

Configurable VHDL Models for Field Programmable Gate Arrays

Mohammad Reza Movahhedin and Zainalabedin Navabi
Electrical and Computer Engineering Department
Faculty of Engineering ; University of Tehran
North Kargar Avenue ; 14399 Tehran , Iran
Fax: +(98-21) 688690

ABSTRACT

We have developed a VHDL modeling strategy for modeling Field Programmable Gate Arrays (FPGAs). Different modeling levels have been implemented for Xilinx XC2000 products as well as other series such XC3000 and XC4000. These models can be used for FPGA simulation from LCA file format. This file is used for configuring our programmable models as well as the actual FPGA parts.

INTRODUCTION

Just as gate arrays are programmable integrated circuits that can be programmed by data from a certain configuration file, their equivalent VHDL models must be configurable and can be configured with the same configuration data file. Chip configuration files include routing, placement, fan out and detailed timing information and can be converted to real chip's program bit stream without requiring additional information. This data is used for generation of a configurable VHDL model for the programmed gate arrays.

Modeling strategy presented here uses separate VHDL description for the individual gate array cells, and it uses an overall description for specifying wiring and interconnections of the cells.

Various forms of configurable VHDL FPGAs models can be useful in simulation and testing of the gate arrays, or in obtaining an FPGA model as part of a larger system. Several modeling styles are used for fast simulation or detailed timing analysis.

The paper consists of five sections. In section 1 top-down design and related requirements will be discussed. Section 2 includes a brief review of Xilinx XC2000 series FPGA products. In section 3 we will explain our strategy for FPGA modeling which consists of two subsections of Cell Modeling and Chip or Interconnection Modeling. Section 4 discusses data extraction methods. The last section presents conclusion, simulation performance and its accuracy.

1. Using Gate Array Models in a Design Environment

A large digital system design is usually an integration of many parts developed by various designers at different levels of abstraction. It is important that at one point in the design process, various components are put together and simulated under the same simulation and verification environment. In a VHDL based design environment, if part of design consists of an FPGA, it becomes necessary to form an accurate VHDL model for that part of the design and simulate that part along with the rest of the circuit. In addition to this integration, a VHDL model of an FPGA enables the FPGA designers to take advantage of the powerful VHDL simulation tools for the verification of their gate array designs.

For various applications in a top-down process, three modeling strategies are being proposed:

A) A system consisting of only FPGAs.

These systems need a high level equivalent VHDL model so that developers and designers can use VHDL simulation environment for their verification and debugging. This can be achieved by a C program which converts LCA file formats to VHDL description and must be compiled and used every

time for simulation, verification and debugging.

B) An FPGA as a part of a larger system requiring debugging.

A programmable and configurable VHDL model with debugging abilities is useful for designers of such systems. This VHDL model is compiled only once and configured to a specific function by different LCA file formats. The model is capable of reading and configuring itself according to LCA files with no need for recompilation. This model is at the intermediate level and can be used in any application.

C) An FPGA as a part of a larger system required fast simulation purposes.

Such systems need a configurable VHDL model. The model based on an LCA file to configure itself by a table look up. The contents of the table is extracted from an LCA file. Because of this table look up, this modeling strategy presents a fast model simulation.

The work presented here concentrates as part (B) discussed above.

2. Xilinx XC2000 series FPGAs.

Every FPGA is formed by an $n * n$ ($8 * 8$ or $10 * 10$) array of CLBs (Configurable Logic Blocks) and IOBs (Input Output Blocks) which are connected together by programmable interconnection switches (Fig. 1).

IOBs have one output and three input ports (Fig. 2). The *IN* output of IOB is programmed to be connected directly to a pad or through a D flip flop. The *CLK* input is connected to the D flip flop clock. The *OUT* input of IOB is connected to an I/O pad by a three state buffer which can be programmed to be ON, OFF or controlled by *TS* input of IOB.

