

Figure 54: Asynchronous backwards counter

## Synchronous Counters

In a synchronous counter all flip-flops receive the same clock pulse so changes at the outputs appear simultaneously. As in synchronous counters flip-flops are connected serially while being triggered simultaneously they are usually built using Master-Slave flip-flops. Figure 55 shows a synchronous upwards counter that makes use of the fact that any digit of a dual number only toggles in that moment, when all less significant digits are 1 (compare in Table 30 the steps from 3 to 4, 7 to 8 and 15 to 0). Toggling an output is done by setting both inputs of a JK flip-flop to 1, so the upward counting can be done by doing an AND for all less significant bits and feeding the result to both input of the actual digit's flip-flop:

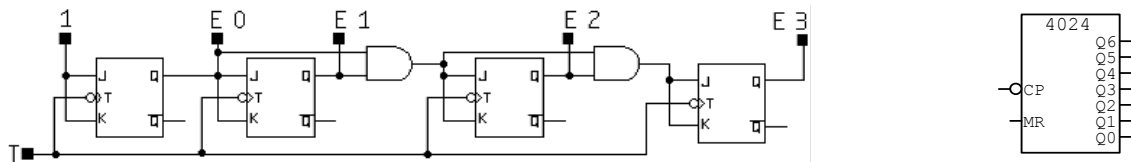


Figure 55: Synchronous upwards counter a) circuit b) symbol for Counter IC 4024

As counters are needed all the time they are available as ready-to-use Integrated Circuits. Figure 55.b shows the IC 4024, a 7 bit counter. Apart from the clock input CP it offers a Master Reset pin to reset the counter at any time. The synchronous backward counting is done the same way as with the asynchronous counter: by using the inverted outputs:

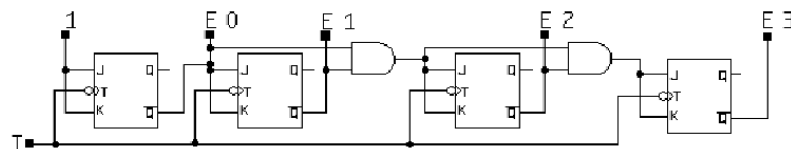


Figure 56: Synchronous backward counter

## Counting any Sequence - Counter Design

In some applications it might be necessary to have a counter counting something different from just up or down dual numbers. Let's say we want a counter that counts a circular sequence 0 – 4 – 7 – 2 – 3 – 0. The sequence contains 5 different numbers so we need  $n = \lceil \log_2 5 \rceil + 1 = 3$  Flip-Flops. As the highest number is 7, which can be represented with 3 bit, we can interpret the Flip-Flop's outputs  $E_0$  to  $E_2$  as dual numbers. The state table in Table 31 lists the desired next states for all possible states of this 3 flip-flop circuit.

Actual State				Next State			
Number	$E_2$	$E_1$	$E_0$	$E_2^+$	$E_1^+$	$E_0^+$	Number <sup>+</sup>
0	0	0	0	1	0	0	4
1	0	0	1	x	x	x	x
2	0	1	0	0	1	1	3
3	0	1	1	0	0	0	0
4	1	0	0	1	1	1	7
5	1	0	1	x	x	x	x
6	1	1	0	x	x	x	x
7	1	1	1	0	1	0	2

Table 31: State table for a sequence counter

Note that some of the state do not have a succeeding state and are treated as “don’t care”. From the state table we read the KV Maps to determine the simplified transitions functions  $E_i^+ = f(E_1, E_2, E_3)$  for all outputs.

KV Map for  $E_0^+$ KV Map for  $E_1^+$ KV Map for  $E_2^+$ 

$E_2$				$E_1$
x	0	0	0	
1	x	x	1	
$E_0$				

$E_2$				$E_1$
x	1	0	1	
1	x	x	1	
$E_0$				

$E_2$				$E_1$
x	0	0	0	
1	x	x	0	
$E_0$				

$$E_0^+ = E_1 \bar{E}_0 + E_2 \bar{E}_1$$

$$E_1^+ = E_2 + \bar{E}_1 + \bar{E}_0$$

$$E_2^+ = \bar{E}_1$$

These equations tell us, how the next state of every single flip-flop is determined by the actual state of all three flip-flops. In the last step we compare these equations to the flip-flop’s characteristic equation to find out, how the outputs have to be fed back to the inputs. In this example JK flip-flops are used:

