PANDUAN PRAKTIKUM DASAR MIKROKONTROLER KELUARGA MCS-51

MENGGUNAKAN

DT-51 MINIMUM SYSTEM VER 3.0 DAN DT-51 TRAINER BOARD

PANDUAN PRAKTIKUM DASAR
MIKROKONTROLER KELUARGA MCS-51
MENGGUNAKAN
DT-51 MINIMUM SYSTEM VER 3.0 DAN DT-51 TRAINER BOARD

Danny Christanto, S.T.
Kris Pusporini, S.T., M.T.

Sampul & Tata Letak: Gersom Sutedjo, S.T.
Diterbitkan pertama kali oleh:
Innovative Electronics, Surabaya 2004
Website: www.innovativeelectronics.com

Semua pertanyaan tentang produk ini dapat dikirimkan melalui e-mail ke:
tutorial51@innovativeelectronics.com

KATA PENGANTAR

Puji Syukur kepada Tuhan Yang Maha Kuasa atas berkat yang telah diberikanNya dalam penyusunan Panduan Praktikum Dasar Mikrokontroler Keluarga MCS-51 ini. Terima kasih juga kepada semua pihak yang telah membantu baik secara langsung maupun tidak.

Panduan Praktikum Dasar Mikrokontroler Keluarga MCS-51 ini digunakan berdampingan dengan Panduan Dasar Mikrokontroler Keluarga MCS-51 yang menyimpan hampir semua referensi modul ini. Semua pelajaran dasar yang ada dapat dipraktekkan dengan menggunakan modul ini.

Besar harapan penyusun agar para Pengguna, baik Praktikan maupun Pembimbing, dapat memberikan kritik dan saran mengenai isi ataupun penyusunan Panduan Praktikum Dasar Mikrokontroler Keluarga MCS-51 ini.

Akhir kata, penyusun mengucapkan terima kasih kepada para Pengguna. Selamat berlatih!

Surabaya, Agustus 2003

Tim Penyusun

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

9. AKSES LCD

9.1. TUJUAN PRAKTIKUM

- Praktikan mampu membuat program untuk menampilkan karakter ke layar LCD dengan menggunakan "PORT LCD" dan rutin-rutin pada DT-51 MinSys.
- Praktikan mampu membuat kreasi karakter sendiri dan menampilkannya pada layar LCD.

9.2. DASAR TEORI

- DT-51 MinSys memiliki rutin-rutin yang dapat langsung digunakan untuk menampilkan karakter ke layar LCD.
- Rutin-rutin tersebut adalah:

1. CBF (0715h)

Fungsi CBF adalah untuk memeriksa *Busy Flag* LCD jika LCD akan diakses secara manual tanpa menggunakan rutin-rutin. Jika menggunakan rutin-rutin DT-51 MinSys, tidak perlu lagi mengakses CBF.

2. InitLCD (0740h)

Rutin ini digunakan untuk menginisialisasi LCD sebelum menggunakan LCD.

3. CommandLCD (07B0h)

Rutin ini digunakan untuk memberikan perintah pada LCD. Rutin CommandLCD dipanggil setelah kita mengisi *Accumulator* dengan nilai *command word* (perintah). Ada beberapa *command word* yang memiliki nilai tersendiri seperti dalam tabel 3.

Tabel 3. Tabel Daftar *Command Word*

Command Word	Nilai	Fungsi
DisplayClear	01h	Menghapus semua tampilan di LCD.
CursorHome	02h	Meletakkan kursor dan tampilan pada posisi awal.
DecCursor	04h	Kursor *decrement* setiap kali selesai menulis atau membaca LCD.
IncCursor	06h	Kursor *increment* setiap kali selesai menulis atau membaca LCD.
CDDSR	05h	Kursor *decrement* dan tampilan bergeser ke kanan setiap kali selesai menulis atau membaca LCD.
ICDSL	07h	Kursor *increment* dan tampilan bergeser ke kiri setiap kali selesai menulis atau membaca LCD.

Tabel 3. Tabel Daftar *Command Word* (sambungan)

Command Word	Nilai	Fungsi
DisplayOff	08h	Memadamkan layar LCD, tampilan dapat muncul kembali jika ada perintah untuk menyalakan layar (dapat dilihat di manual LCD). Tetapi dari *command word* yang ada di sini, mengakses kursor (memadamkan, menghidupkan, atau *blinking*) dapat digunakan untuk menyalakan layar. Selama layar padam, LCD masih dapat ditulisi tapi tidak menampilkan apa-apa.
CursorOff	0Ch	Memadamkan kursor.
CursorOn	0Eh	Menghidupkan kursor.
CursorBlink	0Fh	Kursor menjadi *blinking* (berkedip).
CurShLeft	10h	Menggeser kursor ke kiri.
CurShRight	14h	Menggeser kursor ke kanan.
DispShLeft	18h	Menggeser tampilan ke kiri.
DispShRight	1Ch	Menggeser tampilan ke kanan.

