Architecture

HanSeong Kim

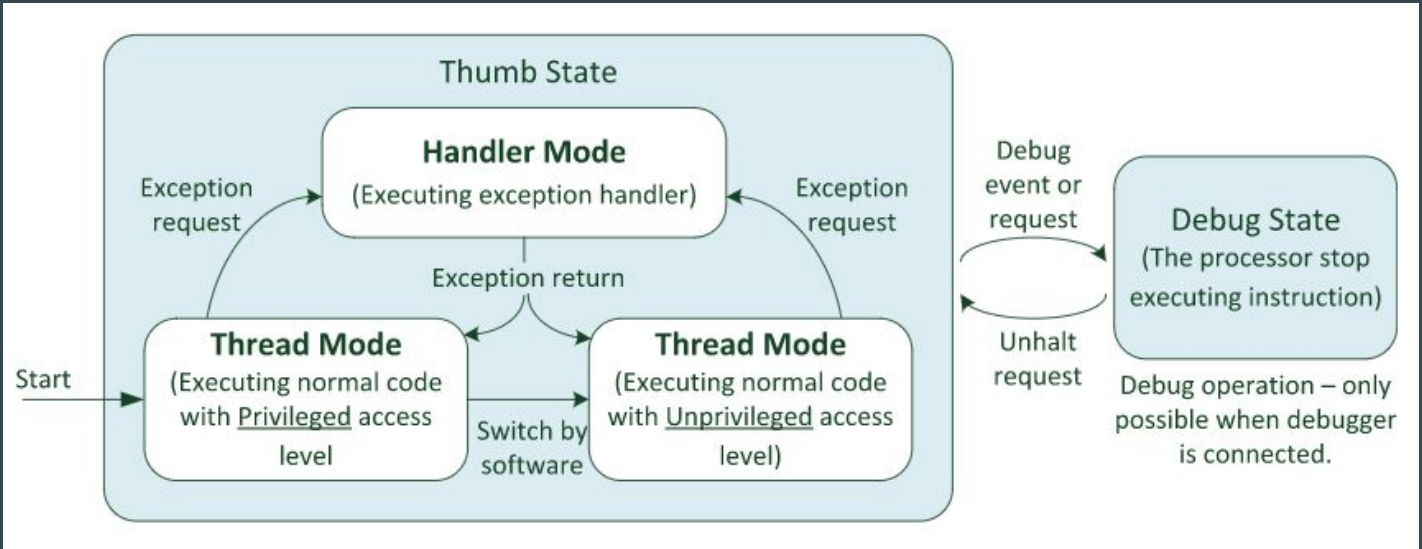
Introduction to the architecture

- Cortex-M3 : ARMv7-M
- Cortex-M4 : ARMv7E-M

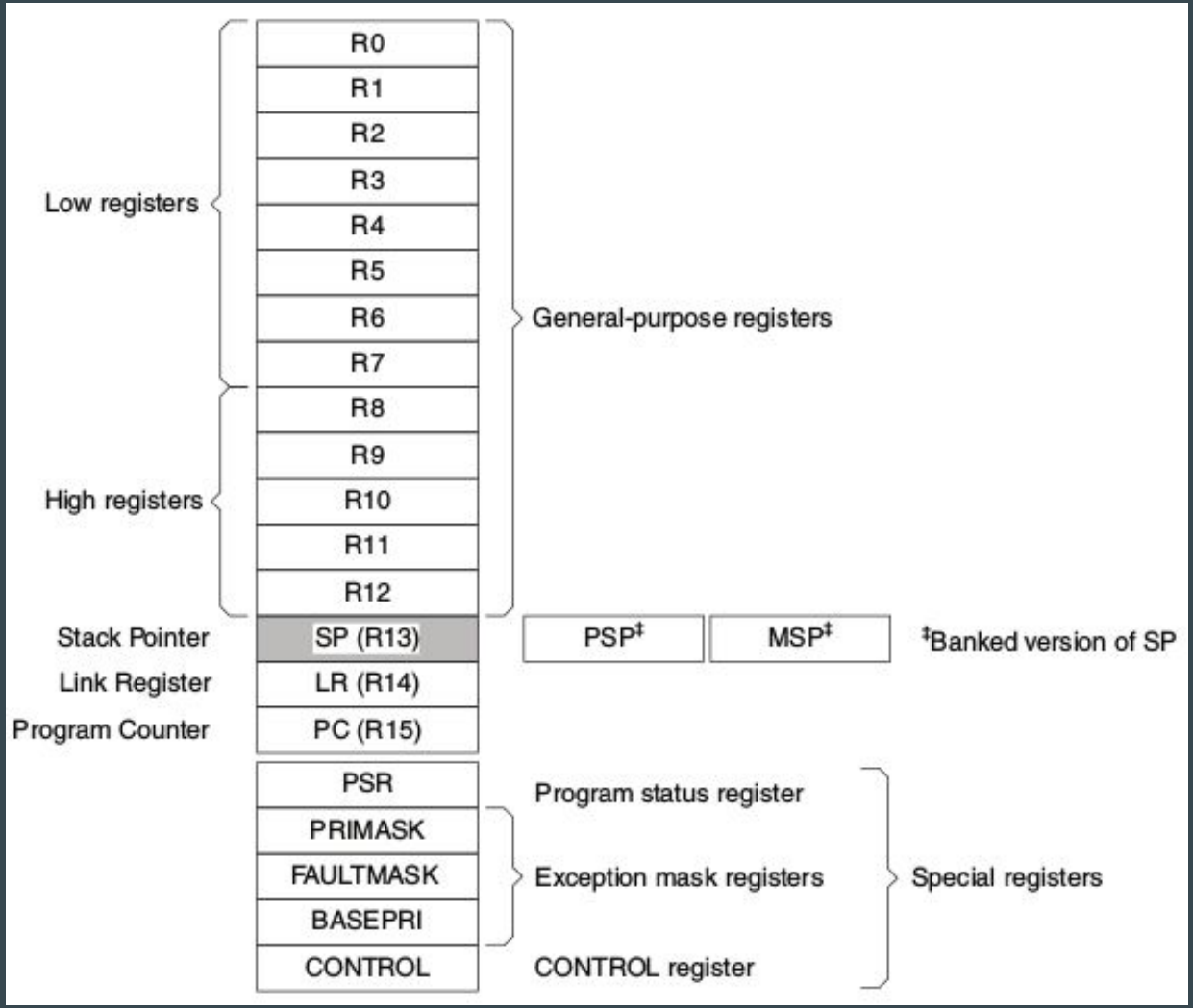
- Programmer’s model of the processor
  - Software development point of view

