BEE 271, Digital circuits and systems
Autumn 2018
Lab 4: keyboard multiplier*

1 Objectives - revised

In this Lab 4, you will extend the Lab 3 design in one new way: add more arithmetic operations between the two operands. For this lab, redefine the UI as described below.

In Lab 3, you captured a 24-digit binary number (6 digit in hex display). In this lab, you will perform a 24-digit x 24-digit add, subtract, multiply or divide. In the later cases, the results may be much larger: 48 bits. Open question: how to display the results?

Redefine Key[1] as ENT (ENTer) key. Simple operation: ACC <= ENTRY

Redefine Key[0] as OPR button. Read two switches to select the operation:

- For ADD, SW[9:8] = 00; // compute 12 digit sum
- For SUB, SW[9:8] = 01; // compute 12 digit difference
- For MUL, SW[9:8] = 10; // compute 48 bit/12 digit product
- For DIV, SW[9:8] = 11; // compute quotient and remainder

Use a third switch (e.g. SW[7]) to choose what to display

- If ADD, no display change
- If SUB, no display change
- If MUL, switch between 6 MSB digits and 6 LSB digits
- If DIV, switch between 6 Quotient digits (and 6 Remainder digits)

Verilog syntax: (see textbook page 690)

- ADD A + B should be part of Lab 3
- SUB A – B should be part of Lab 3
- MUL A * B
- DIV A / B
- REM A % B

2 Work product

This lab will be an extension of Lab 3, with a new mathematic function.

Your lab report: you must demo your design and submit your code as a .v file or .vs file.

* This lab was written and revised by Joe Decuir

3 Procedure

In this lab, you'll scan the keypad, identify when a key is pressed and display it, same as Lag 3. Also, like Lab 3, a crucial component is to know when to detect that the key has been released.

The state machine then needs to scan the buttons (KEYs) to decide what to do.

1. Use the System Builder tool and Quartus Prime to create an empty Verilog project with the necessary inputs and outputs. Suggestion: use the Lab3 project, rename it and extend it.
2. Keep the keyboard scanning logic from Lab 3.
3. Keep the button (KEY) scanning logic, including the ability to wait for that key to be released.
4. Keep existing CLR (Key[3]) and CLE (Key[2]) button logic
5. Reassign the ADD button to ENT: ACC <= ENTRY
6. Reassign the SUB button to OPR: do the specified operations chosen by SW[9:8]
7. Suggestion: use SW[7] as a display toggle.
8. For Addition, SW[9:8] = 00:
   8.1. enter number, push ENT button (ACC <= ENTRY)
   8.2. enter second number, push OPR with SW[9:8] = 00.
   8.3. Display 6-digit sum: ACC = ACC + ENTRY
9. For Subtraction, SW[9:8] = 01:
   9.1. enter number, push ENT button (ACC <= ENTRY)
   9.2. enter second number, push OPR with SW[9:8] = 01.
   9.3. Display 6-digit difference: ACC = ACC - ENTRY
10. For Multiplication, SW[9:8] = 10:
    10.1. Enter number, push ENT button (ACC <= ENTRY)
    10.2. Enter second number, push OPR with SW[9:8] = 10.
    10.3. 48-bit product = ACC * ENTRY
    10.4. ACC <= product[23:0]
    10.5. If SW[7] = 0, display 6-digit product LSB (bits 23:0)
    10.6. If SW[7] = 1, display 6-digit product MSB (bits 47:24)
11. For Division, SW[9:8] = 11:
    11.1. Enter number, push ENT button (ACC <= ENTRY)
    11.2. Enter second number, push OPR with SW[9:8] = 11.

11.3. 24-bit quotient = ACC / ENTRY

11.4. 24-bit remainder = ACC % ENTRY

11.5. If SW[7] = 0, display 6-digit quotient

11.6. If SW[7] = 1, display 6-digit remainder

11.7. Optional: display "Error" for divide by zero. [extra credit]

12. Debug your design, demo it and submit your .v file or .sv file.

4 Example test vectors

A question has arisen for Lab 3: what to use for test vectors?

The following are some suggested inputs and expected results.

4.1 Addition SW[9:8] set to 00

4.1.1 Plain addition:

- Enter any 6-digit number, Press ENT (Key[1]), see it displayed.
- Enter another 6-digit number, Press OPR
- Expect to see the sum displayed.

4.1.2 Addition with overflow

- Enter two 6-digit numbers, chosen so that the result will be larger than 0xFFFFFF
- Observe that the result is smaller than 0xFFFFFF

4.2 Subtraction SW[9:8] set to 01

4.2.1 Plain subtraction:

- Enter a 6-digit number, Press ENT to see it displayed
- Enter a second smaller 6-digit number, Press OPR
- Expect to see the positive difference displayed

4.2.2 Subtract with negative result

- Enter a 6-digit number, Press ENT to see it displayed
- Enter a second larger 6-digit number, Press OPR
- Expect to see the negative difference displayed

4.3 Multiplication SW[9:8] set to 10

4.3.1 Simple multiplication

- Enter a 3-digit number, Press ENT
- Enter a second 3-digit number, Press OPR
- See a product of 6 digits or less

4.3.2 Long multiplication

- Enter a 6-digit number, Press ENT
- Enter a second 6-digit number, Press OPR
- See LSB 6-digits OR MSB 6-digits, chosen by SW[7]

4.4 Division with Remainder SW[9:8] set to 11

- Enter a 6-digit number, Press ENT
- Enter a second number, Press OPR
- See 6-digit quotient or 6-digit remainder, chosen by SW[7]

4.4.1 Denominator is even multiple of numerator

- Look for correct integer result, with remainder = 0

4.4.2 Denominator is not even multiple of numerator

- Look for correct integer result, with non-zero remainder.

4.4.3 Denominator is larger than the numerator

- Look for a quotient of 0 and remainder = first entry.

4.4.4 Denominator of zero (0)

- The Verilog logic will likely generate 0xFFFFFF as the quotient
- The Verilog logic remainder result is unknown.
- Extra credit: detect divide-by-0 and replace it with an “Error” message