Experimenting with the MAX6956 and MAX6957 SPI and I2C GPIO and LED Display Drivers from a PC

This application note describes a PC program, downloadable free, to assist evaluation of the MAX6956 and MAX6957 LED display drivers and GPIO (port expanders).

The MAX6956 and MAX6957 are versatile 28-segment LED display drivers which allow any of the outputs to be configured as general purpose inputs/outputs (GPIO). Control is made through a high-speed SPI (MAX6957) or I^{2}C (MAX6956) serial interface.

This application note describes a utility program which allows a MAX6957 or MAX6956 driver to be controlled from a PC. The utility can be used as a standalone to help an engineer become familiar with the registers and functions of the drivers. Moreover, it can be used to 'prove' an application board prototype by directly controlling the MAX6957 or MAX6956 registers before the equipment's software is designed.

Requirements

A PC running Windows 95, 98, 98SE, ME, NT, or 2000 with a parallel printer port configured for either LPT1 or LPT2.

Description

The utility is a Visual Basic 5 program called MAX6957.EXE which requires the standard Visual Basic run time library MSVBVM50.DLL in order to run at all. The program uses the DriverLINX™ freeware parallel port driver DLPortIO.DLL which provides the Win32 DLL hardware I/O functions not available as standard in Visual Basic. Windows NT and 2000 users also require the DLPortIO.SYS kernel mode driver. Both of these drivers are copyright Scientific Software Tools, Inc. (http://www.driverlinx.com). DriverLINX™ is a registered trademark of Scientific Software Tools, Inc.

Installation

To install to a Windows 95, 98, 98SE, ME platform, download the MAX6956-57.EXE file. This is a WinZIP self-extracting archive that contains ReadMe.txt, ReadMeSST.txt, MAX6957.EXE, DLPortIO.DLL, and MSVBVM50.DLL. The default download directory is C:\MAX6956. MSVBVM50.DLL may be deleted if the library is already registered on the computer.

To install to a Windows NT or 2000 platform, download the MAX6956-NT.EXE file. This is a WinZIP self-extracting archive that contains ReadMe.txt, ReadMeSST.txt, MAX6956.EXE, PORT95NT.EXE, and MSVBVM50.DLL. The default download directory is C:\MAX6956. PORT95NT.EXE is the install program for the DriverLINX drivers which installs and registers the DLPortIO.DLL library and DLPortIO.SYS driver. PORT95NT.EXE can be deleted after installation. Windows 95, 98, 98SE, ME users can also use this installation procedure if they wish.

Source Code

The source code for this application note is available at http://www.maxim-ic.com/products/display/software/.

