

- 1. Pendahuluan**
- 2. Simulasi 1, 2 dan 3**
- 3. Simulasi 4, 5 dan 6**
- 4. Simulasi 7, 8 dan 9**
- 5. Simulasi Heater, Lift, Motor**
- 6. Simulasi Snake, Keypad, Seven Segment**
- 7. Ujian Praktikum**
- 8. Set Instruksi 8051 dan isis**
- 9. LED dan control push button**
- 10. Seven Segment dan Keypad**
- 11. Motor DC, steper dan servo**
- 12. ADC dan LCD display (Arduino)**
- 13. Komunikasi serial (Arduino) potensiometer dan sensor ping.**
- 14. Pengerjaan proyek**

PRAKTIKUM ARITMATIKA

(PENAMBAHAN, PENGURANGAN, PERKALIAN DAN PEMBAGIAN)

A. TUJUAN**B. TEORI SINGKAT****C. ALAT DAN BAHAN**

Laptop atau PC yang sudah dilengkapi program sms32v5.0

D. PROSEDUR KERJA

1. Buka program simulasi sms32v5.0
2. Ketik *source code* di bawah ini pada halaman editor program sms32v5.0

```

; =====
; ===== PENJUMLAHAN =====
    CLO          ;Tutup Jendela yang tidak diinginkan.
    MOV  AL,05   ;Masukkan nilai 05H ke register AL.
    MOV  BL,03   ;Masukkan nilai 03H ke register BL.
    ADD  AL,BL   ;Jumlahkan nilai di register AL dan
                  ;register BL, Simpan Hasilnya di AL
    MOV  AL,06   ;Masukkan nilai 06H ke register AL.
    MOV  BL,08   ;Masukkan nilai 08H ke register BL.
    ADD  AL,BL   ;Jumlahkan nilai di register AL dan
                  ;register BL, Simpan Hasilnya di AL
    MOV  AL,7E   ;Masukkan nilai 7EH ke register AL.
    MOV  BL,01   ;Masukkan nilai 01H ke register BL.
    ADD  AL,BL   ;Jumlahkan nilai di register AL dan
                  ;register BL, Simpan Hasilnya di AL
    ADD  AL,BL
    ADD  AL,BL
    ADD  AL,BL

; ===== PENGURANGAN =====
    MOV  AL,0F   ;Masukkan nilai 0FH ke register AL.
    MOV  BL,04   ;Masukkan nilai 04H ke register BL.
    SUB  AL,BL   ;Kurangkan nilai di register BL dengan
                  ;nilai yang ada di register Al, simpan
                  ;hasilnya di register BL
    MOV  AL,04   ;Masukkan nilai 04H ke register AL.
    MOV  BL,0E   ;Masukkan nilai 0EH ke register BL.
    SUB  AL,BL   ;Kurangkan nilai di register BL dengan
                  ;nilai yang ada di register Al, simpan
                  ;hasilnya di register BL
    MOV  AL,82   ;Masukkan nilai 82H ke register AL.
    MOV  BL,01   ;Masukkan nilai 01H ke register BL.
    SUB  AL,BL   ;Kurangkan nilai di register BL dengan
                  ;nilai yang ada di register Al, simpan
                  ;hasilnya di register BL
    SUB  AL,BL
    SUB  AL,BL
    SUB  AL,BL

```

```

; ===== PERKALIAN =====
    MOV  AL,05      ;Masukkan nilai 05H ke register AL.
    MOV  BL,08      ;Masukkan nilai 08H ke register BL.
    MUL  AL,BL      ;Kalikan nilai di register BL dengan
                    ;nilai yang ada di register Al, simpan
                    ;hasilnya di register BL
    MOV  AL,0A      ;Masukkan nilai 0AH ke register AL.
    MOV  BL,00      ;Masukkan nilai 00H ke register BL.
    MUL  AL,BL

    MOV  AL,32      ;Masukkan nilai 32H ke register AL.
    MOV  BL,03      ;Masukkan nilai 03H ke register BL.
    MUL  AL,BL

    MOV  AL,FB      ;Masukkan nilai FBH ke register AL.
    MOV  BL,02      ;Masukkan nilai 02H ke register BL.
    MUL  AL,BL

    MOV  AL,FB      ;Masukkan nilai FBH ke register AL.
    MOV  BL,FD      ;Masukkan nilai FDH ke register BL.
    MUL  AL,BL

; ===== PEMBAGIAN =====
    MOV  AL,08      ;Masukkan nilai 08H ke register AL.
    MOV  BL,02      ;Masukkan nilai 02H ke register BL.
    DIV  AL,BL      ;Kalikan nilai di register BL dengan
                    ;nilai yang ada di register Al, simpan
                    ;hasilnya di register BL
    MOV  AL,08      ;Masukkan nilai 08H ke register AL.
    MOV  BL,03      ;Masukkan nilai 03H ke register BL.
    DIV  AL,BL

    MOV  AL,08      ;Masukkan nilai 08H ke register AL.
    MOV  BL,05      ;Masukkan nilai 05H ke register BL.
    DIV  AL,BL

    MOV  AL,02      ;Masukkan nilai 02H ke register AL.
    MOV  BL,08      ;Masukkan nilai 08H ke register BL.
    DIV  AL,BL

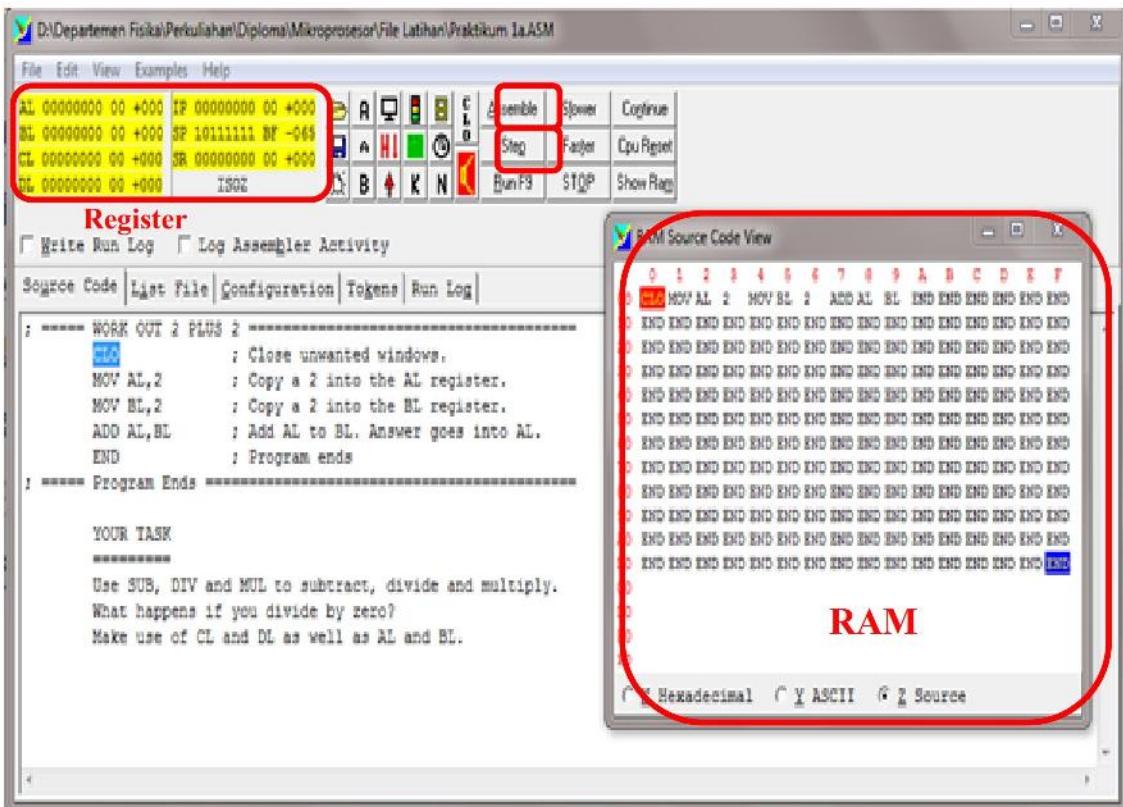
    MOV  AL,E0      ;Masukkan nilai E0H ke register AL.
    MOV  BL,08      ;Masukkan nilai 08H ke register BL.
    DIV  AL,BL

    MOV  AL,E0      ;Masukkan nilai E0H ke register AL.
    MOV  BL,FC      ;Masukkan nilai FCH ke register BL.
    DIV  AL,BL      ;Kalikan nilai di register BL dengan

    END          ;Akhiri program
; ===== Program Ends =====

```

3. Setelah diketik, simpan file tersebut didirectory hardisk anda dengan nama “Praktikum 1a.ASM”.
4. Klik assamble, kemudian klik Step. Lihat Gambar di bawah ini:



- Setiap kali mengklik step maka sintak yang kita ketikkan di jendela editor (kecuali komen) akan dieksekusi perbaris. Sintak yang akan dieksekusi di jendela editor ketika mengklik Step adalah yang diblok dengan warna biru, sedangkan pada RAM adalah yang diblok berwarna merah.
5. Amati perubahan isi register-register dan RAM setiap kali mengklik Step dan cek apakah hasil penjumlahan, pengurangan, perkalian dan pembagiannya sudah benar. Jika program telah berhenti dan ingin mengeksekusi program lagi maka klik assemble, kemudian klik step lagi.
 6. Diskusikan dengan teman satu kelompok pertanyaan-pertanyaan berikut:
 - a. Apa operator yang digunakan untuk memasukkan angka ke register umum simulator?
 - b. Apa operator untuk melakukan operasi aritmatika (penjumlahan, pengurangan, perkalian dan pembagian)?
 - c. Berapa batas minimum negatif dan maksimum positif nilai yang bisa ditampung oleh register umum simulator?
 - d. Kapan register status (flag) bernilai 1 pada bagian S,O dan Z?
 - e. Ada berapa operasi aritmatika pada program di atas yang menyebabkan status register flag S, O dan Z bernilai 1?

LANGKAH PERCOBAAN 2.

1. Klik file kemudian New pada simulator untuk membuka jendela editor baru
2. Tuliskan *source code* di bawah ini:

```

; ===== CONTROL THE TRAFFIC LIGHTS
=====

CLO          ;Tutup jendela yang tidak diinginkan.
Start:
    MOV AL, 0      ;Mematikan lampu lalu lintas
    OUT 01         ;Masukkan nilai 00000000b ke register Al
    OUT 01         ;Kirim nilai pada register Al ke port 1
                    ;(Lampu lalu lintas)
    MOV AL, FC     ;Menghidupkan lampu lalu lintas
    OUT 01         ;Masukkan nilai 11111100 ke register AL
    OUT 01         ;Kirim nilai pada register Al ke port 1
    JMP Start      ;Kembali (Lompat) ke label start.
    END            ;Program ends.

; ===== Program Ends
=====
```

3. Setelah diketik, simpan file tersebut didirectory hardisk anda dengan nama “Praktikum 1b.ASM”.
4. Klik assamble, kemudian Step atau Run untuk menjalankan atau mengeksekusi sintak:
Setiap kali mengklik step maka sintak yang kita ketikkan di jendela editor (kecuali komen) akan dieksekusi perbaris. Sintak yang akan dieksekusi di jendela editor ketika mengklik Step adalah yang diblok dengan warna biru, sedangkan pada RAM adalah yang diblok berwarna merah.
5. Amati perubahan isi register-register dan RAM setiap kali mengklik Step. Jika program telah berhenti dan ingin mengeksekusi program lagi maka klik assemble, kemudian klik step lagi.
6. Diskusikan dengan teman satu kelompok pertanyaan-pertanyaan berikut:
 - a. Apa operator yang digunakan untuk mengirim data dari register AL ke port dan port berapa yang digunakan untuk menjalankan simulator lampu laulintas?
 - b. Apa operator yang digunakan untuk kembali ke baris sintak sebelumnya?
 - c. Bisakah nilai dari register selain AL dikirim ke port?
 - d. Tuliskan bit keberapa yang digunakan untuk mengontrol hidup dan mati setiap lampu lalu lintas?
 - e. Modifikasi sintak di atas sehingga urutan menyala lampu lalu lintas sesuai dengan yang sebenarnya!

LANGKAH PERCOBAAN 3

1. Klik file kemudian New pada simulator untuk membuka jendela editor baru
2. Tuliskan *source code* di bawah ini:

```

; A program to demonstrate MOV commands. Mov is short for
move.
; -----
; CLO          ; Close unwanted windows.
; ===== IMMEDIATE MOVES =====
MOV  AL,15    ; Masukkan nilai 15H ke Register AL
MOV  BL,40    ; Masukkan nilai 40H ke Register BL
MOV  CL,50    ; Masukkan nilai 50H ke Register CL
MOV  DL,60    ; Masukkan nilai 60H ke Register DL
Foo:
    INC AL        ; Naikkan nilai Register Al satu tingkat

; ===== INDIRECT MOVES =====
MOV  [A0],AL ; Copy nilai di Register AL ke alamat
             ; RAM [40]
MOV  BL,[40] ; Copy nilai yang ada di alamat RAM [A0]
             ; ke dalam Register BL

; ===== REGISTER INDIRECT MOVES =====
MOV  [CL],AL ; Copy nilai yang ada pada Register AL ke
             ; alamat RAM yang bersesuaian dengan
             ; nilai yang ada pada Register CL
MOV  BL,[CL] ; Copy nilai yang ada pada alamat RAM
             ; yang bersesuaian dengan nilai pada
             ; Register CL ke Register BL

JMP Foo       ; Kembali (Lompat) ke Label Foo.

END
; -----

```

3. Setelah diketik, simpan file tersebut didirectory hardisk anda dengan nama “Praktikum 1c.ASM”.
4. Klik assamble, kemudian Step atau Run untuk menjalankan atau mengeksekusi sintak:
5. Setiap kali mengklik step maka sintak yang kita ketikkan di jendela editor (kecuali komen) akan dieksekusi perbaris. Sintak yang akan dieksekusi di jendela editor ketika mengklik Step adalah yang diblok dengan warna biru, sedangkan pada RAM adalah yang diblok berwarna merah.
6. Amati perubahan isi register-register dan RAM setiap kali mengklik Step. Jika program telah berhenti dan ingin mengeksekusi program lagi maka klik assemble, kemudian klik step lagi.
7. Diskusikan dengan teman satu kelompok pertanyaan-pertanyaan berikut:
 - a. Bisakah memindahkan nilai antar Register umum secara langsung (misalnya ingin memindahkan nilai pada Register AL ke Register DL)?
 - b. Tuliskan sintak untuk memindahkan isi Register BL ke Register CL!

- c. Bisakah memasukkan nilai ke alamat RAM secara langsung?
- d. Tuliskan sintak untuk memasukkan nilai 45H ke alamat RAM 80!
- e. Berapakah nilai ASCII hurup “H”, “E”, “L”, “L”, dan “O” dalam bilangan HEX? Simpan nilai HEX tersebut ke alamat RAM C0, C1, C2, C3 dan C4. Ketika nilai HEX ascii disimpan di alamat RAM C0 sampai FF maka kode ascii tersebut akan diterjemahkan menjadi karakter pada layar VDU simulator.
- f. Buatlah Program untuk menampilkan Nama kelompok anda pada layar VDU simulator dengan format seperti di bawah ini!
Kelompok 1: => ganti sesuai dengan kelompok anda
 - 1. Nama mahasiswa
 - 2. Nama mahasiswa
 - 3. Nama mahasiswa
 - 4. Nama mahasiswa
- g. Berapa banyak baris dan kolom karakter yang bisa ditampilkan pada layar VDU simulator?

