School of Business Leadership & Enterprise

FdSc Communication Technologies:

Software Engineering and Network Engineering

Level: 4

Module: Computer Platforms

Assessment: Mock TCA

Module Tutor: Dr Nicholas HM Caldwell

Weighting in Module:50%

Hand out: 18th March and 21st March 2014

Hand in: **on or before 18th March and 21st March 2014, noon** [to Assessment Centre, WF 1st floor or SafeAssign  
 Please make sure you obtain and keep a receipt]

|  |
| --- |
| **What is required?**  An electronic copy of your answers uploaded via ‘SafeAssign‘ to the LearnUCS (in a document format readable at UCS, i.e. Word 2003-2010 or PDF ). You should use your UCS userid as the document filename  You may need to copy&paste screenshots and source code from Rolecks or another ARM assembler package into the document |
| **Learning outcomes to be assessed:**  **Learning Outcomes 1, 2, 3, 4 and 5 from Handbook**   1. Identify and explain the purpose and principles of operation of the hardware and software components of computer and microprocessor based systems. 2. Use basic computer/network terminology 3. Describe how data is represented and processed within a computer 4. Describe and understand the core features of a computer operating system, and demonstrate an understanding of scheduling, memory management, file systems and input/output 5. Compare and contrast different ways that different operating systems provide features to users |
| **Graduate Key Skills:**  The assignment can give evidence for GKS: IT1 and IT2 |
| **Assessment & Grading Criteria:**  overleaf |
| **Assessment Brief.**  See page 4 |

**Assessment Criteria**

To achieve a Pass in this Assignment, the stated Pass criteria must be achieved

|  |  |
| --- | --- |
| **Learning Outcomes Assessed in this Assessment** | **PASS criteria** |
| 1. Identify and explain the purpose and principles of operation of the hardware and software components of computer and microprocessor based systems. 2. Use basic computer/network terminology, and perform number conversions/subnet calculations 3. Describe how data is represented and processed within a computer 4. Describe and understand the core features of a computer operating system, and demonstrate an understanding of scheduling, memory management, file systems and input/output 5. Compare and contrast different ways that different operating systems provide features to users 6. Explain the basic components of network architectures and supporting protocols. | In order to be awarded a pass grade (P− or above) for this assignment you must meet the following criteria.   * You must score a minimum of 40% (40 marks) for the 21 questions * You must submit your work: * anonymously (i.e. you must identify yourself using only your student number); * word processed;   with the answers to all questions appropriately identified. |

Grading criteria follow…….

To achieve a higher grade it is the quality of work that will be considered, rather than the amount of work done, and will be assessed against the given criteria:

**Generic Grading criteria for Level 4**

|  |  |
| --- | --- |
| **Good Pass** | In order to be awarded a grade of “Good Pass” (G−, G= or G+)  your score for the 21 questions must greater than or equal to 50% (50 marks) and less than 60% (50 marks) |
| **Merit** | In order to be awarded a grade of “Merit” (M−, M= or M+)  your score for the 21 questions must greater than or equal to 60% (60 marks) and less than 70% (70 marks) |
| **Distinction** | In order to be awarded a grade of “Distinction” (D−, D= or D+)  your score for the 21 questions must greater than or equal to 70% (70 marks) |

**Mock TCA**

**Attempt all questions**

Qu 1: Describe some key differences in terms of hardware and software between computers of the first generation and the second generation. (5 marks)

1st Generation

* + **vacuum tubes for circuitry, magnetic drums, punched cards and paper tape input, printouts for output**
  + **software "machine language" (0s and 1s), one problem at a time, millisecond computation times**
  + **room-sized**

**2nd Generation**

* + **transistors replaced tubes, so computers became smaller, faster, cheaper, more efficient, more reliable**
  + **magnetic core for primary memory, magnetic tapes and disks for secondary storage**
  + **programs written using assembly language (has mnemonics for instructions) and first high-level languages such as COBOL (COmmon Business Oriented Language) and FORTRAN (FORmula Translator) appear**
  + **computation time in microseconds**
  + **software industry is born**