Connecting a MAX6956 or MAX6957 to the Parallel Port

This utility uses 3 of the 8 parallel port printer output lines to simulate SPI serial interface activity, and another 2 output lines to simulate I^{2}C serial interface activity. Parallel port control input lines are used to read back SPI data, I^{2}C data, and interrupt status. Multiple (up to 16, daisy-chained) MAX6957s and/or multiple (up to 16, configured with different I^{2}C addresses) MAX6956s may be operated at once. Either the LPT1 or LPT2 port may be selected from the software. The port can be a standard, ECP, or EPP type. The port connections are shown in Figure 1.
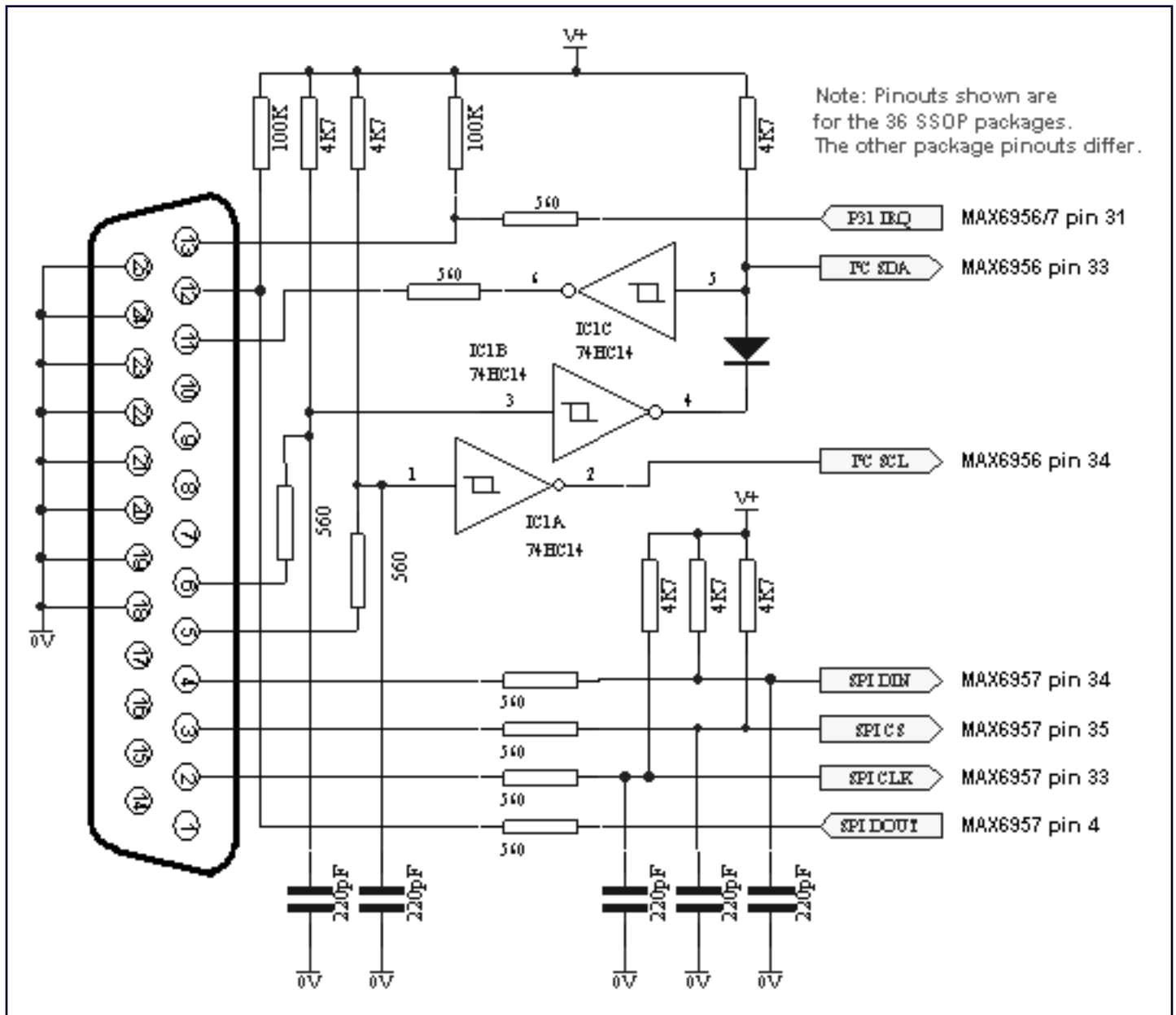
Figure 1. The MAX6956 and MAX6957 connections to the parallel port.

The program brings up 4 windows on startup. These are the main control, intensity control, I/O control, and segment control windows. The register settings on start-up mirror the register power-up conditions of the MAX6956 and MAX6957.

