**CS 8339 – COMPUTER SYSTEM DESIGN ASSIGNMENTS**

**Group 1. Digital Logic and Microarchitecture Fundamentals**

**Key Concepts:**

* **Boolean Logic and Logic Gates**:
  + Basics of AND, OR, NOT, NAND, NOR, XOR gates.
  + Truth tables and practical applications.
* **Combinational vs. Sequential Logic**:
  + Differences between the two.
  + Examples of combinational circuits (adders, multiplexers).
  + Examples of sequential circuits (flip-flops, counters).
* **Microarchitecture Basics**:
  + Definition and components (e.g., datapath, control unit).
  + Instruction execution cycle (fetch-decode-execute).
* **Performance Metrics**:
  + Throughput, latency, and clock speed.

**Group 2. Processor Design and Architectures**

**Key Concepts:**

* **Processor Basics**:
  + Von Neumann vs. Harvard architectures.
  + RISC vs. CISC architectures.
* **Instruction Set Architecture (ISA)**:
  + Role of ISA in defining operations, operands, and instruction formats.
  + Examples: x86, ARM.
* **Pipeline Design**:
  + Purpose and stages of pipelining.
  + Hazards (data, control, and structural) and solutions.
* **Advanced Features**:
  + Superscalar execution, speculative execution.
  + Out-of-order execution.

**Group 3. Memory Systems and Cache Design**

**Key Concepts:**

* **Types of Memory**:
  + Volatile (RAM) vs. Non-Volatile (ROM, flash).
  + Hierarchical memory structure (registers, cache, main memory, secondary storage).
* **Cache Design**:
  + Cache levels (L1, L2, L3).
  + Mapping techniques (direct-mapped, fully associative, set-associative).
  + Cache performance: hit rate, miss penalty.
* **Virtual Memory**:
  + Concept and benefits.
  + Paging and segmentation.
* **Memory Access Optimization**:
  + Prefetching, write-back vs. write-through policies.

**Group 4. Input/Output Systems and Peripherals**

**Key Concepts:**

* **I/O Basics**:
  + Role of I/O in system design.
  + Types of I/O devices (input, output, storage).
* **I/O Communication**:
  + Programmed I/O, interrupt-driven I/O, and Direct Memory Access (DMA).
  + Buses and protocols (e.g., PCI, USB).
* **Storage Devices**:
  + HDD vs. SSD.
  + RAID configurations and their use in data protection.
* **Device Drivers**:
  + Role in communication between OS and hardware.
  + Basic concepts of driver design.

**Group 5. Power Management and Energy-Efficient Design**

**Key Concepts:**

* **Power Management Basics**:
  + Power vs. energy consumption.
  + Dynamic power vs. static power dissipation.
* **Energy-Efficient Techniques**:
  + Clock gating, dynamic voltage, and frequency scaling (DVFS).
  + Power-aware scheduling and workload distribution.
* **Low-Power Architectures**:
  + Design considerations for mobile and embedded devices.
  + Examples of energy-efficient processors (e.g., ARM Cortex-M).
* **Thermal Management**:
  + Heat generation in CPUs.
  + Cooling techniques and thermal design.

**Group 6. Parallelism and Multicore Design**

**Key Concepts:**

* **Parallelism Basics**:
  + Data-level parallelism (SIMD) vs. task-level parallelism (MIMD).
  + Amdahl’s Law and its implications.
* **Multicore Architectures**:
  + Advantages over single-core designs.
  + Challenges in multicore programming (e.g., synchronization, communication).
* **Performance Metrics**:
  + Speedup, scalability, and efficiency.

**Group 7. Security in Computer System Design**

**Key Concepts:**

* **Security Fundamentals**:
  + Importance of secure system design.
  + Common threats (e.g., side-channel attacks, buffer overflows).
* **Hardware Security**:
  + Trusted Platform Module (TPM).
  + Hardware-based encryption.
* **Processor Security Features**:
  + Examples: Intel SGX, AMD SEV.
* **Memory Security**:
  + Memory protection techniques (e.g., Address Space Layout Randomization).
  + Secure boot and firmware security.
* **Designing for Security**:
  + Secure-by-design principles.
  + Balancing security with performance.

**General Assignment Presentation Tips:**

* **Define Key Concepts**: Start with definitions and importance.
* **Use Real-World Examples**: Relate concepts to practical applications (e.g., ARM processors for mobile devices).
* **Visual Aids**: Include diagrams, charts, and flowcharts to explain technical details.
* **Comparison Tables**: Use tables to compare architectures, techniques, or algorithms.