

Lecture1

CORTEX M3/M4 OPERATION | MODES | REGISTER

By: Abdelrahman matarawy

Processor mode:

- **Thread mode:** Used to execute application software. The processor enters Thread mode when it comes out of reset.
- **Handler mode:** The processor returns to Thread mode when it has finished all exception processing.

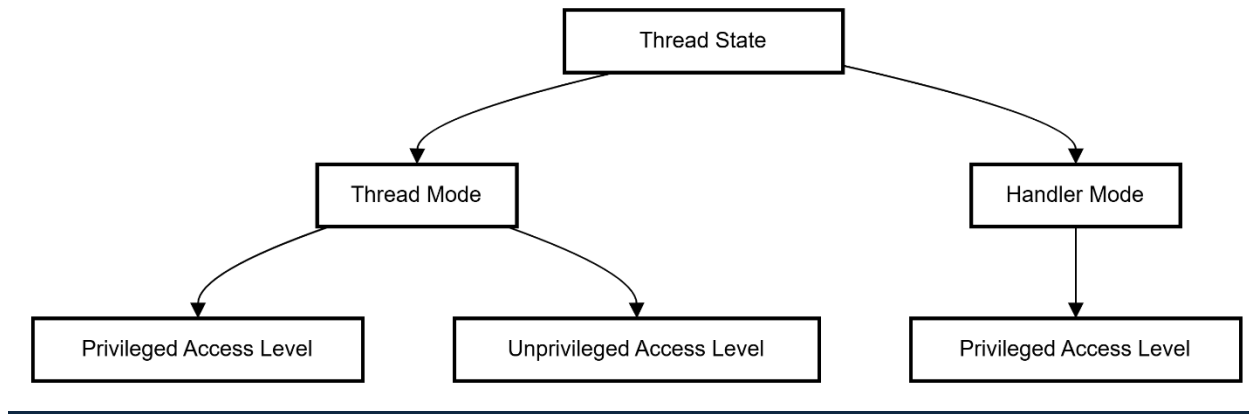
The privilege levels for software execution:

- **Unprivileged:**
 - The software has limited access to the MSR and MRS instructions, and cannot use the CPS instruction
 - The software cannot access the system timer, NVIC, or system control block
 - The software might have restricted access to memory or peripherals.

“Unprivileged software executes at the unprivileged level”
- **Privileged:**
 - The software can use all the instructions and has access to all resources. Privileged software executes at the privileged level.

“In Thread mode, the CONTROL register controls whether software execution is privileged or unprivileged, In Handler mode, software execution is always privileged.”

+ Diagram:



+ The Difference Between Thumb/ARM/Thumb2:

○ **Thumb:**

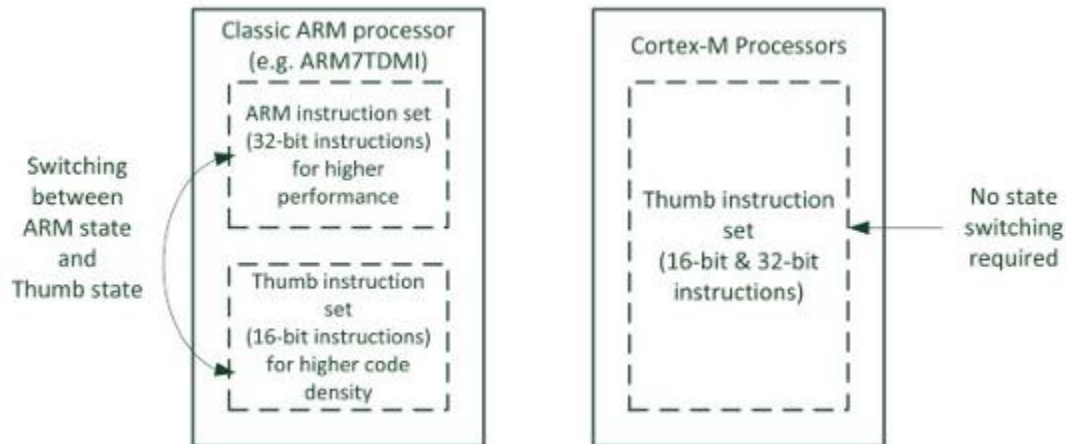
- Thumb instructions Set architecture Having Fixed Length 16 Bits.
- Low performance But Small in Size
- Used in Cortex M0.

○ **ARM:**

- ARM instructions Set architecture Having Fixed Length 32 Bits.
- High performance But Big in Size
- Used in Cortex M1/M2

○ **Thumb2:**

- Thumb2 instructions Set architecture Having Fixed Length 16 & 32 Bits.
- Best in performance and in size
- Used in Cortex M3/M4



Output in Keil Micro Vision:

- In Main the operation mode is thread and privileged

The screenshot displays the Keil Micro Vision IDE interface during a debug session. The main window shows the assembly code for the `main.c` file, with the `while(1)` loop highlighted. The `Registers` window on the left shows the current state of the processor registers, including `R0` through `R15` and `xPSR`. The `Logic Analyzer` window shows a trace of the execution, with a time scale of 8.002 ms. The `External Interrupts (EXTI)` window shows the configuration of the EXTI registers, including the `EXTI_IMR`, `EXTI_RTSR`, `EXTI_EMR`, and `EXTI_FTSR` registers. The `General Purpose I/O B (GPIOB)` window shows the configuration of the GPIOB pins, including the `GPIOB_CRH`, `GPIOB_CRL`, `GPIOB_IDR`, `GPIOB_ODR`, and `GPIOB_LCKR` registers.

- In ISR the operation mode is Handler and privileged

The screenshot displays the STM32CubeIDE interface with several windows open:

- Registers Window:** Shows the current state of registers. R15 (PC) is highlighted at address 0x08000938, and xPSR is at 0x21000027. The Mode is set to Handler.
- Logic Analyzer Window:** Shows a trace at 81.17731 s. The selected line is 35, which is the start of the EXTI callback function.
- Disassembly Window:** Shows the assembly code for the selected line 35: `void EXTI_Callback(void)`. The code includes setting the IRQ flag and initializing the clock.
- External Interrupts (EXTI) Window:** Shows the configuration for EXTI lines 3 to 10. Line 3 is selected, and the source is PA.3. The event is configured as a rising edge trigger.
- General Purpose I/O B (GPIOB) Window:** Shows the configuration for GPIOB. The mode is set to Input, and the pin is configured as a floating input.