LANGKAH PERCOBAAN 4

1. Klik file kemudian New pada simulator untuk membuka jendela editor baru
2. Tuliskan *source code* di bawah ini:

```

; -----
; Input key presses from the keyboard until Enter is
; pressed.
; -----
CLO          ; Tutup Windows yang tidak diinginkan
Rep:
    IN   00      ; Tunggu sampai keyboard ditekan, setelah
                  ; itu simpan nilainya di Register AL
    CMP  AL,0D    ; Apakah yang ditekan adalah Pad
                  ; "ENTER"? (Kode ASCII tobol ENTER adalah
                  ; 0D)
    JNZ  Rep      ; Jika bukan "ENTER" maka kembali ke
                  ; label Rep, jika benar maka jalankan
                  ; sintak dibawahnya
    END
; -----

```

3. Setelah diketik, simpan file tersebut didirectory hardisk anda dengan nama “Praktikum 2a.ASM”.
4. Klik assamble, kemudian Step atau Run untuk menjalankan atau mengeksekusi sintak:
5. Setiap kali mengklik step maka sintak yang kita ketikkan di jendela editor (kecuali komen) akan dieksekusi perbaris. Sintak yang akan dieksekusi di jendela editor ketika mengklik Step adalah yang diblok dengan warna biru, sedangkan pada RAM adalah yang diblok berwarna merah.
6. Amati perubahan isi register-register dan RAM setiap kali mengklik Step. Jika program telah berhenti dan ingin mengeksekusi program lagi maka klik assemble, kemudian klik step lagi.
7. Diskusikan dengan teman satu kelompok pertanyaan-pertanyaan berikut:
 - a. Ketika simulator menjalankan sintak CMP, bagaimanakah kondisi nilai dari register flag S, O, dan Z saat yang ditekan adalah pad bukan “ENTER”?
 - b. Ketika simulator menjalankan sintak CMP, bagaimanakah kondisi nilai dari register flag S, O, dan Z saat yang ditekan adalah pad “ENTER”?
 - c. Bagaimanakah kondisi nilai dari register flag S, O, dan Z saat program kembali ke label Rep ataupun menjalankan program berikutnya?
 - d. Bukalah Help Simulator dengan menekan F1 atau klik Help kemudian content. Pilihlah menu “Detailed Instruction Set” pada jendela help, kemudian pilih menu “Jump Instructions”, pelajarilah operator-operator “JUMP” dan syarat-syaratnya. Tanyakan kepada asisten atau dosen jika belum paham.
 - e. Modifikasi sintak di atas:
 1. Ketika menekan pad keyboard maka karakter keyboard tersebut akan muncul di VDU simulator. (ingat alamat RAM yang bisa digunakan untuk memunculkan karakter di VDU)

2. Sama seperti nomor 1, tetapi karakter yang baru ditampilkan disebelah karakter yang lama dan seterusnya. (Tips. Gunakan register BL, CL, atau DL untuk menyimpan alamat RAM yang akan diakses dan gunakan operator INC untuk meningkatkan nilai pada register).
3. Ketika menekan pad keyboard maka code ascii-nya disimpan di alamat RAM (tetapi belum dimunculkan di VDU), setelah pad “ENTER” ditekan maka semua code ascii yang telah disimpan di RAM tadi ditampilkan ke VDU dengan urutan yang sama. Misalkan tadi ditekan karakter “A”, “B”, “C”, “D” maka yang muncul di VDU adalah karakter tersebut dengan urutan “A B C D”.
4. Sama seperti nomor 3, tetapi yang akan tampil di VDU adalah urutanya berkebalikan. Misalkan tadi ditekan karakter “A”, “B”, “C”, “D” maka yang muncul di VDU adalah karakter tersebut dengan urutan “D C B A”.

LANGKAH PERCOBAAN 5.

1. Klik file kemudian New pada simulator untuk membuka jendela editor baru
2. Tuliskan *source code* di bawah ini:

```

; -----
; Program berikut adalah contoh prosedur delay yang
; dikontrol oleh Register AL.
; Ketika prosedur selesai, nilai awal register AL
; dikembalikan menjadi nilai sebelumnya dengan
; menggunakan operator Push, Pop, Pushf and Popf. Pada
; contoh ini menggunakan tiga jenis prosedur delay.

;----- Program Utama -----

Start:
    MOV AL,8      ; Sebagai delay pendek.
    CALL 30       ; Memanggil prosedur dengan alamat RAM
                  ; [30]
    MOV AL,10     ; Sebagai delay menengah.
    CALL 30       ;
    MOV AL,20     ; Sebagai delay panjang.
    CALL 30       ;

JMP Start        ; Kembali ke label Start.

; ----- Procedur delay disimpan pada alamat RAM [30] -----
ORG 30          ; Menjalankan machine code dari alamat
                  ; [30]

    PUSH AL      ; Simpan nilai Register AL di stack.
    PUSHF         ; Simpan CPU flags di stack.

Rep:
    DEC AL       ; Kurangi nilai AL satu nilai.
    JNZ REP       ; Kembali ke label Rep jika nilai di AL
                  ; tidak Nol.

    POPF          ; Kembalikan CPU flags dari stack.
    POP AL        ; Kembalikan nilai AL dari stack.

    RET           ; Kembali ke procedure awal (Program
                  ; Utama).

; -----
END
; -----

```

3. Setelah diketik, simpan file tersebut didirectory hardisk anda dengan nama “Praktikum 2b.ASM”.
4. Klik assamble, kemudian Step atau Run untuk menjalankan atau mengeksekusi sintak:

5. Setiap kali mengklik step maka sintak yang kita ketikkan di jendela editor (kecuali komen) akan dieksekusi perbaris. Sintak yang akan dieksekusi di jendela editor ketika mengklik Step adalah yang diblok dengan warna biru, sedangkan pada RAM adalah yang diblok berwarna merah.
6. Amati perubahan isi register-register dan RAM setiap kali mengklik Step. Jika program telah berhenti dan ingin mengeksekusi program lagi maka klik assemble, kemudian klik step lagi.
7. Diskusikan dengan teman satu kelompok pertanyaan-pertanyaan berikut:
 - a. Apa fungsi operator CALL [30] pada sintak di atas? Bisakah diganti dengan angka selain 30?
 - b. Apa fungsi operator PUSH, PUSHF, POPF dan POP?
 - c. Pada saat menjalankan sintak mana saja register Stack Pointer aktif?
 - d. Pada saat simulator menjalankan sintak PUSH, PUSHF, POPF dan POP apakah register flag aktif?
 - e. Modifikasi sintak di atas untuk mengatur hidup dan mati lampu lalu lintas, dimana delay singkat untuk lampu kuning, sedang lampu hijau dan lama lampu merah. Ganti lamanya delay jika terlalu lama atau cepat.

LANGKAH PERCOBAAN 6.

1. Klik file kemudian New pada simulator untuk membuka jendela editor baru
2. Tuliskan *source code* di bawah ini:

```

; -----
; Program untuk membaca teks dan menyimpannya di RAM.
; Akhir teks ditandai dengan karakter ASCII null.
; -----
; Program Utama
    MOV BL,70 ; [70] adalah alamat dimana teks akan
              ; disimpan.
    CALL 10   ; Procedure pada alamat [10] untuk
              ; membaca text dan menyimpannya pada
              ; alamat RAM yang ditunjukkan nilai di
              ; register BL. Setelah menekan Enter maka
              ; isi Register BL awal [70] akan
              ; dikembalikan lagi menggunakan POPF dan
              ; POP.
    CALL 40   ; Prosedur ini digunakan untuk
              ; menampilkan teks yang telah disimpan di
              ; RAM ke VDU. Modifikasi sintaknya.
    HALT     ; Printah ini hanya untuk menghentikan
              ; program (ganti sintaknyanya).
; -----
; PROSEDUR UNTUK MEMBACA DAN MENYIMPAN TEKS
    ORG 10      ; Code dimulai pada alamat [10]
    PUSH AL     ; Simpan AL distack
    PUSH BL     ; Simpan BL distack
    PUSHF       ; Simpan CPU flags distack

Rep:
    IN 00        ; Input dari port 00 (keyboard)
    CMP AL,0D   ; Apakah keypad Enter ditekan?
    JZ Stop      ; Jika YA maka lompat ke LABEL Stop
    MOV [BL],AL  ; Masukkan nilai yang ditekan ke alamat
                  ; RAM [BL]
    INC BL       ; Tambah Satu nilai BL.
    JMP Rep      ; Kembali ke LABEL Rep

Stop:
    MOV AL,0     ; Masukkan nilai 00H ke AL sebagai
                  ; penanda
    MOV [BL],AL  ; Copy NULL character ke pada alamat
                  ; RAM [BL]
    POPF        ; Kembalikan kondisi flags dari stack
    POP BL      ; Kembalikan BL dari stack
    POP AL      ; Kembalikan AL dari the stack

    RET         ; Kembali ke prosedur.
; -----
; Prosedur untuk menampilkan TEKS pada VDU

```

```
ORG 40      ; Code dimulai dari alamat [40]
; ***** Lengkapi Sintak di bagian ini*****
RET
; -----
END          ; It is correct to use END at the end.
; -----
```

3. Setelah diketik, simpan file tersebut didirectory hardisk anda dengan nama “Praktikum 2c.ASM”.
4. Klik assamble, kemudian Step atau Run untuk menjalankan atau mengeksekusi sintak:
5. Setiap kali mengklik step maka sintak yang kita ketikkan di jendela editor (kecuali komen) akan dieksekusi perbaris. Sintak yang akan dieksekusi di jendela editor ketika mengklik Step adalah yang diblok dengan warna biru, sedangkan pada RAM adalah yang diblok berwarna merah.
6. Amati perubahan isi register-register dan RAM setiap kali mengklik Step. Jika program telah berhenti dan ingin mengeksekusi program lagi maka klik assemble, kemudian klik step lagi.
7. Diskusikan dengan teman satu kelompok pertanyaan-pertanyaan berikut:
 - a. Ada berapa prosedur pada contoh sintak di atas?
 - b. Lengkapi sintak pada bagian yang belum lengkap di program di atas supaya code ascii yang telah disimpan di RAM dapat ditampilkan di VDU!
 - c. Buatlah program dengan 3 prosedur yang terdiri dari:
 1. Prosedur membaca dan menyimpan teks dari keyboard.
 2. Prosedur mengkonversi huruf besar menjadi huruf kecil.
 3. Prosedur menampilkan huruf ke VDU yang sudah dirubah menjadi huruf kecil.

LANGKAH PERCOBAAN 7.

1. Klik file kemudian New pada simulator untuk membuka jendela editor baru
2. Tuliskan *source code* di bawah ini:

```

; ----- DATA TABLES -----
    JMP Start ; Skip past the data table.
    DB 84      ; Permulaan data table.
    DB C8      ; nilai untuk mengontrol traffic lights
    DB 31      ; This sequence is simplified.
    DB 58      ; Last entry is also used as end marker

Start:
    MOV BL,02   ; 02 merupakan permulaan alamat di RAM
                  ; tempat menyimpan database

Rep:
    MOV AL,[BL] ; Copy data dari database ke AL
    OUT 01      ; keluarkan nilai Al ke port 01
    CMP AL,58   ; nilai terakhir databese???
    JZ Start    ; Jika YA lompat ke LABEL Start
    INC BL      ; Jika TIDAK naikkan nilai BL
    JMP Rep     ; Kembali ke LABEL Rep untuk membaca
                  ; database berikutnya
    END
; -----

```

3. Setelah diketik, simpan file tersebut didirectory hardisk anda dengan nama “Praktikum 2c.ASM”.
4. Klik assamble, kemudian Step atau Run untuk menjalankan atau mengeksekusi sintak:
5. Setiap kali mengklik step maka sintak yang kita ketikkan di jendela editor (kecuali komen) akan dieksekusi perbaris. Sintak yang akan dieksekusi di jendela editor ketika mengklik Step adalah yang diblok dengan warna biru, sedangkan pada RAM adalah yang diblok berwarna merah.
6. Amati perubahan isi register-register dan RAM setiap kali mengklik Step. Jika program telah berhenti dan ingin mengeksekusi program lagi maka klik assemble, kemudian klik step lagi.
7. Diskusikan dengan teman satu kelompok pertanyaan-pertanyaan berikut:
 - a. Dialamat berapa saja tempat menyimpan database di RAM?
 - b. Bagaimana cara program mengetahui telah membaca database yang terakhir?
 - c. Buat program di bawah ini:
 1. Gunakan database untuk mengontrol lampu lalu lintas agar tidak terjadi overlap!
 2. Gunakan database untuk mengontrol navigasi snake melalui port 04. Heks FF untuk meriset.
 3. Gunakan database untuk menggerakkan stepper motor ke depan dan kebelakang.

LANGKAH PERCOBAAN 8.

1. Klik file kemudian New pada simulator untuk membuka jendela editor baru
2. Tuliskan *source code* di bawah ini:

```

; ----- Melawatkan Parameters -----

; -- Menggunakan Registers untuk melewatkkan parameters ---
JMP Start ;
DB 0      ; Nilai database disimpan di RAM [02]
DB 0      ; Nilai database disimpan di RAM [03]

Start:
MOV AL,5
MOV BL,4
CALL 30    ; Prosedur menambahakan AL dan BL,
            ; kemudian hasilnya disimpan di AL.

; ---- Menggunakan alamat RAM untuk melewatkkan parameter --
MOV AL,3
MOV [02],AL ; Simpan nilai di AL ke alamat RAM [02]
MOV BL,1
MOV [03],BL ; Simpan nilai di AL ke alamat RAM [03]
CALL 40

; ---- Menggunakan Stack untuk melewatkkan parameter -----
MOV AL,7
PUSH AL
MOV BL,2
PUSH BL
CALL 60
POP BL
POP AL      ; Akan berisi hasil perhitungan

JMP Start ;

; ----- Prosedur menambahakan dua nilai -----
; Parameters dilewatkan pada procedure menggunakan
; Register AL and BL. Hasilnya disimpan di AL.
; Metode ini sederhana, tetapi kurang bagus jika ada
; banyak parameter yang dilewatkan.

ORG 30      ; Kode dimulai di alamat RAM [30]

ADD AL,BL   ; Lakukan penambahan, hasilnya simpan di
            ; Register AL
RET         ;

; ----- Prosedur menambahakan dua nilai -----
; Parameters dilewatkan pada procedure menggunakan
; alamat RAM. Hasil simpan di alamat RAM
; Metode ini lebih kompleks dan tidak ada batasan
; banyaknya parameter yang akan disimpan di RAM, kecuali

```

```

; kapasitas RAM dan RAM itu sedang digunakan.

ORG 40      ; Kode dimulai di alamat [40]

PUSH CL      ; Simpan registers and flags di stack
PUSH DL
PUSHF

MOV CL,[02] ; ambil parameter dari RAM
MOV DL,[03] ; ambil parameter dari RAM
ADD CL,DL   ; lakukan penambahan
MOV [02],CL ; simpan hasil di RAM

POPF          ; Kembalikan kondisi awal register
POP DL        ; dan nilianya
POP CL

RET

; ----- Prosedur menambahkan dua nilai -----
; Nilia akan ditambahkan di stack.
; POP parameters mengakhiri stack
; lakukan penambahan
; simpan jawaban di stack
; mayoritas dari prosedur pemanggilan dalam kenyataanya
; menjadikan sebagai pelewatt parameter.
; Pada umumnya untuk alamat dengan struktur data
; kompleks di RAM dilewatkan pada prosedur menggunakan
; stack.

ORG 60      ; Kode dimulai pada alamat [60]

POP DL      ; Kembali ke alamat
POP BL      ; sebuah parameter
POP AL      ; sebuah parameter

ADD AL,BL

PUSH AL      ; jawaban ; banyaknya data yang di
              ; masukkan
PUSH AL      ; jawaban; pencocokan jawaban yang
              ; dimasukkan.
PUSH DL      ; kembalikan stack seperti sebelumnya.

RET
; -----
END
; -----

```

3. Setelah diketik, simpan file tersebut didirectory hardisk anda dengan nama “Praktikum 2c.ASM”.
4. Klik assamble, kemudian Step atau Run untuk menjalankan atau mengeksekusi sintak:
5. Setiap kali mengklik step maka sintak yang kita ketikkan di jendela editor (kecuali komen) akan dieksekusi perbaris. Sintak yang akan dieksekusi di jendela editor

ketika mengklik Step adalah yang diblok dengan warna biru, sedangkan pada RAM adalah yang diblok berwarna merah.

6. Amati perubahan isi register-register dan RAM setiap kali mengklik Step. Jika program telah berhenti dan ingin mengeksekusi program lagi maka klik assemble, kemudian klik step lagi.
7. Diskusikan dengan teman satu kelompok pertanyaan-pertanyaan berikut:
 - a. Ada berapa cara melewatkkan parameter? Tuliskan!
 - b. Dari beberapa cara untuk melewatkkan parameter manakah lebih memungkinkan melewatkkan parameter lebih banyak?
 - c. Buat program di bawah ini:
 1. Buat program melipatgandakan bilangan, lewatkan satu parameter ke prosedur menggunakan register. Gunakan register yang sama untuk menyimpan hasilnya!
 2. Buat program untuk merubah tanda bilangan, yang negatif menjadi positif dan yang positif menjadi negatif. Lewatkan paremeter menggunakan RAM. Gunakan alamat RAM yang sama untuk menyimpan hasilnya.
 3. Buat program untuk menghitung factorial 1, 2, 3, 4 atau 5. Gunakan stack untuk melewatkkan parameter dan mengembalikan hasilnya.

LANGKAH PERCOBAAN 9.

