MODUL 3 Kendali LED Menggunakan ATmega16

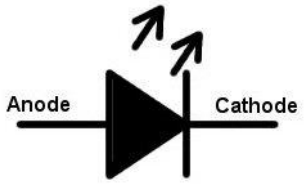
A Tujuan

- Siswa dapat menjelaskan rangkaian antarmuka LED untuk ATmega16.
- Siswa dapat mensimulasikan pengendalian nyala LED dengan ATmega16.
- Siswa dapat menganalisis hasil simulasi kendali LED menggunakan ATmega16.

B Uraian Materi

1. Pengenalan LED (*Light Emmitting Diode*)

LED merupakan singkatan dari Light Emitting Diode. Dari namanya LED termasuk dalam jenis dioda, lebih singkatnya LED adalah dioda yang dapat memancarkan cahaya. Simbol LED dapat dilihat pada Gambar 3.1.



Gambar 3.1 Simbol LED

Seperti yang sudah dijelaskan sebelumnya, LED termasuk jenis dioda sehingga dalam LED juga mengenal adanya bias maju (*forward bias*) dan bias mundur (*reverse bias*). Jika dalam dioda bias maju artinya mengalirkan arus dari anoda ke katoda, maka dalam LED juga sama tetapi dalam peristiwa bias maju ini akan menyalakan cahaya pada LED. sedangkan ketika dibias mundur, maka LED tidak akan menyala yang artinya arus listrik tidak bisa mengalir.

LED dapat memancarkan berbagai warna cahaya, hal ini bergantung pada material semikonduktor penyusun LED. Terdapat berbagai material semikonduktor yang digunakan dalam menghasilkan berbagai warna cahaya pada LED, selengkapnya dapat dilihat pada Tabel 3.1.

Tabel 3.1 Material Semikonduktor Penyusun LED

Material Semikonduktor	Panjang Gelombang Cahaya	Warna Cahaya
Gallium Arsenide (GaAs)	850-940nm	Infra Merah
Gallium Arsenide Phosphide (GaAsP)	630-660nm	Merah
Gallium Arsenide Phosphide (GaAsP)	605-620nm	Jingga
Gallium Arsenide Phosphide Nitride (GaAsP)	585-595nm	Kuning
Aluminium Gallium Phosphide (AlGaP)	550-570nm	Hijau

Lanjutan Tabel 3.1 Material Semikonduktor Penyusun LED

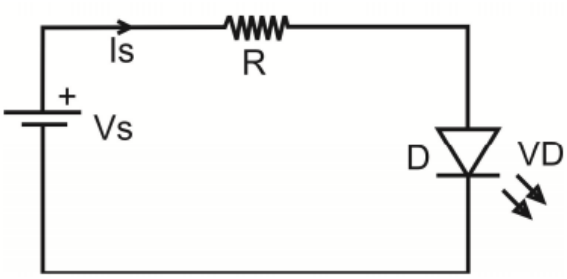
Material Semikonduktor	Panjang Gelombang Cahaya	Warna Cahaya
Silicon Carbide (SiC)	430-505nm	Biru
Gallium Indium Nitride (GaInN)	450nm	Putih

Pada penjelasan sebelumnya diketahui bahwa masing-masing warna LED tersusun dari material semikonduktor yang berbeda, hal ini juga berdampak pada tegangan bias maju LED. masing-masing warna LED memiliki tegangan bias maju yang berbeda-beda. Untuk tegangan bias maju setiap warna LED dapat dilihat pada Tabel 3.2.

Tabel 3.2 Tegangan Bias Maju LED

Warna Cahaya	Tegangan Bias Maju @20mA
Infra Merah	1,2V
Merah	1,8V
Jingga	2,0V
Kuning	2,2V
Hijau	3,5V
Biru	3,6V
Putih	4,0V

Dalam penggunaan LED, hal yang perlu diperhatikan yaitu tegangan pada LED sesuai dengan spesifikasi LED. Penggunaan catu daya yang tidak sesuai dengan spesifikasi LED dapat memutuskan LED, hal ini dikarenakan arus yang mengalir melebihi kemampuan LED. Sehingga perlu ditambahkan resistor untuk menyesuaikan arus yang mengalir pada LED agar LED tidak putus. Pada Tabel 3.2 telah ditunjukkan nilai masing-masing tegangan bias maju untuk setiap warna LED, nilai tersebut digunakan sebagai acuan dalam menentukan besarnya nilai resistor yang harus ditambahkan agar LED bekerja secara normal. Pemasangan resistor dalam rangkaian LED dapat dilihat pada Gambar 3.2.



