Microprocessors & Microcontrollers
: Arm Cortex M0+
(Using RP2040)

ESM_13

Arm Cortex-M0+ Register Set and Stack

Mouli Sankaran

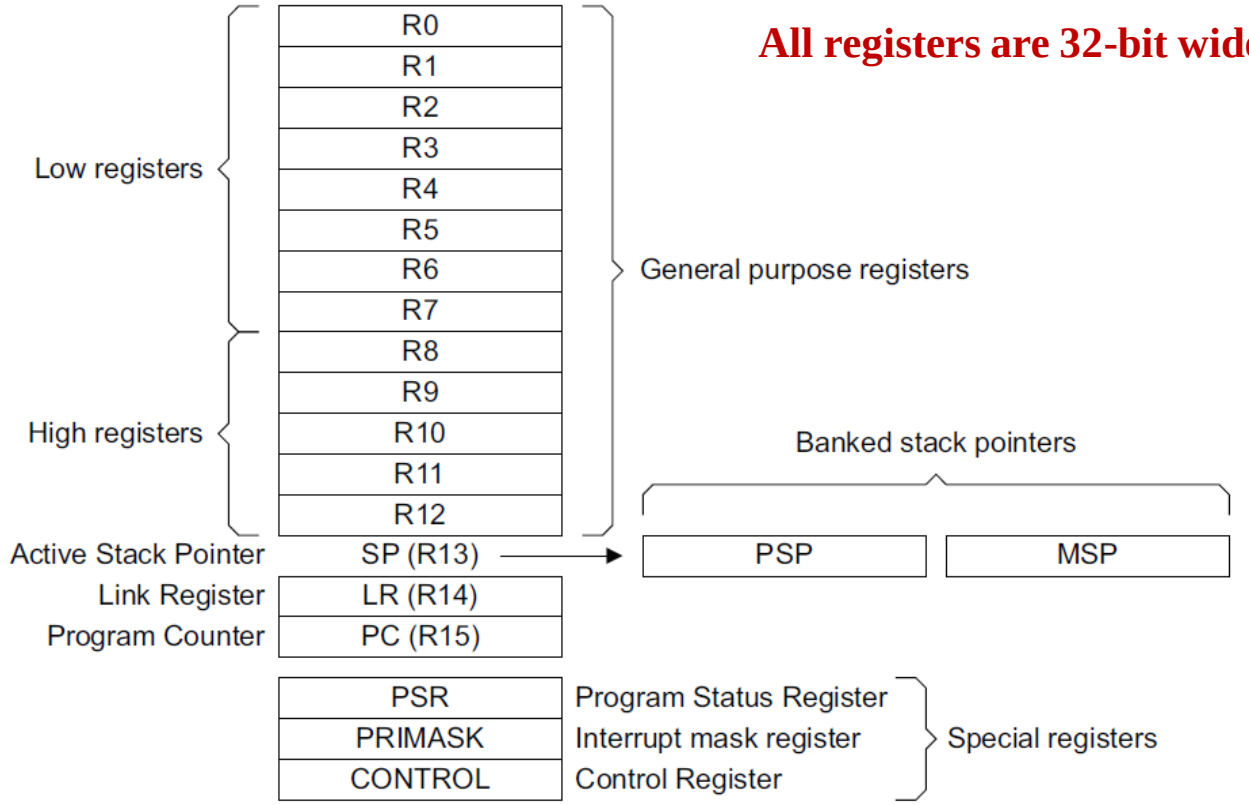
Lecture 2.1.5 Focus

- Arm- Cortex-M0+
  - Register Set
  - Stack Implementation in ARM
  - Different modes
  - Full Descending Stack (Cortex-M0+)

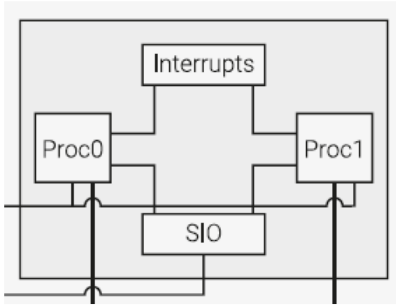
Arm –Cortex-M0+

(Register Set)

Arm-Cortex-M0+: Register Set



All registers are 32-bit wide



PSP: Process Stack Pointer

MSP: Main Stack Pointer (default on reset)

