pollev.com/kakiryan

COMP311: *COMPUTER ORGANIZATION!*

Lecture 17: Pipelining

tinyurl.com/comp311-fa25

PIPELINING

Recall: Min Clock Pd + Max Frequency

Min Clock Period

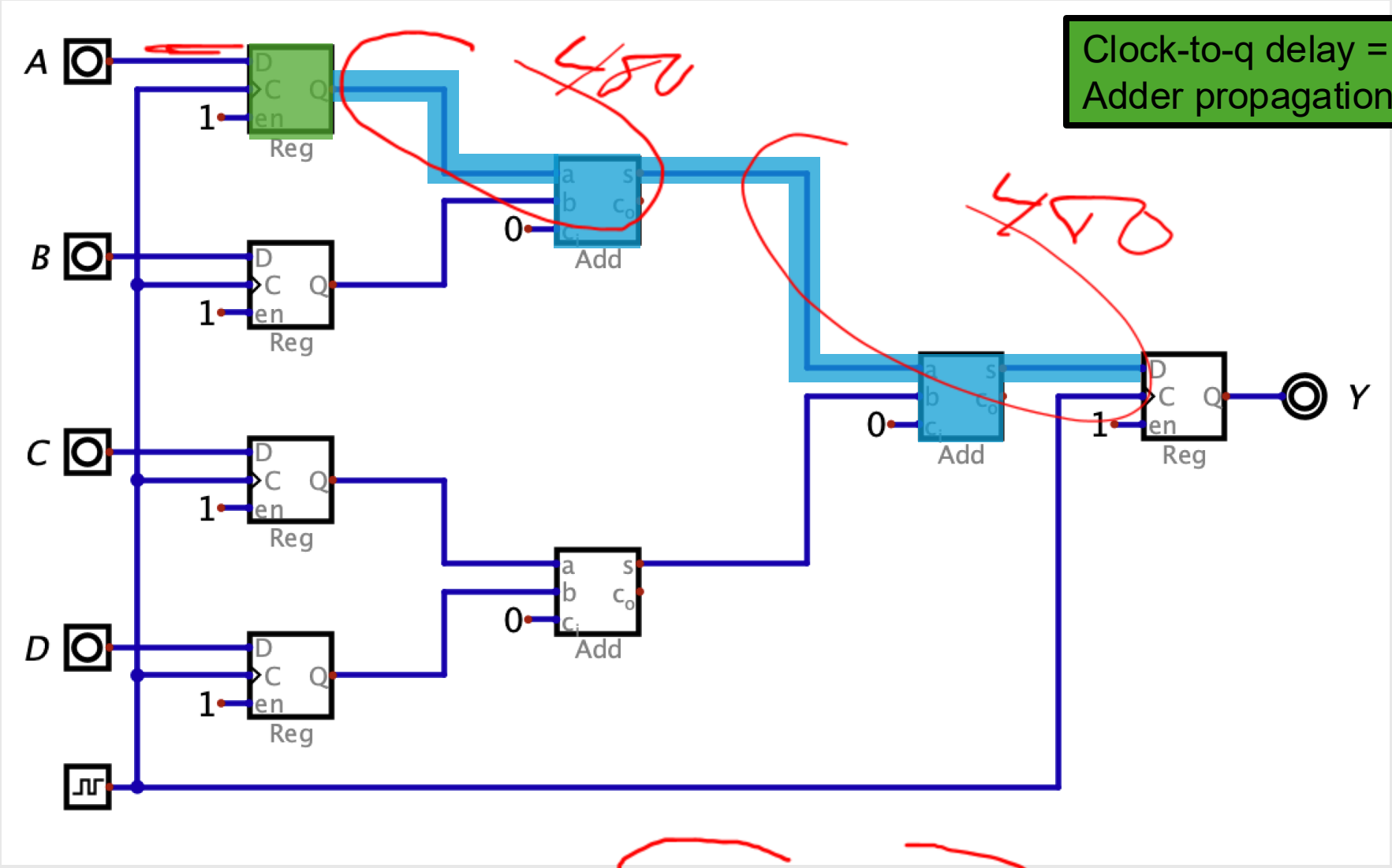
clk-to-q delay + longest combinational delay

Max Clock Frequency

1 / min period

The longest combinational path is the **critical path** or sequence of gates between two registers that has the **maximum propagation delay**

Single-Cycle: Longest Combinational Path



Min clock period = clk-to-q delay + longest comb path = 20ps + 960ps = 980ps

Clock Period

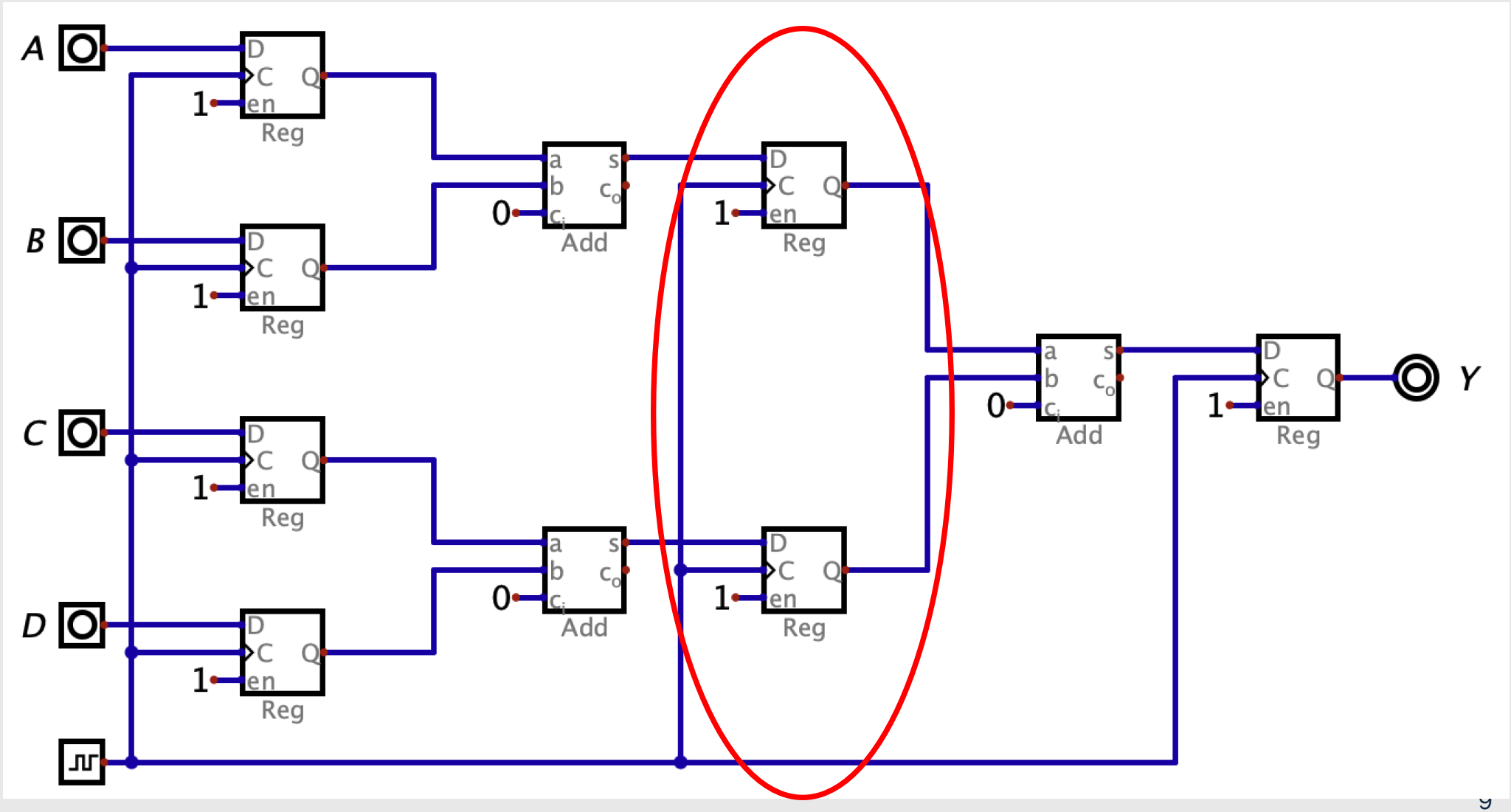
- At which of the following clock periods will the previous circuit work properly?
  - *1000ps*
  - *960ps*
  - *2000ps*
  - *4000ps*

Any clock period greater than or equal to the min clock period

Pipelining

- The previous circuit is referred to as a single-cycle implementation because it completes one operation within a single clock cycle.
- We'll improve the performance of our implementation by using a technique called pipelining
- We'll break down our circuit into **two stages** that can operate simultaneously
- This will make more efficient use of our hardware

Pipelining

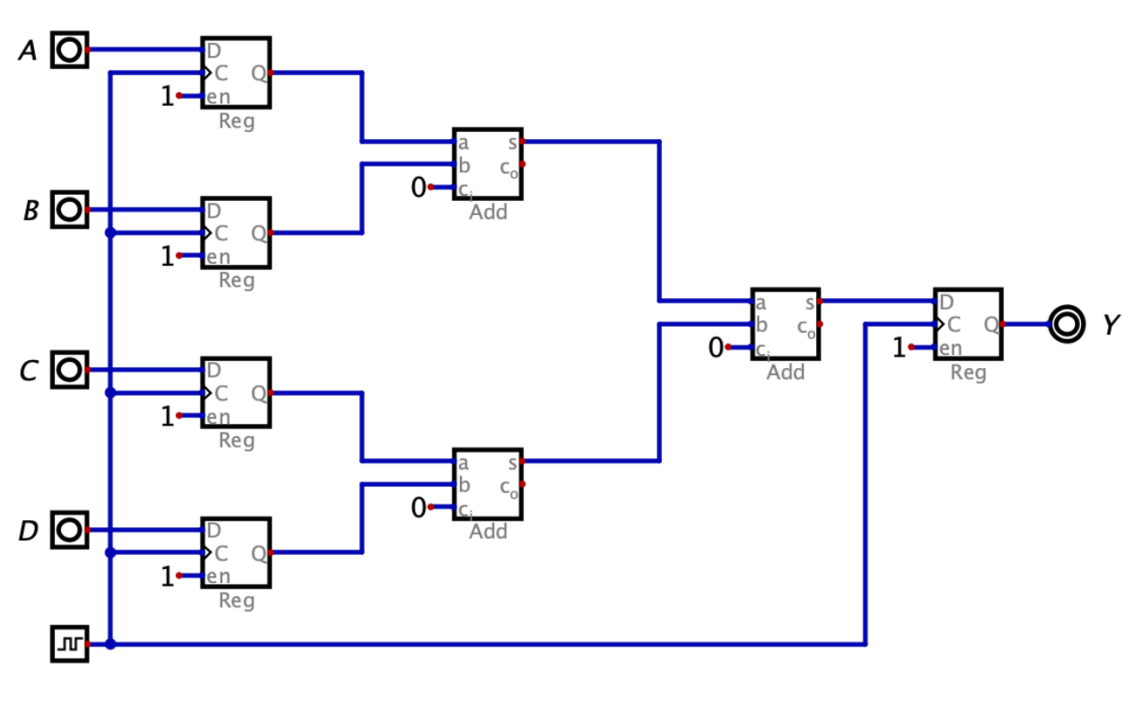


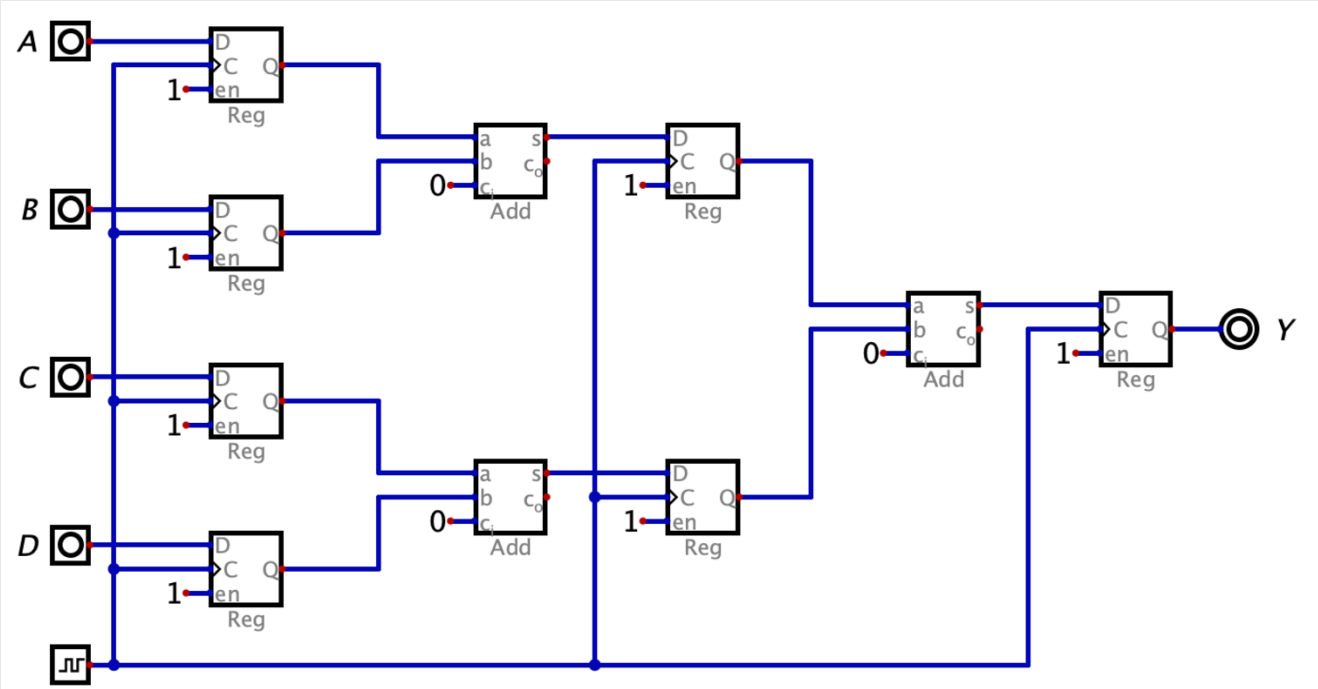
Single Cycle vs Pipelined

Single-Cycle Implementation

Pipelined Implementation

- Below are two circuits that perform the operation Y = A + B + C + D





Laundry Analogy: Single-Cycle

1. Put the first load of clothes in the washer.
2. When the washer is done, put the clothes in the dryer
3. When the dryer is done, take the clothes out. Then place the next load of laundry in the washer….

Let's say you have five loads of laundry and the washer and dryer each take one hour.

Time (h)	0	1	2	3	4	5	6	7	8	9	10
Load 0											
Load 1											
Load 2											
Load 3											
Load 4											

Laundry Analogy: Single-Cycle

It takes 2n hours to complete n loads of laundry

1. Put the first load of clothes in the washer.
2. When the washer is done, put the clothes in the dryer
3. When the dryer is done, take the clothes out. Then place the next load of laundry in the washer….

Let's say you have five loads of laundry and the washer and dryer each take one hour.

Time (h)	0	1	2	3	4	5	6	7	8	9	10
Load 0	W	D									
Load 1			W	D							
Load 2					W	D					
Load 3							W	D			
Load 4									W	D	

Laundry Analogy: Pipelined

1. Put the first load of clothes in the washer.
2. When the washer is done, put the clothes in the dryer. Place the next load in the washer.
3. When the dryer is done, take the clothes out. Then move the clothes from the washer into the dryer. Repeat.

Let's say you have five loads of laundry and the washer and dryer each take one hour.

Time (h)	0	1	2	3	4	5	6
Load 0							
Load 1							
Load 2							
Load 3							
Load 4							

Laundry Analogy: Pipelined

It takes n + 1 hours to complete n loads of laundry

1. Put the first load of clothes in the washer.
2. When the washer is done, put the clothes in the dryer. Place the next load in the washer.
3. When the dryer is done, take the clothes out. Then move the clothes from the washer into the dryer. Repeat.

Let's say you have five loads of laundry and the washer and dryer each take one hour.

Time (h)	0	1	2	3	4	5	6
Load 0	W	D					
Load 1		W	D				
Load 2			W	D			
Load 3				W	D		
Load 4					W	D	

Laundry Analogy: Pipelined

It takes n + 1 hours to complete n loads of laundry

1. Put the first load of clothes in the washer.
2. When the washer is done, put the clothes in the dryer. Place the next load in the washer.
3. When the [illegible] om the washer in[illegible]

Let's say you h[illegible] one hour.

