Proceedings of the 5th WSEAS Int. Conf. on APPLIED INFORMATICS and COMMUNICATIONS, Malta, September 15-17, 2005 (pp415-418)

The 4 Factorial Phase Decoder Circuit

Dr. SERAFIM PORIAZIS
Phasetronic Laboratories
6 Depasta Str., 17122, N.Smirni, Athens
GREECE
http://www.phasetroniclab.com

Abstract: - The design of the 4 Factorial Phase Decoder (FCT4) circuit is described by using a tree-like modular core structure written in synthesizable VHDL. The circuit produces the factorial pattern of four encoded phases of the input line by streaming sets of twenty four (4!) clock phases. A phase difference equal to the half period of a clock signal is used at the input line and this is maintained between the consecutive output line transitions. The components of the core model of the FCT4 is given in behavioral VHDL description and the simulation and synthesis results are generated for targeting an FPGA device.

Key-Words: - Circuit design, Decoder, Factorial, FPGA, Modular structure, Phase, VHDL

1 Introduction

VHDL (VHSIC Hardware Description Language) is intended to describe digital electronics systems from the abstract to the concrete level. VHDL CAD tools allow a design written in VHDL to be synthesized and targeted to a suitable technology. An FPGA (Field Programmable Gate Array) can be the target technology of a specific decoder, such as the BCH (15, 7, 5) minimum weight decoder for double error as given by [1].

The VHDL description is run through a VHDL simulator which demonstrates the effectiveness of the model of a 3 to 8 line decoder as given by [2]. The detailed VLSI design of a turbo decoder based on Log-MAP decoding algorithm is given by [3] where the VLSI implementation is performed using VHDL as design and simulation entry. The logic design of the Serial Viterbi Decoder (SVD) is applied with top-down design methodology using VHDL, and simulated by VHDL logic simulator [4], while the logic timing is verified using output VHDL file by VHDL synthesis and FPGA place and routing. A Multi-Amplitude Continuous Phase Receiver that uses a new design of phase generating circuit has been designed by generating and synthesizing the VHDL code of the behavioral model [5].

In this paper the 4 Factorial Phase Decoder (FCT4) circuit is designed by a modular tree-like structure of basic decoding cores written in VHDL, which forms a reliable solution to the challenging problem of synchronizing the multiphase model [6], whose operation adopts a (4!)-phase timing pattern. The present work is based on the principles of operation of the Two-Phase Twisted Ring Counter (2P-TRC) circuit [7] and is targeting data streaming applications. The unit phase duration that is used throughout the text is equal to the half period of an input clock signal. The VHDL simulation and synthesis results are presented, which verify the correct operation of this circuit targeting an FPGA device.

2 The FCT4 Core Modules

2.1 The modular structure

The structure of the 4 Factorial Phase Decoder circuit is given in Figure 1, where three levels of basic decoding modules are connected in a tree-like form. The first level on the left side (with label 'a') receives the input signal and performs a two-phase decoding function via module PD2. At the second level in the middle (with label 'b'), each module PD3 receives the outputs of the first level and performs a three-phase decoding function. Finally, at the third level on the right side (with label 'c'), each module PD4 receives the outputs of the second level and performs a four-phase decoding function. The output of this circuit consists of twenty-four (4!) signals (labelled by PCLK[24..1]) each having the phase index (PHS-i, i=1...24) of the phase information that was encoded in the input signal. We note that a clock signal is used as the input line signal of the FCT4 throughout this work for implementing a simple pattern of a sequence of twenty-four phases.

