|  |  |
| --- | --- |
| **Name of the Teacher: Dr. Prasad S.Halgaonkar**  **Class: BE Subject: High Performance Computing**  **AY: 2020-21 SEM: I** | |
| **UNIT-1** | |
|  | Conventional architectures coarsely comprise of a\_\_\_\_\_\_ |
|  | 1. processor 2. Memory system 3. Datapath. 4. All of Above |
| **Ans:** | **d** |
| **Explanation:** |  |
|  | Data intensive applications utilize\_\_\_\_\_\_ |
|  | * + 1. High aggregate throughput     2. High aggregate network bandwidth     3. High processing and memory system performance.     4. None of above |
| **Ans:** | **a** |
| **Explanation:** |  |
|  | A pipeline is like\_\_\_\_\_ |
|  | * 1. Overlaps various stages of instruction execution to achieve performance.   2. House pipeline   3. Both a and b   4. gas line |
| **Ans:** | **a** |
| **Explanation:** |  |
|  | Scheduling of instructions is determined \_\_\_\_ |
|  | 1. True Data Dependency 2. Resource Dependency 3. Branch Dependency 4. All of above |
| **Ans:** | **d** |
| **Explanation:** |  |
|  | VLIW processors rely on\_\_\_\_\_\_ |
|  | * 1. Compile time analysis   2. Initial time analysis   3. Final time analysis   4. Mid time analysis |
| **Ans:** | **a** |
| **Explanation:** |  |
|  | Memory system performance is largely captured by\_\_\_\_\_ |
|  | 1. Latency 2. Bandwidth 3. Both a and b 4. none of above |
| **Ans:** | **c** |
| **Explanation:** |  |
| 7) | The fraction of data references satisfied by the cache is called\_\_\_\_\_ |
|  | 1. Cache hit ratio 2. Cache fit ratio 3. Cache best ratio 4. none of above |
| **Ans:** | **a** |
| **Explanation:** |  |
| 8) | A single control unit that dispatches the same Instruction to various processors is\_\_ |
|  | 1. SIMD 2. SPMD 3. MIMD 4. None of above |
| **Ans:** | **a** |
| **Explanation:** |  |
| 9) | The primary forms of data exchange between parallel tasks are\_ |
|  | * 1. Accessing a shared data space   2. Exchanging messages.   3. Both A and B   4. None of Above |
| **Ans:** | **c** |
| **Explanation:** |  |
| 10) | Switches map a fixed number of inputs to outputs. |
|  | * 1. True   2. False |
| **Ans:** | **a** |
| **Explanation:** |  |
| 11) | The stage in which the CPU fetches the instructions from the instruction cache in superscalar organization is |
|  | a) **Prefetch stage** b) D1 (first decode) stage c) D2 (second decode) stage d) Final stage |
| **Ans:** | **a** |
| **Explanation:** | In the prefetch stage of pipeline, the CPU fetches the instructions from the instruction cache, which stores the instructions to be executed. In this stage, CPU also aligns the codes appropriately. |
| 12) | The CPU decodes the instructions and generates control words in |
|  | * 1. Prefetch stage   2. D1 (first decode) stage   3. D2 (second decode) stage   4. Final stage |
| **Ans:** | **b** |
| **Explanation:** | In D1 stage, the CPU decodes the instructions and generates control words. For simple RISC instructions, only single control word is enough for starting the execution. |
| 13) | The fifth stage of pipeline is also known as |
|  | * 1. read back stage   2. read forward stage   3. write back stage   4. none of the mentioned |
| **Ans:** | **c** |
| **Explanation:** | The fifth stage or final stage of pipeline is also known as “Write back (WB) stage”. |
| 14) | In the execution stage the function performed is |
|  | 1. CPU accesses data cache 2. executes arithmetic/logic computations 3. executes floating point operations in execution unit 4. all of the mentioned |
| **Ans:** | **d** |
| **Explanation:** | In the execution stage, known as E-stage, the CPU accesses data cache, executes arithmetic/logic computations, and floating point operations in execution unit. |
| 15) | The stage in which the CPU generates an address for data memory references in this stage is |
|  | 1. prefetch stage 2. D1 (first decode) stage 3. D2 (second decode) stage 4. execution stage |
| **Ans:** | **c** |
| **Explanation:** | In the D2 (second decode) stage, CPU generates an address for data memory references in this stage. This stage is required where the control word from D1 stage is again decoded for final execution. |
| 16) | The feature of separated caches is |
|  | 1. supports the superscalar organization 2. high bandwidth 3. low hit ratio 4. all of the mentioned |
| **Ans:** | **d** |
| **Explanation:** | The separated caches have low hit ratio compared to a unified cache, but have the advantage of supporting the superscalar organization and high bandwidth. |
| 17) | In the operand fetch stage, the FPU (Floating Point Unit) fetches the operands from |
|  | 1. floating point unit 2. instruction cache 3. floating point register file or data cache 4. floating point register file or instruction cache |
| **Ans:** | **C** |
| **Explanation:** | In the operand fetch stage, the FPU (Floating Point Unit) fetches the operands from either floating point register file or data cache. |
| 18) | The FPU (Floating Point Unit) writes the results to the floating point register file in |
|  | 1. X1 execution state b) X2 execution state c) write back stage d) none of the mentioned |
| **Ans:** | **c** |
| **Explanation:** | In the two execution stages of X1 and X2, the floating point unit reads the data from the data cache and executes the floating point computation. In the “write back stage” of pipeline, the FPU (Floating Point Unit) writes the results to the floating point register file. |
| 19) | The floating point multiplier segment performs floating point multiplication in |
|  | 1. single precision b) double precision c) extended precision d) all of the mentioned |
| **Ans:** | **d** |
| **Explanation:** | The floating point multiplier segment performs floating point multiplication in single precision, double precision and extended precision. |
| **20)** | The instruction or segment that executes the floating point square root instructions is |
|  | 1. floating point square root segment b) floating point division and square root segment c) floating point divider segment d) none of the mentioned |
| **Ans:** | **c** |
| **Explanation:** | The floating point divider segment executes the floating point division and square root instructions. |
| **21)** | The floating point rounder segment performs rounding off operation at |
|  | 1. after write back stage b) before write back stage c) before arithmetic operations d) none of the mentioned |
| **Ans:** | b |
| **Explanation:** | The results of floating point addition or division process may be required to be rounded off, before write back stage to the floating point registers. |
| **21)** | Which of the following is a floating point exception that is generated in case of integer arithmetic? |
|  | a) divide by zero b) overflow c) denormal operand d) all of the mentioned |
| **Ans:** | D |
| **Explanation:** | In the case of integer arithmetic, the possible floating point exceptions in Pentium are: 1. divide by zero 2. overflow 3. denormal operand 4. underflow 5. invalid operation. |

Name and Sign of Subject Teacher