Lab 9: Character LCD Control

National Chiao Tung University

Chun-Jen Tsai

11/27/2015

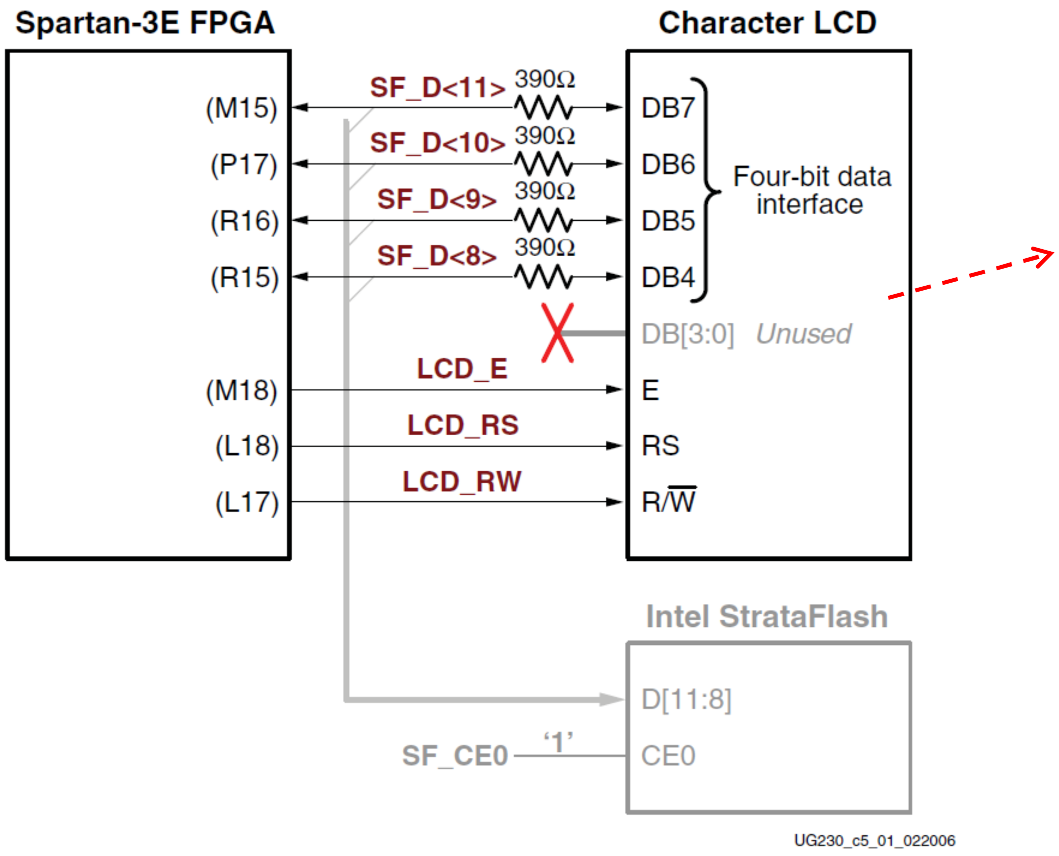
Lab 9: Character LCD Control

- In this lab, you will use the sieve algorithm to find all the primes from 2 to 1021, and use the character LCD to display the prime numbers

- You will demo the design to your TA during the lab hours on 12/7

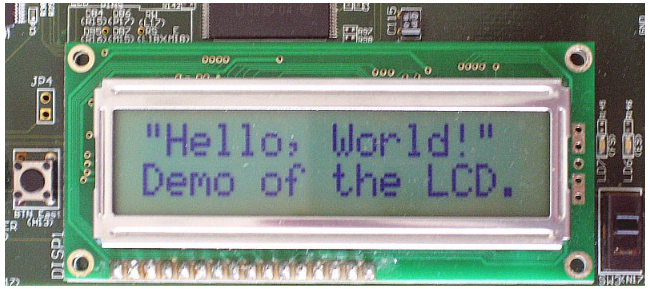
Character LCD Display

- ❑ There is a 2×16 character LCD display on the board:



