Autonomous Program Structure
Third Year B. Tech. Sixth Semester
Computer Engineering
Academic Year: 2022-2023 Onwards

Course Code	Course Title	Teaching Scheme Hours /Week			Examination Scheme				Marks	Credits
		Lecture	Tutorial	Practical	In Semester	End Semester	Oral	Practical		
20CE601	Microprocessor and Microcontroller	3	1	0	50	50	0	0	100	4
20CE602	Software Engineering	3	1	0	50	50	0	0	100	4
20CE603	Cloud Computing	3	0	0	50	50	0	0	100	3
20HS601	Professional and Societal Awareness for Engineers	3	0	0	50	50	0	0	100	3
20PECE 601	Programme Elective-III	3	0	0	50	50	0	0	100	3
20OE601	Open Elective-II	3	0	0	50	50	0	0	100	3
20CE601L	Microprocessor and Microcontroller Laboratory	0	0	2	25	0	0	25	50	1
20CE603L	Cloud Computing Laboratory	0	0	2	25	0	25	0	50	1
20AC 601	Audit Course	0	0	2	0	0	0	0	0	No Credits
	Total	**18**	**2**	**6**	**350**	**300**	**25**	**25**	**700**	**22**
	Grand Total	**24**			**700**					**22**

Programme Elective-III
20PECE601A DevOps Fundamentals
20PECE601B Compiler Construction
20PECE601C Deep Learning
20PECE601D Data Management, Protection and Governance by Veritas Technologies

APPROVED BY
Secretary Governing Body
MKSSS's Cummins College of Engineering
For Women, Pune-411052

[illegible]

APPROVED BY
Chairman Governing Body
MKSSS's Cummins College of Engineering
For Women, Pune-411052

M.K.S.S.S. Cummins college of Engg. for Women ★ KARVENAGAR Pune -52

20OE601 Open Elective-II			Eligible Departments				
Sr. No.	Course Code	Course Title	EnTC	Comp	IT	Mech	Instru
1	20OE601A	Automation and Control Engineering	Y	Y	Y	Y	Y
2	20OE601B	Automotive Electronics	Y	Y	Y	Y	Y
3	20OE601C	Avionics	Y	Y	Y	Y	Y
4	20OE601D	Bioinformatics	Y	Y	Y	N	Y
5	20OE601E	Computer Vision	Y	Y	Y	Y	Y
6	20OE601F	Design Thinking	Y	Y	Y	Y	Y
7	20OE601G	e-Business	Y	Y	Y	Y	Y
8	20OE601H	Electric Vehicles	Y	Y	Y	Y	Y
9	20OE601I	Gamification	Y	Y	Y	Y	Y
10	20OE601J	Geographical Information Systems	Y	Y	Y	Y	Y
11	20OE601K	Multimedia Systems	Y	Y	Y	N	Y

[illegible]

APPROVED BY
Secretary Governing Body
MKSSS's Cummins College of Engineering
For Women, Pune-411052

[illegible]

APPROVED BY
Chairman Governing Body
MKSSS's Cummins College of Engineering
For Women, Pune-411052

M.K.S.S.S. Cummins college of Engg. for Women
KARVENAGAR
Pune -52

Autonomous Program Structure
Third Year B. Tech. Sixth Semester
Computer Engineering
Academic Year: 2022-2023 Onwards

Course Code	Course Title	Teaching Scheme Hours /Week			Examination Scheme				Marks	Credits
		Lecture	Tutorial	Practical	In Semester	End Semester	Oral	Practical		
20CE601	Microprocessor and Microcontroller	3	1	0	50	50	0	0	100	4
20CE602	Software Engineering	3	1	0	50	50	0	0	100	4
20CE603	Cloud Computing	3	0	0	50	50	0	0	100	3
20HS601	Professional and Societal Awareness for Engineers	3	0	0	50	50	0	0	100	3
20PECE 601	Programme Elective-III	3	0	0	50	50	0	0	100	3
20OE601	Open Elective-II	3	0	0	50	50	0	0	100	3
20CE601L	Microprocessor and Microcontroller Laboratory	0	0	2	25	0	0	25	50	1
20CE603L	Cloud Computing Laboratory	0	0	2	25	0	25	0	50	1
20AC 601	Audit Course	0	0	2	0	0	0	0	0	No Credits
	Total	**18**	**2**	**4**	**350**	**300**	**25**	**25**	**700**	**22**
	Grand Total	**24**			**700**					**22**

Programme Elective-III
20PECE601A DevOps Fundamentals
20PECE601B Compiler Construction
20PECE601C Deep Learning
20PECE601D Data Management, Protection and Governance by Veritas Technologies

20OE601 Open Elective-II			**Eligible Departments**				
Sr. No.	Course Code	Course Title	EnTC	Comp	IT	Mech	Instru
1	20OE601A	Automation and Control Engineering	Y	Y	Y	Y	Y
2	20OE601B	Automotive Electronics	Y	Y	Y	Y	Y
3	20OE601C	Avionics	Y	Y	Y	Y	Y
4	20OE601D	Bioinformatics	Y	Y	Y	N	Y
5	20OE601E	Computer Vision	Y	Y	Y	Y	Y
6	20OE601F	Design Thinking	Y	Y	Y	Y	Y
7	20OE601G	e-Business	Y	Y	Y	Y	Y
8	20OE601H	Electric Vehicles	Y	Y	Y	Y	Y
9	20OE601I	Gamification	Y	Y	Y	Y	Y
10	20OE601J	Geographical Information Systems	Y	Y	Y	Y	Y
11	20OE601K	Multimedia Systems	Y	Y	Y	N	Y

20CE 601 Microprocessor And Microcontroller

Teaching Scheme:	Examination Scheme:
Lectures: 3 hours/Week	In Semester: 50 Marks
Tutorial: 1 hour/Week	End Semester: 50 Marks
	Credits: 4

Prerequisite:

1. Digital Systems and Computer Organization (20CE 304)

Course Objectives:

To facilitate the learners

1. To understand basic architecture and programming of Pentium microprocessor.
2. To understand and analyze the protected mode of the Pentium processor.
3. To understand the architecture of an 8051 microcontroller.

