Performance Evolution of 16 Bit Processor in FPGA using State Encoding Techniques

Madhavi Anupoju[1], M. Sunil Prakash[2]

[1]M.Tech (VLSI) Student, Department of Electronics & Communication Engineering,
MVGR College of Engineering, Vizianagaram, Andhra Pradesh, India

[2]Professor, Department of Electronics and Communication Engineering,
MVGR College of Engineering, Vizianagaram, Andhra Pradesh, India

Abstract: ***In present days, there is a need for ever increasing high performance and low power devices, these devices need to meet performance constraints like speed, area& power. This paper describes the area and speed constraints of a 16 bit processor with the implementation of three state encoding techniques binary, one- hot & gray coding technique. The processor architecture is designed using Verilog HDL, simulated on Modelsim and synthesized on Precision RTL synthesis tool & on XILINX ISE 12.1 for the Spartan3E FPGA. From the synthesis reports it is observed that One-hot encoding would perform with speed 28% and Gray code would perform with speed 14% more than binary encoding technique, but both of them require more area compared to binary encoding technique.***

Keywords: FPGA, Precision synthesis, State encoding techniques, VLSI circuits, VERILOG hardware description language

1. Introduction

In today's era of area optimization, high speed systems and ubiquitous computing, the need for real-time embedded systems is always on rise. These embedded systems must operate within stringent requirements that are often at the intersection of the conflict between speed and area. Increasing the complexity of signal processing in embedded and real time application that requires very high computational power and area also optimizes the area. This power can be achieved by high performance programmable components like RISC processor.

This paper describes the different state encoding techniques to increase the speed as well as to optimize the area. The optimization of area can be achieved by implementing the processor design on FPGA instead of ASIC. FPGAs are usually slower than ASICs but have the ability to be re-programmed in the field where the errors need to be corrected and upgraded, flexibility and low-cost.

This paper deals with some of the state encoding techniques namely, the One-Hot Code, Binary/Sequential Code, and Gray Code state assignment. In the one –hot encoding one flip-flop will be in on state, remaining all flip-flops will be in off state in a state machine, so one-hot encoding reduces the switching activity. No state decoding is necessary. One-hot state machines are typically faster. In Gray encoding only one flip-flop will transfer its state while the state changes so it also reduces the switching activity, and reduces the delay. Whereas binary encoding uses less no of flip-flops but the encoding and decoding of information is more complex. Binary encoding technique will occupy less amount of area compared to other techniques, but the speed is less. The design of the processor is carried out through VERILOG hardware description language.

This paper has been organized as follows: in the next section2 we will review the three state encoding techniques. The next sections 3&4 dedicated to the review of design 16 bit RISC processor and implementation of the state encoding techniques on processor, in these sections the simulation and synthesis results were presented. Section5 deals with the conclusion.

2. State Encoding Techniques

There are several methods of encoding the state assignments when designing synchronous finite state machines (FSM) according to [2] - [4]. The decision on which method to use is a function of design constraints such as speed, area, power consumption, and the type of programmable logic device targeted to. This paper deals with some of the state encoding techniques used in synchronous finite state machine: namely, the One-Hot Code, Binary/Sequential Code, and Gray Code state assignment [6]

2.1. One-Hot Encoding

In the one-hot encoding (OHE) only one bit of the state variable is "1" or "hot" for any given state. All other state bits are zero. (See Table 1) Therefore, one flip-flop (register) is used for every state in the machine i.e. n states uses n flip-flops. State decoding is simplified, since the state bits themselves can be used directly to indicate whether the machine is in a particular state. In addition, with a one-hot encoded state machine, the inputs to the state bits are often simply the functions of other state bits. Often times, no state decoding is necessary, and state encoding can only require the OR-in of state bits.

Table 1: An example of state Encoding for a 4 state Machine

State	State variables		
	One-Hot code	*Binary code*	*Gray code*
S0	00001	000	000
S1	00010	001	001
S2	00100	010	011
S3	01000	011	010
S4	10000	100	110

2.1.1 Why use One Hot Code?

One-hot encoding (OHE) is better suited for use with the fan-in limited and flip-flop-rich architectures of the higher gate count field-programmable gate arrays (FPGAs), such as offered by Xilinx, Actel, and others. OHE maps very easily in these architectures. This is because OHE requires a larger number of flip-flops. It offers a simple and easy-to-use method of generating performance optimized state-machine designs because there are few levels of logic between flip-flops. One-hot state machines are typically faster. Speed is independent of the number of states, and instead depends only on the number of transitions into a particular state.

In some cases, the one-hot method may not be the best encoding technique for a state machine implemented in an FPGA. For example, if the number of states is small, the speed advantages of using the minimum amount of combinatorial logic may be offset by delays resulting from inefficient CLB (configurable logic blocks) use, e.g. a Xilinx device. This assignment allows the designer to create state machine implementations that are more efficient in FPGA architectures in terms of area and logic depth (speed). FPGA have plenty of registers but the LUTs are limited to few bits wide. One-hot increases the flip-flop usage (one per state) and decreases the width of combinatorial logic. It makes it easy to decode the next state, resulting in large FSMs. It consumes less power.

2.2 Binary Encoding

In a binary encoding scheme, the relationship between the number of state variables (q) and number of states (n) is given by the equation:

$$q=\log 2\ (n)$$

With this formula, one can easily determine the minimum number of state variables required for a binary encoded state machine. Clearly, the number of flip-flops used is equal to the number of state variables (q). In this technique, the states are assigned in binary sequence where the states are numbered starting from binary 0 and up.

2.2.1 Why binary encoding?

Binary encoding uses fewer flip-flops/registers than one-hot encoding. For example, binary encoding requires only seven (flip-flops) registers to implement a 100-state machine while a one-hot encoding needs 100 flip-flops. Binary encoding uses the minimum number of state variables (flip-flops) to encode a machine since the flip-flops are maximally utilized. As a result, it generally increases the amount of combinatorial logic because more combinatorial logic is required to decode each state. Therefore, binary encoding is implemented more efficiently when using PLAs and CPLDs. These devices have wider gates and a large amount of combinatorial logic per register. It is usually the preferred method when implementing machines that are fewer than 8 states. Their wide 'AND-architecture' allows any number of state variable (bits) to be included in each product term with no speed (or area) penalty.

The disadvantages of using a binary encoded FSM include the fact that more than one bit can flip at any time and can result in a glitch (hazard) especially in design of counters. It also requires a more complex decoding logic to determine the present state. If power consumption is an issue, then this technique may not be suitable since the more registers/flip-flop changes, the more power the device consumes.

2.3 Gray encoding

Gray code assignment is a state assignment in which consecutive states codes only differ by one bit (adjacent). In other words, only one flip-flop changes at a time when changing consecutive states. In this encoding, the number of flip-flops (register) used is also determined by:

Number of state variables = number of flip-flops = log 2 (number of states)

2.3.1 Why gray code?

Gray code uses the same number of register (flip-flops) as the binary coding technique and the decoding logic can also be as complex if not more. Therefore, gray code assignment is also ideal for PLA and CPLD applications since they have wide gates and a large amount of combinatorial logic per register. Gray code is highly recommended in PLA and CPLD applications when designing for low power requirement. One approach to low power design is to choose a state assignment that diminishes (minimizes) the switching activity of state transitions. The ideal case would be if only one state variable changes in each possible transition. Gray encoding has minimal switching between consecutive states.

3. RISC Processor Design

A 16-bit RISC microprocessor based on a simplified version of the MIPS architecture was designed [1]. The processor normally consists of control unit, data path unit and memory. The processor has 16-bit instruction words and 16 general purpose registers. The ISA of this processor consists of 16 instructions with a 4-bit fixed size operation code. The instruction words are 16-bits long. The entire design of processor is carried out using Verilog hardware description language using [5]-[7].

Table 2: The instruction set architecture

Operation	*Opcode*				*Destination Reg*				*Source Reg*				*Target Reg*			
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADD	0	0	0	0	Rd				Rs				Rt			
SUB	0	0	0	1	Rd				Rs				Rt			
AND	0	0	1	0	Rd				Rs				Rt			
OR	0	0	1	1	Rd				Rs				Rt			
XOR	0	1	0	0	Rd				Rs				Rt			
NOT	0	1	0	1	Rd				Rs							
SLA	0	1	1	0	Rd				Rs							
SRA	0	1	1	1	Rd				Rs							
LI	1	0	0	0	Rd				Immediate							
LW	1	0	0	1	Rd				Rs							
SW	1	0	1	0					Rs				Rt			
BIZ	1	0	1	1	Rs				Offset							
BNZ	1	1	0	0	Rs				Offset							
JAL	1	1	0	1	Rd				Offset							
JMP	1	1	1	0					Offset							
JR	1	1	1	1					Rs							

Here Rd defines the destination register address, Rs defines the source register address, Rt defines the target register address

3.1 Control Unit Design

The Control FSM has only three distinct states that determine the operation of the processor: IDLE, FETCH and EXECUTE. Here fetch and Execute is further divided into two states, Fetch instruction state and Fetch operands state. Similarly Execute state also divided into two parts. The state diagram for the control unit is shown in fig1. The three states are described below.

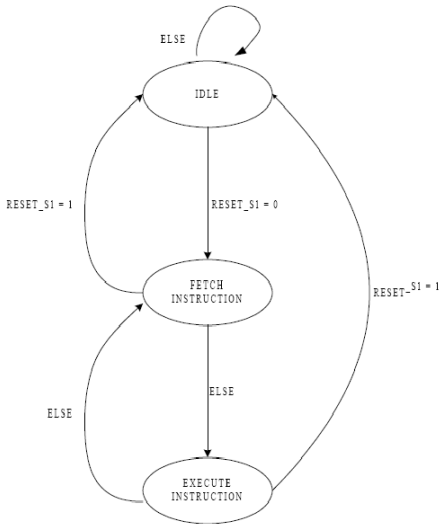

Figure 1: State diagram of control unit

IDLE: In idle state the current program count should be zero.
FETCH INSTRUCTION:

Part 1

- Retrieve instruction word from main memory
- Increment Program Counter and store in ALU Out

Part 2

- Write Incremented Program Count
- Load Operands into latches from Register File

Execute Instruction

Part 1

- Perform ALU Operation based instruction word and store in ALU Out
- Move Memory Word into MDR for Load Word operation
- Write Data into Memory from Register File for Store Word operation

Part 2

- Write ALU, IR (Immediate), or MDR data into Register File
- Write new Program Count for Jump Operation or it Branch taken

3.2 Data Path Unit

The data path unit includes the registers, program counter ,memory unit and an ALU capable of performing arithmetic and logic operations on its operands, subject to the op-code held in the instruction registers, memory unit will store the data and instructions, program counter will hold the address of the next instruction to be fetched. Registers which contains the operand address. Depending on the control signals produced by the control unit, the operation on the data and instruction should be carried out.

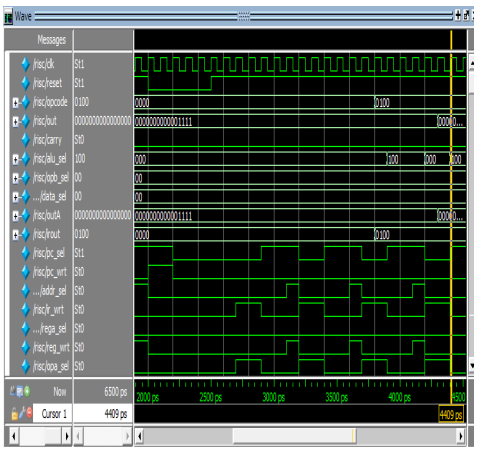

Figure 2: Simulation of RISC Processor

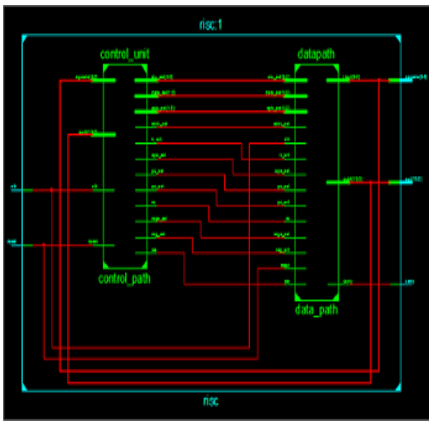

Figure 3: Internal RTL Schematic OF RISC processor

4. Implementation of State Encoding Techniques on Processor

The three state encoding techniques were applied on the RISC processor design and simulated on MODELSIM, synthesized on PRECISION RTL SYNTHESIS so as to get the area and timing (speed) report. The synthesis reports of the RISC processor with the implementation of state encoding techniques are shown below.

4.1 Synthesis Reports

4.1.1 Binary Encoding Technique

a) Area Report

Device Utilization for 3S500EFG320

Resource	*Used*	*Avail*	*Utilization*
IOs	22	232	9.48%
Global Buffers	1	24	4.17%
Function Generators	211	9312	2.27%
CLB Slices	106	4656	2.28%
Dffs or Latches	143	9776	1.46%
Block RAMs	2	20	10.00%
Block Multipliers	0	20	0.00%

b) Clock Frequency Report

Domain Clock	*Name*	*Min Period (Freq)*
ClockDomain0	clk	22.323ns (44.797 MHz)

4.1.2 Gray encoding technique:

a) Area Report

Device Utilization for 3S500EFG320

Resource	*Used*		*Utilization*
IOs	22	232	9.48%
Global Buffers	1	24	4.17%
Function Generators	257	9312	2.76%
CLB Slices	129	4656	2.77%
Dffs or Latches	143	9776	1.46%
Block RAMs	2	20	10.00%
Block Multipliers	0	20	0.00%

b) Clock frequency report

Domain Clock	*Name*	*Min Period (Freq)*
ClockDomain0	clk	18.622 (53.700 MHz)

4.1.3 One-Hot Encoding Technique

a) Area Report

Device Utilization for 3S500EFG320

Resource	*Used*	*Avail*	*Utilization*
IOs	22	232	9.48%
Global Buffers	1	24	4.17%
Function Generators	318	9312	3.41%
CLB Slices	159	4656	3.41%
Dffs or Latches	199	9776	2.04%
Block RAMs	2	20	10.00%
Block Multipliers	0	20	0.00%

b) Clock frequency report

Domain Clock	*Name*	*Min Period (Freq)*
ClockDomain0	clk	16.057 (62.278 MHz)

5. Conclusion

The 16 bit processor was designed and the simulation results for the processor was shown in fig.2 and the RTL schematic is shown in fig.3. The three state encoding techniques binary, gray and one-hot are implemented in the design of processor respectively. The synthesis reports for the target device xc3s500e-4fg320 (SPARTAN 3E) shows that One-hot encoding takes 199 flip-flops, binary & gray coding takes 143 flip-flops each. For One-hot encoding the delay is 16.057ns, corresponds to maximum frequency 62.278MHz,for gray encoding the delay is 18.622ns, corresponds to the maximum frequency 53.70MHz, for binary coding the delay is 22.323ns , corresponds to the maximum frequency of 44.797MHz. From the synthesis reports it is observed that one-hot encoding uses 40% more flip-flops but we can achieve 14% to 28% more speed than gray and binary encoding techniques respectively. The gray encoding require more area than binary encoding technique but it works with high speed, so it is best suitable for PLD & CPLD devices. One-hot encoding is best for FPGA devices because FPGA contains plenty of flip-flops unused.

References

[1] Neeraj Jain, "VLSI Design and Optimized Implementation of a MIPS RISC Processor using XILINX Tool", International Journal of Advanced Research in Computer Science and Electronics Engineering (IJARCSEE), Volume 1 Issue 10, December 2012.

[2] E.V.Nagalakshmi, "Low Power Using Hardware Description Languages for Reduction of Power dissipation in VLSI systems", Research Journal of Engg. And Tech, Vol 5 issue1, ISSN0974-2824.

[3] Bipul C. Paul, Amit Agarwal, Kaushik Roy, "Low-power design techniques for scaled technologies" INTEGRATION, the VLSI journal 39 (2006) 64–89.

[4] N.Siva Sankara Reddy, "Minimization of Power Dissipation in VLSI Circuits using Low Power Techniques", Asian Journal of Applied Sciences, 4(6): 657-662, 2011, ISSN1996-3343.

[5] J.Bhaskar "A Verilog HDL primer", 3rd edition.

[6] Steve Golson, "State machine design techniques for Verilog and VHDL",1994

[7] Doulos, "Verilog Golden reference guide", Version 1.0, August 1996.

Authors Profile

Madhavi Anupoju is studying M. Tech VLSI in MVGR College of Engineering. Vizianagaram District, Andhra Pradesh, India. Her areas of interests are VLSI design, embedded systems and communications.

Dr. M. Sunil Prakash did his B. Tech from Nagarjuna University and M.E from Jadavpur University, Kolkata. He is at present Professor in the Department of Electronics and Communication Engineering and Dean Training & Placements and PG Courses, MVGR College of Engineering, Vizianagaram, Andhra Pradesh. He has done his research under the guidance of Prof. G. S. N Raju in Department of ECE, AU College of Engineering (A). He also presented 7 National and 3 International papers in varies Conferences. He published 4 technical papers in National and International journals.His research interests are Slot antennas, Antenna arrays and EMI/EMC. Dr. Sunil Prakash is also a member of IEEE and life member of IETE, ISTE, IE, ISOI and SEMCE (India).