RV College of Engineering®

CS344AI: IOT and Embedded Computing

Unit-1:Introduction to Embedded Systems

Go, change the world®

Embedded System Products

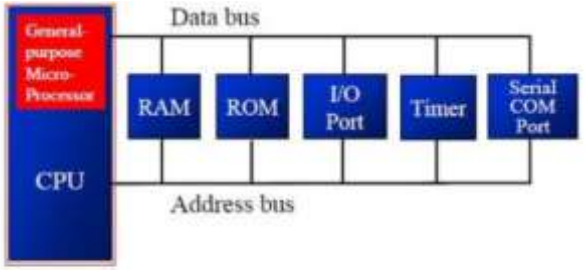
Examples of Embedded Systems

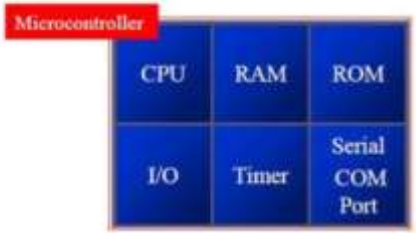

TELEVISION
REMOT CONTROL
REFRIGERATORS
VIDEO GAMES
ELEVATORS
SET-TOP BOX
PLANES
CARS

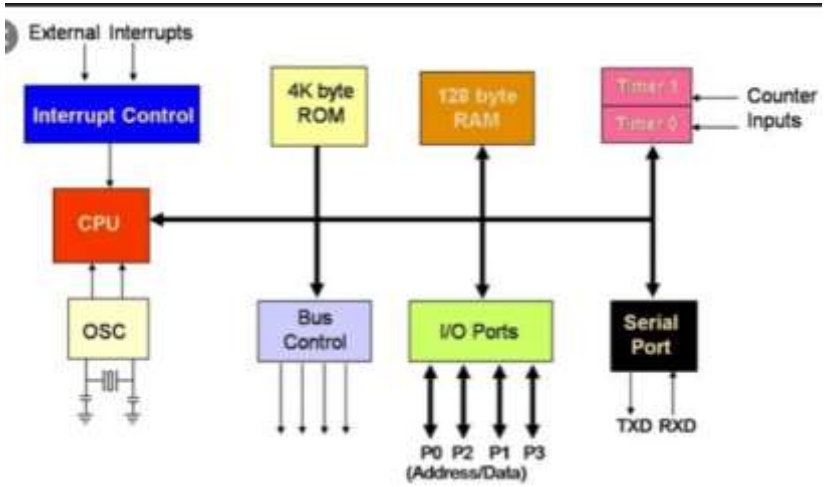

Wearable Computers
Smart Cards
Home Networking
Medical Imaging
Data Center
THE INTERNET
Wireless Pocket PC
XBOX
Internet Backbone Routers
Internet Appliances
Multimedia Distribution


ON/OFF
MODE
FAN
TEMP
COMFORT
OFF
CANCEL
ON
TIMER


SIEMENS
iSensoric
Off
Super
Hygiene
Freshen Up
Jeans
Sportswear
Shirts/Business
Rinse
Cottons
Cottons 60°C
Synthetics
Mix
Delicates
Wool
Drum Descale
Spin/Drain
iQ500
60
0:45
Temp.
Spin
Delay Start
varioSpeed
Easy Iron
Water Plus
Start/Reload
Status
ecoInfo


General-purpose Micro-Processor
CPU
Data bus
RAM
ROM
I/O Port
Timer
Serial COM Port
Address bus


Microcontroller
CPU
RAM
ROM
I/O
Timer
Serial COM Port


External Interrupts
Interrupt Control
4K byte ROM
128 byte RAM
Timer 1
Timer 0
Counter Inputs
CPU
OSC
Bus Control
I/O Ports
Serial Port
TXD RXD
P0 P2 P1 P3
(Address/Data)

