ASICS1 Project (2015) 25%

Project Description.

A lottery checker system is modelled to the block diagram (Figure 1) and pin description tables (Table 1 and Table 2).

The winning numbers (from 1 to 31) are presented through inputs **W1**,**W2**,**W3** and **W4**. The numbers to be checked come from an external scanner and are presented through a synchronisation input (**scan**) and a data input (**N_in**). Up to 6 sets of numbers can be read at a single scan. There are four control/information outputs (**SYSRDY**, **winner**, **not_a_win**, and **RD_ERR**), and the monetary value of the win is displayed in BCD form through outputs **Eur100**, **Eur010** and **Eur001**.

Matching any 1 number wins €1, any 2 numbers wins €5, any 3 numbers wins €25, and all 4 numbers wins €125. The maximum possible winning total is €750.

Figure 1: Block Diagram

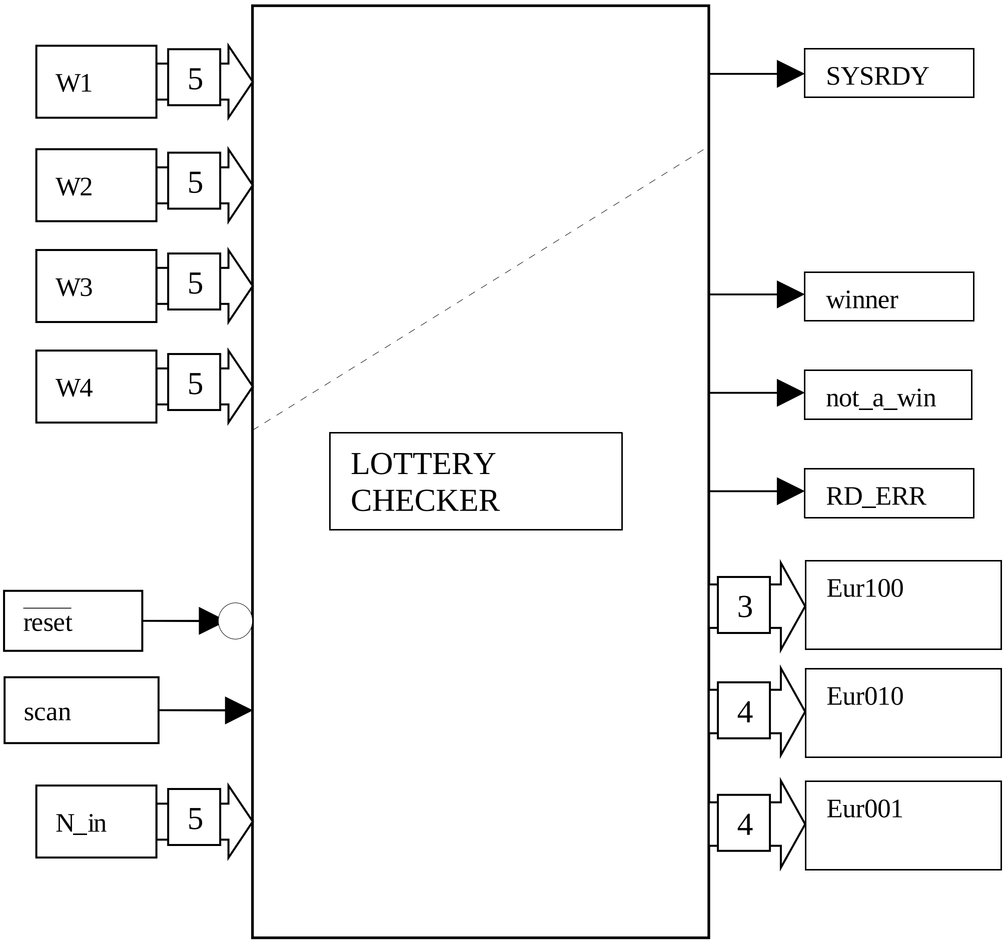


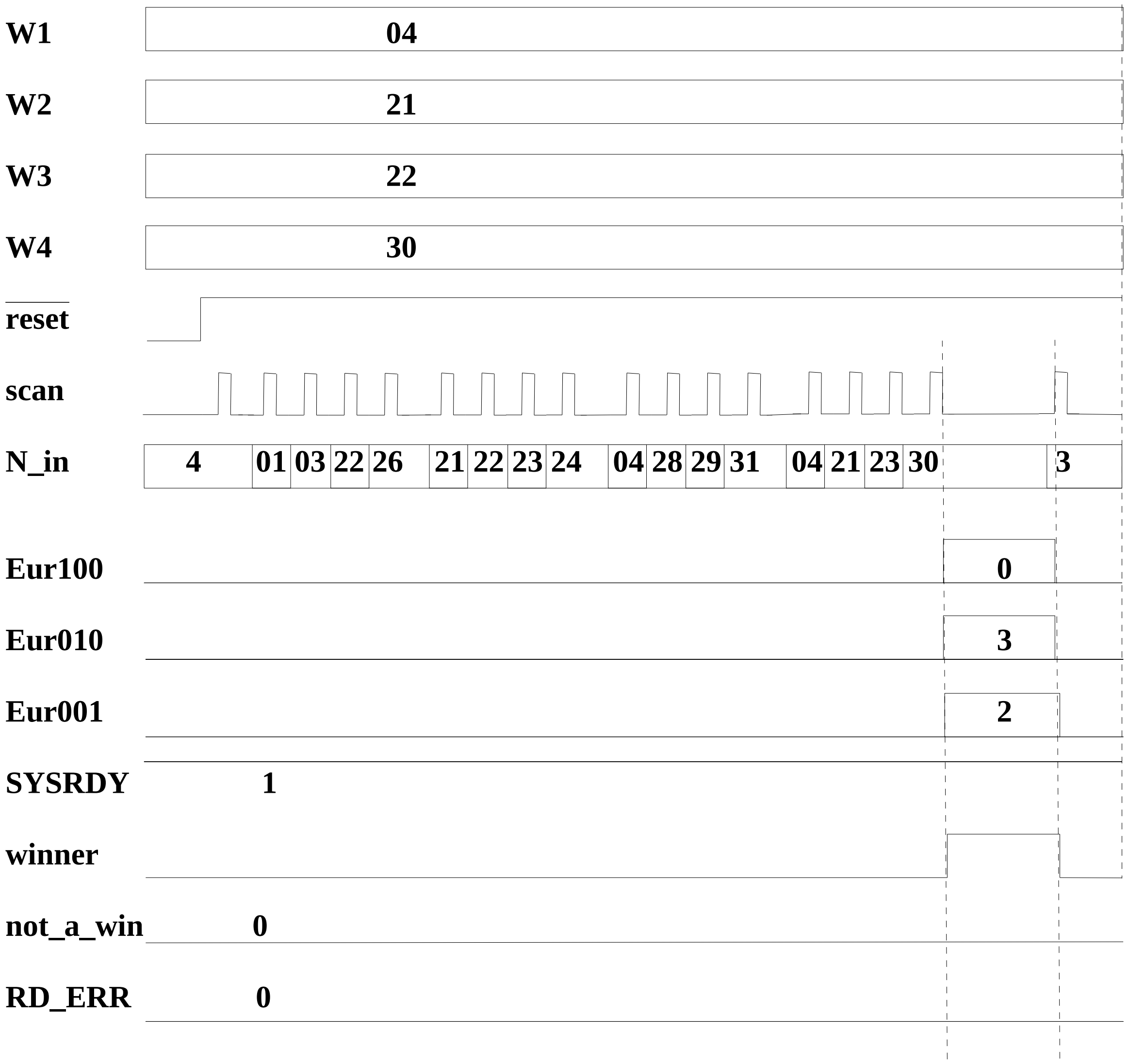
Table 1: Inputs Pin Description

PIN	Input / Output	Description
reset	Input	Active low **reset** pulse, to reset the system. It should set the outputs **winner**, **not_a_win**, and **RD_ERR** low, and the winning cash value (as defined by **Eur100, Eur010** and **Eur001** outputs) to 000. It should also reset the system internally so that the **scan** input is waiting for "number of lines" input (see **N_in** description below), and should reset all appropriate internal registers. For normal operation, this input pin should be 1.
scan	Input	An active high scan valid input (if **reset** is low). If **scan** is high the data on input **N_in** is read into the system.
