Design and Simulation Analysis of Timing Instructions for RISC Instruction Set Architecture

Student's Name	S. Deepak	RTES (FT-2012)
Academic Supervisor(s)	P. Padma Priya Dharishini and Sanket Dessai	
Industrial Supervisor(s)		

S. Deepak

Deepaks20288@gmail.com
Ph. No: 0 96636 32074

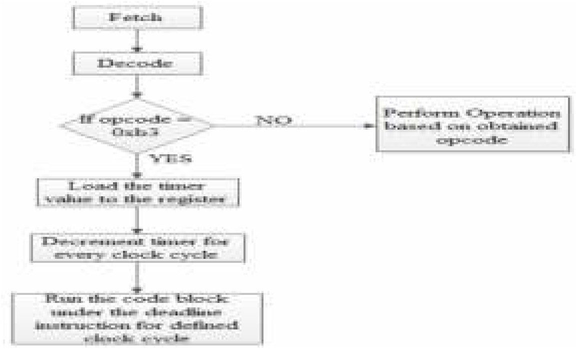
Keywords: Time Predictable Computer Architecture, Deadstart0, Deadend0, Deadloadbranch, Dead

Abstract:

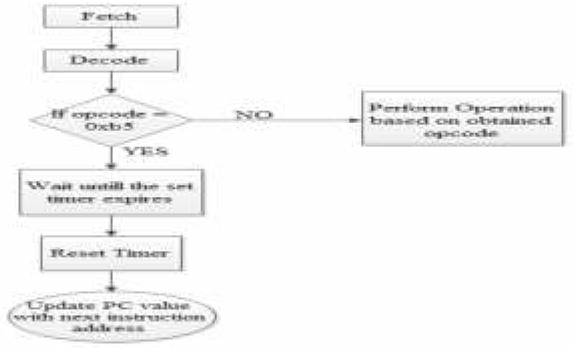
Modern processors use RISC architecture which currently strives for superior average-case performance that regrettably ignores predictability and repeatability of timing properties. The Timing semantics can be defined in Instruction Set Architecture by adding instructions with timing. RISC architecture is not dealing with timing semantics of the system and can be made time predictable by adding timing instructions.

In this project, timing instructions for RISC Instruction Set Architecture is developed and simulated using SimpleScalar Tool Suite. The design of the four timing instruction is carried based on the derived specifications. The developed timing instructions are deadstart0, deadend0, deadloadbranch and dead. The instruction deadstart0 sets the timer and the instruction deadend0 handles the execution until the timer expires by adding delay to the enclosed code. The instruction deadloadbranch sets the timer and checks whether the enclosed code can complete its execution within the set timer. If the enclosed code takes less than the set timer then dead instruction handle the execution time by adding delay. But if the code block enclosed takes more time for execution than defined timer value then the program is aborted.

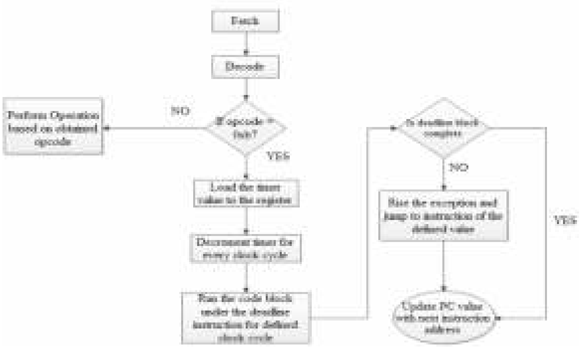
The semantics of the timing instructions are defined in the assembler and simulator of the tool suite and simulated using sim-outorder simulator in SimpleScalar tool suite. It is seen that the timing instructions allow user to set timer values and execute the code sections for the defined value of time. The simulation results show the overhead of clock cycles in simulation results after using timing instructions in application. The simulated instructions can be implemented on the hardware to check the functionality in real time environment.


Fetch
Decode
If opcode = 0xb3
NO
Perform Operation based on obtained opcode
YES
Load the timer value to the register
Decrement timer for every clock cycle
Run the code block under the deadline instruction for defined clock cycle

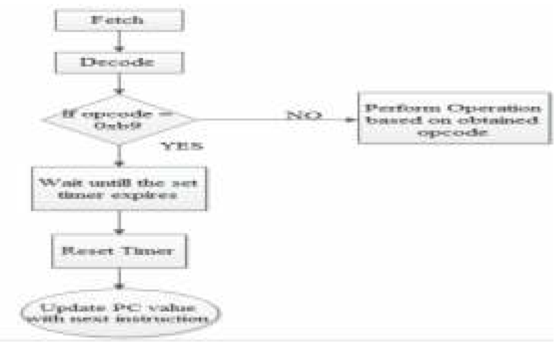
Flow chart of instruction deadstart 0


Fetch
Decode
If opcode = 0xb5
NO
Perform Operation based on obtained opcode
YES
Wait untill the set timer expires
Reset Timer
Update PC value with next instruction address

Flow chart of instruction deadend0


Fetch
Decode
NO
If opcode = 0xb7?
Perform Operation based on obtained opcode
YES
Load the timer value to the register
Decrement timer for every clock cycle
Run the code block under the deadline instruction for defined clock cycle
Is deadline block complete
NO
Raise the exception and jump to instruction of the defined value
YES
Update PC value with next instruction address

Flow chart of instruction deadloadbranch


Fetch
Decode
If opcode = 0xb9
NO
Perform Operation based on obtained opcode
YES
Wait untill the set timer expires
Reset Timer
Update PC value with next instruction

Flow chart of instruction dead