

2011

Teknik Komputer

kiezane

Modul Praktikkum Mikroprosesor 2



Oleh : Maskie Z.Oematan

[TEKNIK KOMPUTER UNIKOM]

Modul Praktikkum Mikroprosesor 2 Teknik Komputer UNIKOM, Simulasi Software PROTEUS dan BASCOM AVR untuk Mikroprosesor Keluarga AVR (ATMEGA 8535 DAN ATMEGA16)

Modul 1

ATMEGA 16 ,AVR &

pengenalan Proteus Simulator dan IDE BASCOM AVR

A. Pengenalan AVR

Secara histories microcontroller seri AVR pertama kali diperkenalkan ke pasaran sekitar tahun 1997 oleh perusahaan Atmel, yaitu sebuah perusahaan yang sangat terkenal dengan produk microcontroller seri AT89S51/52-nya yang sampai sekarang masih banyak digunakan di lapangan. Tidak seperti microcontroller seri AT89S51/52 yang masih mempertahankan arsitektur dan set instruksi dasar microcontroller 8031 dari perusahaan INTEL. Microcontroller AVR ini diklaim memiliki arsitektur dan set instruksi yang benar-benar baru dan berbeda dengan arsitektur microcontroller sebelumnya yang diproduksi oleh perusahaan tersebut. Tetapi walaupun demikian, bagi para programmer yang sebelumnya telah terbiasa dengan microcontroller seri AT89S51/52, dan berencana untuk beralih ke microcontroller AVR, maka secara teknis tidak akan banyak kesulitan yang berarti, hal ini dikarenakan selain relative tidak jauh berbeda.

Berdasarkan arsitekturnya, AVR merupakan microcontroller RISC (*Reduce Instruction Set Computer*) dengan lebar bus data 8 bit. Berbeda dengan sistem AT89S51/52 yang memiliki frekuensi kerja seperduabelas kali frekuensi oscilator, frekuensi kerja microcontroller AVR ini pada dasarnya sama dengan frekuensi oscilator, sehingga hal tersebut menyebabkan kecepatan kerja AVR untuk frekuensi oscilator yang sama, akan dua belas kali lebih cepat dibandingkan dengan microcontroller keluarga AT89S51/52.

Dengan instruksi yang sangat variatif (mirip dengan sistem CISC-*Complex Instruction Set Computer*) serta jumlah register serbaguna (*general Purpose Register*) sebanyak 32 buah yang semuanya terhubung secara langsung ke ALU (*Arithmetic Logic Unit*), kecepatan operasi microcontroller AVR ini dapat mencapai 16 MIPS (enam belas juta instruksi per detik) —sebuah kecepatan yang sangat tinggi untuk ukuran microcontroller 8 bit yang ada di pasaran sampai saat ini.

Untuk memenuhi kebutuhan dan aplikasi industri yang sangat beragam, microcontroller keluarga AVR ini muncul di pasaran dengan tiga seri utama: tinyAVR, ClasicAVR (AVR), megaAVR.

Berikut ini beberapa seri yang dapat anda jumpai di pasaran:

-ATtiny13 AT90S2313 ATmega103
 -ATtiny22 AT90S2323 ATmega128
 -ATtiny22L AT90S2333 ATmega16
 -ATtiny2313 AT90S4414 ATmega162
 -ATtiny2313V AT90S4433 ATmega168
 -ATtiny26 AT90S8515 ATmega8535

Keseluruhan seri AVR ini pada dasarnya memiliki organisasi memori dan set instruksi yang sama (sehingga dengan demikian jika kita telah mahir menggunakan salah satu seri AVR, untuk beralih ke seri yang lain akan relative mudah). Perbedaan antara tinyAVR, AVR dan megaAVR pada kenyataannya hanya merefleksikan tambahan-tambahan fitur yang ditawarkannya saja (misal adanya tambahan ADC internal pada seri AVR tertentu, jumlah Port I/O serta memori yang berbeda, dan sebagainya). Diantara ketiganya, megaAVR umumnya memiliki fitur yang paling lengkap, disusul oleh AVR, dan terakhir tinyAVR.

| Microcontroller AVR | | Memori (byte) | | |
|---------------------|-----------|---------------|-----------|-----------|
| Jenis | Paket IC | Flash | EEPROM | SRAM |
| TinyAVR | 8–32 pin | 1 – 2K | 64 – 128 | 0 – 128 |
| AVR (classic AVR) | 20–44 pin | 1 – 8K | 128 – 512 | 0 – 1 K |
| MegaAVR | 32–64 pin | 8 – 128 K | 512 – 4 K | 512 – 4 K |

