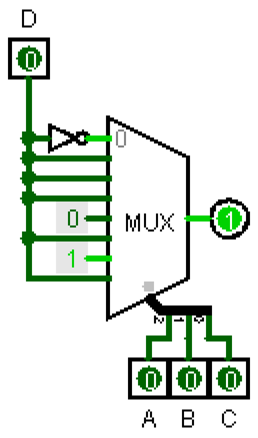
John Gangemi
U6871-4612
CDA 3201

Homework Assignment 3

1. Given a four input Boolean function F(A,B,C,D) = Σm(0,3,5,7,11,12,13,15)

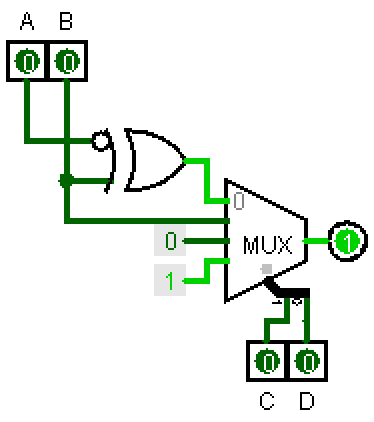
a.

Control Inputs			Data Input	Output	
A	B	C	D	Q	
0	0	0	0	1	D'
0	0	0	1	0	
0	0	1	0	0	D
0	0	1	1	1	
0	1	0	0	0	D
0	1	0	1	1	
0	1	1	0	0	D
0	1	1	1	1	
1	0	0	0	0	0
1	0	0	1	0	
1	0	1	0	0	D
1	0	1	1	1	
1	1	0	0	1	1
1	1	0	1	1	
1	1	1	0	0	D
1	1	1	1	1	



b.

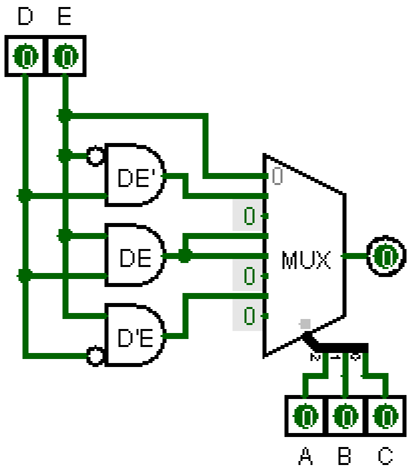
Control Inputs		Data Inputs		Output	
C	D	A	B	Q	
0	0	0	0	1	
0	0	0	1	0	A' XOR B
0	0	1	0	0	
0	0	1	1	1	
0	1	0	0	0	
0	1	0	1	1	B
0	1	1	0	0	
0	1	1	1	1	
1	0	0	0	0	
1	0	0	1	0	0
1	0	1	0	0	
1	0	1	1	0	
1	1	0	0	1	
1	1	0	1	1	1
1	1	1	0	1	
1	1	1	1	1	



