubah selama program berlangsung.

Contoh:

```
#define F_CPU 1000000  //mendefinisikan freq CPU
#define led PORTB      //menggantikan  PORTB dengan LED
#define nilai 100      //mendefinisikan variabel nilai 100
```

PENULISAN INSTRUKSI

Penulisan instruksi dengan mode Byte

Untuk menuliskan data sekaligus 8 bit

Contoh:

```
DDRA  = 0xff;                 //dengan bilangan hexa decimal
DDRA  = 0b11111111;           //dengan bilangan biner
DDRA  = 255;                  //dengan bilanga decimal
PORTA = 0xff;                 //dengan bilangan hexa decimal
PORTA = 0b11111111;           //dengan bilangan biner
PORTA = 255;                  //dengan bilanga decimal
uint8_t a =  0xa0;
uint8_t a =  0b10100000;
uint8_t a =  160;
```

Penulisan instruksi dengan mode Byte

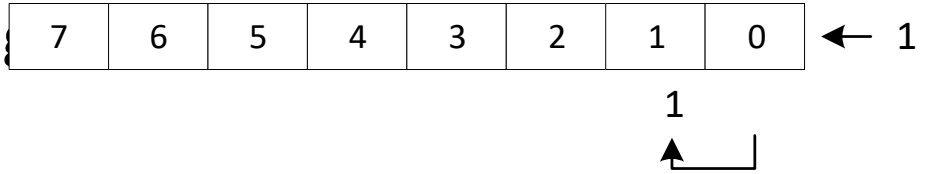
Untuk menuliskan data sekaligus 8 bit

Contoh:

```
DDRA  = 0xff;                 //dengan bilangan hexa decimal
DDRA  = 0b11111111;           //dengan bilangan biner
DDRA  = 255;                  //dengan bilanga decimal
PORTA = 0xff;                 //dengan bilangan hexa decimal
PORTA = 0b11111111;           //dengan bilangan biner
PORTA = 255;                  //dengan bilanga decimal
a =  0xa0;
a =  0b10100000;
a =  160;
```

Penulisan instruksi dengan mode BIT

Untuk menuliskan data hanya pada bit yang diinginkan dan tidak mempengaruhi bit-bit yang



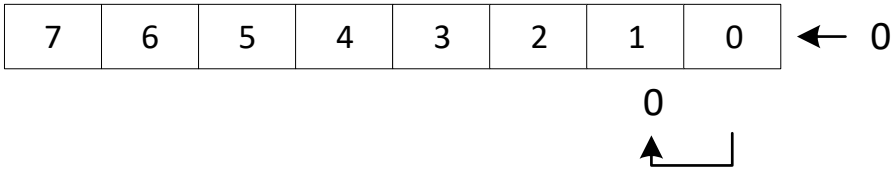
Contoh:

```
DDRA  |= (1<<PA1);
```

Atau:

```
DDRA  |= _BV(PA1);
```

```
DDRA  &=  ~(1<<PA1);
```

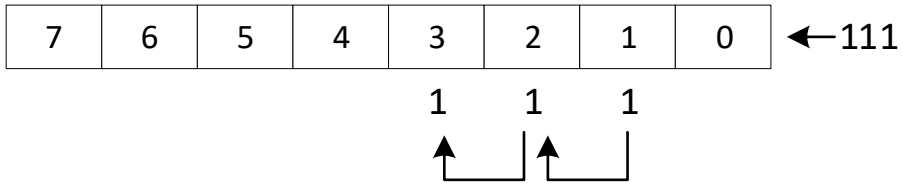


Atau:

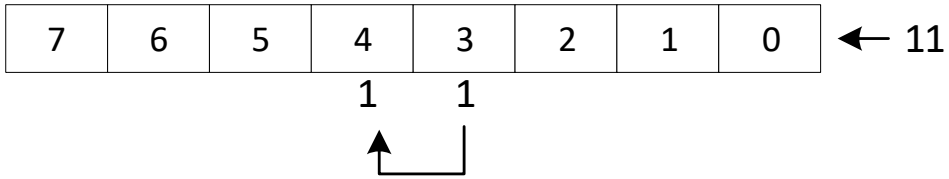
```
DDRA  &= ~_BV(PA1);
```

Contoh lain:

