ECE 327 Solution to Final

2013t2 (Spring)

Q1 (15 Marks) Dataflow Diagram Design

(*estimated time: 20 minutes*)

Draw a dataflow diagram for the system described below:

```
z = max( 3a - (b+c+d),  3b - (a+c+d),  3c - (a+b+d),  3d - (a+b+c) )
```

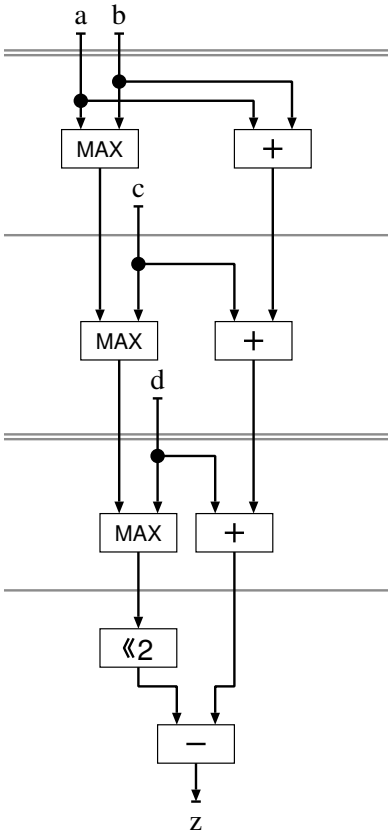
NOTES:

1. The inputs are `a`, `b`, `c`, and `d`.
2. The output is `z`.
3. Hint: $3i - j = 4i - (i + j)$.
4. Hint: $\texttt{max}(i-k,\ j-k) = \texttt{max}(i,j) - k$
5. The *inputs* shall be registered.
6. The *output* may be either registered or combinational.
7. The `max` operation shall be implemented using MAX components with two inputs (*e.g.*, m = MAX(a,b)).
8. The *throughput* shall be at least 1/2.
9. Clock period:
   - Each 2-input MAX component has a delay of t_m
   - Each 2-input MULT component has a delay of t_m (same as a MAX)
   - Each 2-input ADD component has a delay of t_n
   - Each 2-input SUB component has a delay of t_n (same as an ADD)
   - $t_n < t_m < 2t_n$
   - The clock period shall be at most FLOP + $2t_n$. Reducing the clock period below this limit is an optimization goal.
10. Optimization goals, in order of *decreasing* importance:
    (a) Minimal area, in order of *decreasing* importance:
        i. Inputs
        ii. Registers
        iii. Multipliers
        iv. MAX components
        v. Total number of adders and subtracters
    (b) Minimal clock period
    (c) Minimal latency
11. You may use *algebraic optimizations*, as long as the value of `z` is correct.
12. You may schedule the input values to arrive in any order, but you may read each input value only once.
13. You do *not* need to do any allocation.

Answer:

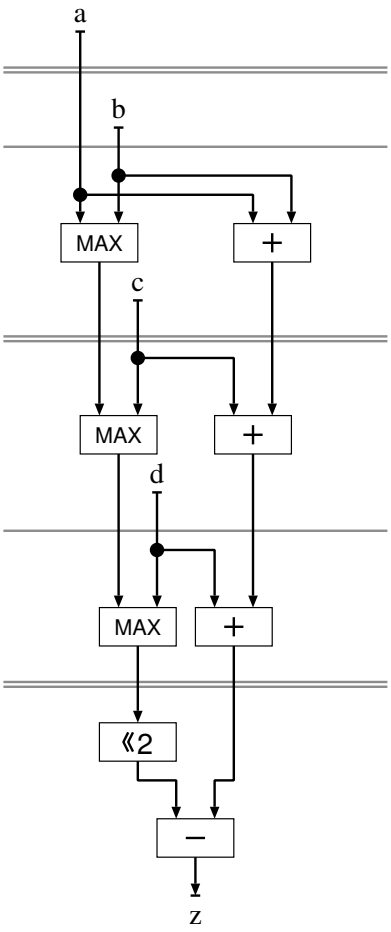
1. Algebraic simplificiations

```
z  =  MAX( 4a - a+b+c+d,  4b - a+b+c+d,  ...  )
   =  MAX( a, b, c, d)×4 - (a+b+c+d)
   =  MAX( a, MAX(b, MAX(c, d)))×4 - (a+b+c+d)
```



