|  |  |  |
| --- | --- | --- |
| **PES University Logo.jpg** | **PES University, Bangalore**  (Established under Karnataka Act No. 16 of 2013) | UE18CS253 |
| **Model Question Paper B.TECH. IV SEMESTER**  **UE18CS253- Microprocessors & Computer Architecture** | | |
| Time: 3 Hrs Answer All Questions Max Marks: 100 | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 1. | a) | Differentiate the following   1. Microprocessors and Microcontroller. 2. Computer Organization and Computer Architecture. 3. RISC and CISC.   **Solution:**  Differentiate between   1. Computer Organization and Computer Architecture   Organization: Operational Units and their interconnections. Hardware transparent to the user.  Architecture: Instruction Set, Number of bits to represent the datatypes, clock rate, I/o mechanism.   1. Microprocessor & Microcontroller   Microprocessor: CPU, Does not have RAM, ROM and other peripherals on chip.  Microcontroller: Processor, Memory(RAM, ROM), I/o devices all emded within a chip.   1. RISC & CISC   RISC : Hardwired Control , Pipeline Execution, Higher performance, Single cycle execution, 3 operand instruction format  CISC: Microprogrammed Control , takes many cycles for execution of instructions, 2 operand instruction format | 06 |
| b) | Write a program using ARM ISA to find the length of a string.  **Solution:**  Str: .asciz "hello World"  Ldr R0, =str  Mov R1, #0  Loop: Ldrb R2, [r0], #1  Add R1, R1, #1  Cmp R2, #0  Bne Loop  Sub R1, R1, #1  Swi 0x11 | 03 |
| **c)** | The address for the memory system starts from 1000. It is byte addressable and follows little endian. Show the memory allocated for the following data declaration in ARM memory organization.  a:.halfword 400  b:.byte 40,80,90  c:.halfword 50  d:.halfword 10  e:.word 600  **solution:**   |  |  |  |  |  | | --- | --- | --- | --- | --- | | 31 |  |  | 0 | 1010 | | 600 | | | | 100c | |  |  | 10 | | 1008 | | 50 | |  | 90 | 1004 | | 80 | 40 | 400 | | 1000 | | 05 |
| **d)** | Write the equivalent ARM code snippet for the following C–language statement.  I)  int fun (int I, int j)  {  while (i!=j)  {  if (i>j)  i -= j;  else  j -= i;  }  }  **solution:**  Loop:  CMP R1, R2  SUBGT R1, R1, R2  SUBLT R2, R2, R1  BNE loop  While( i < 5 )  { x[i] = y + i;  i + = 1;  }  **Solution**  .TEXT  LDR R3, =y  LDR R4, =x  LDR R8, [R3]  MOV R6, #0  MOV R5, #0 ;i  LOOP: ADD R6, R8, R5  STR R6, [R4, R5]!  ADD R5, R5, #4  CMP R5, #40  BNE LOOP  EXIT : SWI 0X11  .DATA  y: .WORD 0  x: .WORD 0,0,0,0,0,0,0,0,0,0 | 06 |
|  | | | |
| 2 | a) | Explain pipelining with an example.  **Solution:**  Pipelining - Multiple instructions are overlapped in execution.  - Advantage of parallelism that exists among the actions needed  to execute an instruction.  - In current day processors, pipelining is the KEY implementation feature to make fast processors.  Ex: Assembly line in the construction of an automobile.  - Each step operates in parallel with other steps,  although on a different automobile [instructions]  - In a computer pipeline, each step in the pipeline completes part of the instruction  - Different steps complete different parts of different instructions in parallel.  - Each of these steps is called pipe stage / segment.  - All these stages are connected one to next to form a pipe.  - Instructions enter from one end , Progress through the stages & exit through the other end. | 03 |
| b) | Consider two instructions *i* and *j*, with *i* occurring before *j*. Explain the possible data hazards.  **Solution:**  **RAW** (read after write) - j tries to read a source before i writes it, so j incorrectly gets the old value.  **WAW** (write after write) - j tries to write an operand before it is written by i. The writes end up being performed in the wrong order, leaving the value written by i rather than the value written by j in the destination.  This hazard is present only in pipelines that write in more than one pipe stage or allow an instruction to proceed even when a previous instruction is stalled.  **WAR** (write after read) - j tries to write a destination before it is read by i , so i  incorrectly gets the new value. | 06 |
| c) | How data hazards are minimized using data forwarding in a 5 stage pipeline architecture? Explain with a neat diagram.  **Solution:**  **Data Hazard:**    **Solution:**    Eliminate the stalls for the hazard involving SUB and AND instructions using a technique called ***Data forwarding*** | 04 |
| d) | Consider the following RISC assembly code.    load r1,45(r2) (1)  add r7 <- r1, r5 (2)  sub r8 <- r1, r6 (3)  or r9 <- r5, r1 (4)  brneq r7, target (5)  add r10 <- r8, r5 (6)  xor r2 <- r3, r4 (7)  Identify each dependence; list the two instructions involved; identify which instruction is dependent; and, if there is one, name the storage location involved (register or memory).  **Solution:**     Inst (2) is dependent on inst (1) through r1   Inst (3) is dependent on inst (1) through r1   Inst (4) is dependent on inst (1) through r1   Inst (5) is dependent on inst (2) through r7   Inst (6) is dependent on inst (3) through r8   Inst (6) is dependent on inst (5) through control   Inst (7) is dependent on inst (5) through control | 07 |