Operation and states

- Operation states : Debug state, Thumb state (No ARM state, No ARM instruction sets)
- Operation modes : Handler mode, Thread mode



Registers


R0
R1
R2
R3
R4
R5
R6
R7
R8
R9
R10
R11
R12
SP (R13)
LR (R14)
PC (R15)
PSR
PRIMASK
FAULTMASK
BASEPRI
CONTROL
Low registers
High registers
General-purpose registers
Stack Pointer
Link Register
Program Counter
PSP‡
MSP‡
‡Banked version of SP
Program status register
Exception mask registers
CONTROL register
Special registers

Registers

- R0 - R14 : General Purpose Registers
- R0 - R7 : Low registers
  - Many 16 bit instructions can only access low registers only
- R8 - R12 : High registers
  - 32 bit instructions and a few 16 bit instructions (for example, MOVE)
- R13 : stack pointer
  - PUSH, POP operaion use this.
  - MSP (Main Stack Pointer) : default stack pointer.
  - PSP (Processor Stack Pointer) : used in Thread Mode. required in embedded OS is involved. the stack for the OS kernel and application tasks are separated
  - CONTROL : select stack pointer
- R14 : Link register
  - used for return address when calling a function
  - updated automatically when a function is calling
- R15 : program counter (PC)
  - read and write operation
  - When read operation, return a current instruction address + 4 (due to pipeline architecture)
  - When write PC, then jump to the address

Special Registers - Program Status Register

- MRS <reg>, <special_reg>; Read special register into register
- MSR <special_reg>, <reg>; write to special register
- Application PSR (APSR)
- Execution PSR (EPSR)
- Interrupt PSR (IPSR)

	31	30	29	28	27	26:25	24	23:20	19:16	15:10	9	8	7	6	5	4:0
APSR	N	Z	C	V	Q				GE*							
IPSR												Exception Number				
EPSR						ICI/IT	T			ICI/IT						

*GE is available in ARMv7E-M processors such as the Cortex-M4. It is not available in the Cortex-M3 processor.

	31	30	29	28	27	26:25	24	23:20	19:16	15:10	9	8	7	6	5	4:0
xPSR	N	Z	C	V	Q	ICI/IT	T		GE*	ICI/IT		Exception Number				

Special Registers - Program Status Register

- MRS r0, PSR ; Read the combined program status word
- MSR PSR, r0 ; Write combined program state word