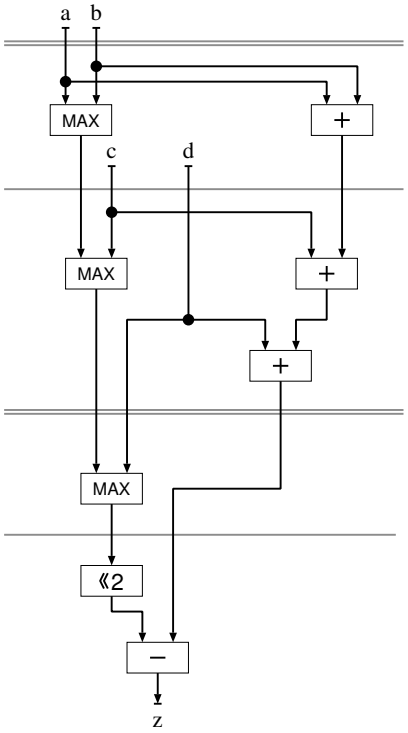
Latency	*4*
Throughput	*1/2*
Clock period	*Flop+t_m*
Number of inputs	*2*
Number of registers	*6*
Number of multipliers	*0*
Number of MAX*s*	*2*
Number of adders	*2*
Number of subtracters	*1*

15 marks



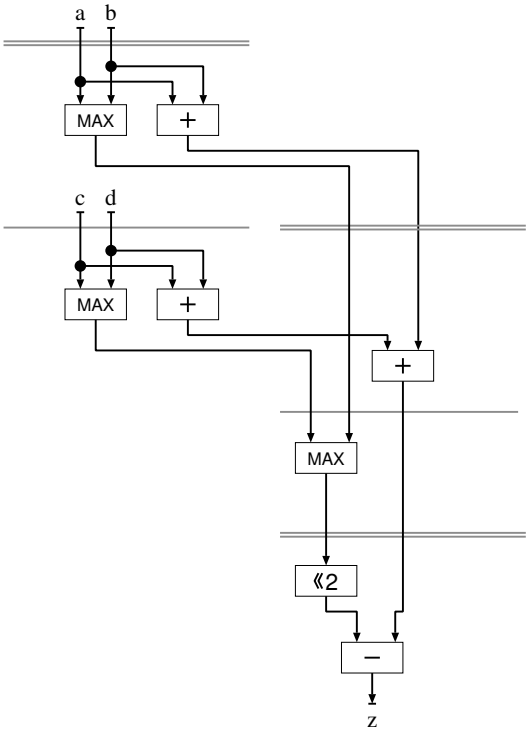
Latency	*5*
Throughput	*1/2*
Clock period	*Flop+t_m*
Number of inputs	*2*
Number of registers	*7*
Number of multipliers	*0*
Number of MAX*s*	*2*
Number of adders	*2*
Number of subtracters	*1*

14 marks



Latency	*4*
Throughput	*1/2*
Clock period	*Flop+$2t_n$*
Number of inputs	*2*
Number of registers	*7*
Number of multipliers	*0*
Number of MAX*s*	*2*
Number of adders	*2*
Number of subtracters	*1*

13 marks



Latency	*4*
Throughput	*1/2*
Clock period	*Flop+$2t_n$*
Number of inputs	*2*
Number of registers	*7*
Number of multipliers	*0*
Number of MAX*s*	*2*
Number of adders	*2*
Number of subtracters	*1*

13 marks

Other answers:

2 inputs, 8 registers, Flop+t_n: 13 marks

3 inputs, 6 registers: 11 marks

Marking:

-1 mark *missing stage boundary at top of DFD*

-2 marks *combinational inputs*

-1 mark *registered outputs*

-2 marks *used a multiplier*

-5 marks *throughput = 1/3*

-8 marks *throughput $\geq$ 1/4*

Q2 (15 Marks) Allocation and Control

(*estimated time: 25 minutes*)

For the dataflow diagram on the next page, perform: input/output allocation, datapath allocation, and register allocation; then draw the control table.

