2013 ARM Design Contest

STM32F4x Technical Training

July 2013 V1.0

life.augmented

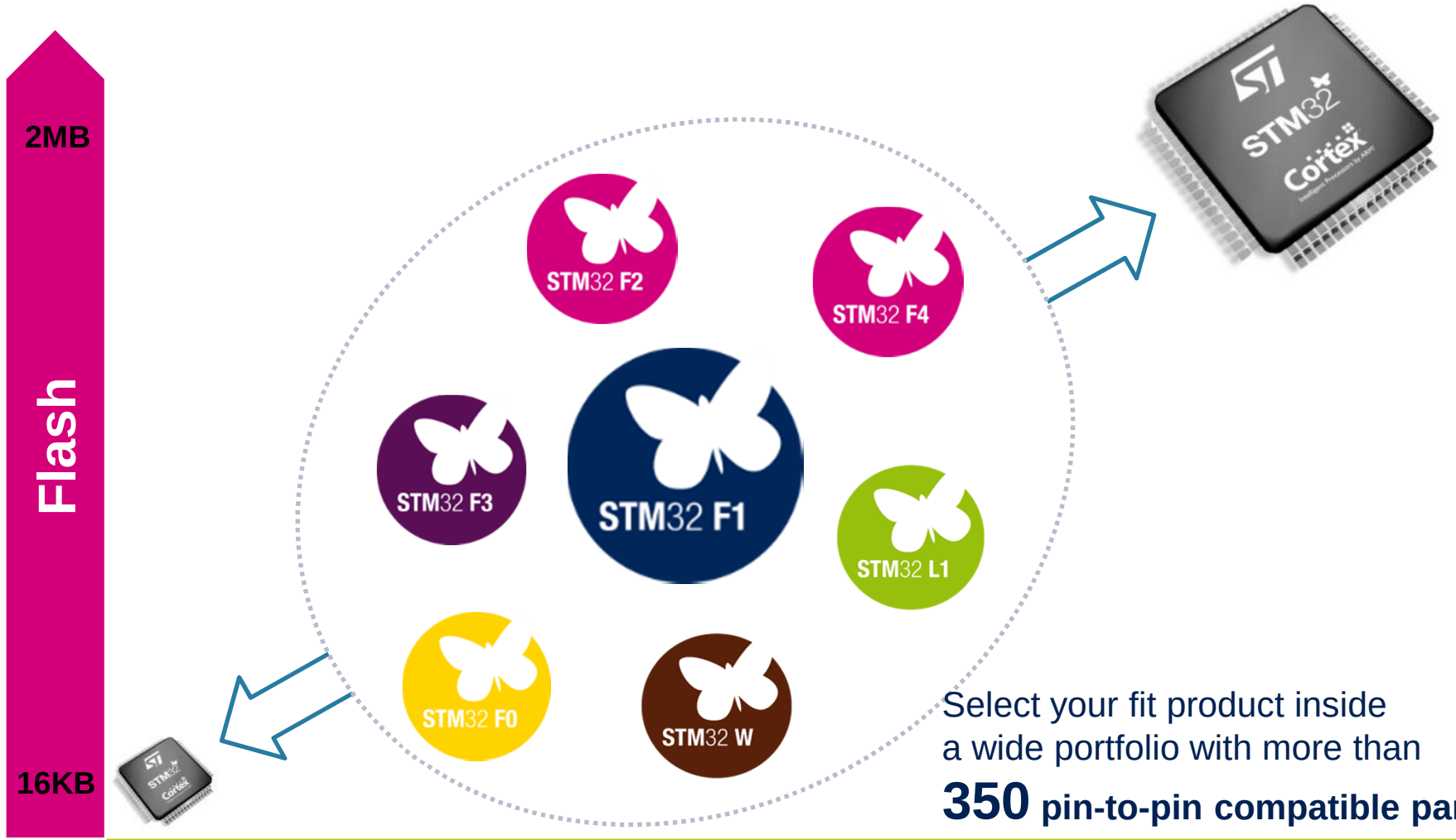
Agenda

- STM32 Products
- STM32F4 Block Diagram & Features
- Embedded Flash
- Power Control (PWR)
- Reset and Clock Control (RCC)
- External interrupt/Event (EXTI), GPIO.
- Direct Memory Access (DMA).
- Analog to Digital (ADC).
- Flexible Static Memory Controller (FSMC)
- STM32Fx Firmware.
- Hands-on.

STM32 32-bit ARM Cortex MCUs

life.augmented

STM32 platform effect



STM32 Cortex™-M microcontrollers

Cortex-M4

High-perf. DSP MCUs
168MHz / 210DMIPS
Up to 2MB Flash
Up to 192+64KB SRAM

Analog & DSP MCUs
72MHz / 94DMIPS (from CC-SRAM)
64 to 256KB Flash
Up to 40+8KB SRAM

32-bit/DSC applications

Cortex-M3

Mainstream MCUs
24 to 72MHz / 61DMIPS
16KB to 1MB Flash
Up to 96KB SRAM

Ultra-low-power MCUs
32MHz / 33.3DMIPS
32 to 384KB Flash
Up to 48KB SRAM

High-performance MCUs
120MHz / 150DMIPS
256KB to1MB Flash
Up to 128KB SRAM

Wireless MCUs
24MHz / 30DMIPS
64 to 128KB Flash
Up to 8KB SRAM

16/32-bit applications

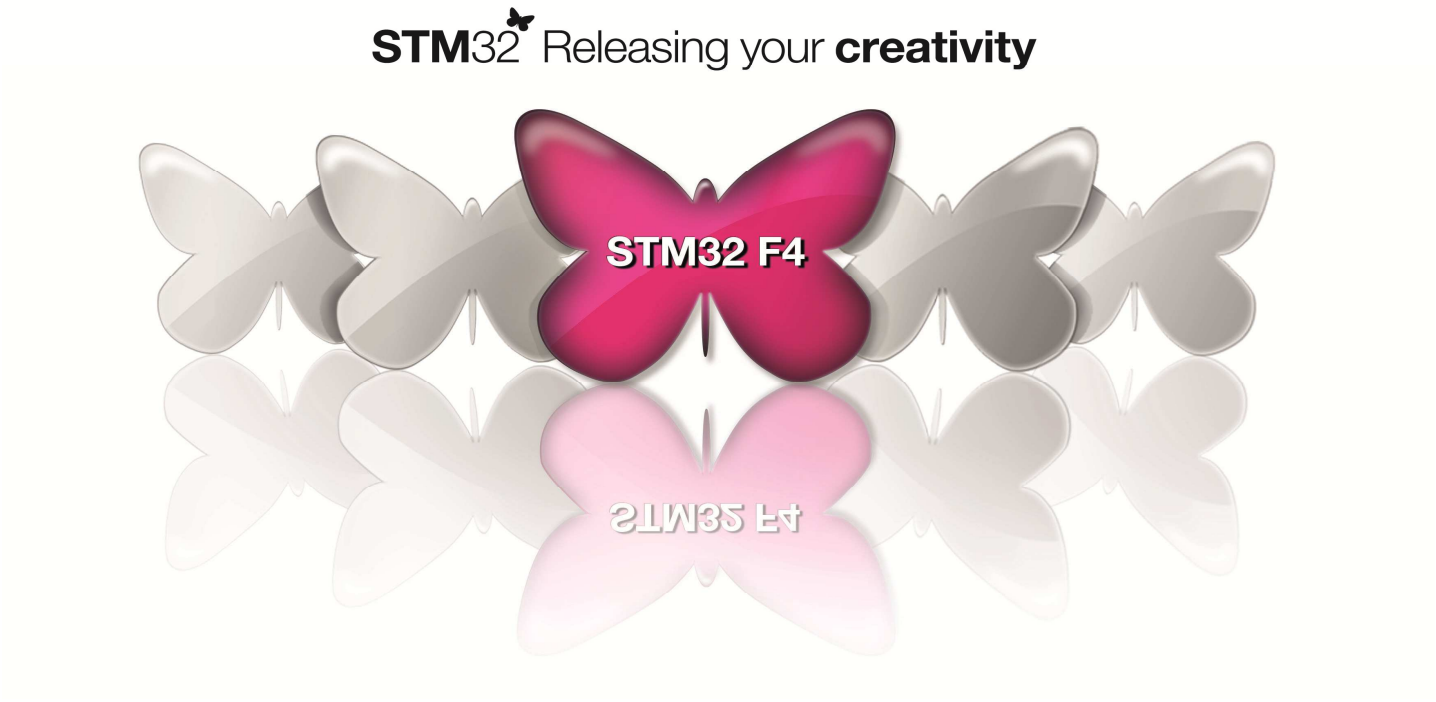
Cortex-M0

Entry-level MCUs
48MHz / 38DMIPS
16 to 128KB Flash
Up to 20KB SRAM

8/16-bit applications

STM32 F4 series

High-performance Cortex™-M4 MCU



life.augmented

ARM Cortex M4 Core

ARM

Cortex

Low-Power Leadership from ARM

FPU

Single precision
Ease of use
Better code efficiency
Faster time to market
Eliminate scaling and saturation
Easier support for meta-language tools

MCU

Ease of use of C programming
Interrupt handling
Ultra-low power

Cortex-M4

DSP

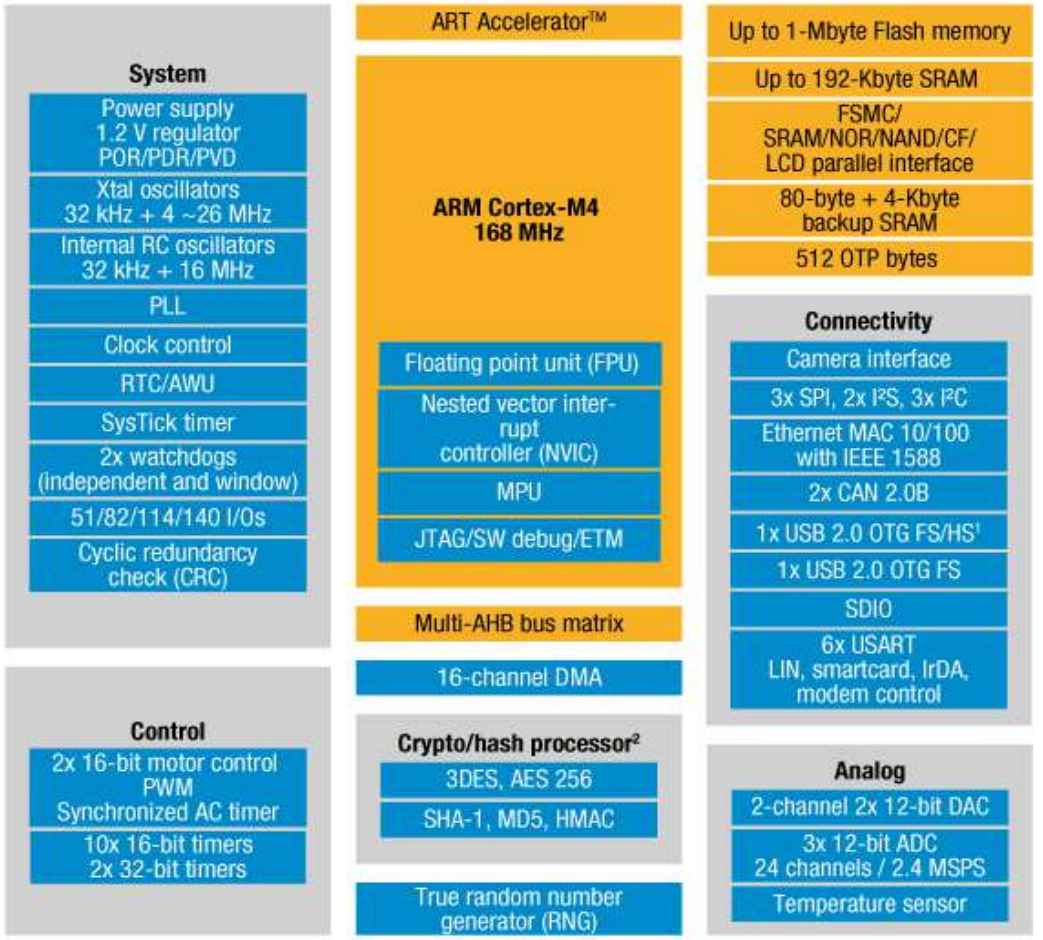
Harvard architecture
Single-cycle MAC
Barrel shifter

life.augmented

STM32 F4 block diagram 1/2

Feature highlight

- 168 MHz Cortex-M4 CPU
  - Floating point unit (FPU)
  - ART Accelerator ™
  - Multi-level AHB bus matrix
- 1-Mbyte Flash, 192-Kbyte SRAM
- 1.7 to 3.6 V supply
- RTC: <1 µA typ, sub second accuracy
- 2x full duplex I²S
- 3x 12-bit ADC 0.41 µs/2.4 MSPS
- 168 MHz timers

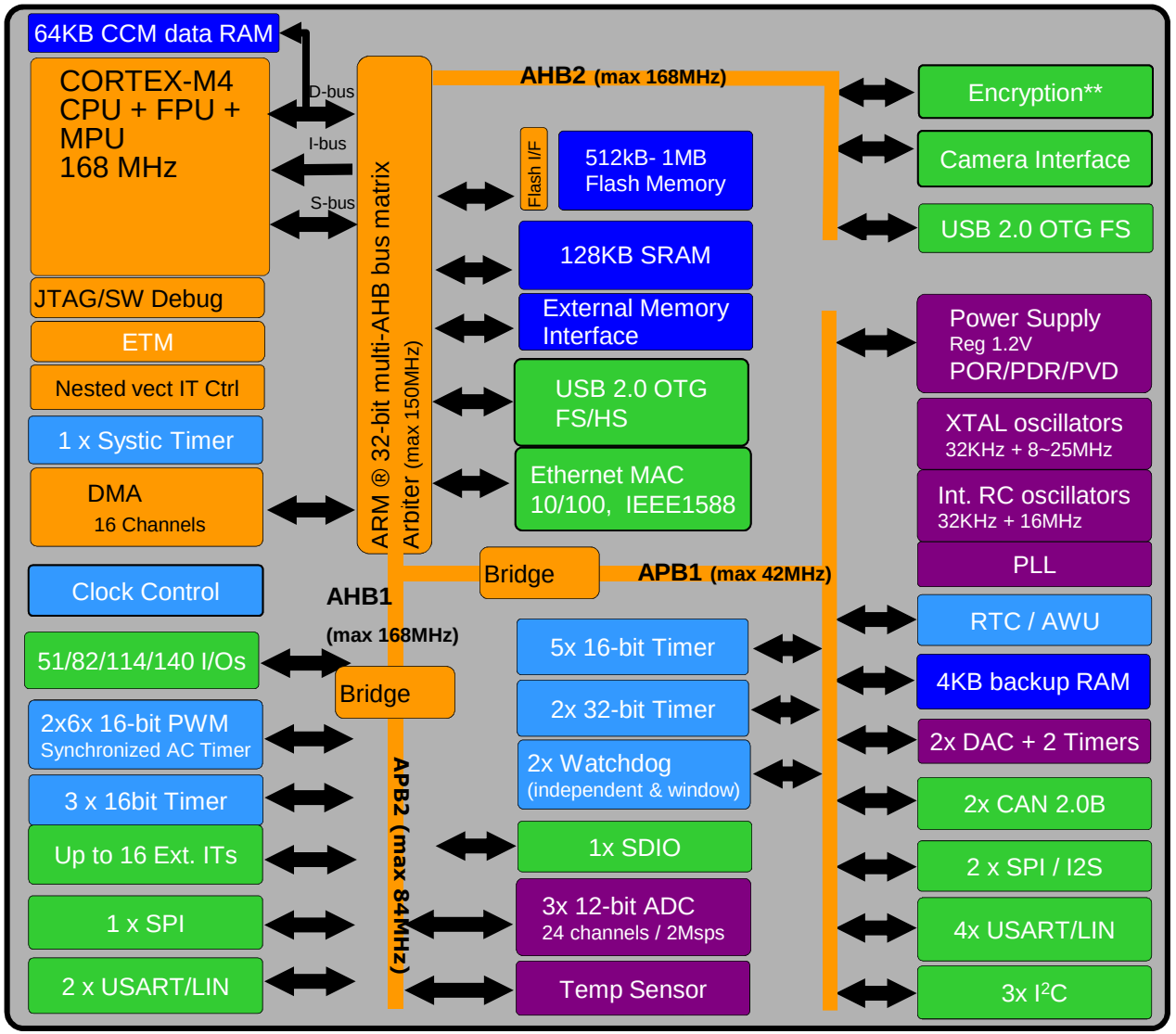


Notes:
1. HS requires an external PHY connected to the ULPI interface
2. Crypto/hash processor on STM32F417 and STM32F415

life.augmented

STM32F4xx Block Diagram 2/2

- **Cortex-M4 w/ FPU, MPU and ETM**
- **Memory**
  - Up to 1MB Flash memory
  - 192KB RAM including 64KB CCM data RAM
  - FSMC up to 60MHz
- **New application specific peripherals**
  - USB OTG HS w/ ULPI interface
  - Camera interface
  - HW Encryption**: DES, 3DES, AES 256-bit, SHA-1 hash, RNG.
- **Enhanced peripherals**
  - USB OTG Full speed
  - ADC: 0.416µs conversion/2.4Msps, up to 7.2Msps in interleaved triple mode
  - ADC/DAC working down to 1.8V
  - Dedicated PLL for I2S precision
  - Ethernet w/ HW IEEE1588 v2.0
  - 32-bit RTC with calendar
  - 4KB backup SRAM in VBAT domain
  - Pure 1% RC
  - 2 x 32bit and 8 x 16bit Timers
  - high speed USART up to 10.5Mb/s
  - high speed SPI up to 37.5Mb/s
- **RDP (JTAG fuse)**
- **More I/Os in UFBGA 176 package**

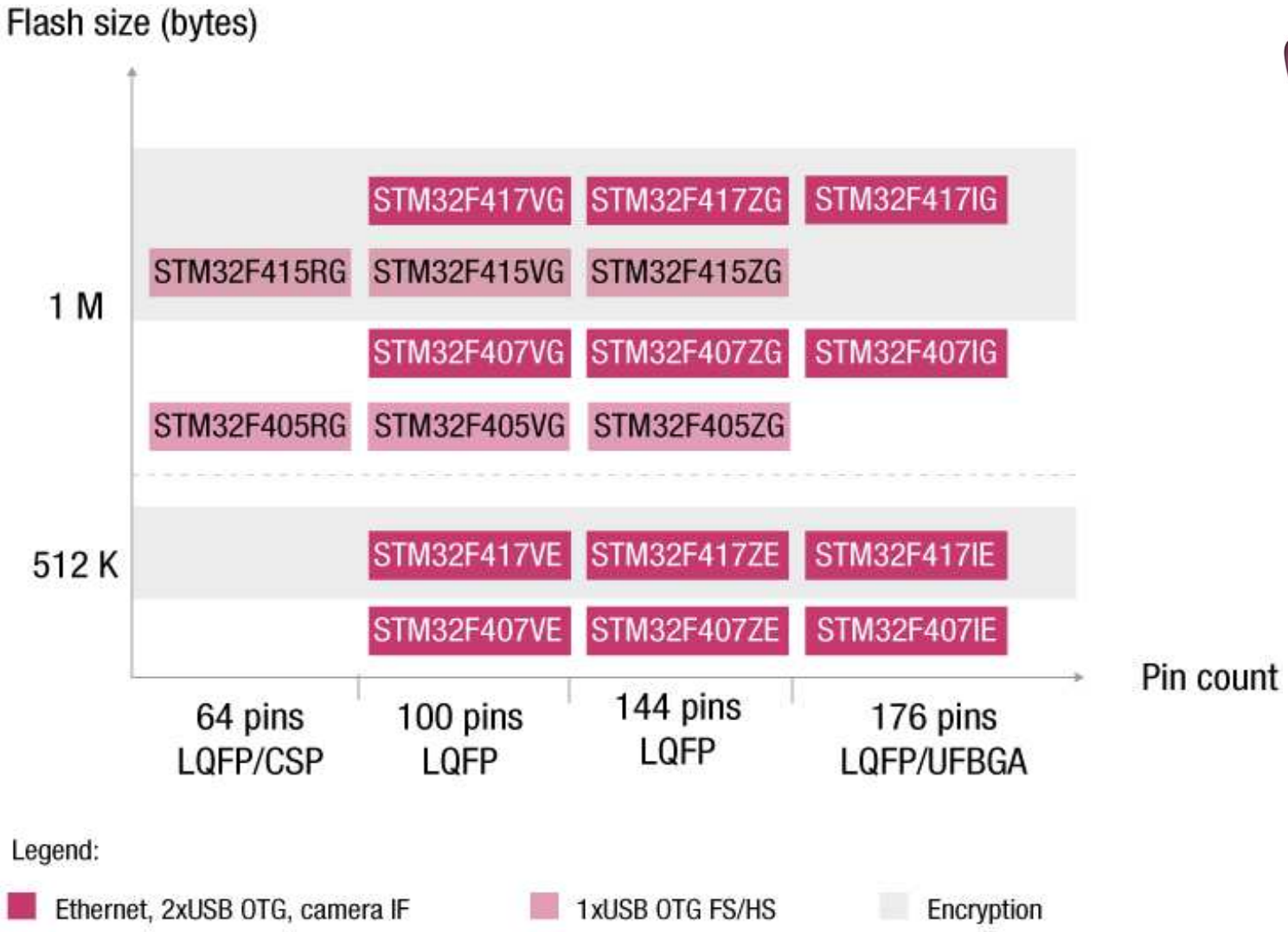


HS requires an external PHY connected to ULPI interface,
** Encryption is only available on STM32F415 and STM32F417

STM32 F4 key features

life.augmented

STM32 F4 portfolio



life.augmented

System Peripherals

EMBEDDED FLASH

life.augmented

System Architecture - Bootloader

BOOT Mode Selection Pins		Boot Mode	Aliasing
BOOT1	BOOT0		
x	0	**Flash memory**	Main Flash memory is selected as boot space
0	1	**System memory**	System memory is selected as boot space
1	1	**Embedded SRAM**	Embedded SRAM is selected as boot space

- The Bootloader supports
  - USART1(PA9/PA10)
  - USART3(PC10/PC11 or PB10/PB11)
  - CAN2(PB5PB13)
  - USB OTG FS in Device mode (PA11/PA12) through DFU (device firmware upgrade)

  Note
  - The DFU/CAN may work w/ different value of external quartz in the range of 4-26 MHz, and the USART uses the internal HSI
  - This Bootloader uses the same USART, CAN and DFU protocols as for STM32F2xx/STM32F10x

Flash Features Overview

- Up to 1 Mbytes
- 128 bits wide data read
- Byte, half-word, word and double word write
- 512 Bytes One Time Programmable (OTP)
- Sector and mass erase
  - 32-bit Word Program time: 12µs(Typ)
  - Sector Erase time:
    - 16KB: 400ms(Typ)
    - 64KB: 700ms(Typ)
    - 128KB: 1s(Typ)
    - 1MB: around 2s(Typ)
- 10K Cycles by sector / 20 years retention
- Protections
  - 2 RDP levels/sector Write protection

Block	Name	Block base address	Size
Main Memory	Sector 0	0x0800 0000 - 0x0800 3FFF	16 Kbyte
	. . .	. . .	. . .
	Sector 3	0x0800 C000 - 0x0800 FFFF	16 Kbyte
	Sector 4	0x0801 0000 - 0x0801 FFFF	64 Kbyte
	Sector 5	0x0802 0000 - 0x0803 FFFF	128 Kbyte
	. . .	. . .	. . .
	Sector 11	0x080E 0000 - 0x080F FFFF	128 Kbyte
System memory		0x1FFF 0000 - 0x1FFF 77FF	30 Kbyte
OTP		0x1FFF 7800 - 0x1FFF 7A0F	528 Bytes
Option bytes		0x1FFF C000 - 0x1FFF C00F	16 bytes

Flash Interface features Overview

- Flash interface Features
  - Read Interface
    - Instruction prefetch
    - Instruction cache: 64 cache lines of 128 bits
    - Data cache: 8 cache lines of 128 bits
  - Flash program/Erase operations
    - Program: Byte, Half word, word and double word
    - Sector erase and Mass erase
  - Types of Protection
    - Read Protection
      - Level0
      - Level1
      - Level2 (JTAG fuse)
    - Write Protection
  - Option Bytes

life.augmented

Flash Operations (1/3)

- After reset the Flash Control Register is protected, an unlocking sequence should be performed (write of 2 key values) to unlock the Flash

- The Flash can be programmed, depending on the supply voltage, with:
  - 8 bits: PSIZE = 00
  - 16 bits: PSIZE = 01
  - 32 bits: PSIZE = 10
  - 64 bits: PSIZE = 11 (with an external 9v supply on boot0 pin).

