

CHARACTER MODULE INITIALIZATION

Internal Reset Circuit

The module is automatically initialized when the power is applied. The following commands are executed during initialization. The busy flag is kept in the busy state until initialization is complete. The busy state lasts for 10 ms after V_{DD} reaches 4.5 volts.

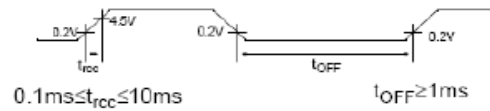
- 1) Clear Display
DL=1.....8-bit data length for interface
- 2) Function set
N=0.....Single-line display
F=0.....5x7 dot matrix character font
- 3) Display ON/OFF Control
D=0.....Display OFF
C=0.....Cursor OFF
B=0.....Blink function OFF
- 4) Entry Mode Set
I/D=1.....Increment Mode
S=0.....Display shift OFF

NOTE:

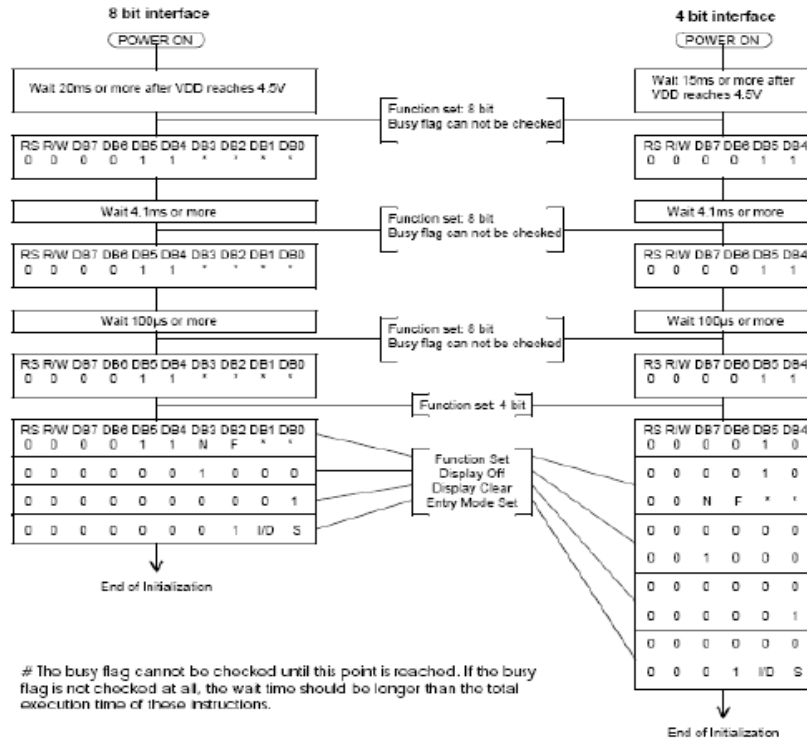
If the following power conditions are not satisfied, the internal reset circuit does not function properly. In this case, the initialization should be executed by the series of instructions from outside the MPU (Software Initialization).

Power Conditions for Internal Reset

ITEM	SYMBOL	MIN	TYP	MAX	UNIT
Power Supply Rise Time	t_{rcc}	0.1	-	10	ms
Power Supply Off time	t_{off}	1.0	-	-	ms



