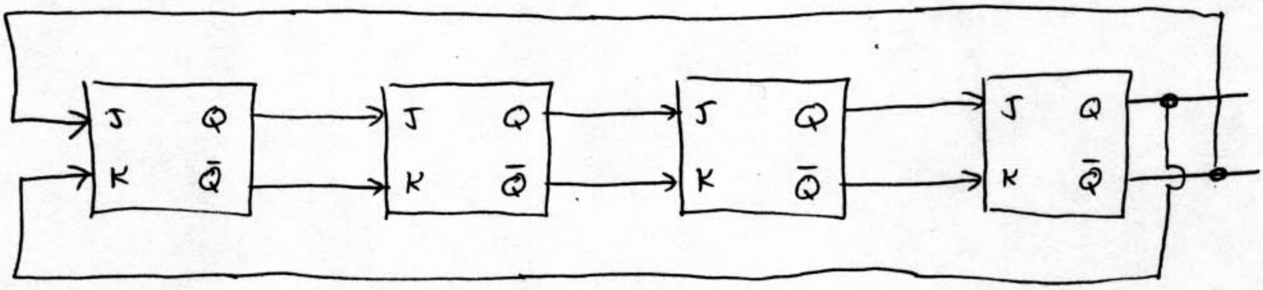
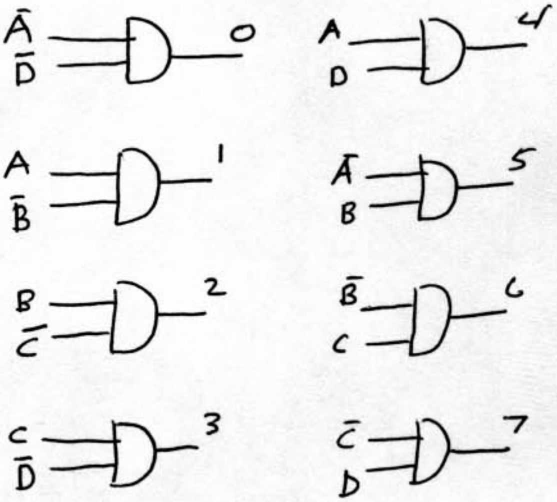
F.B. shift register as a decoded "Johnson" counter,
AKA a "Mobius" counter, or a "twisted tail" counter.



initially cleared, then states on succesive clocks will be:

state	A	B	C	D
0	0	0	0	0
1	1	0	0	0
2	1	1	0	0
3	1	1	1	0
4	1	1	1	1
5	0	1	1	1
6	0	0	1	1
7	0	0	0	1
repeats	0	0	0	0

Decoding for unique count
i.e. state is



N stages, 2N states

Most common chip of this type is the 4017 decade Johnson with 5 stages and 10 outputs. Each output goes hi in turn. Can e.g. direct a sequence of 10 events.

Note: above counter has one "false" possible sequence, Exercise: deduce what it is.

Possible to add circuitry to detect and correct it.

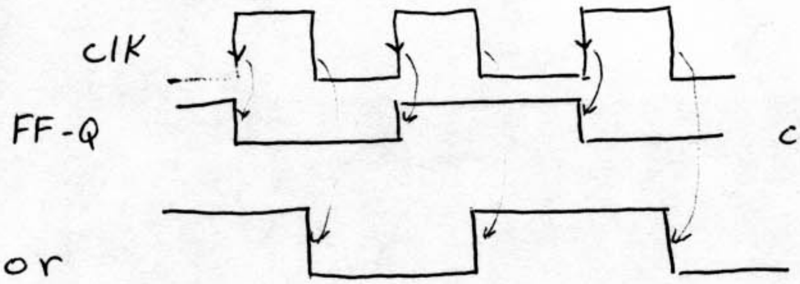
Counters

Two broad classes of counters:

1. Asynchronous (AKA ripple): individual FF's in a cascade are not controlled by a common system clock. Instead each stage receives its "clock" input from the output of the preceding stage (or perhaps from stages and comb. logic). Used as event totalizers or frequency dividers.

2. Synchronous: all FF's have a common clock so that all transitions are "locked" to the system clock transitions. More common than ripple.

Basic element of both is (are) FF operating in toggle mode.



