EMBEDDED SYSTEM BASICS AND APPLICATION

TOPICS TO BE DISCUSSED

- System
- Embedded System
- Components
- Classifications
- Processors
- Other Hardware
- Software
- Applications

INTRODUCTION

What is a system?

A system is a way of working, organizing or doing one or many tasks according to a fixed plan, program or set of rules.

A system is also an arrangement in which all its units assemble and work together according to the plan or program.

SYSTEM EXAMPLES

WATCH is a time display SYSTEM

Its Parts: Hardware, Needles, Battery, Dial, Chassis and Strap

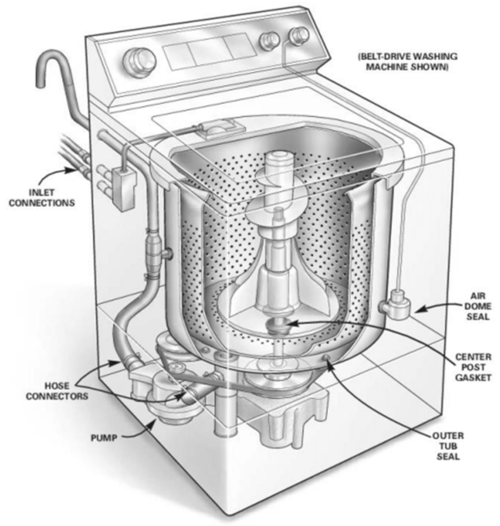
<u>Rules</u>

1. All needles move clockwise only
2. A thin needle rotates every second
3. A long needle rotates every minute
4. A short needle rotates every hour
5. All needles return to the original position after 12 hours

SYSTEM EXAMPLES

WASHING MACHINE

It is an automatic clothe washing SYSTEM
Parts: Status display panel, Switches & Dials, Motor, Power supply & control unit, Inner water level sensor and solenoid valve.



<u>Rules</u>

1.Wash by spinning
2.Rinse
3.Drying
4.Wash over by blinking
5.Each step display the process stage
6.In case interruption, execute only the remaining

Embedded System Characteristics

- Perform a single function
- Form part of a larger system
- Not intended to be independently programmable by the user
- Are expected to work with minimal or no human interaction
- Reactive, real-time operation
- Tightly constrained

APPLICATIONS

- Household appliances:
  - Microwave ovens, Television, DVD Players & Recorders
- Audio players
- Integrated systems in aircrafts and missiles
- Cellular telephones
- Electric and Electronic Motor controllers
- Engine controllers in automobiles
- Calculators
- Medical equipments
- Videogames
- Digital musical instruments, etc.

TELEVISION

REMOT CONTROL

REFRIGERATORS

ELEVATORS

VIDEO GAMES

SET-TOP BOX

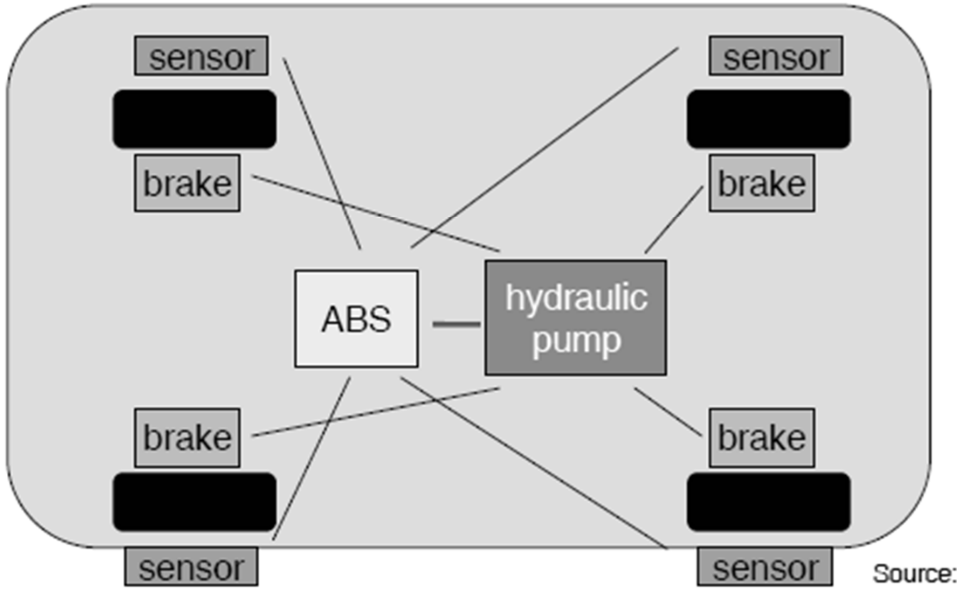
PLANES

CARS

BMW 850i brake and stability control system

- Anti-lock brake system (ABS): pumps brakes to reduce skidding.
- Automatic stability control (ASC+T): controls engine to improve stability.
- ABS and ASC+T communicate.
  - ABS was introduced first---needed to interface to existing ABS module

BMW 850i, cont'd.

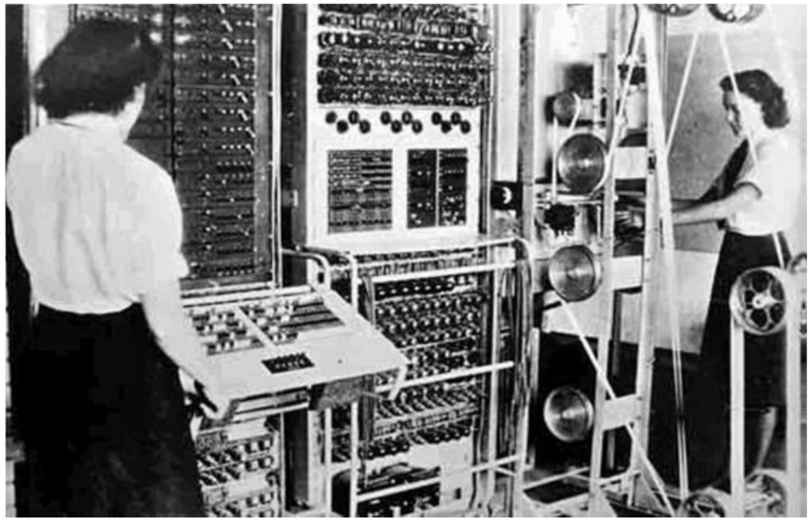


Source: Wolf 2000,
© 2000 Morgan Kaufmar

- **Automotive embedded**
- **systems**
- Today’s high-end automobile may have 100
- microprocessors:
- 4-bit microcontroller checks seat belt;
- microcontrollers run dashboard devices;
- 16/32-bit microprocessor controls engine.
- Source:

- **Embedded Systems...**
- react on the environment at the speed of the environment
- often real-time requirements
- designed for one single task
- have often to be power-efficient
- mass products and have to be cheap
- must be reliable

History of Embedded Systems


Control Panel of Colossus Mark II Computer

Early Computers

- Single functioned
- Not user programmable
- Large and power thirsty
- Not integrated

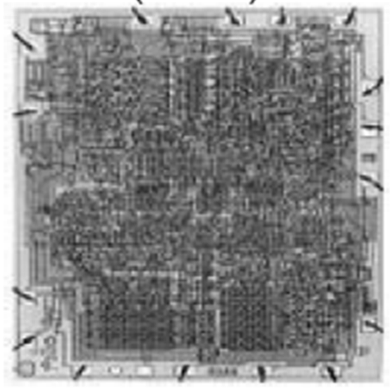
First Embedded System

- The Apollo Guidance Computer (AGC)
- Guidance & Navigation
- CPU + MEM + I/O
- Low-power mode
- AGC Assembly Programmed

History of Embedded Systems

Development in Electronics

Intel 4004
(1971)



108 KHz
2,300 transistors

1.5 GHz
42 million transitors

Intel Pentium 4
(2000)

If automobile speed had increased similarly over the same period, we could now drive from San Francisco to New York in about 13 seconds (Intel).

