Product Information
SSD1821Z Development Kit

DVK1821Z-A1

DVK1821Z-A1 is a demonstration of SSD1821Z working on a (128 X 80 + 1 icon line) panel. It is intended to help users expedite their design-in of SOLOMON LCD driver.

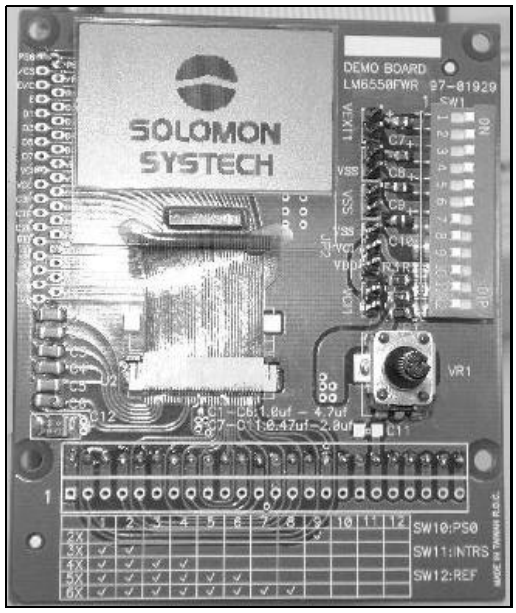

Figure 1 DVM1821Z-A0 outlook

PACKAGE CONTENTS

DVK1821Z-A1 consists of the following:

1) LCD Module (128 X 80 + 1 icon line) (DVM1821Z-A0)
2) LCD Drawing (optional)
3) Programmed 8051 MCU Board (EVM89C52-A0)
4) 8051 MCU Board Schematics (EVM89C52-A0)
5) Demo Program in C Language (PRG1821Z-A1)

SYSTEM REQUIREMENT

DVK1821Z-A1 is good enough to serve as a standalone demo. 2 X AA 1.5V battery is required for normal operation. For engineering evaluation on the LCD driver and/or LCD panel, a 8051 Incircuit emulator (ICE) is required. The one Solomon Systech is using is EMMIT 8051 ICE from Syber Electronics Co Ltd. User can get its information from the email: syber@public1.pt

LCD MODULE

The LCD module has been configured as follows
1) size: 128 X 80 + 1 icon line
2) COG Package
3) 6X DC-DC converter to generate VEE
4) Internal feedback resistor which is software programmable is used to generate VL6.
5) Serial interface (SPI 4 Pins).
6) 24 pins single inline (SIL) header is used to connect 8051 MCU Board to LCD Module. Please refer to **Table 1** for pin assignment.
7) DIP SW 1-9 are to control the DC-DC conveter. SW 10 is the PS0 setting, trun on in SPI mode. SW 11 & 12 are for internal testing only, please trun off for normal use.

ORDERING INFORMATION

Item	Ordering Part Number
SSD1821Z Development Kit	DVK1821Z-A1
SSD1821Z Development Module	DVM1821Z-A0

Pin Number	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
LCD Module	VDD	$\overline{RES}$	PS1	D/$\overline{C}$	R/$\overline{W}$	$\overline{CS1}$	VSS	D0	D1	D2	D3	D4	D5	D6 (SCK)	D7 (SDA)	E	NC	VSS	NC	NC	NC	NC	NC	NC
8051 MCU Board	VOUT	PA-04	PA-05	PA-01	PA-07	PA-06	GND	PB-00	PB-01	PB-02	PB-03	PB-04	PB-05	PB-06	PB-07	PA-00	VOUT	GND	+9V	P34	P35	P36	P37	NC

Table 1 DVM1821Z-A1pin assignment

8051 MCU BOARD

The 8051 MCU Board is powered up by 2 X AA 1.5V battery. A 2X DC converter (ICL7660) is used to generate 4.8V to supply 8051 MCU. This voltage is also regulated by LM317 to generate VOUT (adjustable from 1.8V to 3.5V) which in turn powers up solomon LCD driver. All logic output from 8051 are down-converted from 4.8V to VOUT through 74HC4050 non-inverting buffer. User can change VOUT by tuning T1 (1K trimmer).The MCU board is configured to run at a speed of 12MHz. 4 dummy keys are reserved for developing user-interactive application such as tuning contrast.

