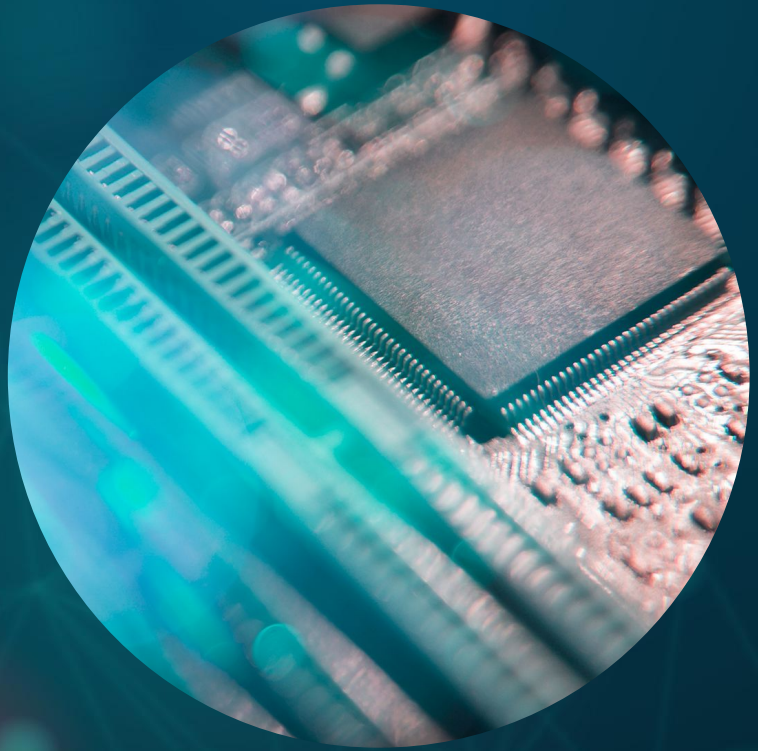
ARM Based Microcontroller

ARM Architecture

Lecture 1



Advanced RISC Machines

Introduction 01

ARM Architecture


ARM

ARM

1990

ARM was formed in as Advanced RISC Machines Ltd., a joint venture of Apple Computer, Acorn Computer Group, and VLSI Technology.

1991

ARM introduced the ARM6 processor family to meet Apple requirement for its product "Personal Digital Assistant" called Newton.
Unfortunately, the Newton was not a great success and so Robin Saxby, ARM's CEO, decided to grow the business by pursuing what we now call intellectual property "IP" business model.

The ARM processor was licensed to many semiconductor companies for an upfront license fee and then royalties on production silicon. This effectively incentivized ARM to help its partner get to high volume shipments as quickly as possible.

ARM

1993

Nokia approached TI to produce a chipset for an upcoming GSM mobile phone and TI proposed an ARM7 based system to meet Nokia's performance and power requirements. Unfortunately Nokia rejected the proposal !
History

ARM came up with a radical idea to create a subset of the ARM instruction set that required just 16 bits per instruction. This improved the code density by about 35% and brought the memory footprint down to a size comparable with 16 bit microcontrollers.

The first ARM-powered GSM phone was the hugely popular Nokia 6110 and the ARM7TDMI.

ARM

1997

ARM had grown to become a £27m business with a net income of £3m !
ARM then decided to build software-based systems on a single chip, the so-called system-on-chip, or SoC.

2001

ARM9 was announced. It was fully synthesizable with a 5 stage pipeline and a proper MMU, as well as hardware support for Java acceleration and some DSP extension.

2002

ARM11 families had extended the capability of the ARM architecture in the direction of higher performance with the introduction of multi-processing, SIMD multimedia instructions, DSP capability, Java acceleration etc

ARM

2005

The ARM Cortex..