History of Embedded Systems

Plenty of Vendors

- TI (MSP430)
- Microchip (PIC)
- Intel (8051, 80x86)
- Freescale (HC11, HC08)
- ARM Limited (ARM7)
- Atmel (ATmega)

Plenty of Sizes

- 4-, 8-, 16-, 32-, and 64-bit
- CISC Vs. RISC
- Harvard Vs. vonNeumann
- **Wide Market**
- Over 6 Billion chips per year
- Over $50 billion sales

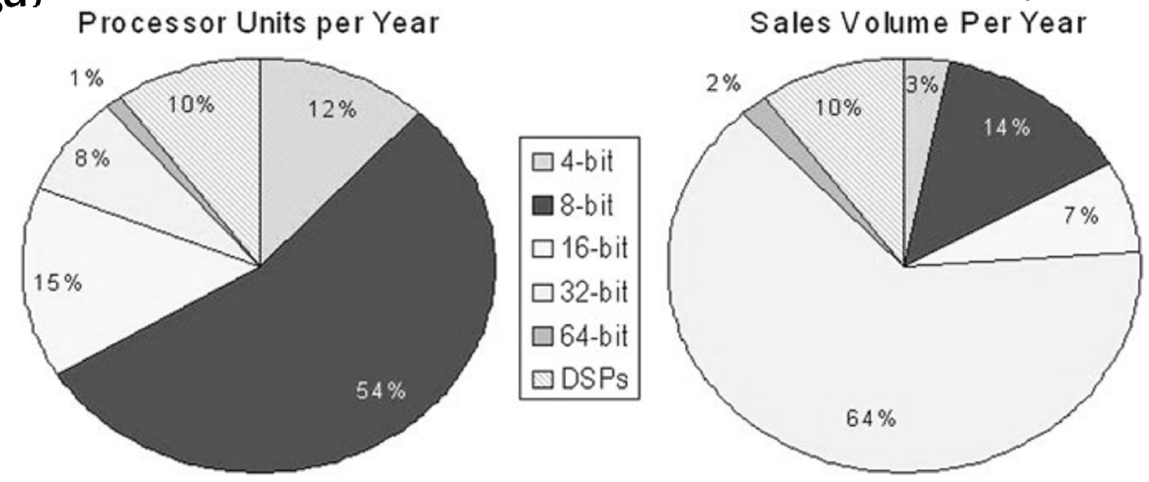


Fig. 1.5 Estimates of processor market distribution (*Source* Embedded systems design—www.embedded.com)

Contemporary Embedded Systems

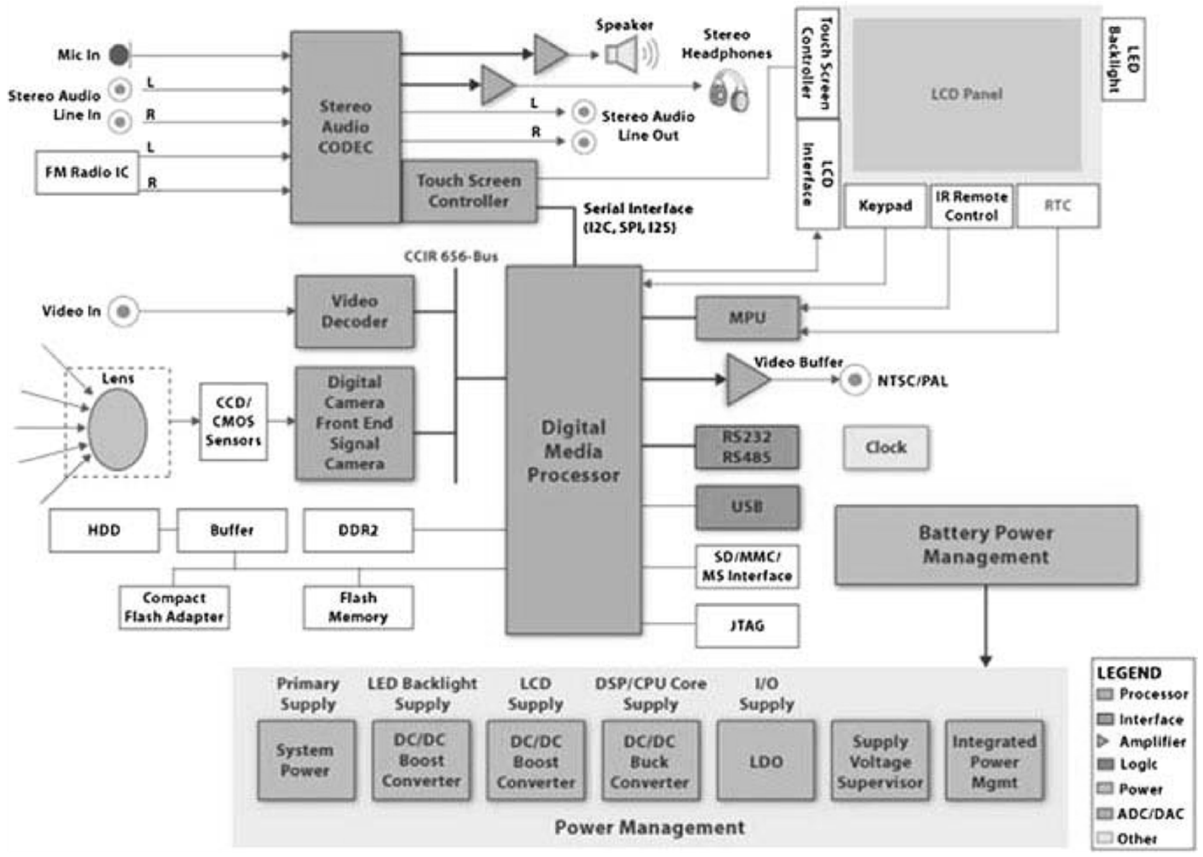


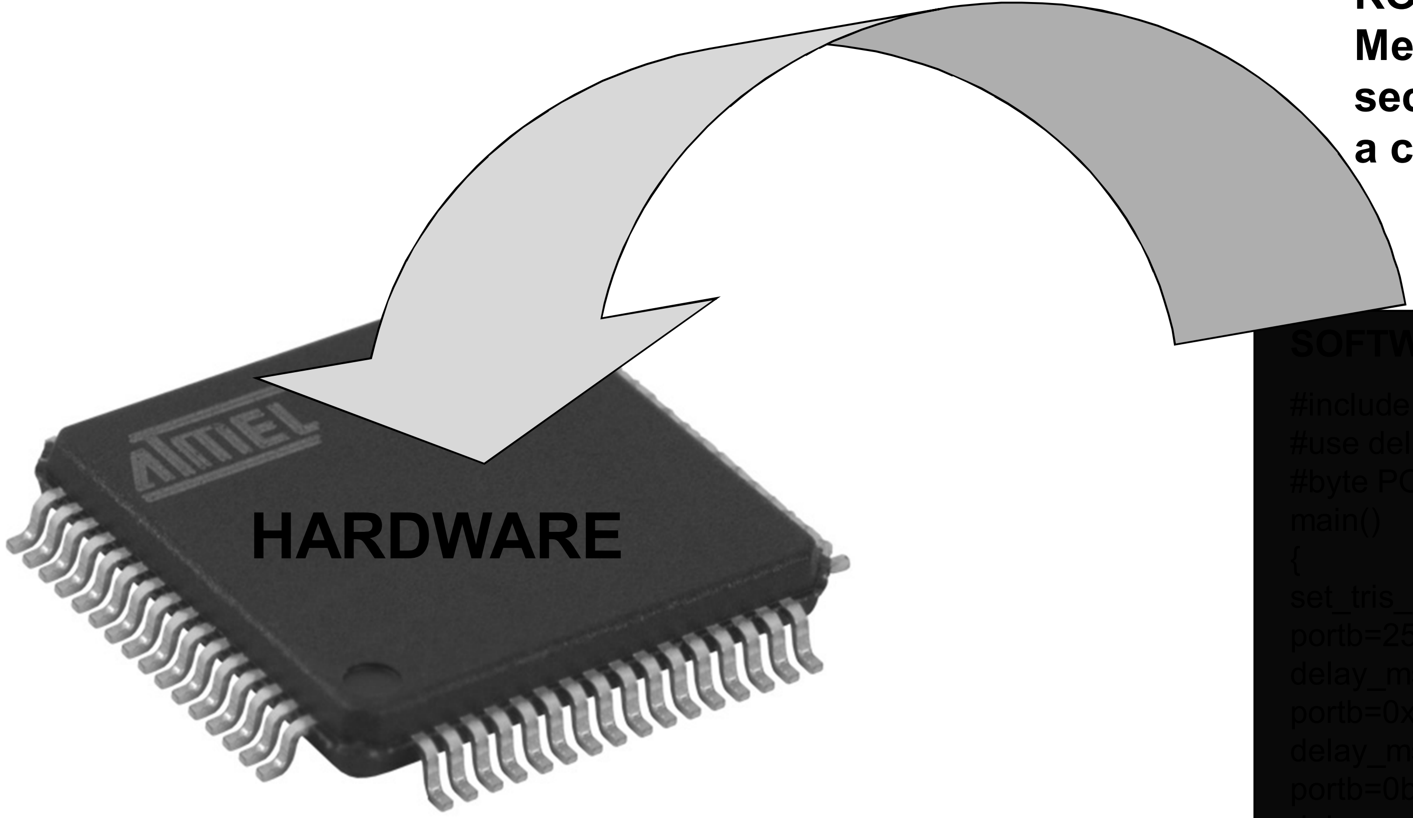
Fig. 1.6 Generic multi-function media player (*Courtesy of Texas Instruments, Inc.*)

- System Components
- Power management
- Video processing
- Audio processing
- Communications
- User interfaces
- Dedicated ASICs
- Memory management
- Storage

EMBEDDED SYSTEM

Definition: An Embedded System is a system that has electronic <u>hardware with software tightly coupled</u> together.