4. WriteLCD (07D0h)

Rutin ini berfungsi untuk menuliskan karakter ke LCD. Kode karakter disimpan pada *Accumulator*. Karakter pada LCD membutuhkan 8 *bit* data. Karena DT-51 MinSys menggunakan *mode* 4 *bit*, maka seharusnya kita mengirimkan dua set data untuk menampilkan satu karakter. Dengan adanya rutin ini, kita tidak perlu mengirimkan dua set data. Untuk menampilkan satu karakter kita cukup mengisi *Accumulator* satu kali dengan karakter yang diinginkan.

Kode DDRAM (*Display Data Random Access Memory*) dari setiap karakter dapat dilihat pada manual LCD. Jika tidak ada, maka tabel 4 memuat tabel referensi dari manual LCD buatan *Seiko Instruments, Inc*. Tabel ini hanya memuat karakter yang umum digunakan (LCD Seiko juga dapat menampilkan karakter Jepang, Yunani, dan karakter lain yang jarang digunakan). Alamat ‘00000000h’ sampai ‘0000FFFFh’ digunakan untuk menyimpan 8 karakter CGRAM (*Character Generator Random Access Memory*). Karena yang digunakan hanya 3 bit terakhir, maka alamat ‘00000000h’ sama dengan alamat ‘00001000h’. Bagian kosong pada kolom “Upper 0010” memang berupa karakter “spasi”.

Tabel 4. Kode Data DDRAM

Upper / Lower	0000	0010	0011	0100	0101	0110	0111
xxxx 0000	(1)		0	@	P	`	p
xxxx 0001	(2)	!	1	A	Q	a	q
xxxx 0010	(3)	“	2	B	R	b	r
xxxx 0011	(4)	#	3	C	S	c	s
xxxx 0100	(5)	$	4	D	T	d	t
xxxx 0101	(6)	%	5	E	U	e	u
xxxx 0110	(7)	&	6	F	V	f	v
xxxx 0111	(8)	‘	7	G	W	g	w
xxxx 1000	(1)	(	8	H	X	h	x
xxxx 1001	(2)	)	9	I	Y	i	y
xxxx 1010	(3)	*	:	J	Z	j	z
xxxx 1011	(4)	+	;	K	[	k	{
xxxx 1100	(5)	,	<	L	¥	l	|
xxxx 1101	(6)	-	=	M	]	m	}
xxxx 1110	(7)	.	>	N	^	n	→
xxxx 1111	(8)	/	?	O	_	o	←

Jika ingin menampilkan huruf J, *Accumulator* diberi nilai ‘01001010b’ atau ‘4Ah’ dan memanggil rutin WriteLCD. Sebagian dari karakter di tabel sama dengan karakter ASCII, sehingga tidak menutup kemungkinan untuk menampilkan karakter lain.

5. **ReadLCD (07F0h)**
   Rutin ini berfungsi untuk membaca karakter dari LCD atau CGRAM kemudian memindahkannya ke *Accumulator*. Pembacaan karakter dari LCD didahului dengan penentuan alamat DDRAM yang diinginkan dan pemanggilan rutin SetDDRAM. Sedangkan pembacaan karakter dari CGRAM didahului dengan penentuan alamat CGRAM yang diinginkan dan pemanggilan rutin SetCGRAM.

6. **ReadAddrLCD (0820h)**
   Rutin ini berfungsi untuk membaca alamat LCD atau CGRAM kemudian memindahkannya ke *Accumulator*. Pembacaan alamat dari LCD didahului dengan penentuan alamat yang diinginkan dan pemanggilan rutin SetDDRAM. Sedangkan pembacaan alamat dari CGRAM didahului dengan penentuan alamat yang diinginkan dan pemanggilan rutin SetCGRAM.

7. **SetDDRAM (0850h)**
   Rutin ini berfungsi untuk menentukan alamat DDRAM pada alamat tertentu yang terdapat pada *Accumulator* sebelum memulai menulis atau membaca LCD. Alamat DDRAM adalah alamat pada tampilan LCD.
   Konfigurasi alamat DDRAM untuk dua baris adalah sebagai berikut:
   Display baris I : 00h – 27h
   Display baris II : 40h – 67h
   Jika LCD yang ada berformat 16 x 2, maka pada baris I adalah alamat DDRAM ‘00h – 0Fh’ dan baris II adalah alamat ‘40h – 4Fh’. jika *display* digeser ke kiri satu kali, maka yang tampil pada baris I adalah alamat ‘01h – 10h’ dan baris II adalah alamat ‘41h – 50h’.