Characteristic Equation for JK FF :  $Q^+ = J\bar{Q} + \bar{K}Q$

$$E_0^+ = E_1 \bar{E}_0 + E_2 \bar{E}_1 = (E_1 + E_2 \bar{E}_1) \bar{E}_0 + E_2 \bar{E}_1 E_0 = J_{E_0} \bar{E}_0 + \bar{K}_{E_0} E_0$$

$$J_{E_0} = E_1 + E_2 \bar{E}_1 = E_1 + E_2$$

$$K_{E_0} = \overline{E_2 \bar{E}_1} = E_1 + \bar{E}_2$$

$$E_1^+ = E_2 + \bar{E}_1 + \bar{E}_0 = (E_2 + 1 + \bar{E}_0) \cdot \bar{E}_1 + (E_2 + \bar{E}_0)E_1 = J_{E_1} \bar{E}_1 + \bar{K}_{E_1} E_1$$

$$J_{E_1} = 1$$

$$K_{E_0} = \overline{E_2 + \bar{E}_0} = E_0 \bar{E}_2$$

$$E_2^+ = \bar{E}_1 = \bar{E}_1 \cdot \bar{E}_2 + \bar{E}_1 \cdot E_2 = J_{E_2} \bar{E}_2 + \bar{K}_{E_2} E_2$$

$$J_{E_2} = \overline{E}_1$$

$$K_{E_2} = E_1$$

The actual counter is shown in

Figure 57.

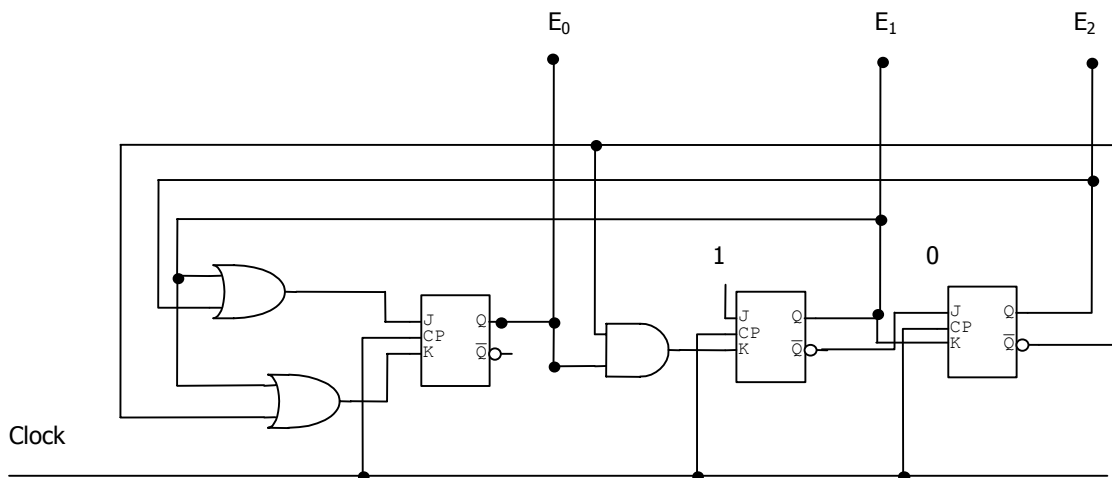


Figure 57: Synchronous counter for the sequence 0-4-7-2-3

## Registers

With flip-flop we can store data bitwise but usually data does not appear as single bits. Instead it is common to store data words of  $n$  bit with typical word lengths of 4, 8, 16, 32 or 64 bit. Thus, several flip-flops are combined to form a *register* to store whole data words. Registers are synchronous circuits thus all flip-flops are controlled by a common clock line. As registers are often use to collect serial data they are also called accumulators. There exist several types of registers as there are several ways of receiving and sending data.

## Shift Registers

Information often comes bitwise i.e. one bit every clock pulse. To store such data *shift registers* are used. A shift register has one input. Every clock pulse one bit is loaded into the first flip-flop of the register while all the actual flip-flop contents are shifted and the “oldest” bit got dropped. A circuit as in Figure 58 works as a *Serial In – Serial Out* or *Serial-In – Parallel Out* register: data input is done with line *D*. If the