N_in	Input (5-bit)	A 5-bit data input. This should use either the positive or negative edge of **scan** to load valid data (**N_in**) to the system. The first data read defines the "number of lines" which is the number of sets of numbers to be read. Valid values are 1 to 6. If a 0 or a number greater than 6 is read, then **RD_ERR** should be set high and nothing further should happen, until the system is reset by **reset**. If a valid number is read, **winner**, **not_a_win**, and **RD_ERR** outputs should be set to 0. The next 4 to 24 numbers (as defined by "number of lines") should read a lottery number between 1 and 31. The numbers are read in sequences of 4. If any of these numbers is 0, or if any two of the sequence of 4 are the same, then **RD_ERR** should be set high and nothing further should happen, until the system is reset by **reset**. When the final number is read, the system should check the numbers. If there is a winning combination, the **winner** output should go high and the winning cash value should appear on the **Eur100, Eur010** and **Eur001** outputs. If there is no winning combination, the **not_a_win** output should go high, and 000 should appear on the **Eur100, Eur010** and **Eur001** outputs.
W1	Input (5-bit)	A hard-wired winning number (provided by the lottery organiser). Valid values are 1 to 31. If this number is 0, the **SYSRDY** output should be forced to a 0.
W2	Input (5-bit)	As the **W1** input description.
W3	Input (5-bit)	As the **W1** input description.
W4	Input (5-bit)	As the **W1** input description.