With pipelining, we can overlap the execution of multiple instructions by breaking them into stages, similar to an assembly line. ☺

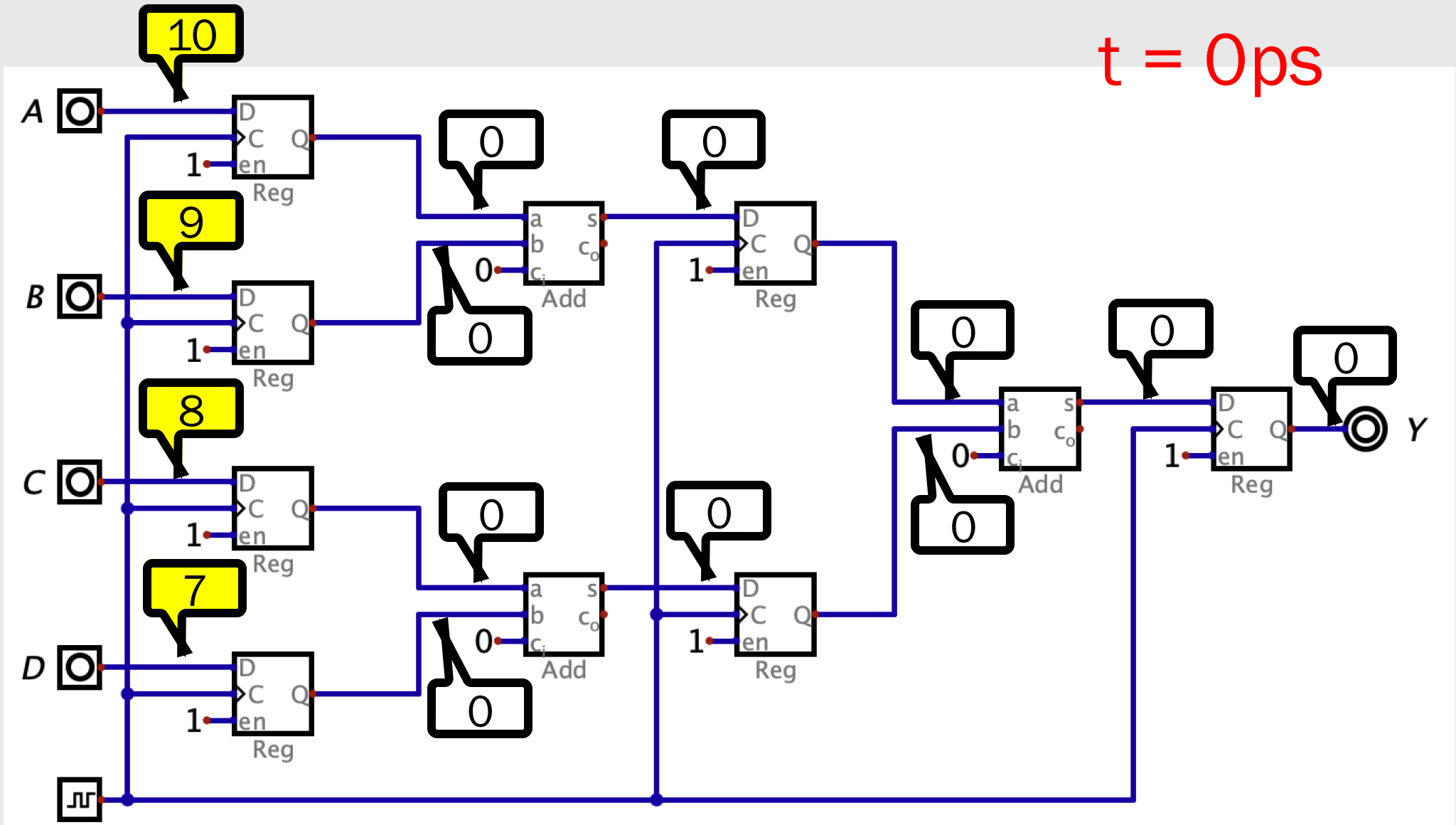
Tim[illegible]

L[illegible]						
Load 3				W	D	
Load 4					W	D

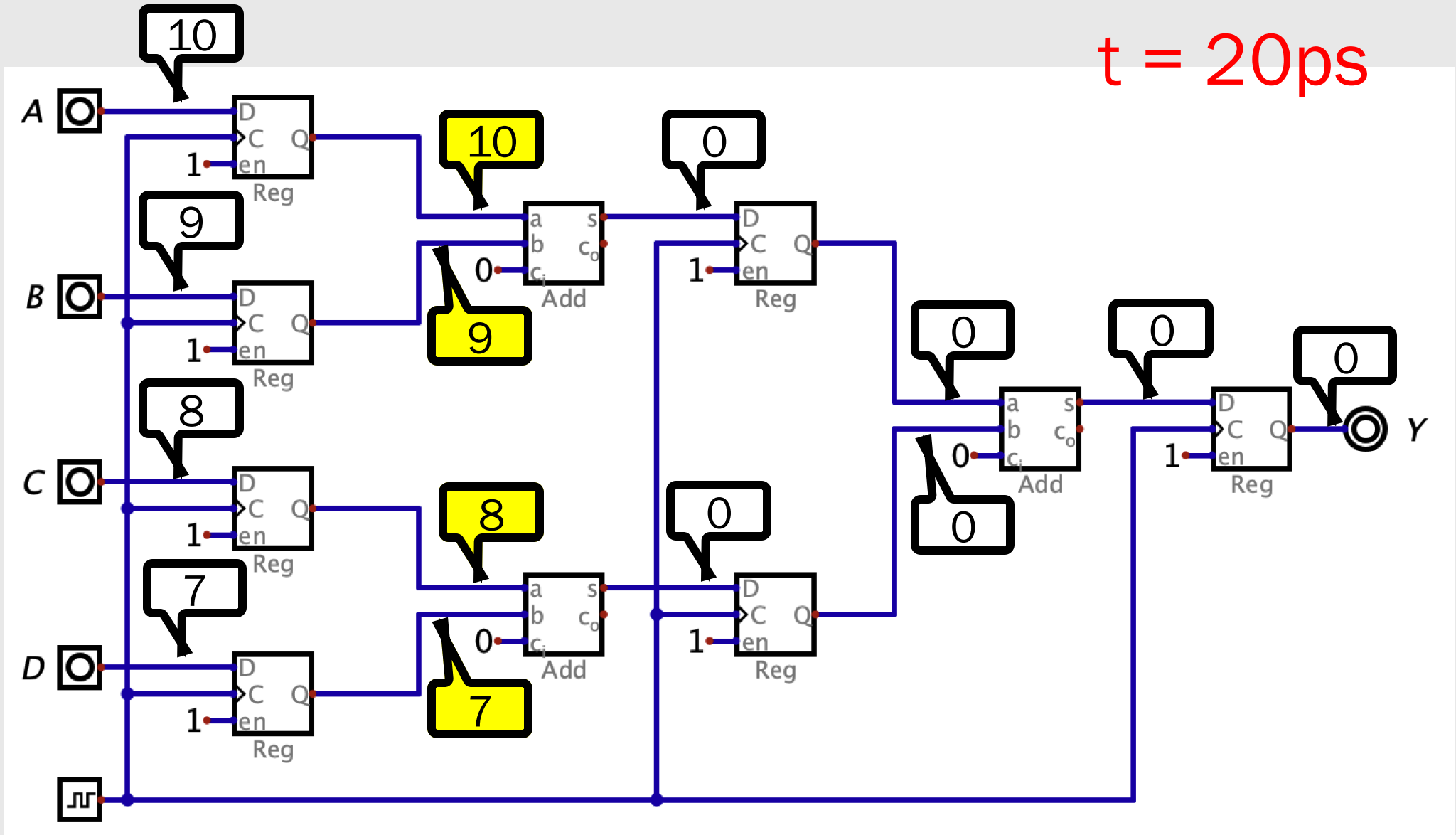
Pipelining

- Let's say that we want to perform the following operations
  - *10 + 9 + 8 + 7*
  - *5 + 4 + 3 + 2*
  - *10 + 10 + 20 + 20*
  - *3 + 3 + 4 + 4*
- We'll start at time t = 0
- We'll set our clock period to 500ps

Clock-to-q delay = 20ps
Adder propagation delay = 480ps

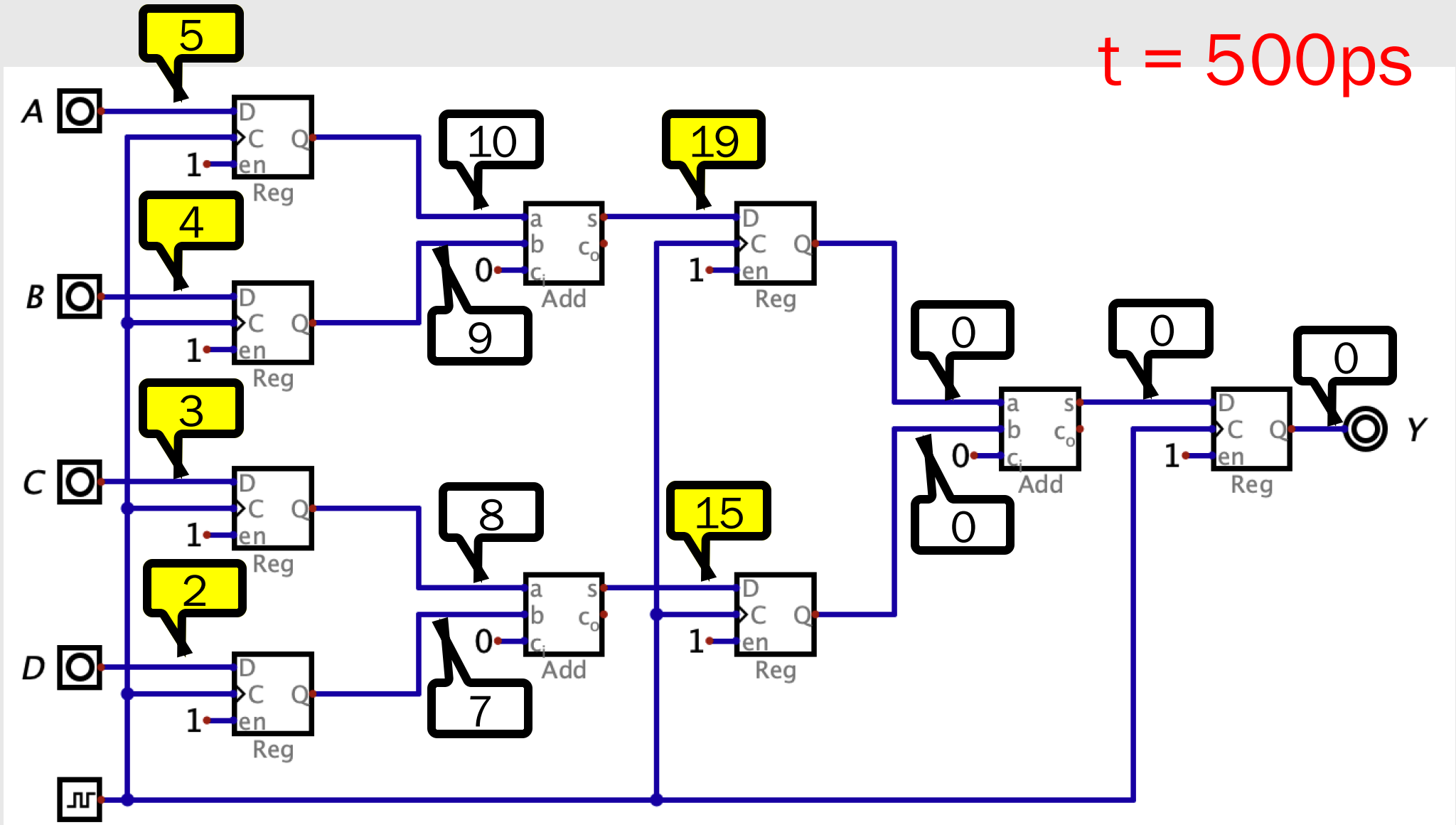


Clock-to-q delay = 20ps
Adder propagation delay = 480ps



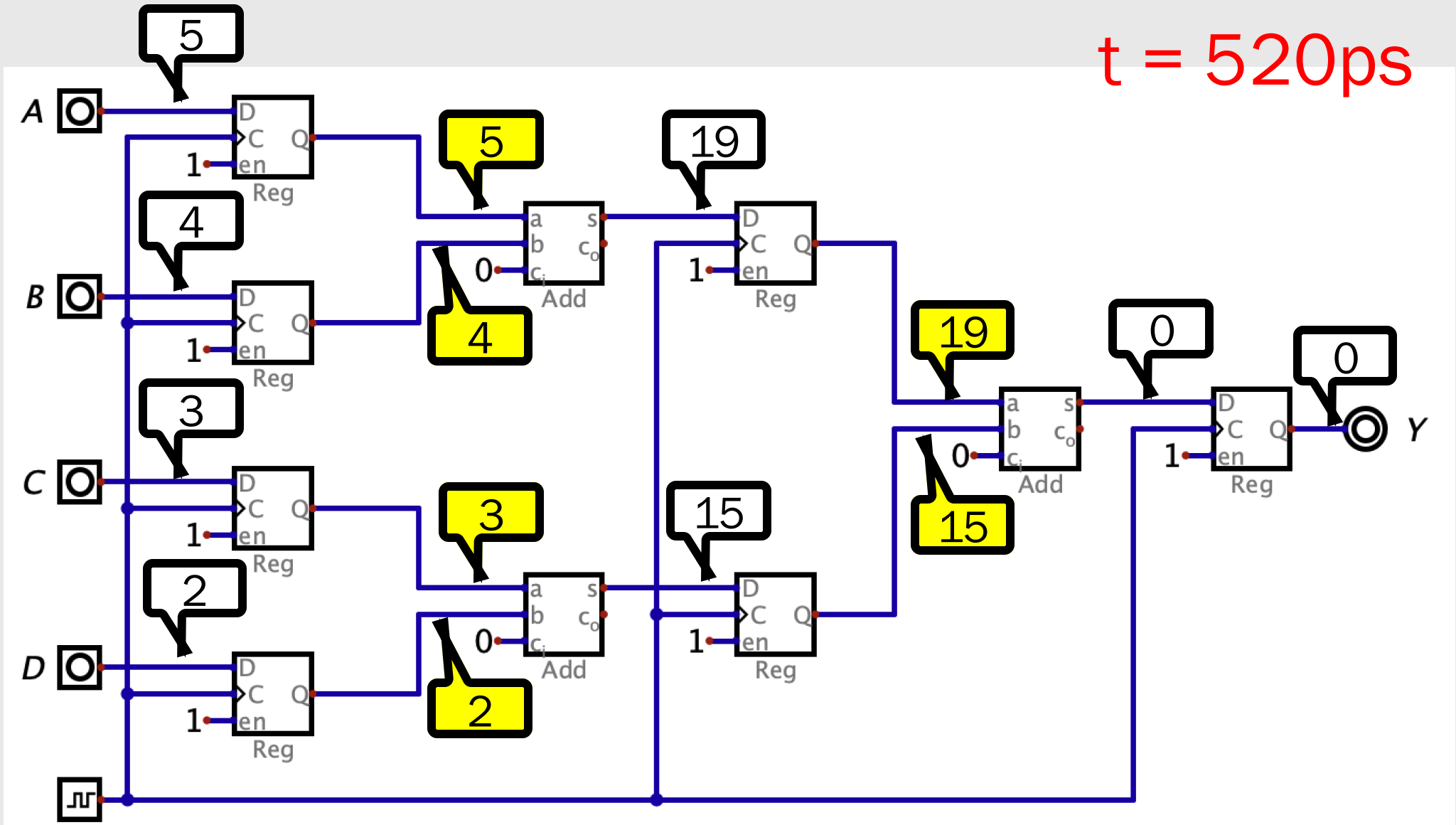
Clock-to-q delay = 20ps
Adder propagation delay = 480ps

t = 500ps



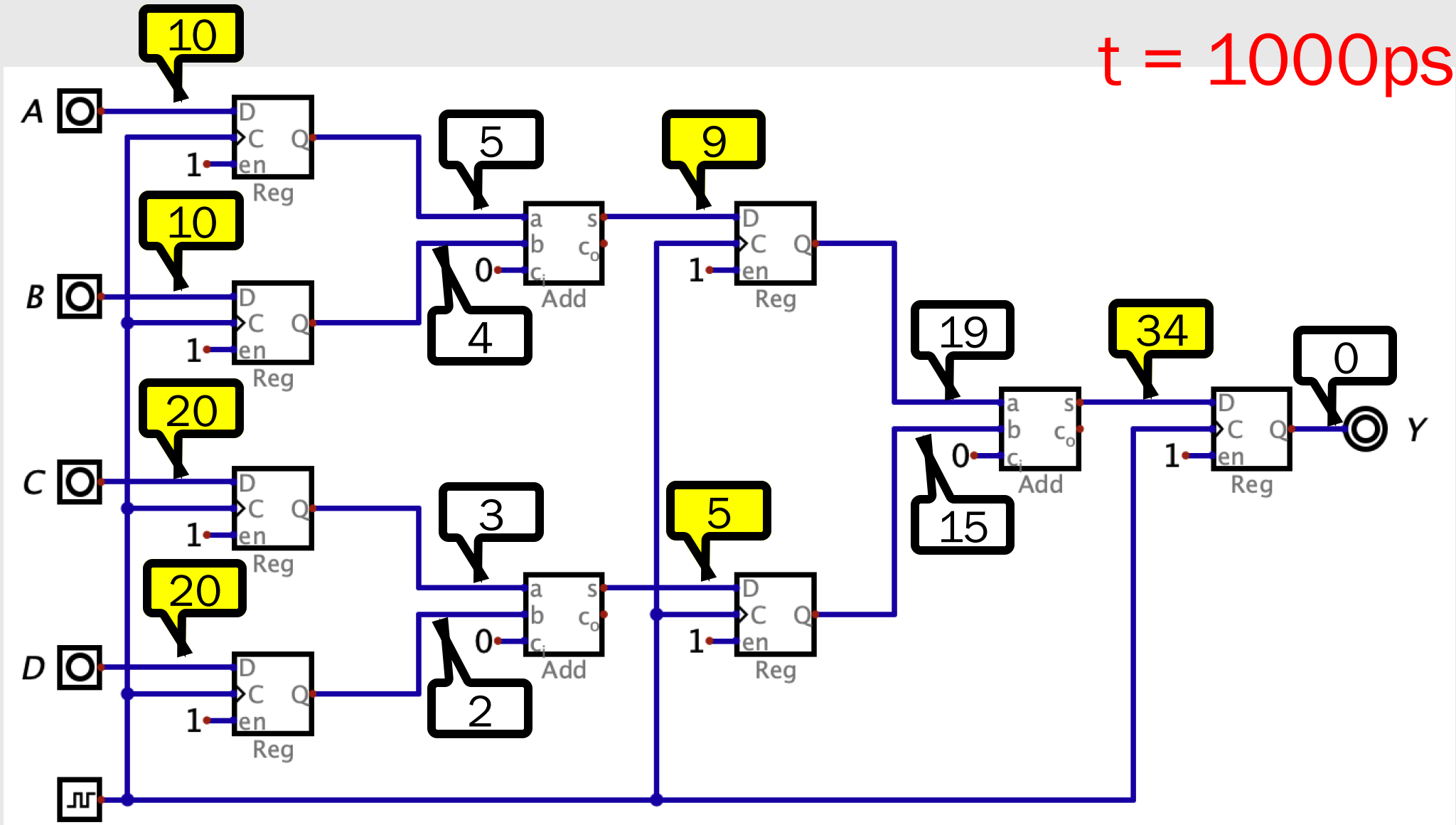
Clock-to-q delay = 20ps
Adder propagation delay = 480ps

