

EMBEDDED SYSTEMS

Late Dr. RAJ KISHOR PRASAD
Dr. GAURI V. GHULE
Dr. PALLAVI D. DESHPANDE

EMBEDDED SYSTEMS

Late Dr. Raj Kishor Prasad

BE. (Honors), ME, MS, Ph.D

Ex. Head of Department

Department of E&TC Engineering,

Army Institute of Technology,

Dighi Hills. Pune.

Dr. Gauri V. Ghule

Ph.D, E&TC

Assistant Professor,

Department of E&TC,

Vishwakarma Institute of

Information Technology, Pune.

Dr. Pallavi D. Deshpande

Ph.D, E&TC

Assistant Professor,

Department of E&TC,

Vishwakarma Institute of

Information Technology, Pune.



Preface ...

Children need embedded systems to play smart video games and to operate automatic chocolate vending machines. Young people need embedded systems to borrow smart cards from parents to see movies. Housewives need embedded systems for smart Internet Compliant home appliances, such as, microwave, television, music systems and so on. The driver needs it for automatic cruise control of car. Organizations need embedded systems for networking and products. In developed countries, every house is equipped with more than hundred embedded systems.

In early days, embedded systems were designed using 8085 microprocessors. Applications were simple, for example data acquisition systems using ADC and DAC, temperature monitoring systems, music systems, simple robotic system using stepper motor.

Since the early eighties small-scale embedded systems have used microcontrollers. With the advent of SOC technology, we are now developing embedded systems using a single VLSI chip. These embedded systems are very smart and highly sophisticated. As example we have a smart card and smart digital cameras.

Embedded systems have definitions that vary with time and perceptions! An embedded system is defined as one that has computer hardware with software embedded in it as one of the most important components.

This book will be useful for engineering students, readers and keen learners of embedded systems. It is intended as a text book for students and a reference book for practicing engineers.

We also take this opportunity to express our sincere thanks to Shri Dineshbhai Furia, Shri Jignesh Furia, Mrs. Nirali Verma, Mrs. Deepali Lachake (coordinator), and the entire team at Nirali Prakashan for their keen interest and tireless efforts in publishing this book.

The advice and suggestions of our esteemed readers to improve the text are most welcome and will be highly appreciated.

Authors

Contents

| | | |
|-----------|--|-----------|
| 1. | INTRODUCTION TO EMBEDDED SYSTEMS | 1 |
| 1.1 | Introduction to Embedded Systems | 1 |
| 1.2 | Embedded System Design and Development | 11 |
| 1.3 | Development Tools for an Embedded System | 17 |
| 1.4 | System Design Specifications | 18 |
| 1.5 | System Specification Versus System Requirements | 19 |
| 1.6 | Partitioning and Decomposing a System | 19 |
| 1.7 | Functional Design | 20 |
| 1.8 | Architectural Design | 20 |
| 1.9 | Functional Model Versus Architectural Model | 21 |
| 1.10 | Prototyping | 21 |
| 1.11 | Other Consideration for the Design Process | 23 |
| 1.12 | Archiving the Project | 23 |
| • | Exercise | 24 |
| 2. | OVERVIEW OF EMBEDDED SYSTEMS | 25 |
| 2.1 | Introduction | 25 |
| 2.2 | Design Challenge | 29 |
| 2.3 | Processor Technology | 32 |
| 2.4 | IC Technology | 34 |
| 2.5 | Design Technology | 36 |
| 2.6 | Custom Single-Purpose Processors Technology | 37 |
| 2.7 | Sequential Logic | 41 |
| 2.8 | Custom Single-purpose processor Design | 42 |
| 2.9 | RT-level Custom Single – Purpose Processor Design | 45 |
| 2.10 | Optimizing Custom Single – Purpose Processors | 47 |
| • | Exercise | 48 |
| 3. | GENERAL PURPOSE PROCESSOR SOFTWARE | 49 |
| 3.1 | Introduction | 49 |
| 3.2 | Basic Architecture | 49 |
| 3.3 | Operation | 51 |
| 3.4 | Programmer's View | 52 |
| 3.5 | Development Environment | 56 |
| 3.6 | Application Specific Instruction-Set Processor (ASIPS) | 57 |
| 3.7 | Selecting a Microprocessor | 58 |
| 3.8 | General-Purpose Processor Design | 59 |
| 3.9 | Introduction to LPC 214X | 60 |
| • | Exercise | 64 |