01

CORTEX-A

OPTION A

Application Processors for full OS and Open Application Platforms

02

CORTEX-R

OPTION B

Embedded Processors for real time signal processing and control applications

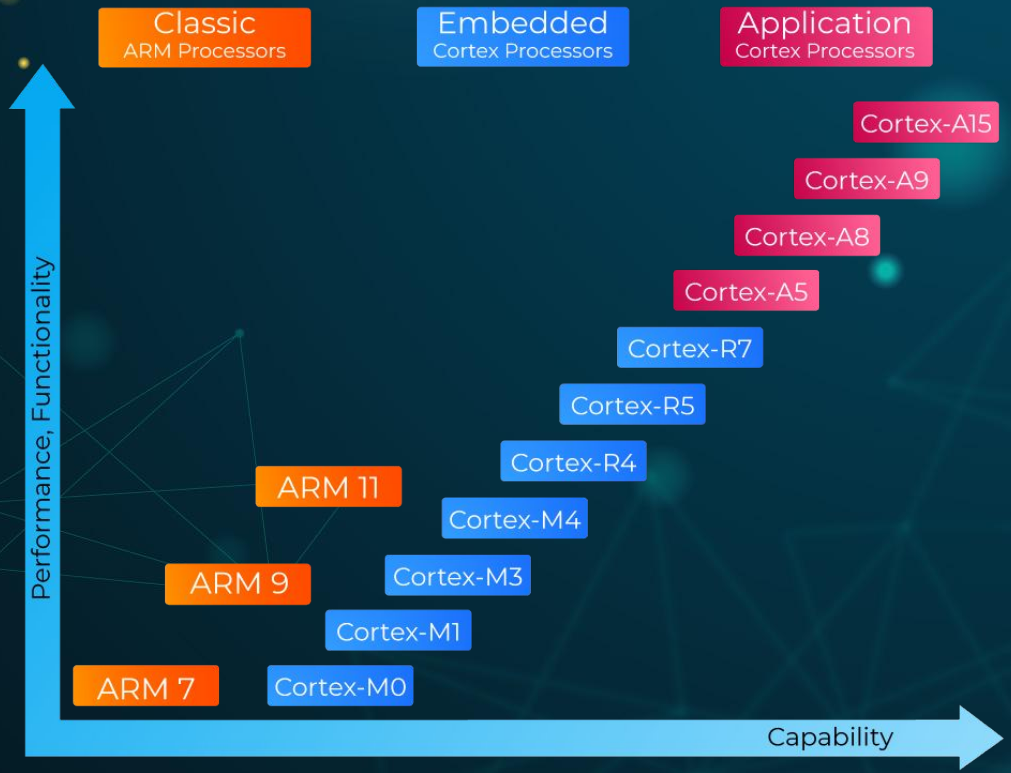
03

CORTEX-M

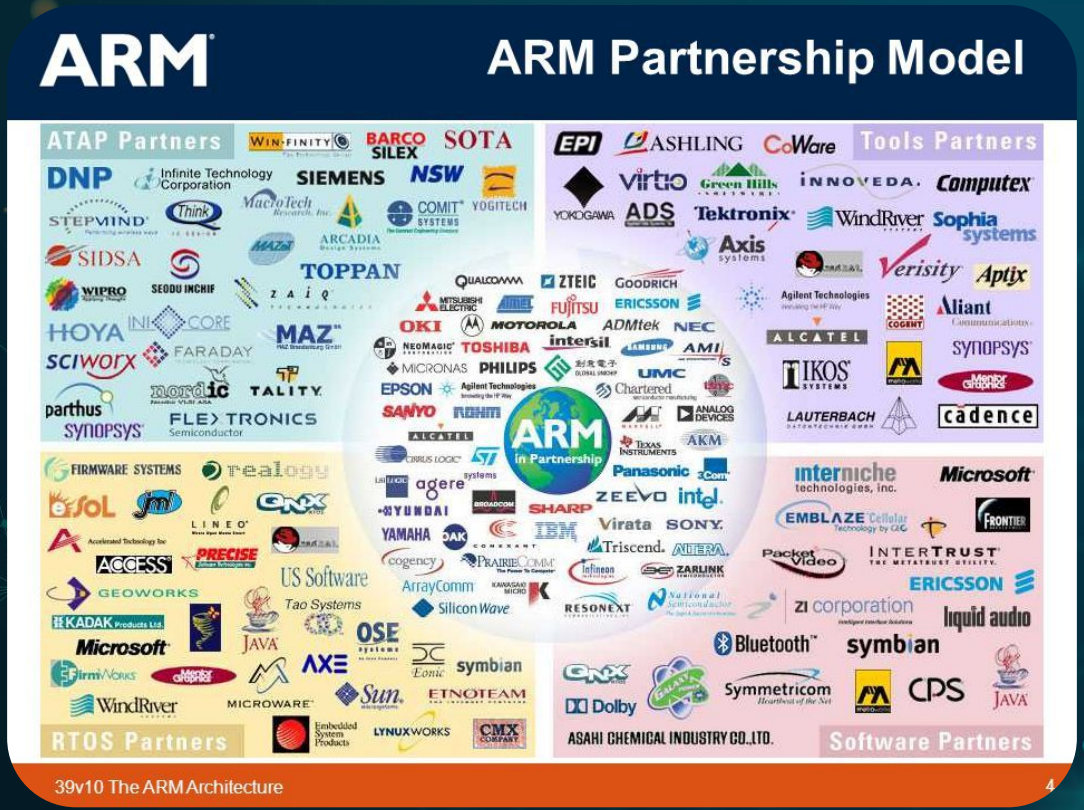
OPTION C

Microcontroller Oriented Processors

ARM Processor Roadmap