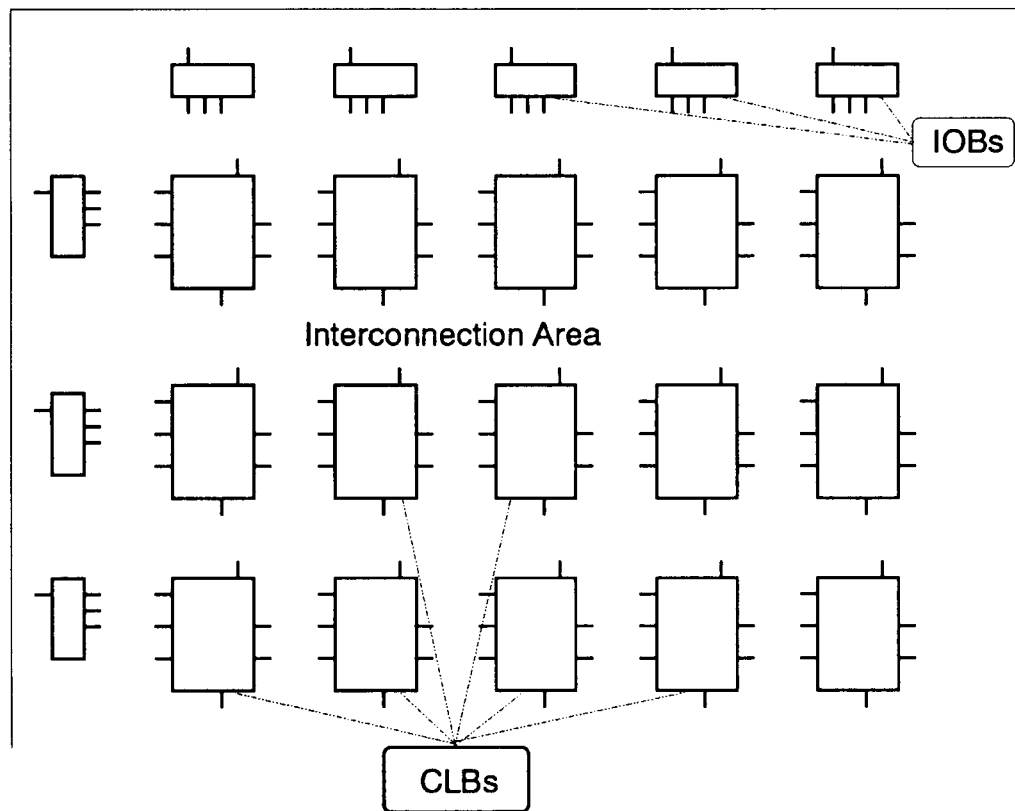


Figure 1. FPGA structure: CLBs, IOBs and interconnections

CLBs have a higher programmability. They have four inputs (*A, B, C, D, K*) and two outputs *X* and *Y* (Fig. 3). Each CLB can be used for implementing combinatorial logic or storage circuits. Combinatorial logic uses a memory table look up to implement any Boolean function. Therefore, the

propagation delay through the combinatorial is independent of its function and depends on memory delays only. Combinatorial logic can be used in one of three ways, shown in Fig. 4 Two 8 by 1 Boolean function memory and all program controlled multiplexers that are part of CLBs must be programmed during FPGA configuring.

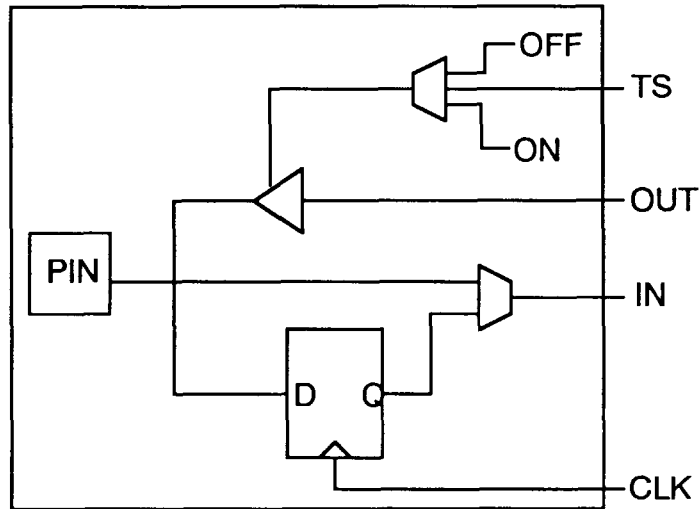


Figure 2. Input Output Blocks (IOBs)

