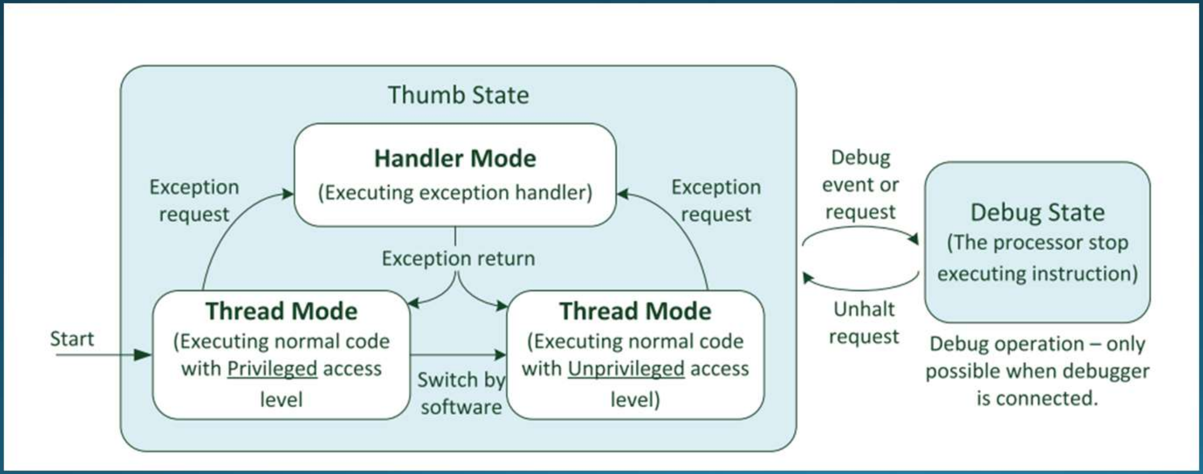
Operation Modes

The Cortex – M3 and Cortex – M4 processors have two Operation modes.

1) **Handler Mode** : When executing an exception handler such as an Interrupt Service Routine(ISR). When in handler mode, the processor always has privileged access level.

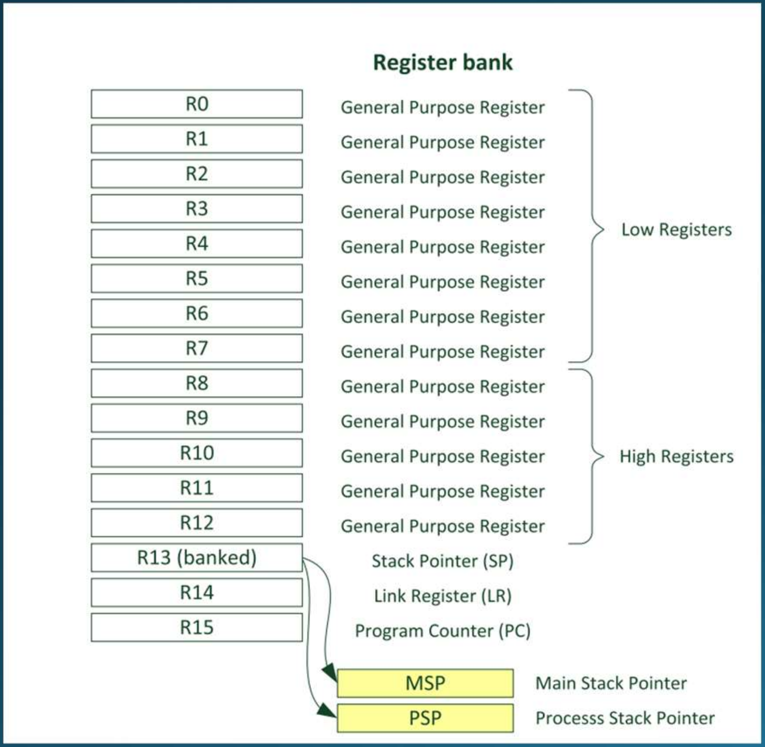
2) **Thread Mode** : When executing normal application code, the processor can be either in privileged access level or unprivileged access level. This is controlled by a special register called "CONTROL."

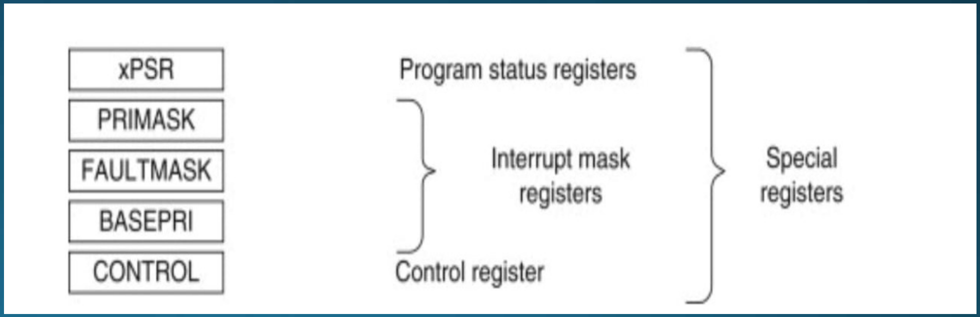


Thumb State
Handler Mode
(Executing exception handler)
Exception request
Exception request
Exception return
Start
Thread Mode
(Executing normal code with Privileged access level
Switch by software
Thread Mode
(Executing normal code with Unprivileged access level)
Debug event or request
Unhalt request
Debug State
(The processor stop executing instruction)
Debug operation – only possible when debugger is connected.

Registers

Most of the registers are grouped in a unit called the register bank.



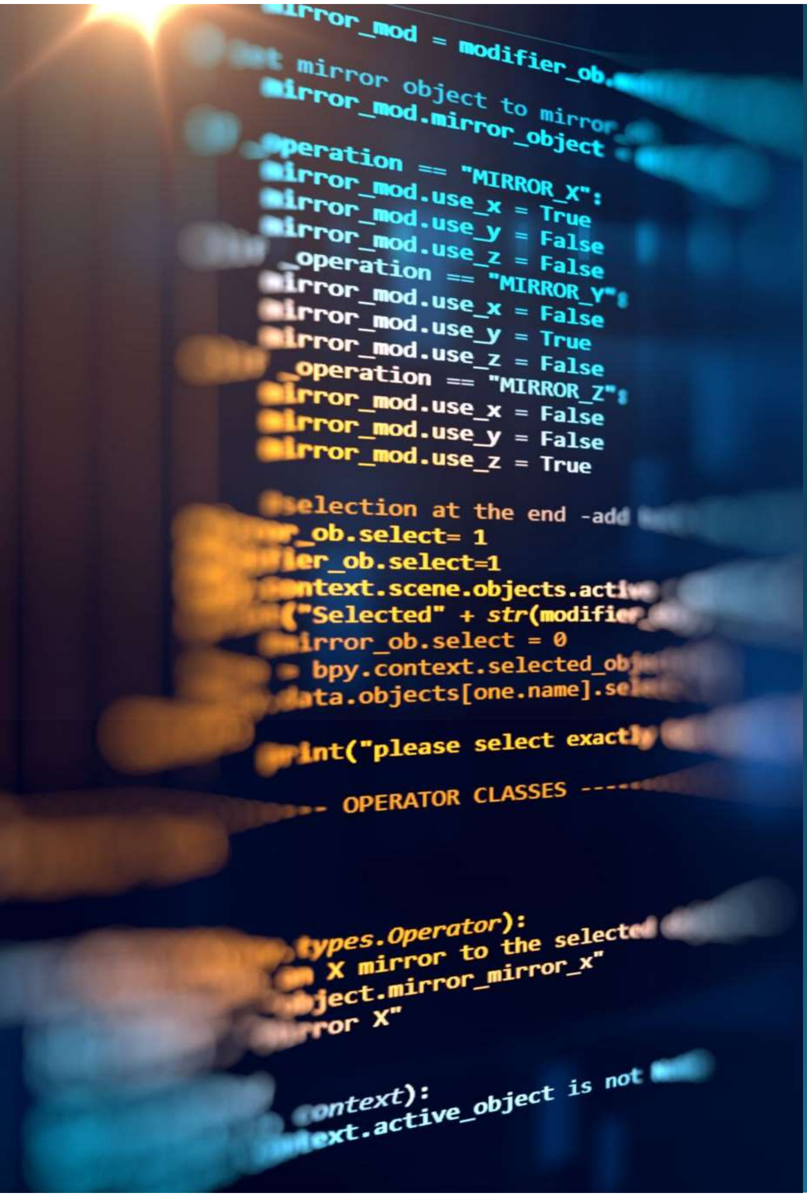


R13, stack pointer (SP)

R13 is the Stack Pointer. It is used for accessing the stack memory via PUSH and POP operations. Physically there are two different Stack Pointers: the Main Stack Pointer (MSP) is the default Stack Pointer. It is selected after reset, or when the processor is in Handler Mode. The other Stack Pointer is called the Process Stack Pointer (PSP). The PSP can only be used in Thread Mode. The selection of Stack Pointer is determined by a special register called CONTROL

R14, Link register (LR)

R14 is also called the Link Register (LR). This is used for holding the return address when calling a function or subroutine. At the end of the function or subroutine, the program control can return to the calling program and resume by loading the value of LR into the Program Counter (PC).