| | | |
|-----------|--|------------|
| 4. | ARM PROCESSOR FUNDAMENTALS AND INSTRUCTION SET | 65 |
| 4.1 | Introduction | 65 |
| 4.2 | Registers | 66 |
| 4.3 | Current Program Status Register (CPSR) | 67 |
| 4.4 | Pipeline | 69 |
| 4.5 | Exception | 70 |
| 4.6 | Interrupts and Vector Table | 71 |
| 4.7 | Data Processing Instruction | 76 |
| 4.8 | Branch Instruction | 76 |
| 4.9 | Load-Store Instructions | 77 |
| 4.10 | Software Interrupt Instruction | 78 |
| 4.11 | Program Status Register Instructions | 79 |
| 4.12 | Loading Constants | 80 |
| 4.13 | Thumb Register Usage | 80 |
| 4.14 | ARM-Thumb Interworking | 81 |
| 4.15 | Other Branch Instructions | 81 |
| 4.16 | Thumb Data Processing Instructions | 82 |
| 4.17 | Stack Instructions | 83 |
| 4.18 | Thumb Single-Register Load-Store instructions | 83 |
| 4.19 | Thumb Multiple Register Load-Store Instructions | 83 |
| 4.20 | Thumb Software Interrupt Instructions | 84 |
| • | Exercise | 84 |
| 5. | COMMUNICATION PROTOCOLS | 85 |
| 5.1 | Communication Protocols | 85 |
| 5.2 | Use of Communication Protocols in Embedded System | 85 |
| 5.3 | Types of Communication Protocols in Embedded Systems | 85 |
| 5.4 | Inter System Communication Protocols | 86 |
| 5.5 | Intra System Communication Protocol | 88 |
| 5.6 | Serial Communication Basics | 92 |
| 5.7 | Synchronous/Asynchronous Interfaces | 93 |
| • | Exercise | 94 |
| 6. | SYSTEM CONTROL, GPIO AND TIMER COUNTER | 95 |
| 6.1 | System Control | 95 |
| 6.2 | External Interrupt Inputs | 100 |
| 6.3 | Other System Controls | 105 |
| 6.4 | VPB Divider | 109 |
| 6.5 | GPIO | 110 |
| 6.6 | Timer | 113 |
| • | Exercise | 122 |
| 7. | ARM COMMUNICATION PROTOCOL – UART, SPI & I²C | 123 |
| 7.1 | UART | 123 |
| 7.2 | UART1 | 137 |
| 7.3 | Serial Peripheral Interface (SPI) | 144 |
| 7.4 | I ² C Bus Interface | 149 |
| • | Exercise | 152 |

| | | |
|------------|---|------------|
| 8. | REAL TIME OPERATING SYSTEM (RTOS) | 153 |
| 8.1 | RTOS Fundamentals | 153 |
| 8.2 | Multitasking in Small Embedded Systems | 156 |
| 8.3 | Memory Management | 157 |
| 8.4 | Task Management | 158 |
| 8.5 | Queue Management | 162 |
| 8.6 | Software Timer Management | 165 |
| 8.7 | Interrupt Management | 166 |
| 8.8 | Resources Management | 167 |
| 8.9 | Event Groups | 167 |
| 8.10 | Task Notification | 167 |
| 8.11 | Design of Rudimentary Kernel Operating System | 168 |
| • | Exercise | 174 |
| 9. | EMBEDDED LINUX | 175 |
| 9.1 | Embedded Linux | 175 |
| 9.2 | System Architecture of Embedded Lynax | 175 |
| 9.3 | BIOS Versus Boot-Loader | 176 |
| 9.4 | Booting the Kernel | 176 |
| 9.5 | Kernel Initialization | 178 |
| 9.6 | Space Initialization | 179 |
| 9.7 | Boot-Loaders | 180 |
| 9.8 | Storage Consideration | 182 |
| 9.9 | Linux Kernel Construction | 184 |
| 9.10 | Kernel Build System | 188 |
| 9.11 | Obtaining a Custom Linux Kernel | 191 |
| 9.12 | File Systems | 191 |
| 9.13 | Device Drivers | 196 |
| 9.14 | Kernel Configuration | 202 |
| • | Exercise | 206 |
| 10. | RTOS & DESIGN EXAMPLE OF EMBEDDED SYSTEM | 207 |
| 10.1 | Introduction to RTOS | 207 |
| 10.2 | Process Scheduling | 208 |
| 10.3 | Examples of RTOS | 210 |
| 10.4 | Microprocessor and Microcontroller Based System Design | 210 |
| 10.5 | Design Examples | 212 |
| 10.6 | System Design and Simulation Software such as Proteus VSM | 218 |
| 10.7 | Digital Camera Example | 220 |
| 10.8 | Design of Digital Camera | 228 |
| • | Exercise | 234 |
| 11. | INTRODUCTION TO IoT BASED EMBEDDED SYSTEMS | 235 |
| 11.1 | Introductions to Embedded Systems | 235 |
| 11.2 | Real Time Scheduling | 243 |
| 11.3 | Processor Basics | 245 |
| 11.4 | Components of Embedded System | 250 |
| 11.5 | Introduction to Embedded Processor | 251 |
| • | Exercise | 255 |

