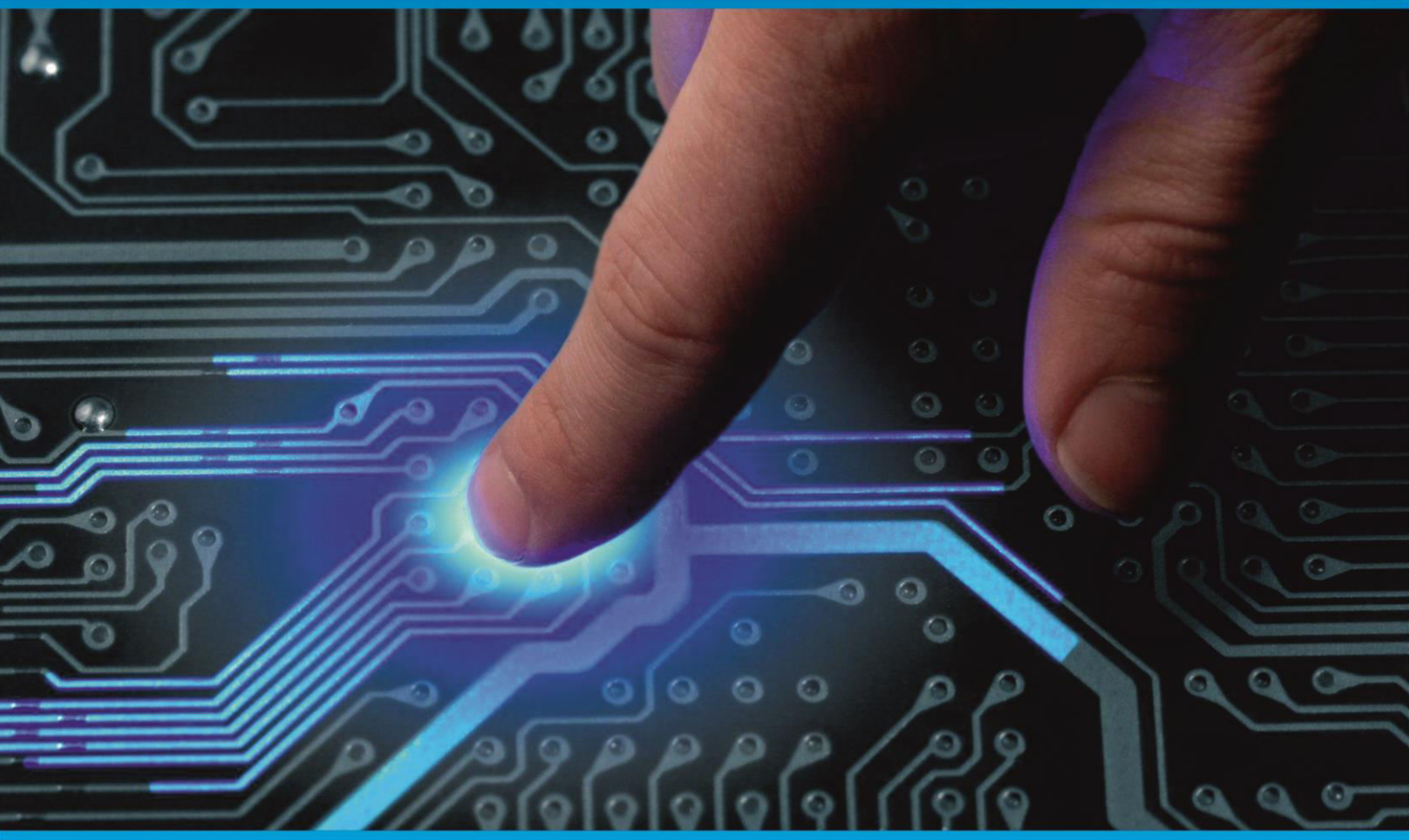
IJIRCCE

e-ISSN: 2320-9801 | p-ISSN: 2320-9798



INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN COMPUTER & COMMUNICATION ENGINEERING

Volume 9, Issue 2, February 2021

INTERNATIONAL STANDARD SERIAL NUMBER INDIA

Impact Factor: 7.488

International Journal of Innovative Research in Computer and Communication Engineering

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 7.488 |

|| Volume 9, Issue 2, February 2021 ||

| DOI: 10.15680/IJIRCCE.2021.0902041 |

A Comprehensive Review on the Characteristics and Life Cycle Models of an Embedded System