|  | | | |
| 3. | a) | A computer has an 8 GByte memory with 64 bit word sizes. Each block of memory stores 16 words. The computer has a direct-mapped cache of 128 blocks. The computer uses word level addressing. What is the address format? If we change the cache to a 4- way set associative cache, what is the new address format?  **Solution:**  With 8 GB and a 64 bit word size, there are 8 GB / (8 bytes / word) = 1 GW of memory. This requires 30 bits for an address. Of the 30 bits, we need 4 bits for the word on the line and 7 bits for the block number, leaving 30 – (7 + 4) = 19 bits for the tag. So the address format is 19 – 7 – 4.  If we have a 4-way set associative cache instead, then there will be 4 sets with 128 / 4 = 32 blocks per set. So we would only need 5 bits for the block number, leaving 30 – (5 + 4) = 21 bits for the tag. So the address format is 21 – 5 – 4. | 08  (4+4) |
| b) | Find the average memory access time for a processor given the following:  - The clock rate is 1 ns  - The miss penalty is 25 clock cycles  - 1% of instructions are not found in the cache.  - 5% of data references are not found in the cache  - 15% of memory accesses are for data.  - The memory system has a cache access time (including hit detection) of 1 clock cycle.  - Assume that the read and write miss penalties are the same and ignore other write stalls.  **Solution:**  AMAT=HT+MRxMP  = 1ns + (0.01\*0.85+0.05\*0.15) x 25ns  **= 1.4ns** | 4 |
| c) | Briefly explain the different write strategies.  • **Write-through.**  All write operations are passed to main memory; if the addressed location is currently held in the cache, the cache is updated to hold the new value. The processor must slow down to main memory speed while the write takes place.  • **Write-through with buffered write.**  Here all write operations are still passed to main memory and the cache updated as appropriate, but instead of slowing the processor down to main memory speed the write address and data are stored in a write buffer which can accept the write information at high speed. The write buffer then transfers the data to main memory, at main memory speed, while the processor continues with its next task.  • **Copy-back (also known as write-back).**  A copy-back cache is not kept coherent with main memory. Write operations update only the cache, so cache lines must remember when they have been modified (usually using a dirty bit on each line or block). If a dirty cache line is allocated to new data it must be copied back to memory before the line is reused. | 06 |
| d) | What is principle of locality of reference ?  **Solution:**   * Temporal locality (locality in time): if an item is referenced, it will tend to be referenced again soon. * Spatial locality (locality in space): if an item is referenced, items whose addresses are close by will tend to be referenced soon. | 02 |
|  | | | |
| 4. | a) | “Multi level cache reduces miss penalty”. Justify the statement with an example.  **Solution**   * All modern computers uses caches to close the gap between memory and CPU. * To reduce the time required to access DRAMs, todays processors support multi level caches. * When a miss occurs, Instead of going to main memory, if the data block is available in the next level of cache, then that reduces the time taken to access the required block.   Example  Image result for multi level cache | 05 |
| b) | Write a note on DMA.  **Solution:**   1. Think about the overhead in both polling and interrupting mechanisms when a large block of data needs to be transferred between the processor and the I/O device. 2. A special control unit may be provided to allow transfer of a block of data directly between an external device and the main memory, without continuous intervention by the processor – direct memory access (DMA). 3. The DMA controller provides the memory address and all the bus signals needed for data transfer, increment the memory address for successive words, and keep track of the number of transfers. 4. Processor sends the starting address, the number of data, and the direction of transfer to DMA controller. 5. Processor suspends the application program requesting DMA, starts DMA transfer, and starts another program. 6. After the DMA transfer is done, DMA controller sends an interrupt signal to the processor.   The processor puts the suspended program in the Runnable state. | 05 |