| | | |
|------------|---|------------|
| 12. | INTERNET OF THINGS: CONCEPTS | 257 |
| 12.1 | Introduction to Internet of Things (IoT): Definition and Vision | 257 |
| 12.2 | Physical Design of IoT | 268 |
| 12.3 | Logical Design of IoT | 275 |
| 12.4 | Case study: Raspberry PI | 285 |
| • | Exercise | 304 |
| 13. | DESIGN OF IoT BASED EMBEDDED SYSTEMS | 305 |
| 13.1 | Introduction | 305 |
| 13.2 | Basics of IoT Networking | 307 |
| 13.3 | IoT Communication Models | 316 |
| 13.4 | IoT Communication APIs | 318 |
| 13.5 | Sensor Networks | 319 |
| 13.6 | Four Pillars of IoT | 320 |
| 13.7 | Case Study: Home Automation | 330 |
| • | Exercise | 333 |
| 14. | IoT PROTOCOLS FOR EMBEDDED SYSTEMS | 335 |
| 14.1 | Introduction | 335 |
| 14.2 | Protocol Standardization for IoT | 340 |
| 14.3 | M2M and WSN Protocols | 354 |
| 14.4 | RFID Protocols | 356 |
| 14.5 | MODBUS Protocols | 356 |
| 14.6 | Zigbee Architecture | 358 |
| 14.7 | IP Based Protocols | 359 |
| 14.8 | Case Study: Smart Irrigation System | 360 |
| • | Exercise | 363 |
| 15. | CLOUD PLATFORMS FOR IoT BASED EMBEDDED SYSTEM | 365 |
| 15.1 | Software Defined Networking | 365 |
| 15.2 | Introduction to Cloud Storage Model | 366 |
| 15.3 | Communication API | 366 |
| 15.4 | Python Web Application Framework | 369 |
| 15.5 | IoT Based Embedded Systems | 379 |
| • | Appendix - A | 380 |
| • | Appendix - B | 381 |
| • | Appendix - C | 384 |
| • | Exercise | 386 |
| 16. | SECURITY IN IoT BASED EMBEDDED SYSTEM | 387 |
| 16.1 | IoT Security | 387 |
| 16.2 | Key Element of IoT Security | 391 |
| 16.3 | Case Study: Home Intrusion Detection | 394 |
| • | Exercise | 396 |

Introduction to Embedded Systems



OUTLINE

- 1.1 *Introduction to Embedded Systems*
 - 1.2 *Embedded System Design and Development*
 - 1.3 *Development Tools for an Embedded System*
 - 1.4 *System Design Specifications*
 - 1.5 *System Specification Versus System Requirements*
 - 1.6 *Partitioning and Decomposing a System*
 - 1.7 *Functional Design*
 - 1.8 *Architectural Design*
 - 1.9 *Functional Model Versus Architectural model*
 - 1.10 *Prototyping*
 - 1.11 *Other Consideration for the Design Process*
 - 1.12 *Archiving the Project*
-