Microprocessor vs. Microcontroller

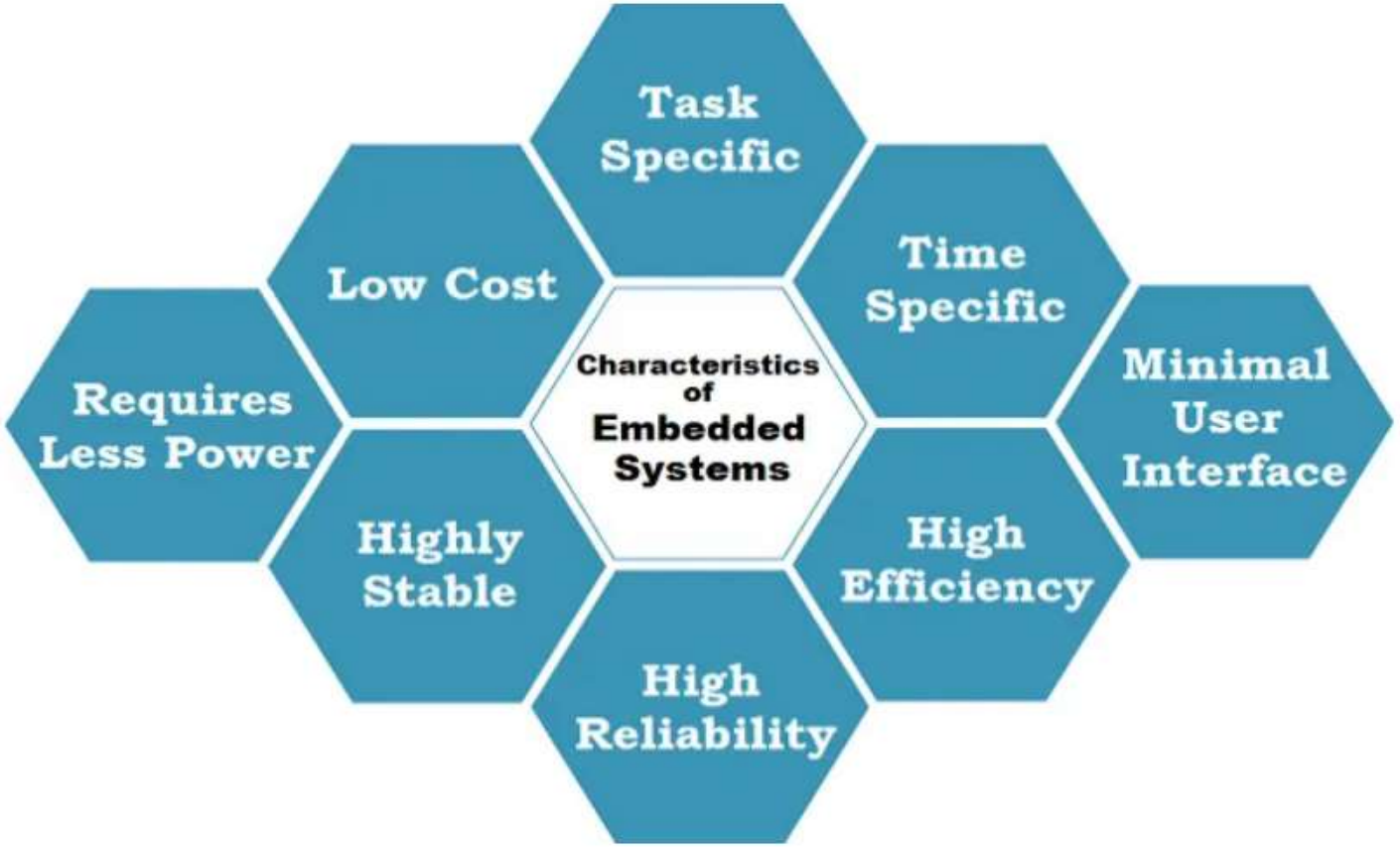
Microprocessor

- CPU is stand-alone, RAM, ROM, I/O, timer are separate
- designer can decide on the amount of ROM, RAM and I/O ports.
- expensive
- versatility
- general-purpose
- High processing power
- High power consumption
- Instruction sets focus on processing-intensive operations
- Typically 32/64 – bit
- Typically deep pipeline (5-20 stages)

Microcontroller

- CPU, RAM, ROM, I/O and timer are all on a single chip
- fixed amount of on-chip ROM, RAM, I/O ports
- for applications in which cost, power and space are critical
- single-purpose (control-oriented)
- Low processing power
- Low power consumption
- Bit-level operations
- Instruction sets focus on control and bit-level operations
- Typically 8/16 bit
- Typically single-cycle/two-stage pipeline

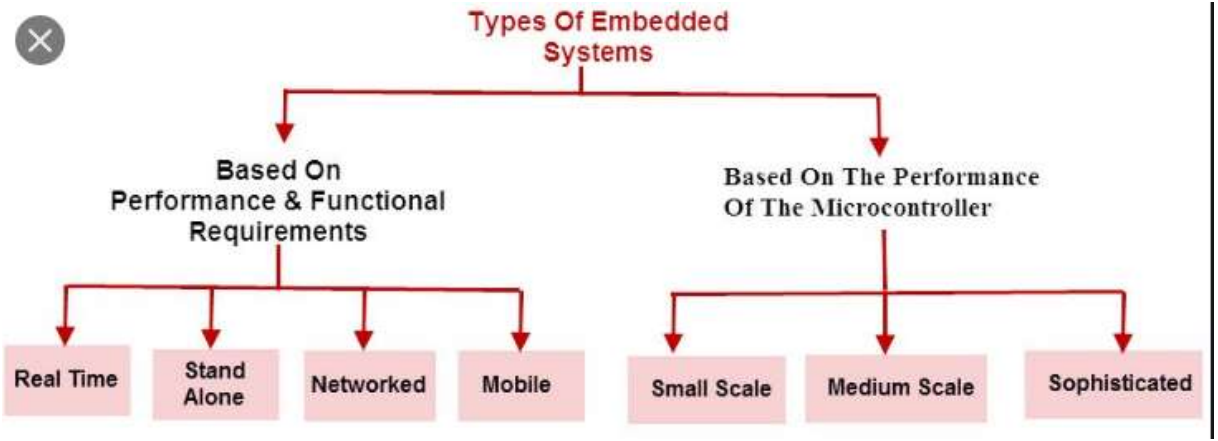
Characteristics of embedded systems



Some of the key characteristics of Embedded Systems are as mentioned below.

