MISC-V HANDOUT

Reference Sheet

inst	fmt	func	opcode	description
+	R	0000	000	R[rd] = R[rs1] + R[rs2]
-	R	0001	000	R[rd] = R[rs1] – R[rs2]
|	R	0010	000	R[rd] = R[rs1] | R[rs2]
&	R	0011	000	R[rd] = R[rs1] & R[rs2]
+_	I	00	001	R[rd] = R[rs1] + SE(imm)
<<_	I	01	001	R[rd] = R[rs1] << imm
>>_	I	10	001	R[rd] = R[rs1] >> imm
X|_	I	11	001	R[rd] = R[rs1] ^ SE(imm)
<-	M		010	R[rd] = M[R[rs1] + SE(imm)]
->	M		011	M[R[rs1] + SE(imm)] = R[rd]

Y=	Y		100	If(rs1==rs2)
				PC += SE(imm) << 1

Y<	Y		101	If(rs1<rs2)
				PC += SE(imm) << 1

\/	J		110	R[rd] = PC+2
				PC += SE(imm) << 1

/\	J		111	PC = R[rd]

Register Names

Register	Name	Description	Saver
x0	zero	This register is always zero	-
x1	ra	This is the return address	caller
x2	sp	This is the stack pointer	-
x3	at	This is the assembler temporary	-
x4	a0	This is a temporary register that is used for function inputs and function return values	caller
x5	a1		
x6	s0	These are usable saved registers	callee
x7	s1		

Writing Instructions:

Type	Layout
R	rs1 (op) rs2, rd
I	rs1 (op) imm, rd
M	rd (op) rs1+imm
Y	rs1 (op) rs2, imm
J	rd (op) imm

Example Program, Diffsums:

```
Diffsums:   sp - 4, sp

            s0 -> sp + 0

            s1 -> sp + 2

            a1 <- a0 + 0

            s0 <- a0 + 2

            a1 + s0, a1

            s0 <- a0 + 4

            s1 <- a0 + 6

            s0 + s1, s0

            a1 + s1, a0

            s1 <- sp + 2

            s0 <- sp + 0

            sp + 4, sp

            ra /\ 0
```

The above program takes in an array location in memory that holds 4 values, then returns the following:

(m0+m1)-(m2+m3)