Integer Conversion

(2pts) 4. Preform the following conversions. You must show your work to receive credit.

(a) Convert the following signed 8-bit integer to decimal. Assume that the 7th bit (furthest left) is a signed bit such that 1 is negative and 0 is positive.

(.25pts) a.1 $1001\ 1111_2$ $= -(0\times2^6 + 0\times2^5 + 0\times2^4 + 1\times2^3 + 1\times2^2 + 1\times2^1 + 1\times2^0) = (-31)_{10}$

(.25pts) a.2 $0110\ 0110_2$ $= 102\ (2^1 + 2^2 + 2^5 + 2^6)$

(.25pts) a.3 $1010\ 1110_2$ $= -46 = -(2^1 + 2^2 + 2^3 + 2^5)$

(.25pts) a.4 $0000\ 1111_2$ $= 15\ (2^0 + 2^1 + 2^2 + 2^3)$

(b) Convert the following hexadecimal integer to a binary integer.

(.25pts) b.1 EECA 6420_{16}

(.25pts) b.2 AB23 4205_{16}

b1 E E E C A 6 4 2 0

1110 1110 1110 1100 1010 0110 0100 0010 0000

1110 1110 1100 1010 0110 0100 0010 0000

b1 1110 1110 1110 1100 1010 0110 0100 0010 0000

b2 AB23 4205

Same as before we get

1010 1011 0010 0011 0100 0010 0000 0101

RISC-V R-Format and I-Format

(2pts) 5. Convert the following RISC-V instruction into the appropriate R-Format and I-Format shown below. Use the table provided to determine OP codes and funct. codes.

funct7	rs2	rs1	funct3	rd	opcode
7 bits	5 bits	5 bits	3 bits	5 bits	7 bits

Figure 1. R-Format

immediate	rs1	funct3	rd	opcode
12 bits	5 bits	3 bits	5 bits	7 bits

Figure 2. I-Format

[31:25] 7	[24:20] 5	[19:15] 5	[14:12] 3	[11:7] 5	[6:0] 7
function 7	source 2	source 1	function 3	destination	opcode
0000000	xR	xL	000 : ADD	xD	0110011 : OP
0100000	xR	xL	000 : SUB	xD	0110011 : OP
0000000	xR	xL	001 : SLL	xD	0110011 : OP
0000000	xR	xL	010 : SLT	xD	0110011 : OP
0000000	xR	xL	011 : SLTU	xD	0110011 : OP
0000000	xR	xL	100 : XOR	xD	0110011 : OP
0000000	xR	xL	101 : SRL	xD	0110011 : OP
0100000	xR	xL	101 : SRA	xD	0110011 : OP
0000000	xR	xL	110 : OR	xD	0110011 : OP
0000000	xR	xL	111 : AND	xD	0110011 : OP

[31:20] 12	[19:15] 5	[14:12] 3	[11:7] 5	[6:0] 7
IMMEDIATE[11:0]	source 1	function 3	destination	opcode
CONSTANT[11:0]	xL	000 : ADDI	xD	0010011 : OP-IMM
CONSTANT[11:0]	xL	010 : SLTI	xD	0010011 : OP-IMM
CONSTANT[11:0]	xL	011 : SLTIU	xD	0010011 : OP-IMM
CONSTANT[11:0]	xL	100 : XORI	xD	0010011 : OP-IMM
CONSTANT[11:0]	xL	110 : ORI	xD	0010011 : OP-IMM
CONSTANT[11:0]	xL	111 : ANDI	xD	0010011 : OP-IMM

[31:25] 7	[24:20] 5	[19:15] 5	[14:12] 3	[11:7] 5	[6:0] 7
IMMEDIATE[11:5]	IMMEDIATE[4:0]	source 1	function 3	destination	opcode
0000000	SHIFT	xL	001 : SLLI	xD	0010011 : OP-IMM
0000000	SHIFT	xL	101 : SRLI	xD	0010011 : OP-IMM
0100000	SHIFT	xL	101 : SRAI	xD	0010011 : OP-IMM

Figure 3. opcode

(1pts) a. add x9, x20, x21

func 7	rs2	rs1	func 3	rd	op
0000000	10101	10100	000	01001	0110011

(1pts) b. addi x10, x22, 36

immediate	rs1	func 3	rd	op
000000100100	10110	000	01010	0010011

addi

1)

```
mul x23, x20,x21
addi, x24, x22, 10
sub x19, x23, x24
```

2)

```
addi x19, x0, 25
blt x20, x19, Else
addi x21, x0, 35
beq x0,x0, Exit
Else: addi x19, x19, 1
Exit: // continue code
```

3)

```
addi x19, x0,0# initializing i
addi x20, x0, 45 # initializing c
addi x29, x0,7 # control value for loop
loop1:slli   x28, x19, 2 # x28 is temporary  x28 = i * 4
add x28, x25, x28
addi x19, x19,1
addi x20, x20,1
lw x5, 0(x28)
beq x5,x29, loop1
```