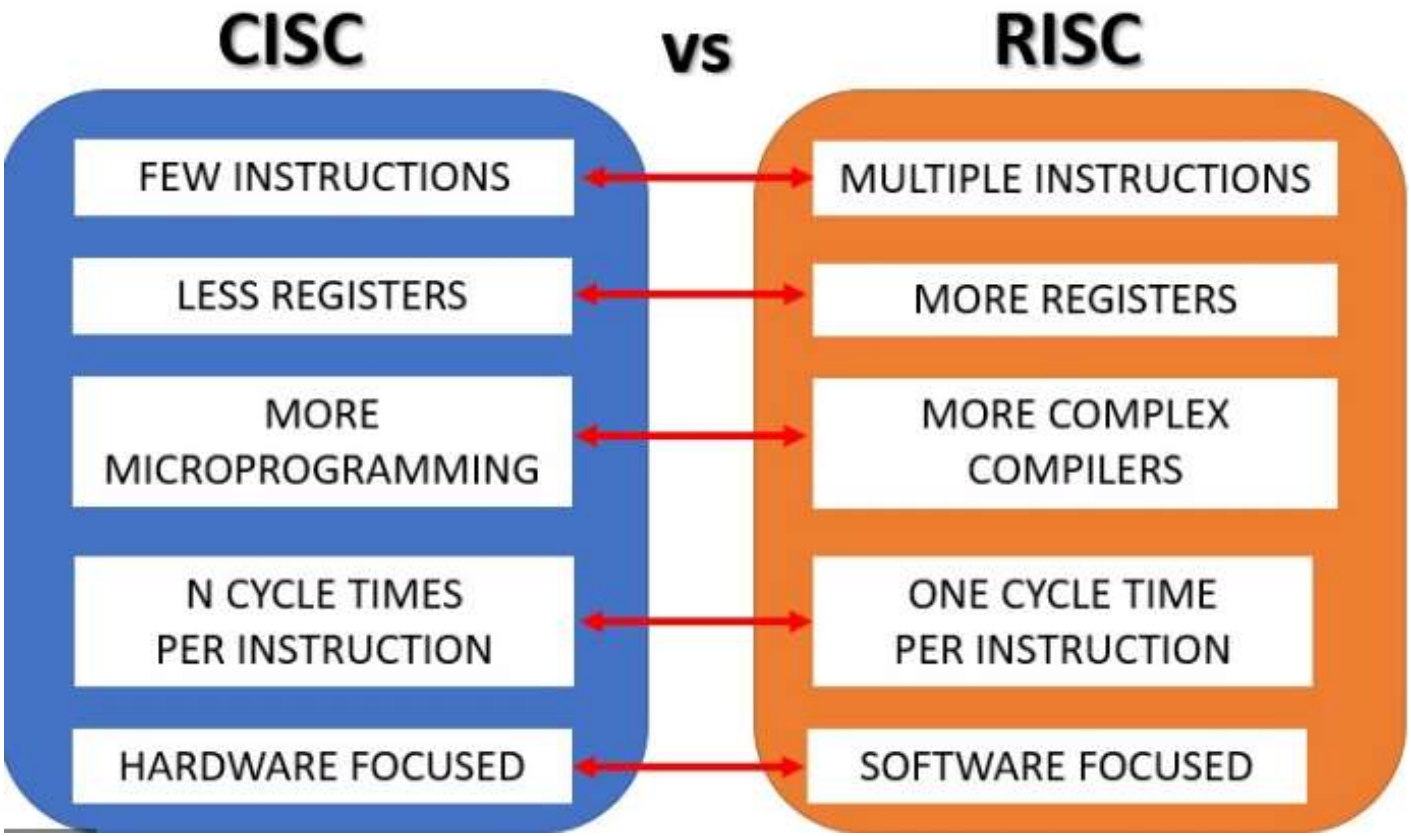
1. **All Embedded Systems are task specific. They do the same task repeatedly /continuously over their lifetime. An mp3 player will function only as an mp3 player.**
2. **Embedded systems are created to perform the task within a certain time frame. It must therefore perform fast enough. A car's brake system, if exceeds the time limit, may cause accidents.**
3. **They have minimal or no user interface (UI). A fully automatic washing machine works on its own after the program is set and stops once the task is over.**
4. **Some embedded systems are designed to react to external stimuli and react accordingly. A thermometer, a GPS tracking device.**
5. **Embedded systems are built to achieve certain efficiency levels. They are small sized, can work with less power and are not too expensive.**
6. **Embedded systems cannot be changed or upgraded by the users. Hence, they must rank high on reliability and stability. They are expected to function for long durations without the user experiencing any difficulties.**
7. **Microcontroller or microprocessors are used to design embedded systems.**
8. **Embedded systems need connected peripherals to attach input & output devices.**
9. **The hardware of an embedded-system is used for security and performance. The Software is used for features.**

Criteria	General Purpose Computing System	Embedded System
Contents	A system which is a combination of a generic hardware and a General Purpose Operating System for executing a variety of applications.	A system which is a combination of special purpose hardware and embedded OS for executing a specific set of applications.
OS	It contains a general purpose operating system (GPOS).	It may or not contain an operating system for functioning.
Alterations	Applications are alterable (programmable) by the user. (It is possible for the end user to re-install the OS and also add or remove user applications.)	The firmware of the embedded system is pre-programmed and it is non-alterable by the end-user.
Key factor	Performance is the key deciding factor in the selection of the system. Faster is better.	Application specific requirements (like performance, power requirements, memory usage, etc.) are key deciding factors.
Power Consumption	More	Less
Response Time	Not critical	Critical for some applications
Execution	Need not be deterministic	Deterministic for certain types of ES like 'Hard Real Time' systems.



- **Real-Time Embedded Systems:**
  - Traffic control system
  - Military usage in defense sector
  - Medical usage in health sector
- **Stand Alone Embedded Systems** :
  - MP3 players
  - Microwave ovens
  - calculator

- **Networked Embedded Systems :**
  - Home security systems
  - ATM machine
  - Card swipe machine
- **Mobile Embedded Systems :**
  - MP3 player
  - Mobile phones
  - Digital Camera


CISC
vs
RISC
FEW INSTRUCTIONS
MULTIPLE INSTRUCTIONS
LESS REGISTERS
MORE REGISTERS
MORE MICROPROGRAMMING
MORE COMPLEX COMPILERS
N CYCLE TIMES PER INSTRUCTION
ONE CYCLE TIME PER INSTRUCTION
HARDWARE FOCUSED
SOFTWARE FOCUSED

Introduction to ARM

- Introduction
- Architecture
- Programmers Model
- Instruction Set

RV College of Engineering®

ARM Processor Families

ARM Cortex Processor Families

- ***A Profile*** **:** Application processors which are designed to handle complex
  applications

  such as high-end embedded operating systems (OSs)

  ●-These processors requiring the highest processing power, virtual memory system support with memory management units (MMUs).

  ● E.g: High-end mobile phones(Samsung S6: Samsung Eqynos).

- ***R Profile*****:** Real-time, high-performance processors targeted primarily at the higher end of the real-time market.

