Logical Networks Final Project

Prof. Fornaciari, Prof. Palermo and Prof.
Salice Academic Year 2022 - 2023

(UPDATED 15 December 2022)

General description

The specification of the "*Final Proof (Logical Networks Project)*" for the Academic Year 2022/2023 calls for the implementation of a HW module (described in VHDL) that interfaces with a memory and complies with the following specification.

At a high level of abstraction, the system receives indications about a memory location, the contents of which must be directed to one of the four available output channels.
The indications as to which channel to use and which memory address to access are provided via a one-bit serial input, while the system outputs, i.e. the aforementioned channels, provide all the bits of the memory word in parallel.

Interfaces

The module to be implemented has **two 1-bit primary inputs** (**W** and **START**) and **five primary outputs**. The outputs are as follows: four 8-bit (**Z0, Z1, Z2, Z3**) and one 1-bit (**DONE**). In addition, the module has a **CLK** clock signal, unique for the entire system, and a **RESET** reset signal, also unique.

Operation

At the initial instant, the instant of system **reset**, the outputs have the following values:

Z0, Z1, Z2 and Z3 are 0000 0000, DONE is 0.

The input data, obtained as sequences on the **primary serial input W** read on the rising edge of the clock, are organised as follows:

- 2 header bits (the first in the sequence) followed by
- N memory address bits.

N bits make it possible to construct a memory address (read the specification for these N bits below). The memory address stores **the 8-bit message** that is to be addressed to an **output** channel.

The two **header** bits identify the **output channel** (Z0, Z1, Z2 or Z3) to which the message is to be addressed. The first bit is the most significant bit of the output channel, the second the least significant bit:

00 identifies Z0, 01 identifies Z1, 10 identifies Z2 and, finally, 11 identifies Z3.

The **N** address bits can vary from 0 to a **maximum of 16 bits**. Memory addresses are all 16 bits.

If the number of bits of N is less than 16, the address is **extended** with **0 on the most significant bits**. For example:

```
(N = 7)                      1010111-> 0000000001010111
(N = 16)1110000001010111   -> 1110000001010111
(N = 0)    0000000000000000 -> 0000000000000000
```

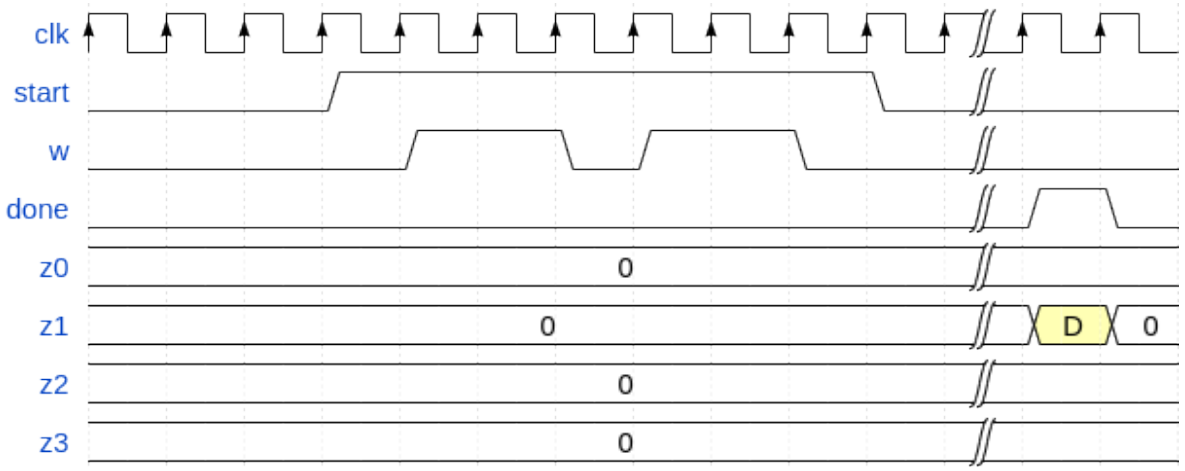
All bits on W must be read on the rising edge of the clock.

The input sequence is valid when the START signal is high (=1) and ends when the START signal is low (=0).
The START signal remains high for at least 2 clock cycles and no more than 18 clock cycles (2 channel bits and 16 bits for the maximum number of bits to address the memory). Assume this condition always occurs (it is not necessary to handle the case where the START signal remains high less than 2 clock cycles or more than 18).