1. Klik file kemudian New pada simulator untuk membuka jendela editor baru
2. Tuliskan *source code* di bawah ini:

```

; -----
; Contoh interupsi software.
; -----
    JMP  Start    ; Melewati table vector interupsi
    DB   51        ; Vector 02 ditempatkan pada alamat [51]
    DB   71        ; Vector 03 ditempatkan pada alamat [71]
Start:
    INT  02        ; Kerjakan interrupt 02
    INT  03        ; Kerjakan interrupt 03
    JMP  Start
; -----


    ORG  50
    DB   E0        ; Data table
                  ; Kode interrupt dimulai di sini
    MOV  AL,[50]  ; Copy nilai bits dari RAM ke AL
    NOT  AL        ; invers bits di AL
    MOV  [50],AL  ; Copy kembali invers bits ke RAM.
    OUT  01        ; Kirim data ke traffic lights
    IRET
; -----


    ORG  70
    DB   0         ; Data Table

; Kode interupsi dimulai di sini
    MOV  AL,[70]  ; Copy bits dari RAM ke AL
    NOT  AL        ; Invert bits di AL
    AND  AL,FE    ; Rubah semua bit menjadi nol
    MOV  [70],AL  ; Copy kembali invers bits ke RAM
    OUT  02        ; Kirim data ke tampilan seven segment
    IRET
; -----


    END
; -----

```

3. Setelah diketik, simpan file tersebut didirectory hardisk anda dengan nama “Praktikum 2c.ASM”.
4. Klik assamble, kemudian Step atau Run untuk menjalankan atau mengeksekusi sintak:
5. Setiap kali mengklik step maka sintak yang kita ketikkan di jendela editor (kecuali komen) akan dieksekusi perbaris. Sintak yang akan dieksekusi di jendela editor ketika mengklik Step adalah yang diblok dengan warna biru, sedangkan pada RAM adalah yang diblok berwarna merah.
6. Amati perubahan isi register-register dan RAM setiap kali mengklik Step. Jika program telah berhenti dan ingin mengeksekusi program lagi maka klik assemble, kemudian klik step lagi.

7. Diskusikan dengan teman satu kelompok pertanyaan-pertanyaan berikut:
 - a. Ada berapa software interupt pada contoh program di atas?
 - b. Berapa port untuk mengaktifkan simulasi traffic light dan seven segment?
 - c. Buat program di bawah ini dengan memodifikasi contoh program sebelumnya:
 1. Buat interrupt 02 untuk membaca dan menyimpan karakter yang ditekan dari key board ke RAM.
 2. Buat interrupt baru untuk menampilkan karakter keyboard di VDU saat ditekan keypad enter dan menghapus karakter di VDU saat ditekan keypad Back Space.

LANGKAH PERCOBAAN 10.

1. Klik file kemudian New pada simulator untuk membuka jendela editor baru
2. Tuliskan *source code* di bawah ini:

```

; -----
; Contoh menggunakan interrupts hardware.
; Program ini untuk memutar motor stepper secara
; continuous dan menjalankan lampu lalu lintas setiap ada
; hardware interrupt
; Uncheck the "Show only one peripheral at a time" box
; to enable both displays to appear simultaneously.
; -----


JMP Start ; Jump past table of interrupt vectors
DB 50      ; Vector at 02 pointing to address 50

Start:
    STI          ; Set I flag. Enable hardware interrupts
    MOV AL,11   ;
Rep:
    OUT 05      ; Stepper motor
    ROR AL      ; Rotate bits in AL right
    JMP Rep
    JMP Start
; -----
; -----
ORG 50

PUSH al       ; Save AL onto the stack.
PUSH bl       ; Save BL onto the stack.
PUSHF        ; Save flags onto the stack.

JMP PastData

DB 84        ; Red           Green
DB c8        ; Red+Amber     Amber
DB 30        ; Green          Red
DB 58        ; Amber          Red+Amber
DB 57        ; Used to track progress through table

PastData:
    MOV BL,[5B] ; BL now points to the data table
    MOV AL,[BL] ; Data from table goes into AL
    OUT 01      ; Send AL data to traffic lights
    CMP AL,58   ; Last entry in the table
    JZ Reset    ; If last entry then reset pointer

    INC BL      ; BL points to next table entry
    MOV [5B],BL ; Save pointer in RAM
    JMP Stop

Reset:
    MOV BL,57   ; Pointer to data table start address

```

```

MOV [5B],BL ; Save pointer into RAM location 54
Stop:
    POPF      ; Restore flags to their previous value
    POP bl     ; Restore BL to its previous value
    POP al     ; Restore AL to its previous value

    IRET
; -----
-----

END
;
-----

```

Setelah diketik, simpan file tersebut didirectory hardisk anda dengan nama “Praktikum 2c.ASM”.

3. Klik assamble, kemudian Step atau Run untuk menjalankan atau mengeksekusi sintak:
4. Setiap kali mengklik step maka sintak yang kita ketikkan di jendela editor (kecuali komen) akan dieksekusi perbaris. Sintak yang akan dieksekusi di jendela editor ketika mengklik Step adalah yang diblok dengan warna biru, sedangkan pada RAM adalah yang diblok berwarna merah.
5. Amati perubahan isi register-register dan RAM setiap kali mengklik Step. Jika program telah berhenti dan ingin mengeksekusi program lagi maka klik assemble, kemudian klik step lagi.
6. Diskusikan dengan teman satu kelompok pertanyaan-pertanyaan berikut:
 - d. Ada berapa software interupt pada contoh program di atas?
 - e. Berapa port untuk mengaktifkan simulasi traffic light dan seven segment?
 - f. Buat program di bawah ini dengan memodifikasi contoh program sebelumnya:
 1. Gunakan hardware interrupt untuk mengontrol heater dan thermostat.
Dengan menggunakan hardware interrupt kita bisa mengatur lama mati dan hidup agar suhu tetap terjaga pada kisaran 20 °C.
 2. .

MENGINSTAL WIN8051 IDE

Menginstal win8051 IDE

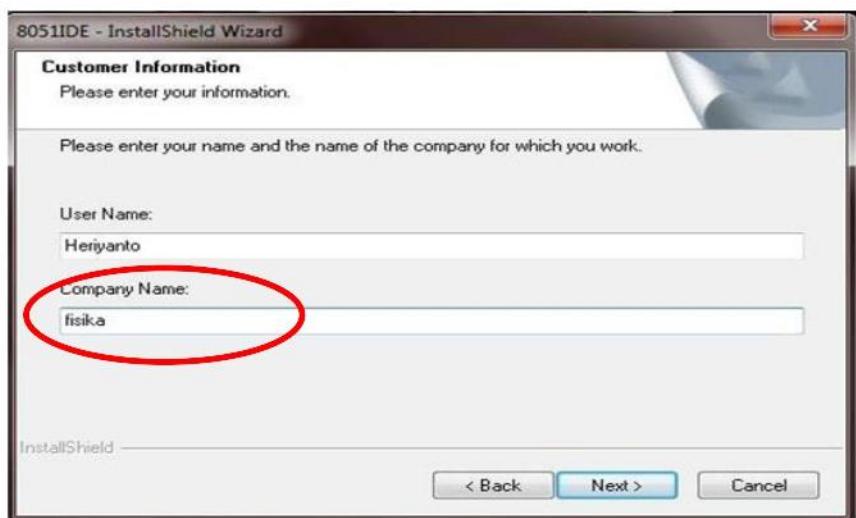
1. Download software win8051 IDE di <http://www.acebus.com/win8051.htm#download>.
Download software win8051 dan keywin8051 agar semua fiturnya dapat diaktifkan atau minta ke asisten.
2. Ekstrak filenya menggunakan winRAR, setelah file diekstrak klik dua kali file **setup.exe** seperti pada Gambar di bawah ini;

Name	Date modified	Type	Size
data1.cab	2/2/2009 1:16 PM	WinRAR archive	342 KB
data1.hdr	2/2/2009 1:16 PM	HDR File	12 KB
data2.cab	2/2/2009 1:16 PM	WinRAR archive	863 KB
engine32.cab	1/24/2008 11:16 AM	WinRAR archive	541 KB
layout.bin	2/2/2009 1:16 PM	BIN File	1 KB
setup.exe	11/14/2005 3:24 AM	Application	119 KB
setup.ibt	2/2/2009 1:16 PM	IBT File	388 KB
setup.ini	09 1:16 PM	Configuration sett...	1 KB
setup.inx	09 1:16 PM	INX File	209 KB

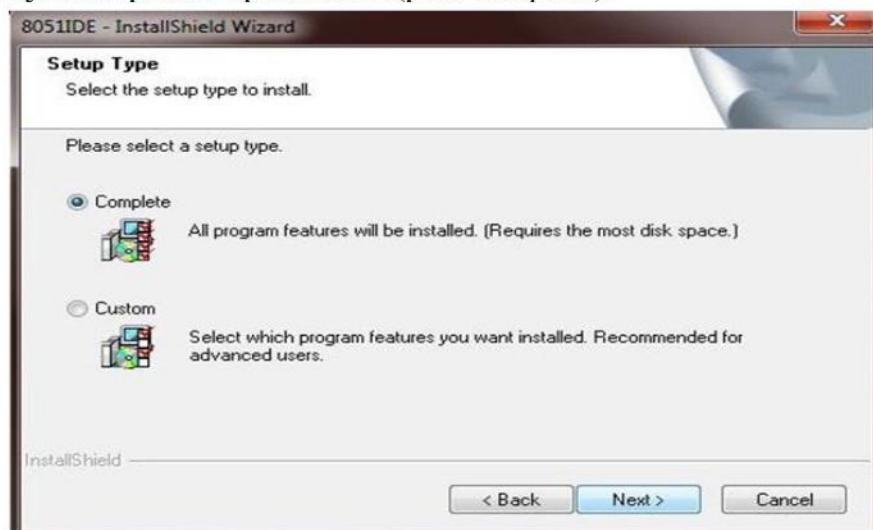
3. Klik NEXT.



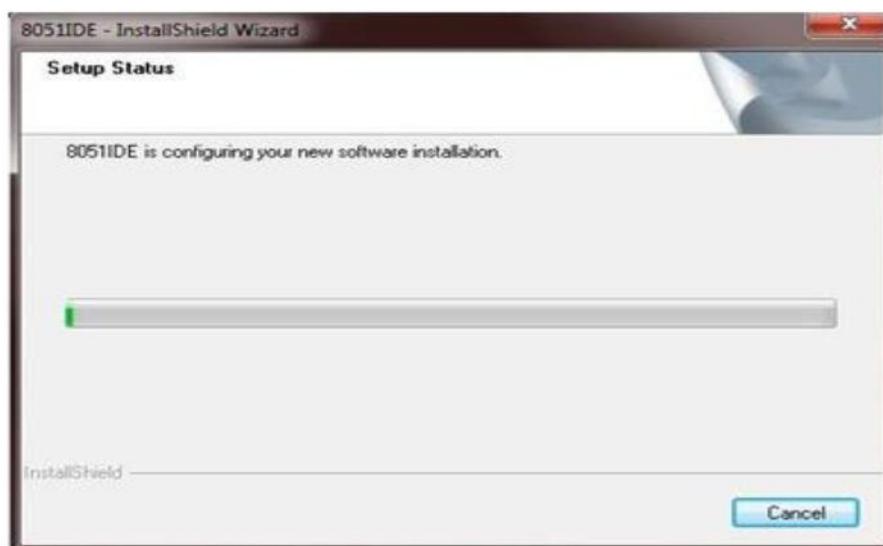
4. Masuk ke jendela untuk memasukkan Company Name: Tuliskan apa saja (contoh Tek_diploma).



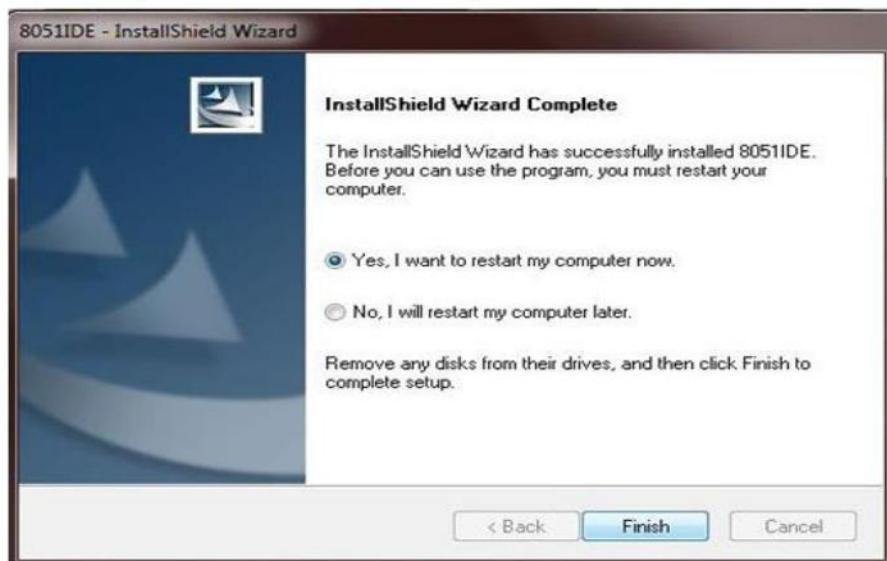
5. Masuk ke jendela pilihan tipe instalasi (pilih *Complete*).



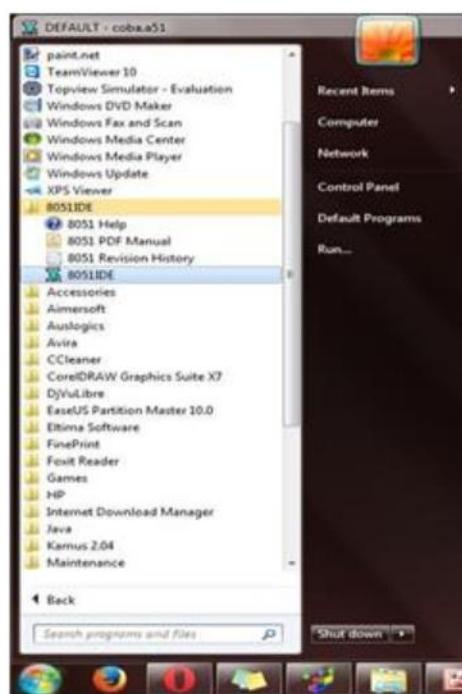
6. Proses intalasi sedang berlangsung.



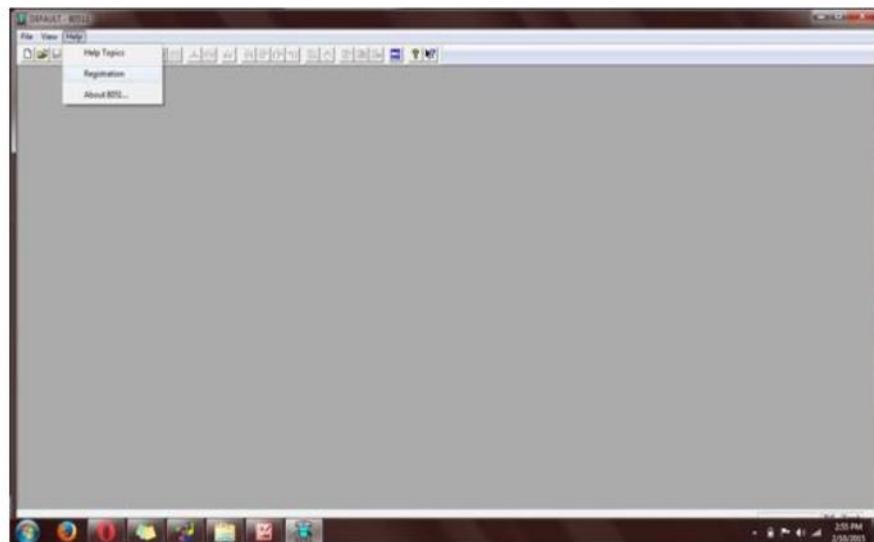
7. Proses Selesai dan pilih untuk merestart komputer.



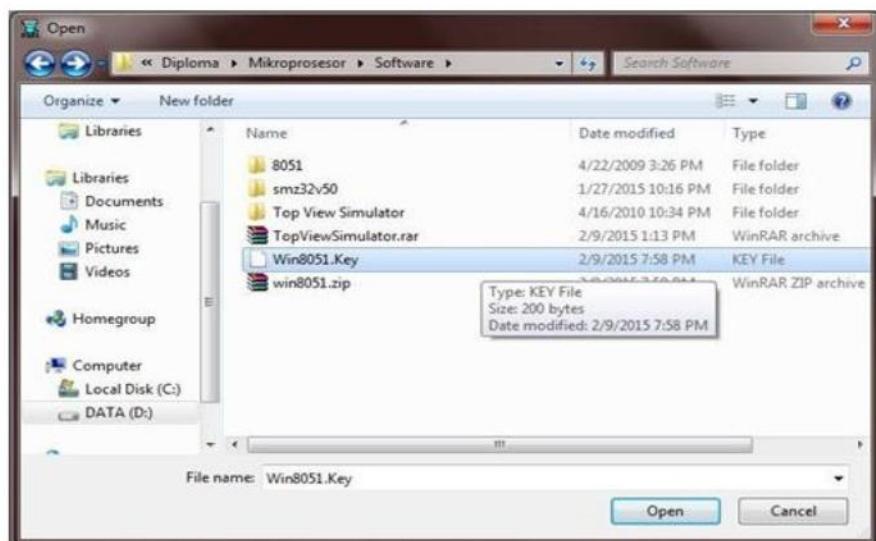
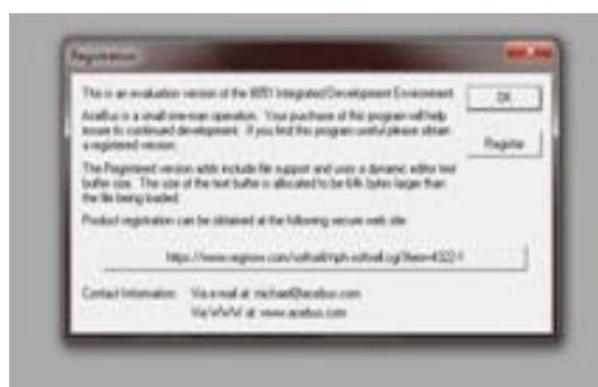
8. Buka aplikasi win8051 IDE yang telah terinstal.