Course Outcomes:

By taking this course, the learner will be able to

1. Demonstrate the knowledge of basic Pentium processor concepts.
2. Infer the advanced microprocessor architectures.
3. Make use of the 8051 microcontrollers for interfacing the devices.
4. Apply the programming concepts using x86 and 8051 assembly level language.

Unit – 1: PENTIUM MICROPROCESSOR ARCHITECTURE (06)

Pentium Architecture, Pipeline stages, Superscalar pipeline issues, Instruction pairing rules, Branch prediction, Memory organization with Instruction and Data caches Pentium programmers' model, register set, Addressing modes and instructions.

Unit – 2: PROTECTED MODE ARCHITECTURE IN PENTIUM (08)

Real Mode vs. Protected mode, Memory management with segmentation and paging Protection mechanism in segmentation and paging, Virtual 8086 Mode (support registers, descriptors, privilege-level, protection, exclusive instructions, inter-privilege level, transfer control, Paging-support registers, Descriptor, linear to physical address translation, TLB, page level protection).

Unit – 3: MULTITASKING, INTERRUPTS, EXCEPTION AND INPUT/OUTPUT (08)

Multitasking, support registers, Descriptors, Task switching, Nested task, I/O handling in Pentium, I/O instructions, I/O Permission bitmap, Interrupts and Exceptions structure in real, protected and virtual modes.

Unit – 4: 8051 MICROCONTROLLER ARCHITECTURE (06)

Features, Microcontroller MCS-51 family architecture. Programmers model-register set, register bank, SFR's, addressing mode, instruction set, Memory organization on-chip data memory External data memory and program memory. Memory interfacing-external RAM/ROM interface.

Unit – 5: 8051 AND INPUT-OUTPUT INTERFACING (08)

CPU timings, Interrupt structure, Timers and their programming, Serial port and programming, Serial Data Communication using RS-232C. I/O devices-ADC / DAC and Stepper Motor, Power saving modes in 8051.

Unit–6: INTRODUCTION TO ADVANCED MICROPROCESSORS AND MICROCONTROLLERS (06)

Introduction to multicore architectures i3/i5/i7, Cache coherency,
Processor Architectures for Mobile Application, Embedded Application and Enterprise Application. Introduction to Advanced Microcontrollers for Embedded Systems, A case study in embedded or IoT based systems.

Text Books:

1. 8086 and peripherals – Intel Manual
2. Pentium Architecture – Intel Manual
3. Intel 8-bit Microcontroller Manual

Reference Books:

1. Douglas Hall, **'Microprocessors & Interfacing'**, *McGraw Hill*, (Revised 2nd Edition), (2006)
2. James Antonakos, **'The Pentium Microprocessor'**, *Pearson Education,* (2nd Edition), (2004)
3. Sivarama P. Dandamudi, **'Introduction to Assembly Language Programming For Pentium and RISC Processors'**, *Springer*,(2nd Edition), (2004)
4. Muhammad Ali Mazidi and Janice Gillispie Mazidi, **"The 8051 Microcontroller and embedded systems**", 2009, Pearson education. ISBN – 81-7808-574-7
5. W. Stallings, '**Computer Organization and Architecture - Designing for Performance'**, *Prentice Hall of India*,(8th edition), (2002)

Web References:

1. NPTEL series – nptel.ac.in/courses/Webcourse-contents/IIT-KANPUR/
2. NPTEL series for 8051 – https://nptel.ac.in/courses/108/105/108105102/
3. service.scs.carleton.ca/sivarama/org_book/org_book_web/slides/chap_1_versions/ch7_1.pdf

List of Tutorial Assignments:
The subject Microprocessor and Microcontroller introduces the processor evolution from basic to advanced. It also signifies the use of microcontrollers in multiple real-life applications.
The tutorial is designed to develop the assembly language programming ability of an individual student.
The teachers can design different problems based on the topics suggested below –

1. & 2. Write small code snippets using arithmetic, logical and conditional jump instructions.

3. Learning how to use the DOS/LINUX system calls for program I/O.

4. Write small codes using string instructions.

5. Evaluate the output of small ALP's.

6. Numerical examples solving logical, linear and physical address translation for x86.

7. Draw memory maps and evaluate the changes after a particular instruction.

8. Develop 8051 program snippets.

9. Understand the basics of SFRs and register banks.

10. 8051 addressing modes.

11. Design delays using 8051 timers.

12. A design case study using 8051 and its interfacing with memory, I/O, sensors.

20CE 602 Software Engineering

Teaching Scheme	**Examination Scheme**
Lectures: 3 Hours/Week	In Semester: 50 marks
Tutorial: 1 Hour/Week	End Semester: 50 marks
	Credits: 4

Prerequisites: Software Design and Architecture (20CE 503)

Course Objectives:
To facilitate the learner to -

1. Develop familiarity with the software design and component based software engineering.
2. Get exposure to the various facets of agile software process model.
3. Learn the basic concepts of refactoring.
4. Gain knowledge about the various aspects of designing and testing of web applications.

Course Outcomes:
By taking this course, the learner will be able to -

1. Apply the concepts of component-level design to realize the solution of a system.
2. Analyze the agile software process model for application development.
3. Analyze the refactoring methods to restructure the classes.
4. Make use of various concepts of designing and testing for web applications.

Unit 1: Software Design Concepts and Component-Level Design (07)
Design within the context of Software Engineering, The design process, Design concepts, Design model. Component-Level Design: What is a component, Designing class-based components, Steps of component-level design, Component-based development.