Table 4.2 Bit Fields in Program Status Registers

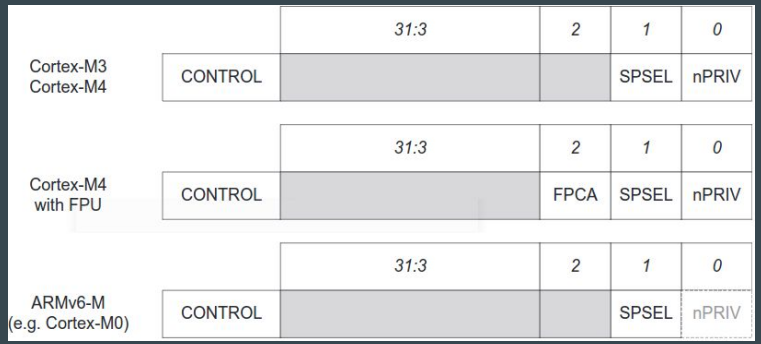
Bit	Description
N	Negative flag
Z	Zero flag
C	Carry (or NOT borrow) flag
V	Overflow flag
Q	Sticky saturation flag (not available in ARMv6-M)
GE[3:0]	Greater-Than or Equal flags for each byte lane (ARMv7E-M only; not available in ARMv6-M or Cortex®-M3).
ICI/IT	Interrupt-Continuable Instruction (ICI) bits, IF-THEN instruction status bit for conditional execution (not available in ARMv6-M).
T	Thumb state, always 1; trying to clear this bit will cause a fault exception.
Exception Number	Indicates which exception the processor is handling.

Special Registers - PRIMASK, FAULTMASK and BASEPRI

- PRIMASK : When set, blocks all exceptions (except NMI, HardFault exception)
  - To disable all interrupts for a time critical process
- FAULTMASK : PRIMASK + HardFault exception
  - For fault handling codes to supress triggering further faults
- BASEPRI : Masks exceptions based on priority level

```
x = __get_BASEPRI();     // Read BASEPRI register
x = __get_PRIMARK();     // Read PRIMASK register
x = __get_FAULTMASK();   // Read FAULTMASK register
__set_BASEPRI(x);        // Set new value for BASEPRI
__set_PRIMASK(x);      // Set new value for PRIMASK
__set_FAULTMASK(x);    // Set new value for FAULTMASK
__disable_irq();       // Set PRIMASK, disable IRQ
__enable_irq();        // Clear PRIMASK, enable IRQ
```

Special Registers - CONTROL



		31:3	2	1	0
Cortex-M3 Cortex-M4	CONTROL			SPSEL	nPRIV
Cortex-M4 with FPU	CONTROL		FPCA	SPSEL	nPRIV
ARMv6-M (e.g. Cortex-M0)	CONTROL			SPSEL	nPRIV

```
MRS   r0, CONTROL ; Read CONTROL register into R0
MSR   CONTROL, r0 ; Write R0 into CONTROL register
```

```
int in_privileged(void)
{
if (__get_IPSR() != 0) return 1; // True
else
 if ((__get_CONTROL() & 0x1)==0) return 1; // True
 else return 0; // False
}
```

Table 4.3 Bit Fields in CONTROL Register

Bit	Function
nPRIV (bit 0)	Defines the privileged level in Thread mode: When this bit is 0 (default), it is privileged level when in Thread mode. When this bit is 1, it is unprivileged when in Thread mode. In Handler mode, the processor is always in privileged access level.
SPSEL (bit 1)	Defines the Stack Pointer selection: When this bit is 0 (default), Thread mode uses Main Stack Pointer (MSP). When this bit is 1, Thread mode uses Process Stack Pointer (PSP). In Handler mode, this bit is always 0 and write to this bit is ignored.
FPCA (bit 2)	Floating Point Context Active – This bit is only available in Cortex-M4 with floating point unit implemented. The exception handling mechanism uses this bit to determine if registers in the floating point unit need to be saved when an exception has occurred. When this bit is 0 (default), the floating point unit has not been used in the current context and therefore there is no need to save floating point registers. When this bit is 1, the current context has used floating point instructions and therefore need to save floating point registers. The FPCA bit is set automatically when a floating point instruction is executed. This bit is clear by hardware on exception entry. There are several options for handling saving of floating point registers. This will be covered in Chapter 13.

