

DVK1815T1-A0

Product Information
SSD1815T1 Development Kit

DVK1815T1-A0 is a demonstration of SSD1815T1 working on a (132 X 64 + 1 icon line) panel. It is intended to help users expedite their design-in of SOLOMON LCD driver.

PACKAGE CONTENTS

DVK1815T1-A0 consists of the following:

- 1) LCD Module (132 X 64 + 1 icon line) (DVM1815T1-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 (PRG1815T1-A0)

SYSTEM REQUIREMENT

DVK1815T1-A0 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

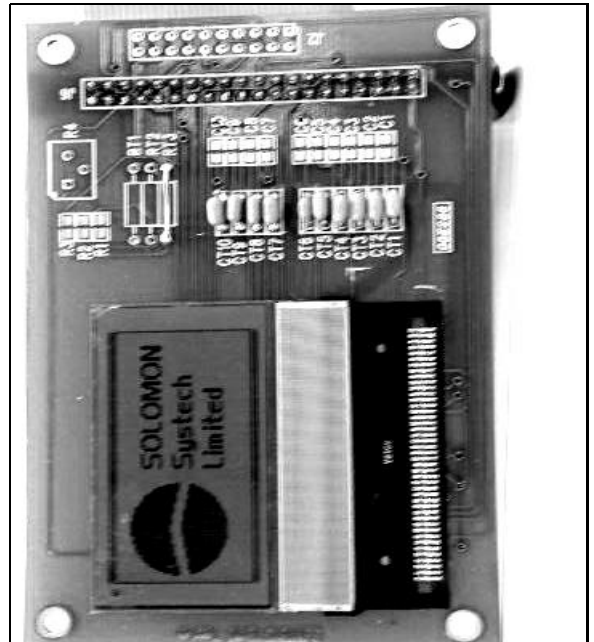


Figure 1 DVM1815T1-A0 outlook

LCD MODULE

The LCD module has been configured as follows

- 1) size: 132 X 64 + 1 icon line
- 2) TAB Package
- 3) 4X DC-DC converter to generate VEE
- 4) Internal feedback resistor which is software programmable is used to generate VL6.
- 5) 6800 8-bit parallel interface.
- 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.

ORDERING INFORMATION

Item	Ordering Part Number
SSD1815T1 Development Kit	DVK1815T1-A0
SSD1815T1 Development Module	DVM1815T1-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	RES	NC	D/C	R/W	CS1	VSS	D0	D1	D2	D3	D4	D5	D6	D7	CE	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 DVM1815T1-A0 pin 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 4MHz. 4 dummy keys are reserved for developing user-interactive application such as tuning contrast.