R15, program counter (PC)

- R15 is the Program Counter (PC). It is readable and writeable: a read returns the current instruction address plus 4 (this is due to the pipeline nature of the design, and compatibility requirement with the ARM7TDMI processor). Writing to PC (e.g., using data transfer/processing instructions) causes a branch operation.

(Allowed Register Names as Assembly Code)

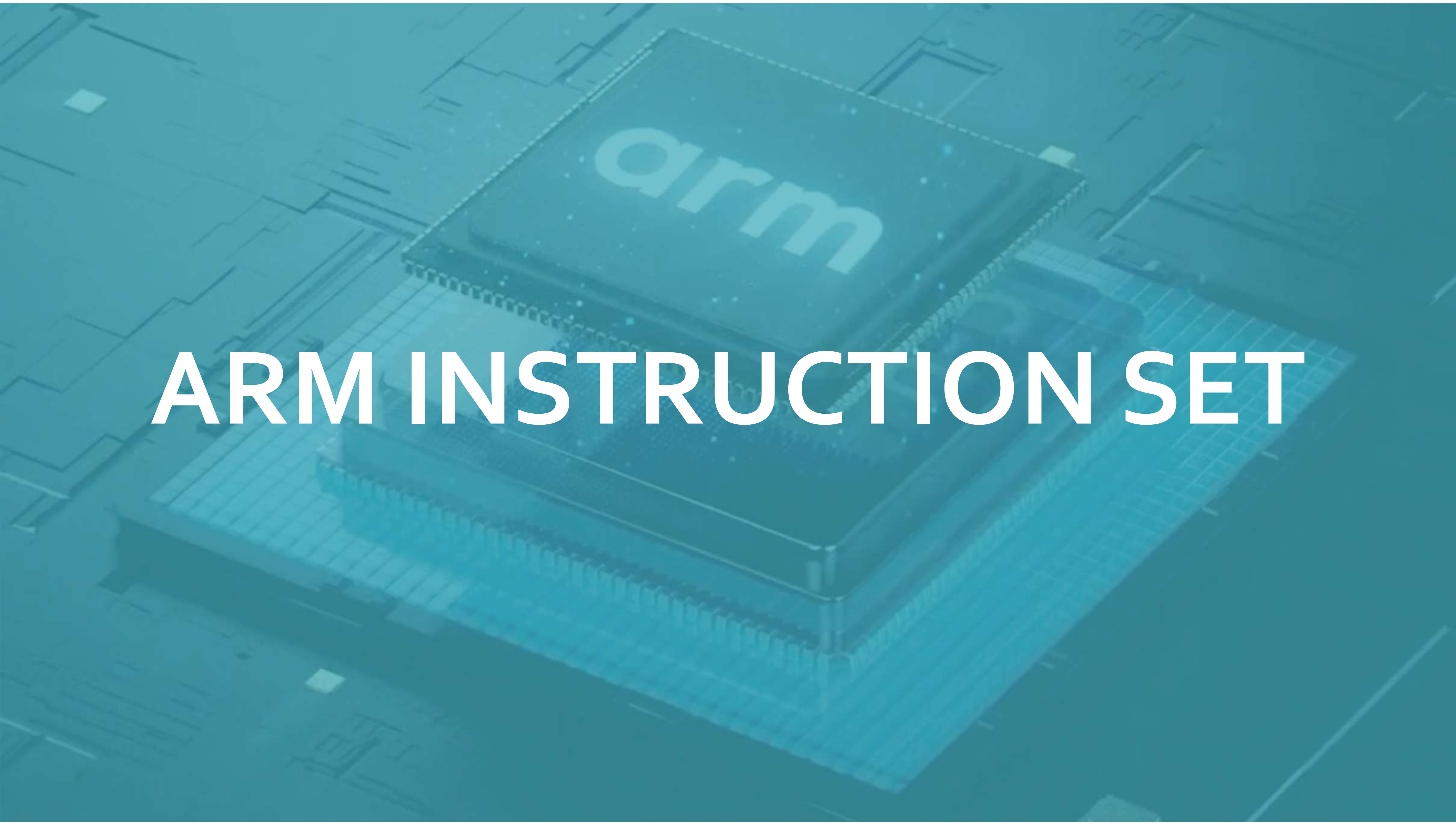
Special Registers

- Besides the registers in the register bank, there are a number of special registers. These registers contain the processor status and define the operation states and interrupt/exception masking.
- Special registers are not memory mapped, and can be accessed using special register access instructions such as MSR and MRS.

Program status registers

The Program Status Register is composed of three status registers:

- Application PSR (APSR)
- Execution PSR (EPSR)
- Interrupt PSR (IPSR)



ARM INSTRUCTION SET

The ARM instruction set can be divided into six broad classes of instruction:

- Branch instructions
- Data-processing instructions
- Status register transfer instructions
- Load and store instructions
- Coprocessor instructions
- Exception-generating instructions

1. Branch instructions

Branch instructions which can switch instruction set, so that execution continues at the branch target using the Thumb instruction set
Ex : BX, BL etc

2. Data-processing instructions

The data-processing instructions perform calculations on the general-purpose registers. There are five types of data-processing instructions:

1. Arithmetic/logic instructions
2. Comparison instructions
3. Single Instruction Multiple Data (SIMD) instructions(SUBT)
4. Multiply instructions
5. Miscellaneous Data Processing instructions

3. Status register transfer instructions

The status register transfer instructions transfer the contents of the CPSR or an SPSR to or from a general-purpose register.

4.Load and Store Instructions

The following load and store instructions are available:

- Load and Store Register (LDR/STR)
- Load and Store Multiple registers (LDM/STM)
- Load and Store Register Exclusive(LDREX/STREX)

5. Coprocessor Instructions

There are three types of coprocessor instructions:

Data-processing instructions

These start a coprocessor-specific internal operation.

Data transfer instructions

These transfer coprocessor data to or from memory. The address of the transfer is calculated by the ARM processor.