1.1 INTRODUCTION TO EMBEDDED SYSTEMS

Background and History: The microprocessor is an integrated circuit chip fabricated from MOSFETs (Metal-Oxide-Semiconductor Field-Effect Transistors) and was developed in early 1960. We have LSI (Large Scale Integration) with hundreds of transistors on a single MOS chip by Late 1960. The application of MOS LSI chips was the basis for first microprocessor. The first single-chip microprocessor was the Intel 4004 using MOS technology in 1971. The first embedded system (calculators) using intel 4004 was developed by Busicom engineer of Japan. In 1980, memory, input and output system components had been integrated into processor chip. Now, processor chip with memory and input/output components is called microcontroller (μC) Microcontroller based embedded systems are cheaper of than that of general purpose computer. Today, low-cost μC are programmed to carry out the roles of various components. It is possible to replace consumer products, potentiometers variable capacitor by a microcontroller. In 2020, 29 billions of ARM microcontrollers and 1 billion PIC μC are manufactured. Other firms have also manufactured μC only 2% of μC are used for new PCs, Macs and Unix Workstations. 98% of manufacture μC are used for design and development of embedded systems.

Embedded system covers the range from 4-bit μC based toy and greeting cards to 128-bit μp and specialized DSPs and network processors. Some of the embedded systems run a short assembly program from ROM with no operating system. Many embedded systems run on RTOS and complex multithreaded C or C++ programs. Virtually every electronic device designed and manufactured today is an embedded system. An electronic glucose meter (embedded system) is developed using 8 bit parasonic MN101 microcontroller (Fig. 1).

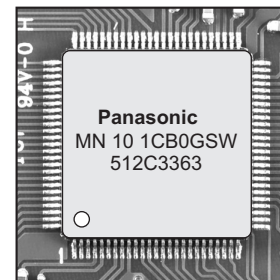


Fig. 1: 8 bit panasonic MN101 microcontroller

What is an Embedded System?

An embedded system is a microcontroller based system which is designed to perform a specific task. For example, fire alarm is an embedded system it senses only smoke. An embedded system has three components:

1. Hardware (μ C), sensors,
2. Application software,
3. Real Time Operating Systems (RTOS).

RTOS defines the way the system works, it sets rules to execute the application program.

Characteristics of Embedded System

1. **Single-Functioned:** An embedded system perform a specific task and does the same repeatedly. ATM always function as ATM.
2. **Tightly Constrained:** All system have constraints on design metrics, but embedded system is tightly constraints.

Design metrics is a measure of the features of implementation such as cost, size, Power and performance. An embedded system must be reliable, must be of a size to fit on a single chip, must work fast to process data in real time and consume minimum power to extend battery life. Example: Smart phone.

Reactive and Real Time

Embedded system must react to changes in the parameter of system and must compute required results in real time.

Examples: A car cruise controller continually checks and reacts to speed and brake sensors. It must compute acceleration or de-accelerations repeatedly within a limited time a delayed computations may result in failure to control the car.

- **Microcontroller Based:** It must be microcontroller based system.
- **Memory:** It must have memory as its software usually embeds in ROM. It does not need any secondary memory.
- **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 of Embedded Systems

- Easily customizable
- Low power consumption
- Low cost
- Enhanced performance
- Reliable

Disadvantages of Embedded Systems

- High Development cost
- Larger time to market
- Poor Heat Management in circuit.

Basic Structure of Embedded System

The block diagram of Basic structure of Embedded system is given in Fig. 2.

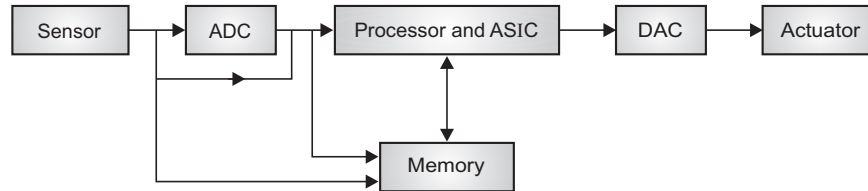


Fig. 2: Basic structure of embedded system

Following are the Categories of Processors

1. General Purpose Processor (GPP)
 - (a) Microprocessor
 - (b) Microcontroller
 - (c) Digital Signal Processor
 - (d) Media Processor
2. Application Specific System Processor (ASSP)
3. Application Specific Instruction Processor (ASIP)

Types of Embedded Systems

The various types of embedded systems are shown in Fig 1.3.