9. Setelah terbuka lakukan registrasi dengan cara klik *Registration* agar semua fitur bisa digunakan.



10. Akan terbuka jendela baru. Klik *Registration* kemudian browse Key (Win8051.key) ditempat anda menyimpan file Setup instalasi.



SET INSTRUKSI MCS51

A. TUJUAN

1. Memahami fungsi-fungsi instruksi dari set instruksi MCS51
2. Dapat menggunakan software program bantu simulator/emulator untuk memahami kerja dari mikrokontroler.

B. ALAT DAN BAHAN

Laptop yang telah di install program simulator WIN8051IDE

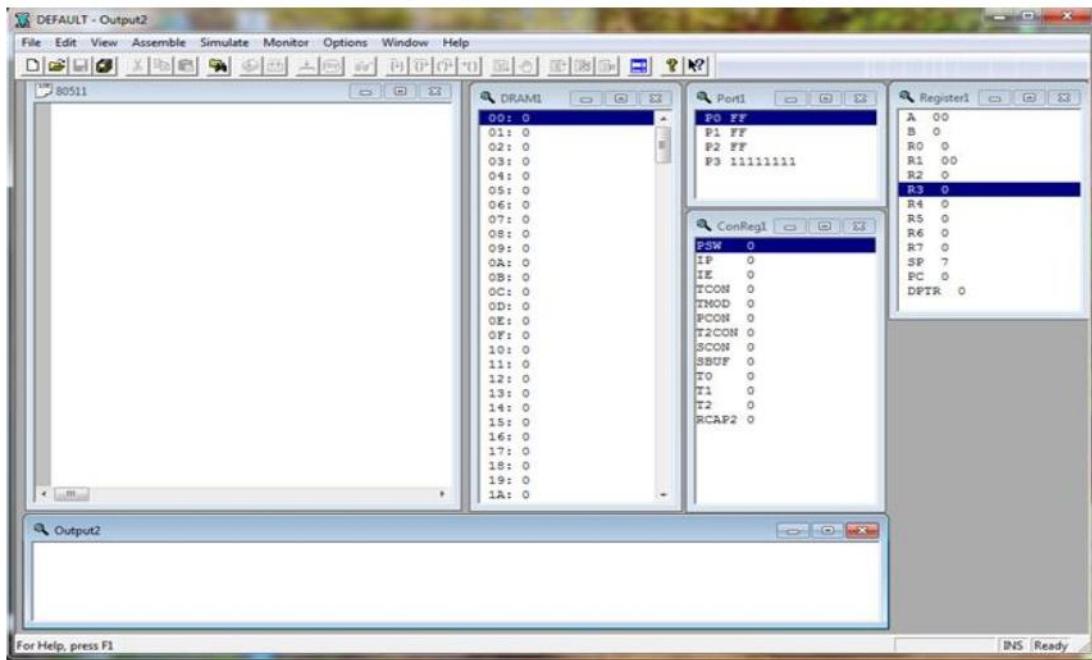
C. PROSEDUR KERJA

1. Pengenalan Program Simulasi WIN8051

Program simulator WIN8051IDE adalah program terintegrasi untuk pembangunan sistem berbasis MCS51 di mana sudah dilengkapi teks editor, compiler assembler, dan simulatornya. 8051 integrated Development Environment (IDE) sudah berbasis windows sebagai pengoperasiannya semudah aplikasi lain yang menggunakan GUI.

Langkah percobaan

1. Buka aplikasi win8051 IDE.
2. Kenali window yang tampil di layar.
3. Klik View dan buka jendela Registers, Control Register, Port, Internal Memory satu persatu, kemudian atur posisinya agar terlihat semua. Seperti pada Gambar di bawah ini;



4. Jendela-jendela yang dimunculkan tersebut menggambarkan keadaan internal dari Mikrokontroler 8051.

2. Simulasi dari Set Instruksi MCS51

A. Mode Pengalamatan dan Pemindahan Data

Langkah Percobaan

1. Buka aplikasi win8051 IDE.
2. Klik file kemudian New pada program win8051 IDE untuk membuka jendela editor baru
3. Ketik *source code* di bawah ini:

```
MOV A, #15           ; A diisi nilai konstan 15 desimal
MOV R1, A            ; isi Akumulator disalin ke register R1
MOV A, #0Ah          ; A diisi nilai konstan 0A heksa
MOV @R1, A           ; isi A disalin ke memori dengan lokasi
                     ; ditunjukkan oleh isi register R1
MOV P3, A            ; Isi A disalin ke Port 3
MOV P0, 0Fh          ; Isi memori dengan alamat 0Fh disalin
                     ; ke Port 0.
END
```

4. Setelah selesai klik simpan dan kemudian klik assemble. Setelah diassemble akan keluar jendela output yang berisi informasi apakah sintak yang dituliskan terdapat kesalahan atau tidak. Jika tidak ada pesan terjadi kesalahan maka file HEX sudah berhasil di generate jika ada pesan kesalahan maka lihat pada jendela output kesalahan apa yang ada pada sintak yang telah dituliskan.
5. Simulasikan sintak yang telah dibuat dengan mengklik menu Simulate, Start Simulator kemudian klik Step Into atau tekan tombol F11.
6. Perhatikan perubahan yang terjadi pada jendela register dan Internal memori untuk setiap penekanan F11 atau step program.

B. Operasi Aritmatika

Langkah Percobaan

1. Buka aplikasi win8051 IDE.
2. Klik file kemudian New pada program win8051 IDE untuk membuka jendela editor baru
3. Ketik *source code* di bawah ini:

```
MOV A, #12
MOV B, #7
ADD A, B
MOV 05h, A
INC A
MOV B, #20
MUL AB
MOV 06h, A
MOV 07h, B
MOV A, #12
MOV B, #7
DIV AB
MOV 08h, A
MOV 09h, B
INC A
DEC B
END
```

4. Klik simpan kemudian assemble. Setelah diassemble akan keluar jendela output yang berisi informasi apakah sintak yang dituliskan terdapat kesalahan atau tidak. Jika tidak ada pesan terjadi kesalahan maka file HEX sudah berhasil di generate jika ada pesan kesalahan maka lihat pada jendela output kesalahan apa yang ada pada sintak yang telah dituliskan
7. Simulasikan sintak yang telah dibuat dengan mengklik menu Simulate, Start Simulator kemudian klik Step Into atau tekan tombol F11.
5. Perhatikan perubahan yang terjadi pada jendela register dan Internal memori untuk setiap penekanan F11 atau step program.
6. Buatlah program untuk operasi $(5 \times 6) + 2(2+3) / 4$ hasil akhir disimpan di R0.

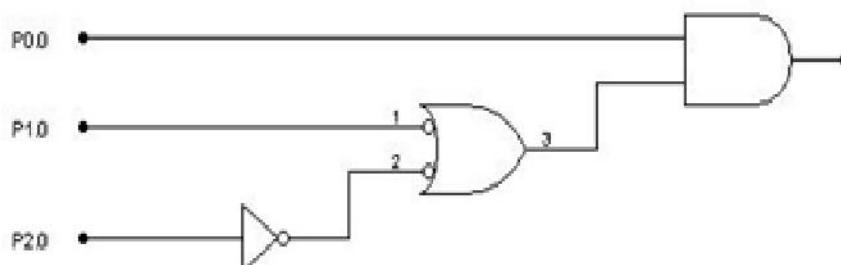
C. Operasi BIT

Langkah Percobaan

1. Buka aplikasi win8051 IDE.
2. Klik file kemudian New pada program win8051 IDE untuk membuka jendela editor baru
3. Ketik *source code* di bawah ini:

```
SETB    C
CLR     P1.7
MOV    P1.7,C
CPL    P0.0
ANL    C,P0.0
END
```

4. Klik simpan kemudian assemble. Setelah diassemble akan keluar jendela output yang berisi informasi apakah sintak yang dituliskan terdapat kesalahan atau tidak. Jika tidak ada pesan terjadi kesalahan maka file HEX sudah berhasil di generate jika ada pesan kesalahan maka lihat pada jendela output kesalahan apa yang ada pada sintak yang telah dituliskan
5. Simulasikan sintak yang telah dibuat dengan mengklik menu Simulate, Start Simulator kemudian klik Step Into atau tekan tombol F11.
6. Perhatikan perubahan yang terjadi pada jendela register dan Internal memori untuk setiap penekanan F11 atau step program
7. Implementasikan rangkaian digital di bawah ini secara software.



APLIKASI I/O PORT MCS51 UNTUK PENGGERAK LED DAN MASUKAN PUSH BUTTON

A. TUJUAN

1. Memahami aplikasi penggunaan port paralel untuk menjalankan LED
2. Memahami aplikasi penggunaan port paralel untuk membaca masukan lewat *push-button*
3. Mampu membuat program untuk menjalankan LED berdasarkan masukan *push-button*.

B. ALAT DAN BAHAN

1. DT-51 Low Cost Micro System Ver2.0
2. DT-51 Trainer Kit

C. PROSEDUR KERJA

1. Aplikasi menghidupkan dan mematikan kelompok LED secara bergantian

1. Buka aplikasi win8051 IDE.
2. Klik file kemudian New pada program win8051 IDE untuk membuka jendela editor baru
3. Ketik *source code* di bawah ini:

```

; -----
; Menghidupkan dan Mematikan Kelompok LED Secara Bergantian
; Menggunakan PORT 1.
;

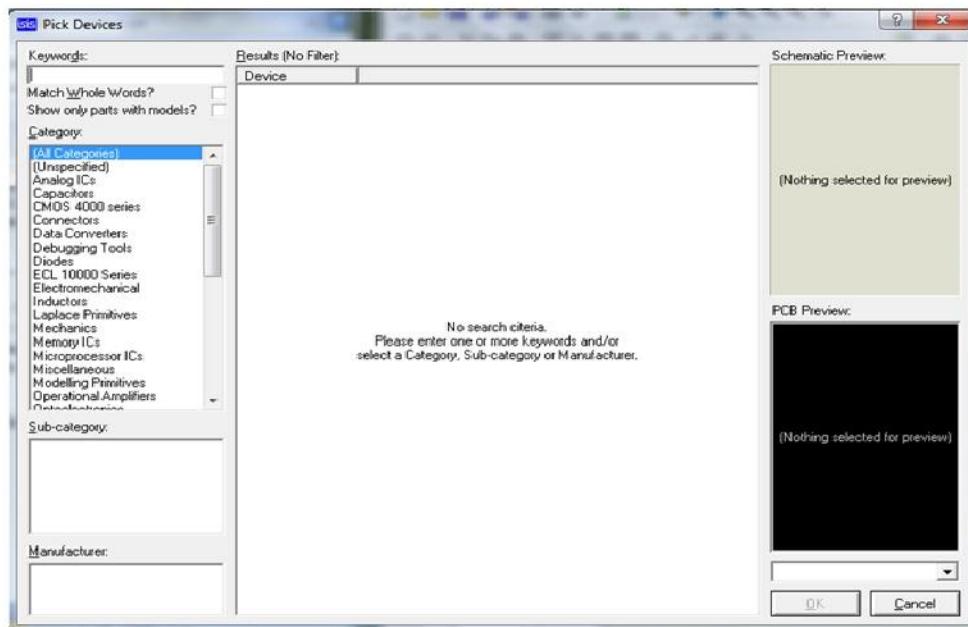
ORG 0H ; Program ditempatkan pada 00H
MULAI:
    MOV     P1,#00001111B ;LED di P1.4 - P1.7 nyala
    ACALL  DELAY          ;lakukan penundaan sesaat
    MOV     P1,#11110000B ; LED di P1.0 - P1.3 nyala
    ACALL  DELAY          ;lakukan penundaan sesaat
    SJMP   MULAI

; Subrutin Delay
DELAY:
    MOV     R0,#5H          ; Isi register R0 dengan 5d
DELAY1:
    MOV     R1,#0FFH         ; Isi register R1 dengan 255d
DELAY2:
    MOV     R2,#0FFH         ; Isi register R1 dengan 255d
    DJNZ   R2,$              ; R2=R2-1, Jika R2 belum = 0
                           ; ulangi lagi.
    DJNZ   R1,DELAY2        ; R1=R1-1 Jika R1 belum = 0
                           ; ulangi DELAY2.
    DJNZ   R0,DELAY1        ; R0=R0-1 Jika R0 belum = 0
                           ; ulangi DELAY1.
    RET                  ; Kembali ke pemanggil subrutin
                           ; DELAY
END
;

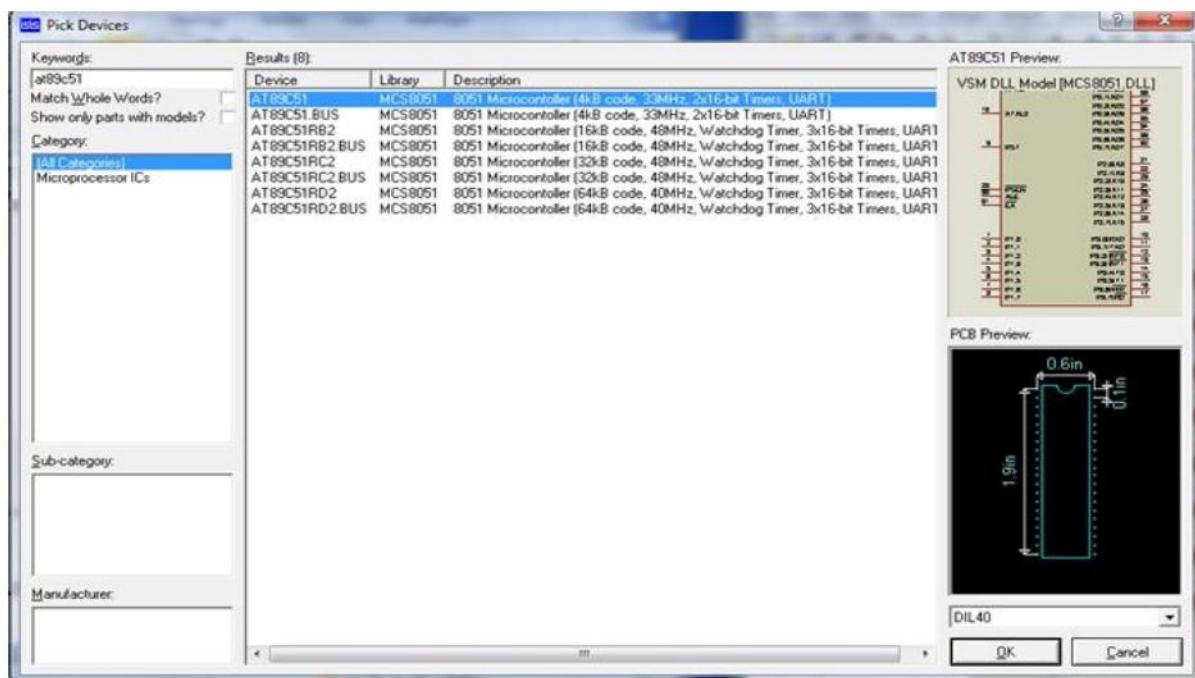
```

4. Setelah selesai klik simpan dan kemudian klik assemble. Setelah diassemble akan keluar jendela output yang berisi informasi apakah sintak yang dituliskan terdapat kesalahan atau tidak. Jika tidak ada pesan terjadi kesalahan maka file HEX sudah berhasil di generate jika ada pesan kesalahan maka lihat pada jendela output kesalahan apa yang ada pada sintak yang telah dituliskan.