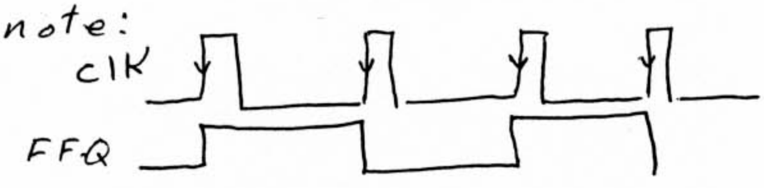
can toggle on leading edge

or on trailing edge

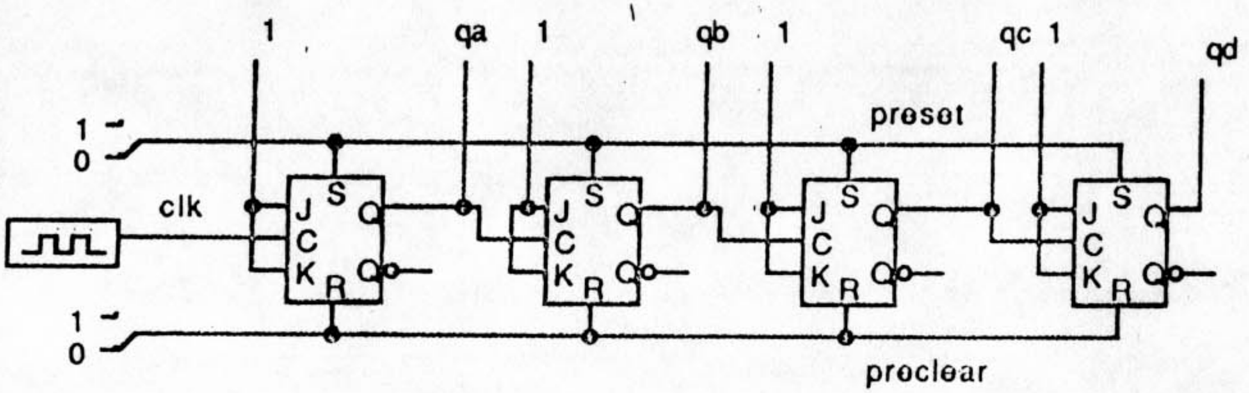
depending on use of +ve o -ve edge type FF

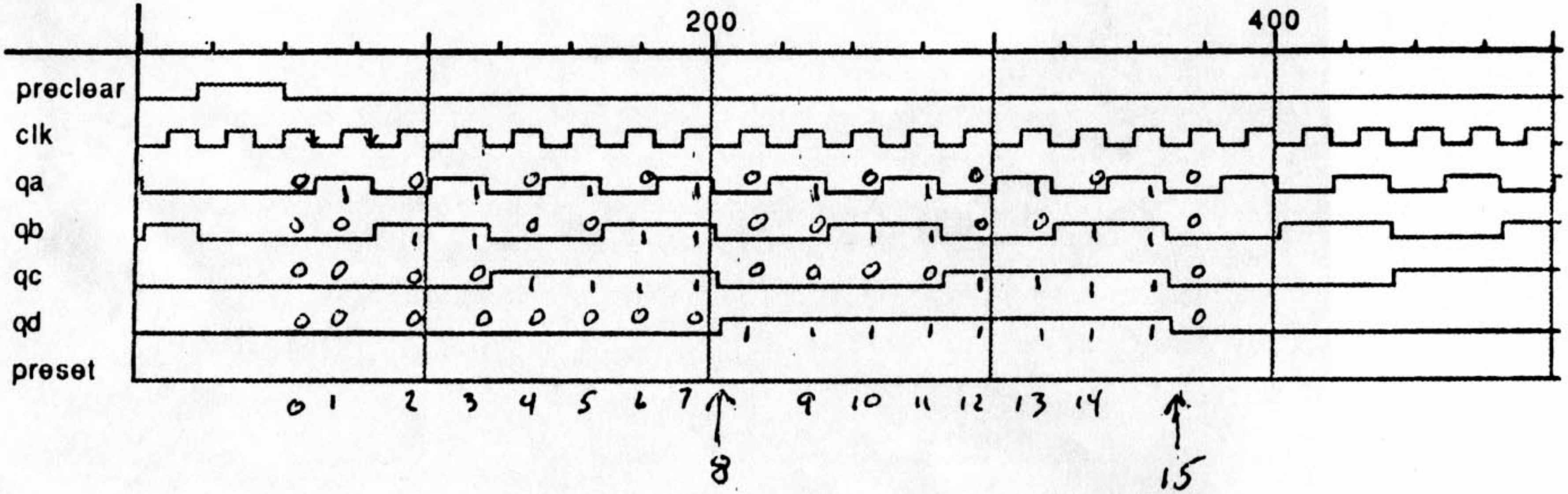
note that toggling implements a divide-by-two, i.e, rate is 1/2 that of input clock (period is twice)

also note:



i.e. toggle output has 50% duty cycle even if clK does not.