Unit 2: Introduction to Agile Software Development (07)
Why agile software development - Limitations of traditional process models, Evaluating Agile Benefits, Understanding the Agile Manifesto, Outlining the Four Values of the agile Manifesto, Defining the 12 Agile Principles, Agile approaches - Lean, Scrum and Extreme Programming, Agile team.

Unit 3: Agile Project Planning and Software Practices (07)
Agile project inception, User stories, Estimation, Agile plan.
Agile software practices: Refactoring, Test-driven development, Continuous Integration Continuous Delivery (CICD); DevOps: Lifecycle, Benefits, Use cases, DevOps and Deployment.

Unit 4: Introduction to Refactoring **(07)**
What is Refactoring, Why and when to refactor, Code smells, Duplicated code, Long method, Extract method, Large class, Extract class, Alternative classes with different interfaces, Move method, Move field, Rename method, Rename variable, Replace method with method object.

Unit 5: Refactoring Methods **(07)**
Replace data value with object, Change unidirectional association to bidirectional, Switch statements, Replace conditional with polymorphism.

Remove control flag, Introduce assertion, Replace constructor with factory method, Replace error code with exception.
Pull up field, Pull up method, Push down method, Push down field, Extract subclass, Extract superclass, Extract interface, Replace inheritance with delegation.

Unit 6: Design and Testing of Web Applications **(07)**
WebApp design quality, Design goals, Design pyramid, WebApp interface design, Asthetic design, Content design, Architecture design, Navigation design, Component-level design, Object-oriented hypermedia design method.
Testing concepts for WebApps, Testing process - overview, Content testing, User interface testing, Component-level testing, Navigation testing, Configuration testing, Security testing, Performance testing.

Text books:

1. Roger S. Pressman, '**Software Engineering: A Practitioners Approach**', *Tata McGraw Hill*, (7th Edition) (2010).
2. Jonathan Rasmusson, **'The Agile Samurai: How Agile Masters Deliver Great Software',** *Shroff Publishers and Distributers (SPD)*, ISBN: 978-93-5213-411-3, (2016).
3. Martin Fowler, Kent Beck, John Brant, William Opdyke and Don Roberts, **'Refactoring: Improving The Design of Existing Code'**, *Pearson Education,* ISBN: 978-81-317-3466-7, (2017).
4. Mark C. Layton, Steven J. Ostermiller, **'Agile Project Management for Dummies',** *Wiley*, (2nd Edition), (2017).

Reference books:

1. Sanjeev Sharma and Bernie Coyne, **'DevOps for Dummies'**, *IBM Limited Edition*, John Wiley and Sons, Inc. (2015).
2. Ian Sommerville, '**Software Engineering'**, *Person Education,* (8th Edition) (2008).
3. Grady Booch, James Rumbaugh, Ivar Jacobson, **'The Unified Modeling Language User Guide'**, *Pearson Education,* (2nd Edition) (2008).

Web References:

1. Official website of R. S. Pressman and Associates, Inc: http://www.rspa.com/
2. Agile Software process model: https://www.agilealliance.org/
3. Basics of Scrum: https://www.scrumalliance.org/
4. https://www.bmc.com/blogs/devops-basics-introduction/

Tutorials - Preamble:

The scope of tutorials for "Software Engineering" includes exercises based on component-level design concepts, agile software practices, refactoring concepts and design and testing of web applications. During tutorials, problem solving and system design skills of students are challenged and improved.
For a chosen hypothetical system, students are expected to identify its scope, suitable classes, modules and build the component and deployment models. The students are also expected to apply the relevant refactoring techniques to improve the quality of the design. The following is a sample list of tutorials, covering the various concepts in the course. The objective of tutorials is to provide an opportunity for students to explore as per their interests. Consequently, these tutorial statements will be further detailed during conduction, according to the scenarios under consideration.

Example List of Tutorials:

1. Draw Component Diagrams to model components, interfaces and dependencies as part of an implementation view of a given system.

2. Draw Deployment Diagrams to show the configuration of run-time processing nodes and the components that reside on them, depicting the working scenario for a given application.

3. Apply CRC modeling to identify classes, their responsibilities and the collaborators for a given system. Also for this system, identify the relevant user interface classes, business domain classes, system classes, process classes and persistent classes.

4. For a given large and complex system, create a "NOT List". Also for this system, identify the various modules and specify the responsibilities of these modules.

5. For a given system, identify the different users and write down the User Stories for the various features of this system, using the user story template.

6. Imagine that you are part of an agile team involved in the development of some software product. For this product, design a Product Box.

7. Write an Elevator Pitch in proper format for any software product of your choice.

8. For the given requirement / function, apply and describe with code/pseudo code the 3 steps of Test Driven Development.

9. Refactor the given code using "Rename method" and "Rename variable" techniques. Write the refactored code.

10. Refactor the given code using "Extract method" technique and Write the refactored code.

11. Refactor the given code using "Replace error code with exception" technique and Write the refactored code.

12. Refactor the given code using "Remove control flag" technique and Write the refactored code.

13. Refactor the given code using "Introduce Assertion" technique and Write the refactored code.

14. Refactor the given code using "Move method" and "Move field" techniques. Write the refactored code.

15. Refactor the given code using "Replace conditional with polymorphism" technique and Write the refactored code.

16. For a given an inheritance hierarchy, apply "Pull up field/method" technique and Write the refactored code.

17. For a given an inheritance hierarchy, apply "Push down field/method" technique and Write the refactored code.

18. For the given code, apply the relevant refactoring techniques from "Extract class", "Extract subclass", "Extract superclass" and Write the refactored code.

19. For any typical web application of your choice, specify the various features and write down a set of test cases to test these features/functionalities.

20CE 603 Cloud Computing

Teaching Scheme	**Examination Scheme**
Lecture: 3 Hours/week	In Semester: 50 marks
	End Semester: 50 marks
	Credits: 3

Prerequisites: Operating Systems (20CE 403)

Course Objectives:
To facilitate the learner to -
1. Understand the basic concepts related to cloud computing.
2. Analyze the underlying principles of different cloud service models.
3. Understand and apply the security techniques in cloud computing.
4. Get exposure to emerging trends in cloud computing.