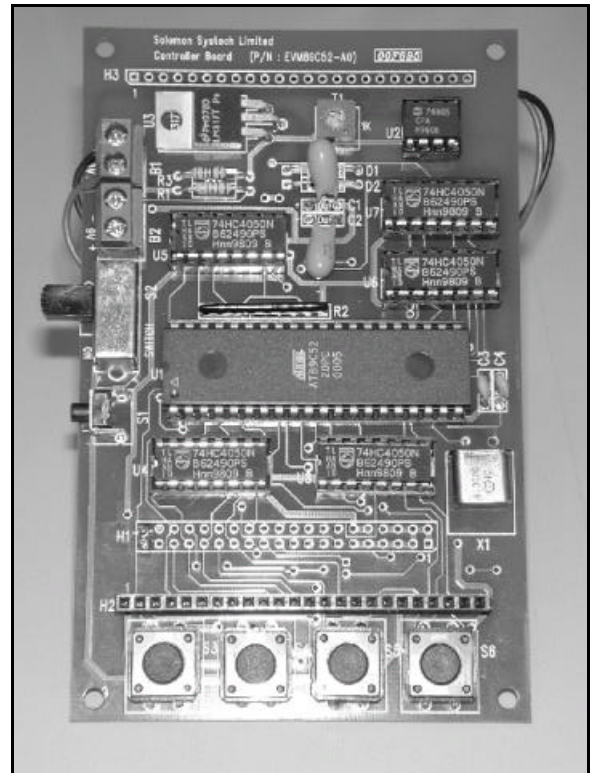


Figure 2 EVM89C52-A0 outlook

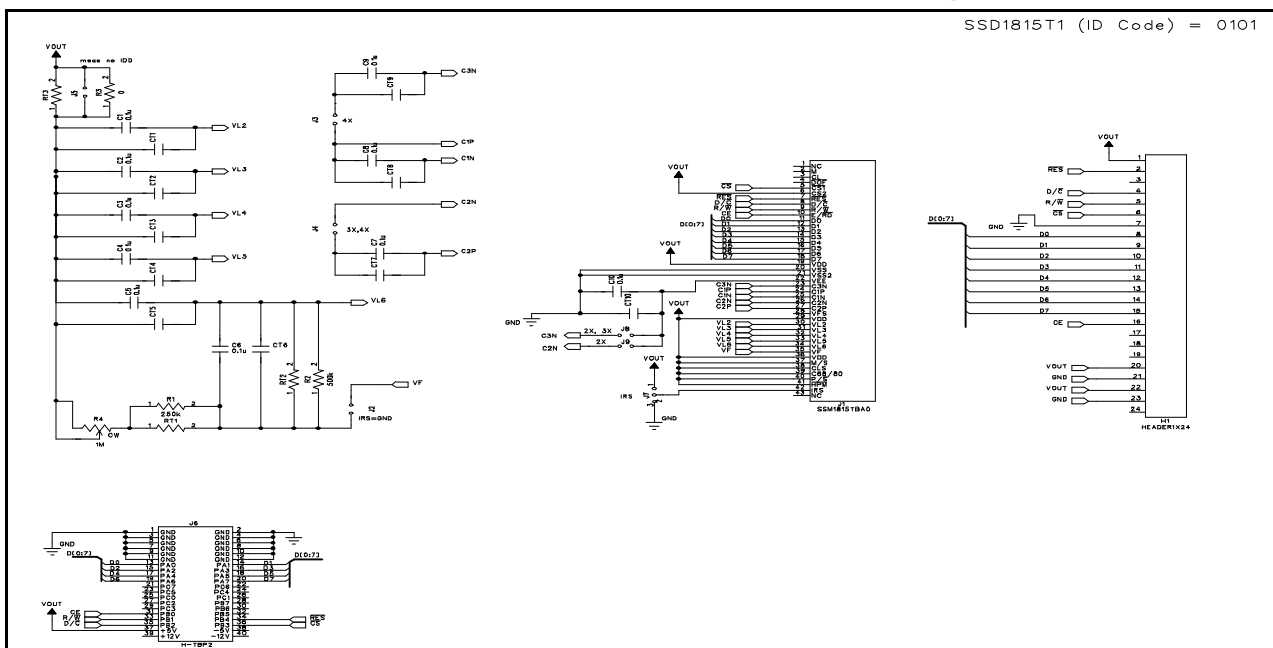


Figure 3 DVM1815T1-A0 schematics

PROGRAMMING NOTE

```
/*
/* Prog: PRG1815T1-A0
/* Device: SSD1815T1
/* Descp: 1815 Pearl Demo
/* LCM: 132X64
/* written by: David Wan Ka Ho / Chu Chi Wa
/* modified by: Tom Tsang
/* Date: 2k0912
/* Structure: A) Hardware Interface
/* B) Command Table per device
/* C) Global Variables Definition
/* D) Hardcoded Graphics
/* E) Function Prototypes
/* F) Main Function
/* Note: For LCM Board(P/N: EVK1815T1-A0) with Controller Board
/* (P/N: EVM89C52A0), the 74HC4050 buffer(U8) should not be inserted.*/
/*
#include "reg51.h"
#define controlport P0
#define dataport P1
/*****
* A) HardWare Interface
*
* PORT A 7 6 5 4 3 2 1 0
* RW CS PS 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
*
*****/
/*****
* 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 OneSixBias 0xA2
#define OneFiveBias 0xA3
#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 IndicateOff 0xAC
#define ContCtrlReg 0x81
#define IndicateOn 0xAD
#define IndicateOff 0xAC

#define CmdMuxRatio 0xA8
#define CmdBiasRatio 0xA9
#define SetBiasQtr 0xAA
#define SetBiasNorm 0xAB
#define DispOffset 0xD3
#define IconModeOn 0xD1
#define IconModeOff 0xD0
```

```

#define PageBlinkg 0xD5

#define Device SSD1815T1 /* device under demo */
#define ColNo 132 /* number of Column/Seg on LCD glass*/
#define RowNo 64 /* number of Row/Com/Mux */
#define PS 1 /* fixed to Parallel mode */
#define PageNo 8 /* Total no of RAM pages */
#define IconPage 8 /* Icon Page number */

#define All0 6 /* 3 for all 0, 4 for all 1 */
#define All1 4
#define iIntRegValue 5 /*Internal Regulator Resistor Ratio Value */
#define iContCtrlRegValue 40 /* 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 16
#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 96
#define ypageq 4
#define horizontal 0
#define d_time 20

/*****
* 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 PAGE0[] = { 0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x80,0x80,0x80,0x80,0x80,0x80,0x80,0x80,0x80,0x80,0x80,0x80,0x80,0x80,0x80,0x80,
0xC0,0xC0,0xC0,0xC0,0xC0,0xC0,0xC0,0xC0,0xC0,0xC0,0xC0,0xC0,0xC0,0xC0,0xC0,0xC0,
0xC0,0xC0,0xC0,0xC0,0xC0,0xC0,0xC0,0xC0,0xC0,0xC0,0xC0,0xC0,0xC0,0xC0,0xC0,0xC0,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00 };

unsigned char code PAGE1[] = { 0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x80,0x80,0xE0,0xF8,0xFC,0xFE,0xFE,0xFF,0xFF,
0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,
0xFE,0xF8,0xF8,0xE0,0xE0,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00 };

