Course: ENCM 369
Lab Section: B03
Lab 8
Student Name: Mitchell Sawatzky
Date Submitted: Mar 18, 2016

Exercise A

CLK runs at 2.00 GHz, $\frac{1}{2.00*10^9 Hz} = 5.00 * 10^{-10}s = 0.5ns = 500ps$

For the circuit to be safe, D2 and D7 must both be stable by 42ps before the next clock cycle hits. The sign extender is way faster than the multiplier, so only the multiplier's $t_{pd}$is relevant here.

$$500ps \geq 42ps + 35ps + 246ps + t_{pd_{adder}}$$
$$500ps - 42ps - 35ps - 246ps \geq t_{pd_{adder}}$$
$$177ps \geq t_{pd_{adder}}$$

Since Q6 is directly connected to D2, the t_{ccq}if the registers but be at least 10ps

Exercise B

2.)

```
    That is the end of the Fetch stage of the sub instruction, and
    the beginning of the Decode stage. The PC gets the output of the
    adder in the Fetch stage, which will be 0x0040_0190. InstrD gets
    the instruction, which will be 0x0222_9822.
    The instructions about to be written into the D/E pipeline
    register are from the sw instruction. RD1 holds the value inside
    the A1 register, which is 0b11101 (Register 29), which is
    0x7fff_ec80. RD2 holds the value inside the A2 register, which is
    0b10010 (Register 18), which is whatever value is in register 18.
    SignExtend receives 0b0000000000101000, and extends it with
    zeros.

    PC: 0x0040_0190
    InstrD: 0x0222_9822
    RD1: 0x7fff_ec80
    RD2: whatever was in register 18
    SignExtend: 0x0000_0028
```

3.)

```
    Shortly before 51.5ns is when the sub instruction is just done
    its Decode stage. RD1 holds the A1 register value (Register
    0b10001 (17)), which is 0x1001_0050. RD2 holds the A2 register
    value (Register 0b00010 (2)), which is 0x0000_0008. InstrD values
    are the subsets of the sub instruction.
    The output of the ALU is dependant on the sw instruction. The ALU
    will be in ADD mode to add the sign-extended offset with the A1
    register value. This is 0x28 + 0x7fff_ec80, which is 0x7fff_eca8.
    The sw instruction does not write a value back into a register
    when it is done, so it is undefined whether RegDstE is 0 or 1.
    That being said, the section of the instruction holding a
    register address is InstrD20:16, so WriteRegE will most likely be
    register 18, or 0b10010
```

RD1: 0x1001_0050
RD2: 0x0000_0008
$\text{InstrD}_{20:16}$: 0b00010
$\text{InstrD}_{15:11}$: 0b10011
ALU: 0x7fff_eca8
$\text{WriteRegE}_{4:0}$: 0b10010

4.)

Shortly before 52.0ns is when the sub instruction is just done its Execute stage. The ALU will be in SUB mode, and will subtract the RD2 value from the RD1 value. This is 0x1001_0050 – 0x0000_0008, which is 0x1001_0048. WriteRegE will be register 19, or 0b10011
ALU: 0x1001_0048
$\text{WriteRegE}_{4:0}$: 0b10011

5.)

Shortly before 52.5ns is when the sub instruction is just done its Memory stage. ALUOutM is the same as ALU in the previous stage. WriteRegM is the same as WriteRegE in the previous stage.
ALUOutM:0x1001_0048
$\text{WriteRegM}_{4:0}$: 0b10011