8. **SetCGRAM (0870h)**
   Rutin ini berfungsi untuk menentukan alamat CGRAM sebelum memulai menulis atau membaca pada CGRAM. Alamat tersebut disimpan pada *Accumulator*. Alamat CGRAM adalah alamat per baris pada setiap kotak tampilan. Satu karakter memiliki 8 alamat CGRAM (7 baris untuk karakter dan 1 baris terakhir umumnya untuk kursor) yang masing-masing datanya selebar 8 *bit*. Tetapi dari 8 *bit* tersebut, yang digunakan hanya 5 *bit* LSB (*bit* 4 – *bit* 0). Karakter I dan E pada program “TESLCD” DT-51 MinSys juga menggunakan CGRAM.
   Konfigurasi alamat CGRAM terdapat pada ‘00h – 3Fh’ (karena alamat CGRAM hanya selebar 6 *bit*) dengan kemampuan untuk menampung 8 karakter (1 karakter butuh 8 alamat). Satu contoh untuk membuat karakter “I” dan “E” pada alamat DDRAM ‘00h’ dan ‘01h’ akan diberikan pada tabel 5.

Tabel 5. Contoh Data CGRAM

Data DDRAM	Alamat CGRAM	Data CGRAM
0 0 0 0 * 0 0 0	0 0 0	0 0 0 : 1 1 1 1 1 0 0 1 : 1 1 1 1 1 0 1 0 : 0 0 1 0 0 0 1 1 : 0 0 1 0 0 1 0 0 : 0 0 1 0 0 1 0 1 : 1 1 1 1 1 1 1 0 : 1 1 1 1 1 1 1 1 : 0 0 0 0 0
0 0 0 0 * 0 0 1	0 0 1	0 0 0 : 1 1 1 1 1 0 0 1 : 1 1 1 1 1 0 1 0 : 1 1 0 0 0 0 1 1 : 1 1 1 1 1 1 0 0 : 1 1 0 0 0 1 0 1 : 1 1 1 1 1 1 1 0 : 1 1 1 1 1 1 1 1 : 0 0 0 0 0

Jadi untuk membuat karakter sendiri, yang pertama dilakukan adalah menentukan alamat CGRAM, kemudian diisi dengan dengan data CGRAM. Penampilan karakter tersebut dilakukan dengan mengisi *Accumulator* dengan data DDRAM (bernilai antara '00h-0Fh' pada tabel 4) lalu memanggil perintah WRITELCD.

9.3. PERINTAH YANG DIGUNAKAN

- SJMP, AJMP, LJMP
- LCALL, RET
- MOV, MOVX
- DJNZ
- PUSH, POP

9.4. RUTIN DT-51 MINSYS YANG DIGUNAKAN

- INITLCD (0740H)
- COMMANDLCD (07B0H)
- WRITELCD (07D0H)
- READLCD (07F0H)
- READADDRLCD (0820H)
- SETDDRAM (0850H)
- SETCGRAM (0870H)

9.5. PEMBERITAHUAN

- Semua contoh dan latihan pada bagian ini lebih sesuai untuk LCD dengan ukuran 20 x 2. Tidak menutup kemungkinan untuk menggunakan

LCD dengan ukuran berbeda tetapi mungkin tampilannya akan sedikit berbeda.

9.6. PROSEDUR PERCOBAAN

1) Percobaan I: Penampilan Karakter dan Variasinya

Persiapan:

- Hubungkan "PORT LCD" DT-51 MinSys dengan modul LCD.
- Hubungkan "CONTROL" DT-51 MinSys dengan "CONTROL" DT-51 *Trainer Board* (sebagai sumber tegangan) menggunakan kabel tipe X.
- Hubungkan DT-51 MinSys dengan PC menggunakan kabel serial.
- Hubungkan DT-51 MinSys dengan sumber tegangan.

