COMP7300 Fall 2021 EXAM1 Review

Updated by **Libo Sun** lzs0101@auburn.edu
SPECIAL Thanks: **Michael Johnson** for providing the most answers!!
Compute Science and Software Engineering ,Auburn University, Auburn, AL, U.S.

Chapter 01 - Introduction

1. *What is Microprogramming?*
   Microprogramming is the technique of making machine instructions generate sequences of microinstructions in accordance with a microprogram rather than initiate the desired operations directly.
2. *What is the function of the control unit?*
   Provides control signals for the operation and coordination of all processor components. The control unit controls the operation of the CPU and hence the computer. The control unit contains Things like sequential logic units, registers and decoders, and memory.
3. *Basic IAS Structure, MAR/MBR*

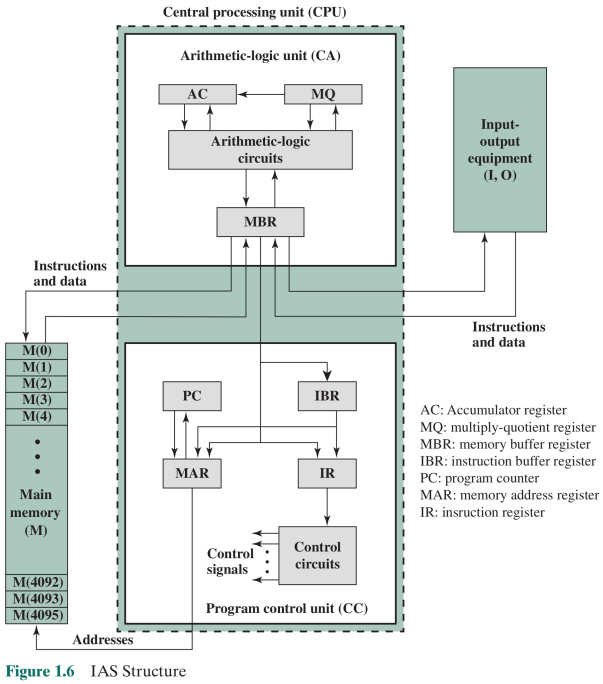

Figure 1.6 IAS Structure

Figure 1: Basic IAS Structure

- The IAS Structure Contains Four Main Components:
  - **A Main Memory**, which stores both data and instructions
  - **An arithmetic and logic unit (ALU)** capable of operating on binary data
  - **A control unit**, which interprets the instructions in memory and causes them to be executed.
  - **Input–output (I/O) equipment** operated by the control unit
- **Memory buffer register (MBR):** Contains a word to be stored in memory or sent to the I/O unit, or is used to receive a word from memory or from the I/O unit.
- **Memory address register (MAR):** Specifies the address in memory of the word to be written from or read into the MBR.

4. *Wafers/Chips*
   A thin wafer of silicon is divided into a matrix of small areas, each a few millimeters square. The identical circuit pattern is fabricated in each area and the wafer is broken up into chips. Each chip consists of many gates and/or memory cells plus several input and output attachment points. The chip is then packaged in a housing that protects it and provides pins for attachment to devices beyond the chip.

5. *Moore's Law*

   - **Moore's Law:** The number of components on chip doubles every 18 months.
   - **The consequences of Moore's Law:**
     - The cost of computer logic and memory circuitry has fallen at a dramatic rate
     - The electrical path length is shortened, increasing operating speed
     - Computer becomes smaller and is more convenient to use in a variety of environments
     - Reduction in power and cooling requirements
     - Fewer interchip connections

6. *What is an embedded system?*
   An embedded system refers to the use of electronics and software within a product, rather than a general-purpose computer. Embedded systems often interact with the outside world.

7. *What is cloud computing?*
   Could Computing is a model for enabling convenient on-demand network access to a shared pool of configurable computing resources that can be rapidly provisioned and released with minimal management effort or service provider interaction.

8. *What is Internet of Things?*
   The IoT is primarily driven by deeply embedded devices that are low-bandwidth, low-repetition data-capture, and low-bandwidth data-usage appliances that communicate with each other and provide data via user interfaces.