Gambar 3.2 Rangkaian LED

Pada Gambar 3.2 ditunjukkan bahwa resistor dipasang seri dengan LED untuk membatasi arus yang mengalir pada LED agar sesuai dengan kebutuhan LED. untuk menghitung nilai resistansi resitor yang harus dapasang dapat menggunakan persamaan berikut.

$$R = \frac{V_s - V_D}{I_D}$$

Keterangan:

R : resistansi resistor (Ohm)

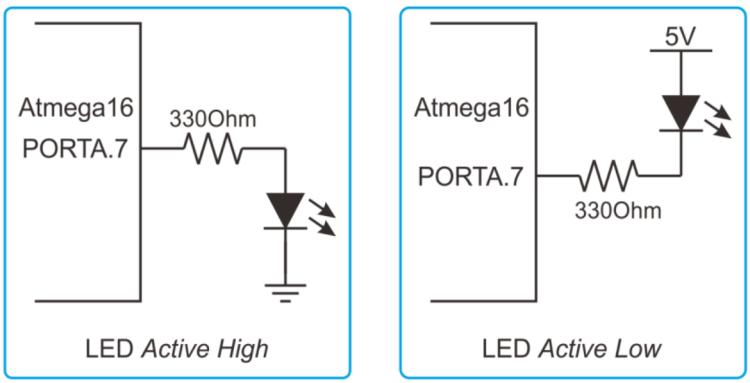
V_s : Tegangan catu daya (Volt)

V_D : Tegangan bias maju LED (Volt)

I_D : Arus ideal LED (Ampere)

2. Antarmuka LED untuk ATmega16

Pemasangan LED pada mikrokontroler ATmega16 terdapat dua cara, yaitu dengan cara aktif *LOW* dan aktif *HIGH*. Perbedaan kedua cara ini terletak pada pemasangan kutub LED terhadap *port* ATmega16. Pada pemasangan aktif *LOW*, LED akan menyala jika keluaran *port* ATmega16 yang terhubung dengan LED bernilai *LOW* (0). Sedangkan pada pemasangan aktif *HIGH*, LED akan menyala jika *port* ATmega16 yang terhubung dengan LED bernilai *HIGH* (1). Agar lebih mempermudah pemahaman konsep aktif *LOW* dan aktif *High* amati Gambar 3.3.



Gambar 3.3 LED Aktif LOW dan Aktif HIGH

3. Kendali Nyala LED menggunakan ATmega16

Sebelum membahas mengenai pengendalian nyala LED menggunakan ATmega16, perlu diperhatikan mengenai cara mengatur penggunaan *pin/port* ATmega16. Sebagaimana telah dibahas pada Modul 1 mengenai fungsi masing-masing *pin/port* pada ATmega16, diketahui masing-masing *pin/port* dapat digunakan sebagai *input* atau *output*. Hal ini perlu diperhatikan dalam penggunaan ATmega16. Pada modul ini pemrograman ATmega16 dilakukan melalui *software* CodeVision AVR, pada tahap awal pembuatan *project* di CodeVision AVR bisa dilakukan pengaturan kegunaan masing-masing *pin/port* ATmega16, digunakan sebagai *input* atau *output*.