a. Program 1:
Ketiklah program berikut ini, *assemble*, *download* ke DT-51 MinSys, dan amati hasilnya:

```
$mod51
;built-in routine
INITLCD     EQU   0740H
COMMANDLCD  EQU   07B0H
WRITELCD    EQU   07D0H
;command word
CURSORHOME  EQU   02H
DISPLAYOFF  EQU   08H
CURSOROFF   EQU   0CH
CURSORON    EQU   0EH
CURSORBLINK EQU   0FH
CURSHLEFT   EQU   10H
CURSHRIGHT  EQU   14H
DISPSHLEFT  EQU   18H
DISPSHRIGHT EQU   1CH

            CSEG
            ORG   4000H
            LJMP  START

            ORG   4100H
LDELAY:     PUSH  07H
            PUSH  06H
            PUSH  05H
            MOV   R7, #08H
LOP1:       MOV   R6, #0FFH
LOP2:       MOV   R5, #0FFH
            DJNZ  R5, $
            DJNZ  R6, LOP2
```

```
            DJNZ  R7, LOP1
            POP   05H
            POP   06H
            POP   07H
            RET

START:      MOV   SP, #40H
            LCALL INITLCD

            LCALL LDELAY
;menampilkan "ABC"
            MOV   A, #41H
            LCALL WRITELCD
            LCALL LDELAY
            MOV   A, #42H
            LCALL WRITELCD
            LCALL LDELAY
            MOV   A, #43H
            LCALL WRITELCD
            LCALL LDELAY
;menampilkan kursor blinking
            MOV   A, #CURSORBLINK
            LCALL COMMANDLCD
            LCALL LDELAY
;memadamkan kursor
            MOV   A, #CURSOROFF
            LCALL COMMANDLCD
            LCALL LDELAY
;menampilkan kursor
            MOV   A, #CURSORON
            LCALL COMMANDLCD
            LCALL LDELAY
;memadamkan display
            MOV   A, #DISPLAYOFF
            LCALL COMMANDLCD
            LCALL LDELAY
;menyalakan   tampilan   dengan   mengakses
;kursor
            MOV   A, #CURSORON
            LCALL COMMANDLCD
            LCALL LDELAY
;menggeser kursor ke kanan
            MOV   A, #CURSHRIGHT
            LCALL COMMANDLCD
            LCALL LDELAY
```

```
;menggeser kursor ke kiri
            MOV   A, #CURSHLEFT
            LCALL COMMANDLCD
            LCALL LDELAY
;meletakkan kursor pada posisi awal
            MOV   A, #CURSORHOME
            LCALL COMMANDLCD
            LCALL LDELAY
;menggeser display ke kanan 1 kali
            MOV   A, #DISPSHRIGHT
            LCALL COMMANDLCD
            LCALL LDELAY
;menggeser display ke kiri 2 kali
            MOV   A, #DISPSHLEFT
            LCALL COMMANDLCD
            LCALL LDELAY
            MOV   A, #DISPSHLEFT
            LCALL COMMANDLCD
            SJMP  $
            END
```

Jika tidak ada kesalahan, program tersebut akan bekerja sama seperti yang sudah tercantum pada komentar di awal masing-masing bagian.

Catatan:

Perhatikan bahwa meskipun layar LCD dipadamkan, karakter "ABC" masih tersimpan dalam memori .
Salah satu cara untuk menyalakan layar adalah dengan menyalakan kursor.
Perhatikan juga pada bagian terakhir dimana *accumulator* diisi ulang dengan perintah untuk menggeser tampilan ke kiri meskipun perintahnya sama. Hal ini dilakukan untuk mengantisipasi berubahnya *accumulator* akibat pemangilan rutin CommandLCD.

b. Program 2:
Ketiklah program berikut ini, *assemble*, *download* ke DT-51 MinSys, dan amati hasilnya:

```
$mod51
;built-in routine
INITLCD           EQU   0740H
COMMANDLCD        EQU   07B0H
WRITELCD          EQU   07D0H
SETDDRAM          EQU   0850H
;command word
DISPLAYCLEAR      EQU   01H
DECCURSOR         EQU   04H
```

```
INCCURSOR   EQU   06H

            CSEG
            ORG   4000H
            LJMP  START

            ORG   4100H
LDELAY:     PUSH  07H
            PUSH  06H
            PUSH  05H
            MOV   R7, #08H
LOP1:       MOV   R6, #0FFH
LOP2:       MOV   R5, #0FFH
            DJNZ  R5, $
            DJNZ  R6, LOP2
            DJNZ  R7, LOP1
            POP   05H
            POP   06H
            POP   07H
            RET

START:      MOV   SP, #40H
            LCALL INITLCD

;memindahkan kursor ke alamat 6h
            MOV   A, #6H
            LCALL SETDDRAM
            LCALL LDELAY
;menulis  "15-TD"  dengan  diawali  perintah
;DecCursor
            MOV   A, #DECCURSOR
            LCALL COMMANDLCD
            LCALL LDELAY
            MOV   A, #31H
            LCALL WRITELCD
            LCALL LDELAY
            MOV   A, #35H
            LCALL WRITELCD
            LCALL LDELAY
            MOV   A, #2DH
            LCALL WRITELCD
            LCALL LDELAY
            MOV   A, #54H
            LCALL WRITELCD
            LCALL LDELAY
            MOV   A, #44H
```

```
            LCALL WRITELCD
            LCALL LDELAY
;memindahkan kursor ke alamat 8h
            MOV   A, #8H
            LCALL SETDDRAM
            LCALL LDELAY
;mengembalikan ke penulisan normal dengan
;IncCursor lalu menuliskan "MinSys"
            MOV   A, #INCCURSOR
            LCALL COMMANDLCD
            LCALL LDELAY
            MOV   A, #4DH
            LCALL WRITELCD
            LCALL LDELAY
            MOV   A, #69H
            LCALL WRITELCD
            LCALL LDELAY
            MOV   A, #6EH
            LCALL WRITELCD
            LCALL LDELAY
            MOV   A, #53H
            LCALL WRITELCD
            LCALL LDELAY
            MOV   A, #79H
            LCALL WRITELCD
            LCALL LDELAY
            MOV   A, #73H
            LCALL WRITELCD
            LCALL LDELAY
            SJMP  $
            END
```