The outputs Z0, Z1, Z2 and Z3 are initially 0. The values remain unchanged except for the channel on which the message read in memory is sent; the values are only visible when the value of DONE is 1.
When the DONE signal is 0 all channels Z0, Z1, Z2 and Z3 must be zero (32 bits to 0). At the same time as the message is written to the channel, the DONE signal changes from 0 to 1 and remains active for only one clock cycle (after 1 clock cycle DONE changes from 1 to 0). In practice, when DONE=1 the channel associated with the message will change its value, while the other channels will display the last transmitted value derived from the messages associated with them.
The START signal is guaranteed to remain at 0 until the DONE signal has returned to 0.
The maximum time to produce the result (i.e. the time elapsed between START=0 and DONE=1) must be less than 20 clock cycles.

The module must be designed considering that before the first START=1 (and before requesting correct module operation) a RESET (RESET=1) will *always* be given. A second (or subsequent) processing with START=1 will not have to wait for the module to be reset. Whenever the RESET signal (RESET=1) is given, the module is re-initialised.

Example (time diagrams)
Reading a datum 'D' from memory address 00000000010110.
Data 'D' is written to the specified output Z1 (header bit '01')



Component Interface

The component to be described must have the following interface.

```
entity
    project_networks_logical
    is port (
        i_clk  :  in  std_logic;
        i_rst  :  in  std_logic;
        i_start : in std_logic;
        i_w : in std_logic;

        o_z0  :  out  std_logic_vector(7  downto  0);
        o_z1  :  out  std_logic_vector(7  downto  0);
        o_z2  :  out  std_logic_vector(7  downto  0);
        o_z3  :  out  std_logic_vector(7  downto  0);
        o_done : out std_logic;

        o_mem_addr : out std_logic_vector(15 downto 0);
        i_mem_data : in std_logic_vector(7 downto 0);
        o_mem_we   : out std_logic;
        o_mem_en   : out std_logic
    );
```

In particular:

- the module name **must be** `project_reti_logic` and there must be **only one architecture** for each entity; violation of these instructions will result in the Test Bench not being able to be carried out and a subsequent zero rating;
- i_clk is the input CLOCK signal generated by the Test Bench;
- i_rst is the RESET signal that initialises the machine ready to receive the first START signal;

- i_start is the START signal generated by the Test Bench;
- i_w is the previously described W signal generated by the Test Bench;
- o_z0, o_z1, o_z2, o_z3 are the four output channels;
- o_done is the output signal communicating the end of processing;
- o_mem_addr is the output signal (vector) that sends the address to the memory;
- i_mem_data is the signal (vector) that arrives from memory following a read request;
- o_mem_en is the ENABLE signal that must be sent to the memory in order to communicate (both read and write);
- o_mem_we is the WRITE ENABLE signal that must be sent to memory (=1) in order to write to it. To read from memory it must be 0.

APPENDIX: Memory Description

NOTE: **Memory is already instantiated within the Test Bench and should not be synthesised**

The memory and its protocol can be extracted from the following VHDL description, which is part of the test bench and derived from the VIVADO user guide available at the following link: https://www.xilinx.com/support/documentation/sw_manuals/xilinx2017_3/ug901-vivado-synthesis.pdf