Course Outcomes:
By taking this course, the learner will be able to -
1. Apply cloud computing concepts and the emerging trends to cloud based systems.
2. Analyze the cloud services and models.
3. Analyze various cloud platforms and tools for realization of different services.
4. Apply security concepts to the cloud environment.

Unit 1: Introduction **(06)**

Introduction to Cloud Computing, Cloud Economics, National Institute of Standards and Technology (NIST) Definition of Cloud Computing, Cloud Characteristics, Cloud Service Models, Cloud Deployment Models, Benefits, Challenges and Risks.

Unit 2: Infrastructure-as-a-Service (IaaS) **(08)**

Introduction to Infrastructure-as-a-Service (IaaS), Virtualization – Introduction, Taxonomy, Characteristics, Pros and Cons, Types of Service Level Agreement (SLA), Hypervisors - Xen, Kernel Virtual Machine (KVM), VMware, Docker Containers, Serverless computing, Microservices, Microservices architecture, Case Study- Amazon Web Services (AWS).

Unit 3: Platform-as-a-Service (PaaS) **(07)**

Introduction to Platform-as-a-Service (PaaS), Data in Cloud: Relational Databases, NoSQL Databases, Big Data, Cloud File System: Hadoop Distributed File System (HDFS), HBase, Map-Reduce Model, Case Study- Google App Engine (GAE).

Unit 4: Software-as-a-Service (SaaS) **(08)**

Introduction to Software-as-a-Service (SaaS), Multi-tenancy, Mashups, Service Oriented Architecture (SOA), Web Services based on Simple Object Access Protocol (SOAP) and REpresentational State Transfer (REST), SaaS Applications, Case Study- Salesforce.com.

Unit 5: Cloud Security **(07)**
Cloud Security Fundamentals, Cloud Security Challenges and Risks, Virtualization Security, Identity Management and Access Control, Secure Execution Environment and Communication.

Unit 6: Recent Trends **(06)**
Inter-cloud / Federated Cloud, Internet of Things (IoT) and Cloud Computing, Mobile and Cloud Computing, Data Centers- Introduction, Cloud Applications, Cloud and DevOps, Research trends in Cloud Computing.

Text books:

1. Rajkumar Buyya, Christian Vecchiola, S Thamarai Selvi, '**Mastering Cloud computing**', *McGraw Hill Education*, (2013), ISBN 978-1-25-902995-0.
2. Gautam Shroff, '**Enterprise Cloud Computing**', *Cambridge University Press*, (2010), ISBN 978-0-521-13735-5.
3. Ronald Krutz and Russell Dean Vines, '**Cloud Security**', *Wiley India Pvt. Ltd.*, (2010), ISBN 978-81-265-2809-7.
4. Kailash Jayaswal, Jagannath Kallakurchi, Donald Houde, Dr. Deven Shah, '**Cloud Computing Black Book**', *DreamTech Press*, (2015), ISBN 978-93-5119-418-7.

Reference books:

1. Thomas Erl, Zaigham Mahmood and Ricardo Puttini, '**Cloud Computing Concepts, Technology and Architecture**', *Prentice Hall*, (2013), ISBN 978-01-333-8751-3.
2. Barrie Sosinsky, '**Cloud Computing Bible**', *Wiley India Pvt. Ltd.*, (2015), ISBN 978-81-265-2980-3.
3. Rajkumar Buyya, James Broberg, Andrzej Goscinski, '**Cloud Computing Principles and Paradigms**', *Wiley India Pvt. Ltd.*, (2015), ISBN 978-81-265-4125-6.
4. Dr. Kumar Saurabh, '**Cloud Computing**', *Wiley India Pvt. Ltd.*, (2011), ISBN 978-81-265-2883-7.
5. Tim Mather, Subra Kumaraswamy, Shahed Latif, '**Cloud Security and Privacy**', *O'Reilly*, (2011), ISBN 13:978-81-8404-815-5.
6. A. Srinivasan, J. Suresh, '**Cloud Computing: A Practical Approach for Learning and Implementation**', *Pearson*, (2014), ISBN 978-81-317-7651-3.

Web References:

1. http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.500-291r2.pdf
2. https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-145.pdf
3. https://docs.docker.com
4. https://www.bmc.com/blogs/devops-basics-introduction/
5. http://searchdatacenter.techtarget.com/definition/data-center
6. http://www.sapdatacenter.com/article/data_center_functionality/
7. https://www.salesforce.com

20HS 601 Professional And Societal Awareness For Engineers

Teaching Scheme	**Examination Scheme**
Lectures: 3 Hours / Week	In Semester: 50 Marks
	End Semester: 50 Marks
	Credits: 3

Course Objectives:

To facilitate the learner to

1. Understand professional ethics, communication and practices
2. Relate Intellectual property concepts to various documents , products
3. Study Sustainability issues and green computing in environmental context
4. Study social issues in the computing world

Course Outcomes:

After completion of the course, students will be able to

1. Apply professional and computing ethics
2. Relate Intellectual property basics to information management, storage and sharing
3. Apply sustainability paradigms to various computing centric issues
4 Relate green computing basics to IT systems
5. Apply sustainability principles to new world

Unit I: Professional Ethics and communication (08)

Morals, values and Ethics, Integrity, Work ethic, Civic virtue, Valuing time, Cooperation, Commitment, Empathy, Self-confidence, stress management, Senses of Engineering Ethics, Kohlberg‟s theory, Gilligan‟s theory, Models of professional roles, Uses of Ethical Theories, Communicating professionally with stakeholders

Unit II: Intellectual Property (08)

Philosophical foundations of intellectual property, Intellectual property rights (cross-reference IM/Information Storage and Retrieval/intellectual property and protection) ,Intangible digital intellectual property (IDIP), Copyrights, patents, trade secrets, trademarks, Plagiarism, non disclosure agreement

Unit III: Sustainability & CSR (09)

Basics of sustainability in IT and computing, Global social and environmental impacts of computer use and disposal, Business Ethics, Ethics Vs Social Responsibility, A view of corporate social responsibility (Legal, Ethical, Economic, Philanthropic) and its importance, ESG(Environmental, Social and Governance standards), Evolution of ESG from CSR

Unit IV: Green Computing (09)

Green IT Fundamentals: Business, IT, and the Environment , Green computing: carbon footprint, scoop on power, Green IT Strategies: Drivers, Dimensions, and Goals , Environmentally Responsible Business: Policies, Practices, and Metrics, Virtualization of IT systems, Role of electric utilities, Telecommuting, teleconferencing and teleporting , Materials recycling , Best ways for Green PC, Green Data center, Green Grid framework.