Chapter 02 - Performance Issues

1. *Performance Balance*
   An adjustment/tuning of the organization and architecture to compensate for the mismatch among the capabilities of the various components.
   Some example solutions:

- Increase the number of bits that are retrieved at one time by making DRAMs "wider" rather than "deeper" and by using wide bus data paths.
- Change the DRAM interface to make it more efficient by including a cache or other buffering scheme on the DRAM chip.
- Reduce the frequency of memory access by incorporating increasingly complex and efficient cache structures between the processor and main memory.
- Increase the interconnect bandwidth between processors and memory by using higher-speed buses and a hierarchy of buses to buffer and structure data flow.

2. *Wider vs Deeper memories*

   - A "wider" memory means that more bits can be retrieved at once from the memory. This can be achieved by taking the single memory unit and splitting it into two parts. One part is called the odd bank, and the other is called the even bank. All odd addresses are stored in the odd bank, and all even addresses are stored in the even bank. In a narrow memory, one read cycle yields one byte of data. In a wide memory, one read cycle yields two bytes of data. On a wide memory, usually the even address is presented on the address bus and is changed internally to odd for the odd bank, read directly for the even bank. The number of times you can widen the memory is dependent on the size of the data bus.
   - A "deeper" memory means that more bits can be held on the main memory at once.

3. **I/O Data Rates**
   Human interface I/O devices are the slowest, then things like scanners and printers are briefly faster, then hard discs, Wi-Fi, graphics display, and ethernet modem (highest speed) The rates of I/o devices are constantly changing, and we need to take this into account to make computer design a constantly evolving art form.

4. **Power and RC Delays**

   - Power density increases with density of logic and clock speed, thus increasing heat density. Therefore we have fans to dissipate heat on computers.
   - RC delay is the fact that the speed at which electrons flow is limited by resistance and capacitance of the metal wires connecting the devices. As components on the chip increase in size, the wire interconnects become thinner, thus increasing resistance. As the wires become closer together, this increases capacitance.

5. **Amdahl's Law**
   Amdahl's law, first proposed by Gene Amdahl in 1967, is a formula which is often used in parallel computing to predict the theoretical speedup when using multiple processors compared to the single processor.

   - A fraction $(1 - f)$ of the execution time involves code that is inherently sequential
   - A fraction f that involves code that is infinitely parallelizable with no scheduling overhead
   - Let T be the total execution time of the program using a single processor.
   - Then the speedup using a parallel processor with N processors that fully exploits the parallel portion of the program is as follows equation(1):

$$
\begin{aligned}
SpeedUp &= \frac{Time\,to\,execute\,program\,on\,a\,single\,processor}{Time\,to\,execute\,program\,on\,N\,parallel\,processors} \\
&= \frac{T(1-f)+Tf}{T(1-f)+\frac{Tf}{N}} = \frac{1}{(1-f)+\frac{f}{N}}
\end{aligned} \tag{1}
$$

(a) When f is small, the use of parallel processors has little effect.

(b) As N approaches infinity, speedup is bound by

6. **AM, GM, HM**
   Arithmetic Mean, Geometric mean, and Harmonic mean are all part of Pythagorean Means. They offer consistent ways to evaluate performance between different computers.

   - **AM**: Arithmetic mean is good for things like comparing the execution times of different systems.
   - **GM**: Geometric mean is good for comparing normalized metrics.
   - **HM**: Harmonic Mean is good for comparing rates. Such as instruction execution rate.

7. **Benchmarks and their roles**

   - Benchmarks are used to measure the capability of different computer systems. Benchmarks should have these desirable traits:
     (a) It is written in a high-level language. Making it portable across different machines.
     (b) It is representative of a particular kind of programming domain or paradigm such as systems programming numerical programming, or commercial programming.
     (c) It can be measured easily.
     (d) It has wide distribution.
   - SPEC (System Performance Evaluation Corporation) has a benchmark suite to provide a representative test of a computer in a particular application or system programming area.
   - Benchmarks can play huge roles in helping major businesses, or even individual users, decide on which machine is the best for them and their needs. Also, benchmarks can play a large role for research purposes in testing new architectural improvements.

Chapter 03 - Buses

1. **Memory Read/Write cycles**

   - **Memory write**: causes data on the bus to be written into the addressed location.
     For a write operation, the processor puts the data on the data lines at the start of the second cycle, and issues a write command after the data lines have stabilized. The memory module copies the information from the data lines during the third clock cycle.
   - **Memory read**: causes data from the addressed location to be placed on the bus.
     For a read operation, the processor issues a read command at the start of the second cycle. A memory module recognizes the address and, after a delay of one cycle, places the data on the data lines. The processor reads the data from the data lines and drops the read signal.