Memory Map of the LCD

- The LCD device can be treated as a 32-byte memory
  - Each 8-bit cell corresponds to a character on the display
  - Writing an ASCII code to a cell will display the character on the corresponding location on the LCD:



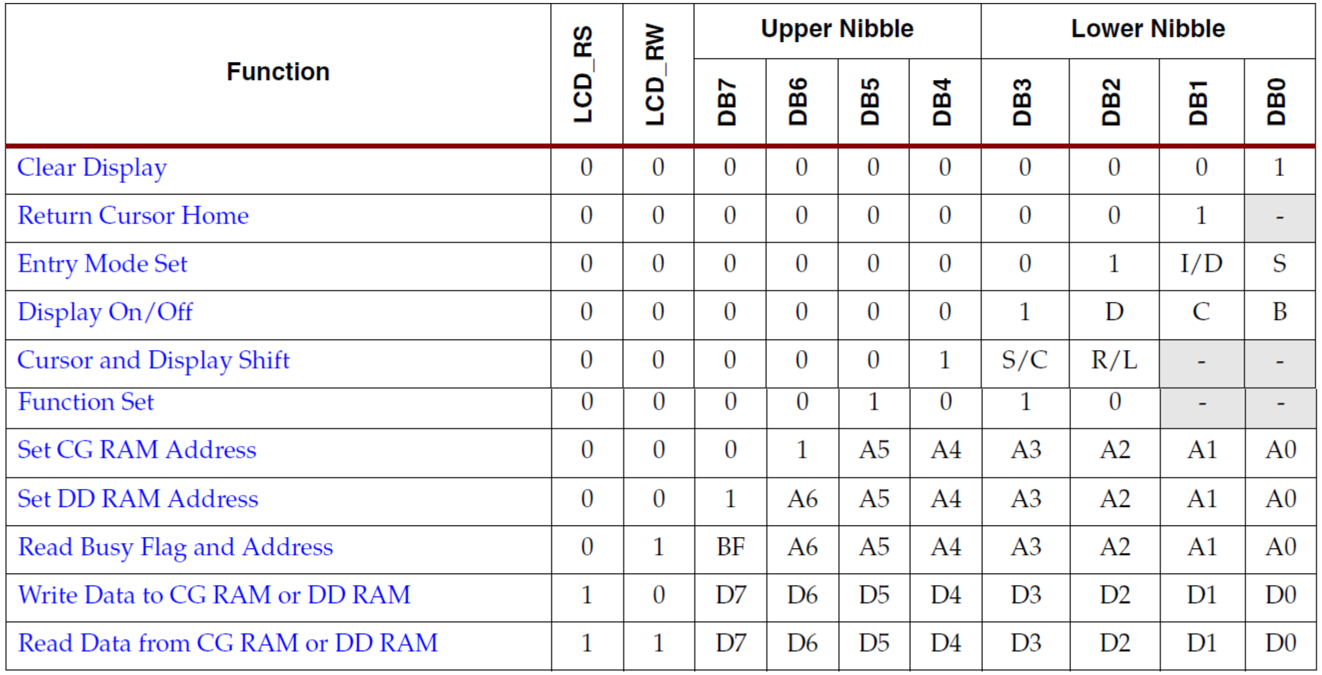
Note: the LCD is slow, you should not update the screen faster than every 0.5 seconds!