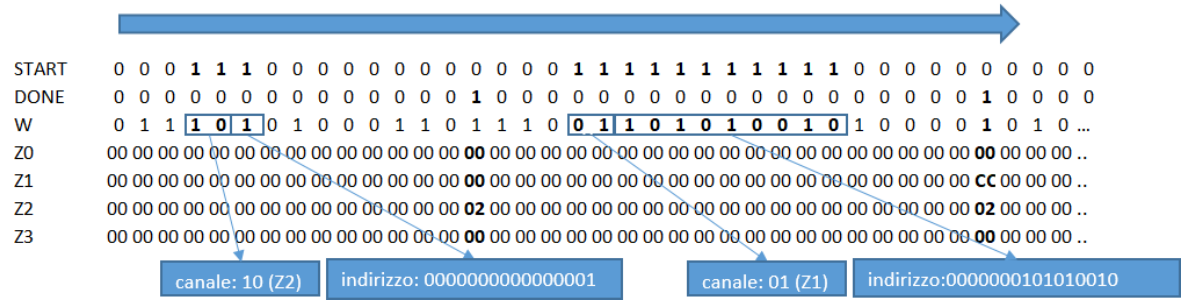
```
-- Single-Port Block RAM Write-First Mode (recommended template)
--
-- File: rams_02.vhd
--
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
entity rams_sp_wf is
port(
  clk  :  in  std_logic;
  we : in std_logic; en
  :  in std_logic;
  addr : in std_logic_vector(15 downto 0);
  di:  in std_logic_vector(7 downto 0); do
       : out std_logic_vector(7 downto 0)
);
end rams_sp_wf;

architecture syn of rams_sp_wf is
type ram_type is array (65535 downto 0) of std_logic_vector(7 downto 0);
signal RAM : ram_type;
begin
  process(clk)
    begin
    if clk'event and clk = '1' then
      if en = '1' then
        if we = '1' then
          RAM(conv_integer(addr)) <= di;
          do<=                    of after 2 ns;
        else
          do <= RAM(conv_integer(addr)) after 2 ns;
        end if;
      end if;
    end if;
  end process;
end syn;
```

EXAMPLES

The examples below are intended to exemplify the relationship between the useful input string (denoted as useful W), i.e. during the period in which START takes value 1, the data (DATE column) present in the requested memory location (ADDRESS column) and the configuration assumed by the outputs at the only useful time, i.e. during the period in which DONE takes value 1.

Example 1



		reset	00	00	00	00	[LSB . MBS | LSB MSB] . .
CHANNEL	ADDRESS	FACT	Z0	Z1	Z2	Z3	W useful
02	0001	**02**	00	00	02	00	**101**
01	0152	**CC**	00	CC	02	00	**01001010110**
02	045B	A1	00	CC	A1	00	**1101101000101**
00	0751	7E	7E	CC	A1	00	**1000101011100**
00	07B2	69	69	CC	A1	00	**0100110111100**
02	0A58	7B	69	CC	7B	00	**00011010010101**
00	0B1C	AC	AC	CC	7B	00	**00111000110100**
03	0DFC	EA	AC	CC	7B	EA	**00111111101111**
00	1025	03	03	CC	7B	EA	**101001000000100**
01	1335	D8	03	D8	7B	EA	**101011001100110**
02	1605	4F	03	D8	4F	EA	**101000000110101**

...	...	...	...	...	...	...	...

Example 2

		RESET	00	00	00	00	
CHANNEL	ADDRESS	FACT	Z0	Z1	Z2	Z3	W useful
03	EA60	04	00	00	00	04	0000011001010111111
03	EBC3	59	00	00	00	59	1100001111010111111
03	EBC9	66	00	00	00	66	1001001111010111111
02	ECEB	EE	00	00	EE	66	1101011100110111101
02	F063	2E	00	00	2E	66	1100011000001111101
03	F391	94	00	00	2E	94	1000100111001111111
03	F700	3A	00	00	2E	3A	0000000011101111111
01	FAA0	52	00	52	2E	3A	0000010101011111110
03	FE1D	14	00	52	2E	14	1011100001111111111
02	FB78	F4	00	52	F4	14	0001111011011111101
01	FD5F	43	00	43	F4	14	1111101010111111110
02	FEA1	1F	00	43	1F	14	1000010101111111101
03	FEB7	E6	00	43	1F	E6	1110110101111111111
03	FFA2	86	00	43	1F	86	0100010111111111111
03	FC4F	E4	00	43	1F	E4	1111001000111111111
02	FEBC	31	00	43	31	E4	0011110101111111101
...	...	...	...	...	...	...	...