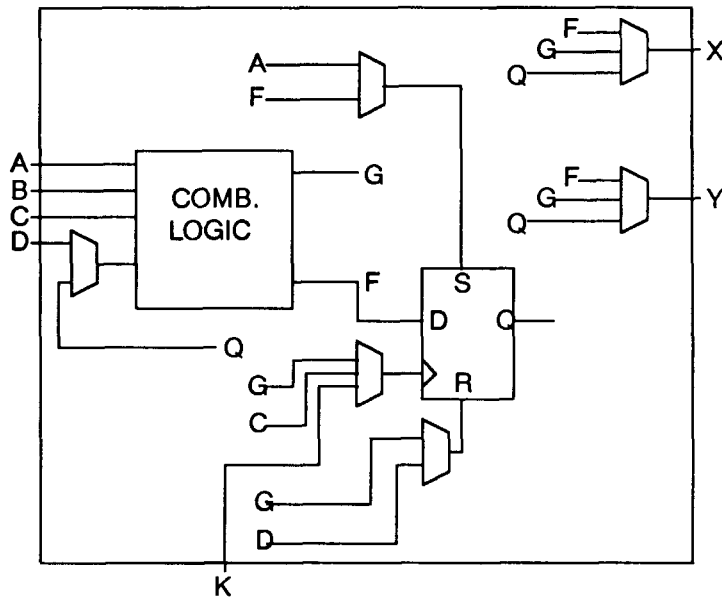


Figure 3. Configurable Logic Blocks (CLBs)

Outside the cells, the chip has many interconnection routings. Output of any CLB can be connected to its near CLBs input by direct connections. Two vertical and one horizontal long lines between the cells, access all CLBs in their route. Five vertical and four horizontal general purpose interconnections are also implemented and can be programmed by programmable interconnection points (PIPs) or switch matrices.

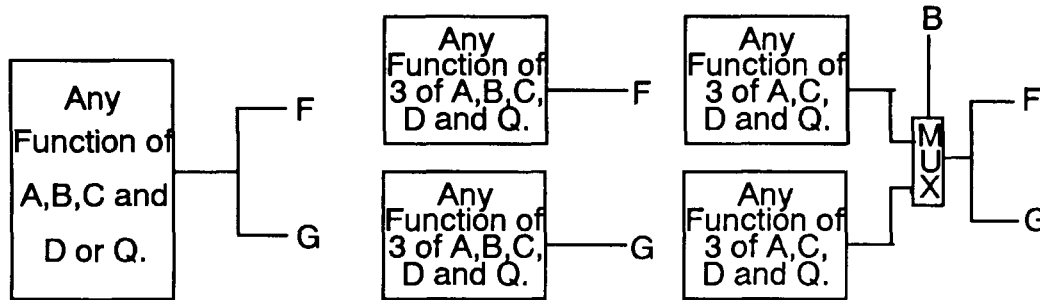


Figure 4. Three options of combinatorial logic usage

An XC2000 FPGA is equivalent to 1200 (8*8 CLBs) or 1800 (10*10 CLBs) gates which must be programmed by about 12 K bit or 18 K bit program data. XC3000 and XC4000 series of Xilinx have more advanced CLBs and more blocks. 320 CLBs in XC3090 equivalent to 9000 gates and 900 CLBs in XC4020 are equivalent to 20,000 gates in highest capacity product in every series.

3. Modeling Strategy

3.1. Cell Modeling

VHDL models of CLBs and IOBs directly correspond to the structure of the hardware that they model. Figure 5 shows a portion of the VHDL description for the CLB of Fig. 3. The selected signal assignments in this description model the multiplexers whose outputs are *S* and *X* respectively. The process statement shown in this figure models the D-flip flop, and the rest of this code consist of signal assignments that set value to signals used to specify internal cell delays. Such models also include signals for configuring the function of the cell. The configuration data is extracted from LCA files. Configuration information could be passed through generic parameters, but a set of signals used for this purpose more closely correspond to the actual hardware used for cells. Additionally, signals used for this purpose can be assigned values that are read from LCA configuration files.

```

ARCHITECTURE model OF CLB2000 IS
  SIGNAL f,g,q,q_d:BIT;
  SIGNAL set_d,res_d,clk_d,res_g:BIT;
  SIGNAL memory_input:bit_vec_2_3;
  SIGNAL memory_output:BIT_VECTOR(0 TO 1);
BEGIN
  WITH clbs_config(row,col).set SELECT set_d<=
    a AFTER set_res_delay WHEN 'a',
    f AFTER set_res_delay WHEN 'f',
    '0' WHEN ' ';
  . . .
  WITH clbs_config(row,col).x_i SELECT x<=
    f AFTER output_delay WHEN 'f',
    g AFTER output_delay WHEN 'g',
    q AFTER output_delay WHEN 'q';
  . . .
  q_d<=f AFTER combinational_delay;
  res_g<=global_reset AFTER global_res_delay;
  PROCESS (q_d,set_d,res_d,clk_d,res_g)
    VARIABLE internal_state:BIT;
  BEGIN
    IF set_d='1' THEN internal_state:='1';END IF;
    IF res_d='1' THEN internal_state:='0';END IF;
    IF res_g='0' THEN internal_state:='0';END IF;

```

```

CASE clbs_config(row,col).f_l IS
  WHEN 'f' => IF clk_d='1' AND clk_d'EVENT THEN
    internal_state:=q_d;
  END IF;
  WHEN 'l' => IF clk_d='1' THEN
    internal_state:=q_d;
  END IF;
  WHEN OTHERS => NULL;
END CASE;
q <= internal_state AFTER latch_delay;
END PROCESS;
. . .
combinational_delay <= clb_delay_table(spd,Trlo) -
clb_delay_table(spd,Trio);
latch_delay <= clb_delay_table(spd,Tito) - clb_delay_table(spd,Tilo);
output_delay <= clb_delay_table(spd,Tilo) - combinational_delay;
set_res_delay <= clb_delay_table(spd,Trio) - latch_delay - output_delay;
k_clock_delay <= clb_delay_table(spd,Tcko) - latch_delay - output_delay;
c_clock_delay <= clb_delay_table(spd,Tcco) - latch_delay - output_delay;
q_delay <= clb_delay_table(spd,Tqlo);
global_res_delay <= clb_delay_table(spd,Tmro)-latch_delay -
output_delay;
END model;

```

Figure 5. CLB model

Corresponding to the loading of bit stream data into configuration memory of an FPGA chip after power on, we use global signals for cell configuration (Fig. 6). These signals configure CLBs and IOBs and their values are set by a procedure that reads LCA files. This procedure is called when the model is powered on.

```

PACKAGE xilinx2000 IS
. . .
TYPE clb_config IS RECORD
  mode:CHARACTER;
  memory_input_config:string_2_3;
  memory:bit_vec_2_8;
  d_q:CHARACTER;
  set:CHARACTER;
  res:CHARACTER;
  clk:STRING (1 TO 2);
  f_l:CHARACTER;
  x_i:CHARACTER;
  y_i:CHARACTER;
END RECORD;
. . .
TYPE iob_config IS RECORD
  in_i:CHARACTER;
  trst:CHARACTER;
END RECORD;
. . .
TYPE clb_loc IS (A,B,C,D,E,F,G,H,I,J);
SUBTYPE iob_loc IS INTEGER RANGE 1 TO 74;

```

```

. . .
TYPE clbs_config_type IS ARRAY (clb_loc,clb_loc) OF clb_config;
TYPE iobs_config_type IS ARRAY (iob_loc) OF iob_config;
. . .
CONSTANT clb_default:clb_config:=
    ('g', ("abc","abc"), ("01111111","00000000"), 'd',' ',' ',' k','f','f','g');
CONSTANT iob_default:iob_config:=('q','t');
. . .
END xilinx2000;

```

Figure 6. Configuration information signals

3.2. Chip and Interconnection Modeling

An actual FPGA hardware has many PIPs and switch matrices. Data sheets on FPGAs contain little information on PIP and switch matrices. However, an LCA file contains complete interconnection information.

A connection between a source (a CLB or IOB output) and multiple targets (some CLB or IOB inputs) have different delays according to the path through one or more PIPs and switch matrices. This delay is caused by pass transistors that are used for PIPs and switch matrix implementation. Since our models do not have direct correspondence to such switches, we have implemented delays by use of scheduling signal assignments.

On the other hand, target cells should be able to use any cell as source. We have, therefore, made provision such that all input signals to all CLBs and IOBs could be mapped to any source from other cells. This is achieved by making individual assignment of a general array of output signals to every cell input. For example, the last five signal assignments of Figure 7 each shows assignment of an indexed array of *im* to specific CLB and IOB input ports. The instantiation statement in this figure individually map lines of the *im* intermediate signal to output of all CLBs. Related to any input signal, we used a program record that its fields is signal source index and its transmission delay of source. This record used in a part of code as shown in Fig. 7 and can connect any source to desired target with transmission delay. These records of signals is a subset of configuration signals which must be set by LCA file reader procedure.

```

TYPE net_delay IS RECORD
    src_net:INTEGER; delay:TIME;
END RECORD;
. . .
ARCHITECTURE model OF XC2000 IS
. . .
    SIGNAL clb_a,clb_b,clb_c,clb_d,clb_k:clb_signal;
    SIGNAL iob_o,iob_k,iob_t:iob_signal;
    SIGNAL clb_net_a,clb_net_b,clb_net_c,clb_net_d,clb_net_k:clb_net;
    SIGNAL iob_net_o,iob_net_k,iob_net_t:iob_net;
    SIGNAL clk_aa,clk_kk:net_delay:=net_def;
    SIGNAL im:BIT_VECTOR(0 TO 2*clb_size**2+iob_size+2);
. . .
    ALIAS osc_out:BIT IS im(2*clb_size**2+iob_size);
    ALIAS clk_aa_out:BIT IS im(2*clb_size**2+iob_size+1);
    ALIAS clk_kk_out:BIT IS im(2*clb_size**2+iob_size+2);

```

```

BEGIN
  clb_i:FOR i IN clb_loc'RANGE GENERATE
    clb_j:FOR j IN clb_loc'RANGE GENERATE
      clb:CLB2000
        GENERIC MAP(i,j)
        PORT MAP
          (clb_a(i,j),clb_b(i,j),clb_c(i,j),clb_d(i,j),clb_k(i,j),
            im(2*(clb_size*clb_loc'POS(i)+clb_loc'POS(j))),
            im(2*(clb_size*clb_loc'POS(i)+clb_loc'POS(j))+1));
        clb_a(i,j) <=
          im(clb_net_a(i,j).src_net) AFTER clb_net_a(i,j).delay;
        clb_b(i,j) <=
          im(clb_net_b(i,j).src_net) AFTER clb_net_b(i,j).delay;
        clb_c(i,j) <=
          im(clb_net_c(i,j).src_net) AFTER clb_net_c(i,j).delay;
        clb_d(i,j) <=
          im(clb_net_d(i,j).src_net) AFTER clb_net_d(i,j).delay;
        clb_k(i,j) <=
          im(clb_net_k(i,j).src_net) AFTER clb_net_k(i,j).delay;
      END GENERATE;
    END GENERATE;
  . . .
  . . .
END model;

```

Figure 7. Programmable delayed interconnection implementation

4. Data Extraction

As explained earlier, a LCA file reading is necessary to set all configuration signals which includes cell configuration, interconnections and their delay information. This procedure is also sensitive to *PROG* pad which reads new program bit stream in the actual hardware.

Similar to this VHDL procedure, a *LCA2VHDL* program is implemented that converts any LCA file to its VHDL description and useful for single FPGA system designers.

5. Conclusions

We have shown VHDL modeling strategy for Xilinx FPGAs. By use of these models, FPGA users can benefit from powerful VHDL simulation and design environments. In addition, VHDL users can easily integrate FPGAs in their designs. Although, Xilinx XC2000 series have been primarily addressed in this paper, extending the methods to other Xilinx series and other FPGA manufactures is easily possible. Preliminary simulation performance analysis of our models indicates an improvement of the simulation speed when compared to gate level simulation.

REFERENCES:

1. Navabi, Z, "VHDL: Analysis and Modeling of Digital Systems", McGraw Hill Publishing, New York, N.Y., 1993.
2. "XACT 2000/3000 Programmable Gate Arrays Development Systems", Xilinx Inc., 1992.
3. "The Programmable Gate Arrays Data Book", Xilinx Inc., 1991.