Unit V: Sustainability in Healthcare (08)

Basics, Societal expectations, Sustainability and Pharmaceutical products-Role in Human health, Sustainable Concerns All Along the Life Cycle of the Health-care Industry, Global corporate governance and IT

Text Books

1. Bhuvan Unhelkar, "Green IT Strategies and Applications-Using Environmental Intelligence", CRC Press, June 2014
2. Ming din, "Sustainable development for health care industry" , Springer
3. Niraja Pandey, Khushdeep Dharni, "Intellectual Property Rights", PHI
4. Caroline Whitbeck, "Ethics in Engineering Practice and Research", Cambridge Press, ISBN:978-1-107-66847-8

Reference books

1. Woody Leonhard and Katherine Murray, "Green IT for Dummies", Wiley Publications (2009),ISBN: 978-0-470-74349-2

Online resources

NPTEL on Professional Ethics :https://nptel.ac.in/courses/110/105/110105097/

20CE 601L Microprocessor And Microcontroller Laboratory

Teaching Scheme:
Practical: 2 hours./Week

Examination Scheme:
In Semester : 25 marks
Practical :25
Credits: 1

Prerequisite:

1. Digital Systems and Computer Organization (20CE 304)
2. Digital Systems and Computer Organization Laboratory (20CE 307)

Course Objectives:
To facilitate the learners

1. To understand and apply x86 instructions to write assembly language programs.
2. To learn, apply and analyze microprocessor and peripherals interfacing techniques.
3. To learn and use the interfacing of assembly language and higher-level language.
4. To be able to solve moderately complex problems using modular assembly language programming.
5. To understand and use privileged instructions.

Course Outcomes:
By taking this course, the learner will be able to

1. Apply x86 instructions to write assembly language programs.
2. Apply modular programming using assembly level language.
3. Apply 8051 instructions to develop simple microcontroller programs.
4. Build a small system using microcontroller interfacing techniques.

The Microprocessor and Microcontroller laboratory assignments are designed for problem solving using assembly language programming. The laboratory work also covers the introduction to microcontroller assembly language and real-life case studies. It also aims to familiarize the concepts of use of modular programming and higher level language with ALP. The assignments in Group A cover basic concepts of assembly language programming whereas Group B cover structured assignments with advanced assembly language approaches. The indicative titles are mentioned in Group C, where students will be able to select a title and apply the learned concepts to understand the requirements of a real world application.

Group A Assignments (Perform all assignments)

1. Develop an application using x86 ALP to perform data declarations, arithmetic and logical operations and check the output in debugger.
2. Develop an application using x86 ALP to accept a signed number and check if it is positive or negative.
3. Develop an application using x86 ALP to accept a string from user and perform operations like
   (a) Convert a string to uppercase / lowercase
   (b) Toggle the case of the string

(c) Concatenation of another string
(d) Find if it is palindrome
(e) Find a substring
(Use of macros and procedures is recommended.
For this assignment make a group of 3-4 students, each one performing each task and then combine all functions to apply modular programming.)

4. Develop an application using 8051 Assembly language programming for addition, subtraction, multiplication and division of two 8-bit numbers.
5. Develop an application using 8051 Assembly language programming for block data transfer between internal and external memory including overlapping blocks.

Group B Assignments

Part B-1 – Select any one assignment from the following.

1. Develop an application using x86 ALP to simulate TYPE or COPY commands with the help of command line arguments.
2. Develop a modular application using x86 ALP PUBLIC/GLOBAL and EXTERN. Choose any application of your choice.
3. Develop an application by selecting any high-level language and insert assembly language code into it.

Part B-2 – Select any one assignment from the following.

1. Develop a suitable application using 8051 ALP with the help of timer, counter and interrupts.
2. Develop a suitable application using 8051 ALP for serial communication.
3. Develop a suitable application using 8051 ALP to interface I/O and DAC.

Group C Assignments

1. Design and build system for any real world application using 8051. Suggestive titles can be -
   i) Digital clock programming using 7- segment display.
   ii) Programming of LCD.
   iii) Programming on the keyboard.
   iv) Programming of parallel ADC.
   v) Interfacing Stepper Motor.
   vi) Speed Control of DC motor.
   vii) Interfacing Relay.
2. Study assignment - Perform a case study for a real time application of microprocessor or microcontroller. E.g. Application using DSP processor, ARM, embedded systems, vector, multi-core or array processor.