1
qa
1
qb
1
qc
1
qd
preset
1
0
clk
J
S
C
K
R
1
0
preclear


200
400
preclear
clk
qa
qb
qc
qd
preset
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

Misc notes on foil of basic binary ribble counter:

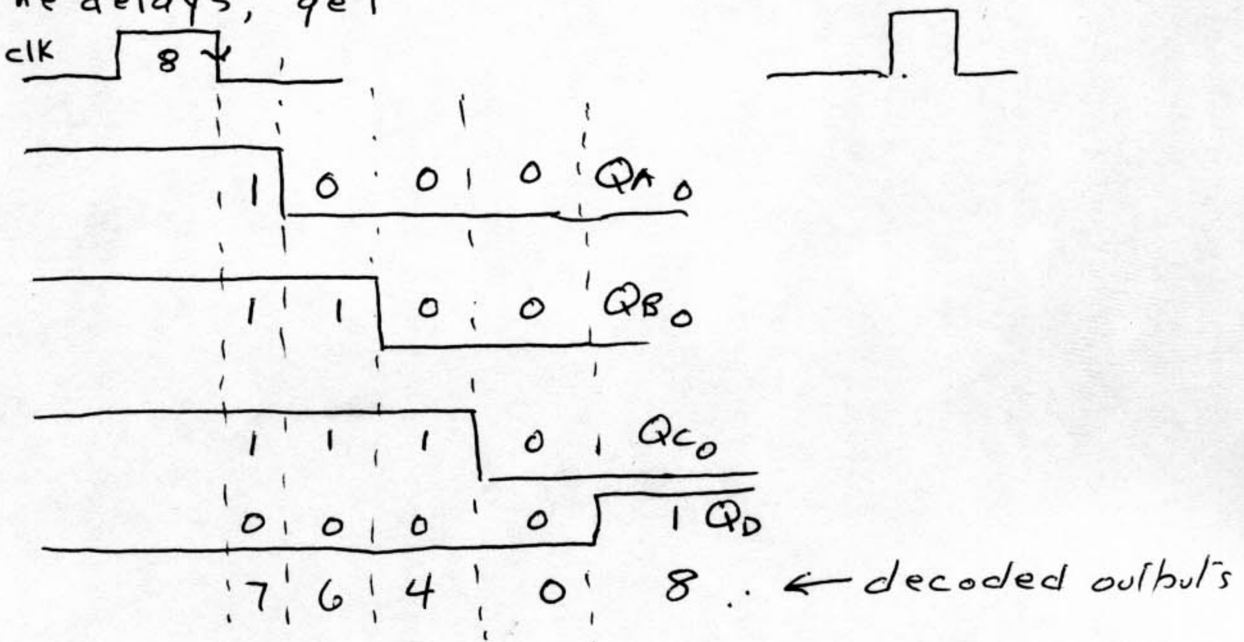
1. Waveforms indicate -ve edge triggering, ⇒ diagrom missing bubble at clK inputs.

2. If outputs were taken from $\bar{Q}$ instead of from Q as shown - then would be a "count-down" counter as count would then be 15→14-13→etc.

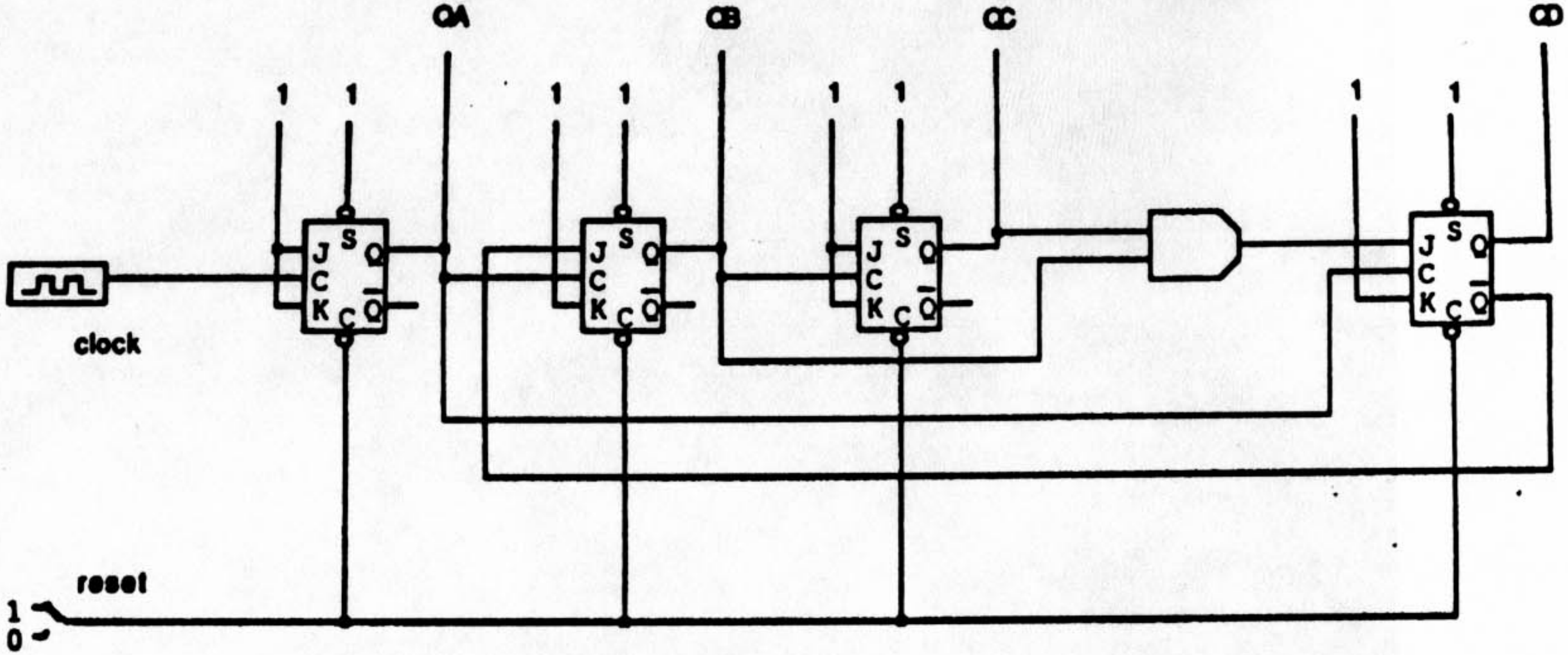
   Implies if put a 2 to1 MUX to select outputs from either the FF's Q or $\bar{Q}$ would have a counter that either counts UP or counts DN.