- Flash can be sector or mass erased

- The Read access can be performed with the following configuration:
  - Latency: Number of wait state for a read operation
  - Instruction Prefetch : For faster linear CPU execution
  - Instruction Cache: To limit the time lost due to jumps, it is possible to retain 64 lines of 128 bits in an instruction cache memory.
  - Data Cache: This feature work like the instruction cache memory, but the retained data size is limited to 8 rows of 128 bits.

Flash Operations (2/3)

- Relation between CPU clock frequency and Flash memory read time

Wait states(WS) (LATENCY)	HCLK clock frequency (MHz)			
	Voltage range 2.7 V - 3.6 V	Voltage range 2.4 V - 2.7 V	Voltage range 2.1 V - 2.4 V	Voltage range 1.8V - 2.1 V
0WS(1CPU cycle)	0 < HCLK <= 30	0 < HCLK <= 24	0 < HCLK <= 18	0 < HCLK <= 16
1WS(2CPU cycle)	30 < HCLK <= 60	24 < HCLK <= 48	18 < HCLK <= 36	16 < HCLK <= 32
2WS(3CPU cycle)	60 < HCLK <= 90	48 < HCLK <= 72	36 < HCLK <= 54	32 < HCLK <= 48
3WS(4CPU cycle)	90 < HCLK <= 120	72 < HCLK <= 96	54 < HCLK <= 72	48 < HCLK <= 64
4WS(5CPU cycle)	120 < HCLK <= 150	96 < HCLK <= 120	72 < HCLK <= 90	64 < HCLK <= 80
5WS(6CPU cycle)	150 < HCLK <= 168	120 < HCLK <= 144	90 < HCLK <= 108	80 < HCLK <= 96
6WS(7CPU cycle)		144 < HCLK <= 168	108 < HCLK <= 126	96 < HCLK <= 112
7WS(8CPU cycle)			126 < HCLK <= 144	112 < HCLK <= 128

Note: Latency when **VOS** bit in **PWR_CR** is equal to '**1**'

life.augmented

Flash Operations (3/3)

- The maximum program/erase parallelism is limited by the supply voltage and by whether the external Vpp supply is used or not.

	Program/Erase parallelism				
	Voltage range 2.7 V - 3.6 V	**Voltage range 2.4 V - 2.7 V**	**Voltage range 2.1 V - 2.4 V**	**Voltage range 1.8V - 2.1 V**	**External Vpp 9 V**
Maximum Parallelism	x32	x16	x8		x64
PSIZE[1:0]	10	01	00		11

Flash user configuration and specific block

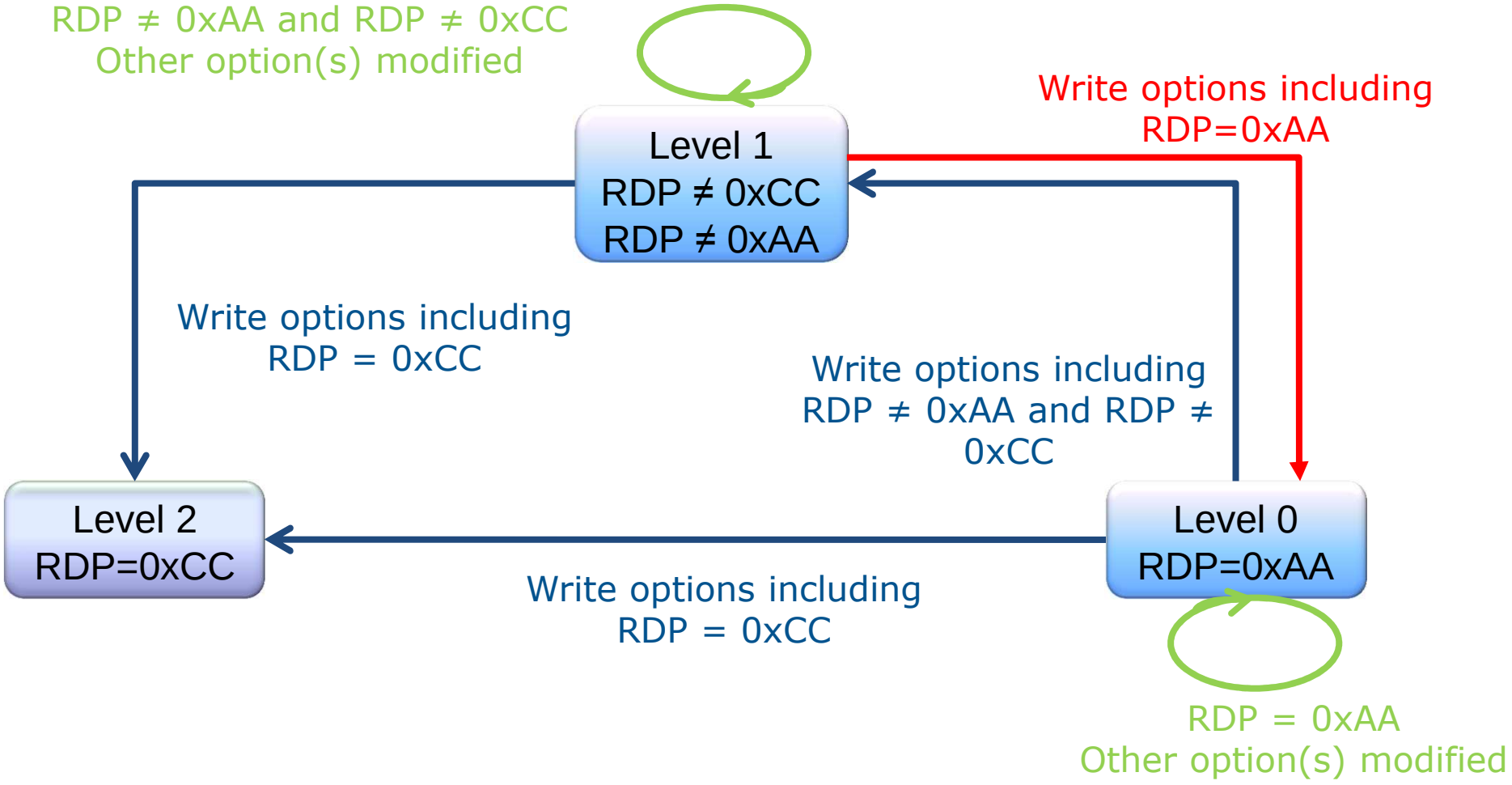
- The Flash user-specific block consists:
  - 30 Kbyte of system Memory
  - 512 Bytes OTP: one-time programmable
- The Flash user configuration block, is a 16 bytes size reserved for:
  - Read protection
  - Write protection
  - Software/hardware Watchdog
  - Reset when in stop/standby mode
  - BOR reset Level
- Flash option control register is s/w protected with key values
- To modify the Option bytes
  - Enter the 2 key register values
  - Write the desired option value in the FLASH_OPTCR
  - Then set the OPTSTRT bit and wait BSY to clear

Flash Read Protection (1/2)

- The user area in the Flash memory can be protected against read operations from an entrusted code.

  - ***Level 0***: Read protection disabled
    - Activated by writing 0xAA to the RDP byte register
    - All operations from/to the Flash memory or the backup SRAM are possible in any boot configuration.
  - ***Level 1***: Read protection enabled
    - Activated by writing any other value of 0xAA and 0xCC to the RDP byte register
    - In debug mode, any Flash memory or backup RAM accesses are disabled.
    - Debug is still permitted in system SRAM
  - ***Level 2***: Device is “locked-up”
    - Activated by writing 0xCC to the RDP option byte register
    - All protections provided by Level 1 are active
    - Debug features (CPU JTAG and single-wire) are disabled
    - User options can no longer be changed.
    - Boundary scan disabled

Flash Read Protection (2/2)



Option byte write (RDP level increase) includes: Option byte erase and New option byte programming
Option byte write (RDP level decrease) includes: Option byte erase, New option byte programming and Mass Erase
Option byte write (RDP level identical) includes : Option byte erase and New option byte programming

life.augmented

Flash Write Protection

- Each sector can be protected against unwanted write operations
  - 12 bits to write protect independently the 12 Flash sectors

- To enable/disable the write protection:
  - Write the desired option value in the FLASH_OPTCR register: nWRPi (0<=i<=11) should be set to 1 in case of write protection disable.
  - Set the option start bit (OPTSTRT) in the FLASH_OPTCR register: once this bit is set, the value of an option is automatically modified by first erasing the user configuration sector and then programming all the option bytes with the values contained in the FLASH_OPTCR register
  - Wait for the BSY bit to be cleared

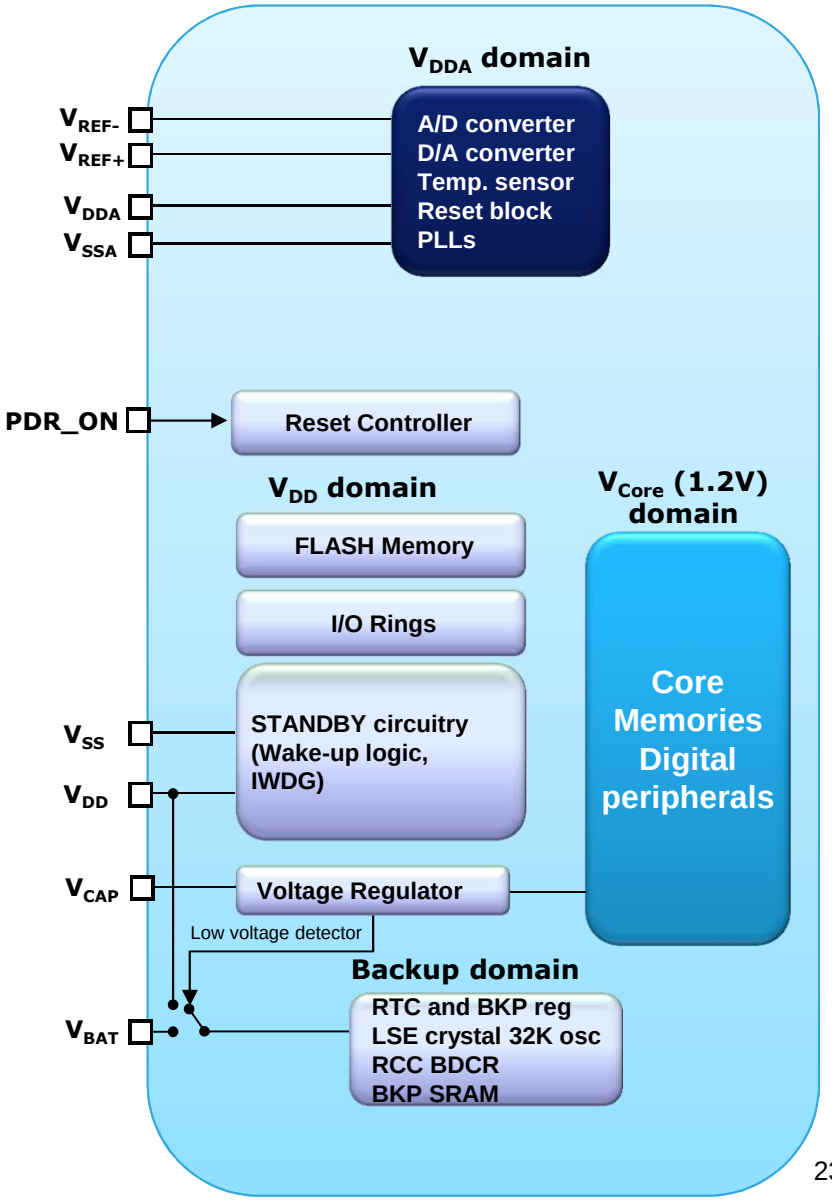
System Peripherals

POWER CONTROL (PWR)

life.augmented

Power Supply

- Power Supply Schemes
  - **V_{DD}** = 1.8 V to 3.6 V. External Power Supply for I/Os and the internal regulator. The supply voltage can drop to 1.7 when the PDR_ON is connected to VSS and the device operates in the 0 to 70° C.
  - **V_{DDA}** = 1.8 V to 3.6 V : External Analog Power supplies for ADC, DAC, Reset blocks, RCs and PLLs.
  - **V_{CAP}** = Voltage regulator external capacitors (also 1.2V supply in Regulator bypass mode)
  - **V_{BAT}** = 1.65 to 3.6 V: power supply for Backup domain when V_{DD} is not present.
  - Power pins connection:
    - V_{DD} and V_{DDA} must be connected to the same power source
    - V_{SS}, V_{SSA} must be tight to ground
    - $2.4V \leq V_{REF+} \leq V_{DDA}$ when $V_{DDA} \geq 2.4$
    - $V_{REF+} = V_{DDA}$ when $V_{DDA} < 2.4$



Limitations depending on the operating power supply range

Limitations depending on the operating power supply range					
Operating power supply range	**ADC operation**	**Maximum CPU frequency (fCPUmax)**	**I/O operation**	**FSMC controller operation**	**Possible Flash memory operations**
V_{DD} = 1.8 to 2.1 V	Conversion time up to **1 .2Msps**	– **128 MHz** with 7 Flash memory wait states – **16MHz** with no Flash memory wait state	– Degraded speed performance – No I/O compensation	up to **30 MHz**	**8-bit** erase and program operations only
V_{DD} = 2.1 V to 2.4 V	Conversion time up to **1.2 Msps**	– **138 MHz** with 7 Flash memory wait states – **18MHz** with no Flash memory wait state	– Degraded speed performance – No I/O compensation	up to **30 MHz**	**16-bit** erase and program operations
V_{DD} = 2.4 V to 2.7 V	Conversion time up to **2 .4Msps**	– **168 MHz**(*) with 6 Flash memory wait states – **24MHz** with no Flash memory wait state	– Degraded speed performance – I/O compensation works	up to **48 MHz**	**16-bit** erase and program operations
V_{DD} = 2.7 V to 3.6 V	Conversion time up to **2.4 Msps**	– **168 MHz**(*) with 5 Flash memory wait states – **30MHz** with no Flash memory wait state	– Full-speed operation – I/O compensation works	– up to **60 MHz** when VDD = 3.0 V to 3.6 V – up to **48 MHz** when VDD =2.7 V to 3.0 V	**32-bit** erase and program operations

(*) when **VOS** bit in **PWR_CR** is equal to '**1**'

life.augmented

What is the I/O Compensation cell?

- An automatic slew rate adjustment depending on PVT for high frequency IO toggling:
  - IOs Rise and fall edge slopes are adjusted according to Power Voltage and Temperature
  - Recommended for 50MHz/100MHz drive IO settings
  - Allows a greater I/O noise immunity from VDD
- Feature is enabled by s/w (disabled by default) and switched off in low power mode
- Enabled with VDD operating range: 2.4V-3.6V
- Power consumption Increased when enabled

Voltage Regulators (1/2)

- 2 voltage regulators are embedded
  - A Main linear voltage regulator supplies all the digital circuitries (except for the Standby circuitry and Backup domain). The regulator output voltage (V_{CORE}) is 1.2 V (typical) and can supply up to 200mA.
  - Low voltage regulator exclusively for the backup RAM (in VBAT mode)
- The Main Voltage regulator has three different modes
  - Run and Sleep modes (200mA max)
  - Low power mode for STOP mode (5mA max)
  - Regulator OFF in STANDBY/VBAT mode.
- Regulator bypass mode
  - It allows to supply externally a 1.2 V voltage source through V_{CAP_1} and V_{CAP_2} pins, in addition to a second external V_{DD} supply source.

Voltage Regulators (2/2)

- In order to achieve a tradeoff between performance and power consumption, ‘VOS’ dedicated bit in ‘PWR_CR’ register, allows to controls the main internal voltage regulator output voltage

Condition	Max AHB clock frequency
VOS bit in **PWR_CR** register equal to ‘**0**’	**144 MHz**
VOS bit in **PWR_CR** register equal to ‘**1**’	**168 MHz**

- The voltage scaling allows to optimize the power consumption when the device is clocked below the maximum system frequency.

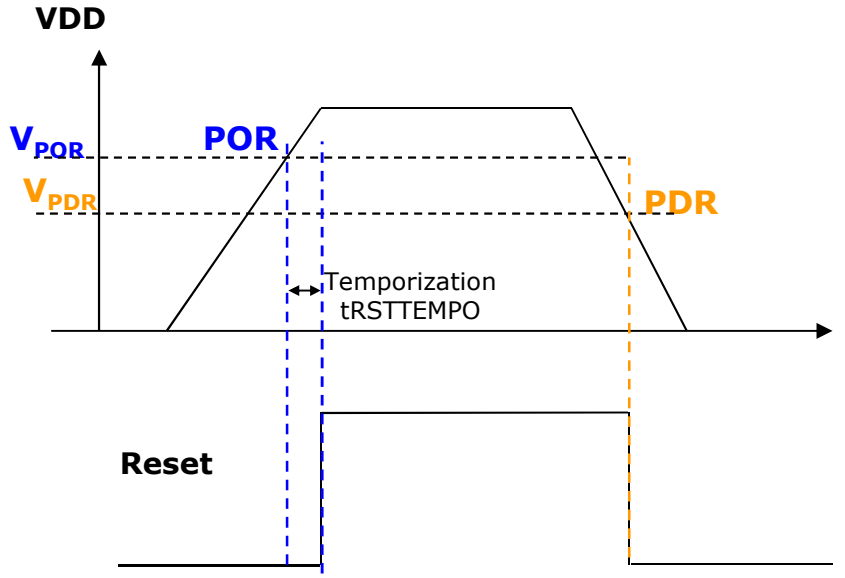
Supply monitoring and Reset circuitry

At startup:

- POR/PDR - Always ON (or CSP64 bounding option)
- Brown Out Reset (BOR) - Always ON (can be switched off after option byte loading)
- Programmable Voltage Detection (PVD) – ON/OFF.
  - PVD enable/disable bit is controlled by software via a dedicated bit (PVDE).

Power On Reset (POR)/Power Down Reset (PDR)

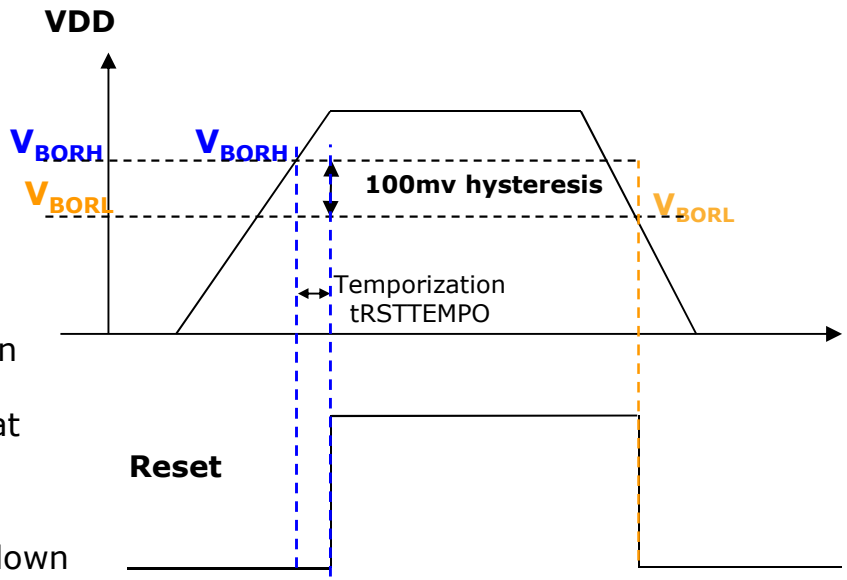
- **Integrated POR / PDR circuitry:**
  - The device has an integrated POR/PDR circuitry that allows proper operation starting from/down to 1.8 V.
- **POR and PDR have 40mV hysteresis**



$V_{PDR} = V_{POR} = 1.8V$

Brown Out Reset (BOR)

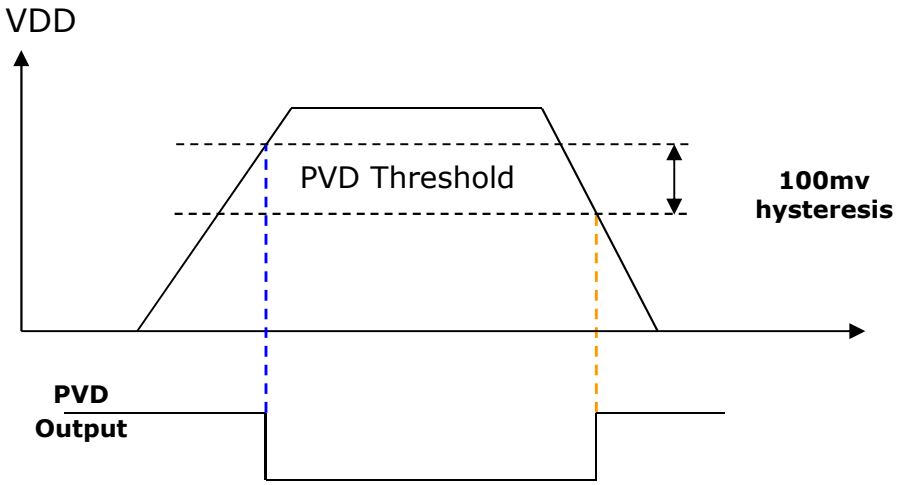
- During power on, the Brown out reset (BOR) keeps the device under reset until the supply voltage reaches the specified V_{BOR} threshold.
  - No need for external reset circuit
- BOR have a typical hysteresis of 100mV
- BOR Levels are configurable by option bytes:
  - BOR OFF: **2.1 V** at power on and **1.62 V** at power down
  - BOR LOW **(DEFAULT)** : **2.4 V** at power on and **2.1 V** at power down
  - BOR MEDIUM: **2.7 V** at power on and **2.4 V** at power down
  - BOR HIGH: **3.6 V** at power on and **2.7 V** at power down



Programmable Voltage Detector (PVD)

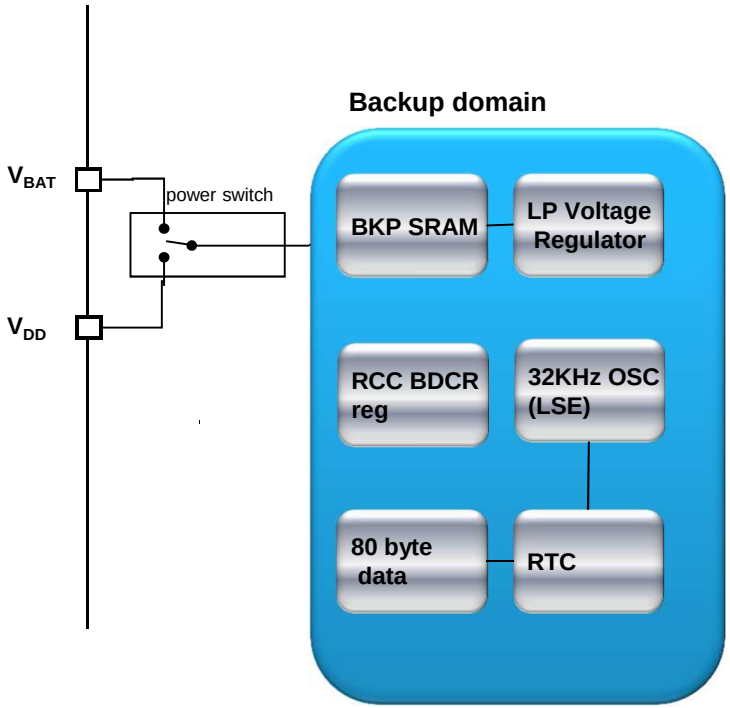
- Programmable Voltage Detector
  - Enabled by software
  - Monitor the V_{DD} power supply by comparing it to a threshold
  - Threshold configurable from 2.0V to 2.9V by step of 100mV
  - Generate interrupt through EXTI Line16 (if enabled) when VDD < Threshold and/or VDD > Threshold

    ➔ Can be used to generate a warning message and/or put the MCU into a safe state



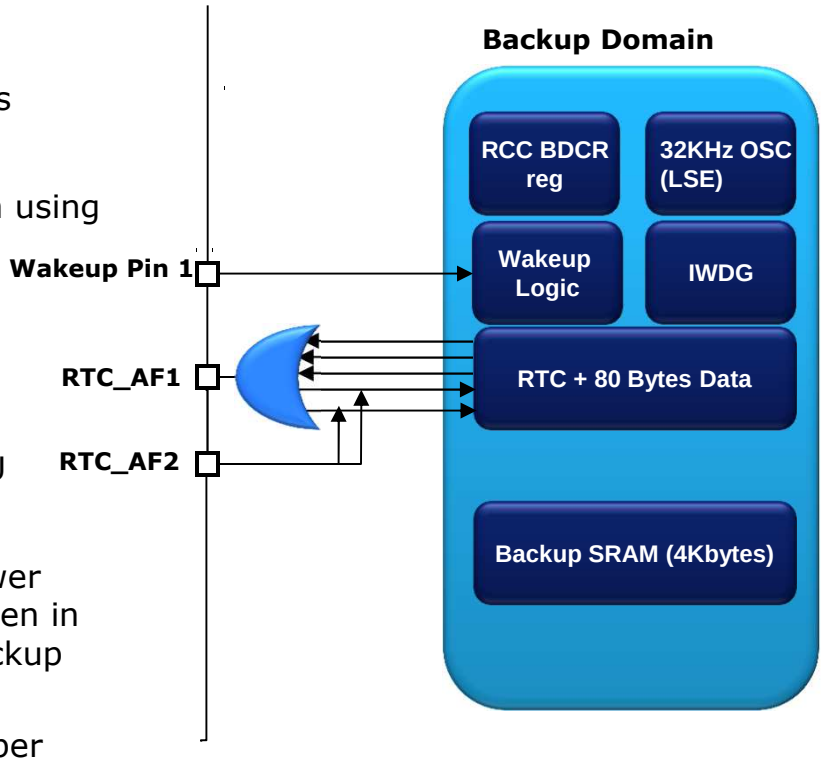
Backup Domain (1/2)

- Backup Domain
  - RTC unit and 4KB Backup RAM
  - LVR for the backup RAM
    - Switch off option
- VBAT independent voltage supply
  - Automatic switch-over to V_{BAT} when V_{DD} goes below PDR level
    - No current sunk on V_{BAT} when V_{DD} present
    - Prevent from power line down
- 1 Wakeup pin and 2 RTC Alternate functions pins (RTC_AF1 and RTC_AF2)



Backup Domain (2/2)

- RTC Alternate functions (RTC_AF1/RTC_AF2)
  - Time stamp/Tamper detection
    - resets all RTC user backup registers
    - The calendar is saved in the time-stamp registers
  - RTC Alarm Output: Alarm A, Alarm B (RTC_AF1)
  - RTC Clock calibration Output: RTCCLK/64, 512Hz when using 32.768Hz crystal. (RTC_AF1)
  - RTC Clock input reference 50/60 Hz (RTC_AF1)
- Wakeup pin
- Backup SRAM
  - 4 Kbytes of backup SRAM accessible only from the CPU
    - Can store sensitive data (crypto keys)
    - Backup SRAM is powered by a dedicated low power regulator in V_{BAT} mode. Its content is retained even in Standby and V_{BAT} mode when the low power backup regulator is enabled.
    - The backup SRAM is not mass erased by an tamper event.



