

Design and Implementation of 7-Segments-Board 1.0

Basic Extension Module for Embedded System Prototype

By:

Ivan Christian (e-mail: chris.ivan@gmail.com)

Erwin (e-mail: erwin@dsp.itb.ac.id)

Introduction - When developing embedded system, it would be helpful if we could have a module for monitoring purpose. For instance, let say that your system has to process data streams. In testing and verification step, we need to compare each input and output bytes, so that we can verify that your system is doing right. Having a monitor module is surely a great help for engineers. Some development board has monitor module integrated, such as Altera UP1X FPGA Development Board. Inspired by its usage and benefit of such monitor, 7-Segments-Board 1.0 is designed as an extension module for embedded system prototype.

1. Description

7-Segments-Board 1.0 is a low-cost low-power MCU extension module for monitoring purpose. Its aim is to help engineers doing the firmware testing and debugging on hardware prototype. For those who build embedded system prototype device from scratch and do not have access to sophisticated debugging instruments, using this module would make testing and verification process less painful. 7-Segments-Board 1.0 is designed for 8-bit microcontroller system.

2. Module Specifications

The module specifications are as follows:

1. Input:

- General purpose push-button (PB) switches (dry contact).

2. Output:

- Seven-segments LED to display 8-bit data in hex format at minimum.

3. Low power consumption

Driving LED continuously consumes much amount of power. So it is desirable to minimize power loss in limiting resistors.

4. Minimum port usage

7-Segments-Board 1.0 is designed to work well with minimum I/O pin resources, as I/O ports are strictly limited in common embedded system.

3. Block Diagram

Figure 3.1 shows the block diagram of 7-Segments-Board 1.0. In general the module consists of decoders, LED drivers, 7-segments LED display, serial-in parallel-out (SIPO), and PB switches.

We only allocated 1 MCU Port (8 pins + 2 power lines) for module interfacing. SIPO is used to minimize port pins. The minimum total number of pin for driving SIPO is 3 pins only.

In order to drive 7-segments LED display with small power, decoder and SIPO works together to multiplex the display. It takes input signal frequency of 20-25 Hz (i.e. 40-50 ms period) at minimum to flash the display on and off so that human eyes don't see the flicker. The number of 7-segments display depends on MCU I/O port

availability and it sizes up the decoder as well. In this module two 2-digit-in-1-package of 7-segments LED components are used. It means this module can display 4 digits at once. Therefore 2-to-4 decoder is used. Total number of pin for decoder is 2 pins.

PB switches is integrated in the module for any purposes, such as to switch the display from one mode to another. The rest pin available (3 pins) is used for these switches.

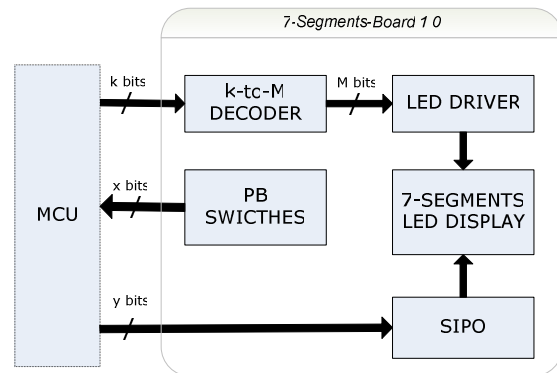


Figure 3.1 Block diagram of 7-Segments-Board 1.0

4. Tools

Primary tools used for design and implementation of 7-Segment-Board 1.0 are as follows:

1. Hardware

- AVR ISP Development Board (designed by Erwin)
- Atmel ATmega8535

2. Software

- Altium Design Explorer Version 7.2.85
- WinAVR 20050214

5. Schematic Capture

See schematic file (SevenSeg_Sch_04172006 - .SchDoc) in the attached zip file. Notice that Output Enable pin (N_OE) and Master Reset pin (N_MR) is tied to VCC and GND respectively. Therefore total number of pin to control SIPO is reduced from 5 pins to 3 pins.

6. PCB Layout Capture

See PCB file (SevenSeg_Pcb_04182006.PcbDoc) in the attached zip file. Note that physical board is implemented using 2-layer PCB with through hole. **Figure 6.1** is the picture of physical board. Pin assignments for interfacing with MCU is shown in **Figure 6.2**. **Tabel 6.1** gives us list of module components.

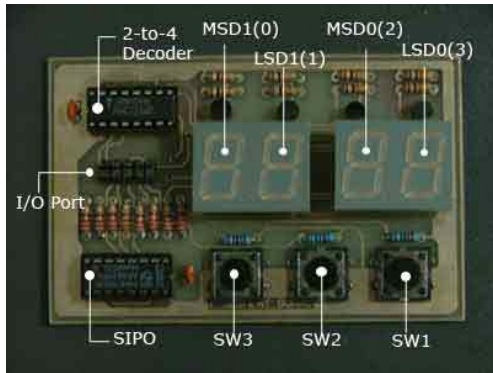


Figure 6.1 Physical Board of 7-Segments-Board 1.0

DEC0	STCLK	SW1	SW3	GND
DEC1	DSER	SHCLK	SW2	VCC

Legend:

DEC1 (pin 1) = decoder input (msb)	SW1 (pin 6) = PB switch 1
DEC0 (pin 2) = decoder input (lsb)	SW2 (pin 7) = PB switch 2
DSER (pin 3) = SIPO serial data	SW3 (pin 8) = PB switch 3
STCLK (pin 4) = SIPO store clock	VCC (pin 9) = +5V Power line
SHCLK (pin 5) = SIPO shift clock	GND (pin10) = Ground line

Figure 6.2 Pin configuration of I/O Port

Table 6.1 List of Primary Components

Component Name	Units	Notes
2-digit-in-1-package 7-segments LED	2	Display units
74HCT595	1	3-States SIPO Buffer
74HC139	1	2-to-4 Decoder
2N2907	4	PNP LED driver
PB Switch	3	General purpose
C 100nF Ceramic	2	Decoupling C
R220 1/2W	8	LED current limiting R
R10K 1/2W	8	LED driver bias circuit
R1K 1/2W	3	Pull up R for switches

7. Firmware Design

This project use Atmel AVR family microcontroller for driver implementation. No particular reason but tools availability that makes the decision. The module driver, written in C, consists of 4 separate files to simplify the maintenance: main.c, port.h, sevenseg_drv.c, and sevenseg_drv.h. The main.c file is the main program that is used to test and verify the driver. The

port.h and sevenseg_drv.h files are self-explanatory. The sevenseg_drv.c contents 3 functions that do display multiplexing, that is:

1. void init_sevenseg(void)

This function initiates data direction register (DDRx) and Data Register (PORTx) of AVR ports used. PORTA is used for module interfacing, and PORTC is used for debugging purpose.

2. void send_data(uint8_t data)

It sends 8-bits data in MCU register serially to SIPO. Each bit is transmitted consecutively from MSB to LSB along with clock signals (SHCLK, STCLK). SHCLK is used to shift bit serially, and STCLK is used to pass the shifted data into SIPO output register. Data can only be seen at SIPO output register after SIPO receives STCLK pulse. Thus it takes 8 pulse of SHCLK and 1 pulse of STCLK to transmit 8-bit data from MCU to SIPO.

3. void mx_display(uint8_t MSD1, uint8_t LSD1, uint8_t MSD0, uint8_t LSD0)

This function is responsible to multiplex display digits. Each digit is named and coded after its position. As seen on Figure 7.1, from right to left, MSD1, LSD1, MSD0, and LSD0 is coded 0, 1, 2, and 3, respectively. When this function is called, each digit would be flashed (based on the parameters entered) one by one from code 0 to code 3. Three main steps are used as follows:

- It selects which digit is activated by controlling the decoder.
- Then the program sends the parameter to SIPO by function send_data(uint8_t data).
- After short delay, the program erases the displayed digit before it activates the next digit, so the next digit to come doesn't shadow the last active digit.

The steps above are repeated after all digit are flashed.

8. Testing and Verification

To do testing and verification, a simple program was made in main.c file. A LED board is used in PORTC as debugging tools. Program first initializes module and supporting peripheral. **Figure 8.1** shows that each point of available digits would be lighted after program finishes the process. LED board would shows 0x7E.



Figure 8.1 After Initialization

Then it would detect the press of one of three integrated PB switches. If SW1 is pressed, LED board would show 0x01, and none is displayed on the module (**Figure 8.2.a**). Module would display number "0123" after SW1 is released (**Figure 8.2.b**).



(a)



(b)

Figure 8.2 Display (a) when SW1 is pressed, (b) when SW1 is released

When SW2 is pressed LED board would show 0x02 (Figure 9.3.a) and module would display “4567” (Figure 8.3.b).



(a)



(b)

Figure 8.3 Display (a) when SW2 is pressed, (b) when SW2 is released

If SW3 is pressed, LED board would show 0x03 (Figure 8.4.a), and module would display “89ab” (Figure 8.4.b). If no switch is pressed, LED board would show 0x7E, and module would display the last number.



(a)



(b)

Figure 8.4 Display (a) when SW3 is pressed, (b) when SW3 is released

Doing multiplexing influences the brightness of the display. Figure 8.5 shows the brightness of the display in a dark room. When the surrounding is dark, we can easily see the display, but in a bright the display is not as contrast as it is in the dark place. The reading is even more difficult with sunlight. Note that the number “1” seems to be brighter than the others, which means greater current flows through digit LSD0. The possible cause is the PNP LED drivers are not all identical.



Figure 8.5 Display brightness in dark room

9. Portability

The source can be translated to other MCU (8051, PIC) with minor modification.

10. Conclusion

From this project, we can conclude several points as follows:

1. Multiplexing display is less bright than continuous display, but the power loss is reduced.
2. By multiplexing the display in frequency greater than 25Hz, we can avoid our eyes seeing flickering display.

11. Advices

To improve 7-Segments-Board 1.0, it is recommended to lower the value of current limiting resistors of the display to less than 220 ohms.

References

- [1] 74HC/HCT139 Dual 2-to-4 line Decoder / Demultiplexer – datasheet. Philips Semiconductors. September 1993
- [2] 2N2907 PNP General Purpose Amplifiers and Switches – datasheet. SGS-Thomson Microelectronics. November 1997
- [3] 74HC/HCT595 8-bit Serial-in/Serial or Parallel-out Shift Register with Output Latches; 3-state – datasheet. Philips Semiconductors. June 1998
- [4] ATmega8535 8-bit AVR Microcontroller with 8 Kbytes In-System Programmable Flash – Preliminary datasheet. Atmel Corporation. June 2004