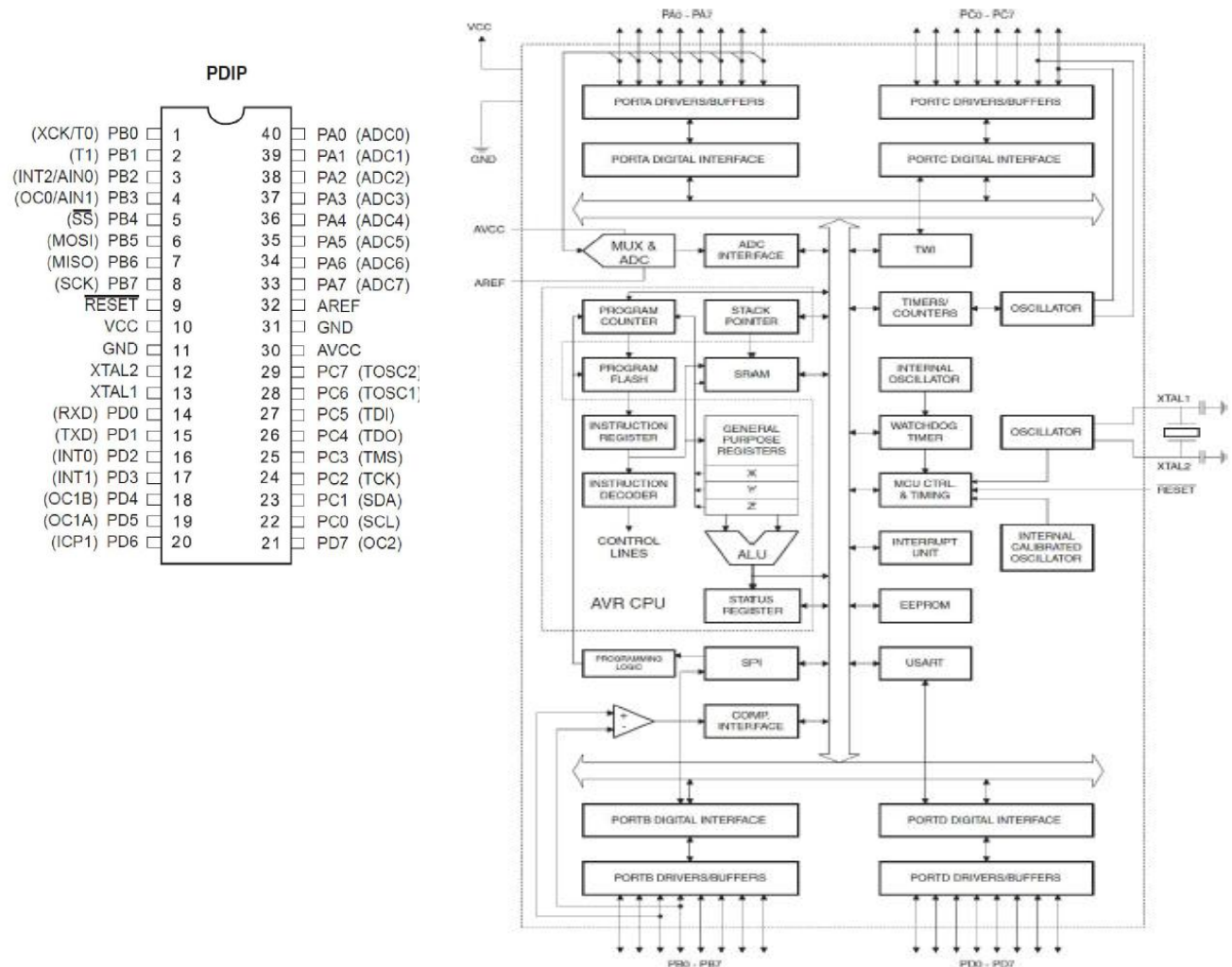
Seperti terlihat pada tabel tersebut, Semua jenis AVR ini telah dilengkapi dengan memori flash sebagai memori program. Tergantung serinya, kapasitas memori flash yang dimiliki bervariasi dari 1K sampai 128 KB. Secara teknis, memori jenis ini dapat diprogram melalui saluran antarmuka yang dikenal dengan nama *Serial Peripheral Interface (SPI)* yang terdapat pada setiap seri AVR tersebut. Dengan menggunakan perangkat lunak *programmer (downloader)* yang tepat, pengisian memori *Flash* dengan menggunakan saluran SPI ini dapat dilakukan bahkan ketika chip AVR telah terpasang pada sistem akhir (*end system*), sehingga dengan demikian pemrogramannya sangat fleksibel dan tidak merepotkan pengguna (Secara praktis metoda ini dikenal dengan istilah *ISP-In System Programming* – sedangkan perangkat lunaknya dinamakan *In System Programmer*).

