ENGR 271
Philip Nevins

Problem #1

Given: Use VMLAB to write a register-level AVR program that makes R0 and R1 into a 16-bit Cylon eye. Start with a 1 in the LSB of R1. With each time interval the 1 will move to the left (aka 1-hot). The movement will continue into R0. When the 1 gets to the MSB of R0, it changes to moving to the right until it gets back to the starting position, at which point it repeats the pattern, as this animated gif shows.

Find: Design Cyclon Eye

Solution:

```
.include "C:\VMLAB\include\m168def.inc"

loop5:

ldi r16, 1     ; loads r16 with 00000001
mov r1, r16    ; moves r16 into r1
clr r16        ; clears r16

loop1: ; loop1 shifts left through r1, stops when all 0
lsl r1
brne loop1

ldi r17, 1  ; loads r17 with 00000001
mov r0, r17  ; moves r17 into r0
clr r17     ; clears r17

loop2: ; loop2 shifts left through r0, stops when all 0
lsl r0
brne loop2

ldi r18, 128 ; loads r18 with 10000000
mov r0, r18   ; moves r18 into r0
clr r18      ; clears r18

loop3: ; loop3 shifts right through r0, stops when all 0
lsr r0
```

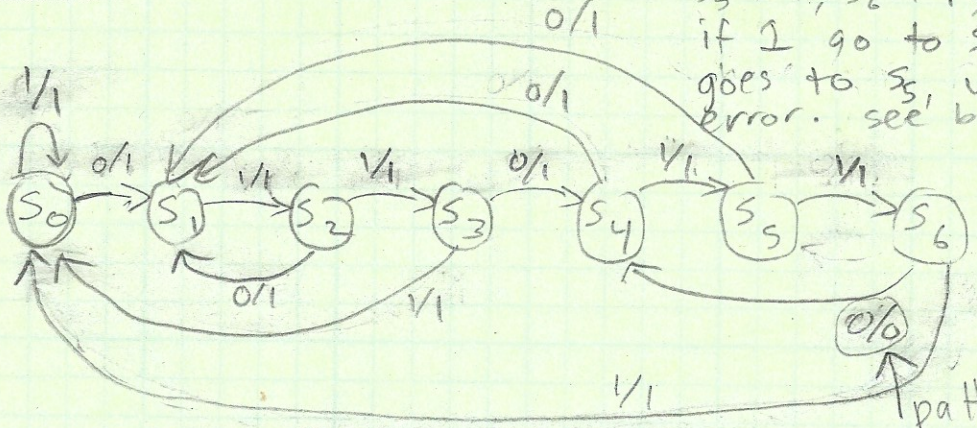
```
brne loop3

ldi r19, 128  ;loads r19 with 10000000
mov r1, r19   ; moves r19 into r1
clr r19       ; clears r19

loop4: ; loop4 shifts right through r1, stops when all 0
lsr r1
brne loop4

breq loop5 ; loops back to start
```

Given: Draw a state diagram for a Mealy machine to find the pattern "0110110" in a series of binary #s, where each new bit appears on the input to the circuit w/ each Clk pulse. Active-low output indicates when the last # in the pattern is detected. Allow overlaps

Find: Stat diagram described above

Solution:

S_0 => "ready" state. $S_1 \to 0$, $S_2 \to 1$, $S_3 \to 1$, $S_4 \to 0$, $S_5 \to 1$, $S_6 \to 1$ => if 0, put active low, if 1 go to S_0. If S_6 (out=1) goes to S_5, we will have error. see below



pattern detected, output 0 for active low. to allow overlaps, it signals 0 that pattern is found, and goes to S_4 to "keep going"

scenario: S_6 goes to S_5 on 1/1

S_5 (1/1) -> S_6; we see 1 pattern

S_6 (1/1) -> S_5; we see 11 pattern

S_5 (1/1) -> S_6; we see 111 pattern

we are stuck detecting 1s forever and if a 0 is detected at S_6, it will signal pattern detected, but it will have seen 1111...0, this is why S_6 (1/1) feeds to S_0

Given:

S	X=0	X=1	Z
A	B	D	0
B	C	B	0
C	B	A	1
D	B	C	0

S*

State assignments

A = 00

B = 01

C = 11

D = 10

Find: Synthesize a state machine w/ given information. Use 2 state variables, Q_1, Q_2. Write out the excitation equations & draw schematic using NAND gates and DFFs.