ARM Silicon Partners

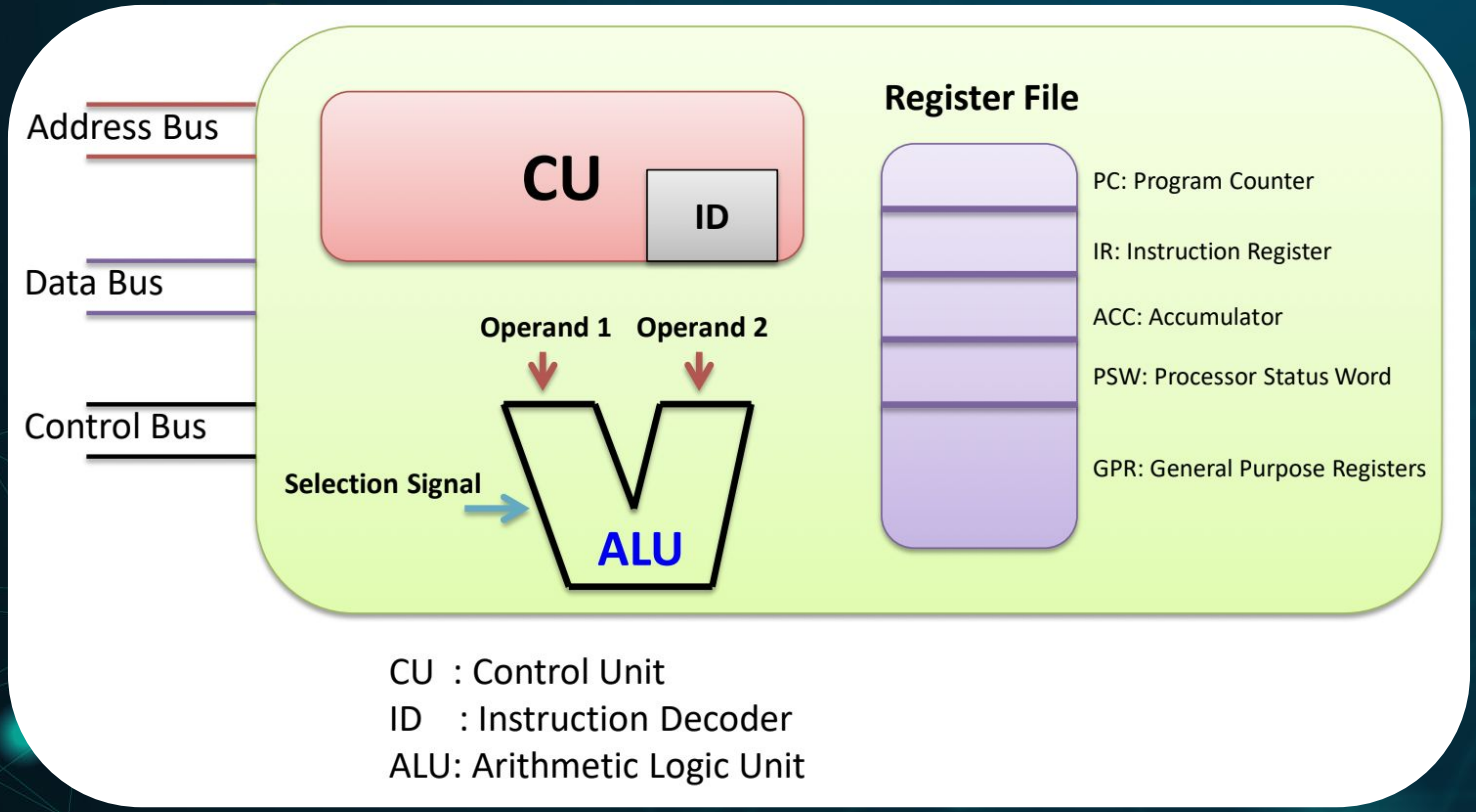


ARM Major Characteristic

Architecture 01

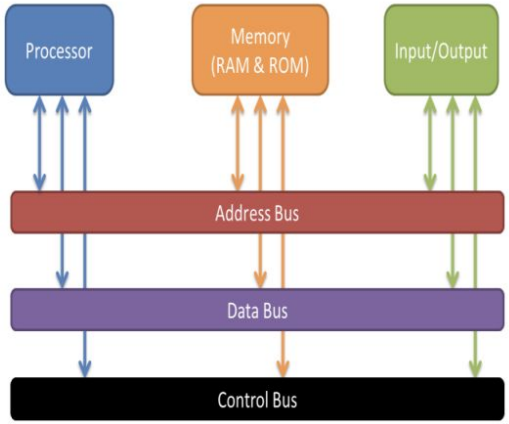
ARM Architecture

General Purpose Architecture



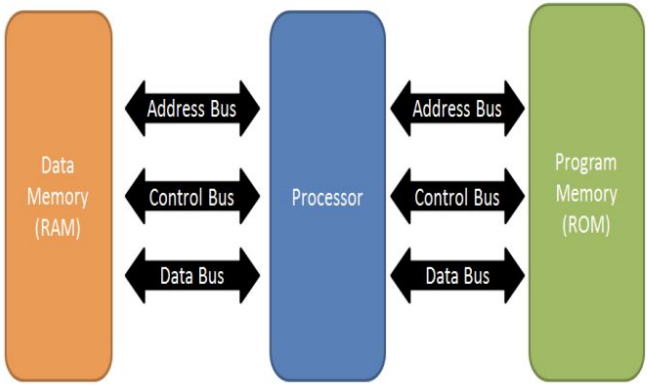
General Purpose Architecture

1- Von Numann



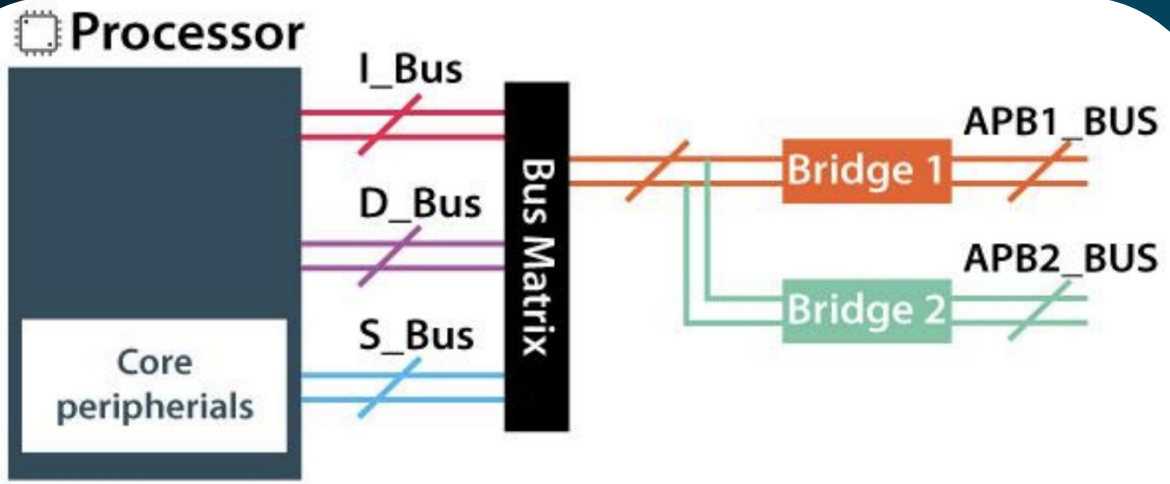
- Simple in Design
- The code is executed serially