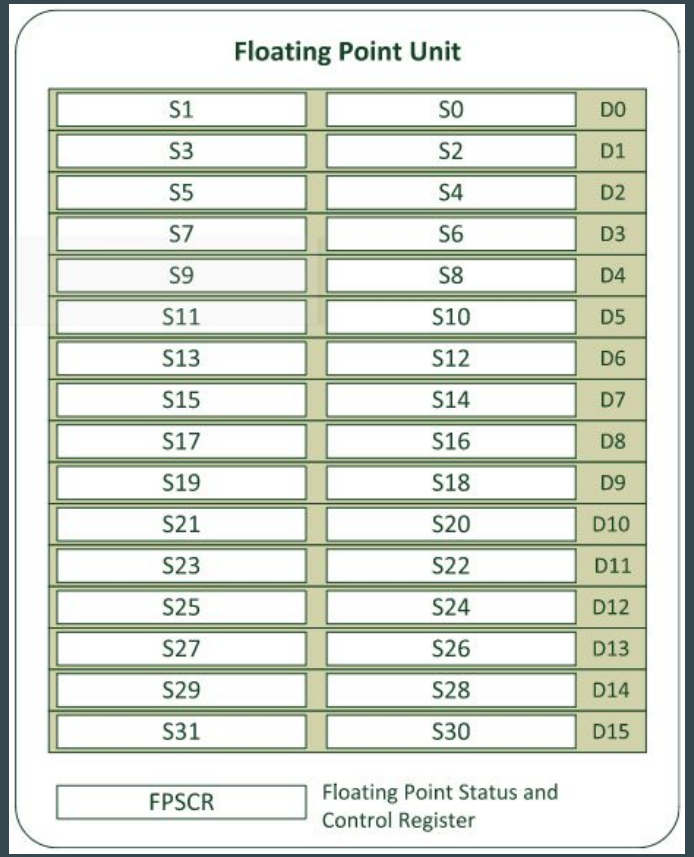
Floating Point Registers

- FPSCR : Floating Point Status and Control Register
- Single precision operation
- But can transfer double precision data
- CPACR : Coprocessor Access Control Register
  - Enable or disable Floating Point Unit
- Table 4.5 (92 page)

```
SCB->CPACR |= 0xF << 20; // Enable full access to the FPU
```

	31:24	23:22	21:20	19:0
CPACR	Reserved	CP11	CP10	Reserved

Bit field encoding:
00 – Access denied
01 – Privileged access only
10 – Reserved (unpredictable)
11 – Full accesses



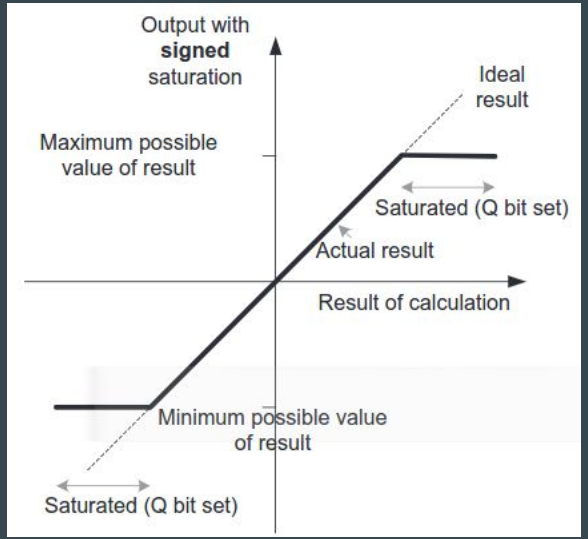
APSR - Integer status flags

Table 4.6 ALU Flags on the Cortex-M Processors

Flag	Descriptions
N (bit 31)	Set to bit[31] of the result of the executed instruction. When it is "1," the result has a negative value (when interpreted as a signed integer). When it is "0," the result has a positive value or equal zero.
Z (bit 30)	Set to "1" if the result of the executed instruction is zero. It can also be set to "1" after a compare instruction is executed if the two values are the same.
C (bit 29)	Carry flag of the result. For unsigned addition, this bit is set to "1" if an unsigned overflow occurred. For unsigned subtract operations, this bit is the inverse of the borrow output status. This bit is also updated by shift and rotate operations.
V (bit 28)	Overflow of the result. For signed addition or subtraction, this bit is set to "1" if a signed overflow occurred.

APSR - Q status flag

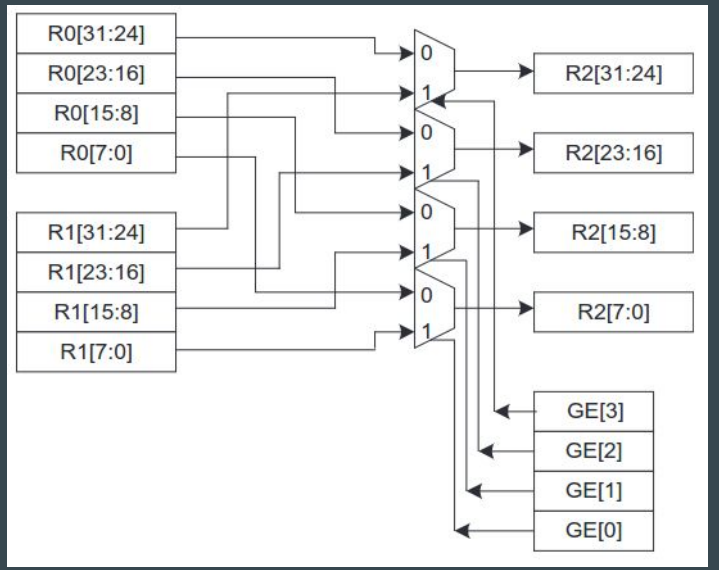
- Q status flag is used to indicate a saturation
- Saturation arithmetic/adjustment does not clear this bit
- ARMv7-M not available in ARMv6-M
- Saturation arithmetic starts with “Q”
  - ex) QADD16