2. Implement the function using an 8:1 multiplexer

a. F(a,b,c,d,e) = m1 + m3 + m6 + m15 + m19 + m25

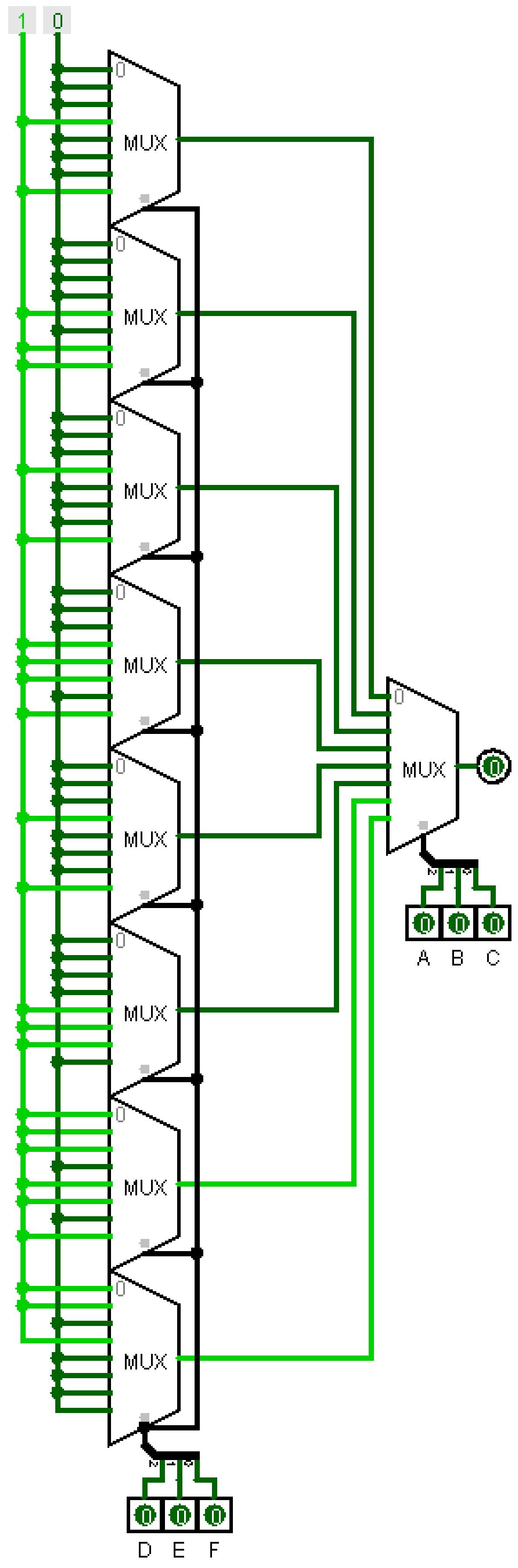
Control Inputs			Data Inputs		Output	
A	B	C	D	E	Q	
0	0	0	0	0	0	
0	0	0	0	1	1	E
0	0	0	1	0	0	
0	0	0	1	1	1	
0	0	1	0	0	0	
0	0	1	0	1	0	DE'
0	0	1	1	0	1	
0	0	1	1	1	0	
0	1	0	0	0	0	
0	1	0	0	1	0	0
0	1	0	1	0	0	
0	1	0	1	1	0	
0	1	1	0	0	0	
0	1	1	0	1	0	DE
0	1	1	1	0	0	
0	1	1	1	1	1	
1	0	0	0	0	0	
1	0	0	0	1	0	DE
1	0	0	1	0	0	
1	0	0	1	1	1	
1	0	1	0	0	0	
1	0	1	0	1	0	0
1	0	1	1	0	0	
1	0	1	1	1	0	
1	1	0	0	0	0	
1	1	0	0	1	1	D'E
1	1	0	1	0	0	
1	1	0	1	1	0	
1	1	1	0	0	0	
1	1	1	0	1	0	0
1	1	1	1	0	0	
1	1	1	1	1	0	


D
E
DE'
DE
D'E
MUX
0
0
0
0
A
B
C

3. Exercise 4.12

Control Lines Second Stage			Control Lines First Stage			Output
A	B	C	D	E	F	Q
0	0	0	0	0	0	0
			0	0	1	0
			0	1	0	0
			0	1	1	1
			1	0	0	0
			1	0	1	0
			1	1	0	0
			1	1	1	1
0	0	1	0	0	0	0
			0	0	1	0
			0	1	0	0
			0	1	1	0
			1	0	0	1
			1	0	1	0
			1	1	0	1
			1	1	1	1
0	1	0	0	0	0	0
			0	0	1	0
			0	1	0	0
			0	1	1	1
			1	0	0	0
			1	0	1	0
			1	1	0	0
			1	1	1	1
0	1	1	0	0	0	0
			0	0	1	0
			0	1	0	0
			0	1	1	1
			1	0	0	1
			1	0	1	1
			1	1	0	0
			1	1	1	1
1	0	0	0	0	0	0
			0	0	1	0
			0	1	0	0
			0	1	1	1
			1	0	0	0
			1	0	1	0
			1	1	0	0
			1	1	1	1

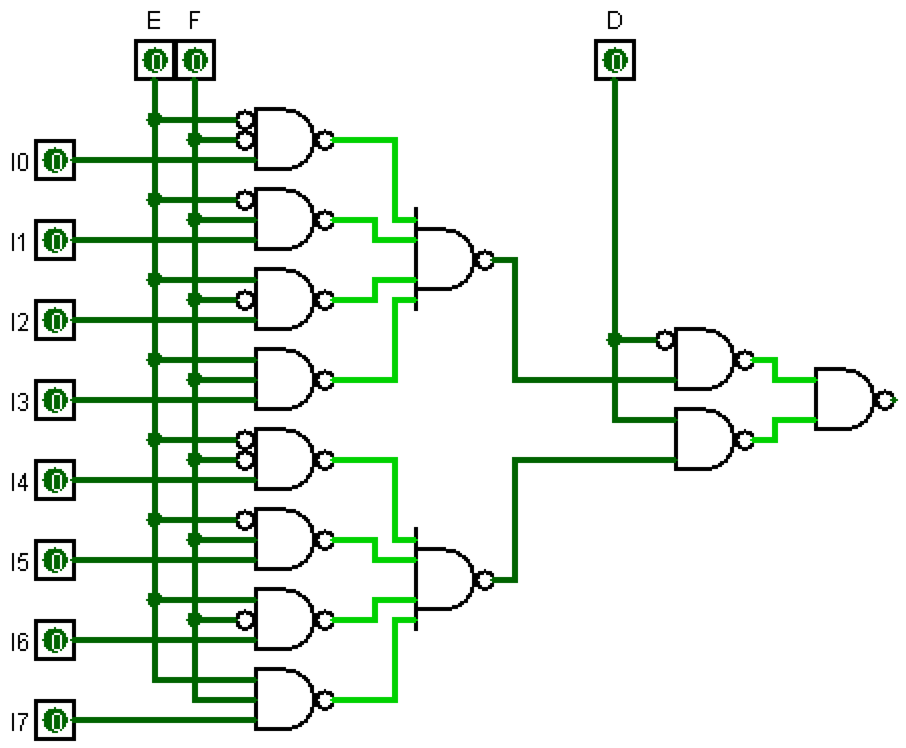
1	0	1	0	0	0	0
			0	0	1	0
			0	1	0	0
			0	1	1	0
			1	0	0	1
			1	0	1	1
			1	1	0	1
			1	1	1	0
1	1	0	0	0	0	1
			0	0	1	1
			0	1	0	1
			0	1	1	0
			1	0	0	1
			1	0	1	1
			1	1	0	0
			1	1	1	1
1	1	1	0	0	0	1
			0	0	1	1
			0	1	0	0
			0	1	1	1
			1	0	0	0
			1	0	1	0
			1	1	0	0
			1	1	1	0



1
0
MUX
MUX
MUX
MUX
MUX
MUX
MUX
MUX
MUX
A
B
C
D
E
F

a. Assuming there is one 8:1 multiplexer or two 4:1 multiplexers per logic package, then the total number of packages required for a two-stage multiplexer network using only 8:1 multiplexers is **9** logic packages.

b. Assuming there are four 2-input NAND gates, three 3-input NAND gates, or two 4-input gates per package then a single 8:1 multiplexer can be modeled using only inverters and NAND gates as shown below. The configuration is duplicated *8* times to create the "first level" multiplexer network.

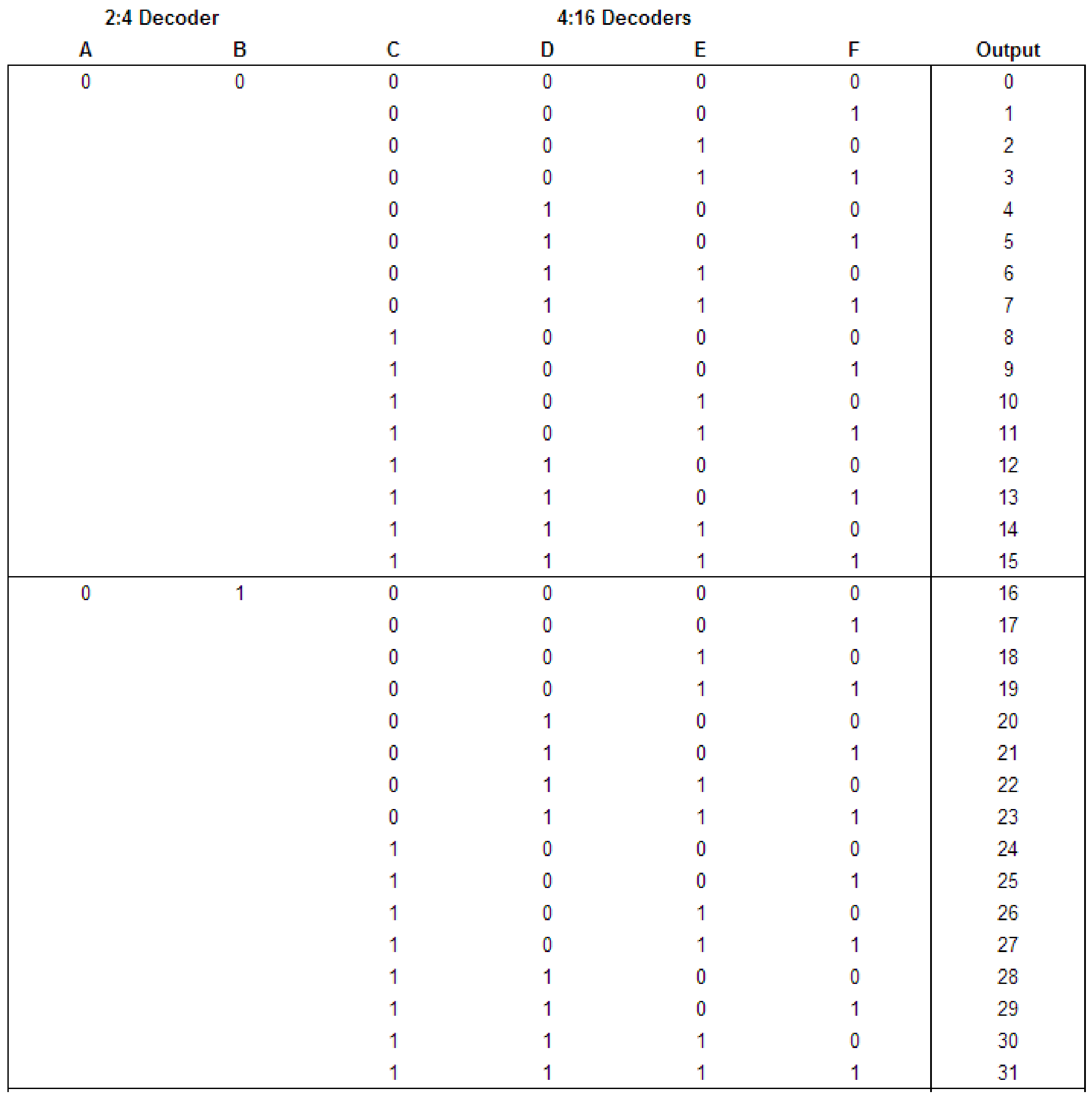


Total number of packages for "first level" multiplexer network (8 x 8:1 MUX):
30 3-input NAND
12 Inverters
8 4-input NAND
6 2-input NAND

The "second level" multiplexer network is a single 8:1 multiplexer NAND gate implementation, thus the total number of packages for "second level" multiplexer network:
3 ⅔ 3-input NAND
1 ½ Inverters
1 4-input NAND
¾ 2-input NAND

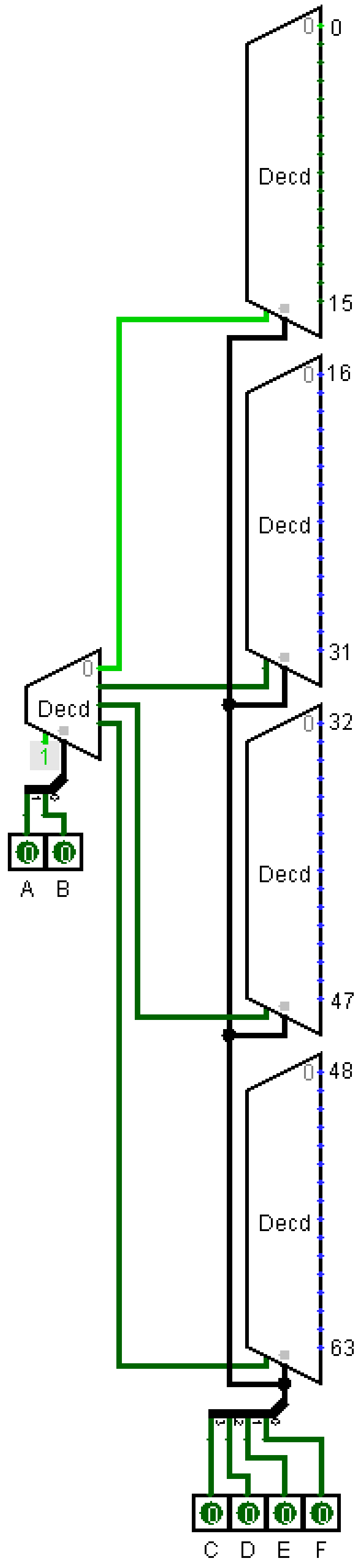
Overall number of packages for the entire 6 variable two-stage multiplexer network:
34 3-input NAND
14 Inverters
9 4-input NAND
7 2-input NAND
= approximately 64 TTL packages

4. Exercise 4.15



2:4 Decoder		4:16 Decoders				
A	B	C	D	E	F	Output
0	0	0	0	0	0	0
		0	0	0	1	1
		0	0	1	0	2
		0	0	1	1	3
		0	1	0	0	4
		0	1	0	1	5
		0	1	1	0	6
		0	1	1	1	7
		1	0	0	0	8
		1	0	0	1	9
		1	0	1	0	10
		1	0	1	1	11
		1	1	0	0	12
		1	1	0	1	13
		1	1	1	0	14
		1	1	1	1	15
0	1	0	0	0	0	16
		0	0	0	1	17
		0	0	1	0	18
		0	0	1	1	19
		0	1	0	0	20
		0	1	0	1	21
		0	1	1	0	22
		0	1	1	1	23
		1	0	0	0	24
		1	0	0	1	25
		1	0	1	0	26
		1	0	1	1	27
		1	1	0	0	28
		1	1	0	1	29
		1	1	1	0	30
		1	1	1	1	31

1	0	0	0	0	0	32
		0	0	0	1	33
		0	0	1	0	34
		0	0	1	1	35
		0	1	0	0	36
		0	1	0	1	37
		0	1	1	0	38
		0	1	1	1	39
		1	0	0	0	40
		1	0	0	1	41
		1	0	1	0	42
		1	0	1	1	43
		1	1	0	0	44
		1	1	0	1	45
		1	1	1	0	46
		1	1	1	1	47
1	1	0	0	0	0	48
		0	0	0	1	49
		0	0	1	0	50
		0	0	1	1	51
		0	1	0	0	52
		0	1	0	1	53
		0	1	1	0	54
		0	1	1	1	55
		1	0	0	0	56
		1	0	0	1	57
		1	0	1	0	58
		1	0	1	1	59
		1	1	0	0	60
		1	1	0	1	61
		1	1	1	0	62
		1	1	1	1	63



0
Decd
15
16
Decd
31
32
Decd
47
48
Decd
63
Decd
1
A
B
C
D
E
F

5. Exercise 4.14

c. f(P,Q,R,S,T) = P'Q'R'(S + T)

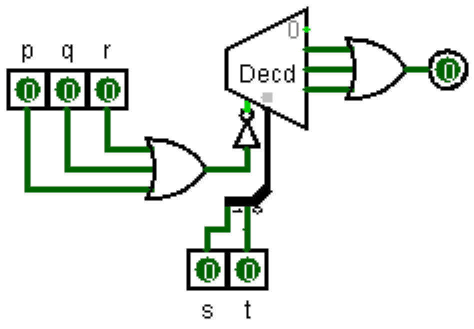
= P'Q'R'S + P'Q'R'T

= P'Q'R'S(T) + P'Q'R'S(T') + P'Q'R'(S)T + P'Q'R'(S')T

= P'Q'R'ST + P'Q'R'ST' + P'Q'R'S'T

Given the requirement for a 2:4 decoder only two bits can be used to select the appropriate output, noticing that P'Q'R' is constant where S & T is variable then S & T can be used as the select bits. Allowing the constant P'Q'R' to control the enable input ensures the decoder is operational only in a specific manner.

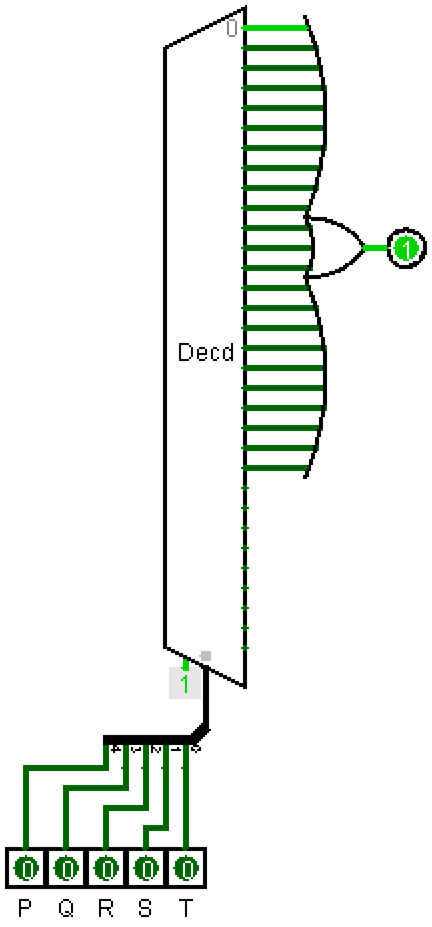
Note: It is assumed that the enable bit for the decoder is active low, given the requirements of using only OR gates along with decoders. In the logic diagram provided the enable bit is active high and thus had to be negated for proper functionality in Logisim, therefore disregard the Inverter in the logic diagram.



d. f(P,Q,R,S,T) = (P(Q + RST))’

P	Q	R	S	T	RST	Q + RST	P(Q + RST)	f = P(Q + RST)'
0	0	0	0	0	0	0	0	1
0	0	0	0	1	0	0	0	1
0	0	0	1	0	0	0	0	1
0	0	0	1	1	0	0	0	1
0	0	1	0	0	0	0	0	1
0	0	1	0	1	0	0	0	1
0	0	1	1	0	0	0	0	1
0	0	1	1	1	1	1	0	1
0	1	0	0	0	0	1	0	1
0	1	0	0	1	0	1	0	1
0	1	0	1	0	0	1	0	1
0	1	0	1	1	0	1	0	1
0	1	1	0	0	0	1	0	1
0	1	1	0	1	0	1	0	1
0	1	1	1	0	0	1	0	1
0	1	1	1	1	1	1	0	1
1	0	0	0	0	0	0	0	1
1	0	0	0	1	0	0	0	1
1	0	0	1	0	0	0	0	1
1	0	0	1	1	0	0	0	1
1	0	1	0	0	0	0	0	1
1	0	1	0	1	0	0	0	1
1	0	1	1	0	0	0	0	1
1	0	1	1	1	1	1	1	0
1	1	0	0	0	0	1	1	0
1	1	0	0	1	0	1	1	0
1	1	0	1	0	0	1	1	0
1	1	0	1	1	0	1	1	0
1	1	1	0	0	0	1	1	0
1	1	1	0	1	0	1	1	0
1	1	1	1	0	0	1	1	0
1	1	1	1	1	1	1	1	0

MINTERMS = 0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22



6. Given the sequential logic circuit in Figure 6.19
a. Assuming no propagation delay in the combinational logic block, find the maximum allowable frequency of the clock that controls the sub system.

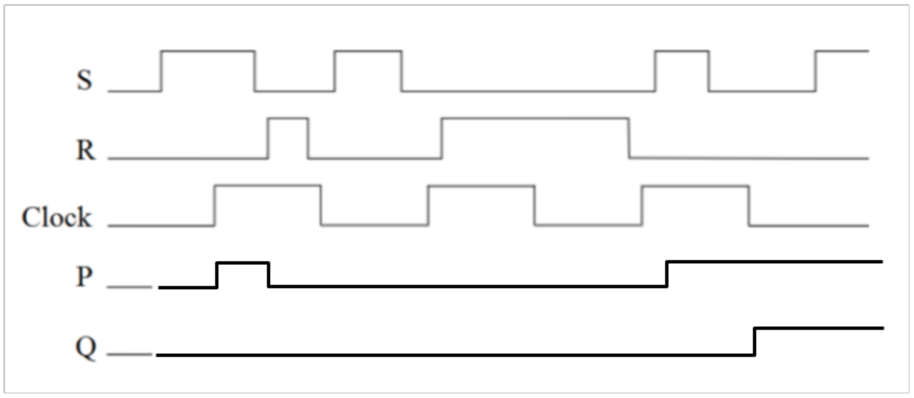
- Set-up time = 25ns & Propagation delay = 15ns
- Minimum clock period (Tmin clk) = Tsu + Tpd = 25ns + 15ns = 40ns
- Maximum frequency (Fmax) = 1/Tmin clk = 1/40ns = 1/40e-9 sec = *25 MHz*

b. Assuming a worst-case combinational logic delay of 110ns, how does the answer from part (a) change?

- Set-up time = 25ns & Propagation Delay = 15ns & Worst-case combinational logic = 110ns
- Minimum clock period (Tmin clk) = Tsu + Tpd + Tcpd = 25ns + 15ns + 110ns = 150ns
- Maximum frequency (Fmax) = 1/Tmin clk = 1/150ns = 1/150e-9 sec = *6.67 MHz*

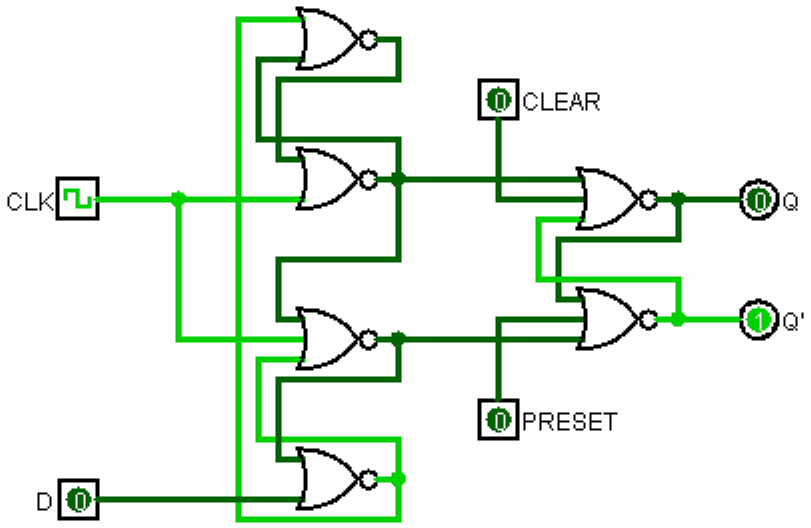
The minimum clock period increased and as a result the maximum frequency decreased compared to part (a).

