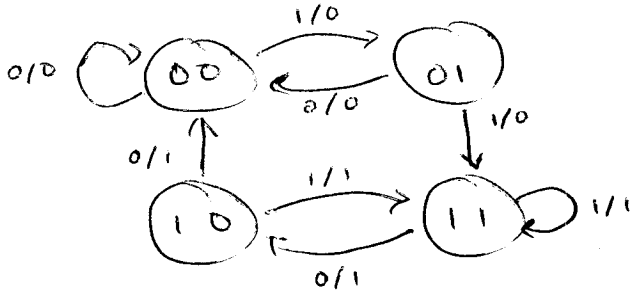
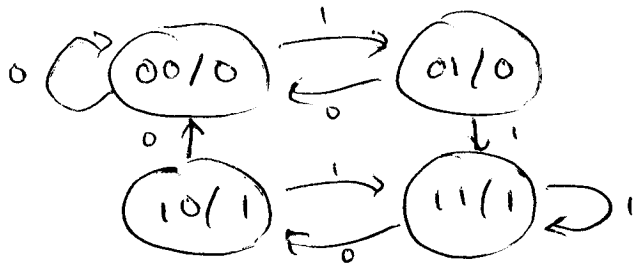
1. Implement the following state diagram, where the output will be the first bit of the two-bit state value.



notice:
- there are four states, enumerated 00 to 11 (0 to 3)
- there is one input, 0 or 1
- so there are eight transitions, two per state (one for input 0, and one for input 1)
- the output can be attached to the transitions, but more simply in this case to the states

redrawing the state diagram with the outputs attached to the states:



Let the state be represented as AB, where A and B each represent the current value of a flip-flop.

Let the input be represented as I. (The output is merely A.)

The truth table is therefore:

A(t)	B(t)	I	A(t+1)	B(t+1)	Out
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	0	0	0
0	1	1	1	1	0
1	0	0	0	0	1
1	0	1	1	1	1
1	1	0	1	0	1
1	1	1	1	1	1

If we design with D flip-flops, the inputs to the flip-flops are merely A(t+1) and B(t+1) - rather than needed extra J and K columns for each flip-flop.

So we use K-maps to find the simpliest logic expressions for A(t+1) and B(t+1). Let's do one for A(t+1):

A(t+1) \ B(t)*I	00	01	11	10
0	0	0	1	0
1	0	1	1	1

There are three pairs of 1's we can draw to cover the four 1's. (Remember that it is okay to have an individual 1 in multiple groups.)

A(t+1) \ B(t)*I	00	01	11	10
0	0	0	1	0
1	0	1	1	1

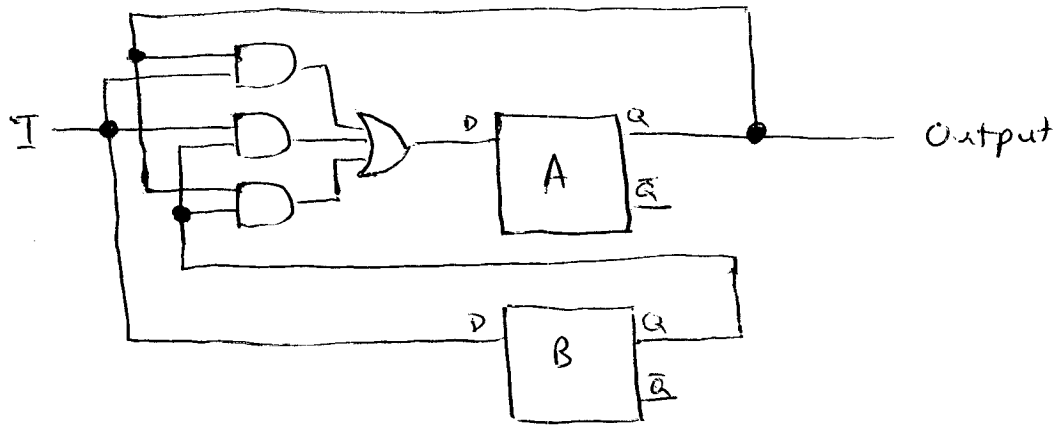
so A(t+1) = A(t)*I // middle bottom pair
+ B(t)*I // pair in the 11 column
+ A(t)*B(t) // rightmost bottom pair

Next is the B(t+1) K-map:

B(t+1) \ B(t)*I	00	01	11	10
0	0	1	1	0
1	0	1	1	0

There's a group of four 1's in the middle, so B(t+1) = I.

Then the circuit is given below (with two D flip-flops, three 2-input AND gates, and one 3-input OR gate).



A program implementing the state machine is

```
#include<stdio.h>

#define START 0
#define STOP -1

int main(){

  int state, next_input;
  int transition_table[4][2] = { {0,1}, {0,3}, {0,3}, {2,3} };
  int output_table[4] = {0,0,1,1};

  state = START;

  scanf( "%d", &next_input );
  while( next_input != STOP ){

    printf( "in state %d with output of %d,",      // output
            state, output_table[state] );
    printf( " input of %d causes transition to state %d\n",
            next_input, transition_table[state][next_input] );

    state = transition_table[state][next_input];  // update state
    scanf( "%d", &next_input );                   // next input

  }
}
```

When run with the input 0 1 0 1 1 0 1 1 1 0 1 0 0 1 1 -1, the program produces:

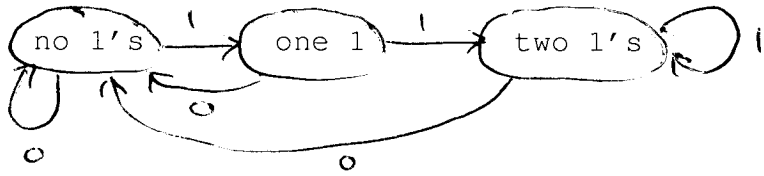
```
in state 0 with output of 0, input of 0 causes transition to state 0
in state 0 with output of 0, input of 1 causes transition to state 1
in state 1 with output of 0, input of 0 causes transition to state 0
in state 0 with output of 0, input of 1 causes transition to state 1
in state 1 with output of 0, input of 1 causes transition to state 3
in state 3 with output of 1, input of 0 causes transition to state 2
in state 2 with output of 1, input of 1 causes transition to state 3
in state 3 with output of 1, input of 1 causes transition to state 3
in state 3 with output of 1, input of 1 causes transition to state 3
in state 3 with output of 1, input of 0 causes transition to state 2
in state 2 with output of 1, input of 1 causes transition to state 3
in state 3 with output of 1, input of 0 causes transition to state 2
in state 2 with output of 1, input of 0 causes transition to state 0
in state 0 with output of 0, input of 1 causes transition to state 1
in state 1 with output of 0, input of 1 causes transition to state 3
```

2. Consider a state machine that outputs a 1 whenever it recognizes three 1's in a row in the input.

That is, the state machine behaves like this

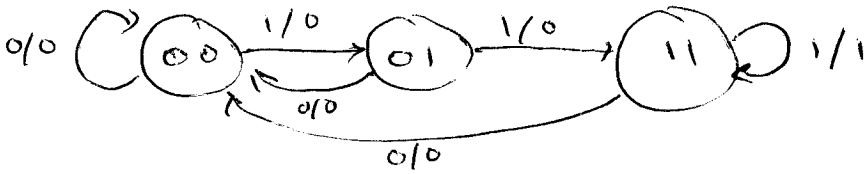
```
input:   0 1 0 1 1 0 1 1 1 0 1 1 1 1 1 0
output:  0 0 0 0 0 0 0 0 1 0 0 0 1 1 1 0
```

The state diagram can be formulated this way:



Output is 1 whenever there is an input of 1 while in state "two 1's".

Assigning bit values to the states:



Note that the 10 state is not needed. (We assume that we can add a "reset" signal to the circuit in order to start it off in state 00.)

Let the state be represented as AB as before, and let "d" represent "don't care". The truth table is therefore:

A(t)	B(t)	I	A(t+1)	B(t+1)	Out
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	0	0	0
0	1	1	1	1	0
1	0	0	d	d	0
1	0	1	d	d	0
1	1	0	0	0	0
1	1	1	1	1	1

The K-maps are as follows

A(t+1) \ B(t)*I

A(t) \	00	01	11	10
0	0	0	1	0
1	d	d	1	0

so A(t+1) = B(t)*I

(we don't use either "d")

B(t+1) \ B(t)*I

A(t) \	00	01	11	10
0	0	1	1	0
1	d	d	1	0

so A(t+1) = I

(we use only one "d")

The circuit diagram is easily obtained.