  ●-Those applications, such as high-end breaking systems and hard drive controllers, in which high processing power and high reliability are essential and for which

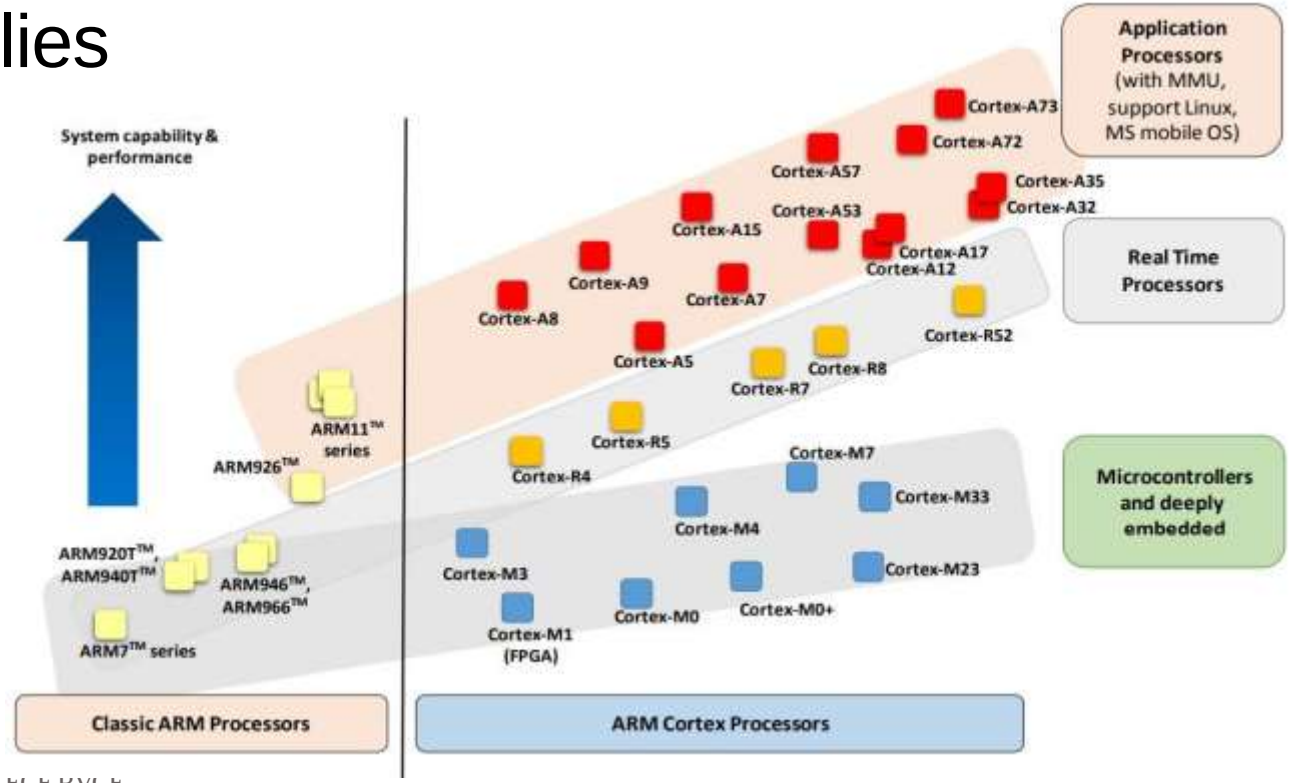
  low latency is important

ARM Processor Families contd..

- ***Neoverse N1:*** Delivering market-leading server performance at the half the power, the N1 CPU ushers in a revolutionary era of compute efficiency or performance/watt. Targeted for server work loads and data centres.
- ***Ethos :*** Highly scalable and efficient NPU, the Ethos enables new immersive applications with a 2.5x increase in single-core performance now scalable from 1 to 10 TOP/s and beyond through many-core technologies. It provides flexibility to optimize the ML capability and IoT applications.

Source: www.arm.com

ARM Processor Families



Typical Application Domains

Cortex - A

Highest performance

Optimised for rich operating systems

Cortex - R

Fast response

Optimised for high performance, hard real-time applications

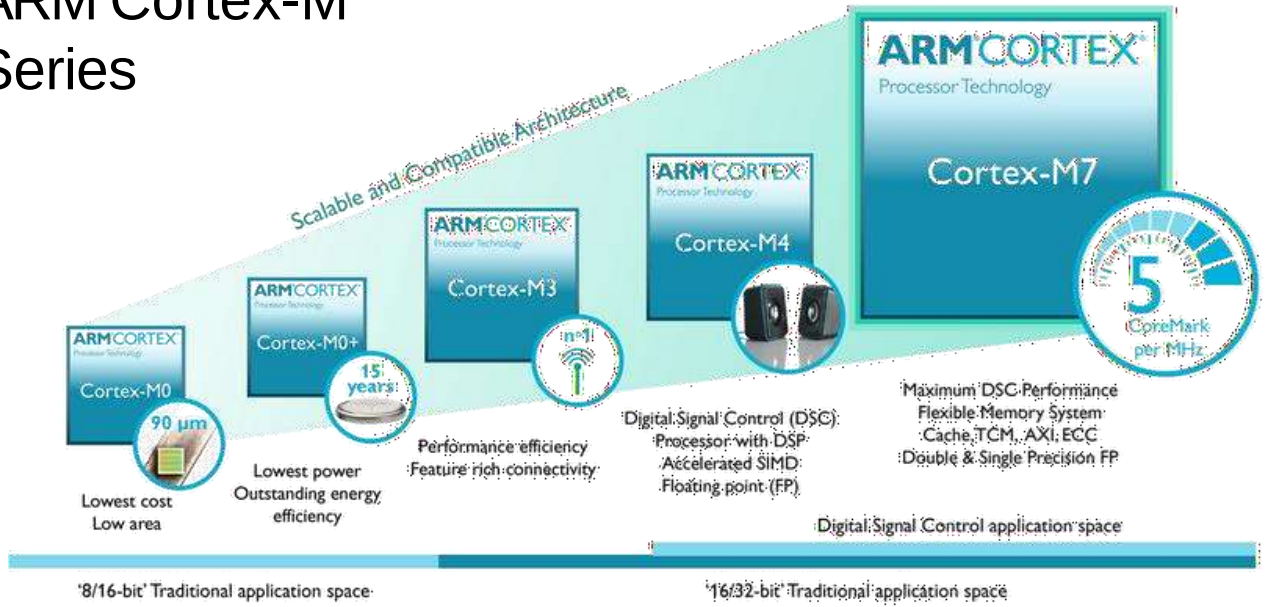
Cortex - M

Smallest/lowest power

Optimised for discrete processing and microcontrollers

Source: google

ARM Cortex-M Series

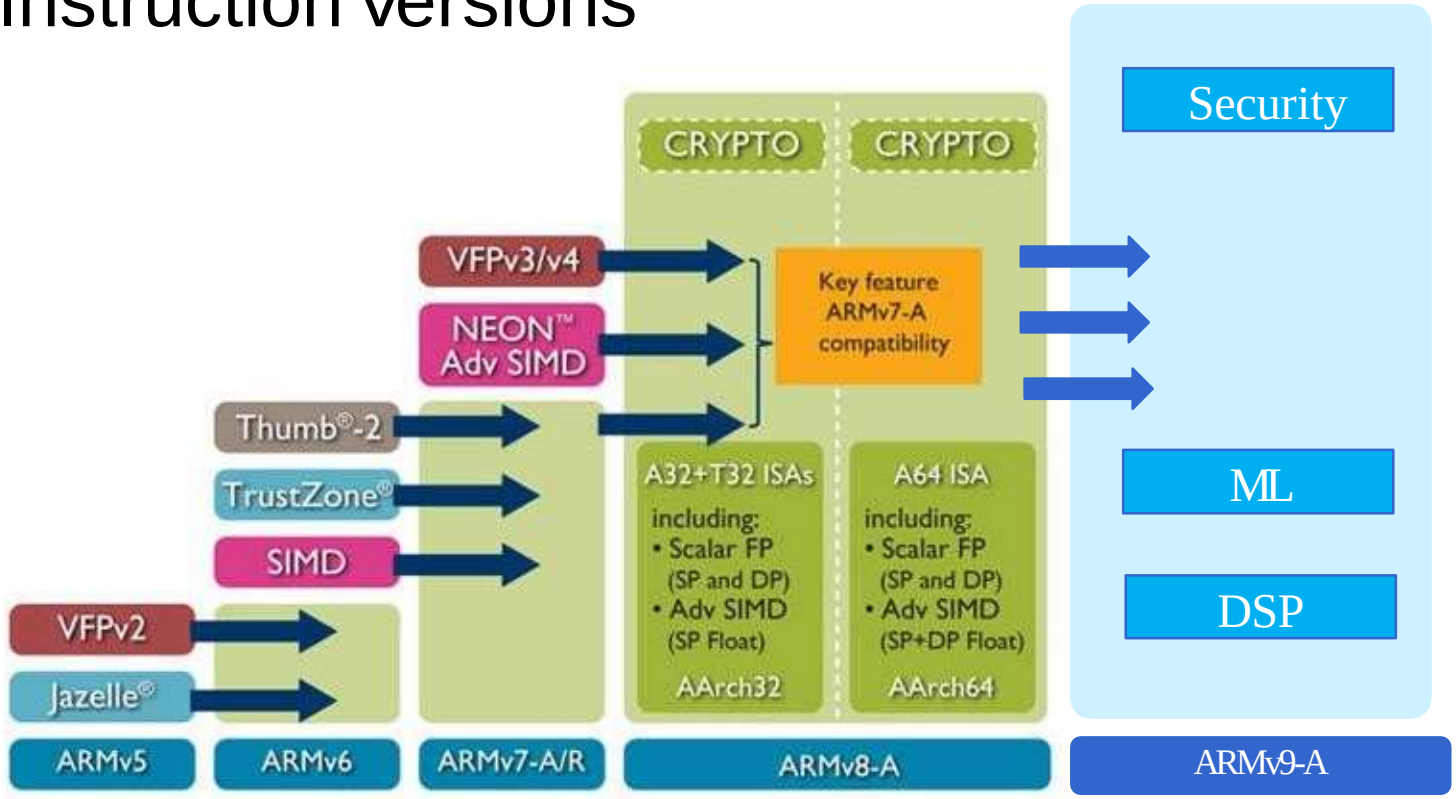


For more info on CoreMark:
www.eembc.org

CoreMar k

- CoreMarkis small, portable, easy to understand, free, and displaysa single number benchmark score to represent the speed of processors.
- A processorwith higher CoreMarknumber is faster. This rating is given by CoreMark consortium.
- Earlier DMIPS(Dhrystone Million Instructions Per second) is the metrics used specify the speed of processors.
- To avoid the problems associated with the DMIPS(Dhrystone MIPS), CoreMark rating is introduced.
- More info on CoreMark: https://www.eembc.org/coremark/

