SAPIENZA
UNIVERSITÀ DI ROMA
DIPARTIMENTO DI INFORMATICA

Synthesis of sequential nets

Prof. Daniele Gorla

The aim

SAPIENZA
UNIVERSITÀ DI ROMA
DIPARTIMENTO DI INFORMATICA

Like for combinatorial circuits, the synthesis procedure aims at creating digital circuits starting from an abstract specification.

	Combinatorial	**Sequential**
Formal specification	TT	Automaton
Intermediate Representation	BEs	Future states table & excitation function EB
Final Circuit	Logical gates + (acyclic) interconnections	Logical gates + (even cyclic) interconnections + FF

Synthesis procedure through an example

SAPIENZA
UNIVERSITÀ DI ROMA
DIPARTIMENTO DI INFORMATICA

Design a circuit that, taken in input a sequence of bits, gives in output 1 if and only if the number of 1s received so far is a multiple of 3.

Step 1. From the verbal specification to the (minimal) automaton

In our example, a number is multiple of 3 if it equals $3 \cdot k$, where k is a natural number ($3 \cdot 0$, $3 \cdot 1$, $3 \cdot 2$, $3 \cdot 3$, $3 \cdot 4$, …). If n is the number of 1s received so far, the desired automaton must check whether:

$n = 3 \cdot k$ $\quad$ $n = 3 \cdot k + 1$ $\quad$ $n = 3 \cdot k + 2$

And only in the first case produce in output 1.

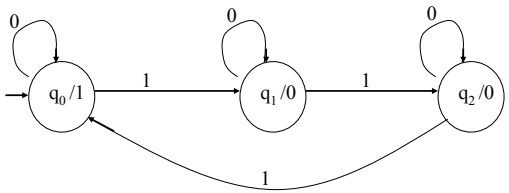
Let's asociate the first condition to q_0, the second condition to q_1 and the third condition to q_2. The initial state is q_0.

A transition from q_i to $q_{(i+1) \mathrm{MOD}\ 3}$ happens every time a 1 arrives; by contrast, with 0 we remain in the current state.

q_0 is the only state that produces 1 (Moore model).

1. Automaton of the example

SAPIENZA
UNIVERSITÀ DI ROMA
DIPARTIMENTO DI INFORMATICA



Or, equivalently (it will be useful soon):

	0	1
→ q_0	q_0 / 1	q_1 / 0
q_1	q_1 / 0	q_2 / 0
q_2	q_2 / 0	q_0 / 1

It is easy to check that the automaton is minimal (all states are distinguishable)

2. Binary encoding of the automaton

SAPIENZA Università di Roma Dipartimento di Informatica

We have to represent the 3 sets Q, Σ and Δ in binary.

REMARK: a set with n elements requires $\lceil \log_2 n \rceil$ bits to be represented.

- every bit needed to represent the state is stored in a FF (it is the output y of the FF);
- every bit needed to represent an input character is an input of the circuit (it is one of the input variables x);
- every bit needed to represent an output character is an output of the circuit (it is one of the output variables z).

In our example, the input and output alphabeths are already coded in binary (they are exactly the set {0,1}).

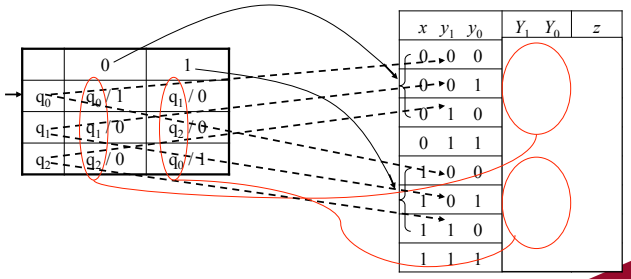
Let's codify state q_0 as $y_1 y_0 = 00$ in the FFs, q_1 with $y_1 y_0 = 01$ and q_2 with $y_1 y_0 = 10$ (the combination $y_1 y_0 = 11$ is not used).

3. Future states table

SAPIENZA Università di Roma Dipartimento di Informatica

We now turn the automaton to a table (that we call *future states*) where we provide the next state (i.e., that at time t+1) and output (at time t) as a function of the current state and of the input (both at time t).

This can be very easily done by starting from the tabular representation of the automaton (also from its drawing, of course…)



4. Excitation functions of the FFs

SAPIENZA Università di Roma Dipartimento di Informatica

For every input of every FF, we add a new column to the future states table.