Example 3

		RESET	00	00	00	00	
CHANNEL	ADDRESS	FACT	Z0	Z1	Z2	Z3	W useful
00	03E8	16	16	00	00	00	000101111100
03	2176	FROM	16	00	00	FROM	0110111010000111
02	3873	EB	16	00	EB	FROM	1100111000011101
02	495D	E3	16	00	E3	FROM	10111010100100101
01	6CDA	89	16	89	E3	FROM	01011011001101110
03	8C28	24	16	89	E3	24	0001010000110000111
02	9A3F	4D	16	89	4D	24	1111110001011001O1
00	B25C	4B	4B	89	4D	24	0011101001001101O0
00	CBAD	BB	BB	89	4D	24	1011010111010011O0
03	D202	5D	BB	89	4D	5D	0100000001001011111
00	D322	73	73	89	4D	5D	0100010011001011O0
03	E536	DF	73	89	4D	DF	0110110010100111111
03	F680	C7	73	89	4D	C7	0000000101101111111
01	EB9E	C9	73	C9	4D	C7	0111100111010111110
01	C98F	81	73	81	4D	C7	1111000110010011110
00	CF7A	0B	0B	81	4D	C7	0101111011110011O0
...	...	...	.....		...	...	. . .

Example 4

		RESET	00	00	00	00	
CHANNEL	ADDRESS	FACT	Z0	Z1	Z2	Z3	W useful
03	03E8	29	00	00	00	29	000101111111
01	A2E5	1D	00	1D	00	29	101001110100010110
00	5CA4	39	39	1D	00	29	00100101001110100
03	8EED	1A	39	1D	00	1A	101101110111000111
02	6616	15	39	1D	15	1A	01101000011001101
01	F177	07	39	07	15	1A	111011101000111110
03	5680	27	39	07	15	27	00000001011010111
02	7E09	E9	39	07	E9	27	10010000011111101
02	1585	7C	39	07	7C	27	101000011010101
03	CC42	05	39	07	7C	05	010001000011001111
02	3ACD	8F	39	07	8F	05	1011001101011101
02	170F	BC	39	07	BC	05	111100001110101
00	B73E	CA	CA	07	BC	05	011111001110110100
03	3B89	FE	CA	07	BC	FE	1001000111011111
02	09FD	FB	CA	07	FB	FE	10111111100101
01	4630	82	CA	82	FB	FE	00001100011000110
...	...	...	.....		...	...	. . .

Example 5

		RESET	00	00	00	00	
CHANNEL	ADDRESS	FACT	Z0	Z1	Z2	Z3	W useful
02	03E8	D6	00	00	D6	00	000101111101
01	2FCD	A6	00	A6	D6	00	1011001111110110
01	3E1C	1C	00	1C	D6	00	0011100001111110
01	891B	DE	00	DE	D6	00	110110001001000110
00	ACD7	14	14	DE	D6	00	111010110011010100
02	BEBD	3B	14	DE	3B	00	101111010111110101
01	020E	AB	14	AB	3B	00	011100000110
00	20BB	43	43	AB	3B	00	1101110100000100
02	4E5F	FF	43	AB	FF	00	11111010011100101
01	7B5F	FROM	43	FROM	FF	00	11111010110111110
03	9CCA	15	43	FROM	FF	15	010100110011100111
00	D717	FROM	FROM	FROM	FF	15	111010001110101100
01	0633	7E	FROM	7E	FF	15	1100110001110
02	29C1	C2	FROM	7E	C2	15	1000001110010101
02	591B	B9	FROM	7E	B9	15	11011000100110101
03	89CB	EB	FROM	7E	B9	EB	110100111001000111
...	...	...	.....		...	...	. . .

Example 6

		RESET	00	00	00	00	
CHANNEL	ADDRESS	FACT	Z0	Z1	Z2	Z3	W useful
01	000A	20	00	20	00	00	010110
02	000F	07	00	20	07	00	111101
03	0012	1D	00	20	07	1D	0100111
00	0015	F7	F7	20	07	1D	1010100
02	001E	26	F7	20	26	1D	0111101
01	0023	6D	F7	6D	26	1D	11000110
03	0025	C6	F7	6D	26	C6	10100111
03	002F	CA	F7	6D	26	CA	11110111
01	0036	35	F7	35	26	CA	01101110
00	003F	03	03	35	26	CA	11111100
00	0043	E9	E9	35	26	CA	110000100
00	0049	65	65	35	26	CA	100100100
01	0050	40	65	40	26	CA	000010110
02	005A	6E	65	40	6E	CA	010110101
02	005F	F6	65	40	F6	CA	111110101
00	0068	26	26	40	F6	CA	000101100
...	...	...	.....	...	...	...	