t = 520ps



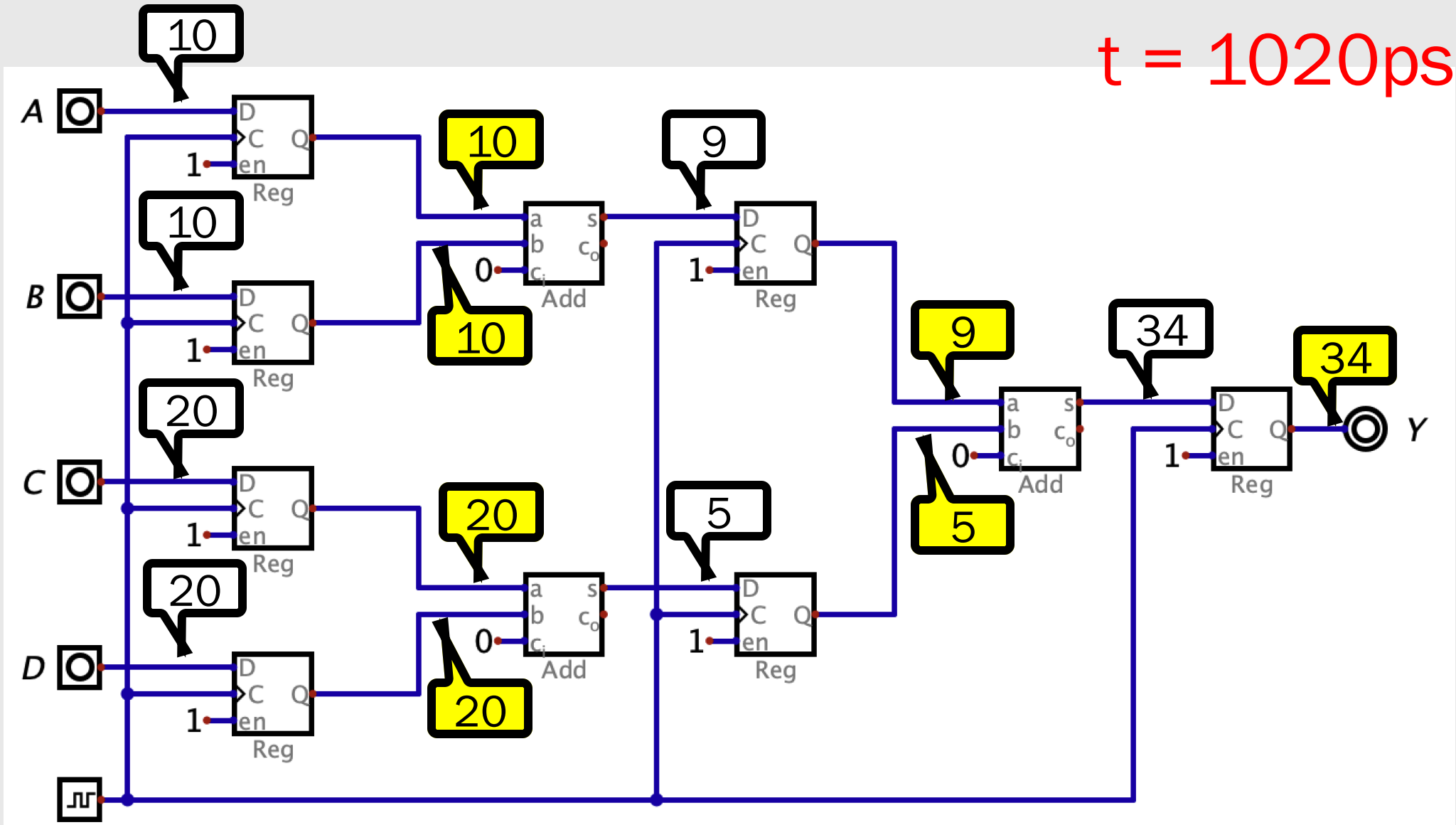
Clock-to-q delay = 20ps
Adder propagation delay = 480ps

t = 1000ps



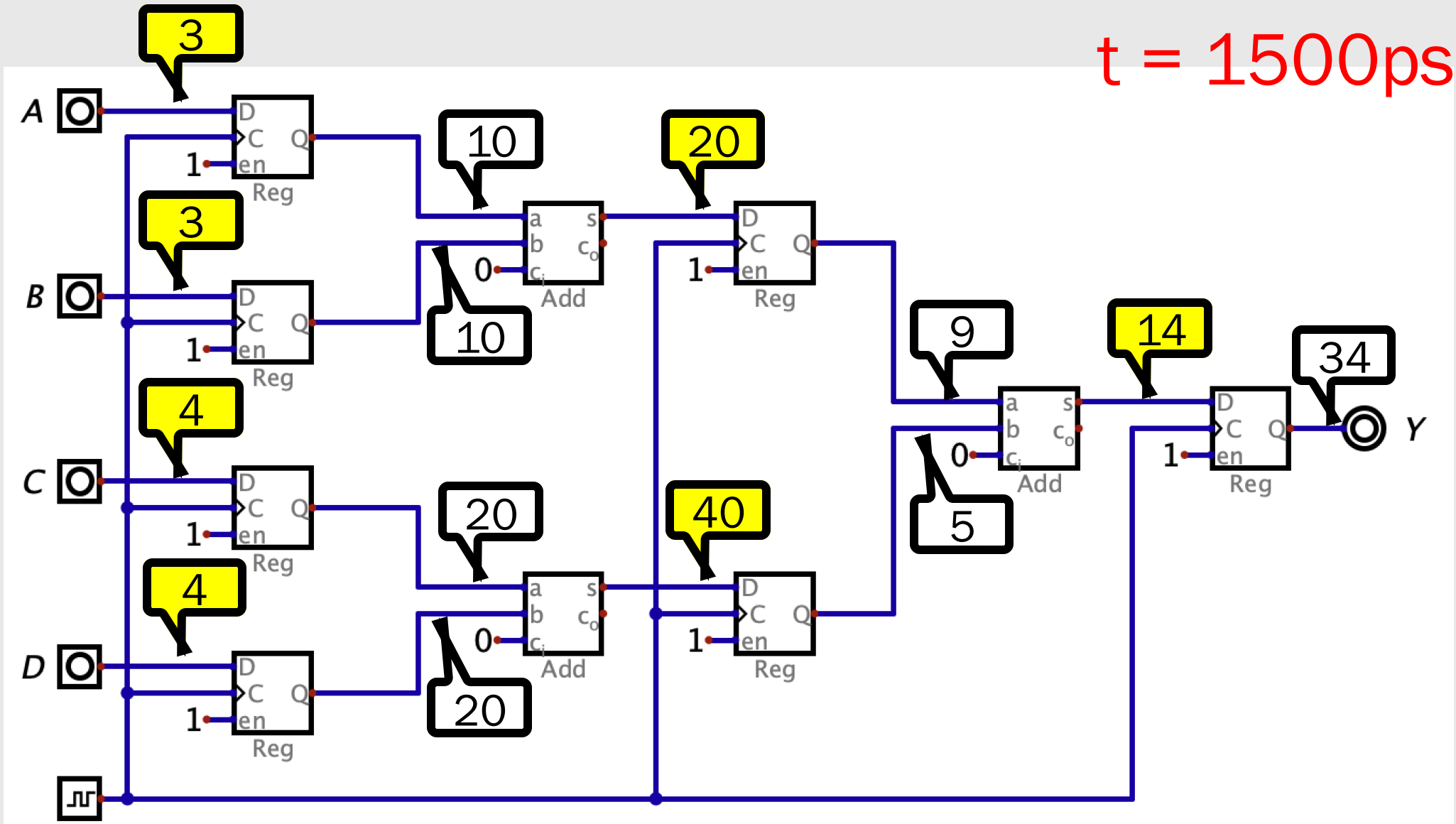
Clock-to-q delay = 20ps
Adder propagation delay = 480ps

t = 1020ps



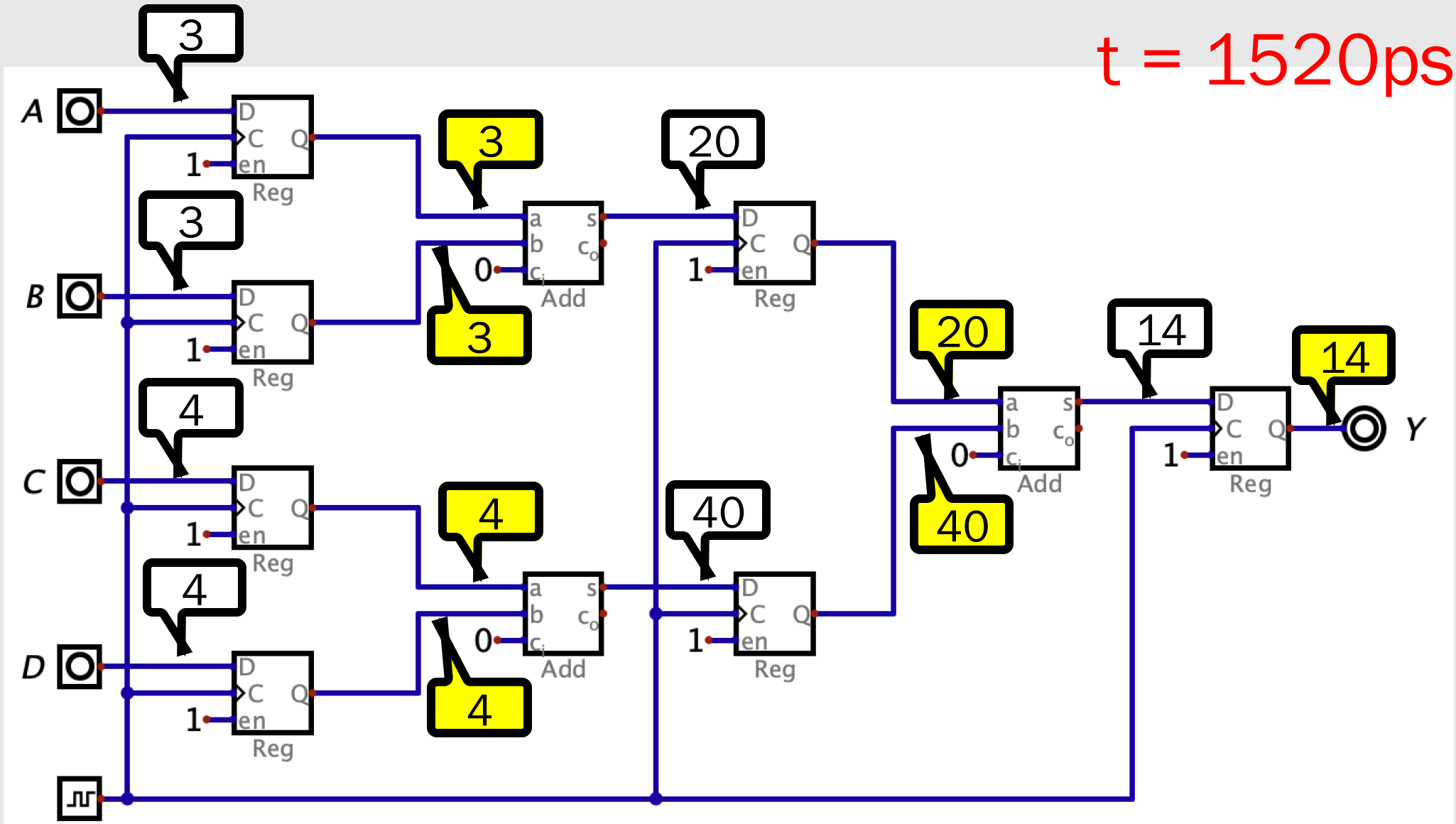
Clock-to-q delay = 20ps
Adder propagation delay = 480ps

t = 1500ps



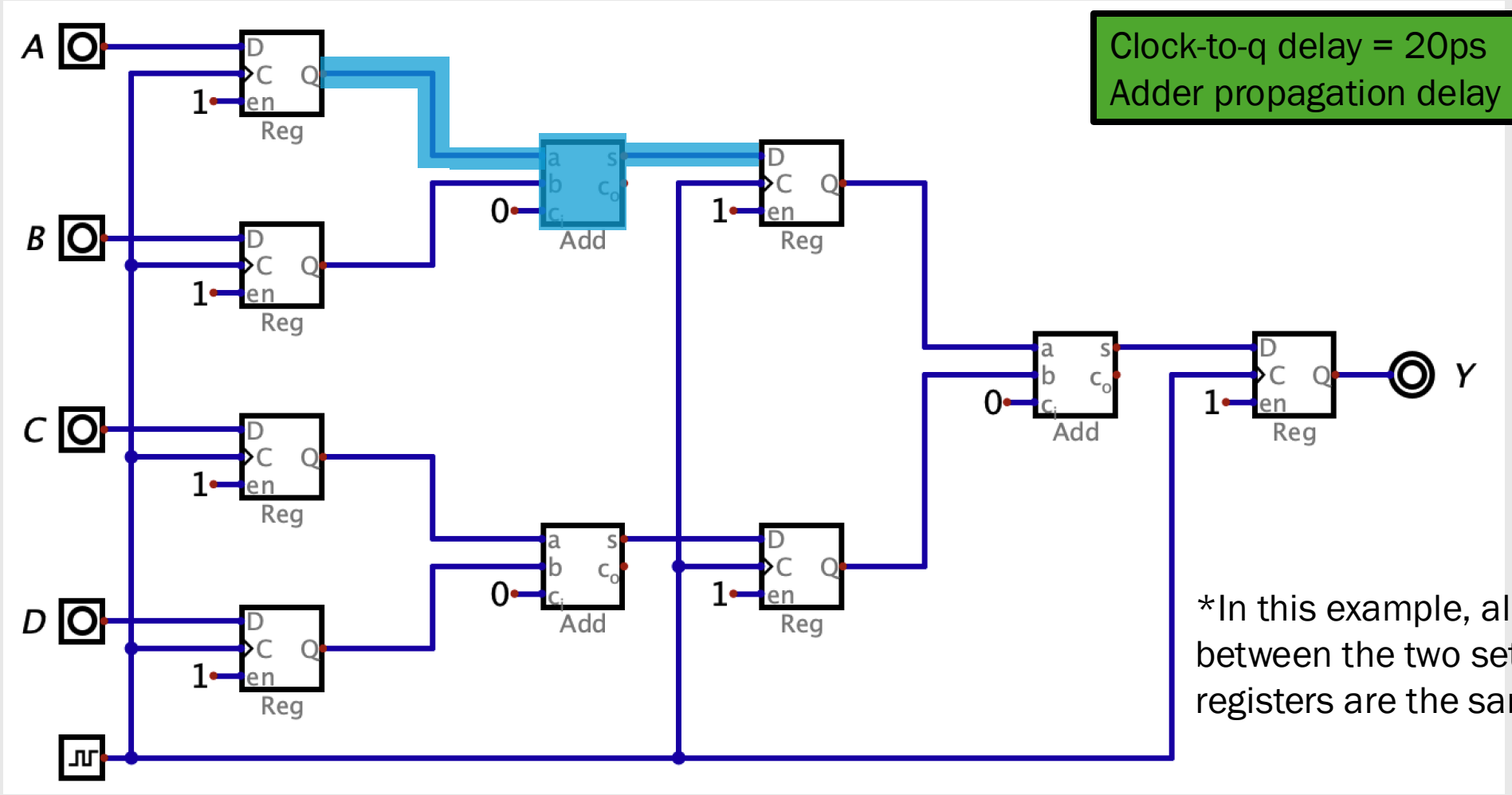
Clock-to-q delay = 20ps
Adder propagation delay = 480ps

t = 1520ps



Clock-to-q delay = 20ps
Adder propagation delay = 480ps

Pipelined: Longest Combinational Path

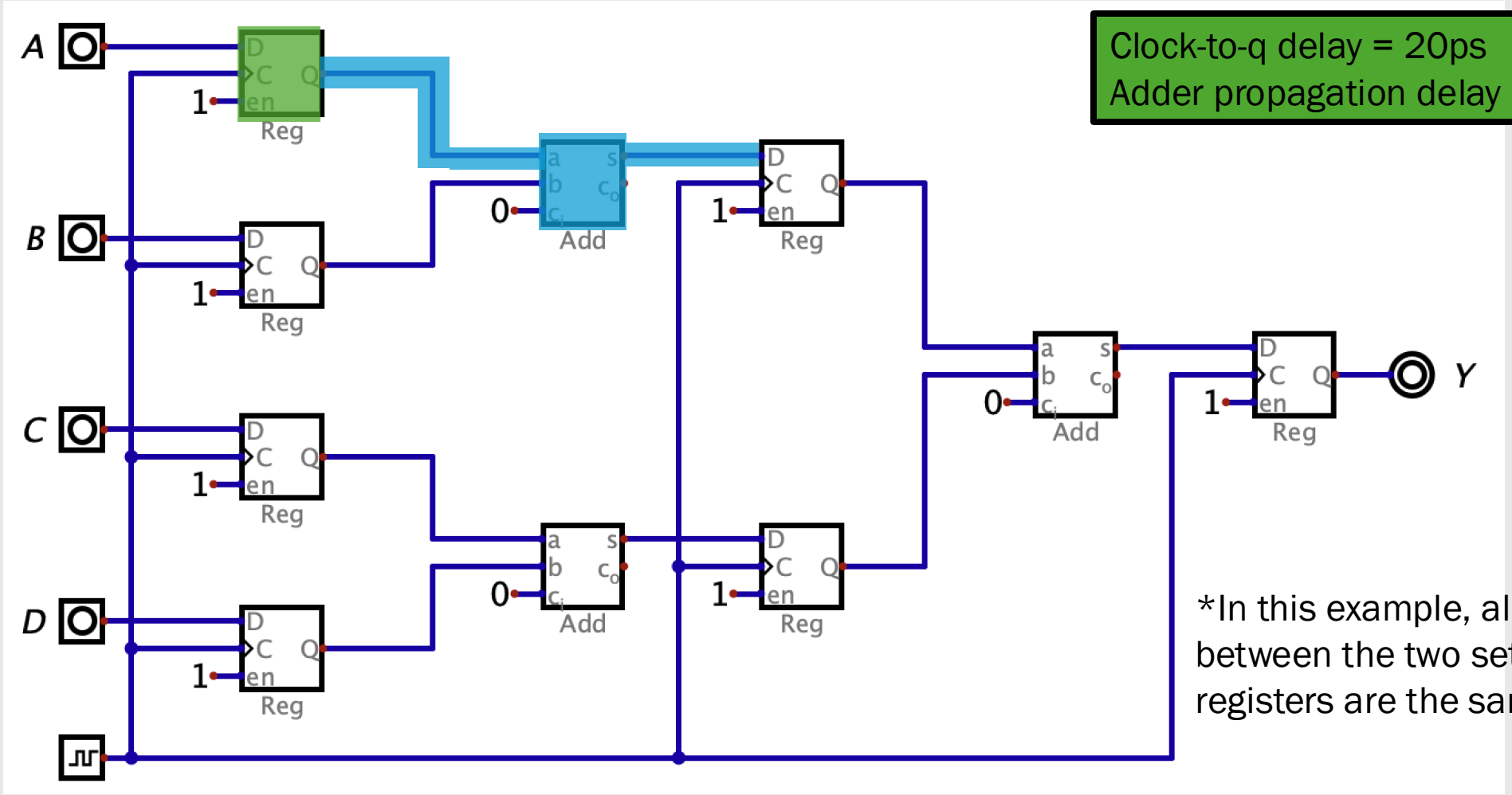


Clock-to-q delay = 20ps
Adder propagation delay = 480ps

*In this example, all paths between the two sets of registers are the same length

