Course Outcomes (COs):

CO 1. To create (K6) knowledge for analyzing (K4) problems and interpret/explain (K6) them with the acquired knowledge develop (K5) solutions through a mathematical framework that supports engineering, science, and mathematics.

CO 2. To characterize (A5) given problems by analyzing their arguments, in relation to their premises, assumptions, contexts, and conclusions and associating (K2) them with the mathematical designs (S5) used to express physical and natural laws, followed by computing (K3)

CO 3. Problem solving (S4) is naturalized (S5) through critical and analytical thinking.

CO 4. Be conversant with the mathematical precision and rigor that is necessary for computer science and be able to present (A2) it with accomplished (S3) arguments.

Course Articulation matrix

Graph Theory and Combinatorics	Program Outcomes									Program Specific Outcomes						
CO	PO 1	PO 2	P O 3	PO 4	PO 5	PO 6	PO 7	PO 8	PO 9	PO 10	PO 11	PO 12	PSO 1	PSO 2	PSO 3	PSO 4
CO 1	3	3	2	3		2					2	3	3	2		1
CO 2	3		3		2				2			3	3	2		
CO 3		3	2			3	2			3		3			2	1
CO 4	3	3		2	3					3		3	3	2	1	

Course code	CSE/PC/B/T/226
Category	Professional Core
Course title	**Computer Architecture**
Scheme and Credits	L–T–P: 3-0-0; Credits: 3.0; Semester – II
Pre-requisites (if any)	

Syllabus:

1. Introduction: Design objectives of a computer architect; cost and performance measures; benchmark & metrics; instruction set architecture classification; instruction format and semantics; memory addressing modes; instruction encoding principles; role of compilers; formal description of architecture; VHDL; AADL.

[2L]

2. Instruction level parallelism:

Basic principles of pipelines; structural, control and data hazards; instruction pipelines; branch prediction; pipeline scheduling and collision avoidance; optimizing pipeline performance; RISC & CISC pipeline examples. VLIW architecture; overview of proposed and commercial VLIW Systems.

[2L]

Superscalar architecture; basic objectives of superscalar processing; superscalar instruction issues; issue policies; instruction pairing rules; shelving; register renaming; load/ store reordering; the reorder buffer; instruction pipeline – D1, D2 execution and write-back stages; branch handling – delayed branch, multiway branch; case study – Power PC620, Pentium Pro. **[4L]**

Code scheduling for ILP processor; basic block scheduling; loop scheduling; global scheduling. **[2L]**

Data parallel architecture: Basic idea of data parallelism; connectivity – nearest neighbour, tree, pyramid, mesh, hypercube and reconfigurable networks; different classes of data parallel architecture – SIMD, associative, neural, data parallel pipeline, systolic and vector architectures. **[3L]**

SIMD architecture; features – granularity, connectivity, processor complexity & local autonomy; fine grained SIMD overview; an example – the Massively Parallel Processor; coarse grained SIMD overview; an example – the CM5; SIMD algorithm examples – matrix multiplication/ inversion, sorting/ searching. **[3L]**

Systolic architecture; introduction; systolic design space; comparison with multidimensional pipeline; spatial convolutions; case study – the WARP processor. **[2L]**

Vector architecture; principles of vectorization; pipelined & parallel stream implementation of vector machine; case study – the CRAY-1, C-90 and the Convex C4/X4 system. **[2L]**

Thread/ Process level parallelism: Basic architectural concepts; scalable parallel architecture; design issues for scalable MIMD computers. **[1L]**

Multi-thread implementation on sequential control flow model; case study – the Dencolor HEP machine, the MIT ‘Sparcle’ machine. **[3L]**

Dataflow architecture; the classical static dataflow machine proposed by J Dennis; tagged token dataflow machine; explicit token-store architecture; dataflow model verification using simple/ coloured Petri Net. **[3L]**

Shared memory MIMD architecture; systems using single & multiple shared buses; blocking & non-blocking interconnection networks such as cross-bar and other MINs. **[2L]**

Cache coherence problem; hardware & software coherence policies – write-invalidate, write- update, write-through and write-back policies; snoopy protocol. **[2L]**

Synchronization; spin-lock; event ordering in coherent systems. **[1L]**

Uniform memory access (UMA) machine example; non-uniform memory access (NUMA) machine example; cache coherent NUMA (CCNUMA) machine example; case study – the SUN Enterprise 6000. **[2L]**

RISC Architecture: comparison between CISC & RISC concepts; RISC machine, features; hardwired control; horizontal machine code format; register file; jumps & delay slots. **[2L]**

Special Architecture: Architectural considerations for low power hand held mobile devices, embedded systems. **[1L]**

Parallelization: Parallel program development environment and software tools; mapping application onto multi-computers. **[1L]**

Performance Evaluation: Role of performance; performance metrics; Amdahl’s law; benchmarks; the SPEC benchmarks; SPEC95 for Pentium & Pentium Pro; SPEC 2000 benchmarks; MIPS as performance metric; native, peak & relative MIPS & FLOPs as performance measure; synthetic benchmarks; price-performance metric. **[2L]**