Its software embeds in ROM (Read Only Memory). It does not need secondary memories as in a computer



SOFTWARE PROGRAM

```
#include <16f876a.h>
#use delay (clock=20000000)
#byte PORTB=6
main()
{
set_tris_b(0);
portb=255;        //decimal
delay_ms(1000);
portb=0x55;        //hexadecimal
delay_ms(1000);
portb=0b10101010;  //binary
delay_ms(500);
}
```

COMPONENTS OF EMBEDDED SYSTEM

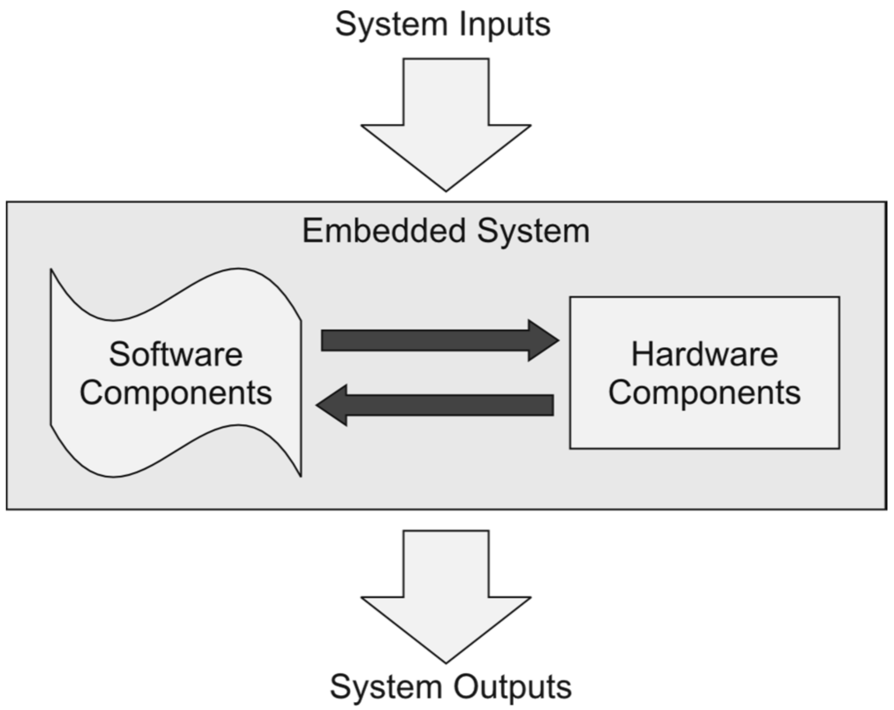


Fig. 1.7 General view of an embedded system

COMPONENTS OF EMBEDDED SYSTEM

- **Hardware**

  Processor, Timers, Interrupt controller, I/O Subsystem, Memories, Ports, etc.

- **Software**

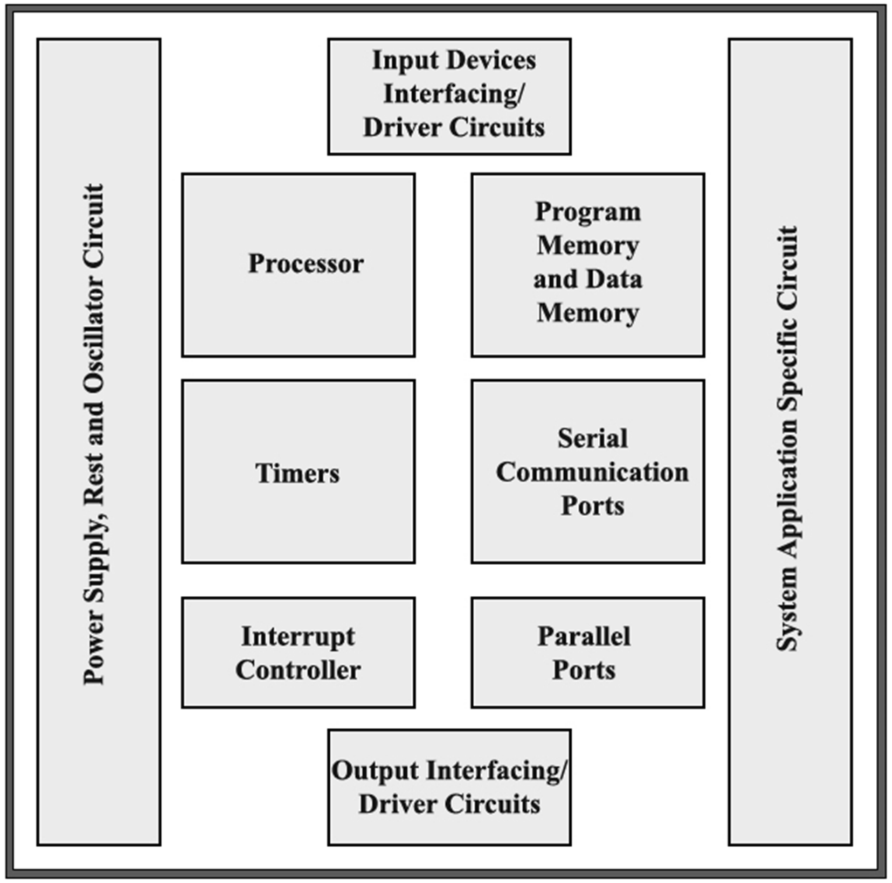
  - **Application Software**

    Which may perform concurrently the series of tasks or multiple tasks

  - **Kernel or Real Time Operating System (RTOS)**

    RTOS defines the way the system work. Which supervise the application software. It sets the rules during the execution of the application program. A small scale embedded system may not need an RTOS.

EMBEDDED SYSTEM HARDWARE


Power Supply, Rest and Oscillator Circuit
Input Devices Interfacing/ Driver Circuits
Processor
Program Memory and Data Memory
Timers
Serial Communication Ports
Interrupt Controller
Parallel Ports
Output Interfacing/ Driver Circuits
System Application Specific Circuit

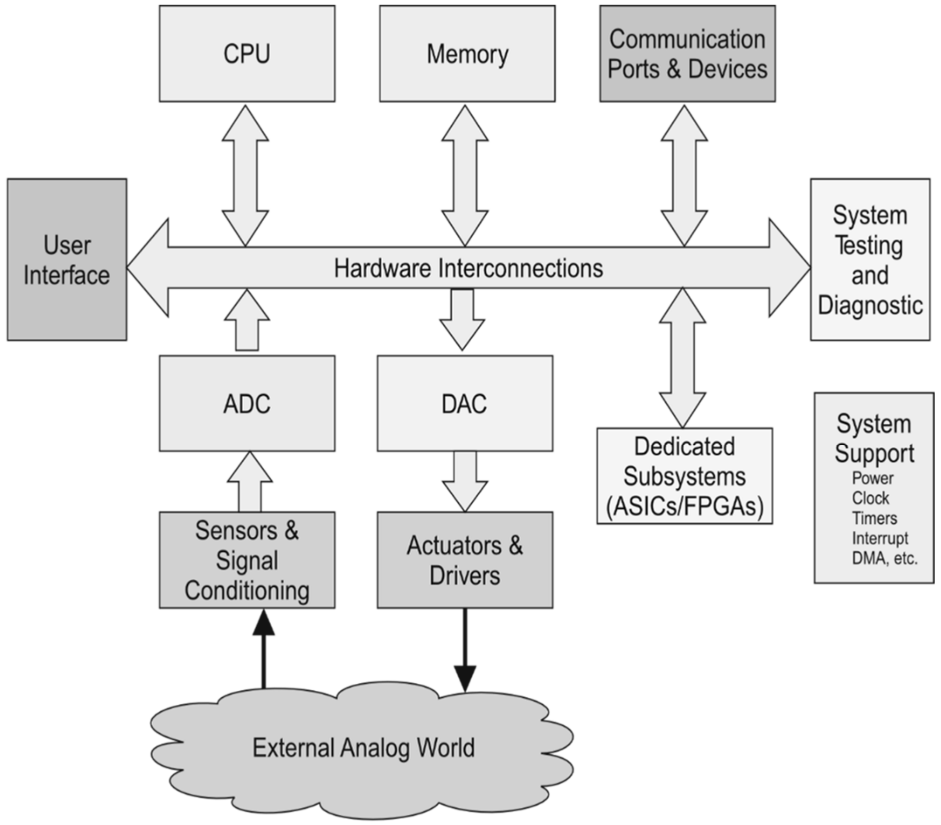

CPU
Memory
Communication Ports & Devices
User Interface
Hardware Interconnections
System Testing and Diagnostic
ADC
DAC
Dedicated Subsystems (ASICs/FPGAs)
System Support
Power
Clock
Timers
Interrupt
DMA, etc.
Sensors & Signal Conditioning
Actuators & Drivers
External Analog World

