Rose-Hulman Institute of Technology
Computer Science and Software Engineering
Prof. Laurence D. Merkle

Term Project Milestone 1 Evaluation

Evaluation Criteria Categories	Specific Criteria	Comments	Score
Consistency with higher level specifications	❑ Given the semantics of the Assembly Language (AL) specification, the sample program can be implemented ❑ Every instruction allowed by the assembly language (AL) specification has a unique machine language (ML) representation o Each instruction type includes enough fields to represent the information specified in the corresponding AL statements o Each field is allocated enough bits to represent all values allowed by the AL specification o For each instruction type, the total number of bits allocated to fields is not greater than the number of bits available ❑ Sample program is translated into binary as described in ML specification	AL appears to be a useful instruction set. ML specifications are provided for AL instructions. ML specification makes bit allocation unclear. Sample program is translated into machine code. ML code should contain AL code in comments.	2/3
Self-consistency	❑ Sample program uses the syntax described in AL specification ❑ Sample program uses the registers described in AL specification (number and type) ❑ Sample program uses the representation given in the ML specification, including correct values for fields specifying branch and jump targets	Sample program uses beq differently than specified; uses unspecified "move" instruction. Sample program uses appropriate registers. Branch / Jump targets do not appear correct. Sample code seems rather short to accomplish goal.	2/3
Demonstration of design principles 1. Simplicity favors regularity 2. Smaller is faster 3. Good design demands good compromises 4. Make the common case fast	❑ AL instructions are easy to understand and are not overly specialized ❑ Number of instructions is minimized ❑ Number of registers is minimized ❑ Where the above criteria conflict, good compromises are made (to make the common case fast) ❑ Number of instruction types is small ❑ Instruction types have regularity	Number of instructions may be slightly excessive. Use of "extra-opcode" fields mitigates this somewhat. Number of registers is reasonable. What advantages does a '1' register have? Does the PC need to be a general register? Second LOADI instruction is the same as first. What does LOAD do to the other 8 bits? Instructions are divided into several instruction types, perhaps a few too many for simplicity	3/3

Evaluation Criteria Categories	Specific Criteria	Comments	Score

Documentation
❑ Organization
❑ Completeness
❑ Conciseness
❑ Grammar and style
• Memo
  • Objective assessment of design and status
• Design Documentation
  • Demonstration of conceptual understanding
  • Highlights interesting features
• Design Process Journal
  • Alternatives considered
  • Tradeoffs
  • Decisions
• Website
	❑ Clear English specifications
	  o Instruction set (incl. prototypical AL statements)
	  o Registers
	  ▪ Number of general purpose registers
	  ▪ Specification of special purpose registers (if applicable)
	  ▪ Naming conventions
	  ▪ Usage conventions
	  o Instruction types
	  o Representation of each instruction
		AL instructions are clearly specified.
		
		Documentation is lacking in several key areas:
		compensation for 8-bit addressing
		no logical operators (are they necessary?)
		what addressing mode or modes do instructions use?
		no explicit register usage conventions
		will a stack be provided?
		
		Website shows some effort.
		
		Design Process Documentation should include more alternatives and reasoning behind decisions.
		
		Jump Return and Chain: overloads jr, interesting idea, What are the advantages of this instruction?
		
		Multiply/Divide are not “general math” for a microprocessor—while you may implement them, they will require significant effort and expense.
			14/16