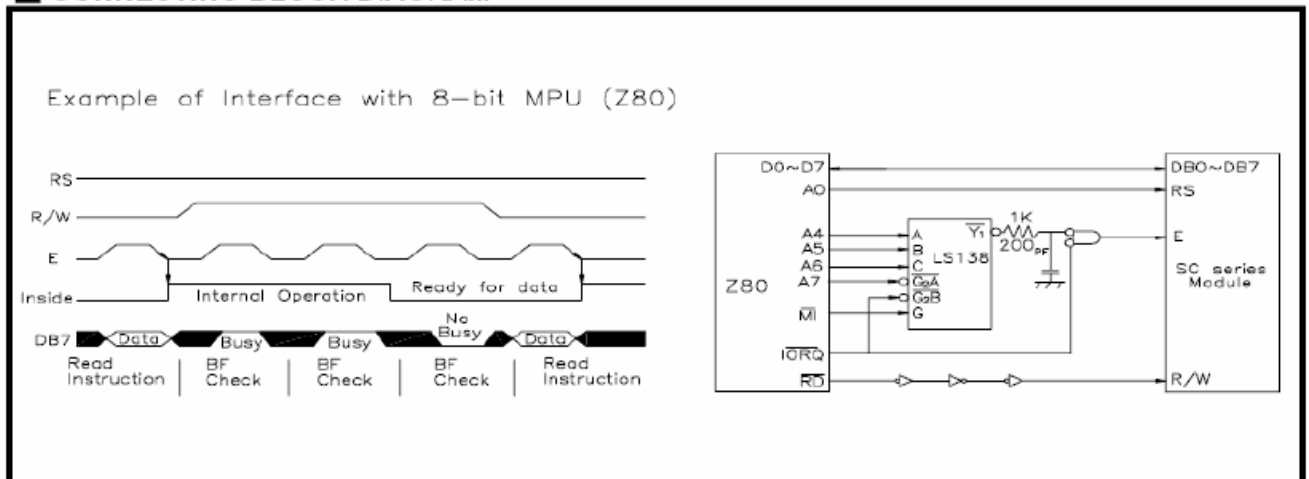
Software Initialization



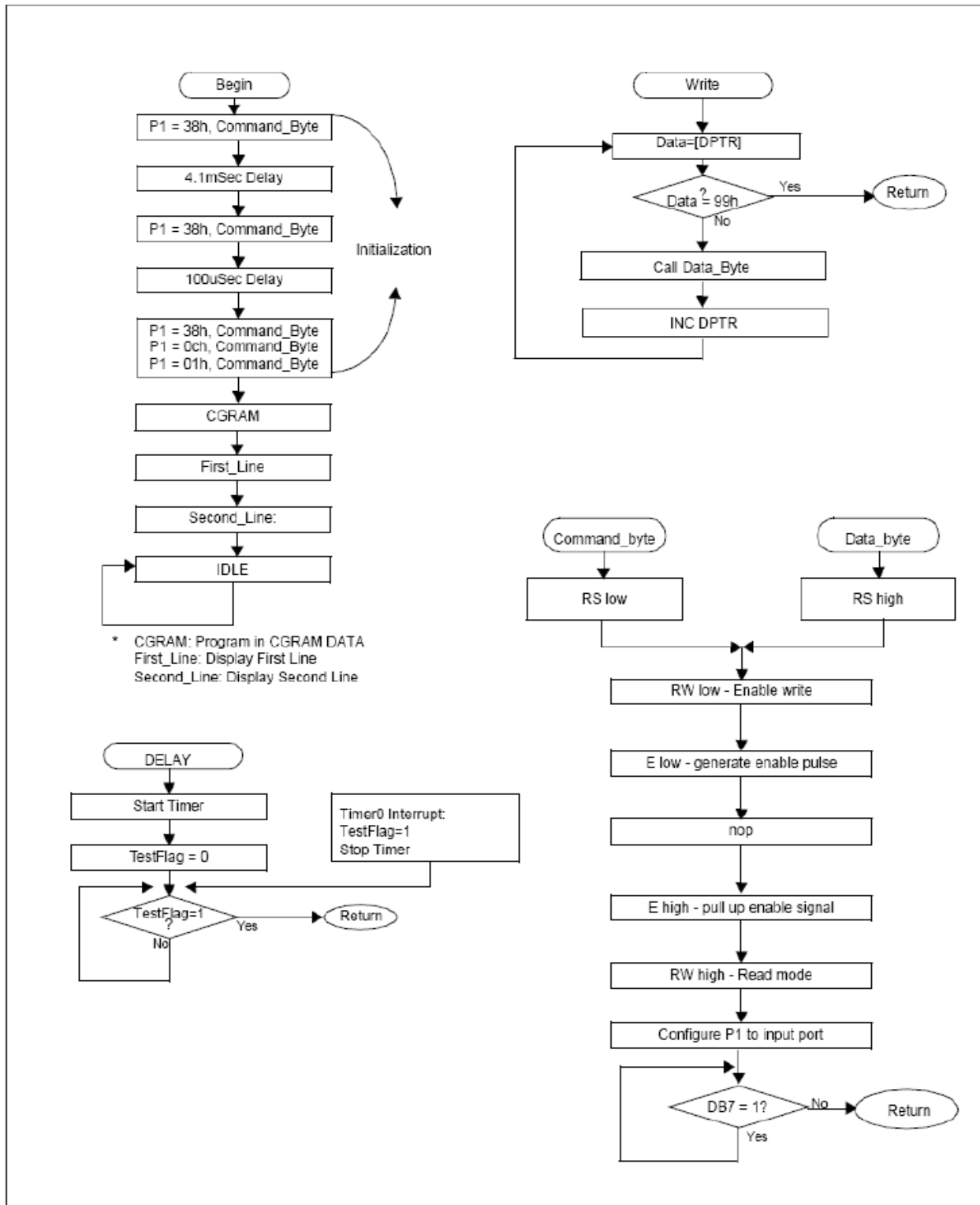
■ DISPLAY COMMANDS

INSTRUCTION	CODE										Description
	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	
1: Clear Display	0	0	0	0	0	0	0	0	0	1	Clears entire display and sets DD RAM address 0 in address counter .
2: Return Home	0	0	0	0	0	0	0	0	1	*	Sets DD RAM address 0 in address counter . Also returns display from being shifted to original position . DD RAM contents remain unchanged .
3: Entry Mode Set	0	0	0	0	0	0	0	1	I/D	S	I/D=1 : Increment I/D=0 : Increment S=1 : Accompanies display shift
4: Display On/Off	0	0	0	0	0	0	1	D	C	B	I/D=1/0 : Display on/off I/D=0/1 : cursor on/off S=1 : Blink of cursor
5: Cursor /Display shift	0	0	0	0	0	1	S/C	R/L	*	*	S/C=1 : Display shift S/C=0 : Cursor move R/L=0 : Shift to left R/L=1 : Shift to right
6:Function Set	0	0	0	0	1	DL	N	F	*	*	DL=1 : 8 bits , DL=0 : 4 bits N=1 : 2 lines , N=0 : 1 line F=1 : 5*10 dots , F=0 : 5*8 dots
7: Set CGRAM Address	0	0	0	1	Acc					Acc : CGRAM address	
8: Set DD RAM address	0	0	1	Acc					Acc : DD RAM address corresponds to cursor address		
9: Read busy flag/address counter	0	1	BF	Ac					BF=1 : Busy , BF=0 : Not busy Ac : Address counter used for both of CG and DD RAM address		
