CONTOH-CONTOH PROGRAM MIKROKONTROLER

Yoyo Somantri dan Erik Haritman
Dosen Jurusan Pendidikan Teknik Elektro
FPTK Universitas Pendidikan Indonesia

Pendahuluan

Dalam bab ini akan dibahas tujuan perkuliahan, contoh-contoh program aplikasi pada mikrokontroler : lampu led berjalan, lampu Flip-flop, key pad, lampu berputar ke kiri-ke kanan, penghitung, dan stepper motor.

Tujuan Perkuliahan

Setelah mempelajari bab ini, diharapkan mahasiswa mampu untuk :

1. Membuat programLed berjalan.
2. Membuat lampu Flip-Flop.
3. Membuat program key pad.
4. Membuat program penghitung (Counter)
5. Membuat program stepper motor

1. Contoh-contoh program :

Aplikasi pada lampu LED

1. Program Lampu Flip Flop pada Port 0

```
; Program Lampu Flip Flop pada Port 0

        $mod51
mulai:      mov p0,#0ffh
        call delay
        mov p0,#0
        call delay
        jmp mulai
```

```
; Sub rutin Delay
delay:      mov r0,#0
delay1:     mov r1,#0
            djnz r1,$
        djnz r0,delay1
        ret
        end
```

2. Program Lampu Flip Flop pada Port 0

```
; Program Lampu Flip Flop pada Port 0
        $mod51
mulai:      mov p0,#00fh
        call delay
        mov p0,#0f0h
        call delay
        jmp mulai
; Sub rutin Delay
delay:      mov r0,#0
delay1:     mov r1,#0
            djnz r1,$
        djnz r0,delay1
        ret
        end
```

3. Program Lampu berjalan pada Port 0

```
; Program Lampu berjalan pada Port 0
```

```
        $mod51
mulai:      mov p0,#11111110b
        call delay
        mov p0,#11111101b
        call delay
            mov p0,#11111011b
        call delay
        mov p0,#11110111b
            call delay
            mov p0,#11101111b
        call delay
        mov p0,#11011111b
            call delay
            mov p0,#10111111b
        call delay
        mov p0,#01111111b
            call delay
        jmp mulai
; Sub rutin Delay
delay:      mov r0,#0
delay1:     mov r1,#0
            djnz r1,$
        djnz r0,delay1
        ret
        end
```

3. Program Lampu Flip Flop pada Port 0

```
; Program Lampu Flip Flop pada Port 0
        $mod51
mulai:
                mov a,#11111111b
mulai1:
                rrc     a
                mov p0,a
        call delay
                jmp mulai1
; Sub rutin Delay
delay: mov r0,#0
delay1:         mov r1,#0
                djnz r1,$
        djnz r0,delay1
        ret
        end
```

4. Program Lampu Flip Flop pada Port 0

```
; Program Lampu Flip Flop pada Port 0
        $mod51
mulai:
                mov a,#11111111b
mulai1:
                rlc     a
                mov p0,a
        call delay
        jmp mulai1
```

```
; Sub rutin Delay
delay:      mov r0,#0
delay1:     mov r1,#0
            djnz r1,$
        djnz r0,delay1
        ret
        end
```

Aplikasi pada 7 Segment

1. Program –1

```
        $mod51
        org     0h
main:
        mov     p2,#11000000b       ;0
        clr     p1.4
        call    delay
        mov     p2,#11110011b       ;1
        call    delay
        mov     p2,#10001001b       ;2
        call    delay
        mov     p2,#10100001b       ;3
        call    delay
        mov     p2,#10110010b       ;4
        call    delay
        mov     p2,#10100100b       ;5
        call    delay
        mov     p2,#10000100b       ;6
```

```
        call    delay
        mov     p2,#11110001b       ;7
        call    delay
        mov     p2,#10000000b       ;8
        call    delay
        mov     p2,#10100000b       ;9
        call    delay
        jmp     main
delay:
        mov     r7,#100
delay_loop1:
        mov     r6,#100
delay_loop2:
        mov     r5,#100
        djnz    r5,$
        djnz    r6,delay_loop2
        djnz    r7,delay_loop1
        ret
end
```

2. Program -2

```
        $mod51
        org             0h
main:
        mov     p2,#11000000b       ;0
        setb    p1.4
```

```
        call    delay
        mov     p2,#11110011b       ;1
        call    delay
        mov     p2,#10001001b       ;2
        call    delay
        mov     p2,#10100001b       ;3
        call    delay
        mov     p2,#10110010b       ;4
        call    delay
        mov     p2,#10100100b       ;5
        call    delay
        mov     p2,#10000100b       ;6
        call    delay
        mov     p2,#11110001b       ;7
        call    delay
        mov     p2,#10000000b       ;8
        call    delay
        mov     p2,#10100000b       ;9
        call    delay
        jmp     main
delay:
        mov     r7,#100
delay_loop1:
        mov     r6,#100
delay_loop2:
        mov     r5,#100
        djnz    r5,$
```

```
        djnz    r6,delay_loop2
        djnz    r7,delay_loop1
        ret
end
```

3. Program -3

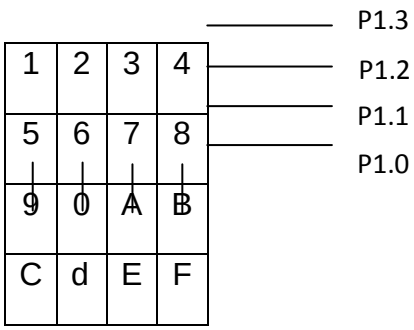
```
        $mod51
        Counter_Lowequ          30h
        Counter_High    equ     31h
        Scanning    equ         32h
        org     0
main:
        mov     Counter_Low,#0
        mov     Counter_High,#0
        mov     Scanning,#100
main_loop:
        clr     a
        mov     p2,a
        clr     P1.4
        mov     a,Counter_Low
        call    Tabel_Data
        mov     p2,a
        call    delay
        clr     a
        mov     p2,a
        setb    P1.4
        mov     a,Counter_High
```

```
        call    Tabel_Data
        mov     p2,a
        call    delay
        djnz    Scanning,Main_Loop
        mov     Scanning,#100
        inc     Counter_Low
        mov     a,Counter_Low
        cjne    a,#10,main_loop
        mov     Counter_Low,#0
        inc     Counter_High
        mov     a,Counter_High
        cjne    a,#10,main_loop
        mov     Counter_High,#0
        jmp     main_loop
Tabel_Data:
        cjne    a,#0,TabelData_1
        mov     a,#11000000b            ;0
        ret
TabelData_1:
        cjne    a,#1,TabelData_2
        mov     a,#11110011b            ;1
        ret
TabelData_2:
        cjne    a,#2,TabelData_3
        mov     a,#10001001b            ;2
        ret
TabelData_3:
```

```
        cjne    a,#3,TabelData_4
        mov     a,#10100001b            ;3
        ret
TabelData_4:
        cjne    a,#4,TabelData_5
        mov     a,#10110010b            ;4
        ret
TabelData_5:
        cjne    a,#5,TabelData_6
        mov     a,#10100100b            ;5
        ret
TabelData_6:
        cjne    a,#6,TabelData_7
        mov     a,#10000100b            ;6
        ret
TabelData_7:
        cjne    a,#7,TabelData_8
        mov     a,#11110001b            ;7
        ret
TabelData_8:
        cjne    a,#8,TabelData_9
        mov     a,#10000000b            ;8
        ret
TabelData_9:
        cjne    a,#9,TabelData_Out
        mov     a,#10100000b            ;9
TabelData_Out:
```

```
        ret
delay:
        mov    r7,#1
delay_loop1:
        mov    r6,#10
delay_loop2:
        mov    r5,#100
        djnz   r5,$
        djnz   r6,delay_loop2
        djnz   r7,delay_loop1
        ret
end
```

