Mark Agaton
2013-58222

CoE 113 Lab 2 Documentation

I. Faulty Instructions

SUB: This performs subtraction for the lower 16 bits while for the upper 16 bits, it just copies the upper 16 bits of the first operand ($t1 in the example code below)

```
sub $t0, $t1, $t2
```

SLT: “slt $t0, $t1, $t2” -> When the value of $t2 is greater than 2^15 == 32768 and 0 < $t1 < $t2, the return value of slt is zero.

ADDI: always returns a zero

SLTI: always returns a zero

BEQ: branch is always not taken regardless of the values of the registers

BNE: branch is always not taken regardless of the values of the registers

II. Checking of branch instructions:

For the branch instructions, there are three cases of branch error that we need to check. These three cases are the ff: 1) branch is always taken , 2) branch is always not taken , 3) beq acts as a bne and bne acts as a beq.

Each branch instructions was checked by storing three values at three different memory locations. Using these values and the table below, I determined the type of branch error that occurred.

Case#	Mem[0]	Mem[4]	Mem[8]	Type of branch error
1	0	1	0	Branch is always taken
2	1	1	1	Branch is always not taken
3	0	1	1	Branch inst is doing the opposite (beq -> bne, bne -> beq)

From the output of the data memory, both branch (beq, bne) instructions have an error of case 2 so when using beq or bne, the branch is always not taken.