3. Often have decoding logic to indicate when count1 or count 2 or count 3 etc have occurred:

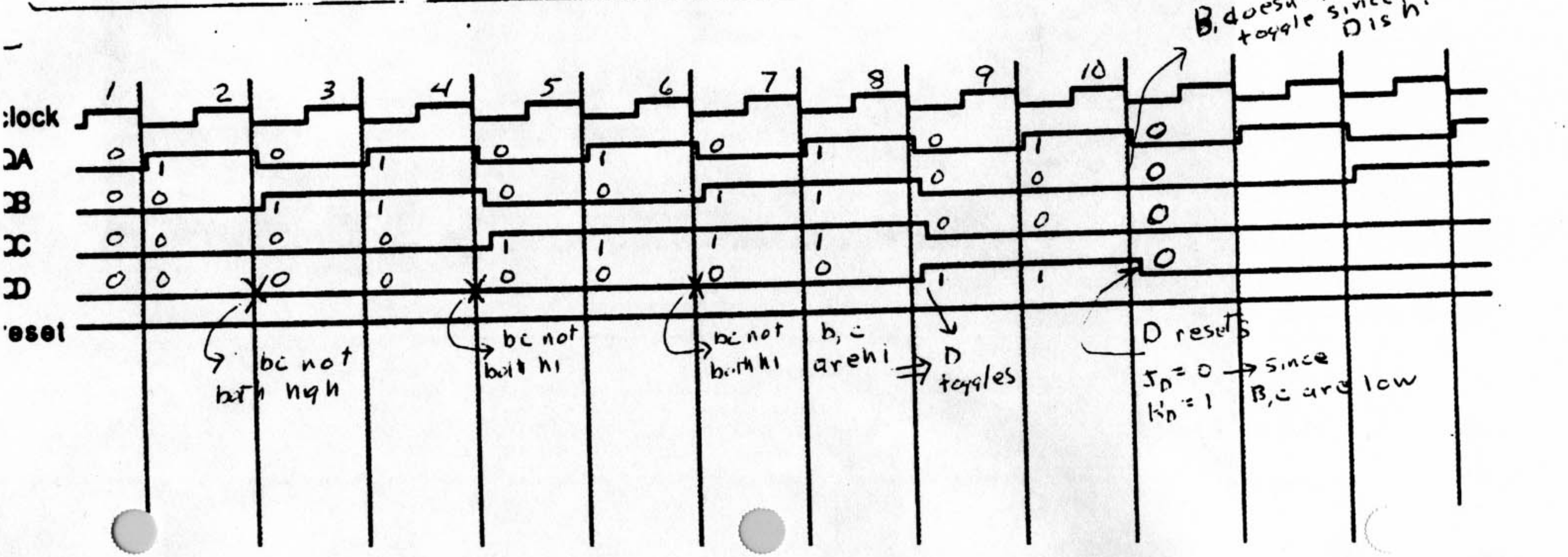
   Now look at outputs just after clK 8 and exaggerote the delays, get



After 8th clK count should be 8 but between 7and 8 decoders for counts 6, 4, 0 would go high — have false decoding due to the ripple effect, i.e. the "effective system clock" ripples thru the counter. Can give false decoding and also limits speed of operation,



Decade Ripple Counter



Notes on decade ripple counter:

1. FF-A just toggles on clock edge ↓
2. FF-C " " " clock from Q_B
3. FF-B toggles on A as long as its J input is high, ie as long as $Q_D = 0$ ($\overline{Q}_D = 1$) which is up until count 8 is finished.
4. FF-D is initially 0 and can't set (on clock from A) until its J input is high, ie until both B and C are both hi. FF-D toggles then on clk 8.
5. Entire counter is reset to zero after 10 clocks i.e. it is a decade counter (modulo 10 counter).
6. Since FF-A is just a ÷2 then FF's B, C, D form a divide by 5 counter.