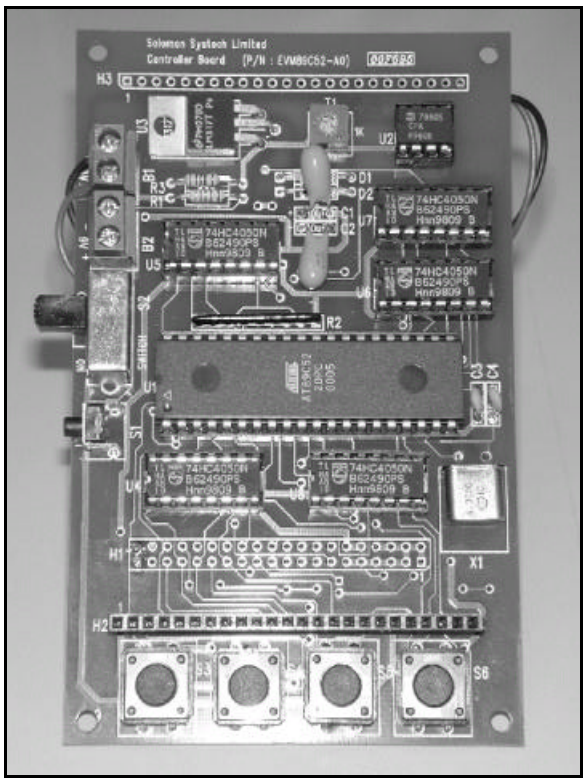

Figure 2 EVM89C52-A0 outlook

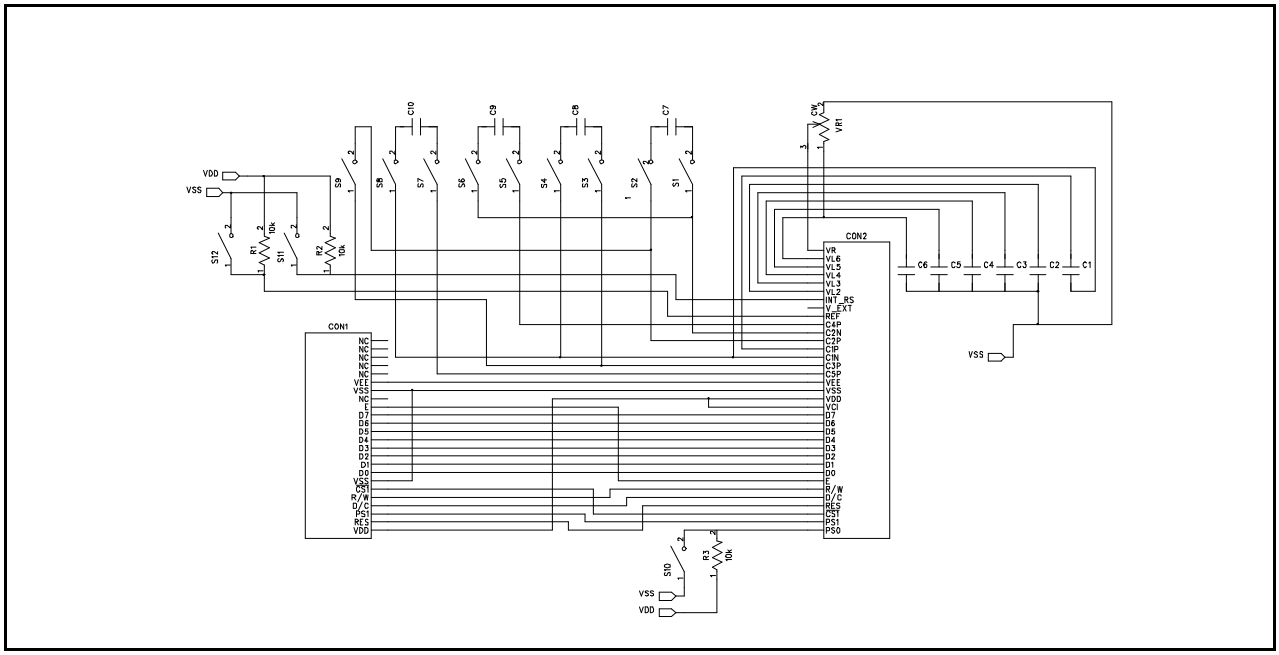

Figure 3 DVM1821Z-A0 schematics

PROGRAMMING NOTE

```
/*_____________________________________________________________________________*/
/*  Prog:       PRG1821Z-A1.C                                                  */
/*  Device:     DVK1821Z-A1                                                    */
/*  Descp:      1821 Demo in SPI 4 Pins (in 12MHz)                             */
/*  LCM:        128x81                                                         */
/*  Written by: Tom Tsang                                                      */
/*  Date:       2k0824                                                         */
/*  Structure:  A) Hardware Interface                                          */
/*              B) Command Table per device                                    */
/*              C) Global Variables Definition                                 */
/*              D) Hardcoded Graphics                                          */
/*              E) Function Prototypes                                         */
/*              F) Main Function                                               */
/*                                                                             */
/*  Remark: PS0 is hardwire on DVM1821Z-A0 by DIP SW, PS1 is programable pin*/
/*_____________________________________________________________________________*/
#include “reg8751.h”
#define controlport P0
#define dataport    P1

/************************************************
* A) HardWare Interface                         *
*                                               *
* PORT A  7   6   5    4    3   2   1   0       *
*         RW  CS  PS1  RES  --  --  DC  E       *
*                                               *
* E and PS may be tied high                     *
*                                               *
* PORT B  7   6   5    4    3   2   1   0       *
*         D7  D6  D5   D4   D3  D2  D1  D0      *
*        SDA SCK                                *
************************************************/

/************************************************
* B) Command Table per device                   *
************************************************/
#define DisplayOff      0xAE
#define DisplayOn       0xAF
#define DisplayStart    0x40
#define PageAddr        0xB0
#define ColAddrHi       0x10
#define ColAddrLo       0x00
#define SegRemapOff     0xA0
#define SegRemapOn      0xA1
#define NormalDisp      0xA6
#define ReverseDisp     0xA7
#define ExitEntireD     0xA4
#define EntEntireD      0xA5
#define EnterRMW        0xE0
#define ExitRMW         0xEE
#define SWRest          0xE2
#define ComRemapOff     0xC0
```

```
#define ComRemapOn      0xC8
#define PwrCtrlReg      0x28
#define OPampBuffer     0x01
#define IntReg          0x02
#define IntVolBstr      0x04
#define IntRegRatio     0x20
#define ContCtrlReg     0x81

#define CmdMuxRatio     0x48
#define CmdBiasRatio    0x50
#define DispOffset      0x44
#define IconModeOn      0xA3
#define IconModeOff     0xA2

#define NlineInver      0x4C
#define DCDCconver      0x64
#define PowersavStandby 0xA8
#define PowersavSleep   0xA9
#define PowersavOff     0xE1
#define InterOsc        0xAB

#define Device      SSD1821     /* device under demo */
#define ColNo       128         /* number of Column/Seg on LCD glass*/
