Table 1. Core Instruction Set

NAME	MNE- MONIC	FOR- MAT	OPERATION (in Verilog)	OPCODE/FUNCT (Hex)
Add[1]	`add`	R	R[rd]=R[rs]+R[rt]	0/20
Add Unsigned	`addu`	R	R[rd]=R[rs]+R[rt]	0/21
Subtract[1]	`sub`	R	R[rd]=R[rs]-R[rt]	0/22
And	`and`	R	R[rd]=R[rs]&R[rt]	0/24
Or	`or`	R	R[rd]=R[rs]|R[rt]	0/25
Shift Left Logical	`sll`	R	R[rd]=R[rs]<<shamt	0/00
Shift Right Logical	`srl`	R	R[rd]=R[rs]>>shamt	0/02
Set Less Than	`slt`	R	R[rd]=(R[rs]<R[rt])?1:0	0/2a
Add Immediate[1][2]	`addi`	I	R[rd]=R[rs]+SignExtImm	8
Add Immediate Unsigned[2]	`addiu`	I	R[rd]=R[rs]+SignExtImm	9
And Immediate[3]	`andi`	I	R[rd]=R[rs]&ZeroExtImm	c
Or Immediate[3]	`ori`	I	R[rd]=R[rs]|ZeroExtImm	d
Load Upper Immediate	`lui`	I	R[rd]={imm,16’d0}	f
Load Word[2]	`lw`	I	R[rt]=M[R[rs]+SignExtImm]	23
Store Word[2]	`sw`	I	M[R[rs]+SignExtImm]=R[rt]	2b
Branch On Equal[4]	`beq`	I	If(R[rs]==R[rt])PC=PC+4+BranchAddr	4
Jump[5]	`j`	J	PC=JumpAddr	2

Note :

(1) May cause overflow exception

(2) SignExtImm={16{immediate[15]}, immediate}

(3) ZeroExtImm={16{1’b0}, immediate}

(4) BranchAddr={14{imeediate[15]}, immediate, 2’b00}

(5) JumpAddr = {PC[31:28], address, 2’b00}

NAME	**MNEMONIC**	**OPERATION**	**BIT FIELD (Hex)**				
		(in Verilog)	**[31-26]**	**[25-21]**	**[20-16]**	**[15-11]**	**[10-0]**
Move From co-Processor	`mfc0`	R[rt]=CP[rd]	10	0	Rt	Rd	0
Move From co-Processor	`mtc0`	CP[rd]=R[rt]	10	4	Rt	Rd	0
Return from Interrupt Service Routine (ISR)	`eret`	PC=CP[epc]	10	10	0	0	10