Solution:

S	X=0 (input)	X=1	Z (output)
00	01	10	0
01	11	01	0
11	01	00	1
10	01	11	0

S*

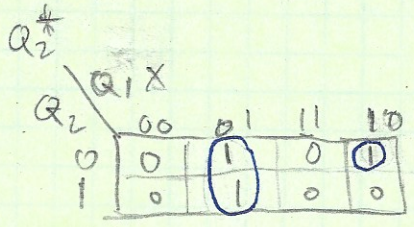
*Based on this state table, we have a Moore machine

State table

Q_2 Q_1	X	Q_2^* Q_1^*	Z
00	0	01	0
00	1	10	0
01	0	11	0
01	1	01	0
11	0	01	1
11	1	00	1
10	0	01	0
10	1	11	0

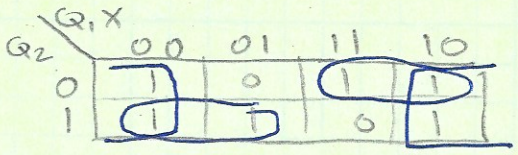
excitation equations

Q_2^*



$$Q_2^* = \bar{Q}_1 X + \bar{Q}_2 Q_1 \bar{X}$$

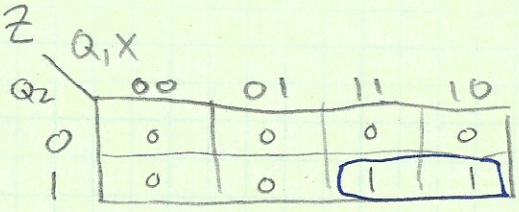
Q_1^*



$$Q_1^* = \bar{X} + \bar{Q}_2 Q_1 + Q_2 \bar{Q}_1$$

$$Q_1^* = \bar{X} + Q_2 \oplus Q_1$$

Z



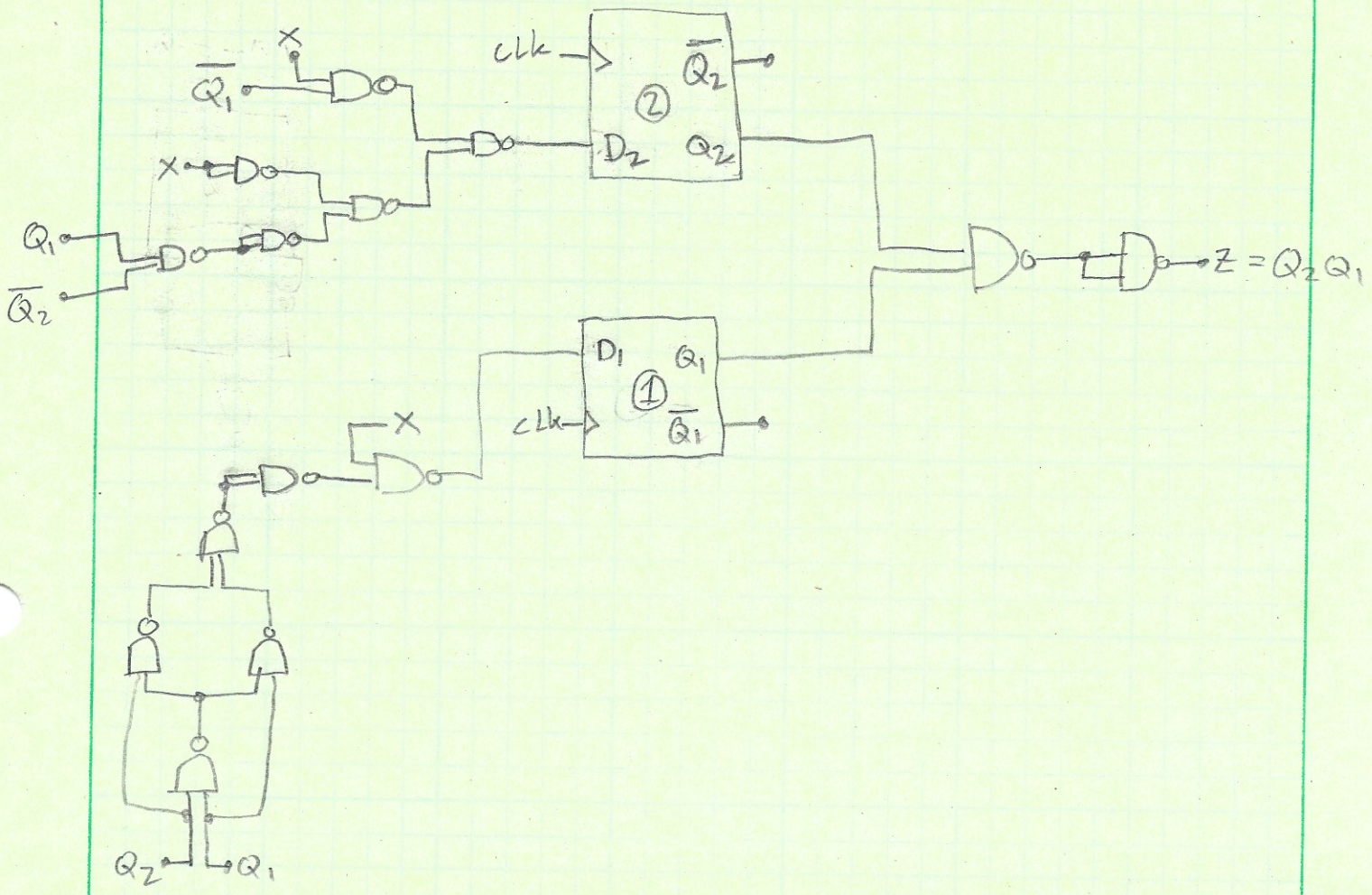
$$Z = Q_2 Q_1$$