2.2 The basic module operation

We consider in this section the design of the PD4 module, while similar principles apply to the design of the PD2 and PD3 modules within FCT4. The fundamental module PD4 that decodes an input line signal CLK_c of frequency f_c into four line phased signals of frequency $f_c/4$, is called the 4-Phase Decoder (PD4) module, which is shown in Figure 1 at the third level on the right side. The four overlapping output line signals CLK_1=cDOUT1, CLK_2=cDOUT2, CLK_3=cDOUT3, CLK_4=cDOUT4 have frequency equal to $f_c/4$ (period of each CLK_i, i=1,2,3,4 equals $4 \cdot T_c$, where T_c the period of the predecessor PD3 module) with signal CLK_1 leading CLK_2, CLK_2 leading CLK_3, CLK_3 leading CLK_4 by a $T_c/2$ phase difference. We note that the above signals CLK_i are the outputs of a module PD3 of the second

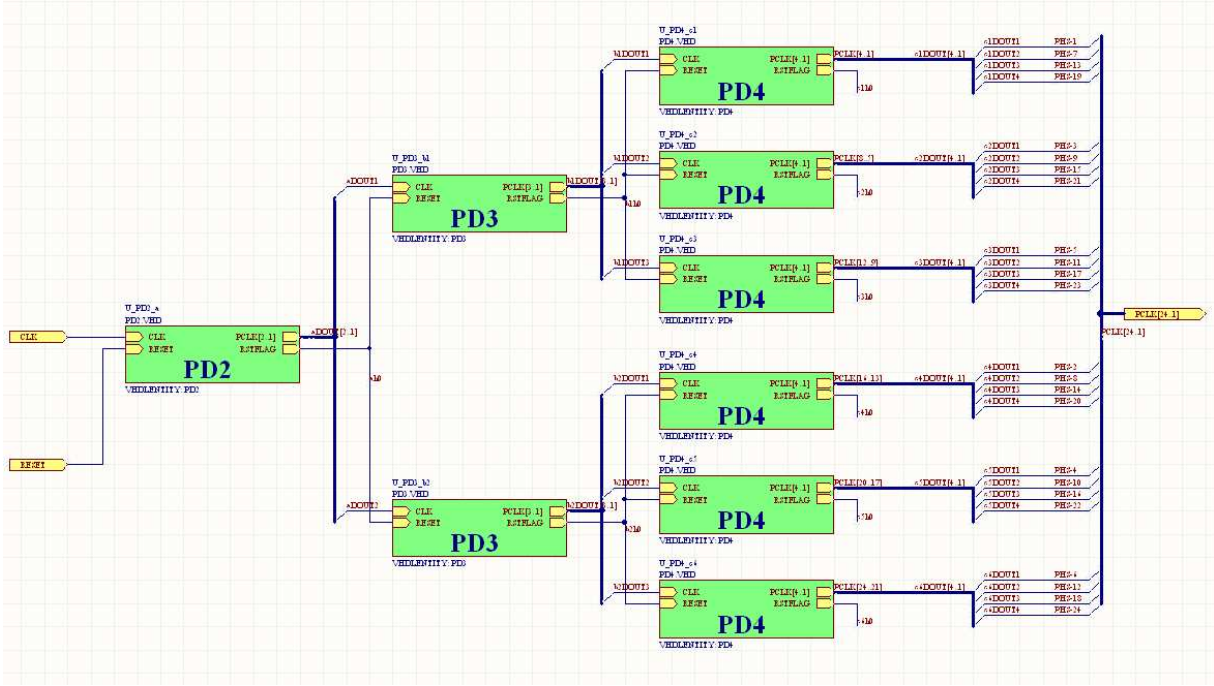

Fig. 1 The FCT4 modular block diagram

internal level of FCT4 and is labelled by 'c' . The logic-'1' or logic-'0' pulse width of each of the above phased signals is equal to $4 \cdot T_c/2$. Consecutive changes of logic value at each signal CLK_i, for i=1,2,3,4 occur alternatively at the rising and falling edges of CLK_c at a distance of $4 \cdot T_c/2$.

2.3 The valid line codeword sequence

When the line signal CLK_c is applied to the PD4 module, the following cyclic sequence of line codewords is presented at the outputs: CLK_1,CLK_2,CLK_3,CLK_4= 0000→1000→1100→1110→1111→0111→0011→0001, which is considered as being the normal circuit operation. Each line codeword remains stable for the state time of the module, that is $T_c/2$, and the above sequence is repeated throughout the operation of the module. Thus the cycle time for the output pattern of PD4 is defined by the eight-tuple of codewords of length equal to $4 \cdot T_c$. This duration forms the period of each phased output line signal of the third level in association to the outputs of the second level.

The additional 8 line codewords out of the total 16 possible line codewords for the PD4 module that are not included in the above cyclic sequence should be considered during the design of this module for achieving reliable operation of the core. If this module reaches any of these 8 invalid codewords, then an invalid codeword flag is set at the output of the module. This flag maintains the proper initializing behavior until a valid line codeword appears on the output port of the module.

2.4 The algorithm aspects of core operation

The PD4 core operation is implemented by the behavioral VHDL description shown in Figure 2. The VHDL entity section has an input port CLK_c on which the line signal of frequency f_c is applied and an input port $RESET_c$ on which a reset flag is applied. The phased output line signals of frequency $f_c/4$ of the module are assigned to port $PCLK_c$[4..1] of width four, that is PCLK1=cDOUT1, PCLK2=cDOUT2, … , PCLK4= cDOUT4. The output port $RSTFLAG_c$ is signaling the invalid line codeword status of $PCLK_c$, or the core reset state, which suspends the streaming of phases from CLK_c toward the outputs of the module. The VHDL architecture section is of type "behavioral" and utilizes a state machine model, where two internal registers are being used, reg1 and reg2, one for the present state named "present_state1" clocked by the rising edge of the clock and the other for the present state named "present_state2" clocked by the falling edge of the clock, respectively. The next state logic block and the ouput logic block of the model are specified by the corresponding processes "next_state_logic" and "output_logic". The set of eight valid line codewords of the circuit are stored in an indexed array of size 8*4=32 bits, that is represented by the constant named "phased_output". The index of the above array cycles through the integer values 1 to 8 specifying the valid line codeword entry for the next state signal. Whenever the $RESET_c$ input is set, that is $RESET_c$='1', each output signal from $PCLK_c$[4..1] can cycle only through the values "0000" and "1111" thus assuring proper initialization of the circuit at either the rising or the falling edge of CLK_c.

3 The VHDL simulation and Synthesis results

The VHDL testbench simulation results for the FCT4 core are given in Figure 3. The duration of this simulation is defined by the value of the signal "done". Each of the signals "next_state", "present_state1", "present_state2" and PCLK per PD4 module have each a width of 4 bits. Each value shown on these signal waveforms is hexadecimal. The output port PCLK is analyzed into twenty-four individual

```
1   library IEEE;
2   use ieee.std_logic_1164.all;
3
4   entity PD4 is
5       port (CLK , RESET : in std_logic;
6             RSTFLAG : out std_logic;
7             PCLK : out std_logic_vector(4 downto 1));
8   end PD4;
9
10  architecture behavioral of PD4 is
11        type validcodewords is array(1 to 8) of std_logic_vector(4 downto 1);
12        constant phased_output : validcodewords :=
13        (  ( '0', '0', '0', '0'), ( '0', '0', '0', '1'), ( '0', '0', '1', '1'),
14           ( '0', '1', '1', '1'), ( '1', '1', '1', '1'), ( '1', '1', '1', '0'),
15           ( '1', '1', '0', '0'), ( '1', '0', '0', '0') );
16         signal index : integer := 0;
17         signal present_state1, present_state2, next_state: std_logic_vector(4 downto 1);
18         signal invalidcode_flag : std_logic := '0';
19        begin
20         reg1 : process (CLK, RESET)
21            begin
22               if  RESET = '1' then  present_state1 <= phased_output(1);
23               elsif (CLK='1' and CLK'event )  then present_state1 <= next_state ;
24               end if;
25         end process;
26         reg2 : process (CLK, RESET)
27            begin
28               if  RESET = '1' then  present_state2 <= phased_output(5);
29               elsif (CLK='0' and CLK'event )  then present_state2 <= next_state ;
30               end if;
31         end process;
32         next_state_logic : process (CLK, RESET)
33            begin
34               case CLK is
35                when '1' =>  if RESET = '1' then index <= 1;  else index <= index + 1; end if;
36                when '0' =>  if RESET = '1' then index <= 5;  else index <= index + 1; end if;
37                when others   => null;
38               end case;
39               if index < 8 then next_state <= phased_output(index + 1);
40               else next_state <=  phased_output(1); index <= 1;  end if;
41               for i in 1 to 8 loop
42                 if next_state = phased_output(i) then invalidcode_flag <= '0'; exit;
43                 else invalidcode_flag <= '1'; end if;
44               end loop;
45          end process;
46          output_logic : process (index, present_state1, present_state2)
47            begin
48               case CLK is
49                when '1' =>  PCLK <= present_state1;
50                                    if RESET = '1' then RSTFLAG <= '1'; else
51                                    RSTFLAG <= invalidcode_flag;  end if;
52                when '0' =>  PCLK <= present_state2;
53                                    if RESET = '1' then RSTFLAG <= '1'; else
54                                    RSTFLAG <= invalidcode_flag;  end if;
55                when others   => null;
56               end case;
57          end process;
58  end behavioral;
```