#define RowNo       80          /* number of Row/Com/Mux */
#define PS          1           /* fixed to Parallel mode */
#define PageNo      10          /* Total no of RAM pages */
#define IconPage    10          /* Icon Page number */

#define All0 6          /* 3 for all 0, 4 for all 1    */
#define All1 4
#define iIntRegValue           4 /*Internal Regulator Resistor Ratio Value */
#define iContCtrlRegValue     27 /* Contrast Control Register Value */
#define MSGNo       16
#define MSGLength   22
#define SSLNameNo   4
#define DevicePg    0   //RAM page for showing device name
#define FeaturePg   1   //RAM page for showing feature
#define GRAPHICNo   13
#define xlogo       38
#define ylogo       5
#define xsolomon    91
#define ysolomon    2
#define xsystech    81
#define ysystech    2
#define xlimited    70
#define ylimited    2
#define xcc         16
#define ycc         2
#define xpageq      128
#define ypageq      4
#define horizonal   0
#define d_time      60
```

```
/*************************************************
* C) Global Variable Definition                  *
*************************************************/
unsigned char WC_CSH;
unsigned char WC_CSL;
unsigned char WD_CSH;
unsigned char WD_CSL;
unsigned char RES_CSH;
unsigned char RES_CSL;
unsigned char RD_CSH;
unsigned char RD_CSL;
unsigned char RC_CSH;
unsigned char RC_CSL;

/*************************************************
* D) Hardcoded Graphics and String               *
*************************************************/


 unsigned char code npic1[64] = { 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
                                  0x00, 0x00, 0x00, 0x00, 0x00, 0x80, 0xC0,
                                  0xE0, 0xE0, 0x1F, 0xF0, 0xF8, 0xF8, 0xFC,
                                  0xFC, 0xFC, 0xFE, 0xFE, 0xFE, 0xFE, 0xFF,
                                  0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,
                                  0xFF, 0xFF, 0xFE, 0xFE, 0xFE, 0xFE, 0xFC,
                                  0xFC, 0xFC, 0xF8, 0xF8, 0xF0, 0xF0, 0x1F,
                                  0xE0, 0xC0, 0x80, 0x00, 0x00, 0x00, 0x00,
                                  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
                                  0x00 } ;

 unsigned char code npic2[64] = { 0x80, 0x80, 0x80, 0x80, 0x80, 0x40, 0x60,
                                  0x70, 0x78, 0x7E, 0x7E, 0x7F, 0x7F, 0x7F,
                                  0x7F, 0x7F, 0x80, 0x7F, 0x7F, 0x7F, 0x7F,
                                  0x7F, 0x7F, 0x7F, 0x7F, 0x7F, 0x7F, 0x7F,
                                  0x7F, 0x7F, 0x7F, 0x7F, 0x7F, 0x7F, 0x7F,
                                  0x7F, 0x7F, 0x7F, 0x7F, 0x7F, 0x7F, 0x7F,
                                  0x7F, 0x7F, 0x7F, 0x7F, 0x7F, 0x7F, 0x80,
                                  0x7F, 0x7F, 0x7F, 0x7F, 0x7E, 0x7E, 0x78,
                                  0x70, 0x60, 0x40, 0x80, 0x80, 0x80, 0x80,
                                  0x80 } ;

 unsigned char code npic3[64] = { 0x00, 0x80, 0xF0, 0xFC, 0xFF, 0xFF, 0xFF,
                                  0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,
                                  0xFF, 0xFF, 0x00, 0xFF, 0xFF, 0xFF, 0xFF,
                                  0x7F, 0x1F, 0x1F, 0x0F, 0x07, 0x07, 0x03,
                                  0x03, 0x03, 0x01, 0x01, 0x01, 0x01, 0x01,