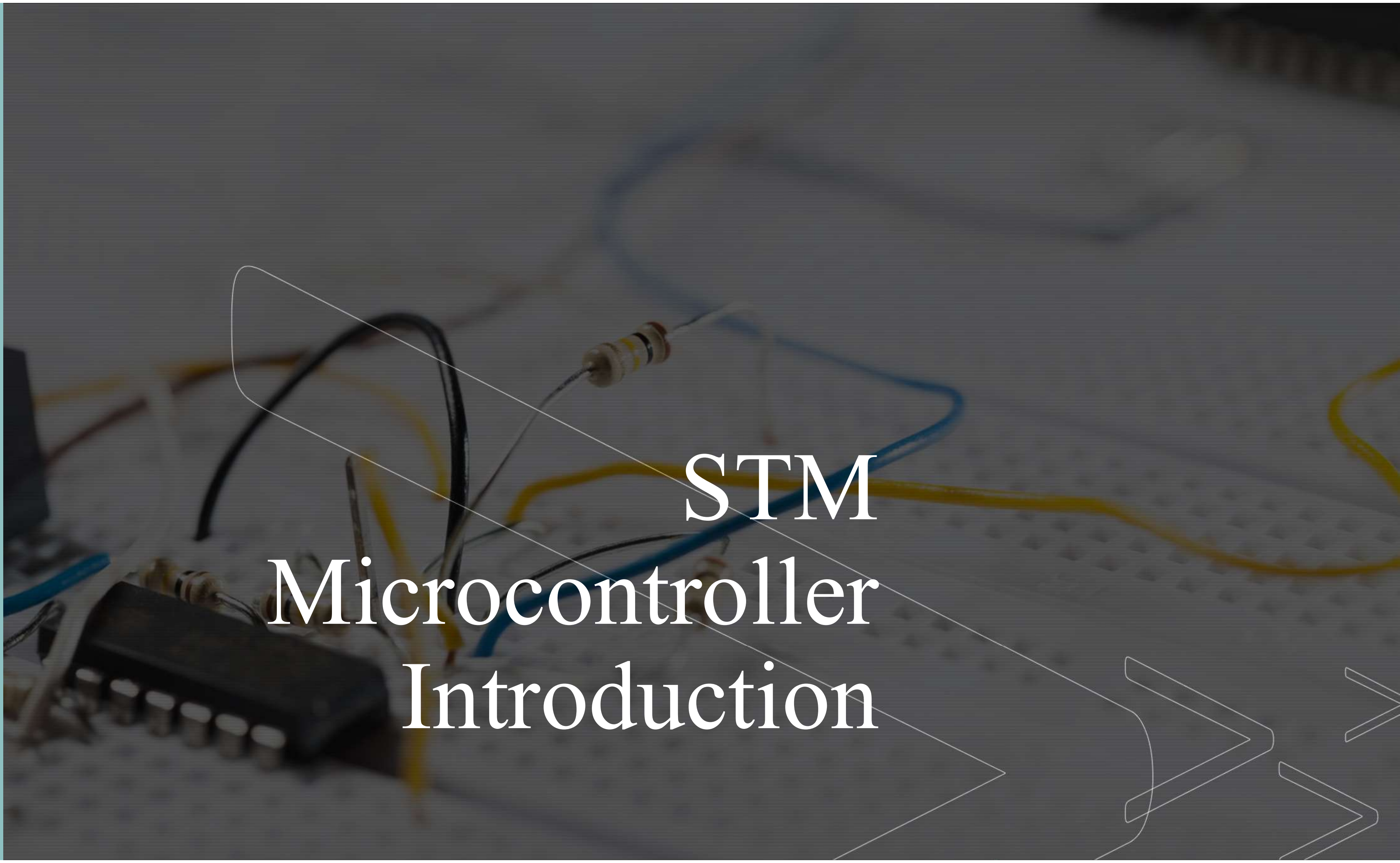
Register transfer instructions

These allow a coprocessor value to be transferred to or from an ARM register, or a pair of ARM registers.

6. Exception-generating instructions

Two types of instruction are designed to cause specific exceptions to occur.

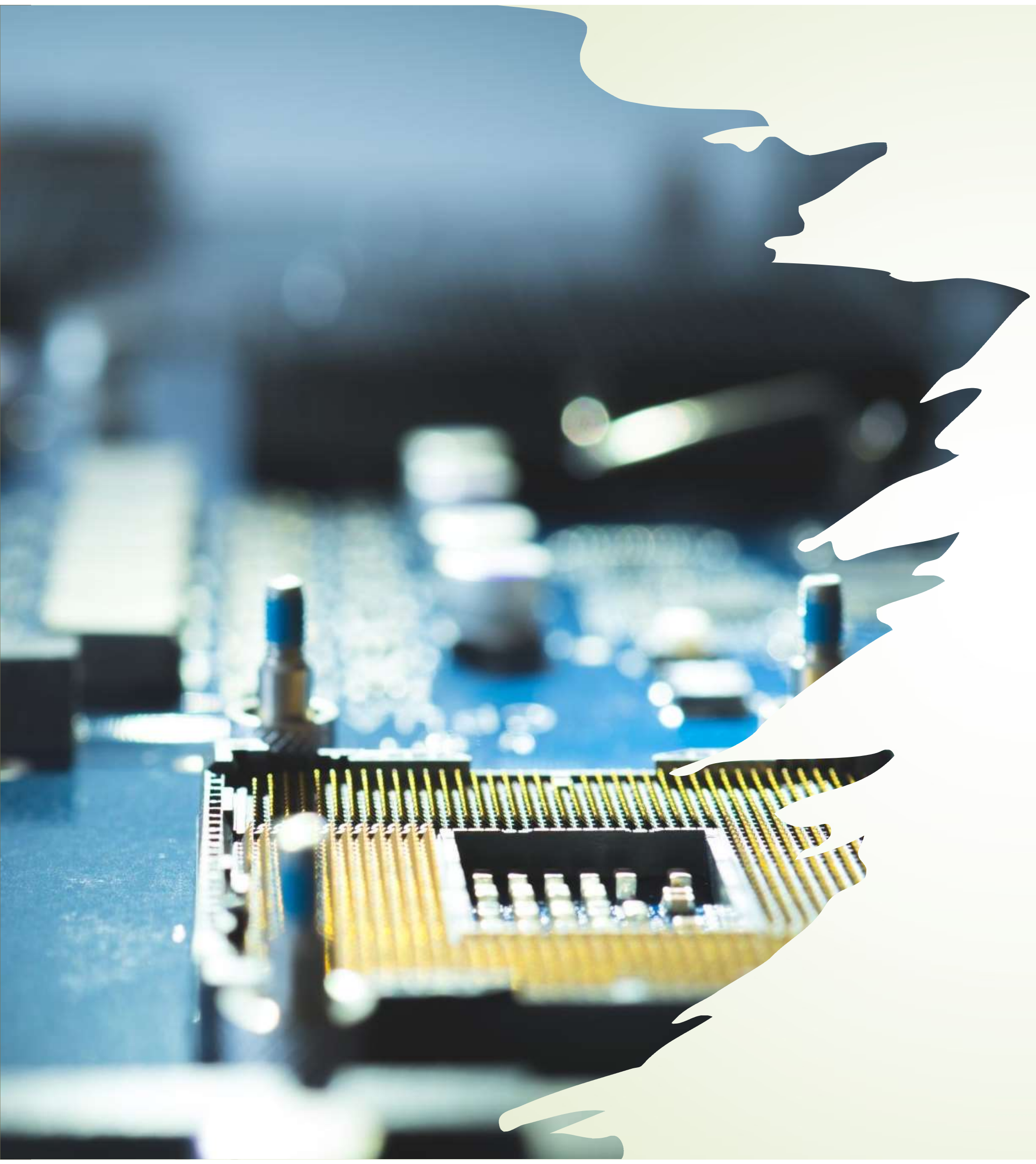
I. Software interrupt instructions
II. Software breakpoint instructions



STM Microcontroller Introduction

First things first

- What is Microcontroller ?
- What is Arm ?
- What is STM32 ?



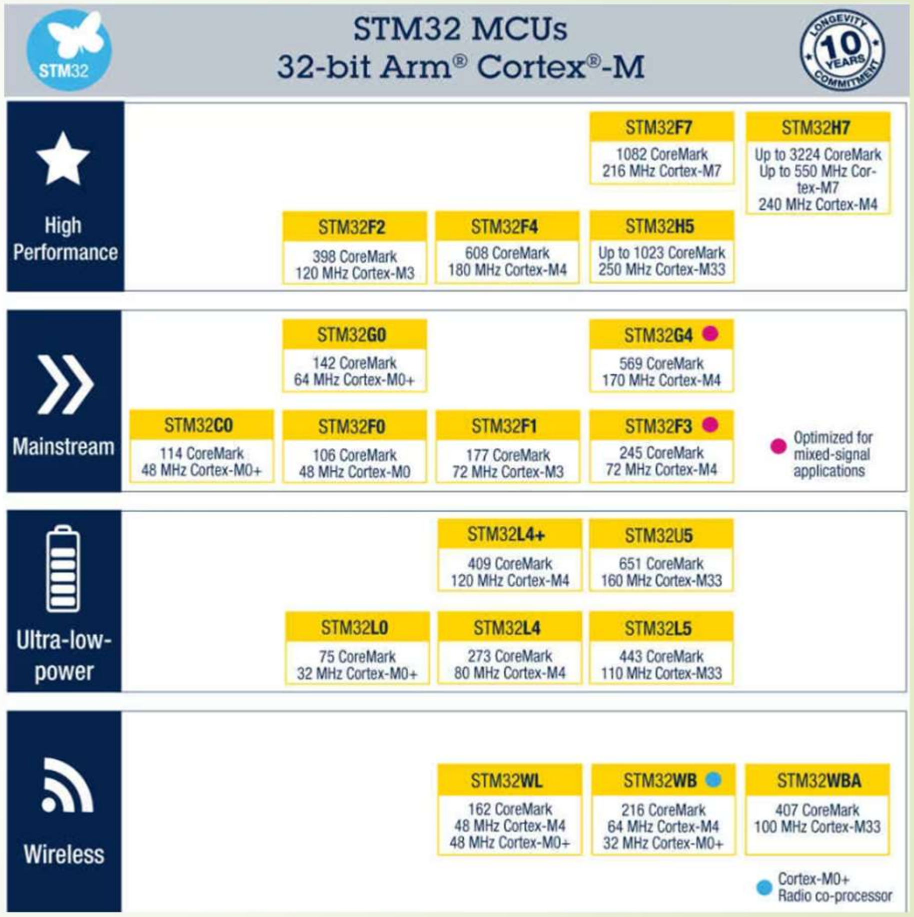
Processor vs Controller(Cortex-M vs STM32)

- Arm is the leading technology provider of processor IP, offering the widest range of processors to address the performance, power, and cost requirements of every device.
- STM32 is a family of 32-bit microcontroller integrated circuits by STMicroelectronics.

STM32 Intro

- The STM32 is a family of microcontroller ICs based on various 32-bit RISC ARM Cortex - M cores. STMicroelectronics licenses the ARM Processor IP from ARM Holdings.
- The ARM core designs have numerous configurable options, and ST chooses the individual configuration to use for each design. ST attaches its own peripherals to the core before converting the design into a silicon die.
- The STM32 chips are grouped into related series that are based around the same 32-bit ARM processor core: Cortex-M0, Cortex-M0+, Cortex-M3, Cortex-M4, Cortex-M7, Cortex-M33.

