ICT1011: Computer Organisation and Architecture Tutorial

1: Computer Architecture Basics

Q1. Reduced Instruction Set Computing

Consider the following instructions and decide whether they are from a RISC or CISC processor:

a) MPX: Multiply the content of two memory locations, then add the result to an accumulator.

It is RISC because an accumulator is a type of register and RISC instructions operate only on registers.

b) BCDD: Perform a binary-coded decimal division on two registers, format the result in scientific notation and store as ASCII to a memory block ready for display to the screen.

It is CISC as the result is stored to a memory block.

c) ADD: ADD one operand to another and return the result as a third operand. The operands and result are register contents only.

It is RISC because the instructions and result are only stored in the registers.

d) LMUL Rc, Ra, Rb: Perform a 100 clock cycle long multiplication of Ra * Rb and place the result in register Rc.

Since the instruction is more than 100 clock cycle, it is therefore CISC.

Q2. Flynn's classification

Michael Flynn was the first to classify machines based upon how they handle multiple items of data in relation to their instruction words. Consider how he would rate the following (mainly fictitious) types of instruction that different processors and systems might implement:

Q2a) ADD R1, R2, R3 ; perform addition R1 = R2 + R3

Single data stream + single instruction stream, hence SISD.

Q2b) MADD {R0-R7}, {R8-R15}, #0x10 ; R0=R8+16, R1=R8+16, R2=R9+16... R7=R15+16

Multiple instruction stream + single data stream, hence MISD.

Q2c) ADD R0, R1, R2 || SUB R4, R5, R6 ; performs R0=R1+R2, R4=R5-R6 simultaneously

Multiple instruction stream + multiple data stream (simultaneously), hence MIMD.

Q3. Memory Bus Architecture

Fig. Q3 shows three CPUs each of which connect to memory in slightly different ways. Classify CPU A, B and C into von Neumann or Harvard architectures and rank the three in terms of:

CPU A: Von Neumann

CPU B and C: Harvard

(i) system cost,

CPU A is the cheapest (1 memory block and 1 bus), CPU B cheaper than CPU C (2 memory blocks and 1 bus) and CPU C (2 memory blocks and 2 buses) is the most expensive

(ii) device cost,

CPU A is the cheapest (1 memory block and 1 bus), CPU B cheaper than CPU C (2 memory blocks and 1 bus) and CPU C (2 memory blocks and 2 buses) is the most expensive

(iii) ease of programming,

CPU A is the easiest (Von Neumann), CPU C easier than CPU B (because the memory is more directly accessible), CPU B

(iv) CPU pin count requirements,

CPU A requires least number of pins because it manages only one memory interface with shared resources, CPU C lesser number of pins because there are lesser bus than CPU B, CPU B needs the most number of pins.

(v) performance.

Extension: modern PCs are highly graphical. Can you guess which memory connection method is used on most modern PCs to prevent display updates from disrupting normal program memory access?

CPU C offers the best performance because the data stream and address stream are the nearest to the CPU, compared to CPU B. CPU B performance is better than CPU A because it is a Harvard architecture, which allows simultaneous fetching of instructions, which eliminates the Von Neumann bottleneck (CPU must fetch both instruction and data from memory over a common bus).