                                  0x03, 0x03, 0x03, 0x07, 0x07, 0x0F, 0x1F,
                                  0x1F, 0x3F, 0xFF, 0xFF, 0xFF, 0xFF, 0x00,
                                  0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,
                                  0xFF, 0xFF, 0xFF, 0xFF, 0xFC, 0xF0, 0x80,
                                  0x00 } ;

 unsigned char code npic4[64] = { 0xF8, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,
```

```
                            0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,
                            0xFF, 0xFF, 0x00, 0xFF, 0x1F, 0x03, 0x00,
                            0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
                            0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
                            0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
                            0x00, 0x00, 0x00, 0x01, 0x1F, 0xFF, 0x00,
                            0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,
                            0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,
                            0xF8 } ;

unsigned char code npic5[64] = { 0x1F, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,
                            0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,
                            0xFF, 0xFF, 0x00, 0xFF, 0xFC, 0xE0, 0x80,
                            0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
                            0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
                            0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
                            0x00, 0x00, 0x80, 0xC0, 0xFC, 0xFF, 0x00,
                            0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,
                            0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,
                            0x1F } ;

unsigned char code npic6[64] = { 0x80, 0x81, 0x8F, 0xBF, 0xFF, 0x7F, 0x7F,
                            0x7F, 0x7F, 0x7F, 0x7F, 0x7F, 0x7F, 0x7F,
                            0x7F, 0x7F, 0x80, 0x7F, 0x7F, 0x7F, 0x7F,
                            0x7F, 0x7C, 0x7C, 0x78, 0x70, 0x70, 0x60,
                            0x60, 0x60, 0x40, 0x40, 0x40, 0x40, 0x40,
                            0x60, 0x60, 0x60, 0x60, 0x70, 0x78, 0x78,
                            0x7C, 0x7E, 0x7F, 0x7F, 0x7F, 0x7F, 0x80,
                            0x7F, 0x7F, 0x7F, 0x7F, 0x7F, 0x7F, 0x7F,
                            0x7F, 0x7F, 0x7F, 0xFF, 0xBF, 0x8F, 0x81,
                            0x80 } ;

unsigned char code npic7[64] = { 0x00, 0x00, 0x00, 0x00, 0x00, 0x03, 0x07,
                            0x0F, 0x1F, 0x7F, 0x7F, 0xFF, 0xFF, 0xFF,
                            0xFF, 0xFF, 0x00, 0xFF, 0xFF, 0xFF, 0xFF,
                            0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,
                            0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,
                            0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,
                            0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0x00,
                            0xFF, 0xFF, 0xFF, 0xFF, 0x7F, 0x7F, 0x1F,
                            0x0F, 0x07, 0x03, 0x00, 0x00, 0x00, 0x00,
                            0x00 } ;

unsigned char code npic8[64] = { 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