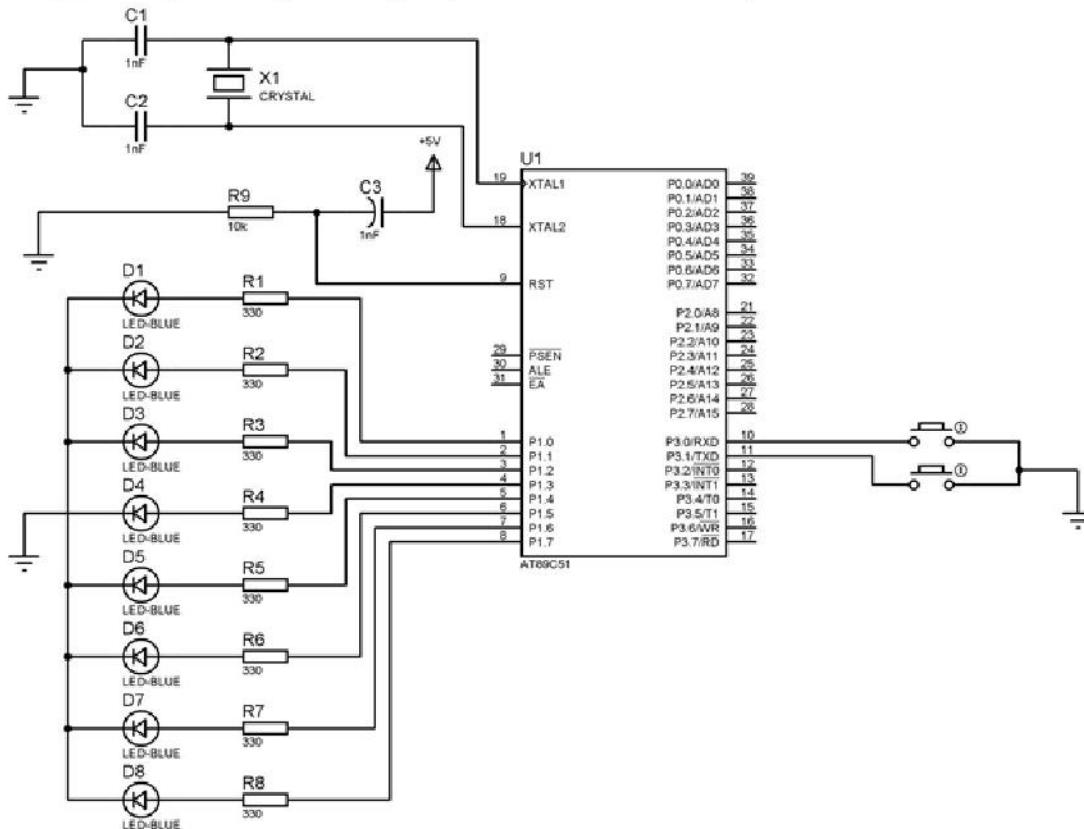
5. Buka aplikasi simulator ISIS 7 Profesional. Klik icon  , kemudian , setelah itu akan keluar jendela untuk mencari komponen seperti pada Gambar di bawah;



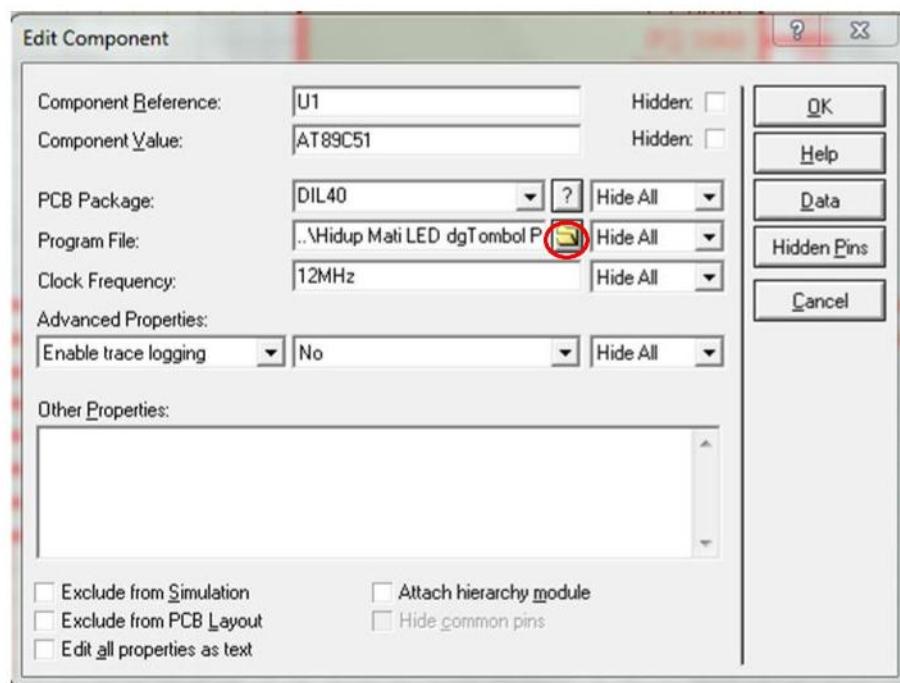
6. Ketikkan komponen AT89C51 pada kolom isian  , akan muncul pada jendela komponen seperti Gambar di bawah. Double klik komponen yang di blok berwarna biru maka akan otomatis masuk pada kolom *Devices* yang digunakan pada simulasi yang akan dirangkai.



7. Ketikkan kembali pada kolom di langkah 6 komponen; BUTTON, CAP, CAP-POL, CRYSTAL, LED-BLUE, RES. Lakukan prosedur yang sama seperti pada langkah 6.
8. Rangkai komponen seperti tampak pada Gambar di bawah;



9. Download file HEXA yang telah dibuat dengan cara *double klik* mikrokontroler AT89C51 maka akan muncul jendela seperti;



10. Klik bagian yang dilingkari, kemudian cari dan arahkan di tempat file HEXA disimpan.
11. Klik pada bagian pojok bawah kiri untuk mensimulasikan, amati LED apakah sudah sesuai dengan yang diharapkan? Jika belum, lihat dan evaluasi kembali sintak yang telah diketikkan.

2. Aplikasi menghidupkan dan mematikan dengan tombol push-button

1. Klik file kemudian New pada program win8051 IDE untuk membuka jendela editor baru
2. Tuliskan source code di bawah ini:

```

; -----
; Menghidupkan dan Mematikan LED Dengan Tombol Push-Bottom.
; -----
        ORG 0H                                ; Program ditempatkan pada 00H
MULAI:
        MOV     A, P3      ; Baca Status Port 3 dan simpan
                           ; isinya ke akumulator
        CJNE   A, #0FEH, TERUS ; Apakah isi akumulator
                           ; = 11111110B (P3.0 ditekan) ?
                           ; Tidak! Lompat ke TERUS
        MOV     P1, #0B      ; Ya! Hidupkan LED di Port 1
        SJMP   MULAI       ; Ulangi lagi dari awal (Label
                           ; MULAI)
TERUS:
        CJNE   A, #0FDH, MULAI ; Apakah Isi Akumulator
                           ; = 11111101B (P3.1 ditekan)?
                           ; Tidak! Lompat ke awal (Label
                           ; MULAI)
        MOV     P1, #0FFH    ; Ya! Matikan LED di Port 1
        SJMP   MULAI       ; Ulangi lagi dari awal ...
END
; -----

```

3. Setelah selesai klik simpan dan kemudian klik assemble. Setelah diassemble akan keluar jendela output yang berisi informasi apakah sintak yang dituliskan terdapat kesalahan atau tidak. Jika tidak ada pesan terjadi kesalahan maka file HEX sudah berhasil di generate jika ada pesan kesalahan maka lihat pada jendela output kesalahan apa yang ada pada sintak yang telah dituliskan.
4. Download file Hexa yang telah dibuat pada mikrokontroler AT89C51.
5. Simulasikan kembali rangkaian. Klik push-button pada rangkaian untuk mematikan dan menyalakan LED.

3. Aplikasi menghidupkan dan mematikan LED hanya dari satu tombol-tekan (P3.0) secara bergantian.

1. Tuliskan *source code* di bawah ini:

```

; -----
; Menghidupkan dan mematikan LED pada port 1 dengan cara
; menekan tombol pada P3.0 sebagai toggle switch.
; -----
        ORG 0H                                ; Program ditempatkan pada 00H
MULAI:
        MOV     A, P3      ; Baca Status Port 3 dan simpan
                           ; isinya ke akumulator
        CJNE   A, #0FEH, MULAI    ; Apakah tombol P3.0 ditekan?
                           ; = 1111 1110
                           ; Tidak! Ulangi lagi dari awal
        CJNE   R0, #0, TERUS    ; Ya! Apakah R0=0? (artinya lampu
                           ; sedang mati)?
                           ; Tidak! Loncat ke proses
                           ; mematikan lampu LED (TERUS)
                           ; = 11111110B (P3.0 ditekan) ?
                           ; Tidak! Lompat ke TERUS
        MOV     R0, #1      ; Ya! Ubah status R0=1 (LED Nyala)
        MOV     P1, #0      ; dan matikan LED di port 1
TUNGGU:
        MOV     A, P3      ; Untuk menghindari bunching
                           ; Tunggu hingga tombol P3.0
                           ; dilepas
        CJNE   A, #0FFH, TUNGGU
        SJMP   MULAI       ; Ulangi lagi dari awal
TERUS:
        MOV     R0, #0      ; ubah status R0=0 (lampu mati)
        MOV     P1, #0FFh    ; dan matikan LED di Port 1
        SJMP   TUNGGU      ; untuk menghindari bouching
                           ; lompat ke TUNGGU
        END
; -----

```

2. keluar jendela output yang berisi informasi apakah sintak yang dituliskan terdapat kesalahan atau tidak. Jika tidak ada pesan terjadi kesalahan maka file HEX sudah berhasil di generate jika ada pesan kesalahan maka lihat pada jendela output kesalahan apa yang ada pada sintak yang telah dituliskan.
3. Download file Hexa yang telah dibuat pada mikrokontroler AT89C51.
4. Simulasikan kembali rangkaian. Klik *push button* pada rangkaian untuk mematikan dan menyalakan LED.

5. Diskusikan bersama anggota kelompok;
 1. Port berapa yang digunakan sebagai port input dan port output?
 2. Apa perbedaan menyalakan dan mematikan LED pada sintak ke 2 dan ke 3?
 3. Tuliskan sintak yang membedakan cara mematikan dan menyalakan LED pada sintak ke 2 dan ke 3?
 4. Jika MSB pada port input pada percobaan ke 2 dan ke 3 bernilai LOW, apakah masih bisa menyalakan dan mematikan LED? Berikan penjelasan!
 5. Buat program LED berjalan yang dikendalikan oleh dua buah saklar push-button (A dan B) sebagai berikut;

A	B	Running LED
0	0	LED nyala dari tepi ke tengah
0	1	LED nyala dari kiri ke kanan
1	0	LED nyala dari kanan ke kiri
1	1	LED nyala dari tengah ke tepi

SEVEN SEGMENT DAN KEYPAD

A. TUJUAN

1. Memahami cara menyalakan *Seven Segment Common Cathode* dan *Common Anode*.
2. Memahami teknik Scanning display 2x7 dan 8x7-Segment
3. Mampu membuat program untuk menjalankan LED berdasarkan masukan *push-button*.

B. ALAT DAN BAHAN

1. DT-51 Low Cost Micro System Ver2.0
2. DT-51 Trainer Kit

C. PROSEDUR KERJA

1. Menampilkan karakter pada 2 x 7-segment dan 8 x 7-segment dengan teknik scanning

Langkah Percobaan;

1. Buka aplikasi win8051 IDE.
2. Klik file kemudian New pada program win8051 IDE untuk membuka jendela editor baru
3. Tuliskan *source code* di bawah ini:

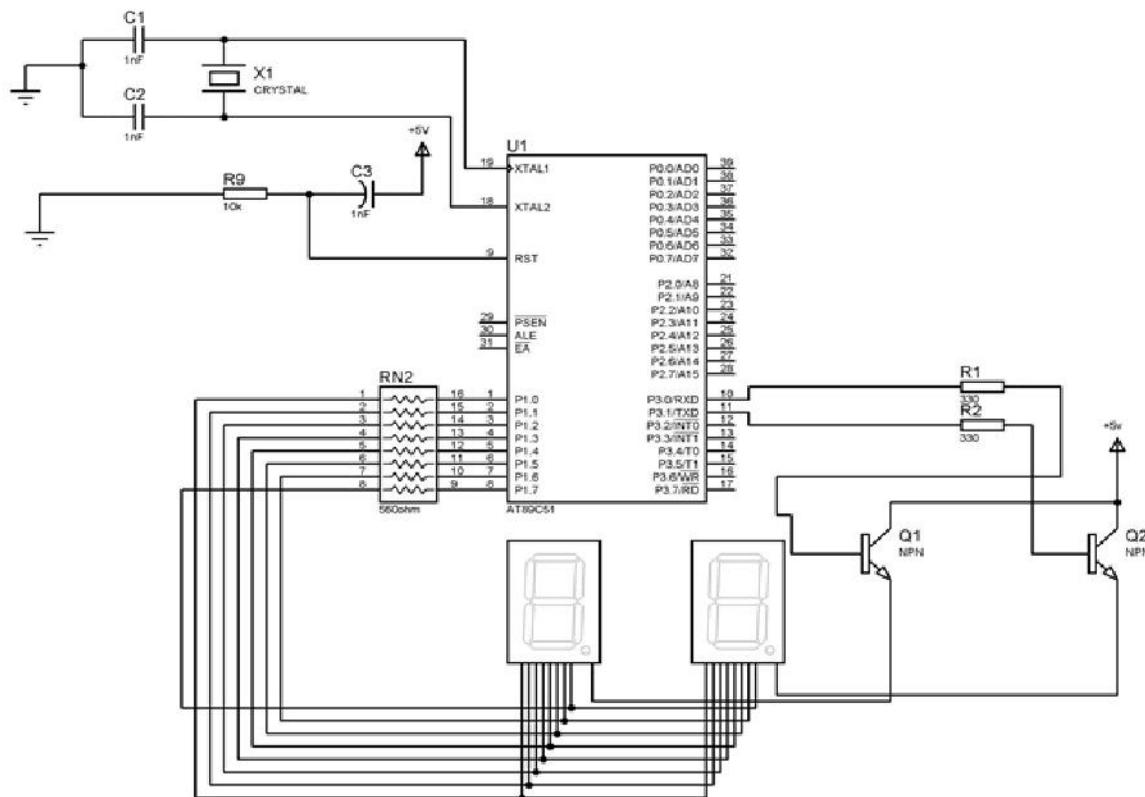
```
; Program untuk menghidupkan tampilan 2 x 7-segment
$mod51
        ORG      0H

MULAI:
        MOV DPTR, #Kamar
        MOV R6, #02H
        MOV R1, #02H
ULANG: CLR A
        MOVC A, @A+DPTR
        INC      DPTR
        MOV P1, A
        MOV A, R1
        MOV P3, A
        RR      A
        MOV R1, A
;MOV R2, #OFFH
;DELAY3: DJNZ R2, DELAY3
;        ACALL DELAY ;lakukan penundaan sesaat
;        MOV P1, #00H
;        ACALL DELAY ;lakukan penundaan sesaat
;        DJNZ R6, ULANG
;        JMP MULAI
DELAY:
        MOV R2, #02H ; Isi register R2 dengan 02h
DELAY1:
        MOV R3, #03H ; Isi register R3 dengan 03h
DELAY2:
```

```

MOV R4,#0FFH ; Isi register R4 dengan 0FFh
DJNZ R4,$ ; R4=R4-1, Jika R4 belum = 0
; ulangi lagi.
DJNZ R3,DELAY2 ; R3=R3-1 Jika R3 belum = 0
; ulangi DELAY2.
DJNZ R2,DELAY1 ; R2=R2-1 Jika R2 belum = 0
; ulangi DELAY1.
RET ; Kembali ke pemanggil subrutin
; DELAY
Kamar: DB 086H, 0DBH ; Database penyimpan nilai yang akan
; ditampilkan
END
;
;
```

4. Setelah selesai klik simpan dan kemudian klik assemble. Setelah diassemble akan keluar jendela output yang berisi informasi apakah sintak yang dituliskan terdapat kesalahan atau tidak. Jika tidak ada pesan terjadi kesalahan maka file HEX sudah berhasil di generate jika ada pesan kesalahan maka lihat pada jendela output kesalahan apa yang ada pada sintak yang telah dituliskan.
5. Rangkai komponen seperti tampak pada Gambar di bawah; Gunakan 7 segment *Common Cathode*.