Fig. 2 The VHDL description of the PD4 module at level 'c'

output line signals with waveforms that verify the correct operation of the circuit. The "index" signal has decimal values and defines the index value of the array of valid line codewords. The logic value changes of PCLK occur at each rising and at each falling edge of the input signal CLK.

The synthesis of the FCT4 core targeting an FPGA device was successfully performed giving us the following results:

- flip flops with asynchronous reset = 32
- flip flops with asynchronous preset = 32
- combinational feedback paths = 319
- combinational logic area estimate = 917 LUTs

4 Conclusion

The operation and the design concepts of the 4 Factorial Phase Decoder (FCT4) circuit are considered in this paper. The behavioral VHDL description of the FCT4 core that is based on a three-level tree-like structure of interconnected decoding modules is given. The resulting twenty-four (4!) output signals decode the phase information of the input line signal. The corresponding circuit simulation and synthesis results targeting an FPGA device successfully verify the proper core behavior.

References:

[1] El-Medany, et al. "*VHDL implementation of a BCH minimum weight decoder for double error*," in Proceedings of the Eighteenth National Radio Science Conference, NRSC 2001, Vol.2, pp.361-368, 2001.

[2] M.B.I. Reaz, S.Z. Islam, "*VHDL modeling of built-in-self-test (BIST) for system on chip (SOC) design*," in Proceedings of the IEEE International Conference on Semiconductor Electronics, ICSE 2002, pp.222-225, 19-21 Dec. 2002.