STM32F4xx Low power modes features

- The STM32F4xx features 3 low power modes as in STM32F2xx family
  - SLEEP
  - STOP
  - STANDBY
- VBAT mode.
- The STM32F4xx features options to decrease the consumption during low power modes
  - Peripherals clock stopped automatically during sleep mode (S/W)
  - Flash Power Down mode
  - LVR and Backup RAM disable option
- The STM32F4xx features many sources to wakeup the system from low power modes
  - Wakeup pin (PA0) / NRST pin
  - RTC Alarm (Alarm A and Alarm B)
  - RTC Wakeup Timer interrupt
  - RTC Tamper events
  - RTC Time Stamp Event
  - IWDG Reset event

Low Power Modes (1/3)

- **SLEEP Mode**: Core stopped, peripherals kept running
  - Entered by executing special instructions
    - **WFI** (Wait For Interrupt)
      - Exit: any peripheral interrupt acknowledged by the Nested Vectored Interrupt Controller (NVIC)
    - **WFE** (Wait For Event)
      - An *event* can be an interrupt enabled in the peripheral control register but NOT in the NVIC or an EXTI line configured in event mode
      - Exit: as soon as the event occurs ➔ No time wasted in interrupt entry/exit
  - Two mechanisms to enter this mode
    - **Sleep Now**: MCU enters SLEEP mode as soon as WFI/WFE instruction are executed
    - **Sleep on Exit**: MCU enters SLEEP mode as soon as it exits the lowest priority ISR
  - To further reduce power consumption you can save power of unused peripherals by gating their clock

Low Power Modes (2/3)

- **STOP Mode**: all peripherals clocks, PLL, HSI and HSE are disabled, SRAM and register contents are preserved.
  - If the RTC and IWDG are running they are not stopped in STOP (either as their clock sources)
  - To further reduce power consumption the Voltage Regulator can be put in Low Power mode
  - V_{REFINT} and BOR can be OFF
  - Wake-up sources:
    - WFI was used for entry: any EXTI Line configured in Interrupt mode (the corresponding EXTI Interrupt vector must be enabled in the NVIC)
    - WFE was used for entry: any EXTI Line configured in event mode
    - EXTI line source can be: one of the 16 external lines, PVD, RTC alarms, RTC Tamper, RTC Time Stamp, RTC Wakeup, Ethernet, USB HS & FS wake-up

    ➔ After resuming from STOP the clock configuration returns to its reset state (MSI used as system clock).

Wake-up time from Stop mode on HSI RC at 16MHz: regulator in run mode	**13us**
Wake-up time from Stop mode on HSI RC at 16MHz: Regulator in low power mode	**17us(Typ)**
Wake-up time from Stop mode on HSI RC at 16MHz: Regulator in low power mode and Flash memory in Deep power down mode	**110us**
Supply current in Stop mode w/ main regulator in run mode	**0.45mA**
Supply current in Stop mode w/ main regulator in in low power mode	**0.31mA**

life.augmented

Low Power Modes (3/3)

- **STANDBY Mode**: Voltage Regulator off, the entire V_{CORE} domain is powered off.
  - SRAM and register contents are lost except registers in the Standby circuitry and Backup Domain.
  - RTC and IWDG are kept running in STANDBY (if enabled)
  - In STANDBY mode all IO pins are high impedance except
    - Reset pad (still available)
    - RTC_AF1 pin if configured for tamper, timestamp, Alarm A out, Alarm B out, RTC Wakeup out or calibration out
    - RTC_AF2 pin if configured for tamper or timestamp
    - WKUP1 pins if enabled
  - Wake-up sources:
    - WKUP1 pins rising edge
    - RTC alarm A, RTC alarm B, RTC Wakeup, Tamper event, TimeStamp
    - External reset in NRST pin
    - IWDG reset

    ➔ After wake-up from STANDBY mode, program execution will restart in the same way as after a RESET.

Supply current in Standby mode w/ Backup SRAM OFF, RTC OFF.(VDD=3.3V)	**2.2uA**

life.augmented

Wakeup time from Low Power

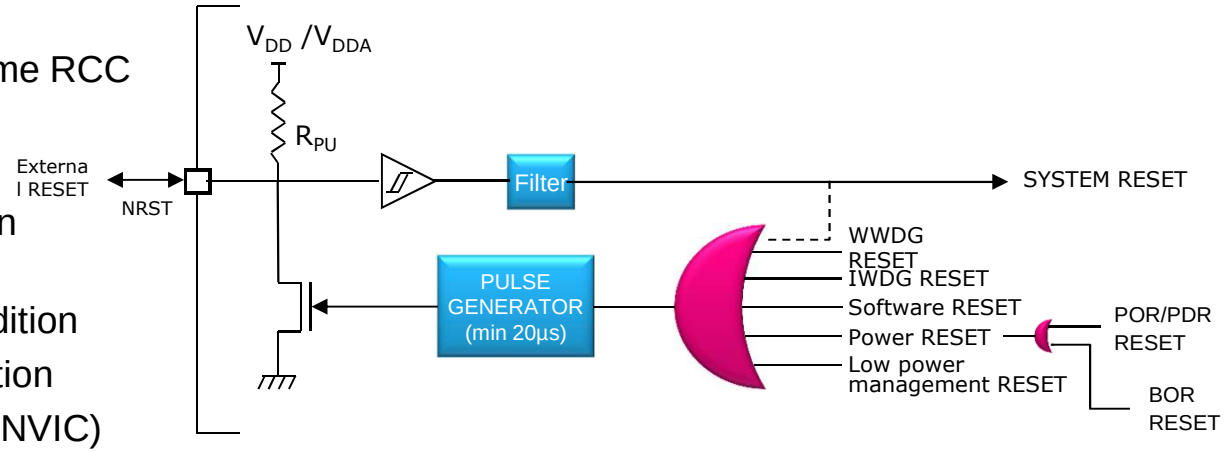
Low power mode	Conditions	Wakeup time in µs
Sleep mode		1 Typ
Stop mode	regulator in Run mode	13 Typ
Stop mode	regulator in low power mode	17 Typ
Stop mode	regulator in low power mode and Flash in Deep power down mode	110 Typ
Standby mode		375 Typ

System Peripherals

RESET AND CLOCK CONTROL (RCC)

RESET Sources

- System RESET
  - Resets all registers except some RCC registers and Backup domain
  - Sources
    - Low level on the NRST pin (External Reset)
    - WWDG end of count condition
    - IWDG end of count condition
    - A software reset (through NVIC)
    - Low power management Reset



- Power RESET
  - Resets all registers except the Backup domain
  - Sources
    - Power On/Power down Reset (POR/PDR)
    - BOR
    - Exit from STANDBY

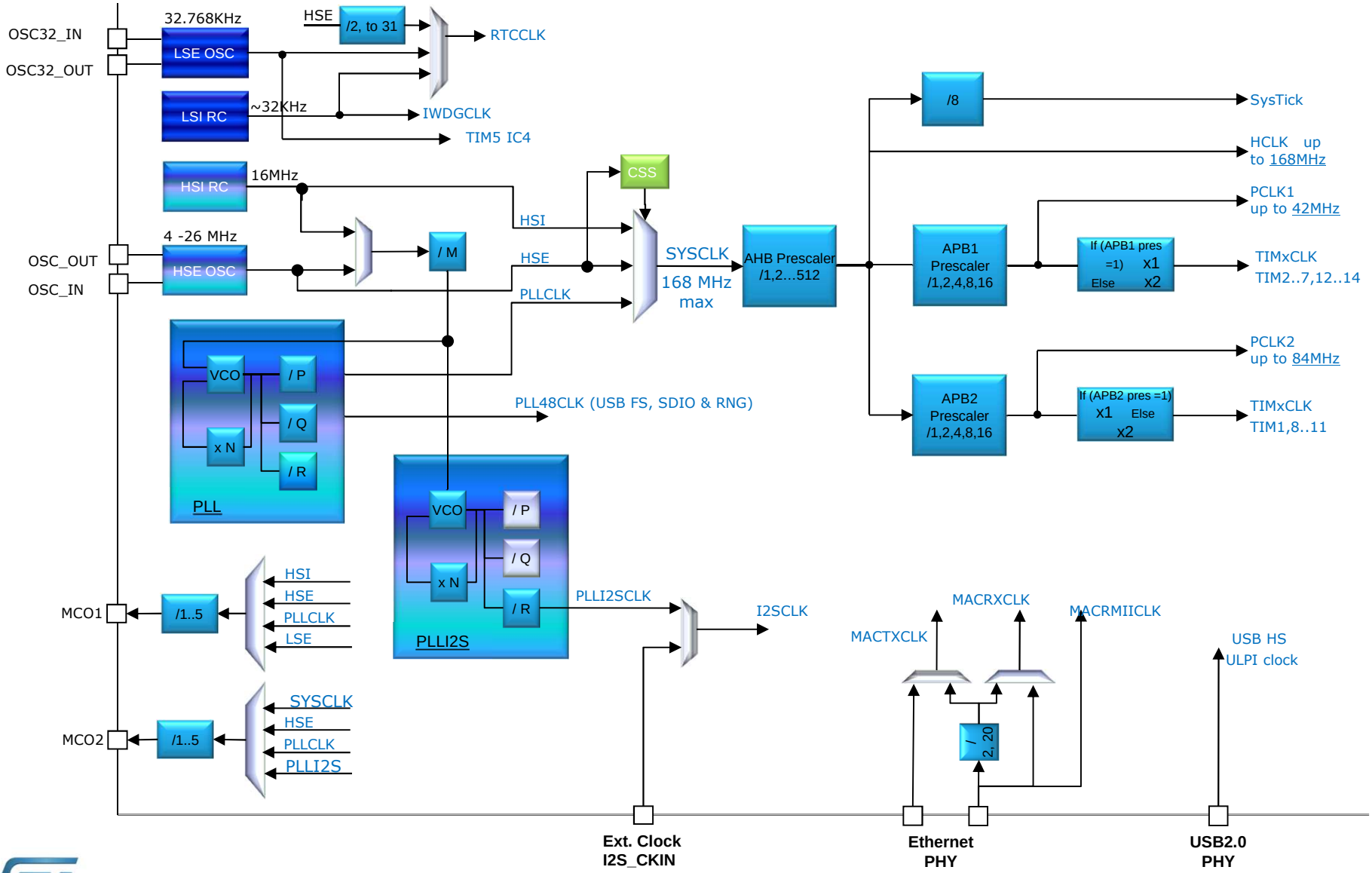
- Backup domain RESET
  - Resets in the Backup domain: RTC registers + Backup Registers + RCC BDCR register
  - Sources
    - BDRST bit in RCC BDCR register
    - POWER Reset

life.augmented

Clock Features

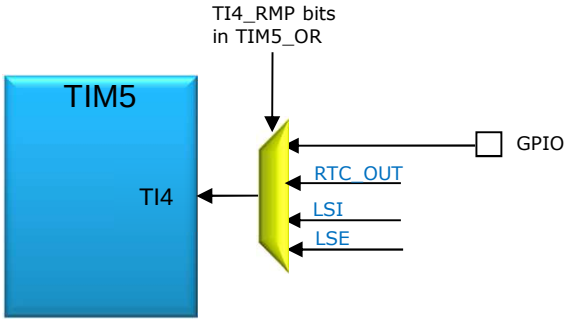
- **HSE**:(High Speed External Osc) 4MHz to 26MHz
  - Can be bypassed by and external Oscillator (ie:50MHz)
- **HSI** (High Speed Internal RC): factory trimmed internal RC oscillator 16MHz +/- 1
- **LSI** (Low Speed Internal RC): 32KHz internal RC
  - IWDG and optionally for the RTC
  - used for Auto Wake-Up (AWU) from STOP/STANDBY mode
- **LSE** (Low Speed External oscillator): 32.768kHz osc
  - precise time base with very low power consumption (max 1µA).
  - Optionally drives the RTC for Auto Wake-Up (AWU) from STOP/STANDBY mode.
  - Can be Bypassed by an external Osc
- Two PLLs
  - Main PLL (**PLL**) clocked by HSI or HSE used to generate the System clock (up to 168MHz), and 48 MHz clock for USB OTG FS, SDIO and RNG. PLL input clock in the range 1-2 MHz, 2MHz is the preferred PLL input frequency for Jitter purpose.
  - A dedicated PLL (**PLLI2S**) used to generate an accurate clock to achieve high-quality audio performance on the I2S interface.
- System Clock (SYSCLK) sources: HSI, HSE and PLL
- RTC Clock sources: LSE, LSI and HSE clock divided by 2 to 31
- Clock-out capability on the MCO1 (PA8) and MCO2 pins (PC9)
- Clock Security System (CSS) to backup clock in case of HSE clock failure (HSI feeds the system clock)
  - Enabled by SW w/ interrupt capability linked to Cortex NMI

Clock Scheme

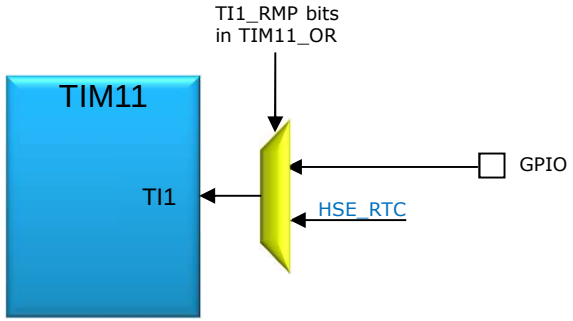


Internal/External clock measurement using TIM5/TIM11

- TIM5 channel4 input capture can be triggered by an I/O or by LSE, LSI and RTC_OUT signal ➔ Purpose:
  - Use the precise LSE clock to measure HSI frequency; HSI used as system clock, knowing the LSE frequency we can determine the HSI frequency (w/ the precision of the LSE)
  - Measure the LSI frequency (w/ the precision of the HSE) to fine tune IWDG and/or RTC timing



- TIM11 channel1 input capture can be triggered by an I/O or by HSE_RTC clock (HSE divided by a programmable prescaler)➔ Purpose:
  - Have rough indication of the external crystal frequency (require to have HSI used as system clock)



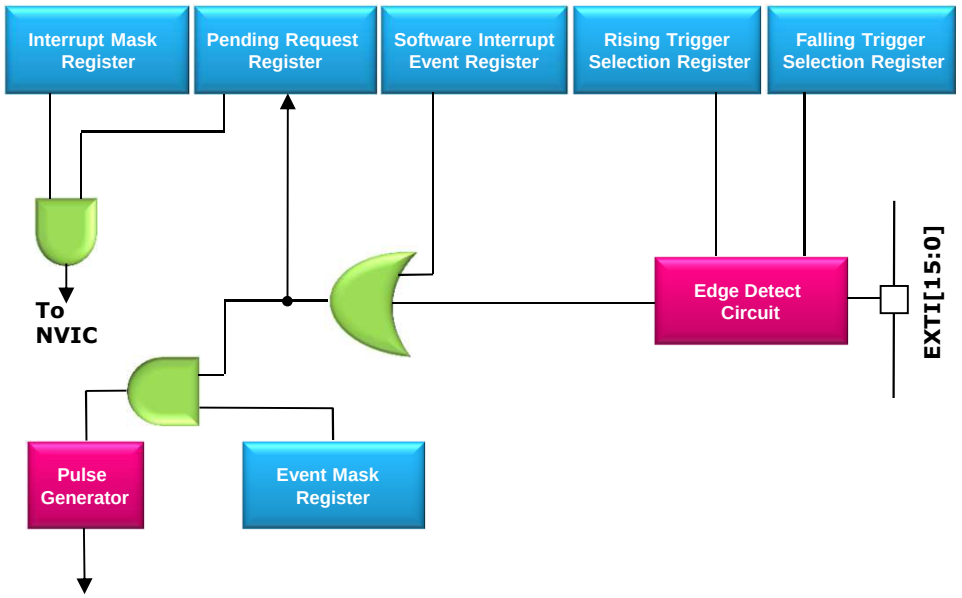
System Peripherals

EXTERNAL INTERRUPT/EVENT CONTROLLER (EXTI)

life.augmented

EXTI Features

- Up to 23 Interrupt/Events requests
  - Up to 140 GPIOs can be used as EXTI line(0..15)
  - EXTI line 16 connected to PVD output
  - EXTI line 17 connected to RTC Alarm event
  - EXTI line 18 connected to USB OTG FS Wakeup event
  - EXTI line 19 connected to Ethernet Wakeup event
  - EXTI line 20 connected to USB OTG HS (configured in FS) Wakeup event
  - EXTI line 21 connected to RTC Tamper and TimeStamp events
  - EXTI line 22 connected to RTC Wakeup event



- Two configuration modes
  - Interrupt mode: generate interrupts with external lines edges
  - Event mode: generate pulse to wake-up system from SLEEP and STOP modes
- Independent trigger (rising, falling, rising & falling) and mask on each interrupt/event line
- Dedicated status bit for each interrupt line
- Generation of up to 23 software interrupt/event requests

- Minimum Pulse Width: < 1 * T_{PCLK2} (Fast APB)
- EXTI mapped on high speed APB (APB2) to save time entering in the External Interrupt routine

System Peripherals

GENERAL-PURPOSE I/OS (GPIO)

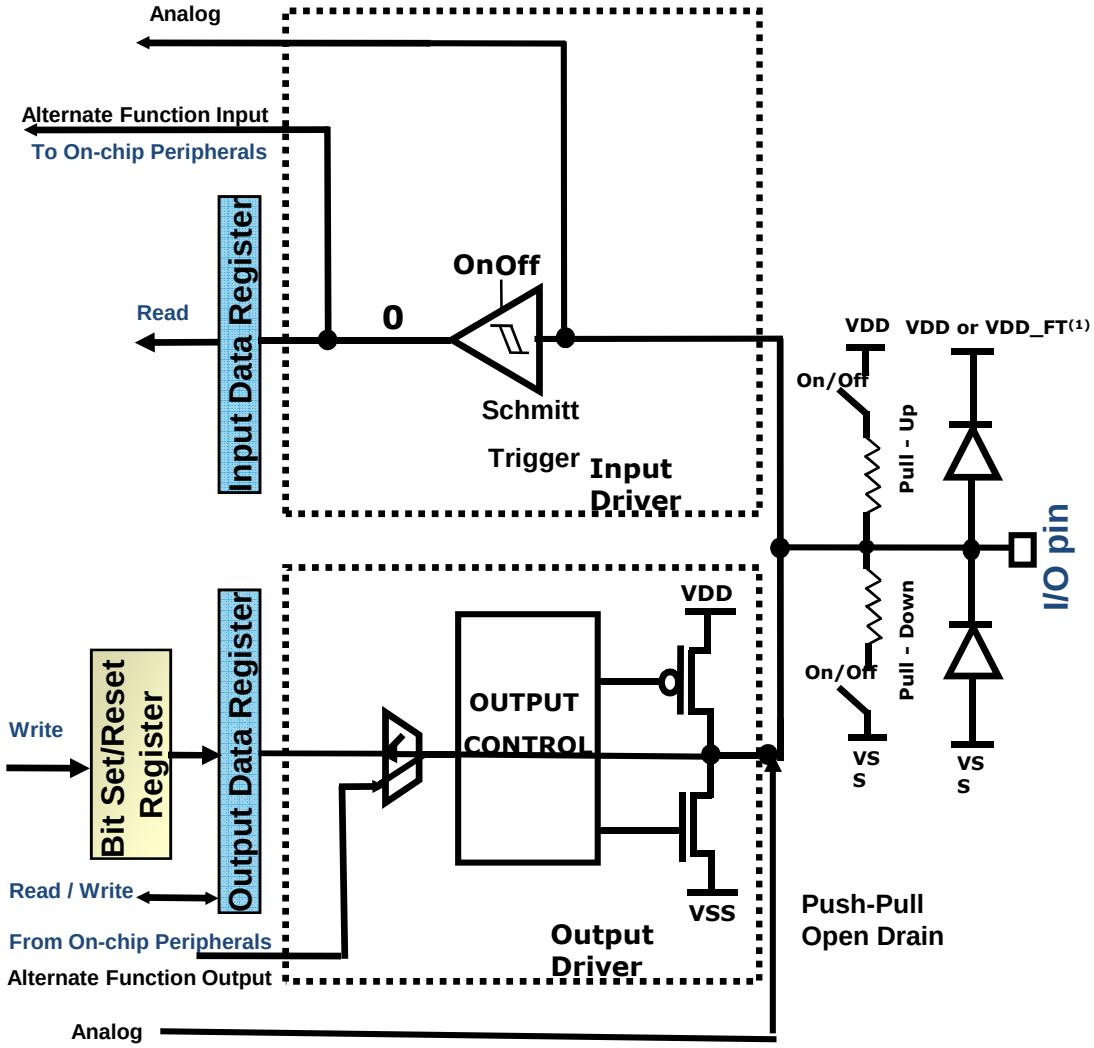
life.augmented

GPIO features

- Up to 140 multifunction bi-directional I/O ports available on biggest package 176 pin.
- Almost standard I/Os are 5V tolerant*
- All Standard I/Os are shared in 9 ports (GPIOA..GPIOI)
- Atomic Bit Set and Bit Reset using BSRR register
- GPIO connected to AHB bus: max toggling frequency 84 MHz
- Configurable Output Speed up to 100 MHz
- Locking mechanism (GPIOx_LCKR) provided to freeze the I/O configuration
- Up to 140 GPIOs can be set-up as external interrupt (up to 16 lines at time) able to wake-up the MCU from low power modes

GPIO Configuration Modes

MODER(i) [1:0]	OTYPER(i) [1:0]	PUPDR(i) [1:0]	I/O configuration
01	0	0 0 0 1 1 0	Output Push Pull with Pull-up with Pull-down
	1	0 0 0 1 1 0	Output Open Drain with Pull-up with Pull-down
10	0	0 0 0 1 1 0	Alternate Function Push Pull with Pull-up with Pull-down
	1	0 0 0 1 1 0	Alternate Function Open Drain with Pull-up with Pull-down
00	x	0 0 0 1 1 0	Input floating Pull-up Pull-down
11	x	x	Analog mode

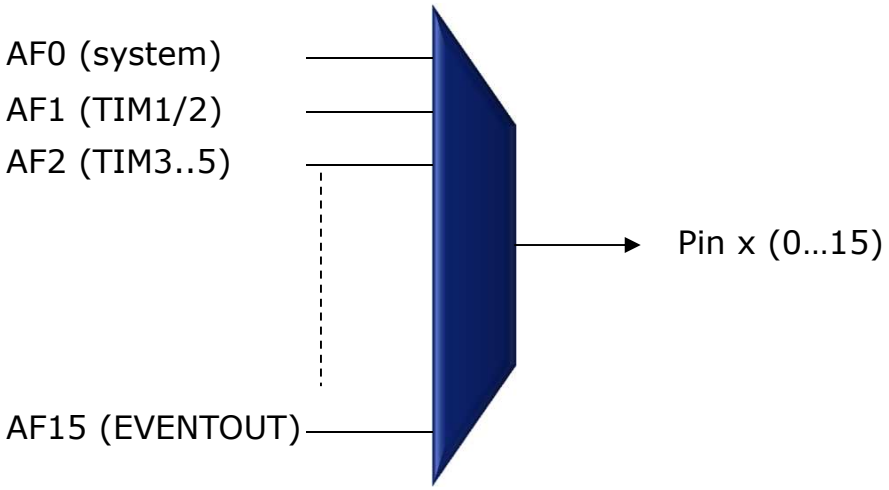


* In output mode, the I/O speed is configurable through OSPEEDR register: 2MHz, 25MHz, 50MHz or 100 MHz

(1) VDD_FT is a potential specific to five-volt tolerant I/Os and different from VDD.