- Character display ram memory addresses:

1	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
2	40	41	42	43	44	45	46	47	48	49	4A	4B	4C	4D	4E	4F
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16

Character LCD Interface (1/2)

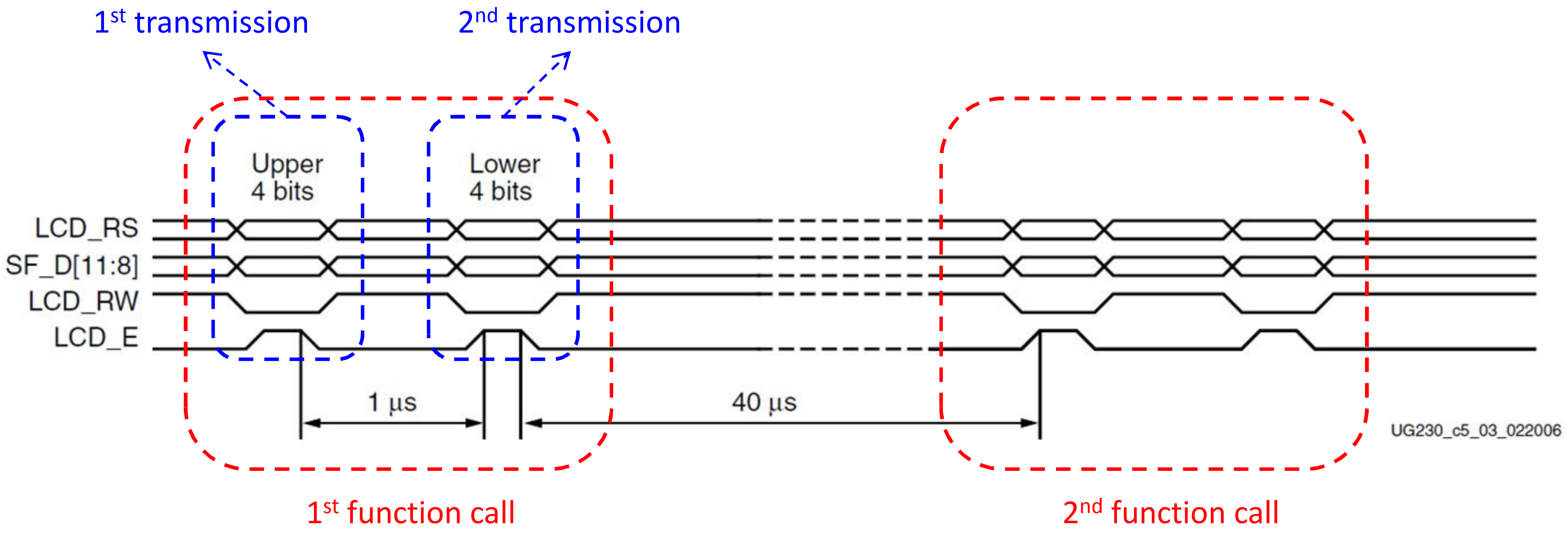
- The LCD interface has 8 data wires (DB0 ~ DB7) and 3 control wires (LCD_E, LCD_RS, LCD_RW):
  - LCD_E enable/disable the LCD module
  - The rest of the wires are defined depending on the functions:



Function	LCD_RS	LCD_RW	Upper Nibble				Lower Nibble			
			DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
Clear Display	0	0	0	0	0	0	0	0	0	1
Return Cursor Home	0	0	0	0	0	0	0	0	1	-
Entry Mode Set	0	0	0	0	0	0	0	1	I/D	S
Display On/Off	0	0	0	0	0	0	1	D	C	B
Cursor and Display Shift	0	0	0	0	0	1	S/C	R/L	-	-
Function Set	0	0	0	0	1	0	1	0	-	-
Set CG RAM Address	0	0	0	1	A5	A4	A3	A2	A1	A0
Set DD RAM Address	0	0	1	A6	A5	A4	A3	A2	A1	A0
Read Busy Flag and Address	0	1	BF	A6	A5	A4	A3	A2	A1	A0
Write Data to CG RAM or DD RAM	1	0	D7	D6	D5	D4	D3	D2	D1	D0
Read Data from CG RAM or DD RAM	1	1	D7	D6	D5	D4	D3	D2	D1	D0