Untuk penyimpanan data, microcontroller AVR menyediakan dua jenis memori yang berbeda: EEPROM (*Electrically Erasable Programmable Read Only Memory*) dan SRAM (*Static Random Access memory*). EEPROM umumnya digunakan untuk menyimpan data-data program

yang bersifat permanen, sedangkan SRAM digunakan untuk menyimpan data variabel yang dimungkinkan berubah setiap saatnya. Kapasitas simpan data kedua memori ini bervariasi tergantung pada jenis AVR-nya (lihat tabel 1). Untuk seri AVR yang tidak memiliki SRAM, penyimpanan data variabel dapat dilakukan pada register serbaguna yang terdapat pada CPU microcontroller tersebut.

Selain seri-seri diatas yang sifatnya lebih umum, Perusahaan Atmel juga memproduksi beberapa jenis microcontroller AVR untuk tujuan yang lebih khusus dan terbatas, seperti seri AT86RF401 yang khusus digunakan untuk aplikasi *wireless remote control* dengan menggunakan gelombang radio (RF), seri AT90SC yang khusus digunakan untuk peralatan sistem-sistem keamanan kartu SIM GSM, pembayaran via internet, dan lain sebagainya.

Berikut ialah diagram blok AVR ATmega 16



Vcc : Masukkan tegangan catu daya

Port A (A7..A0) : Port a berfungsi sebagai masukkan analog ke ADC internal pada mikrokontroller ATMEGA16, selain itu juga berfungsi sebagai port I/O dwi arah 8 bit, jika ADC tidak digunakan. Masing-masing pin menyediakan resistor pull-up internal yang bias di aktifkan untuk masing-masing bit

Port B (B7..B0) : port B sebagai port I/O dwi aarah 8 bit, masing-masing pin menyediakan resistor pull up internal yang bias diaktifkan untuk masing-masing bit.

Port C (C7..C0) : port C juga digunakan sebagai antarmuka JTAG

Port D (D7..D0) : port D berfungsi sebagai port I/O dwi arah 8 bit. Masing-masing pin menyediakan resistor pull-up internal yang bias diaktifkan untuk masing-masing bit

/Reset : masukkan reset. Level rendah pada pin ini selama lebih dari lama waktu minimum yang ditentukan akan menyebabkan reset, walaupun clock tidak dijalankan

XTAL1 : masukkan ke penguat osilator terbalik(inverting) dan masukan ke rangkaian clock internal

XTAL2 : luaran dari penguat osilator terbalik

AVCC : merupakan masukkan tegangan catu daya untuk port A sebagai ADC, biasanya dihubungkan ke VCC, walaupun ADC-nya tidak digunakan. Jika ADC digunakan sebaiknya dihubungkan ke VCC melalui *low pass filter*

AREF : Merupakan tegangan referensi untuk ADC

B. Basic Compiler (BASCOM) AVR

1. Tipe Data

Berikut ialah table dari tipe data Bascom

| Tipe Data | Ukuran (Byte) | Jangkauan Data |
|-----------|---------------|------------------------------------------------|
| Bit | 1/8 | 0 atau 1 |
| Byte | 1 | 0 s/d 255 |
| Integer | 2 | -32768 s/d 3.767 |
| Word | 2 | 0 s/d 65535 |
| Long | 4 | -2147483648 s/d 2147483647 |
| Single | 4 | $1,5 \times 10^{-45}$ s/d $3,4 \times 10^{38}$ |
| Double | 8 | 5×10^{-324} s/d $1,7 \times 10^{308}$ |
| String | s/d 254 | |

2. Variable

Variable atau pengubah digunakan untuk menyimpan data sementara. Variable diberi nama dan dideklarasikan terlebih dahulu, berikut ialah aturan pemberian nama variable :

- A) harus dimulai dengan huruf (bukan angka)
- B) tidak ada nama variable yang sama pada sebuah program
- C) Maksimum 32 karakter
- D) Tidak menggunakan spasi, pemisah atau garis bawah
- E) tidak menggunakan karakter khusus yang biasa digunakan sebagai operator

Variable dapat dideklarasikan dengan beberapa cara :