APSR - GE bits

- The “Greater-Equal” (GE) is a 4-bit wide field in the APSR in the Cortex-M4, not available in the Cortex-M3 processor.
- It is updated by a number of SIMD instructions
- Table 4.8 GE flags

UADD16	If lower half-word result >= 0x10000 then GE[1:0] = 2’b11 else GE[1:0] = 2’b00 If upper half-word result >= 0x10000 then GE[3:2] = 2’b11 else GE[3:2] = 2’b00

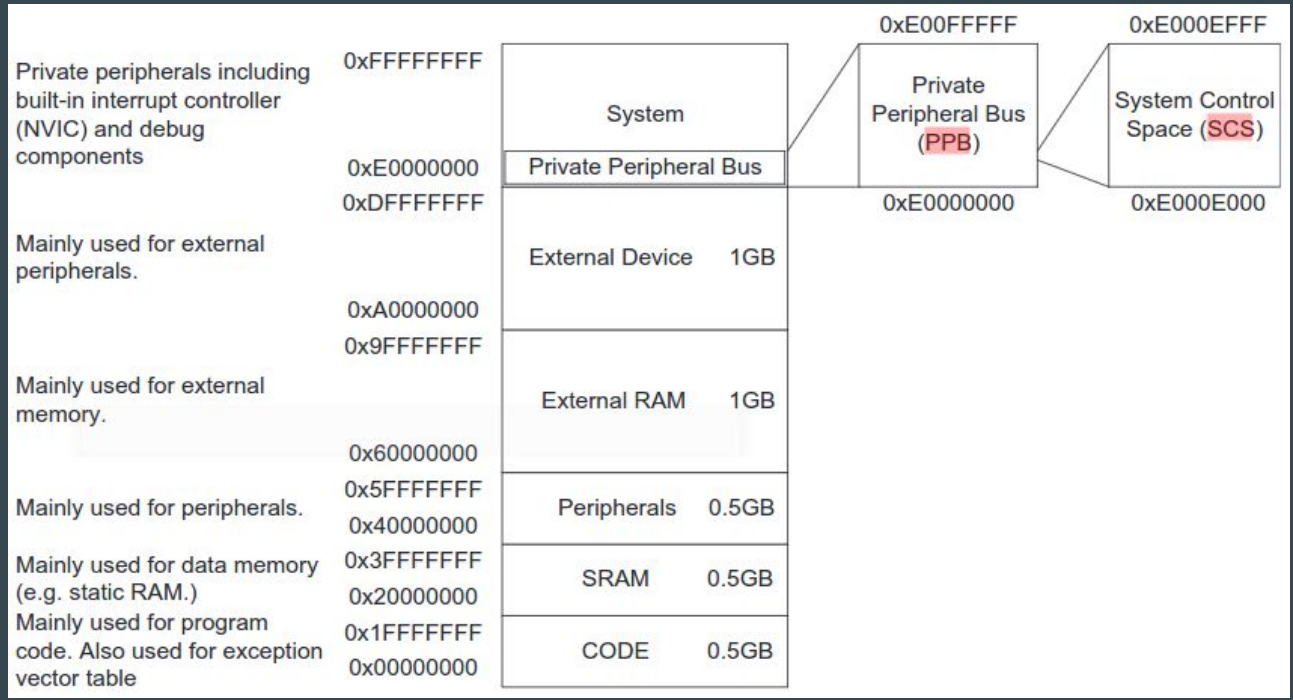


Memory System Features

- 4GB linear address space
- Architecturally defined memory map
- Support for little endian and big endian memory systems
- Bit band accesses (optional) : two regions are addressable via two bit-bands region
- Write buffer : the transfer can be buffered -> increase execution speed
- Memory Protection Unit (Optional)
- Unaligned transfer support : For ARMv7-M

