LCD Board Functions

Contents

0.1 Display . . . 1
0.1.1 Display formats . . . 1
0.1.2 Data Inputs . . . 2
0.2 Programming through This Board . . . 3
0.2.1 Silicon Labs Programming ("C2" Protocol); Dallas Programming (UART link) . . . 3
0.3 UART Links . . . 4
0.3.1 Dallas ("Big Board") UART link . . . 4
0.3.2 SiLabs UART Link . . . 4
0.4 Schematic . . . 6

REV 0; December 23, 2016.

0.1 Display

With no input connections, all inputs are likely to float *high*. So, one could make no connections to inputs while doing an initial test.

0.1.1 Display formats

Try the several slide switches that select display formats.

Figure 1: Slide switches select display options

Here's what the several display choices mean:

Format: hexadecimal versus binary

Hex is much more compact and therefore easier to understand at a glance. It should be your default display mode.

Binary is good when you want to see bit patterns at work: watching a counter count Up versus Down, for example. Also very useful when the Hex display looks wrong (say, the count sequence goes "A, B, E, F;" not at all obvious what's wrong. But the binary sequence, "1010, 1011, 1110, 1111" shows that bit 1 is stuck High).

Data: this switch determines whether the *data* display (on the lower, second line) is 8-bits or 16-bits wide. The 8-bit choice is normal for the "Big Board" microcomputer, but occasionally, even with that computer, we need a 16-bit display in order to show two bytes put out in two successive output operations.

An alternative use for 16-bit data might be to show the output of, say, a 16-bit hard-wired counter.

Note that the *data* display (whether 8 or 16 bits) differs from the *address* display in that a *transparent* latch stands between input and display. So, data can be *latched*, whereas the address display always is continuous (‘live”).

Lines: selects whether to show *two* lines or *one*. *Two* shows first and second lines; *One* shows only the second line.

Labels: Selecting *Yes* shows the word “Address” to the left of first line (in hexadecimal mode), and the word “Data” to the left of the value shown on the second line.

This switch has no effect on the display when format is *binary*, because the wide binary display leaves no room for such a label.

Selecting ’No’ suppresses the labels. Suppressing these makes sense when the display is used for a purpose other than showing address and data of the ‘Big Board’ computer. For example, one might want to show a 16-bit counter output on the second line. The word “data” would be misleading in that application.

0.1.2 Data Inputs

0.1.2.1 Latching

The two 8-bit data latches are controlled by the two latch *enables* shown in fig. 3, labelled “LATCH_EN,” “BYTE LOW” and “BYTE HIGH.” The “LOW” and “HIGH” designations refer to the two data connectors at J3, as shown in fig. 2:

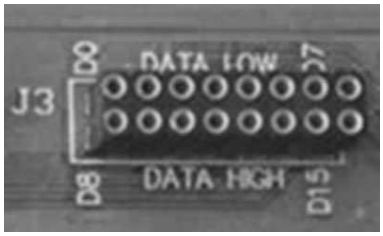

Figure 2: Data input connectors, associated with two “LATCH_EN” signals

These two 8-bit transparent latches normally are held in *transparent* mode (by a pull-down resistor of 100k, on each of the active-low control inputs). So, you may well find yourself forgetting that the latches are present. They operate as latches only when you apply a signal that overrides the effect of the pull-down resistor.

0.1.2.2 Data Multiplexing

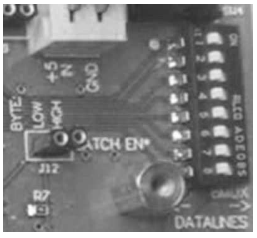

Figure 3: Latch and Multiplexer controls, for Data inputs

The 8-position DIP switch shown at the right side of fig. 3 determines whether the two 8-bit data latches are fed independently from the two separate connectors of fig 2 (DATA HIGH, DATA LOW at connector J3) or,

instead, both are fed from the "DATA LOW" connector at J3. To feed from one input, choose "MUX"; to feed from two independent inputs, choose "16".

The latter ("multiplexed") scheme may at first seem perverse (why short two set of inputs together?). But this merging of two sets of inputs is not perverse but instead is necessary when the signal source is, like the bus of our 8-bit computer, a time-shared or "multiplexed" resource. The computer can put out only one byte at a time; but in two passes it can load first one data register, then the other. To take advantage of this multiplexing option, the program "16_SUM_02.a51" in lab 20 writes first to Port 0, then to Port 1.