Different families of STM32 Cortex - M



STM32 MCUs
32-bit Arm® Cortex®-M

LONGEVITY 10 YEARS COMMITMENT

High Performance

- STM32F7 — 1082 CoreMark, 216 MHz Cortex-M7
- STM32H7 — Up to 3224 CoreMark, Up to 550 MHz Cortex-M7, 240 MHz Cortex-M4
- STM32F2 — 398 CoreMark, 120 MHz Cortex-M3
- STM32F4 — 608 CoreMark, 180 MHz Cortex-M4
- STM32H5 — Up to 1023 CoreMark, 250 MHz Cortex-M33

Mainstream

- STM32G0 — 142 CoreMark, 64 MHz Cortex-M0+
- STM32G4 ● — 569 CoreMark, 170 MHz Cortex-M4
- STM32C0 — 114 CoreMark, 48 MHz Cortex-M0+
- STM32F0 — 106 CoreMark, 48 MHz Cortex-M0
- STM32F1 — 177 CoreMark, 72 MHz Cortex-M3
- STM32F3 ● — 245 CoreMark, 72 MHz Cortex-M4

● Optimized for mixed-signal applications

Ultra-low-power

- STM32L4+ — 409 CoreMark, 120 MHz Cortex-M4
- STM32U5 — 651 CoreMark, 160 MHz Cortex-M33
- STM32L0 — 75 CoreMark, 32 MHz Cortex-M0+
- STM32L4 — 273 CoreMark, 80 MHz Cortex-M4
- STM32L5 — 443 CoreMark, 110 MHz Cortex-M33

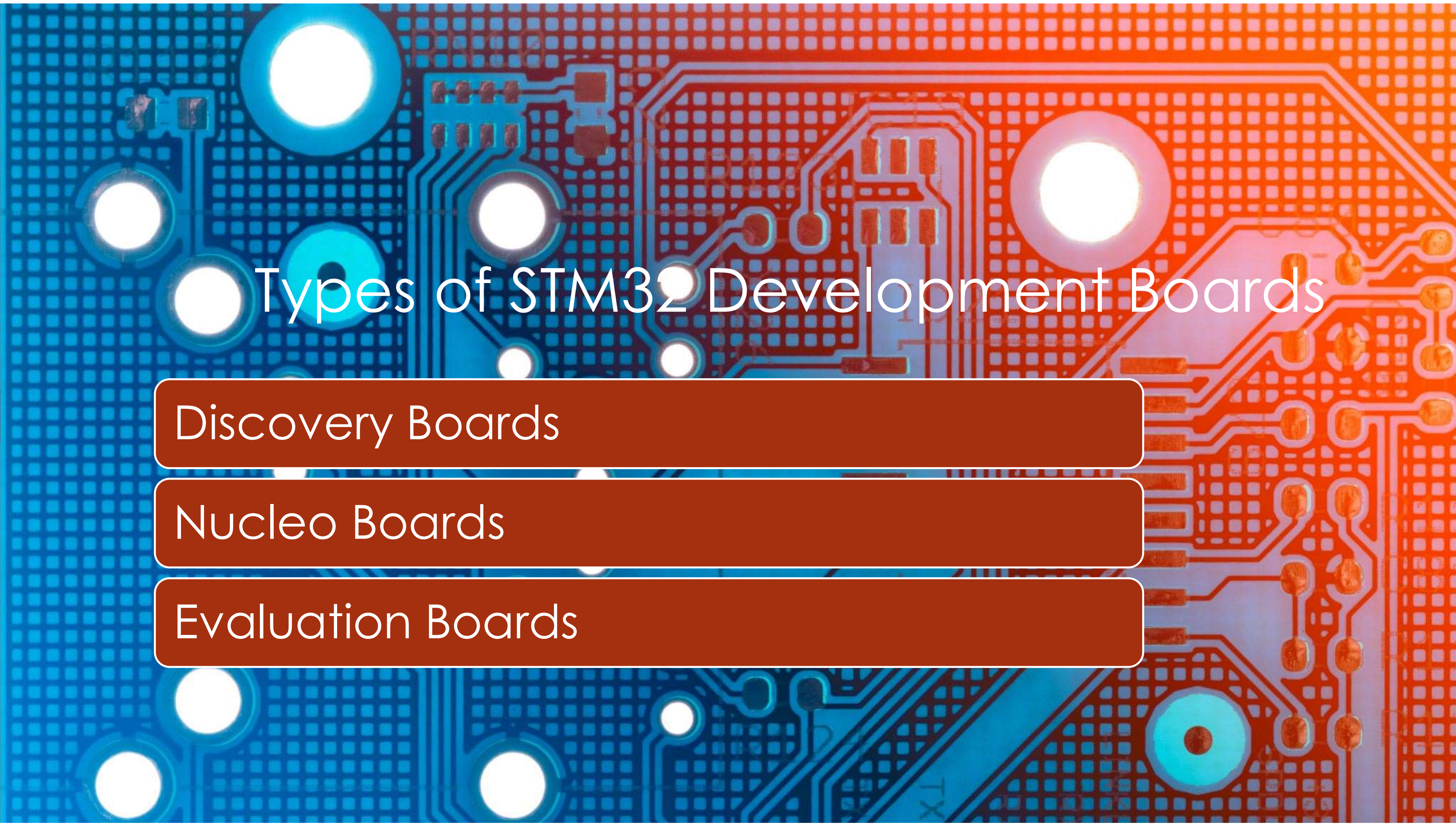
Wireless

- STM32WL — 162 CoreMark, 48 MHz Cortex-M4, 48 MHz Cortex-M0+
- STM32WB ● — 216 CoreMark, 64 MHz Cortex-M4, 32 MHz Cortex-M0+
- STM32WBA — 407 CoreMark, 100 MHz Cortex-M33

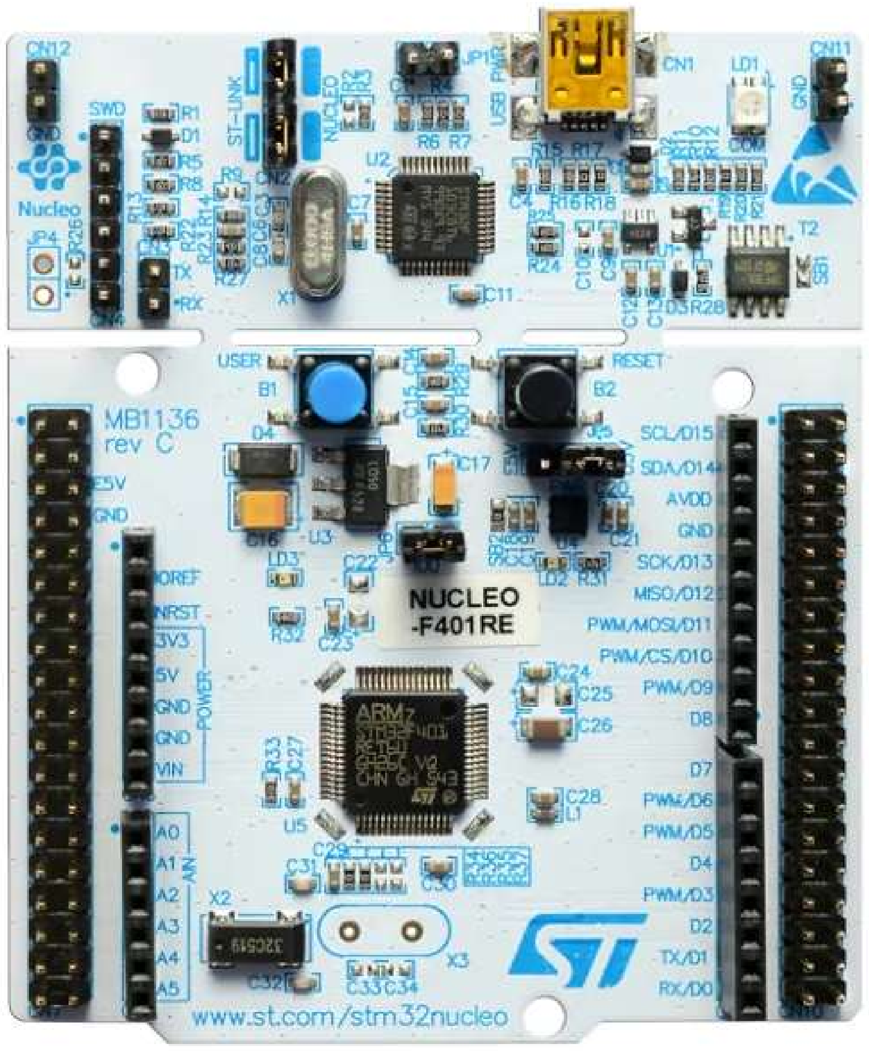
● Cortex-M0+ Radio co-processor

Types of STM32 Development Boards

- Discovery Boards
- Nucleo Boards
- Evaluation Boards



Nucleo Boards



- The STM32 Nucleo board series are based on ARM Cortex-M 32-bit RISC cores optimised for high performance and energy efficiency.
- STM32 Nucleo boards can easily be extended with a large number of specialized application hardware add-ons like Arduino Uno Rev3 and ST morpho connectors on Nucleo-144 and Nucleo-64 and Arduino Nano connectors on Nucleo-32.
- On-board ST-LINK debugger/programmer and it also provides Virtual COM port.
- Ex : STM32F411RE,STM32L152RE etc

