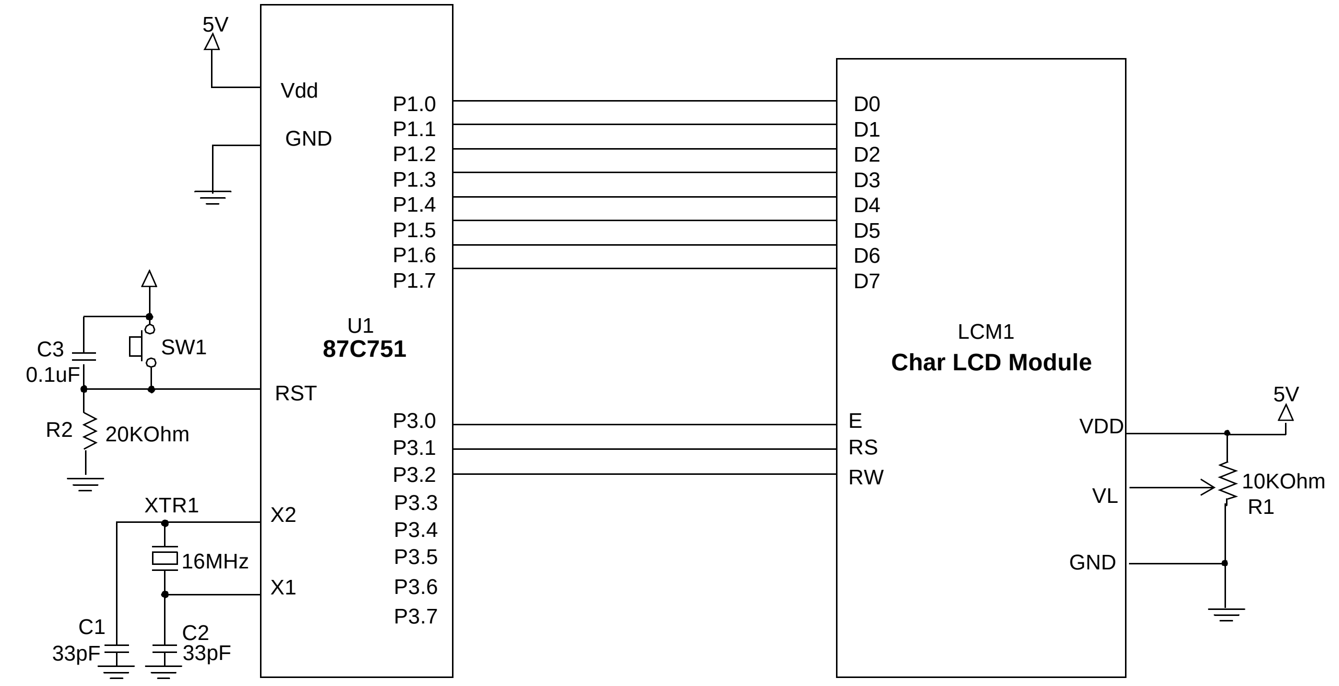
Displaying Characters on an LCD Character Module

I. Introduction:

This application note describes a simple technique to display characters from both the internal character generator and user designed characters on an LCD character module.The controlling microcontroller is a Phillips 87C751, a derivitive of the popular Intel 8051. The LCD module is connected to the microcontroller through its I/O ports. It could also be connected directly to the data bus with the addition of address decoding logic.

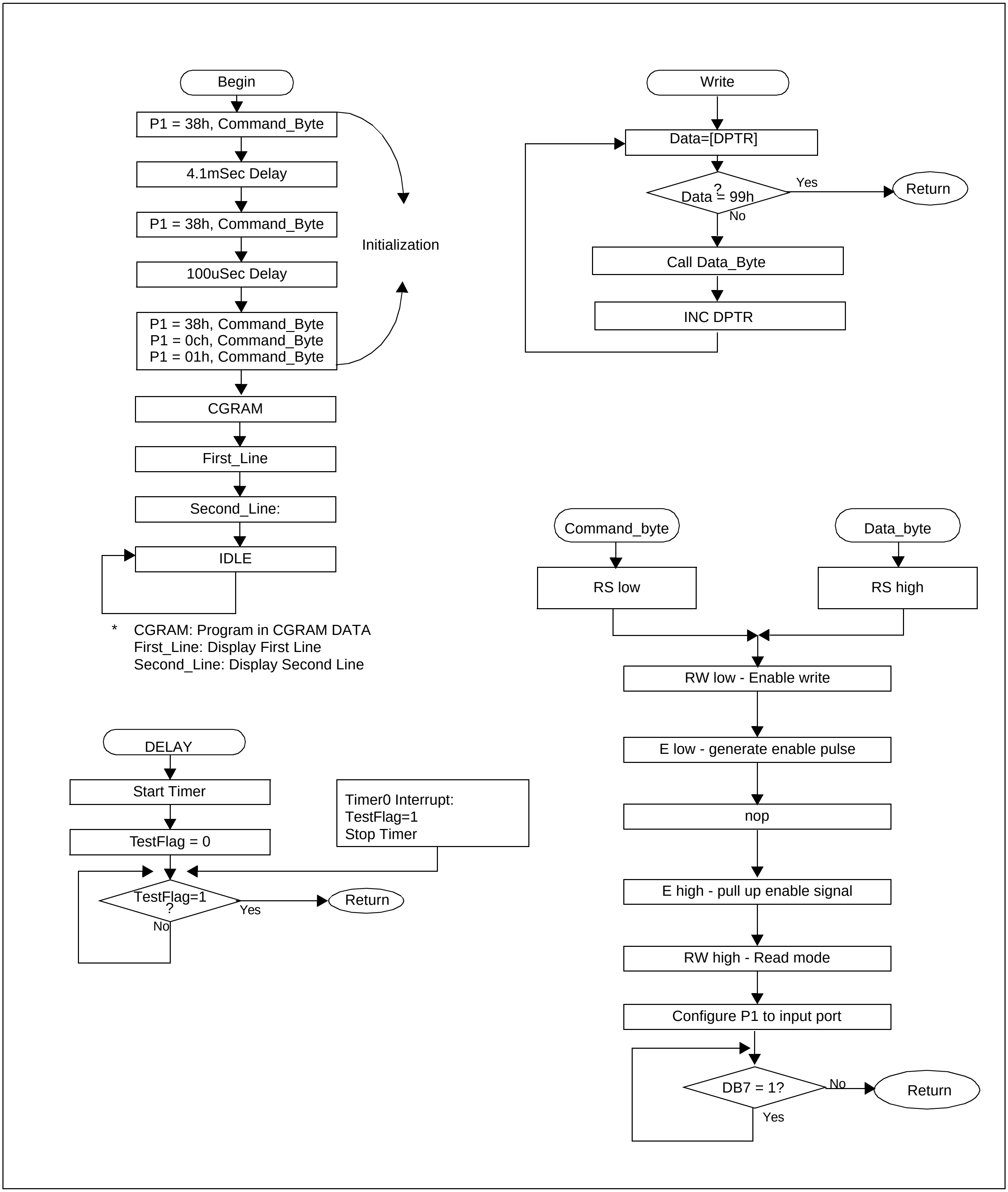
The process of displaying character to this module is divided into three steps. First the module must be initialized. This sets up the built-in LCD controller chip. Second, some user designed characters are uploaded to the CGRAM. This allows the displaying of up to 8 custom characters in addition to the 192 character permanently stored in the module. Lastly, a message consisting of a mix of standard ASCII characters and custom designed characters is displayed on the module.