The DIP switch determines whether the two registers are fed from *one* input ("MUX") or from *two* ("16"). The Big Board program 16_SUM_02.a51 uses the "MUX" setting; the equivalent program for SiLabs, keysum_16bit.a51 uses the "16" setting. The SiLabs program does that because that controller presents its 16-bit result on two byte-wide ports.

0.2 Programming through This Board

The USB connection can provide power to the board, but need not. A switch at the top right corner selects the source of board power: *USB* (the choice you'll want to use whenever a USB line is connected) versus *EXT*. The latter source is provided for the instances when the board is used simply as a display device, not for linking to an eternal computer (a laptop, ordinarily). When plugged into the big green breadboard (through a DIN connector), the LCD board is powered from the breadboard. In other cases, the LCD board can take power through either of two "+5 IN" connectors near the top right corner of the board.

The USB signal can be routed to either of two translator IC's: one talks to the SiLabs controller; the other can talk to the Dallas controller. A DPDT slide switch selects the path that is connected to the USB signals:

Figure 4: Slide switch selects whether USB talks with SiLabs or Dallas controller

"C2" selects the SiLabs path ("C2" is the name for SiLabs proprietary communications protocol).

0.2.1 Silicon Labs Programming ("C2" Protocol); Dallas Programming (UART link)

To use SiLabs programming functions, slide the switch of fig. 4 to the *C2* position. Connect USB cable to LCD board.

To protect against accidental re-programming of the display microcontroller (the device that controls the LCD), make sure the slide switch in fig. 6 on the next page is in the NORM position, not REPROG.[1]

[1] We have provided *double* protection against accidental reprogramming: to reprogram the USB's display controller you need also to install two shorting jumpers just to the right of the REPROG switch.

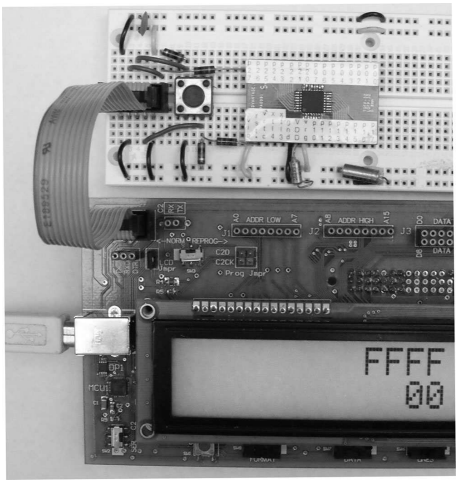

Figure 5: SiLabs programming: done through 10-line jumper cable

Connect the flat ribbon cable between the top-left socket on the board and a C8051F410 controller, as in fig.5. If you want, you can test whether the SiLabs IDE on a laptop can *connect*, and when it does you can load a little test program. bitflip.a51, which blinks an LED would do.

0.3 UART Links

The LCD board includes *two* USB-to-UARTs.[2]

0.3.1 Dallas ("Big Board") UART link

One of these UARTs, used by the Dallas controller, we adopt only to load code into the Big Board computer. The three lines used by that UART appear near the top left corner of the board, as shown in fig. 6, and are also wired directly (through the board's DIN connector) to the big green breadboard's headers that are labelled the same way: TX, RD, DTR. Those three lines must be wired, on the Big Green board, to RX, TX and LOADER*, respectively (note that "TX" goes to RxD on the 8051, "RX" to TxD).

Note, also, that the programming protection switch shown in fig.6 must be kept in the left-hand position, labelled "NORM."

0.3.2 SiLabs UART Link

The other UART, used by the SiLabs controller, we apply so as to permit communication between a laptop and the controller. These two programs appear in Lab 24L.2.5, pp. 986 and following. The two lines used by this UART (only *two* because SiLabs does not use DTR) appear to the right of the header for the 3 Dallas UART lines.

[2] "UART" is an acronym for the traditional RS232 serial protocol: "Universal Asynchronous Receiver/Transmitter." This protocol is described in classnotes 24, pp. 966 and following.

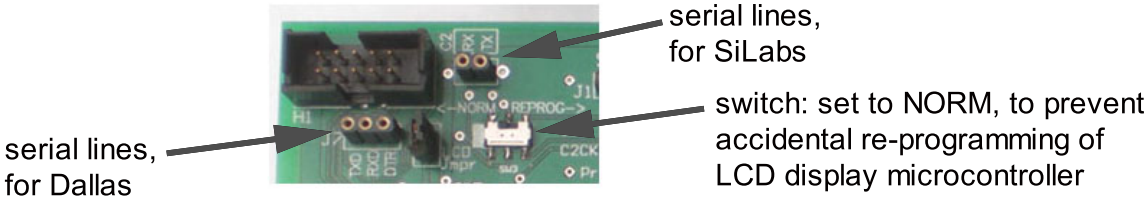

Figure 6: Serial port connections: one set for SiLabs, another for Dallas

Note that when using the LCD board with the "big green breadboard," you need not wire the UART connection points that are shown in fig.6, because those lines are connected to Big Green through the DIN connector, as we said in §0.3.