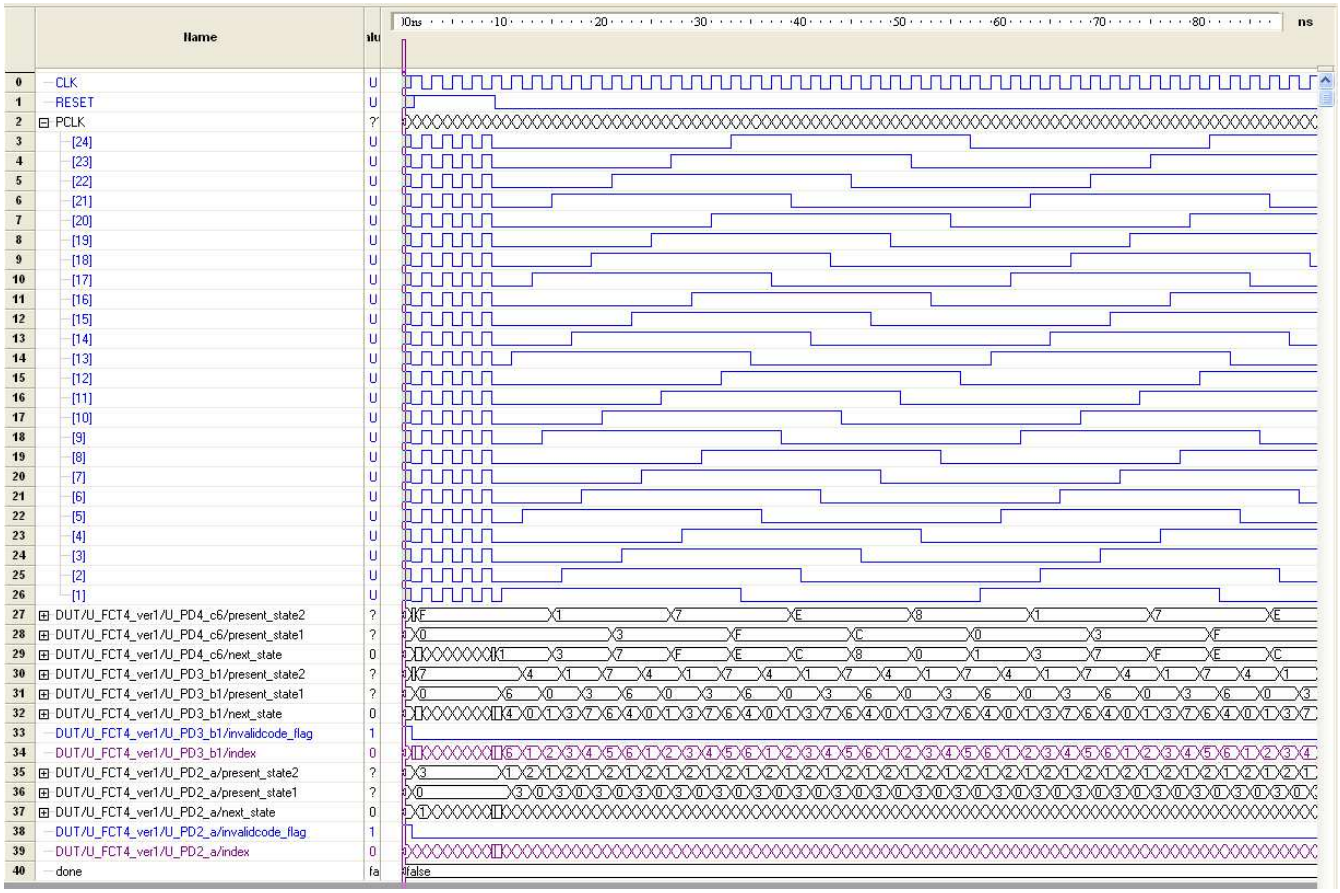

Fig. 3 The FCT4 circuit operation (VHDL simulation results)

[3] Tong Yanhui, T.H.Yeap, J.Y.Chouinard, "*VHDL implementation of a turbo decoder with log-MAP-based iterative decoding*," in IEEE Transactions on Instrumentation and Measurement, Vol.53, Issue 4, pp.1268-1278, Aug. 2004.

[4] Park Jang-Hyun, Rho Yea-Chul, "*Performance test of Viterbi decoder for wideband CDMA system*," in Proceedings of the ASP-DAC 1997, Asia and South Pacific Design Automation Conference, pp.19-23, 28-31 Jan. 1997.

[5] H.H. Ali, "*VLSI implementation of a multi-amplitude continuous phase receiver*," in Proceedings of the 44th IEEE 2001 Midwest Symposium on Circuits and Systems, MWSCAS 2001, Vol.1, pp.393-396, 2001.

[6] S. Poriazis, *Logic Design of Multiphase Finite State Machines*, Monograph, Phasetronic Laboratories, http://www.phasetroniclab.com Athens, Greece, 2001.

[7] S. Poriazis, "*The Two-Phase Twisted-Ring Counter Circuit,*" in Proceedings of the 2002 IEEE International Symposium on Circuits and Systems (ISCAS'02), Phoenix, Arizona, USA, vol. IV, pp. 858-861, 26-29 May, 2002.