Longest comb path = 480ps

Pipelined: Min Clock Period = Critical Path



Min clock period = clk-to-q delay + longest comb path = 20ps + 480ps = 500ps

Clock Period

- At which of the following clock periods will the pipelined implementation work properly?
  - *480ps*
  - *580ps*
  - *1000ps*
  - *4000ps*

Clock Period

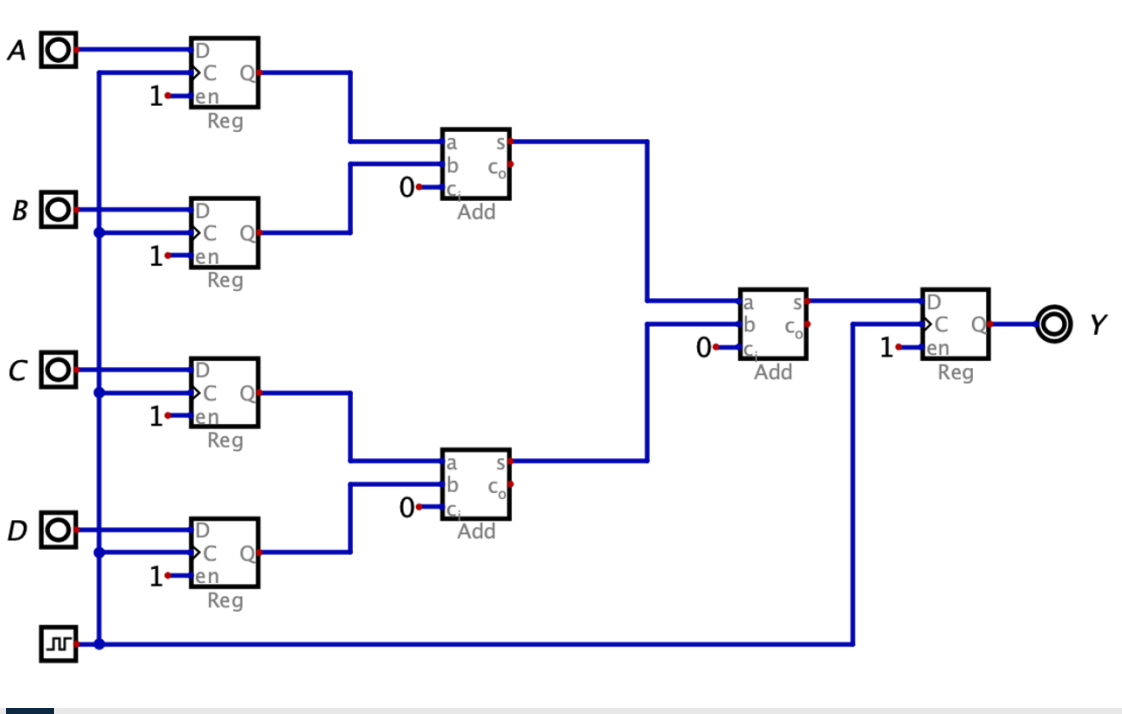
- At which of the following clock periods will the pipelined implementation work properly?
  - *480ps*
  - *580ps*
  - *1000ps*
  - *4000ps*

Any clock period greater than or equal to the min clock period

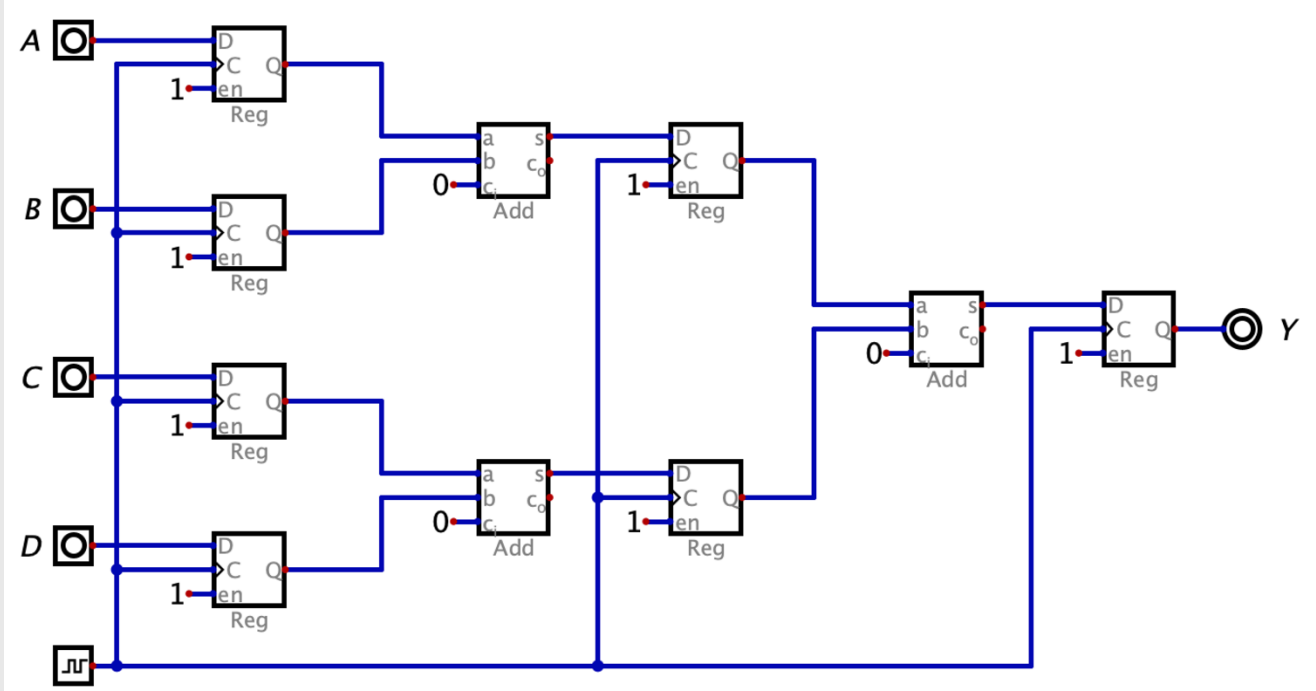
Single Cycle vs Pipelined

Clock-to-q delay = 20ps
Adder propagation delay = 480ps

Single-Cycle Implementation



Pipelined Implementation

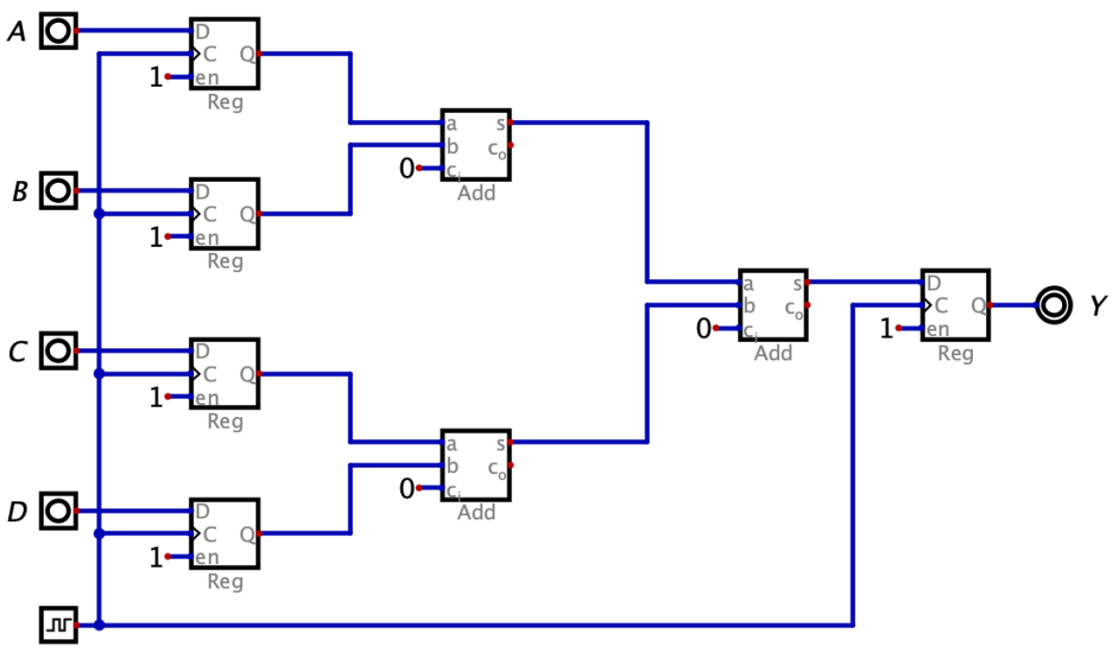


	Single-Cycle	Pipelined
Longest combinational path		
Min clock period		
# cycles to complete one operation		
Time to complete 10 ops at min clock speed		

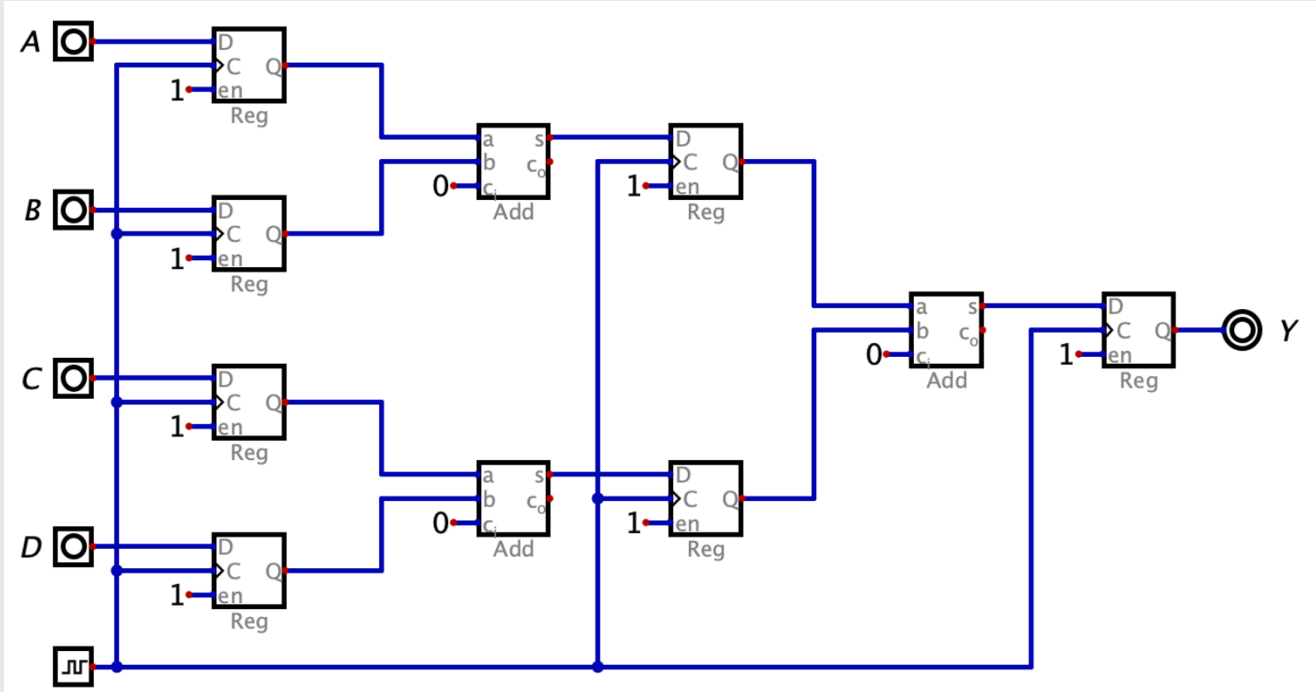
Single Cycle vs Pipelined

Clock-to-q delay = 20ps
Adder propagation delay = 480ps

Single-Cycle Implementation



Pipelined Implementation



	Single-Cycle	Pipelined
Longest combinational path	960ps	480ps
Min clock period	980ps	500ps
# cycles to complete one operation	1	2
Time to complete 10 ops at min clock speed	9800ps	5500ps

Single Cycle

1. What is the longest combinational path in this circuit?
   - *The delay through the two adders*
   - *480ps + 480ps = 960ps*
2. What is the min clock period?
   - *Clock-to-q delay + longest combinational delay*
   - *20ps + 960ps = 980ps*
3. If we set the clock to the min period, how many clock cycles does it take to perform one operation?
   - *1 cycle*
4. If we run this circuit at the minimum clock period, how long will it take to complete 10 operations?
   - *Final result is available at t = (980ps)(10) = 9800ps*
   - *Technically, the result is not available on Y until 9800ps + 20ps = 9820ps, but when we are talking about metrics, we keep it simple and just say 9800ps*

Pipelined

1. What is the longest combinational path in this circuit?
   - *The longest combinational delay between any two registers is through one adder*
   - *480ps*
2. What is the min clock period?
   - *Clock-to-q delay + longest combinational delay*
   - *20ps + 480 = 500ps*
3. If we set the clock to the min period, how many clock cycles does it take to perform one operation?
   - *2*
4. If we run this circuit at the minimum clock period, how long will it take to complete 10 operations?
   - *Final result is available at t = (500ps)(10+1) = 5500ps*
   - *Technically, the result is not available on Y until 5500ps + 20ps = 5520ps, but when we are talking about metrics, we keep it simple and just say 5500ps*

Single-Cycle vs Pipelined Execution Times

- To perform n operations
  - *Single-cycle: (n)(clock period)*
  - *Pipelined (n + 1)(clock period)*

Metrics

- Latency
  - *The amount of time that it takes to complete a single operation*
- Throughput
  - *The number of operations that can be completed in a given amount of time*
- Speedup
  - *The measure of how much faster the pipelined implementation is in comparison to the single-cycle implementation.*

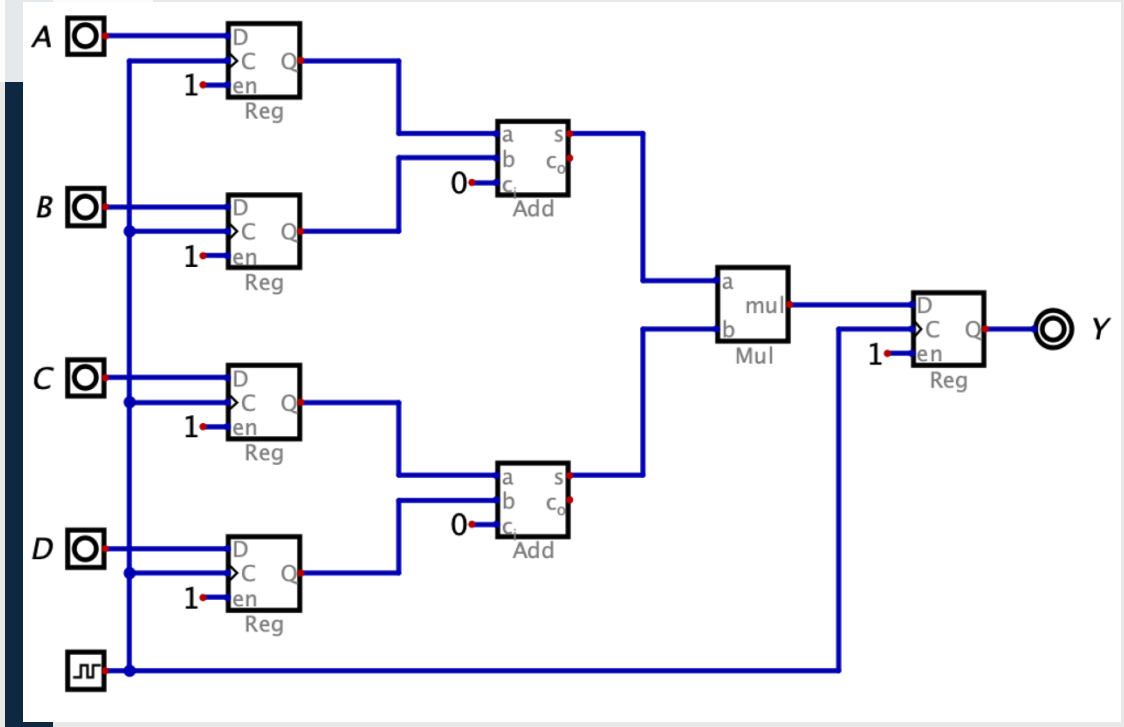
$$speedup = \frac{Time\ it\ takes\ to\ complete\ a\ given\ set\ of\ operations\ on\ the\ single\ cycle\ implementation}{Time\ it\ takes\ to\ complete\ the\ same\ set\ of\ operations\ on\ the\ pipelined\ implementation}$$

Performance Metrics

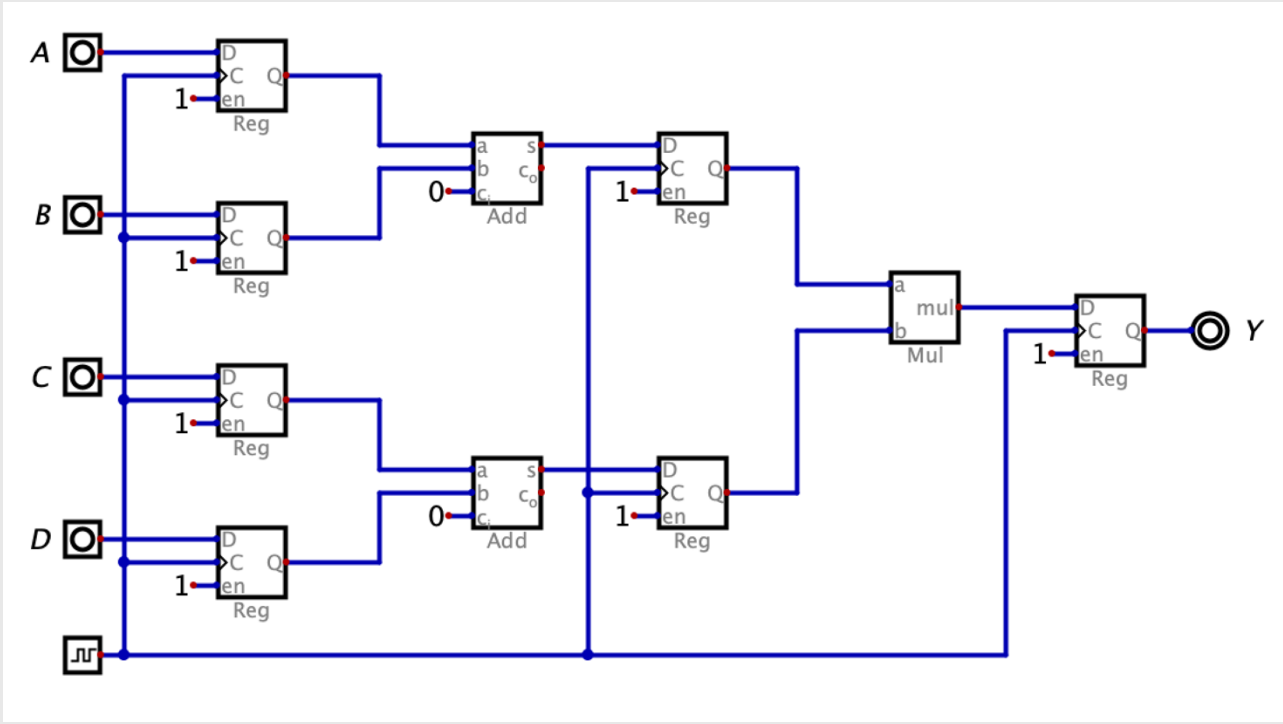
- What is the latency of an operation in the single-cycle implementation?
- What is the latency of an operation in the the pipelined implementation?
- Is the throughput higher in the single cycle example or the pipelined example?
- What is the speedup for executing 10 operations?
- If we had performed 20 operations, would the speedup be higher?
  - *Note, you shouldn't have to do any math for this*

Y = (A + B)(C + D)

- Let's compare two circuits that perform the operation Y = (A + B)(C + D)



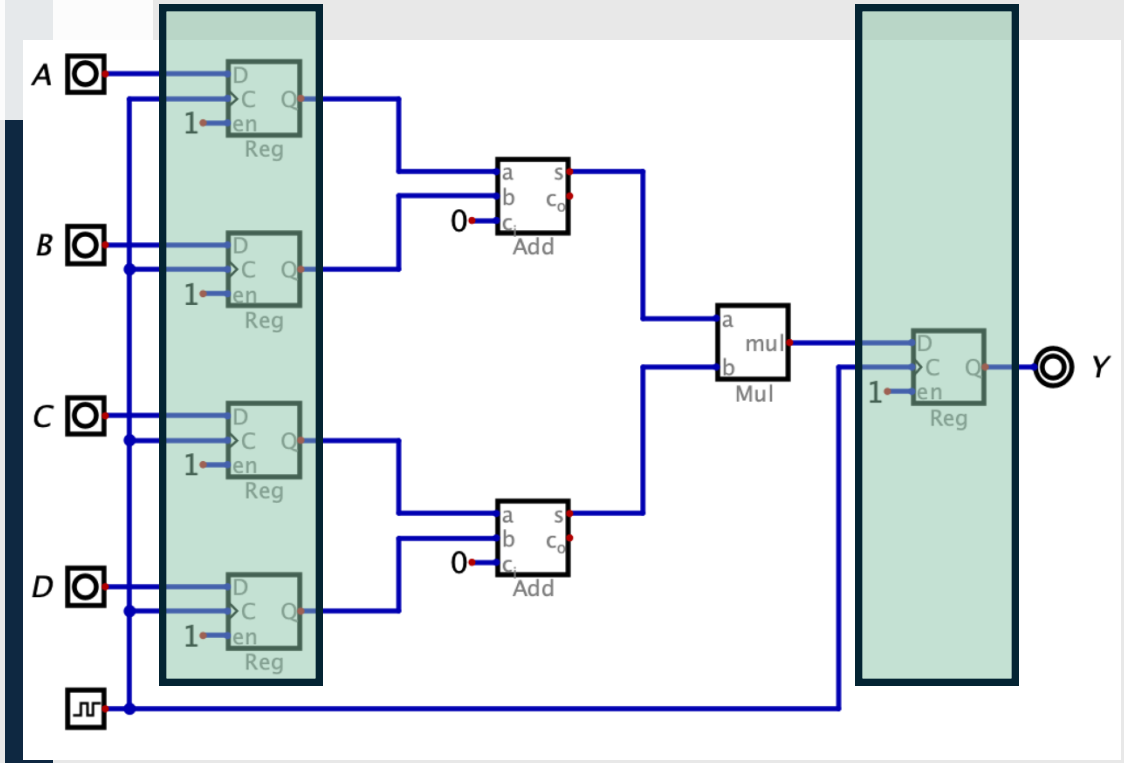
Single-cycle



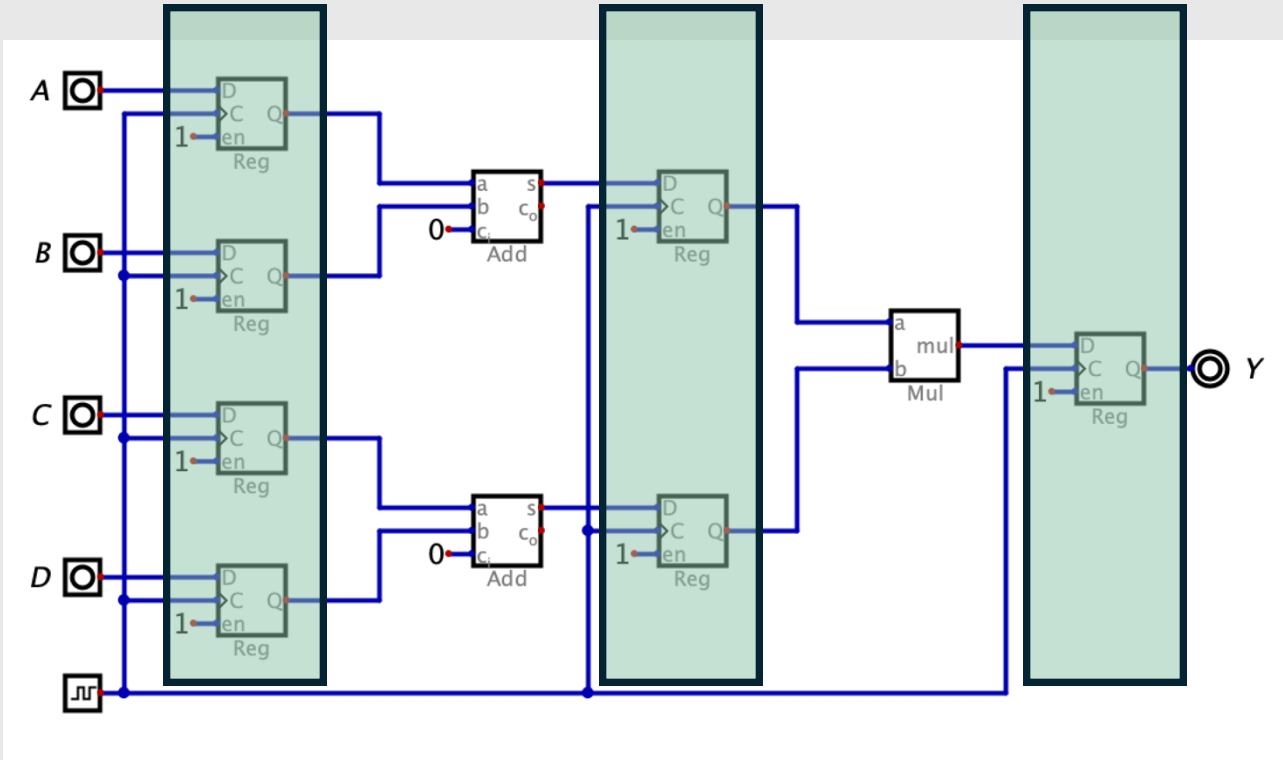
Pipelined

Y = (A + B)(C + D)

- Adding boxes around registers to make diagrams easier to read



Single-cycle

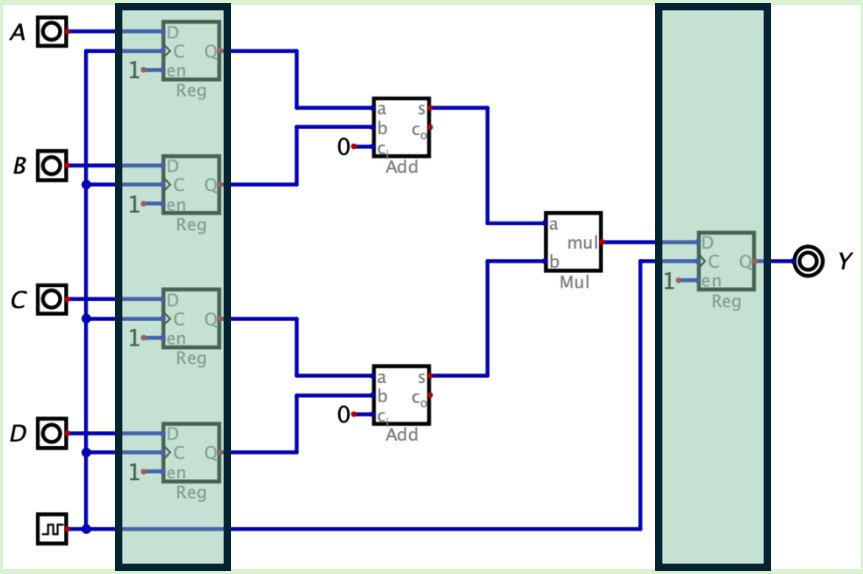


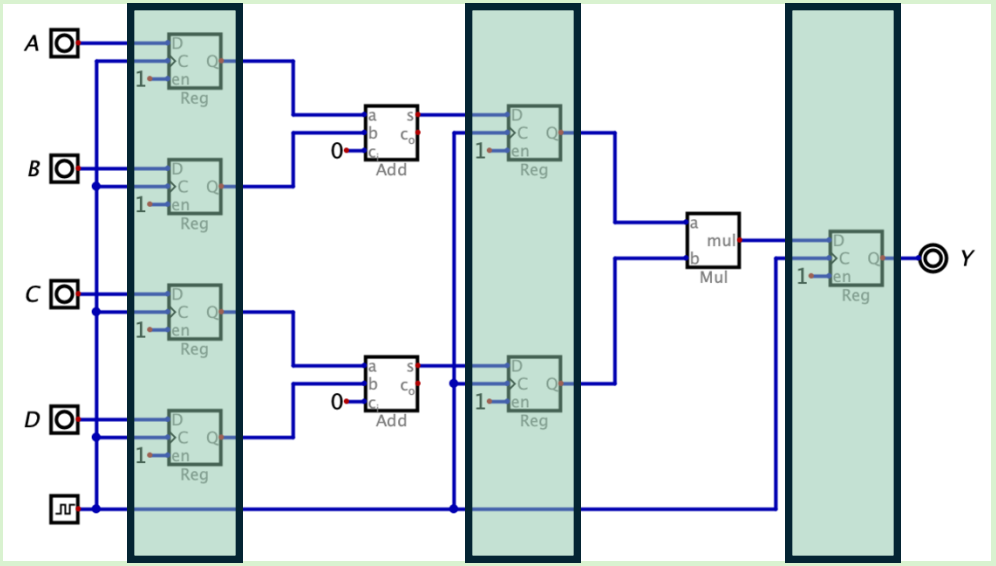
Pipelined

For each circuit…

Clock-to-q delay = 20ps
Adder propagation delay = 480ps
Multiplier propagation delay = 1200ps

1. What is the longest combinational path?
2. What is the min clock period (critical path)?
3. If the clock period is at least the min period, how many clock cycles does it take to perform one operation?
4. If we run this circuit at the minimum clock period, how long will it take to complete 10 operations?





Performance Metrics

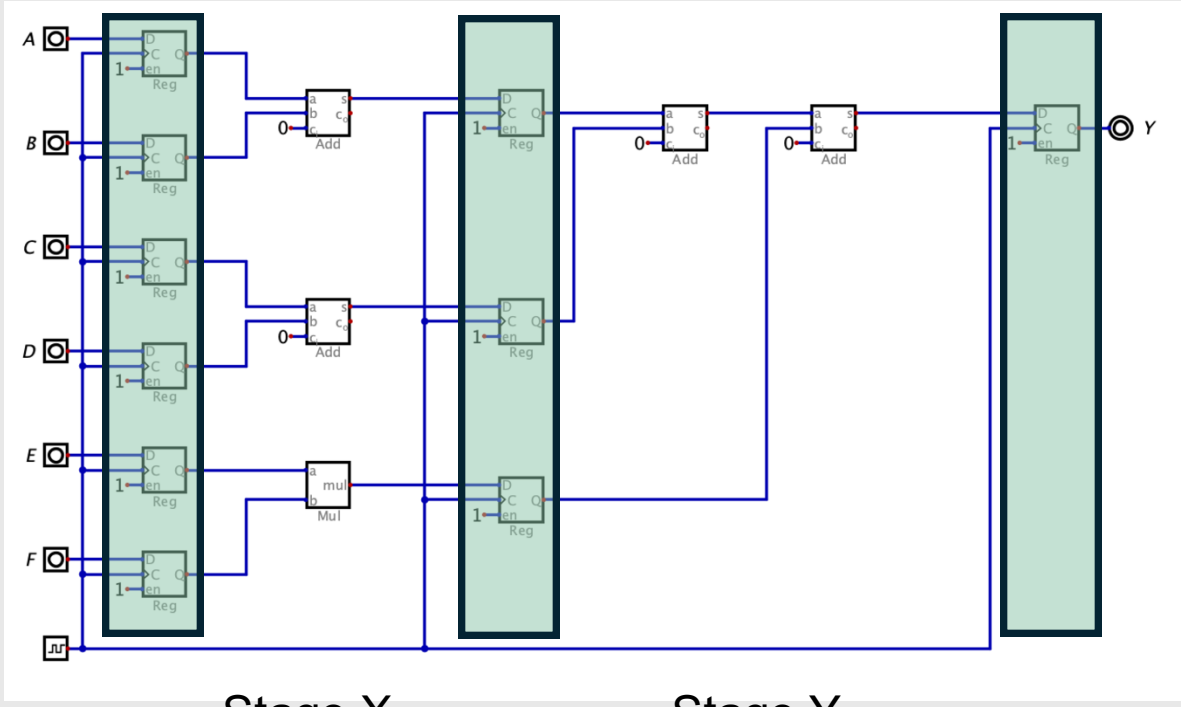
- What is the latency of an operation in the single-cycle implementation?
- What is the latency of an operation in the the pipelined implementation?
- Is the throughput higher in the single cycle example or the pipelined example?
- What is the speedup for executing 10 operations?

Performance Metrics

- What is the latency of an operation in the single-cycle implementation?
  - *One clock cycle = 1700ps*
- What is the latency of an operation in the pipelined implementation?
  - *2 clock cycles = 2(1220ps) = 2440ps*
- Is the throughput higher in the single cycle example or the pipelined example?
  - *Pipelined*
- Speedup
  - *17000ps / 13420ps = 1.27*

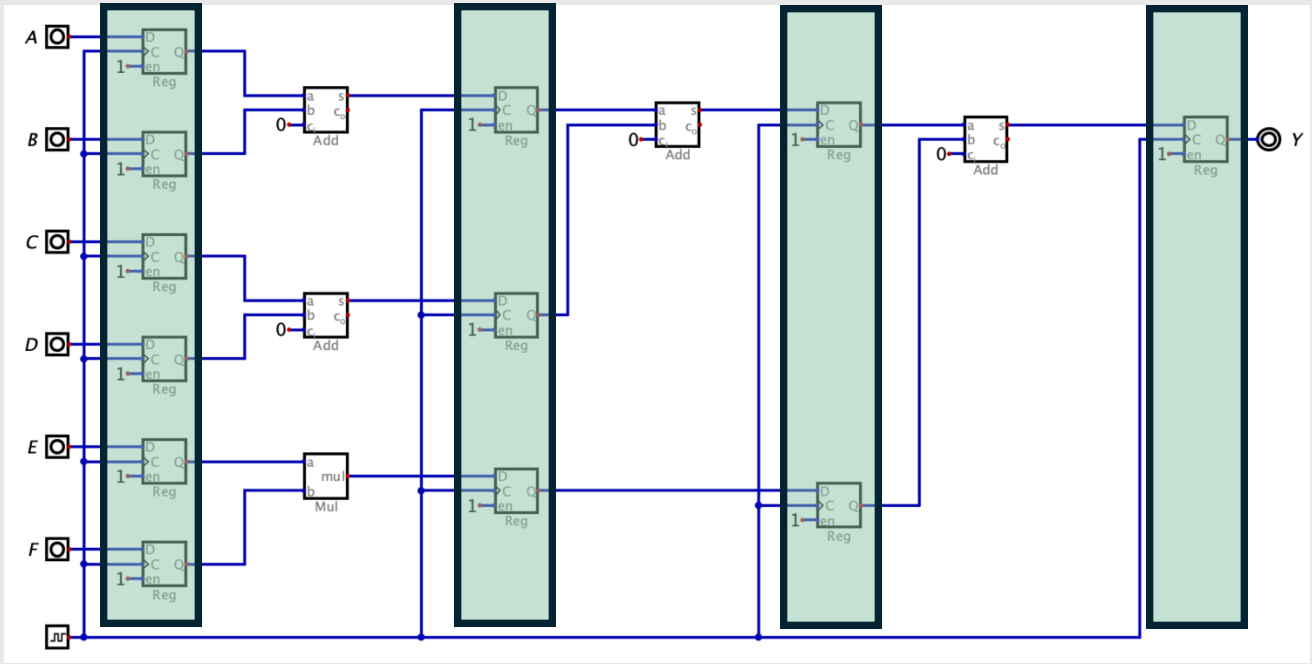
Y = A + B + C + D + EF

- Let's compare two circuits that perform the operation Y = A + B + C + D + EF



Stage X Stage Y

2-stage Pipeline



Stage X Stage Y Stage Z

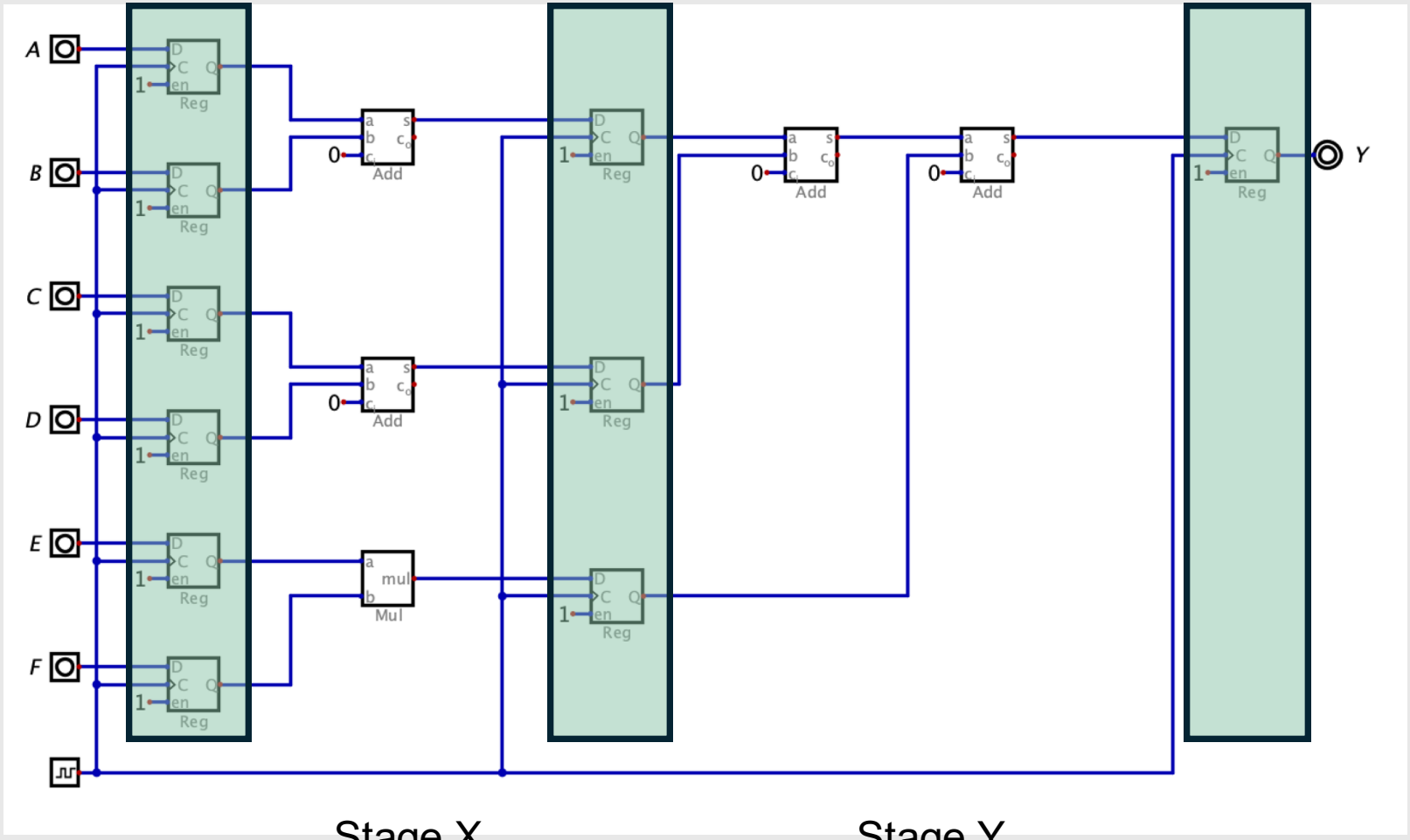
3-stage Pipeline

2-Stage Pipeline

Clock-to-q delay = 20ps
Adder propagation delay = 480ps
Multiplier propagation delay = 1200ps

Let's say we want to perform the following operation:
Y = 1 + 2 + 3 + 4 + (5)(6)

- In Cycle 0, we will compute

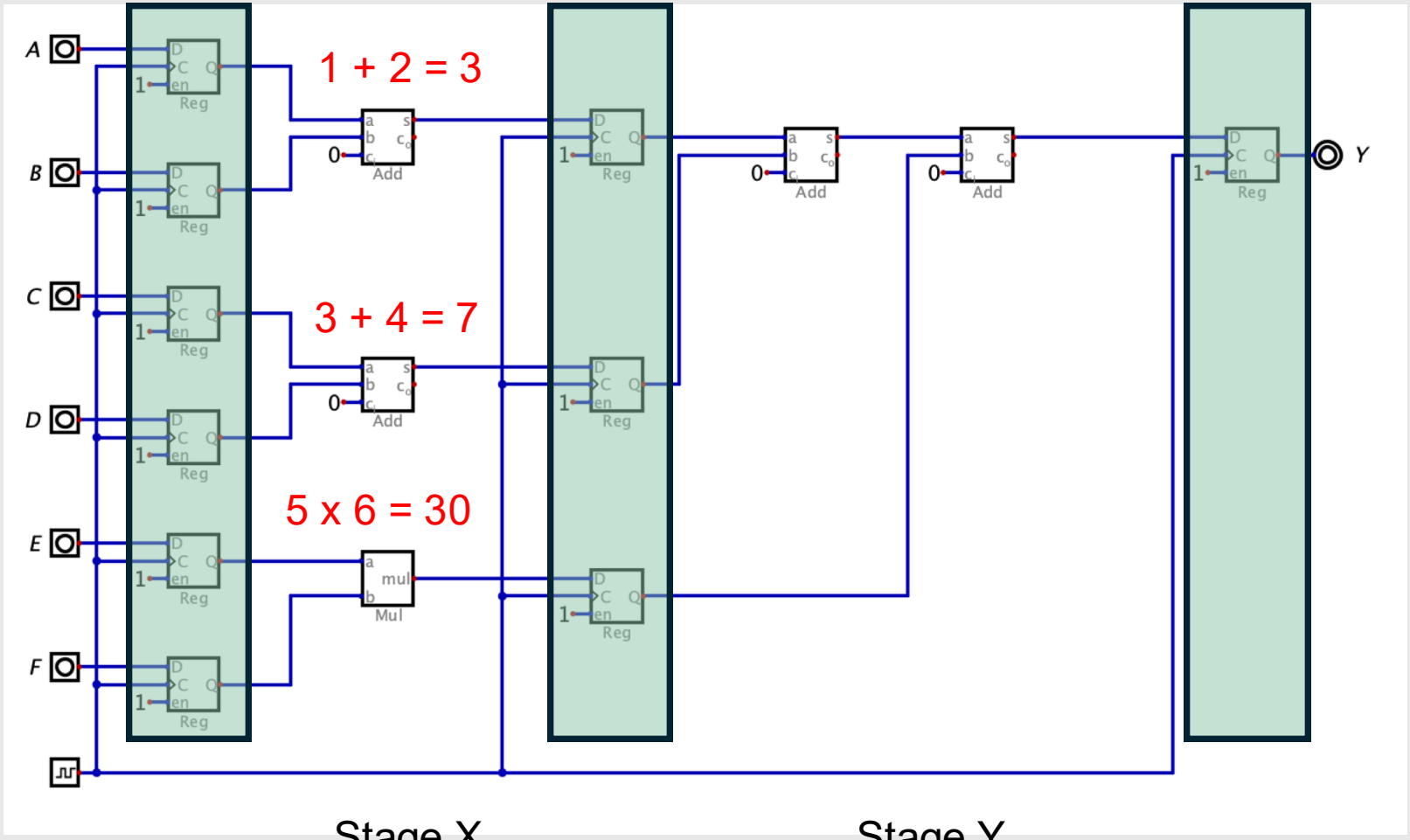


2-Stage Pipeline

Clock-to-q delay = 20ps
Adder propagation delay = 480ps
Multiplier propagation delay = 1200ps

Let's say we want to perform the following operation:
Y = 1 + 2 + 3 + 4 + (5)(6)

- In Cycle 0, we will compute
  - 1 + 2 = 3
  - 3 + 4 = 7
  - 5 x 6 = 30

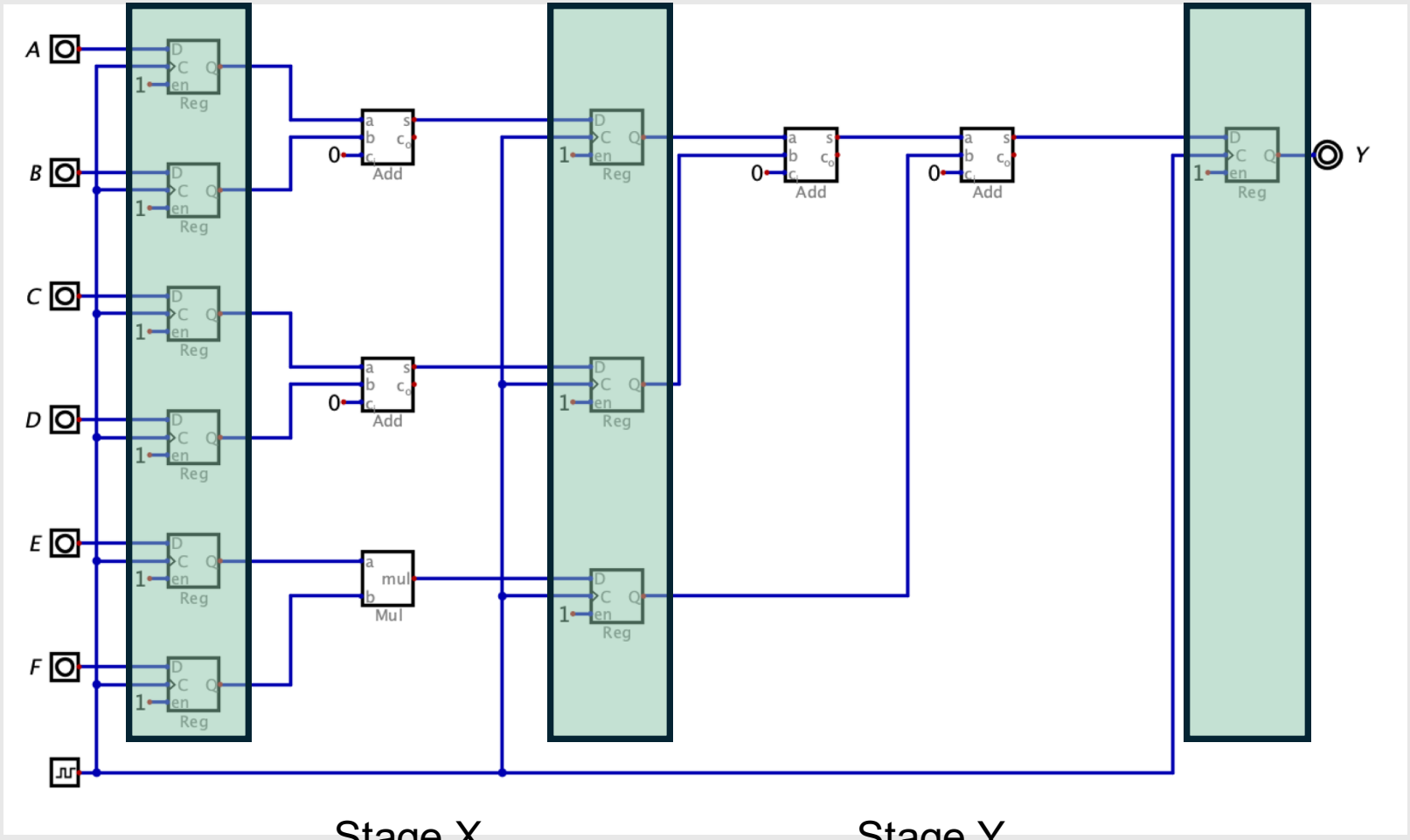


2-Stage Pipeline

Clock-to-q delay = 20ps
Adder propagation delay = 480ps
Multiplier propagation delay = 1200ps

Let's say we want to perform the following operation:
Y = 1 + 2 + 3 + 4 + (5)(6)

- In Cycle 0, we will compute
  - 1 + 2 = 3
  - 3 + 4 = 7
  - 5 x 6 = 30
- In Cycle 1, we will compute

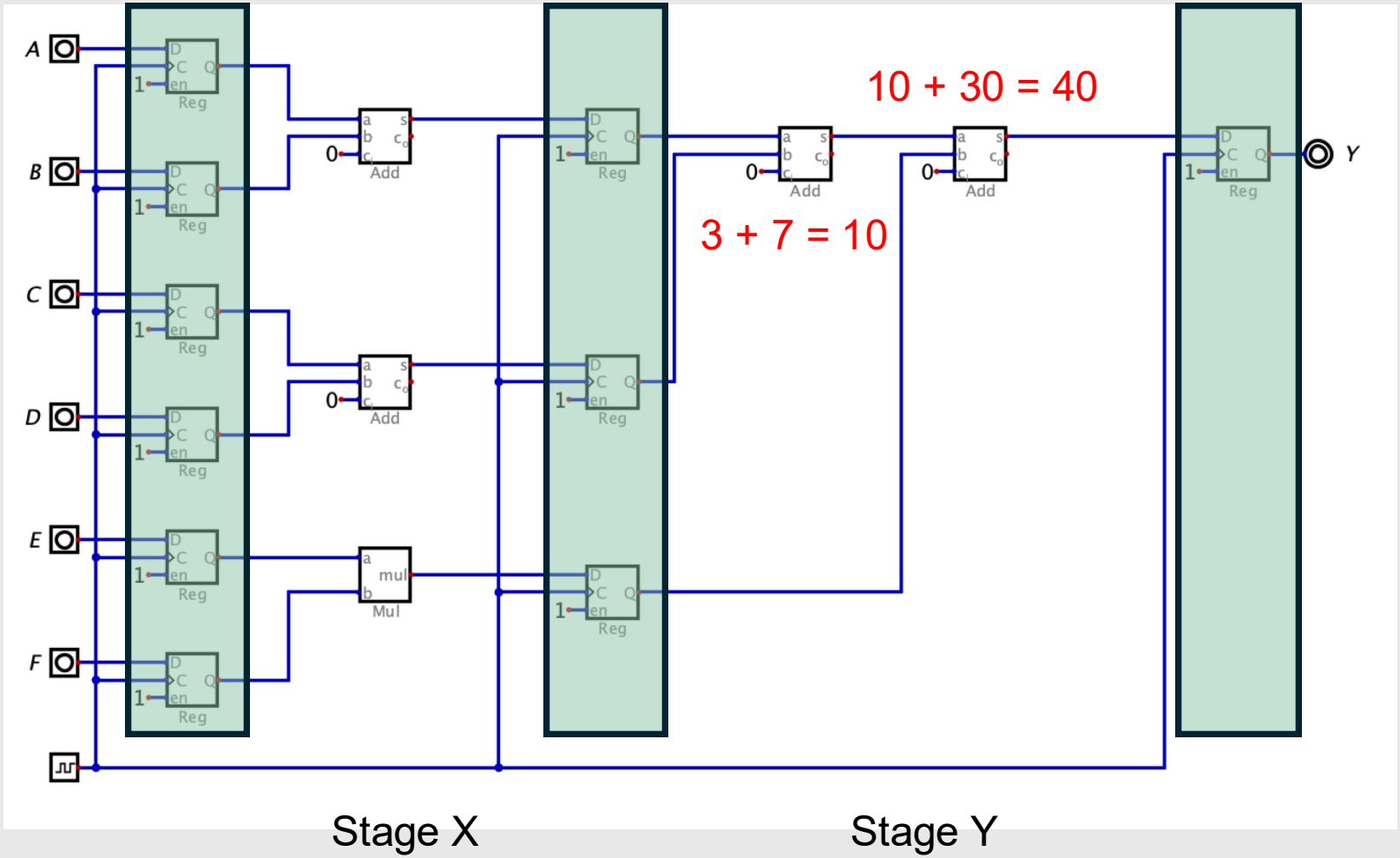


2-Stage Pipeline

Clock-to-q delay = 20ps
Adder propagation delay = 480ps
Multiplier propagation delay = 1200ps

Let's say we want to perform the following operation:
Y = 1 + 2 + 3 + 4 + (5)(6)

- In Cycle 0, we will compute
  - 1 + 2 = 3
  - 3 + 4 = 7
  - 5 x 6 = 30
- In Cycle 1, we will compute
  - 3 + 7 + 30 = 40