1) pernyataan DIM

Singkatan dari *dori dimension* dengan turunan sebagai berikut

DIM <nama variable> As <Tipe Data>

Contoh :

```
Dim bilangan As Integer
```

```
Dim poin As byte
```

2) dengan DEFINT, DEFBIT, DEFBYTE,DEFWORD

Berikut ialah keterangan dari masing-masing pendeklarasian tersebut :

DEFINT = untuk tipe data integer

DEFBIT = untuk tipe data bit

DEFBYTE = untuk tipe data byte

DEFWORD = untuk tipe data word

DEFLNG = untuk tipe data long

DEFSNG = untuk tipe data single

DEFDBL = untuk tipe data double

Cara pendeklarasiannya ialah sebagai berikut :

DEFBIT/DEFINT/DEFBYTE/DEFWORD <variable>

Contoh :

```
DEFINT angka
```

```
DEFBYTE bilangan
```

Bias juga dipisahkan dengan titik koma, missal :

```
DEFINT bil_1; bil_2; bil_3
```

3. Konstanta

Berbeda dengan variable, sebuah konstanta akan bernilai tetap, sebelum digunakan, konstanta terlebih dahulu dideklarasikan dengan 2 cara :

```
Dim nama_Konstanta As nilai_konstanta
```

```
Const nama_konstanta = nilai_konstanta
```

Contoh :

```
Dim A as 66
```

```
Const X = 99
```

4. Penulisan Bilangan

Ada 3 bentuk penulisan bilangan pada BASCOM AVR :

1. Desimal ditulis biasa, contoh : 16
2. Biner diawali dengan &B, contoh : &B1000111
3. Heksadesimal diawali dengan &H, contoh : &H8F

5. Alias

Untuk mempermudah memprogram biasanya nama register dibuat identik dengan hardware, misalnya :

```
LED_1 alias PORTC.0
```

```
SW_1 alias PINC.1
```

6. Array atau Larik

Array atau larik merupakan sekumpulan variable dengan nama dan tipe yang sama yang berbeda indeks keanggotaannya

Cara deklarasi array sebagai berikut :

```
Dim nama_array (jumlah anggota) as tipe_data
```

Contoh :

```
Dim data(8) as byte
```

Artinya variable data dengan tipe data byte memiliki 8 anggota

Adapun untuk mengaksesnya ialah dengan cara :

```
Data(1) = 25
```

PORTC=data (1)

7. Operator

Berikut ialah table beberapa operator BASCOM AVR

Operator Aritmatika

| Operator | Keterangan | Contoh |
|----------|------------------------|--------|
| + | Operasi penjumlahan | A + B |
| - | Operasi pengurangan | A - B |
| * | Operasi perkalian | A * B |
| / | Operasi pembagian | A / B |
| % | Operasi sisa pembagian | A % B |

Operator relasional

| Operator | Keterangan | Contoh |
|----------|------------------------------|--------|
| = | Sama dengan | A = B |
| <> | Tidak sama dengan | A <> B |
| > | Lebih besar dari | A > B |
| < | Lebih kecil dari | A < B |
| >= | Lebih besar atau sama dengan | A >= B |
| <= | Lebih kecil atau sama dengan | A <= B |

Operator Logika

| Operator | Keterangan | Contoh |
|----------|-------------|------------------------------|
| AND | Operasi AND | &B110 And &B101 = &B100 |
| OR | Operasi OR | &B11001 Or &B10111 = &B11111 |
| NOT | Operasi NOT | NOT &HFF = &H0 |
| XOR | Operasi XOR | &B1001 Xor &B0111 = &B1110 |

8. Operasi Bersyarat

a. IF – Then

digunakan untuk memenuhi syarat atau kondisi tertentu, format penggunaannya ialah sebagai berikut :

```
if <kondisi> then <perintah>
```

jika lebih dari satu perintah, maka format penulisannya ialah sebagai berikut

```
if<kondisi> then
```

```
<perintah 1>
<perintah 2>
<perintah n>
Endif
```

b. If Then Else

format ini digunakan untuk 2 kondisi. Formatnya ialah sebagai berikut

```
if <kondisi> then
<perintah 1>
.....
Else
<perintah 2>
.....
Endif
```

c. If Then - ElseIf

jika lebih dari 2 kondisi maka anda dapat menggunakan format di bawah ini

```
if <kondisi 1> then
<perintah 1>
.....
Else if <kondisi2> then
<perintah 2>
.....
Else if <kondisi3> then
<perintah 3>
.....
Endif
```

d. Select – Case

format penggunaannya ialah sbagai berikut

```
select case <variable>
    case 1 : <perintah1>
    case 2 : <perintah2>
end select
```

