The University Of Sheffield.

Electronic & Electrical Engineering.

EEE6207 ADVANCED COMPUTER SYSTEMS

Credits: 15

Course Description including Aims

This module looks at modern computer systems from operating systems down to the underlying computer architectures to provide a coherent view of how such systems work and how their performance can be improved, looking, in particular, at parallelism

1. To provide students with an understanding of the structure of modern multi-tasking operating systems.
2. To identify the links between systems and the underlying architectures.
3. To equip the student with an understanding of high-performance processing system architectures.
4. To introduce the concepts of parallelism as a means of enhancing system performance.

Outline Syllabus

System Organisation. Historic introduction and origins of the operating system. Elements of the multi-user operating system - Hardware considerations, memory protection, system mode operation, time slicing. **Processes:** Scheduling, process synchronisation, inter-process communication. **Threads:** Comparison with heavyweight processes. **Deadlocks:** detection, avoidance and recovery. **File systems. Memory Systems :** virtual memory, memory hierarchies, locality, coherence, address translation, paging/segmentation, memory management. **Processor Classifications**. **Pipelining:** speed-up constraints, static/dynamic pipelines, reservation tables and collision vectors. **Pipelined Processors:** architecture, constraints, interlock and register forwarding/scoreboarding, stalling, branching, superscalar, VLIW. **Parallel Processors:** array processors, loosely-coupled processors, tightly-coupled processors, vector processors. **Connection Networks :** structures, complexity, performance, limitations, memory organisation and interleaving, multi-processing caches. **Task partitioning :** compute/IO bounds.

Time Allocation

36 lectures plus 18 hours of additional support material.

Recommended Previous Courses

EEE336 “Digital Design”

Assessment

2 hour examination, answer 3 out of 4 questions (75%) and two pieces of courswork (each worth 12.5%)

Recommended Books

Silberschatz, Galvin & Gagne	*Operating System Concepts*	Addison-Wesley
Hennessy & Patterson	*Computer Architecture - A Quantitative Approach*	Morgan Kaufmann

Hwang & Briggs	*Computer Architecture and Parallel Processing*	McGraw-Hill
Stone, H.S.	*High Performance Computer Architectures*	Addison-Wesley
Van de Goor, A.J.	*Computer Architecture and Design*	Addison-Wesley

Objectives

On successful completion of this module students will be able to

1. Describe the key elements and functionality of a modern computer operating system.
2. Demonstrate an understanding of computational processes and their interaction, particular process interactions and synchronisation.
3. Demonstrate an appreciation of the issues surrounding the management of resources such as: memory, disk space and the CPU.
4. Understand the interaction between an operating system and the underlying hardware, and the hardware extensions which facilitate key functionality in a modern operating system.
5. Identify the crucial importance of memory system design in high performance computer systems,
6. Analyse/estimate the performance of a given memory system and to evaluate the design options and alternatives.
7. Identify appropriate forms of parallelism (temporal and spatial) for particular problems/application, to analyse the effect on system performance and evaluate the costs/benefit.
8. Describe various structures and issues which are important to the design of parallel processors, identifying their strengths and weaknesses and to analyse the likely performance of key components.

Detailed Syllabus

1. Historic introduction and origins of the operating system: OS functions - user interface, Management of physical resources, Hardware/software co-evolution, Development of multi-user concept.
2. Multi-user systems: Overview of memory management, processes and process scheduling security. Preemptive and non-preemptive multitasking.
3. Other computing paradigms: Parallel processors, distributed processing, real-time systems.
4. Hardware aspects: Interrupts, System calls, Memory protection, Mode - bits - system and user modes, Rings, Time-sharing implementation.
5. OS functions. Process management, memory, I/O, protection, kernels.
6. Virtual machines: Example of MS-DOS processes under Windows NT.
7. Processes: Properties and status, process control blocks, context switching, child processes.
8. Threads: Comparison with heavyweight processes, advantages and disadvantages.
9. Interprocess Communication: Shared memory, message passing, pipes.
10. CPU Scheduling: Burst duration model, preemptive vs. non-preemptive scheduling, scheduling algorithms - FCFS, SJF, prioritisation, round-robin, multi-level prioritisation.
11. Process Synchronisation: Critical sections, semaphores, 'wait-spinning' vs. blocking, atomic instructions. The bounded buffer problem.
12. Deadlocks: Characteristics of deadlocks. Resource allocation graphs. Handling deadlocks avoidance, prevention recovery.
13. File Systems: Files as abstract entities, physical organisation of disks, disk partitions, logical vs. physical records, file operations, directory structures, file handles, security and access, sequential and random access, clusters, disk fragmentation.
14. Real-time operating systems. Scheduling.
15. Memory systems, virtual and physical memory, technology / speed / volume / cost trade off,

memory hierarchy.

