**UE21CS251B - Microprocessor and Computer Architecture # Hrs - 105**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class #** | **Chapter Title / Reference Literature** | **Topics to be Covered** | **% of portions covered** | |
| Reference Chapter | Cumulative |
| **UNIT 1: Introduction to Microprocessor Architecture &ISA** | | | | |
| 1 | **1.6,2.3 of T2,**  **A-3 of T1,**  **pg no: 51-55 of T2**  **Chapter 3.1 to 3.5 of T3**  **6.8,5.6 of T2** | Introduction and Motivation. How Program Execute? Relationbetween Processor,Operating System, Compiler and Memory.  1.1 Interrupts, Context Switching an overview. 1.2 Classification CISC Vs RISC and Introduction to ARM Processor | 22% | 22% |
| 2 | ARM Processor: Register set, Introduction to ARM ISA and Instruction Layout |
| 3 | Data Processing Instructions: Addition and Subtraction with programming Examples |
| 4 | Data Processing Instruction variants |
| 5 | Data Transfer Instructions: Load and Store with programming examples |
| 6 | Data Transfer instruction and STACK operations |
| 7 | Branch Instructions |
| 8 | Multiplication Instructions and Instruction Encoding |
| 9-10  L1 | Implementation of ARM7TDMI-ISA to Block transfer of data items, Find sum of N data items in the memory. |
| 11-12  L2 | Find the product of two 32bit numbers using barrel shifter |
| 13 | Interrupts and Programming Examples |
| 14-15  L3 | Convert the statement in C language into an ALP.  Find Factorial, GCD, search for an element, sum of n elements in an array using various addressing modes |
| 16  A1.1 | Write a program in ARM7TDMI-ISA to search for an element in an array. Display appropriate messages on the standard output device.For Successful search display as “Successful Search” and if the search is unsuccessful, display as “Unsuccessful Search”.  Use Binary search Technique. |
| 17  A1.2 | Write a program in ARM7TDMI-ISA to find a sub string in a given main string.  Example1: Main string : My name is Bond.  Character : ‘name’.  **Expected Output : “String Present”**  Example2: Main string : My name is Bond.  Character : ‘James’.  **Expected Output : “String Absent”** |
| 18 | Instruction Encoding 1: Data Processing Instruction |
| 19 | Instruction Encoding 2: Data Transfer Instruction |
| 20 | Instruction Encoding 3: Branch and other Instructions |  |  |
| 21 |  | Revision |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **UNIT 2 Pipelining** | | | | |
| 22 | **4.1,4.2 of Text T2**  **Appendix C-1, C-2,**  **Sec 1.1 , 1.4, 1.5 of T1** | Introduction to Pipelining (3 & 5 Stage) | 20% | 42% |
| 23 | Performance Analysis, Speed up Calculations…..etc |
| 24 | Introduction to Pipeline hazards, Structural Hazards |
| 25  A2.1 | Consider the following sequence of instructions in MIPS architecture.  LDR R1, [R2,#40]  ADD R2, R3, R3  ADD R1, R1, R2  STR R1, [R2,#20]  a. Find all dependencies in this instruction sequence.  b. Find all hazards in this instruction sequence for a five stage pipeline with and without data forwarding.  c. Find whether NOPs are required to be introduced inspite of data forwarding in this instruction sequence. |
| 26-27  L4 | Implementation of ARM7TDMI code to generate Fibonacci series, smallest, largest in an array. |
| 28 | Data Hazards 1 |
| 29 | Data Hazards 2 |
| 30 | Control Hazard 1 |
| 31-32 | Control Hazard 2 & Introduction to Branch Prediction |
| 33  A2.2 | Consider the following sequence of instructions in MIPS architecture.  LDR R1, [R6,#40]  BEQ R2, R3, LABEL2 ; BRANCH TAKEN  ADD R1, R6, R4  LABEL2:BEQ R1,R2, LABEL1 ; BRANCH NOT TAKEN  STR R2,[R4, #20]  AND R1, R1, R4  a. Draw the pipeline execution diagram for this code, assuming there are no delay slots and that branches execute in the EX stage.  b. Repeat the exercise mentioned in a and draw the pipeline execution diagram for this code, assuming that delay slots are used by writing a “SAFE INSTRUCTION” in the delay slot. |
| 34-35  L5 | Usage of Multiple Load and Store instructions to perform the parameter passing techniques to a function. |
| 36 | Branch Prediction 1 |
| 37 | Branch Prediction 2 |
| 38 | Branch Prediction 3 |
| 39 | Performance Analysis, Speed up calculations …..etc |
| 40-41  L6 |  | Implementation of Matrix Operations – Addition, Multiplication. |
| 42 |  | Revision |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **UNIT 3: Memory Hierarchy** | | | | |