Table 2: Output Pin Description

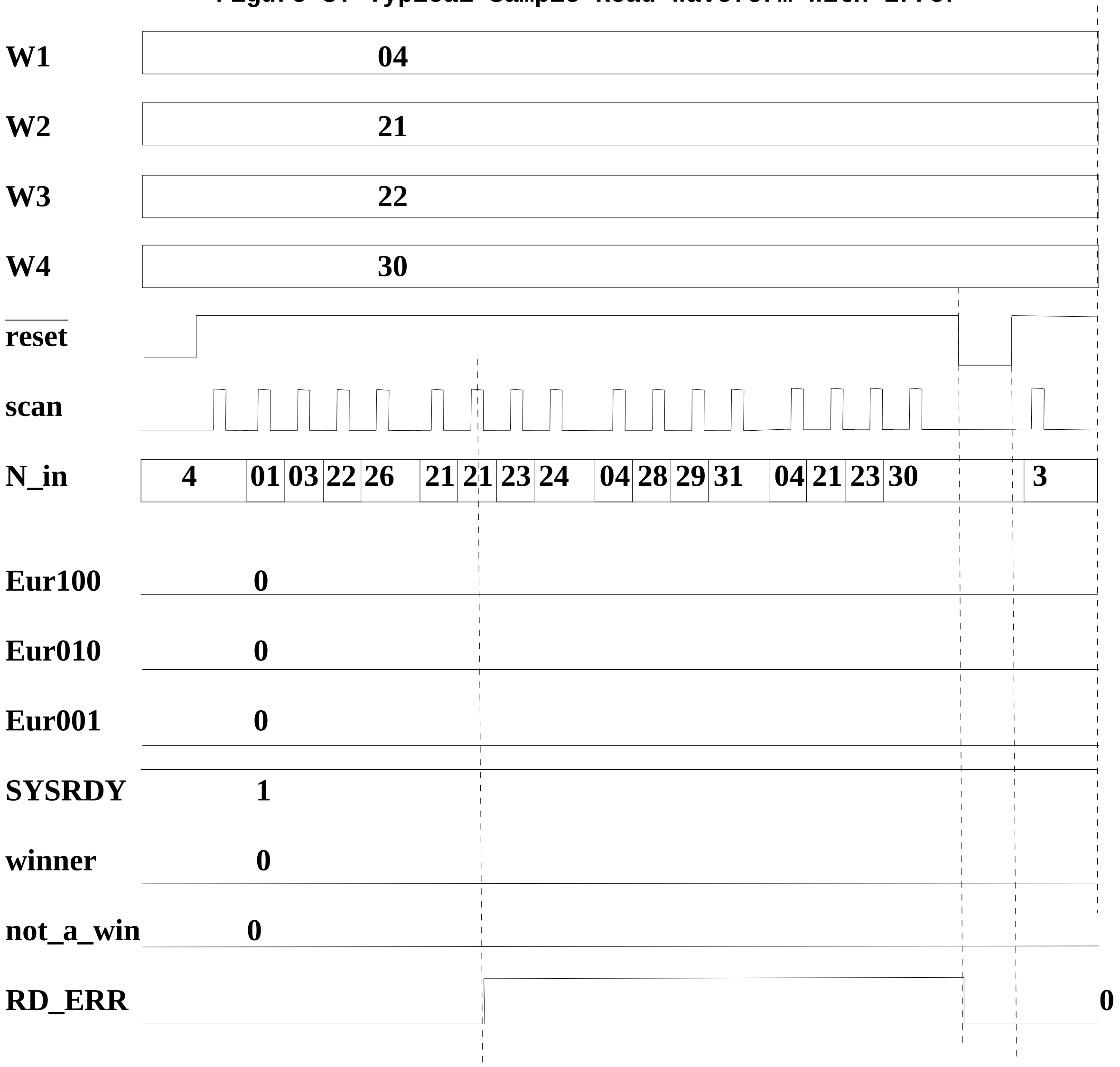
PIN	Input / Output	Description
SYSRDY	Output	The **SYSRDY** (System Ready) will go high if there is a valid input on pins **W1**,**W2**,**W3** and **W4.** If any of the input pins **W1**,**W2**,**W3** and **W4** are 0, then **SYSRDY** will go low. If **SYSRDY** is low, nothing further should happen, until there are valid values on inputs **W1**,**W2**,**W3** and **W4**.
winner	Output	This will go high after a valid read cycle if there is a match of 1, 2, 3 or 4 between the scanned sets of 4 numbers and the winning numbers (**W1**,**W2**,**W3** and **W4**), and **reset** is high. This will remain high, until **reset** goes low, or until the next read cycle is started, which will bring **winner** back low.
not_a_win	Output	This will go high after a valid read cycle if there is no match of 1, 2, 3 or 4 between the scanned sets of 4 numbers and the winning numbers (**W1**,**W2**,**W3** and **W4**), and **reset** is high. This will remain high, until **reset** goes low, or until the next read cycle is started, which will bring **not_a_win** back low.
RD_ERR	Output	This will go high if an invalid value is read through the **N_in** input. (See "Input Pin Description"). This will then remain high, until **reset** goes low which will bring **RD_ERR** back low.