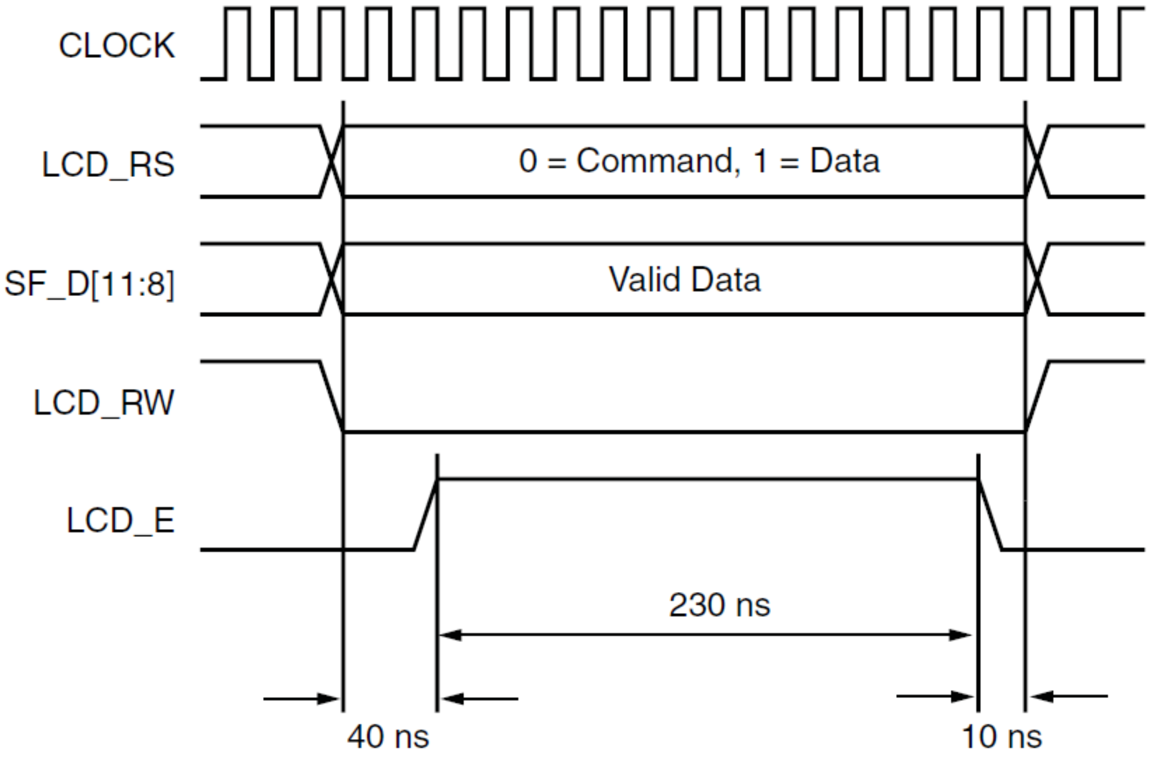
Character LCD Interface (2/2)

- However, the Spartan-3E Starter Board uses the 4-bit operating mode of the LCD device, that is, only DB4~DB7 are connected to the FPGA
  - Execution of a function will need two transmissions, using only LCD_E, LCD_RS, LCD_RW, and DB4~DB7:



Timing Diagrams for Transmission

- The timing diagram for one transmission in four-bit mode is as follows:
  - Note that execution of a function requires two transmissions



The Sample LCD Module

- A ISE workspace that contains an LCD module that handle all the timing issues will be provided for you to use; the workspace also has a “Hello, World!” circuit that shows you how to use the LCD module

```
module LCD_module(
    input clk,
    input reset,
    input [127:0] row_A,
    input [127:0] row_B,
    output reg LCD_E,
    output reg LCD_RS,     //register select
    output reg LCD_RW,     //read / write
    output reg [3:0]LCD_D //data
    );
```

top-row of text

bottom-row of text

The Sieve Algorithm (1/2)