2. **Figure 3.5 (10th edition)– Example Program Execution**

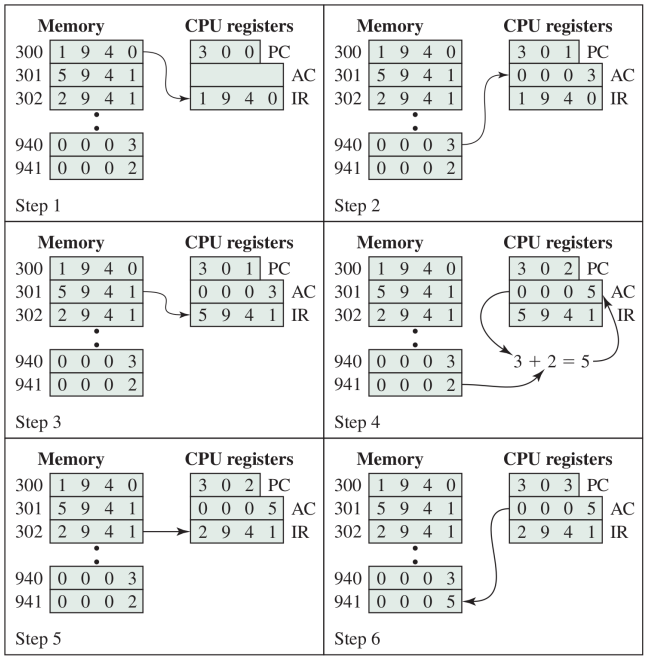

Figure 3.5 Example of Program Execution (contents of memory and registers in hexadecimal)

Figure 2: Example Program Execution

(a) The PC contains 300, the address of the first instruction. This instruction (the value 1940 in hexadecimal) is loaded into the instruction register IR, and the PC is incremented. Note that this process involves the use of a memory address register and a memory buffer register. For simplicity, these intermediate registers are ignored.

(b) The first 4 bits (first hexadecimal digit) in the IR indicate that the AC is to be loaded. The remaining 12 bits (three hexadecimal digits) specify the address(940) from which data are to be loaded.

(c) The next instruction (5941) is fetched from location 301, and the PC is incremented.

(d) The old contents of the AC and the contents of location 941 are added, and the result is stored in the AC.

(e) The next instruction (2941) is fetched from location 302, and the PC is incremented.

(f) The contents of the AC are stored in location 941.

3. **Classes of Interrupts**

- **software(Program**: Generated by some condition that occurs as a result of an instruction execution, such as arithmetic overflow, division by zero, attempt to execute an illegal machine instruction, or reference outside a user's allowed memory space.
- **Timer**: Generated by a timer within the processor. This allows the operating system to perform certain functions on a regular basis.

- **I/O**: Generated by an I/O controller, to signal normal completion of an operation, request service from the processor, or to signal a variety of error conditions.
- **Hardware Failure**: Generated by a failure such as power failure or memory parity error.

4. **When is a check for an interrupt made**
   In the interrupt cycle, the processor checks to see if any interrupts have occurred, indicated by the presence of an interrupt signal.

5. **How are multiple Interrupts handled?**

   - **Disabled Interrupts:** The processor can and will ignore that interrupt request signal. If an interrupt occurs during this time, it generally remains pending and will be checked by the processor after the processor has enabled interrupts. Thus, when a user program is executing and an interrupt occurs, interrupts are disabled immediately. After the interrupt handler routine completes, interrupts are enabled before resuming the user program, and the processor checks to see if additional interrupts have occurred. This approach is nice and simple, as interrupts are handled in strict sequential order.
   - **Nested Interrupts:** This approach is to define priorities for interrupts and to allow an interrupt of higher priority to cause a lower-priority interrupt handler to be itself interrupted. Then current state of Interrupt Service Routine (ISR) for low priority device will be saved on stack and the processor executes the ISR of higher priority device. After completion, the processor executes the ISR of the current device. It can be happening multiple times and the execution sequence is only based on the priority.

6. **Data Bus, Address Bus, Control Bus**

   - **Address bus**: The address lines are used to designate the source or destination of the data on the data bus. The width of the address bus determines the maximum possible memory capacity of the system. Furthermore, the address lines are generally also used to address I/O ports.
   - **Data bus**: The data bus provides a path for moving data among system modules. The data bus may consist of 32, 64, 128, or even more separate lines, the number of lines being referred to as the width of the data bus.
   - **Control bus**: The control bus is used to control the access to and the use of the data and address lines. Control signals transmit both command and timing information among system modules. Timing signals indicate the validity of data and address information. Command signals specify operations to be performed.