Eur100	Output (3-bit)	If output **winner** is 1, this is the winning value in multiples of €100. Valid values are 0 to 7.
Eur010	Output (4-bit)	If output **winner** is 1, this is the winning value in multiples of €10. Valid values are 0 to 9.
Eur001	Output (4-bit)	If output **winner** is 1, this is the winning value in multiples of €1. Valid values are 0 to 9.

Figure 2: Typical Sample Valid Read Waveform



Input Numbers to be checked	Numbers Matched	Winning Outcome
01 03 **22** 26	1	€ 1
21 **22** 23 24	2	€ 5
04 28 29 31	1	€ 1
04 **21** 23 **30**	3	€ 25
		€ 32

Figure 3: Typical Sample Read Waveform with Error



Design the Verilog Behavioural Code for the above system.

```
module lottery  (..............);
.
.
.
.
.
.
endmodule
```

Complete the following test stimulus to test the full functionality of the circuit.

```
module test;
.
.
.
.
lottery i1 (................);
.
.
.
.
endmodule
```

Simulate and Synthesise the design.

Work to be done.

Task	Marks
Write a behavioural VERILOG description of the lottery checker. Compile without Errors.	10
Simulate the full functionality of the design with an appropriate set of test vectors.	9
Synthesize the design.	5

Project Deadline.

End of Week 13. (Friday 4 December 2015).