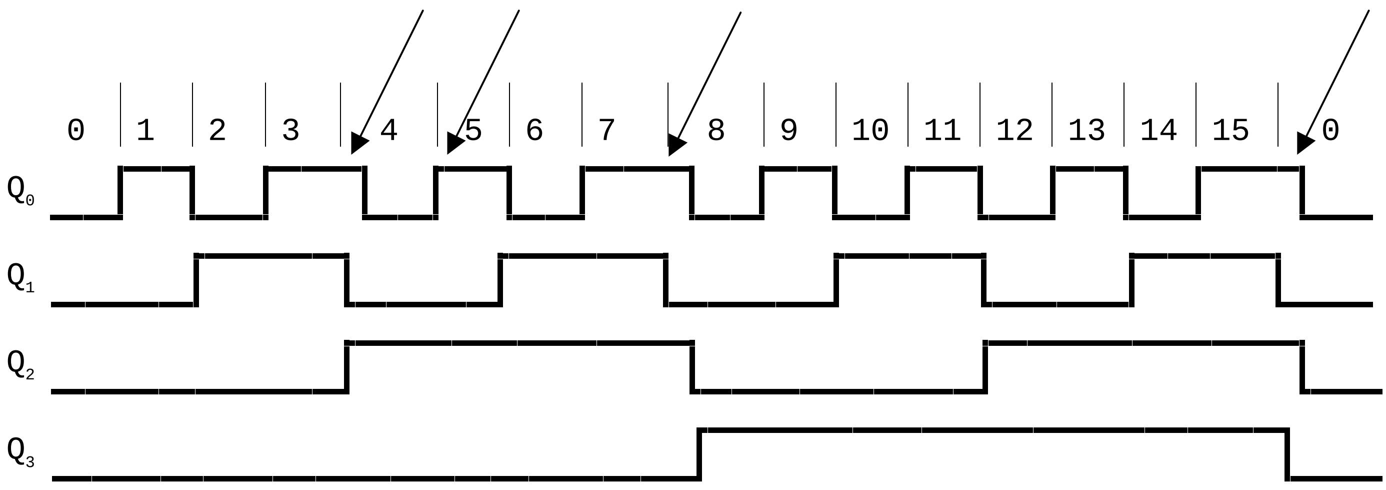
CS150 PS 5 Solutions M. Brudno

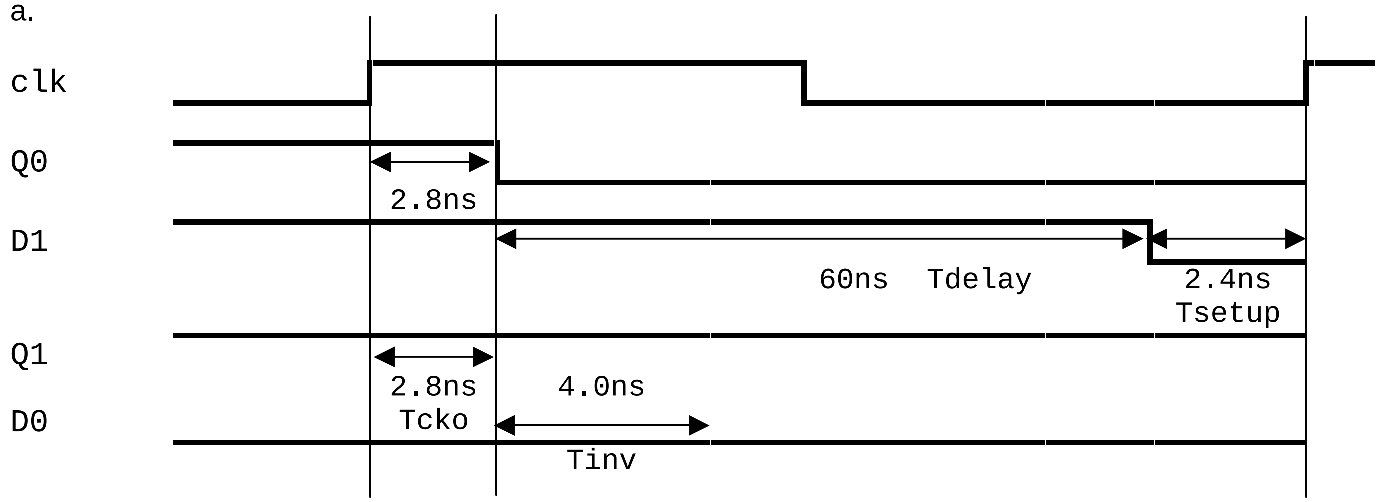
1.