Koppula Geetha[1]

Assistant Professor, Department of ECE, SVS Group of Institutions, India[1]

ABSTRACT: In Organizational development, performance improvement is the concept of organizational change in which the managers and governing body of an organization put into place and manage a program which measures the current level of performance of the organization and then generates ideas for modifying organizational behavior and infrastructure which are put into place to achieve higher output. The primary goals of organizational improvement are to increase organizational effectiveness and efficiency to improve the ability of the organization to deliver goods and or services. This paper provides a comprehensive review on the characteristics and life cycle models of an embedded system.

KEYWORDS: Embedded systems, life cycle models, characteristics.

I. EMBEDDED SYSTEM DESIGN AND DEVELOPMENT

Basic Structure of an Embedded System

The following illustration shows the basic structure of an embedded system:

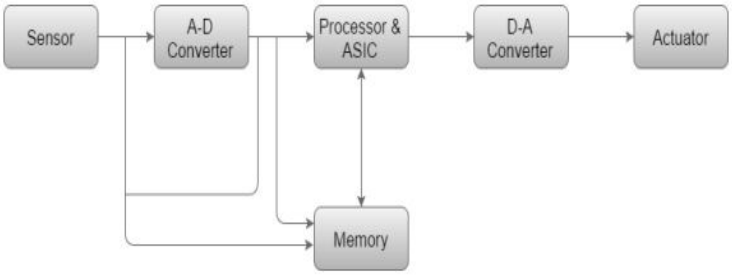


Figure 1

- **Sensor** – It measures the physical quantity and converts it to an electrical signal which can be read by an observer or by any electronic instrument like an A2D converter. A sensor stores the measured quantity to the memory.

- **A-D Converter** – An analog-to-digital converter converts the analog signal sent by the sensor into a digital signal.

- **Processor & ASICs** – Processors process the data to measure the output and store it to the memory.

- **D-A Converter** – A digital-to-analog converter converts the digital data fed by the processor to analog data.

Actuator – An actuator compares the output given by the D-A Converter to the actual (expected) output stored in it and stores the approved output.

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 7.488 |

|| Volume 9, Issue 2, February 2021 ||

| DOI: 10.15680/IJIRCCE.2021.0902041 |

II. CHARACTERISTICS OF AN EMBEDDED SYSTEM

- **Single-functioned** – An embedded system usually performs a specialized operation and does the same repeatedly. For example: A pager always functions as a pager.
- **Tightly constrained** – All computing systems have constraints on design metrics, but those on an embedded system can be especially tight. Design metrics is a measure of an implementation's features such as its cost, size, power, and performance. It must be of a size to fit on a single chip, must perform fast enough to process data in real time and consume minimum power to extend battery life.
- **Reactive and Real time** – Many embedded systems must continually react to changes in the system's environment and must compute certain results in real time without any delay. Consider an example of a car cruise controller; it continually monitors and reacts to speed and brake sensors. It must compute acceleration or de- accelerations repeatedly within a limited time; a delayed computation can result in failure to control of the car.
- **Microprocessors based** – It must be microprocessor or microcontroller based.
- **Memory** – It must have a memory, as its software usually embeds in ROM. It does not need any secondary memories in the computer.
- **Connected** – It must have connected peripherals to connect input and output devices.
- **HW-SW systems** – Software is used for more features and flexibility. Hardware is used for performance and security.

Advantages

- Easily Customizable
- Low power consumption
- Low cost
- Enhanced performance

Disadvantages

- High development effort
- Larger time to market

III. LIFE CYCLE MODELS

The fundamentals of design are

- Find out what the customers want.
- Think of a way to give them what they want.
- Prove what you have done by building and testing it.
- Build a lot of the product to prove that it wasn‘t an accident. Use the product to solve the customer‘s problem.

The common life cycle models are:

- Waterfall model V cycle model. Spiral

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 7.488 |

|| Volume 9, Issue 2, February 2021 ||

| DOI: 10.15680/IJIRCCE.2021.0902041 |

- Rapid prototype

Waterfall model

The waterfall model represents a cycle- specifically a series of steps appearing much like a waterfall. It is the model which is use to linear process development. It is a sequential design process, often used in software process development in which progress is seen as flowing steadily downwards through the phases of Conception,Initiation, Analysis, Design, Construction, Testing, Production / Implementation and Maintenance.

The waterfall development model originates in the manufacturing and construction industries: highly structured physical environments in which after-the-fact changes are prohibitively costly, if not impossible. Since no formal software development methodologies existed at the time, this hardware-oriented model was simply adapted for software development.

The steps are:

Phases:

Specification. Preliminary design. Design review Detailed design.
Design review. Implementation. Review.

1. Requirement : In this phase we gather necessary information which will use for development of any project . For above example we gather information like which types of characteristics client wants. It also defines system requirement specification. This phase defines what to do.

2. Design: In design phase we then construct design to how to implement that requirements gathered into phase 1 .This phase define how to do .For this phase we then write algorithms

3. Coding: Now base on design phase we then write actual code to implement algorithms. This code should be efficient.

4. Testing : This phase use to test our coding part it checks all the validation...like our code should work for each and every possibilities of input if any bug occur then we have to report that bug to design phase or development phase.

5. Maintenance: In this phase we need keep updating information.

1. The implementation process contains software preparation and transition activities, such as the conception and creation of the maintenance plan; the preparation for handling problems identified during development; and the followup on product configuration management.

2. The problem and modification analysis process, which is executed once the application has become the responsibility of the maintenance group. The maintenance programmer must analyze each request, confirm it (by reproducing the situation) and check its validity, investigate it and propose a solution, document the request and the solution proposal, and, finally, obtain all the required authorizations to apply the modifications.

3. The process considering the implementation of the modification itself.

4. The process acceptance of the modification, by confirming the modified work with the individual who submitted the request in order to make sure the modification provided a solution.

5. The migration process is exceptional, and is not part of daily maintenance tasks. If the software must be ported to another platform without any change in functionality, this process will be used and a maintenance project team is likely to be assigned to this task.

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 7.488 |

|| Volume 9, Issue 2, February 2021 ||

| DOI: 10.15680/IJIRCCE.2021.0902041 |

6. Finally, the last maintenance process, also an event which does not occur on a daily basis, is the retirement of a piece of software.

The V model

The V-Model is a term applied to a range of models, from a conceptual model designed to produce a simplified understanding of the complexity associated with systems development to detailed, rigorous development lifecycle models and project management models. Each test phase is identified with its matching development phase as shown in figure 2.

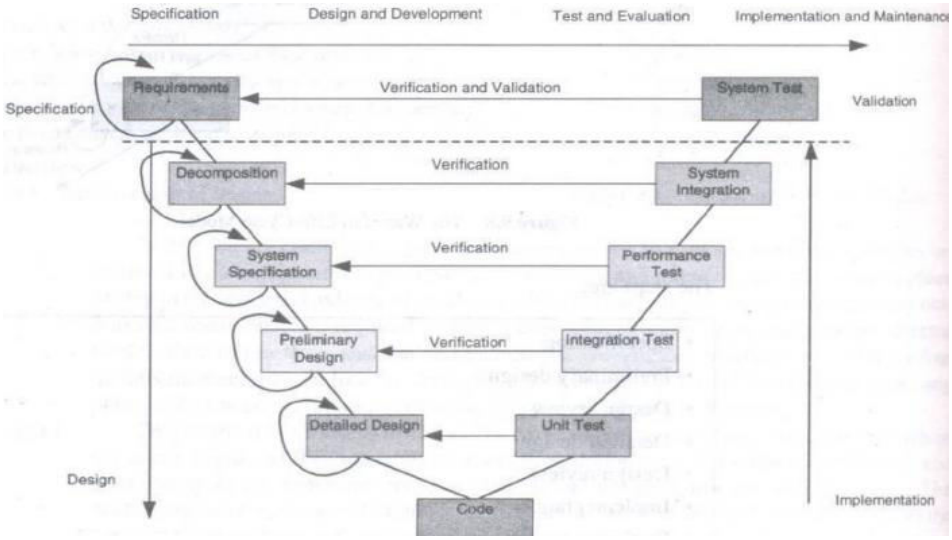


Figure 2: The V life cycle model

The V-model is a graphical representation of the systems development lifecycle. It summarizes the main steps to be taken in conjunction with the corresponding deliverables within computerized system validation framework.