9. Operasi Perulangan

a. For – Next

format perintah nya ialah sebagai berikut

```
For <Var> = <nilai_awal> to <nilai_akhir> <step_angka>
<perintah>
Next <var>
```

b.Do – Loop

```
Do
<perintah>
Loop
```

c. Do – Loop Until

```
Do
<perintah>
Loop until <kondisi>
```

d. While – Wend

```
While <kondisi>
<perintah>
Wend
```

10. Lompatan Proses

a. Gosub <nama_subrutin>

contoh :

```
print "Teknik Komputer"
```

```
gosub next
```

```
print "UNIKOM"
```

```
End
```

```
next:
```

```
print"Bandung 2011"
```

```
return
```

b. Goto <Label>

contoh :

```
Dim A as Byte
```

```
Mulai :
```

```
A = A +1
```

```
If A < 10 Then
```

```
Goto Mulai
```

```
Endif
```

```
Print"Sukses"
```

11. Exit

Untuk keluar langsung dari perulangan Do-Loop, For-Next, While-Wend. Cara penulisannya ialah sebagai berikut :

```
Exit For
```

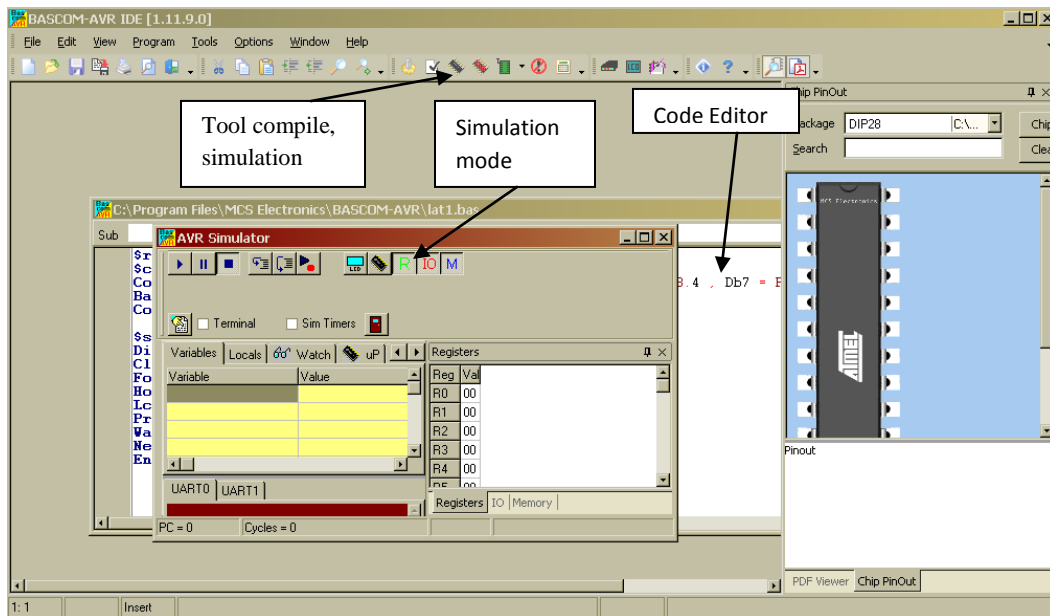
```
Exit Do
```

```
Exit While
```

```
Exit Sub           Exit Function
```

C. Jendela IDE Bascom AVR

Berikut ialah tampilan dari jendela IDE Bascom AVR



D. Latihan Program dasar

(1) Loop

```
$regfile = "m16def.dat"
$crystal = 8000000
$baud = 19200
$hwstack = 32
$swstack = 10
$framesize = 40

Dim A As Byte , B1 As Byte , C As Integer
For A = 1 To 10 Step 2
    Print " A ke " ; A
Next A

Print "sekarang counter turun"
For C = 10 To -5 Step -1
```

```

    Print "C ke " ; C
Next
Print "ini menggunakan FOR..NEXT statements."
For A = 1 To 10
    Print " A ke " ; A
    For B1 = 1 To 10
        Print "B1 ke" ; B1
    Next
Next A
A = 0
Do
    Print A
    Incr A
Loop Until A = 10
End

```

(2). Configure Port

```

$regfile "m8def.dat"
Dim A As Byte , Count As Byte
Config Portd = Input
$sim
A = Pind
A = A And Portd
Print A
Bitwait Pind.7 , Reset
Config Portb = Output
Portb = 10
Portb = Portb And 2
Set Portb.0
Incr Portb

```

```

Count = 0
Do
    Incr Count
    Portb = 1
    For A = 1 To 8
        Rotate Portb , Left
        Wait 1
    Next
Portb = 1
Rotate Portb , Left , 8
Loop Until Count = 10
Print "Ready"
End

```

1) Amati dan pelajari mekanisme sintaks untuk pengaksesan PORT di Bascom AVR. Buat kesimpulan anda?