10:Write data	1	0	WRITE DATA					Write data to CG or DD RAM			
11:Read data	1	1	READ DATA					Read data from CG or DD RAM			
<p>☆ Execution Time (Et) of Instruction : (Under condition of fosc = 270 KHz)</p> <p>1 & 2 : Et=1.52 ms</p> <p>3 ~ 11 : Et=37 μs</p> <p>☆ "*" : Either 0 or 1</p>											

■ CONNECTING BLOCK DIAGRAM



III. Software Flowchart:



PIN ASSIGNMENT

PIN NO.	SYMBOL	LEVEL	FUNCTION
1	VSS	-	Power Supply
2	VDD	-	
3	Vo	-	
4	RS	H / L	Selects Registers H: Data register (When Writing And Reading) L: Instruction Register (Writing) Busy Flag And Address Counter (Reading)
5	R / W	H / L	Read/Write Select Signal H: Data read (Module→MPU) L: Data write (Module→MPU)
6	E	H, H→L	Enable Signal
7	DB0	H / L	Databus lines, see description below
8	DB1	H / L	DB4~DB7:
9	DB2	H / L	High-order lines of data bus with three-state, bi-directional function for use
10	DB3	H / L	in data transfer with the MPU. DB7 may also be used to check the busy flag
11	DB4	H / L	DB0~DB3:
12	DB5	H / L	Low-order lines of data bus with three-state, bi-directional function for use
13	DB6	H / L	in data transfer with the MPU. These lines are not used when interfacing
14	DB7	H / L	with a 4-bit microprocessor.