Aplikasi pada KEYPAD



P3.7 P3.6 P3.5 P3.4

1. Program -1

```
        $mod51
        org     0
        clr     p1.4
mulai: mov     P1,#11110111b
        jb      p3.7,key_1
```

```
        mov   p2,#11110011b;1
        sjmp  mulai
key_1:
        jb    p3.6,key_2
        mov   p2,#10001001b;2
        sjmp  mulai
key_2:
        jb    p3.5,key_3
        mov   p2,#10100001b;3
        sjmp  mulai
key_3:
        jb    p3.4,key_4
        mov   p2,#10110010b;4
        sjmp  mulai
key_4:
        mov   p1,#11111011b
        jb    p3.7,key_5
        mov   p2,#10100100b;5
key_5:
        jb    p3.6,key_6
        mov   p2,#10000100b;6
        sjmp  mulai
key_6:
        jb    p3.5,key_7
        mov   p2,#11110001b;7
        sjmp  mulai
key_7:
```

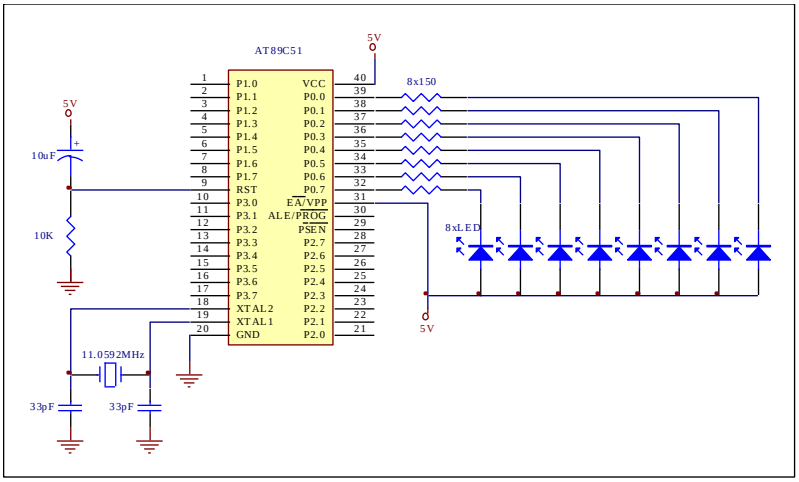
```
        jb      p3.4,key_8
        mov     p2,#10000000b;8
        sjmp    mulai
key_8:
        mov     p1,#11111101b
        jb      p3.7,key_9
        mov     p2,#10100000b;9
key_9:
        jb      p3.6,key_10
        mov     p2,#11000000b;0
        ljmp    mulai
key_10:
        jb      p3.5,key_11
        mov     p2,#10010000b       ;A
        ljmp    mulai
key_11:
        jb      p3.4,key_12
        mov     p2,#10000110b       ;b
key_12:
        mov     p1,#11111110b
        jb      p3.7,key_13
        mov     p2,#11001100b;C
        sjmp    mulai
key_13:
        jb      p3.6,key_14
        mov     p2,#10000011b;d
key_14:
```

```
        jb      p3.5,key_15
        mov     p2,#10001100b;E
        ljmp    mulai
key_15:
        jb      p3.4,key_16
        mov     p2,#10011100b;0
        ljmp    mulai

key_16:
        ljmp    mulai
end
```

Contoh program dan rangkaiannya

1. Merancang aplikasi lampu berjalan dari kiri kekanan dan dari kanan ke kiri lengkap dengan program dan perhitungan delaynya.



Gambar 1. Rangkaian Untuk Menyalakan Led 8 buah.