ARM Instruction Versions

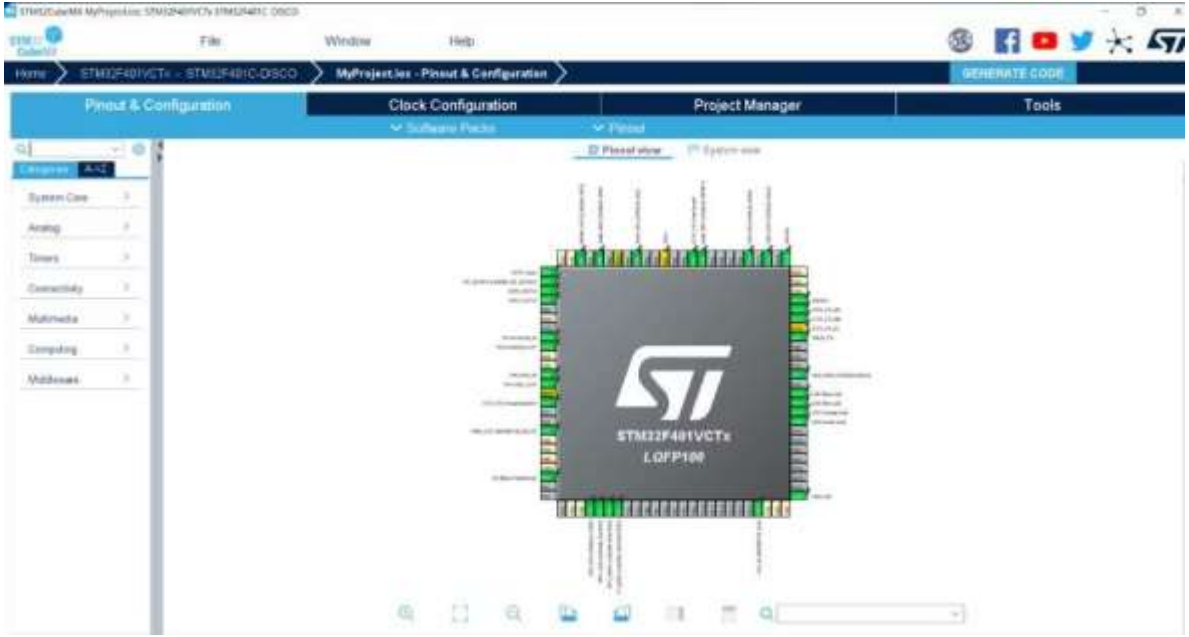


ARMv7-M ARMv8-R/M

www.arm.com

STM32 CubeMX

- ✓ Embedded finder: MCUs/MPUs, boards, examples
- ✓ Pinout, Peripherals & Middleware configuration
- ✓ Clock configuration
- ✓ Project generation for Keil, IAR & STM32CubeIDE
- ✓ Power consumption estimation
- ✓ Software package manager
- ✓ Embedded tutorial videos and documentation



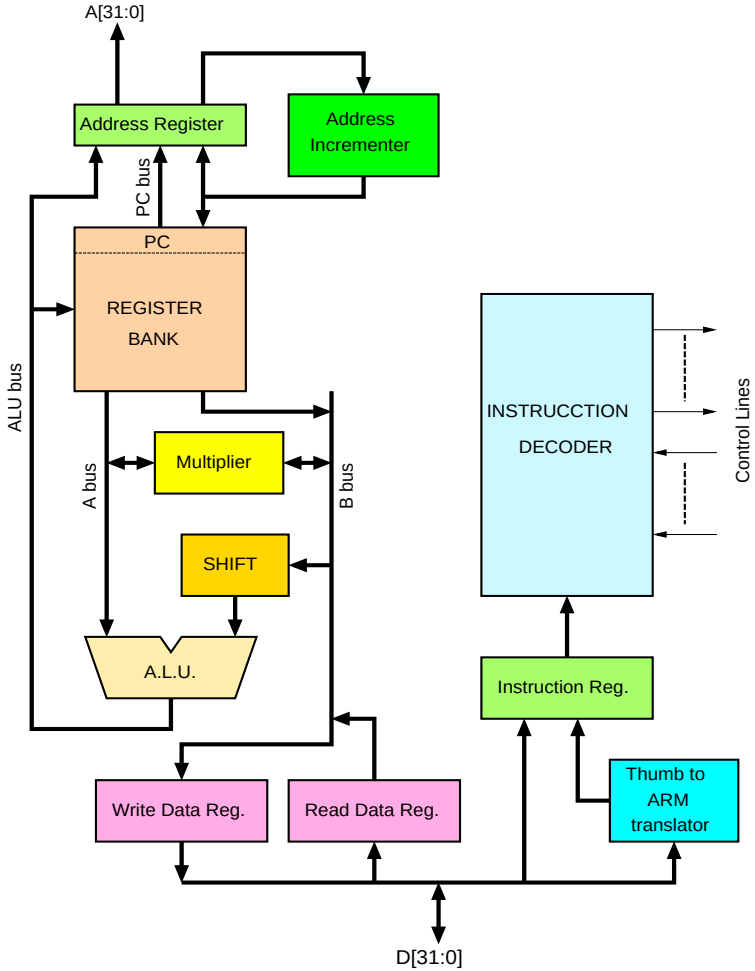
History of ARM

- ARM (Acorn RISC Machine) started as a new, powerful, CPU design for the replacement of the 8-bit 6502 in Acorn Computers (Cambridge, UK, 1985)
- First models had only a 26-bit program counter, limiting the memory space to 64 MB (not too much by today standards, but a lot at that time).
- 1990 spin-off: ARM renamed Advanced RISC Machines
- ARM now focuses on Embedded CPU cores
  - IP licensing: Almost every silicon manufacturer sells some microcontroller with an ARM core. Some even compete with their own designs.
  - Processing power with low current consumption
    - Good MIPS/Watt figure
    - Ideal for portable devices
  - Compact memories: 16-bit opcodes (Thumb)
- New cores with added features
  - Harvard architecture (ARM9, ARM11, Cortex)
  - Floating point arithmetic
  - Vector computing (VFP, NEON)
  - Java language (Jazelle)

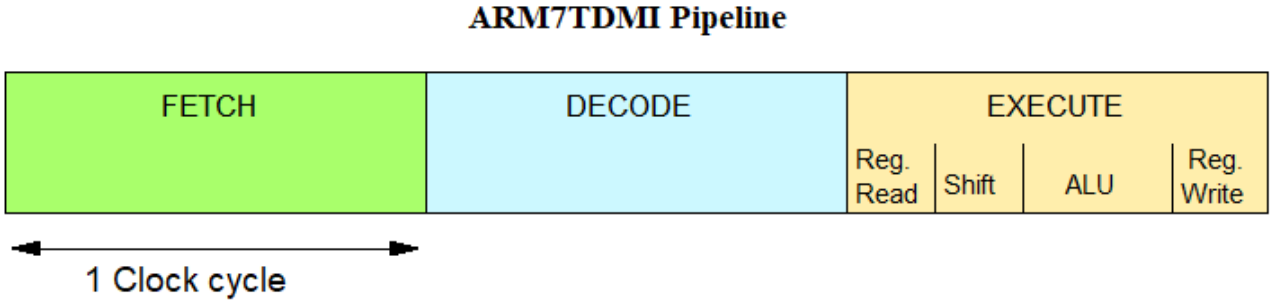
Facts

- 32-bit CPU
- 3-operand instructions (typical): ADD Rd,Rn,Operand2
- RISC design…
  - Few, simple, instructions
  - Load/store architecture (instructions operate on registers, not memory)
  - Large register set
  - Pipelined execution
- … And some very specific details
  - No stack. Link register instead
  - PC as a regular register
  - Conditional execution of all instructions
  - Flags altered or not by data processing instructions (selectable)
  - Concurrent shifts/rotations (at the same time of other processing)
  - …