Text Books:

1. 8086 and peripherals – Intel Manual
2. Pentium Architecture – Intel Manual
1. Intel 8-bit Microcontroller Manual

Reference Books:

1. Douglas Hall, **'Microprocessors & Interfacing'**, *McGraw Hill*, (Revised 2nd Edition), (2006)
2. James Antonakos, **'The Pentium Microprocessor'**, *Pearson Education,* (2nd Edition), (2004)
3. Sivarama P. Dandamudi, **'Introduction to Assembly Language Programming For Pentium and RISC Processors'**, *Springer*,(2nd Edition), (2004)
4. Muhammad Ali Mazidi and Janice Gillispie Mazidi, **"The 8051 Microcontroller and embedded systems**", 2009, Pearson education. ISBN – 81-7808-574-7
5. W. Stallings, '**Computer Organization and Architecture - Designing for Performance'**, *Prentice Hall of India*,(8th edition), (2002)

Web References:

1. NPTEL series – nptel.ac.in/courses/Webcourse-contents/IIT-KANPUR/
2. NPTEL series for 8051 – https://nptel.ac.in/courses/108/105/108105102/
3. service.scs.carleton.ca/sivarama/org_book/org_book_web/slides/chap_1_versions/ch7_1.pd f

20CE 603L Cloud Computing Laboratory

Teaching Scheme	**Examination Scheme**
Practical: 2 Hours/week	In Semester: 25 Marks
	Oral: 25 Marks
	Credits: 1

Course Objectives:
To facilitate the learners to -

1. Explore the underlying principles of Infrastructure-as-a-Service (IaaS), virtualization and containers.
2. Understand the use of the Hadoop ecosystem.
3. Get exposure to the use of cloud Application Programming Interfaces (APIs) for developing sample application(s).
4. Study different cloud platforms and tools for various cloud service models.

Course Outcomes:
By taking this course, the learner will be able to -

1. Apply the hypervisor and container-based virtualization.
2. Experiment with the Hadoop ecosystem by implementing sample programs for Hive/HDFS/Map-Reduce.
3. Make use of CloudSim framework for understanding cloud computing infrastructure and services.
4. Analyze the use of different cloud platforms and tools/APIs for various cloud service models.

Preamble:

The intent of Cloud Computing Laboratory is to enable the understanding and implementation of the basic concepts of Cloud Computing. Assignment statements are in brief and can be implemented with Java/Python programming language. Motivation here is that students should be able to experiment with different aspects of IaaS, PaaS and SaaS using various APIs/libraries. Faculty members are encouraged to expand problem statements with variations. Assignments can be framed and expanded in such a way that it explores concepts, logic of solution and simple application. Students will be encouraged to explore different cloud platforms and tools. Faculty will appropriately adopt assignments on similar lines as the examples shown here. The basic and the next level experimentation with CloudSim, Docker container, virtualization and Hadoop ecosystem is covered by the assignments in Group A and those in Group B, respectively. Group B assignments are also on exploring the various cloud APIs. Group C assignments are on exploring the various cloud platforms.

Suggestive List of Assignments:

Group A: (Mandatory)

1. Explore the CloudSim platform for cloud modelling. For example: Create a data centre with one host and run one cloudlet on it using CloudSim.
2. Demonstrate the use of Docker container by exploring its related commands. Also, show the use of Fedora/Ubuntu images over the Docker engine.
3. Demonstrate the use of MySQL/Tomcat/MongoDB image over the Docker engine.
4. Demonstrate the use of Hive query language (HQL) to process the data using Hadoop ecosystem.
5. Create a virtual machine using Kernel Virtual Machine (KVM) and explore commands for virtualization.

Group B: (Any Three)

1. Experiment with the CloudSim platform for modelling and simulation of cloud infrastructure. For example: Create and configure the data centre and user base to show response time, request servicing time and data centre loading.
2. Frame Python scripts to perform operations (for e.g. start/pause/stop) on the Virtual Machine using Libvirt and Operating System (OS) calls for virtualization.
3. Build the Docker image from a Docker file and demonstrate the use of it over the Docker engine.
4. Using Hadoop ecosystem, implement Map-Reduce word count program on single node cluster for the given sample data.
5. Using Hadoop ecosystem, implement Map-Reduce program for the given log file data.
6. Explore and configure the Xen/VirtualBox/VMware hypervisor.
7. Execute Hadoop Distributed File System (HDFS) commands on Hadoop ecosystem.
8. Install Google App Engine. Create hello world application and other simple web applications using Python/Java.
9. Explore the use of API for cloud storage application (for e.g. DropBox API) with the Linux command line interface and Python script.
10. Create an application using Force.com API.
11. For a sample application, implement and consume web service using social networking APIs with Simple Object Access Protocol (SOAP).

12. For a sample application, implement and consume web service using cloud APIs with REpresentational State Transfer (REST).

Group C: (Any One)

1. Installation and configuration of an open source cloud platform.
2. Explore the use of different cloud platforms such as Google App Engine (GAE), Amazon Platform Services, Microsoft Azure services, Openstack and Rackspace.

20PECE 601A DevOps Fundamentals

Teaching Scheme
Lectures: 3 Hours/Week

Examination Scheme
In Semester: 50 marks
End Semester: 50 marks
Credits: 3

Prerequisites: Software Design and Architecture (20CE 503)

Course Objectives:
To facilitate the learner to -

1. Understand and appreciate the need for the DevOps as a state-of-art software engineering practice.
2. Learn the basic concepts related to DevOps.
3. Get acquainted with the various tools which are used in different phases of DevOps model.
4. Get exposure to emerging trends in software development related to DevOps.

Course Outcomes:
By taking this course, the learner will be able to -

5. Apply the fundamental concepts and emerging trends of DevOps to software development.
6. Analyze the various concepts of application development such as agile software model, microservices to understand the need of DevOps based solution of a system.
7. Compare various concepts and tools of continuous integration, continuous delivery, continuous testing and monitoring for DevOps based realization of solution of a system.
8. Analyze the various deployment platforms as part of DevOps lifecycle.

Unit 1: Introduction to DevOps **(06)**
Overview, Features, Components, Why to use DevOps - Benefits, Business need for DevOps, Using DevOps to solve new challenges like enabling mobile applications, scaling agile and managing multitier applications; DevOps lifecycle - DevOps stages: Develop, Code/build, Test, Deploy; DevOps use cases DevOps techniques: Continuous improvement, Release planning, Continuous integration, Continuous delivery, Continuous testing, Continuous monitoring and feedback; DevOps tools.