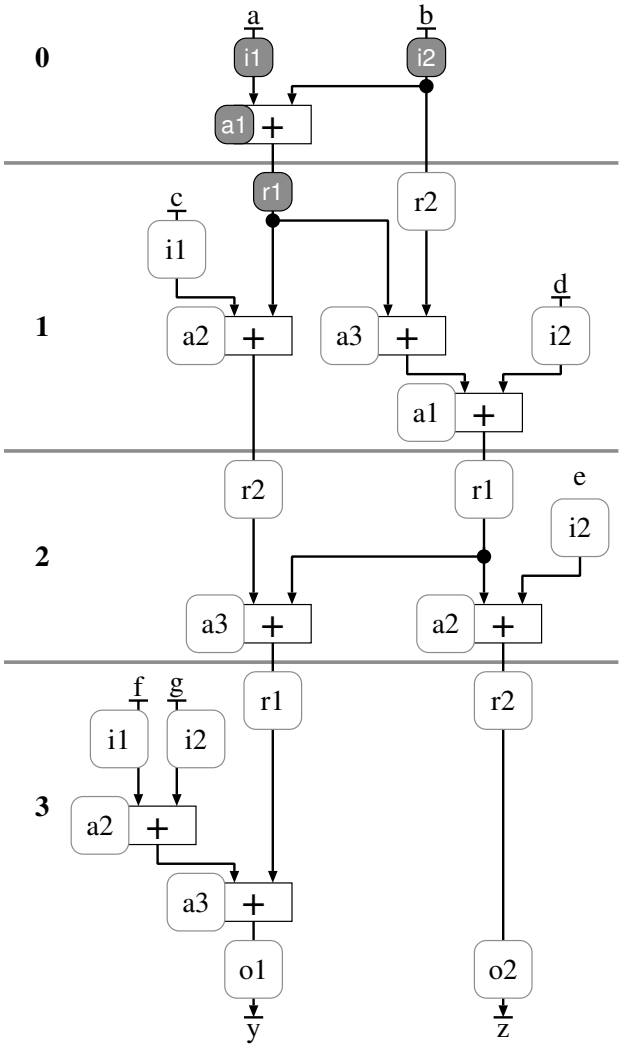
NOTES:

1. You may *not* change the allocations that have already been done.
2. You may use 2:1 multiplexers, and may combine 2:1 multiplexers to create larger multiplexers. (e.g. two 2:1 multiplexers may be combined to create a 3:1 multiplexer)
3. Optimization goals in order of highest priority to lowest:
   (a) Minimize the number of adders
   (b) Minimize the number of input and output ports
   (c) Minimize the number of registers
   (d) Minimize the number of 2:1 multiplexers (including the 2:1 multiplexes used to build larger multiplexers)
4. You may *not* perform scheduling optimizations on the dataflow diagram.
5. The *only* algebraic optimization that you may perform is *commutativity*.
6. The system has combinational inputs. If you do not know how to do allocation and write the control table for combinational inputs, you may earn *part marks* by writing a $\surd$ in the box below and using registered inputs.

   I am using registered inputs, not combinational inputs. ☐

Q2a (10 Marks) Allocation

Answer:



Marking:

10 marks *5 muxes*
8 marks *6 muxes*
7 marks *7 muxes*
6 marks *8 muxes*
5 marks *9 or more muxes*
-1 mark *1 missing or incorrect allocation*
-2 marks *2–3 missing or incorrect allocations*
-2 marks *used a1 in both cycle 0 and cycle 3*
-2 marks *more than 2 inputs, 2 registers, or 3 adders*
-2 marks *using same adder more than once in a single clock cycle*

Q2b (5 Marks) Control Table

Draw the control table for the *registers* and *adders* from the dataflow diagram.

NOTES:

1. Label each row and column of the table clearly.
2. The system shall support an *unpredictable number of bubbles* between valid parcels.
3. For *each* register and adder, write down the number of 2:1 multiplexers that it uses as an extra line at the bottom of the control table.
4. Mark cells as "don't-care" or "unused" using the symbols from class. If you do not know the symbols, you may earn *part marks* by writing a $\surd$ in the box at at the end of this question and using "DC" for "don't-care" and "UN" for unused.