ARM7TDMI Block Diagram

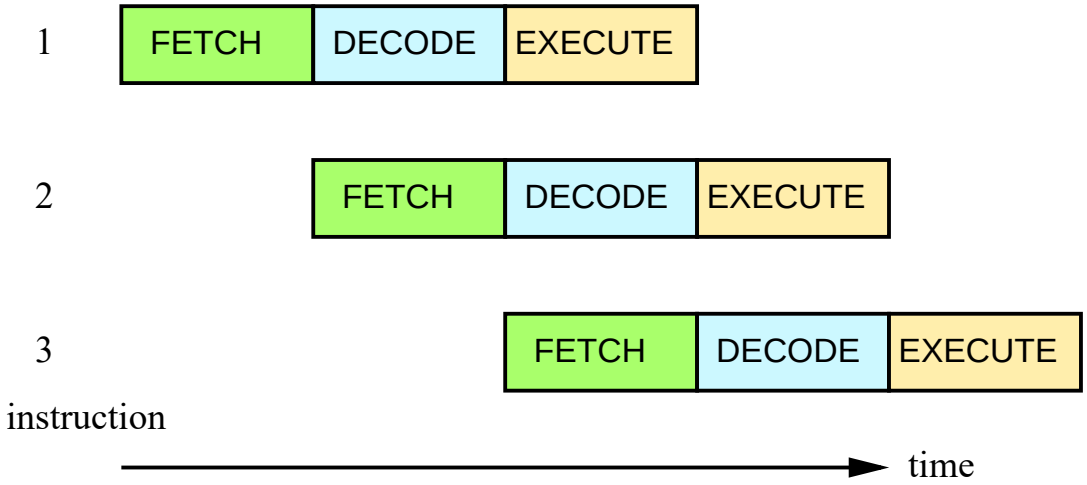


ARM Pipelining examples



- Fetch: Read Op-code from memory to internal Instruction Register
- Decode: Activate the appropriate control lines depending on Opcode
- Execute: Do the actual processing

ARM7TDMI Pipelining (I)



- Simple instructions (like ADD) Complete at a rate of one per cycle

Data Sizes and Instruction Sets

- The ARM is a 32-bit architecture.
- When used in relation to the ARM:
  - **Byte** means 8 bits
  - **Halfword** means 16 bits (two bytes)
  - **Word** means 32 bits (four bytes)
- Most ARM's implement two instruction sets
  - 32-bit **ARM** Instruction Set
  - 16-bit **Thumb** Instruction Set

Processor Modes

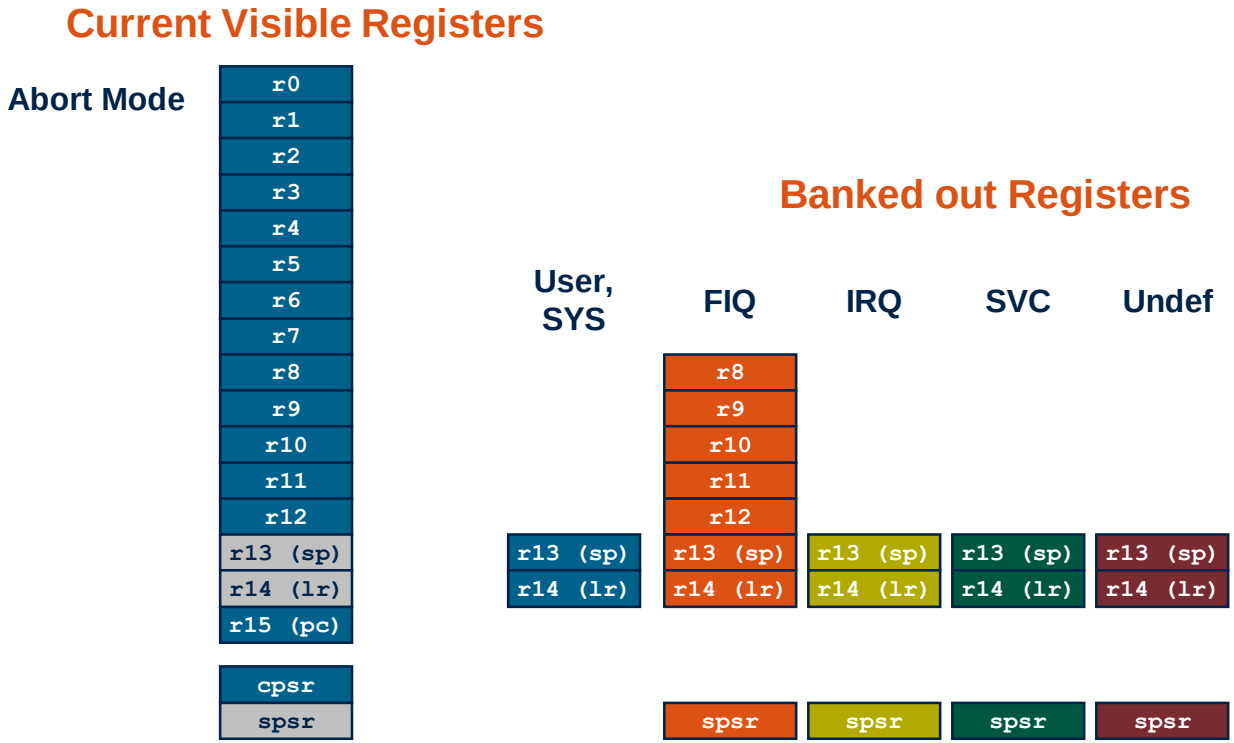
- The ARM has seven operating modes:
  - **User** : unprivileged mode under which most tasks run
  - **FIQ** : entered when a high priority (fast) interrupt is raised
  - **IRQ** : entered when a low priority (normal) interrupt is raised
  - **SVC** : (Supervisor) entered on reset and when a Software Interrupt instruction is executed
  - **Abort** : used to handle memory access violations
  - **Undef** : used to handle undefined instructions
  - **System** : privileged mode using the same registers as user mode