Unit 2: Application Development **(07)**
Agile software development, User stories, Automating SDLC and DevOps, DevOps team. Serverless computing. Microservices and DevOps: Monolithic to microservices Project in terms of microservices, need/applicability of DevOps Microservices - what, characteristics, how are they used, nature in continuous release, architecture, services design.

Unit 3: Continuous Integration Continuous Delivery (CICD) Pipeline (08)

CICD pipeline - basics, Source code repository, Version control and source code management - GitHub, Git commands. Creating Automated build, Automated build frameworks like Maven, Ant for Java. Automated configuration and automated deployment, Configuration management with tools like Puppet and Chef, Ansible; Continuous integration with Jenkins.

Unit 4: Continuous testing and Continuous monitoring (08)

Continuous Unit testing and Integration testing, Test Driven Development (TDD), Release testing, Testing in development, Testing in production, Selenium: Introduction, Why to use, automating test cases for testing web elements, Creating test cases in Selenium WebDriver. Testing and Bug tracking Frameworks such as JUnit, TestNG and JIRA. Continuous monitoring and logging with tools like Nagios.

Unit 5: Deployment Platforms (07)

Containerization and DevOps: Virtualization, Application runtime environment, Application needs/dependencies, Containerization, Containerization Tools, Benefits of containers in enabling DevOps workflow.

Dockers for DevOps: Docker - Overview, Docker lifecycle, Docker Image, Docker file, Docker registry, Docker engine, Docker runtime container, Docker installation, commands, Namespaces, Docker layered approach, Docker applications/use cases. Container Orchestration: Kubernetes.

Unit 6: DevOps - Applications, Case studies and Trends (06)

Cloud's benefit to DevOps, Web Applications on Cloud Platform

Automation of infrastructure: Terraforms - Infrastructure as code (IaC) software tool.

DevSecOps: Agile security, DevOps and security, Low code solutions.

Text books:

1. Sanjeev Sharma and Bernie Coyne, 'DevOps for Dummies', IBM Limited Edition, John Wiley and Sons, Inc., ISBN- 978-1-119-04705-6, (2015).
2. Viktor Farcic, 'The DevOps 2.0 Toolkit: Automating the Continuous Deployment Pipeline with Containerized Microservices', CreateSpace Independent Pub, (2016).
3. Katrina Clokie, 'A Practical Guide to Testing in DevOps', Leanpub, (2017).

Reference books:

1. Bass, L., Weber, I.M., Zhu, L., 'DevOps: a software architect's perspective'. Pearson Education, ISBN: 9789332570375, (2016).
2. Davis J., Daniels K., 'Effective DevOps: Building a Culture of Collaboration, Affinity and Tooling at Scale', O'Reilly, ISBN- 9789352133765, (2018).

3. Farooqui S. M., 'Enterprise DevOps Framework: Transforming IT Operations', CA Press / Apress, ISBN- 9781484240618, (2019).
4. Sanjeev Sharma, 'The DevOps Adoption Playbook: A Guide to Adopting DevOps in a Multi-Speed IT Enterprise', Wiley, ISBN- 9788126569083, (2017).
5. Humble, J., Farley, D.: 'Continuous Delivery: Reliable Software Releases Through Build, Test, and Deployment Automation'. 1st edn. Addison-Wesley Professional (2010).

Web References:

1. https://devops.com/
2. https://docs.docker.com
3. https://www.bmc.com/blogs/devops-basics-introduction/
4. https://www.ibm.com/in-en/cloud/devops
5. https://aws.amazon.com/devops/what-is-devops/

20PECE 601B Compiler Construction

Teaching Scheme	**Examination Scheme**
Lectures: 3 Hours/week	In Semester : 50 Marks
	End Semester : 50 marks
	Credits : 3

Course Objectives:

To facilitate the learners -

1. To describe the phases of the compiler and the translation process.
2. To understand various parsing techniques
3. To discuss the effectiveness of optimization.
4. To learn and use tools for automatic compiler generation.

Course Outcomes:

By taking this course, the learner will be able to -

1. Build the knowledge of various system software.
2. Make use of Finite automata and tools to tokenize the given source code.
3. Construct a parser for a small context-free grammar.
4. Create symbol table and intermediate code for a simple programming language.
5. Apply the code optimization and code generation algorithms to get the machine code for the optimized code.

Unit 1: Introduction to System Programming and Compilation (06)

Components of System Software, Language Processing Activities, Fundamentals of Language Processing, Assembler, Compiler, Interpreter, Linkers and Loader, Dynamic Link Libraries. What is a Compiler, what is the Challenge, Compiler Architecture, Front end and Back end model of compiler, Cross compiler, Incremental compiler, Bootstrapping.

Unit 2: Lexical Analysis (06)

Concept of Lexical Analysis, Regular Expressions, Deterministic finite automata (DFA), Non- Deterministic finite automata (NFA), Converting regular expressions to DFA, Converting NFA to DFA, Hand coding of Lexical analyzer, Introduction to LEX Tool and LEX file specification

Unit 3: Syntax Analysis (10)

Context Free Grammars (CFG), Concept of parsing, Parsing Techniques, Top-Down Parsers: Introduction, Predictive Parsing - Removal of left recursion, Removal of left factoring, Recursive Descent Parsing, Predictive LL(k) Parsing Using Tables, Bottom Up parsing: Introduction, Shift-Reduce Parsing Using the ACTION/GOTO Tables, Table Construction, SLR(1), LR(1), and LALR(1) Grammars, Practical Considerations for LALR(1) Grammars, Introduction to YACC Tool & YACC file specification

MKSSS'S CUMMINS COLLEGE OF ENGG. FOR WOMEN KARVENAGAR PUNE-52.