2- Harvard Architecture



- Complex in design
- The code is executed in parallel

ARM Hybrid Architecture



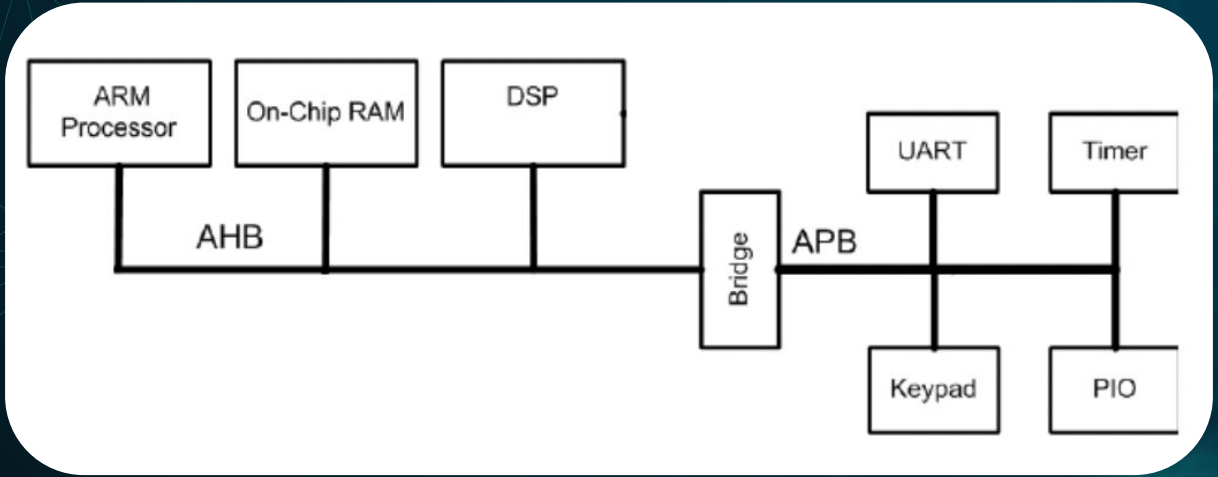
- Complex in design
- Easy to software
- The code sees the memory system as if it is one

- Bus matrix function is mapping the addresses written by software to physical addresses by hardware circuit.

ARM Hybrid Architecture

AMBA Bus

Advanced Microcontroller Bus Architecture
AMBA is a description or a documentation of how to connect the external peripherals. Because of success of AMBA documentation, the other microcontroller companies use it into their microcontroller products.



ARM Hybrid Architecture

AHB

stands for Advanced High Performance Bus

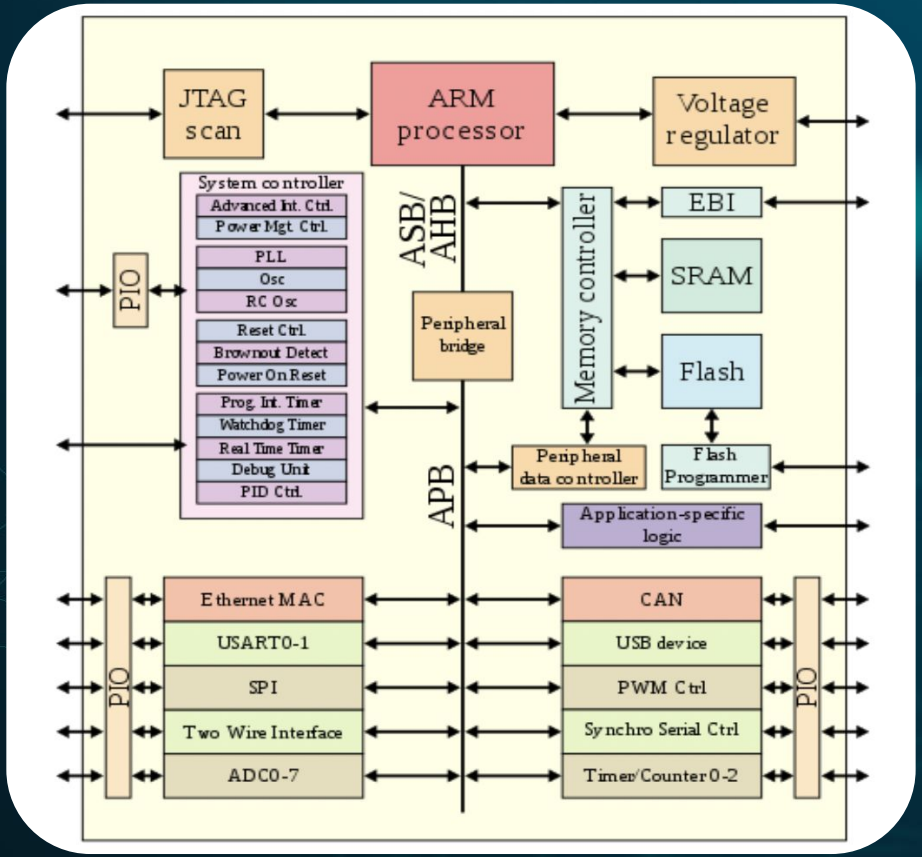
- High Performance
- Full Duplex
- Support Pipelining
- MultiMaster operation
- Complex in design
- Max speed is 72 MHz .

APB

stands for Advanced Peripheral Bus

- Low Power
- No Pipelining
- Simple in design
- Used for connecting peripherals
- Max speed is 36 MHz

ARM Hybrid Architecture



Connection to Microcontroller

controller Bus Architecture
AMBA is a description or a documentation of how to connect the external peripherals.
Because of success There are two types of connections: Forward Connection and Reverse Connection.
Forward connection is also called “Active High or current Source”.
Reverse connection is also called “Active Low or Current Sink”.

Source Current of PORT A,B equals Sink Current.
Typical Current is “8 mA”.
Max Current is “20 mA”.
PORT C at STM32f104 is run as sink current only at “3 mA” maximum.
f AMBA documentation, the other microcontroller companies use it into their microcontroller products.

Register Bank in ARM Microcontroller

General Purpose Registers (GPRs)

There are 13 registers, each size is 4 Bytes (32 bits).

Processor uses them in his operations like that: if there is x variable stored in RAM, then the operation is adding 10 to x, the processor will load the x value into its GPRs then adding 10, then storing the new value to the location address of x. This operation is called “Load-Modify-Store” or “Read-Modify-Write”.
GPRs also are used for storing value with keyword “register”.

Register Bank in ARM Microcontroller

Stack Pointer (r13)

It is a register stores the RAM address value which it points immediately

It is considered as a gate to two registers called MSP “Main Stack Pointer” and PSP “Process Stack Pointer”.
The reason to be existed two register is sometimes user needs to divide stack as example within using RTOS “Real Time Operating System”.
To switch from PSP to MSP and vice versa, Control Register can do this. We will mention it in details later.

Register Bank in ARM Microcontroller

Link Register (r14)
It is a register is used to save PC "Program Counter" value within context switching.

The PC value is stored in the Link Register before context switching. After finishing of function execution, the old PC value stored in Link Register will be loaded again to PC

register, but if fun ( ) contains another function -that means nested context switching- the PC value will be stored at the first context switching but the next context switching,

PC value will be stored in stack as AVR.

Register Bank in ARM Microcontroller

Programmer Counter Register (r15)

It is a register stores the ROM address value which it will point next.

user can change the value of PC by software by:

```
store    pc   r0    (dereference to address)
store    pc   #10   (assigning value directly)
```

user can change the value of PC by hardware by: Link Register

Register Bank in ARM Microcontroller

Special Function Register

A. Program Status Register:

It contains arithmetic flags as (Zero flag, Overflow flag, Carry flag, Half flag, ...) and some logic flags.
It is used to know the current executing interrupt.

Register Bank in ARM Microcontroller

Special Function Register

B. Interrupt Mask:

Premask (1_bit Register):
If this bit equals one, all interrupts will be disabled except NMI "Non-Maskable Interrupt", Reset, and Hard Fault interrupts.

Fault Mask (1_bit Register):
If this bit equals one, hard fault interrupt will be disabled.

Base PRI:
It is used to disable all interrupts that is lower than the assigned into this register.

Register Bank in ARM Microcontroller

Special Function Register

C. Control Register(2 bits register):

control [1]:
This bit control on which stack pointer is enabled PSP or MSP.

control [0]:
This bit controls on which mode is enabled Privileged or User.

Operations Mode

Access Level:

Privileged Level:

At this mode, Processor can access any thing at microcontroller.

User Level:

At this mode, Processor is prohibited from accessing somethings at microcontroller like MPU “Memory Protection Unit”.

Processor Mode:

Thread Mode:

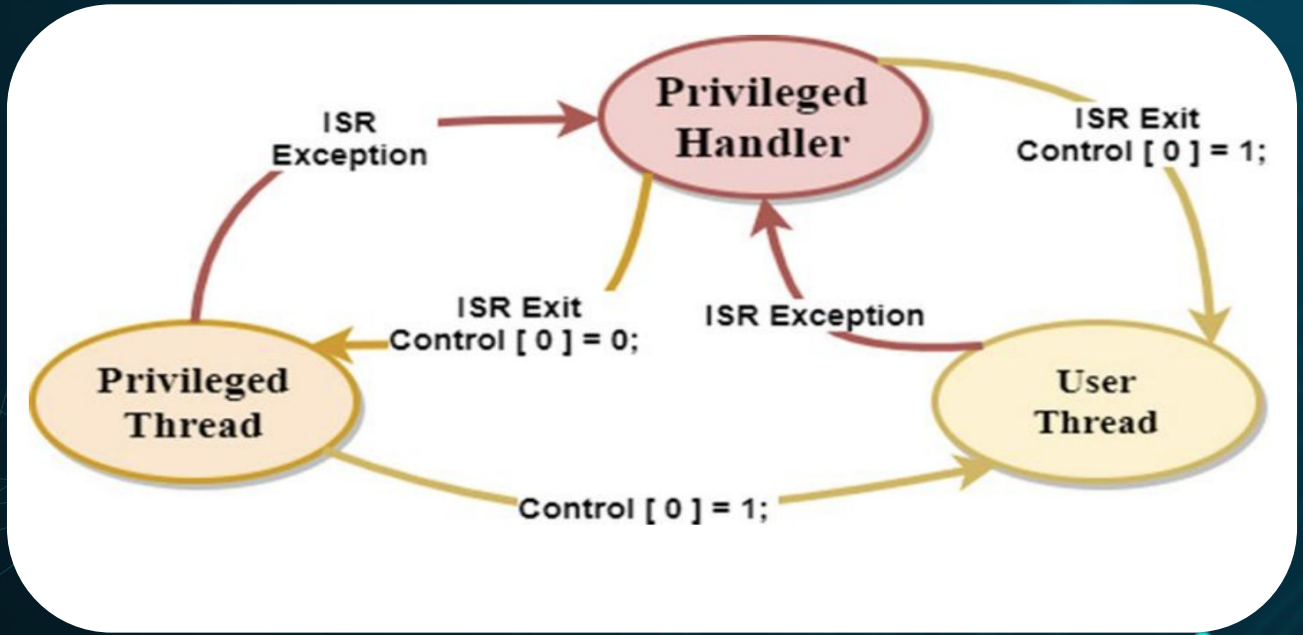
At this mode, program runs at normal code. Processor can be Privileged or user at this mode.

Handler Mode:

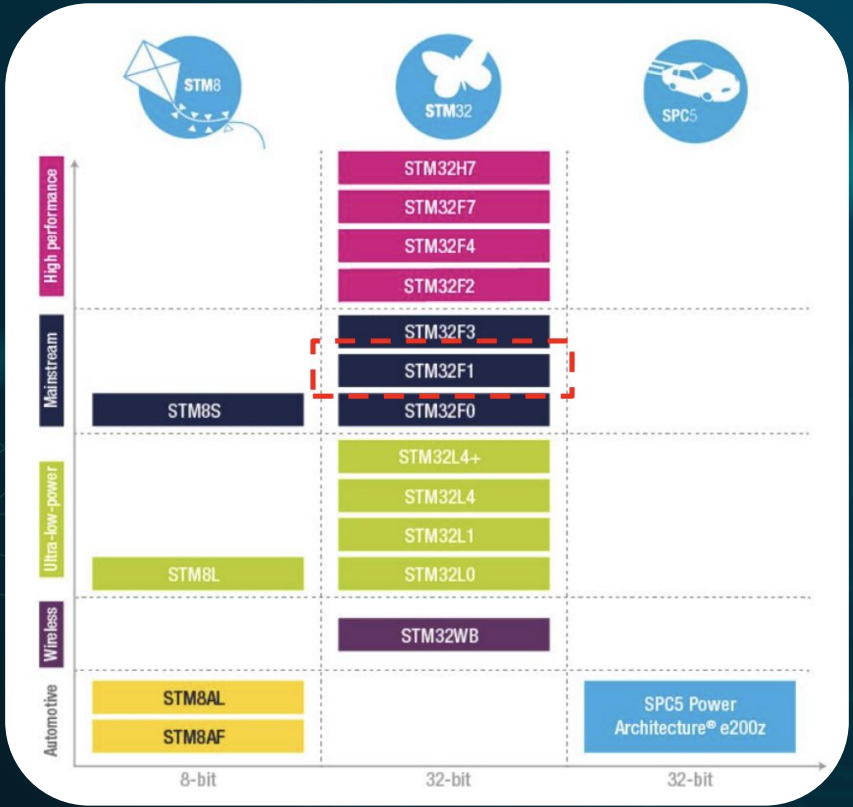
At this mode, program runs at interrupt level. Processor can be Privileged only.

Operations Mode

Conversion from Privileged to User and vice versa:



ST Production Lines



STM32F401CC

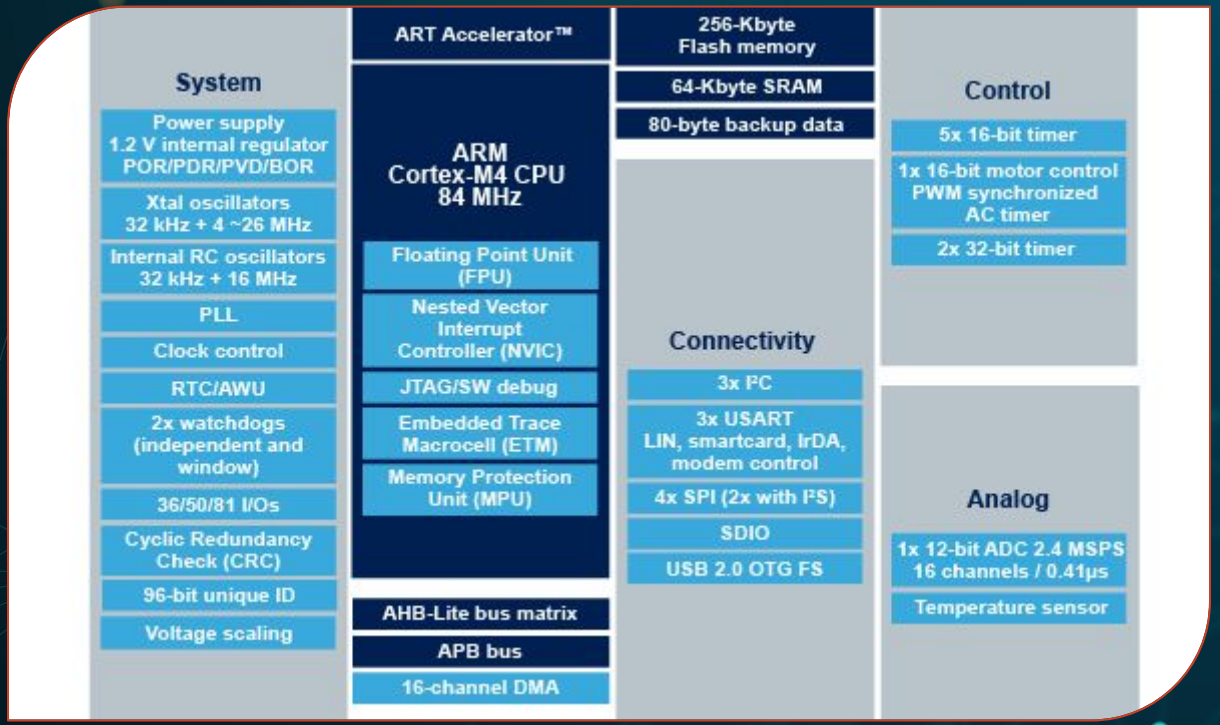
The STM32F401xB/STM32F401xC devices are based on the high-performance Arm® Cortex® -M4 32-bit RISC core operating at a frequency of up to 84 MHz. The Cortex®-M4 core features a Floating point unit (FPU) single precision which supports all Arm single-precision data-processing instructions and data types. It also implements a full set of DSP instructions and a memory protection unit (MPU) which enhances application security. The STM32F401xB/STM32F401xC incorporate high-speed embedded memories (up to 256 Kbytes of Flash memory, up to 64 Kbytes of SRAM), and an extensive range of enhanced I/Os and peripherals connected to two APB buses, two AHB buses and a 32-bit multi-AHB bus matrix.

All devices offer one 12-bit ADC, a low-power RTC, six general-purpose 16-bit timers including one PWM timer for motor control, two general-purpose 32-bit timers. They also feature standard and advanced communication interfaces.

The STM32F401xB/STM32F401xC operate in the - 40 to + 125 °C temperature range from a 1.7 (PDR OFF) to 3.6 V power supply. A comprehensive set of power-saving mode allows the design of low-power applications.

These features make the STM32F401xB/STM32F401xC microcontrollers suitable for a wide range of applications.

STM32F401CC



STM32F401CC Specifications

Dynamic efficiency line with BAM (batch acquisition mode)

- 1.7 V to 3.6 V power supply
- -40 °C to 85/105/125 °C temperature range

Core: Arm® 32-bit Cortex®-M4 CPU with FPU, Adaptive real-time accelerator (ART Accelerator™) allowing 0-wait state execution from Flash memory, frequency up to 84 MHz, memory protection unit, 105 DMIPS/1.25 DMIPS/MHz (Dhrystone 2.1), and DSP instructions

Memories Up to 256 Kbytes of Flash memory, 512 bytes of OTP memory, Up to 64 Kbytes of SRAM

Clock, reset and supply management

- 1.7 V to 3.6 V application supply and I/Os
- POR, PDR, PVD and BOR
- 4-to-26 MHz crystal oscillator
- Internal 16 MHz factory-trimmed RC
- 32 kHz oscillator for RTC with calibration
- Internal 32 kHz RC with calibration

STM32F401CC Specifications

Power consumption

- Run: 128 µA/MHz (peripheral off)
- Stop (Flash in Stop mode, fast wakeup time): 42 µA typ @ 25 °C;65 µA max @25 °C
- Stop (Flash in Deep power down mode, slow wakeup time): down to 10 µA typ@ 25 °C; 28 µA max @25 °C
- Standby: 2.4 µA @25 °C / 1.7 V without RTC; 12 µA @85 °C @1.7 V
- VBAT supply for RTC: 1 µA @25 °C

1×12-bit, 2.4 MSPS A/D converter: up to 16 channels

General-purpose DMA: 16-stream DMA controllers with FIFOs and burst support

Up to 11 timers: up to six 16-bit, two 32-bit timers up to 84 MHz, each with up to 4 IC/OC/PWM or pulse counter and quadrature (incremental) encoder input, two watchdog timers (independent and window) and a SysTick timer

Debug mode

- Serial wire debug (SWD) & JTAG interfaces
- Cortex®-M4 Embedded Trace Macrocell™

Up to 81 I/O ports with interrupt capability

- All IO ports 5 V tolerant
- Up to 78 fast I/Os up to 42 MHz

STM32F401CC Specifications

Up to 11 communication interfaces

- Up to 3 × I2C interfaces (1Mbit/s, SMBus/PMBus)
- Up to 3 USARTs (2 x 10.5 Mbit/s, 1 x 5.25 Mbit/s), ISO 7816 interface, LIN, IrDA, modem control)
- Up to 4 SPIs (up to 42 Mbits/s at fCPU= 84 MHz), SPI2 and SPI3 with muxed full-duplex I2S to achieve audio class accuracy via internal audio PLL or external clock
- SDIO interface

