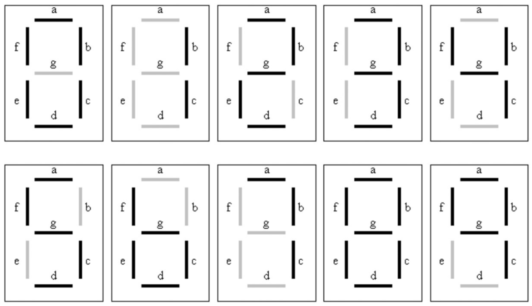
HOMEWORK #3

EE/CS 119a

Adam Krivka

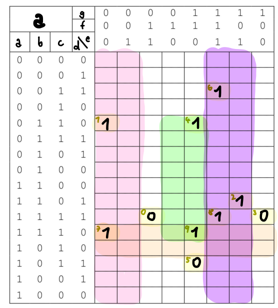
1



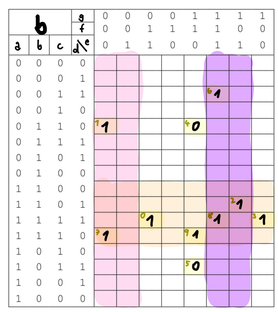
	input							output						
	a	b	c	d	e	f	g	a	b	c	d	e	f	g
0	1	1	1	1	1	1	0	0	1	1	0	0	0	0
1	0	1	1	0	0	0	0	1	1	0	1	1	0	1
2	1	1	0	1	1	0	1	1	1	1	1	0	0	1
3	1	1	1	1	0	0	1	0	1	1	0	0	1	1
4	0	1	1	0	0	1	1	1	0	1	1	0	1	1
5	1	0	1	1	0	1	1	0	0	1	1	1	1	1
6	0	0	1	1	1	1	1	1	1	1	0	0	0	0
7	1	1	1	0	0	0	0	1	1	1	1	1	1	1
8	1	1	1	1	1	1	1	1	1	1	0	0	1	1
9	1	1	1	0	0	1	1	1	1	1	1	1	1	0

Form:

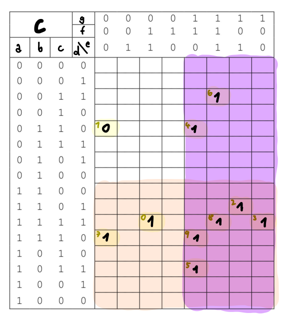
			g	0	0	0	0	1	1	1	1
			f	0	0	1	1	1	1	0	0
a	b	c	d \ e	0	1	1	0	0	1	1	0
0	0	0	0								
0	0	0	1								
0	0	1	1						6		
0	0	1	0								
0	1	1	0	1				4			
0	1	1	1								
0	1	0	1								
0	1	0	0								
1	1	0	0								
1	1	0	1							2	
1	1	1	1			0			8		3
1	1	1	0	7				9			
1	0	1	0								
1	0	1	1					5			
1	0	0	1								
1	0	0	0								



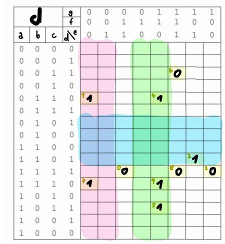
$a = \bar{g}\bar{f} + ac\bar{d} + bf\bar{e} + ge$



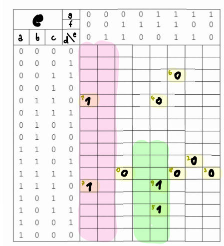
$b = \bar{g}\bar{f} + ab + ge$



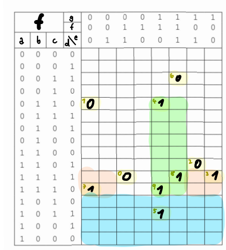
$c = g + a$



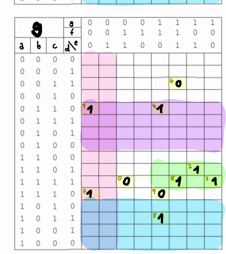
$d = \bar{g}\bar{f} + f\bar{e} + b\bar{c}$



$e = \bar{g}\bar{f} + af\bar{e}$



$f = abc\bar{f} + bgf + a\bar{b}$



$g = \bar{g}\bar{f} + a\bar{b} + abdg + \bar{a}b$

Final equations:

$$a = \bar{g}\bar{f} + ac\bar{d} + bf\bar{e} + ge$$

$$b = \bar{g}\bar{f} + ab + ge$$

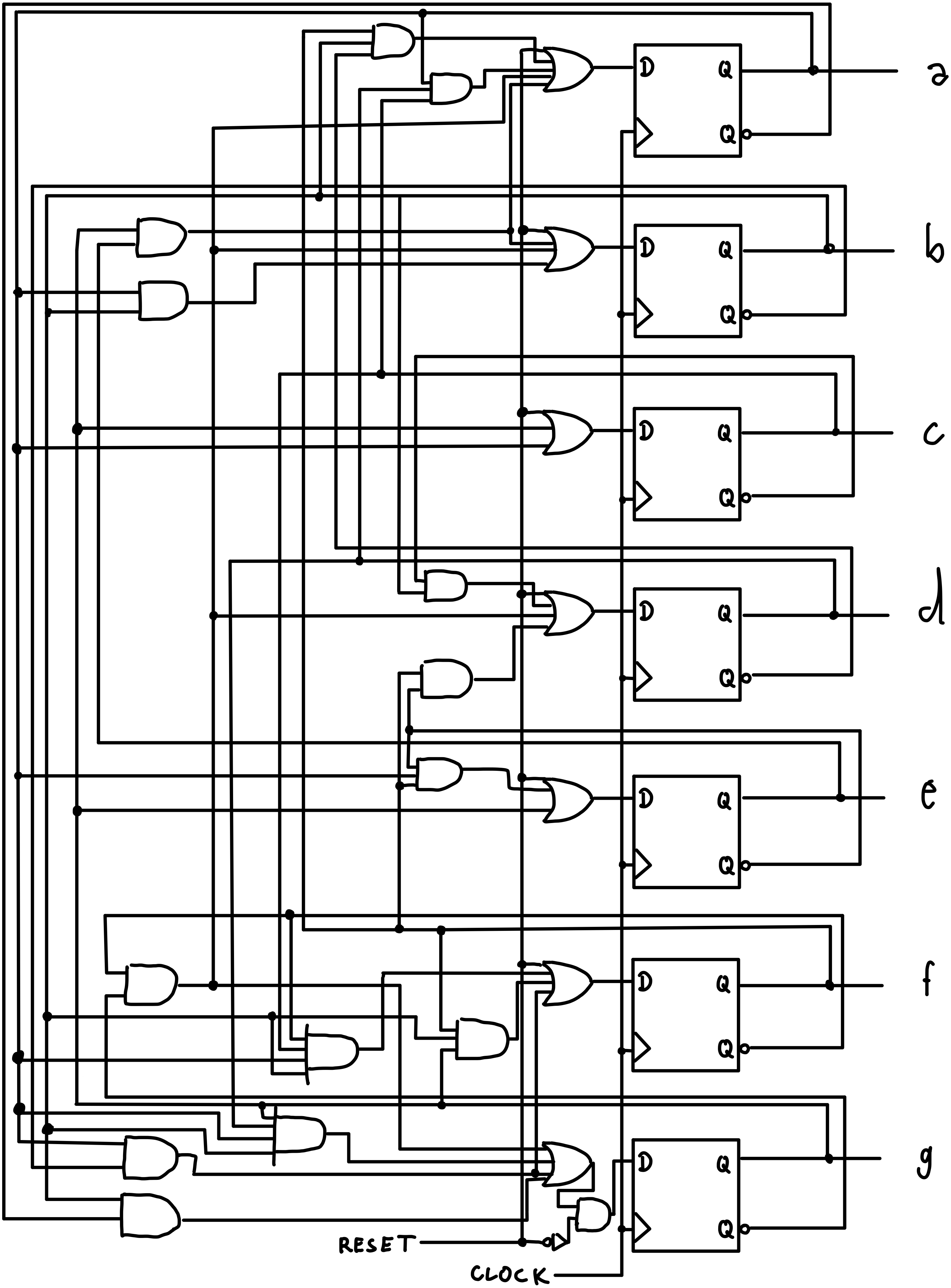
$$c = g + a$$

$$d = \bar{g}\bar{f} + f\bar{e} + b\bar{c}$$

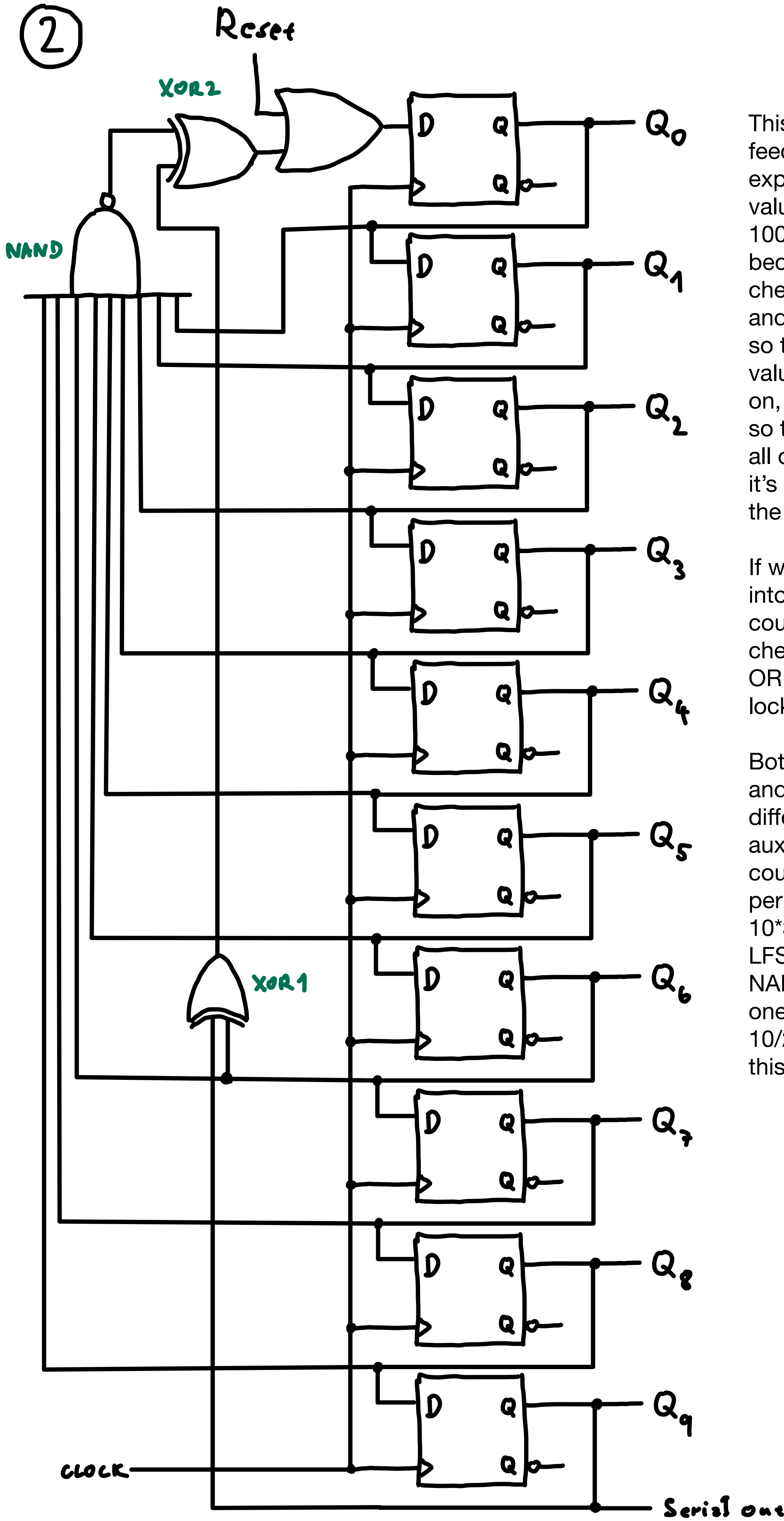
$$e = \bar{g}\bar{f} + af\bar{e}$$

$$f = abc\bar{f} + bgf + a\bar{b}$$

$$g = \bar{g}\bar{f} + a\bar{b} + abdg + \bar{a}b$$


a
b
c
d
e
f
g
D
Q
Q
RESET
CLOCK

2



This is a standard LFSR with the 9 and 6 feedback bits (as in the lecture notes), expect when the counter reaches the value 0000000001, instead of going to 1000000000, it goes to 0000000000, because the both the NAND gate, which checks for Q0Q1Q2Q3Q4Q5Q6Q7Q8, and the XOR1 gate is on, so XOR2 is off, so the next pushed bit is 0. When the value is 0000000000, the NAND gate still on, but XOR1 is now off, so XOR2 is on, so the next pushed bit is 1000000000. In all other cases, the NAND gate is off, so it's as if the XOR2 gate wasn't there and the circuit acts as a standard counter.

If we didn't care about the circuit looping into and back to 0000000000, we could've just added a single AND gate checking for all-zeros going into the main OR gate, which would get us *from* the lock state but never *into* it.

Both the regular synchronous counter and the LFSR use 10 D-flip-flops, so the difference in the gate count will be in the auxiliary gates. A regular synchronous counter uses 1 AND gate and 1 OR gate per bit, so for a 10-bit counter that's 10*3=30 gates. In our extended 10-bit LFSR counter, we have one 10 input NAND gate, two 2 input XOR gates and one 2 input OR gate, which is 10/2+2*1+1.5=8.5 gates. We can see that this is much less.

3

decimal

99	1100011
↓ 10	0001010

$$RST_{HELPER} = RST + Q_0 Q_1 Q_5 Q_6$$

$$Q_0' = \overline{RST_{HELPER}} \cdot \overline{Q_0}$$

$$Q_1' = RST_{HELPER} + (Q_0 \oplus Q_1)$$

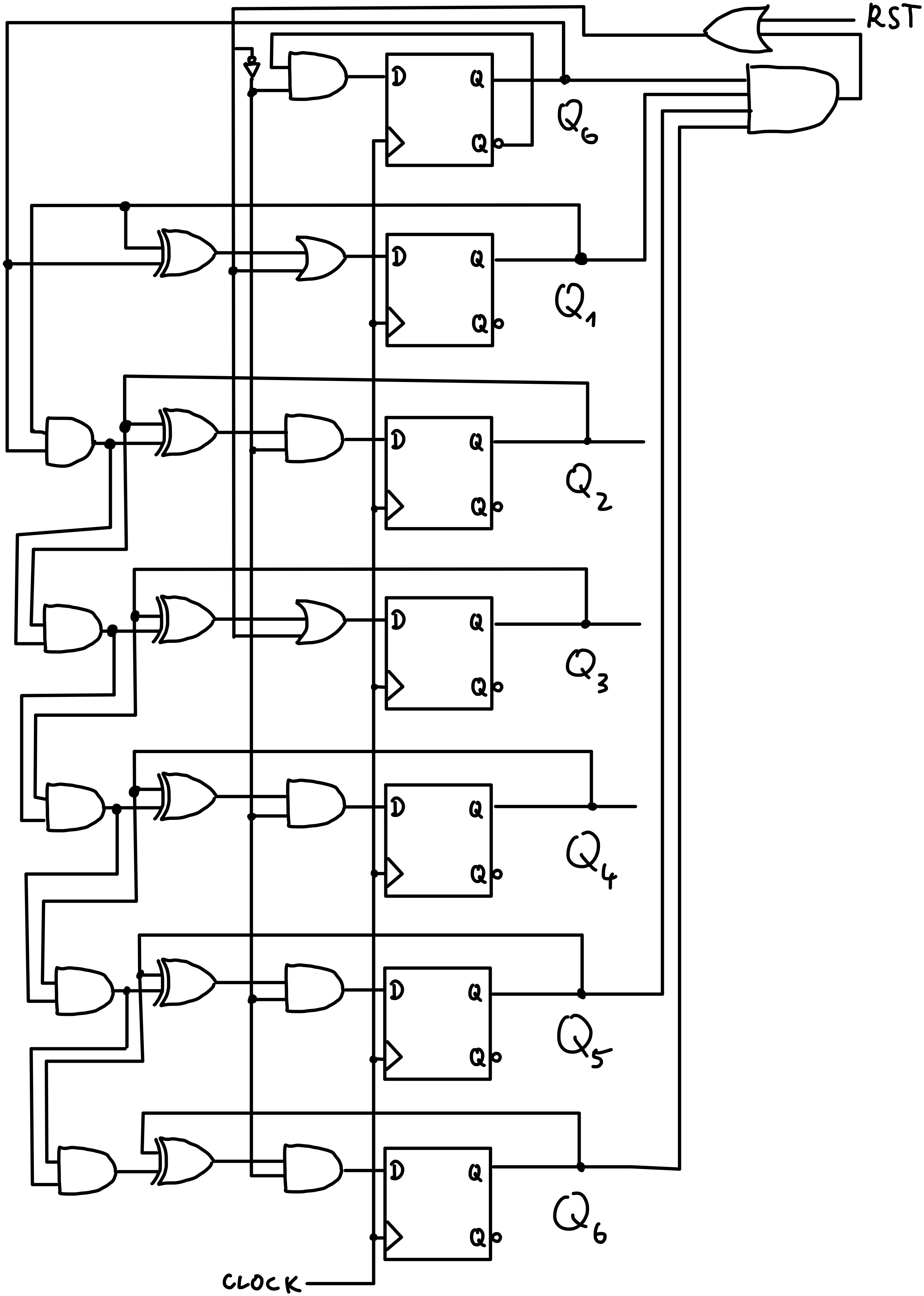
$$Q_2' = \overline{RST_{HELPER}} \cdot Q_0 Q_1 \oplus Q_2$$