Alternate Functions features

- Most of the peripherals shares the same pin (like USARTx_Tx, TIMx_CH2, I2Cx_SCL, SPIx_MISO, EVENTOUT…)
- Alternate functions multiplexers prevent to have several peripheral's function pin to be connected to a specific I/O at a time.
- Some Alternate function pins are remapped to give the possibility to optimize the number of peripherals used in parallel.
- For detail of alternate function map to each I/O, please refer the table "Alternate function mapping" in the datasheet.



DIRECT MEMORY ACCESS (DMA)

life.augmented

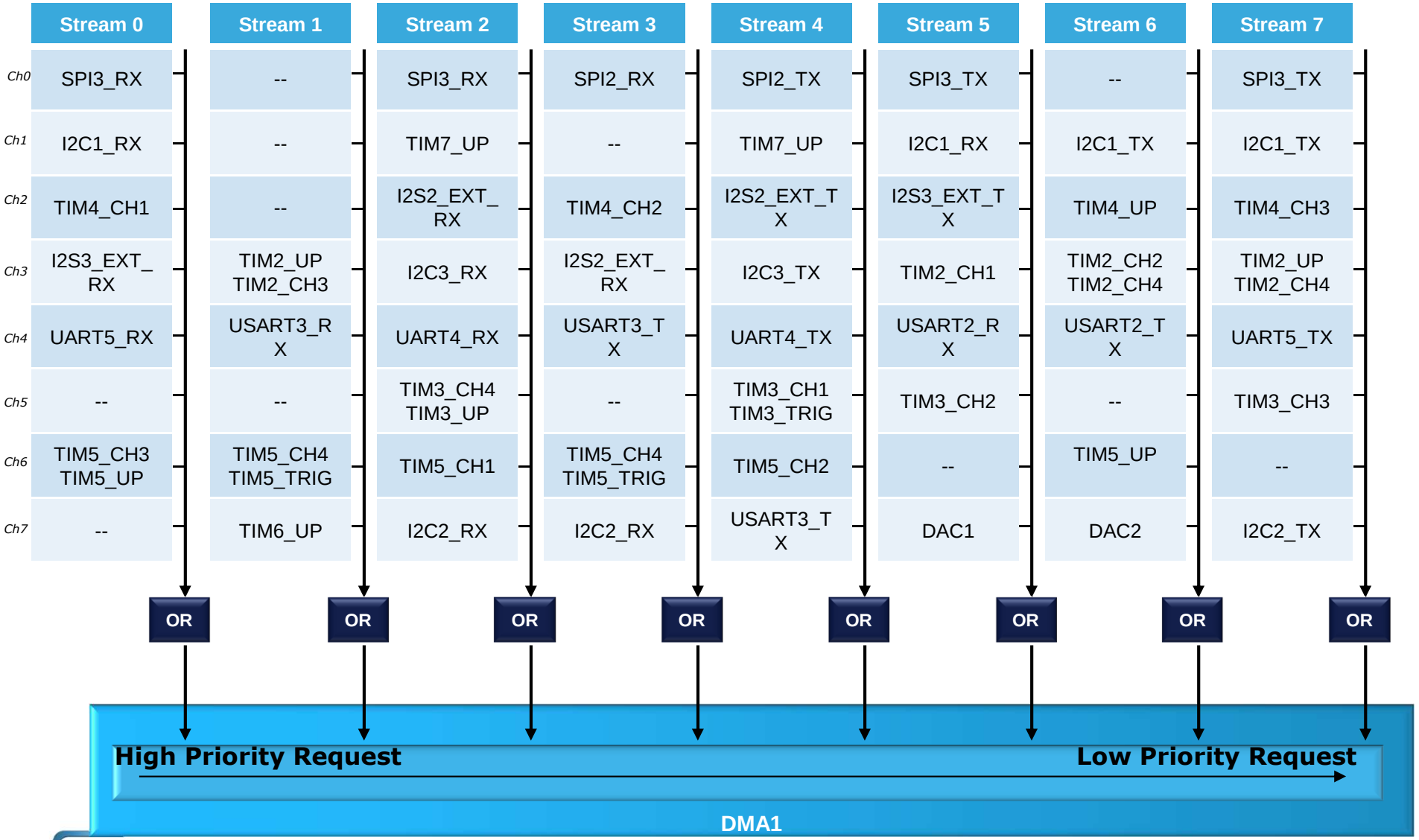
DMA Features

- Dual AHB master bus architecture, one dedicated to memory accesses and one dedicated to peripheral accesses.
- 8 streams for each DMA controller, up to 8 channels (requests) per stream (2 DMA controllers in STM32F4xx family). Channel selection for each stream is software-configurable.
- 4x32-Bits FIFO memory for each Stream (FIFO mode can be enabled or disabled).
- Independent source and destination transfer width (byte, half-word, word): when the source and destination data widths are different, the DMA automatically packs/unpacks data to optimize the bandwidth. (this feature is available only when FIFO mode is enabled)
- Double buffer mode (double buffer mode can enabled or disabled).
- Support software trigger for memory-to-memory transfers (available for the DMA2 controller streams only)

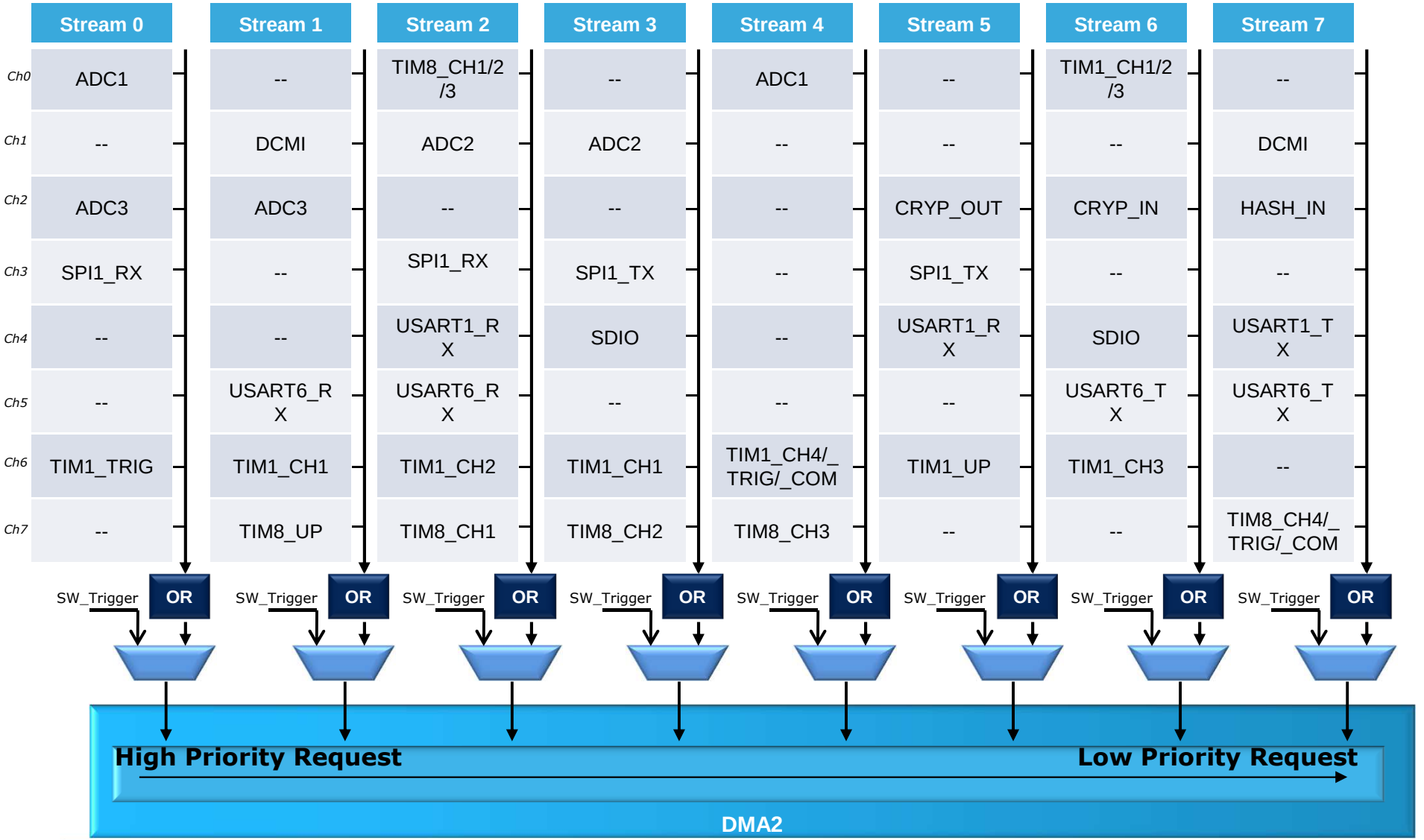
DMA Features

- The number of data to be transferred can be managed either by the DMA controller or by the peripheral:
  - DMA flow controller: the number of data to be transferred is software programmable from 1 to 65535
  - Peripheral flow controller: the number of data items to be transferred is unknown and controlled by the source or the destination peripheral that signals the end of the transfer by hardware.
- Independent Incrementing or Non-Incrementing addressing for source and destination. Possibility to set increment offset for peripheral address.
- Supports incremental burst transfers of 4, 8 or 16 beats. The size of the burst is software-configurable, usually equal to half the FIFO size of the peripheral
- Each stream supports circular buffer management.
- 5 event flags (DMA Half Transfer, DMA Transfer complete, DMA Transfer Error, DMA FIFO Error, Direct Mode Error) logically ORed together in a single interrupt request for each stream
- Priorities between DMA stream requests are software-programmable (4 levels consisting of very high, high, medium, low) or hardware in case of equality (request 0 has priority over request 1, etc.)

DMA1 Controller



DMA2 Controller



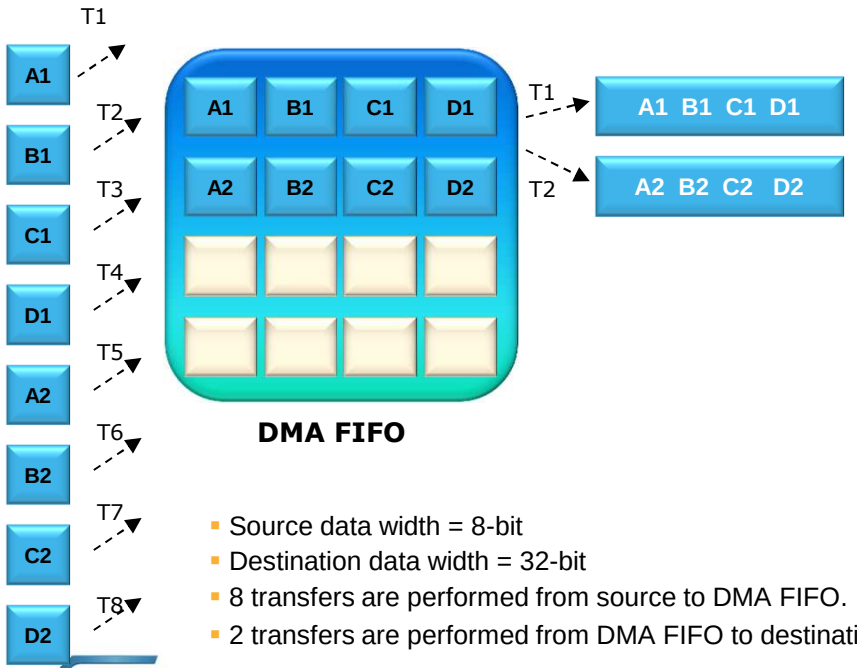
Transfer size and Flow controller

- Either the DMA or the Peripheral determine the amount of data to transfer
  - **DMA is the flow controller**: (to most applied)
    - Number of data items to be transferred is determined by the DMA through the value in register DMA_SxNDTR.
    - DMA_SxNDTR register: from 1 to 65535 bytes/half-words/words and decrements
    - Number of data items is relative only to Peripheral side
      - in Memory-to-Memory mode, the source memory is considered as peripheral
  - **Peripheral is the flow controller: SDIO only**
    - The number of transfers is determined only by the peripheral.
    - Used when the transfer size is unknown to the DMA
    - When transfer is complete, the peripheral sends End of Transfer Signal to DMA when number of transfers is reached.
- DMA_SxNDTR register can be read when transfer is ongoing to know the remaining number of transfers.

FIFO: Data Packing/Unpacking

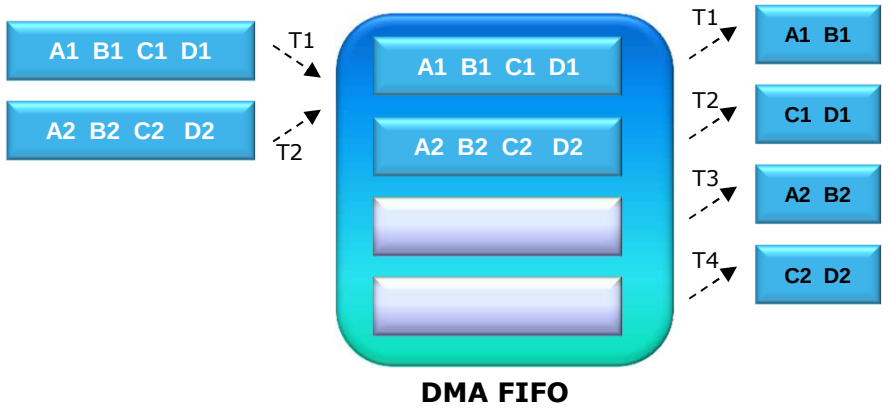
- When FIFO mode is enabled (direct mode disabled) the DMA manage the data format difference between source and destination (data Packing and Unpacking).
- Supported operations:
  - 8-bit / 16-bit → 32-bit / 16-bit (Packing)
  - 32-bit / 16-bit → 8-bit / 16-bit (Unpacking)
- This feature allows to reduce software overhead and CPU load.

Data Packing Example (8-bit → 32-bit)



- Source data width = 8-bit
- Destination data width = 32-bit
- 8 transfers are performed from source to DMA FIFO.
- 2 transfers are performed from DMA FIFO to destination.

Data Unpacking Example (32-bit → 16-bit)



- Source data width = 32-bit
- Destination data width = 16-bit
- 2 transfers are performed from source to DMA FIFO.
- 4 transfers are performed from DMA FIFO to destination.

life.augmented

FIFO: Threshold & Burst mode

- **Threshold:**
  - Threshold level determines when the data in the FIFO should be transferred to/from Memory.
  - There are 4 threshold levels:
    - ¼ FIFO Full ,1/2 FIFO Full, ¾ FIFO Full, FIFO Full
  - When the FIFO threshold is reached, the FIFO is filled/flushed from/to the Memory location.

- **Burst/Single mode:**
  - Burst mode is available only when FIFO mode is enabled (direct mode disabled)
  - Burst mode allows to configure the amount of data to be transferred without CPU/DMA interruption.
  - Available Burst modes:
    - INC4: 1 burst = 4-beats (4 Words, 8 Half-Words or 16 Bytes)
    - INC8: 1 burst = 8-beats (8 Half-Words or 16 Bytes)
    - INC16: 1 burst = 16-beats (16 Bytes)
  - When setting Burst mode, the FIFO threshold should be compatible with Burst size:

Memory Data Size	Burst Size	Allowed Threshold levels
Byte	4-Beats (INC4)	¼, ½, ¾ and Full
	8-Beats (INC8)	½ & Full
	16-Beats (INC16)	Full
Half-Word	4-Beats (INC4)	½ & Full
	8-Beats (INC8)	Full
Word	4-Beats (INC4)	Full

Notes:

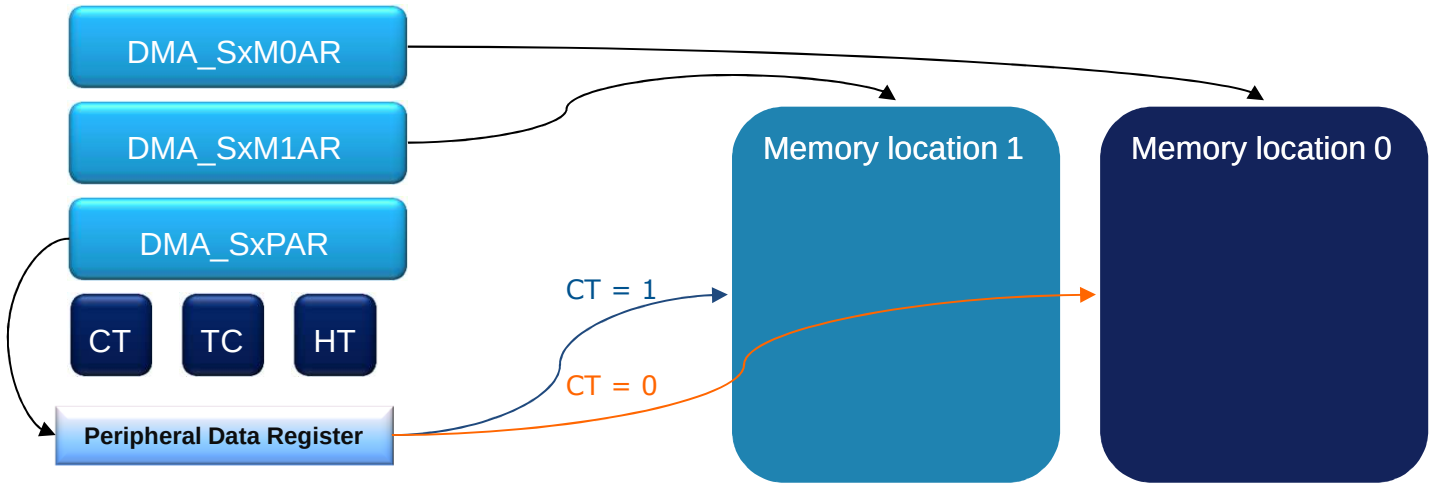
- For **Half-Word** Memory size, **INC16** is not possible.

- For **Word** Memory size, INC8 and **INC16** are not possible.

life.augmented

Circular & Double Buffer modes

- Circular mode:
  - All FIFO features and DMA events (TC, HT, TE) are available in this mode.
  - The number of data items is automatically reloaded and transfer restarted
  - This mode is NOT available for Memory-to-Memory transfers .
- Double Buffer mode: (circular mode only)
  - Two Memory address registers are available (DMA_SxM0AR & DMA_SxM1AR)
  - Allows switch between two Memory buffers to be managed by hardware.
  - Memory-to-Memory mode is not allowed
  - A flag & control bit (CT) is available to monitor which destination is being used for data transfer.
  - TC flag is set when transfer to memory location 0 or 1 is complete.



DMA Events and Interrupt Flags (1/2)

- Each DMA Stream has its own events flags and interrupts.
- **DMA Enable (EN):**
  - EN bit is a control AND status bit.
    - If the DMA is disabled while a transfer is ongoing, the current transfer is completed then the DMA Enable bit is updated (cleared).
    - If the DMA is disabled while some data are still present in FIFO, these data are flushed to memory then the DMAE flag is updated (cleared).
    - DMA allows suspending the current transfer by clearing EN bit then resume the transfer by setting EN bit again.
    - EN bit is cleared by hardware when:
      - End of transfer is reached (not applicable in circular mode).
      - A transfer error occurs on AHB buses.
      - FIFO threshold is not compatible with the burst size.
- **Transfer Completion (TC):**
  - TC flag is set when all the data configured in DMA_SxNDTR register has been actually transferred to destination.
- **Half Transfer Complete (HT):**
  - HT is set when half of data configured in DMA_SxNDTR register has been transfered.

DMA Events and Interrupt Flags (2/2)

- **Transfer Error (TE):**
  - Can indicate a bus error occurs during a DMA read or a write access.
  - Can indicate a write access is requested by software on a memory address register in Double buffer mode whereas the stream is enabled and the current target memory is the one impacted by the write into the memory address register
- **Direct mode Error (DME):**
  - Is available only in: Peripheral-to-Memory mode, in Direct mode, when Memory Incrementation is disabled.
  - Indicates that a new data is being transferred to memory location whereas the previous transfer is not complete yet.
- **FIFO Error (FE):**
  - Indicates FIFO Underrun/overrun condition or Threshold-burst size incompatibility.
- Each of TC, HT, TE, DME and FE events can independently generate an interrupt when relative interrupt enable bits is set.

Transfer modes summary

DMA transfer mode	Flow Controller	Circular mode	Transfer Type	Direct Mode	Double Buffer mode
Peripheral-to-Memory	DMA	Possible	Single	Possible	Possible
			Burst	Forbidden	
	Peripheral	Forbidden	Single	Possible	Forbidden
			Burst	Forbidden	
Memory-to-Peripheral	DMA	Possible	Single	Possible	Possible
			Burst	Forbidden	
	Peripheral	Forbidden	Single	Possible	Forbidden
			Burst	Forbidden	
Memory-to-Memory	DMA	Forbidden	Single	Forbidden	Forbidden
			Burst		

Analog Peripherals

ANALOG –TO-DIGITAL CONVERTER (ADC)

life.augmented

ADC Features (1/3)

- 3 ADCs : ADC1 (master), ADC2 and ADC3 (slaves).
- Maximum frequency of the ADC analog clock is 36MHz.
- 12-bits, 10-bits, 8-bits or 6-bits configurable resolution.
- ADC conversion rate with 12 bit resolution is up to:
  - 2.4 M.sample/s in single ADC mode,
  - 4.5 M.sapmle/s in dual interleaved ADC mode,
  - 7.2 M.sample/s in Triple interleaved ADC mode.
- Conversion range: 0 to 3.6 V.
- ADC supply requirement: VDDA = 2.4V to 3.6V at full speed and down to 1.65V at lower speed.
- Up to 24 external channels.
- 3 ADC1 internal channels connected to:
  - Temperature sensor,
  - Internal voltage reference : V_{REFINT} (1.2V typ),
  - VBAT for internal battery monitoring.

life.augmented

ADC Features (2/3)

- External trigger option for both regular and injected conversion.
- Single and continuous conversion modes.
- Scan mode for automatic conversion of channel 0 to channel ‘n’.
- Left or right data alignment with in-built data coherency.
- Channel by channel programmable sampling time.
- Discontinuous mode.
- Dual/Triple mode (with ADC1 and ADC2 or all 3 ADCs) with various possibilities :
  - Injected simultaneous mode,
  - Regular simultaneous mode,
  - Interleaved mode,
  - Alternate trigger mode,
  - Injected simultaneous mode + Regular simultaneous mode,
  - Regular simultaneous mode + Alternate trigger mode.