   I am answering with *DC and UN*, not the symbols from class. ☐

Answer:

	r1		r2		a1		a2		a3		
	ce	**d**	**ce**	**d**	**src1**	**src2**	**src1**	**src2**	**src1**	**src2**	
0	1	a1	1	i2	i1	i2					
1	1	a1	1	a2	a3	i2	i1	r1	r1	r2	
2	1	a3	1	a2	—	—	i1	r1	r1	r2	
3							i1	i2	r1	a2	**Total**
muxes	1		1		1		1		1		5

Marking:

- **+1 mark** *unused is correct*
- **+1 mark** *don't care is correct*
- **+1 mark** *chip enable is correct*
- **+1 mark** *sources are correct*
- **+1 mark** *mux count is correct*
- **-2 marks** *mux count is missing*
- **-2 marks** *used DC and UN*

Q3 (10 Marks) Optimality Metrics

(*estimated time: 10 minutes*)

You have just been promoted to your dream job: drawing dataflow diagrams for the next-generation Waterluvian filter. The details of Waterluvian filters are not relevant to this question. At your first design review, you present several different dataflow diagrams.

Your manager asks you if you can define a formula to *estimate the optimality of a dataflow diagram*. The optimality of Waterluvian filters is measured using Megapixels-per-second per Watt (MPPS/Watt).

The estimate of optimality will be used to compare the *relative* optimality of dataflow diagrams. The estimate is *not* intended to give the actual optimality of a system that implements the dataflow diagram. That is, if you *estimate* the optimality of DFD1 to be 500 and DFD2 to be 400, then an implementation of DFD1 will have a higher actual optimality than an implementation of DFD2, but the actual optimality of DFD1 is not necessarily 500 MPPS/Watt.

NOTES:

1. The inputs to the formula shall be some or all of the following, as measured on a dataflow diagram:
   - **A** = area (FPGA cells)
   - **F** = clock speed (MHz)
   - **L** = latency (clock cycles)
   - **T** = throughput (pixels per clock cycle)
2. The output of the formula shall be an estimate of optimality.
3. The estimate of optimality will be used only to compare Waterluvian filters for the same type of FPGA.

Is it possible to estimate optimality? **For full marks, you must justify your answer.**

Answer:

MPPS = $T \times F$
Power:

- *Static power is proportional to capacitance (area)*
- *Dynamic power is proportional to capacitance (area) and clock speed.*
- *Cannot estimate activity factor from dataflow diagram, so assume that all DFDs have the same activity factor.*

From above, relative power can be approximated by $A \times F$.

$$\begin{aligned} MPPS/Watt &= TF/AF \\ &= T/A \end{aligned}$$

Q4 (15 Marks) Performance

(*estimated time: 15 minutes*)

The simple version of Amdahl's law is "make the common case fast". This question explores the issues of "how common" and "how fast".

You are the project leader for an edge-detection circuit. Two of your group members have proposed performance optimizations. You have time to implement only one of the optimizations.

NOTES:

1. Eric Edge has proposed an optimization that will give a 90% performance improvement for pixels that are on an edge.
2. Mark Dark has proposed an optimization that will give a 10% performance improvement for pixels that are *not* on an edge.
3. On average, 10% of the pixels are on an edge.
4. Currently, it takes n_e clock cycles to process a pixel that is on an edge and n_m clock cycles to process a pixel that is *not* on an edge
5. Your system is not pipelined. Neither Eric's nor Mark's optimization will introduce pipelining.

Your task is to determine under what condition will Eric Edge's optimization be better than Mark Dark's optimization. **For full marks, you must justify your answer.**

Answer:

$$
\begin{aligned}
n_e &= \textit{time to process one edge pixel now} \\
n_m &= \textit{time to process one non-edge pixel now} \\
T_0 &= \textit{time to process average image of 100 pixels now} \\
&= 10n_e + 90n_m \\
n'_e &= \textit{optimized time for one edge pixel} \\
&= (1/1.90)n_e \\
T_e &= \textit{time to process average image of 100 pixels with edge optimization} \\
&= 10n'_e + 90n_m \\
&= 10 \cdot (1/1.9)n_e + 90n_m \\
&= 5.26 \cdot n_e + 90n_m \\
n'_m &= \textit{optimized time for one non-edge pixel} \\
&= (1/1.10)n_m \\
T_m &= \textit{time to process average image of 100 pixels with non-edge optimization} \\
&= 10n_e + 90n'_m \\
&= 10n_e + 90(1/1.10)n_m \\
&= 10n_e + 82n_m
\end{aligned}
$$

Eric's option is better when: $T_e < T_m$

$$
\begin{aligned}
T_e &< T_m \\
5.26 \cdot n_e + 90n_m &< 10n_e + 82n_m \\
n_m &< 0.5925n_e \\
1.6877n_m &< n_e
\end{aligned}
$$

If the time to process an edge pixel is at least 69% more than the time to process a non-edge pixel, then Eric's optimization to increase the performance for edge pixels is better.