The V represents the sequence of steps in a project life cycle development. It describes the activities to be performed and the results that have to be produced during product development. The left side of the "V" represents the decomposition of requirements, and creation of system specifications. The right side of the V represents integration of parts and their validation

Validation. The assurance that a product, service, or system meets the needs of the customer and other identified stakeholders. It often involves acceptance and suitability with external customers. Contrast with verification."

"Verification. The evaluation of whether or not a product, service, or system complies with a regulation, requirement, specification, or imposed condition. It is often an internal process. Contrast with validation."

The spiral model

The spiral model is a software development process combining elements of both design and prototyping-in-stages, in an effort to combine advantages of top-down and bottom-up concepts. Also known as the spiral lifecycle model (or spiral development), it is a systems development method (SDM) used in information technology (IT). This model of development combines the features of the prototyping and the waterfall model. The spiral model is intended for large, expensive and complicated projects.

The steps in spiral model life cycle are Determine objective, alternatives, and constraints. Identify and resolve risks.

Evaluate alternatives.

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 7.488 |

|| Volume 9, Issue 2, February 2021 ||

| DOI: 10.15680/IJIRCCE.2021.0902041 |

Develop deliverables-verify that they are correct. Plan the next iteration.
Commit to an approach for the next iteration.

The spiral model combines the idea of iterative development (prototyping) with the systematic, controlled aspects of the waterfall model. It allows for incremental releases of the product, or incremental refinement through each time around the spiral. The spiral model also explicitly includes risk management within software development. Identifying major risks, both technical and managerial, and determining how to lessen the risk helps keep the software development process under control.

The spiral model is based on continuous refinement of key products for requirements definition and analysis, system and software design, and implementation (the code). At each iteration around the cycle, the products are extensions of an earlier product. This model uses many of the same phases as the waterfall model, in essentially the same order, separated by planning, risk assessment, and the building of prototypes and simulations

Documents are produced when they are required, and the content reflects the information necessary at that point in the process. All documents will not be created at the beginning of the process, nor all at the end (hopefully). Like the product they define, the documents are works in progress. The idea is to have a continuous stream of products produced and available for user review.

The spiral lifecycle model allows for elements of the product to be added in when they become available or known. This assures that there is no conflict with previous requirements and design. This method is consistent with approaches that have multiple software builds and releases and allows for making an orderly transition to a maintenance activity. Another positive aspect is that the spiral model forces early user involvement in the system development effort. For projects with heavy user interfacing, such as user application programs or instrument interface applications, such involvement is helpful

Note that the requirements activity takes place in multiple sections and in multiple iterations, just as planning and risk analysis occur in multiple places. Final design, implementation, integration, and test occur in iteration 4. The spiral can be repeated multiple times for multiple builds. Using this method of development, some functionality can be delivered to the user faster than the waterfall method. The spiral method also helps manage risk and uncertainty by allowing multiple decision points and by explicitly admitting that all of anything cannot be known before the subsequent activity starts.

Rapid prototype

The Rapid prototyping model is intended to provide a rapid implementation of high level portions of both the software and the hardware . the approach allows developers to construct working portion of hardware and software in incremental stages.Each stage through the cycle,one incorporates a little more of the intended functionality.The prototype is useful for both the designer and the customer. The prototype can be either evolutionary or throughway. It has the advantage of having a working system early in development process.

IV. PROBLEM SOLVING-FIVE STEPS TO DESIGN

The 5 steps to a successful design are Requirement definition.

System specification Functional design Architectural design Prototyping.

The design process

The design process comprises five distinct stages although it may vary for particular projects or design disciplines. This information may be useful when working with a designer to understand the processes involved. Before the project is started however, a vital question has to be asked:
—Why do you need a new identity, brochure or website etc?‖ This question is the key to undertaking a successful project.

Identifying and formulating the requirement specification

Requirements analysis in systems engineering and software engineering, encompasses those tasks that go into determining the needs or conditions to meet for a new or altered product, taking account of the possibly

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 7.488 |

|| Volume 9, Issue 2, February 2021 ||

| DOI: 10.15680/IJIRCCE.2021.0902041 |

conflicting requirements of the various stakeholders, analyzing, documenting, validating and managing software or system requirements.

Requirements analysis is critical to the success of a systems or software project. The requirements should be documented, actionable, measurable, testable, traceable, related to identified business needs or opportunities, and defined to a level of detail sufficient for system design.