ADC Features (3/3)

- DMA capability
  - DMA request generation during regular channel conversion in single mode,
  - “ADC-DMA mode 1” used in regular simultaneous Dual/Triple ADC mode,
  - “ADC-DMA mode 2” used in interleaved Dual/Triple ADC mode as well as in regular simultaneous Dual ADC mode,
  - “ADC-DMA mode 3” used in interleaved Dual/Triple ADC mode with 6-bits and 8-bits resolution.
- Analog Watchdog on high and low thresholds.
- Interrupt generation on:
  - End of Conversion
  - End of Injected conversion
  - Analog watchdog
  - Overrun

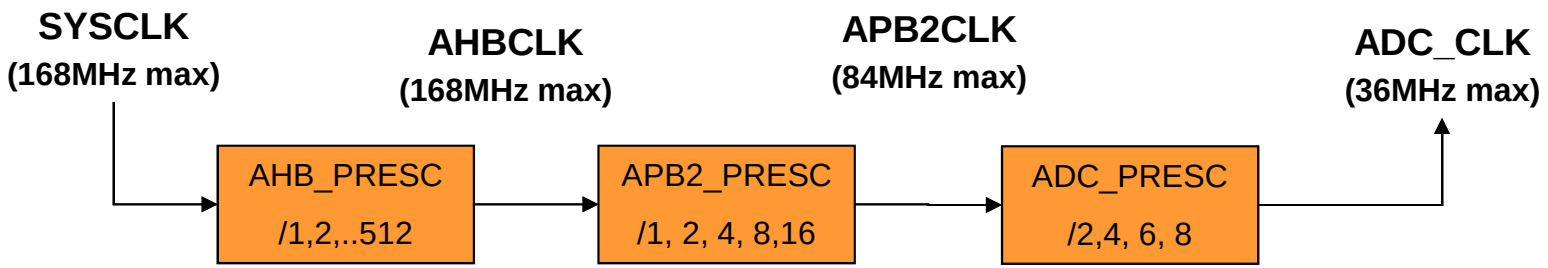
ADC speed performances

AHBCLK	APB2CLK	ADC_CLK	ADC speed (15 cycles)
168MHz	(a) 84MHz	(2) 21MHz	0.714µs **1.4 Msample/s**
144MHz	(a) 72MHz	(1) 36MHz	0.416µs **2.4 Msample/s**
120MHz	(a) 60MHz	(1) 30MHz	0.5µs **2 Msample/s**
96MHz	(a) 48MHz	(1) 24MHz	0.625µs **1.6 Msample/s**
72MHz	(b) 72MHz	(1) 36MHz	0.416µs **2.4 Msample/s**

(1). ADC_PRESC = /2
(2). ADC_PRESC = /4

(a) APB_PRESC = /2

(b) APB_PRESC = /1

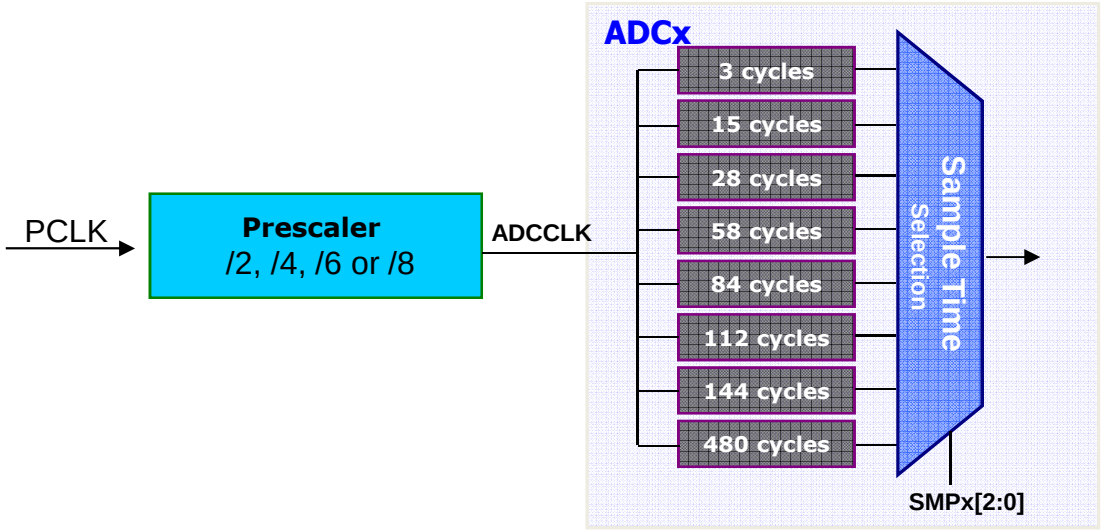


life.augmented

ADC Sampling Time ($T_{Sampling}$)

- ADCCLK, up to 36MHz, taken from PCLK through a prescaler (Div2, Div4, Div6 and Div8).

- Three bits programmable sample time for each channel:
  - 3 cycles
  - 15 cycles
  - 28 cycles
  - 58 cycles
  - 84 cycles
  - 112 cycles
  - 144 cycles
  - 480 cycles



Total Conversion Time

- Total conversion Time = $T_{Sampling}$ + $T_{Conversion}$

Resolution	$T_{Conversion}$
12 bits	12 Cycles
10 bits	10 Cycles
8 bits	8 Cycles
6 bits	6 Cycles

- With Sample time= 3 cycles @ ADC_CLK = 36MHz ➔ total conversion time is equal to :

resolution	Total conversion Time	
12 bits	**12 + 3 = 15cycles**	**0.416 us → 2.4 Msps**
10 bits	10 + 3 = 13 cycles	0.361 us → 2.71 Msps
8 bits	8 + 3 = 11 cycles	0.305 us → 3.27 Msps
6 bits	6 + 3 = 9 cycles	0.25 us → 4 Msps

life.augmented

ADC Regular / Injected channels group

- Programmable number of regular channels: Up to 16 conversions.
- Programmable sample time and conversion sequence.
- Conversion started by:
  - Software: through start bit,
  - External trigger generated by:
    - EXTI IT11,
    - 15 triggers from 6 TIMERS,
- Programmable number of injected channels: Up to 4 conversions.
- Programmable sample time and conversion sequence.
- Conversion started by:
  - JAUTO: automatic injected conversion after regular channels conversion,
  - Software: through start bit,
  - External trigger generated by:
    - EXTI IT15,
    - 15 triggers from 6 TIMERS.

life.augmented

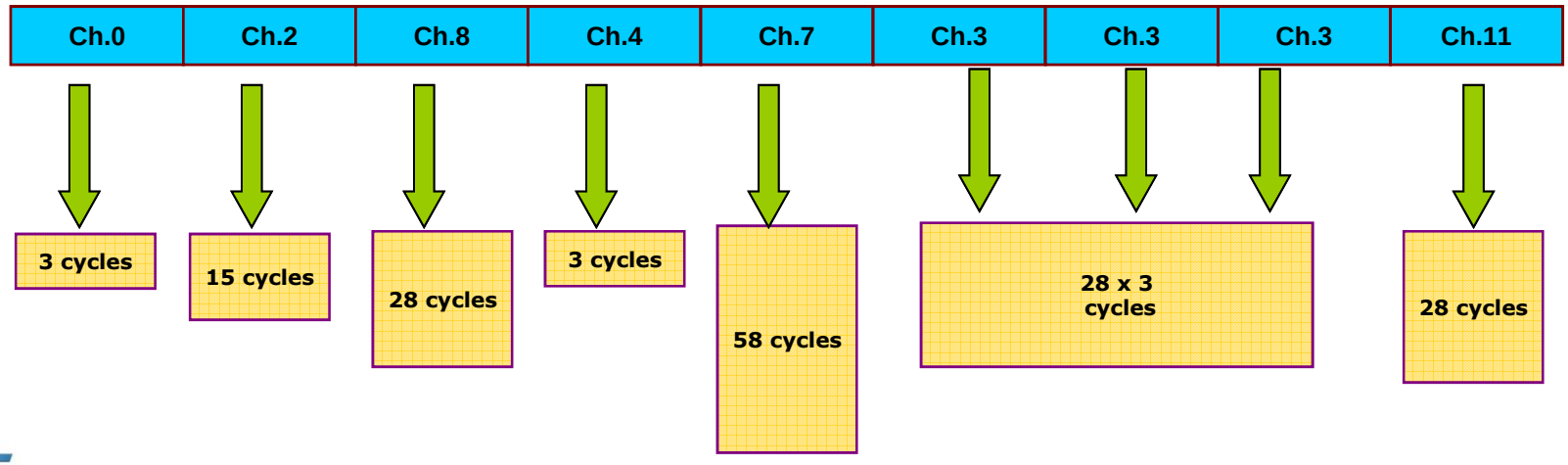
ADC Sequencer

- Up to 16 regular and 4 injected conversions with programmable order, programmable sampling time and over-sampling possibility.

Example: - Conversion of channels: 0, 2, 8, 4, 7, 3, 3 ,3 and 11

- Different sampling time.

- Over-sampling of channel 3.



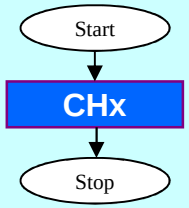
life.augmented

ADC conversion modes

- Five conversion modes are available:

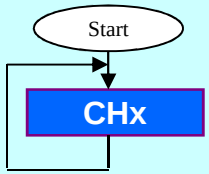
Single channel

single conversion mode



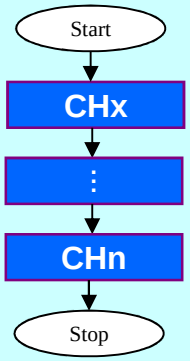
Single channel

Continuous conversion mode



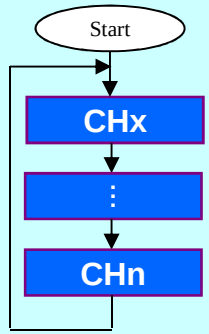
Multi-channels (Scan)

single conversion mode



Multi-channels (Scan)

continuous conversion mode



Discontinuons conversion mode

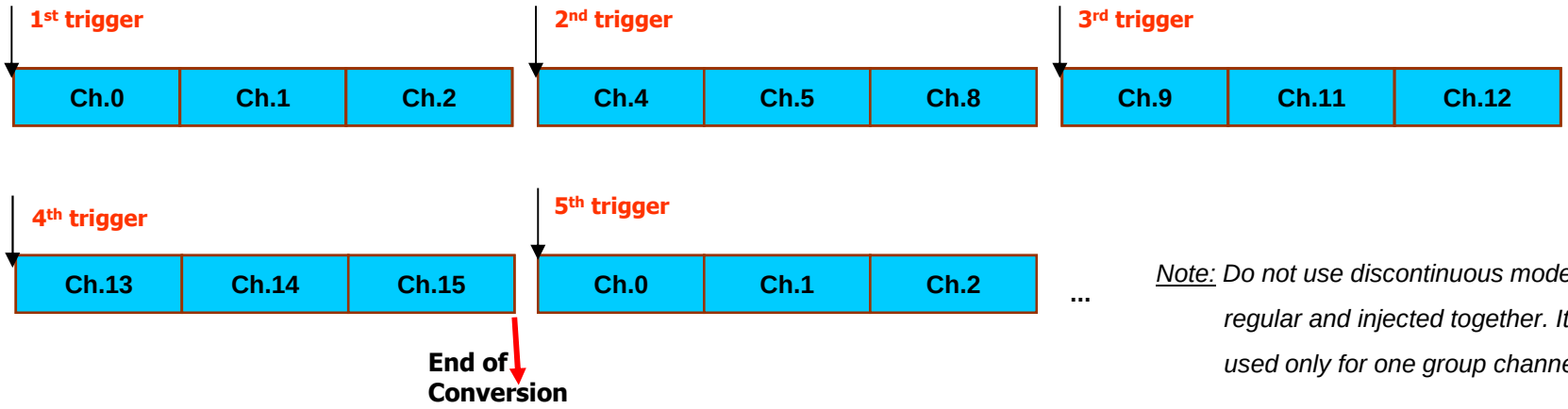
ADC discontinuous conversion mode

- Split channels conversion sequence into sub-sequences
- Available for both regular and injected groups:
  - Up to 8 conversions for regular group
  - Up to 3 conversions for injected group

Example:

- Conversion of regular channels: 0, 1, 2, 4, 5, 8, 9, 11, 12, 13, 14 and 15

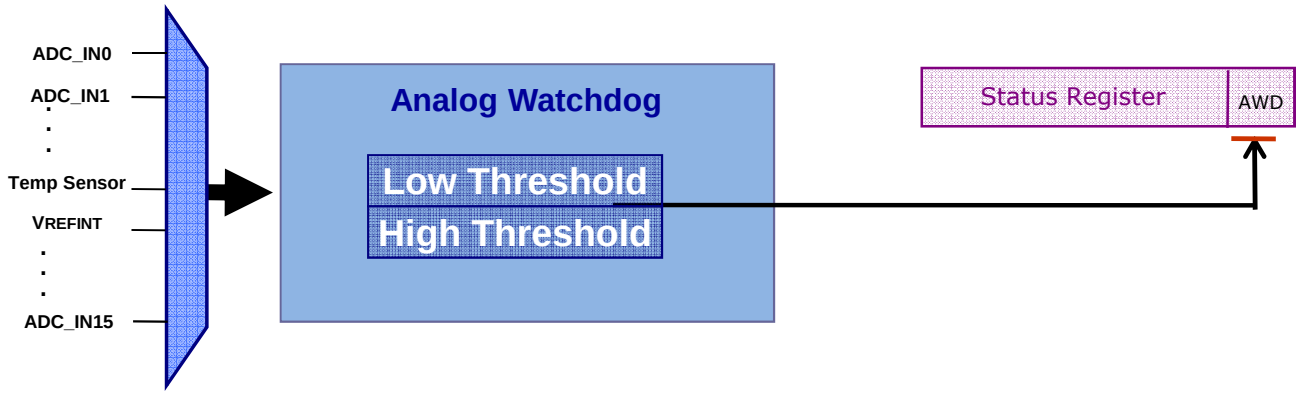
- Discontinuous mode Number of conversions: 3



Note: Do not use discontinuous mode for both regular and injected together. It can be used only for one group channel

ADC Analog Watchdog

- 12-bit programmable analog watchdog low and high thresholds
- Enabled on one or all converted channels: one regular or/and injected channel, all injected or/and regular channels.
- Interrupt generation on low or high thresholds detection

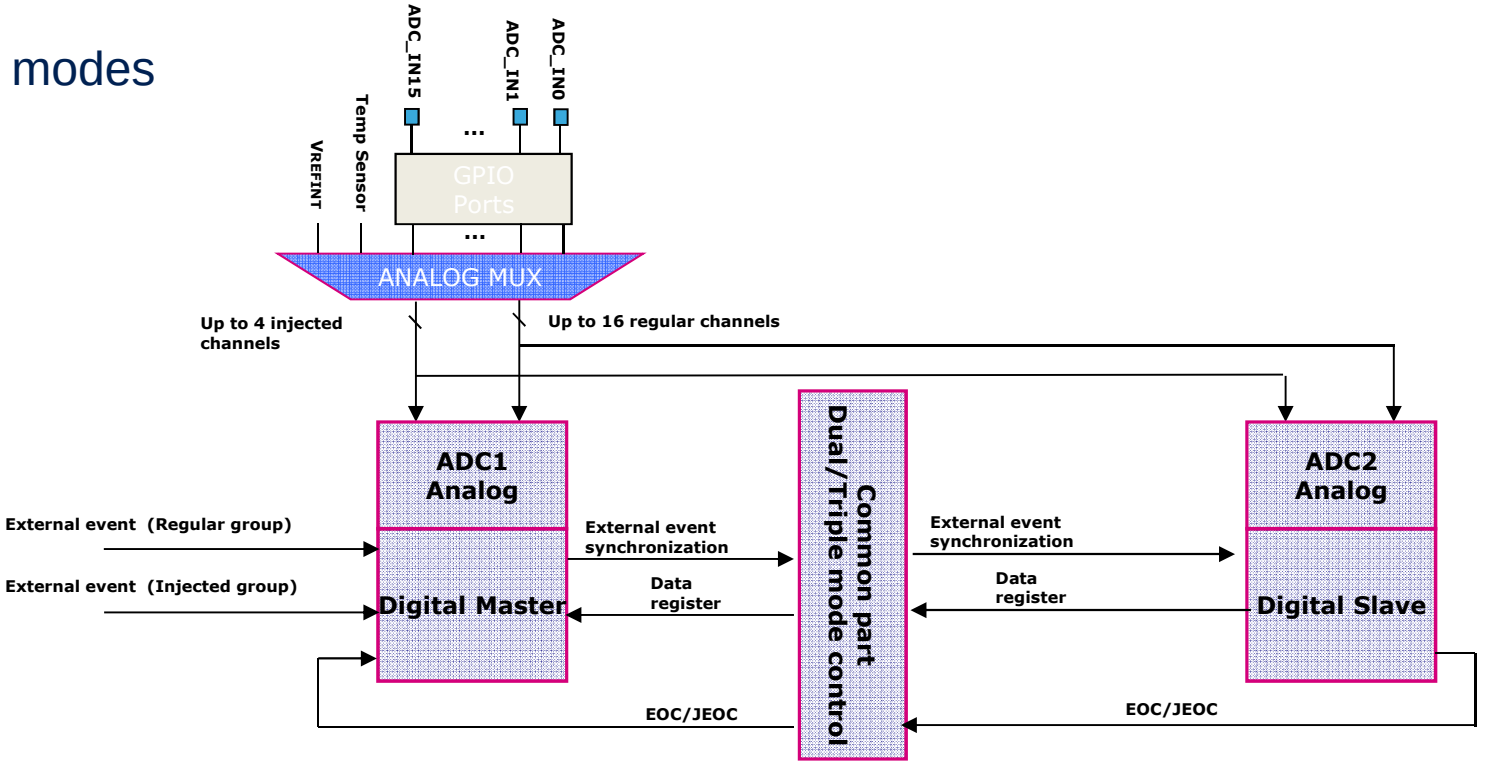


Temperature sensor and V_{BAT} monitoring

- The Temperature Sensor is internally connected to the ADC1_IN16 input channel.
- The temperature sensor can be used to measure the junction temperature (TJ) of the device.
- Accuracy: +/- 1.5°C.
- V_{BAT} input voltage can be converted on ADC1_IN18 input for Battery monitoring.
- As V_{BAT} voltage could be higher than VDDA, to assure the correct operating condition of the ADC, the Input voltage is in fact V_{BAT} followed by a bridge divider by 2.

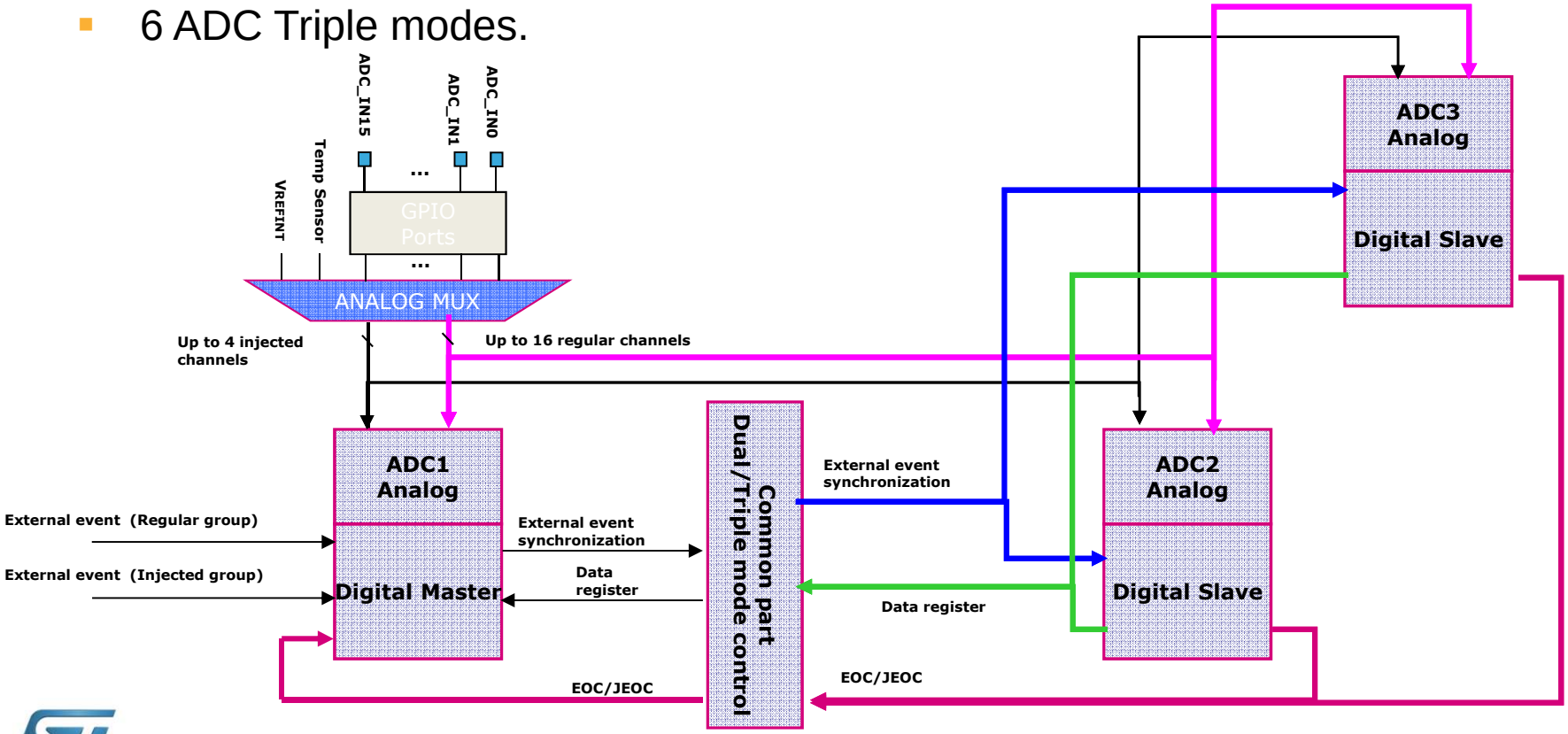
ADC dual modes

- ADCs: ADC1 master and ADC2 slave, ADC3 is independently.
- The start of conversion is triggered alternately or simultaneously by the ADC1 master to the ADC2 slave depending on the mode selected.
- 6 ADC dual modes



ADC Triple modes(1/7)

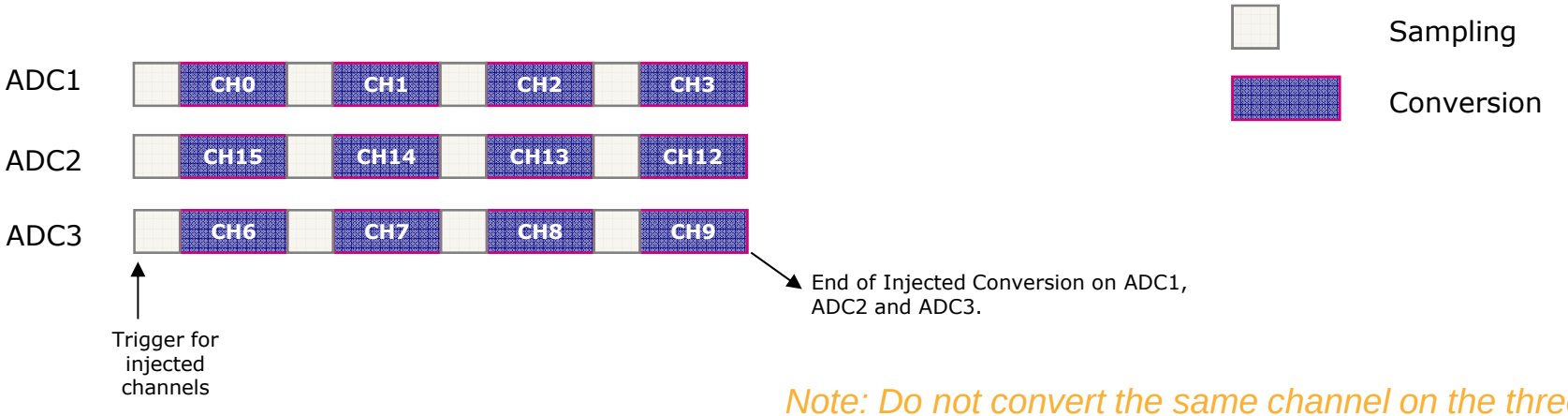
- ADCs: ADC1 master, ADC2 and ADC3 slaves.
- The start of conversion is triggered alternately or simultaneously by the ADC1 master to the ADC2 and ADC3 slaves depending on the mode selected.
- 6 ADC Triple modes.



