Name	Signature	Student Id Nr	*Points*
Model solutions			

581305-6 Computer Organization I, miniexam 2, 18.11.2019 (12 p)

Write your answer on this exam paper in the space given. Please notice, that the exam paper is 2-sided.

a) [1 p] What are the 32-bit Little-Endian *sign-magnitude* representations for integer values +35 and -35?
+35 = 10 0011 = 0x 00 00 00 23, -35 = 0x 80 00 00 23
Little-Endian: +35 = 0x 23 00 00 00, -35 = 0x 23 00 00 80

b) [1 p] What are the 64-bit Big-Endian *twos complement* representations for integer values +35 and -35?
+35 = 10 0011 = 0x 00 00 00 00 00 00 00 23, -35 = 0x FF FF FF FF FF FF FF DD (not … DC)

c) [1 p] What are the 8-bit *biased 127* representations for integer values +2 and -2?
+2 -> 127+2 =129 = 1000 0001 = 0x81, -2 -> 127-2 = 125 = 0111 1101 = 0x7D

d) [1 p] What two advantages do you get, when the bias is specifically 127 with 8-bit biased representation?
127 is almost in the middle of range 0-255, and so there is about equal number of positive and negative numbers. Also, the leftmost bit works now as a sign bit, 1 for positive and 0 for negative numbers.

e) [2 p] What is the IEEE floating point standard normalized Big-Endian 32-bit representation of +5.75?
$5.75 = 101.11_2 = 1.0111 * 2^2$. Exponent = 2 → 2+127 = 129 = 1000 0001.
Mantissa = 011 1000 0000…0. Sign '+' → 0.
+5.755.75: 0 100 0 000 1 011 1000 0000 0000 0000 0000 = 0x 40 B8 00 00.

-- TURN --

f) [2 p] What are the main tasks for operating system? What resources are managed in them?
Process management, memory management, file management, device management (including network management).

g) [2 p] What is a process and what is it's presentation in the operating system?
What is processor context and what data does it contain?
Process is the representation of one program in execution in the system. It is represented by a process descriptor (PCB, process control block, kuvaaja), which contains all data relating to that process (process identifier, CPU priority, processor context, open files, used memory areas, etc).
Processor context contains the values for those CPU registers that must be saved when the process is suspended for a while. When the process resumes execution, processor context registers are recovered and the execution can resume from the interrupted instruction in exactly the same manner as if the execution would not have been interrupted at all.

h) [2 p] The operating system wants to switch the process in execution every time when the executing process has used 20 ms CPU-time (in one turn to execute), even if the process would be able to continue the execution longer. How is this implemented in the system? As an example, use the case where the execution of process P is interrupted and process Q gets it's turn to execute. How is the process switch from process P to process Q implemented?
This requires systems support in the form of clock interrupt. Assume now, that there is a clock interrupt every 20 ms. (If it is every 10 ms, then do the following only every other time, etc.) The clock interrupt handler invokes the scheduler, which will check whether current process (P) has been running for 20 ms. If it has, then it first selects the next process to run (Q) and makes a process switch from P to Q. In process switch, P's processor context is first copied from CPU registers into P's PCB, and P is moved back to Ready-to-Run queue. Then Q is removed from Ready-to-Run queue, and it's processor context is copied to CPU registers. The last register to copy is the PC, and at that instant Q will start to execute.