Conceptually, requirements analysis includes three types of activities

Eliciting requirements: the task of identifying the various types of requirements from various sources including project documentation, business process documentation, and stakeholder interviews. This is sometimes also called requirements gathering.

Analyzing requirements: determining whether the stated requirements are clear, complete, consistent and unambiguous, and resolving any apparent conflicts.

Recording requirements: Requirements may be documented in various forms, usually including a summary list and may include natural-language documents, use cases, user stories, or process specifications.

Characterizing the system

Requirements analysis can be a long and arduous process during which many delicate psychological skills are involved. New systems change the environment and relationships between people, so it is important to identify all the stakeholders, take into account all their needs and ensure they understand the implications of the new systems. Analysts can employ several techniques to elicit the requirements from the customer. These may include the development of scenarios, the identification of use cases, the use of workplace observation or ethnography, holding interviews, or focus groups and creating requirements lists. Prototyping may be used to develop an example system that can be demonstrated to stakeholders. Where necessary, the analyst will employ a combination of these methods to establish the exact requirements of the stakeholders, so that a system that meets the business needs is produced.

The specification of the external environment should contain the following for each entity: Name and description of the entity.

For each I/O variable, the following information is available

¬
The name of the signal.

the use of the signal as an i/p or o/p.

¬
The nature of the signal as an event,data,state variable.
Responsibilities-activities.

Relationships. Safety and reliability.

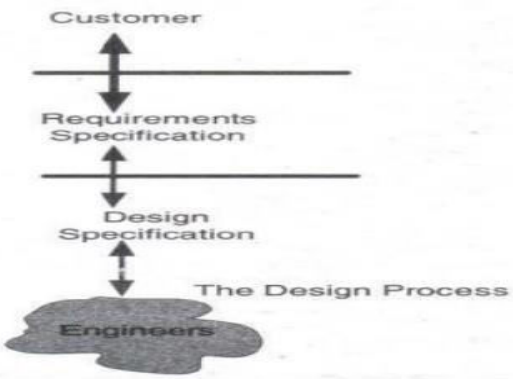


Figure 3: The Customer, the requirement, the design and the engineer

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 7.488 |

|| Volume 9, Issue 2, February 2021 ||

| DOI: 10.15680/IJIRCCE.2021.0902041 |

The system design specification

The System Design Specification (SDS) is a complete document that contains all of the information needed to develop the system. Systems design is the process of defining the architecture, components, modules, interfaces, and data for a system to satisfy specified requirements. Systems design could be seen as the application of systems theory to product development. There is some overlap with the disciplines of systems analysis, systems architecture and systems engineering. System design specification serves as a bridges between the customers and designers as shown in figure 3.

The requirement specifications provides a view from the outside of the system, design specification provides a view from the inside looking out as well. Design specification has 2 masters:

It must specify the system‘s public interface from inside the system.

It must specify how the requirements defined for and by the public interface are to be met by the initial functions of the system.

Five areas should be considered are:

Geographical constraints.

Characterization of and constraints on interface signals. User interface requirements
Temporal constraints.

Electrical infrastructure consideration Safety and reliability

V. SYSTEM SPECIFICATION VERSUS SYSTEM REQUIREMENTS

Requirements give a description of something wanted or needed. They are a set of needed properties.¬
¬
Specification is a description of some entity that has or implements those properties.

A System Requirements Specification is a structured collection of information that embodies the requirements of a system.

Requirements and specifications are very important components in the development of any embedded system. Requirements analysis is the first step in the system design process, where a user's requirements should be clarified and documented to generate the corresponding specifications. While it is a common tendency for designers to be anxious about starting the design and implementation, discussing requirements with the customer is vital in the construction of safety-critical systems. For activities in this first stage has significant impact on the downstream results in the system life cycle.

For example, errors developed during the requirements and specifications stage may lead to errors in the design stage. When this error is discovered, the engineers must revisit the requirements and specifications to fix the problem. This leads not only to more time wasted but also the possibility of other requirements and specifications errors. Many accidents are traced to requirements flaws, incomplete implementation of specifications, or wrong assumptions about the requirements. While these problems may be acceptable in non-safety-critical systems, safety- critical systems cannot tolerate errors due to requirements and specifications. Therefore, it is necessary that the requirements are specified correctly to generate clear and accurate specifications.