16. Address translation, look up tables, CAM
17. TLB, limitations, sparse tables / TLBs.
18. Temporal and spatial locality, coherence, block transferring (for performance), attribute information.
19. Paging and segmentation, relationship to programs, memory management (LRU, LFU, FIFO, RAND, MIN), pre-fetching, thrashing. A practical paging system.
20. Pipelining, concepts, speed-up, reservation-tables and collision vectors, dynamic pipelines.
21. Pipelined processors, internal organisation and relationship to RISC processors (load-and-store architecture), examples.
22. Read/Write interlocks, pipeline stalls, scoreboarding
23. Register forwarding.
24. Branching instructions, choice, branch-history caches, delayed jumps.
25. RISC processors, high-level support, architectures, registers, optimising compilers.
26. Superscalar and VLIW processors.
27. Loosely-coupled processors, message-passing paradigm, local memory, hyper-cubes.
28. Tightly-coupled processors, organisation, processing paradigm, connection network.
29. Memory modules, high and low interleaving, home modules, multi-processor caches (coherency, ownership, and snooping).
30. Communication Networks : non-blocking,
31. Communication Networks: blocking (e.g. Banyan, Delta).
32. Network complexity issues and performance.
33. Task partitioning, degree of parallelism, sequential / parallel - Amdahl's rule, communication / computational bounds on performance, example calculation.

UK-SPEC/IET Learning Outcomes

Outcome Code	Supporting Statement
SM1m/fl	Moore's Law, VLSI design. Quantitative analysis of performance of systems. Stress on economic drivers in the development of OSs. Evolution of technologies as agents of change in Oss. Assessed by examination.
SM2fl	Moores Law, technological development and roadmaps. Assessed by examination and as part of the coursework submissions.
SM4m	Moore's Law, technological development and roadmaps. Assessed by examination and as part of the coursework submissions.
SM2p	Problems in modelling complex system performance often require approximation/linearisation, probabilities, permutations/combinations. The ability to perform general algebraic manipulations and to understand numbering systems/arithmetic are considered. Assessed by examination.
SM3m	The design and organization of computer systems is fundamentally dependent on implementation limitations and capability derived from VLSI design. The systematic integration of OS features. Assessed by examination and as part of the coursework submissions.
EA1m/fl	This course is fundamentally associated with some of the descriptors associated with the architecture, performance and organization of systems for different domains, operating systems, distributed computing, and computational methods. Considers systematic integration of OS features.

	Assessed by examination.
EA2m	Many of the aspects of the course are concerned with examining the trade-offs and limits of system design: cost .v. performance .v. complexity. Assessed by examination and as part of the coursework submissions.
EA3m	Various scenarios are presented and, in some cases, analyses are presented to allow students to understand and apply a 'principled' approach to computational system design as well as the heuristics that often underpin design.. These are supported by a mathematical treatment of a number of different cases. Assessed by examination.
EA4p	The emphasis in the course is the various limitations and costs and design options available to designers to allow them to make rational decisions about system design underpinned by either mathematical analyses (where tractable) or empirical studies (only described). Assessed by examination and as part of the coursework submissions.
D1p	Considers economic versus performance drivers. Assessed by examination.
D4p	The course looks at the critical relationship between system design/organization and fitness-for-purpose in specific domains. Assessed by examination.
D5m	The relationship between design choices (made early on in the design process) and their effect on the outcome – associated with the costs of making poor decisions and not adopting a well-thought-out methodology. This is addressed partly via an individual assignment.
D6m	Students are expected to complete an assignment on a topic within the subject area but beyond the scope of the teaching.
EP2p	Performance issues studied extensively. Assessed by examination.
EP4p	The SIA roadmap, appropriate research papers. Assessed as part of the corsework submission.