Qu 2: Describe briefly some of the key capabilities provided by the kernel of an operating system (5 marks)

**Enlarge upon:**

**KERNEL provides many internal services (others provided by other modules controlled by kernel)**

**KERNEL manages memory- i.e. locates and allocates space for programs; schedules time for each application to execute, provides communication between executing programs, provides security, control other modules**

Qu 3: Convert the following denary numbers into binary, octal and hexadecimal (6 marks)

A) 75

**1001011, 113, 4B**

B) 494  
**111101110, 756, 1EE**

Qu 4: Convert the following unsigned binary values into decimal (4 marks)

**A) 11010011 = 211**

**B) 11001100 = 204**

**C) 101011001111 = 2767**

**D) 11001101.0110 = 205.375**

Qu 5. Convert the following octal numbers into decimal numbers. (2 marks)

1. 67 = **55**
2. 324 = **212**

Qu 6. Convert the following hexadecimal numbers into decimal numbers. (2 marks)

A) D4A =**3402**

B) 2BC = **700**

Qu 7. Perform the following binary additions (3 marks)

A. 1 0 1 1 0 B. 1 1 0 1 0 1 C. 1 0 0 1 1 0

+ 1 1 0 1 + 1 0 1 1 + 1 1 0

**A: 22 + 13 = 35 = 100011**

**B : 53 + 11 = 64 = 1000000**

**C: 38 + 6 = 44 = 101100**

Qu 9. Perform the following binary multiplications (3 marks)

A. 1 1 1 1 0 B. 1 0 1 0 1 1 C. 1 1 1 0 1 0

x 1 0 0 1 x 1 1 0 1 x 1 1 1

**A: 30 x 9 = 270 = 100001110**

**B: 43 x 13 = 559 = 1000101111**

**C: 58 x 7 = 406 = 110010110**

Qu 10: What is the denary value being stored in the 8 bit binary value 10011111 if the format of the data is a) unsigned binary, b) sign and magnitude, and c) 2's complement? (3 marks)

1. **159 b) -31 c) -97**

Qu 11: Represent -68 in a) 8 bit sign and magnitude, and b) 8 bit 2's complement (2 marks)

1. **11000100**

**b) 10111100**

Qu 12: a. convert the following address and the subnet mask (written in denary) into their equivalent binary bits

b. Perform bitwise logical AND between the two

c. Convert the answer to an address in denary (5 marks)

address: 192.168.75.43

subnet mask: 255.255.000.000

1. **11000000. 10101000. 01001011. 00101011**

**11111111. 11111111. 00000000. 00000000**

1. **11000000.10101000. 00000000. 00000000**
2. **192. 168. 0 . 0**

Qu 13: Assuming the bit patterns below represent 32-bit floating point numbers according to the IEEE standard with a sign bit first, 8 bit exponent and remaining bits as significand, with an emax of +127, convert the numbers to decimal (5 marks)

1. **01111111100000000000000000000000 = Infinity**
2. **11000000011000000000000000000000 = - 3.5**
3. **01000001100000000000000000000000 = 16**

Qu 14: Explain the difference between lossless and lossy compression with two examples of when to use lossless and two of when to use lossy compression (10 marks)

Lossless data compression is a class of data compression algorithms that allows the original data to be perfectly reconstructed from the compressed data. By contrast lossy compression technique permits reconstruction only of an approximation of the original data, though this usually allows for improved compression rates (and therefore smaller sized files).

Two examples: lossless zip for docs, email, code. Lossless tar,

Two examples: lossy picture jpg, sound mp3, etc

Qu 15: Describe the sequence of actions involved in a memory read including details of address valid, R/, and chip select wires (10 marks)

**1. Set the address (of the memory location) on the address bus.**

**2. Set the read/write wire of the control bus high (i.e. request a read operation).**

**3. Set the address valid control wire high.**

**4. The address valid signal, together with the value on the address bus will activate the chip select wire on the appropriate memory chip**