II. Circuit Schematic



Parts List	Part Number	Description
LCM1	HDM16216H-5	Hantronix Alphanumeric LCD Module
U1	87C751	Phillips Microcontroller
XTR1	NMP160	16MHz Crystal
C1,C2	33pF	Capacitor
SW1	--	Push Button Switche (Reset)
R1	10Kohm	Pot
R2	20Kohm	Resistor
C3	0.1 uF	Capacitor

III. Software Flowchart:



* CGRAM: Program in CGRAM DATA
First_Line: Display First Line
Second_Line: Display Second Line

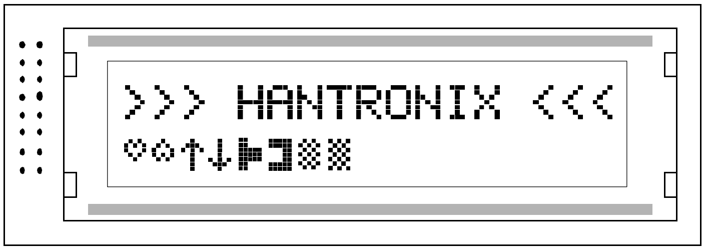
```
;**********************************************************
;;          Application Note:
;           =================
;           Displaying Characters on an LCD Character Module
;
;           Description:            Demo software to display “canned”
                                    message and custom characters.
.;          Controller:             Phillips 87C751
;           LCD controller:         HD44780, KS0066, SED1278
;
;*************************************************************
;           Constant Definition
;*************************************************************
EnableT0                equ         082h        ;enable timer0
Disable                 equ         000h        ;disable timer
D4100h                  equ         00ch        ;timer reload high byte def.
D4100l                  equ         003h        ;timer reload = 4.1mSec.
D100h                   equ         000h        ;timer reload
D100l                   equ         04ch        ;timer reload = 100uSec.
;*****************************************
;           Ram Definition
;*****************************************
Flags                   DATA        020h        ;flag
TstFlag                 BIT         Flags.0     ;interrupt flag bit
;*****************************************
;           Port Connections
;           =================
;           P1.0 -> D0
;           P1.1 -> D1
;           P1.2 -> D2
;            . . .
;           P1.7 -> D7
;           P3.0 -> Enable
;           P3.1 -> RS
;           P3.2 -> RW
;*******************************************************
;           Interrupt Vectors
;           ------------------
            org         000h
            jmp         PowerUp                 ; Power up reset vector
            org         003h
            jmp         ExInt0                  ; External interrupt 0 vector
            org         00bh
            jmp         Timer0                  ; Counter/ Timer 0 int vector
            org         013h
            jmp         ExInt1                  ; External int 1 vector
            org         01bh                    ;
            jmp         Timer1                  ; Timer 1 int vector
            org         023h
            jmp         I2C                     ; I2C serial int vector

            org         50h
PowerUp:
;*****      Timer 0 Interrrupt preparation
            clr         TR                      ;disable timer
            clr         TF                      ;clear overflow
;**************************************
;           LCD Initialization Routine
;**************************************
cinit:      clr         P3.1                    ;RS low
            clr         P3.2                    ;RW low
            setb        P3.0                    ;Enable

            mov         RTL,#D4100l             ;set timer reload value
            mov         RTH,#D4100h             ;dealy time = 4.1mSec
```

```
            mov         p1,#38h
            acall       command_byte
            acall       ddelay                  ;initial delay 4.1mSec

            mov         RTL,#D100l              ;set timer reload value
            mov         RTH,#D100h              ;dealy time = 100uSec
            mov         p1,#38h                 ;function set
            acall       command_byte
            acall       ddelay                  ;busy flag not avail. yet

            mov         p1,#38h                 ;function set
            acall       command_byte
            mov         p1,#0ch                 ;display on
            acall       command_byte
            mov         p1,#01h                 ;clear display
            acall       command_byte

            acall       cgram                   ;define custom fonts
            acall       first_line              ;display first line
            acall       second_line             ;display second line
sdone:
            setb        IDLE                    ;power down mode
            jmp         sdone

;********************************************************
;Subroutine: WRITE
;=================
;Purpose: To feed in data/command bytes to the LCD module
;Parameters:dptr =      should be set to the beginning of
;                                   the data byte address