| c) | Write a note on Flynn’s Classification of parallel computing. And Explain any one classification with an example.  **Solution:**  Parallel systems are more difficult to program than computers with a single processor because the architecture of parallel computers varies accordingly and the processes of multiple CPUs must be coordinated and synchronized.  The crux of parallel processing are CPUs. Based on the number of instruction and data streams that can be processed simultaneously, computing systems are classified into four major categories:    **Single-instruction, single-data (SISD) systems –**  An SISD computing system is a uniprocessor machine which is capable of executing a single instruction, operating on a single data stream. In SISD, machine instructions are processed in a sequential manner and computers adopting this model are popularly called sequential computers. Most conventional computers have SISD architecture. All the instructions and data to be processed have to be stored in primary memory. | 06 |
| d) | Explain the following techniques   1. Polling 2. Daisy Chain Technique   **Solution:**  Polling:Polling is the process where the computer or controlling device waits for an [external device](https://en.wikipedia.org/wiki/External_device) to check for its readiness or state, often with low-level hardware. For example, when a [printer](https://en.wikipedia.org/wiki/Printer_(computing)) is connected via a [parallel port](https://en.wikipedia.org/wiki/Parallel_port), the computer waits until the printer has received the next character. These processes can be as minute as only reading [one bit](https://en.wikipedia.org/wiki/Status_register). Τhis is sometimes used synonymously with [busy-wait](https://en.wikipedia.org/wiki/Busy_waiting) polling  The daisy-chaining method involves connecting all the devices that can  1) request an interrupt in a serial manner.  2) This configuration is governed by the priority of the devices.  3) The device with the highest priority is placed first followed by the  second highest priority device and so on. | 04 |
|  | | | |
| 5. | a) | Write the limitations of serial computing and the applications of parallel computing.  **Solution:**  Limits to serial computing - both physical and practical reasons pose significant constraints to simply building ever faster serial computers.  Transmission speeds - the speed of a serial computer is directly dependent upon how fast data can move through hardware. Absolute limits are the speed of light (30 cm/nanosecond) and the transmission limit of copper wire (9 cm/nanosecond). Increasing speeds necessitate increasing proximity of processing elements.  Limits to miniaturization - processor technology is allowing an increasing number of transistors to be placed on a chip. However, even with molecular or atomic-level components, a limit will be reached on how small components can be.  Economic limitations - it is increasingly expensive to make a single processor faster. Using a larger number of moderately fast commodity processors to achieve the same (or better) performance is less expensive. | 05 |
| b) | What is instruction level parallelism?Consider the following program:  e = a + b  f = c + d  m = e \* f  If we assume that each operation can be completed in one unit of time what is the total amount of time required to execute the above program?  **Solution:**  Instruction-level parallelism (ILP) is a measure of how many of the [instructions](https://en.wikipedia.org/wiki/Instruction_set) in a [computer program](https://en.wikipedia.org/wiki/Computer_program) can be executed simultaneously.  Operation 3 depends on the results of operations 1 and 2, so it cannot be calculated until both of them are completed. However, operations 1 and 2 do not depend on any other operation, so they can be calculated simultaneously. If we assume that each operation can be completed in one unit of time then these three instructions can be completed in a total of two units of time, giving an ILP of 3/2. | 05 |
| c) | What is Cache Coherency? Explain with an example.  **Solution:**    Cache coherence is a concern raised in a multi-core system distributed L1 and L2 caches. Each core has its own L1 and L2 caches and they need to always be in-sync with each other to have the most up-to-date version of the data.  The Cache Coherence Problem is the challenge of keeping multiple local caches synchronized when one of the processors updates its local copy of data which is shared among multiple caches.  Imagine a scenario where multiple copies of same data exists in different caches simultaneously, and if the processors are allowed to update their own copies freely, an inconsistent view of memory can result. | 05 |
| d) | Mention the limitations of ILP  **Solution:**  Factors limiting ILP  • Rename Registers  • Reservation Stations, Reorder Buffer  • Branch Prediction  • Has its own limit  • Memory Aliasing  • Harder to determine  • Window Size  • Limited by the buffer size and basic block size  • Issue Width  • Hardware and Software limitations | 05 |