| 43 | **Appendix B.1, B.2, B.3 of T1** | Introduction to Memory Subsystem, Bottle neck, Memory Hierarchy Introduction to Cache, Locality of reference and Cache Design Philosophy | 20% | 62% |
| 44 | Cache Design Philosophy Continued: Block Placement, Block Identification, BlockReplacement, Read / Write issues with cache |
| 45 | Direct Map Cache Memory |
| 46-47  L7 | Demonstration of MIPS5 simulator to understand pipeline architecture |
| 48 | Set Associative Cache Memory |
| 49 | Fully Associative Cache Memory |
| 50  A3.1 | How many total bits are required for a direct –mapped cache with 16KB of data and 4 word blocks, assuming a 32-bit address? |
| 51 | Page Replacement Algorithms |
| 52 | Read / Write Policy |
| 53-54  L8 | Working with Memory simulator : PARACACHE. |
| 55 | Performance Analysis |
| 56 | 1st Optimization |
| 57 | 1st Optimization&2nd Optimization with examples |
| 58 | 3rd Optimization with examples |
| 59 | 4th Optimization with examples |  |  |
| 60-61  L9 |  | Demonstration of Direct mapping cache and Associative cache. |
| 62  A3.2 |  | Consider a cache with 64 blocks and a block size of 16bytes. To what block number does the byte address 1200map?Assume all are decimal numbers. |  |  |
| 63 |  | Revision |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **UNIT 4 :Memory Optimization Continued &IO Sub system** | | | | |
| 64 | **Appendix B.3 of T1** | 5th Optimization with examples | 16% | 78% |
| 65 | 5th Optimization with examples |
| 66 | 6th Optimization with examples |
| 67 | 6th Optimization with examples |
| 68  A4.1 | Increasing associativity requires more comparators and more tag bits per cache block. Assuming ac cache of 4K blocks, 4 word block size, and a 32-bit address, find the total number of sets and the total number of tag bits for caches that are direct mapped, two-way and four-way set associative , and fully associative. |
| 69-70  L10 | Working with Memory simulator : PARACACHE and  demonstration of Set Associative memory write with all options. |
| 71-72 | Memory Introduction to flash storage, Connecting Processors, and I/O devices. |
| 73 | Interfacing I/O Devices to the Processor |
| 74 | DMA Controller |
| 75 | Memory and Operating System |
| 76 | Examples |
| 77  A4.2 |  | Recall that we have two write policies and write allocate policies, their combinations can be implemented in either in L1 or L2 cache.   |  |  |  | | --- | --- | --- | |  | **L1 cache:** | **L2 cache:** | | a | Write back, write allocate | Write –through, non write allocate | | b | Write back, write no allocate. | Write –through, write allocate |   i. Describe the procedure of handling an L1 write miss, considering the component involved and the possibility of replacing a dirty block.  ii. For a multilevel exclusive cache( a block can only reside in one of the L1 and L2 caches) configuration, describe the procedure of handling an L1 write miss, considering the component involved and the possibility of replacing a dirty block. |
| 78-79  L11 |  | Working with Memory simulator : PARACACHE  DemonstrationFully Associative mapping, memory write with all options. |
| 80-81  L12-P |  | Introduction to Arduino Board: Working of various Sensors with Arduino board. |
| 80 |  | Revision |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **UNIT Advances in ArchitectureoursHour** | | | | |
| 81 | **Sec 1.9 of T1,**  **Sec 3.1, 4.1** | Introduction to Parallel Computing | 22% | 100% |
| 82 | Parallel Computing : Introductory concepts and terminology-Flynn’s taxonomy, |
| 83 | Parallel computing memory architectures, |
| 84 | parallel programming models |
| 85 | parallel examples: matrix multiplication |
| 86 | Amdahl’s Law, Gustafson Law, |
| 87 | Hardware Multi threading |
| 88 | Multi-Core Architecture |
| 89 | Multi-Core Architecture continued.. |
| 90 | Introduction to GPU Computing |
| 91-100  L13-P | Project Work using sensors |
| 101-105 |  | ISA1-ISA5(CBT) |

**Literature:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Book Type | Code | Title & Author | Publication Info | | |
| Edition | Publisher | Year |
| Text Book | T1 | Hennessy Patterson | Fifth Edition | MK Morgan Kaufmann | 2012 |
| Text Book | T2 | ARM System on Chip, Steve  Furber | Second Edition, | Pearson Education | 2000 |
| Text Book | T3 | ARM System Developer's Guide | Reprint 2009 | Elsevier | 2009 |