The FIVE signal could potentially go high during the 3→4, 4→5 , 7→8 and 15→0 transitions. A counter with an asynchronous clear driven by FIVE could reset when a number other than 5 is reached, since it does not require that FIVE be asserted at the rising edge of a clock.

2.

a.

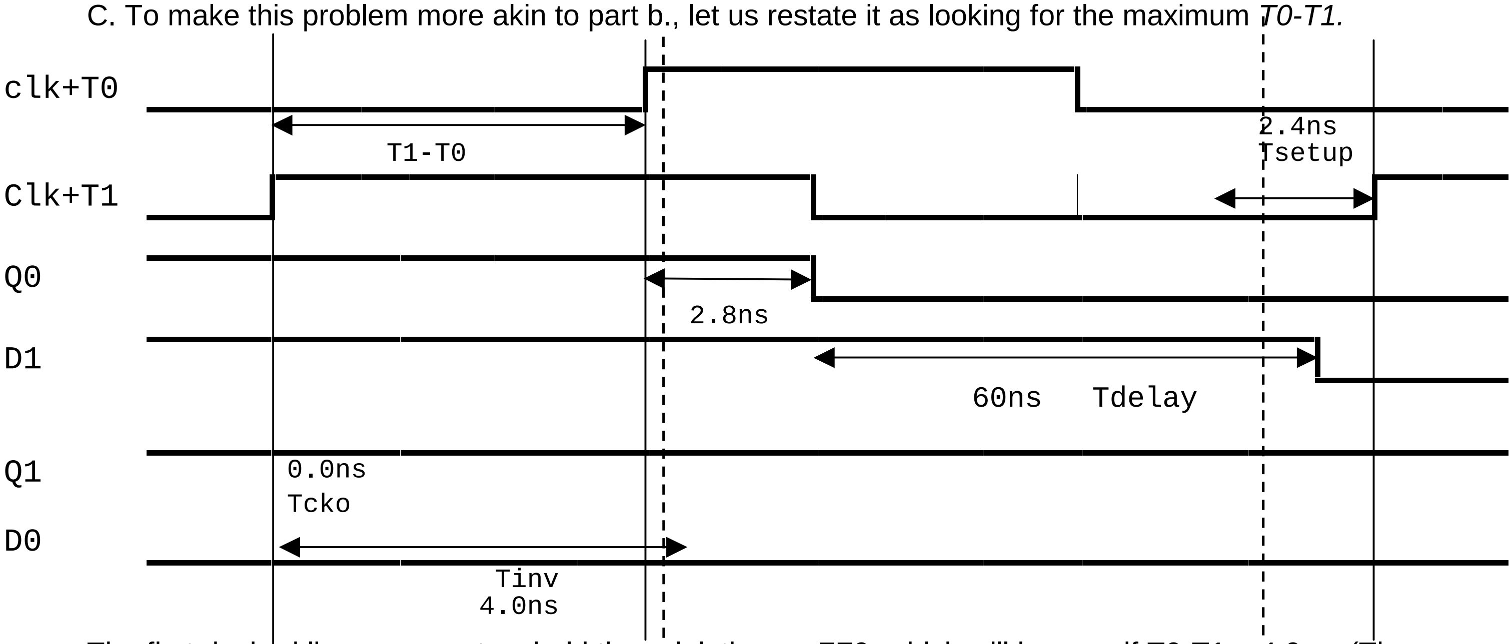


So the clock period must be at least 65.2ns, and the frequency cannot be greater than 1/65.2ns=15.3MHz.

B. There are two potential problems: if T1-T0 is too large we could either cause a setup time violation on Q0 or the delay can be so large that we will clock in the wrong (future) value in FF1. The timing diagram demonstrates both possible problems.

The first dashed line represents a hold time violation on FF1, which will happen if T1-T0 > 60.0ns. (The T1 clock must rise before D1 goes low, otherwise you are loading the next-next state into the FF#1.) The second one is a setup time violation on FF0 during the next clock, which will happen if T1-T0 > (ω - 9.2ns), where ω is the period of the clock.