Fig. 1.8 Hardware elements in an embedded system

EMBEDDED SYSTEM HARDWARE

Central Processing Unit

▪Registers, ALU, CU

Memory

▪Program Memory

▪Data Memory

I/O Devices

▪Communication ports

▪User Interfaces

▪Sensors & actuators

▪Diagnostics support

▪System controllers

▪Power management

▪Specialized ASICs

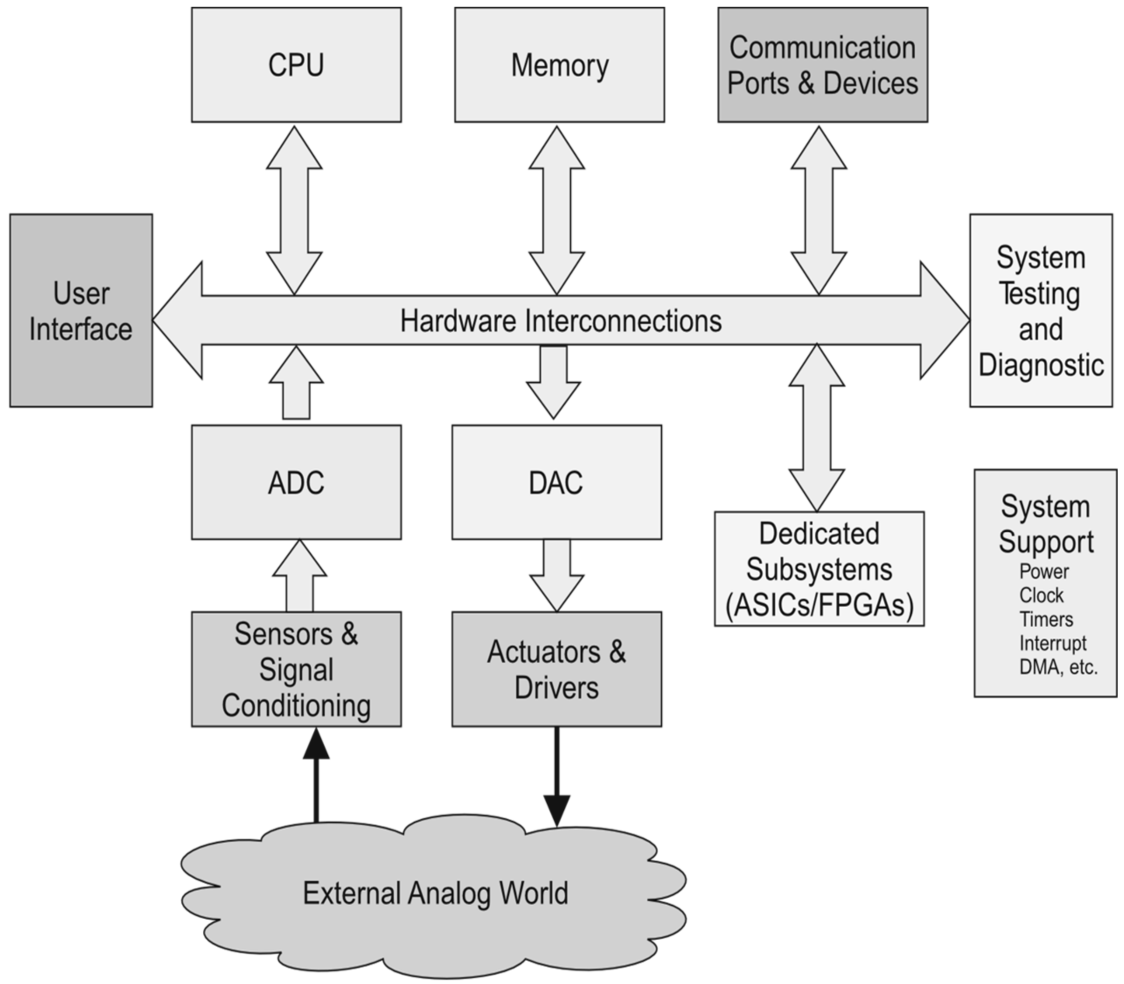

CPU
Memory
Communication Ports & Devices
User Interface
Hardware Interconnections
System Testing and Diagnostic
ADC
DAC
Dedicated Subsystems (ASICs/FPGAs)
System Support
Power
Clock
Timers
Interrupt
DMA, etc.
Sensors & Signal Conditioning
Actuators & Drivers
External Analog World

Fig. 1.8 Hardware elements in an embedded system

OTHER HARDWARE

- Clock Oscillator
- Real Time Clock (RTC)
- Reset Circuit, Power-up Reset and watchdog timer Reset
- I/O Ports, I/O Buses
- Interrupt Handler
- DAC and ADC
- LCD and LED Display
- Keypad/Keyboard

