LCD Debugging & NAND's 8-bit Bus

ENEE 245: Digital Circuits and Systems Laboratory, Fall 2014
Lab 10

Objectives

The objectives of this laboratory are the following:

- To work with a full 8-bit DQ bus
- To use the LCD on the FPGA board to observe the 8-bit bus as hexadecimal characters

In this lab you will use an LCD Driver that we have prepared for you, familiarize yourself with it and integrate it into your controller design, and then use it to observe the operation of your circuit. You will extend your 2-bit bus to be compliant with the 8-bit NAND Flash data bus.

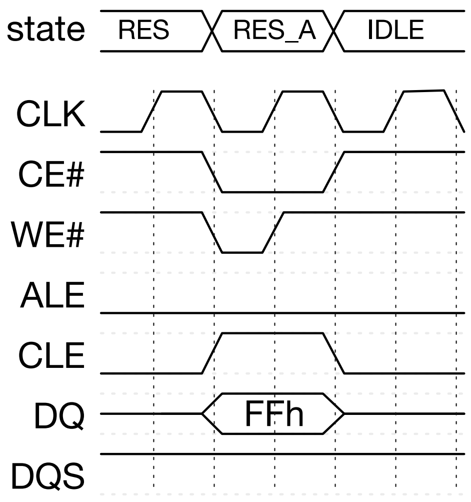
Pre-Lab Preparation

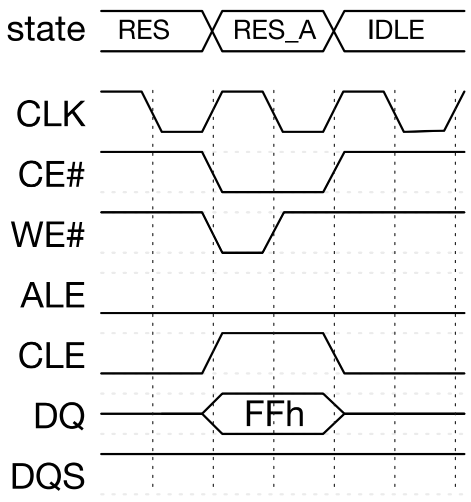
8-bit DQ Bus

The following timing diagrams show the updated behavior, which is more NAND-flash compliant. Note several things: the command codes have changed, and the page-program command has a final command state at the end that indicates to the memory device to actually write the data into the flash array, after it has received all of it over the bus. This latter change will require a non-trivial modification to your state machine.

Reset Command

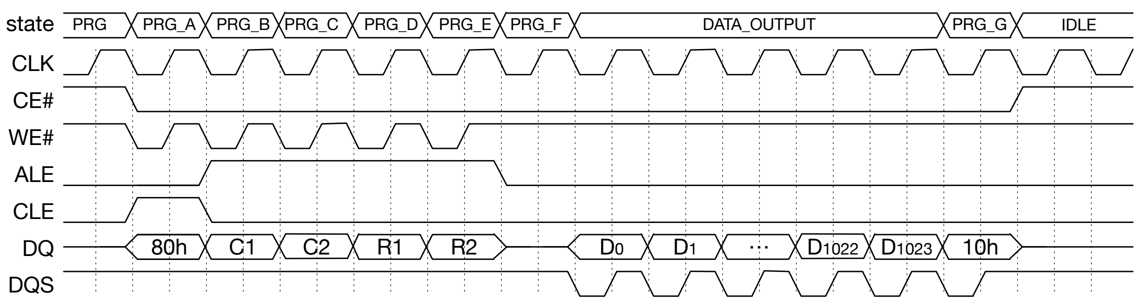
The timing for the *Reset* command is given below, in two variations, depending on whether you want to use the rising or falling edge of the clock to drive your state machine.

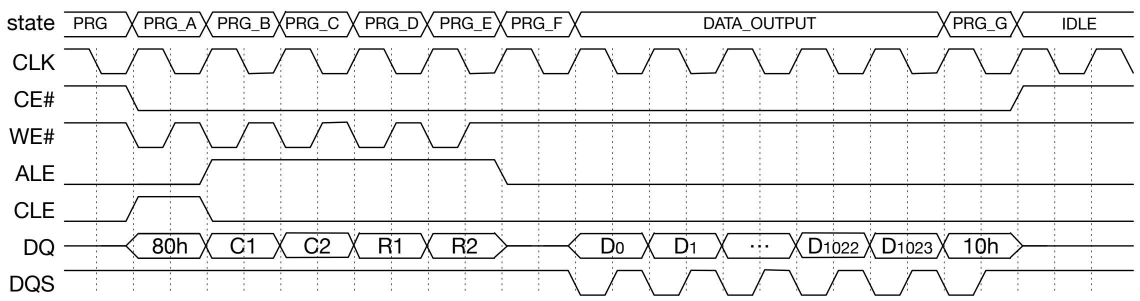




Page Program Command

The timing for the *Page Program* command is given below, in two variations, depending on whether you want to use the rising or falling edge of the clock to drive your state machine.