- ❑ The sieve algorithm is an interesting algorithm to find prime numbers using only addition operations:

```
#define LIMIT 1024

unsigned char primes[LIMIT];

memset((void *) primes, 1, sizeof(primes));

/* sieve algorithm */
for (idx = 2; idx < LIMIT; idx++)
{
    if (primes[idx])
    {
        for (jdx = idx+idx; jdx < LIMIT; jdx += idx)
        {
            primes[jdx] = 0;
        }
    }
}
```

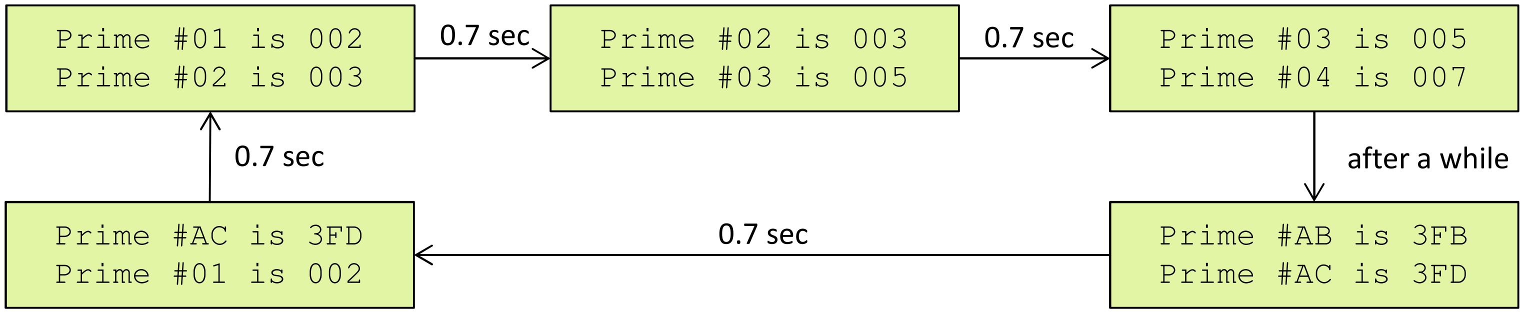
Turn these nested loops into an FSM!

The Sieve Algorithm (2/2)

- After running the Sieve algorithm, the array `primes[]` contains flags of whether a number is a prime or not
  - `primes[n] == 1` means `n` is a prime number,
`primes[n] == 0` means `n` is not a prime number
- In your circuit, you must declare the `primes[1024]` array as a 1K-byte SRAM (see the Lab 7 sample code)
  - If your circuit does not synthesize any BRAM in the synthesis report, you do not get full credit for this lab

What to Do in Lab9

- ❑ In Lab 9, it is mandatory for you to do following things:
  - Design an FSM to implement the sieve algorithm
  - Use a 1K-byte SRAM to store the array `primes[1024]`
  - Once all primes between 0 and 1024 are found, the LCD will start to display prime numbers in the following format
    - Roughly every 0.7 sec, the LCD scrolls up one number cyclically
    - If the west button is pressed, the scrolling direction will be reversed (scroll-up becomes scroll-down, and vice versa)
  - Example display: cyclic scroll-up (numbers are hexadecimal)



```
Prime #01 is 002
Prime #02 is 003
```

0.7 sec

```
Prime #02 is 003
Prime #03 is 005
```

0.7 sec

```
Prime #03 is 005
Prime #04 is 007
```

after a while

```
Prime #AB is 3FB
Prime #AC is 3FD
```

0.7 sec

```
Prime #AC is 3FD
Prime #01 is 002
```

0.7 sec

References

- ❑ Chapter 5 of *Spartan-3E Starter Kit Board User Guide, UG230 (v1.0)*.

- ❑ Xilinx, *Using Block RAM in Spartan-3 FPGAs, Xilinx Application Note, XAPP 463 v1.1.2*, July 23, 2003.