C. To make this problem more akin to part b., let us restate it as looking for the maximum *T0-T1.*



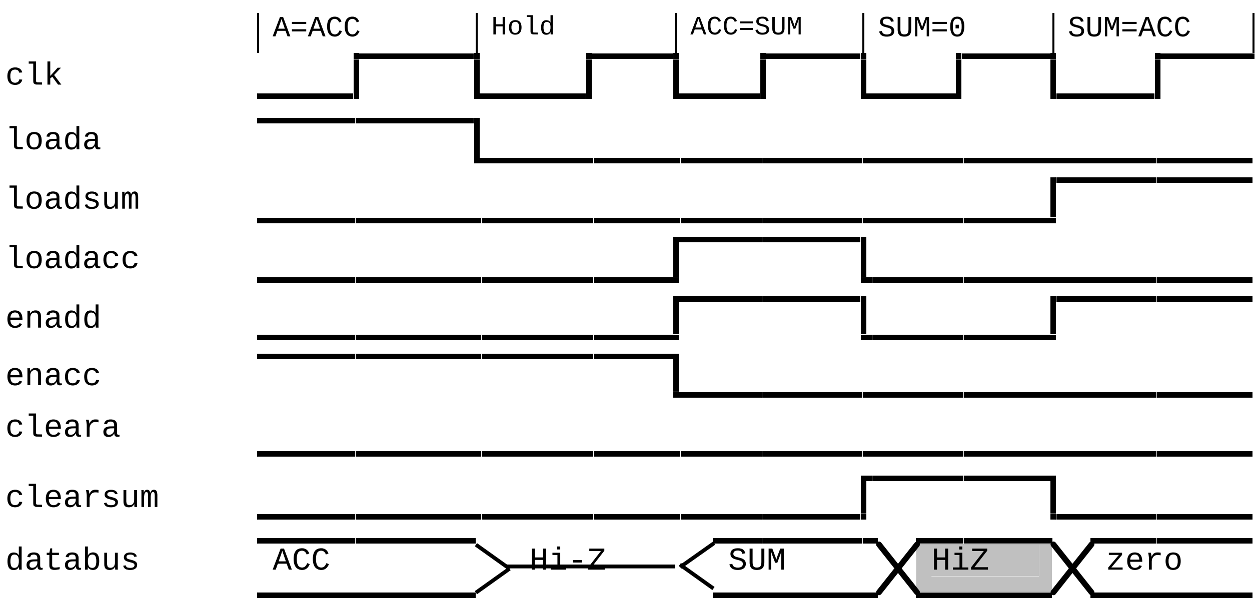
The first dashed line represents a hold time violation on FF0, which will happen if T0-T1 > 4.0ns. (The T0 clock must rise before D0 goes low, otherwise you are loading the next-next state into the FF0.) The second one is a setup time violation on FF1 during the next clock, which will happen if T0-T1 > (ω - 65.2ns).

D. You should always run your clock through a BUFGS, which tells the Xilinx compiler to put the signal on a special wire in the board. This will considerably reduce the clock delay. However you can only use up to 6 BUFGSes in a schematic.

3. Signals in parenthesis are to be asserted during the same clock. After the colon is the explanation. All signals low unless otherwise asserted. SUM, ACC, A are the initial values of the registers. SUM’, ACC’, and A’ are their current values at the time of the operation. Because of the delay in going to tri-state, we need to include extra cycles to avoid bus conflicts. These cycles are designated as ‘()’. Note that we should not enable the other BUFEs during those cycles.
The values are shown to change at the falling edge of the clock to indicate the delay which occurs between the rising edge and when all the outputs have stabilized.

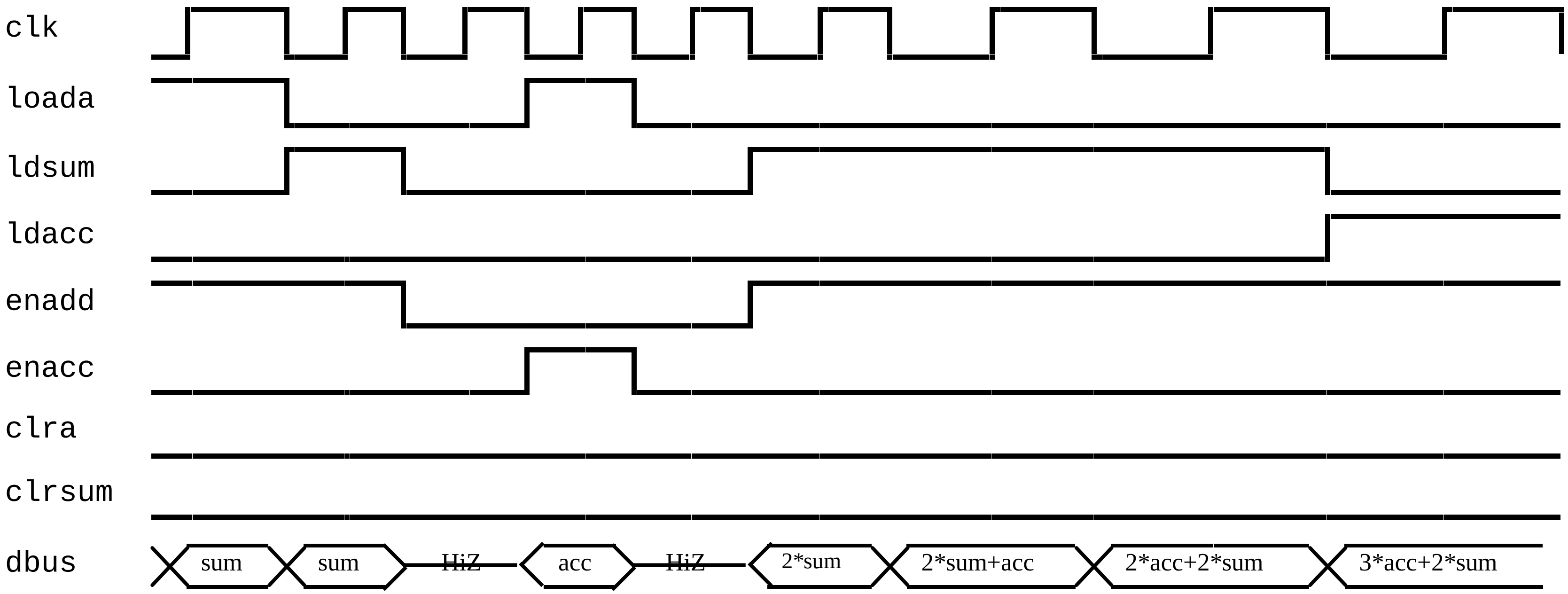
a.