7. **Hierarchical Bus Configuration and elements of bus design**

8. **Arbitration - centralized and distributed**

   - In a centralized scheme, a single hardware device referred to as a bus control or arbiter, is responsible for allocating time on the bus. The device may be a separate module or part of the processor.
   - In a distributed scheme, there is no central controller. Rather, each module contains access control logic, and the modules act together to share the bus.

9. **Synchronous and Asynchronous Buses**

- With synchronous timing, the occurrence of events on the bus is determined by a clock. The bus includes a clock line upon which a clock transmits a regular sequence of alternating 1s and 0s of equal duration. A single 1-0 transmission is referred to as a clock cycle or a bus cycle and defines a time slot. All other devices on the bus can read the clock line, and all events start at the beginning of a clock cycle.
- With asynchronous timing, the occurrence of one event on a bus follows and depends on the occurrence of a previous event, not the timer.
- Synchronous timing is simpler to implement and test. However, it is less flexible that asynchronous timing. Because all devices on a synchronous bus are tied to a fixed clock rate, the system cannot take advantage of advances in device performance. With asynchronous timing, a mixture of slow and fast devices, using older and new technology can share a bus.

10. **Multi-core QPI configuration (concepts here, not details)**

    - **Multiple direct connections**: Multiple components within the system enjoy direct pairwise connections to other components. This eliminates the need for arbitration found in shared transmission systems.
    - **Layered protocol architecture**: As found in network environments, such as TCP/IP-based data networks, these processor-level interconnects use a layered protocol architecture, rather than the simple use of control signals found in shared bus arrangements.
    - **Packetized data transfer**: Data are not sent as a raw bit stream. Rather, data are sent as a sequence of packets, each of which includes control headers and error control codes.

11. **PCI and PCI Express (concepts only, not details)**

    - The peripheral component interconnect (PCI) is a popular high-bandwidth, processor-independent bus that can function as a mezzanine or peripheral bus.Compared with other common bus specifications, PCI delivers better system performance for high-speed I/O subsystems
    - PCIe, known as PCI Express (PCIe), as with QPI, is a point-to-point interconnect scheme intended to replace bus-based schemes such as PCI. A key requirement for PCIe is high capacity to support the needs of higher data rate I/O devices, such as Gigabit Ethernet.
    - **3 Layers**: Physical, Data link, Transaction

Chapter 04 - Cache

1. **Access/Cycle/Transfer time**

    - **Access time (latency)**: For random-access memory, this is the time it takes to perform a read or write operation, that is, the time from the instant that an address is presented to the memory to the instant that data have been stored or made available for use. For non-random-access memory, access time is the time it takes to position the read–write mechanism at the desired location.
    - **Memory cycle time**: This concept is primarily applied to random-access memory and consists of the access time plus any additional time required before a second access can commence. This additional time may be required for transients to die out on signal lines or to regenerate data if they are read destructively. Note that memory cycle time is concerned with the system bus, not the processor.

- **Transfer Rate:** This is the rate at which data can be transferred into or out of a memory unit. For random-access memory it is equal to **1/(cycle time)**. For non-random-access memory:

$$T_n = T_A + \frac{n}{R}$$

  - T_n = Average time to read or write n bits
  - T_A = Average access time
  - n = Number of bits
  - R = Transfer rate, in bits per second(bps)

2. **Sequential/Direct/Random Access**

- **Sequential access:** Memory is organized into units of data called records, Access must be made in a specific linear sequence and the access time is highly variable. **Tape units**, e.g., magnetic tape, are sequential access.
- **Direct access:** As sequential access, it involves a shared read-write mechanism.Individual blocks or records have a unique address based on physical location. Access time is variable. **Disk units**, e.g., magnetic disk, CD-ROM, are direct access.
- **Random access:**Each addressable location in memory has a unique, physically wired-in addressing mechanism. The time to access a given location is independent of the sequence of prior accesses and is constant. Thus, any location can be selected at random and directly addressed and accessed. **Main memory and some cache systems**, e.g., DRAM, are random access.