Jika tidak ada kesalahan, program tersebut akan bekerja sama seperti yang sudah tercantum pada komentar di awal masing-masing bagian.

Catatan:

Perintah "DecCursor" hanya menggeser kursor ke kiri setiap kali program selesai menuliskan satu karakter pada LCD. Perintah ini tidak menggeser tampilan.

Perintah "IncCursor" menampilkan penulisan normal dimana kursor akan bergeser ke kanan setiap kali selesai menuliskan satu karakter. Perintah ini juga tidak menggeser tampilan.

Jika sebelumnya terdapat perintah "DecCursor", perintah "DisplayClear" akan menghapus semua tampilan sekaligus mengembalikan ke penulisan normal.

c. Program 3:

- Hubungkan *Port* B DT-51 MinSys dengan "PORT OUTPUT" DT-51 *Trainer Board* menggunakan kabel tipe Y.

Ketiklah program berikut ini, *assemble*, *download* ke DT-51 MinSys, dan amati hasilnya:

```
$mod51
;built-in routine
INITLCD     EQU   0740H
COMMANDLCD  EQU   07B0H
WRITELCD    EQU   07D0H
READLCD     EQU   07F0H
READADDRLCD EQU   0820H
SETDDRAM    EQU   0850H
;command word
DISPLAYCLEAR      EQU   01H
INCCURSOR         EQU   06H
CDDSR             EQU   05H
ICDSL             EQU   07H

            CSEG
            ORG   4000H
            LJMP  START

            ORG   4100H
LDELAY:     PUSH  07H
            PUSH  06H
            PUSH  05H
            MOV   R7, #08H
LOP1:       MOV   R6, #0FFH
LOP2:       MOV   R5, #0FFH
            DJNZ  R5, $
            DJNZ  R6, LOP2
            DJNZ  R7, LOP1
            POP   05H
            POP   06H
            POP   07H
            RET

START:      MOV   SP, #40H
            LCALL INITLCD

;inisialisasi Port B untuk LED
            MOV   DPTR, #2003H
            MOV   A, #80H
            MOVX  @DPTR, A
```

```
;memindahkan kursor ke alamat 45h
            MOV   A, #45H
            LCALL SETDDRAM
            LCALL LDELAY
;menulis  "SHOW"  diawali  dengan  perintah
;ICDSL
            MOV   A, #ICDSL
            LCALL COMMANDLCD
            LCALL LDELAY
            MOV   A, #53H
            LCALL WRITELCD
            LCALL LDELAY
            MOV   A, #48H
            LCALL WRITELCD
            LCALL LDELAY
            MOV   A, #4FH
            LCALL WRITELCD
            LCALL LDELAY
            MOV   A, #57H
            LCALL WRITELCD
            LCALL LDELAY
;memindahkan kursor ke alamat 4Ch
            MOV   A, #4CH
            LCALL SETDDRAM
            LCALL LDELAY
;menulis  "DCL"  diawali  dengan  perintah
;CDDSR
            MOV   A, #CDDSR
            LCALL COMMANDLCD
            LCALL LDELAY
            MOV   A, #44H
            LCALL WRITELCD
            LCALL LDELAY
            MOV   A, #43H
            LCALL WRITELCD
            LCALL LDELAY
            MOV   A, #4CH
            LCALL WRITELCD
            LCALL LDELAY
;membaca  karakter  LCD  dan  menampilkan  ke
;Port B
            LCALL READLCD
            MOV   DPTR, #2001H
            MOVX  @DPTR, A
            LCALL LDELAY
```

```
;membaca alamat LCD dan menampilkan ke Port
;B
            LCALL READADDRLCD
            MOV   DPTR, #2001H
            MOVX  @DPTR, A
            LCALL LDELAY
;menghapus semua tampilan
            MOV   A, #DISPLAYCLEAR
            LCALL COMMANDLCD
            LCALL LDELAY
;memindahkan kursor ke alamat 4h dan
;menampilkan "ABC"
            MOV   A, #04H
            LCALL SETDDRAM
            LCALL LDELAY
            MOV   A, #41H
            LCALL WRITELCD
            LCALL LDELAY
            MOV   A, #42H
            LCALL WRITELCD
            LCALL LDELAY
            MOV   A, #43H
            LCALL WRITELCD
            LCALL LDELAY
;menghapus semua tampilan dan kembali ke
;normal
            MOV   A, #DISPLAYCLEAR
            LCALL COMMANDLCD
            LCALL LDELAY
            MOV   A, #INCCURSOR
            LCALL COMMANDLCD
            LCALL LDELAY
;memindahkan kursor ke alamat 4h dan
;menampilkan "ABC"
            MOV   A, #04H
            LCALL SETDDRAM
            LCALL LDELAY
            MOV   A, #41H
            LCALL WRITELCD
            LCALL LDELAY
            MOV   A, #42H
            LCALL WRITELCD
            LCALL LDELAY
            MOV   A, #43H
            LCALL WRITELCD
            LCALL LDELAY
```

```
        SJMP   $
        END
```