0.4 Schematic

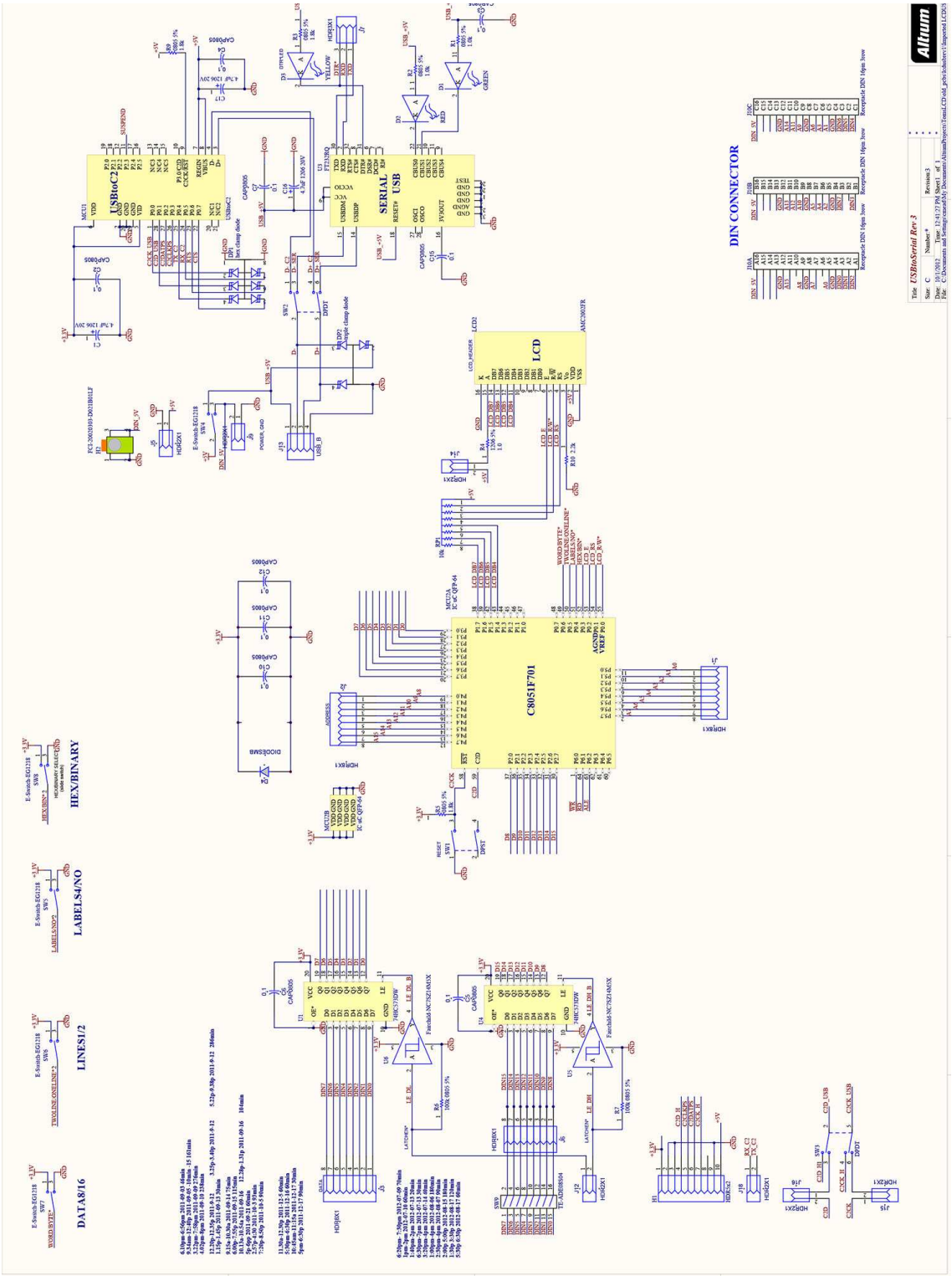

Figure 7: LCD board schematic