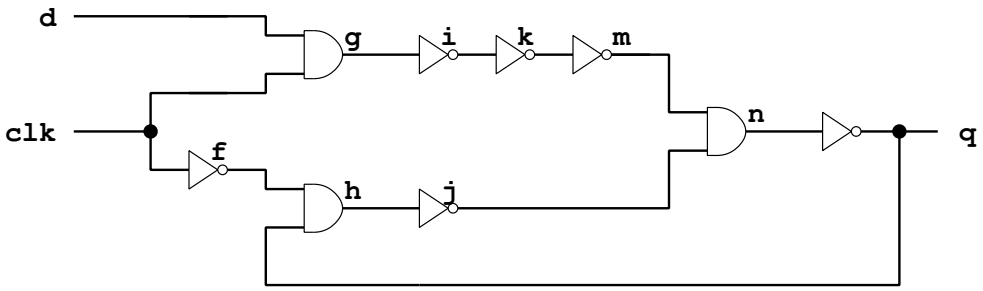
Marking:

+2 marks *time for one edge pixel with Eric's optimization*
+2 marks *time for one non-edge pixel with Mark's optimization*
+2 marks *concept of an average image and total execution time*
+2 marks *equation for total time with Eric's optimization*
+2 marks *equation for total time with Mark's optimization*
+2 marks *equation to compare total times*
+2 marks *algebra*
+1 mark *final answer*

Q5 (15 Marks) Latch Analysis

(*estimated time: 20 minutes*)

In this question, you will analyze the circuit below to determine if it is a correct implementation of a latch.



NOTES:

1. The delay through each gate is 1 ns.

Q5a (3 Marks) Good or Bad?

Answer:

Yes, the circuit is a correct implementation of a latch.

Marking:

3 marks *correct answer*

1 mark *incorrect answer (if rest of question is answered)*

Q5b (7 Marks) Analysis

Determine if the latch is active-high or active-low and calculate the timing parameters below.

Answer:

Active-high or active-low?	*Active high*	*1 mark*
Clock-to-Q	*6*	*2 marks*
Setup	*5*	*2 marks*
Hold	*0*	*2 marks*

Marking:

latch is correct

1 mark *answer is* ± 1 *the correct answer*

latch is incorrect

7 marks *glitch on transition from load to store caused by delay from clk to n on load path being longer than along storage loop*

5 marks *problem on transition from load to store caused by improper choice of gate delays*

4 marks *glitch on transition from load to store*

2 marks *glitch on transiton from store to load, or no mention of which mode transition has the glitch*

2 marks *odd number of inverters on storage loop*

Q5c (5 Marks) Modification

If you answered that the latch is correct, then modify the circuit to *reduce the clock period*; otherwise modify the circuit to make it a correct latch.

Answer:

We cannot modify the latch to reduce the clock period.

The answer below is longer than would be expected on an exam.

Clock period includes setup and clock-to-Q. For the latch to multiplex between the load path and storage loop, we need an odd number of inverters between g and n and between h and n. Also, we need the delay from clk to n along the load path to be longer than the delay from clk to along along the storage loop.

We need the inverter j, therefore we need the three inverters i, k, and m.

We could reduce clock-to-Q by moving the inverter for q to the feeback loop, but then to have an even number of inverters on the load path, we would need to add an inverter between d and g, which would increase the setup time, thus negating the reduction in clock-to-Q.

If we try to reduce the setup time by adding a pair of inverters between clk and h, we will cause the latch to be a bad latch, because the delay from clk to j will become greater than the delay from clk to m.

Marking:

if answered that cannot reduce clock period

- **+1 mark** *must reduce setup and/or clock-to-Q*
- **+1 mark** *must delete or move an inverter*
- **+1 mark** *must have odd number of inverters between g and n and between h and n*
- **+1 mark** *must have longer delay from g to n than from h to n*
- **+1 mark** *if decrease clock-to-Q then will increaset setup*

if modified latch (either for functionality or to reduce clock period)

- **+1 mark** *q is not inverted*
- **+1 mark** *even number of inverters on storage loop*
- **+1 mark** *mux between load-path and storage loop works*
- **+1 mark** *delay from clk to n along load path is is greater than or equal to delay from clk to n along storage loop*
- **+1 mark** *active high*

if tried to reduce clock period

- **-1 mark** *did not reduce either setup or clock-to-Q*
- **-1 mark** *reduced hold-time to reduce clock period*

Q6 (15 Marks) Timing Analysis

(*estimated time: 20 minutes*)

In the circuit below, the longest path, $\langle \mathtt{c},\mathtt{e},\mathtt{f},\mathtt{g},\mathtt{i},\mathtt{k},\mathtt{m},\mathtt{n}\rangle$, is a false path. The second longest path is $\langle \mathtt{b},\mathtt{h},\mathtt{k},\mathtt{m},\mathtt{n}\rangle$.

NOTES:

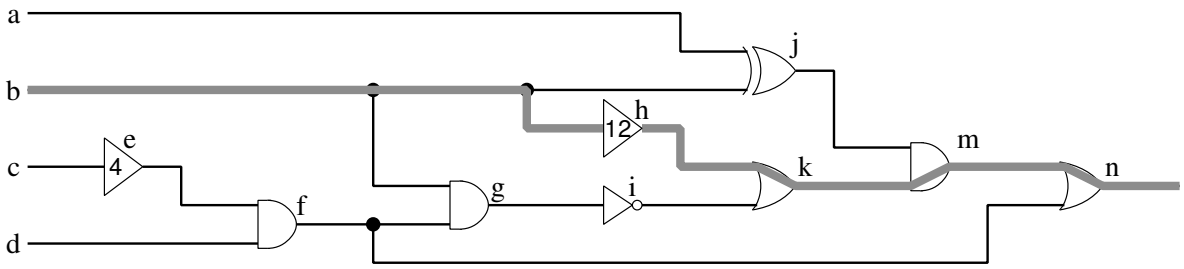
1. A *buffer* is drawn as a triangle with its delay written inside the triangle. The output of the buffer is simply a delayed version of the input.

Delay through gates

	2
	4
	6
	12

Q6a (12 Marks) Viability of Path

Answer whether the second longest path $\langle \mathtt{b},\mathtt{h},\mathtt{k},\mathtt{m},\mathtt{n}\rangle$ is viable. If it is viable, give an excitation, otherwise, explain why the path is *not* viable.



1. Prelude: check longest path (not needed in real solution, because we are told that this is a false path.)

   path #1 (longest) = $\langle \mathtt{c},\mathtt{e},\mathtt{f},\mathtt{g},\mathtt{i},\mathtt{k},\mathtt{m},\mathtt{n}\rangle$

f[d]	1	d	d
g[b]	1	b	b
k[h]	0	$\overline{h}$	$\overline{b}$
m[j]	1		
n[f]	0	$\overline{f}$	

   Contradiction on b between g[b] and k[h].

2. Begin actual solution: viability of path #2 = $\langle \mathtt{b},\mathtt{h},\mathtt{k},\mathtt{m},\mathtt{n}\rangle$.

k[i]	0	$\overline{i}$	=	g
			=	bf
m[j]	1			
n[f]	0	$\overline{f}$	$\overline{f}$	

   Contradiction on f. We may stop as soon as we find a contradiction. We do not have to push the constraints all the way to to the inputs. Also, we do not have to explore the constraint on m[j], because we already have a contradiction.

3. k[i] is on the prefix of a previously discovered false path. Try k[i] as a late-arriving side input.

   viable($\langle \mathtt{c},\mathtt{e},\mathtt{f},\mathtt{g},\mathtt{i}\rangle$) AND ctrl(k[h])

viable($\langle$ c, e, f, g, i $\rangle$)
= nonctrl(f[d]) AND nonctrl(g[b])
= d AND b
= bd

ctrl(k[h])
= b

viable($\langle$ c, e, f, g, i $\rangle$) AND ctrl(k[h])
= bd AND b
= bd

4. add condition for late side into constraint table

k[i]	$bf+bd$	
	= $bcd+bd$	
	= bd	
m[j]	1	j
n[f]	$\overline{f}$	
	= $\overline{c}+\overline{d}$	

combine constraints together

bd AND j AND $(\overline{c}+\overline{d})$
= $b\overline{c}dj$
to get $j=1$ with $b=1$, need $a=0$
= $\overline{a}b\overline{c}d$

5. excitation a='0', b=↑, c=*downarrow*, d=1

Marking:

+2 marks *viable="yes"*

If deriviation is clear and understandable, or said "viable"

+2 marks *non-controlling values on side inputs*
+2 marks *need late side input*
+1 marks *late side is viable*
+1 marks *path input is controlling*
+1 marks *propagate constraints to inputs*
+1 marks *conjunction of constraints*
+1 marks *translate constraint into excitation*
+1 marks *late side input has edge*

If derivation is unclear

+3 marks *a='0'*
+3 marks *b=rising edge*
+3 marks *c=falling edge*
+3 marks *d-1*

Q6b (3 Marks) Monotone Speedup

Does this circuit demonstrate the benefits of taking monotone speedup into account when calculating the critical path? **For full marks, you must justify your answer.**

Answer:

No, the circuit does not illustrate the benefits of taking into account monotone speedup, because an edge will propagate along the critical path with the given delays through the gates.

Marking:

+1 mark *does not illustrate monotone speedup*

+1 mark *need late side input*

+1 mark *excitation generates edge on output*

Q7 (15 Marks) Power

(*estimated time: 25 minutes*)

You are developing a circuit that has a strict power budget. The current plan for meeting the power budget is to run the clock well below the maximum clock speed that the circuit could support. Your task is to determine how much you could increase the clock speed if you add a clock-gating circuit. More precisely: *calculate the maximum clock speed at which you can run the circuit such that the total power with clock gating is the same as the power consumption now.*

Statistics of main circuit

Latency	10
Throughput	1
Average number of consecutive valid parcels	5
Average number of consecutive bubbles	20
Activity factor	25%
Clock frequency	300 MHz
Area	500 cells
Switching power	27.0 mW
Leakage power	5.0 mW
Short circuiting power	3.0 mW

Statistics of clock-gating circuit

Area	60 cells
Effectiveness	85%

NOTES:

1. The clock-gating circuit and the main circuit shall be run at the same clock speed.
2. Even with clock gating, the power budget will be the limiting factor for the clock speed. The delay through the circuit will *not* limit the clock speed.
3. All circuits shall use the same type of FPGA, same supply voltage, and same threshold voltage.
4. Clearly list any assumptions you use.

Answer:

Notation:

	Main circuit	*Clock-gating state machine*
Total power	$P_{m.tot}$	$P_{c.tot}$
Switching power	$P_{m.sw}$	$P_{c.sw}$
Short circuiting power	$P_{m.sh}$	$P_{c.sh}$
Leakage power	$P_{m.lk}$	$P_{c.lk}$
Activity factor	A_m	$A_c = A_m$
Area (Capacitance)	A_m	$A_c = A_m$
Clock speed	F	

1. *Assume: activity factor for clock gating circuit is same as original activity factor for main circuit.*
2. *The derivation that follows is aimed to be complete and detailed. Shorter answers are possible.*
3. *Combine $P_{m.sw}$ and $P_{m.sh}$ for dynamic power of main circuit, because switching and short-circuting power will be affected identically by clock gating.*

$$\begin{aligned} P_{m.dyn} &= P_{m.sw} + P_{m.sh} \\ &= 27.0 + 3.0 \\ &= 30.0 \end{aligned}$$