Synchronous Counters

Casual look with a scope at the outputs of ripple and sync. counters would tend to indicate they are the same. Important difference tho, sync. counters employ common clock to all stages so that all output transitions are sync'd to the clock. Avoids the false decoding and speed problems of ripple counters and in addition allows for more formal methods of design.

Synchronous binary counter - again toggle FF's are the basic stage.

Consider what the count sequence would be for say 3 stages:

clk	Q_2	Q_1	Q_0
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1
repeat	0	0	0

note 1st stage toggles on each clock pulse

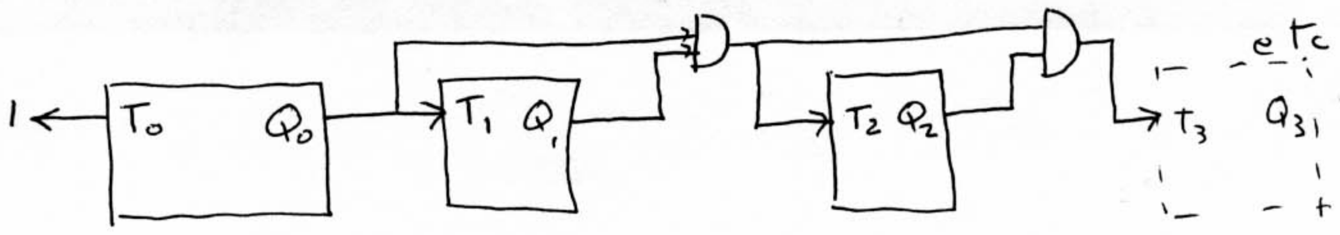
also note: each higher stage toggles when all previo (lower order) stages are 1

Hence the T (toggle) input on the stages are

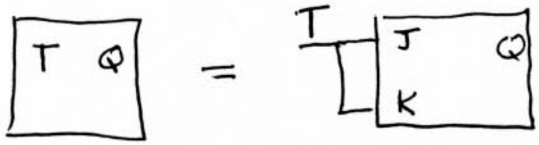
$$T_0 = 1 \quad \text{(i.e toggles on each clock)}$$
$$T_1 = Q_0$$
$$T_2 = Q_1 \cdot Q_0 = Q_1 \cdot T_1$$

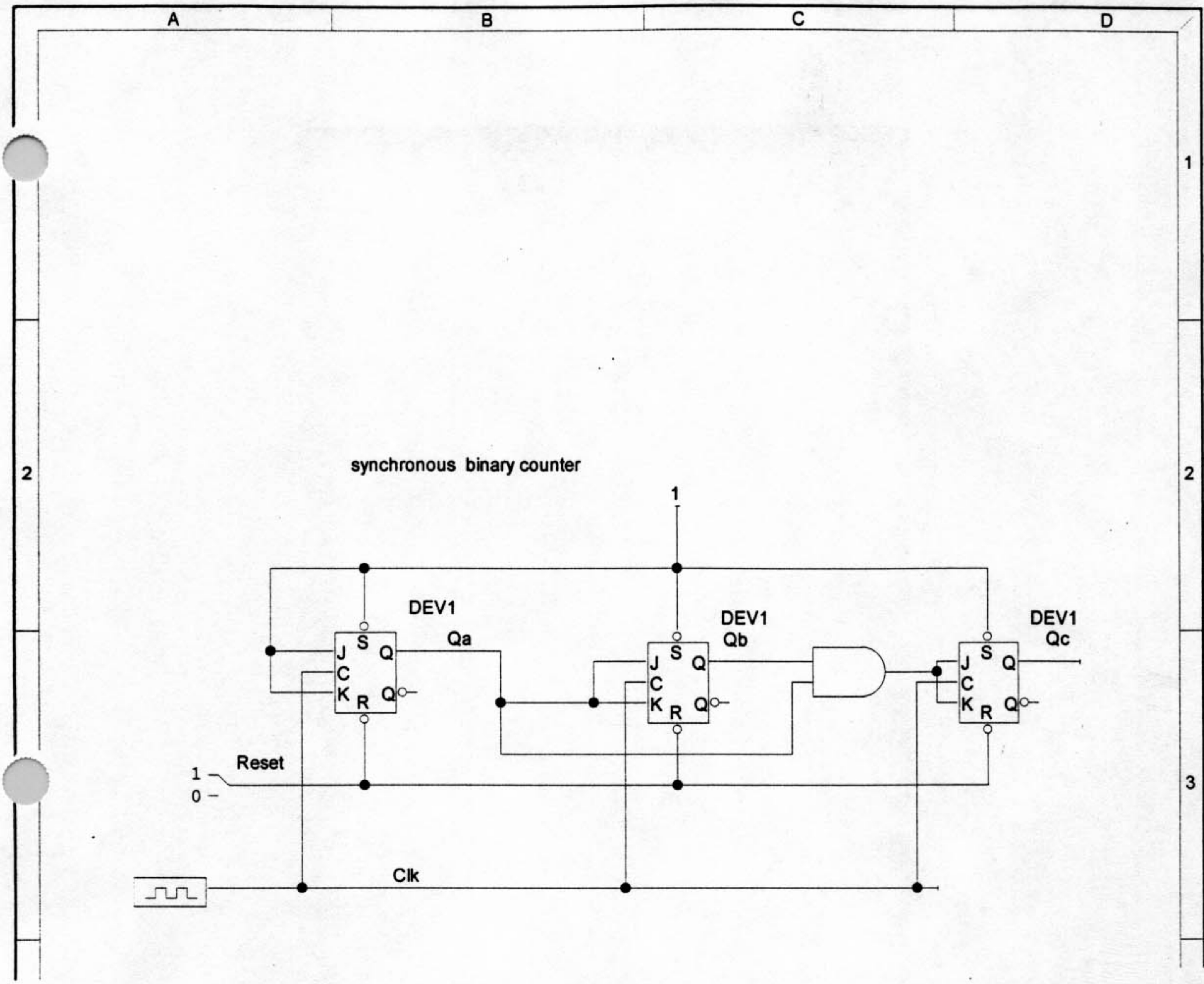
if there were 4 stages would have $T_3 = Q_2 \cdot Q_1 \cdot Q_0 = Q_2 \cdot T_2$