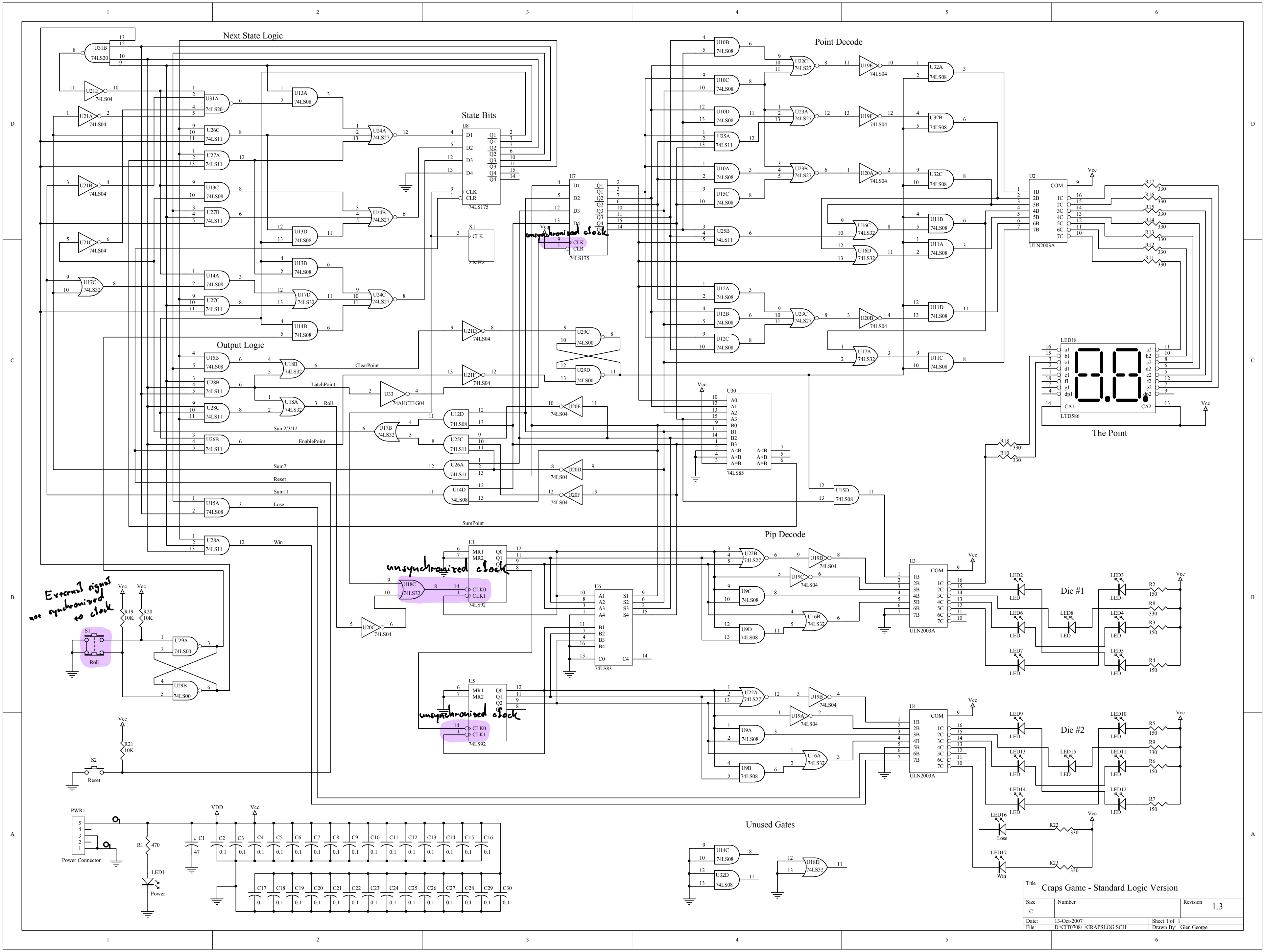
$$Q_3' = RST_{HELPER} + (Q_0 Q_1 Q_2 \oplus Q_3)$$

$$Q_4' = \overline{RST_{HELPER}} \cdot Q_0 Q_1 Q_2 Q_3 \oplus Q_4$$

$$Q_5' = \overline{RST_{HELPER}} \cdot Q_0 Q_1 Q_2 Q_3 Q_4 \oplus Q_5$$

$$Q_6' = \overline{RST_{HELPER}} \cdot Q_0 Q_1 Q_2 Q_3 Q_4 Q_5 \oplus Q_6$$


RST
D
Q
Q
Q0
D
Q
Q
Q1
D
Q
Q
Q2
D
Q
Q
Q3
D
Q
Q
Q4
D
Q
Q
Q5
D
Q
Q
Q6
CLOCK



Next State Logic
State Bits
Point Decode
Output Logic
Pip Decode
The Point
Die #1
Die #2
Unused Gates
ClearPoint
LatchPoint
Roll
Sum2/3/12
EnablePoint
Sum7
Reset
Sum11
Lose
Win
SumPoint
2 MHz
Power Connector
Power
unsynchronized clock
unsynchronized clock
unsynchronized clock
External signal not synchronized to clock
Title
Craps Game - Standard Logic Version
Size
C
Number
Revision
1.3
Date: 13-Oct-2007
Sheet 1 of 1
File: D:\CIT0708\..\CRAPSLOG.SCH
Drawn By: Glen George