S-87.3190 Computer Architecture (5 cr) Exam 7.1.2009
Please write clearly. **Voit myös vastata suomeksi.**
Mark your project completion year / Merkitse harjoitustyön suorittamisvuosi

1. a) Explain what is included in an Instruction Set Architecture. (5p)
   b) Consider two competing 5-bit floating point formats. Each contains the same fields (sign, exponent, significand) and follows the same general rules as the 32-bit IEEE standard (denorms, biased exponent, non-numeric values, etc.), but allocates its bits differently. For both, calculate Exponent Bias, Denorm implicit exponent, Number of NANs, Smallest non-zero pos denorm, and Largest non-infinite pos value. (5p)

Implementation "LEFT": `S | EE | FF` | Implementation "RIGHT": `S | EEE | F`

2. a)Explain the meaning of a Direct Mapped Cache. (3p)
   b) Consider a 4-kilobyte direct-mapped cache with a block size of 2 words. Indicate below how many bits of a 32-bit address form the tag, how many form the cache index, and how many form the byte offset (the position in the block). (2p)
   c) Consider now a 8-word direct-mapped cache with 2-word blocks, and suppose that the following sequence of memory accesses is made (e.g. with a sequence of loads) with an initially empty cache. Identify which accesses are hits, which are misses that fill in a block, and which are misses that cause a block to be replaced, by marking each with hit (H), miss (M), or miss with replacement (MR)? (5p)
   Hex byte address sequence: 4512, 4514, 4504, 4501, 4508, 4584, 4518, 4501

3. a) Explain the concept of Pipelining (3p)
   b) Assume the following instruction sequence executes on a pipelined MIPS processor with the solutions to control and data hazards (branch delay slot, load interlock, register forwarding). Determine the number of clock cycles needed to execute each sequence. Count from the cycle to fetch the first instruction through the cycle to completely drain the pipeline of the final instruction. (Drain the pipeline between instruction sequences.)

```
        addi  $1, $0, 2
        add   $0, $0, $0 # assume that $0 can be written
loop:   beq   $1, $0, done
        add   $4, $3, $2
        add   $5, $4, $3
        add   $6, $5, $4
        addi  $1, $1, -1
        beq   $0, $0, loop
        addi  $1, $1, -1
done:   beq   $0, $0, exit
        addi  $1, $0, 3
exit:   addi  $1, $0, 1
```

TURN OVER =>

4. For the two MIPS instructions shown below (instruction format also shown):

`ADDU (Add Unsigned)`

op	rs	rt	rd	shamt	funct
6 bits	5 bits	5 bits	5 bits	5 bits	6 bits

`ADDI (Add Immediate)`

op	rs	rt	Immediate
6 bits	5 bits	5 bits	16 bits

a) Explain the meaning of the instruction fields (3p)
b) Construct a single-cycle 32-bit datapath which can implement the two instructions (include bit-widths, explain control signals) (7p)