PORTB |= (7<<PB1);



PORTC |= (3<<PC3);



Membuat beberapa bit berlogika 1, yg lain tidak terpengaruh

```
PORTD |= (1<<PD2) | (1<<PD4) | (1<<PD7);
```

PD7	PD6	PD5	PD5	PD4	PD3	PD3	PD2	PD1	PD0	
x	x	x	x	x	x	x	x	x	x	
1				1			1			
										OR
1	x	x	x	1	x	x	1	x	x	

Membuat beberapa bit berlogika 0, yg lain tidak terpengaruh

PORTD &= ~(1<<PD2) & ~(1<<PD4) & ~ (1<<PD7);

```
PD7  PD6  PD5  PD5  PD4  PD3  PD3  PD2  PD1  PD0
 x    x    x    x    x    x    x    x    x    x
 0              0              0
-------------------------------------------------------  &
 0    x    x    x    0    x    x    0    x    x
```

Instruksi geser (Shift instruction)

Yaitu instruksi untuk menggeser nilai biner ke kanan atau kekiri dari data pada Register atau Variable tertentu

Contoh:
PORTC >>= 1 //isi register PORTC digeser kanan 1 bit
Misal mula-mula:

	PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0
→	0	1	0	1	0	0	0	0

Menjadi:

PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0
0	1	0	1	0	0	0	0

Berlaku juga untuk geser kiri

Instruksi logika

Logika AND:

```
uint8_t a = 0b00111111;
PORTA = 0b11000011;
PORTA = PORTA & a;  //atau
PORTA &= a;
```

Maka PORTA menjadi:

```
PORTA = 1 1 0 0 0 0 1 1
a     = 0 0 1 1 1 1 1 1
--------------------------- AND
PORTA  = 0 0 0 0 0 0 1 1
```

Logika OR:

uint8_t a = 0b00111111;
PORTB = 0b11000011;
PORTB = PORTB | a; //atau
PORTB |= a;

Maka PORTB menjadi:
PORTB = 1 1 0 0 0 0 1 1
a = 0 0 1 1 1 1 1 1
---------- OR
PORTB = 1 1 1 1 1 1 1 1

Logika XOR:

unt8_t a = 0b00111111;
PORTC = 0b11000011;
PORTC = PORTC ^ a; //atau
PORTC ^= a;

Maka PORTC menjadi:
PORTC = 1 1 0 0 0 0 1 1
a = 0 0 1 1 1 1 1 1
---------- XOR
PORTC = 1 1 1 1 1 1 0 0

Logika NOT:

PORTC = 0b11000011;
PORTC = ~PORTC

Maka PORTC menjadi:
PORTC = 1 1 0 0 0 0 1 1
--------------------------------- NOT
PORTC = 0 0 1 1 1 1 0 0

Instrksi looping

PERULANGAN for

Contoh:

```
for(uint8_t  a=0; a<=5; a++){
    -------
}
```

Jika syarat terpenuhi maka perintah dlm kurung kurawal dieksekusi dan variabel a bertambah 1 hingga = 5

```
for(; ;){
    -------
}
```

Perulangan terus menerus (tanpa syarat)

PERULANGAN while (kondisi)

Contoh:

```
while( a<=5){
    -------
}
```

Jika kondisi terpenuhi maka perintah dlm kurung kurawal dieksekusi, jika tidak maka akan dilompati

```
while(1){
    -------
}
```

Perulangan terus menerus (tanpa syarat)

if (kondisi) else

Contoh:

```
if( a<=5){
    -------
}
Else{
    -------
}
```

Jika kondisi terpenuhi maka perintah dlm kurung kurawal dieksekusi dan melompat keluar else, jika tidak maka akan ke else

Terima kasish