ADC Triple modes(2/7) Injected simultaneous mode

- Converts an injected channel group.
- The external trigger source, which start the conversion, comes from ADC1 (simultaneous trigger provided to ADC2 and ADC3 slaves).
- An end of injected conversion is generated at the end of all channels conversion.
- Results stored on injected data registers of each ADC.

Injected simultaneous mode on 4 injected channels:

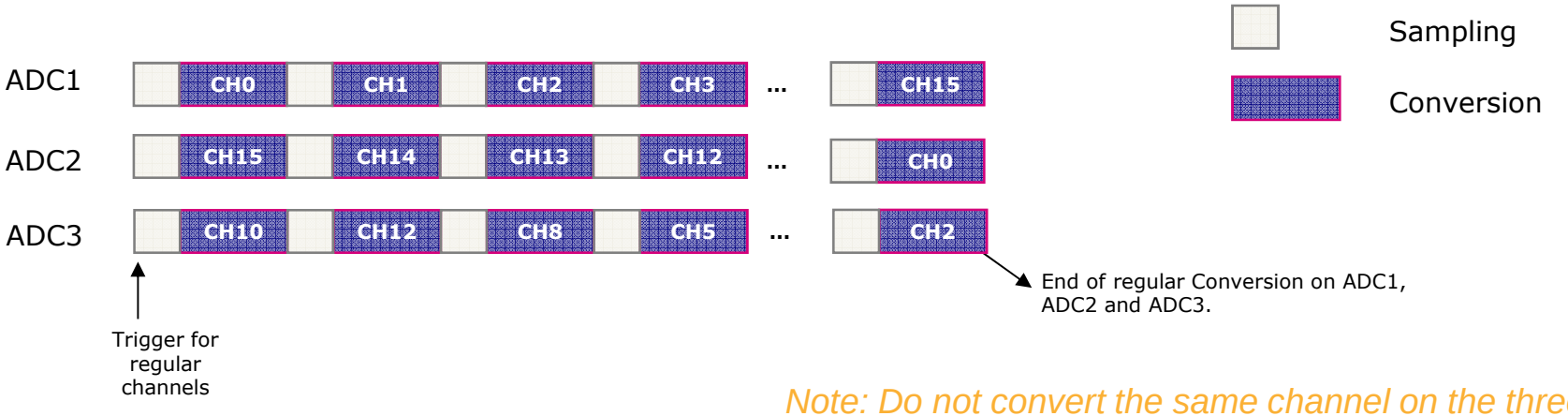


Note: Do not convert the same channel on the three ADCs.

ADC Triple modes(3/7) Regular simultaneous mode

- Converts a regular channel group.
- The external trigger source, which start the conversion, comes from ADC1 (simultaneous trigger provided to ADC2 and ADC3 slaves).
- An end of regular conversion is generated at the end of all channels conversion.
- Results stored on the common data register ADC_CDR.

Regular simultaneous mode on 16 regular channels:

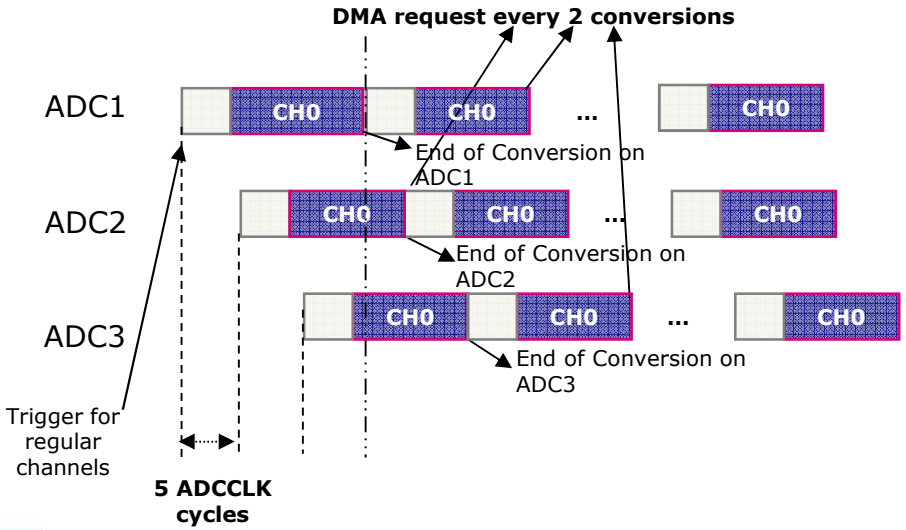


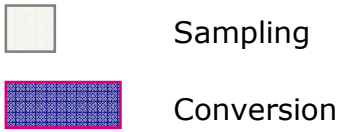
Note: Do not convert the same channel on the three ADCs.

ADC Triple modes(4/7) Interleaved mode

- Converts a regular channel group (usually one channel).
- The external trigger source, which start the conversion, comes from ADC1:
  - ADC1 starts immediately,
  - ADC2 starts after a delay of 5 ADC clock cycles,
  - ADC3 starts after a delay of 5 ADC clock cycles referred to the ADC2 conversion.
- In this mode the sampling criteria to respect is always "Tsampling + 2 ADC Cycles “ as minimum delay.
- Results stored on the common data register ADC_CDR.

Interleaved mode on 1 regular channel in continuous conversion mode:



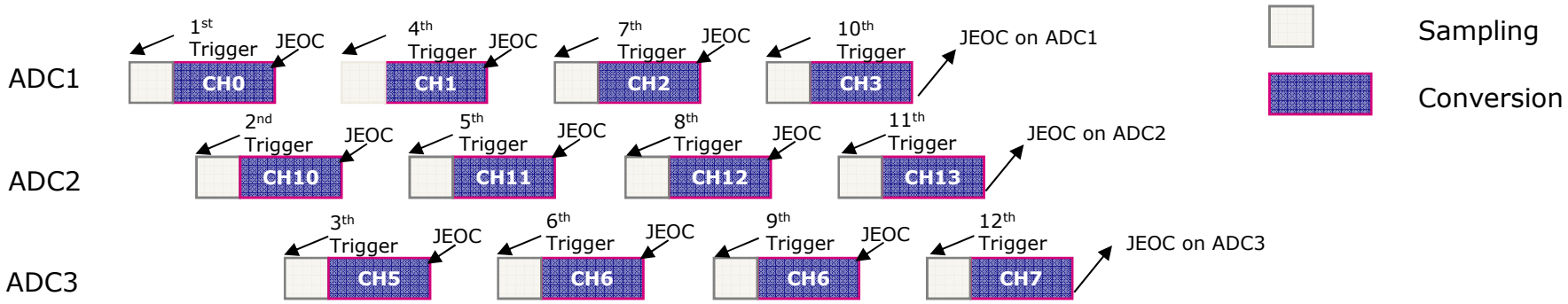


7.2M sample/s with 3 cycles of sampling time

ADC Triple modes (5/7) Alternate Trigger mode

- Converts an injected channel group.
- The external trigger source, which start the conversion, comes from ADC1:
  - On 1st trigger, the first injected group channel in ADC1 is converted
  - On 2nd trigger, the first injected group channel in ADC2 is converted
  - On 3rd trigger, the first injected group channel in ADC3 is converted
  - On 4th trigger, the second injected group channel in ADC1 is converted…
- An end of injected conversion is generated at the end of each conversion
- Results stored on injected data registers of each ADC.

Alternate Trigger mode on 4 injected channels:

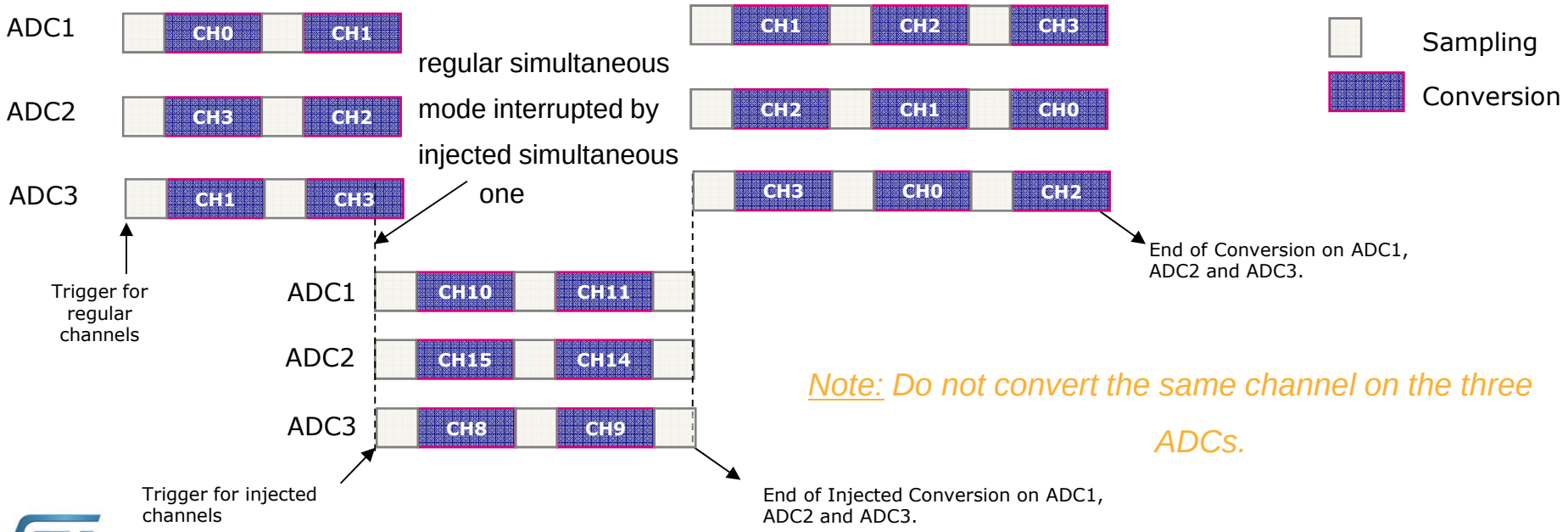


ADC Triple modes (6/7)

Combined Regular + Injected simultaneous mode

- Converts an injected and regular channel groups.
- The external triggers sources, which start the conversions, comes from ADC1 (simultaneous trigger provided to ADC2 and ADC3): injected simultaneous mode can interrupt all channels conversions.
- Results of injected channels stored on injected data registers of each ADC, and regular channels on the common data register ADC_CDR.

Combined Regular/Injected simultaneous mode on 4 regular channels and 2 injected channels:



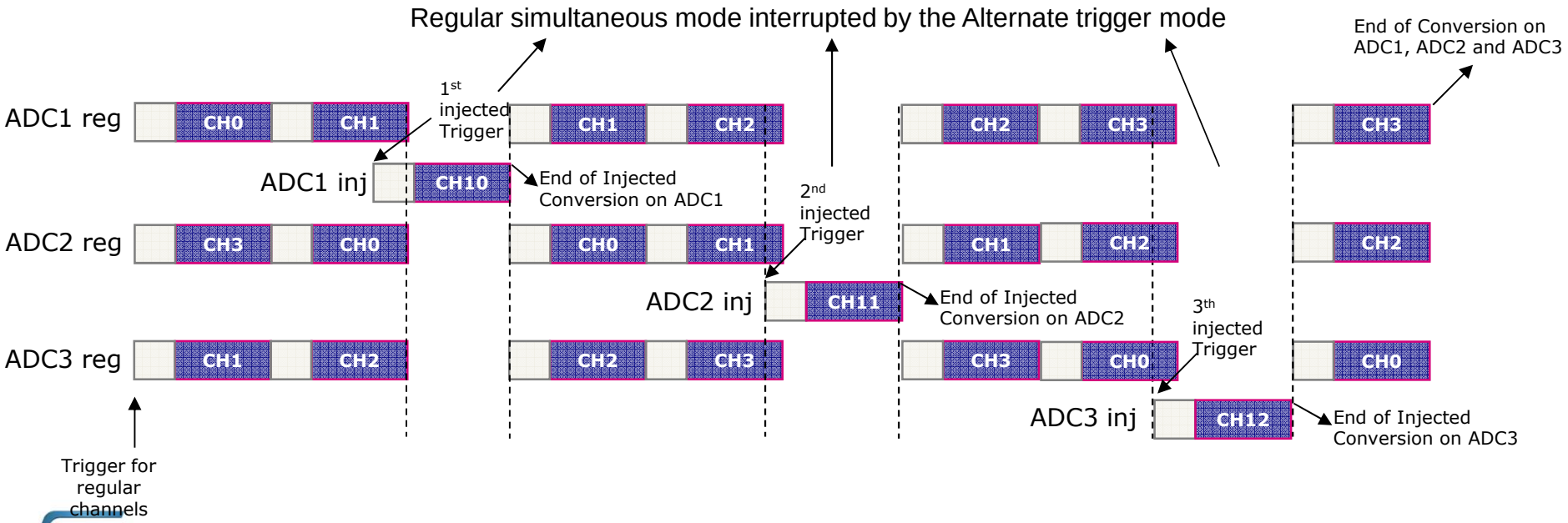
Note: Do not convert the same channel on the three ADCs.

ADC Triple modes (7/7)

Combined Regular simultaneous + Alternate Trigger mode

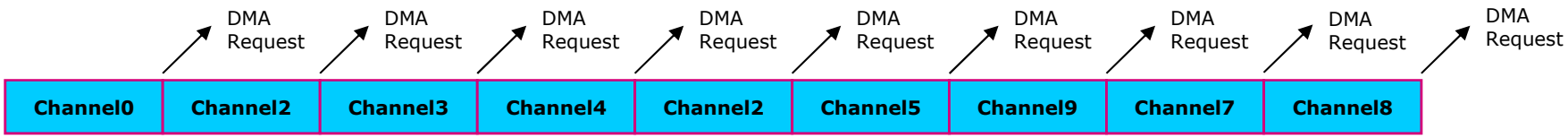
- Converts an injected and regular channel groups.
- The external triggers sources, which start the conversions, comes from ADC1 (simultaneous trigger provided to ADC2 and ADC3): alternate trigger mode can interrupt all channels conversions.
- Results of injected channels stored on injected data registers of each ADC, and regular channels on the common data register ADC_CDR.

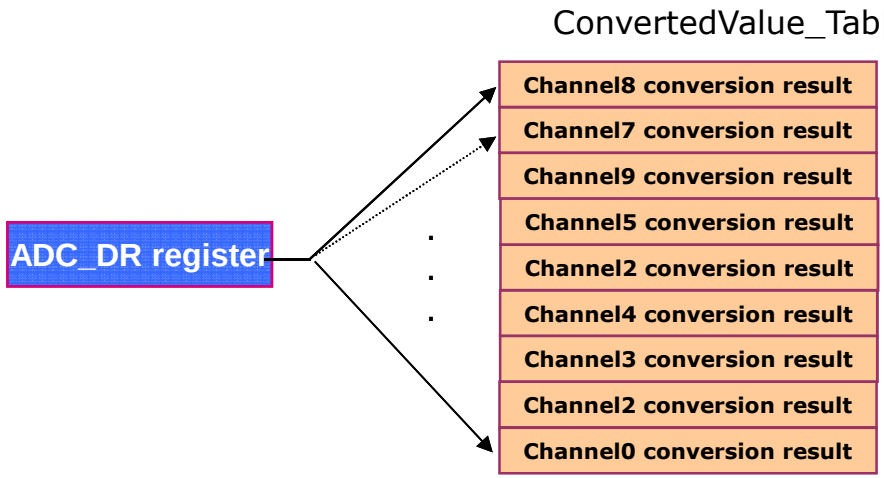
Combined Regular simultaneous + Alternate trigger mode:



ADC and DMA(1/4): Single mode

- DMA request generated on each ADC end of regular channel conversion (Not in injected channels).





Example:

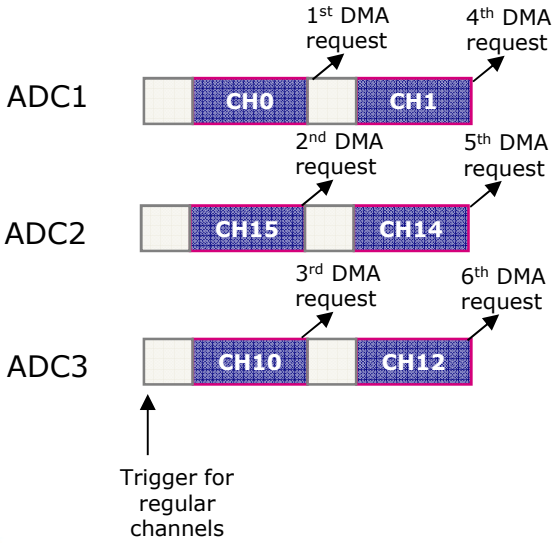
- Conversion of regular channels: 0, 2, 3, 4, 2, 5, 9, 7 and 8
- Converted data stored in ConvertedValue_Tab[9]
- DMA transfer enabled (destination address auto incremented)

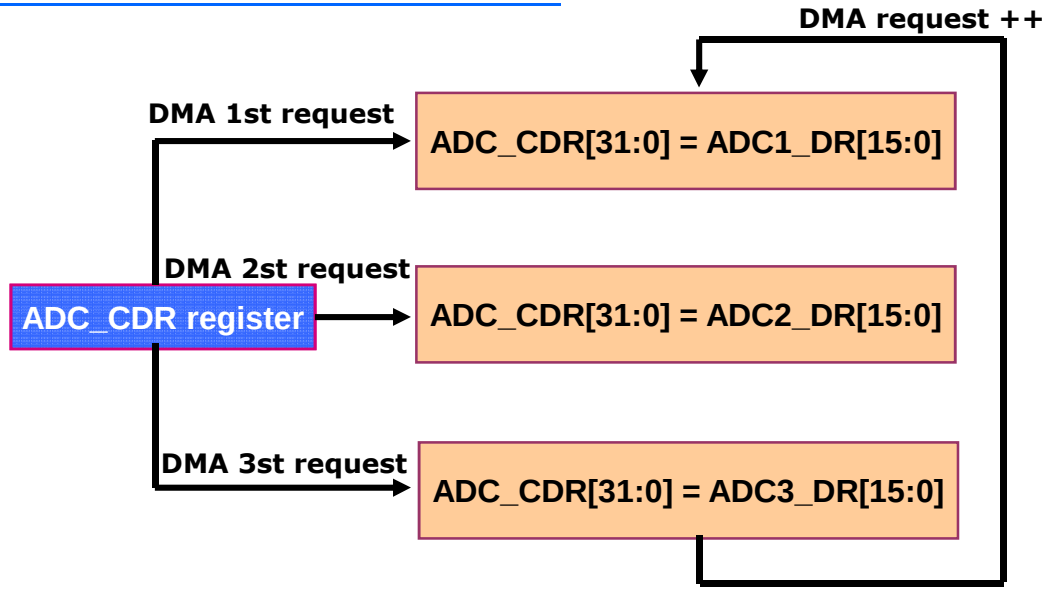
Note: Each time DMA accessed to ADC_DR register, EOC flag is automatically cleared

ADC and DMA(2/4): ADC_DMA mode “1”

- Used in regular simultaneous Dual/Triple ADC mode.
- DMA ensure the access to the converted regular channel values which are stored into the common data register ADC_CDR.
- On each DMA request, a half-word representing an ADC-converted data item is transferred.
- DMA bits in common control register ADC_CCR must be set at 0b01.

DMA transfer in Triple ADC Regular simultaneous mode:

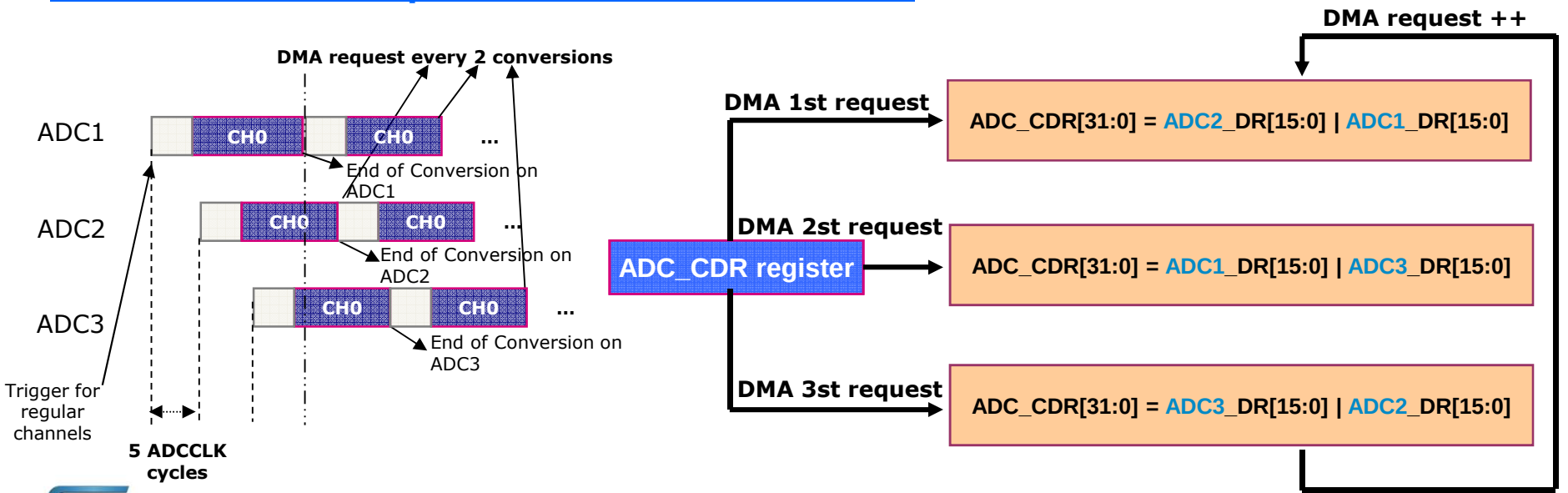




ADC and DMA(3/4): ADC_DMA mode “2”

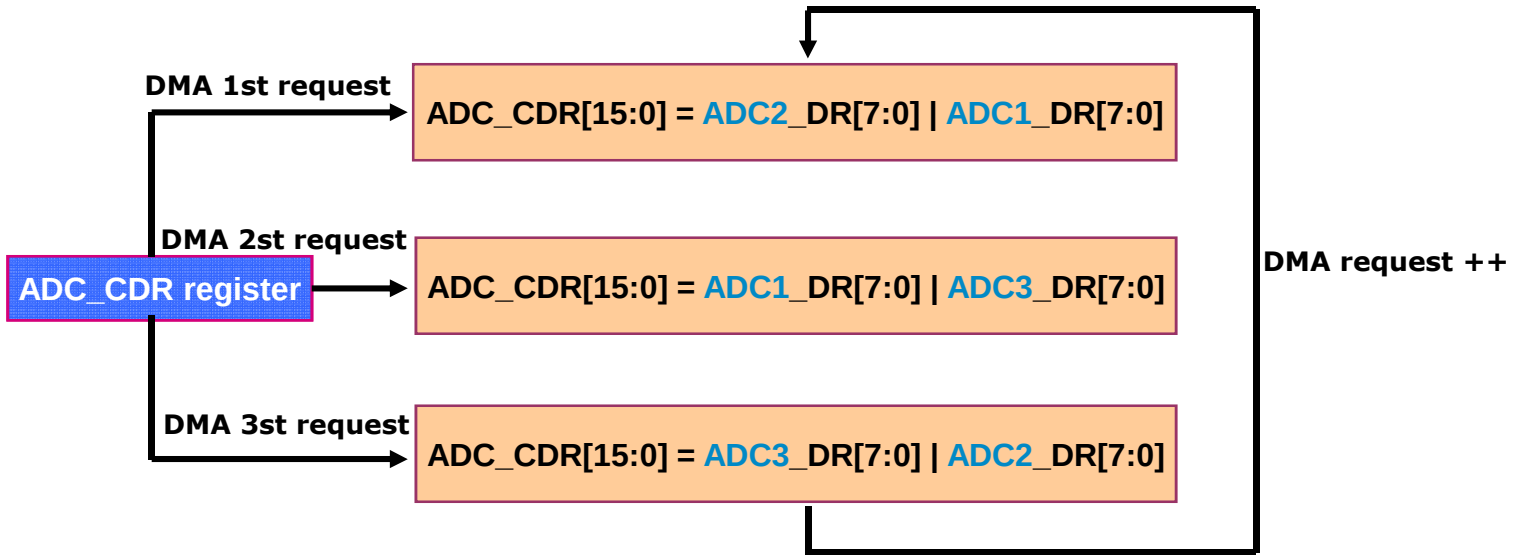
- Used in interleaved Dual/Triple ADC mode and in regular simultaneous Dual ADC mode.
- DMA ensure the access to the converted regular channel values which are stored into the common data register ADC_CDR.
- On each DMA request, two half-words representing two ADC-converted data items are transferred as a word.
- DMA bits in common control register ADC_CCR must be set at 0b10.

