R32V2020 Programmer's Reference Card (2019-08-10)

Category	Name	Format	Syntax
System	No Operation	NO_ARGS	`nop`
	Halt and Catch Fire		`hcf`
Arithmetic	Add registers Add immediate	BIN_DEST TBD	`add rd,rs2,rs1` `addi r6,rs,imm16`
	Subtract registers (rd=rs1-rs2) Subtract immediate	BIN_DEST TBD	`sub rd,rs2,rs1` `subi rd,rs,imm16`
	Multiply registers Multiply immediate	BIN_DEST TBD	`mul rd,rs2,rs1` `muli rr,r2,imm16`
Logical	OR registers OR immediate	BIN_DEST TBD	`or rd,rs2,rs1` `ori r6,rs,imm16`
	AND registers AND immediate	BIN_DEST TBD	`and rd,rs2,rs1` `andi r6,rs,imm16`
	XOR registers XOR immediate	BIN_DEST TBD	`xor rd,rs2,rs1` `xori r6,rs,imm16`
Shift	Shift left by 1	UN_DEST	`sl1 rd,rs1`
	Shift left by 8	UN_DEST	`sl8 rd,rs8`
	Shift right by 1	UN_DEST	`sr1 rd,rs1`
	Shift right by 8	UN_DEST	`sr8 rd,rs8`
	Rotate left by 1	UN_DEST	`rol1 rd,rs1`
	Rotate right by 1	UN_DEST	`ror1 rd,rs1`
	Arithmetic Shift right by 1	UN_DEST	`asr rd,rs1`
Compare	Compare Compare Immediate	BIN_CMP TBD	`cmp rs2,rs1` `cmpi rs,imm16`
Swap	Swap Endian	UN_DEST	`ens rd,rs1`
Immediate	Load immediate lower	IMM_DEST	`lil rd,imm16`
	Load immediate upper	IMM_DEST	`liu rd,imm16`
	Load immediate extended	IMM_DEST	`lix rd,imm20`
Load/Stores Data [p] post incr	Load Data Byte	R6_DEST	`ldb[p] rd`
	Load Data Short	R6_DEST	`lds[p] rd`
	Load Data Long	R6_DEST	`ldl[p] rd`
	Store Data Byte	UN_R6_DEST	`sdb[p] rs1`
	Store Data Short	UN_R6_DEST	`sds[p] rs1`
	Store Data Long	UN_R6_DEST	`sdl[p] rs1`
Load/Stores Peripheral [p] post incr	Load Peripheral Byte	R5_DEST	`lpb[p] rd`
	Load Peripheral Short	R5_DEST	`lps[p] rd`
	Load Peripheral Long	R5_DEST	`lpl[p] rd`
	Store Peripheral Byte	UN_R5_DEST	`spb[p] rs1`
	Store Peripheral Short	UN_R5_DEST	`sps[p] rs1`
	Store Peripheral Long	UN_R5_DEST	`spl[p] rs1`
Stack	Push to stack	UN_R4_DEST	`push rs1`
	Pull from stack	R5_DEST	`pull rd`

	Store to stack	UN_R4_DEST	**sss rs1**
	Load from stack	R5_DEST	**lss rd**
Branches	Branch Always	ADDR	**bra addr**
	Branch if equal to zero (ALU)	ADDR	**bez addr**
	Branch if equal to one (ALU)	ADDR	**be1 addr**
	Branch if not zero (ALU)	ADDR	**bnz addr**
	Branch if carry clear (ALU)	ADDR	**bcc addr**
	Branch if carry set (ALU)	ADDR	**bcs addr**
	Branch if less than (cmp)	ADDR	**blt addr**
	Branch if greater than (cmp)	ADDR	**bgt addr**
	Branch if equal (cmp)	ADDR	**beq addr**
	Branch if not equal (cmp)	ADDR	**bne addr**
	Branch to subroutine	ADDR	**bsr addr**

Instruction Format

Format	**D31..D24**	**D23..D20**	**D19..D16**	**D15..D12**	**D11..D00**
ADDR	OPCODE	Sign-Extended Offset (24-bits) *			
BIN_CMP	OPCODE	X	rs2	rs1	X
BIN_DEST	OPCODE	X	rs2	rs1	X
IMM_DEST	OPCODE	rd	Signed-Extended Immed (20-bits) **		
NO_ARGS	OPCODE	X	X	X	X
R4_DEST	OPCODE	rd	X	(r4)	X
R5_DEST	OPCODE	rd	X	(r5)	X
R6_DEST	OPCODE	rd	X	(r6)	X
R7_DEST	OPCODE	rd	X	(r7)	X
UN_DEST	OPCODE	rd	X	rs1	X
UN_R4_DEST	OPCODE	(r4)	X	rs1	X
UN_R5_DEST	OPCODE	(r5)	X	rs1	X
UN_R6_DEST	OPCODE	(r6)	X	rs1	X

* 24-bit range = -8,388,608 to 8,388,607
** 20-bit range = -524,288 to 524,287
*** 16-bit range = -32,768 to 32,767
(rN) = register as pointer to address space

Register Aliases

Special Purpose Registers

r0 = ZERO (0x00000000)	r4 = SAR (Stack Pointer)
r1 = ONE (0x00000001)	r5 = PAR (Peripheral Pointer)
r2 = MINUS 1 (0xFFFFFFFF)	r6 = DAR (Data Pointer)
r3 = Condition Code Register	r7 = PC (Program Counter)

General Purpose Registers

r8 = GP0
r9 = GP1
r10 = GP2
r11 = GP3
r12 = GP4
r13 = GP5
r14 = GP6
r15 = GP7