Advanced connectivity

- USB 2.0 full-speed device/host/OTG controller with on-chip PHY

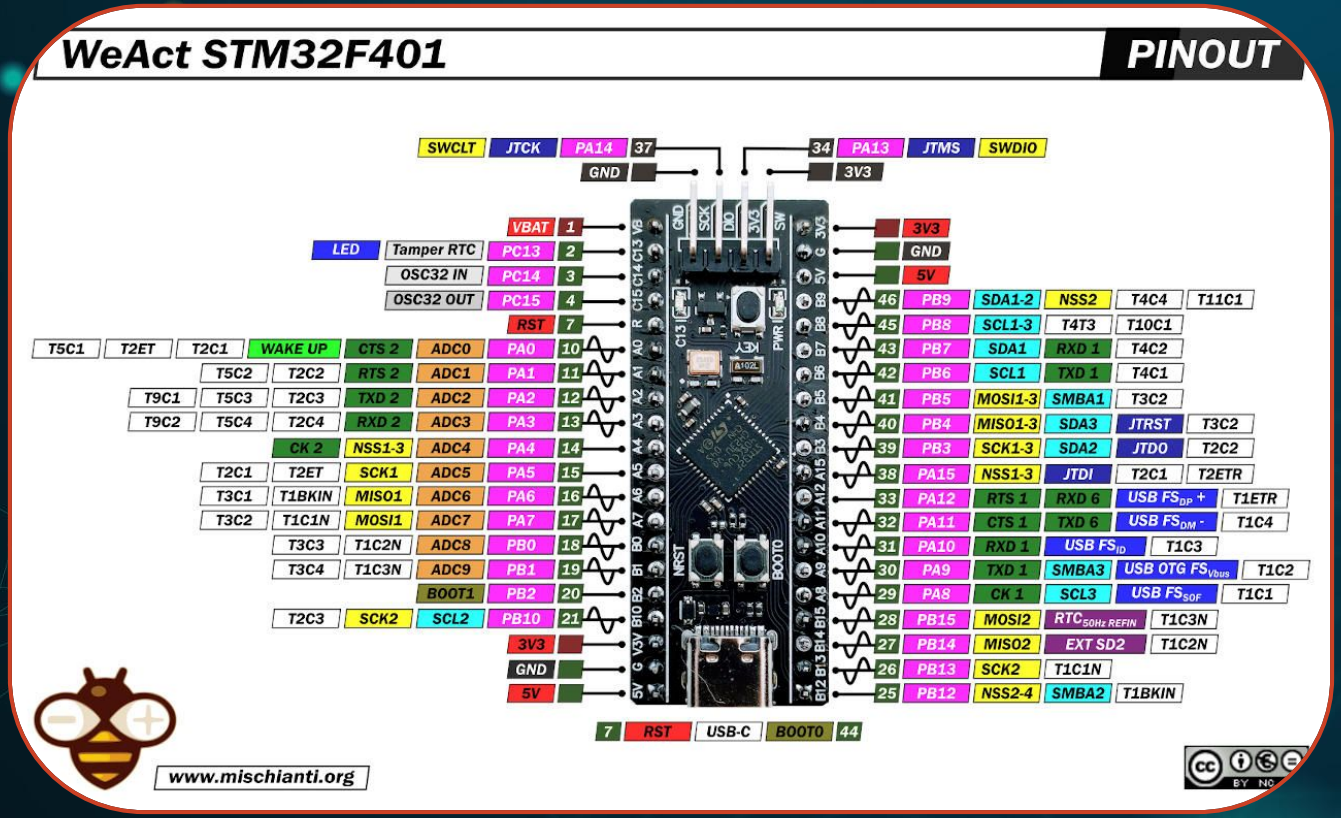
CRC calculation unit

96-bit unique ID

RTC: subsecond accuracy, hardware calendar

All packages are ECOPACK2

Development Kit



Bus Connections

Boundary address	Peripheral	Bus
0x4002 3000 - 0x4002 33FF	CRC	AHB
0x4002 2400 - 0x4002 2FFF	Reserved	
0x4002 2000 - 0x4002 23FF	Flash memory interface	
0x4002 1400 - 0x4002 1FFF	Reserved	
0x4002 1000 - 0x4002 13FF	Reset and clock control RCC	
0x4002 0400 - 0x4002 0FFF	Reserved	
0x4002 0000 - 0x4002 03FF	DMA1	
0x4001 4C00 - 0x4001 FFFF	Reserved	APB2
0x4001 4800 - 0x4001 4BFF	TIM17 timer	
0x4001 4400 - 0x4001 47FF	TIM16 timer	
0x4001 4000 - 0x4001 43FF	TIM15 timer	
0x4001 3C00 - 0x4001 3FFF	Reserved	
0x4001 3800 - 0x4001 3BFF	USART1	
0x4001 3400 - 0x4001 37FF	Reserved	
0x4001 3000 - 0x4001 33FF	SPI1	
0x4001 2C00 - 0x4001 2FFF	TIM1 timer	
0x4001 2800 - 0x4001 2BFF	Reserved	
0x4001 2400 - 0x4001 27FF	ADC1	
0x4001 1C00 - 0x4001 23FF	Reserved	
0x4001 1800 - 0x4001 1BFF	GPIO Port E	
0x4001 1400 - 0x4001 17FF	GPIO Port D	
0x4001 1000 - 0x4001 13FF	GPIO Port C	
0x4001 0C00 - 0x4001 0FFF	GPIO Port B	
0x4001 0800 - 0x4001 0BFF	GPIO Port A	
0x4001 0400 - 0x4001 07FF	EXTI	
0x4001 0000 - 0x4001 03FF	AFIO	

Boundary address	Peripheral	Bus
0x4000 7C00 - 0x4000 FFFF	Reserved	APB1
0x4000 7800 - 0x4000 7BFF	CEC	
0x4000 7400 - 0x4000 77FF	DAC	
0x4000 7000 - 0x4000 73FF	Power control PWR	
0x4000 6C00 - 0x4000 6FFF	Backup registers (BKP)	
0x4000 5C00 - 0x4000 6BFF	Reserved	
0x4000 5800 - 0x4000 5BFF	I2C2	
0x4000 5400 - 0x4000 57FF	I2C1	
0x4000 4C00 - 0x4000 53FF	Reserved	
0x4000 4800 - 0x4000 4BFF	USART3	
0x4000 4400 - 0x4000 47FF	USART2	
0x4000 3C00 - 0x4000 3FFF	Reserved	
0x4000 3800 - 0x4000 3BFF	SPI2	
0x4000 3400 - 0x4000 37FF	Reserved	
0x4000 3000 - 0x4000 33FF	Independent watchdog (IWDG)	
0x4000 2C00 - 0x4000 2FFF	Window watchdog (WWDG)	
0x4000 2800 - 0x4000 2BFF	RTC	
0x4000 1800 - 0x4000 27FF	Reserved	
0x4000 1400 - 0x4000 17FF	TIM7 timer	
0x4000 1000 - 0x4000 13FF	TIM6 timer	
0x4000 0C00 - 0x4000 0FFF	Reserved	
0x4000 0800 - 0x4000 0BFF	TIM4 timer	
0x4000 0400 - 0x4000 07FF	TIM3 timer	
0x4000 0000 - 0x4000 03FF	TIM2 timer	

STM32
Is AWESOME

THANKS!

Do you have any questions?

www.imtschool.com

www.facebook.com/imaketechnologyschool/