The main control is shown in Figure 2 below. The 'Driver type' radio buttons select whether the software will transmit to MAX6957 (using the SPI connection) or the MAX6956 (using the I2C connection).
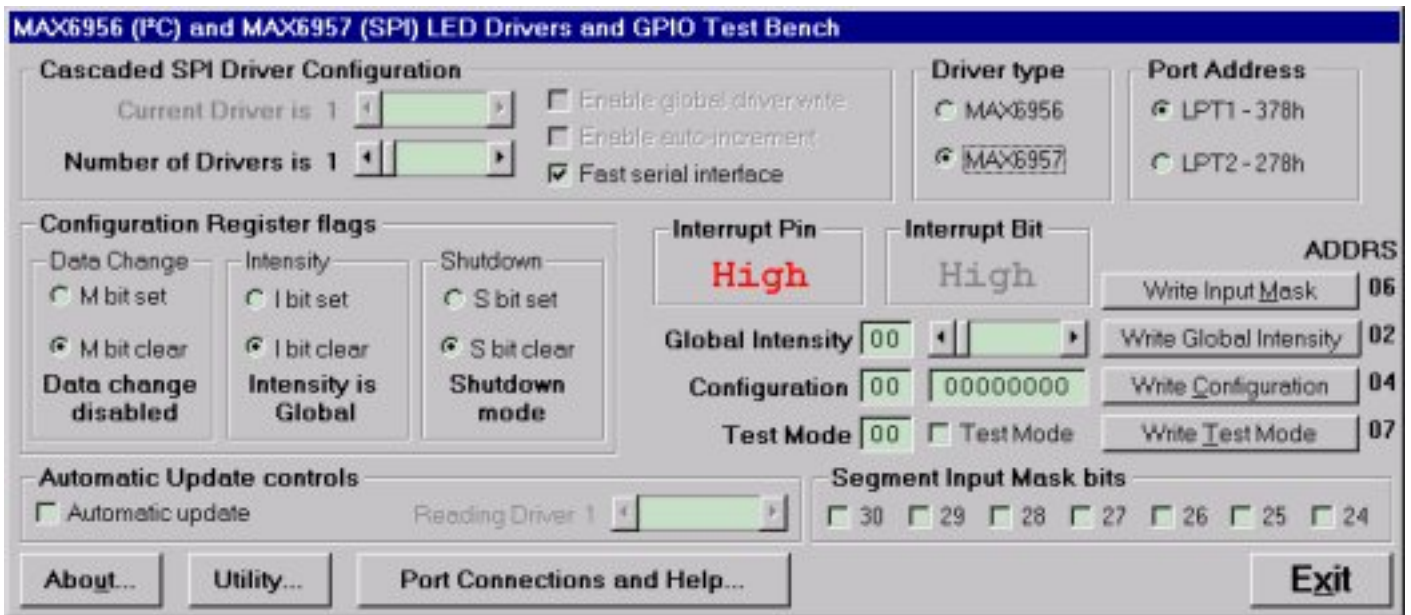
Figure 2. Program display on startup

In SPI mode, the software controls up to 16 MAX6957 drivers. The drivers are presumed to be cascaded, (i.e., the DOUT pin of the first MAX6957 connects to the DIN pin of the second MAX6957, whose DOUT pin connects to the DIN pin of the third MAX6957, and so on). The total number of MAX6957 devices is set by the 'Number of Drivers' slider. When this is set to more than 1, the 'Enable global driver write' and 'Enable auto-increment' check boxes are available. When 'Enable global driver write' is clear, only the MAX6957 driver selected by the 'Current Driver' slider is written to when a write command is selected - the others receive the no-op instruction. When 'Enable global driver write' is checked, all the MAX6957 drivers are written to with the same data. When 'Enable auto-increment' is checked, the current driver number is automatically incremented after each write action. This allows the user to quickly send the same data for a series of MAX6957 drivers.

The SPI interface connection to the parallel port can be tested with the 'Test Stream' facility which can be found under 'Port Connections and Help...' when the program is running. The 'Test Stream' facility transmits the no-op instruction continuously to the MAX6957(s) (as set by the 'Number of Drivers' slider) allowing the interface connections to be verified without affecting register contents.