6. Download file Hexa yang telah dibuat pada mikrokontroler AT89C51. Kemudian simulasi.
7. Modifikasi sintak di atas untuk 7 segment yang digunakan adalah *common anode*
8. Tuliskan *source code* di bawah ini:

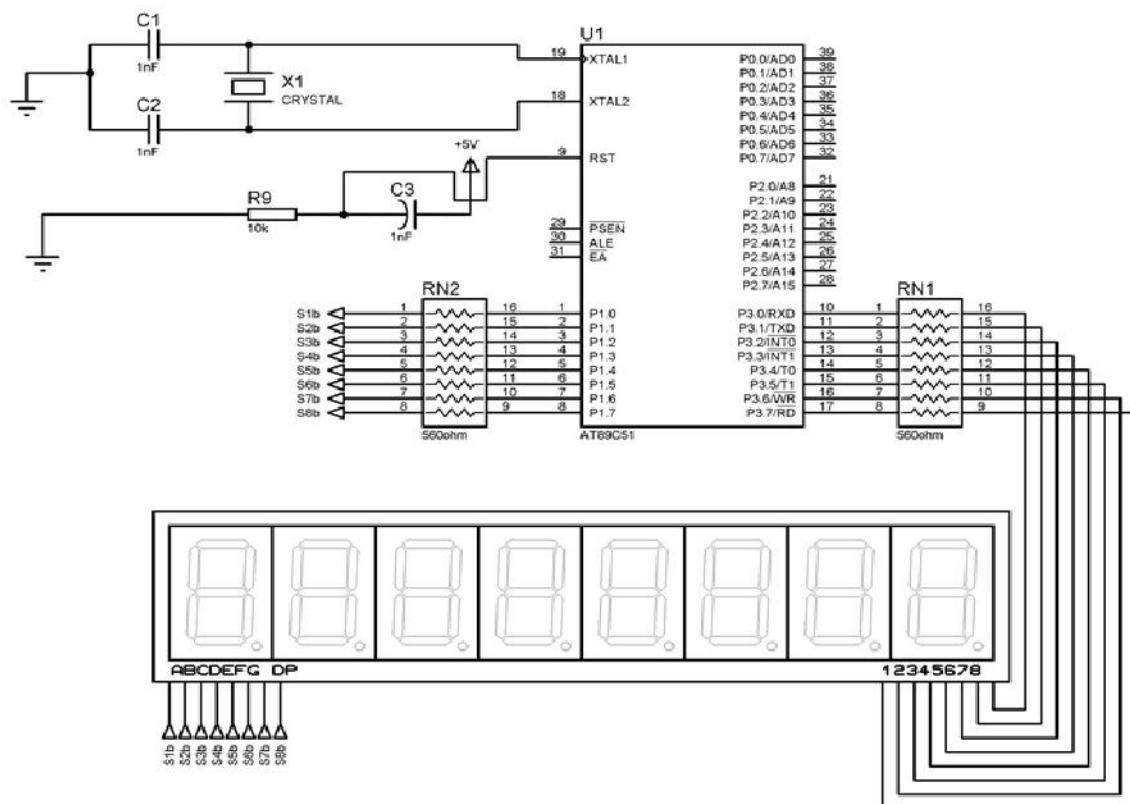
```
; Program untuk menghidupkan tampilan 8 x 7-segment
; untuk menampilkan tulisan 'n0'
$mod51
        ORG      0H

MULAI:
        MOV DPTR, #Kamar
        MOV R6, #08H
        MOV R1, #07FH
ULANG: CLR A
        MOVC A, @A+DPTR
        INC     DPTR
        MOV P1,A
        MOV A,R1
        MOV P3,A
        RR    A
        MOV R1,A
;MOV R2, #0FFH
;DELAY3: DJNZ R2, DELAY3
        ACALL DELAY ;lakukan penundaan sesaat
        MOV P1, #00H
        ACALL DELAY ;lakukan penundaan sesaat
        DJNZ R6, ULANG
        JMP MULAI

DELAY:
        MOV R2, #02H ; Isi register R0 dengan 5d
DELAY1:
        MOV R3, #03H ; Isi register R1 dengan 255d
DELAY2:
        MOV R4, #0FFH ; Isi register R1 dengan 255d
        DJNZ R4,$ ; R2=R2-1, Jika R2 belum = 0
        ; ulangi lagi.
        DJNZ R3,DELAY2 ; R1=R1-1 Jika R1 belum = 0
        ; ulangi DELAY2.
        DJNZ R2,DELAY1 ; R0=R0-1 Jika R0 belum = 0
        ; ulangi DELAY1.
        RET ; Kembali ke pemanggil subrutin
        ; DELAY

; Common Cathode
Kamar: DB 0BFH, 086H, 0DBH, 0CFH, 0E6H, 0EDH, 0FCH, 087H
;           OFFH, 0E7H,
        END
; -----
```

9. Rangkai komponen seperti tampak pada Gambar di bawah; Gunakan 7 segment *Common Cathode*.



10. Download file Hexa yang telah dibuat pada mikrokontroler AT89C51. Kemudian simulasikan
11. Modifikasi sintak di atas untuk 7 segment yang digunakan adalah *commont annode*

2. Sistem scanning keypad dan menampilkan pada 7-segment

Langkah Percobaan

1. Buka aplikasi win8051 IDE.
2. Klik file kemudian New pada program win8051 IDE untuk membuka jendela editor baru
3. Tuliskan *source code* di bawah ini:

```

; -----
; Program untuk membaca tombol-tombol yang disusun secara
; matrik, input scanner → P2.2, P2.0, P2.4 (K1, K2, K3) output
; scanner → P2.1, P2.6, P2.5, P2.3 (B1, B2, B3, B4) hasil
; ditampilkan pada 7-segment Commont Anode.
; -----
org 0h

satu:
mov p2,#11111110b
jb p2.4,dua
mov p1,#01111001b
ljmp satu

dua:
jb p2.5,tiga
mov p1,#00100100b

```

```

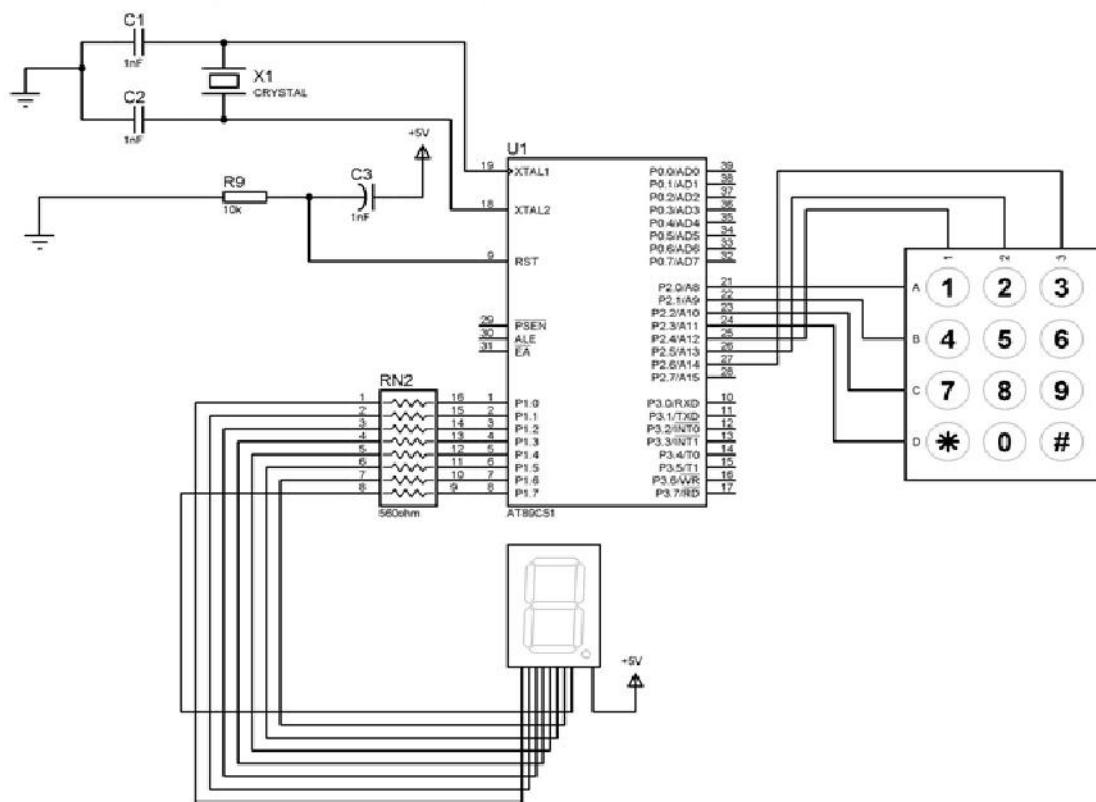
ljmp satu

tiga:
jb p2.6,satu
mov p1,#00110000b
ljmp satu

ok:
ljmp satu
end
; -----

```

4. Setelah selesai klik simpan dan kemudian klik assemble. Setelah diassemble akan keluar jendela output yang berisi informasi apakah sintak yang dituliskan terdapat kesalahan atau tidak. Jika tidak ada pesan terjadi kesalahan maka file HEX sudah berhasil di generate jika ada pesan kesalahan maka lihat pada jendela output kesalahan apa yang ada pada sintak yang telah dituliskan.
5. Rangkai komponen seperti tampak pada Gambar di bawah;



6. Download file Hexa yang telah dibuat pada mikrokontroler AT89C51. Kemudian simulasiakan
7. Modifikasi sintak di atas supaya tombol 4, 5, 6, 7, 8, 9, *, 0 dan # aktif.

KENDALI MOTOR DC, MOTOR STEPPER DAN MOTOR SERVO

A. TUJUAN

1. Memahami prinsip kerja Motor DC, Motor Stepper dan Motor Servo.
2. Memahami teknik pengendalian Motor DC, Motor Stepper dan Motor Servo secara manual.
3. Memahami teknik pengendalian Motor DC dan Motor Stepper menggunakan mikroprosesor.
4. Mampu mengaplikasikan pengendalian Motor DC dan Motor Stepper pada berbagai aplikasi.

B. ALAT DAN BAHAN

1. DT-51 Low Cost Micro System Ver2.0
2. DT-51 Trainer Kit
3. Driver motor DC dan motor stepper
4. Laptop atau PC yang sudah diinstal program proteus dan 8051IDE

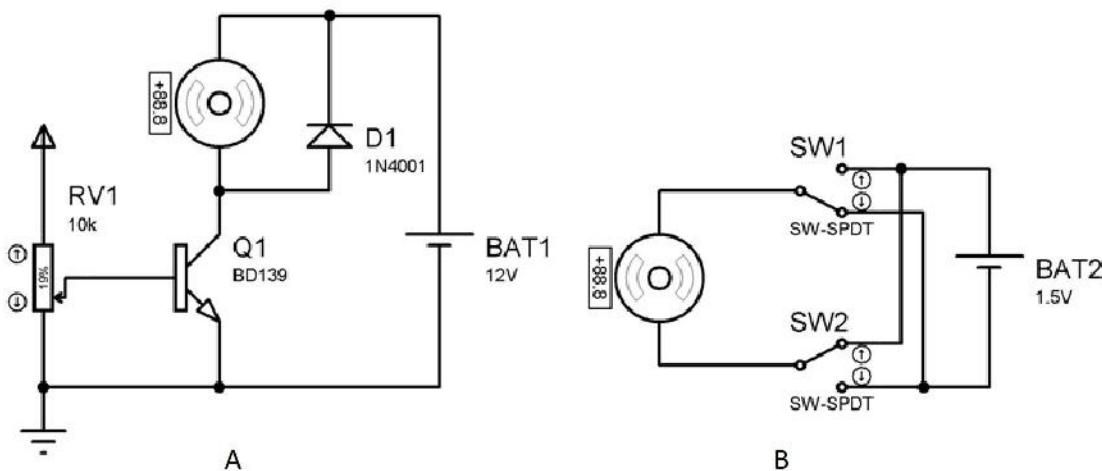
C. PROSEDUR KERJA

1. Memperlajari Cara kerja motor DC

D. Kendali manual motor DC

Langkah Percobaan

1. Buka aplikasi proteus.
2. Susun rangkaian pada aplikasi proteus seperti pada gambar di bawah ini;



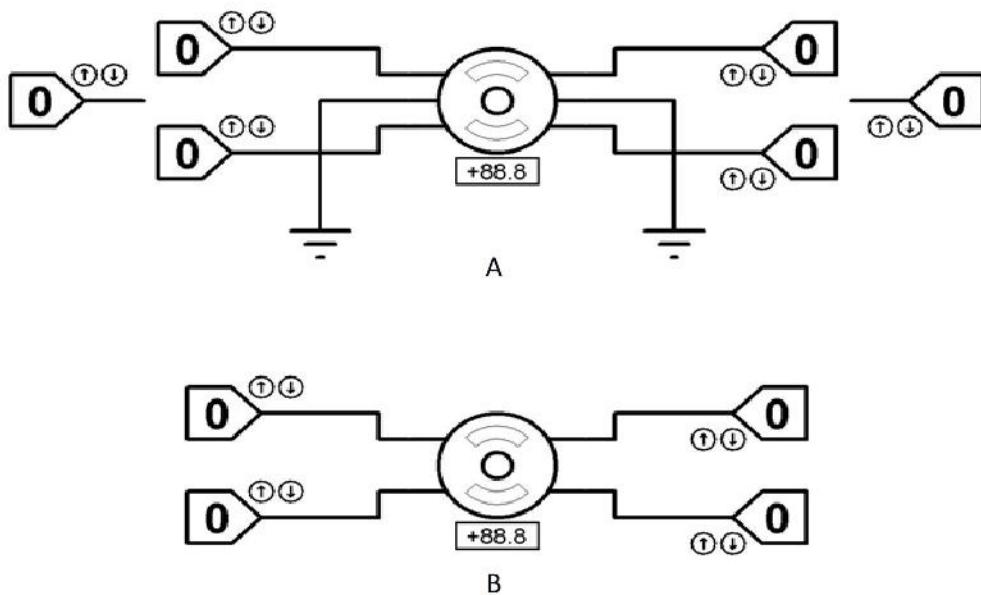
3. Simulasikan rangkaian.
4. Atur kecepatan motor pada rangkaian A dengan merubah potensiometer (RV1), amati perubahan kecepatan rotasi motor.
5. Rubah arah putar motor pada rangkaian B dengan cara memindahkan switch 1 dan 2.
6. Diskusikan dengan teman satu kelompok pertanyaan di bawah ini;
 - a. Bagaimana hubungan antara nilai hambatan RV1 dengan kecepatan motor?

- b. Saat motor DC berputar maksimum, langsung turunkan nilai persen hambatan RV1 menjadi nol. Apakah motor langsung berhenti? Berikan penjelasan hasil pengamatan anda!
- c. Berikan penjelasan, kenapa dengan memindahkan switch pada rangkaian B, arah putar motor dapat berubah.
- d. Berikan penjelasan, bagaimana cara untuk mengatur kecepatan motor DC dan bagaimana pula cara untuk merubah arah pergerakan motor DC!
- e. Tanda (-) dan (+) pada motor DC menunjukkan apa?

E. Kendali manual motor Stepper

Langkah Percobaan

1. Buka aplikasi proteus. Cari komponen POT-HG, BD139, IN4001, CELL, LOGICSTATE, MOTOR-DC, MOTOR-BISTEPPER, MOTOR-STEPPER, SW-SPDT
2. Susun rangkaian pada aplikasi proteus seperti pada gambar di bawah ini;

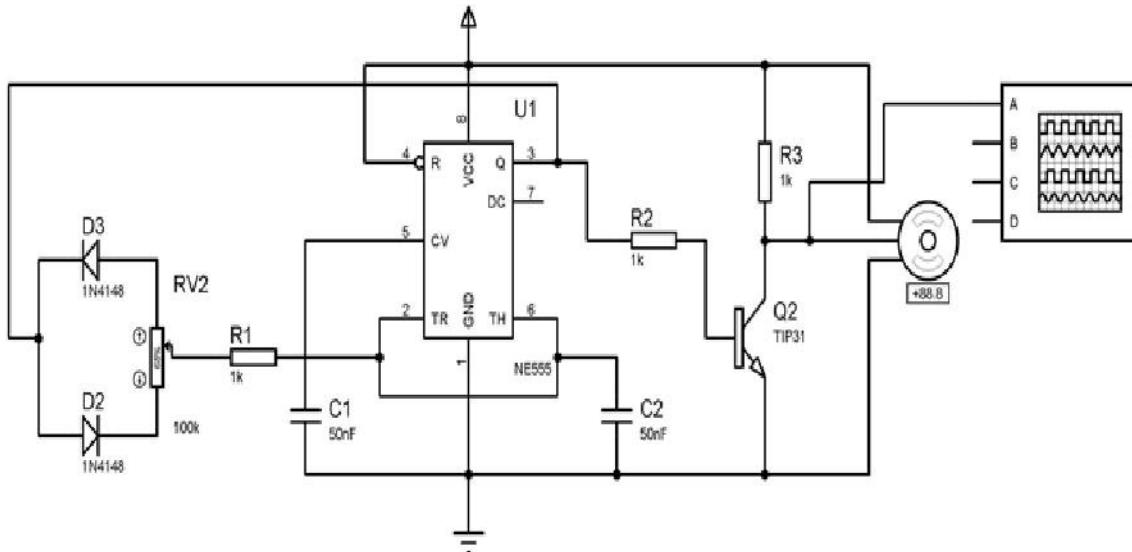


3. Simulasikan rangkaian.
4. Rubah logika LogicState dan amati arah perubahan dan besar sudut putaran motor.
5. Diskusikan dengan teman satu kelompok pertanyaan di bawah ini;
 - a. Nilai yang ditunjukkan pada motor menunjukkan nilai apa?
 - b. Tanda (-) dan (+) pada nilai yang ditunjukkan di motor artinya apa?
 - c. Jika perubahan logika LogicState dari kiri ke kanan maka motor berputar ke arah?
 - d. Jika perubahan logika LogicState dari kanan ke kiri maka motor berputar ke arah?
 - e. Berikan penjelasan, bagaimana cara untuk mengatur kecepatan motor Stepper dan bagaimana pula cara untuk merubah arah pergerakan motor Stepper!