Every such a column is filled by relying on the excitation functions of the FFs, according to the current state (y) and the future one (Y).

The kind of the FFs can be either specified in the project or we can choose the one that generates the simplest final circuit.

x	y_1	y_0	Y_1	Y_0	z	s_1	r_1	s_0	r_0	j_1	k_1	j_0	k_0	d_1	d_0	t_1	t_0
0	0	0	0	0	1												
0	0	1	0	1	0												
0	1	0	1	0	0												
0	1	1	-	-	-												
1	0	0	0	1	0												
1	0	1	1	0	0												
1	1	0	0	0	1												
1	1	1	-	-	-												

5. Minimal BEs

SAPIENZA Università di Roma Dipartimento di Informatica

From the resulting table, we have to derive the (minimal) BEs of the inputs of the FFs and for the outputs of the circuit.

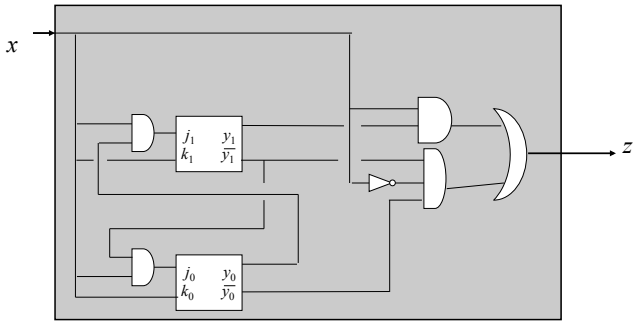
REMARK: for future states (Y) you don't have to calculate a BE since they are automatically given by the FFs according to the current state (stored in the FFs) and their inputs (for which we calculate the BE).

According to the minimal BEs, we shall choose the kind of the FF for every stored bit (it is not mandatory that the FFs are all of the same kind!), if this was not specified by the specification.

$$z = \bar{x}\,\bar{y}_1\bar{y}_0 + xy_1$$

$$\underbrace{\begin{aligned} s_1 &= xy_0 \\ r_1 &= xy_1 \\ s_0 &= x\bar{y}_1\bar{y}_0 \\ r_0 &= xy_0 \end{aligned}}_{\text{4 gates}} \qquad \underbrace{\begin{aligned} j_1 &= xy_0 \\ k_1 &= x \\ j_0 &= x\bar{y}_1 \\ k_0 &= x \end{aligned}}_{\text{2 gates}} \qquad \underbrace{\begin{aligned} d_1 &= \bar{x}y_1 + xy_0 \\ d_0 &= \bar{x}y_0 + x\bar{y}_1\bar{y}_0 \end{aligned}}_{\text{8 gates}} \qquad \underbrace{\begin{aligned} t_1 &= x(y_1 + y_0) \\ t_0 &= x\bar{y}_1 \end{aligned}}_{\text{3 gates}}$$

6. Circuit schema (optional)



Encodings and Optimizations

OBS.: At step 2, we encoded in binary states and alphabeths; such encodings have an influence on the final size of the circuit!!

Ex.: For states, we used the encoding $q_0 \rightarrow 00$, $q_1 \rightarrow 01$, $q_2 \rightarrow 10$.

Let's see what happens with the encoding $q_0 \rightarrow 11$, $q_1 \rightarrow 01$, $q_2 \rightarrow 10$.
For simplicity, let's ignore the output (it is not touched by the encoding of states) and let's consider D FFs only:

$$\underbrace{\begin{aligned} d_1 &= \overline{y}_0 + \overline{x}y_1 + x\overline{y}_1 = \overline{y}_0 + (x \oplus y_1) \\ d_0 &= \overline{x}y_0 + xy_1 \end{aligned}}_{\text{6 gates}}$$

(8 gates with the previous encoding)

x	y_1	y_0	Y_1	Y_0	d_1	d_0
0	0	0	-	-	-	-
0	0	1	0	1	0	1
0	1	0	1	0	1	0
0	1	1	1	1	1	1
1	0	0	-	-	-	-
1	0	1	1	0	1	0
1	1	0	1	1	1	1
1	1	1	0	1	0	1

Optimal sequential net

To obtain the optimal sequential net (i.e., the one that has the smallest number of logical gates) we have to use:

1. The minimal automaton;
2. Consider all the possible binary encodings of states, inputs and outputs;
3. For every possible encoding in point 2, consider all possible combinations by using any of the 4 kinds of FFs.