Unit 4: Semantic Analysis (06)

Need of semantic analysis, Abstract Parse trees for Expressions, variables, statements, functions and class declarations, Syntax directed definitions, Syntax directed translation schemes for declaration processing, type analysis, scope analysis, Symbol Tables (ST), Organization of ST for block structure and non-block structured languages, Symbol Table management, Type Checkers: type checking for expressions, declarations (variable, type, function, recursive), statements

Unit 5: Intermediate Code Generation and Code Optimization (08)

Intermediate languages, Design issues, Intermediate representations: three address, postfix & abstract syntax trees, Intermediate code generation for declaration, assignment, iterative statements, case statements, arrays, structures, conditional statements and Boolean expressions. Model of a program in execution, Stack and static allocation, Activation records. Introduction to optimization, Principal sources of optimization, Machine Independent Optimization and Machine Dependent Optimization.

Unit 6: Code Generation and Advances in Compilation (06)

Issues in the design of code generation, Target machine description, Basic blocks & flow graphs, Expression Trees, Unified algorithms for instruction selection and code generation., Sethi Ullman algorithm for expression trees, Aho Johnson algorithm, Different models of machines, order of evaluation, register allocation. Advances in compilation.

Text Books:

1. Aho, Sethi, Ulman, Lam, "Compilers: Principles, Techniques and Tools", Pearson, 2nd Edition, ISBN 978-93-325-1866-7
2. Dhamdhere D., "Systems Programming and Operating Systems", 2nd Edition, ' McGraw Hill, 1999, ISBN 0 - 07 - 463579 – 4.

Reference Books:

1. Andrew Appel, "Modern Compiler Implementation in C", Cambridge
2. Kenneth C. Louden, "Compiler Construction: Principles and Practice", Cengage Learning, ISBN-13:978-0534939724
3. J. R. Levine, T. Mason, D. Brown, "Lex & Yacc", O'Reilly, 2000, ISBN 81-7366 –061-X

MKSSS'S CUMMINS COLLEGE OF ENGG. FOR WOMEN KARVENAGAR PUNE-52.

20PECE 601C Deep Learning

Teaching Scheme
Lectures: 3 Hours / Week

Examination Scheme
In Semester: 50 Marks
End Semester: 50 Marks
Credits: 3

Course Objectives:

To facilitate the learner to

1. Understand building blocks of Deep Neural Networks.
2. Understand various optimization algorithms used for training Deep Neural Networks.
3. Understand the working of CNN, RNN
4. Have knowledge of Deep Architectures for solving various applications.

Course Outcomes:

After completion of the course, students will be able to

1. Apply mathematical concepts and Machine Learning Basics for understanding Deep Learning topics
2. Apply concepts of Feedforward Networks for understanding Deep Learning topics
3. Apply the basic concepts of CNN and RNN to real time problems
4. Apply available Deep Learning solutions to real time applications.

Unit I: Machine Learning and Deep Learning (07)

What Is Deep Learning and Machine Learning Work? Limitations of Machine Learning, History of Deep Learning, Advantages/ Challenges of Deep Learning, Bias Variance trade off, hyper- parameters, Regularization, Confusion matrix, Building a Machine Learning Algorithm, Deep Learning tools/frameworks.

Unit II: Deep Learning Basics (07)

Linear Algebra, Probabilities and Information theory, Linear Dependence and Span, Norms, Eigen decomposition, The Trace Operator, The Determinant, Principal Components Analysis, Activation Functions, Loss Functions, Perceptron, Sigmoid neurons.

Unit III: Feedforward Networks for Deep Learning (07)

Learning XOR, Gradient-Based Learning, Hidden Units, Architecture Design, Back-Propagation, Regularization and Under-Constrained Problems, Dataset Augmentation, Noise, Early Stopping, Parameter Tying and Parameter Sharing, Dropout, Introduction to Keras, TensorFlow, Theano, and CNTK, Setting up a deep-learning workstation.

Unit IV: Convolution Neural Network (CNN) (08)

Biological Inspiration and Motivation, The Convolution Operation, Pooling, Padding, Overview of CNN Architecture, Input Layers, Convolutional Layers, Pooling Layers, Fully Connected Layers, Back propagation in CNN, Applications of CNNs, Introduction to convnets.

Unit V: Recurrent Neural Network (RNN) (07)

Working with text data, One-hot encoding of words and characters, Using word embeddings, Wrom raw text to word embeddings, Wrapping up, Recurrent Neural Network (RNN), A recurrent layer in Keras, Understanding the LSTM and GRU, Advanced use of recurrent neural networks, A temperature-forecasting

Unit VI: Advanced Deep Learning (06)

Introduction to Deep Learning applications in Computer Vision / NLP / Text Mining, Understanding use of CNNs for classification, Semantic Segmentation, Image denoising, Object Detection. Introduction to Generative Adversarial Networks, Deep Reinforcement Learning, AlexNet/VGG Net/ResNet etc.

Text Books:

1. Deep Learning, Ian Goodfellow and Yoshua Bengio and Aaron Courville, MIT Press Ltd. ISBN:9780262035613, 0262035618, 2016
2. Deep Learning with Python, FRANÇOIS CHOLLET, Manning Publications Co., ISBN 9781617294433, 2017
3. Python Deep Learning,Valentino Zocca, Gianmario Spacagna, Daniel Slater, Peter Roelants, Packt Publishing, ISBN 9781786460660, 2017

Reference Books:

1. Fundamentals of Deep Learning: Designing Next Generation intelligence Alogrithms, Nikhil Baduma, Nicholas Locascio, O'Reilly Publication, ISBN 10: 9352135601 , ISBN 13: 978- 9352135608, 2017

MKSSS'S CUMMINS COLLEGE OF ENGG. FOR WOMEN KARVENAGAR PUNE-52.

2. Deep Learning – A Practitioner's approach, Josh Patterson and Adam Gibson, O'Reilly Publication, 1st edition, ISBN : 9789352136049, 2017

3. Deep Learning with PyTorch, ELI STEVENS, LUCA ANTIGA, AND THOMAS VIEHMANN, Manning Publications Co, ISBN 9781617295263, 2020