Jika tidak ada kesalahan, program tersebut akan bekerja sama seperti yang sudah tercantum pada komentar di awal masing-masing bagian.

Catatan:

Perintah "ICDSL" akan menampilkan penulisan dimana kursor bergeser ke kanan setiap kali selesai menuliskan satu karakter kemudian tampilan bergeser ke kiri. Hal ini membuat kursor seolah-olah diam di tempat, padahal kursor telah menempati alamat yang berbeda (lebih besar).

Perintah "CDDSR" akan menampilkan penulisan dimana kursor bergeser ke kiri setiap kali selesai menuliskan satu karakter kemudian tampilan bergeser ke kanan. Hal ini membuat kursor seolah-olah diam di tempat, padahal kursor telah menempati alamat yang berbeda (lebih kecil).

Perintah "ReadLCD" secara *default* akan membaca karakter yang ditunjuk oleh posisi kursor saat itu. Pembacaan ini dianggap seolah-olah kursor telah menuliskan satu karakter yang tidak nampak sehingga kursor akan bergeser setelah melakukan pembacaan (tanpa menggeser tampilan). Pergeseran kursor tergantung dari adanya perintah "DecCursor"/"CDDSR" atau "IncCursor"/"ICDSL" sebelumnya. Jika sebelumnya terdapat perintah "DecCursor"/"CDDSR", maka kursor akan bergeser ke kiri. Jika sebelumnya terdapat perintah "IncCursor"/"ICDSL", maka kursor akan bergeser ke kanan. Pada program ini yang dibaca oleh perintah "Read LCD" adalah karakter "spasi" sehingga akan tampil di LED sebagai '20h'. Karena sebelumnya terdapat perintah "CDDSR", maka kursor akan bergeser ke kiri, tepat di bawah karakter "W".

Perintah "ReadAddrLCD" secara *default* akan membaca alamat yang ditunjuk oleh posisi kursor pada saat itu. Pembacaan ini tidak menggeser kursor maupun tampilan.

Perhatikan bahwa adanya perintah "ICDSL" atau "CDDSR" sebelum perintah "DisplayClear" akan menghapus tampilan dan membuat penulisan selanjutnya mengikuti cara penulisan "ICDSL" (kursor bergeser ke kanan dan tampilan bergeser ke kiri). Untuk mengembalikan ke penulisan normal (kursor bergeser ke kanan tanpa menggeser tampilan) digunakan perintah "IncCursor".

d. Program 4:

- Hubungkan *Port* B DT-51 MinSys dengan "PORT INPUT" DT-51 *Trainer Board* menggunakan kabel tipe Y.

Ketiklah program berikut ini, *assemble*, *download* ke DT-51 MinSys, dan amati hasilnya:

```
$mod51
;built-in routine
INITLCD     EQU   0740H
WRITELCD    EQU   07D0H
SETDDRAM    EQU   0850H

            CSEG
            ORG   4000H
```

```
START:      MOV   SP, #40H
            LCALL INITLCD
;inisialisasi Port B untuk toggle switch
            MOV   DPTR, #2003H
            MOV   A, #82H
            MOVX  @DPTR, A
;mengambil data dari toggle switch Port B
LOOP:       MOV   A, #00H
            LCALL SETDDRAM
            MOV   DPTR, #2001H
            MOVX  A, @DPTR
            ANL   A, #00001111B
            ADD   A, #30H
            LCALL WRITELCD
            AJMP  LOOP
            END
```

Jika tidak ada kesalahan, program tersebut akan menampilkan angka antara "0 – 9" pada LCD, sesuai dengan kondisi *toggle switch*. Misalkan jika kondisi *toggle switch* bernilai '00000011b' maka LCD akan menampilkan angka "3".