Bahasa Pemograman

a) Lampu berjalan yang bergerak satu per satu dari kiri kekanan secara berulang-ulang.

```
    ORG     0H          ; Awal program dimulai pada alamat 0H
Mulai:
    MOV     A,#07FH     ; Isi Akumululator dengan data 7FH
Putar:
     MOV    P0,A      ; Salin data dari Akumuklator ke P0 (nyalakan 1     lampu)
    RR      A           ; Putar 1 bit data pada Akumulator ke arah kanan
    ACALL   Tunda       ; Panggil subrutin tunda untuk waktu Tunda     penyalaan
    SJMP    Putar               ; Lompat ke label mulai (lakukan secara berulang)
; Led akan nyala jika diberi logik 0
Tunda:
    MOV     R5,# 250            ; Isi Register 5 dengan data 250
Tunda1:
    MOV     R6,#100             ; Isi Register 6 dengan data 100
Tunda2:
    MOV     R7,#10      ; Isi Register 7 dengan data 10
    DJNZ    R7,$                ; Kurangi R7 dengan 1 sampai 0 *
    DJNZ    R6,Tunda2           ; Kurangi R6 jika belum 0 lompat ke label tunda2 **
    DJNZ    R5,Tunda1           ; Kurangi R7 jika belum 0 lompat ke label tunda1 ***
    RET                         ; Kembali ke program utama
End
; Waktu tunda ( siklus DJNZ = 2 µS) :
; *     : 2 (R7) = 2 (10)                     =         20 µS
```

; ** : 2 (R7 x R6) = 2 (10 x 100) = 2000 µS

; *** : 2 (R7 x R6 x R5) = 2 (10 x 100 x 250) = 500000 µS

; ---------- +

; 502020 µS ≈ 500 mS

b) Bahasa pemogramannya jika lampu ingin bergerak dari kanan ke kiri yaitu

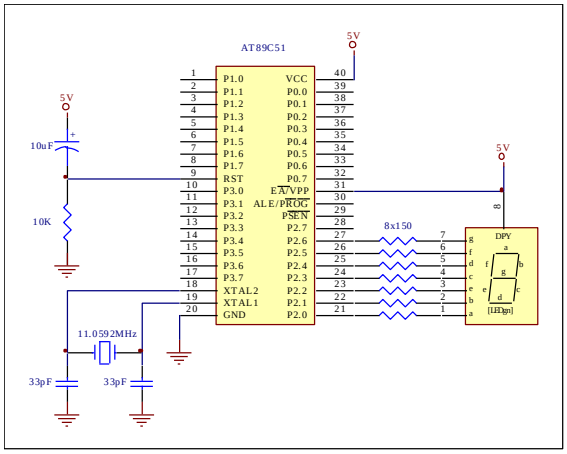
```
    Mulai:
        MOV     A,#0FEH         ; Intruksi yang diganti
    Putar:
        MOV     P0,A
        RL      A               ; Intruksi yang diganti
        ACALL   Tunda
        SJMP    Putar
; Led akan nyala jika diberi logik 0
```

c) Bahasa program untuk menggerakan lampu dari tengah ke pinggir dengan waktu tunda 200 mili detik yaitu

```
        ORG     0H              ; Awal program dimulai pada alamat 0H
    Mulai:
        MOV     P0,#11100111B   ; Isi port 0 dengan data 11100111B
        ACALL   Tunda           ; Panggil subrutin Tunda
        MOV     P0,#11011011B   ; Isi port 0 dengan data 11011011B
        ACALL   Tunda           ; Panggil subrutin Tunda
```

```
        MOV     P0,#10111101B  ; Isi port 0 dengan data 10111101B
        ACALL  Tunda           ; Panggil subrutin Tunda
        MOV     P0,#01111110B  ; Isi port 0 dengan data 01111110B
        ACALL  Tunda           ; Panggil subrutin Tunda
        SJMP    Mulai          ; Lompat ke label mulai (lakukan berulang)
; Led akan nyala jika diberi logik 0
    Tunda:
        MOV     R5,# 250       ; Isi Register 5 dengan data 250
    Tunda1:
        MOV     R6,#40         ; Isi Register 6 dengan data 40
    Tunda2:
        MOV   R7,#10           ; Isi Register 7 dengan data 10
        DJNZ   R7,$            ; Kurangi R7 dengan 1 sampai 0 *
        DJNZ   R6,Tunda2       ; Kurangi R6 jika belum 0 lompat ke label tunda2 **
        DJNZ   R5,Tunda1       ; Kurangi R7 jika belum 0 lompat ke label tunda1 ***
        RET                    ; Kembali ke program utama
    End
    ; Waktu tunda ( siklus DJNZ = 2 µS) :
    ; *     : 2 (R7) = 2 (10)                          =      20 µS
    ; **    : 2 (R7 x R6) = 2 (10 x 40)          =      800 µS
    ; *** : 2 (R7 x R6 x R5) = 2 (10 x 100 x 250) =  200000 µS
    ;                                                  ---------- +
    ;                                                  200820 µS ≈ 200 mS
```