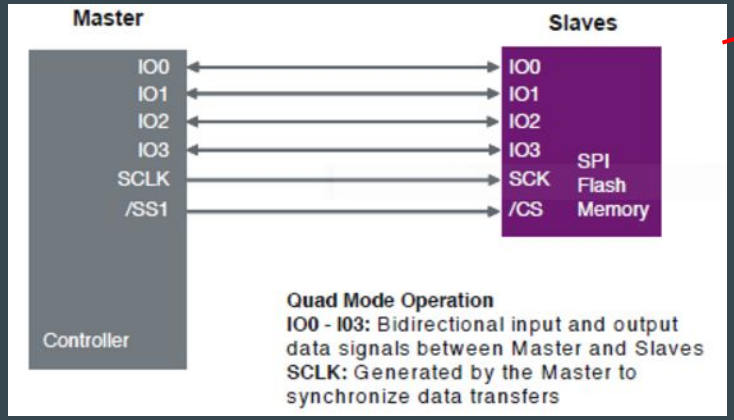
Memory Map

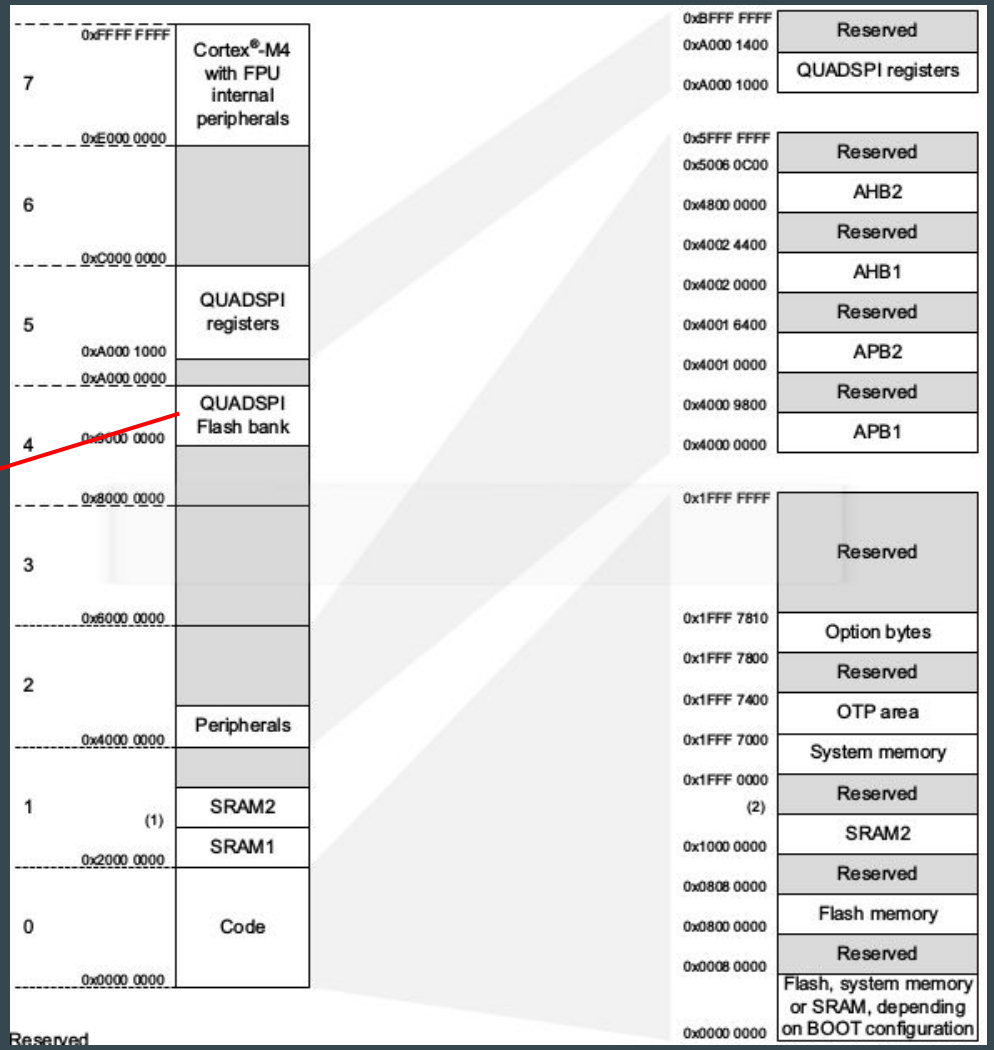
- Memory map arrangement is same on all Cortex-M
- For better software reusability



Memory Map (STM3L432KC)