In I²C mode, the software controls up to 16 MAX6956 drivers. The first driver is presumed to be set to address 1000000x, with the addresses of subsequent devices increasing to 1001111x for the last device. To access, for example, a single MAX6956 driver at address 1001111x, simply set the 'Number of Drivers' slider to 16 and then the 'Current Driver' slider to 16, address 1001111x. The main form in I²C mode is shown in Figure 3.
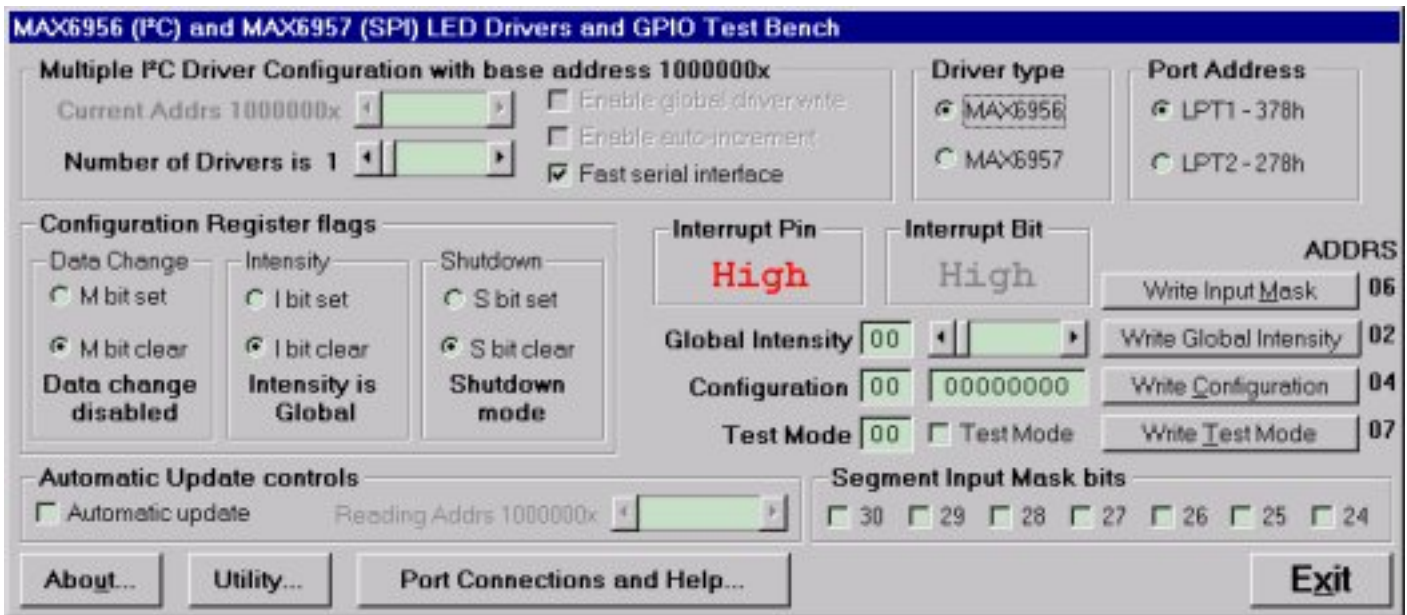
Figure 3. I²C program display

The I2C interface connection to the parallel port can be tested with the 'Test Stream' facility which can be found under 'Port Connections and Help...' when the program is running. The 'Test Stream' facility transmits the no-op instruction continuously to all MAX6956(s) (as set by the 'Number of Drivers' slider) allowing the interface connections to be verified without affecting register contents.

The 'Fast serial interface' check box sets the serial interface speed for both SPI and I2C interface types. When checked, serial interface runs at machine-dependent full speed up to 1 Mbits/sec; unchecked, the speed is limited to 500 bits/sec maximum. The slow speed may be useful when connecting over very long cables.

The program will normally only communicate with the target driver (or drivers) when a 'Write' or 'Read' button is pressed. However, the program attempts to monitor an interrupt output pin presumed to be connected to the parallel port pin 13. This is displayed on the main window and displays status as High or Low.

The 'Automatic Update control' button, when checked, makes the program continually poll all 28 port input registers and the interrupt register bit. The reading rate depends on the setting of the 'Fast serial interface' check box, and is many times a second (Fast) or every many seconds (Slow). Note: slow mode is *very* slow! The 'Reading driver x' slider allows the user to set the device being automatically read independently to the driver(s) accessible through the 'Current driver' control.

The main window is the route to Exit the program. The Exit button, or pressing the Esc key on the main window, closes all 4 windows. Pressing the Esc key in one of the other 3 windows takes the user to the main window; therefore, pressing the Esc key twice will close the program

quickly. Before closing, the program stores the last settings for device configuration in a file called MAX6956.ini in the same directory that the program was executed from. Nothing is written to Windows registry. The settings includes the positions of the 4 windows, and the interface settings on the main window. If the MAX6956.ini doesn't exist then it is created; otherwise it is overwritten. To restore program default settings, or if the MAX6956.ini is corrupted, simply delete the MAX6956.ini file.

Figures 4, 5, and 6 show snapshots of the other three windows.

