12 System-on-Chip Design (DJG)

Programmed I/O, also known as memory-mapped I/O, uses a processor's load and store instructions to read and write peripheral registers.

(*a*) How can programmed I/O adversely interact with compiler optimisations and how can this be avoided? [2 marks]

(*b*) How can programmed I/O adversely interact with cache subsystems and how can this be avoided? [2 marks]

(*c*) How might the memory addresses for progammed I/O typically be decided and how are these decisions typically embodied in hardware and in software? [3 marks]

(*d*) Give two code fragments that both implement a pair of registers in a peripheral device that, perversely, when one register is read (e.g. by programmed I/O) it returns the value last written to the other.

(*i*) The first fragment should be for a physical implementation (use synthesisable RTL/SystemC or a schematic diagram), [3 marks]

(*ii*) The second fragment should be for a transactional model. [3 marks]

(*e*) Estimate, to the nearest order of magnitude, how many instructions are needed by the simulator when modelling a single programmed I/O write to a device register using the following two modelling styles:

(*i*) the system bus and the peripheral device modelled as a gate-level net list; [4 marks]

(*ii*) the system bus and the peripheral device hand-coded as a cycle-accurate model. [3 marks]

You may disregard the modelling of the initiating CPU core and answer only for the peripheral device connected by a simple bus. Greater credit will be awarded for arguments supporting your answers than for the figures themselves. State assumptions as to whether the modelling language is interpreted, compiled or otherwise in each case.