2) jelaskan apa yang terjadi pada program , lakukan step per line program?

(3). Boolean Instruction

```

$regfile = "m16def.dat"
$crystal = 8000000
$baud = 19200
$hwstack = 32
$swstack = 10
$framesize = 40
Dim A As Byte , B1 As Byte , C As Byte
Dim Aa As Bit , I As Integer
A = 5 : B1 = 3
C = A And B1
Print "a AND c = " ; C
C = A Or B1
Print "a OR b1 = " ; C

```

```

C = A Xor B1
and for xor
Print "a XOR b1 = " ; C
A = 1
C = Not A
Print "c = NOT a " ; C
C = C Mod 10
Print "c MOD 10 = " ; C
If Portb.1 = 1 Then
    Print "Bit set"
Else
    Print "Bit not set"
End If
Aa = 1
Set Aa
If Aa = 1 Then
    Print "set Bit (aa=1)"
Else
    Print "Bit not set(aa=0)"
End If
Aa = 0
Reset Aa
If Aa = 1 Then
    Print "Bit set (aa=1)"
Else
    Print "Bit not set(aa=0)"
End If
B1 = 255
Reset B1.0
Print B1
Set B1.0
Print B1

```

Toggle B1.0

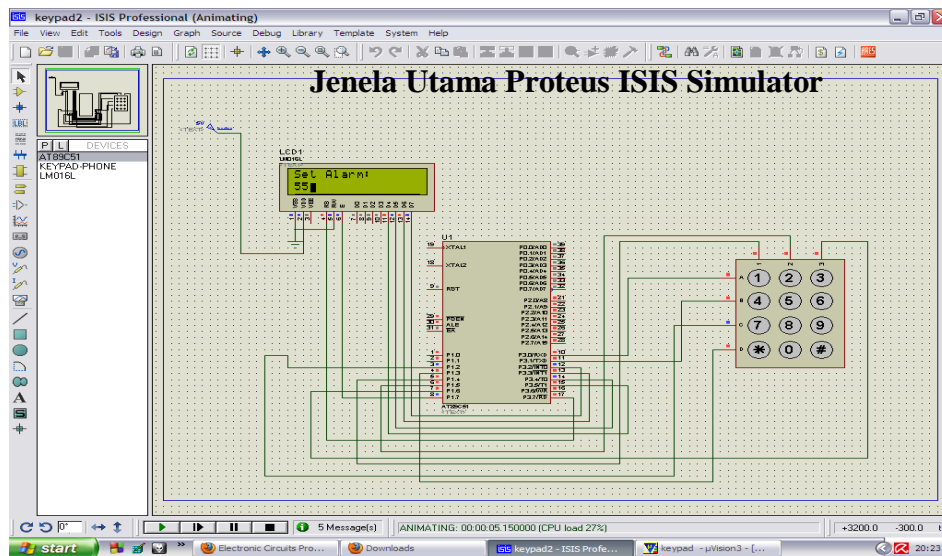
Print B1

End

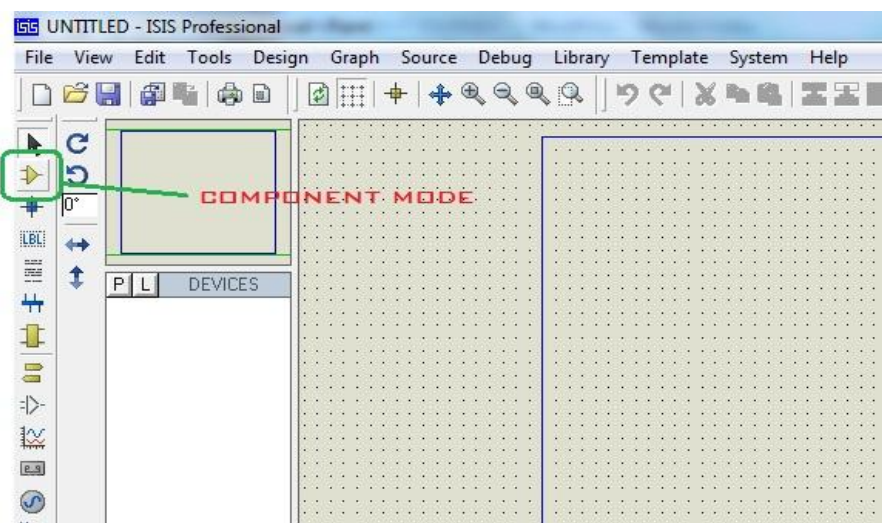
1) Amati dan pelajari mekanisme sintaks untuk pengaksesan instruksi Boolean dan seleksi kondisi di Bascom AVR. Buat kesimpulan anda?