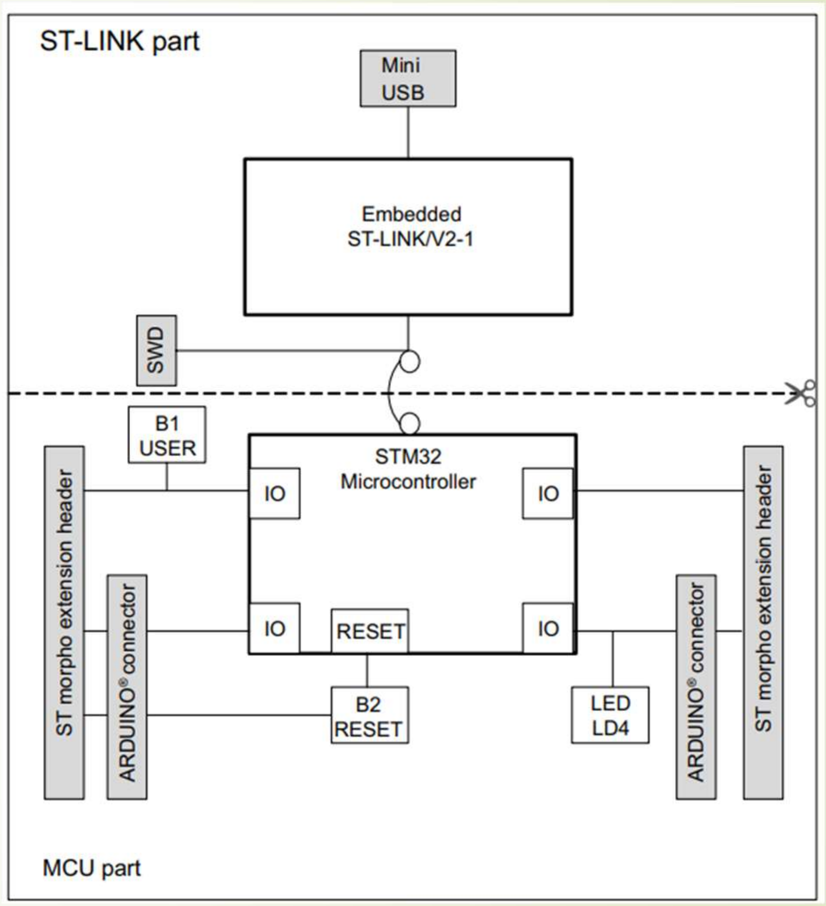
Discovery Boards

- STM32 discovery kits provide affordable and complete solutions to evaluate the application-specific features of STM32 MCUs and MPUs.
- Most discovery boards have additional peripherals like displays, external memory, sensors etc. The STM32 discovery board is rich in features compared to the Nucleo board.
- Ex : STM32F407VGT6, STM32F051R8T6 etc

Evaluation Boards

- The STM32 eval boards have been designed as a complete demonstration and development platform for the STM32 MCUs and MPUs.
- They carry external circuitry, such as transceivers, sensors, memory interfaces, displays and many more. The evaluation boards can be considered as a reference design for application development.
- Ex : STM32G0C1VE , STM32H753XI etc

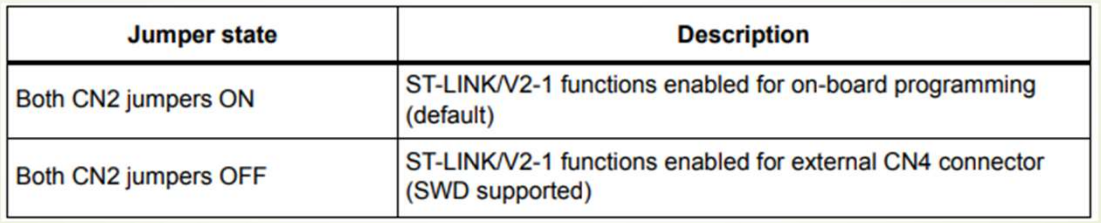
Nucleo Hardware Block diagram



Embedded ST-LINK/V2-1

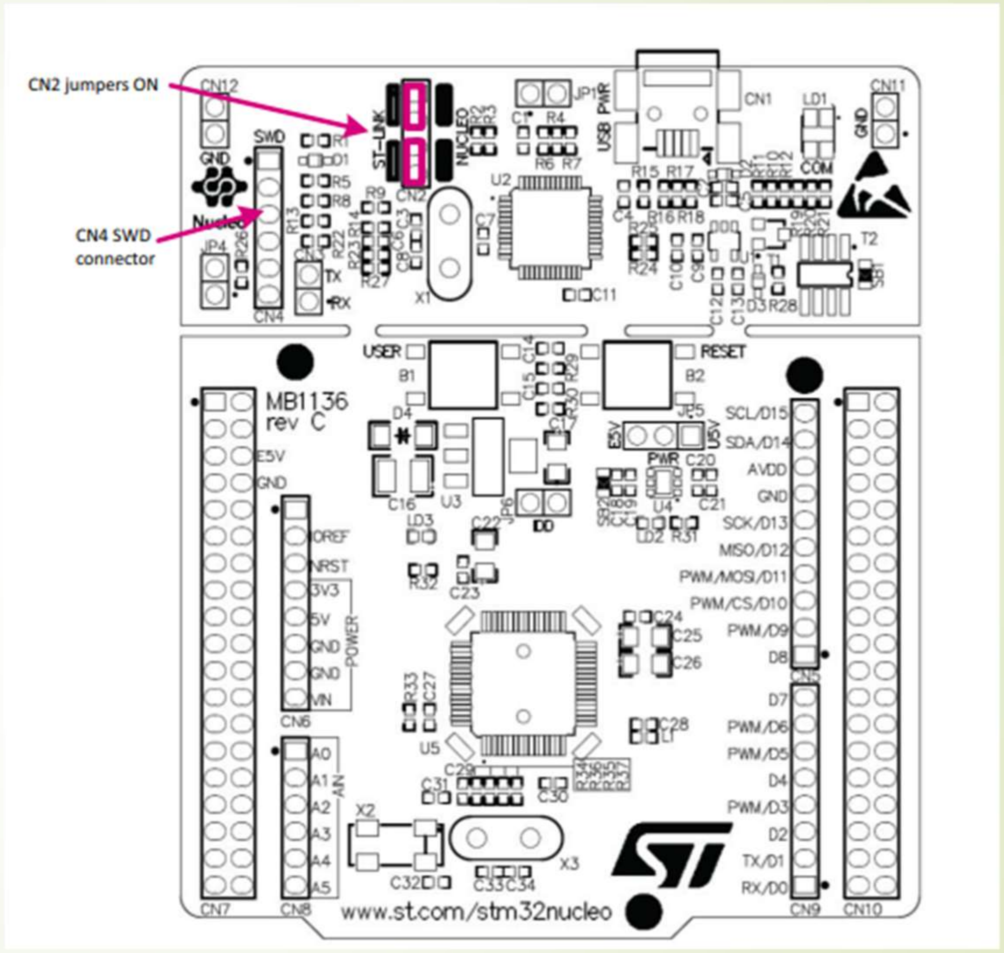
- The ST-LINK/V2-1 programming and debugging tool is integrated into the STM32 Nucleo board.
- Virtual COM port interface on USB

