BAB II
SIMULASI DAN PEMROGRAMAN MIKROKONTROLER

2.1. Pengenalan Bahasa C dan Simulasi Mikro kontroler

2.1.1. Struktur bahasa C

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. Banyak *software* yang dapat digunakan untuk memprogram mikrokontroler 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). Untuk 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: int data_2; => terdapat *variable* bertipe data *integer* dengan nama data_2. Selain tipe data, bahasa C memiliki struktur penulisan dengan simbol-simbol operasi aritmatik. Setiap penggunaan simbol-simbol aritmatik memiliki fungsi masing-masing. Berikut tabel 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	~	NOT
!		!=	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.

```
Void setup() {
  //put your setup code here, to run once:
}

Void loop() {
  //put your main code here, to run repeatedly:
}
```

Induk program

Anak program

Deklarasi sebuah variabel 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 (program inti) adalah blok program yang akan selalu dikerjakan terus-menerus oleh mikroprosesor selama mikrokontroler hidup.

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

A. Void setup()

Fungsi setup() digunakan untuk inisialisasi program, fungsinya dijalankan sekali yaitu ketika arduino pertama kali dihidupkan atau setelah program selesai di *upload*.

```
void setup() {
    pinMode(13, OUTPUT);
}
```

B. Void loop()

Fungsi loop() dijalankan terus-menerus (*looping forever*) hingga Arduino dimatikan.

```
void loop() { //isi kode}
```

C. pinMode()

Fungsi pinMode() adalah inisialisasi bahwa pinLED berupa *Output*. Dengan demikian mikrokontroler tidak akan "membaca" logika pin tersebut, akan tetapi dia hanya akan "menulis" logika pada pin tersebut. Dengan kata lain, jika kita ingin mendefinisikan bahwa pin ini adalah *input*, maka kita tinggal mengubah OUTPUT menjadi INPUT.

```
Int pinLED=13;
void setup() {
    pinMode(pinLED, OUTPUT;
}
```

D. digitalWrite()

Fungsi digitalWrite() yaitu menyatakan dan memberikan perintah kondisi ke perangkat yang terhubung dengan pin digital. Karena menggunakan pin Digital sehingga untuk perintahnya terdapat dua kondisi yaitu HIGH dan LOW.

```
void loop() {
    digitalWrite(pinLED, HIGH);
    digitalWrite(pinLED, LOW);
}
```

E. analogWrite()

Fungsi analogWrite() sama dengan digitalWrite() hanya saja yang membedakan adalah perintah yang diberikan ke perangkat yang terhubung dengan pin Analog. Karena menggunakan pin

analog untuk kondisi perintah yang diberikan 0 - 255 seperti contoh berikut.

```
void loop() {
    analogWrite(pinLED, 0);
    analogWrite(pinLED, 255);
}
```

F. digitalRead()

digitalRead() berfungsi untuk menerima data yang dibaca dari suatu perangkat, biasanya perintah digitalRead() ini digunakan pada perangkat sensor-sensor yang terhubung dengan pin Digital yang hanya mendeteksi 2 kondisi yaitu HIGH (1) atau LOW (0), dimana datanya bisa langsung dapat disimpan kedalam sebuah *variable* seperti pada contoh.

```
void loop() {
    var = digitalRead(pinSENSOR);
}
```

G. analogRead()

analogRead() sama seperti digitalRead() yang ditujukan untuk menerima sebuah data dari suatu perangkat yang terhubung dengan pin analog, biasanya digunakan pada sensor yang pembacaan datanya lebih detail (kontinyu), dan pembacaan datanya dapat langsung di simpan kedalam sebuah variable.

```
void loop() {
    var = analogRead(pinSENSOR);
}
```

H. Delay()

fungsi delay() adalah memberikan jeda eksekusi program selama waktu yang telah ditentukan. Untuk satuan waktu yang digunakan pada delay() adalah milisecond.

```
void loop() {
    digitalWrite(pinLED, HIGH);
    delay(1000);
}
```