2) jelaskan apa yang terjadi pada program , lakukan step per line program?

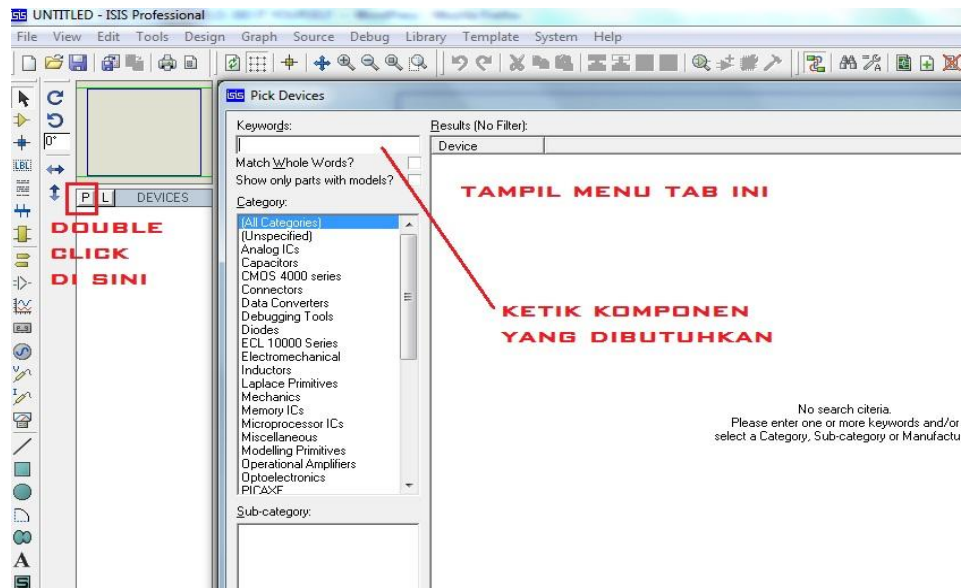
PENGENALAN PROTEUS ISIS SIMULATOR



Adapun beberapa bagian dari simulatornya ialah sebagai berikut :

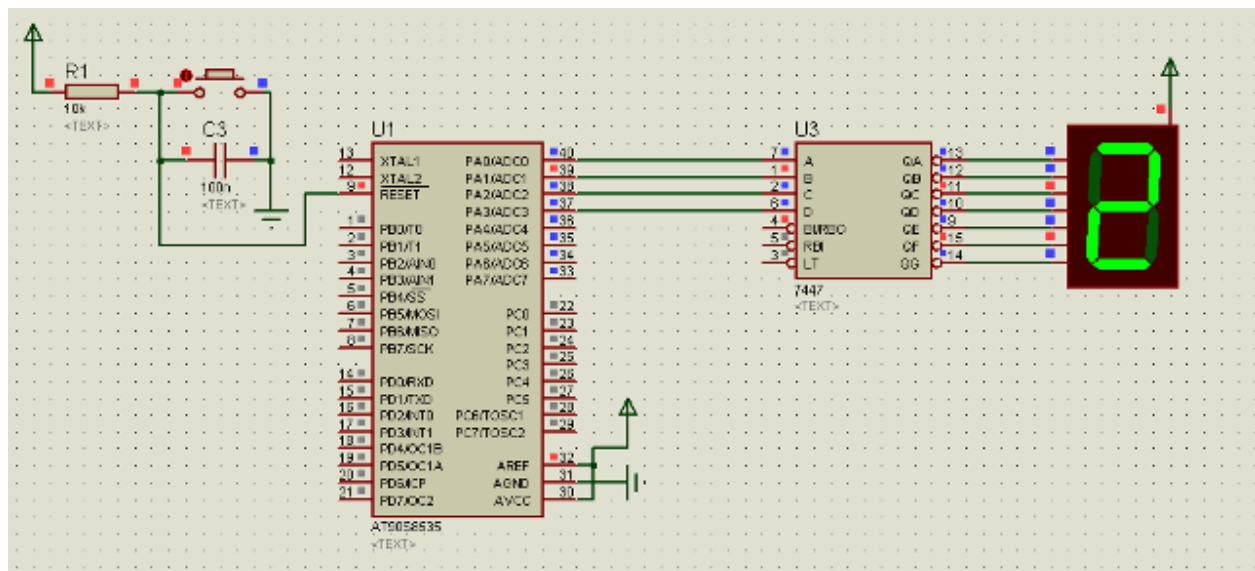


Untuk memilih komponen yang kita perlukan pada rancangan schematic nya ialah sebagai berikut