There is a distinct difference between requirements and specifications. A requirement is a condition needed by a user to solve a problem or achieve an objective. A specification is a document that specifies, in a complete, precise, verifiable manner, the requirements, design, behavior, or other characteristics of a system, and often, the procedures for determining whether these provisions have been satisfied. For example, a requirement for a car could be that the maximum speed to be at least 120mph. The specification for this requirement would include technical information about specific design aspects. Another term that is commonly seen in books and papers is requirements specification which is a document that specifies the requirements for a system or

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 7.488 |

|| Volume 9, Issue 2, February 2021 ||

| DOI: 10.15680/IJIRCCE.2021.0902041 |

component. It includes functional requirements, performance requirements, interface requirements, design requirements, and developement standards.

A specification is a precise description of the system that meets stated requirements. A specification document should be

Complete Consistent Comprehensible
Traceable to the requirement Unambiguous
Modifiable

Able to be written

Functional design

The functional design is not concerned with the software or hardware that will support the operation of the product or the physical organization of the data or the programs that will accept the input data, execute the processing rules, and produce the required output.

Functional Design is a paradigm used to simplify the design of hardware and software devices such as computer software and increasingly, 3D models. A functional design assures that each modular part of a device has only one responsibility and performs that responsibility with the minimum of side effects on other parts. Functionally designed modules tend to have low coupling.

Architectural design

The major objective of the Architectural design activity is the allocation or mapping of the different pieces of system functionality to the appropriate hardware anf software blocks. Work is based on the detailed functional structure. The important constraints that must be considered include items as

The geographical distribution. Physical and user interfaces System performance specifications.
Timing constraints and dependability requirements Power consumption
Legacy components and cost.

Hardware and software specification and design

For the software design, the following must be analyzed and decided.

Whether to use a real time kernel.

Whether several functions can be combined in order to reduce the number of software tasks and if so, how?
A priority for each task.

An implementation technique for each intertask relationship.
The important criteria that we strive to optimize are Implementation cost

Development time and cost

Performance and dependability constraints Power consumption

Functional model versus architectural model

An appropriate model has to include elements both at the functional and architectural level to be able to represent and evaluate hardware/software system.

Functional model

The functional model describes a system through a set of interacting functional elements. The design proceeds at a high level without initial bias toward any specific implementation. We have freedom to explore and to be creative. The functional models will interact using one of the following 3 types of relations

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 7.488 |

|| Volume 9, Issue 2, February 2021 ||

| DOI: 10.15680/IJIRCCE.2021.0902041 |

The shared variable relation-which defines a data exchange without temporal dependencies.
The synchronization relation- which specifies temporal dependency.

The message transfer by port- which implies a producer/consumer kind of relationship.

Architectural model

Tha architectural model describes the physical architecture of the system based on real components such as microprocessor, arrayed logics, special purpose processors,analog and digital components, and the many interconnections between them.

VI. CONLUSION

The functional design process maps the "what to do" of the Requirements Specification into the "how to do it" of the design specifications. During this stage, the overall structure of the product is defined from a functional viewpoint. The functional design describes the logical system flow, data organization, system inputs and outputs, processing rules, and operational characteristics of the product from the user's point of view. This paper provided a comprehensive review on the characteristics and life cycle models of an embedded system.