Figure 4. Intensity configuration window

Figure 5. I/O configuration window

Segment Registers

REGISTER	WRITE	ADDRS	WRITE	ADDRS	READ	READ	ADDRS	READ	ADDRS
Segment 0 `0`	Write Seg 0	20			?	Read Seg 0	A0	Read 7 - 0	C0
Segment 1 `0`	Write Seg 1	21			?	Read Seg 1	A1	Read 8 - 1	C1
Segment 2 `0`	Write Seg 2	22			?	Read Seg 2	A2	Read 9 - 2	C2
Segment 3 `0`	Write Seg 3	23		ADDRS	?	Read Seg 3	A3	Read 10 - 3	C3
Segment 4 `0`	Write Seg 4	24	Write 11 - 4	44	?	Read Seg 4	A4	Read 11 - 4	C4
Segment 5 `0`	Write Seg 5	25	Write 12 - 5	45	?	Read Seg 5	A5	Read 12 - 5	C5
Segment 6 `0`	Write Seg 6	26	Write 13 - 6	46	?	Read Seg 6	A6	Read 13 - 6	C6
Segment 7 `0`	Write Seg 7	27	Write 14 - 7	47	?	Read Seg 7	A7	Read 14 - 7	C7
Segment 8 `0`	Write Seg 8	28	Write 15 - 8	48	?	Read Seg 8	A8	Read 15 - 8	C8
Segment 9 `0`	Write Seg 9	29	Write 16 - 9	49	?	Read Seg 9	A9	Read 16 - 9	C9
Segment 10 `0`	Write Seg 10	2A	Write 17 - 10	4A	?	Read Seg 10	AA	Read 17 - 10	CA
Segment 11 `0`	Write Seg 11	2B	Write 18 - 11	4B	?	Read Seg 11	AB	Read 18 - 11	CB
Segment 12 `0`	Write Seg 12	2C	Write 19 - 12	4C	?	Read Seg 12	AC	Read 19 - 12	CC
Segment 13 `0`	Write Seg 13	2D	Write 20 - 13	4D	?	Read Seg 13	AD	Read 20 - 13	CD
Segment 14 `0`	Write Seg 14	2E	Write 21 - 14	4E	?	Read Seg 14	AE	Read 21 - 14	CE
Segment 15 `0`	Write Seg 15	2F	Write 22 - 15	4F	?	Read Seg 15	AF	Read 22 - 15	CF
Segment 16 `0`	Write Seg 16	30	Write 23 - 16	50	?	Read Seg 16	B0	Read 23 - 16	D0
Segment 17 `0`	Write Seg 17	31	Write 24 - 17	51	?	Read Seg 17	B1	Read 24 - 17	D1
Segment 18 `0`	Write Seg 18	32	Write 25 - 18	52	?	Read Seg 18	B2	Read 25 - 18	D2
Segment 19 `0`	Write Seg 19	33	Write 26 - 19	53	?	Read Seg 19	B3	Read 26 - 19	D3
Segment 20 `0`	Write Seg 20	34	Write 27 - 20	54	?	Read Seg 20	B4	Read 27 - 20	D4
Segment 21 `0`	Write Seg 21	35	Write 28 - 21	55	?	Read Seg 21	B5	Read 28 - 21	D5
Segment 22 `0`	Write Seg 22	36	Write 29 - 22	56	?	Read Seg 22	B6	Read 29 - 22	D6
Segment 23 `0`	Write Seg 23	37	Write 30 - 23	57	?	Read Seg 23	B7	Read 30 - 23	D7
Segment 24 `0`	Write Seg 24	38	Write 31 - 24	58	?	Read Seg 24	B8	Read 31 - 24	D8
Segment 25 `0`	Write Seg 25	39	Write 32 - 25	59	?	Read Seg 25	B9	Read 32 - 25	D9
Segment 26 `0`	Write Seg 26	3A	Write 33 - 26	5A	?	Read Seg 26	BA	Read 33 - 26	DA
Segment 27 `0`	Write Seg 27	3B	Write 34 - 27	5B	?	Read Seg 27	BB	Read 34 - 27	DB
Segment 28 `0`	Write Seg 28	3C	Write 35 - 28	5C	?	Read Seg 28	BC	Read 25 - 28	DC
Segment 29 `0`	Write Seg 29	3D	Write 36 - 29	5D	?	Read Seg 29	BD	Read 26 - 29	DD
Segment 30 `0`	Write Seg 30	3E	Write 37 - 30	5D	?	Read Seg 30	BD	Read 37 - 30	DE
Segment 31 `0`	Write Seg 31	3F	Write 38 - 31	5D	?	Read Seg 31	BF	Read 38 - 31	DF

Set and write all segments low	Set and write all segments high	Read all segments

Figure 6. Segment registers window

More Information

MAX6956: QuickView -- Full (PDF) Data Sheet -- Free Samples

MAX6957: QuickView -- Full (PDF) Data Sheet -- Free Samples