F. Kendali manual motor Servo

Langkah Percobaan

1. Buka aplikasi proteus. Cari komponen POT-HG, BD139, IN4148, NE555, MOTOR-PWMSERVO, OSCILOSCOP.
2. Susun rangkaian pada aplikasi proteus seperti pada gambar di bawah ini;



3. Simulasikan rangkaian.
4. Rubah nilai persen pada potensiometer RV2 dan amati perubahan bentuk sinyal pada osiloskop serta amati juga perubahan nilai pada motor servo.
5. Diskusikan dengan teman satu kelompok pertanyaan di bawah ini;
 - a. Saat sinyal kotak dengan lebar HIGH berapa sekon nilai pada motor servo + 90?
 - b. Saat sinyal kotak dengan lebar HIGH berapa sekon nilai pada motor servo - 90?
 - c. Apa yang terjadi saat lebar sinyal HIGH lebih besar atau kurang dari batas saat nilai servo +90?
 - d. Apa yang terjadi saat lebar sinyal HIGH lebih besar atau kurang dari batas saat nilai servo -90?
 - e. Berikan kesimpulan saudara dari hasil pengamatan “a” sampai “d” terkait dengan menggerakkan motor servo!

2. Kendali menggunakan mikrokontroler

A. Kendali manual motor DC menggunakan mikrokontroler

Langkah Percobaan

1. Buka aplikasi proteus.
2. Ketik sintak di bawah ini;

```

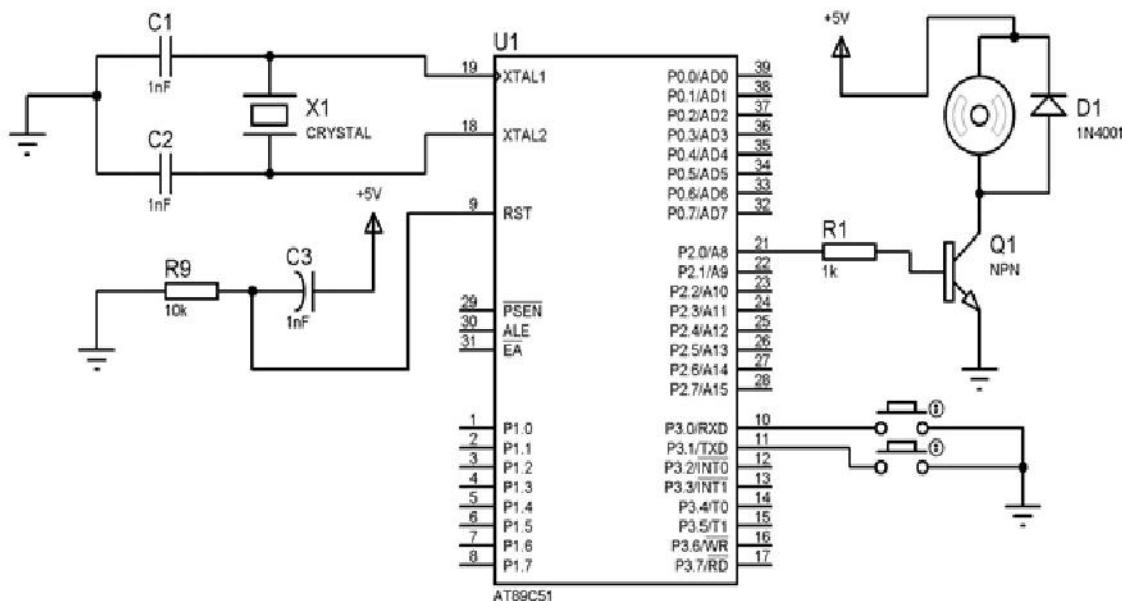
; -----
org 0h
MULAI:
    MOV     A, P3          ; Baca Port 3 dan simpan ke
                        ; akumulator
    CJNE   A, #0FEH, TERUS ; apakah P3.0 ditekan?
                        ; Tidak! Lompat ke terus
    MOV     P2, #01H         ; Ya! Hidupkan Motor DC di Port2.0
    SJMP   MULAI           ; Ulangi lagi dari awal
                        ; (label MULAI)

TERUS:
    CJNE   A, #0FDH, MULAI ; Apakah P3.1 ditekan?
                        ; Tidak! Lompat ke awal
                        ; (label MULAI)
    MOV     P2, #0H          ; Ya! Matikan Motor DC di Port2.0
    SJMP   MULAI           ; Ulangi lagi dari awal.

END
; -----
```

3. Setelah selesai klik simpan dan kemudian klik assemble. Setelah diassemble akan keluar jendela output yang berisi informasi apakah sintak yang dituliskan terdapat kesalahan atau tidak. Jika tidak ada pesan terjadi kesalahan maka file HEX sudah berhasil di generate jika ada pesan kesalahan maka lihat pada jendela output kesalahan apa yang ada pada sintak yang telah dituliskan.

4. Susun rangkaian pada aplikasi proteus seperti pada gambar di bawah ini;



5. Download file Hexa yang telah dibuat pada mikrokontroler AT89C51.
6. Simulasikan rangkaian dan amati apa yang terjadi.
7. Diskusikan bersama anggota kelompok;
8. Diskusikan dengan teman satu kelompok pertanyaan di bawah ini;
 - a. Port berapa yang digunakan sebagai pengontrol untuk mematikan dan menyalakan Motor DC?
 - b. Sintak yang mana pada contoh di sebelumnya yang berfungsi sebagai pemilihan tombol mana yang ditekan?
 - c. Jika port yang digunakan untuk menyalakan dan mematikan Motor DC diganti menjadi Port 1.5 dan Port 1.6 maka sintak manakah yang harus dirubah?
 - d. Buatlah program pengendali motor DC dengan 3 tingkat kecepatan yang diatur melalui 3 tombol, misal tombol 1 untuk putaran cepat, tombol 2 untuk putaran sedang dan tombol 3 untuk putaran pelan.

B. Kendali motor Stepper menggunakan mikrokontroler

Langkah Percobaan

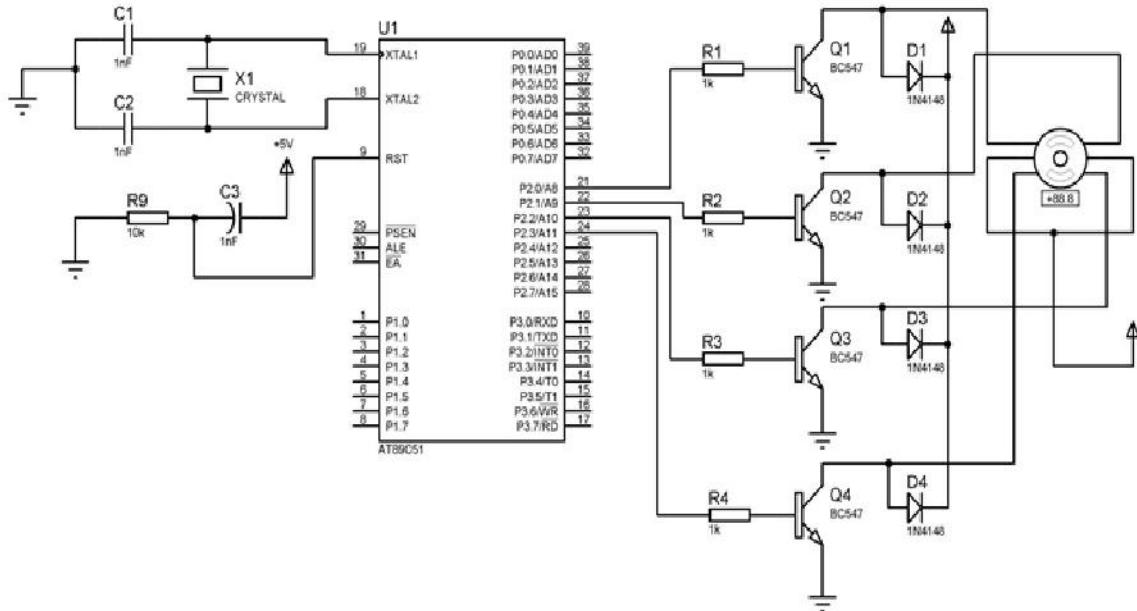
1. Buka aplikasi proteus.
2. Ketik sintak di bawah ini;

```
; -----  
org 0h  
MULAI:  
    MOV    P2,#00000111B  
    ACALL DELAY  
    MOV    P2,#00000011B  
    ACALL DELAY  
    MOV    P2,#00001011B  
    ACALL DELAY  
    MOV    P2,#00001001B  
    ACALL DELAY  
    MOV    P2,#00001101B  
    ACALL DELAY  
    MOV    P2,#00001100B  
    ACALL DELAY  
    MOV    P2,#00001110B  
    ACALL DELAY  
    MOV    P2,#00000110B  
    ACALL DELAY  
    SJMP  MULAI  
  
DELAY:   MOV R0,#1H  
DELAY1:  MOV R1,#050H  
DELAY2:  MOV R2,#0H  
        DJNZ R2,$  
        DJNZ R1,DELAY2  
        DJNZ R0,DELAY1  
        RET  
  
END  
; -----
```

3. Setelah selesai klik simpan dan kemudian klik assemble. Setelah diassemble akan keluar jendela output yang berisi informasi apakah sintak yang dituliskan terdapat kesalahan atau

tidak. Jika tidak ada pesan terjadi kesalahan maka file HEX sudah berhasil di generate jika ada pesan kesalahan maka lihat pada jendela output kesalahan apa yang ada pada sintak yang telah dituliskan.

4. Susun rangkaian pada aplikasi proteus seperti pada gambar di bawah ini;



5. Download file Hexa yang telah dibuat pada mikrokontroler AT89C51.
6. Simulasikan rangkaian dan amati apa yang terjadi.
7. Diskusikan bersama anggota kelompok;
8. Diskusikan dengan teman satu kelompok pertanyaan di bawah ini;
 - a. Apa fungsi transistor?
 - b. Rubah nilai delay pada sintak dan amati yang terjadi! Berikan penjelasan saudara tentang hubungan antara lama delay dengan yang terjadi pada motor stepper!
 - c. Buatlah program pengendali motor Stepper dengan 2 tombol, tombol 1 pergerakan ke kanan, tombol 2 pergerakan ke kiri.

ADC DAN LCD DISPLAY (ARDUINO)

A. TUJUAN

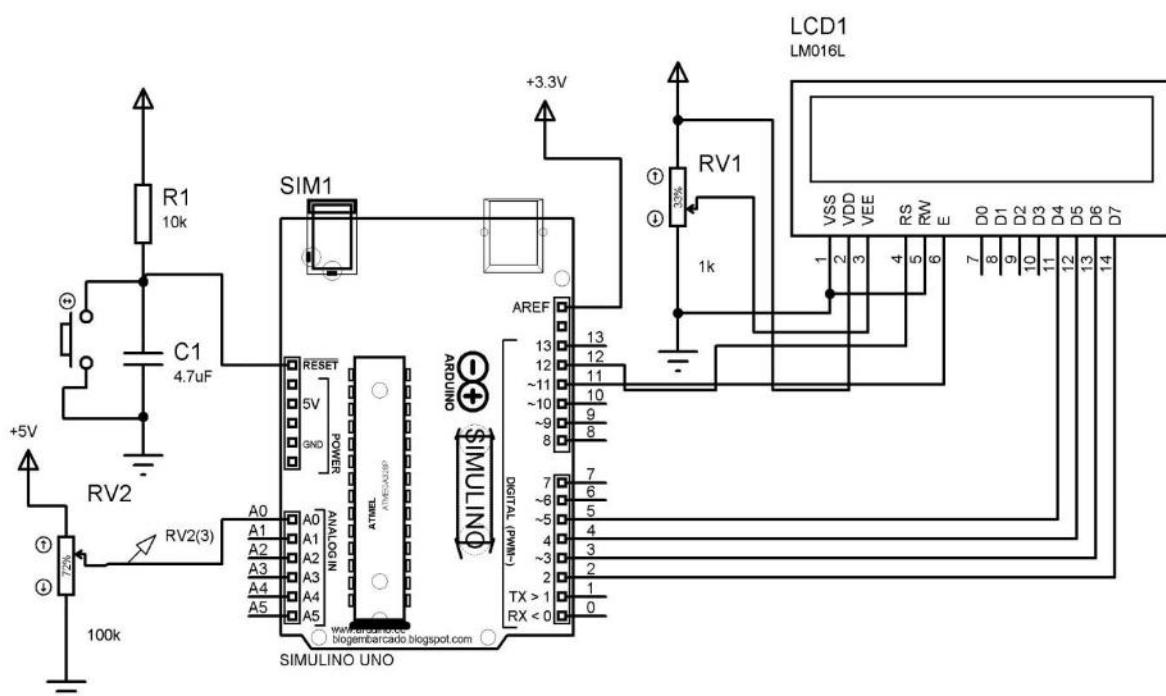
1. Mahasiswa dapat menampilkan karakter pada LCD display.
2. Mahasiswa dapat memanipulasi tampilan pada LCD display
3. Mahasiswa dapat memahami konsep konversi data analog ke digital pada mikrokontroler
4. Mahasiswa dapat mengatur tampilan karakter pada LCD sesuai dengan nilai analog yang dibaca oleh mikrokontroler

B. ALAT DAN BAHAN

1. Board mikrokontroler arduino uno
2. LCD 16x2
3. Laptop atau PC yang sudah diinstal program proteus dan Aduino IDE

C. PROSEDUR KERJA

Rangkaian yang digunakan pada percobaan ini adalah;



1. Menampilkan karakter di LCD

A. Menampilkan dan menghapus karakter pada LCD

Langkah Percobaan:

1. Buka aplikasi Arduino IDE. Ketikkan sintak di bawah ini;

```
/*LiquidCrystal Library
Sintak ini akan menampilkan tulisan "Hello World!" pada LCD.
Rangkaian:
```

```
* pin RS LCD ke pin digital 12
* pin Enable LCD ke pin digital 11
* pin D4 LCD ke pin digital 5
* pin D5 LCD ke pin digital 4
* pin D6 LCD ke pin digital 3
* pin D7 LCD ke pin digital 2
* pin R/W LCD ke ground
* 10K resistor:
* ends to +5V and ground
* wiper to LCD VO pin (pin 3)
*/
// Memanggil Library LiquidCrystal:
#include <LiquidCrystal.h>

// Inisialisasi pin Arduino yang terhubung ke LCD.
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

void setup() {
    // Pendefinisian jumlah kolom dan baris LCD yang digunakan:
    lcd.begin(16, 2);
    // Perintah menuliskan karakter pada LCD.
    lcd.print("hello, world!");
}

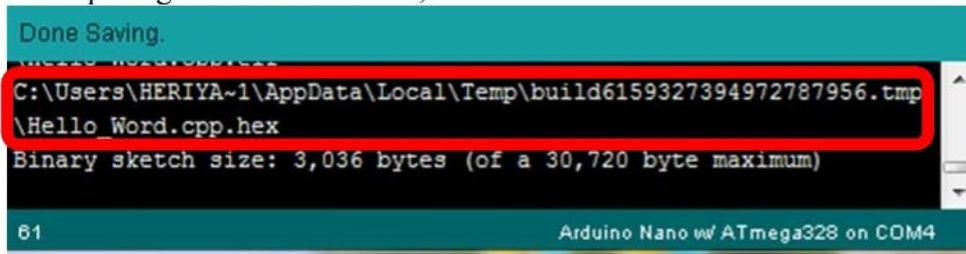
void loop() {

    lcd.noDisplay(); // Mematikan Display LCD:
    delay(2000); // Waktu Tunda selama 2 second
    lcd.display(); // Menghidupkan Display LCD:
    delay(2000);
    lcd.blink(); // Turn on the blinking cursor:
    delay(2000);
    lcd.noBlink(); // Turn off the blinking cursor:
    delay(2000);
    lcd.cursor(); // Turn on the cursor:
    delay(2000);
    lcd.noCursor(); // Turn off the cursor:
    delay(2000);

    lcd.setCursor(0, 1); // Mulai mencetak pada Kolom ke 0,
    // baris ke 1
    lcd.print(millis()/1000); // Cetak jumlah detik pada LCD di
    // posisi kolom dan baris sesuai
    // dengan setCursor

    if (millis()/1000 > 25){ // jika nilai millis()/1000 > 25 maka
        // hapus karakter pada LCD
        // Kemudian cetak kembali karakter
        // "Hello Word 2".
        lcd.clear(); // Hapus karakter pada LCD
        delay(2000);
        lcd.print("Hello Word 2");
    }
}
```

2. Klik File, Preferences, akan muncul jendela preferences. Centrang compilation dan upload untuk melihat direktori file heksa yang telah degenerate.
3. Klik  pada arduino IDE untuk membuat file heksa. Copy alamat direktori file heksa seperti contoh pada gambar di bawah ini;



4. Paste alamat yang telah dicopy tadi ke arduino pada simulator.\
5. Simulasikan dan amati yang terjadi pada layar LCD untuk memahami sintak yang telah diketikkan.

B. Menggeser (Scroll) karakter pada LCD

Langkah Percobaan:

1. Buka aplikasi Arduino IDE. Ketikkan sintak di bawah ini;

```
/*LiquidCrystal Library - Hello World
```

Sintak ini akan menampilkan tulisan "Hello World!" pada LCD.

