CST 305	SYSTEM SOFTWARE	Category	L	T	P	Credit	Year of Introduction
		PCC	3	1	0	4	2019

Preamble:

The purpose of this course is to create awareness about the low-level codes which are very close to the hardware and about the environment where programs can be developed and executed. This course helps the learner to understand the machine dependent and machine independent system software features and to design/implement system software like assembler, loader, linker, macroprocessor and device drivers. Study of system software develops ability to design interfaces between software applications and computer hardware**.**

Prerequisite: A sound knowledge in Data Structures, and Computer Organization

Course Outcomes: After the completion of the course the student will be able to

CO#	Course Outcomes
CO1	Distinguish **softwares** into system and application software categories. **(Cognitive Knowledge Level: Understand)**
CO2	Identify **standard** and extended architectural features of machines. (**Cognitive Knowledge Level: Apply)**
CO3	Identify machine dependent features of system software (**Cognitive Knowledge Level: Apply)**
CO4	Identify machine independent features of system software. (**Cognitive Knowledge Level: Understand)**
CO5	Design **algorithms** for system softwares and analyze the effect of data structures. (**Cognitive Knowledge Level: Apply)**
CO6	Understand the features of device drivers and editing & debugging tools.(**Cognitive Knowledge Level: Understand)**

Mapping of course outcomes with program outcomes

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12
CO1	✓	✓			✓							✓
CO2	✓	✓	✓									✓
CO3	✓	✓	✓									✓
CO4	✓	✓										✓
CO5	✓	✓	✓	✓								✓
CO6	✓	✓			✓							✓

Abstract POs defined by National Board of Accreditation			
PO#	**Broad PO**	**PO#**	**Broad PO**
PO1	Engineering Knowledge	**PO7**	Environment and Sustainability
PO2	Problem Analysis	**PO8**	Ethics
PO3	Design/Development of solutions	**PO9**	Individual and team work
PO4	Conduct investigations of complex problems	**PO10**	Communication
PO5	Modern tool usage	**PO11**	Project Management and Finance
PO6	The Engineer and Society	**PO12**	Lifelong learning

Assessment Pattern

Bloom's Category	Continuous Assessment Tests		End Semester Examination Marks(%)
	Test 1 (%)	Test 2 (%)	
Remember	30	30	30
Understand	30	30	30
Apply	40	40	40
Analyze			
Evaluate			
Create			

Mark Distribution

Total Marks	CIE Marks	ESE Marks	ESE Duration
150	50	100	3

Continuous Internal Evaluation Pattern:

Attendance : **10 marks**
Continuous Assessment Test (Average of series Tests 1&2) : **25 marks**
Continuous Assessment Assignment **: 15 marks**

Internal Examination Pattern:

Each of the two internal examinations has to be conducted out of 50 marks. First series test shall be preferably conducted after completing the first half of the syllabus and the second series test shall be preferably conducted after completing remaining part of the syllabus. There will be two parts: Part A and Part B. Part A contains 5 questions (preferably, 2 questions each from the completed modules and 1 question from the partly completed module), having 3 marks for each question adding up to 15 marks for part A. Students should answer all questions from Part A. Part B contains 7 questions (preferably, 3 questions each from the completed modules and 1 question from the partly completed module), each with 7 marks. Out of the 7 questions, a student should answer any 5.

End Semester Examination Pattern:

There will be two parts; Part A and Part B. Part A contains 10 questions with 2 questions from each module, having 3 marks for each question. Students should answer all questions. Part B contains 2 questions from each module of which a student should answer any one. Each question can have maximum 2 sub-divisions and carry 14 marks.

Syllabus

Module-1 (Introduction)

System Software vs Application Software, Different System Software– Assembler, Linker, Loader, Macro Processor, Text Editor, Debugger, Device Driver, Compiler, Interpreter, Operating System (Basic Concepts only). SIC & SIC/XE Architecture, Addressing modes, SIC & SIC/XE Instruction set , Assembler Directives.

Module-2 (Assembly language programming and Assemblers)

SIC/XE Programming, Basic Functions of Assembler, Assembler Output Format – Header, Text and End Records. Assembler Data Structures, Two Pass Assembler Algorithm, Hand Assembly of SIC/XE Programs.

Module-3 (Assembler Features and Design Options)

Machine Dependent Assembler Features-Instruction Format and Addressing Modes, Program Relocation. Machine Independent Assembler Features –Literals, Symbol Defining Statements, Expressions, Program Blocks, Control Sections and Program Linking. Assembler Design Options- One Pass Assembler, Multi Pass Assembler. Implementation Example-MASM Assembler.

