CE/CZ3001 – Advanced Computer Architecture

Course Code	CE3001/CZ3001
Course Title	Advanced Computer Architecture
Pre-requisites	CE1006/CZ1006
No of AUs	3
Contact Hours	TEL(LAMS) 10 Lecture 26 Tutorial 10 Laboratories 5

Course Aims

This course aims to develop your understanding in the fundamental concepts and in-depth understanding of design principles of a computer system by addressing key issues in instruction set design, micro-architecture design along with the interaction of hardware components in a computer system. This course aids you to acquire necessary skills for analysing and estimating the performance of computing systems, and to design high-performance computing systems. In this course, there is a strong emphasis on the study of techniques for improving the power efficiency of single and multiple processor systems. Students will complete this course with a useful appreciation and understanding of processor design issues relating to simplicity of implementation, performance-enhancement techniques, and power-reduction methods.

Intended Learning Outcomes (ILO)

By the end of this course, you (as a student) would be able to:

1. Interpret the performance of a processor based on metrics such as execution speed and power.
2. Design processors that achieve the desired performance in a step by step manner.
3. Design and describe pipeline data-path for performance enhancement.
4. Predict the challenges of realizing different kinds of parallelism (such as instruction, data and thread level) and leverage them for performance advancement for the present, and in the future.
5. Design cache that takes into account the importance of efficient memory design and virtual memory to overcome memory wall.
6. Develop high performance programs by taking into consideration data-path, memory design and parallelism at instruction, data and thread level.

Course Content

	Topics	LAMS (Hours)	Lectures & discussion (Hours)	Tutorial (Hours)
1	**Introduction and Background:** Review of basic computer architecture, Technology trend and design goals, Performance metrics and performance enhancement techniques, Key concepts of parallel processing and pipelining, Power dissipation in processors, power metrics, and low-power design techniques.	1	4	2
2	**Instruction set architecture design:** Instruction set design: implementation and performance perspectives, relative advantages of RISC and CISC instruction sets. Relation of ISA to assembly, and to compiler, Instruction formats and addressing modes.	1.5	3	2
3	**Micro-architecture design:** Single and multi-cycle data path design. Hardwired and micro-programmed control design. Pipeline data-path.	1.5	4	1
4	**Memory Systems and I/O design:** Memory hierarchy, Cache design considerations, instruction vs. data caches, write-policy and replacement policy, analysis of cache performance, and cache design for performance enhancement, memory technologies (SRAM, DRAM, and flash memory), Virtual memory, TLB.	2	5	2
5	**Instruction-Level Parallelism:** Concept and examples of data-dependence, Challenges in ILP realization, Pipeline hazards and their solutions, Data forwarding, Register renaming, Reordering of instructions, Out-of-order execution, Branch prediction, dynamic scheduling, Limitations of scalar pipelines, VLIW and superscalar processors, Instruction, data and memory-flow challenges in superscalar and out-of-order processors.	2	5	2
6	**Data-Level Parallelism:** Introduction to vector architecture, SIMD instruction set extensions, GPU architecture.	0.5	2	1
7	**Thread-Level Parallelism:** Motivation for multicore and many-core systems, Amdahl's law under power constraint, Challenges in efficient multi-core system design, Cache coherence problem.	0.5	2	0
8	**Emerging Computing Trends:** Application specific architectures: ASIP. FPGA and ASIC. Heterogeneous multicore platform. Introduction to domain-specific computing. Comparison of performances and power consumption of general purpose processors, DSP, GPU, FPGA, and ASIC.	1	1	0
		=10	=26	=13

Assessment

a) Final Examination: 60%
b) Laboratory quizzes: 30%
c) Laboratory Assignment: 10%

CZ 3002 – Advanced Software Engineering

Course Code	CZ3002							
Course Title	Advanced Software Engineering							
Pre-requisites	CZ/CE2006: Software Engineering							
No of AUs	3							
Contact Hours	Lectures	26	TEL	0	Tutorials	13	Student presentations	2(lab)

Course Aims

Building upon software engineering and design concepts covered in Software Engineering (CZ2006), this course aims to introduce advanced software engineering management topics to you including Quality Management, Project Management, Configuration Management, and Maintenance, according to the Software Engineering Body of Knowledge.

Intended Learning Outcomes (ILO)

This course introduces software at an advanced level. Upon the successful completion of this course, you (as a student) shall be able to:

1. Describe the software engineering management and quality processes, their purpose and importance
2. Apply advanced software engineering management techniques
3. Use appropriate methods and tools for the development and management of real world reliable software systems
4. Conduct yourself in an ethically responsible way expected of software professionals to ensure public safety

