Comparative Architectures Supervision Answer (Set 1)

Wei Song 09/02/2016

This answer provides a summary of key points in each question. It is the understanding of the supervisor, which does not necessarily always correct and does not represent the view of lecturers. It is expected to expand the key points into detailed description when similar questions are asked in exams.

Lecture 1. Introduction

Q1: Will we ever converge on a single optimal architecture for all computers? [Sup-work Q 1.2]

No. Computers for different applications have very different requirements, which makes it impossible to have a single optimal architecture which is suitable for all applications.

Q2: Why does power consumption now constrain processor design much more than it has historically? [Sup-work Q 1.3]

1. Although dynamic power is reduced for each generation, supply voltage and interconnect do not scale at the same speed with transistors, slowing down the drop of dynamic power.
2. Leakage power actually increases for each generation due to the drop of supply voltage.
3. The increased transistor density and 3-D technology make heat dissipation more difficult than before.

Q3: Why is MIPS (millions of instructions per second) a very poor measure of processor performance? [Sup-work Q 1.9]

1. The number of instructions is not a good measurement of workload. Work can be done by different ISAs with very different numbers of instructions. (CISC and RISC)
2. Speed performance is not the only metric important for processors.

Q4: (Hard) discuss the important metrics in evaluating the performance of computers targeting different markets, including mobile phones, servers, real-time systems, sensor networks. A non-exhaustive list of available metrics includes: power, energy, MIPS, power efficiency, cost, reliability, flexibility, responding time, etc.

Mobile phone: energy efficiency (longer battery life)

Server: MIPS (high computing power), power efficiency (low operation cost), reliability (less down time), flexibility (customer diversity)

Real-time system: reliability (no down time), responding time (processor should be predictable for worst-case timing calculation)

Sensor network: cost (low cost due to large quantity), energy efficiency (longer battery life), reliability (normally fulfilled by robust software)

Q5: (Hard) how might recent advances in die stacking help to improve microprocessor performance and reduce costs? (2015 Paper 7, Q 5)

1. Die stacking significantly reduces the length of interconnects, which improves speed performance.

2. It also allows dies to use different feature sizes for the best cost reduction and manufacture easiness.
3. The die size of each layer is significantly reduced, which improves manufacturing yield rate.

Lecture 2. Fundamentals of Computer Design

Q6: (Hard) discuss the pros and cons of architectures with a 64-bit word size versus those with a 32-bit size. What applications are likely to benefit most? (2005 Paper 7, Q 1) What hardware extensions are needed for the 64-bit CPU to run a 32-bit application to ensure backward compatibility?

1. Supporting 64-bit word size allows the access to memory > 4GB, which is beneficial for applications requiring large memory.
2. It also reduces the number of instructions and the time to do long calculation, such as double precision floating point and 64-bit integer calculation. It would be helpful for scientific applications need such features.

Extra question: hardware extension

For 32-bit instruction ISAs supporting both 64-bit and 32-bit data:

Special instruction or bit to identify instructions operating 32-bit data. Extra logic in D$ to serve both 64-bit and 32-bit data accesses.

For 64-bit instruction ISAs to support 32-bit instruction:

Special instruction or bit to identify 32-bit instructions. Extension in fetch unit, instruction decoder and PC calculation to support 32-bit instructions. Also extra logic in D$ for 32-bit data accesses.

Q7: (Hard) If you were a processor architect targeting embedded applications where memory is a scarce resource, how might you design a RISC-like instruction set that will achieve efficient use of memory? (2005 Paper 7, Q 1)

Currently there is strong evidence that using shorter instructions for frequent operations can reduce the size of a program. So the common method is profiling the target applications to identify the most utilized operations. Then replace the most utilized operations with shorter instructions, such as 16-bit instructions in a 32-bit ISA.

Reducing the memory need for data is less obvious. Most time it will depend on programmers to write tight programs.

Lecture 3 & 4. Advanced Pipelining

Q8: Provide a case when anti-dependence can cause actual data hazard in a processor.

Quote: “WAR hazards cannot occur in most static issue pipelines” [Hennessy and Patterson]

This is a hazard happens only in out-of-order issue processors and can be resolved by renaming.

Q9: What determines the optimal pipeline length for a microprocessor? (Sup-work Q 3.1)

Optimal pipeline length is a trade-off between clock frequency and stall cost.

Longer pipeline allows higher clock frequency but introduces longer stall penalty.

[side note] As a result, deep pipelined processor must have highly accurate branch predictors, which may significantly increase area.

Q10: How does the existence of multiple parallel pipelines make it more difficult for us to provide support for precise exceptions? (Sup-work Q 3.7)

Precise exception requires all instructions before the excepted instruction to be committed while all instructions after the excepted one to be flushed. For processors using multiple pipelines, it is difficult to figure out the individual instruction on the non-excepted pipeline whether it should be committed or flushed. Extra synchronization or book-keeping logic is required.

Q11: (Hard) discuss the pros and cons of different address formats, from zero-address to three-address format ("specifying an instruction's operands" in Lecture 2). (hint, what is the effect on structure and data hazards?)

From zero to three address formats, implicate register dependency is reduced; therefore, parallelization and speculation become easier. However, the number of read ports to the register file increased, which brings potential structural hazards.

Q12: Why are 2-bit saturating counters often used to predict a branch's direction in preference to single bit schemes? (Sup-work Q 4.3)

It reduces the fluctuation of the miss predication on minor cases.

Q13: Why might a branch target buffer provide a poor prediction of procedure return addresses and what hardware solution may be employed to improve the accuracy of such predictions? (2011 Paper 8, Q 3(a))

BTB is suitable for predicting the target address of a branch that frequently jumps to a single address. Procedure calls hardly return to a single address but the caller addresses. A return address stack can accurately predict the return addresses.

Q14: (Hard) why might a single conditional move instruction be supported rather than the conditional (or predicated) execution of every instruction in the instruction set? (Sup-work Q 4.8)

This is a trade-off between pure predicated execution and branched execution. Predicted execution has the benefits of low instruction count and no jump but processors need to work out whether an instruction should be invalidated according to the condition. Such condition causes implicate dependence, which makes speculation hard. It is normally beneficial to use predicated execution for short code segments rather than long segments. To reach a balance between easier speculation (small forwarding network and simple pipeline control) and lower instruction count (less overhead for miss-prediction as well), some processor may choose to support only conditional move.

[Side note] There are more complicated reasons that even conditional move is dangerous in superscalar processors.