myMIPS Instruction Set Architecture Documentation

Led Robster

January 15, 2025

Overview

myMIPS is an ISA that imitates the classic MIPS architecture. It is a byproduct of my interest in electronics and software and the result of my learning journey. May it be useful for at least one person other than me.

Instruction Format

myMIPS is a 16-bit architetcure. Instructions are 16 bits wide, data is 16 bits wide. Describe the instruction format, including fields such as opcode, operands, and immediate values. Use a table to illustrate this.

15–12	**11–9**	**8–6**	**5–3**	**2–0**	
opcode	rs	rt	rd/shamt	Fcode	R-format
opcode	rs	rt	offset/immediate		I-format
opcode	address				J-format

Table 1: Instruction Format Example

Formal meaning.
In R-format: rs = alu_op1, rt = alu_op2, rd = dest reg
In R-format SLL or SRL : rs = des reg, rt = alu_op1, shamt = alu_op2
In I-format: rt = dest reg, rs = alu_op1, imm = alu_op2 ;
In I-format BEQ : rs = alu op1, rt = alu op2

Instruction Set

List all instructions, grouped by category (e.g., arithmetic, logical, memory, control). Include opcode, function codes, and descriptions.

Arithmetic Instructions

Instructions involving arithmetic operators +, -.

Format	**Instruction**	**Opcode**	**Fcode**	**Description**	**Example (CASM)**
R	ADD	0000	000	Add rs and rt, store in rd	ADD $rd, $rs, $rt
R	SUB	0000	001	Subtract rt from rs, store in rd	SUB $rd, $rs, $rt
I	ADDI	0000	010	Add rt and immediate, store in rs	ADDI $rs, $rt, d'10

Logical Instructions

Instructions involving logical operators &, |, <<, >>.

Format	Instruction	Opcode	Fcode	Description	Example (CASM)
R	AND	0000	010	Bitwise AND rs and rt, store in rd	AND $r2, $r1, $r3
R	OR	0000	011	Bitwise OR rs and rt, store in rd	OR $r2, $r1, $r3
R	SLL	0000	101	Shift-left-logical rt and shamt, store in rs	SLL $rs, $rt, shamt
R	SRL	0000	110	Shift-left-logical rt and shamt, store in rs	SRL $rs, $rt, shamt

Comparison Instructions

Instructions involving relational operators >, ==.

Instruction	Opcode	Fcode	Description	Example (CASM)
SLT	0000	100	Set rd=1 if rs<rt	SLT $rd, $rs, $rt
SLTI	0011	—	Set rt=1 if rs<IMM	SLTI $rs, $rt, IMM

Control Flow Instructions

List instructions used for branching and jumps, including encoding and behavior.

Instruction	Opcode	Fcode	Description	Example (CASM)
BEQ	0110	—	Branch to PC+OFFSET+1 if rs == rt	BEQ rs,rt, OFFSET
J	0111	—	Jump to ADDR	J h'F
JAL	1000	—	Same as J but PC+1 is stored in $ra	JAL h'F
JR	0000	—	Jump to address in register rs	JR $rs

Memory Instructions

Load/store instructions to access data memory.

Instruction	Opcode	Fcode	Description	Example (CASM)
LW	0100	—	Load RAM[rs+OFFSET] in rt	LW $rs, $rt, OFFSET
SW	0101	—	Store rt in RAM[rs+offset]	SW $rs, $rt, OFFSET

Register Set

Describe the registers available in your ISA, including their names, sizes, and special purposes.

Register Name	Size (bits)	Purpose/Description
r0	32	Always zero
r1-r7	32	General Purpose
x2	32	Stack pointer
r15	32	Return address

Table 7: Register Set

Encoding Examples

Provide encoding examples for a few representative instructions.

Instruction	Opcode	rs1	rs2	Binary Encoding
ADD x1, x2, x3	0110011	00010	00011	0000000 00010 00011 000 00001 0110011
SUB x1, x2, x3	0110011	00010	00011	0100000 00010 00011 000 00001 0110011

Table 8: Encoding Examples

Additional Notes

CASM Immediate : immediates can be written in the following format b'XXX, h'YYY, d'ZZZ. Depends on the number base preferred. b' is for binary format, h' for exadecimal format and 'd for decimal format. Immediates are 6-bits long, writing anything higher than 2**6-1 will result in truncation.
Examples: b'111; d'10; h'3F