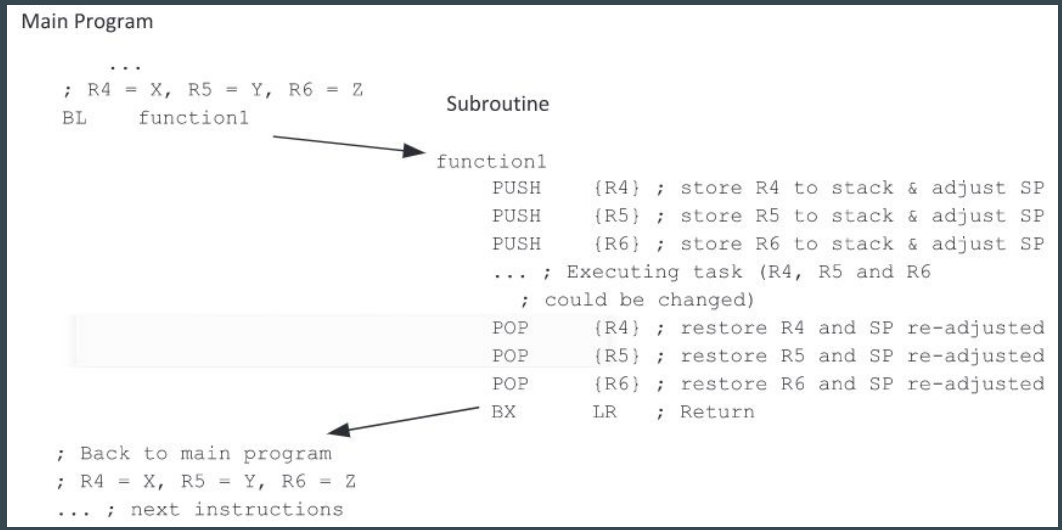
-





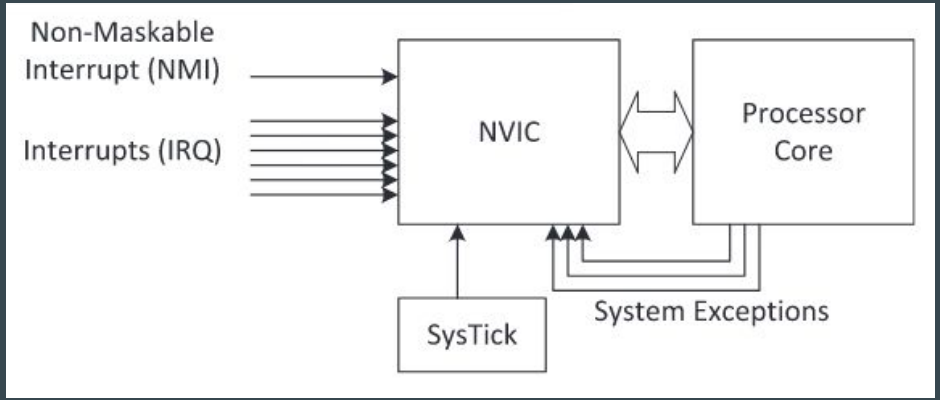
Stack memory

- R13 : Stack pointers
- What for
  - temporary storage
  - passing information for functions or subroutines
  - storing local variables
  - hold process status and register values in case of exceptions
- Main Stack Pointer (MSP)
- Process Stack Pointer (PSP)
- CONTROL' SPSEL



What are exceptions ?

- Exceptions are events that cause changes to program flow
- Interrupts are subset of exceptions
- Exceptions are processed by NVIC
- No FIQ in Cortex-M (12clocks)



Exception numbers

Table 4.9 Exception Types

Exception Number	CMSIS Interrupt Number	Exception Type	Priority	Function
1	—	Reset	−3 (Highest)	Reset
2	−14	NMI	−2	Non-Maskable interrupt
3	−13	HardFault	−1	All classes of fault, when the corresponding fault handler cannot be activated because it is currently disabled or masked by exception masking
4	−12	MemManage	Settable	Memory Management fault; caused by MPU violation or invalid accesses (such as an instruction fetch from a non-executable region)
5	−11	BusFault	Settable	Error response received from the bus system; caused by an instruction prefetch abort or data access error
6	−10	Usage fault	Settable	Usage fault; typical causes are invalid instructions or invalid state transition attempts (such as trying to switch to ARM state in the Cortex-M3)
7–10	—	—	—	Reserved
11	−5	SVC	Settable	Supervisor Call via SVC instruction
12	−4	Debug monitor	Settable	Debug monitor – for software based debug (often not used)
13	—	—	—	Reserved
14	−2	PendSV	Settable	Pendable request for System Service
15	−1	SYSTICK	Settable	System Tick Timer
16–255	0–239	IRQ	Settable	IRQ input #0–239

Nested Vector Interrupt Controller (NVIC)

- Flexible exception and interrupt management
  - NVIC can handle pulsed interrupt type and level triggered interrupt type
- Nested exception/interrupt support
  - Preemption
- Vectored exception/interrupt entry
  - The Cortex-M processors automatically locate the starting point of the exception handler from a vector table in the memory. cf) ARM7TDMI does this by software
- Interrupt masking
  - PRIMASK, BASEPRI