The Registers

- ARM has 37 registers all of which are 32-bits long.
  - 1 dedicated program counter
  - 1 dedicated current program status register
  - 5 dedicated saved program status registers
  - 30 general purpose registers
- The current processor mode governs which of several banks is accessible. Each mode can access
  - a particular set of r0-r12 registers
  - a particular r13 (the stack pointer, sp) and r14 (the link register, lr)
  - the program counter, r15 (pc)
  - the current program status register, cpsr

  Privileged modes (except System) can also access
  - a particular spsr (saved program status register)

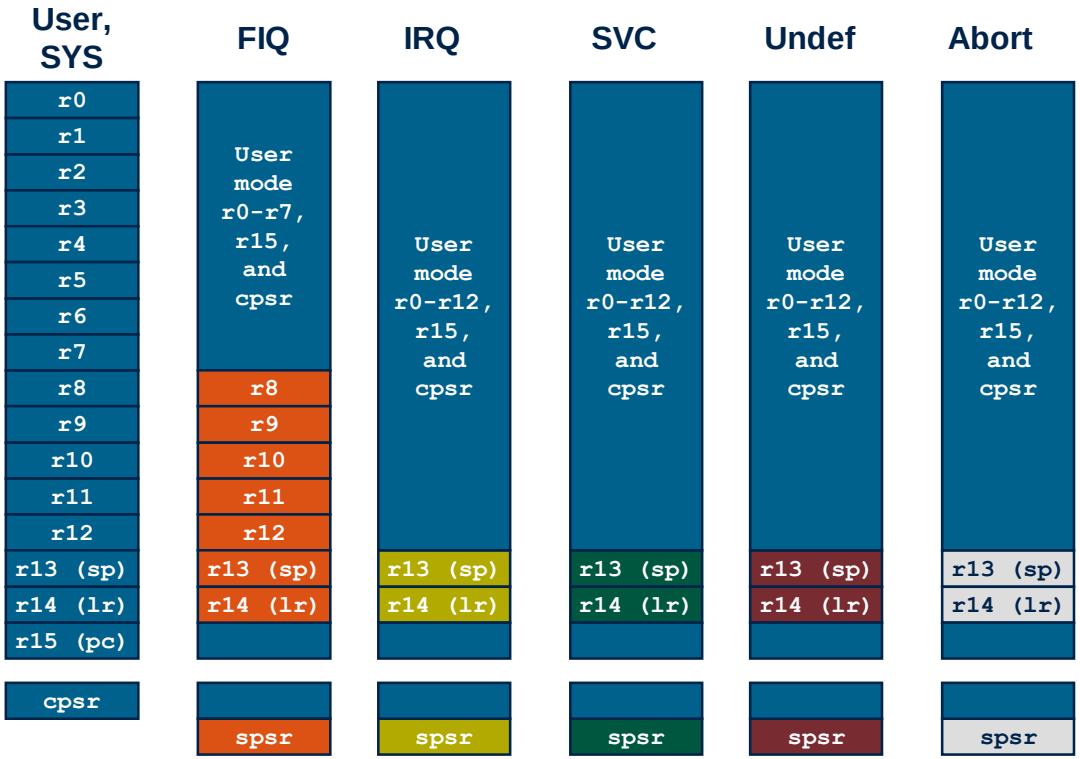
The ARM Register Set



Special Registers

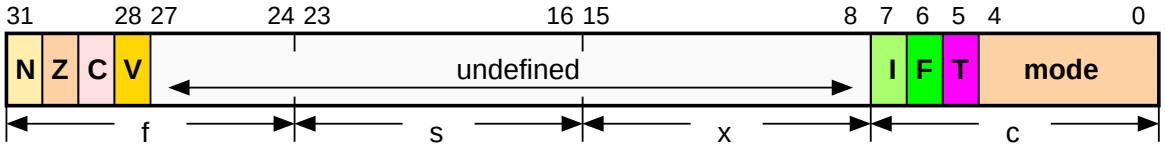
- Special function registers:
  - **PC** (R15): Program Counter. Any instruction with PC as its destination register is a program branch
  - **LR** (R14): Link Register. Saves a copy of PC when executing the BL instruction (subroutine call) or when jumping to an exception or interrupt routine
    - It is copied back to PC on the return from those routines
  - **SP** (R13): Stack Pointer. There is **no stack** in the ARM architecture. Even so, R13 is usually reserved as a pointer for the program-managed stack
  - **CPSR** : Current Program Status Register. Holds the visible status register
  - **SPSR** : Saved Program Status Register. Holds a copy of the previous status register while executing exception or interrupt routines
    - It is copied back to CPSR on the return from the exception or interrupt
    - No SPSR available in User or System modes

Register Organization Summary



Note: System mode uses the User mode register set

Program Status Registers



- Condition code flags
  - N = **N**egative result from ALU
  - Z = **Z**ero result from ALU
  - C = ALU operation **C**arried out
  - V = ALU operation o**V**erflowed

	Mode bits
10000	**User**
10001	**FIQ**
10010	**IRQ**
10011	**Supervisor**
10111	**Abort**
11011	**Undefined**
11111	**System**

Interrupt Disable bits.

I = 1: Disables the IRQ.

F = 1: Disables the FIQ.

T Bit (Arch. with Thumb mode only)

T = 0: Processor in ARM state

T = 1: Processor in Thumb state

Never change T directly (use BX instead)

Changing T in CPSR will lead to unexpected behavior due to pipelining

Tip: Don't change undefined bits.

This allows for code compatibility with newer ARM processors