EMBEDDED SYSTEM SOFTWARE

System Tasks

- Actions making use of system resources

System Kernel

- Manages system resources
- Coordinates task services

Services

- Routines performing specific tasks

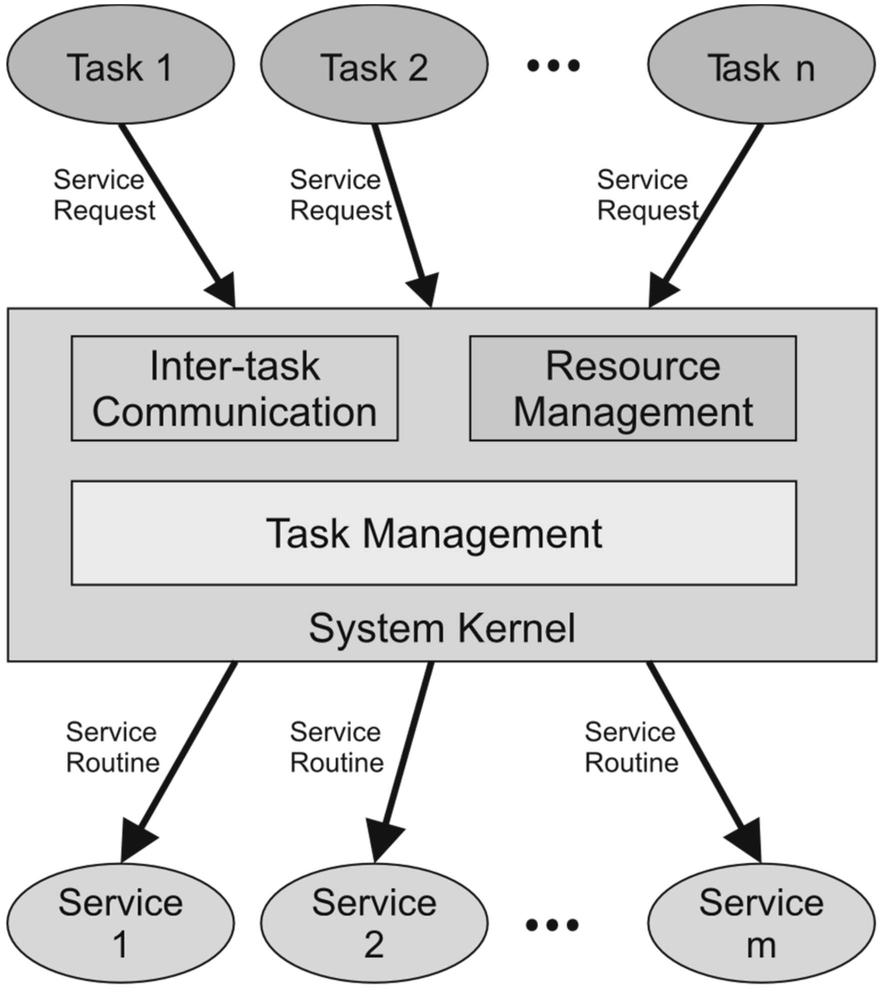


Fig. 1.9 Software structure in an embedded system

SOFTWARE

SOFTWARE

C
C++
Dot Net

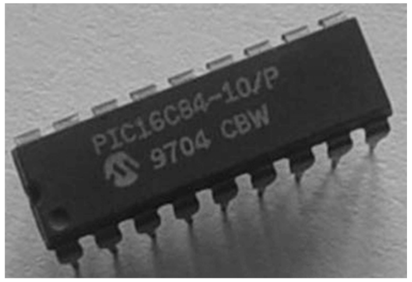
SIMULATOR

Masm

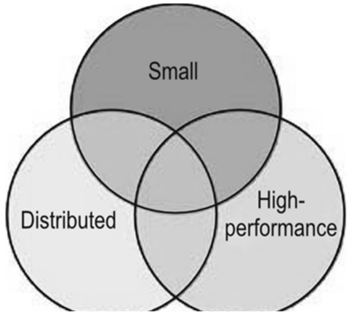
COMPILER

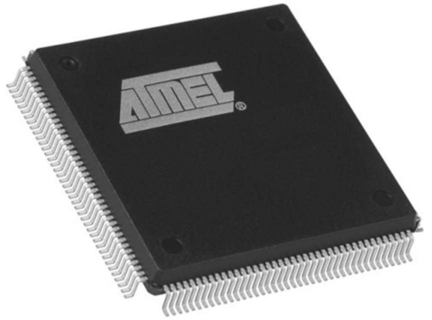
RIDE
KEIL

CLASSIFICATIONS OF EMBEDDED SYSTEM

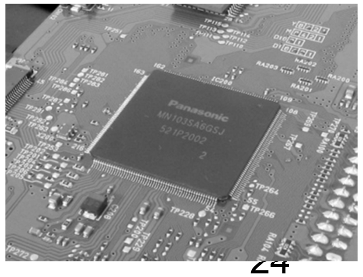


1. **Small Scale Embedded System**



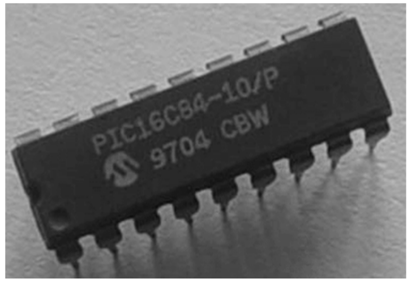


2. **Distributed Embedded System**



3. **High PerformanceEmbedded System**

SMALL SCALE EMBEDDED SYSTEM



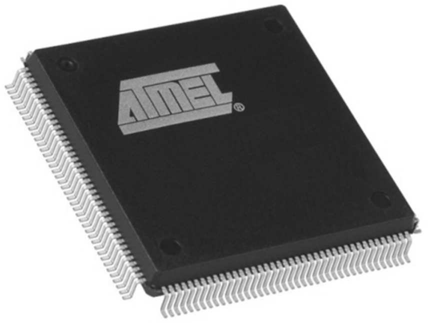
- Single 8 bit or 16bit Microcontroller.
- Low hardware and software complexity.
- They may even be battery operated.
- Usually “C” is used for developing these system.
- The need to limit power dissipation when system is running continuously.
- Single Tasked
- Low-cost maintenance free

Programming tools:

Editor, Assembler and Cross Assembler

DISTRIBUTED EMBEDDED SYSTEM

- **Single or few 16 or 32 bit microcontrollers or Digital Signal Processors (DSP) or Reduced Instructions Set Computers (RISC).**
- **May be multichip, board level distributed**
- **Multi-tasked**
- **Maintainable, upgradeable**
- **Both hardware and software complexity.**

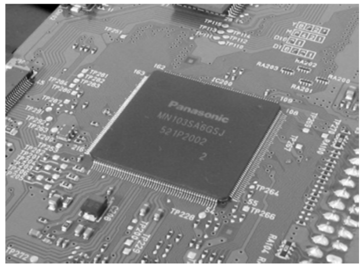


Programming tools:

RTOS, Source code Engineering Tool, Simulator, Debugger and Integrated Development Environment (IDE).

HIGH PERFORMANCE EMBEDDED SYSTEM

- **Enormous hardware and software complexity**
- **Low Volume**
- **High Cost**
- **High Maintenance**
- **Which may need scalable processor or configurable processor and programming logic arrays.**
- **Constrained by the processing speed available in their hardware units.**



Programming Tools:

For these systems may not be readily available at a reasonable cost or may not be available at all. A compiler or retargetable compiler might have to be developed for this.

The Life Cycle of Embedded Designs

- Embedded Design Goal
  - Design must successfully complete all pertinent stages
  - Not all designs go through all stages
- Plan for Each Stage
  - Designer's vision and planning needed for success
  - Good designs do not happen by chance

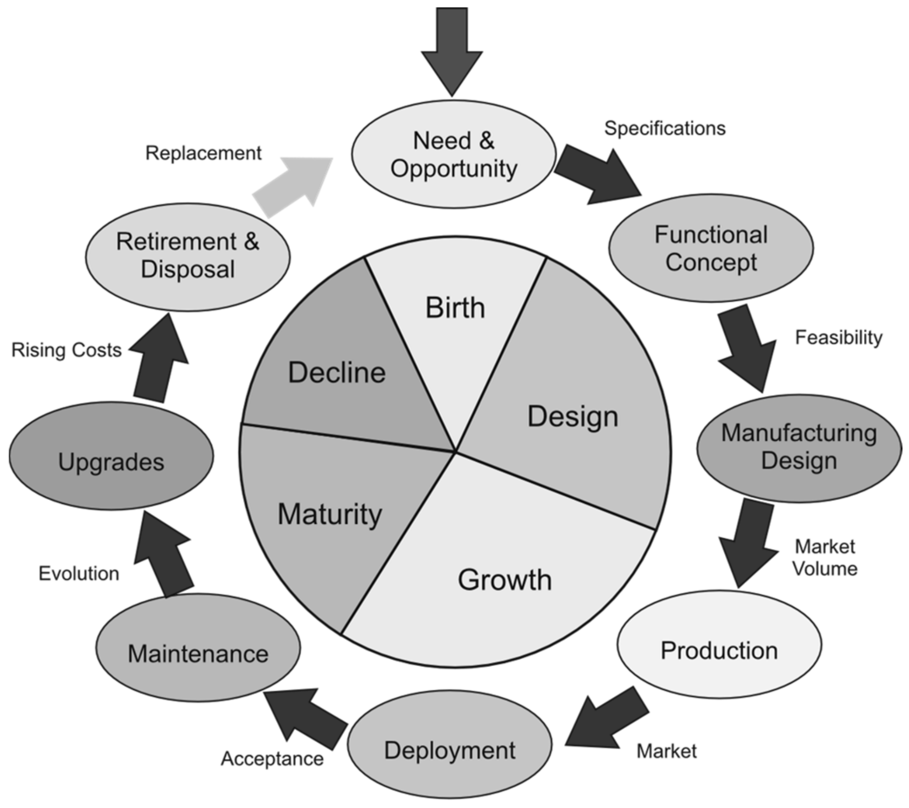


Fig. 1.11 Life cycle of an embedded system

Design Constraints

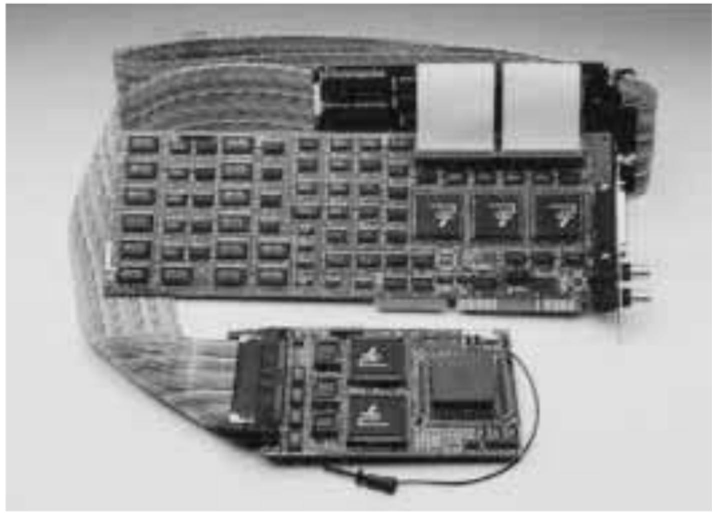
- Functionality
  - System ability to perform the function it was designed for (REQ)
- Cost
  - Amount of resources needed to conceive, design, and produce an embedded system
- Performance
  - System ability to perform its function in time.
  - Affected by both HW & SW factors
- Size
  - Physical space taken by a system solution.
- Power and Energy
  - Energy required by a system to perform its function.
- Time to Market
  - The time it takes from system conception to deployment.
- Maintainability
  - System ability to be kept functional during its mature life.