Rangkaian:

```
* pin RS LCD ke pin digital 12
* pin Enable LCD ke pin digital 11
* pin D4 LCD ke pin digital 5
* pin D5 LCD ke pin digital 4
* pin D6 LCD ke pin digital 3
* pin D7 LCD ke pin digital 2
* pin R/W LCD ke ground
* 10K resistor:
* ends to +5V and ground
* wiper to LCD VO pin (pin 3)
```

```
*/
```

```
// include the library code:
#include <LiquidCrystal.h>

// initialize the library with the numbers of the interface pins
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

void setup() {
    lcd.begin(16, 2); // set up the LCD's number of
columns and rows:
    lcd.setCursor(0,0);
    lcd.print("hello, world!"); // Print a message to the LCD.
    lcd.setCursor(0,1);
```

```
lcd.print("Hello Tekom"); // Print a message to the LCD.  
delay(2000); // Waktu Tunda selama 2 second  
}  
  
void loop() {  
    // Geser ke kiri sebanyak 13 kali (sesuai panjang karakter yang ada)  
    // sampai batas display LCD sebelah kiri  
    for (int positionCounter = 0; positionCounter < 13; positionCounter++) {  
        lcd.scrollDisplayLeft(); // Geser ke kiri satu kali  
        delay(150); // Tunggu 0.15 detik:  
    }  
  
    // Geser ke kanan sebanyak 29 kali (panjang karakter + panjang display LCD/kolom)  
    // sampai batas display LCD sebelah kanan  
    for (int positionCounter = 0; positionCounter < 29; positionCounter++) {  
        lcd.scrollDisplayRight(); // Geser ke kanan satu kali  
        delay(150); // Tunggu 0.15 detik:  
    }  
    // Geser ke kiri sebanyak 16 kali (panjang karakter + panjang display LCD/kolom)  
    // sampai karakter kembali ke ujung kiri LCD  
    for (int positionCounter = 0; positionCounter < 16; positionCounter++) {  
        // Geser ke kiri satu kali  
        lcd.scrollDisplayLeft();  
        delay(150); // Tunggu 0.15 detik:  
    }  
  
    delay(1000); // Tunggu 1 detik sebelum kembali ke looping awal  
}
```

2. Compile sintaknya, kemudian Paste alamat yang telah dicopy tadi ke arduino pada simulator.
3. Simulasikan dan amati yang terjadi pada layar LCD untuk memahami sintak yang telah diketikkan.

C. Membuat karakter tertentu pada LCD

Langkah Percobaan:

1. Buka aplikasi Arduino IDE. Ketikkan sintak di bawah ini;

```
// include the library code:  
#include <LiquidCrystal.h>  
  
// initialize the library with the numbers of the interface pins  
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);  
  
// make some custom characters:  
byte heart[8] = {  
    0b00000,  
    0b01010,  
    0b11111,  
    0b11111,  
    0b11111,  
    0b01110,  
    0b00100,  
    0b00000  
};  
  
byte smiley[8] = {  
    0b00000,  
    0b00000,  
    0b01010,  
    0b00000,  
    0b00000,  
    0b10001,  
    0b01110,  
    0b00000  
};  
  
byte frownie[8] = {  
    0b00000,  
    0b00000,  
    0b01010,  
    0b00000,  
    0b00000,  
    0b00000,  
    0b01110,  
    0b10001  
};  
  
byte armsDown[8] = {  
    0b00100,  
    0b01010,  
    0b00100,  
    0b00100,  
    0b01110,  
    0b10101,  
    0b00100,
```

```
0b01010
};

byte armsUp[8] = {
  0b00100,
  0b01010,
  0b00100,
  0b10101,
  0b01110,
  0b00100,
  0b00100,
  0b01010
};
void setup() {
  lcd.createChar(1, heart); // create a new character
  lcd.createChar(2, smiley); // create a new character
  lcd.createChar(3, frownie); // create a new character
  lcd.createChar(4, armsDown); // create a new character
  lcd.createChar(5, armsUp); // create a new character

  // set up the lcd's number of columns and rows:
  lcd.begin(16, 2);
  lcd.print("Tekom ");
  lcd.print(" IPB ");
  lcd.write(1);

}

void loop() {
  lcd.setCursor(4, 1);
  lcd.write(1); // draw the heart:
  delay(1000);
  lcd.write(2); // draw the smiley:
  delay(1000);
  lcd.write(3); // draw the frownie:
  delay(1000);
  lcd.write(4); // draw armsDown:
  delay(1000);
  lcd.write(5); // draw him arms up:
  delay(1000);
}
```

2. Compile sintaknya, kemudian Paste alamat yang telah dicopy tadi ke arduino pada simulator.
3. Simulasikan dan amati yang terjadi pada layar LCD untuk memahami sintak yang telah diketikkan.
4. Buat nama kelompok masing-masing dan tampilkan di LCD dengan cara bergeser ke arah kiri!

2. Analog to Digital Arduino

Langkah Percobaan:

1. Buka aplikasi Arduino IDE. Ketikkan sintak di bawah ini;

```
/* Analog Input
Pembacaan nilai analog pada mikrokontroler Arduino dapat dilakukan
pada pin A0 sampai A5.
Pada Arduino ATmega328 memiliki resolusi 10 bit (1024 kondisi).
Resolusi pembacaan analog dapat diatur dengan menggunakan mode
tegangan referensi;
- DEFAULT: tegangan referensi 5 Volt (pada arduino 5 V) atau 3.3
Volt (pada arduino 3.3 V)
- INTERNAL: referensi internalnya, 1.1 Volt pada ATmega168 dan
ATmega328, 2.56 Volt pada ATmeg8
- INTERNAL1V1: referensi internal 1.1 Volt pada arduino Mega
- INTERNAL2V56: referensi internal pada arduino Mega.
- EXTERNAL: tegangan referensi yang digunakan pada inputan luar
(pin AREF).
Resolusi = Vreferensi/1023; jika Vreferensi 5 V maka resolusi 4.89
mV
```

Untuk menggunakan tegangan referensi digunakan sintak
"analogReference(type)"

Untuk membaca nilai analog digunakan sintak "analogRead(pin)"

pada percobaan ini hasil pembacaan akan ditampilkan di LCD 16x2.
*/

```
// include the library code:
#include <LiquidCrystal.h>
int sensorPin = A0;      // Pin analog yang digunakan adalah A0
int sensorValue = 0;     // Nilai awal pembacaan input Analog bentuk
digital
float Volt = 0;          // Nilai awal pembacaan Analog setelah dikonversi
// initialize the library with the numbers of the interface pins
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

void setup() {
    lcd.begin(16, 2);    // set up the LCD's number of columns and
rows:
    lcd.print("Nilai Tegangan"); // Print a message to the LCD.
    delay(2000);           // Waktu Tunda selama 2 second
}

void loop() {

    analogReference(DEFAULT);
    sensorValue = analogRead(sensorPin);
    Volt = sensorValue * 5.0/1023.0;
    lcd.clear();
    lcd.setCursor(0, 0); // cetak nilai pada baris ke 0 kolom ke 0.
    lcd.print("DEFAULT");
```

```
lcd.setCursor(0, 1); // cetak nilai pada baris ke 1 kolom ke 0.  
lcd.print(sensorValue);  
lcd.setCursor(6, 1); // cetak nilai pada baris ke 1 kolom ke 6.  
lcd.print(Volt,4); // Cetak Nilai analog yang terbaca  
dengan 4 angka dibelakang koma  
delay(2000);  
  
analogReference(INTERNAL);  
sensorValue = analogRead(sensorPin);  
Volt = sensorValue * 1.1/1023.0;  
lcd.clear();  
lcd.setCursor(0, 0); // cetak nilai pada baris ke 0 kolom ke 0.  
lcd.print("INTERNAL");  
lcd.setCursor(0, 1); // cetak nilai pada baris ke 1 kolom ke 0.  
lcd.print(sensorValue);  
lcd.setCursor(6, 1); // cetak nilai pada baris ke 1 kolom ke 6.  
lcd.print(Volt,4); // Cetak Nilai analog yang terbaca  
dengan 4 angka dibelakang koma  
delay(2000);  
  
analogReference(EXTERNAL);  
sensorValue = analogRead(sensorPin);  
Volt = sensorValue * 3.3/1023.0;  
lcd.clear();  
lcd.setCursor(0, 0); // cetak nilai pada baris ke 0 kolom ke 0.  
lcd.print("EXTERNAL");  
lcd.setCursor(0, 1); // cetak nilai pada baris ke 1 kolom ke 0.  
lcd.print(sensorValue);  
lcd.setCursor(6, 1); // cetak nilai pada baris ke 1 kolom ke 6.  
lcd.print(Volt,4); // Cetak Nilai analog yang terbaca  
dengan 4 angka dibelakang koma  
delay(2000);  
}
```

2. Compile sintaknya, kemudian Paste alamat yang telah dicopy tadi ke arduino pada simulator.
3. Simulasikan dan amati yang terjadi pada layar LCD untuk memahami sintak yang telah diketikkan. Rubahlah nilai hambatan potensio.
4. Rubah nilai tegangan input potensio menjadi 0,5 Volt. Simulasikan kembali dan amati apa yang terjadi pada kemampuan pembacaan nilai tegangan untuk mesing-masing mode tegangan referensi.
5. Diskusikan dengan teman kelompok:
 - a. Carilah nilai tegangan terkecil dan terbesar yang masih bisa terbaca oleh mikrokontroler untuk masing-masing mode referensi.
 - b. Berikan penjelasan terhadap hasil yang didapatkan pada ‘a’!
 - c. Buat sintak yang bisa menampilkan nama anggota kelompok. Kemunculan nama anggota diatur oleh tegangan input analog. Gunakan tegangan masukan pada potensio sebesar 5 Volt.

KOMUNIKASI SERIAL (ARDUINO) POTENSIOMETER DAN SENSOR PING

A. TUJUAN

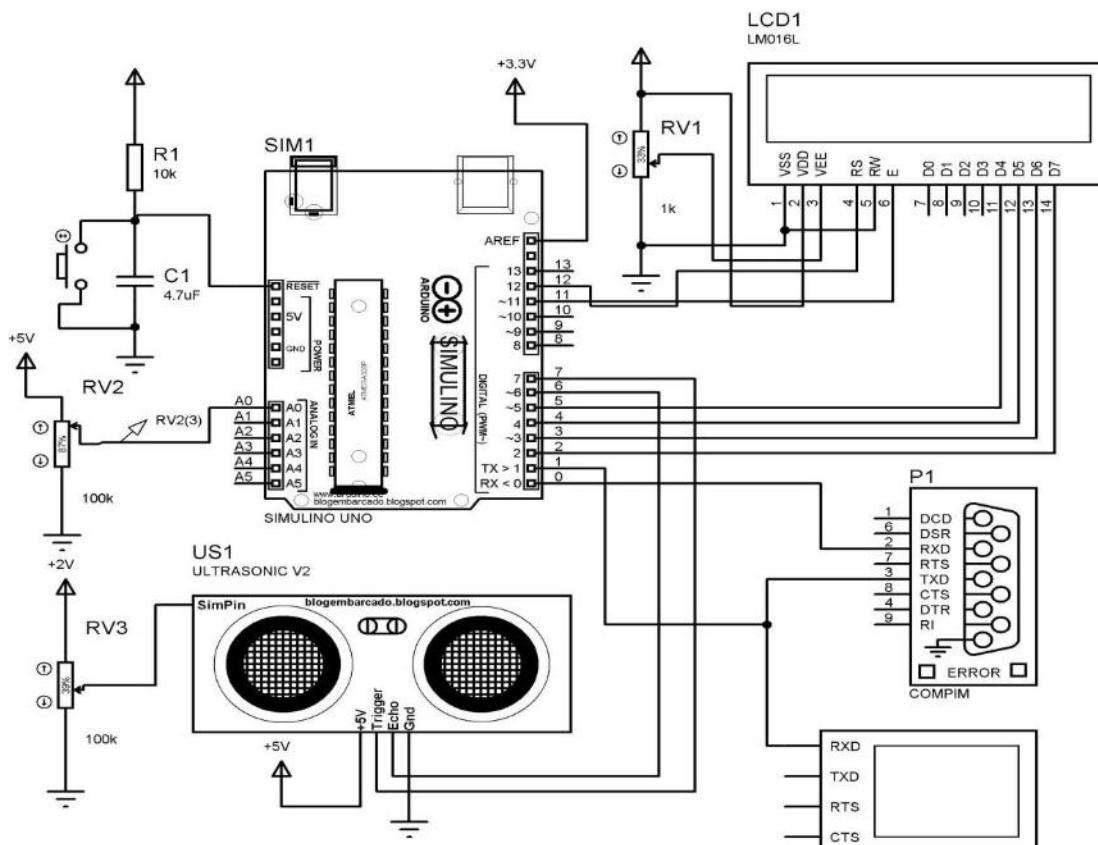
1. Mahasiswa dapat menggunakan sensor ultrasonik.
2. Mahasiswa dapat memahami kerja sensor ultrasonic
3. Mahasiswa dapat mengirim data dari mikrokontroler ke laptop atau sebaliknya menggunakan komunikasi serial.

B. ALAT DAN BAHAN

4. Board mikrokontroler arduino uno
5. LCD 16x2
6. Sensor ultrasonik
7. Beberapa LED dan resistor.
8. Laptop atau PC yang sudah diinstal program proteus dan Aduino IDE

C. PROSEDUR KERJA

Rangkaian yang digunakan pada percobaan ini adalah;



1. Membaca sensor ping dan mengirim hasil pembacaan ke laptop secara serial.

Langkah Percobaan:

1. Buka aplikasi Arduino IDE. Ketikkan sintak di bawah ini;

```
const int trigPin = 7; // mendefinisikan variable trigPin pada pin 7 arduino
const int echoPin = 6; // mendefinisikan variable echoPin pada pin 6 arduino
String string1 = " ";
char b,c;

void setup() {
    Serial.begin(9600);
    pinMode(trigPin, OUTPUT); // mendefinisikan pin 7 arduino sebagai output
    pinMode(echoPin, INPUT); // mendefinisikan pin 6 arduino sebagai input
    c = char (int (65));
    delay(1000);
}
void loop() {

    if (Serial.available()>0) {
        b = Serial.read(); // membaca data yang dikirimkan melalui serial
    }

    if (b == char (int (65))){ // 65 adalah ascii dari A
        c=b;
    }
    else if (b == char (int (66))){ // 66 adalah ascii dari B
        c=b;
    }

    if (c == char (int (65))){ // jika yang diterima adalah huruf A maka
                                // kirim data secara serial ke laptop
        kirimdata();
        //Serial.println("Hasil adalah A");
    }
    else{
        //Serial.println("Hasil adalah B");
    }
}

void kirimdata(){
    float duration; // pendefinisian variable dalam kategori bilangan long.
    long inches, cm; // pendefinisian variable dalam kategori bilangan long.
    digitalWrite(trigPin, LOW); // memberi logika nol pada pin 2 untuk mematikan transmiter sensor
```

```
delayMicroseconds(2); // tunggu selama 2 mikrosekon sebelum
mengaktifkan stransmiter
digitalWrite(trigPin, HIGH); // memberi logika satu pada pin 2
untuk mengaktifkan transmiter
// sensor ping supaya mengirimkan
sinyal
delayMicroseconds(10); // tunggu selama 10 mikrosekon
sebelum mematikan stransmiter
digitalWrite(trigPin, LOW); // memberi logika nol pada pin 2 untuk
mematikan transmiter sensor
duration = pulseIn(echoPin, HIGH); // waktu yang diperlukan sinyal
dari transmiter sampai ke resicer
// setelah dipantulkan oleh objek
dalam mikrosekon.
duration = duration; // merubah dari mikrosekon ke sekon
cm = duration/1.e6/2.*340.*100.; // 340 m/s kecepatan suara di
udara
// dibagi 2 karena waktu gelombang datang
dan pantul
// dikali 100 untuk merubah dari meter ke
cm.
inches = 2.56*cm; // 1 inchi = 2.56 cm.

unsigned long t = millis(); // medapatkan waktu dalam milidetik
String stringOne = string1 + t + string1 + cm + string1;
Serial.println(stringOne); // menampilkan pada serial monitor
delay(100);
}
```

2. Compile sintaknya, kemudian Paste alamat yang telah dicopy tadi ke arduino pada simulator.
3. Buka aplikasi  Serial_communication untuk melihat data yang dikirim oleh arduino ke laptop secara serial. Pilih Boutrate yang bersesuaian dengan yang diketikkan di arduino IDE, dan gunakan COM yang sesuai dengan COM pada simulasi.
4. Simulasikan dan amati yang terjadi, rubah potensio pada sensor ultrasonic untuk mendapatkan nilai jarak yang berubah.
5. Jika simulasi sudah berjalan sesuai dengan yang diharapkan, download file hexa pada board arduino yang ada.
6. Diskusikan dengan teman kelompok:
 - a. Rangkain komponen yang telah disediakan seperti pada rangkaian di simulator kemudian download file heksa untuk mensimulasikan pada kondisi real.
 - b. Buat program untuk menyalakan 3 LED pada arduino dengan inputan hurup 1, 2 dan 3.