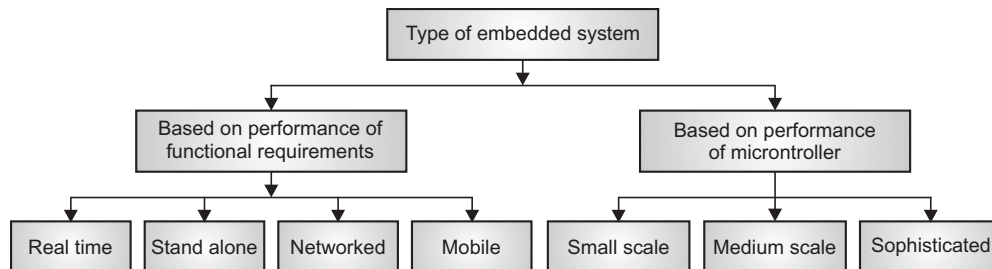


Fig. 3: Types of embedded systems

1. **Real Time Embedded System:** Such system gives the required output in a particular time. Real time embedded systems are of two types Soft and Hard real time systems.
2. **Stand Alone Embedded System:** Stand Alone system does not require computer.
Example MP3 player, digital camera, microwave Ovens, and temperature measurement system.
3. **Network Embedded Systems:** This type of embedded system is connected to LAN, WAN or internet. The connection can be wired or wireless. This type of embedded system is the fastest growing area in embedded system applications. The embedded web serve is a system where all embedded devices are connected to a web server and controlled by web browser.
Example: Home security system where all sensors are connected and run on the TCP/IP protocol.
4. **Mobile Embedded System:** Mobile smart phones, cell phones are all mobile Embedded System.
5. **Small Scale Embedded System:** This type of system uses 8-16 bit microcontrollers. For developing embedded software for small scale embedded systems, we require editor, assembler cross-assembler and IDE.

6. **Medium Scale Embedded System:** There types of systems use 32 bit microcontroller, RISCs or DSPs. Main programming tools for such systems are C, C++, Java, Visual C++, RTOS, debugger, source code engineering tool, simulator and IDE.
7. **Sophisticated Embedded System:** There types of systems have enormous hardware and software complexities. They need ASIPs, IPs, PLAs, Scalable processor. They are used for cutting edge applications.
8. **Applications of Embedded System:** Embedded systems are used in different applications like automobiles, telecommunications, smart cards, missiles, satellites, computer networking and digital consumer electronics.

1.1.1 Architecture of Embedded System

Fig. 4 shows the basic architecture of an Embedded system. The architecture is divided into two parts: embedded hardware and embedded software.

The embedded hardware includes processor, memory bus, peripheral device, I/O ports and various controllers. Embedded software usually contain OS and application software. In embedded system, the hardware and software work together with various input signals and output the required results.