ELECTRIC MAXIMUM RATINGS

ITEM	SYMBOL	MIN	MAX	UNIT	REMARKS
Power Supply For Logic	VDD-VSS	-0.3	7.0	V	
Signal Input Voltage	VIN	$-3 \leq V_{IN} \leq V_{DD}+0.3$		V	
Static Electricity	-	-	100	V	See Note

Note: Electro-static discharge resistance is tested by charging a 200pf capacitor and discharging it by contact with a interface connector pin .

EXAMPLE OF POWER SUPPLY

FIG.1 Normal Temperature Type

FIG.2 Extended Temperature Type

*Note: If V vary from recommended value, you cannot get proper contrast or viewing angle.

■ Examples of Temperature Compensation Circuits for Extended Temp Type. (Only for reference)

(A) 1/8Duty - 1/4Bias

(B) 1/16Duty - 1/5Bias

Thermistor: $R_{th}(25^{\circ}C) = 15[k\text{-ohm}]$, $B = 4200[K]$
 Resistors: $R_p = 30[k\text{-ohm}]$, $R_s = 6.8[k\text{-ohm}]$, $R_m = 3.3[k\text{-ohm}]$
 Transistors: PNP Type.
 $V_{cc} = +5V$, $V_{ss} = 0V$ (Logic Supply)
 $V_z = -8V$ (-7.8 ~ -8.2V)
 $V_{ee} < V_z$, $R_z = (V_z - V_{ee}) / 5$ [k-ohm]

Thermistor: $R_{th}(25^{\circ}C) = 15[k\text{-ohm}]$, $B = 4200[K]$
 Resistors: $R_p = 150[k\text{-ohm}]$, $R_s = 8.2[k\text{-ohm}]$, $R_m = 3.6[k\text{-ohm}]$
 Transistors: PNP Type.
 $V_{cc} = +5V$, $V_{ss} = 0V$ (Logic Supply)
 $V_z = -11V$ (-10.725 ~ -11.275V)
 $V_{ee} < V_z$, $R_z = (V_z - V_{ee}) / 5$ [k-ohm]

Deepakshi Display Devices Pvt Ltd

<pre> Application Note ;***** ;Displaying Characters on DM0801 LCD Module ;Description: Demo software to display ; characters. ; Controller: AT89S52 ;***** ;Equates ;***** lcdport data p1 e equ p3.3 rw equ p3.1 rs equ p3.0 ;***** ; Interrupt Vectors ; *****] org 000h jmp start ; Power up reset vector org 003h reti ; External interrupt 0 vector org 00bh reti ; Counter/ Timer 0 int vector org 013h reti ; External int 1 vector org 01bh ; reti ; Timer 1 int vector org 023h reti ; I2C serial int vector ;***** ;Start Program ;***** start: acall init main: acall delay_large mov dptr,#ka acall nxt acall delay_large acall delay_large mov dptr,#k1 acall nxt here: ajmp start </pre>	<pre> nxt: clr a movc a,@a+dptr cjne a,#0ffh,go_on ret go_on: acall datawr acall delay inc dptr mov p1,a ajmp nxt command: setb e clr rs clr rw mov p1,a clr e ret datawr: setb e setb rs clr rw mov p1,a acall delay clr e ret ;***** ;INITIALIZATION ;***** init: mov a,#38h ;function set acall command acall delay mov a,#38h ; function set acall command acall delay mov a,#0ch ;display on/off acall command acall delay mov a,#01h ;clear display acall command acall delay mov a,#80h ;set ddram acall command acall delay ret </pre>
--	--

```
,*****  
ka:      db   '*DDDPL**',0ffh  
k1:      db   'LCD 0801',0ffh
```

```
,*****  
;DELAY  
,*****
```

```
delay_large:      mov r4,#32  
abc:              mov r5,#100  
sm:              mov r6,#100  
mn:              djnz r6,mn  
                 djnz r5,sm  
                 djnz r4,abc  
                 ret
```

```
delay:           mov r4,#05d  
delay1:         mov r5,#60d  
la:             djnz r5,la  
                 djnz r4,delay1  
                 ret  
end
```