4. *Calculate activity factor of main circuit with clock gating.*

$$\begin{aligned} W &= \mathit{NumValid} + \mathit{NumBubbles} \\ &= 5 + 20 \\ &= 25 \end{aligned}$$

$$\begin{aligned} \mathit{MinClkEn} &= \mathit{NumValid} + \mathit{Latency} \\ &= 5 + 10 \\ &= 15 \end{aligned}$$

$$\begin{aligned} \mathit{MinPctClk} &= \mathit{MinClkEn}/W \\ &= 15/25 \\ &= 0.60 \end{aligned}$$

$$\begin{aligned} \mathit{PctClk} &= 1 - \mathit{Eff}(1 - MinPctClk) \\ &= 1 - 0.85(1 - 0.60) \\ &= 0.6600 \end{aligned}$$

5. Equations for power consumption in terms of original power consumption of main circuit

$$
\begin{aligned}
P'_{m.dyn} &= \tfrac{1}{2}A'_m F' C_m V^2 \\
\frac{P'_{m.dyn}}{P_{m.dyn}} &= \frac{\tfrac{1}{2}A'_m F' C_m V^2}{\tfrac{1}{2}A_m F C_m V^2} \\
&= \frac{A'_m F'}{A_m F} \\
P'_{m.dyn} &= \frac{A'_m F'}{A_m F} P_{m.dyn}
\end{aligned}
$$

$$
P'_{m.stat} = P_{m.stat}
$$

$$
\begin{aligned}
P_{c.dyn} &= \tfrac{1}{2}A_m F' C_c V^2 \\
\frac{P_{c.dyn}}{P_{m.dyn}} &= \frac{\tfrac{1}{2}A_m F' C_c V^2}{\tfrac{1}{2}A_m F C_m V^2} \\
&= \frac{F' C_c}{F C_m} \\
P_{c.dyn} &= \frac{F' C_c}{F C_m} P_{m.dyn}
\end{aligned}
$$

$$
P_{c.stat} = \frac{C_c}{C_m} P_{m.stat}
$$

6. Setup ratio of total power with and without clock gating, solve for ratio of clock speeds.

$$\begin{aligned}
P'_{tot} &= P_{tot} \\
P'_m + P_c &= P_m \\
P'_{m.dyn} + P_{m.stat} + P_{c.dyn} + P_{c.stat} &= P_{m.dyn} + P_{m.stat} \\
P'_{m.dyn} + P_{c.dyn} + P_{c.stat} &= P_{m.dyn} \\
\frac{A'_m F'}{A_m F} P_{m.dyn} + \frac{F' C_c}{F C_m} P_{m.dyn} + \frac{C_c}{C_m} P_{m.stat} &= P_{m.dyn} \\
\frac{F'}{F}\left(\frac{A'_m}{A_m} P_{m.dyn} + \frac{C_c}{C_m} P_{m.dyn}\right) + \frac{C_c}{C_m} P_{m.stat} &= P_{m.dyn} \\
\frac{F'}{F} &= \frac{P_{m.dyn} - \frac{C_c}{C_m} P_{m.stat}}{\frac{A'_m}{A_m} P_{m.dyn} + \frac{C_c}{C_m} P_{m.dyn}}
\end{aligned}$$

Equation for A'/A

$$\begin{aligned}
\frac{A'}{A} &= \textit{PctClk} \\
&= 0.6600
\end{aligned}$$

$$\begin{aligned}
A' &= 0.6600 \times A \\
&= 0.6600 \times 0.25 \\
&= 0.1650
\end{aligned}$$

$$\begin{aligned}
&= \frac{30 - \frac{60}{500} 5}{0.6600 \times 30 + \frac{60}{500} 30} \\
&= 1.3419
\end{aligned}$$

7. Final answer

$$\begin{aligned}
F' &= 1.3419F \\
&= 1.3419 \times 300 \\
&= 403\textit{MHz}
\end{aligned}$$

Marking:

1 mark *window*
1 mark *MinClkEn or MinPctClk*
1 mark *PctClk (used effectivenes)*
1 mark *reduced activity factor of main circuit*
1 mark *clock gating affects switching and short-circuting*
1 mark *clock gating does not affect leakage power*
1 mark *included all three types of power for clk-enable state machine*
1 mark *scaled power of clk-enable state machine by area*
1 mark *did not reduce activity factor of clk-enable state machine*
1 mark *new power scaled by increased clock freq*
1 mark *main circuit and clk-enable state machine use same clk speed*
1 mark *equation for equality of power*
1 mark *final answer*
1 mark *algebra*
1 mark *neatness and clarity*