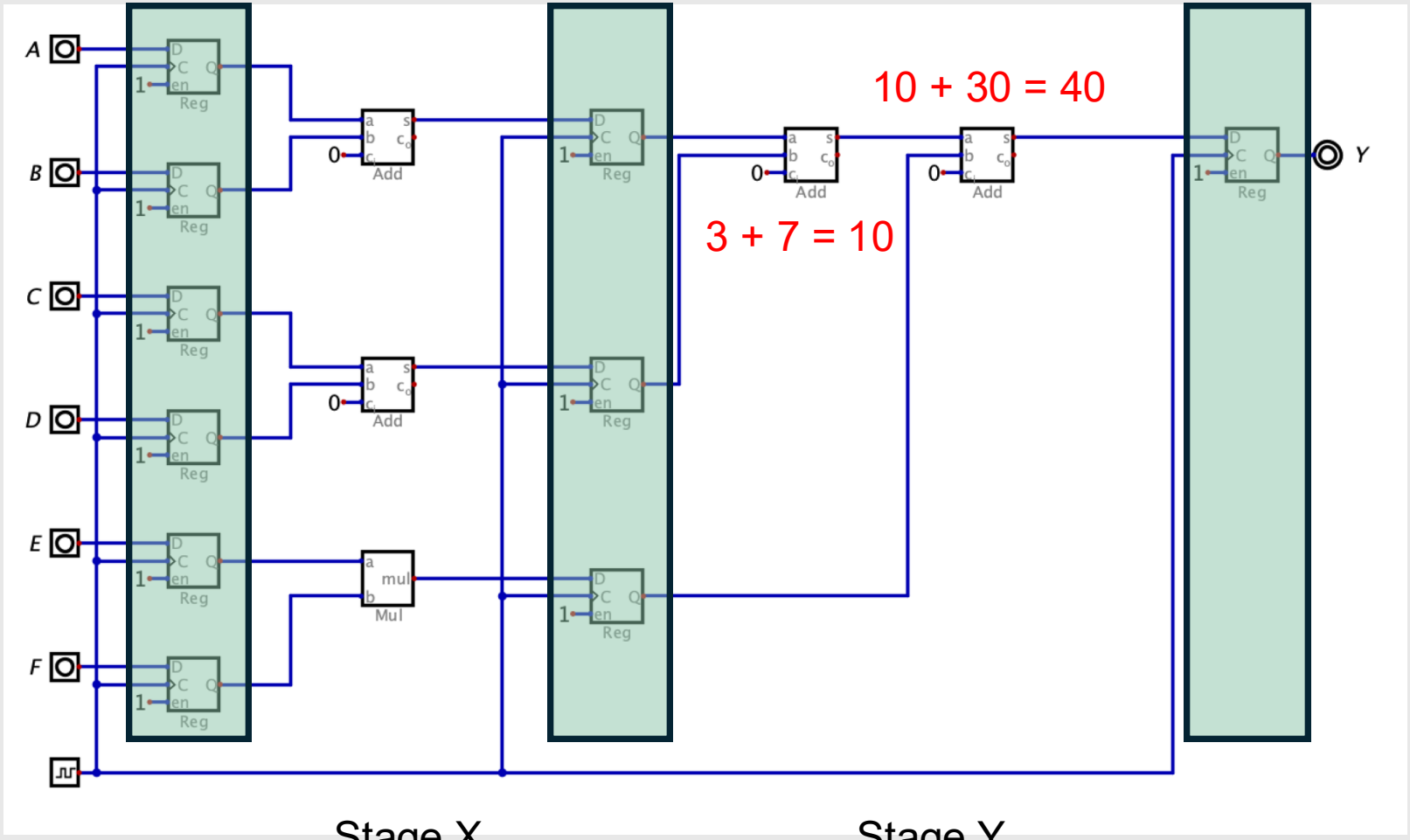


2-Stage Pipeline

Clock-to-q delay = 20ps
Adder propagation delay = 480ps
Multiplier propagation delay = 1200ps

In Cycle 1, Stage X is empty. We can make better use of our resources and start another operation while the first operation is in Stage Y

Let's say we also want to compute
Y = 4 + 2 + 7 + 10 + (9)(8)

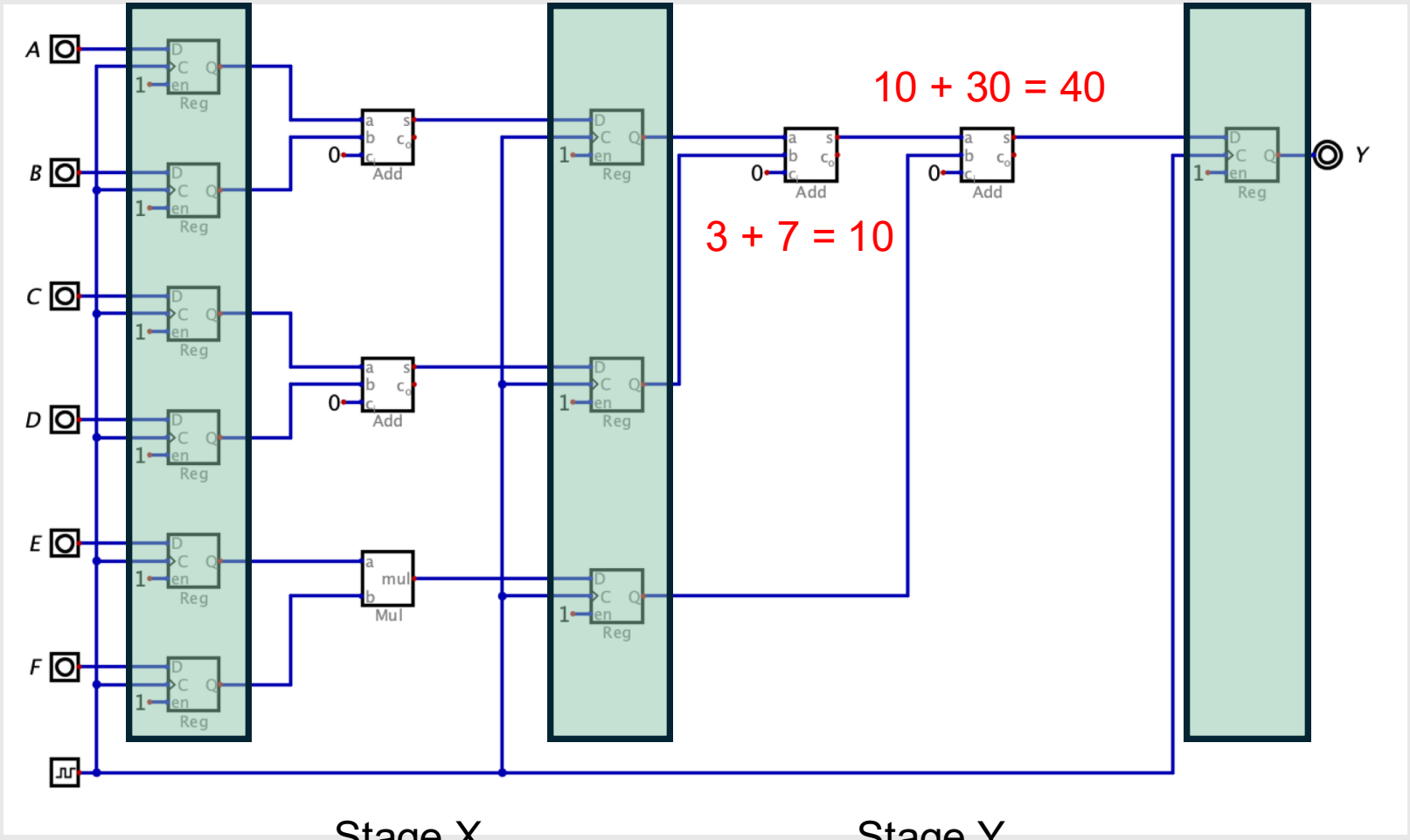


2-Stage Pipeline

Clock-to-q delay = 20ps
Adder propagation delay = 480ps
Multiplier propagation delay = 1200ps

Let's say we also want to perform the following operation:
Y = 4 + 2 + 7 + 10 + (9)(8)

- In Cycle 1, we will compute

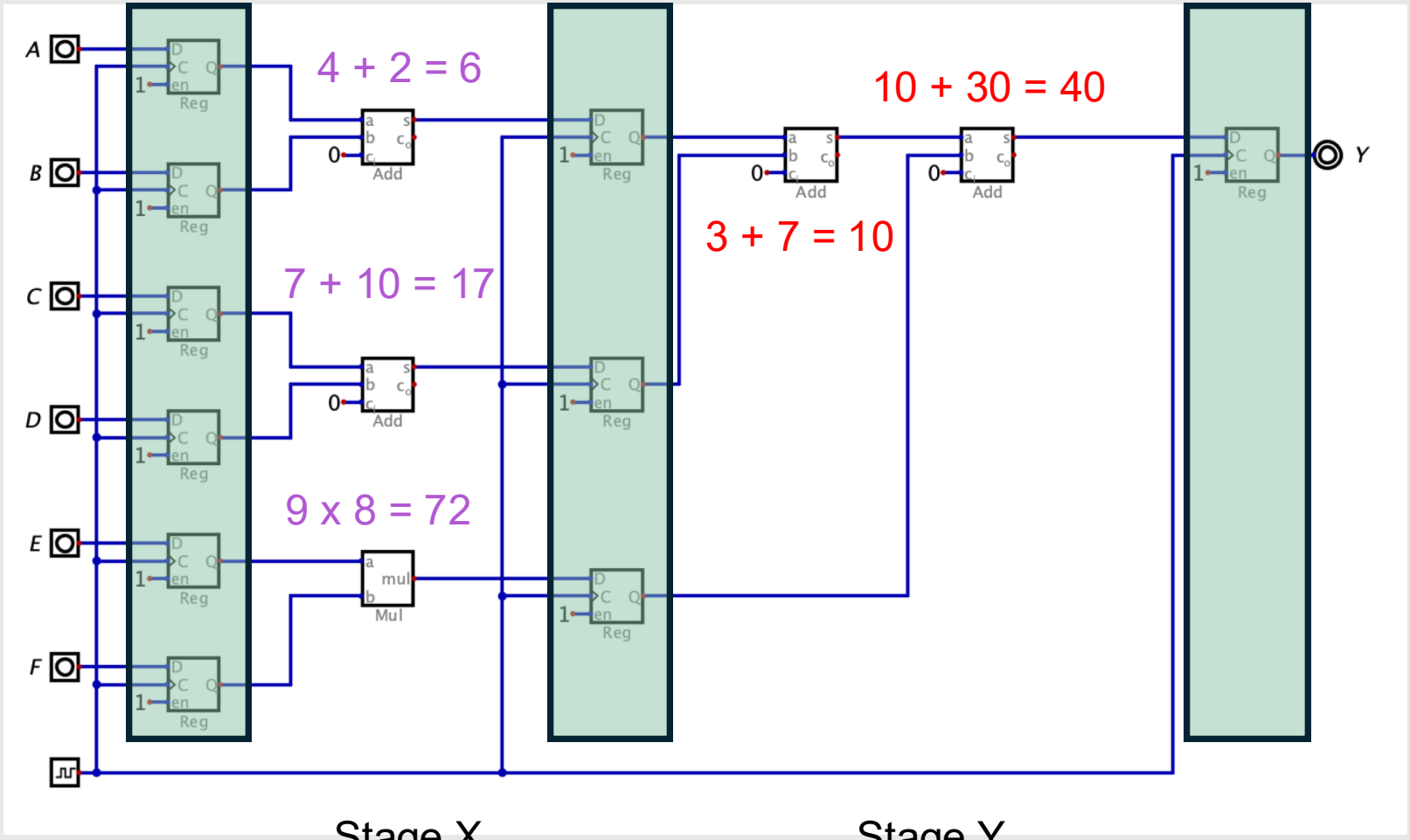


2-Stage Pipeline

Clock-to-q delay = 20ps
Adder propagation delay = 480ps
Multiplier propagation delay = 1200ps

Let's say we also want to perform the following operation:
Y = 4 + 2 + 7 + 10 + (9)(8)

- In Cycle 1, we will compute
  - 4 + 2 = 6
  - 7 + 10 = 17
  - 9 x 8 = 72

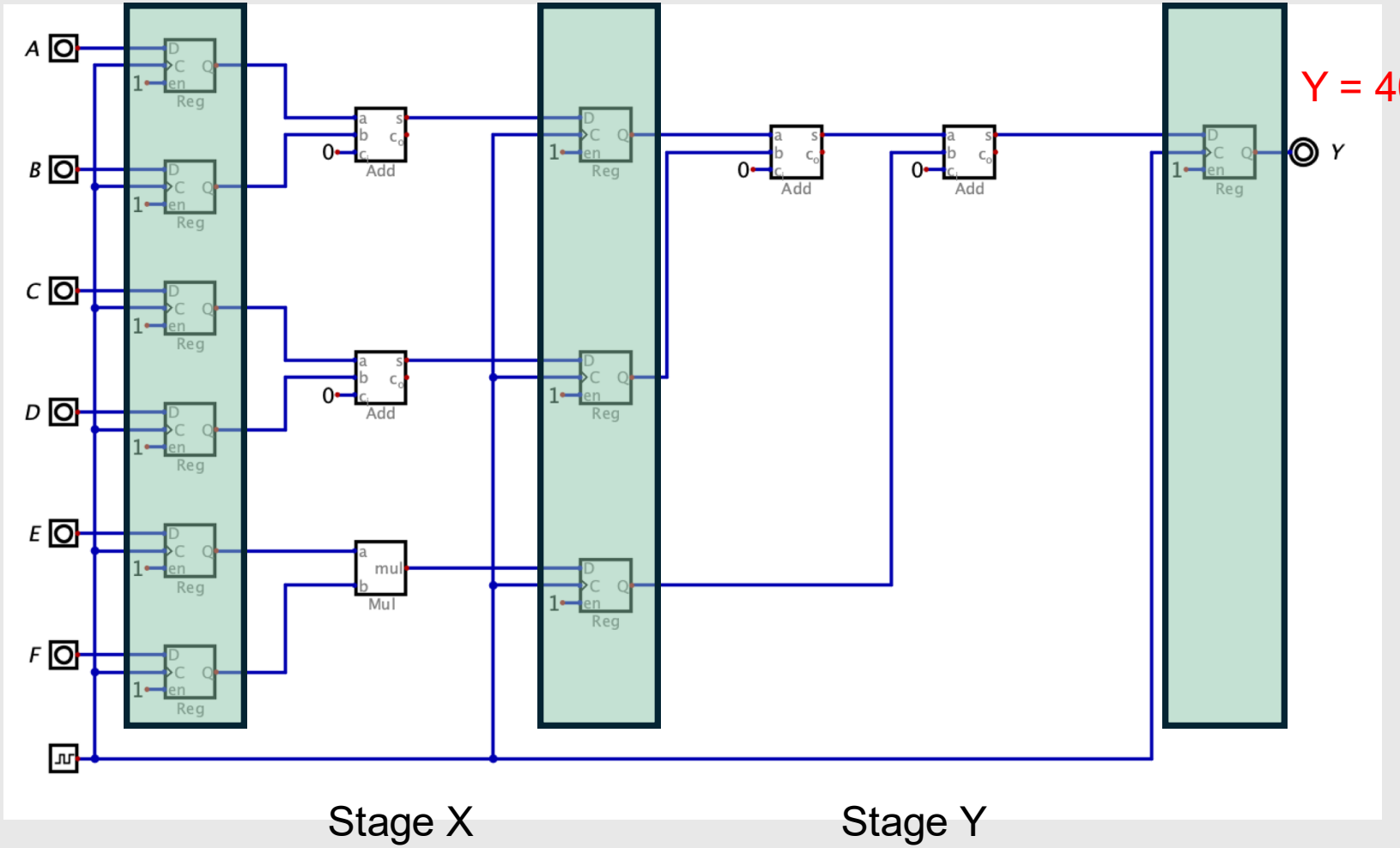


2-Stage Pipeline

Clock-to-q delay = 20ps
Adder propagation delay = 480ps
Multiplier propagation delay = 1200ps

Let's say we also want to perform the following operation:
Y = 4 + 2 + 7 + 10 + (9)(8)

- In Cycle 1, we will compute
  - 4 + 2 = 6
  - 7 + 10 = 17
  - 9 x 8 = 72
- In Cycle 2, we will compute

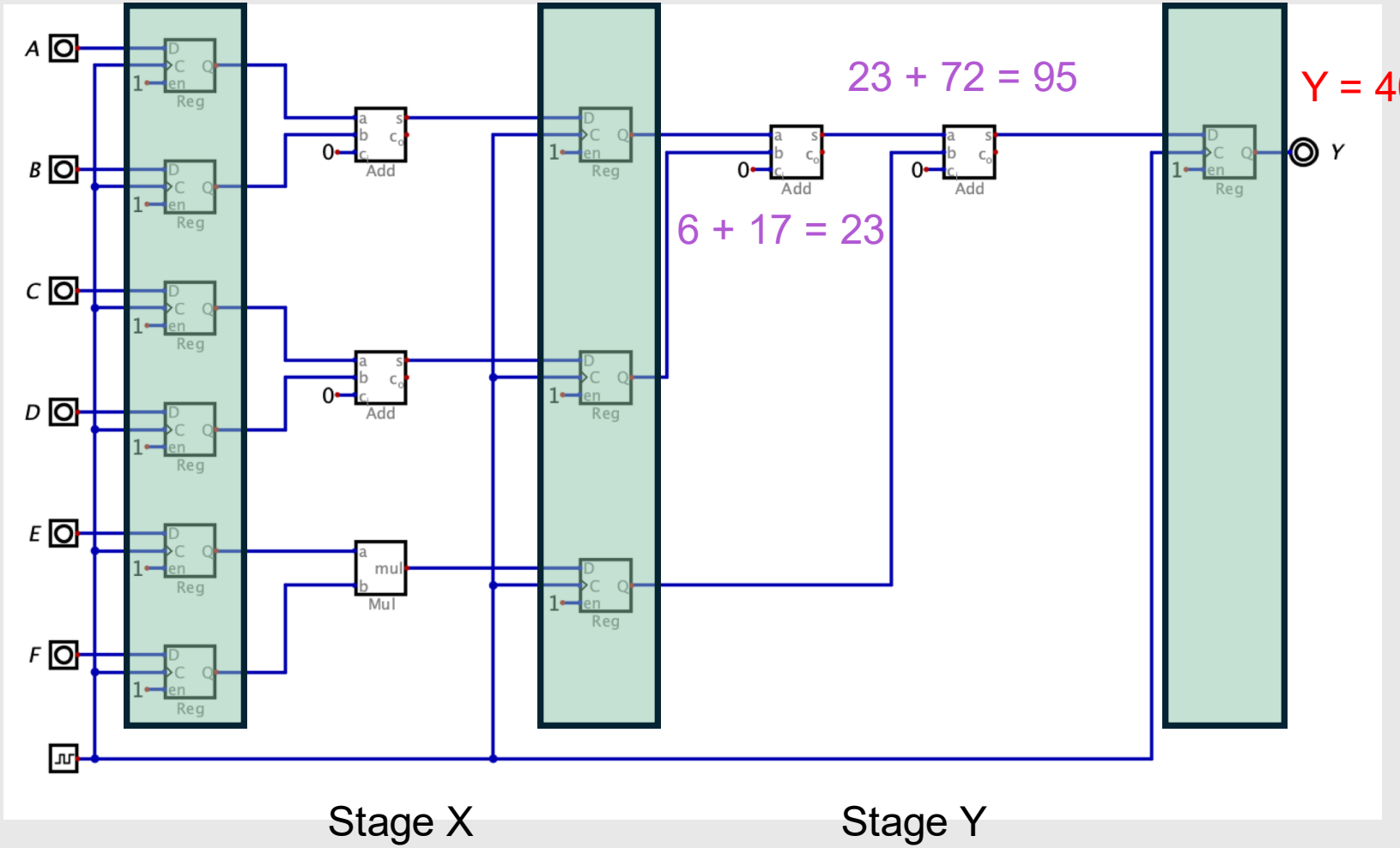


2-Stage Pipeline

Clock-to-q delay = 20ps
Adder propagation delay = 480ps
Multiplier propagation delay = 1200ps

