Homework #1 – 10 points

Due Wednesday, 17 Sep at the beginning of class (or by e-mail before class). Can be turned in up until I hand out the test on Friday, 19 Sep.

You can e-mail me the numerical answers, e.g. “#1 I got 5, #2 I got 7, etc.” and I’ll let you know which ones are right or wrong. This is just for checking your answers, for turning the assignment in you are required to show your work so if you plan to submit by e-mail you can scan your handwritten assignment, use LaTeX, I can send you the OpenOffice source file, etc., just be sure to show your work.

You can always stop by my office hours or e-mail me if you have trouble working the problems.

For numbers 1 through 4 (1 point each), count the number of basic blocks in each snippet of MIPS code and draw lines indicating where the blocks are divided.

#1

```
    move s0, ra
    jal getPid
    bltz v0, exit_prog
    li a0, 24
    li a1, 32
    jal gcd
    move s1, v0
    li a0, 2
    li a1, 3
    jal pow
    bne s1, v0, exit_prog
    move v0, zero
    jr s0
```

#2

```
pow:
     move t1, a0
     li   t2, 1
pow2:
     beq a1, zero, ret1
     beq t2, a1, rett1
     mul t1, a0, t1
     sub a1, a1, t2
     j pow2
rett1:
     move v0, t1
     jr ra
ret1:
     li v0, 1
     jr ra
```

#3

```
gloop1:
     beq t1, zero, ret
     move t2, t1
     div t0, t1
     mfhi t1
     move t0, t2
     j gloop1
```

#4

```
          move $t0,$zero
loop1:    add $t1,$t0,$t0
          add $t1,$t1,$t1
          add $t2,$a0,$t1
          sw $zero,0($t2)
          addi $t0,$t0,1
          slt $t3,$t0,$a1
          bne $t3,$zero,$loop1
```

For numbers 5, 6, and 7, use Amdahl's law. Calculators are okay on the homework, though you should only need one for #6. On the test I'll make sure the fractions work out nicely.

You should use the formula:

OVERALL SPEEDUP = OS = 1/((1 – f) + (f / s))

#5. Suppose you are benchmarking a new type of adder with a benchmark made up of 10000 additions. Suppose also that you find a corner case where you can speed up the adder for 12.5% of those additions, i.e., for most of those 10000 additions the performance stays the same but for the 12.5% that fit my special case you think you can probably make those a lot faster. What is the most overall speedup you could possibly hope for?

#6. Suppose you are designing and benchmarking a DSP processor and multiply/accumulates (MACs) are 40% of your instruction mix measured by clock cycle according to your benchmarks. You make a change to your multiply/accumulate logic that shaves off 5% of your overall execution time for the benchmark. How much faster did you need to make each MAC operation in order to accomplish this?

#7. Suppose you're designing a pipeline for a processor that uses the MIPS ISA. You discover an optimization technique that will cut the number of clock cycles needed for each branch instruction in half. In order to get an overall speedup of 1.10, what percentage of the total clock cycles in your benchmark must be currently being spent on branches?