DMA transfer in Triple ADC interleaved mode:



ADC and DMA(4/4): ADC_DMA mode “3”

- Used in interleaved Dual/Triple ADC mode and in regular simultaneous Dual ADC mode with 6-bits and 8-bits resolutions.
- DMA ensure the access to the converted regular channel values which are stored into the common data register ADC_CDR.
- On each DMA request, two bytes representing two ADC-converted data items are transferred as a half-word.
- DMA bits in common control register ADC_CCR must be set at 0b11.



ADC Flags and interrupts

OVR : Overrun detection when regular converted data are lost

EOC : « Regular channel end of conversion » to indicate (depending on EOCS bit) the end of :

➔ a regular **CHANNEL** conversion

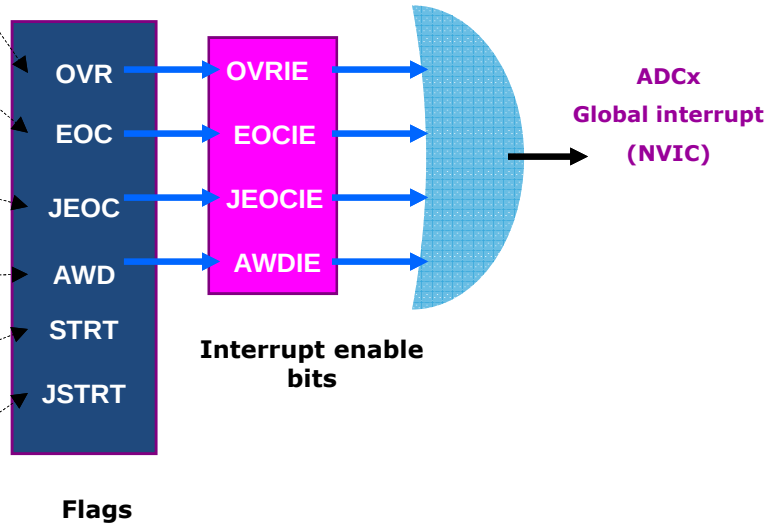
➔ sequence of regular **GROUP** conversions .

JEOC : « Injected channel end of conversion» to indicate at the end of injected **GROUP** conversion

AWD : « Analog watchdog » to indicate if the converted voltage crosses the programmed thresholds values.

STRT: « Regular channel start » to indicate when regular **CHANNEL** conversion starts.

JSTRT: « Injected channel start **»** to indicate hardware when injected **GROUP** conversion starts.



Flag for ADC regular channels

Flag for ADC Injected channels

General Flag for the ADC

FLEXIBLE STATIC MEMORY CONTROLLER (FSMC)

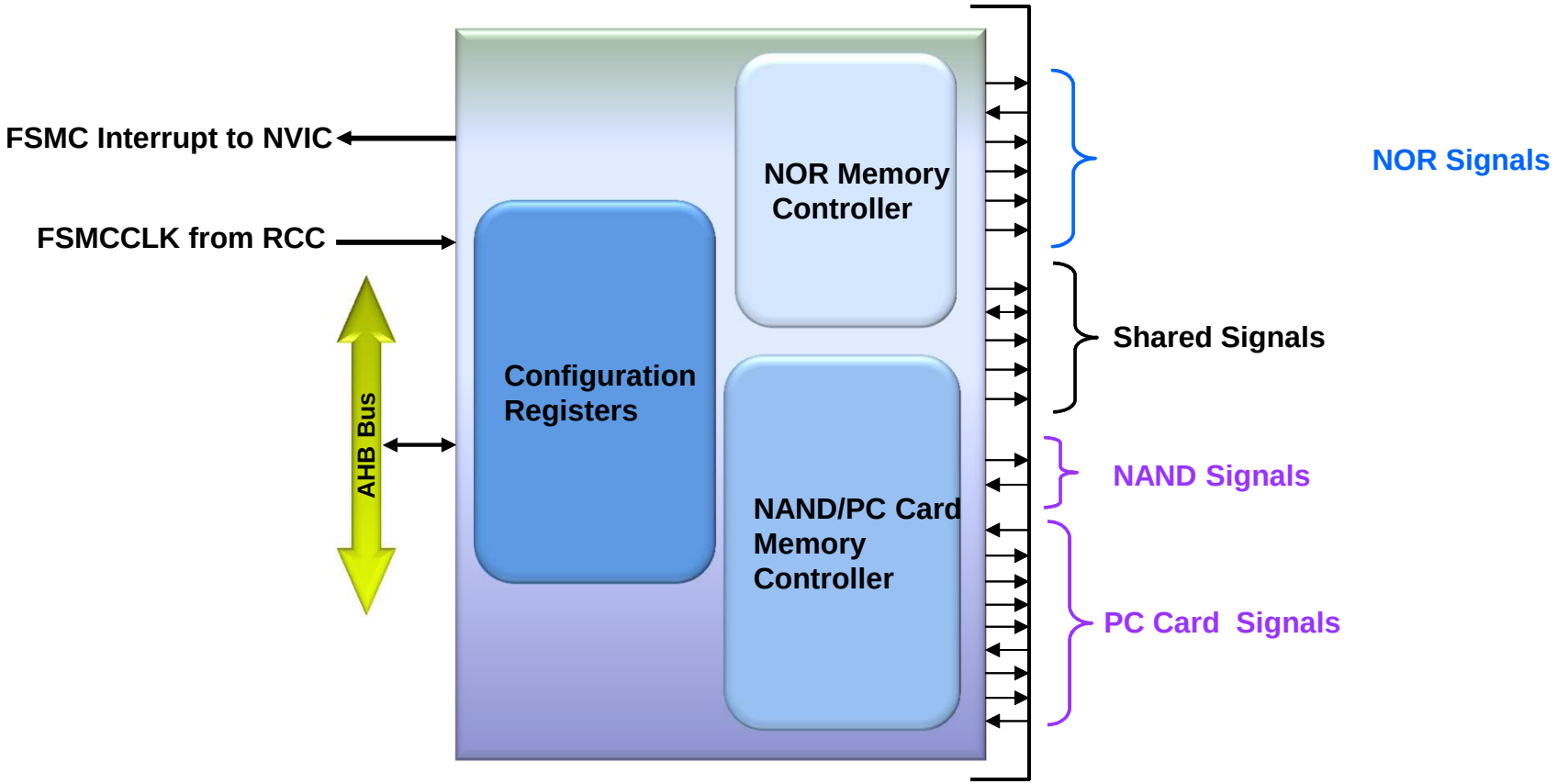
life.augmented

FSMC Features

- The Flexible Static Memory Controller has the following main features:
  - 4 Banks to support External memory
  - FSMC external access frequency is 60MHz when HCLK is at 168Hz
  - Independent chip select control for each memory bank
  - Independent configuration for each memory bank
  - Interfaces with static memory-mapped devices including:
    - static random access memory (SRAM)
    - read-only memory (ROM)
    - NOR/ OneNAND Flash memory
    - PSRAM
  - Interfaces parallel LCD modules: Intel 8080 and Motorola 6800
  - Supports burst mode access to synchronous devices (NOR Flash and PSRAM)
  - NAND Flash and 16-bit PC Cards
    - With ECC hardware up to 8 Kbyte for NAND memory
    - 3 possible interrupt sources (Level, Rising edge and falling edge)
  - Programmable timings to support a wide range of devices
  - External asynchronous wait control
  - Enhanced performance vs. STM32F10x

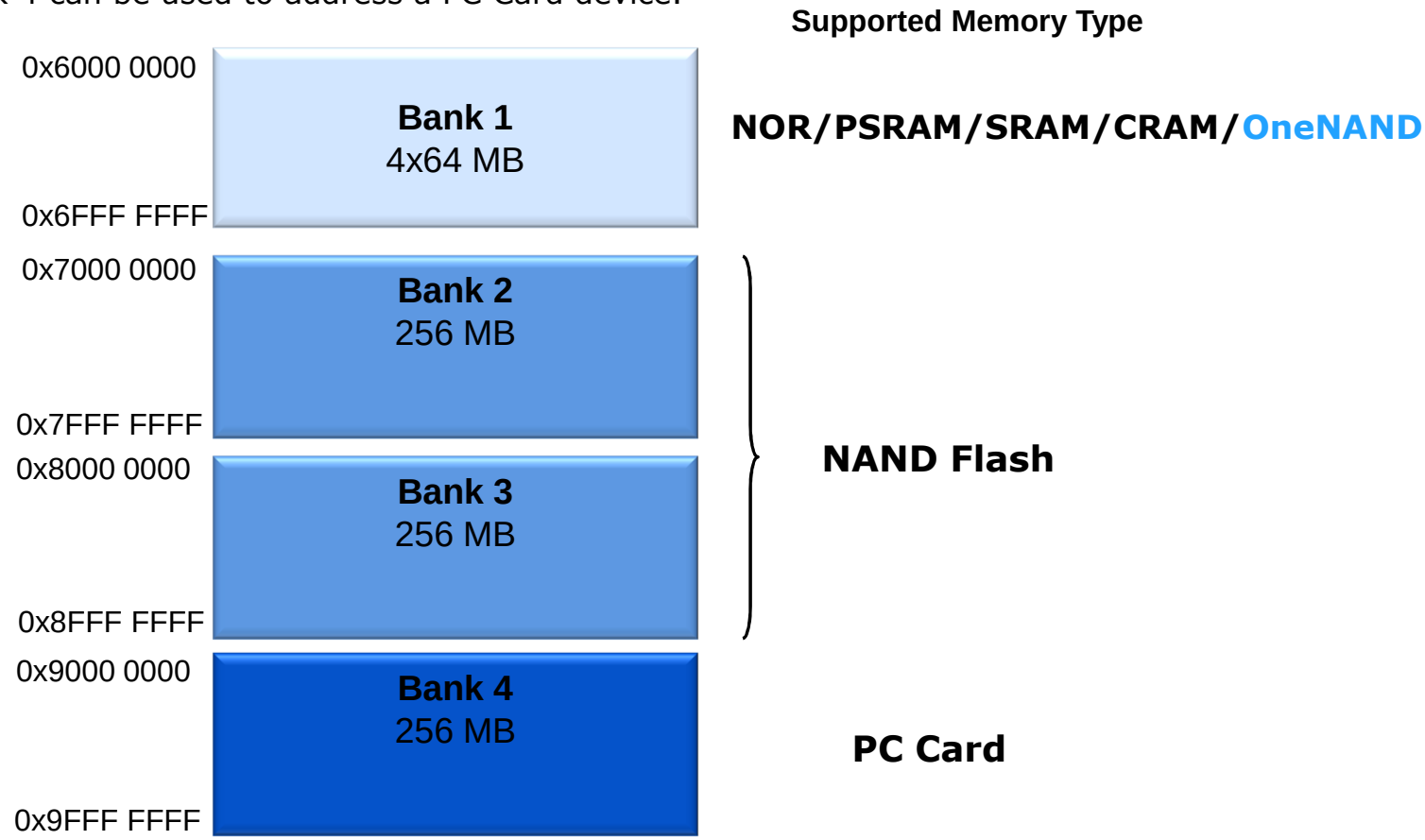
FSMC Block Diagram

- The FSMC consists of four main blocks:
  - The AHB interface (including the IP configuration registers)
  - The NOR Flash/PSRAM controller
  - The NAND Flash/PC Card controller
  - The external devices interface



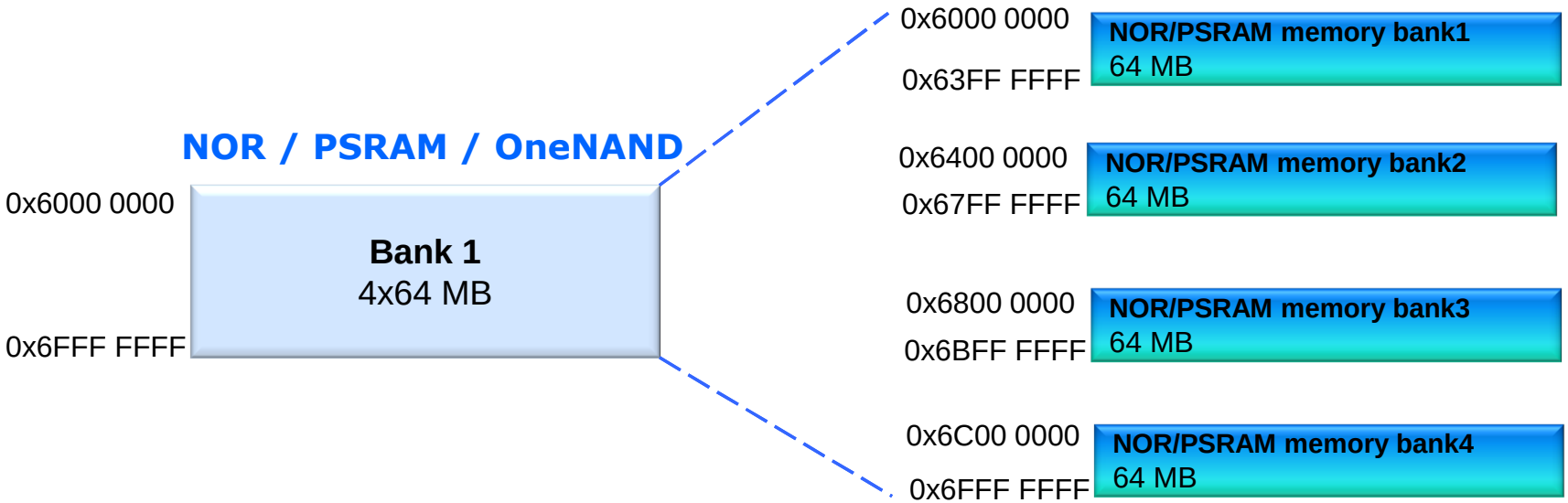
FSMC Bank memory mapping

- For the FSMC, the external memory is divided into 4 fixed size banks of 4x64 MB each:
  - Bank 1 can be used to address NOR Flash, OneNAND or PSRAM memory devices.
  - Banks 2 and 3 can be used to address NAND Flash devices.
  - Bank 4 can be used to address a PC Card device.



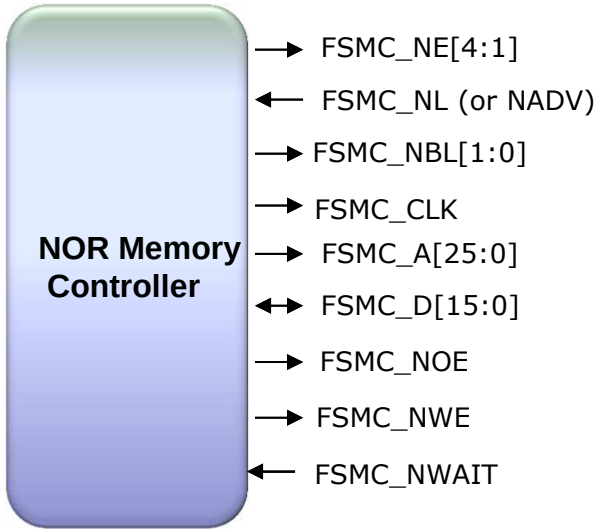
NOR/PSRAM Address mapping

- Bank1 is divided in 4 banks of 64 MB size to interface with 4 NOR/PSRAM external memory
  - NOR/PSRAM Bank selection is done through 4 chip select pins
  - NOR/PSRAM Bank can support the following memory types:
  - NOR FLASH: 8/16 bits synchronous/asynchronous, Multiplexed or non multiplexed
  - SRAM/ROM: 8/16 bits
  - CRAM/PSRAM: 8/16 bits synchronous/asynchronous
  - OneNAND: synchronous/asynchronous



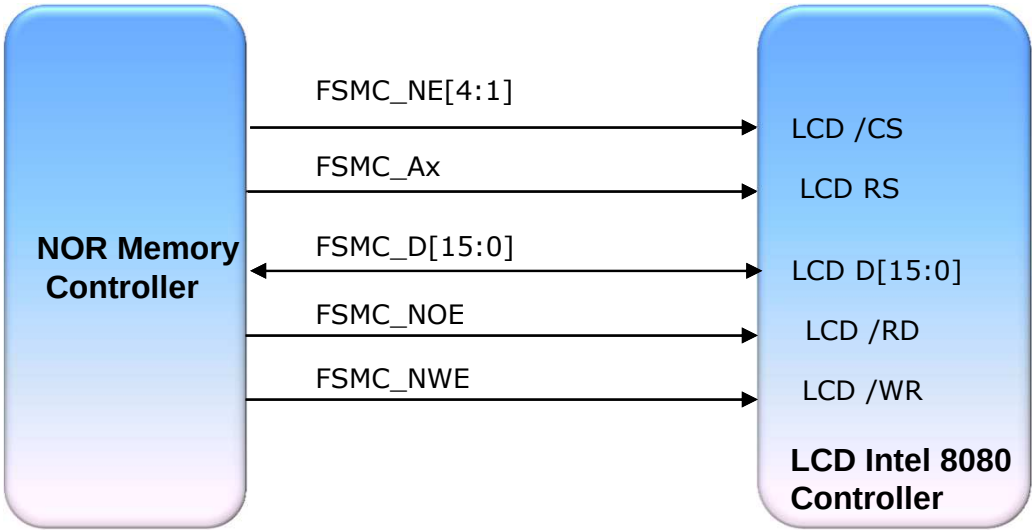
NOR/PSRAM interface signals

- The FSMC generates the appropriate signal to drive the following types of memories:
  - Asynchronous SRAM and ROM
    - 8-bit
    - 16-bit
  - PSRAM (Cellular RAM and Cosmo RAM) and OneNAND
    - Asynchronous mode
    - Burst mode
  - NOR Flash
    - Asynchronous mode or burst mode
    - Multiplexed or non multiplexed



- The FSMC outputs a unique chip select signal NE[4:1] per bank
- For synchronous accesses, the FSMC issues the clock (CLK)
- NOR Flash memories are addressed in 16-bit words.
  The maximum capacity is 512 Mbits
- The NOR Controller can be used in multiplexed mode: Lower address are multiplexed with data Bus

LCD modules interface signals



FSMC_Ax: where x can be (0..25)

LCD /RD: The ready signal indicates to the 8080 that valid memory or input data is available on the 8080 data bus.

LCD /WR: The /WR signal is used for memory write or I/O output control. The data on the data bus is stable while the /WR is active low (/WR = 0).

LCD RS: RAM Data/ Register Data Selection

LCD /CS: Chip Select

LCD D[0:15]: Bidirectional data bus

All LCD Signals are controlled by FSMC

Application Note is available from www.st.com/mcu

AN2790: TFT LCD interfacing with the High-density STM32Fx FSMC

life.augmented

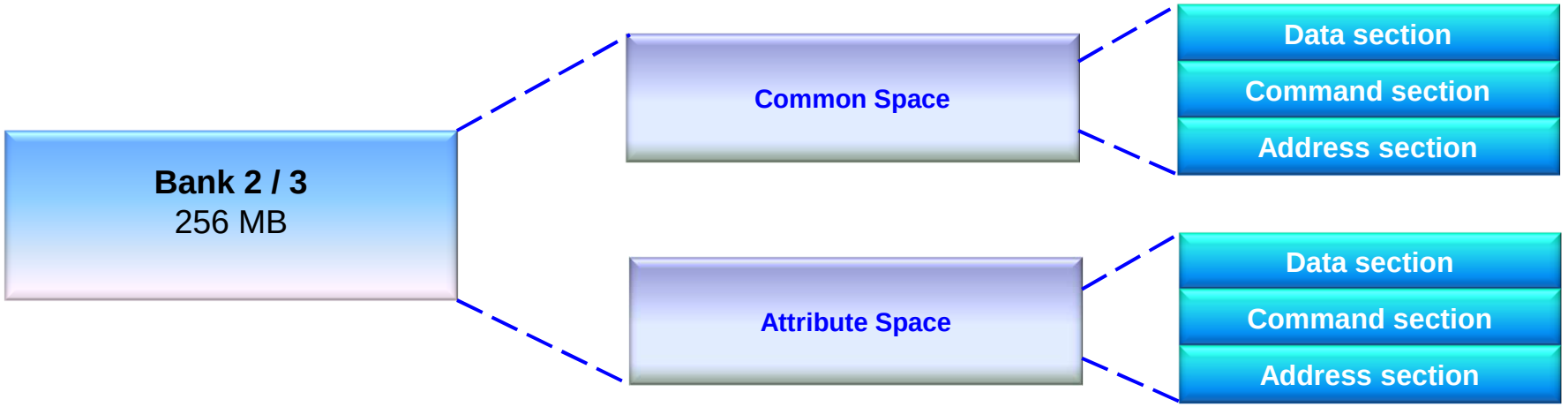
NOR/PSRAM Controller

- The FSMC NOR/PSRAM Controller can control the following features:
  - Memory type:
    - NOR, OneNand, CRAM, SRAM, etc…
  - Memory data width:
    - 8 bit or 16 bit
  - Synchronous or asynchronous Mode
  - Multiplexed or not multiplexed memory
  - Wait feature enable or disable
  - Extended mode: read timings and protocol are different to write.

- The FSMC NOR/PSRAM Controller used also to set the different timings of the memory connected to the bank
  - Address setup phase duration
  - Address-hold phase duration
  - Data-phase duration
  - Bus turnaround phase duration
  - Clock divide ratio
  - Data latency (for synchronous burst NOR Flash)
  - Access mode

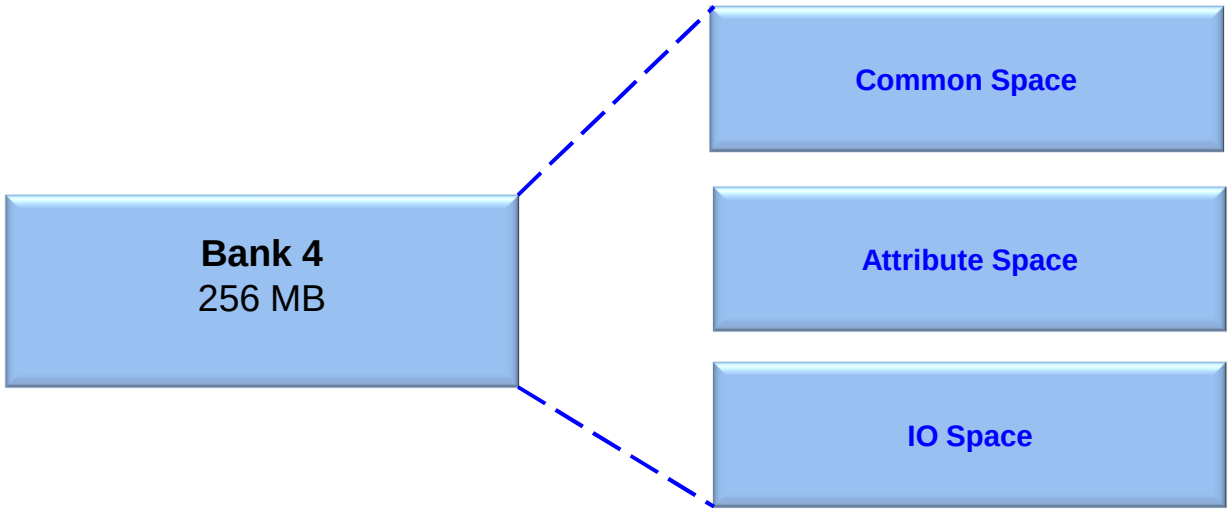
NAND Address mapping

- Bank2 and 3 are used t support the NAND Flash memory
  - Common memory space
  - Attribute memory space
- Each memory space is divided into 3 subsection:
  - Data section: first 32KB in the common/attribute memory space
    - Used to read or write data
  - Command section: second 32 Kbytes in the common / attribute memory space
    - Used to send a command to NAND Flash memory
  - Address section: next 64 Kbytes in the common / attribute memory space
    - Used to specify the NAND Flash address that must be read or written



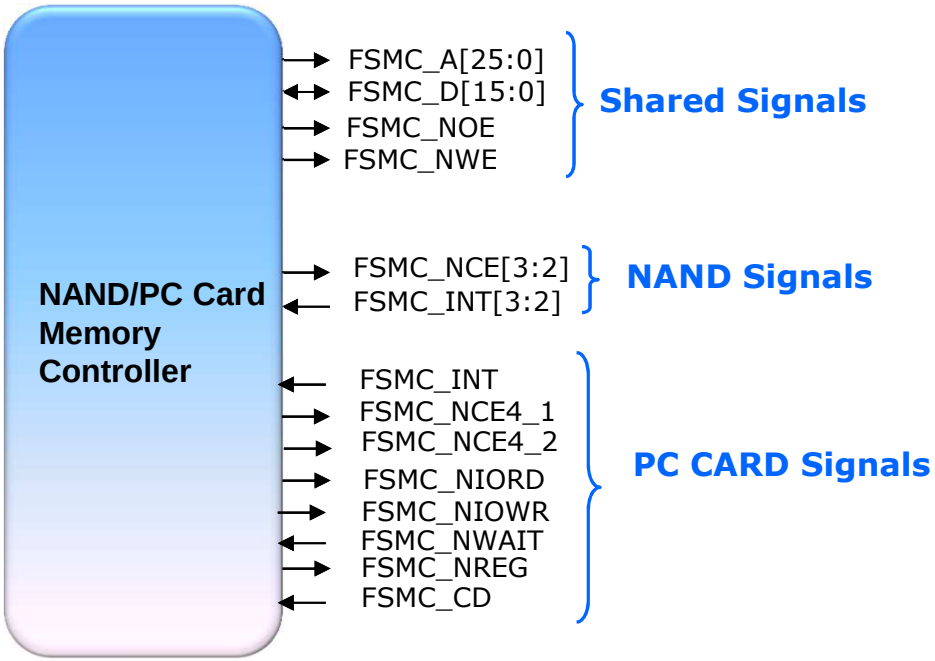
PCCARD Address mapping

- Bank4 is used to support the PC CARD Memory
  - Common memory space
    - To support the PC Card Memory Mode
  - Attribute memory space
  - IO memory space
    - To support PC Card I/O Mode
    - To support True IDE Mode



NAND/PCCARD interface signals

- The FSMC generates the appropriate signal to drive the following types of memories:
  - PC Card, CompactFlash, CF+ or PCMCIA
  - NAND Flash
    - 8-bit
    - 16-bit



NAND/PCCARD Controller

- The FSMC NAND PCCARD Controller can control the following features:
  - ECC hardware acceleration for pages raging from 256 to 8192
  - 3 Interrupt sources for PCCARD Bank and for each NAND Bank:
    - Rising edge,
    - Falling edge,
    - Level of the signal coming from the External memory Ready/nBusy output pin
  - Wait feature management
    - The controller waits for the NAND Flash to be ready (R/NB signal high) to become active, before starting a new access.