Ref: Ref4_ARM-Cortex-M0+ Devices Generic User Guide, Section 2.1.3 Core Registers

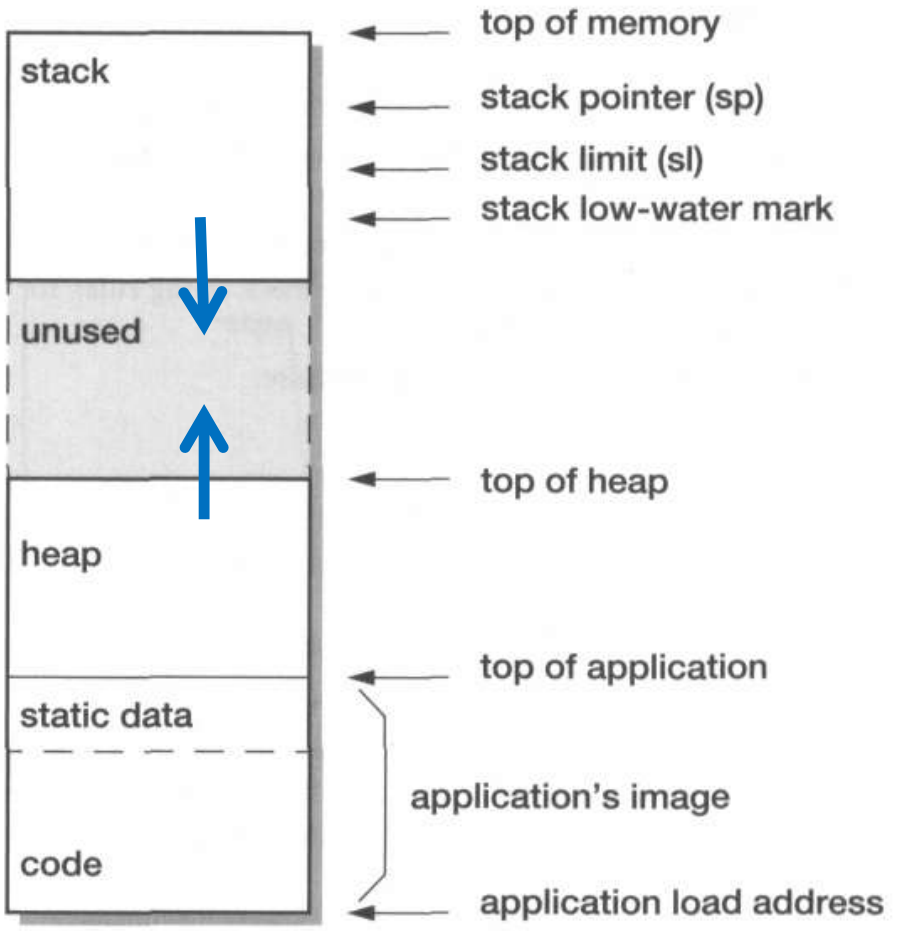
Stack Implementation in ARM

Standard ARM (32 bit) Address Space Model

Upper Address (0xFFFF FFFF)

32 Bit Physical Address Space of size **4 GB**

Lower Address (0x0000 0000)



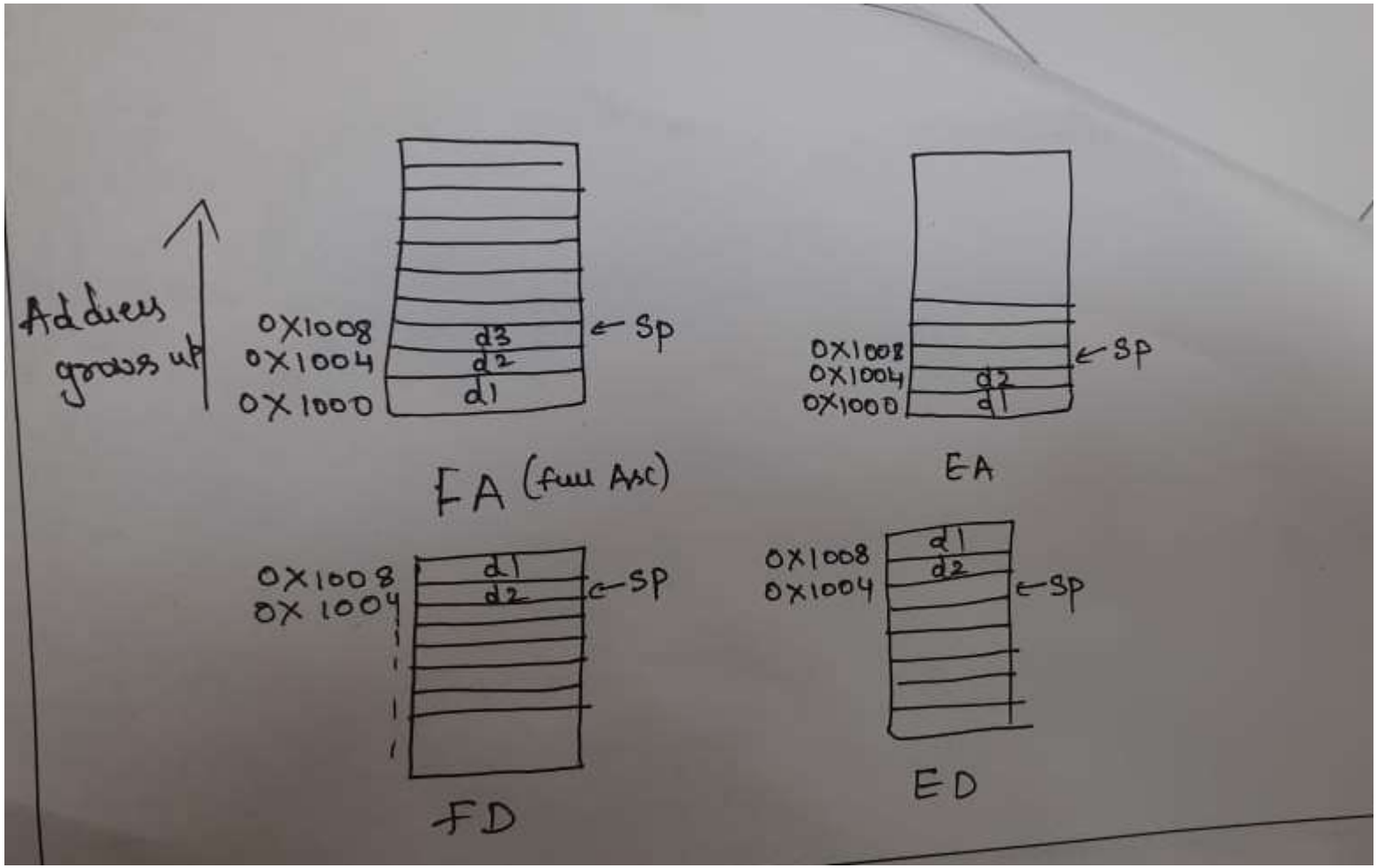
Stack Addressing modes

Stack Processing

ARM support for all four forms of stacks

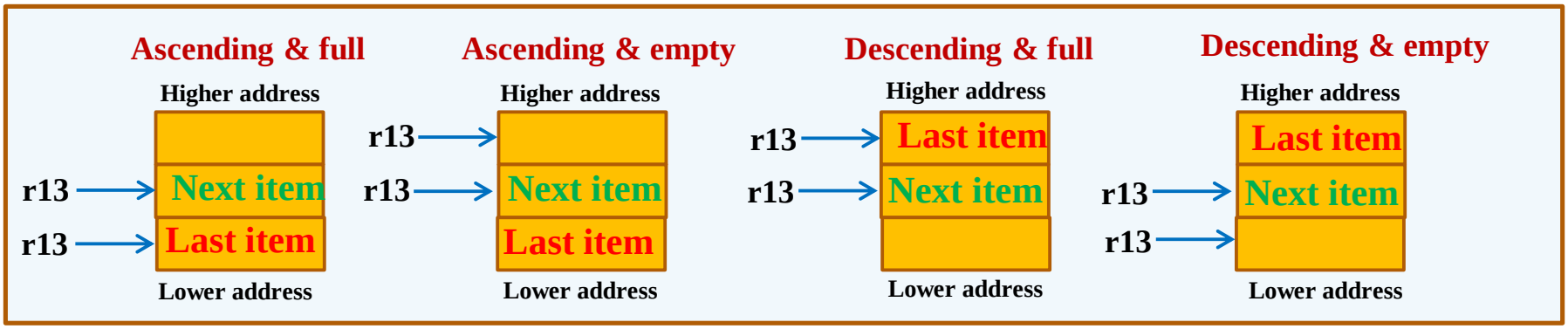
- **Full ascending (FA):** grows up; stack pointer points to the highest address containing a valid data item
- **Empty ascending (EA)**: grows up; stack pointer points to the first empty location
- **Full descending (FD):** grows down; stack pointer points to the lowest address containing a valid data item
- **Empty descending (ED):** grows down; stack pointer points to the first empty location below the stack