Clearly, this is unfeasible also for very small automata:

In our example, we have 24 possible encodings of the states and 4 kinds of FFs: so, 96 columns for the FFs inputs; with 2 FF, 2 with 2 inputs and 2 with 1 input, we would obtain 96×12 = 1152 EB!!!

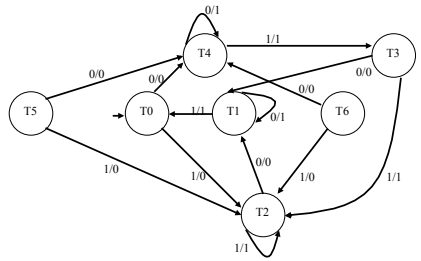
Never done in practice!!

→ With nowadays costs and crossing times, having the really minimal circuit is not really needed

A second example (1)

Implement the sequential net associated to the following automaton by using SR flip-flops.

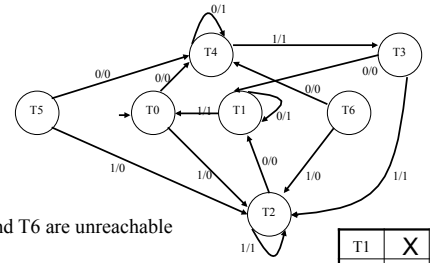
Then, show the temporal diagram with input 1100101.



A second example (2)

SAPIENZA Università di Roma Dipartimento di Informatica

Step 1: the automaton is already given; we only need to check its minimality.



OBS: T5 and T6 are unreachable

T1	X			
T2	X	X		
T3	X	X	O	
T4	X	X	X	X
	T0	T1	T2	T3

Minimal Automaton:

	0	1
S0 (T0)	S3/0	S2/0
S1 (T1)	S1/1	S0/1
S2 (T2+T3)	S1/0	S2/1
S3 (T4)	S3/1	S2/1

A second example (3)

SAPIENZA Università di Roma Dipartimento di Informatica

	0	1
S0 (T0)	S3/0	S2/0
S1 (T1)	S1/1	S0/1
S2 (T2+T3)	S1/0	S2/1
S3 (T4)	S3/1	S2/1

Let's encode the states as follows:

S0 → 00 , S1 → 01 , S2 → 10 , S3 → 11

With such an encoding, the future states table is:

By looking at the values of the state at times t and t+1, let's derive the BEs for the FFs inputs, by using the SR excitation function:

x	y1	y0	Y1	Y0	z	s1	r1	s0	r0
0	0	0	1	1	0	1	0	1	0
0	0	1	0	1	1	0	-	-	0
0	1	0	0	1	0	0	1	1	0
0	1	1	1	1	1	-	0	-	0
1	0	0	1	0	0	1	0	0	-
1	0	1	0	0	1	0	-	0	1
1	1	0	1	0	1	-	0	0	-
1	1	1	1	0	1	-	0	0	1

A second example (4)

SAPIENZA Università di Roma Dipartimento di Informatica

x	y1	y0	z	s1	r1	s0	r0
0	0	0	0	1	0	1	0
0	0	1	1	0	-	-	0
0	1	0	0	0	1	1	0
0	1	1	1	-	0	-	0
1	0	0	0	1	0	0	-
1	0	1	1	0	-	0	1
1	1	0	1	-	0	0	-
1	1	1	1	-	0	0	1

We can finally derive the minimal BEs for z, s1, r1, s0 ed r0:

$$z = y_0 + xy_1$$

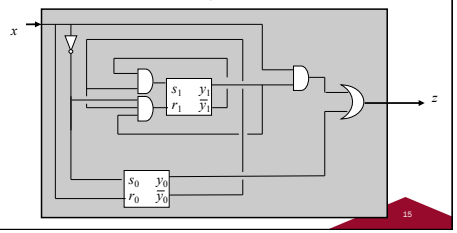
$$s_1 = \bar{y}_1\bar{y}_0$$

$$r_1 = \bar{x}y_1\bar{y}_0$$

$$s_0 = \bar{x}$$

$$r_0 = x$$

From these BEs, we have the circuit:



A second example (4)

SAPIENZA Università di Roma Dipartimento di Informatica

	0	1
S0 (00)	S3/0	S2/0
S1 (01)	S1/1	S0/1
S2 (10)	S1/0	S2/1
S3 (11)	S3/1	S2/1

Temporal diagram for input 1100101: