

# **ESET 349 - Microcontroller Architecture**

## **Architecture**

Dr. Muhammad Faeyz Karim

# Microcontroller Architecture

## History of ARM and Microcontroller

- 1980s and 1990s – Intel and Motorola dominated the field of Microprocessors and Microcontroller
- Late 1990s – ARM Microcontroller started to challenge the dominance of Intel and Motorola in the 32-bit market



# Microcontroller Architecture

## Currently Available Microcontrollers

- 32-bit
  - ARM, AVR32 (Atmel), ColdFire (Freescale), MIPS32, PIC32, (Microchip), PowerPC, TriCore (Infineon), SuperH
- 16-bit
  - MSP430 (TI), HCS12 (Freescale), PIC24 (Microchip), dsPIC (Microchip)
- 8-bit
  - 8051, AVR (Atmel), HCS08 (Freescale), PIC16, PIC18



# Microcontroller Architecture

## The ARM Family History

- The ARM came out of a company called **Acorn Computers** in UK in 1980s
- Professor Steve Furber of Manchester University worked with Sophie Wilson to define the ARM Architecture and instructions
- The VLSI Technology Corp. produced the first ARM chip in 1985 for Acorn Computers.
- The technology designated as **Acorn RISC Machine (ARM)**.

# Microcontroller Architecture

- ARM (Acorn) started single-chip microcontroller embedded system
- Apple used ARM chip for PDA (Personal Digital Assistants)
- Acorn created new company called **ARM**
- ARM's entire fortune is on selling the rights of this new CPU to other silicon manufacturers and design houses.
- Since early 1990s, an ever-increasing number of companies have licensed the right to make the ARM chip.



# Microcontroller Architecture

ARM Company Milestone ([www.arm.com](http://www.arm.com))

- 1983 – Acorn and VLSI began designing of ARM processor
- 1985 – Acorn developed world's first commercial RISC processor
  - ARMv1 had 2500 transistors and worked with frequency of 4MHz
- 1989 – ARM (Acorn) introduced ARMv3 with 4KB cache at 25MHz



# Microcontroller Architecture

- 1990 – VLSI becomes an investor and the first licensee
- 1991 – ARM introduced its first embeddable RISC core, the ARM6 solution using ARMv3 architecture
- 1992 – GEC Plessey and Sharp licensed ARM technology
- 1993 – ARM introduced ARM7 core. TI licensed ARM technology.
- 1995 – ARM introduced Thumb architecture extension, which gives 32-bit RISC performance at 16-bit system cost



# Microcontroller Architecture

- 1996 – ARM and VLSI Technology introduced ARM810 microprocessor
  - ARM and Microsoft worked together to extend Windows CE to the ARM architecture
- 1997 – Hyundai, Lucent, Philips, Rockwell and Sony licensed ARM Technology
  - ARM9TDMI family announced
- 1998 – HP, IBM, Matsushita, Seiko Epson, and Qualcomm licensed ARM Technology
  - ARM developed synthesizable version of the ARM7TDMI core
  - ARM Partners shipped more than 50 million ARM-powered products





# Microcontroller Architecture

- 2002 – ARM shipped over one billion of its microprocessor cores
  - ARM licensed to Seagate, Broadcom, Philips, Matsushita, Micrel, eSilicon, Chip Express and ITI
  - ARM launched the ARM11 microarchitecture
  - ARM launched its RealView family of development tools
  - Flextronics became the first ARM Licensing Partner program member, allowing it to sub-license ARM technology to its own customers
- 2004 -



# Microcontroller Architecture

- 2004 – ARM Cortex family based on ARMv7 architecture announced
  - ARM Cortex-M3 family – the first of a new Cortex family
  - MPCore multiprocessor launched – the first integrated multiprocessor
  - OptimoDE technology launched, the groundbreaking embedded signal processing core
- 2005 – ARM acquired Keil software
  - ARM Cortex-A8 processor announced
- 2007 -



# Microcontroller Architecture

- 2007 – Five billionth ARM Powered processor shipped to the mobile device market
  - ARM Cotex-M1 processor launched – the first ARM processor designed for FPGA (Field Programmable Gate Arrays)
  - RealView Profiler for Embedded Software Analysis introduced
  - ARM unveiled Cortex-A9 processor for scalable performance and low-power designs
- 2008 -



# Microcontroller Architecture

- 2008 – ARM announced 10 billionth processor shipment
  - ARM Mali-200 GPU Worlds First to achieve Khronos Open GL ES 2.0 conformance at 1080p HDTV resolution
- 2009 – ARM announces 2GHz capable Cortex-A9 dual core processor implementation
  - ARM launches its smallest, lowest power, most energy efficient processor, Cortex M0



# Microcontroller Architecture

- 2010 – ARM launches **Cortex-M4** processor for high performance digital signal control
- ARM together with key Partners form Linaro to speed rollout for Linux-based devices
- Microsoft becomes an ARM Architecture Licensee
- ARM & TSMC sign long-term agreement to achieve optimized System-on-Chip based on ARM processors, extending down to 20nm
- ARM launches Cortex-A15 MPCore processor
- ARM Mali becomes the most widely licensed embedded GPU architecture
- ARM Mali-T604 Graphics Processing Unit introduced providing industry-leading graphics performance with energy-efficient profile



# Microcontroller Architecture

- 2011 –
  - Microsoft unveils Windows on ARM at CES 2011
  - IBM and ARM collaborate to provide compressive design performance down to 14nm
  - ARM and UMC extend partnership into 28nm
  - Cortex-A7 processor launched
  - ARMv8 unveiled at TechCon
  - AMP announced license and plans for first ARMv8-based processor
  - ARM Mali-T658 GPU launched
  - ARM expands R&D presence in Taiwan with Hsinchu Design Center
  - ARM and Avnet launch Embedded Software Store (ESS)
  - ARM, Cadence and TSMC tape out first 20nm Cortex-A15 multicore processor
- 2012 –
  - ARM, Gemalto and G&D form joint venture to deliver next-generation mobile security
  - First Windows RT (Windows on ARM) devices revealed
  - ARM, AMD, Imagination, MediaTek and TI founding members of Heterogenous System Architecture (HAS) Foundation
  - ARM and TSMC work together on FinFET processor technology for next-generation 64-bit ARM processors
  - ARM forms first UK forum to create technology blueprint “Internet of Things” devices
  - ARM named one of Britain’s Top Employers
  - MIT Technology Reviews named ARM in its list of 50 Most Innovative Companies



# Microcontroller Architecture

- ARM company receives its entire **revenue from licensing** the ARM to other companies since it does not own state of the art chip fabrication facility
- Selling IP (Intellectual Property) has made ARM one of the most widely used CPU architecture in the world.



## 2 ARM States

- ARM State : Standard 32-bit length instructions are used.
- THUMB State : Compressed 16-bit length instructions are used. Provides flexibility of putting more instructions into the same amount of memory or reduce the amount of memory needed for a given design.


















# 7 ARM Processor Modes

| Mode             | Description   |                   |
|------------------|---|-------------------|
| Supervisor (SVC) | Entered on reset and when a Software Interrupt (SWI) Instruction is executed. | Privileged modes  |
| FIQ              | Entered when a high priority (fast ) interrupt is raised                      |                   |
| IRQ              | Entered when a low priority (normal) interrupt is raised.                     |                   |
| Abort            | Used to handle memory access violations                                       |                   |
| Undef            | Used to handle undefined instructions   |                   |
| System           | Privileged mode using the same registers as User mode                         |                   |
| User             | Mode under which most applications/OS tasks run                               | Unprivileged mode |

# Registers

- A Register is the most fundamental storage area on the chip, can be used to stored any data you wish.
- ARM7TDMI has 37 32-bit registers
  - 30 general-purpose registers
  - 6 status registers
  - A program counter (PC)
- Not all registers are visible at any one time.

## ARM State General Registers and Program Counter

| System & User | FIQ   | Supervisor  | Abort   | IRQ   | Undefined   |
|---------------|---|---|---|---|---|
| R0            | R0  | R0  | R0  | R0  | R0  |
| R1            | R1  | R1  | R1  | R1  | R1  |
| R2            | R2  | R2  | R2  | R2  | R2  |
| R3            | R3  | R3  | R3  | R3  | R3  |
| R4            | R4  | R4  | R4  | R4  | R4  |
| R5            | R5  | R5  | R5  | R5  | R5  |
| R6            | R6  | R6  | R6  | R6  | R6  |
| R7            | R7  | R7  | R7  | R7  | R7  |
| R8            |  R8_fiq  | R8  | R8  | R8  | R8  |
| R9            |  R9_fiq  | R9  | R9  | R9  | R9  |
| R10           |  R10_fiq | R10   | R10   | R10   | R10   |
| R11           |  R11_fiq | R11   | R11   | R11   | R11   |
| R12           |  R12_fiq | R12   | R12   | R12   | R12   |
| R13           |  R13_fiq |  R13_svc |  R13_abt |  R13_irq |  R13_und |
| R14           |  R14_fiq |  R14_svc |  R14_abt |  R14_irq |  R14_und |
| R15 (PC)      | R15 (PC)  | R15 (PC)  | R15 (PC)  | R15 (PC)  | R15 (PC)  |



## ARM State Program Status Registers

|      |  |  |  |  |  |
|------|--|--|--|--|--|
| CPSR | CPSR   | CPSR   | CPSR   | CPSR   | CPSR   |
|      |  SPSR_fiq |  SPSR_svc |  SPSR_abt |  SPSR_irq |  SPSR_und |

 = banked register

# Banking of registers

- For example, when the processor changes to FIQ mode, a large number of registers (r8-r14) are banked, or swap out.
- Done to save the current state of the machine. During an interrupt, it is necessary to stop what you are doing and begin work on a task.
- Rather than backing the original content of the registers into the external memory which takes time, the machine simply used a new set of registers instead. Hence execution speed improves!!!

# Reserved Registers

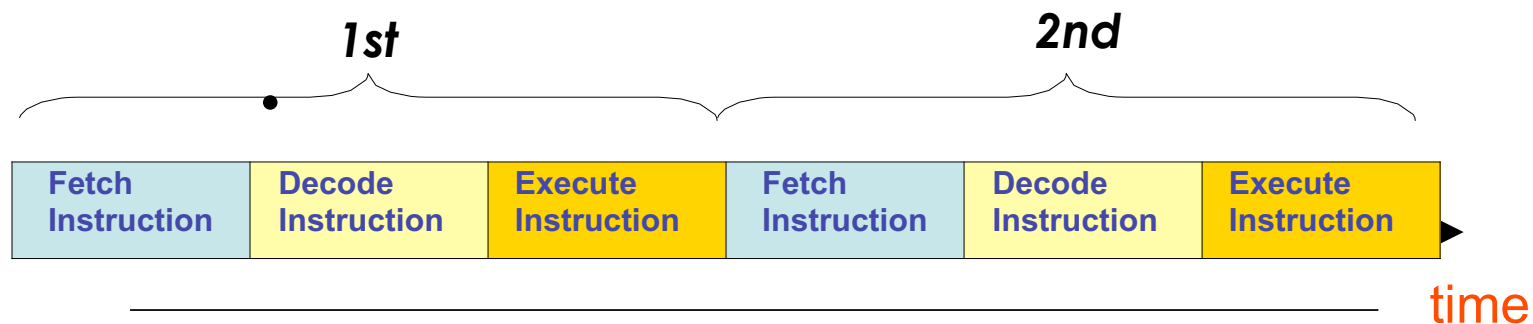
- R13 is also known as Stack Pointer (SP). It holds the address of the stack in memory and a unique stack pointer exists in each mode (except System mode which shares the User mode stack pointer).
- R14 is also known as Link Register (LR). It is used as subroutine return address link register. It is unique except for system mode which shares the same register as User mode.
- R15 is the Program Counter (PC). It holds the address of the instruction being FETCHED (not the one being executed).

# Instruction Execution

Multiple stages are involved in executing an instruction. Example:

- 1) Fetching the instruction code
- 2) Decoding the instruction code
- 3) Executing the instruction code

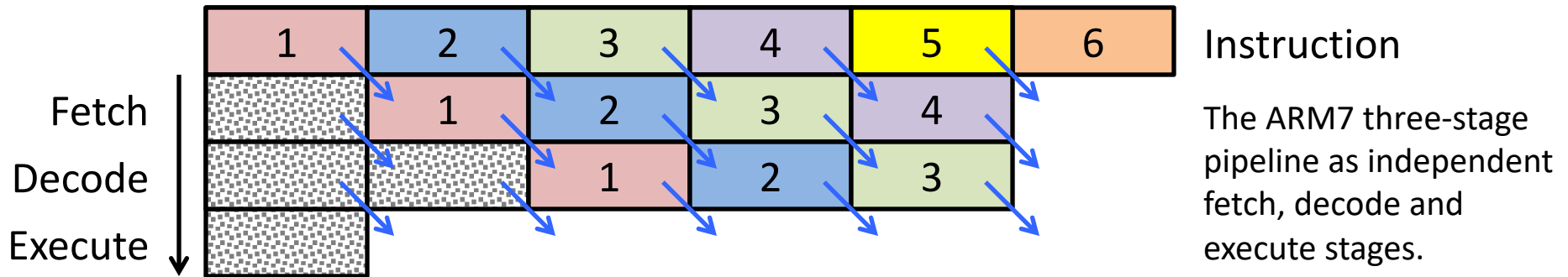
Hence multiple processor clock cycles are needed to execute one single instruction.



# Instruction Pipeline

- Pipeline allows concurrent execution of multiple different instructions
- execution of different stages of multiple instructions at the same time
- During a normal operation
- while one instruction is being executed
- the next instruction is being decoded
- and a third instruction is being fetched from memory
- allows effective throughput to increase to one instruction per clock cycle

# Pipelined Architecture



- **FETCH:** Instruction fetched from memory
- **DECODE :** Decoding of registers used in instruction
- **EXECUTE :**
  - Register(s) read from Register Bank,
  - Shift and ALU operation
  - Write register(s) back to Register Bank

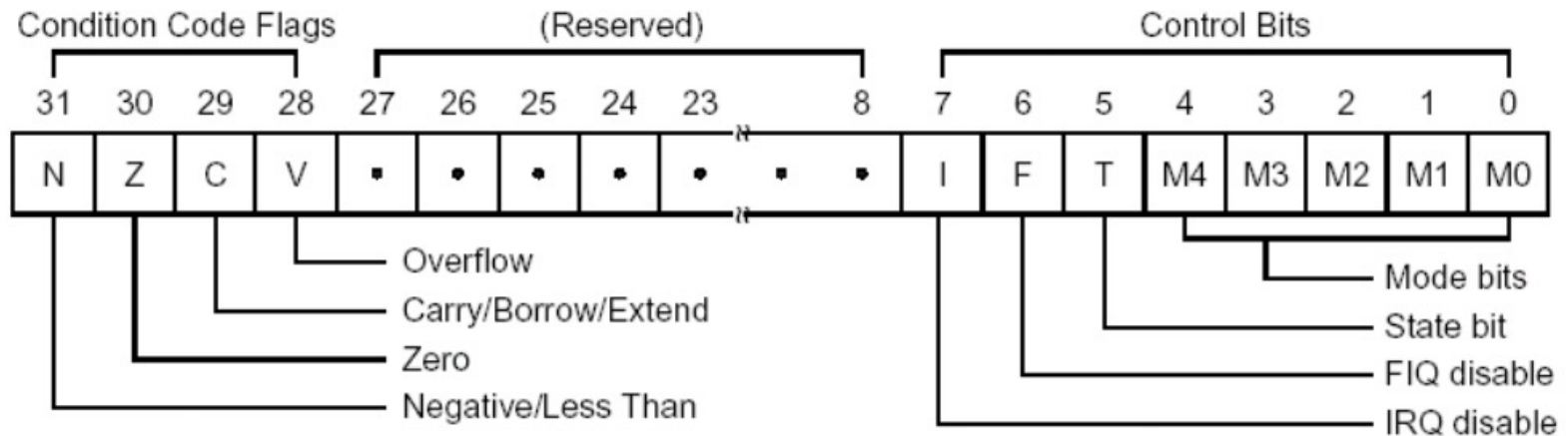


# Current Program Status Register (CPSR)

- CPSR shows the state of the machine.
- It contains condition code flags, interrupt enable flags, the current mode and the current state.
- Each privileged mode (except System mode) has Saved Program Status Register (SPSR) that is used to preserve the value of CPSR when an exception occurs.

# Program Status Register

- Most significant 4 bits are condition code flags
- Least significant 8 bits are control bits



# Control Bits

- I and F bits are interrupt disable bits which disable IRQ interrupts and FIQ interrupts respectively. For example, when  $I = 1$ , no IRQ interrupts are entertained.
- T is the status bits to indicate the state of the machine ( ARM or THUMB).  $T = 1$  implies the machine is currently executing THUMB code. This bit is read only (not writable), you can only change between ARM and THUMB state via a special instruction.

# Mode Bits

- Least significant 5 bits, M[4:0] are the mode bits.

| xPSR[4:0] | Mode       |
|-----------|------------|
| 10000     | User       |
| 10001     | FIQ        |
| 10010     | IRQ        |
| 10011     | Supervisor |
| 10111     | Abort      |
| 11011     | Undefined  |
| 11111     | System     |

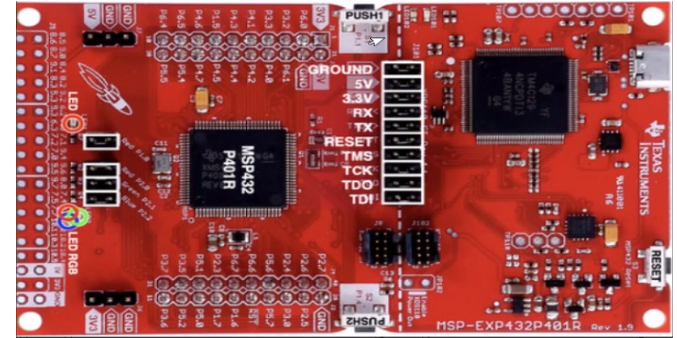
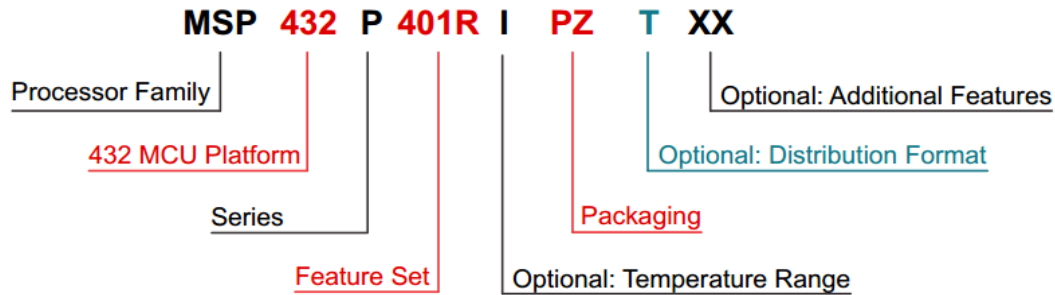
# The Vector Table

- The exception vector table consists of designated addresses in external memory that hold information necessary to handle an exception, an interrupt, or other atypical event such as a reset.
- For example, when an interrupt (IRQ) comes along, the processor will change the PC to 0x18 and fetch the actual ARM Instruction, which is most likely a Branch (B) instruction.

# Exception Vector Table

| Exception Type           | Mode              | Vector Address |
|--------------------------|-------------------|----------------|
| Reset                    | Supervisor (SVC)  | 0x00000000     |
| Undefined Instructions   | Undefined (UNDEF) | 0x00000004     |
| Software Interrupt (SWI) | SVC               | 0x00000008     |
| Prefetch abort           | ABORT             | 0x0000000C     |
| Data abort               | ABORT             | 0x00000010     |
| IRQ                      | IRQ               | 0x00000018     |
| FIQ                      | FIQ               | 0x0000001C     |

# MSP432P401R

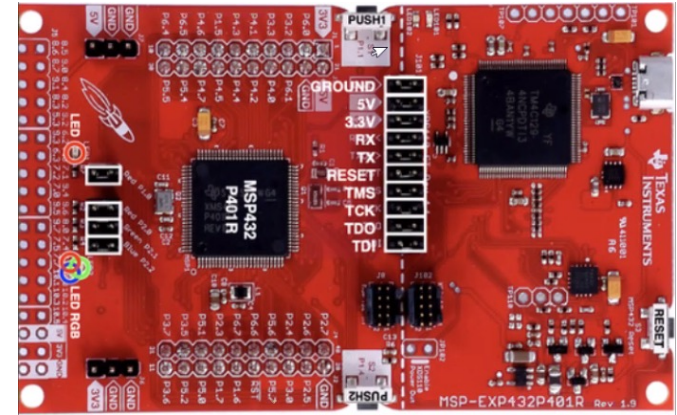


|                                |  |  |                          |  |
|--------------------------------|--|--|--------------------------|--|
| Processor Family               | MSP = Mixed Signal Processor<br>XMS = Experimental Silicon     |  |                          |  |
| 432 MCU Platform               | TI's 32-bit Low-Power Microcontroller Platform                 |  |                          |  |
| Series                         | P = Performance and Low-Power Series                           |  |                          |  |
| Feature Set                    | First Digit<br>4 = Flash based devices<br>up to 48 MHz         | Second Digit<br>0 = General<br>Purpose | Third Digit<br>1 = ADC14 | Fourth Digit<br>R = 256KB of Flash<br>64KB of SRAM<br>M = 128KB of Flash<br>32KB of SRAM |
| Optional:<br>Temperature Range | S = 0°C to 50 °C<br>I = 40 °C to 85 °C<br>T = -40 °C to 105 °C |  |                          |  |
| Packaging                      | PZ = LQFP  |  |                          |  |

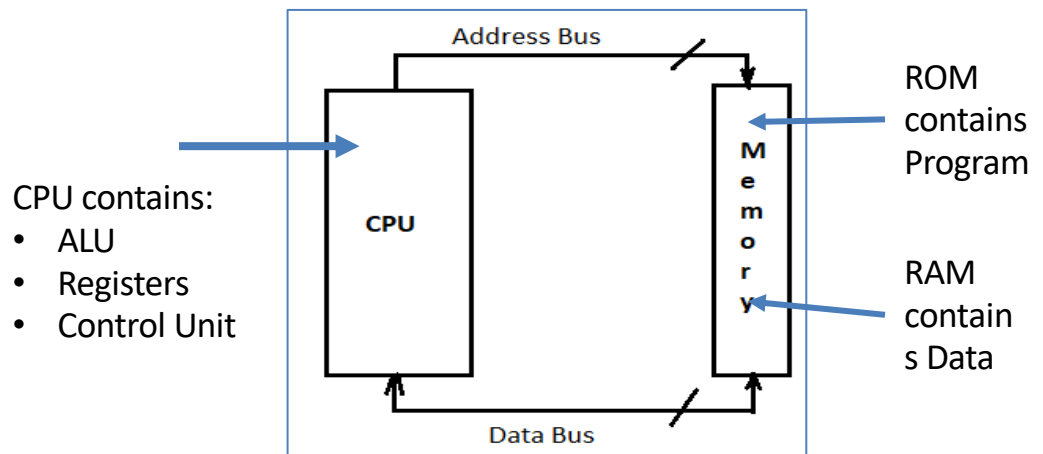
# Microcontroller Architecture

## Main Characteristics

- CPU and Memory are within a single chip
- Connected by Address and Data buses (major buses)
- Address bus: 32 bit wide.
  - Maximum Memory size = 4 GB ( $2^{32} = 2^2 \times 2^{30} = 4\text{GB}$ )
- Data Bus: 32 bit wide
- Working registers: About 16
- Register size: 32 bits



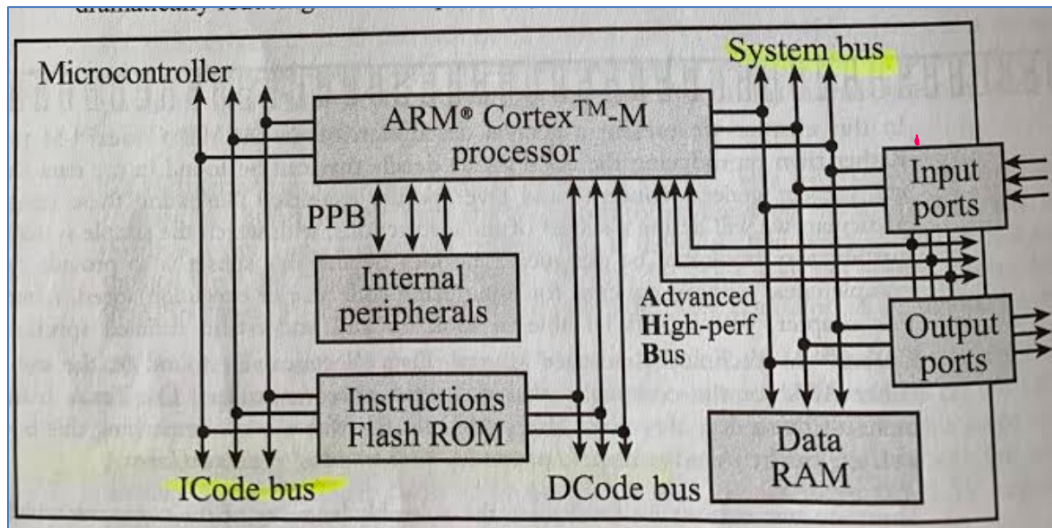
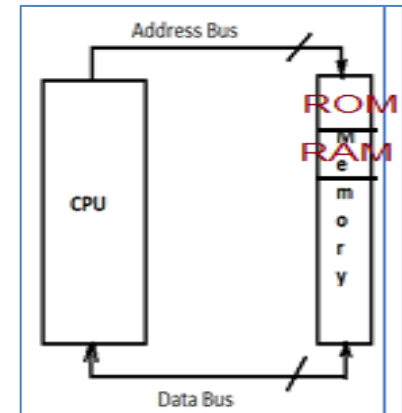
It's a single chip





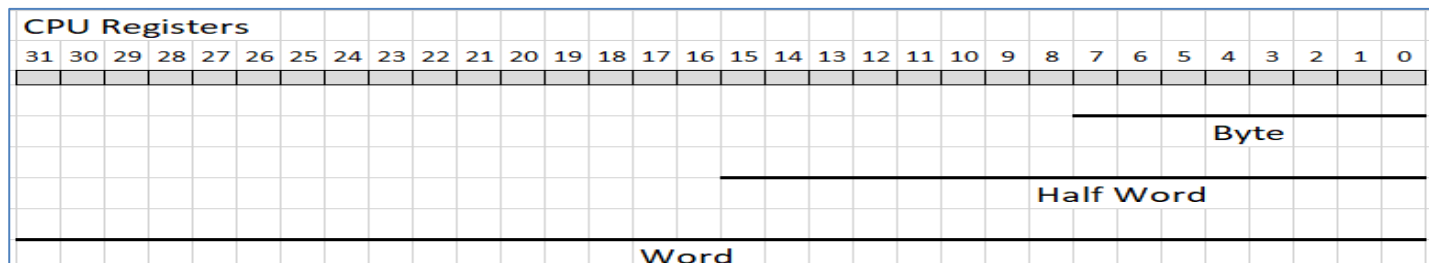
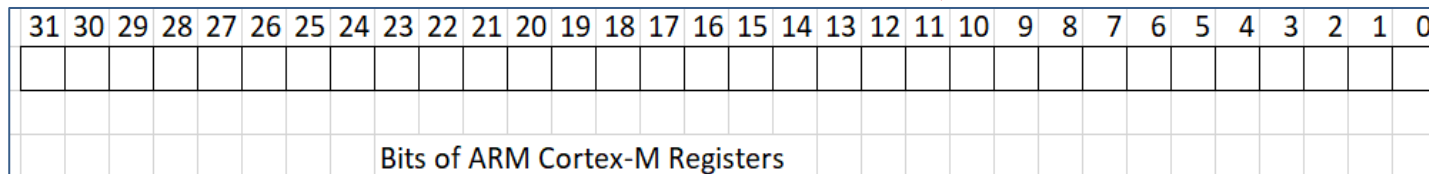
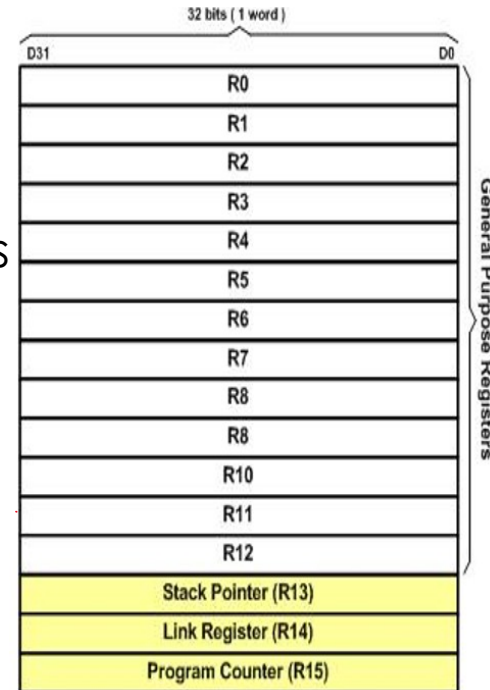
# Cortex-M Architecture (MSP432 LaunchPad)

- ❑ Instructions are fetched from **Flash ROM** using the **ICode Bus**
- ❑ **Data** are exchanged with Memory and I/O devices via the **System Bus**
- ❑ A **second High-Speed I/O Bus** for devices like, USB
- ❑ **DCode** bus for Debugging features
- ❑ Nested Vectored Interrupt Controller (NVIC) communicates with CPU via Private Peripheral Bus (**PPB**)
- ❑ The tight integration of the CPU and NVIC provides fast execution of Interrupt Service Routines (ISRs), reducing interrupt latency
- ❑ **Address bus is not shown**



# Cortex-M Registers

- Register Bank of CPU
- 32-bit wide
- R0- R12 are General-Purpose Registers
- Special-Purpose Registers
  - R13 – Stack Pointer
  - R14 – Link Register
  - R15 – Program Counter



# Cortex-M Registers

- ARM Cortex-M has **3 Status** registers
  - Application Program Status Register (APSR)
  - Interrupt Program Status Register (IPSR)
  - Execution Program Status Register (EPSR)

These status registers can be accessed individually or in combination as **Program Status Register (PSR)**

|                               |    |    |    |    |        |    |          |    |    |    |    |    |    |    |        |    |    |    |    |    |    |                              |   |   |   |   |   |   |   |   |   |
|-------------------------------|----|----|----|----|--------|----|----------|----|----|----|----|----|----|----|--------|----|----|----|----|----|----|------------------------------|---|---|---|---|---|---|---|---|---|
| 31                            | 30 | 29 | 28 | 27 | 26     | 25 | 24       | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16     | 15 | 14 | 13 | 12 | 11 | 10 | 9                            | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| N                             | Z  | C  | V  | Q  | ICI/IT | T  | Reserved |    |    |    |    |    |    |    | ICI/IT |    |    |    |    |    |    | Interrupt Service Routine #s |   |   |   |   |   |   |   |   |   |
|                               |    |    |    |    |        |    |          |    |    |    |    |    |    |    |        |    |    |    |    |    |    |                              |   |   |   |   |   |   |   |   |   |
| Program Status Register (PSR) |    |    |    |    |        |    |          |    |    |    |    |    |    |    |        |    |    |    |    |    |    |                              |   |   |   |   |   |   |   |   |   |



# Cortex-M Registers

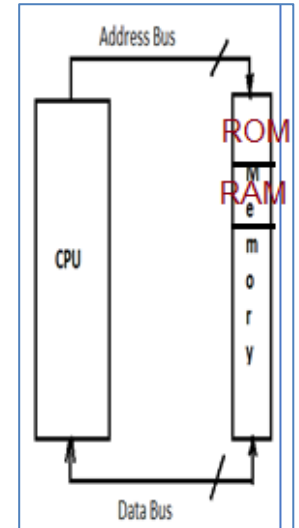
|                               |    |    |    |    |        |    |          |    |    |    |    |    |    |    |        |    |    |    |    |                              |    |   |   |   |   |   |   |   |   |   |   |
|-------------------------------|----|----|----|----|--------|----|----------|----|----|----|----|----|----|----|--------|----|----|----|----|------------------------------|----|---|---|---|---|---|---|---|---|---|---|
| 31                            | 30 | 29 | 28 | 27 | 26     | 25 | 24       | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16     | 15 | 14 | 13 | 12 | 11                           | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| N                             | Z  | C  | V  | Q  | ICI/IT | T  | Reserved |    |    |    |    |    |    |    | ICI/IT |    |    |    |    | Interrupt Service Routine #s |    |   |   |   |   |   |   |   |   |   |   |
| Program Status Register (PSR) |    |    |    |    |        |    |          |    |    |    |    |    |    |    |        |    |    |    |    |                              |    |   |   |   |   |   |   |   |   |   |   |