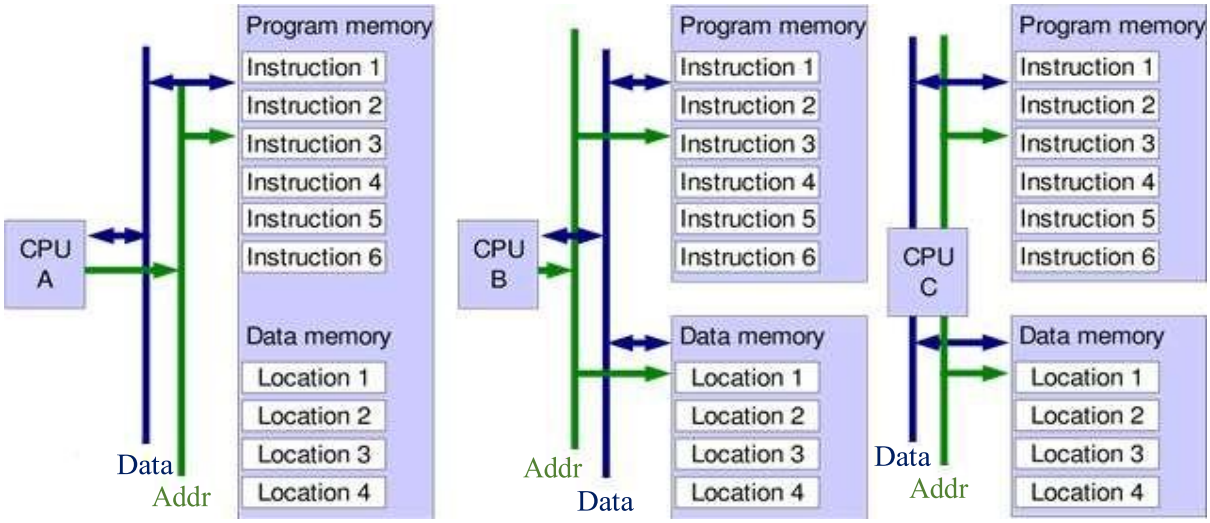


Fig. Q3

Q4. CPU internal bus arrangements

Consider the three types of internal CPU bus arrangements below, all from RISC CPUs. The buses shown in the diagram are used to transfer data. Assume that all the buses are integrated with the appropriate control signals required to enable the data transfers. One clock cycle is required to transfer the contents of one register across the bus. The ALU accepts two input operands and generates one output result. All inputs and output of the ALU are buffered by intermediate registers.

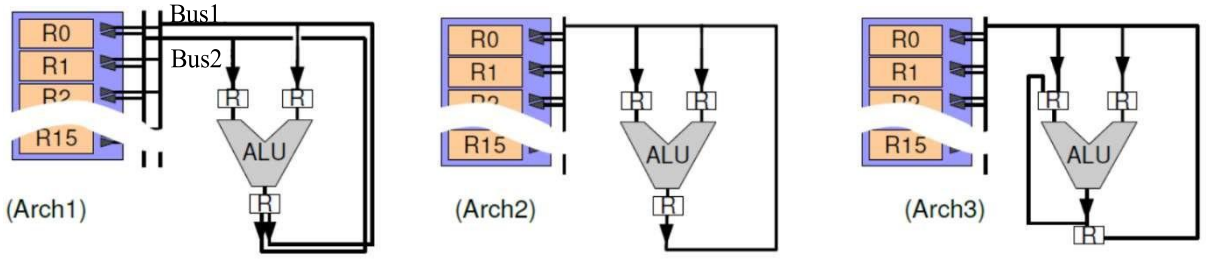
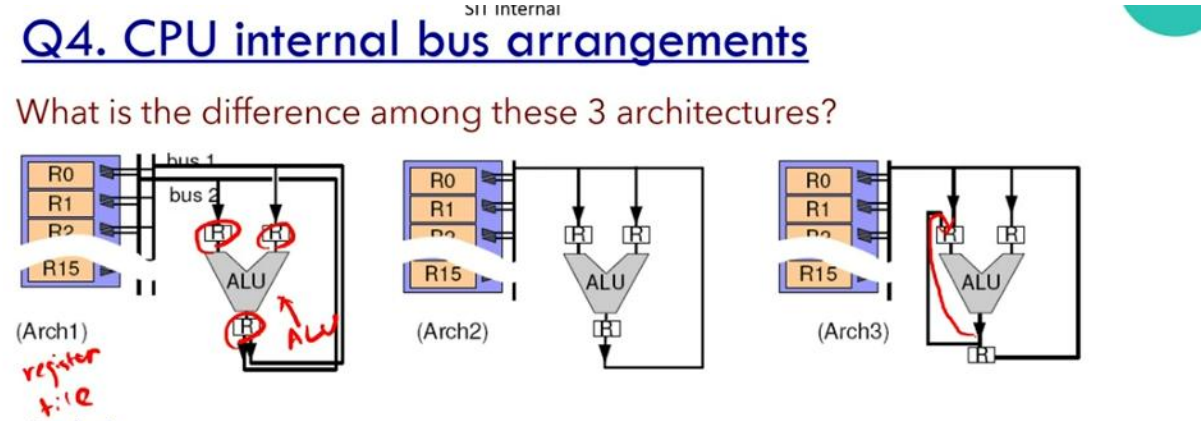


Fig. Q4



Answer the following below, giving a brief explanation to support the reasoning behind your answer:

Q4a) Which of these architectures (if any) can simultaneously transfer two different operands from registers into the ALU during the same clock cycle?

Arch 1 and 3 (result can be sent back)

Q4b) Which of these architectures (if any) can execute R0=R1+R2 in a single clock cycle?

Arch 1,2 and 3 because the instruction is RISC, so single clock cycle.

Q4c) Which of these architectures will be most efficient to perform the following operation:

$$y=\sum_{k=0}^{10} x(k)$$

Assume that the x values are already stored in registers, so that R0 contains x(0), R1 contains x(1)... R10 contains x(10).

Arch 1 because Arch 1 have two buses, which increase bandwidth and reduce latency compared to single bus.