DGITechChronicle

DGI TECH CHRONICLE

CSE EDITION

Volume II, Issue I (Jul-Dec) 2021

DGITechChronicle

EDITORIAL MESSAGE

Dr. Bipin Pandey
(HOD CSE)

Dear Readers,

Welcome to the latest edition of DGI Tech Chronicle. It gives me immense pleasure to present this issue, a culmination of diligent efforts from our talented contributors, showcasing the innovative spirit and technical prowess of our college community.

In these pages, you'll find a diverse array of articles, research findings, and insights that reflect the ever-evolving landscape of technology. As an engineering college community, we stand at the forefront of technological breakthroughs, and it is our mission to empower youths with the knowledge and insights to not only keep pace but to lead in this ever-accelerating race of innovation.

As we continue to push the boundaries of innovation, I extend my gratitude to everyone involved in bringing this publication to life – the writers, editors, designers, and all contributors who have made this edition a reality.

Each piece of writing embodies the spirit of exploration & advancement that defines the creativity of budding technocrats of our institution.

As readers, you play an integral role in this journey. Your curiosity & engagement drive us to explore deeper, question further, and innovate beyond boundaries. We hope this edition complement with your interest, sparks new ideas, and fosters a deeper appreciation for the transformative power of technology.

Warm regards,
Dr. Bipin Pandey
Editor-in-Chief, DGI Tech Chronicle

EDITORIAL BOARD

Editor in Chief

Dr. Bipin Pandey
(HOD CSE)

In this issue, we delve into a captivating array of topics and developments, all tailored to the inquisitive minds of the future engineers. As an engineering college community, we stand at the forefront of technological breakthroughs, and it is our mission to empower you with the knowledge and insights to not only keep pace but to lead in this ever-accelerating race of innovation.

Lovekesh Kumar
(13054; CSE)

Co- Editor

Smita Singh
(13106; CSE)

Editor- Design

Abhishek Singh
(13006; CSE)

Editor- Text

DGITechChronicle

CSE EDITION

TABLE OF CONTENT

VISION

Empowering the future with innovative advancements in computer science technology, fostering human values and principles, through technical education.

MISSION

Advancing computer science technology through cutting-edge research and education, driving innovation and excellence.

Promoting a culture of ethical responsibility and human-centered design in technology, ensuring advancements benefit society.

Empowering individuals with technical skills and values to create a sustainable and impactful future in the digital age.

Program Educational Objectives (PEO)

To enable graduates to pursue higher education and research, or have a successful career in industries associated with Computer Science and Engineering, or as entrepreneurs.

To ensure that graduates will have the ability and attitude to adapt to emerging technological changes.

To prepare students to analyze existing literature in an area of specialization and ethically develop innovative methodologies to solve the problems identified.

Programme Outcome (POs)

To analyze, design and develop computing solutions by applying foundational concepts of Computer Science and Engineering.

To apply software engineering principles and practices for developing quality software for scientific and business applications.

To adapt to emerging Information and Communication Technologies (ICT) to innovate ideas and solutions to existing/novel problems.

Programme Outcome (POs)

Po1. Engineering knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

PO2. Problem analysis: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

PO3. Design/development of solutions: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

PO4. Conduct investigations of complex problems: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

PO5. Modern tool usage: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

PO6. The engineer and society: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

PO7. Environment and sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

PO8. Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

PO9. Individual and team work: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

PO10. Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

PO11. Project management and finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

PO12. Life-long learning: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

Dual Mode Operation in OS

STUDENTS ARTICLES

Priyansh Kamal
(13069; CSE)

Dual Mode Operation, also known as Supervisor Mode and User Mode, is a feature integrated into operating systems to distinguish between privileged and non-privileged instructions. The primary purpose of dual mode operation is to prevent user-level programs from interfering with critical system resources and to ensure the stability and security of the overall system.

User Mode: In User Mode, applications and user-level programs run with restricted access to the system's resources. User Mode is designed to provide a safe environment for applications to execute without directly manipulating critical hardware components. Processes in User Mode have limited permissions, preventing them from tampering with the kernel or other sensitive system data.

Supervisor Mode: Supervisor Mode, on the other hand, is the privileged mode where the operating system kernel operates. In this mode, the OS has unrestricted access to all system resources and can execute privileged instructions that are typically off-limits to user-level programs. Supervisor Mode is crucial for managing system resources, handling interrupts, and ensuring the overall stability and security of the system.

Key Features and Benefits:

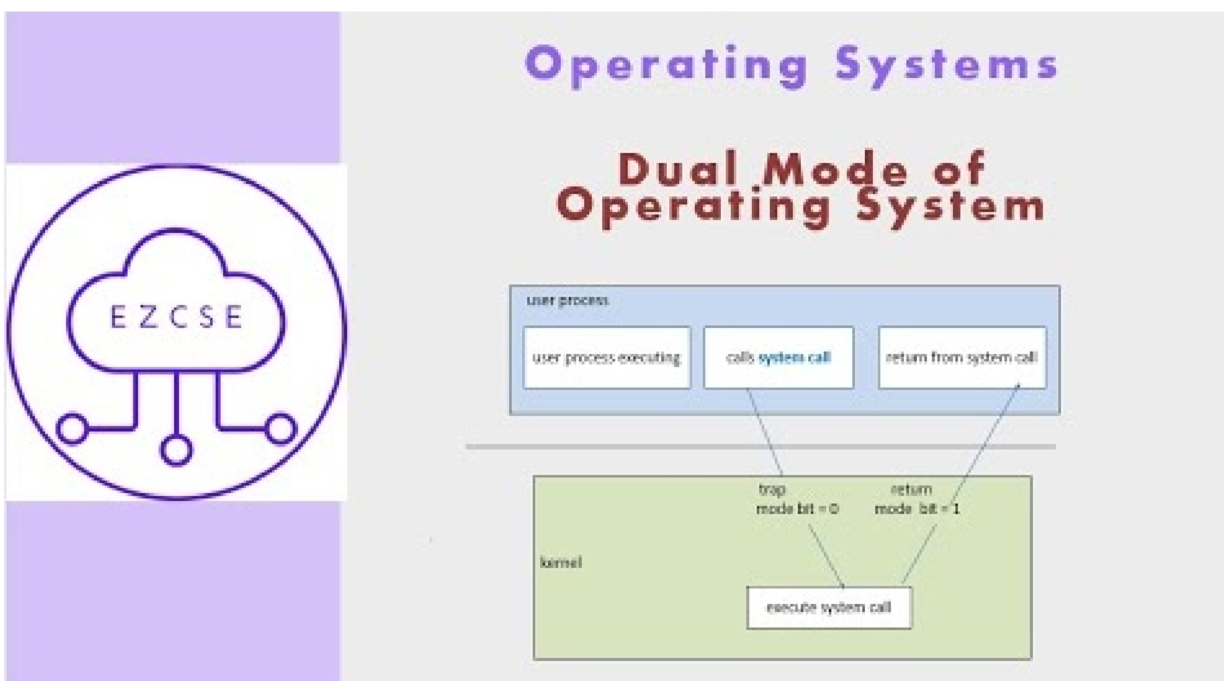
Isolation: Dual Mode Operation provides a robust mechanism for isolating user-level applications from critical system resources. By restricting the access of user programs, the OS ensures that accidental or malicious actions within applications do not compromise the integrity of the entire system.

Security: The separation of User Mode and Supervisor Mode enhances system security. Critical operations, such as modifying hardware configurations or accessing sensitive data, can only be performed in Supervisor Mode, reducing the risk of unauthorized access and potential security breaches.

Stability: Dual Mode Operation contributes to system stability by preventing user-level programs from directly affecting the core functionalities of the OS. This ensures that the operating system remains operational and responsive even when individual applications encounter issues.

Resource Management: In Supervisor Mode, the OS has full control over system resources, allowing it to efficiently manage tasks like memory allocation, process scheduling, and device communication. This enables the OS to optimize resource utilization for improved performance.

Challenges and Considerations: While Dual Mode Operation is a fundamental aspect of modern operating systems, it is not without challenges. Ensuring that the transition between User Mode and Supervisor Mode is seamless and secure is crucial. Additionally, managing the balance between providing flexibility to user-level applications and maintaining system integrity requires careful design and implementation.



Dual Mode Operation is a cornerstone feature in operating systems, providing a crucial layer of security and stability. By dividing the execution environment into User Mode and Supervisor Mode, OS architects can create a robust platform that allows applications to run safely while maintaining control over critical system resources. As computing technologies continue to evolve, the importance of Dual Mode Operation remains steadfast in ensuring the reliability and security of our digital systems.

Computer System Level Hierarchy

STUDENTS ARTICLES

Shivani Dwivedi
(13099; CSE)

User Level: At the top of the hierarchy is the user level, representing the interface between the computer system and the end user. Users interact with applications and software at this level, performing tasks such as word processing, web browsing, or running specific software applications. Users are generally not concerned with the lower levels of the system but rather interact with the system through graphical user interfaces (GUIs) or command-line interfaces.

Application Level: The application level is where software applications are executed. This includes a wide range of programs such as word processors, web browsers, games, and more. Application developers work at this level, creating software that leverages the capabilities provided by the lower levels of the system.

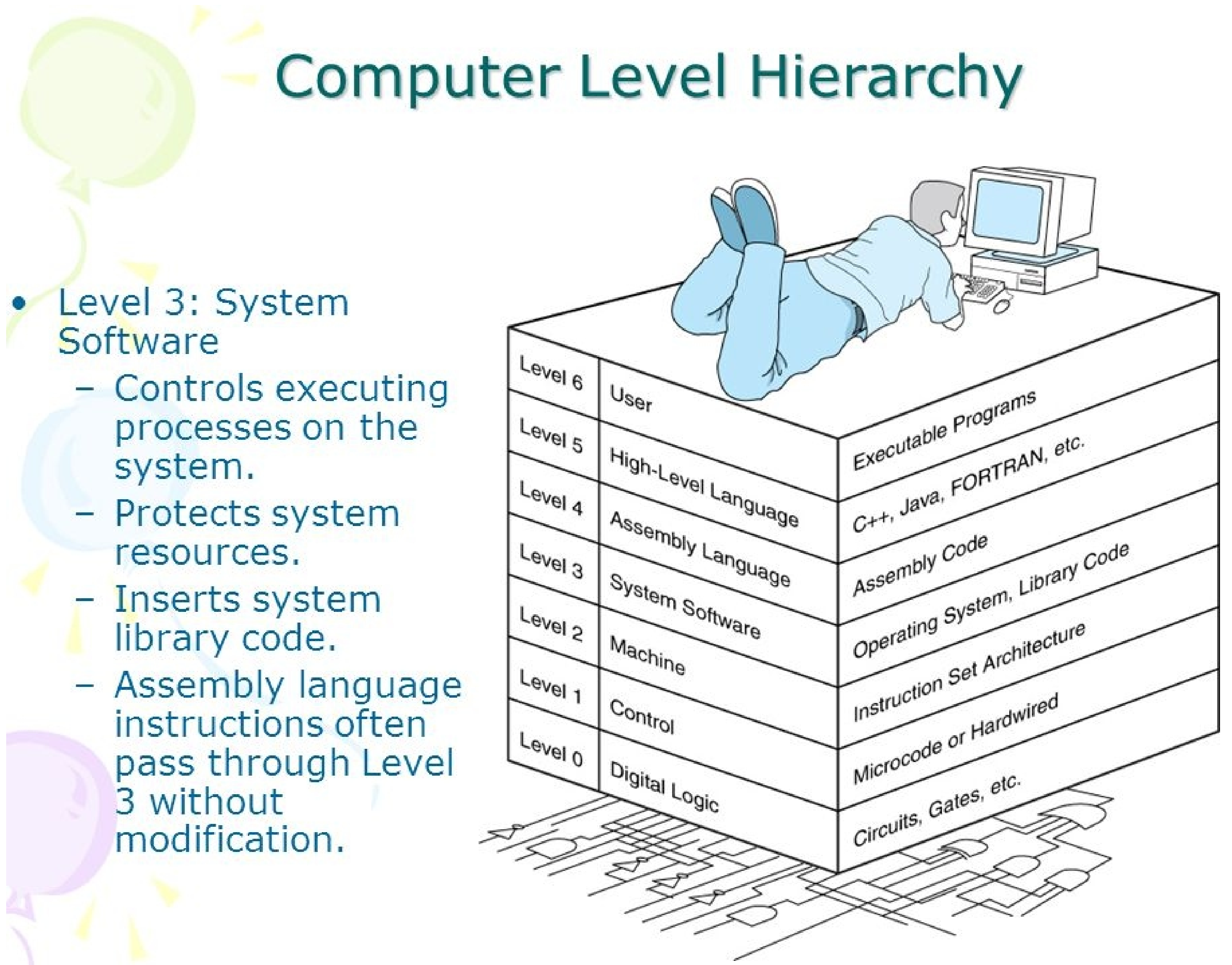
Operating System Level: The operating system (OS) level is an intermediary layer between application software and the computer hardware. The OS manages hardware resources, provides services to applications, and ensures a secure and stable environment for software execution. Tasks at this level include process management, memory management, file system management, and device management.

Hardware Abstraction Level: This level abstracts the details of the underlying hardware, providing a consistent interface for the operating system and applications. It includes components such as device drivers and the hardware abstraction layer (HAL). The goal is to shield higher-level software from the specifics of the hardware, allowing software to be more portable across different hardware configurations.

Instruction Set Architecture (ISA) Level: ISA defines the set of instructions that a processor can execute. It acts as a boundary between hardware and software, providing a standardized interface for software developers. The ISA level includes the architecture of the CPU, registers, instruction formats, and addressing modes.