                            0x00, 0x00, 0x00, 0x00, 0x00, 0x01, 0x03,
                            0x07, 0x07, 0xF8, 0x0F, 0x1F, 0x1F, 0x3F,
                            0x3F, 0x3F, 0x7F, 0x7F, 0x7F, 0x7F, 0xFF,
                            0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,
                            0xFF, 0xFF, 0x7F, 0x7F, 0x7F, 0x7F, 0x3F,
                            0x3F, 0x3F, 0x1F, 0x1F, 0x0F, 0x0F, 0xF8,
                            0x07, 0x03, 0x01, 0x00, 0x00, 0x00, 0x00,
                            0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
                            0x00 } ;
```

```
/************************************************
* E) Function Prototypes                        *
*    SetMode(Mode)                              *
*    delay(n)                                   *
*    resetchip()                                *
*    SingleCmd(i)                               *
*    SingleData(i)                              *
*    SetRAMAddr(Page,Col)                       *
*    ParFillPAGE(Mode,PAGE)                     *
*    ParFillRAM(Mode)                           *
*    SetContrast(Gain,Step)                     *
*    InitDisplay()                              *
*    SendChar(ASCChar)                          *
*    SendMsg(msgoffset)                         *
*    dumppattern(Page,Col,x,y,pat)              *
*    clearRAM(start,stop)                       *
************************************************/

void SetMode(unsigned char Mode)        /* 1 = parallel, 0 = serial */
{
    WC_CSH  = 0x7d;                     //for demoboard MCU
    WC_CSL  = 0x3d;                     /* Parallel mode */
    WD_CSH  = 0x7f;                     /* Master mode */
    WD_CSL  = 0x3f;                     /* 6800 mode */
    RES_CSH = 0xff;
    RES_CSL = 0xef;
    RC_CSH  = 0xfd;                     /* fd */
    RC_CSL  = 0xbd;                     /* bd */
    RD_CSH  = 0xff;                     /* ff */
    RD_CSL  = 0xbf;                     /* bf */
}

void delay(unsigned int n)              /* wait n seconds*/
{
    unsigned int i;
    unsigned int  j;
    for (i=0;i<500;i++)
       for (j=0;j<n*2;j++) { ;}
}

void fdelay(unsigned int n)              /* wait n seconds*/
{
    unsigned int i;
    unsigned int  j;
    for (i=0;i<5;i++)
       for (j=0;j<n*2;j++) { ;}
}
void SingleCmd(unsigned char i)
/* send the value in the accumulator to LCD driver as a command*/
{
    int j;
    controlport=WC_CSL;
```

```
    for (j=0;j<8;j++)
    {
     dataport=(WD_CSL|(i<<j)&0x80);   /* SCK gen & SDA out */
     dataport=(WD_CSH|(i<<j)&0x80);
    }
    controlport=WC_CSL;
    controlport=WC_CSH;
}

void Prog_Reset()
/* send the a reset command*/
{
    int j;
    controlport=RES_CSL;
    for (j=0;j<8;j++)
    {
     dataport=(WD_CSL|(0xff<<j)&0x80);   /* SCK gen & SDA out */
     dataport=(WD_CSH|(0xff<<j)&0x80);
    }
    controlport=RES_CSL;
    controlport=RES_CSH;
}

void resetchip()        // MCU board need to be modified
{                       // cut header pin2 from MCU pin2 and connect to pin7
    SingleCmd(SWRest);
    Prog_Reset();
}



void SingleData(unsigned char i)
/* send the value in the accumulator to LCD driver as a command*/
{
    int j;
    controlport=WD_CSL;
    for (j=0;j<8;j++)
    {
     dataport=(WD_CSL|(i<<j)&0x80);   /* SCK gen & SDA out */
     dataport=(WD_CSH|(i<<j)&0x80);