Module-4 (Loader and Linker)

Basic Loader Functions - Design of Absolute Loader, Simple Bootstrap Loader. Machine Dependent Loader Features- Relocation, Program Linking, Algorithm and Data Structures of Two Pass Linking Loader. Machine Independent Loader Features -Automatic Library Search, Loader Options. Loader Design Options.

Module-5 (Macro Preprocessor ,Device driver, Text Editor and Debuggers)

Macro Preprocessor - Macro Instruction Definition and Expansion, One pass Macro processor Algorithm and data structures, Machine Independent Macro Processor Features, Macro processor design options. Device drivers - Anatomy of a device driver, Character and block device drivers, General design of device drivers. Text Editors- Overview of Editing, User Interface, Editor

Structure. Debuggers - Debugging Functions and Capabilities, Relationship with other parts of the system, Debugging Methods- By Induction, Deduction and Backtracking.

Text book

1. Leland L. Beck, System Software: An Introduction to Systems Programming, 3/E, Pearson Education Asia

References

1. D.M. Dhamdhere, Systems Programming and Operating Systems, Second Revised Edition, Tata McGraw Hill.
2. John J. Donovan, Systems Programming, Tata McGraw Hill Edition 1991.
3. George Pajari, Writing UNIX Device Drivers, Addison Wesley Publications (Ebook : http://tocs.ulb.tu-darmstadt.de/197262074.pdf).
4. Peter Abel, IBM PC Assembly Language and Programming, Third Edition, Prentice Hall of India.
5. Jonathan Corbet, Alessandro Rubini, Greg Kroah-Hartman, Linux Device Drivers, Third Edition, O.Reilly Books
6. M. Beck, H. Bohme, M. Dziadzka, et al., Linux Kernel Internals, Second Edition, Addison Wesley Publications,
7. J Nithyashri, System Software, Second Edition, Tata McGraw Hill.
8. The C Preprocessor http://gcc.gnu.org/onlinedocs/gcc-2.95.3/cpp_1.html -

Course Level Assessment Questions

Course Outcome 1 (CO1):

1. List out two system software and two application software.

Course Outcome 2 (CO2):

1. How is upward compatibility between SIC and SIC/XE machines maintained?
2. Write a sequence of instructions for SIC/XE to divide BETA by GAMMA, setting ALPHA to the integer portion of the quotient and DELTA to the remainder. Use register-to-register instructions to make the calculation as efficient as possible.

Course Outcome 3 (CO3):

1. How do control sections and program blocks differ?
2. Can an assembler incorporating program blocks function using the same data structures as that of a normal two pass assembler? Justify your answer

Course Outcome 4 (CO4):

1. What are literals used for? Does the use of literals change the design of an assembler?

Course Outcome 5 (CO5):

1. Design an assembler that can assemble a source program with different control sections.

Course Outcome 6 (CO6):

1. Describe any one commonly used debugging method.

Model Question Paper

QP CODE:

Reg No: _____

Name: _____ **PAGES : 3**

APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY

FIFTH SEMESTER B.TECH. DEGREE EXAMINATION, MONTH & YEAR

Course Code: CST 305

Course Name: System Software

Max.Marks:100 **Duration: 3 Hours**

PART A

Answer All Questions. Each Question Carries 3 Marks

1. Differentiate between system software and application software.
2. What are assembler directives? List out any five assembler directives in SIC.
3. Explain the different data structures used in the implementation of Assemblers.
4. List out the functions performed by an assembler.

5. What is a Literal? How is a literal handled by an assembler.

6. What are control sections? What is the advantage of using them?

7. Differentiate between linking loader and linkage editor? Which of these is preferable in a program development environment?

8. What is Automatic Library Search?

9. How should a programmer decide whether to use a macro or a subroutine to accomplish a given logical function?

10. Differentiate between character and block device drivers

(10x3=30)

Part B

(Answer any one question from each module. Each question carries 14 Marks)

11. (a) Differentiate between compilers and Interpreters. . **(4)**

(b) Explain the architecture and addressing modes of SIC machine. **(10)**

OR

12. (a) Explain the addressing modes supported by SIC/ XE machine with suitable illustrations. **(8)**

(b) Explain the difference between **(6)**
i) A1 RESW 3 and A1 WORD 3
ii) B BYTE C'23' and B BYTE X'23'
iii) END and END LABEL

13. (a) Let NUMBERS be an array of 100 words. Write a sequence of SIC/XE instructions to find the maximum of these numbers. **(6)**

(b) Perform hand assembly of the above written program using two pass assembler and show the status of various data structures and object program create. **(8)**

OR

14. (a) Write down and explain the second pass of a two pass assembler algorithm. **(8)**

(b) What is a Program Block. What is its advantage? With suitable example, explain how Program Blocks are handled by SIC assembler. **(6)**