Course Content

	Topics	Lectures (Hours)	Tutorials (Hours)	Presentations (Hours)
1	**Introduction** Case studies to illustrate the impact of software on businesses and public well-being.	3	1	3
2	**Software Quality Management** Quality models and characteristics: software product and process quality. Quality management processes overview: quality assurance.	3	1.5	3
3	**Project Management** Project planning: SDLC and planning. Estimation: COCOMO, Scheduling, Critical Path Analysis. Risk management.	7	3.5	7
4	**Agile Methods** Agile Manifesto, Principles, Methods	1	1	1
5	**Configuration Management** Configuration item identification: hardware and software, versions and baselines. Configuration control: change management process. Release management and delivery: build automation.	4	2	6
6	**CMMI** CMMI process models, Industry standards.	2	1	2
7	**Maintenance** Nature of Maintenance. Categories of Maintenance. Maintainability: reuse, design patterns, frameworks. Software evolution: re-engineering, reverse engineering.	3	1.5	3
8	**Software Testing** Test objectives: acceptance, installation, performance, stress testing. Test-driven development. Regression testing. Test automation.	3	1.5	3
	Check for Hours	=26	=13	=28

Assessment (includes summative assessment)

a) Final 2-hour written examination: 40%
b) Coursework assessment in a form of laboratory cum assignments: 60%

CZ3005 - Artificial Intelligence

Course Code	CZ3005										
Course Title	Artificial Intelligence - Problem Solving and Knowledge and Reasoning										
Pre-requisites	NIL										
Pre-requisite for	NIL										
No of AUs	3										
Contact Hours	Lecture	12	TEL	14	Tutorials	13	Laboratories	1	Example Class	2	

Course Aims

Computer / software engineers are involved in effective and efficient building of knowledge agent systems that satisfy the requirements of users; possibly software for intelligent embedded and intelligent information systems. General awareness of theory, knowledge, and practice in all phases of the knowledge based systems and representation techniques for problem solving are necessary for those of you who wants to get into the field artificial intelligence. These are advanced intelligent systems that are finding wide spread applications in finance, banking, manufacturing industries.

Intended Learning Outcomes (ILO)

Upon completion of this introductory course, the student should be able to:

1. Explain what Artificial Intelligence is about and appreciate its relevance to Computer Science and importance for IT and society.
2. Describe the human cognitive organisation in problem solving and appreciate the ethics involved in the application of AI techniques.
3. Formulate a problem, evaluate its complexity and apply the appropriate state space search algorithms to solve it
4. Build knowledge-based systems using techniques relating to computer-based representation and manipulation of complex information, and formal logic inference and reasoning algorithms.
5. Discuss several advance AI applications and topics such as several advanced AI applications and topics such as intelligent agents, constraint satisfaction, game playing, and applied expert systems.

Course Content

	Topics	Lectures (Hours)	TEL (Hours)	Tutorials (Hours)
1	Human brain and Cognitive structure, Thinking and acting, Foundations of AI, AI in Engineering, AI in Society	1	1	0
2	Agent paradigm, Agent tasks and environments	1	1	2
3	Procedural Representation (Algorithmic): -Problem formulation, State space representation, -Uninformed search (breadth-first, depth-first, IDA, uniform-cost) -Informed search (best-first, A*), Heuristics functions -Constraint satisfaction problems -Adversarial search (Minimax) for Game Playing	5	5	5
4	Symbolic Representation (Knowledge Engineering): -Reasoning agents, Knowledge representation and inference, -Propositional logic, Modus ponens – inference engine, Formalizing knowledge,	2	1	2
5	-First order logic, Unification, Generalized Modus ponens – inference engine, semantic network -Production rule system, conflict resolution, Knowledge engineering, design guideline – expert system	2	3	4
6	AI in the Real World, Case studies of intelligent systems (medical diagnosis, transportation, intelligent tutoring, business agents), Future of AI	1	0	0
7	Other methods (uncertainty in knowledge and reasoning, acting under uncertainty, certainty factors, Fuzzy logic)	1	3	0
	Check for Hours	=12	= 14	=13

Assessment (includes both continuous and summative assessment)

a) Final Examination: 60%
b) Quiz: 10%
c) Laboratory: 30%

CZ3006 – Net Centric Computing

Course Code	CZ3006							
Course Title	Net Centric Computing							
Pre-requisites	CE/CZ1006: Computer Organization & Architecture CE/CZ2002: Object Oriented Design & Programming							
No of AUs	3							
Contact Hours	Lectures	26	TEL	0	Tutorials	13	Laboratories	8

Course Aims

This course aims to give a broad knowledge of modern networking technologies and network-based applications, computing systems, and software. The course will cover the background and history, basic concepts and components, mechanisms and protocols of computer networks and Internet. The scope will extend to the World Wide Web computing and information exchange framework built on top of Internet, and introduce key technologies that enable the client–server web application modes. You are expected to finish the course with necessary knowledge and understanding of the rationale in modern computer networking and network centric system and application design.