Functionality

- Functional verification is a difficult task
  - Can consume up to 70% of development time
- Verification Methods
  - Simulation Techniques
    - Behavioral (HDL-based)
    - Logic (Circuit Modeling)
    - Processor (Software)
- JTAG Debugger
  - Hardware supported through dedicated ports
  - Used also for testing (boundary scan test)
  - Cost effective

Functionality

- In-Circuit Emulators
  - Replace MCU in target system
  - A powerful debugger
  - Expensive
- ROM Monitors
  - Monitor functions in ROM
  - Status sent via serial port



System Cost

- The cost of a given Volume (V) of units:

$$C_T = NRE + (RP \cdot V) \therefore$$

$$U_C = \frac{C_T}{V} = \frac{NRE}{V} + RP$$

- NRE = Non-Recurrent Engineering costs (Fixed)
  - Investment to complete all design aspects
  - Very large and independent of volume in CT
  - Include man-hours, infrastructure, and R&D
- RP = Recurrent Production costs (Variable)
  - Expenses in producing each unit of a given volume
  - Small but affected by V in CT
  - Include components, boards, packages, and testing

System Cost

- Commercial off-the-shelf parts-based design
  - Traditional methodology for Embedded Systems
  - Minimizes Hardware costs
  - Increases design & verification costs
- NREs in UC are diluted by a large production volume
- Balance between technology choice and production volume

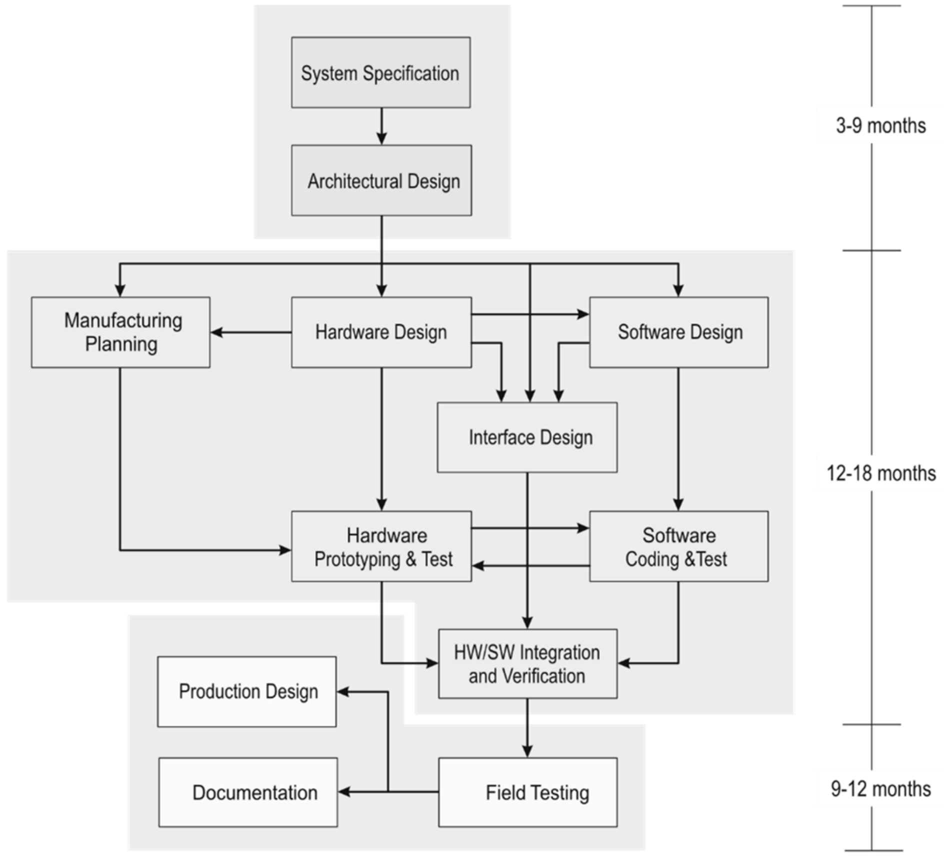


Fig. 1.12 Embedded systems design flow model based on COTS parts [8]

Performance HW

- Clock Frequency
  - System clock speed: not an absolute performance metric
- Architecture
  - Determines how clock cycles are used
- Component Speed
  - Response time and access time
- Handshaking
  - Signalization required to complete a transaction
- Low-power Modes
  - Wake-up times might affect application speed
- High speed is expensive!!!
  - Use it wisely

Performance SW

- Algorithm Complexity
  - Steps and resources needed to complete a task
- Task Scheduling
  - Affects waiting time in multitasking system
- Inter-task Communication
  - Time taken by tasks to exchange information
- Level of Parallelism
  - Software usage of system hardware resources

Power and Energy

- Critical Parameter
  - A long chain of design events depend on it
- System reliability
  - Stress, noise, and heat
- Cooling Costs
  - High power = lot of heat to remove
- Power Supply Requirements
  - Larger batteries of power supply
- Size, Weight, and Form
  - Mechanical system parameters affected by heat density

Power and Energy

- Environmental Impact of Embedded Systems
  - Average individual uses 60 microprocessors per day
  - Household electronics accounts for 11% of all energy consumed in the USA
    - 147,000,000,000 KWh (147TWh) per year
  - Excludes digital TVs and large appliances
  - Excludes industry, schools, hospitals, etc
  - Trend continues to grow… Is there a limit?

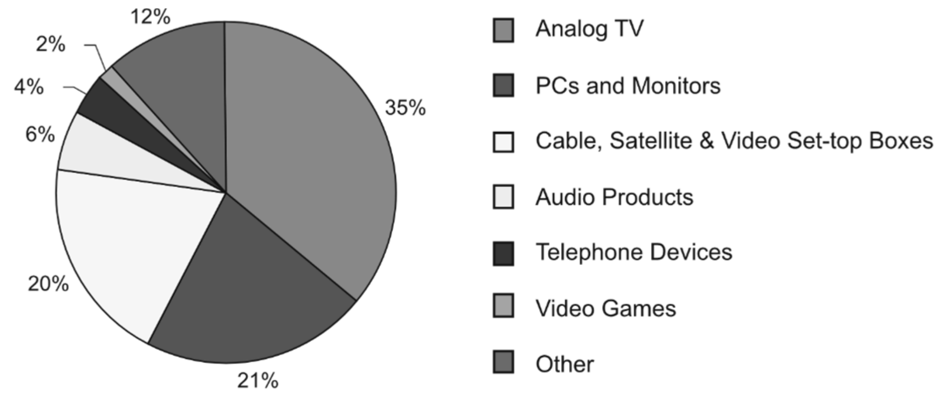


Fig. 1.13 Distribution of U.S. residential consumer electronics (CE) energy consumption in 2006. The total CE energy consumed was 147 TWh (*Source* Consumer Electronics Association)

Tips on power

- Use low-power MCUs and Peripherals
  - Activate CPU standby and sleep modes
  - Let peripherals do the work while the CPU is off
- Stop the Energy Waste
  - Turn off unused peripherals
- Write power efficient code
  - Every wasted CPU cycle is energy that will never come back
- Use power management techniques
  - Power and clock gating plus efficient coding techniques

Maintainability

Maintenance enables reliable system operation throughout entire useful life

- Relevance of maintenance depends on application
  - Expected lifespan
  - Application criticality
- Maintainability is a design requirement
  - Must be included among system specifications
- Must consider both aspects:
  - Hardware Maintenance
  - Software Maintenance

Four maintenance dimensions

- Corrective: Fixes faults
- Adaptive: Copes with a changing environment
- Perfective: Adds enhancements
- Preventive: Anticipates events

