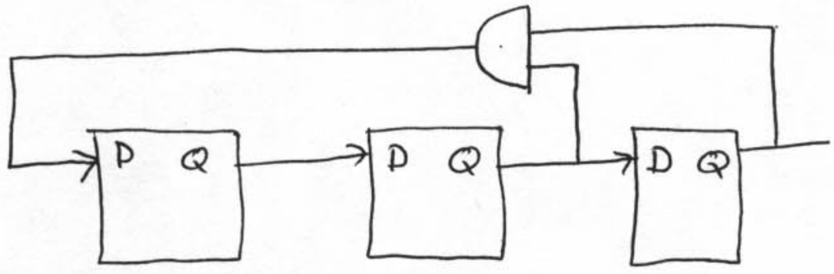
# Shift registers with feed back

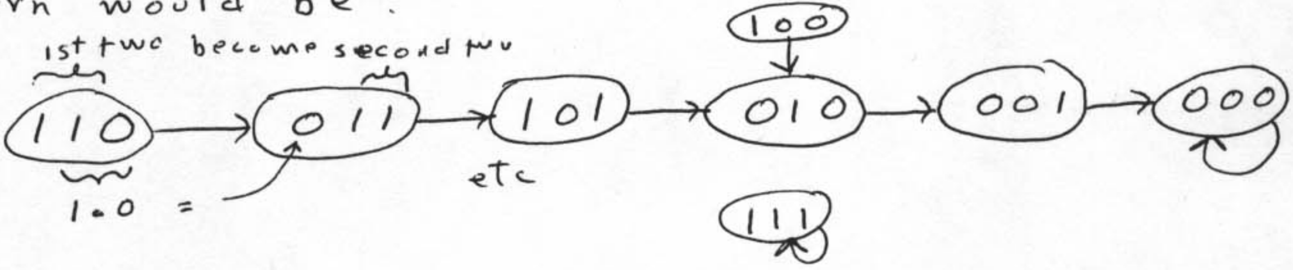
Possible to combine outputs of several stages of a shift register and feed resultant back to input.

e.g.



common clock

Say preset to 110. Then on successive clocks pattern would be:



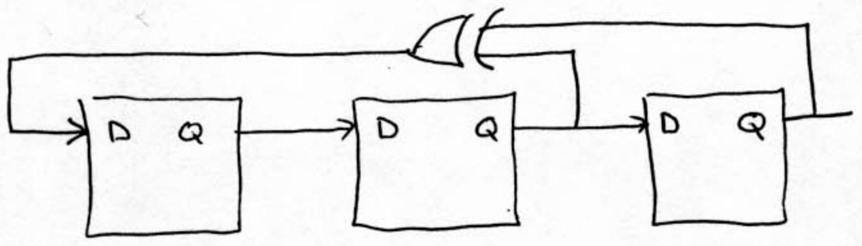
Starting with 110 pattern would shift as above and end with $\boxed{000}$. Six states in this pattern – 3 bits implies 8 possible states; two not in main pattern are 100 and 111 which behave as shown.

Above is a shift register with f.b. acting as a very basic sequence generator. It is <u>non-circulating</u> and <u>not maximal length</u>.

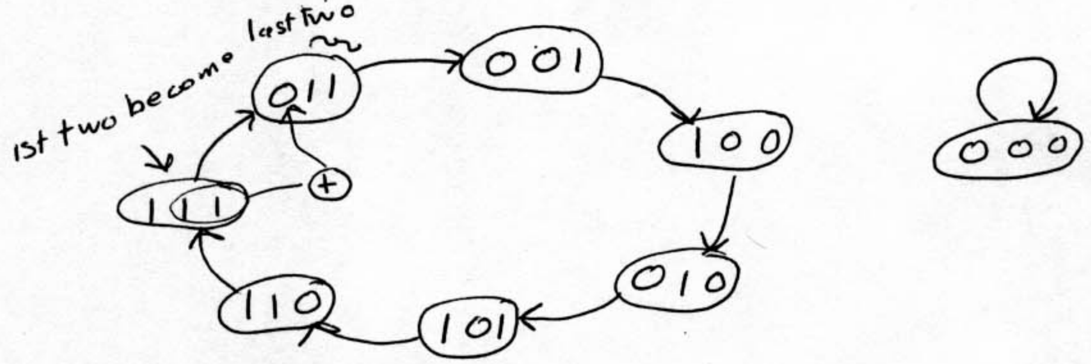
non-circulating – goes thru one sequence and stops

not maximal length – # of states in the sequence is less than max possible (which is $2^N-1$) neglecting the all zero state.

Now consider different f.b. element:



say preset to 111 then sequence will be



This is a <u>circulating</u> sequence generator and is maximal length ($2^N-1 = 7$), all zero state not in seq.

F.B. shift registers of this type are called MLS (maximal length sequence) generators. Note doesn't make any difference for starting state - rest is part of the sequence.

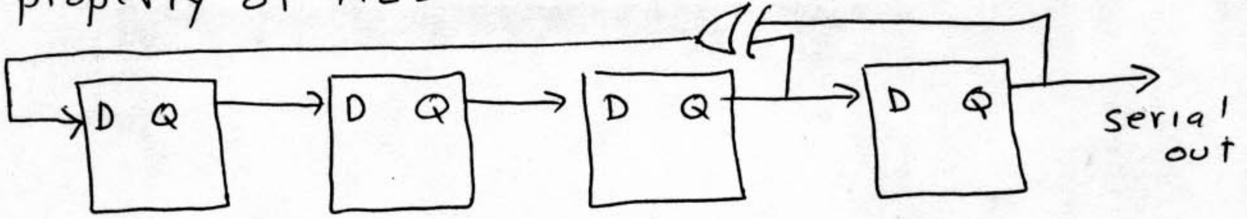
Tables exist which specify what stages can be XOR'd to produce a MLS for different # stages, e.g.

(tables not unique)

| N | stages XOR'd |
|---|---|
| 3 | 2, 3 |
| 4 | 3, 4 |
| 5 | 3, 5 |
| 8 | 4, 5, 6, 8 |
| 10 | 7, 10 |

etc

Special property of MLS



common clock, parallel load

say start with 1011 then seq is

$$1011 \rightarrow 0101 \rightarrow 1010 \rightarrow 1101 \rightarrow 1110 \rightarrow 1111 \rightarrow 0111 \rightarrow \text{etc}$$

serial out bits

serial out sequence will be

$$1\,1\,0\,1\,0\,1\,1\,1\,1\,0\,0\,0\,1\,0\,0 \ldots \left\{ \begin{array}{l} 8 \text{ one's} \\ 7 \text{ zero's} \end{array} \right.$$

There are (essentially) as many one's as zero's in the complete cycle. (The all zero state not included). An observer without knowledge of the circuit would not know whether the next serial out bit will be a 1 or a 0, equal probability. The observer sees a "random bit stream".

The sequence is actually deterministic (given the circuit you know exactly what to expect.)

Hence this is a "pseudo-random noise generator"