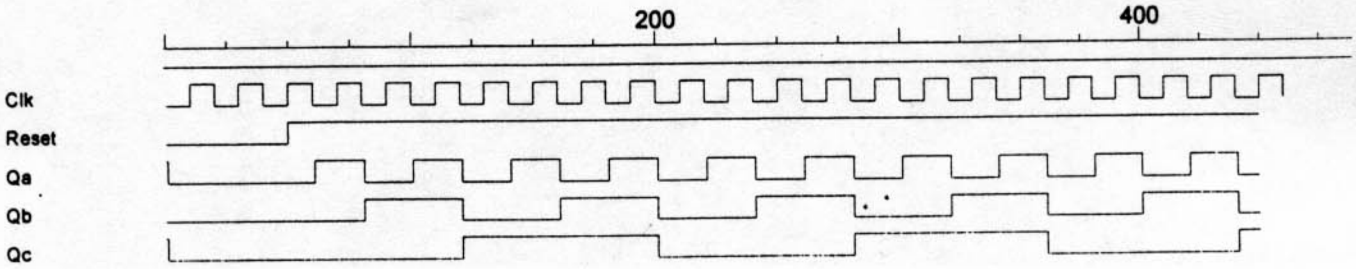
so structure is:



note:

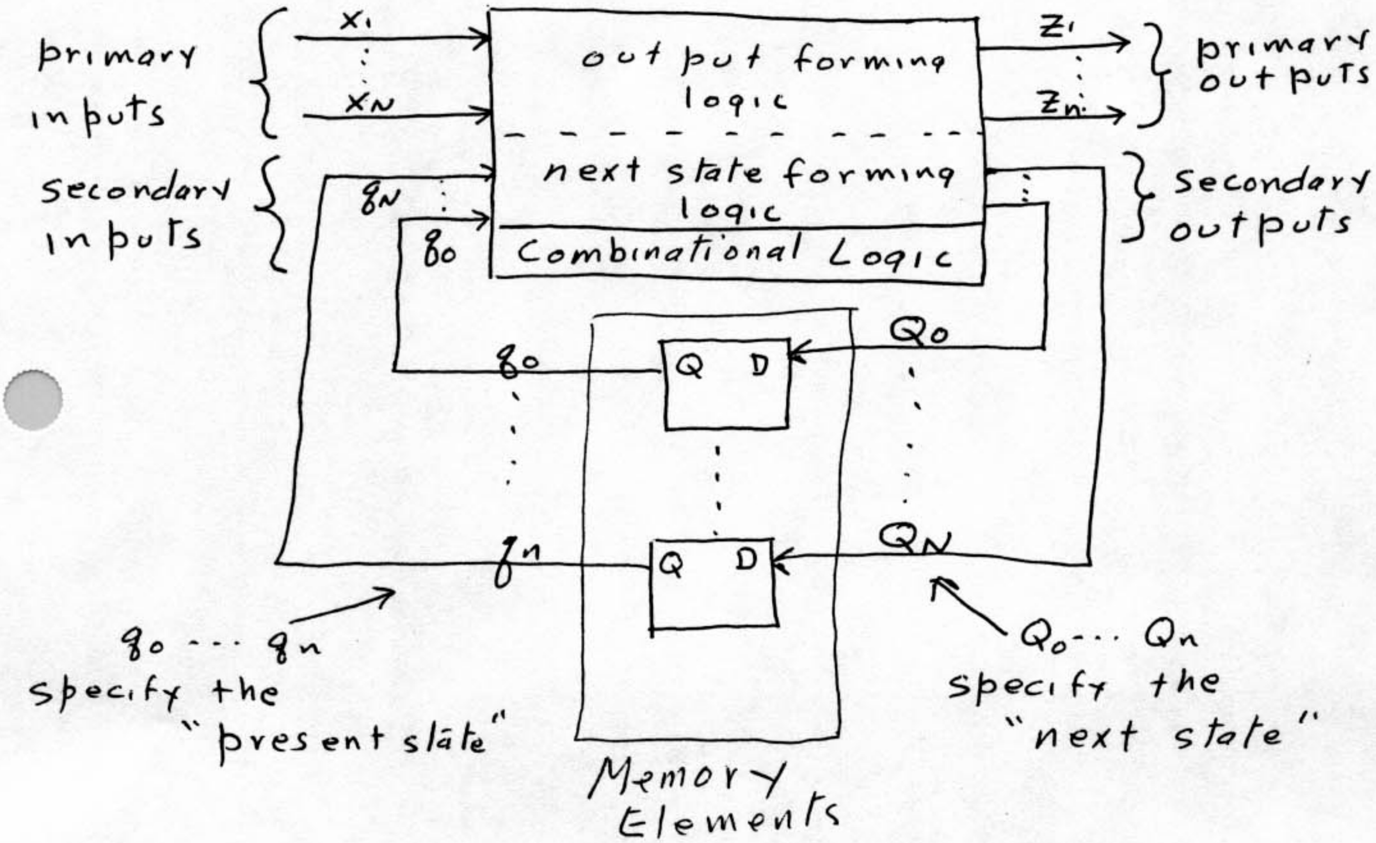



A
B
C
D
1
2
3
synchronous binary counter
1
DEV1
DEV1
Qb
DEV1
Qc
Qa
J
C
K
S
R
Q
Q
Reset
1
0
Clk


200
400
Clk
Reset
Qa
Qb
Qc

Synchronous counters are a special case of synchronous sequential systems which have a high level architecture. Introduces :

Basic Finite State Machine Architecture (Conceptual Model)



Notes: 1. Quite possible for an output to be both primary and secondary.

2. y and Y often used instead of q and Q

3. Actual circuit diagrams seldom drawn this way - it is a conceptual model.

4. Memory elements commonly FF but not always

5. Synchronous ⇒ all share common clock.

6. Asynchrous state machine = no common clock.