Stack Addressing modes



Various Stack Addressing Modes

- The address to be used to store a data value in the stack, is not known at the time the program is compiled or assembled
- A stack is usually implemented as a linear memory space which
  - Grows up (an **ascending** stack) or
  - Grows down (a **descending** stack) in memory as data is pushed into it
- Stack shrinks back as data is removed or popped out
- **Stack Pointer (r13)** holds the address of the current top of the stack
  - Either by pointing to the last valid data item pushed onto the stack (a **full** stack) or
  - By pointing to the vacant slot where the next data item will be placed (an **empty** stack)

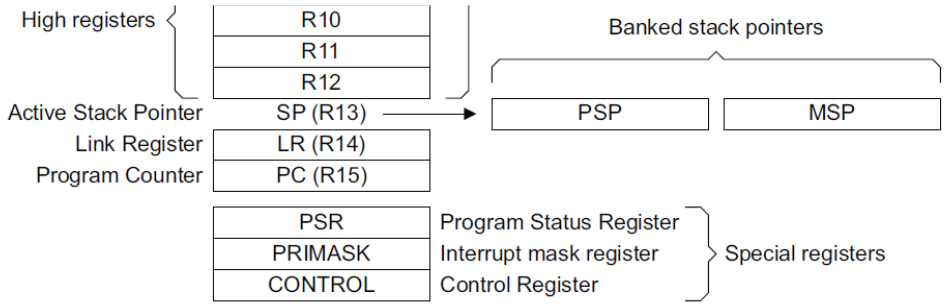


Stack Implementation in ARM

- The choice of implementing one of the **four types** of stack in ARM is with the system designer
  - HW decides the particular type of stack implementation
- A particular type of stack implementation needs to be followed for the application to work together
- There are different instructions available in ARM to implement any of the four types of stacks
- Different register transfer instructions are supported to implement these four types of stack by advanced families of ARM (not in Cortex-M0+)
- **Arm Cortex-M0+** only supports **full-descending mode** and it is not a configurable option here.
- There also **PUSH** and **POP** instructions available in ARM which always assume a **full descending** stack
  - This matches with the convention of stack space growing towards the lower address, towards the heap space
- The width of the stack is 4 bytes (32 bits) always. For every push operation the stack pointer is decremented by 4 before a new value is written into the stack

Arm-Cortex-M0+: Stack Implementation

- The processor uses a **full descending stack**.
  - This means the stack pointer indicates the last stacked item on the stack memory.



  - When the processor pushes a new item onto the stack, it decrements the stack pointer and then writes the item into the new memory location.
- The processor implements two (banked) stacks, the **main stack (MSP)** and the **process stack (PSP)**, with independent copies of the stack pointer.
  - The processor can either be in **Thread** or **Handler mode**.
- In Thread mode, the CONTROL register controls whether the processor uses the MSP or PSP. In Handler mode, the processor always uses the MSP.

Processor mode	Used to execute	Optional privilege level for software execution	Stack used
Thread	Applications	Privileged or unprivileged[a]	Main stack or process stack[a].
Handler	Exception handlers	Always privileged	Main stack.

a. See *CONTROL register* on page 2-8.

Note: Different modes will be covered later.

Ref:Ref4_ARM-Cortex-M0+ Devices Generic User Guide, Section 2.1.2 Stacks

Summary

- Arm- Cortex-M0+
  - Register Set
  - Stack Implementation in ARM
  - Different modes
  - Full Descending Stack (Cortex-M0+)