schematic next Page

$D_1 = \bar{x} + Q_2 \oplus Q_1$ Logic Diagram

$D_2 = \bar{Q}_1 x + \bar{Q}_2 Q_1 \bar{x}$

$Z = Q_2 Q_1$

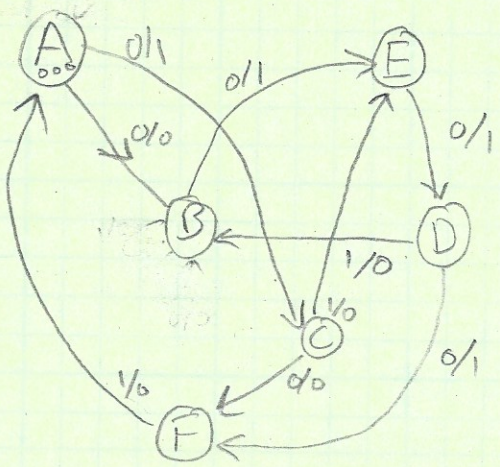


Given:

state	XY = 00	01	11	10
A	B	C	—	A
B	B	E	—	B
C	F	C	—	E
D	D	F	—	B
E	D	E	—	E
F	F	F	—	A

Find: Assign state variables, [Q bits] that avoid critical signal racing. Can add more states, but need to use minimum # Q bits. Assign all 0s to State A. Draw an adjacency diagram for original flow table & write modified flow table & another adjacency diagram to support final state variable assignment

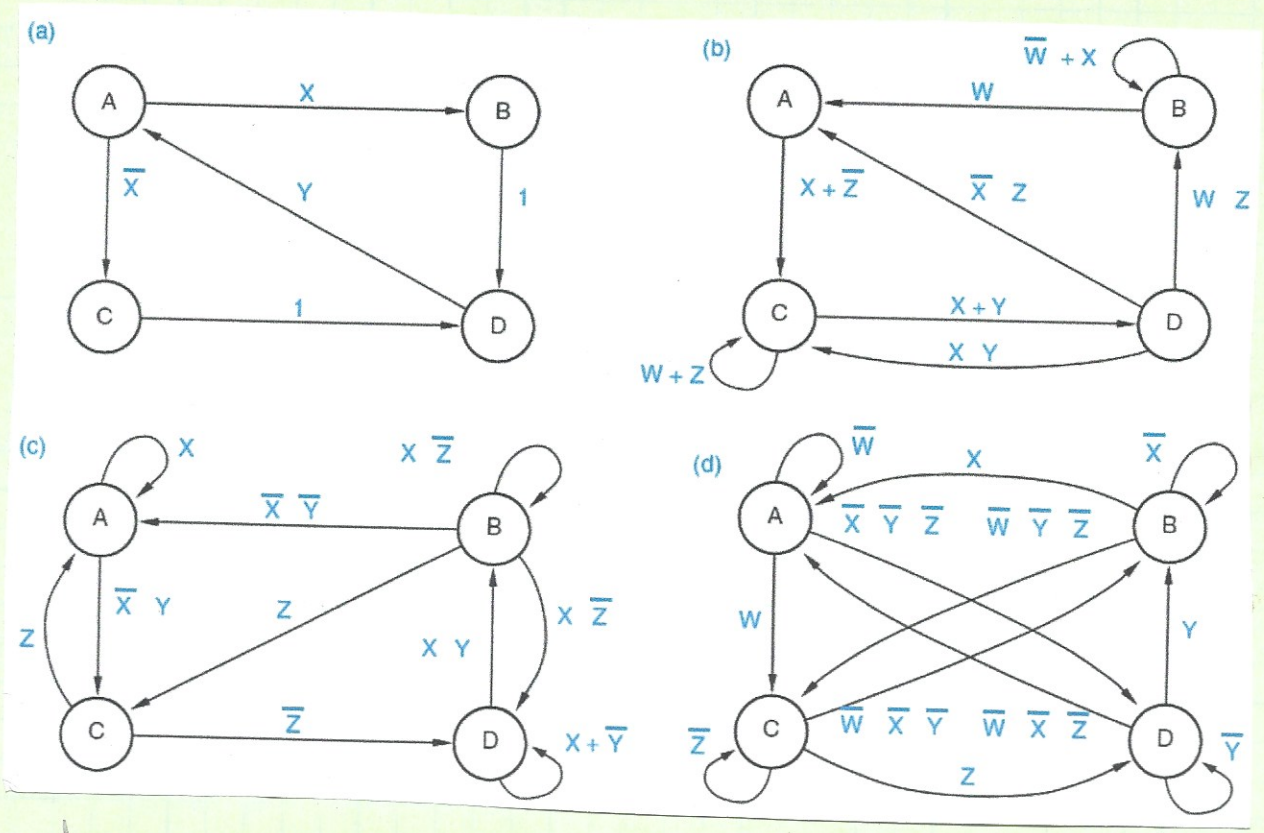
Solution: [X/Y] [input/output]