Microarchitecture Level: Also known as the processor organization level, this level deals with the internal design and implementation details of the CPU. It includes elements such as the control unit, ALU, registers, and pipelines. Microarchitecture focuses on how the CPU interprets and executes instructions at the hardware level.

Digital Logic Level: The lowest level of the hierarchy deals with the digital logic circuits that implement the microarchitecture. This level involves the design of electronic circuits, including gates, flip-flops, and other components that make up the digital logic necessary for processing and storing information.



Simplified Instructional Computer (SIC)

Praveen Kr. Chandra
(13075; CSE)

The Simplified Instructional Computer (SIC) is a hypothetical computer architecture designed for educational purposes to teach the fundamentals of computer organization and assembly language programming. It was introduced by W. A. Wolf and C. S. Papadimitriou in the 1960s.

Here are some key characteristics of the Simplified Instructional Computer (SIC):

Word Size: The SIC has a fixed word size of 3 bytes, meaning each memory location or instruction consists of three bytes.

Memory: SIC features a limited address space with 8,192 (2^13) memory locations, each capable of holding a three-byte word.

Registers: SIC has a small set of registers, including an Accumulator (AC), a Memory Buffer Register (MBR), and an Instruction Counter (IC). The Accumulator is used for arithmetic operations, and the Memory Buffer Register holds data during memory transfers.

Instruction Format: Instructions in SIC have a fixed format, typically consisting of an operation code (opcode) and, in some cases, an address or operand field.

Instruction Set:The instruction set of SIC is intentionally kept simple for educational purposes. It includes basic operations such as addition, subtraction, and comparison. Branching and conditional jump instructions are also part of the instruction set.

Addressing Modes: SIC supports direct addressing, where the operand's address is specified in the instruction. Immediate addressing is also supported for constant values.

Assembler Directives:Assemblers for the SIC often include directives for program control, symbol definition, and addressing mode specification.

Input/Output:SIC provides simple input/output operations. For example, it may have instructions for reading from input devices and writing to output devices.

Assembler and Assembly Language: Assembly language for the SIC is designed to be relatively straightforward, making it suitable for educational purposes. Students can learn the basics of assembly language programming using the SIC model.

Teaching Tool : The primary purpose of the SIC is to serve as a tool for teaching computer organization, machine architecture, and assembly language programming concepts. Its simplicity makes it an ideal starting point for students learning about the fundamentals of computing.

STUDENTS ARTICLES

Enhanced ER Model

Aman Chawla
(13017; CSE)

The Enhanced Entity-Relationship (ER) Model is an extension of the original Entity-Relationship Model, a conceptual data modeling technique used in software engineering to represent the structure of a database. The Enhanced ER Model incorporates additional concepts and features to capture more complex relationships and constraints in a database design. It is often used in the early stages of database development to provide a high-level view of the data requirements.

Entities: Entities represent real-world objects or concepts that have data to be stored in the database. In the Enhanced ER Model, entities can have attributes that describe their properties.

Attributes: Attributes are properties or characteristics of entities. In the Enhanced ER Model, attributes can be classified as simple (atomic), composite (made up of sub-parts), and derived (calculated from other attributes). This allows for a more detailed description of the data associated with entities.

Relationships: Relationships describe connections or associations between entities. Enhanced ER allows for more complex relationship types, including binary, ternary, and higher-degree relationships. Relationships can also have attributes, known as descriptive attributes, which provide additional information about the relationships.

Cardinality and Participation Constraints: Cardinality defines the number of instances of one entity that can be associated with another entity. Enhanced ER allows for more precise specification of cardinality, such as one-to-one, one-to-many, and many-to-many relationships. Participation constraints specify the minimum and maximum number of entity occurrences that must participate in a relationship.

Weak Entities: Weak entities are entities that do not have a primary key attribute. In the Enhanced ER Model, weak entities can have partial key attributes, which are used in conjunction with the primary key of the identifying entity to uniquely identify instances of the weak entity.

Subtypes and Supertypes: Subtypes and supertypes allow for the modeling of inheritance hierarchies within the data model. Entities can be organized into a hierarchy, where a supertype represents a generalization, and subtypes represent specializations of the supertype. This supports the concept of generalization and specialization.

Aggregation: Aggregation is a modeling concept that allows treating a group of related entities and their relationships as a single entity. This is useful when you want to represent a higher-level concept that is composed of smaller, related entities.

Recursive Relationships: The Enhanced ER Model allows for the representation of recursive relationships, where an entity is related to itself through a specific relationship type. This is useful in scenarios where entities have relationships with other instances of the same entity.