Catatan:

Data LCD merupakan data ASCII sehingga angka "0 – 9" bernilai '30h – 39h'. sehingga untuk menampilkan sebuah angka, data pada *toggle switch* harus ditambah dengan nilai '30h' terlebih dahulu.

Perhatikan adanya perintah ANL yang membuat program hanya menggunakan 4 *bit* LSB (*bit* 3 – *bit* 0) karena 4 *bit* MSB dijadikan '0'.

Program ini tidak membatasi angka "0 – 9" sehingga karakter lain akan muncul jika *toggle switch* bernilai lebih dari '00001001b'.

Latihan Mandiri:

a. Buatlah program untuk menampilkan karakter ke LCD dengan langkah:
   - Pada saat pertama kali program dijalankan, kursor berada di ujung kiri baris pertama.
   - Kemudian kursor akan bergeser 5 kali ke kanan.
   - Program akan menampilkan kata "Lat1".
   - Lalu tampilan akan bergeser ke kanan 3 kali.
   - Diikuti dengan pergeseran kursor 7 kali ke kiri.
   - Terakhir tampilan akan bergeser 6 kali ke kiri.
   - Program akan *looping* ke awal lagi.

b. Buatlah program untuk menampilkan tulisan "Trainer Board" ke LCD dengan syarat:

- Tampilkan tulisan “Trainer” dengan menulis terbalik (dimulai dari “r” dan diakhiri “T”) diikuti dengan penulisan “Board” secara normal (dimulai dari “B” dan diakhiri “d”).
- Kedua tulisan berada di tengah-tengah layar.
- Setelah keduanya tampil, pindah kursor ke tengah-tengah tulisan (huruf ‘r’ kedua), baca alamatnya dan tampilkanlah ke LED. Kemudian baca juga karakternya dan tampilkanlah ke LED

c. Buatlah program untuk menampilkan karakter ke LCD dengan alamat DDRAM 04h (dibantu dengan *toggle switch*) dengan syarat:
   - Pada saat *toggle switch* bernilai ‘00h’, maka LCD akan menampilkan karakter “a”.
   - Pada saat *toggle switch* bernilai ‘01h’, maka LCD akan menampilkan karakter “b”.
   - Begitu seterusnya hingga *toggle switch* bernilai ‘19h’, maka LCD akan menampilkan karakter “z”.
   - Pada saat *toggle switch* bernilai ‘1Ah’, maka LCD akan menampilkan karakter “A”.
   - Pada saat *toggle switch* bernilai ‘1Bh’, maka LCD akan menampilkan karakter “B”.
   - Begitu seterusnya hingga *toggle switch* bernilai ‘33h’, maka LCD akan menampilkan karakter “Z”.

2) Percobaan II: Pembuatan dan Penampilan Karakter

Persiapan:

- Hubungkan “PORT LCD” DT-51 MinSys dengan modul LCD.
- Hubungkan DT-51 MinSys dengan PC menggunakan kabel serial.
- Hubungkan DT-51 MinSys dengan sumber tegangan.

Program 1:

Ketiklah program berikut ini, *assemble*, *download* ke DT-51 MinSys, dan amati hasilnya:

```
$mod51
;built-in routine
INITLCD     EQU   0740H
WRITELCD    EQU   07D0H
SETDDRAM    EQU   0850H
SETCGRAM    EQU   0870H

            CSEG
            ORG   4000H
            LJMP  START

            ORG   4100H
```

```
LDELAY:     PUSH  07H
            PUSH  06H
            PUSH  05H
            MOV   R7, #08H
LOP1:       MOV   R6, #0FFH
LOP2:       MOV   R5, #0FFH
            DJNZ  R5, $
            DJNZ  R6, LOP2
            DJNZ  R7, LOP1
            POP   05H
            POP   06H
            POP   07H
            RET

START:      MOV   SP, #40H
            LCALL INITLCD
            LCALL LDELAY
;membuat karakter di data DDRAM 00000000
;(00h) dan 00000100 (04h)
            MOV   A, #00H
            LCALL SETCGRAM
            MOV   A, #00000111B
            LCALL WRITELCD
            MOV   A, #00001000B
            LCALL WRITELCD
            MOV   A, #00011110B
            LCALL WRITELCD
            MOV   A, #00001000B
            LCALL WRITELCD
            MOV   A, #00011110B
            LCALL WRITELCD
            MOV   A, #00001000B
            LCALL WRITELCD
            MOV   A, #00000111B
            LCALL WRITELCD
            MOV   A, #00000000B
            LCALL WRITELCD

            MOV   A, #20H
            LCALL SETCGRAM
            MOV   A, #00011000B
            LCALL WRITELCD
            MOV   A, #00010100B
            LCALL WRITELCD
            MOV   A, #00011000B
            LCALL WRITELCD
```

```
                MOV   A, #00010100B
                LCALL WRITELCD
                MOV   A, #00010110B
                LCALL WRITELCD
                MOV   A, #00010101B
                LCALL WRITELCD
                MOV   A, #00010110B
                LCALL WRITELCD
                MOV   A, #00010100B
                LCALL WRITELCD
                LCALL LDELAY
;menampilkan karakter di data DDRAM 00h dan
;04h pada awal LCD
                MOV   A, #00H
                LCALL SETDDRAM
                MOV   A, #00H
                LCALL WRITELCD
                LCALL LDELAY
                MOV   A, #04H
                LCALL WRITELCD
                LCALL LDELAY
                SJMP  $
                END
```

Jika tidak ada kesalahan, program tersebut akan menampilkan logo menyerupai "€" dan "Rp".

Catatan:

Untuk membuat karakter di data DDRAM '00h' (0000*000), maka alamat CGRAM harus dipindah ke '00h' (000|000).

Untuk membuat karakter di data DDRAM '04h' (0000*100), maka alamat CGRAM harus dipindah ke '20h' (100|000).

Penulisan data CGRAM dilakukan secara berurutan mulai dari baris paling atas (alamat CGRAM terkecil, 000|000 dan 100|000) ke baris paling bawah (alamat CGRAM terbesar, 000|111 dan 000|111). Alamat CGRAM akan bertambah secara otomatis.

Baris terakhir umumnya dikosongkan sebagai tempat untuk kursor, tetapi dapat juga diisi.

Langkah selanjutnya untuk menampilkan karakter yang telah dibuat adalah mengisi *Accumulator* dengan data DDRAM '00h' dan '04h' lalu memanggil perintah "WriteLCD".

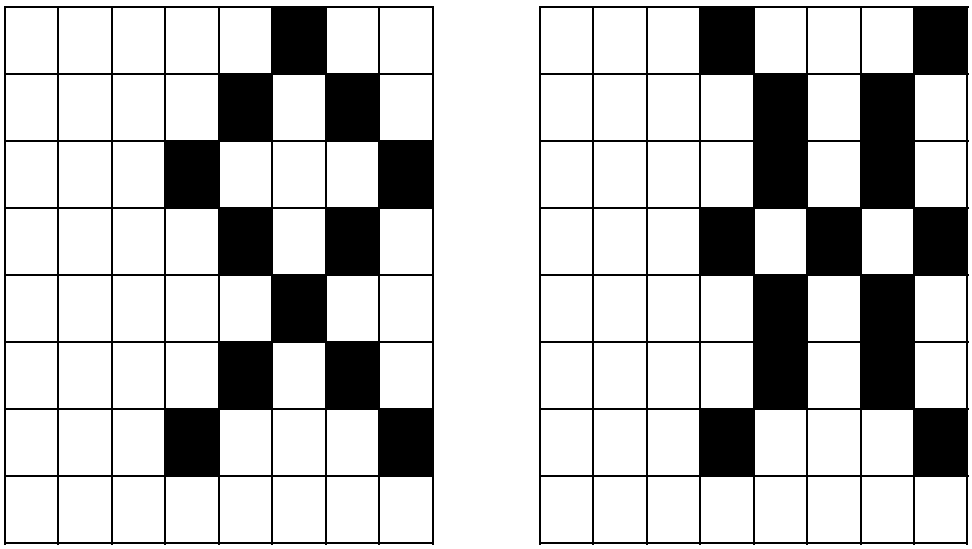
Data CGRAM juga dapat disimpan dalam EEPROM.

Latihan Mandiri:

Buatlah program untuk membuat dan menampilkan karakter dengan tampilan seperti dalam gambar 19 dengan syarat:

- Buat kedua karakter tersebut pada data DDRAM '02h' dan '05h'.

- Tampilkan kedua karakter tepat di tengah-tengah layar LCD baris pertama.
- Setelah program menampilkan kedua karakter ke LCD, geserlah tampilan ke kiri 2 kali dan ke kanan 4 kali kemudian ke kiri lagi 2 kali.



Gambar 19. Dua Karakter CGRAM