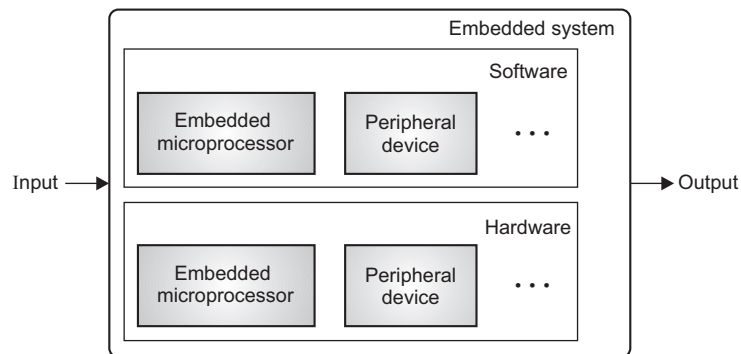


Fig. 4: Block diagram of basic architecture of embedded system

Embedded Hardware Architecture

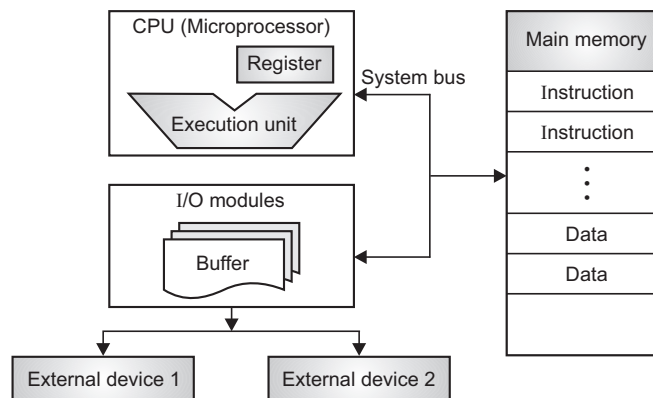


Fig. 5: Block diagram of embedded hardware architecture

The basic embedded hardware components such as microprocessor, memory, input and output modules are interconnected by a system bus in order to communicate and execute a program. The block diagram of embedded Hardware Architecture is shown in Fig. 5.

We have basically two types of embedded hardware Architecture.

1. Von Neumann Architecture
2. Hardware Architecture

The block diagram of Von Neumann Architecture is shown in Fig. 6. In this architecture software and data use the same memory bus. In this architecture, the transmission of information becomes the bottleneck of performance and affects the speed of processing, so this is called Von Neumann bottleneck. In reality cache memory solves this problem.

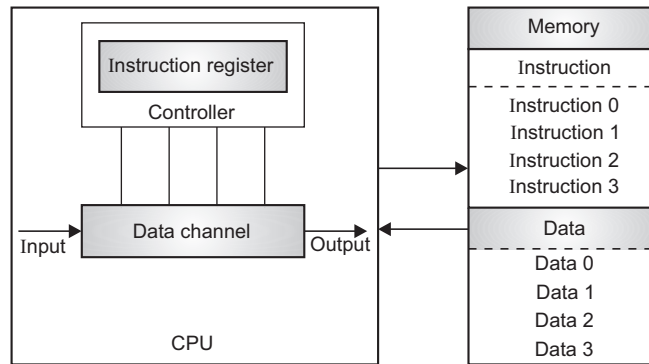


Fig. 6: Block diagram of Von Neumann architecture

The block diagram of Harvard Architecture is shown in Fig. 7. Harvard Architecture has two important features. First, instructions and data are stored in two separate memory modules. Second, two independent buses for program and data are used for communication between CPU and memory.

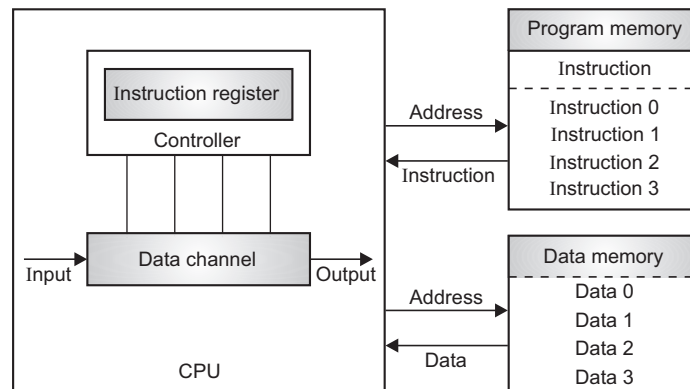


Fig. 7: Block diagram of harvard architecture

Hardware Architecture has greater data-memory bandwidth, it is used for DSP based embedded system. However, the Von Neumann Architecture is simple hardware design and flexible data and program storage. Therefore it is used for most embedded system.

1.1.2 Design Methodology of an Embedded System

Designing and developing embedded systems, is challenging and requires a large number of decisions. Some decisions require knowledge about problem, other decisions require knowledge of tools, techniques and technologies available. The collection of things right from requirement to application is called product life cycle.

The objectives of design methodology is given as:

- Find out what the customers want.
- Think the way to give them what they want.

- Prove the design by building and testing.
- Build a lot of product to prove that it was not an accident.
- Use the product to solve customer's problem.

An embedded system involves the design of Hardware and software.