- N, Z, C, V, and Q bits give information on the most recently ALU operation
  - N bit is set (1) if the result is **Negative**
  - Z bit is set (1), if the result is **Zero**
  - C bit is set (1), if there is **unsigned overflow**
  - V bit is set (1), if there is **signed overflow**
  - Q bit is set (1), if saturation occurred (not for this course)
- T bit is set (1), indicating the ARM Cortex-M is executing Thumb instruction
- The IPSR Number (8 - 0) bits indicates which interrupt, if any, the processor is handling now



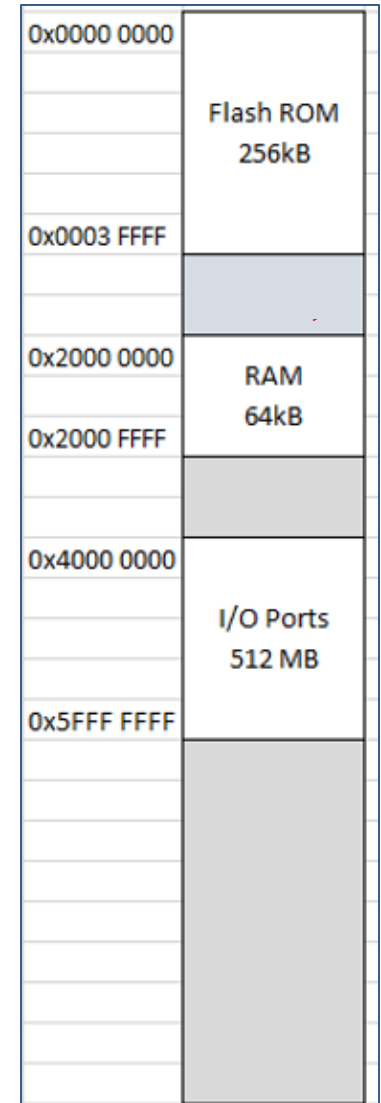
# Cortex-M Reset

- First Reset occurs with Power ON
- And when the Reset Button is pressed
- After Reset
  - Processor runs in Thread (privileged) mode
  - SP points to Flash ROM address, 0x0000 0000
  - PC points to Flash ROM address, 0x0000 0004
- Instructions are half-word aligned. Thus, last bit of PC must be a 0
  - LR points to the last address of memory, 0xFFFF FFFF



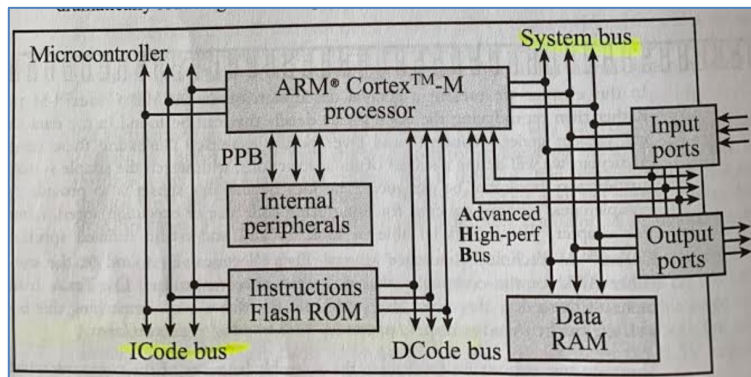
# MSP432 Memory Organization

- Microcontrollers within the same family differs by the
  - Amount of memory and by the types of I/O modules
  - Variations in sizes are for the ending addresses of ROM and RAM
- Range of Flash ROM: **0x0000 0000 to 0x0003 FFFF**. Total amount is **256kB**
- Range of RAM: **0x2000 0000 to 0x2000 FFFF**. Total amount is **64kB**
- Range of I/O Ports: **0x4000 0000 to 0x5FFF FFFF**. Total amount is **512 MB (0.536GB)**



# MSP432 Memory Organization

- Having multiple buses means the processor can perform multiple tasks in parallel.
- The following are some of the tasks that can occur in parallel
- **ICode Bus** – Fetch Instruction from the Flash ROM
- **DCode Bus** – Read Debugging features for Flash ROM
- **System Bus** – Read/Write data from/to Memory or I/O
- **PPB** - Read/Write data from Internal Peripherals, like NVIC
- **AHB** - Read/Write data from High-Speed I/O and parallel Ports



|             |                     |
|-------------|---------------------|
| 0x0000 0000 | Flash ROM<br>256kB  |
|             |                     |
|             |                     |
| 0x0003 FFFF |                     |
|             |                     |
|             |                     |
| 0x2000 0000 | RAM<br>64kB         |
|             |                     |
|             |                     |
| 0x2000 FFFF |                     |
|             |                     |
|             |                     |
| 0x4000 0000 | I/O Ports<br>512 MB |
|             |                     |
|             |                     |
| 0x5FFF FFFF |                     |
|             |                     |
|             |                     |