I. Serial.begin()

Serial.begin() berfungsi untuk mengatur kecepatan aliran data (*baud rate*) dengan satuan kecepatan yaitu bit per second (bps) digunakan untuk mengirimkan data serial.

```
void setup() {
    Serial.begin(9800);
}
```

J. Serial.print() / serial.println()

serial.print() digunakan untuk menampilkan tulisan atau output dari suatu variable di serial monitor, untuk serial.println() digunakan untuk menampilkan tulisan atau data di baris baru secara otomatis.

```
void loop() {
    serial.print("data : ");
    serial.print(variable);
    serial.println();
}
```

2.2. Pemrograman Mikrokontroler Arduino

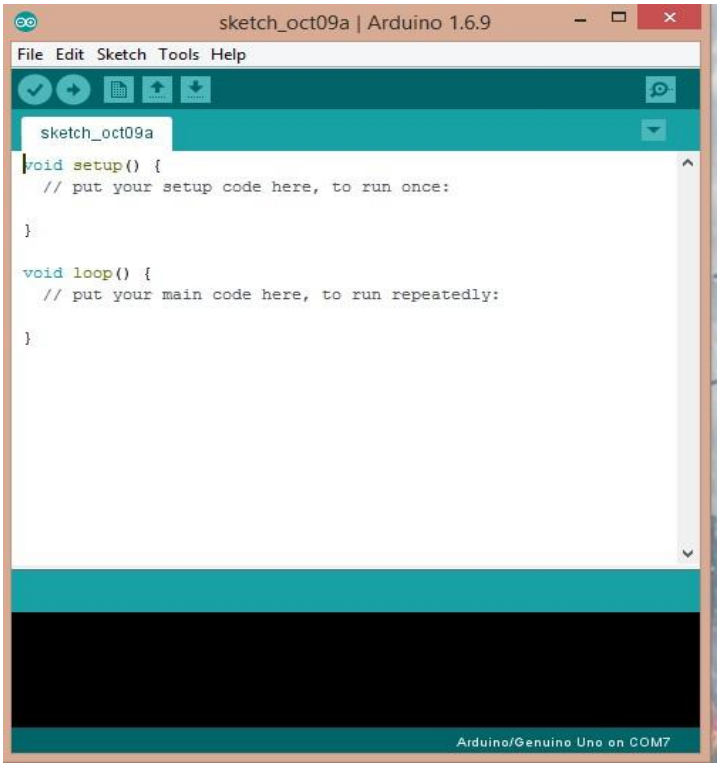
Untuk memprogram board Arduino, kita membutuhkan aplikasi IDE (*Integrated Development Environment*) bawaan dari Arduino. Aplikasi tersebut bisa di *download* di *official website* Arduino. Aplikasi ini berguna sebagai *text* editor untuk membuat, membuka, mengedit, dan juga memvalidasi kode serta untuk di upload ke *board* Arduino. Program yang digunakan pada Arduino disebut dengan istilah "*sketch*" yaitu file *source code* arduino dengan ekstensi .ino

2.2.1. Arduino IDE

IDE itu merupakan kependekan dari *Integrated Development Environment*, atau secara bahasa mudahnya merupakan lingkungan terintegrasi yang digunakan untuk melakukan pengembangan. Disebut sebagai lingkungan karena melalui *software* inilah Arduino dilakukan pemrograman untuk melakukan fungsi-fungsi yang dibenamkan melalui sintaks pemrograman. Arduino menggunakan bahasa pemrograman sendiri yang menyerupai bahasa C. Bahasa pemrograman Arduino (*Sketch*) sudah dilakukan perubahan untuk memudahkan

pemula dalam melakukan pemrograman dari bahasa aslinya. Sebelum dijual ke pasaran, IC mikrokontroler Arduino telah ditanamkan suatu program bernama *Bootlader* yang berfungsi sebagai penengah antara *compiler* Arduino dengan mikrokontroler.