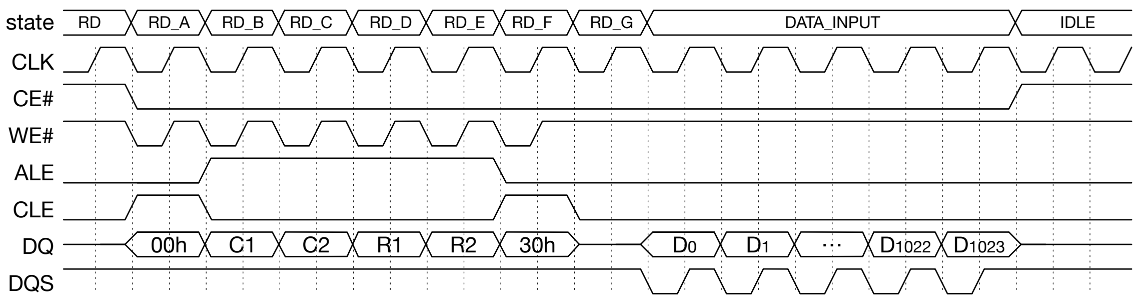


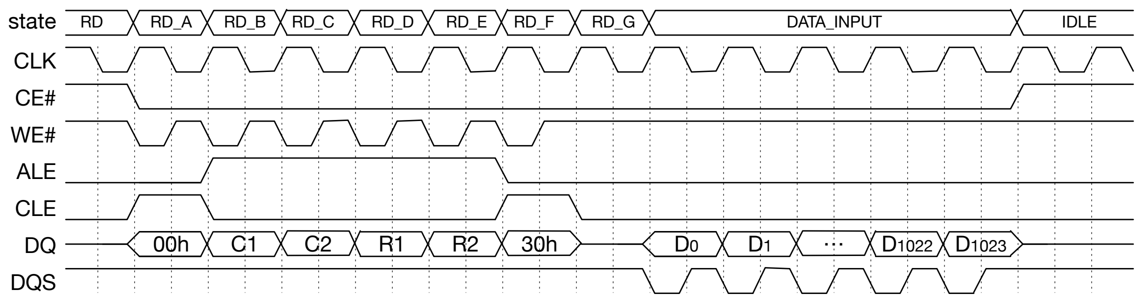


Note that *Page Program* has a lengthy command sequence, as you must send not only the command (80h) but also four address bursts on the bus. For this, simply send the 8-bit value on the switch bus four times. There will be 1024 data transfers from the controller to the memory device: 1K pulses (D0 .. D1023), so a total of 8Kbits, or 1KByte. It is recommended that you have a separate 10-bit counter that gets initialized when you enter the DATA_OUTPUT state, and you simply count down to zero; when done, stop transmitting data, and move into the PRG_G state.

Page Read Command

The timing for the *Page Read* command is given below, in two variations, depending on whether you want to use the rising or falling edge of the clock to drive your state machine.





The only thing different between this lab and the last is the command codes. As with the *Page Program* command, just use the 8-bit value on the switches multiple times.

8-bit Wide Block RAM

You will also need to modify your internal Block RAM to use an 8-bit interface. Instead of the RAMB16_S2 device, use the RAMB16_S9 device, which has the same amount of internal storage (16 Kbits), it just uses an 9-bit data interface (8 bits plus room for an optional parity bit, which you need not implement), and that is wide enough to accommodate your new 8-bit bus.

The Block RAM is instantiated as follows:

```
RAMB16_S9 #(

      .INIT(2’b00),     // Value of output RAM registers @ startup
      .SRVAL(2’b00),                // Output value upon SSR assertion
      .WRITE_MODE(“WRITE_FIRST”),   // WRITE_FIRST|READ_FIRST|NO_CHANGE

) instance_name (

      .DO(DO),          // 9-bit Data Output
      .ADDR(ADDR),      // 11-bit Address Input
      .CLK(CLK),        // Clock
      .DI(DI),          // 9-bit Data Input
      .EN(EN),          // RAM Enable Input
      .SSR(SSR),        // Synchronous Set/Reset Input
      .WE(WE)           // Write Enable Input

);
```

The address bus is now 11 bits instead of 13, so you only need to hardwire one bit to a “0” value.

LCD Driver

On the course website we have posted an LCD Driver with instructions on use. Download it, use it, and install it into your code. You should use it to print out the following information on each operation:

- Operation: *Reset*, *Read*, or *Program*
- Command and address bytes on the bus: hexadecimal values
- Data on the bus: first four bytes as hexadecimal values

Note on addresses: the bus will now be 8 bits wide, which means that you don’t have enough switches to give 4 cycles’ worth of address information. Therefore, use the same 8-bit value seen on the switches for all four address bytes.

Put all of the values up on the LCD, arranged so that all can be seen (do not put them up one by one in the same location, otherwise you will never be able to see them because they will change too rapidly).

In-Lab Procedure

Bring flash drives to store your data.

Ask the TA questions regarding any procedures about which you are uncertain.

Complete the following tasks:

- **Program your system onto the FPGA board.**
- Use the LCD screen to verify your design’s correctness; As the bus is now a bit wider, you do not need to use the DLA, as your timing should have been verified in the previous lab.

Post-Lab Report

Write up your code, schematics, and lab procedures.