2. **Merancang dan membuat rangkaian counter dengan menggunakan mikrokontroler AT89C51.**
    a. Rangkailah *up counter* dengan menggunakan *seven segment* dan IC mikrokontroler AT89C51 yang bekerja pada *port* 2 seperti gambar dibawah.
    b. Buatlah bahasa pemrograman untuk *up counter* yang bergerak naik dari angka 0 ampai 9 secara berulang-ulang dengan waktu tunda 500 mili detik.
    c. Tulis bahasa pemograman yang telah dibuat dikomputer dengan menggunakan *software* Pinnacle. Kemudian simulasikan program tersebut secara *software* sampai tidak ada kesalahan.
    d. Setelah program disimulasikan dan tidak ada kesalahan kemudian isikan program tersebut kedalam IC mikrokontroler dengan menggunakan progrmmer Atmel.
    e. Selanjutnya pindakan IC yang telah diprogram ke Emulator untuk disimulasikan secara hardware.



Gambar 2. Rangkaian Counter dengan display seven segment

.

Bahasa Pemograman

```
ORG      0H                  ; Awal program dimulai pada alamat 0H
Mulai:
 MOV    P2,#11000000B        ; Isi port 2 dengan data 10110000B (angka 0)
ACALL  Tunda                ; Panggil subrutin Tunda
MOV     P2,#11111001B  ; Isi port 2 dengan data 11110011B (angka 1)
ACALL  Tunda                ; Panggil subrutin Tunda
MOV     P2,#10100100B  ; Isi port 2 dengan data 10100100B (angka 2)
ACALL  Tunda                ; Panggil subrutin Tunda
MOV     P2,#10110000B        ; Isi port 2 dengan data 10110000B (angka 3)
ACALL  Tunda                ; Panggil subrutin Tunda
MOV     P2,#10011001B        ; Isi port 2 dengan data 10011001B (angka 4)
ACALL  Tunda                ; Panggil subrutin Tunda
MOV     P2,#10010010B        ; Isi port 2 dengan data 10010010B (angka 5)
ACALL  Tunda                ; Panggil subrutin Tunda
MOV     P2,#10000010B        ; Isi port 2 dengan data 10000010B (angka 6)
ACALL  Tunda                ; Panggil subrutin Tunda
MOV     P2,#11111000B        ; Isi port 2 dengan data 11111000B (angka 7)
ACALL  Tunda                ; Panggil subrutin tunda
MOV     P2,#10000000B        ; Isi port 2 dengan data 10000000B (angka 8)
ACALL  Tunda                ; Panggil subrutin Tunda
MOV     P2,#10010000B        ; Isi port 2 dengan data 10010000B (angka 9)
ACALL  Tunda                ; Panggil subrutin Tunda
SJMP      Mulai             ; Lompat ke label mulai (lakukan berulang)
```

```
; Seven segmen akan menyala jika diberi logik 0
Tunda:
    MOV     R5,# 250            ; Isi Register 5 dengan data 250
Tunda1:
    MOV     R6,#100             ; Isi Register 6 dengan data 100
Tunda2:
    MOV     R7,#10      ; Isi Register 7 dengan data 10
    DJNZ    R7,$                ; Kurangi R7 dengan 1 sampai 0 *
    DJNZ    R6,Tunda2           ; Kurangi R6 jika belum 0 lompat ke label tunda2 **
    DJNZ    R5,Tunda1           ; Kurangi R7 jika belum 0 lompat ke label tunda1 ***
    RET                         ; Kembali ke program utama
End

; Waktu tunda ( siklus DJNZ = 2 µS) :
; *     : 2 (R7) = 2 (10)                       =        20 µS
; **    : 2 (R7 x R6) = 2 (10 x 100)            =      2000 µS
; *** : 2 (R7 x R6 x R5) = 2 (10 x 100 x 250) =  500000 µS
;                                                ---------- +
;                                                502020 µS ≈ 500 mS
```