Arduino IDE dibuat dari bahasa pemrograman JAVA. Arduino IDE juga dilengkapi dengan *library* C/C++ yang biasa disebut *Wiring* yang membuat operasi *input* dan *output* menjadi lebih mudah. Arduino IDE ini dikembangkan dari *software Processing* yang dirombak menjadi Arduino IDE khusus untuk pemrograman dengan Arduino.



Gambar di atas merupakan tampilan dari Software Arduino IDE

Verify

berfungsi untuk melakukan checking kode yang kamu buat apakah sudah sesuai dengan kaidah pemrograman yang ada atau belum

Upload

Berfungsi untuk melakukan kompilasi program atau kode yang kamu buat menjadi bahsa yang dapat dipahami oleh mesih alias si Arduino.

New

Berfungsi untuk membuat *Sketch* baru

Open

Berfungsi untuk membuka *sketch* yang pernah kamu buat dan membuka kembali untuk dilakukan editing atau sekedar upload ulang ke Arduino.

Save

Berfungsi untuk menyimpan *Sketch* yang telah kamu buat.

Serial Monitor

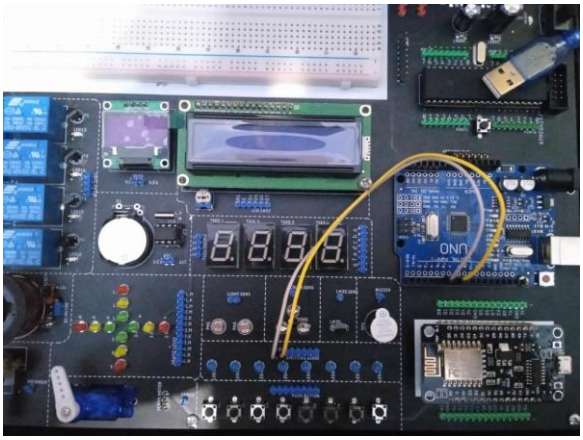
Berfungsi untuk membuka serial monitor. Serial monitor disini merupakan jendela yang menampilkan data apa saja yang dikirimkan atau dipertukarkan antara arduino dengan sketch pada port serialnya. Serial Monitor ini sangat berguna sekali ketika kamu ingin membuat program atau melakukan *debugging* tanpa menggunakan LCD pada Arduino. Serial monitor ini dapat digunakan untuk menampilkan nilai proses, nilai pembacaan, bahkan pesan error.

2.2.2. Tugas Pratikum

1. **Gambar rangkaian**

a. Rangkaian Led Flip – Flop

Pada Rangkaian Dibawah Led A ke pin 6 Dan Led B ke pin 7



Gambar 1 rangkaian Arduino

2. Source Code

```
int a = 6;
int b = 7;

void setup() {
  pinMode(6,OUTPUT);
  pinMode(7,OUTPUT);
}

void loop() {
  digitalWrite(a,HIGH);
  digitalWrite(b,LOW);
  delay(500);
  digitalWrite(a,LOW);
  digitalWrite(b,HIGH);
  delay(500);
}
```

2.3. If Else dan For While

2.3.1. Perintah If dan Else

Perintah IF memiliki beberapa kombinasi, bisa IF saja, IF-ELSE, IF-ELSE IFELSE, dan seterusnya. Semakin kompleks tentu logika yang dipakai akan tampak semakin "rumit"

a. Perintah If

Fungsi IF dalam bahasa pemograman adalah untuk menjalankan (mengeksekusi suatu program) yang apabila syaratnya atau syarat tertentu telah terpenuhi yang mana bisa terdiri hanya atas satu keadaan (syarat) atau banyak keadaan tergantung dari program tersebut. Berikut salah satu sketch LED menggunakan switch(button):

```
int a = 13;
int b = 12;
int c = 11;
int d = 10;
#define Button A1
#define Button A2
void setup() {
  // put your setup code here, to run once:
pinMode(a,OUTPUT);
pinMode(b,OUTPUT);
pinMode(c,OUTPUT);
pinMode(d,OUTPUT);
}
```

b. Perintah If/Else

Pada dasarnya IF-ELSE merupakan pengembangan dari IF. ELSE berarti kondisi yang tidak sesuai dengan kondisi dalam IF. Dengan kata lain, ELSE artinya "jika tidak".