Let's say we also want to perform the following operation:
Y = 4 + 2 + 7 + 10 + (9)(8)

- In Cycle 1, we will compute
  - 4 + 2 = 6
  - 7 + 10 = 17
  - 9 x 8 = 72
- In Cycle 2, we will compute
  - 6 + 17 + 72 = 95

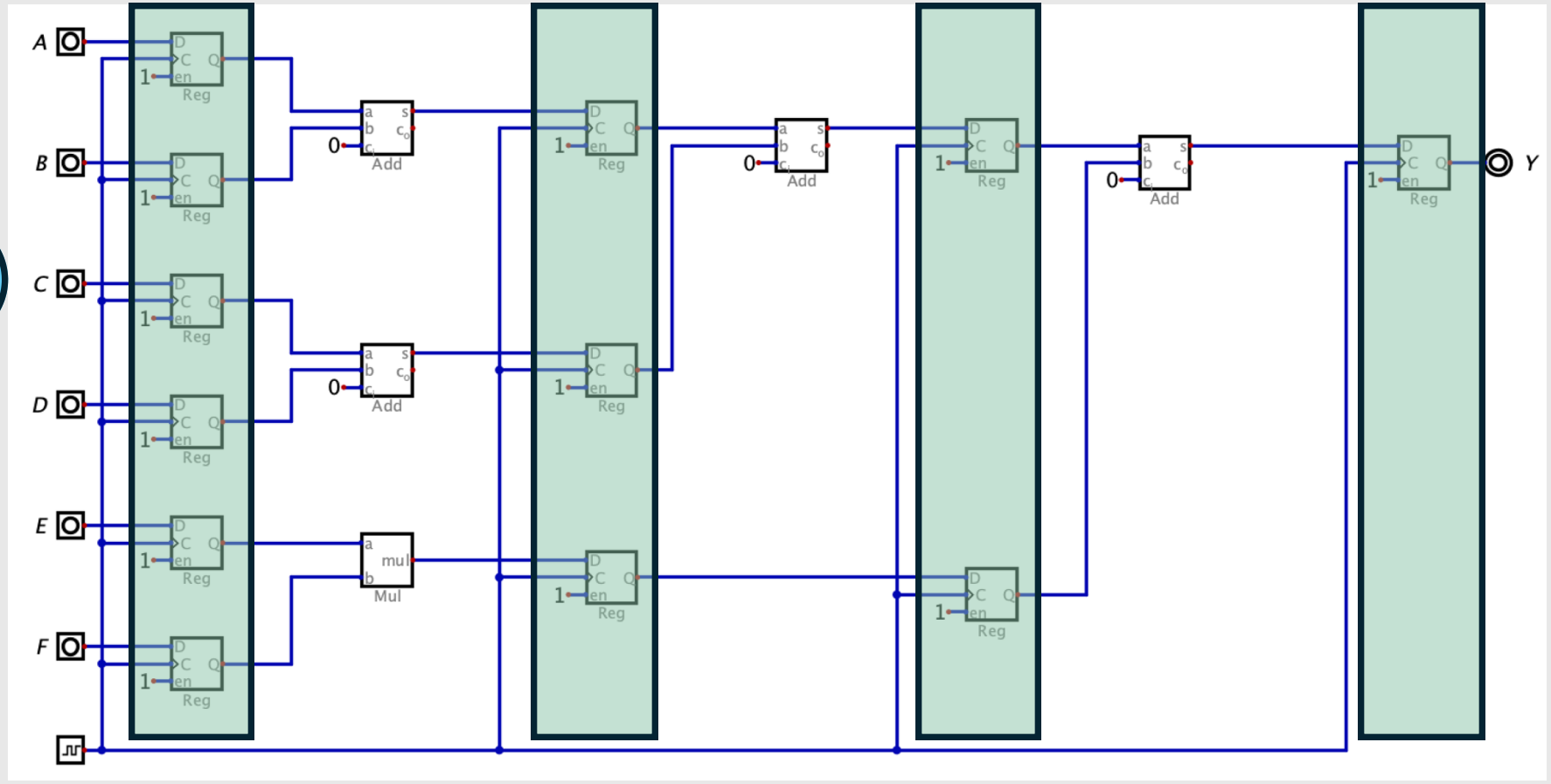


3-Stage Pipeline

Clock-to-q delay = 20ps
Adder propagation delay = 480ps
Multiplier propagation delay = 1200ps

Let's look at the same set of operations in the 3-state pipeline

$Y = 1 + 2 + 3 + 4 + (5)(6)$
$Y = 4 + 2 + 7 + 10 + (9)(8)$

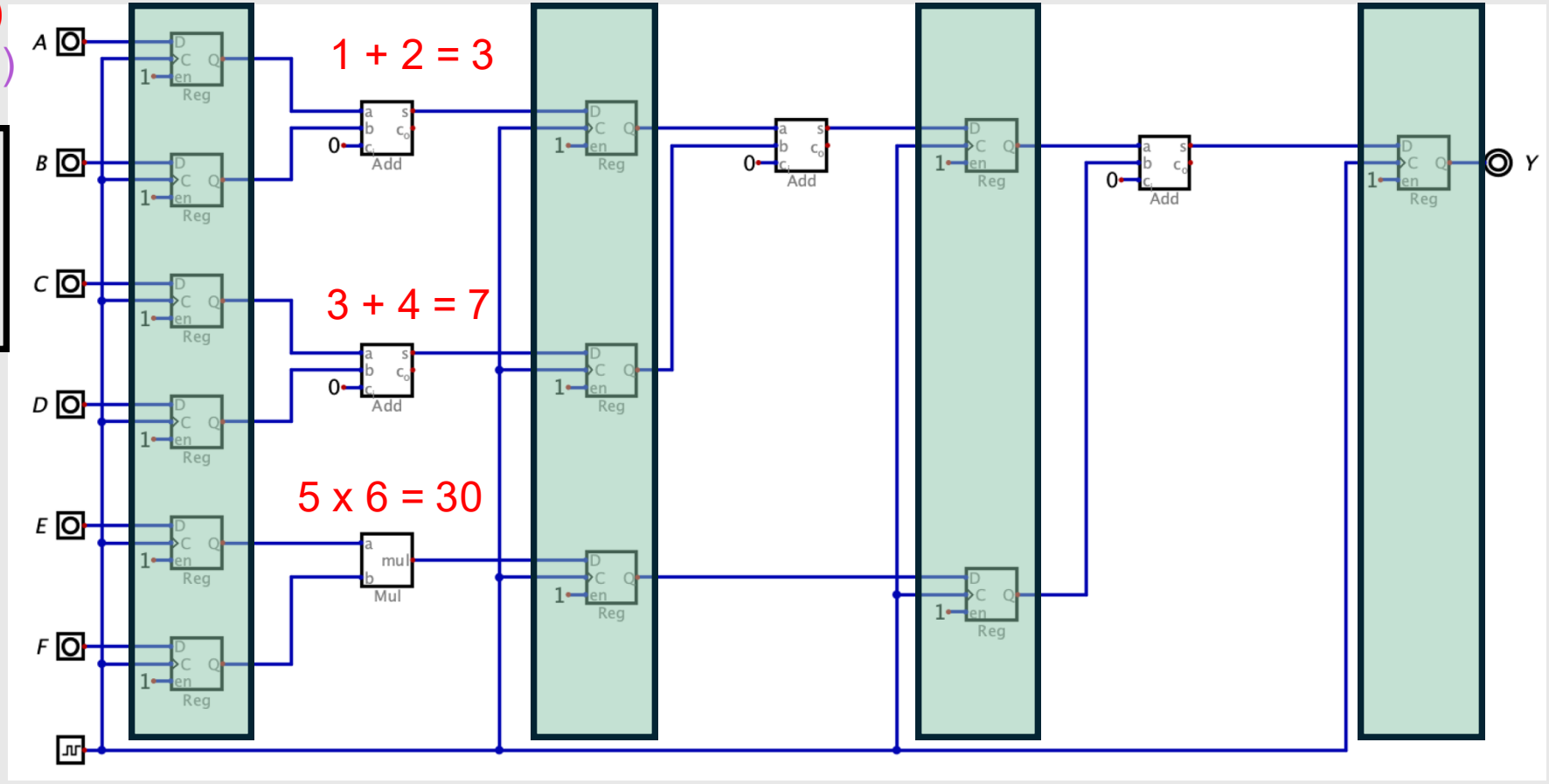


3-Stage Pipeline

Clock-to-q delay = 20ps
Adder propagation delay = 480ps
Multiplier propagation delay = 1200ps

Op0: Y = 1 + 2 + 3 + 4 + (5)(6)
Op1: Y = 4 + 2 + 7 + 10 + (9)(8)

- In Cycle 0, we will compute
  - 1 + 2 = 3
  - 3 + 4 = 7
  - 5 x 6 = 30

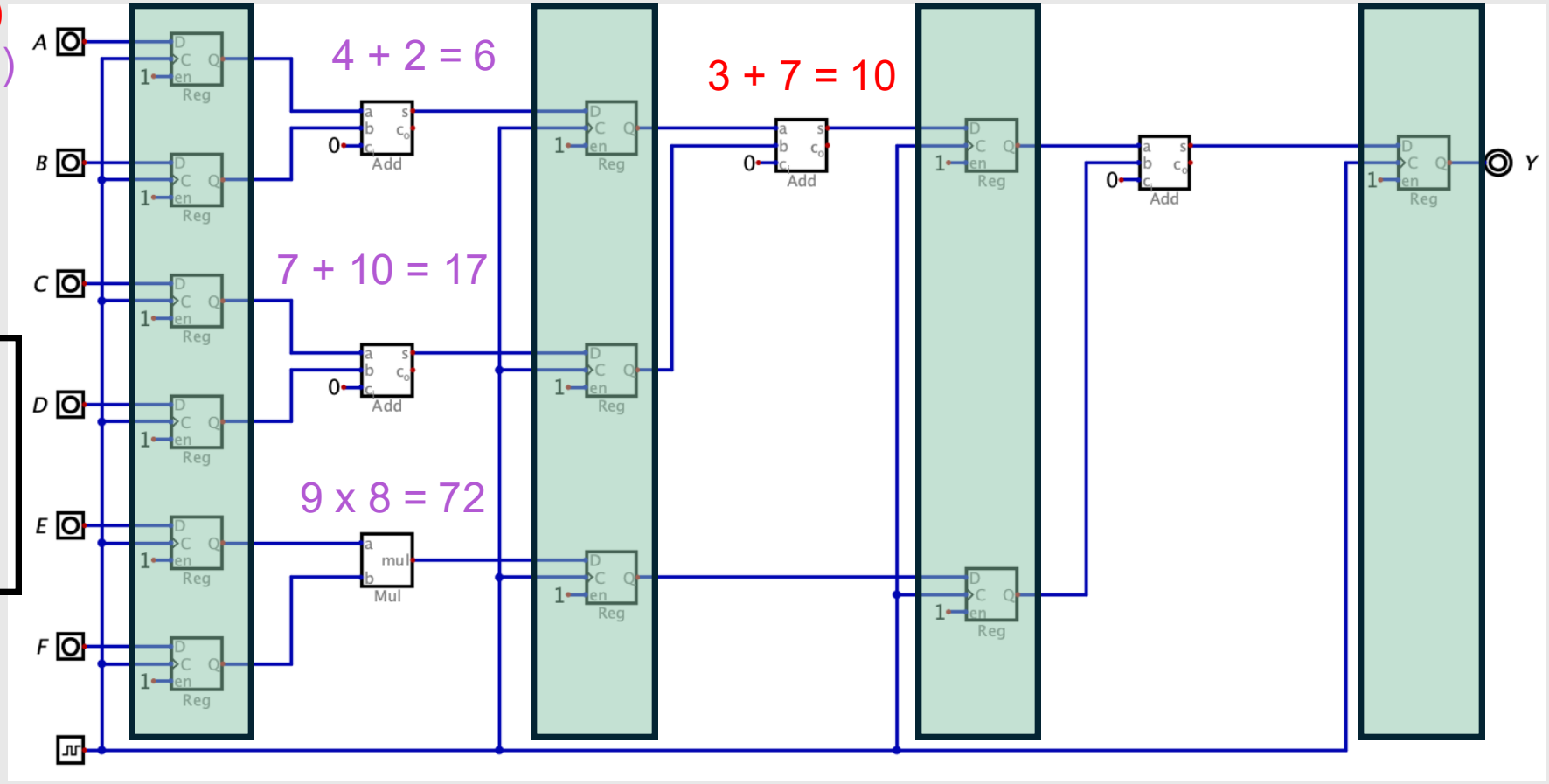


3-Stage Pipeline

Clock-to-q delay = 20ps
Adder propagation delay = 480ps
Multiplier propagation delay = 1200ps

Op0: Y = 1 + 2 + 3 + 4 + (5)(6)
Op1: Y = 4 + 2 + 7 + 10 + (9)(8)

- In Cycle 0, we will compute
  - 1 + 2 = 3
  - 3 + 4 = 7
  - 5 x 6 = 30
- In Cycle 1, we will compute
  - 3 + 7 = 10
  - 4 + 2 = 6
  - 7 + 10 = 17
  - 9 x 8 = 72



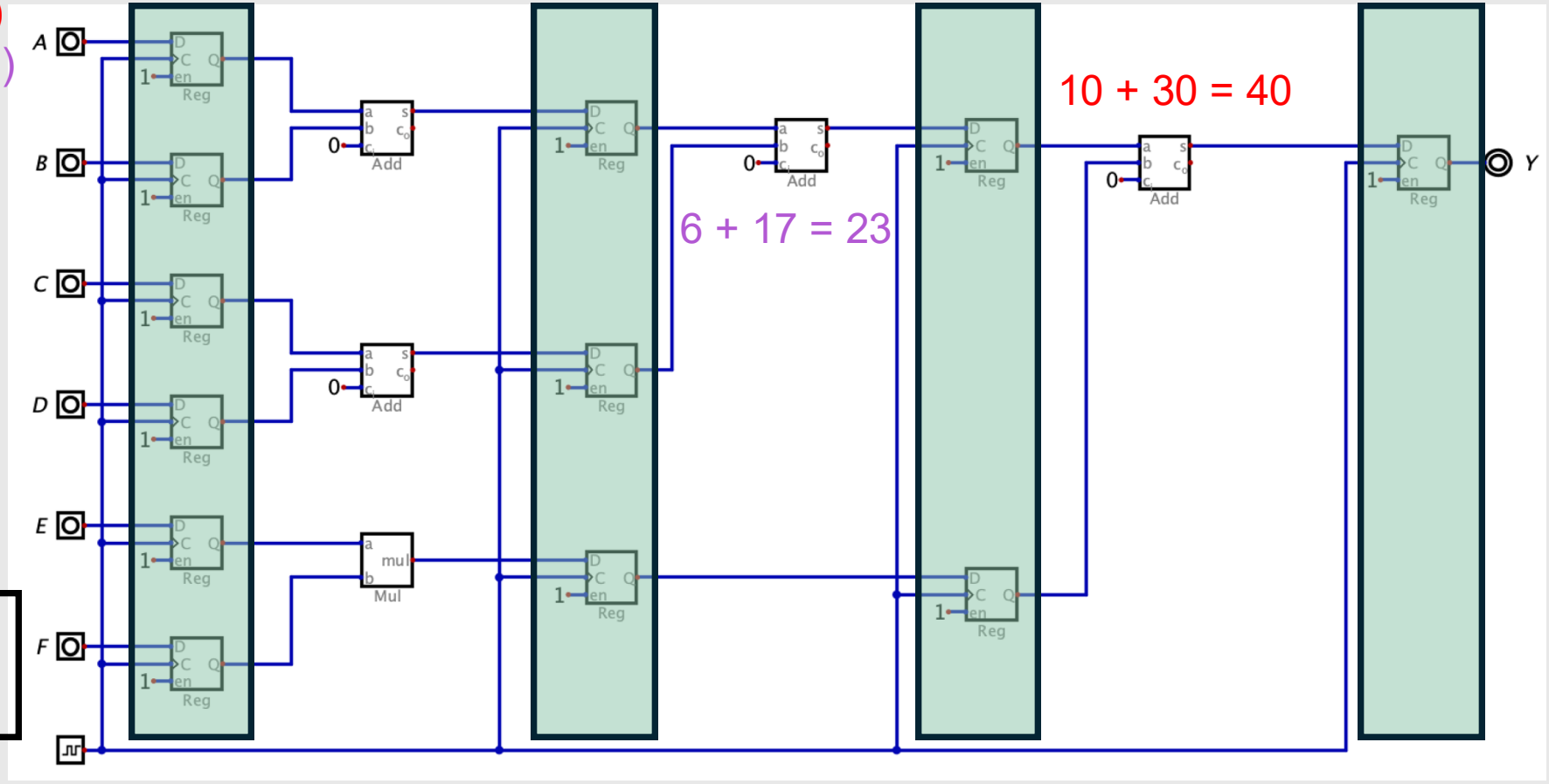
Stage X Stage Y Stage Z

3-Stage Pipeline

Clock-to-q delay = 20ps
Adder propagation delay = 480ps
Multiplier propagation delay = 1200ps

Op0: Y = 1 + 2 + 3 + 4 + (5)(6)
Op1: Y = 4 + 2 + 7 + 10 + (9)(8)

- In Cycle 0, we will compute
  - 1 + 2 = 3
  - 3 + 4 = 7
  - 5 x 6 = 30
- In Cycle 1, we will compute
  - 3 + 7 = 10
  - 4 + 2 = 6
  - 7 + 10 = 17
  - 9 x 8 = 72
- In Cycle 2, we will compute
  - 10 + 30 = 40
  - 6 + 17 = 23



Stage X Stage Y Stage Z

3-Stage Pipeline

Clock-to-q delay = 20ps
Adder propagation delay = 480ps
Multiplier propagation delay = 1200ps

Op0: Y = 1 + 2 + 3 + 4 + (5)(6)
Op1: Y = 4 + 2 + 7 + 10 + (9)(8)

- In Cycle 0, we will compute
  - 1 + 2 = 3
  - 3 + 4 = 7
  - 5 x 6 = 30
- In Cycle 1, we will compute
  - 3 + 7 = 10
  - 4 + 2 = 6
  - 7 + 10 = 17
  - 9 x 8 = 72
- In Cycle 2, we will compute
  - 10 + 30 = 40
  - 6 + 17 = 23
- In Cycle 3, we will compute
  - 23 + 72 = 95