3. Merancang dan membuat rangkaian penggerak motor *stepper* dengan menggunakan Mikrokontroler

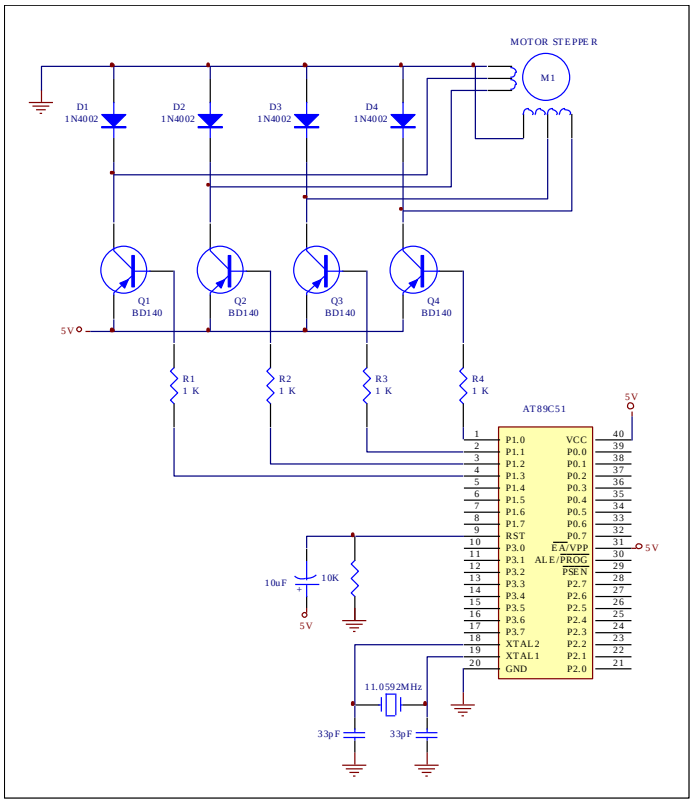
a. Rangkaian penggerak motor *stepper* dengan menggunakan IC mikrokontroler AT89C51 yang bekerja pada *port* 1. Ikuti standar rangkaian seperti gambar dibawah.

b. Buatlah bahasa pemograman untuk penggerak motor *stepper* yang bergerak searah dengan jarum jam secara berulang-ulang dengan waktu tunda 500 mili detik.

c. Tulis bahasa pemograman yang telah dibuat dikomputer dengan menggunakan *software* Pinnacle. Kemudian simulasikan program tersebut secara *software* sampai tidak ada kesalahan.

d. Setelah progaram disimulasikan dan tidak ada kesalahan kemudian isikan program tersebut kedalam IC mikrokontroler dengan menggunakan *programmer* Atmel.

e. Selanjutnya pindahkan IC yang telah diprogram ke *Emulator* untuk disimulasikan secara *hardware*.