Pengaturan penggunaan *pin/port* sebagai *input* atau *output* bisa dilakukan dengan cara menuliskan kode program inisialisasi *port*. Pada bagian inisialisai *port* terdapat dua perintah sebagai berikut.

```
PORTX=0xdata1;
DDRX=0xdata2;
```

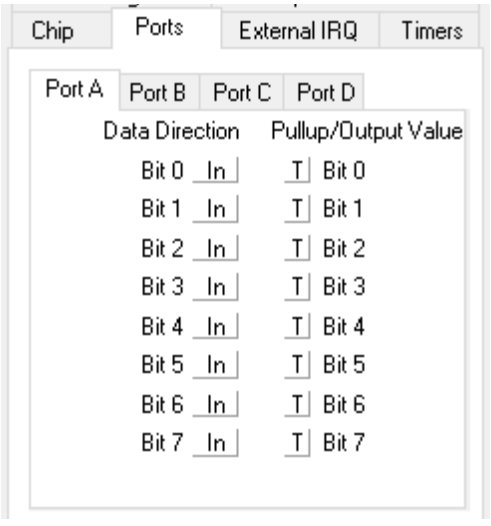
Keterangan:

- Nilai X menunjukkan *port* mana yang diatur, untuk ATmega16 bisa berisi A,B,C, dan D.
- Nilai data1 menunjukkan pengaturan *Data Direction*, sebagai *input* atau *output*. Nilai 0 mewakili *input* dan nilai 1 mewakili *output*.
- Nilai data2 menunjukkan pengaturan *Pullup/Output Value*.
    - Pada *Data Direction* sebagai *Input*, maka nilai 0 mewakili *tristate* dan nilai 1 mewakili *pull up*.
    - Pada *Data Direction* sebagai *Output*, maka nilai 0 mewakili kondisi awal keluaran 0 dan nilai 1 mewakili kondisi awal keluaran 1.

Contoh:

```
PORTA=0x00;     //seluruh input pada port A berjenis tristate.
DDRA=0x00 ;     //seluruh pin pada port A digunakan sebagai input.
PORTB=0x00;     //seluruh output port B berkondisi awal 0.
DDRB=0xff;      //seluruh pin pada port B digunakan sebagai output.
PORTC=0xf0;     //pinC0-pinC3 berjenis tristate, pinC4-pinC7 berkondisi awal 1.
DDRC=0xf0;      //pinC0-pinC3 digunakan sebagai input, pinC4-pinC7 digunakan
                sebagai output
```

Selain dengan cara menulisakan langsung inisialisasi *port* pada kode program, ada cara lain yang dapat digunakan dalam penentuan penggunaan *pin/port* ATmega16. Cara tersebut dapat dilakukan ketika pembuatan *project* baru pada CodeVisionAVR. Terdapat kotak dialog pengaturan *pin/port* ATmega16. Untuk lebih jelasnya perhatikan Gambar 3.4.



Gambar 3.4 Pengaturan *Port* pada CodeVisionAVR

Pada Gambar 3.4 ditampilkan pengaturan *port* A pada ATmega16. Terdapat dua kolom pengaturan, yaitu *Data Direction* dan *Pullup/Output Value.* Pada *Data Direction* terdapat dua pilihan, yaitu *In* dan *Out.* Pengaturan ini akan menentuka pin tersebut akan digunakan sebagai *input* atau *output*. Jika yang dipilih *In*, maka *pin* tersebut akan digunakan sebagai *input*. Sedangkan jika yang dipilih *Out*, maka *pin* tersebut akan digunakan sebagai *output*.

Pada kolom selanjutnya ada pengaturan *Pullup/Output Value*, pengaturan ini menentukan jenis input yang digunakan dan kondisi awal output. Jika pada *Data Direction* dipilih *In*, maka pada pengaturan *Pullup/Output Value* akan muncul pilihan T (*tristate*) dan P (*pull up*). Jika pada *Data Direction* dipilih *Out*, maka pada pengaturan *Pullup/Output Value* akan muncul pilihan 1 dan 0, pilihan ini menunjukkan kondisi awal keluaran *pin*.

Untuk menyalakan LED, *pin/port* ATmega16 harus diatur sebagai *output.* Setelah pengaturan tersebut dilakukan, ATmega16 bisa digunakan untuk mengendalikan nyala LED sesuai dengan program yang dibuat.