Latihan Proteus Simulator (Merancang aplikasi (termasuk system minimum) 7 Segment pada AVR ATMEGA 8535

Buatlah schematic di proteus sebagai berikut :



Keterangan komponen :

R1 10k, C3 100nF, IC decoder 7447, seven segment, U1 AT908535

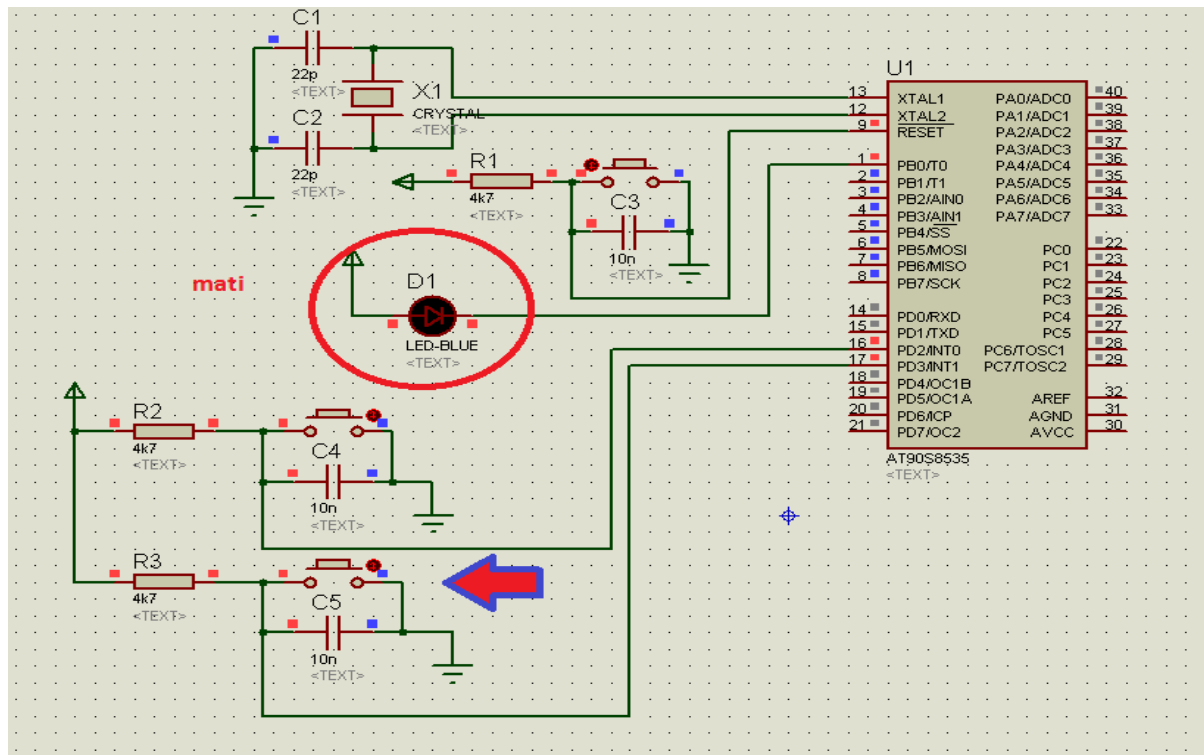
Buatlah listing program sebagai berikut :

```

$crystal = 10000000
$regfile = "m8535.dat"
Dim B As Integer
Config Porta = Output
Do
B = 0
Do
Porta = B
Waitms 60
Incr B
Loop Until B = 10
Loop.

```

Aplikasi LED sederhana menggunakan PROTEUS dan AVR 8535



```

$crystal = 12000000
$regfile = "m8535.dat"
Dim Data1 As Byte
Dim Data2 As Byte
Config Portb = Output
Config Portd = Input
Do
Select Case Pind
Case 4 :

```

```
Set Portb.0          `portb.0 bernilai 1 , akibatnya led mati
Case 8 :
Reset Portb.0        `portb.0 bernilai 0, akibatnya led mati
End Select
Loop
End
```

Pertanyaan dan Tugas

Ubahlah script program diatas, dengan ketentuan sebagai berikut :

Jika **PD.3 dan PD.2 ON** maka lampu akan **ON**, sebaliknya jika **PD.3 atau PD.2 OFF** maka lampu akan **OFF**