7. Indicate the outputs of a negative edge triggered RS Master-Slave flip-flop.



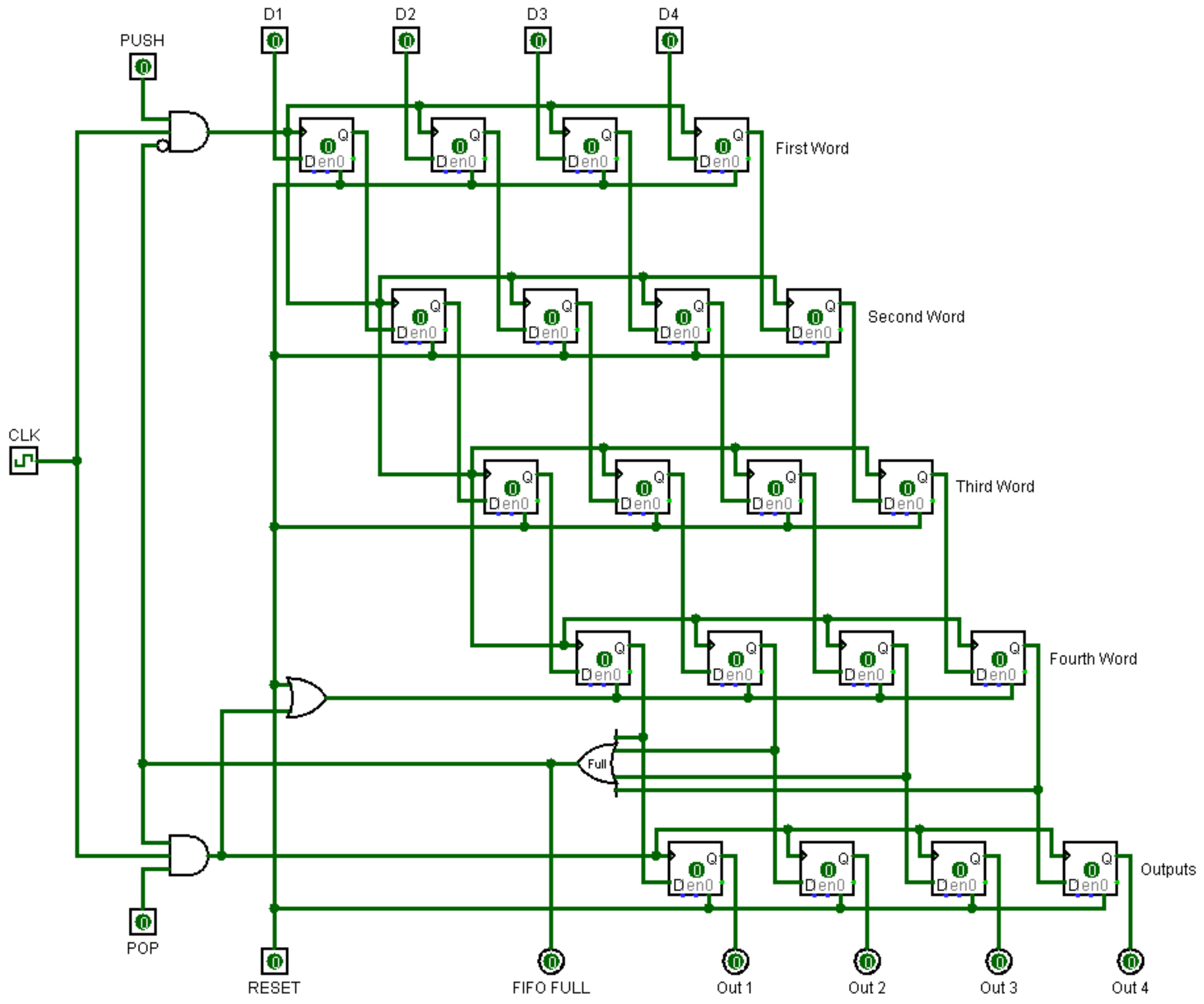
8. Exercise 6.4

Adding an asynchronous preset and clear inputs to the edge-triggered D flip-flop requires the addition of a third input to the R and S NOR gates, as these gates are not dependent upon the presence of clock similar to the asynchronous behavior seen in a regular SR latch.



9. Exercise 6.26

Design of a “flow-through” FIFO queue that stores 4 words by 4-bits using combinational logic and shift-register components.



In this design the FIFO consists of four main register blocks that support 4 bit words. Data is loaded in parallel synchronously from inputs D1,D2,D3, and D4 only when the FIFO is not full and CLK/PUSH are high. For every clock cycle 4-bit data is shifted to the next lower set of registers in the main register block.

To determine if the FIFO is full an OR gate was added between the last stage registers and the output registers. If either of the four registers holds a “1” then the last stage has been populated and the FIFO FULL indicator displays a “1”.

The inclusion of a fifth register block is meant to hold the 4-bit value most recently removed from the FIFO.

The “push” combinational logic is only true for specific cases when PUSH/CLK/FULL’ are true, also the “pop” combinational logic is only true for specific cases when POP/CLK/FULL are true.

When popping the 4-bit value in the last set of registers, the CLEAR asynchronous input is triggered simultaneously effectively storing the 4-bit value in the output registers and clearing the last stage registers of the main block and consequently setting FULL to “0”.

An active high reset input clears all data values stored in every register asynchronously.