Jumper States

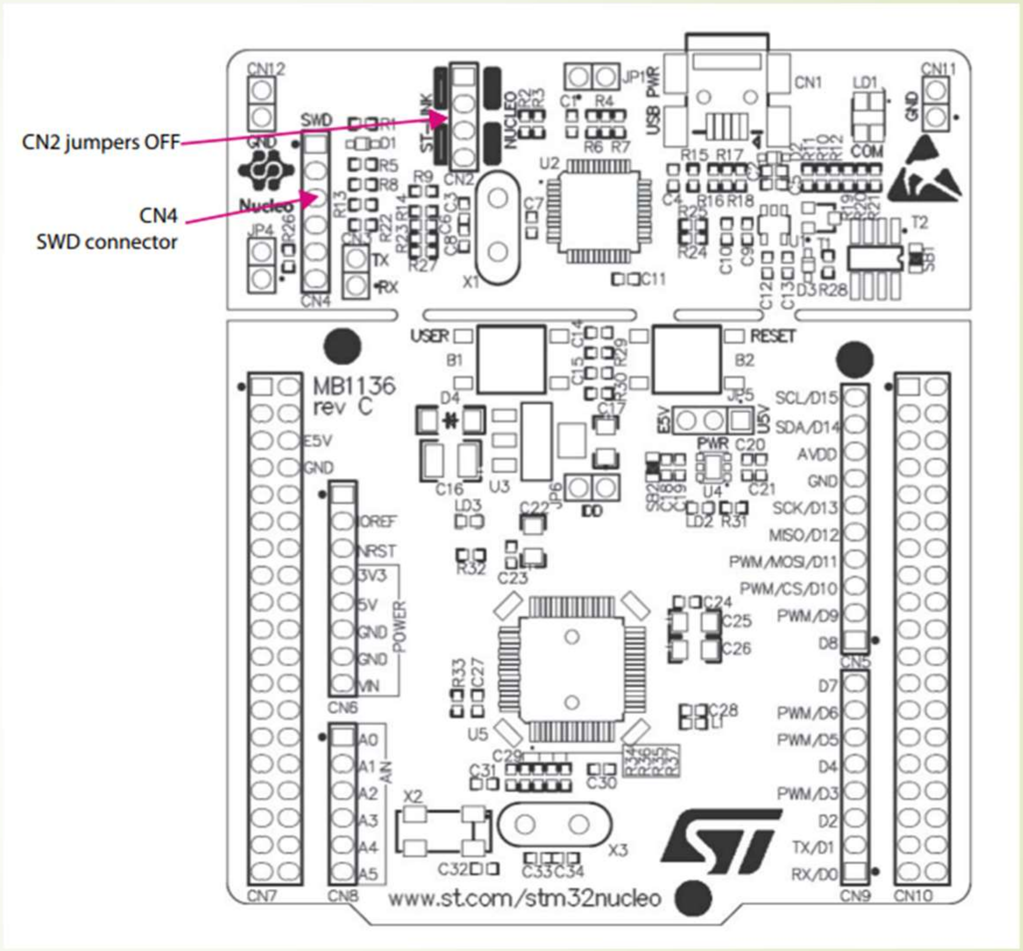


Jumper state	Description
Both CN2 jumpers ON	ST-LINK/V2-1 functions enabled for on-board programming (default)
Both CN2 jumpers OFF	ST-LINK/V2-1 functions enabled for external CN4 connector (SWD supported)

Connecting the STM32 Nucleo board to program the on-board STM32



Using ST-LINK/V2-1 to program the STM32 on an external application



CN4 Pinout :

Pin	CN4	Designation
1	VDD_TARGET	VDD from application
2	SWCLK	SWD clock
3	GND	ground
4	SWDIO	SWD data input/output
5	NRST	RESET of target STM32
6	SWO	Reserved

LEDs

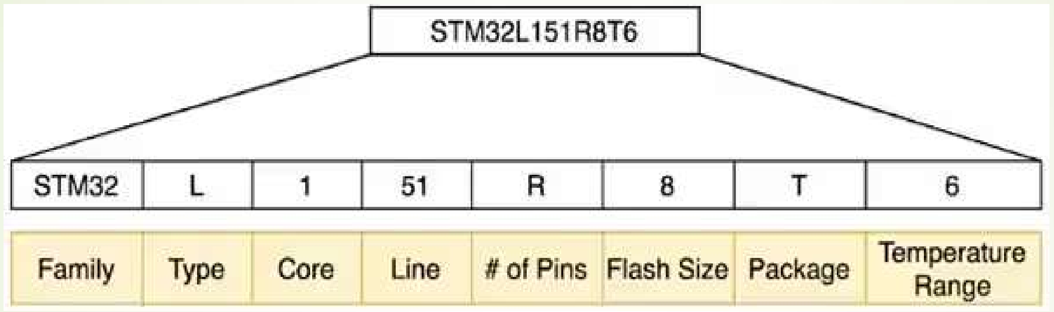
- The tricolor LED (green, orange, red) LD1 (COM) provides information about ST-LINK communication status. LD1 default color is red. LD1 turns to green to indicate that communication is in progress between the PC and the ST-LINK/V2-1, with the following setup:

- **Slow blinking Red/OFF**: at power-on before USB initialization
- **Fast blinking Red/OFF**: after the first correct communication between the PC and ST-LINK/V2-1
- **Red LED ON**: when the initialization between the PC and ST-LINK/V2-1 is complete
- **Green LED ON**: after a successful target communication initialization
- **Blinking Red/Green**: during communication with the target
- **Green ON**: communication finished and successful
- **Orange ON**: Communication failure

User LD2: The green LED is a user LED

LD3 PWR: The red LED indicates that the STM32 part is powered and +5V power is available.

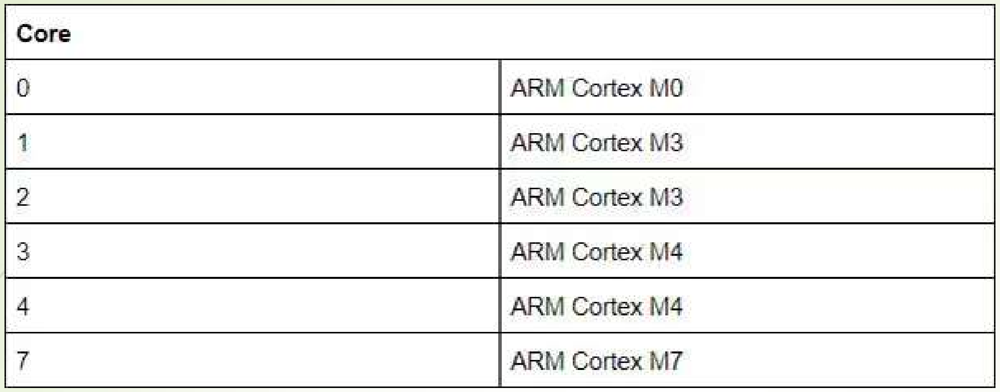
Nomenclature



Type

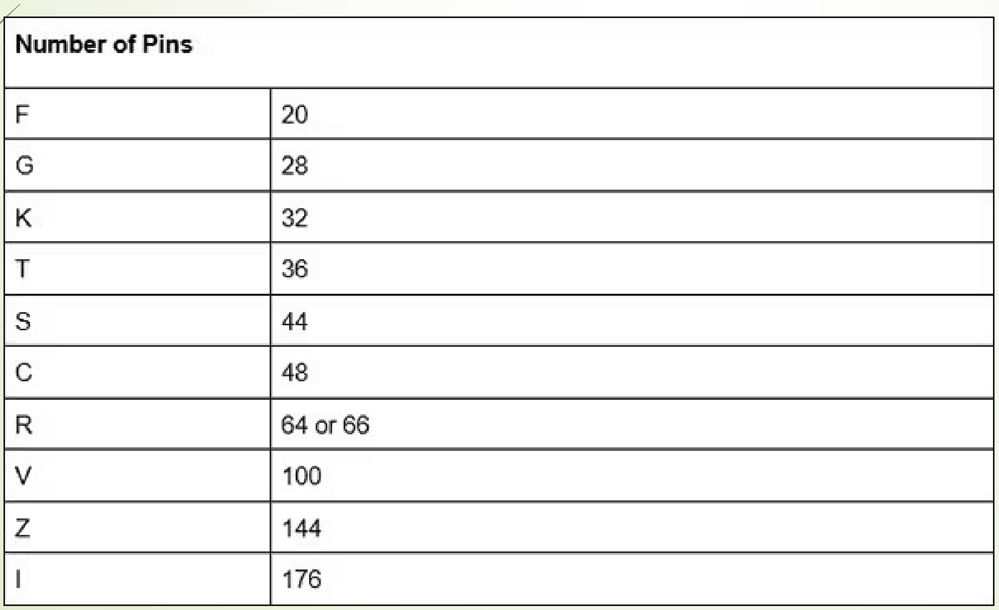
Type	
F	Foundation; Sometimes: High-Performance (for example STM32 F2 series)
G	Mainstream
L	Low-Power
H	High-Performance
W	Wireless

Core



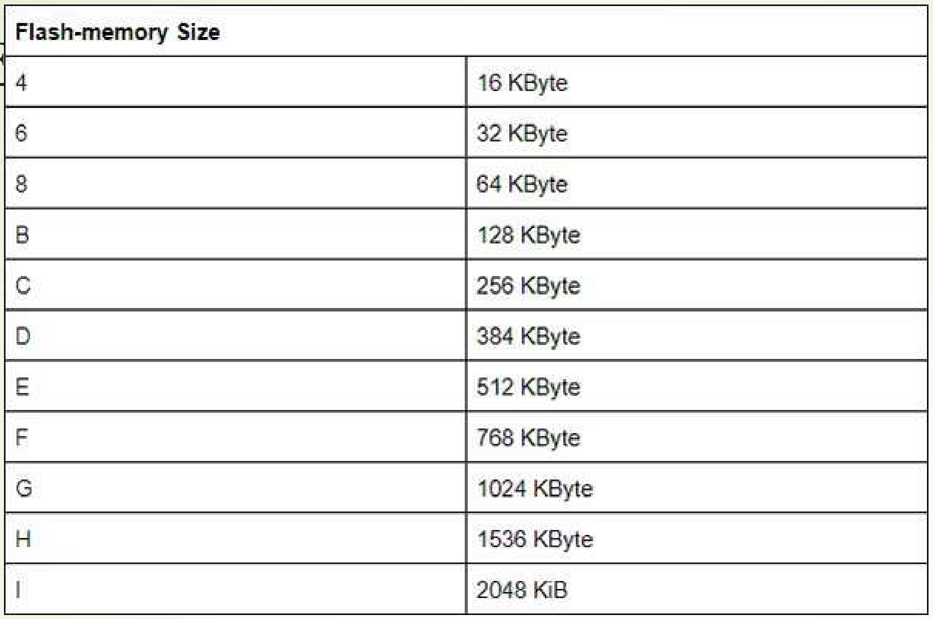
Core	
0	ARM Cortex M0
1	ARM Cortex M3
2	ARM Cortex M3
3	ARM Cortex M4
4	ARM Cortex M4
7	ARM Cortex M7

No Of Pins



Number of Pins	
F	20
G	28
K	32
T	36
S	44
C	48
R	64 or 66
V	100
Z	144
I	176

F

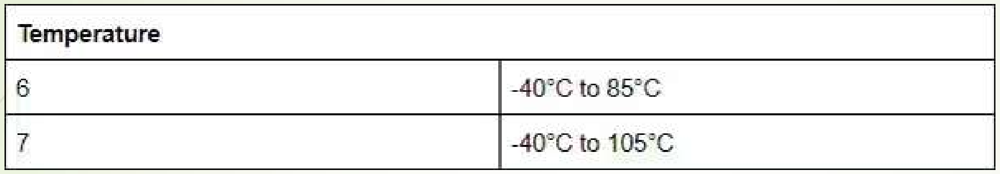


Flash-memory Size	
4	16 KByte
6	32 KByte
8	64 KByte
B	128 KByte
C	256 KByte
D	384 KByte
E	512 KByte
F	768 KByte
G	1024 KByte
H	1536 KByte
I	2048 KiB

Package

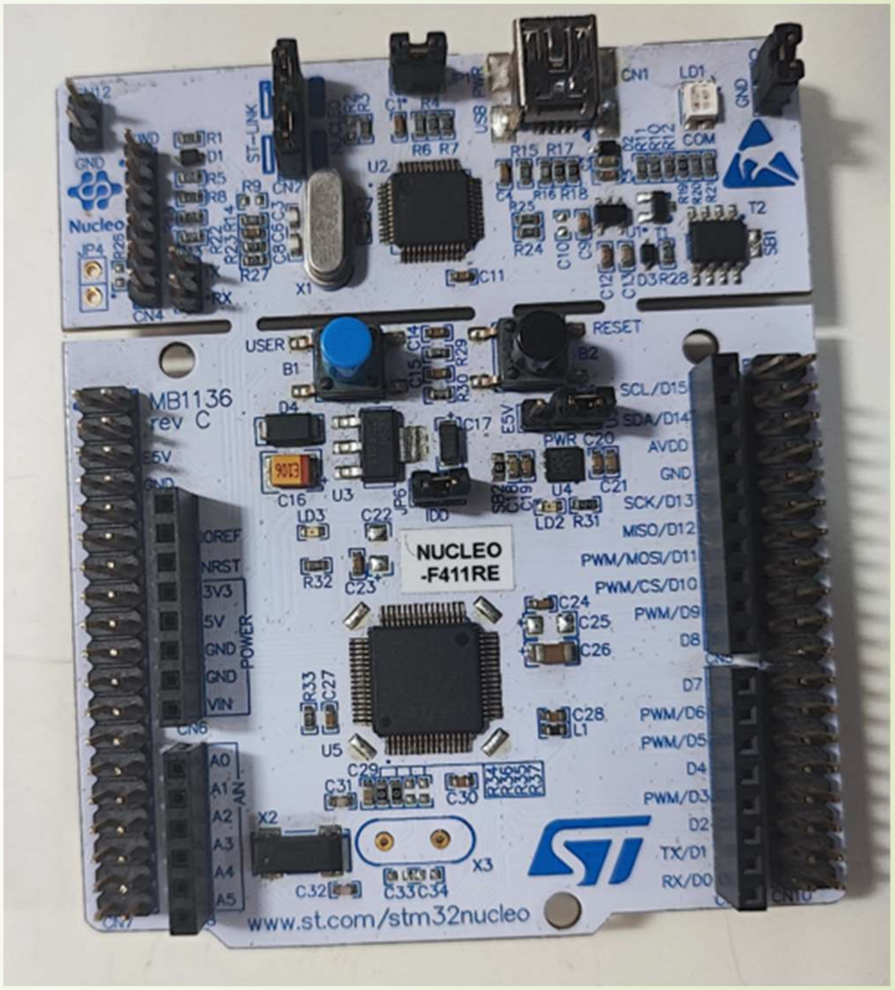
- P → TSSOP(Thin Shrink Small Outline Package)
- H → BGA(Ball Grid Array)
- U → UFQFPN(7mm X 7mm)
- T → LQFP(Low-profile Quad Flat Package)
- Y → WLCSP(Wafer Level Chip Scale Package)

Temperature Range



Temperature	
6	-40°C to 85°C
7	-40°C to 105°C

STM32F411RET6



BUS Protocols and Bus Interfaces

Bus Protocols and Bus interfaces

- In Cortex M processors the bus interfaces are based on Advanced Microcontroller Bus Architecture (AMBA) specification.
- AMBA is a specification designed by ARM which governs standard for on-chip communication inside the system on chip.

Bus Protocols :

- AHB (Main System Bus)
- APB(Peripheral bus)

Advanced High-performance Bus(AHB)

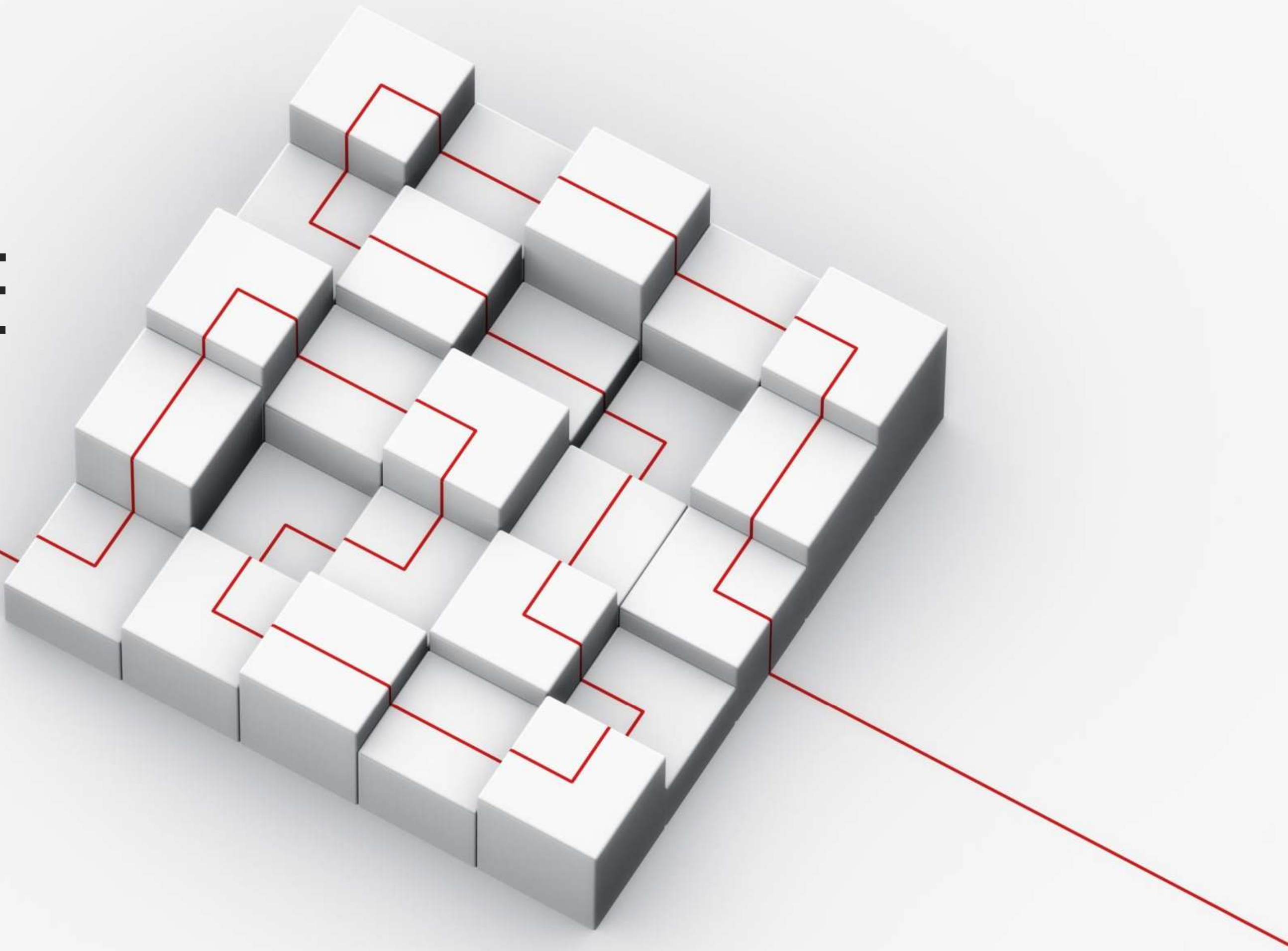
- In The cortex m3 and m4 processors , the AHB protocol used for the main bus interfaces
- It is used for high-speed communication. Common AHB Slaves are internal memory devices, external memory interfaces and high bandwidth peripherals.

Advanced Peripheral Bus (APB)

- The APB is an AMBA-compliant bus optimized for minimum power and reduced interface complexity
- The AHB-APB bridge is an AHB slave that provides an interface between the high- speed AHB domain and the low-power APB domain.
- It is used to interface to any peripherals which are low bandwidth and do not require the high

STM32411RE Block Diagram

(Refer data sheet Pg No 15)



System Architecture

- In STM32F411xC/E, the main system consists of 32-bit multilayer AHB bus matrix that interconnects:

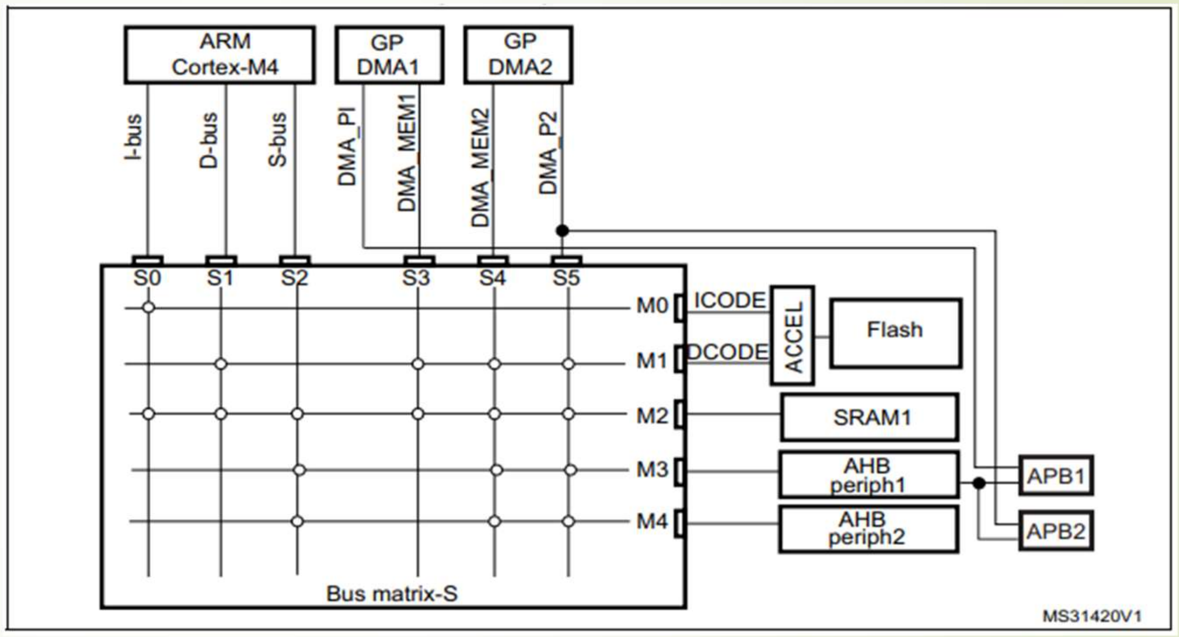
❑ **<u>Six masters:</u>**

- Cortex®-M4 with FPU core I-bus, D-bus and S-bus
- DMA1 memory bus
- DMA2 memory bus
- DMA2 peripheral bus

❑ **<u>Five slaves:</u>**

- Internal Flash memory ICode bus
- Internal Flash memory DCode bus
- Main internal SRAM
- AHB1 peripherals including AHB to APB bridges and APB peripherals
- AHB2 peripherals

Multi-AHB Bus Matrix



Documentation(Might vary for other boards)

Board level

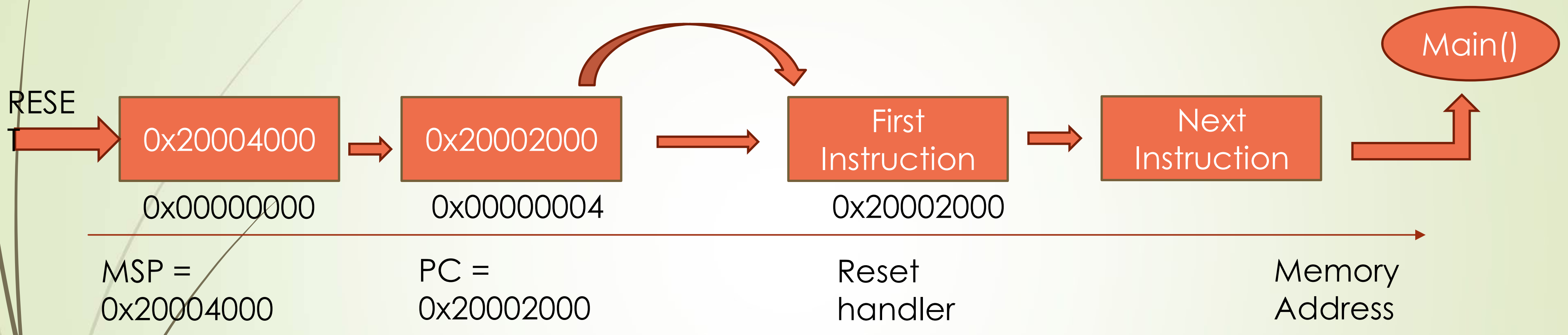
MCU level

CPU level

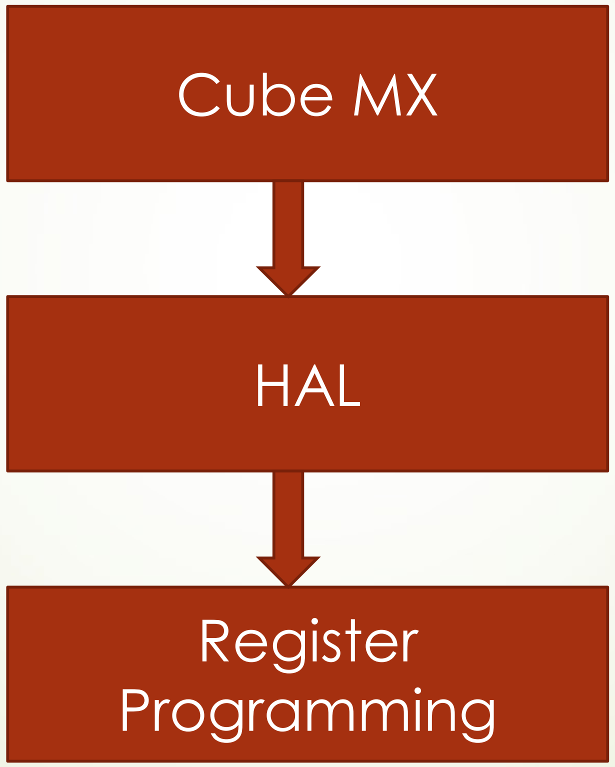
Reset sequence of Processor

- After reset, PC is loaded with the address 0x00000000.
- Processor read the value from 0x00000000 location in to MSP.
- Then processor reads the address of the reset handler from the location 0x00000004
- Then it jumps to reset handler and start executing the instructions
- Then you can call your main() function from reset handler.
- Reset handler is a C or assembly function written by programmer to carry out required initializations.

Reset Sequence flow



Programming flow



Hardware Abstraction Layer(HAL)

The hardware abstraction layer reside below the application programming interface (API) in a software stack.

It bridges the gap between hardware and software and makes developer work easy.

STM32Cube IDE

- STM32CubeIDE is an advanced C/C++ development platform with peripheral configuration, code generation, code compilation, and debug features for STM32 microcontrollers and microprocessors.
- It is based on the Eclipse®/CDT™ framework and GCC toolchain for the development, and GDB for the debugging.