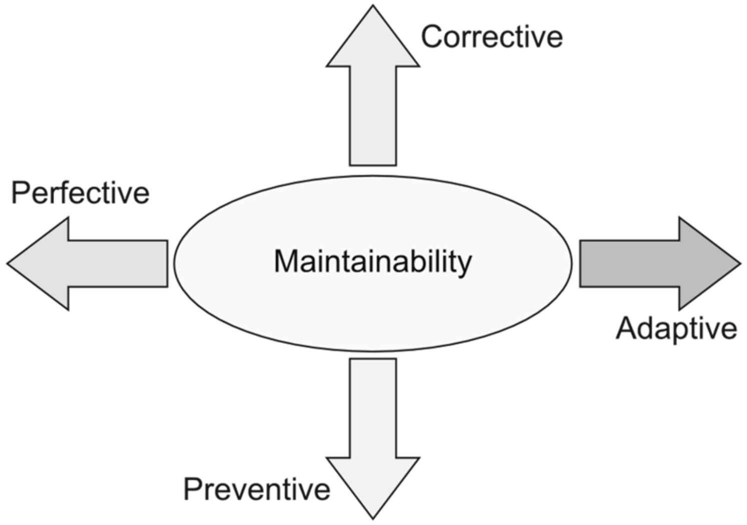


Fig. 1.16 The four actions supporting system maintainability

Hardware Maintenance Issue

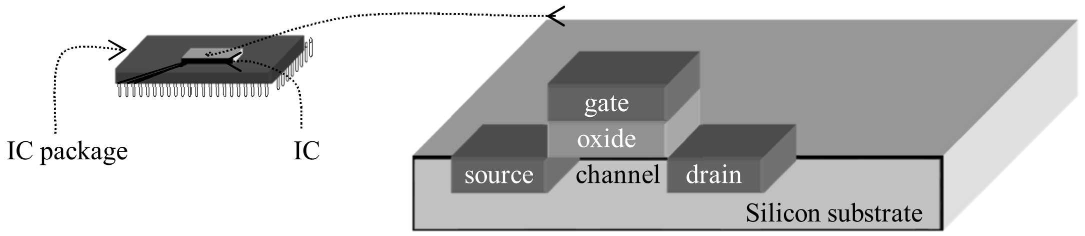
- Increased NREs
  - Design overhead to support HW maintenance
- Time-to-market Impact
  - Additional development time
- Increases Recurrent Cost
  - More components in system
- Component Obsolescence
  - Limit system useful life span

Software Maintenance Issue

- Hardware Constraints
  - Stringent HW constraints leave little room for support functions
- Cost of Verification
  - Undiscovered software bugs become maintenance headaches
- Inadequate Code Documentation
  - Meaningful and up-to-date
- Technology Changes
  - Compatibility with tool newer versions
- Ripple Effect of Changes
  - Identifying effect down the code
- Qualified Personnel
  - Everybody wants to design

IC technology

- The manner in which a digital (gate-level) implementation is mapped onto an IC
  - IC: Integrated circuit, or “chip”
  - IC technologies differ in their customization to a design
  - IC’s consist of numerous layers (perhaps 10 or more)
    - IC technologies differ with respect to who builds each layer and when



IC technology

- Three types of IC technologies
  - Full-custom/VLSI
  - Semi-custom ASIC (gate array and standard cell)
  - PLD (Programmable Logic Device)

Full-custom/VLSI

- All layers are optimized for an embedded system's particular digital implementation
  - Placing transistors
  - Sizing transistors
  - Routing wires
- Benefits
  - Excellent performance, small size, low power
- Drawbacks
  - High NRE cost (e.g., $300k), long time-to-market

Semi-custom

- Lower layers are fully or partially built
  - Designers are left with routing of wires and maybe placing some blocks
- Benefits
  - Good performance, good size, less NRE cost than a full-custom implementation (perhaps \$10k to \$100k)
- Drawbacks
  - Still require weeks to months to develop

PLD (Programmable Logic Device)

- All layers already exist
  - Designers can purchase an IC
  - Connections on the IC are either created or destroyed to implement desired functionality
  - Field-Programmable Gate Array (FPGA) very popular
- Benefits
  - Low NRE costs, almost instant IC availability
- Drawbacks
  - Bigger, expensive (perhaps $30 per unit), power hungry, slower

MICROCONTROLLER

- A **microcontroller** is a functional computer system-on-a-chip. It contains a processor, memory, and programmable input/output peripherals.

- Microcontrollers include an integrated CPU, memory (a small amount of RAM, program memory, or both) and peripherals capable of input and output.

VARIOUS MICROCONTROLLERS

INTEL

8031,8032,8051,8052,8751,8752

PIC

8-bit PIC16, PIC18,

16-bit DSPIC33 / PIC24,

PIC16C7x

Motorola

MC68HC11

MICROPROCESSOR Vs MICROCONTROLLER

MICROPROCESSOR	MICROCONTROLLER
The functional blocks are ALU, registers, timing & control units	It includes functional blocks of microprocessors & in addition has timer, parallel i/o, RAM, EPROM, ADC & DAC
Bit handling instruction is less, One or two type only	Many type of bit handling instruction
Rapid movements of code and data between external memory & MP	Rapid movements of code and data within MC
It is used for designing general purpose digital computers system	They are used for designing application specific dedicated systems

EMBEDDED PROCESSOR

- **Special microprocessors & microcontrollers often called, Embedded processors.**
- **An embedded processor is used when fast processing fast context-switching & atomic ALU operations are needed.**

Examples : ARM 7, INTEL i960, AMD 29050.

DIGITAL SIGNAL PROCESSOR

- DSP as a GPP is a single chip VLSI unit.
- It includes the computational capabilities of microprocessor and multiply & accumulate units (MAC).
- DSP has large number of applications such as image processing, audio, video & telecommunication processing systems.
- It is used when signal processing functions are to be processed fast.

Examples : TMS320Cxx, SHARC, Motorola 5600xx

APPLICATION SPECIFIC SYSTEM PROCESSOR (ASSP)

- **ASSP is dedicated to specific tasks and provides a faster solution.**
- **An ASSP is used as an additional processing unit for running the application in place of using embedded software.**

Examples : IIM7100, W3100A

MULTI PROCESSOR SYSTEM USING GPPs

- Multiple processors are used when a single processor does not meet the needs of different task.
- The operations of all the processors are synchronized to obtain an optimum performance.

Moore’s Law

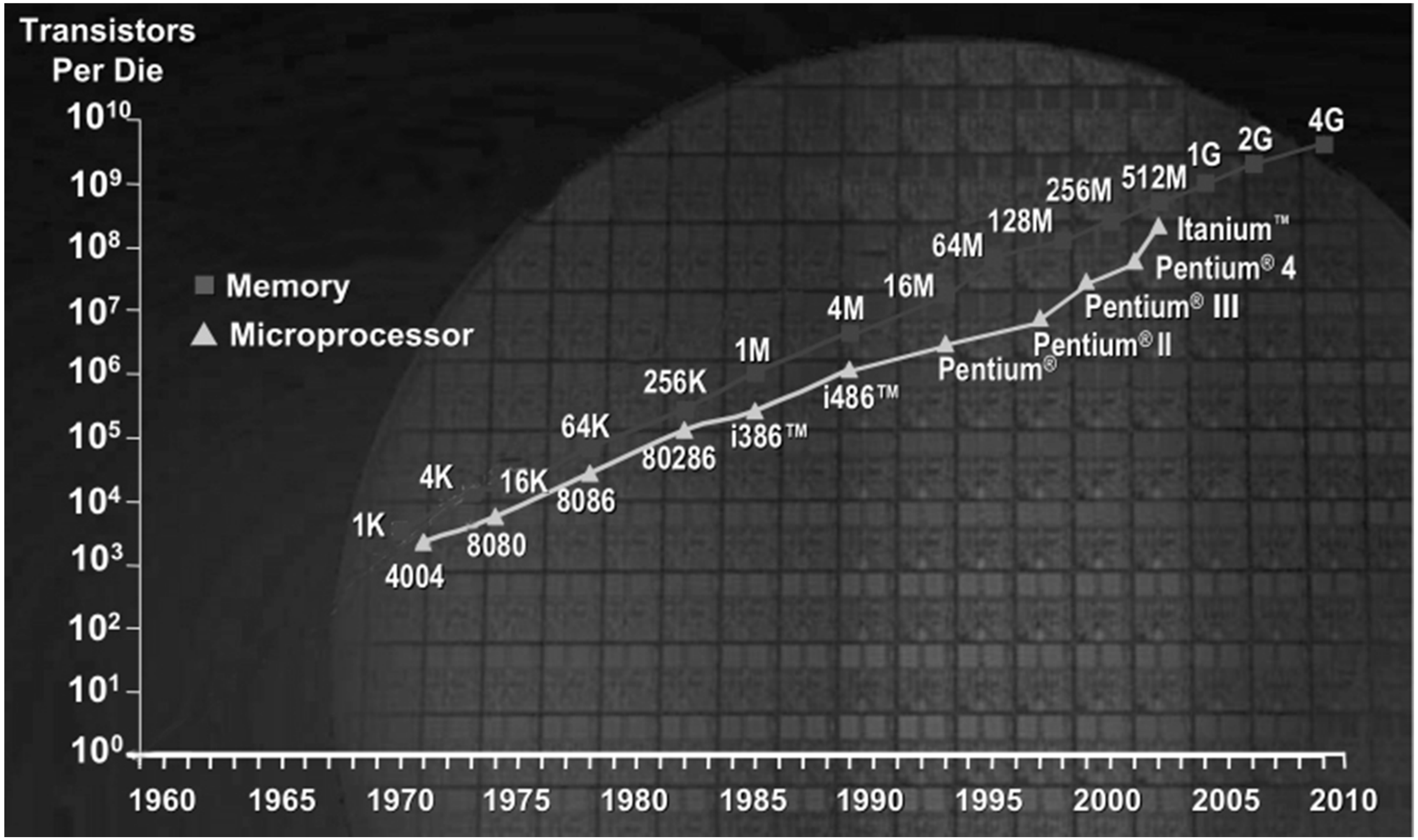
- **Moore's law describes a long-term trend in the history of computing hardware.**