    }
    controlport=WD_CSL;
    controlport=WD_CSH;
}

void SetRAMAddr (unsigned char Page, unsigned char Col)
{
    unsigned char temp;

    temp = 0x0f & Page;
    SingleCmd(PageAddr | temp);
    temp = 0x0f & (Col >> 4);
    SingleCmd(ColAddrHi | temp);
```

```
    temp = 0x0f & Col;
    SingleCmd(ColAddrLo | temp);
}

void SetContrast(unsigned char Gain, Step) { //set overall contrast
    SingleCmd(IntRegRatio | (0x0f & Gain)); //set internal resistor ratio
    SingleCmd(ContCtrlReg);                 //set Contrast Control Register
    SingleCmd((0x3f & Step));
}

void InitDisplay() {        /* turn on normal display */
    SingleCmd(DisplayOff);
    SingleCmd(SegRemapOff);
    SingleCmd(ComRemapOn);
    SetContrast(iIntRegValue, iContCtrlRegValue); //set default contrast
    SingleCmd(PwrCtrlReg | IntVolBstr | IntReg | OPampBuffer); //turn on booster, regulator & divider
    SingleCmd(DisplayOn);
}


void dumppattern(unsigned char page, col, x, y, unsigned char code *pat)
{
unsigned char i;
unsigned char j;
unsigned int k;

    k=0;
    for (j=0;j<y;j++) {
       SetRAMAddr(page+j,col);
       for (i=0;i<x;i++) {
           SingleData(pat[k]);
           k=k+1;
       }
    }
}

void contrastctrl(unsigned char start,stop)
{
unsigned char i;

    if (start < stop)
    {
       for (i=start; i<stop; i+=1)
       {
           SetContrast(iIntRegValue, i);   //slowly turn on display
           fdelay(80);
       }
    }
    else
    {
       for (i=start; i>stop; i-=1)
       {
           SetContrast(iIntRegValue, i);   //slowly turn off display
```

```
           fdelay(120);
       }
    }
}


/************************************************
* F) Main Function                              *
************************************************/
main()
{
    unsigned char iDispStLn, iRAMPgPtr;
    unsigned char ch1,ch2,mask1,mask2;
    unsigned char i, j;
    unsigned char k, l;

start:
       SetMode(PS);
    resetchip();
       SingleCmd(InterOsc);
       SingleCmd(0x67);
       /***************************/
       /* Show device demo         */
       /***************************/
       InitDisplay();                          // initialize normal display environment
       SingleCmd(DisplayOff);
       SingleCmd(0x56);
       contrastctrl(10,iContCtrlRegValue);
       SingleCmd(0x24);
       iDispStLn=0;
    SingleCmd(DisplayOn);                   // blank display update RAM

    /* Fill Screen with 0x00 */

    for (i=0;i<PageNo;i++)
    {
       SetRAMAddr(i,0);
              for (j=0;j<ColNo; j++) SingleData(0x00);
    }
      SetContrast(iIntRegValue,iContCtrlRegValue);

        dumppattern(1,32,64,1,npic1);       /* Display testing pattern */
        dumppattern(2,32,64,1,npic2);       /* 64 * 64 in size         */
        dumppattern(3,32,64,1,npic3);       /* separated in 8 pages    */
        dumppattern(4,32,64,1,npic4);
        dumppattern(5,32,64,1,npic5);       /* offset in the 2nd row   */
        dumppattern(6,32,64,1,npic6);       /* (page 1) & point 32     */
        dumppattern(7,32,64,1,npic7);       /* to start                */
        dumppattern(8,32,64,1,npic8);
        delay(d_time*5);

    /* Fill Screen with 0x00 */
    for (i=0;i<PageNo;i++)
```

```
    {
        SetRAMAddr(i,0);
        for (j=0;j<ColNo; j++) SingleData(0x00);
    }
      SetContrast(iIntRegValue,iContCtrlRegValue);

 goto start;

}
```