**5. The contents of the memory location will now be placed on the data bus.**

**6. Read the value from the data bus - usually into a register in the microprocessor.**

**7. The read/write, address valid and chip select wires can now all be set low.**

**Picture would be great**

Qu 16; What is a register and give two examples of registers with reference to the ARM architecture? (5 marks)

* **Registers are used as temporary storage for instructions and data within the microprocessor.**
* **In ARM processors:**
  1. **Registers R0 to R14 are 32-bit general-purpose registers. These can be used by programmers for almost any purpose.**
  2. **R15 is the Program Counter and is 32-bits wide.**
  3. **The Current Program Status Register contains conditional flags and other status bits that reflect computational results, e.g. arithmetic overflows. It is 32 bits wide but only the first four and last eight bits are currently used.**
  4. **The address register is an internal 32-bit register which can store either a future Program Counter address (so that the next instruction can be fetched in advance) or the address of a value (an operand needed for a computation).**
  5. **The 32-bit data registers ("data in register" and "data out register") are used to hold data read from memory and data written to memory respectively**

Qu 17: Describe the differences between DRAM and SRAM (5 marks)

* **SRAM is Static RAM, which has:**
  1. **a simple interface, good storage density, speedy access and low power consumption (when not in active use).**
  2. **used for fast cache memories on PC motherboards and mobile phone memories**
* **DRAM Dynamic RAM** 
  1. **has a complex interface because it must have its contents refreshed continuously (as it "forgets" in milliseconds)**
  2. **consumes power even when not in use**
  3. **is slower than SRAM.**
  4. **provides very dense storage**
  5. **frequently used as PC main memory**

Qu 18 Design with flowcharts and then write an ARM assembly language program that adds the numbers 1 to 10, storing the final result in a register (10 marks)

**Flowchart worth half**

**AREA Sum1to10, CODE, READONLY**

**ENTRY**

**start**

**MOV r0, 0x0**

**MOV r1, 0x0**

**Loop ADD r1, r1, 0x1**

**ADD r0, r0, r1**

**CMP r1, 0xa**

**BNE loop**

**SWI 0x11**

**END**

Qu 19 Write an ARM assembly language that loads two numbers from memory into registers, multiplies them together and stores the result out to another location in memory (5 marks)

Anything like

**AREA TwoMul, CODE, READONLY**

**ENTRY**

**start**

**ADR r1, FIRSTNUM**

**ADR r2, SECONDNUM**

**ADR r3, PRODUCT**

**LDR r4, [r1]**

**LDR r5, [r2]**

**MUL r7, r4, r5**

**STR r7, [r3]**

**SWI 0x11**

**END**

**AREA data, DATA, READWRITE**

**FIRSTNUM #7**

**SECONDNUM #6**

**PRODUCT #0**

Qu 20 In pipelining, what are data hazards and branch hazards? (5 marks)

**Data hazards occur when a value required in a register is not valid because a previous instruction has not yet updated. May discuss pipeline stalling, data forwarding, delayed loads**

* **Branch hazards because address of the next instruction is not known until the branch instruction is executed.**
* **In the meantime, if the pipeline is not corrected, the wrong instructions will execute and write their values to registers and memory**
* **Always stall on branches with several clock cycles penalty**
* **Assume branch not taken and so instructions after branch start to execute. If branch is taken, the pipeline is flushed, the partial results discarded and execution resumes from branch target address**
* **Use complex branch prediction heuristics (did it branch before?)**
* **Use delayed branches by having the compiler reorder instructions so that useful work can be done without branch hazard risk**

Qu 21 Describe the differences between direct-mapped, fully associative and set associative caches (5 marks)

* **Direct-mapped cache: For every MM address, there is a single cache location which can hold it. If word is in the cache, then the block containing it is identified by the cache index, found directly from the address.**
* **More flexible approach is fully associative cache which allows a memory block to reside anywhere in cache. Accessing a word entails a full search of cache to see if requested block is present.**
* **A set associative cache is a compromise. It has a number of locations (≥ 2) where block can reside. Each MM block maps to a unique set and all blocks in the set must be searched for requested block and tag field extended to identify each set member**
* **Differences in replacement and write strategies as well.**