REFERENCES

1. S. Niu, J. Zhang, B. Wang, and X. Fu, “Analysis and implementation of migrating real-time embedded operating system freertos kernel based on s3c44b0 processor,” in Fourth International Symposium on Information Science and Engineering, 2012, pp. 430 – 433.
2. P. Huang, “Implementation of porting rtos FreeRTOS to arm7,” in J. Chinese electronic business communications market, June 2009, pp. 59 – 64.
3. R. Barry, “Real time engineering ltd: FreeRTOS main website, http://www.freertos.org.”
4. Roopha Shree Kollolu Srinivasa, “DEVELOPMENTS IN WIRELESS NETWORKING TECHNOLOGY AND STUDY ON GERMAN RESEARCHER TEST 40 GBPS WIRELESS BROADBAND”, Wutan Huatan Jisuan Jishu, Volume XIV, Issue II, February 2018.
5. Roopha Shree Kollolu Srinivasa, “REPRESENTATIONOFMAN-IN-MIDDLEATTACK AND WLAN SECURITY ATTACKS”, “Science, Technology and Development “, Volume VIII Issue XII DECEMBER 2019.
6. Roopha Shree Kollolu Srinivasa, “HISTORY, DEPLOYMENT AND SERVICE MODELS TOWARDS THE EVOLUTION OF CLOUD COMPUTING”, Journal of Interdisciplinary Cycle Research, Volume XII, Issue III, March 2020.
7. Roopha Shree Kollolu Srinivasa, “INFRASTRUCTURAL CONSTRAINTS OF CLOUD COMPUTING”, International Journal of Management, Technology And Engineering, Volume X, Issue XII, DECEMBER 2020.
8. Roopha Shree Kollolu Srinivasa, “A REVIEW ONWIDE VARIETY AND HETEROGENEITY OF IOT PLATFORMS”, The International journal of analytical and experimental modal analysis, Volume XII, Issue I, January 2020
9. Roopha Shree Kollolu Srinivasa, “RISKANALYSIS OF PUTTING ATTACKS INTO PERSPECTIVE AND CONDUCTING A VULNERABILITYASSESSMENT”, “Science, Technology and Development “, Volume VIII Issue XII DECEMBER 2019
10. Roopha Shree Kollolu Srinivasa, “TECHNOLOGIES AND ISSUES OF CLOUD COMPUTING”, Journal of Interdisciplinary Cycle Research, Volume XIII, Issue II, February 2021
11. Roopha Shree Kollolu Srinivasa, “CHARACTERISTICS, APPLICATIONS AND USE CASES OF CLOUD COMPUTING”, International Journal of Management, Technology And Engineering, Volume X, Issue VI, JUNE 2020
12. Roopha Shree Kollolu Srinivasa, “A REVIEW ON THE ADVANTAGES AND TYPES OF WIRELESS NETWORKS”, JAC : A Journal Of Composition Theory, Volume X, Issue II, 2017
13. Roopha Shree Kollolu Srinivasa, “CLASSIFICATIONS OF WIRELESS NETWORKING AND RADIO TRANSMISSION TECHNOLOGY”, Wutan Huatan Jisuan Jishu, Volume XIV, Issue XI, November 2018
14. Roopha Shree Kollolu Srinivasa, “A REVIEW ON THE COMPARISON OF CLOUD COMPUTING DEPLOYMENT MODELS”, JASC: Journal of Applied Science and Computations, Volume VIII, Issue VII, July 2021
15. Roopha Shree Kollolu Srinivasa, “RECENT RESEARCH DIRECTIONS TOWARDS INTERNET OF THINGS”, Wutan Huatan Jisuan Jishu Journal, Volume XVI, Issue I, January 2020
16. Roopha Shree Kollolu Srinivasa, “AN OVERVIEW ON THE IOT RESEARCH CHALLENGES”, JASC: Journal

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 7.488 |

|| Volume 9, Issue 2, February 2021 ||

| DOI: 10.15680/IJIRCCE.2021.0902041 |

of Applied Science and Computations, Volume VI, Issue VII, JULY 2019
17. Roopha Shree Kollolu Srinivasa, "A STUDY ON THE DIFFERENCES BETWEEN IOT AND TRADITIONAL NETWORK", JASC: Journal of Applied Science and Computations, Volume VI, Issue II, February 2019
18. Roopha Shree Kollolu Srinivasa, "WLAN TOTOLOGY AND COMPARISON BETWEEN WIRED AND WIRELESS NETWORK", Parishodh Journal, Volume VI, Issue V, May 2017
19. A. J. Pankaj Sagar, Naveen N, A. N. K. H, and L. M., "Implementation of cmnn based industrial controller using vxworks rtos ported to mpc8260," in Third International Conference on Advances in Computing and Communications, 2013, pp. 239 – 242.
20. Christensen HI (2013) A Roadmap for US Robotics - From Internet to Robotics.
21. SME robot (2007) White Paper on Trends and Challenges in Industrial Robot Automation.M. Haegele, Ed., ed: SMErobot consortium
22. ISO (2011) Robots and robotic devices - Safety requirements for industrial robots - Part 1: Robots.

INNO SPACE
SJIF Scientific Journal Impact Factor

Impact Factor:
7.488

ISSN INTERNATIONAL STANDARD SERIAL NUMBER INDIA

निस्केयर
NISCAIR

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN COMPUTER & COMMUNICATION ENGINEERING

9940 572 462 6381 907 438 ijircce@gmail.com

www.ijircce.com

Scan to save the contact details