3. **Direct Mapping, Associative Mapping, Set Mapping** - Be able to do problems

(a) **Direct mapping:**

i. The simplest technique, it maps each block of main memory into only one possible cache line. The mapping is expressed as:

$$i = j \mod m$$

where
- i = cache line number
- j = main memory block number
- m = number of lines in the cache

ii. **Advantage:** Simple and inexpensive to implement.

iii. **Disadvantage:** There is a fixed cache location for any given block. Thus,if a program happens to reference words repeatedly from two different blocks that map into the same line, then the blocks will be continually swapped in the cache, and the hit ratio will be low (a phenomenon known as thrashing).

(b) **Associative mapping:**

i. The cache control logic interprets a memory address simply as a Tag and a Word field. The Tag field uniquely identifies a block of main memory. To determine whether a block is in the cache, the cache control logic must simultaneously examine every line's tag for a match.

ii. **Advantage:** With associative mapping, there is flexibility as to which block to replace when a new block is read into the cache.

iii. **Disadvantage:** The complex circuitry required to examine the tags of all cache lines in parallel.

(c) **Set associative mapping:**

i. Set associative mapping is a compromise that exhibits the strengths of both the direct and associative approaches while reducing their disadvantages. In this case, the cache consists of number sets, each of which consists of a number of lines. The relationships are:

$$m = v \times k$$

$$i = j \mod v$$

Where,

- i = cache set number
- j = main memory block number
- m = number of lines in the cache
- v = number of sets
- k = number of lines in each set

This is referred to as k-way set associative mapping. With set associative mapping, block B_j can be mapped into any of the lines of set j.

ii. **Advantage:** Significantly improves the hit ratio over direct mapping. Four-way set associative ($v = m/4$, $k = 4$) makes a modest additional improvement for a relatively small additional cost. Further increases in the number of lines per set have little effect.

iii. **Disadvantage:** The complexity of the cache increases in proportion to the associativity, and in this case would not be justifiable against increasing cache size to 8 or even 16 kB. A final point to note is that beyond about 32 kB, increase in cache size brings no significant increase in performance.

4. **Set-Associative**

5. **Write Thru/Back**

- **Write through**: Using this technique, all write operations are made to main memory as well as to the cache, ensuring that main memory is always valid. Any other processor–cache module can monitor traffic to main memory to maintain consistency within its own cache. This is the **simplest policy**, while the main disadvantage of this technique is that it generates substantial memory traffic and may create a bottleneck.
- **Write back**: With write back, updates are made only in the cache. When an update occurs, a dirty bit, or use bit, associated with the line is set. Then, when a block is replaced, it is written back to main memory if and only if the dirty bit is set. It **minimizes memory writes**, but the problem with write back is that portions of main memory are invalid, and hence accesses by I/O modules can be allowed only through the cache. This makes for complex circuitry and a potential bottleneck.

6. **Unified and split cache decision**

- There are two potential advantages of a unified cache:
  - For a given cache size, a unified cache has a higher hit rate than split caches because it balances the load between instruction and data fetches automatically. That is, if an execution pattern involves many more instruction fetches than data fetches, then the cache will tend to fill up with instructions, and if an execution pattern involves relatively more data fetches, the opposite

  - Only one cache needs to be designed and implemented.
- The key advantage of the split cache design is that it eliminates contention for the cache between the instruction fetch/decode unit and the execution unit. This is important in any design that relies on the pipelining of instructions. Typically, the processor will fetch instructions ahead of time and fill a buffer, or pipeline, with instructions to be executed.

7. **Replacement Algorithms**

   - **LRU**: least recently used, replace that block in the set that has been in the cache longest with no reference to it.
   - **FIFO**: First-in-first-out, replace that block in the set that has been in the cache longest.
   - **LFU**: least frequently used, replace that block in the set that has experienced the fewest references
   - **Random**: A technique not based on usage (i.e., not LRU, LFU, FIFO, or some variant) is to pick a line at random from among the candidate lines.

8. **Victim Cache**:
   One approach to lower the miss penalty is to remember what was discarded in case it is needed again. Since the discarded data has already been fetched, it can be used again at a small cost. Such recycling is possible using a victim cache. Victim cache was originally proposed as an approach to reduce the conflict misses of direct mapped caches without affecting its fast access time. Victim cache is a fully associative cache, whose size is typically 4 to 16 cache lines, residing between a direct mapped L1 cache and the next level of memory

9. **Hit Ratios**:
   Cache hit ratio is a measurement of how many content requests a cache is able to fill successfully, compared to how many requests it receives.

10. **Cost analysis for Cache and Memory**