DEPARTMENT OF COMPUTER SCIENCE
COLORADOSCHOOLOFMINES EARTH • ENERGY • ENVIRONMENT

CSCI 341: Computer Organization
WS 8: Instruction Formats

1 Why are all the instruction formats somewhat similar?

Solution: to keep the hardware simple

2 What are the basic instruction types? Draw out pictures showing their formats

Solution: R,I,S,U,SB,UJ
(pictures available on quick reference)

3 Convert the following instruction to binary and hex.
lw t0, 32(s3)

immediate(12)	rs1(5)	funct3(3)	rd(5)	opcode(7)
32	19	2	5	3
0000 0010 0000	1001 1	010	0010 1	000 0011
0x0209A283				

4 What range of values can be stored in the I-type immediate value?

-2048 .. 2047 (it is always sign-extended) [-2^11 .. (2^11)-1, it is a 12 bit 2's complement value]

5 What is the range for branch type instructions?

Solution: branch: $+/-2^{10}$ words

6 Convert the following instruction to binary and hex: sw t0, 90(s0)

Solution:
t0 => 5 => 00101
s0 => 8 => 01000
90 => 000001011010
opcode => 0100011
func3 => 010

imm[11:5]	rs2	rs1	func3	imm[4:0]	opcode
0000010	00101	01000	010	11010	0100011

0000 0100 0101 0100 0010 1101 0010 0011
0x04542D23

7 Given the binary format of an instruction as follows

0000 0011 0000 1000 1000 1000 0110 0011

Opcode = 1100011 = SB-type
Imm [4:1|11] = 1000 0
Funct3 = 0x0 => beq
Rs1 = 17
Rs2 = 16
Imm [12|10:5] = 0000 001
Untwisted immediate with hidden bit restored is 0 0 000 001 1000 0 = 0x30

1. What assembly instruction does this correspond to?
    a. Beq (opcode is SB-type and funct3 clarifies it to beq)
2. If the PC = 0X0040 00F4, what is the target address?
    a. 0X0040 00F4 + 0x30 = 0x0040 0124 (0x30 is the untangled immediate value)
3. How many instructions forwards or backwards is that?
    a. 0x30 = 48 / 4 = 12 instructions forward
4. How many bytes is that in decimal?

48

8 This is part of strncpy. What is/are the addressing mode(s) of each instruction?

```
add s0, zero, a1 # q = d;       register (all 3)
add s1, zero, a0 # p = c;       register

# calculate &c[n]
add t1, a0, a2                  register

for_loop:

  lb t0, 0(s1) # t0 = *p        base & register
  sb t0, 0(s0) # *q = t0        base & register
  addi s0, s0, 1 # q++          register & immediate
  addi s1, s1, 1 # p++          register & immediate

  # if (p <&c[n]) goto for_loop;
  blt s1, t1, for_loop          register & PC-relative
```

8 Add the binary numbers shown below without converting to decimal.
0001 1111 0101
0001 0001 0011

Solution:
0011 1110 111 Carry
0001 1111 0101 First number
0001 0001 0011 Second number

0011 0000 1000 Result