Berikut salah satu sketch LED menggunakan *switch*(*button*)

```
      int a = 13;
      int b = 12;
      int c = 11;
      int d = 10;
      #define Button A1
      void setup() {
        // put your setup code here, to run once:
      pinMode(a,OUTPUT);
      pinMode(b,OUTPUT);
      pinMode(c,OUTPUT);
      pinMode(d,OUTPUT);
      }

      void loop() {
        // put your main code here, to run repeatedly:
      if(digitalRead(A1))
      {
        digitalWrite(a,HIGH);
        digitalWrite(b,HIGH);
        digitalWrite(c,HIGH);
        digitalWrite(d,HIGH);
      }
      else
      {
        digitalWrite(a,LOW);
        digitalWrite(b,HIGH);
        digitalWrite(c,LOW);
        digitalWrite(d,HIGH);
      }
}
```

2.3.2. Perulangan dengan For

Perintah "for" digunakan untuk mengulang suatu blok program yang terdapat di dalam kurung kurawal setelah pernyataan "for". Format penulisan pernyataan "for" adalah sebagai berikut:

```
for (inisiasi; test kondisi; proses increment/decrement)
{
pernyataan yang akan diulang;
}
```

Inisiasi dilakukan pertama-tama ketika pernyataan "for" dijalankan dan hanya dilakukan satu kali saja yaitu pada saat awal pernyataan "for" dijalankan. Selama proses pengulangan terjadi, selalu dilakukan test kondisi untuk mengetahui apakah variabel masih masuk dalam range yang ditentukan atau masih bernilai "benar" atau tidak.

Pengulangan terus berlangsung hingga test kondisi “salah”. Bila nilai test kondisi “salah”, maka proses pengulangan dihentikan dan program lanjut membaca pernyataan berikutnya. Berikut salah satu sketch LED menggunakan for

```
int a = 10;

void setup() {
pinMode(a,OUTPUT);
}

void loop() {
 for( int i=1; i<=255; i++)
   {
   analogWrite(a, i);
   delay(10);
   }
}
```

2.3.3. Perulangan dengan While

Perintah WHILE merupakan perintah untuk melakukan perulangan berdasarkan suatu kondisi, jadi banyaknya perulangan tidak bisa ditentukan dengan pasti. Dalam WHILE seakan ada pengecekan kondisi seperti perintah IF untuk melakukan perulangan. Bentuk umum dari perintah WHILE yaitu:

```
while(kondisi) {
    //eksekusi code
}
```

Jika kondisi sesuai, maka perintah atau source code yang ada dalam kurung kurawal “{}” tersebut akan dieksekusi. Untuk lebih memahami tentang perintah WHILE, mari kita modifikasi program di bawah ini dengan penambahan WHILE dan beberapa perubahan lainnya.

```
int i ;

void setup() {
  pinMode(13,OUTPUT);
  pinMode(12,OUTPUT);
  pinMode(11,OUTPUT);
  pinMode(10,OUTPUT);

}

void loop() {
i = 10;
while(i<=13)
{
digitalWrite(i,HIGH);
delay(1000);
digitalWrite(i,LOW);
i++;
}
}
```

2.4. Tugas Rumah

1. Buatlah desain pada tinkercad menggunakan 5 LED dan menyala secara bergantian ke kanan dan ke kiri.
   a. Desain Rangkaian