Lectures will focus on concepts, theory, technology, and practical examples. We will supplement lectures with tutorials, laboratory experiments, course assignments, and e-learning modules. E-learning modules are intended to encourage self-learning, particularly on extended knowledge of the core of this course. Tutorials will focus on topics and knowledge covered in the lectures for you to better understand them. Lab experiments and assignments will enhance your practical programming experience and hands on skills

Intended Learning Outcomes (ILO)

This course introduces a broad knowledge of the structure, implementation, and theoretical underpinning of modern networking technologies and network based applications. Upon completion of this course, you shall be able to:

1. Identify and describe the basic concepts, reference models, and protocols of modern computer networks.
2. Illustrate from layered point of view, the fundamental components and mechanisms that internet is operated on.
3. Identify a set of World Wide Web technologies and applications that enable efficient client-server interaction and information exchange.
4. Describe and analyse practical implementation considerations when designing and developing network based systems and applications.
5. Implement with practical considerations some of the network modules as well as web based systems and applications

Course Content

	Topics	Lecture Hours	Tutorial Hours
1	**Introduction to net-centric computing** Background and history of networking and the Internet, network reference models and architectures, example networks, and network-based applications	2	1
2	**The physical layer and data link layer** Communication technology, packet and circuit switching, error control and flow control, sliding window protocols.	3	2
3	**The MAC Layer and Local Area Networks** Multiple access protocols, CSMA/CD and Ethernet, wireless communication networks.	3	2
4	**The network layer and Internet IP protocols** Network layer service and design issues, routing algorithms, congestion control, internetworking, Internet Protocol (IPv4 and IPv6), Internet control protocols.	3	2
5	**The transport layer and Internet TCP protocols** Transport layer service v.s. protocols, connection establishment, use and release, Internet User Data Protocol (UDP) and Transmission Control Protocol (TCP).	3	2
6	**Web architecture and protocols** Background and fundamentals of Internet and World Wide Web, including web browsers, web servers, and information exchange protocols such as MIME, HTTP, etc.	2	1
7	**Web documentation technologies** Web content and documentation technology and tools, including XML, HTML, and etc.	3	1
8	**Client application programming techniques** Client side dynamic programming and event handling technologies, and how they can be integrated and interacted with web documentation tools.	3	1
9	**Server application programming techniques** Server side programming principle and technologies, including basic scripting syntax, form data processing, file handling, as well as interaction with client side scripts.	4	1
		=26	=13

Assessment (includes both continuous and summative assessment) CZ3006

a) Final Examination: 60%
b) Lab Assignment I: 20%
c) Lab Assignment II: 20%

CZ3007 – Compiler Techniques

Course Code	CZ3007							
Course Title	Compiler Techniques							
Pre-requisites	CE/CZ2001: Algorithms CE/CZ2006: Software Engineering							
No of AUs	3							
Contact Hours	Lectures	26	TEL	0	Tutorials	13	Laboratories	8

Course Aims

The aim of this course is to give you an understanding of compilers and the techniques involved in programming language translation. Compilers are omnipresent in the life of a software developer. Their design and implementation have been studied for more than 50 years, leading to a well-developed body of theory that has been successfully applied in practice many times over. Many techniques originally developed in the context of compiler design (such as regular expressions, parsing and abstract syntax trees) have taken on a life of their own and are widely used in other contexts as well.

While it is not likely that you will ever be involved in the development of a compiler for a general programming language during their professional career, domain specific languages are becoming increasingly important. Implementing an interpreter or a compiler for such a language requires an understanding of compiler techniques. Consequently, it is important that you have at least a high-level grasp of the structure of a compiler and the problems arising during compilation, and that you acquire the most important techniques used to tackle them. Last but not least, a compiler is a non-trivial piece of software; understanding its structure and inner workings is an excellent way for students to become better software engineers.

Intended Learning Outcomes (ILO)

Upon the successful completion of this course, you shall be able to:

1. Identify the steps and techniques involved in programming language translation.
2. Use regular expressions and context free grammars to describe languages, and employ open-source tools to create recognisers for them.
3. Explain the concept of abstract syntax trees and techniques for name binding analysis and type checking.
4. Explain how a compiler generates machine code, and use simple data flow analysis techniques for code optimisation.
5. Apply ideas, techniques, and skills learnt to general software design.

Course Content

	Topics	Lectures (Hours)	Tutorials (Hours)
1	**Introduction to Compilers** Compilers vs. interpreters; components and phases of a compiler; compilation to native code vs. compilation to bytecode.	1	0
2	**Lexical Analysis** Regular languages and regular expressions; finite automata; deterministic and non-deterministic finite automata; lexical analyser generators.	3	2
3	**Parsing** Context-free languages and grammars; recursive descent parsing; LR parsing: LR(0), LALR(1); error recovery; parser generators.	5	3
4	**Semantic Analysis** Abstract syntax trees; abstract grammars, attribute grammars; scope checking; type checking.	5	2
5	**Code Generation** Intermediate languages; run-time storage organisation; stack vs. heap; code generation for stack machines; code generation for physical machines; register allocation.	8	4
6	**Optimisation** Intra-procedural optimisation; data flow analysis; simple local optimisations: liveness analysis, common subexpression elimination; overview of inter-procedural optimisation.	4	2
	Check for Hours	=26	=13

Assessment (includes both continuous and summative assessment) CZ3007

a) Final Examination: 60%
b) Practical Lab Assignments: 21%
c) Online Quiz: 3%
d) Written Quiz: 16%