- **Since the invention of the integrated circuit in 1958, the number of transistors that can be placed inexpensively on an integrated circuit has increased exponentially, doubling approximately every two years.**

- **The trend was first observed by Intel co-founder Gordon E. Moore in 1965.**

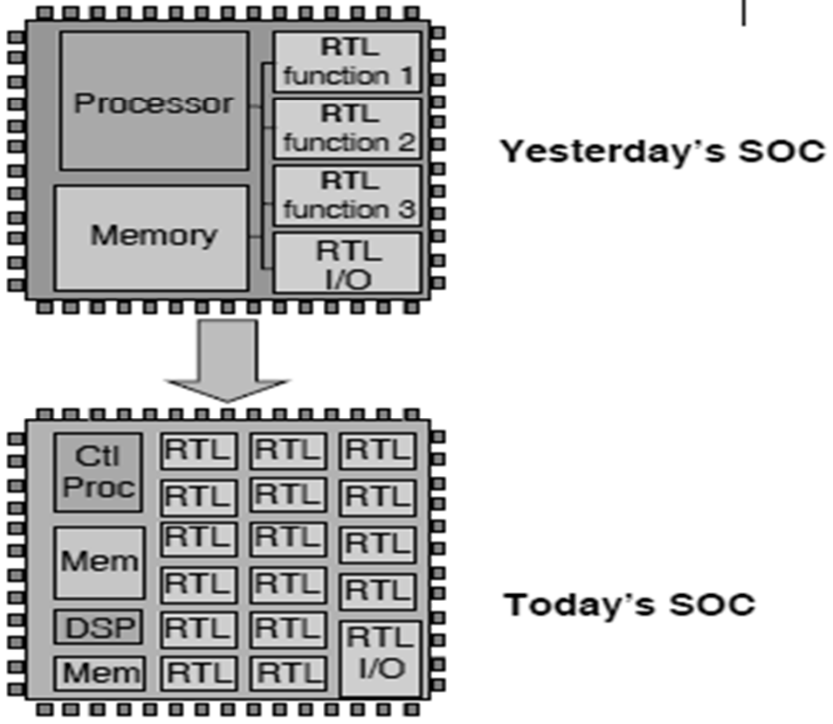
- **Almost every measure of the capabilities of digital electronic devices is linked to Moore's law: processing speed, memory capacity, etc.**

Moore’s law



Transistors Per Die
10^10
10^9
10^8
10^7
10^6
10^5
10^4
10^3
10^2
10^1
10^0
■ Memory
▲ Microprocessor
1K
4K
16K
64K
256K
1M
4M
16M
64M
128M
256M
512M
1G
2G
4G
4004
8080
8086
80286
i386™
i486™
Pentium®
Pentium® II
Pentium® III
Pentium® 4
Itanium™
1960
1965
1970
1975
1980
1985
1990
1995
2000
2005
2010

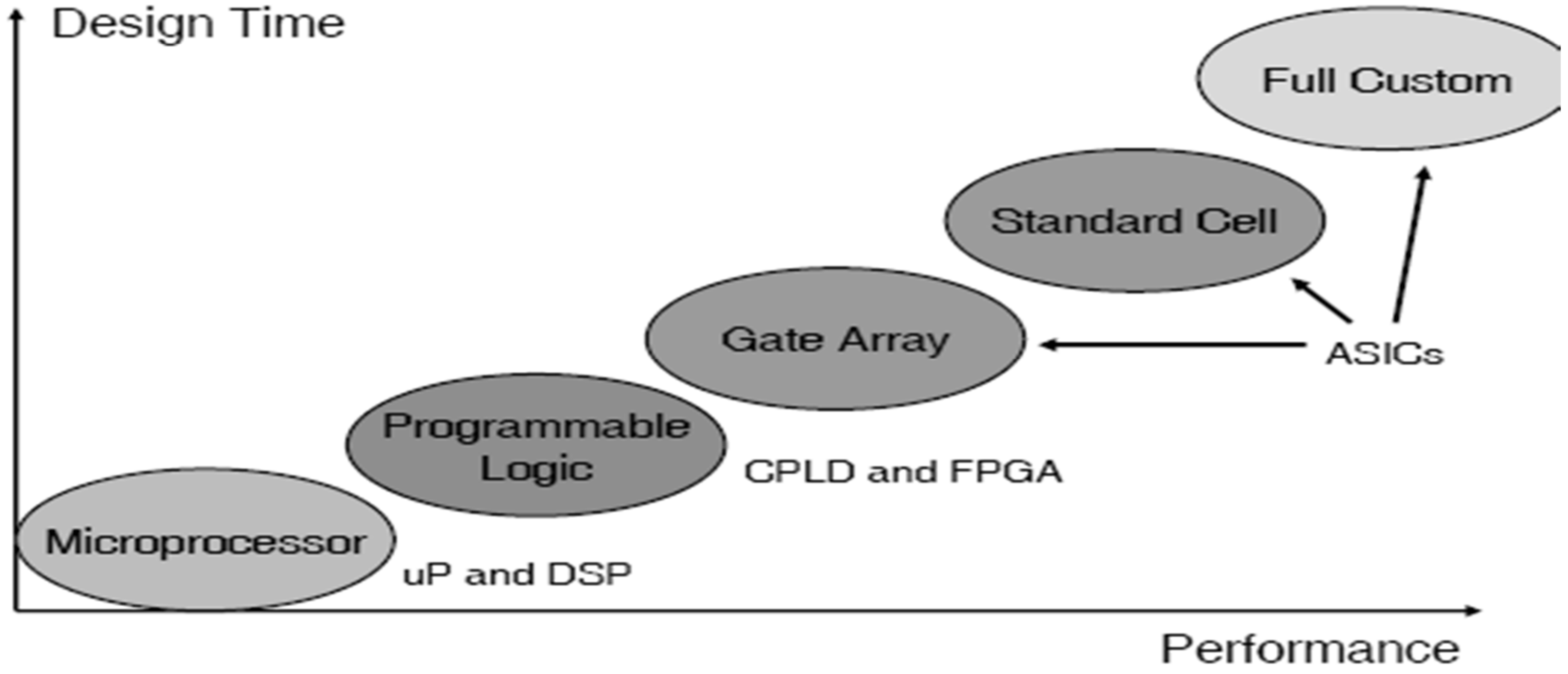
Moore's Law drives the development of System-in-Chip Architectures



The growing number of transistors on an SOC drives the trend towards more RTL blocks on the chip

Source: Leibson (DAC2004)

Alternatives mean longer design time, but allow higher performance



- **Thus ...**
- □ Microprocessors are used
- □ As key components in an embedded design
- □ Programmable Logic and ASICs are used
- □ for critical parts in a design
- □ An obective for an embedded system designer is to
- find the cheapest solution that meets the requirements
- Do not use a Pentium, when you
- only want to control a freezer...

- **Challenges in embedded system design**
- □ How much hardware do we need?
- □ How big is the CPU? Memory?
- □ How do we meet our deadlines?
- □ Faster hardware or cleverer software?
- □ How do we minimize power?
- □ Turn off unnecessary logic? Reduce memory
accesses?

- **Challenges, etc.**
- □ Does it really work?
- □ Is the specification correct?
- □ Does the implementation meet the spec?
- □ How do we test for real-time characteristics?
- □ How do we test on real data?
- □ How do we work on the system?
- □ Observability, controllability?
- □ What is our development platform?

Learn by Doing

Excel Thru Experimentation

Lead by Example

Acquire skills and get employed

Update skills and stay employed

THANK YOU