Shane Cincotta

3/18/19

Lab 1

Benchmark Design

For my calculator, I implemented 5 instructions, *add*, *sub, li*, *compare* and *print.* These instructions fall into one of three categories, R-Format, I-Format or P-Format. The R-Format is used by instructions which do a register-register comparison. The instructions include *add, sub* and *compare.* The I-Format is used by instructions which use immediate values, this pertains to the *li* instruction. The P-Format is a special format used only by the *print* instruction. Each instruction has a unique 2-bit opcode (besides *print* which has a 6-bit opcode).

To test my *add* instruction, I used the following binary command- 00110100. Starting at positions 7/6, I used the 2-bit opcode 00 to denote an add instruction, at position 5/4 I used the 2-bit number 11 to denote the destination register, at position 3/2 I used the 2-bit number 01 to denote the *rs* register and at position 1/0 I used the 2-bit number 00 to denote the *rt* register. Thus, this instruction will add the values in registers 01 & 00 and place the result into register 11.

To test my *sub* instruction, I used the following binary command- 01100001. Starting at position 7/6, I used the 2-bit opcode 01 to denote an *sub* instruction, at position 5/4 I used the 2-bit number 10 to denote the destination register, at position 3/2 I used the 2-bit number 00 to denote the *rs* register and at position 1/0 I used the 2-bit number 01 to denote the *rt* register. Thus, this instruction will subtract the values in registers 00 & 01 and place the result into register 10.

To test my *li* instruction, I used the following binary command- 10110101. Starting at positions 7/6, I used the 2-bit opcode 10 to denote a *li* instruction, at positions 5/4 I used the 2-bit number 11 to denote the destination register, at positions 3-0 I used the 4-bit number 0101 to denote the *immediate*. Thus, this instruction will load the sign extended immediate value into register 11.

To test my *compare* instruction, I used the following binary command- 10110101. Starting at positions 7/6, I used the 2-bit opcode 11 to denote a *compare* instruction, at positions 5/4 I used the 2-bit number 11 to denote the destination register, at positions 3/2 I used the 2-bit number 01 to denote the *rs* register and at positions 1/0 I used the 2-bit number 10 to denote the *rt* register. Thus, this instruction will compare registers *rs* & *rt* and place either a 1 or 0 into register *rd*.

To test my *print* instruction, I used the following binary command- 00000111. Starting at positions 7-2, I used the 6-bit opcode 000001 to denote a *print* instruction, at positions 1/0 I used the 2-bit number 11 to denote the destination register. Thus, this instruction will print to the value in register *rd*.