Operation	Signals
ACC→A	LOADA, ENACC
NoOp	
SUM→ACC	ENADD,LOADACC
0→SUM	CLEARSUM
SUM+A→SUM	ENADD,LOADSUM

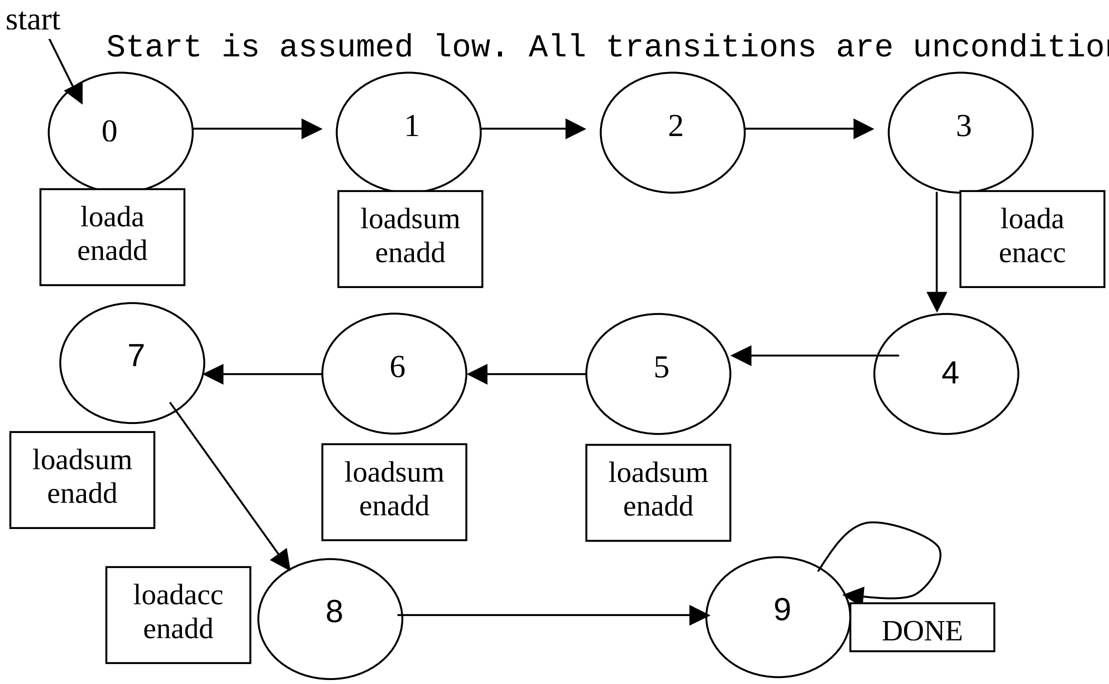


b.

Operation	Signals
SUM→A	LOADA,ENADD
SUM+A→SUM	ENADD,LOADSUM
NoOp	
ACC→A	ENACC,LOADA
NoOp	
SUM+A→SUM	ENADD,LOADSUM
SUM+A→SUM	ENADD,LOADSUM
SUM+A→SUM	ENADD,LOADSUM
SUM→ACC	ENADD,LOADACC)


clk
loada
ldsum
ldacc
enadd
enacc
clra
clrsum
dbus
sum
sum
HiZ
acc
HiZ
2*sum
2*sum+acc
2*acc+2*sum
3*acc+2*sum

C. Start is assumed low. All transitions are unconditional.



Schematic: