Term Project – Milestone One

Introduction

learn machine language and computer architecture. Through software-based simulation execute machine-language programs on the simulator.

The UVSim can only understand Basic Machine Language.

An accumulator – a register into which information is put before UVSim uses it in calculations or examines it in various ways.

Input is a 4 digit signed decimal number.

UVSim equipped with a 100-word memory, and referenced by their location numbers 00, 01, …, 99.

BasicML program must be loaded into 100-word memory starting 00 before executing.

UVSim is going to simulate Arithmetic Logic Unit (ALU) to let the students learn ALU architecture.

The only operation the ALU can have is XOR and AND operation.

The simulation need to simulate an n bit full adder. Missing …

Each instruction in BasicML occupies one word of the UVSim memory (instruction are signed four-digit decimal number).

We shall assume that the sign of a BasicML instruction is always plus, but the sign of a data word may be either plus or minus. Each location in the UVSim memory may contain an instruction, a data value used by a program or an unused area of memory. The first two digits of each BasicML instruction are the operation code specifying the operation to be performed.

registers.

BasicML vocabulary defined as follow.

I/O operation:

READ = 10 Read a word from the keyboard inter a specific location in memory.

WRITE = 11 Write out a string or integer stored in memory

Load/store operations:

LOAD = 20 Load a word from a specific location in memory into the accumulator.

STORE = 21 Store from accumulator to memory location.

Arithmetic operation:

Add = 30 Add a word from a specific location in memory to the word in the accumulator (leave the result in the accumulator)

SUBTRACT = 31 Subtract a word form a specific location in memory to the word in the accumulator (leave the result in the accumulator)

DIVIDE = 32 Divide a word form a specific location in memory to the word in the accumulator (leave the result in the accumulator)

MULTIPLY = 33 Multiply a word form a specific location in memory to the word in the accumulator (leave the result in the accumulator)

Control operation:

BRANCH = 40 Branch to a specific location in memory

BRANCHNEG = 41 Branch to a specific location on a negative number

BRANCHZERO = 42 Branch to a specific location on zero

HALT = 43

Debugging operation:

MEMDUMP Print the memory image on the screen.

BREAK Break point/ Break out of current run.

CONTINUE Continue with execution of code.

The last two digits of a BasicML instruction are the operand – the address of the memory location containing the word to which the operation applies.

Deliverable

(1) (Hard Copy) Meeting log (5 points)

(2) (Hard Copy) Choose SCRUM and one another process model. (5 points)

(a) Give 3 reasons to support, and 3 reasons to against SCRUM process model.

(b) Give 3 reasons to support, and 3 reasons to against the process model your team chose.

(3) (Hard Copy) Telling the story to all stakeholders by using a Use Case Diagram. The number of use cases is no more than 15 and no less than 10. The range of number is used to control the granularity of the use cases. Try you best to make sure the use cases are in the same detailed level. (10 points)

(4) (Hard Copy of the GUI and running results) Develop a prototype of UVSim by using Visual Studio Online. Screen print the check-in and check-out history. Every team member need to have some record on this history. Note: the prototype can skip the some of the implementation detail, such as the implementation of ALU, the debugging operation. When testing, you can introduce stub in the system. (30 Points)

(5) (Hard Copy) Decompose Milestone to backlogs and sprints. Screen print the backlogs and sprints. (5 points)

(6) Team leader ask for test cases from UVU when you have the first prototype released. If you team leader failed to do so, that means your team did not involve all the stakeholders. And will get zero in the part. (5 points)