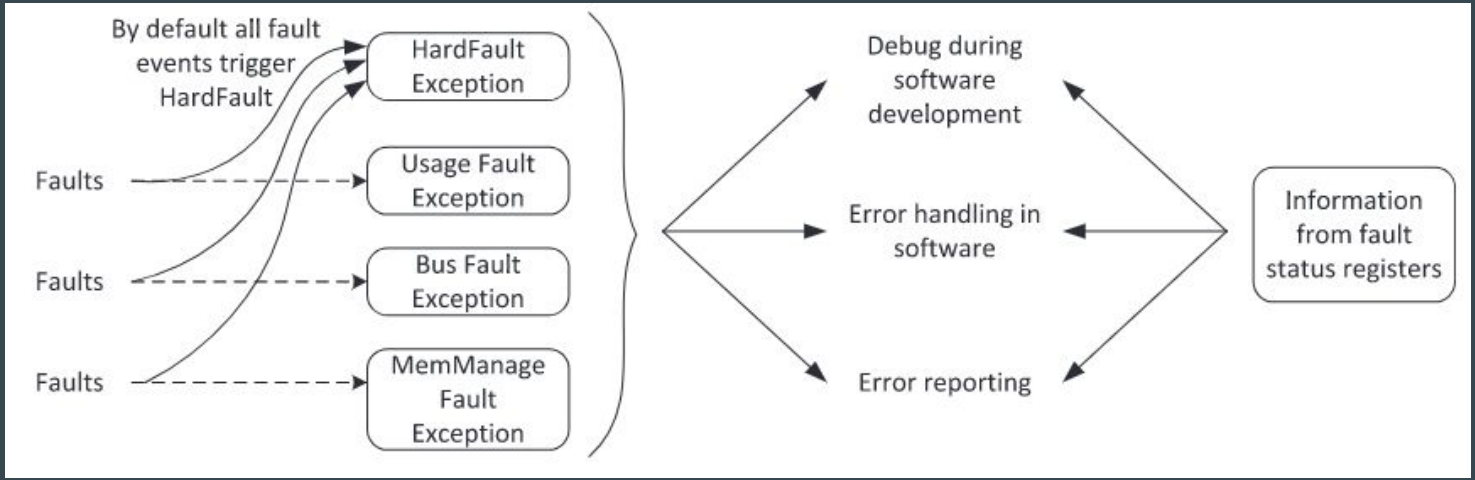
Vector Table

- The vector table is relocatable and the relocation is controlled by a programmable register in the NVIC called the Vector Table Offset Register (VTOR).
- After reset, the VTOR is reset to 0; therefore, the vector table is located at address 0x0 after reset.

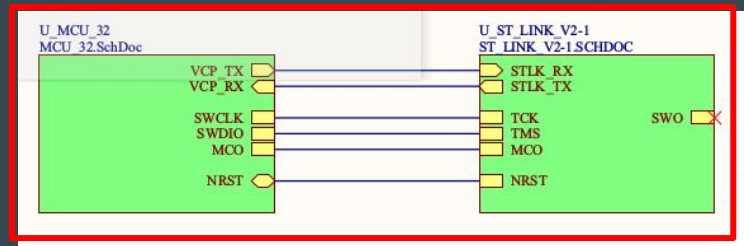
Exception Type	CMSIS Interrupt Number	Address Offset	Vectors
18 - 255	2 - 239	0x48 – 0x3FF	IRQ #2 - #239 [1]
17	1	0x44	IRQ #1 [1]
16	0	0x40	IRQ #0 [1]
15	-1	0x3C	SysTick [1]
14	-2	0x38	PendSV [1]
NA	NA	0x34	Reserved
12	-4	0x30	Debug Monitor [1]
11	-5	0x2C	SVC [1]
NA	NA	0x28	Reserved
NA	NA	0x24	Reserved
NA	NA	0x20	Reserved
NA	NA	0x1C	Reserved
6	-10	0x18	Usage fault [1]
4	-11	0x14	Bus Fault [1]
4	-12	0x10	MemManage Fault [1]
3	-13	0x0C	HardFault [1]
2	-14	0x08	NMI [1]
1	NA	0x04	Reset [1]
NA	NA	0x00	Initial value of MPS

Fault Handling

- By default the Bus Fault, Usage Fault, and Memory Management Fault are disabled and all fault events trigger the HardFault exception.
- The HardFault exception is always enabled.



Debug


U_MCU_32
MCU_32.SchDoc
VCP_TX
VCP_RX
SWCLK
SWDIO
MCO
NRST
U_ST_LINK_V2-1
ST_LINK_V2-1.SCHDOC
STLK_RX
STLK_TX
TCK
TMS
MCO
NRST
SWO

JTAG connection
nTRST
TCK
TDI
TMS
TDO
Serial-Wire connection
not used
Serial-Wire clock
not used
Serial-Wire data
not used
KEIL
Microcontroller
Development Kit
USB
Run
Com
USB
KEIL™
An ARM® Company
ULINK 2
Designed with Real-Time Library
In-Circuit Debugger
Flat cable
IDC
connector
ARM
Cortex-M

Reset and reset sequence

- Power on reset : reset everything in the microcontroller. This includes the processor and its debug support component and peripherals.
- System reset : reset just the processor and peripherals, but not the debug support component of the processor.
- Processor reset : reset the processor only.