```



```

* 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 long 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 long i;
    unsigned int j;
    for (i=0;i<5;i++)
        for (j=0;j<n*2;j++) { ; }
}

void resetchip() // MCU board need to be modified
{ // cut header pin2 from MCU pin2 and connect to pin7
    unsigned long i;

    controlport=RES_CSL;
    for (i=1;i<500;i++);
    dataport = 0xff; /* Fixed for serial mode */
    dataport = 0xff; /* Fixed for serial mode */
    dataport = 0xff; /* Fixed for serial mode */
    for (i=1;i<500;i++);
    controlport=RES_CSH;
}

void SingleCmd(unsigned char i)
/* send the value in the accumulator to LCD driver as a command*/
{
    dataport=i;
    controlport=WC_CSH;
    controlport=WC_CSL;
    controlport=WC_CSH;
}

void SingleData(unsigned char i)
/* send the value in the accumulator to LCD driver as a command*/
{

```

```

    dataport=i;
    controlport=WD_CSH;
    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(SegRemapOn);
    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 clearRAM(unsigned char startpage,stoppage)
{ unsigned char i,j;

    for (j=startpage;j<stoppage;j++) {
        SetRAMAddr(j,0x00);
        for (i=0; i<132; i++) {
            SingleData(0x00);
        }
    }
}

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(100);
        }
    }
    else {
        for (i=start; i>stop; i-=1) {

```

```

        SetContrast(iIntRegValue, i); //slowly turn off display
        fdelay(100);
    }
}

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

start:
    SetMode(PS);
    resetchip();

    /*****/
    /* Show device demo */
    /*****/
    InitDisplay(); // initialize normal display
    SingleCmd(DisplayOff); // blank display update RAM
    clearRAM(0,8);
    contrastctrl(0,iContCtrlRegValue-5);
    dumppattern(0,18,96,8,PAGE0);
    /*****/
    /* Clear IconPage */
    /*****/
    SetRAMAddr(IconPage,0x00);
    for (j=0; j<ColNo; j++) SingleData(0);

    /*****/
    /* Slowly Turn On Diaplay */
    /*****/
    SingleCmd(DisplayOn);
    contrastctrl(0,iContCtrlRegValue);
    delay(d_time*5);

    /*****/
    /* Slowly turn off display */
    /*****/
    contrastctrl(iContCtrlRegValue,0);
    clearRAM(0,8);
    goto start;
}

```

Solomon reserves the right to make changes without further notice to any products herein. Solomon makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Solomon assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters can and do vary in different applications. All operating parameters, including "Typicals" must be validated for each customer application by customer's technical experts. Solomon does not convey any license under its patent rights nor the rights of others. Solomon products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Solomon product could create a situation where personal injury or death may occur. Should Buyer purchase or use Solomon products for any such unintended or unauthorized application, Buyer shall indemnify and hold Solomon and its offices, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Solomon was negligent regarding the design or manufacture of the part.