15. (a) What is a Program Block. What is its advantage? With suitable example, explain how Program Blocks are handled by SIC assembler. **(7)**

(b) What is a forward reference? With example, illustrate how forward references are handled by a single pass assembler? **(7)**

OR

16. (a) With suitable examples explain machine dependent assembler features. **(8)**

(b) Explain with examples, the need and working of multipass assembler. **(6)**

17. (a) With the data structures used, state and explain two pass algorithm for a linking loader. **(10)**

(b) Explain about bootstrap loader. **(4)**

OR

18. (a) Explain about machine independent loader features **(9)**

(b) What is Dynamic Linking? With example, illustrate how dynamic linking is performed. **(5)**

19. (a) Write down the single pass macro processor algorithm and with suitable example illustrate its working. **(10)**

(b) How are unique labels generated during Macro Expansion? **(4)**

OR

20. (a) Explain Text Editor structure in detail with a neat diagram. **(7)**

(b) Explain the different debugging methods in detail. **(7)**

Teaching Plan

No	Contents	No: of Lecture Hours
Module -1 (Introduction) (9 hours)		
1.1	System Software Vs. Application Software , Different System Software– Assembler, Linker, Loader, Macro Processor	1 hour
1.2	Text Editor, Debugger, Device Driver, Compiler, Interpreter, Operating System(Basic Concepts only)	1 hour
1.3	SIC Architecture	1 hour
1.4	SIC Addressing modes	1 hour
1.5	SIC Instruction set & Assembler directives	1 hour
1.6	SIC/XE Architecture	1 hour
1.7	SIC/XE Instruction format	1 hour
1.8	SIC/XE Addressing modes	1 hour
1.9	SIC/XE Instruction set	1 hour
Module -2 (Assembly language programming and Assemblers) (8 hours)		
2.1	SIC Programming	1 hour
2.2	SIC/XE Programming	1 hour
2.3	Basic Functions of Assembler	1 hour
2.4	Assembler output format – Header, Text and End Records	1 hour
2.5	Assembler data structures	1 hour
2.6	Pass 1 of two pass SIC assembler algorithm	1 hour
2.7	Pass 2 of two pass SIC assembler algorithm	1 hour
2.8	Hand assembly of SIC Program	1 Hour
Module-3 (Assembler design options)(11 hours)		

3.1	Machine dependent assembler features-Instruction format and addressing modes, program relocation	1 hour
3.2	Hand assembly of SIC/XE program	1 Hour
3.3	Machine Independent assembler features – Literals	1 hour
3.4	Machine Independent assembler features – Symbol defining statements, expression	1 hour
3.5	Machine Independent assembler features – program blocks	1 hour
3.6	Machine Independent assembler features – program blocks illustration with examples	1 hour
3.7	Machine Independent assembler features – Control sections and program linking.	1 hour
3.8	Machine Independent assembler features – Control sections and program linking. Illustration with example	1 hour
3.9	Assembler design options- One Pass assembler	1 hour
3.10	Multi pass assembler	1 hour
3.11	Implementation example: MASM Assembler	1 hour
Module-4 (Linker and Loader) (8 hours)		
4.1	Basic Loader functions - Design of absolute loader	1 hour
4.2	Simple bootstrap Loader	1 hour
4.3	Machine dependent loader features- Relocation	1 hour
4.4	Machine dependent loader features- Program Linking algorithm and data structures of First pass of two pass Linking Loader	1 hour
4.5	Machine dependent loader features- Program Linking algorithm and data structures of Second pass of two pass Linking Loader	1 hour
4.6	Machine independent loader feature - Automatic library search	1 hour
4.7	Machine independent loader features - Loader options	1 hour
4.8	Loader Design Option- Linking Loader, Linkage Editor, Dynamic Linking	1 hour
Module –5 (Macro Preprocessor, Device drivers, Text Editors, Debuggers) (9 hours)		
5.1	Macro Preprocessor- Macro Instruction Definition and Expansion	1 hour

5.2	One pass Macro processor algorithm and data structures	1 hour
5.3	One pass Macro processor Algorithm and data structures illustration with example	1 hour
5.4	Machine Independent Macro Processor Features- generation of unique labels, Concatenation of macro parameter, Keyword macro parameters	1 hour
5.5	Machine Independent Macro Processor Features- Conditional Macro Expansion	1 hour
5.6	Macro processor design options	1 hour
5.7	Device drivers- Anatomy of a device driver, Character and block device drivers, General design of device drivers	1 hour
5.8	Text Editors- Overview of Editing, User Interface , Editor Structure	1 hour
5.9	Debuggers :- Debugging Functions and Capabilities, Debugging Methods- By Induction, Deduction and Backtracking.	1 hour