Pengendalian nyala LED dengan ATmega16 dapat dilakukan dengan dua cara, yaitu dengan mengendalikan 1 bit LED atau dengan mengendalikan 1 byte (8 bit) LED. pengendalian 1 bit LED dapat dilakukan dengan menuliskan program berikut.

```
PORTX.n = data;
```

Penulisan X menunjukkan posisi *port* dimana LED terhubung mikrokontroler, pada ATmega16 dapat menggunakan *port* A, B, C, atau D. Penulisan n menunjukkan posisi bit pada *port* ATmega16 yang terhubung dengan LED, yaitu dapat bernilai 0 sampai dengan 7. Variabel data menunjukkan logika keluaran *port* yang dikendalikan, dapat diisi dengan nilai 0 atau 1.

Contoh: `PORTA.0 = 1;`

Dari contoh diatas, program yang dibuat akan menghasilkan logika 1 pada keluaran *port* A bit 0.

Sedangkan untuk cara kedua, yaitu dengan mengatur 1 *byte* secara langsung. Terdapat empat cara yang dapat digunakan, yaitu menggunakan sistem bilangan biner, heksadesimal, oktal, dan desimal. Untuk penggunaan sistem bilangan biner penulisan diawali dengan 0b. Untuk penggunaan sistem bilangan heksadesimal penulisan diawali dengan 0x. Untuk penggunaan sistem bilangan oktal penulisan diawali dengan 0. Sedangkan untuk sistem bilangan desimal dengan cara menuliskan langsung nilainya.

```
PORTX = 0bdata; //biner
PORTX = 0xdata; //heksadesimal
PORTX = 0data; //oktal
PORTX = data; //desimal
```

Contoh:
```
PORTA = 0b10000000;
PORTA = 0x80;
PORTA = 0200;
PORTA = 128;
```

Keempat contoh di atas memiliki nilai yang sama jika dikonversi menjadi satu sistem bilangan yang sama. Sehingga hasil keluaran pada *port* A akan memiliki nilai yang sama.

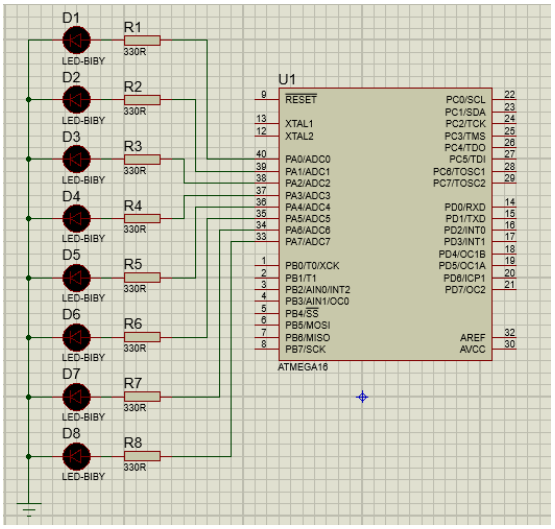
C Rangkuman

- LED merupakan jenis dioda yang dapat memancarkan cahaya.
- Terdapat dua cara penggunaan LED pada mikrokontroler, yaitu Aktif *High* dan Aktif *Low*.
- Terdapat dua cara dalam mengendalikan nyala LED dengan mikrokontroler, yaitu pengendalian 1 bit LED dan 1 byte LED.
- DDRX=0xNN digunakan untuk mengatur *data direction pin/port*, digunakan sebagai *input* atau *output*.
- PORTX=0xNN digunakan untuk mengatur jenis *input* atau kondisi awal *output*.

D Lembar Kerja Simulasi

1. Langkah kerja

1) Buka program ISIS Proteus.
2) Susun rangkaian sesuai Gambar 3.5.



Gambar 3.5 Skema Rangkaian Kendali LED dengan ATmega16

3) Buka program CodeVisionAVR.
4) Buat *file* baru dengan cara klik *File >> New*, atau klik *icon Create a New File or Project*, atau ketik Ctrl + N. Kemudian muncul *dialog box*, pilih *Project* pada *File Type*, kemudian klik *OK*.