7. If outputs are generated solely from the present state inputs (the q's) then the SM is called a "Moore" machine.

8. If output(s) are generated using both the primory inputs and the present state inputs, then the SM is termed a "Mealey" machine.

9. Counters - plain vanilla variety - do not have any primary inputs (clock doesn't count as one). SM's with no primary input are termed "autonomous" Hence these counters are by definition Moore machines.

Without going too deeply into the subject can illustrate a simple design procedure —

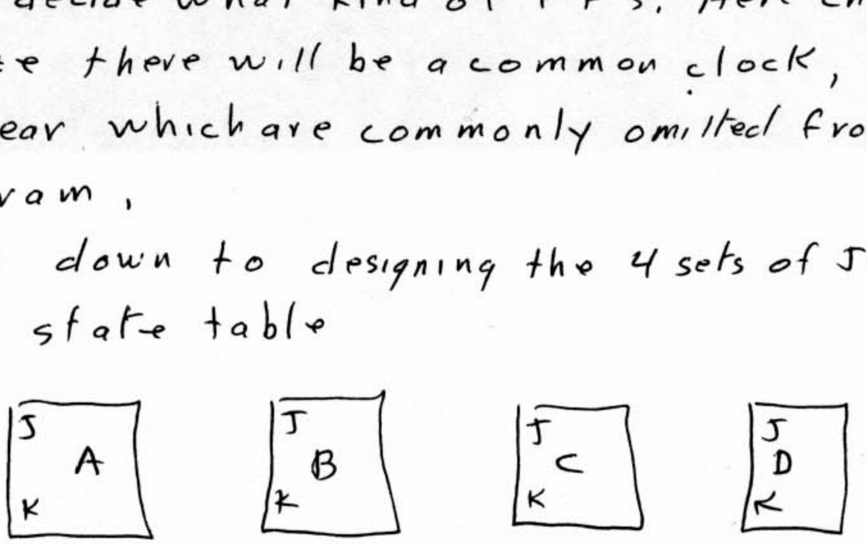
"Design a synchronous decade counter"

First note that 4 FF's will be needed for the 10 states, so there will be 6 don't cares.