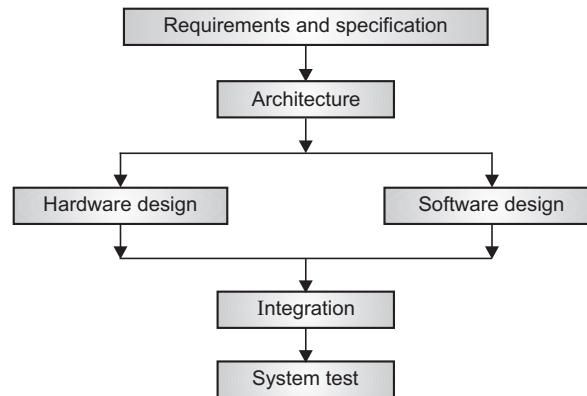


Fig. 8: Block diagram of design methodology of an embedded system

We design a circuit board of Hardware components and writing the code to perform the specific task. Fig. 8, shows the block diagram of design methodology of an embedded system. It is observed from Fig. 8, that front activities such as specification, requirements and architecture are considered for hardware and software Designs. Back-end integration and testing are carried out for entire embedded system. The development of hardware and software component of hardware and software components are carried out independently. The block diagram of the hierarchical design flow for an embedded system is shown in Fig. 9.

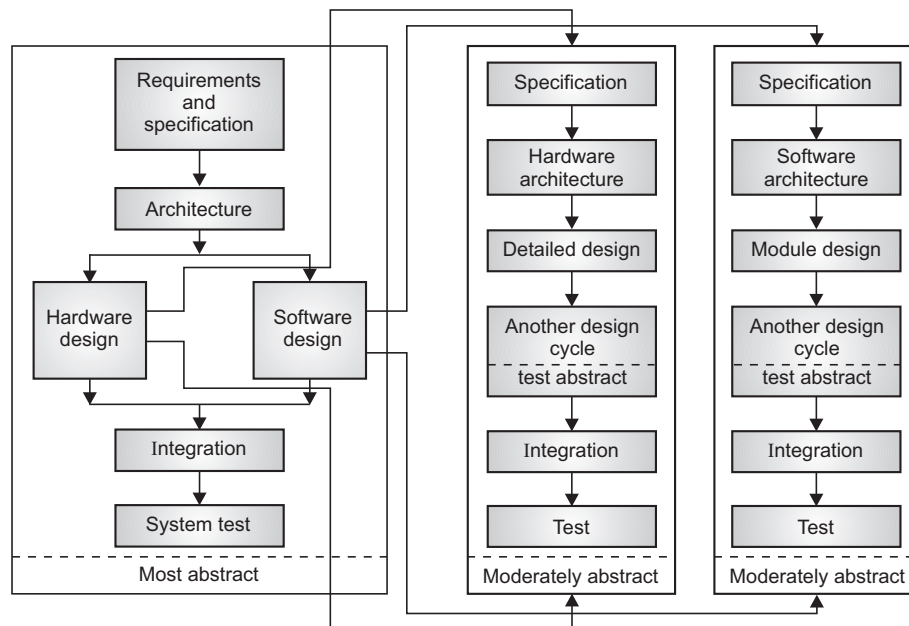


Fig. 9: Block diagram of hierarchical design flow

In Fig. 9, shows the design flow of complex embedded system. In complex embedded system, each flow is handled by separate teams. The teams are dependent on each other results.

EMBEDDED SYSTEMS

- Covers the basics of embedded systems, including microcontrollers, real-time operating systems (RTOS), and system architecture, and explains how hardware and software interact in embedded applications.
- Focuses on embedded programming languages like C, C++ and Python or Assembly language. It also introduces debugging techniques, firmware development and low-level hardware interactions.
- Explores RTOS concepts, task scheduling, inter-process communication and power management for real-time applications and covers systems like Free RTOS and Embedded Linux.
- Includes case studies on embedded applications in automotive systems, IoT, robotics, medical devices and industrial automation.

Your favourite book brand is now on
Play Store !

Buy our e-Books from

PRAGATI BOOKS APP BOOKSTORE AND E-READER

Scan the
QR code
to
download
our app



Note : Do not photocopy the book; it is a punishable offence.

Price : ₹ 600/-

Scan
QR code
to buy
books



Pragati
ONLINE.COM
Digital Bookstore & E-Book Reader

Also available at all bookstores in India.

amazon
Flipkart



NIRALI PRAKASHAN



www.niralibooks.com



niralipune@pragationline.com



(+91-020) 25512336/ 7/ 9



www.facebook.com/niralibooks



@nirali.prakashan



www.pragationline.com