;                       Data bytes should be finished with 99H
;Alg:       get a new data/command byte
;           while (new data != 99h) {
;                       set port1 with new data
;                       call data_byte
;                       increment data pointer
;                       }
;           return
;********************************************************
write:
write_loop:
            mov         a,#0
            movc        a,@a+dptr
            cjne        a,#99h,write_cont
            ret
write_cont:
            mov         p1,a
            acall       data_byte
            inc         dptr
            jmp         write_loop

;**********************************************
;           Delay Routine:
;           Delay period = 4/3uSec * DefRth,l
;**********************************************
ddelay:
            setb        TR                      ;start timer
            mov         IE,#EnableT0            ;enable timer
            clr         TstFlag                 ;reset flag
dloop:      jnb         TstFlag,dloop
            ret
;********************************
;           set address to beginning
;           of CG RAM
;********************************
```

```
cgram:
            mov     p1,#40h
            acall   command_byte
            mov     dptr,#cgram_data
            acall   write
            ret
;********************************
;           Set DDRAM to the beginnig of
;           the first line - 00
;********************************
first_line:
            mov     p1,#080h ;set DDRAM
            acall   command_byte
            mov     dptr,#fline_data
            acall   write
            ret
;********************************
;           Set DDRAM to the beginning of
;           the second line - 40
;********************************
second_line:
            mov     p1,#0c0h ;set DDRAM
            acall   command_byte
            mov     dptr,#sline_data
            acall   write
            ret
;**********************************************************
;           Feed Command/Data to the LCD module
;***********************************************************
command_byte:
            clr     p3.1                    ; RS low for a command byte.
            jmp     bdelay
data_byte:
            setb    p3.1                    ; RS high for a data byte.
            nop
bdelay:
            clr     p3.2                    ; R/w low for a write mode
            clr     p3.0
            nop
            setb    p3.0                    ;enable pulse
            nop
;******************* Check Busy Flag
            mov     p1,#0ffh  ;configure port1 to input mode
            setb    p3.2                    ;set RW to read
            clr     p3.1                    ;set RS to command
            clr     p3.0                    ;generate enable pulse
            nop
            setb    p3.0
bloop:      nop
            mov     a,p1
            anl     a,#80h                  ;check bit#7 busy flag
            cjne    a,#00h,bloop;keep waiting until busy flag clears
;*****************************************
;           check busy flag twice
;*****************************************
bwait:
            mov     a,p1
            anl     a,#80h
            cjne    a,#00h,bloop
            clr     p3.2                    ;return to write mode
            ret
;******************************************
;           Interrupt Routines
;******************************************
ExInt0:
```

```
ExInt1:
Timer1:
I2C:
            reti
;**********************************************
Timer0:
            setb    TstFlag
            clr     TR
            mov     IE,#Disable
            reti
;********************************************
;           Data Bytes
;*******************************************
FLINE_DATA:
            db      ‘>>> HANTRONIX <<<’
            db      099h
SLINE_DATA:
            db      00h,01h,02h,03h,04h,05h,06h,07h
            db      099h
CGRAM_DATA:
font1:      db      0ah,15h,11h,11h,0ah,04h,00h,00h
font2:      db      04h,0ah,11h,11h,15h,0ah,00h,00h
font3:      db      04h,0eh,15h,04h,04h,04h,04h,00h
font4:      db      04h,04h,04h,04h,15h,0eh,04h,00h
font5:      db      18h,18h,1fh,1fh,1fh,18h,18h,00h
font6:      db      1fh,1fh,03h,03h,03h,1fh,1fh,00h
font7:      db      0ah,15h,0ah,15h,0ah,15h,0ah,00h
font8:      db      15h,0ah,15h,0ah,15h,0ah,15h,00h
            db      99h
            end
```

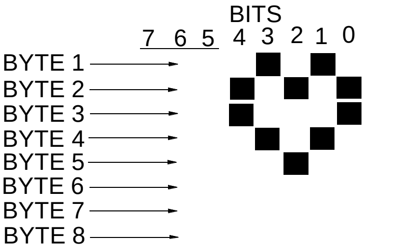
*Display on a 16x2 Character Module



* Display Character Position and DDRAM Address of a 16x2 Character Module.

	Display Position 1	2	3	4		14	15	16
First Line	00	01	02	03	. . . .	0D	0E	0F
Second Line	40	41	42	43	. . . .	4D	4E	4F

DDRAM Address



BITS 5,6,&7 NOT USED

* Custom characters memory map.