Modified state table

Present state	Next state input x=0	Next state input x=1	Output x=0 $y_1(X=0)$	Output x=1 $y_2(X=1)$
A	B, C	—	0, 1	—
B	E	—	1	—
C	F	F	0	0
D	F	B	1	0
E	D	—	0	—
F	—	A	—	0

Given:



Find: List all ambiguities in the given state diagrams

Solution:

a) state A

X \ Y	0	1
0	C	C
1	B	B

No ambiguities

state B & C has unconditional transitions to D

∴ no ambiguities

state D

X \ Y	0	1
0	—	A
1	—	A

since 2 cells have no value, state D is ambiguous

b) state A is ambiguous

wx \ yz	00	01	11	10
00	C	—	—	C
01	C	C	C	C
11	C	C	C	C
10	C	—	—	C

state B is ambiguous

wx \ yz	00	01	11	10
00	B	B	B	B
01	B	B	B	B
11	B+A	B+A	B+A	B+A
10	A	A	A	A

*Note: all circled transitions are why the state is ambiguous. No transition or multiple destinations = ambiguous

cont →

b cont...

State C is ambiguous

wx \ yz	00	01	11	10
00	—	C	C+D	D
01	D	C+D	C+D	D
11	C+D	C+D	C+D	C+D
10	C	C	C+D	C+D

State D is ambiguous

wx \ yz	00	01	11	10
00	—	A	A	—
01	—	—	C	C
11	—	B	B+C	C
10	—	A+B	A+B	—

c

State A is ambiguous

wx \ yz	00	01	11	10
00	—	—	C	C
01	A	A	A	A
11	A	A	A	A
10	—	—	C	C

State B is ambiguous

wx \ yz	00	01	11	10
00	A	A+C	C	—
01	D+B	C	C	D+B
11	D+B	C	C	D+B
10	A	C+A	C	—

State C is not ambiguous

wx \ yz	00	01	11	10
00	D	A	A	D
01	D	A	A	D
11	D	A	A	D
10	D	A	A	D

State D is ambiguous

wx \ yz	00	01	10	11
00	D	D	—	—
01	D	D	D+B	D+B
11	D	D	D+B	D+B
10	D	D	—	—

D

State A is ambiguous

wx \ yz	00	01	11	10
00	D+A	A	A	A
01	D+A	A	A	A
11	C	C	C	C
10	D+C	C	C	C

State B is ambiguous

wx \ yz	00	01	11	10
00	C+B	B	B	B
01	C+A	A	A	A
11	A	A	A	A
10	B	B	B	B

State C is ambiguous

wx \ yz	00	01	11	10
00	B+C	B+D	D	C
01	C	D	D	C
11	C	D	D	C
10	C	D	D	C

State D is ambiguous

wx \ yz	00	01	11	10
00	A+D	D	B	A+B
01	D	D	B	B
10	D	D	B	B
10	D	D	B	B