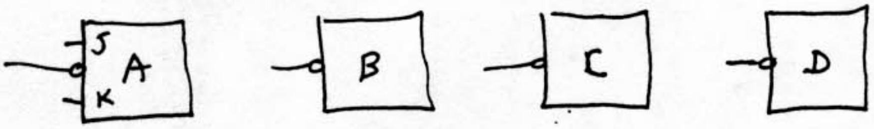
Need to decide what kind of F-F's. Here choose J-K's.

Recognize there will be a common clock, preset, preclear which are commonly omitted from ckt diagram.

So boils down to designing the 4 sets of J-K inputs from a state table



Decade synch. counter: = autonomous state machine
Need 4 FF's (3 FF could only count to 0→7)
Common clock, power, gnd, preset, preclear ; assume use of J-K's)



construct state table:

	Present State				Next State to be				Required excitations				
state #	D	C	B	A	D^+	C^+	B^+	A^+	J_A	K_A	- - -	J_D	K_D
0	0	0	0	0	0	0	0	1	1	φ	.....	0	φ
1	0	0	0	1	0	0	1	0	φ	1		0	φ
2	0	0	1	0	0	0	1	1	1	φ		0	φ
3	0	0	1	1	0	1	0	0	φ	1		0	φ
4	0	1	0	0	0	1	0	1	1	φ		0	φ
5	0	1	0	1	0	1	1	0	φ	1		0	φ
6	0	1	1	0	0	1	1	1	1	φ		0	φ
7	0	1	1	1	1	0	0	0	φ	1		1	φ
8	1	0	0	0	1	0	0	1	1	φ		φ	0
9	1	0	0	1	0	0	0	0	φ	1		φ	1
	0	0	0	0	0	0	0	1					

Required excitations from excitation table for J-K's:

Q_n	Q_{n+1}	J_n	K_n	
0	0	0	φ	leave alone (0,0) or reset (0,1)
0	1	1	φ	toggle (1,1) or set (1,0)
1	0	φ	1	toggle (1,1) or reset (0,1)
1	1	φ	0	leave alone (0,0) or set (1,0)

Have 8 comb. logic designs to do (J_A, K_A - - - - J_D, K_D)
Note 10 → 15 don't care combos.

DC \ BA	00	01	11	10
00	0	1	3	2
01	4	5	7	6
11	ϕ 12	ϕ 13	ϕ 15	ϕ 14
10	8	9	ϕ 11	ϕ 10

don't cares for states 10 → 15

DC \ BA	00	01	11	10
00				
01			1	
11	ϕ	ϕ	ϕ	ϕ
10			ϕ	ϕ

$J_D = ABC$

DC \ BA	00	01	11	10
00	ϕ	ϕ	ϕ	ϕ
01	ϕ	ϕ	ϕ	ϕ
11	ϕ	ϕ	ϕ	ϕ
10	0	1	ϕ	ϕ

$K_D = A$

more ϕ for states 0 → 7

etc for others (do as home exercise)

get:

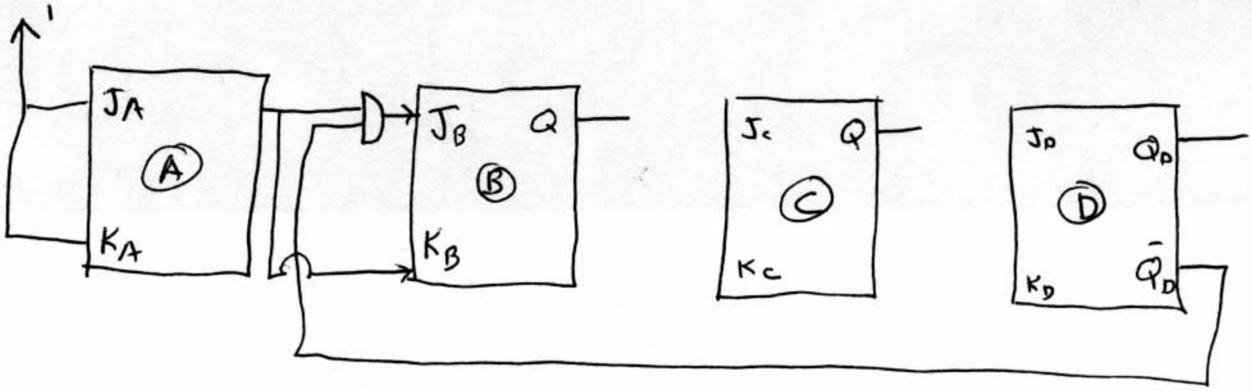
$J_A = 1 \, , \, K_A = 1$ (by inspection)

$J_B = A\bar{D} \, , \, K_B = A$

$J_C = AB \, , \, K_C = AB$

$J_D = ABC \, , \, K_D = A$

so circuit would be (only first two FF connections show)


1
J_A
A
K_A
J_B
Q
B
K_B
J_c
Q
C
K_c
J_D
Q_D
D
K_D
$\bar{Q}_D$