WEEK 3

Last week we learned how to design circuits using logical ports. To do so, on the basis of the description of the circuit we constructed the truth table, we identified the minterms, we used them to obtain a Boolean expression in the form of the sum of products, we simplified it and finally we drew the circuit.

In this third week, we are going to continue working on the optimisation of the circuit.

- We will find out what don’t care terms are and how, if they are well identified, they can be used to obtain even simpler circuits,
- The minimisation of Boolean functions in a relatively high number of variables tends to be a tedious process where it is easy to make mistakes if done manually. In lecture 3.1, we will study some of the principles involved in the automatic optimisation of circuits and will learn about what are known as “automatic synthesis tools”.
- We will introduce a new parameter, propagation time, and will be finding out that the designer often has to reach a compromise between the number of ports and the propagation time.
- And we shall continue to introduce more logical elements; in this case multiplexers, decoders, LUTs and AND-OR planes.
- Finally, lecture L3.4 introduces a very interesting subject, the relationship between certain program structures and digital circuits. We will see how some classic program structures such as *if..then..else* and *loops*, as well as so-called *procedures*, can be implemented directly with combinational circuits. This relation between program structures and circuits justifies the existence of high level hardware description languages, and the existence of synthesis tools.

As for the example-guide, this week we will be completing the processor specifications.

CONTENTS

min (aprox)

L3.1 Combinational systems synthesis tools

Design of combinational circuits: Practical questions 00:20

1. Redundant terms (don’t care) 00:28

2. Synthesis tools

Cube representation 06:23

Adjacency 09:51

Example of a synthesis tool 12:52

Summary 20:56

L3.2	Propagation time	
	1. Propagation time	00:24
	2. Example: n-bit comparator	04:45
	2.1 Component: 1 bit comparator module	07:31
	2.2 Using a 2 bit comparator module	10:13
	Summary	14:20
L3.3	Other logic blocks	
	1. Multiplexers	00:18
	Multiplexers (Boolean functions synthesis)	04:46
	2. Multiplexers and memory blocks (Look Up Tables)	10:25
	3. Planes	14:44
	4. Address decoders and 3-state buffers	17:47
	Summary	23:20
L3.4	Programming language structures	
	1. If ... then … else …	01:08
	2. Case … is	04:18
	3. For … loop …	05:24
	4. Procedure call	09:10
	5. Comments	12:35
	Summary	13:39
P3.1	Structural specification (cont.)	
	2. Block description (cont.)	
	2.3 Register bank	00:35
	2.4 Computation resources	05:08
	2.5 GO TO	06:26
	Summary	08:47