Gambar 3. Rangkaian penggerak *Stepper Motor*

Bahasa Pemograman

```
    ORG     0H
Mulai:
    MOV     P1,#11111110B   ; Isi port 1 dengan data 11111110B
    ACALL   Tunda           ; Panggil subrutin Tunda
    MOV     P1,#11111101B   ; Isi port 1 dengan data 11111101B
    ACALL   Tunda           ; Panggil subrutin Tunda
    MOV     P1,#11111011B   ; Isi port 1 dengan data 11111011B
    ACALL   Tunda           ; Panggil subrutin Tunda
    MOV     P1,#11110111B   ; Isi port 1 dengan data 11110111B
    ACALL   Tunda           ; Panggil subrutin Tunda
    SJMP    Mulai           ; Lompat ke label mulai (lakukan berulang)
; Motor akan berputar jika diberi logik 0.
Tunda:
    MOV     R5,# 250        ; Isi Register 5 dengan data 250
Tunda1:
    MOV     R6,#100         ; Isi Register 6 dengan data 100
Tunda2:
    MOV     R7,#10      ; Isi Register 7 dengan data 10
    DJNZ    R7,$            ; Kurangi R7 dengan 1 sampai 0 *
    DJNZ    R6,Tunda2       ; Kurangi R6 jika belum 0 lompat ke label tunda2 **
    DJNZ    R5,Tunda1       ; Kurangi R7 jika belum 0 lompat ke label tunda1 ***
    RET                     ; Kembali ke program utama
End
```

; Waktu tunda (siklus DJNZ = 2 µS) :

; * : 2 (R7) = 2 (10) = 20 µS

; ** : 2 (R7 x R6) = 2 (10 x 100) = 2000 µS

; *** : 2 (R7 x R6 x R5) = 2 (10 x 100 x 250) = 500000 µS

; ---------- +

; 502020 µS ≈ 500 mS

1. Potongan program yang harus diganti agar motor bergerak berlawanan arah dengan jarum jam yaitu:

```
Mulai:
    MOV   P1,#11110111B ; Isi port 1 dengan data 11110111B
    ACALL  Tunda        ; Panggil subrutin Tunda
    MOV   P1,#11111011B ; Isi port 1 dengan data 11111011B
    ACALL  Tunda        ; Panggil subrutin Tunda
    MOV   P1,#11111101B ; Isi port 1 dengan data 11111101B
    ACALL  Tunda        ; Panggil subrutin Tunda
    MOV   P1,#11111110B ; Isi port 1 dengan data 11111110B
    ACALL  Tunda        ; Panggil subrutin Tunda
    SJMP   Mulai        ; Lompat ke label mulai (lakukan berulang)
```

; Motor akan berputar jika diberi logik 0.

2. Program untuk menggerakan motor *stepper* searah dengan jarum jam sejauh 180º dengan waktu tunda 200 mili detik yaitu :

```
    ORG     0H
Mulai:
    MOV     R0,#24          ; Isi R0 dengan data 24 ( 180o putaran)
Putar:
    MOV     P1,#11110111B ; Isi port 1 dengan data 11110111B
    ACALL  Tunda            ; Panggil subrutin Tunda
    MOV     P1,#11111011B ; Isi port 1 dengan data 11111011B
    ACALL  Tunda            ; Panggil subrutin Tunda
    MOV     P1,#11111101B ; Isi port 1 dengan data 11111101B
    ACALL  Tunda            ; Panggil subrutin Tunda
    MOV     P1,#11111110B ; Isi port 1 dengan data 11111110B
    ACALL  Tunda            ; Panggil subrutin Tunda
    DJNZ    R0,Putar        ; Kurangi R0 dengan 1 jika belum 0 lompat ke Putar
    SJMP    $               ; Lompat kedirinya sendiri (program berhenti)

; 1 putaran motor stepper terdapat 192 step, untuk menjalankan 180o terdapat 96
; step. Dalam menjalankan motor stepper dilakukan setiap 4 step sekali. Sehingga
; untuk memutar 180o terdapat 96 : 4 = 24 kali.

; Motor akan berputar jika diberi logik 0.

Tunda:
    MOV     R5,#250       ; Isi Register 5 dengan data 250
Tunda1:
    MOV     R6,#40        ; Isi Register 6 dengan data 40
Tunda2:
    MOV     R7,#10        ; Isi Register 7 dengan data 10
```

```
    DJNZ    R7,$            ; Kurangi R7 dengan 1 sampai 0
    DJNZ    R6,Tunda2       ; Kurangi R6 jika belum 0 lompat ke label tunda2
    DJNZ    R5,Tunda1       ; Kurangi R7 jika belum 0 lompat ke label tunda1
    RET                     ; Kembali ke program utama
End

; Waktu tunda ( siklus DJNZ = 2 µS) :
; *     : 2 (R7) = 2 (10)                        =        20 µS
; **    : 2 (R7 x R6) = 2 (10 x 40)                       =       800 µS
; *** : 2 (R7 x R6 x R5) = 2 (10 x 40 x 250)   =  200000 µS
;                                                  ---------- +
;                                                  200820 µS ≈ 200 mS
```

2. Soal Latihan

1. Jelaskan organisasi memori pada mikrokontroler 89C51!
2. Jelaskan perbedaan antara *direct addressing* dengan *indirect addressing* dan berikan contohnya !
3. Jelaskan 6 mode penglaman pada mikrokontroler 89C51 serta dilengkapi dengan contoh setiap modenya!
4. Bila *Program Status Word* berisi 18 h.yang terdapat pada RAM Internal 89C51 bank register berapa yang terpilih!
5. Dari alamat berapakah *Special Function Register* dapat dialamati dan pada alamat berapa!
6. Jelaskan fungsi dari pin PSEN dan AE pada mikrokontroler 89C51!
7. Buat program untuk mengurangkan bilangan yang berada pada lokasi memori 2000 dengan bilangan pada lokasi memori 2001!

8. Jelaskan yang dimaksud *Idle Mode* dan *power down mode* pada mikrokontroler 89C51!
9. Buat program untuk menjumlahkan bilangan pada R_0 = 137, R_1 = 33, dan R_3 = 56 !
10. Buat program untuk membagi bilangan pada R_0 : R_1 : R_2. (R_0 = 100; R_1 = 2 ; dan R_2 = 10)!
11. Jelaskan program dibawah ini :

```
SMOD 51
MAIN        CPL  P1,0
              CALL DELAY
              JMP MAIN
DELAY       MOV R7, # 50
DE LAY1     MOV R 6, # 100
DELAY 2     MOV R 5, # 100.
              DJNZ  R5, $
              DJNZ R 6, DELAY 2
              DJNZ R 7, DELAY 1
               RET
END.
```

13. Jelaskan program dibawah ini

```
        ORG     0H
Mulai:
    MOV     P0,#11100111B
    ACALL  Tunda
    MOV     P0,#11011011B
    ACALL  Tunda
    MOV     P0,#10111101B
    ACALL  Tunda
    MOV     P0,#01111110B
    ACALL  Tunda
SJMP     Mulai.
  RET
END.
```

Referensi :

1. Sencer, (1997). *Programming Interfacing 8051 Microcontroller*. Mc Graw Hill.
2. Intel, (1994). *MCS'51 Microcontroller Family User Manual.*
3. Myke Predko, (1995). *Programming and Custumizing The 8051 Microcontroller.* Mc Graw Hill.
4. Allen l Wyatt, (1995). *Using Assembly Language*. Que
5. Atmel, (2005**).** *Data Book Microcontroller.*
6. Ramakart Gayakwad, Leonard Sokolof. (1988). *Analog and Digital Control Systems.* Canada : Prentice- HallInternational, Inc.
7. Greenfield, Joseph D.(1992). *The 68HC11 Microcontroller*. Orlando, FL:
8. Peter Spasov, (2002). *Microcontroller Technology: The 68HC11*, Prentice-Hall. ISBN: 0-13-019579.
9. Toto Budiono (2005). Pemograman Bahasa C dgn SDCC. Gaya Media.
10. 89C51 Development Tools DT51 Version 3. User"S Guide. Manual Book.
11. Agus Bejo, (2008). *C & AVR Rahasia Kemudahan Bahasa C dalam mikrokontroler ATMega 8535*. Graha Ilmu. Yogyakarta.
12. Totok Budioko, (2005). *Belajar dengan mudah dan cepat Pemograman Bahasa C dengan SDCC (Small Device C Compiler).*Penerbit Gaya Media. Yogyakarta.
13. Agfianto Eko Putra, *(2004). Belajar Mikrokontroler AT 89C51/52/55.* Penerbit Gaya Media. Yogyakarta.