- The FSMC NAND PCCARD Controller used also to set the different timings of the memory connected to the bank
  - For each memory space a set of parametes can be configured
    - Setup time: Time to set up the address before the command assertion
    - Wait time: Time to assert the command
    - Hold time: Time to hold address after the command deassertion
    - Data bus HiZ time: Time from address valid to data bus drive

Timers

GENERAL PURPOSE TIMERS (TIM)

life.augmented

STM32F4xx Timer features overview (1/2)

	Counter resolution	Counter type	Prescaler factor	DMA	Capture Compare Channels	Complementary output	Synchronization	
							Master Config	Slave Config
Advanced TIM1 and TIM8	16 bit	up, down and up/down	1..65536	YES	4	3	YES	YES
General purpose (1) TIM2 and TIM5	32 bit	up, down and up/down	1..65536	YES	4	0	YES	YES
General purpose TIM3 and TIM4	16 bit	up, down and up/down	1..65536	YES	4	0	YES	YES
Basics TIM6 and TIM7	16 bit	up	1..65536	YES	0	0	YES	NO
1 Channel (2) TIM10..11 and TIM13..14 (2)	16 bit	up	1..65536	NO	1	0	YES(OC signal)	NO
2 Channel(2) TIM9 and TIM12	16 bit	up	1..65536	NO	2	0	NO	YES

(1) Same as STM32F2xx 32-Bit Timers

(2) These Timers are identical to STM32F2xx and XXL Timers

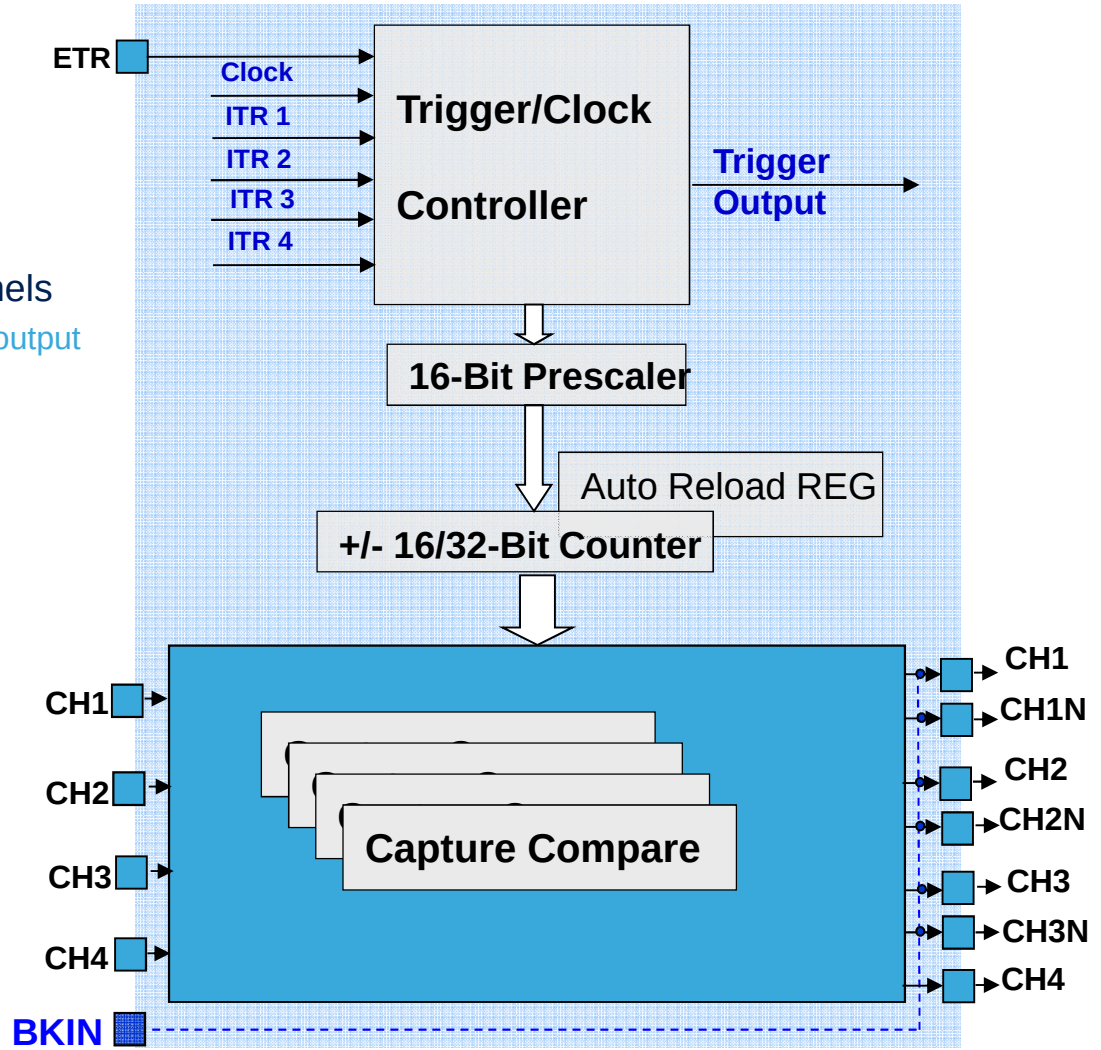
life.augmented

STM32F4xx Timer features overview 2/2

	Counter clock source	Output Compare	PWM	Input Capture	PWMI	OPM	Encoder interface	Hall sensor interface	XOR Input
Advanced TIM1 and TIM8	-Internal clock APB2 -External clock: ETR/TI1/TI2/TI3/TI4 pins -Internal Trigger: ITR1/ITR2/ITR3/ITR4 -Slave mode	7 channels	7 channels	4 channels	2 channels	2 channels	Yes	Yes	Yes
General Purpose TIM2 and TIM5	-Internal clock APB1 -External clock: ETR/TI1/TI2/TI3/TI4 pins -Internal Trigger: ITR1/ITR2/ITR3/ITR4 -Slave mode	4 channels	4 channels	4 channels	2 channels	2 channels	Yes	No	Yes
General Purpose TIM3 and TIM4	-Internal clock APB1 -External clock: ETR/TI1/TI2/TI3/TI4 pins -Internal Trigger: ITR1/ITR2/ITR3/ITR4 -Slave mode	4 channels	4 channels	4 channels	2 channels	2 channels	Yes	No	Yes
Basics TIM6 and TIM7	-Internal clock APB1	No	No	No	No	No	No	No	No
1 Channel TIM10/11 and 13/14	-Internal clock APB1/APB2	1 Channel	1 Channel	1 Channel	No	No	No	No	No
2 Channel TIM9 and TIM12	-Internal clock APB1/APB2 -External clock: TI1/TI2/TI3/TI4 pins -Internal Trigger: ITR1/ITR2/ITR3/ITR4 -Slave mode	2 channels	2 channels	2 channels	2 channels	2 channels	No	No	No

Features overview

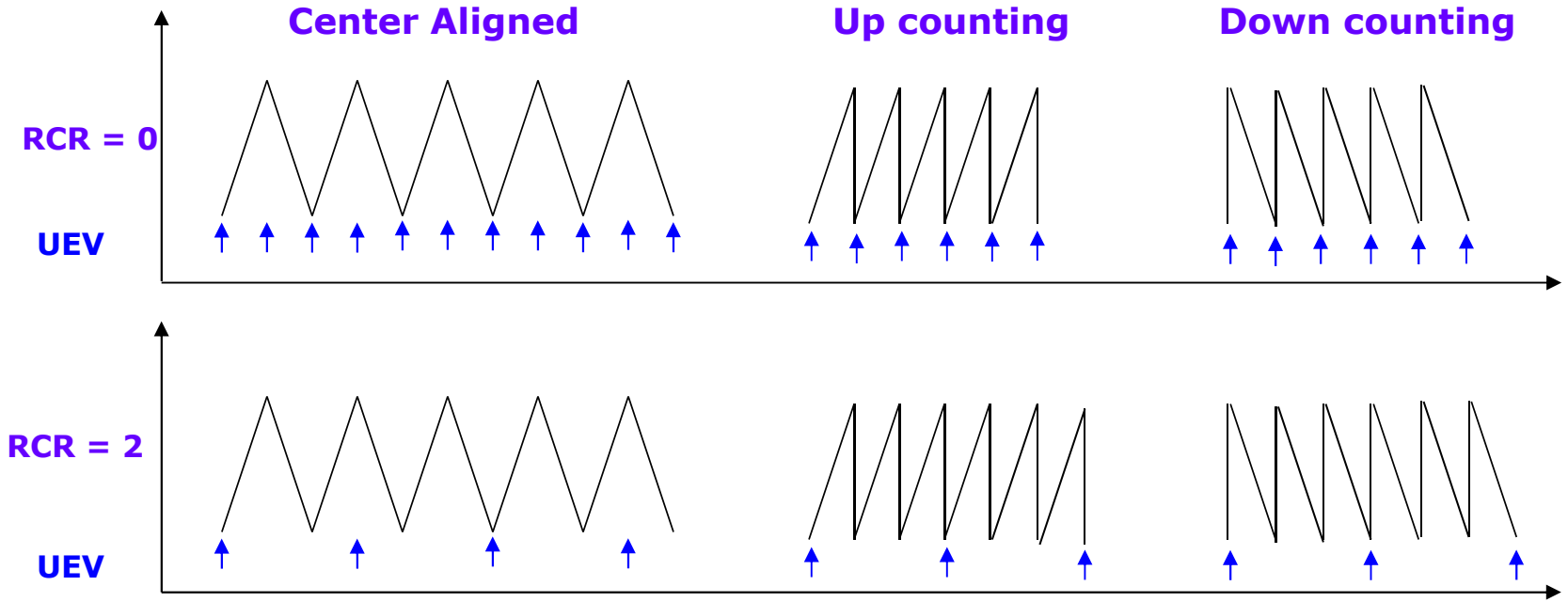
- General Purpose Feature
- 16/32-bit Counter
  - Auto Reload
  - Up, down and centered counting
  - modes
- 4x 16 High resolution Capture Compare channels
  - Programmable direction of the channel: input/output
  - Output Compare: Toggle, PWM
  - Input Capture
  - PWM Input Capture
- Synchronization
- Up to 8 IT/DMA Requests
- Motor Control Specific Feature
- OC Signal Management
  - 6 Complementary outputs
  - Dead-time management
  - Repetition Unit
- Encoder Interface
- Hall sensor Interface
- Embedded Safety features
  - Break sources: BKIN pin/ CSS
  - Lockable unit configuration



Counter Modes

- There are three counter modes:
  - **Up counting mode**
  - **Down counting mode**
  - **Center-aligned mode**

- When using the Repetion Counter (case of TIM1 and TIM8 only)



Update Event

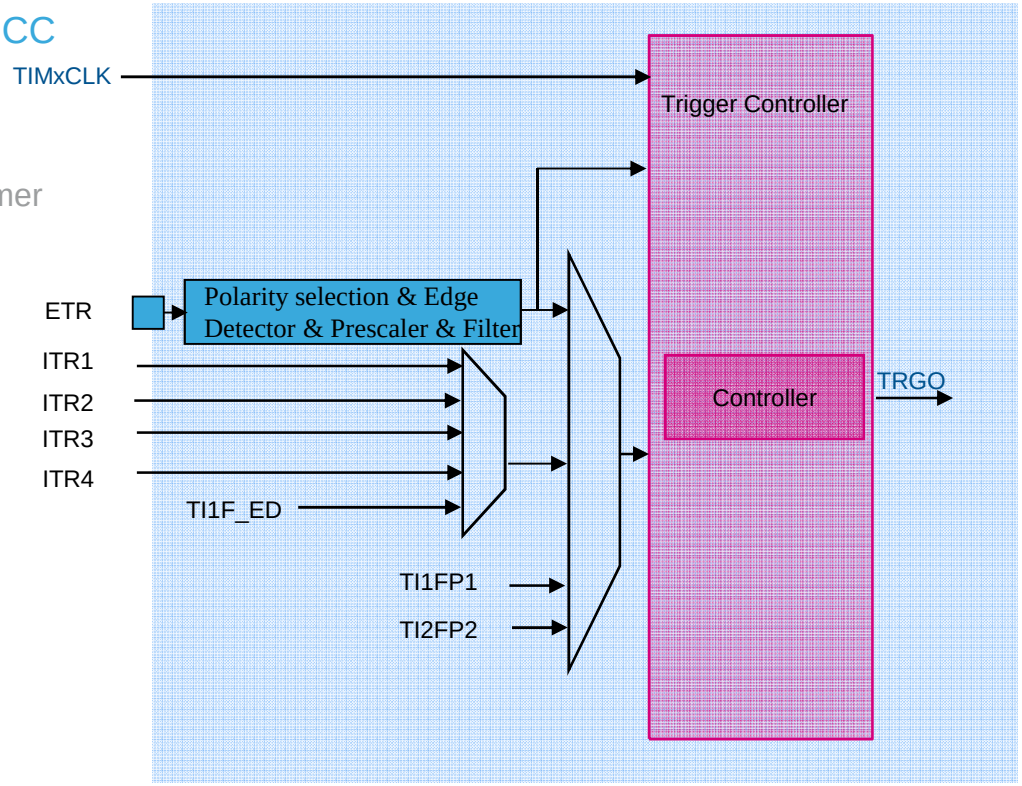
- The content of the preload register is transferred in the shadow register (depending on the auto-reload preload is enable or not):
  - Immediately
  - At each update event UEV

- The Update Event is generated:
  - when the counter reaches overflow/undeflow
  - when the Repetition counter equals to 0 (only for TIM1)
  - By software by setting the UG (Update Generation) bit

- The Update event UEV requests can be selected as follow:
  - The requests are sent only when the counter reachs the overflow/underflow.
  - The request are sent when (counter overflow/underflow or set of UG bit or update generation through the slave mode controller

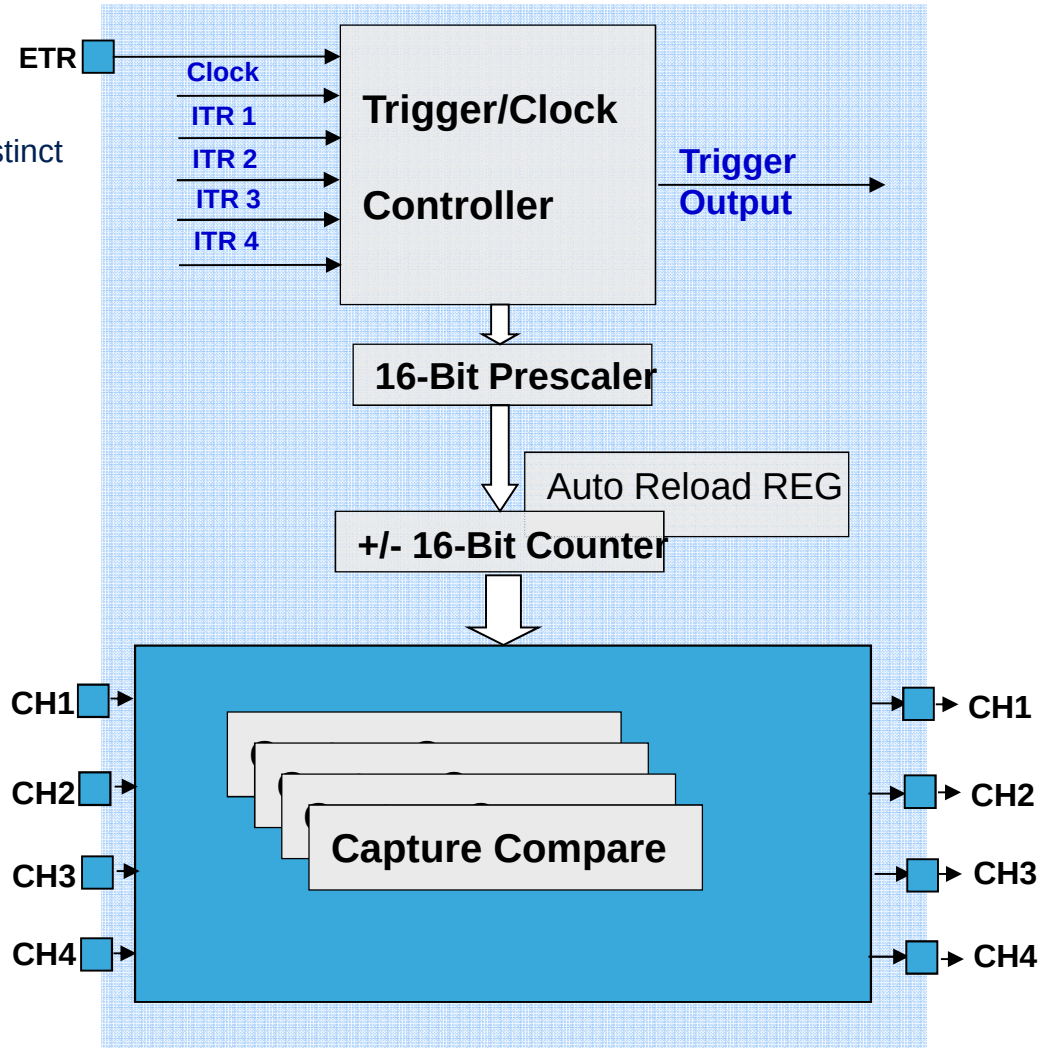
Counter Clock Selection

- Clock can be selected out of 8 sources
  - Internal clock TIMxCLK provided by the RCC
  - Internal trigger input 1 to 4:
    - ITR1 / ITR2 / ITR3 / ITR4
    - Using one timer as prescaler for another timer
  - External Capture Compare pins
    - Pin 1: TI1FP1 or TI1F_ED
    - Pin 2: TI2FP2
  - External pin ETR
    - Enable/Disable bit
    - Programable polarity
    - 4 Bits External Trigger Filter
    - External Trigger Prescaler:
      - Prescaler off
      - Division by 2
      - Division by 4
      - Division by 8



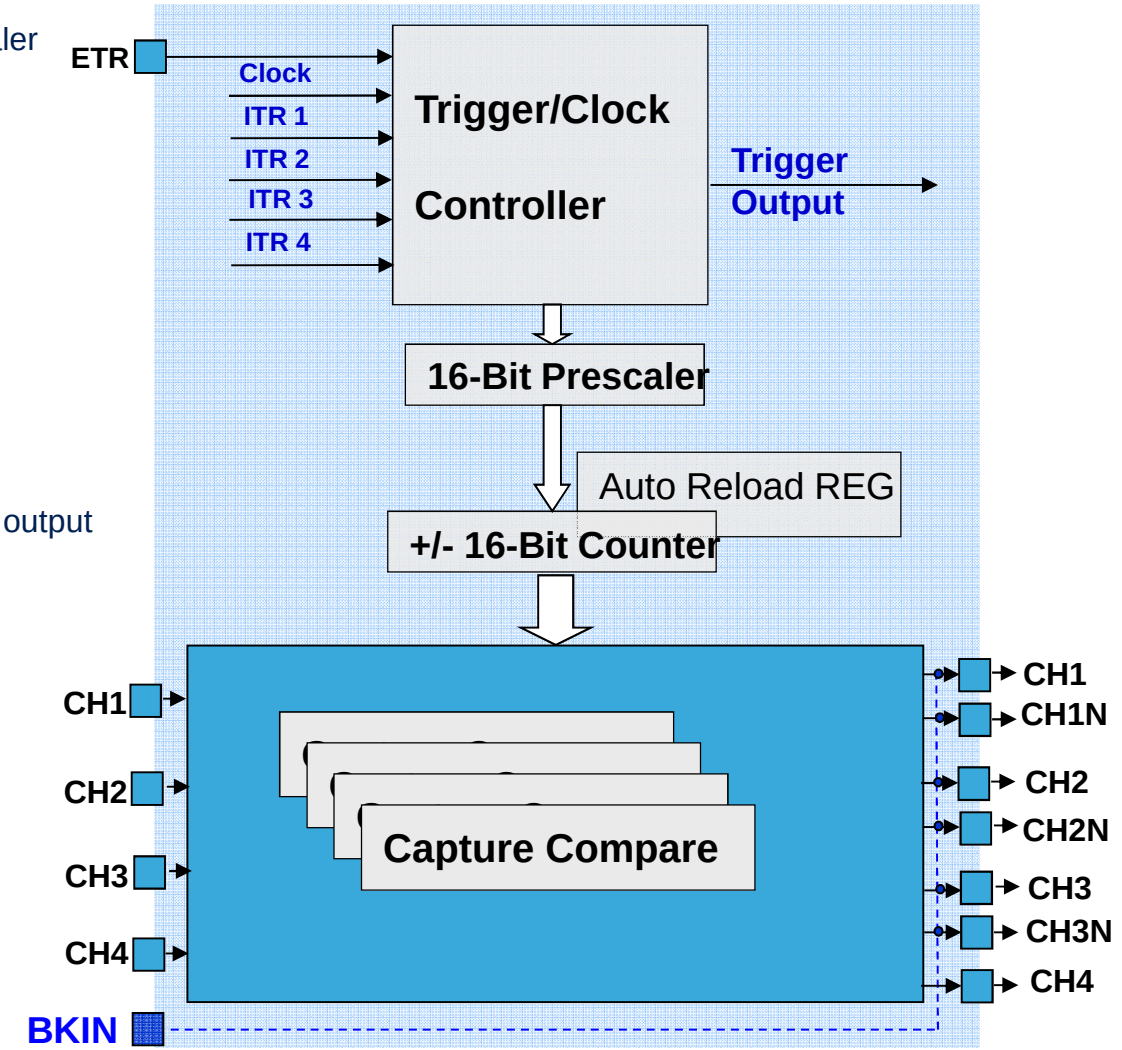
General Purpose timer Features overview

- TIM2, 3, 4 and 5 on Low Speed APB (APB1)
- Internal clock up to **84 MHz** (if AHB/APB1 prescaler distinct from 1)
- 16-bit Counter for TIM3 and 4
- 32-bit Counter for TIM2 and 5
  - Up, down and centered counting modes
  - Auto Reload
- 4 x 16 High resolution Capture Compare Channels
  - Programmable direction of the channel: input/output
  - Output Compare
  - PWM
  - Input Capture, PWM Input Capture
  - One Pulse Mode
- Synchronization
  - Timer Master/Slave
  - Synchronisation with external trigger
  - Triggered or gated mode
- Encoder interface
- 6 Independent IRQ/DMA Requests generation
  - At each Update Event
  - At each Capture Compare Events
  - At each Input Trigger



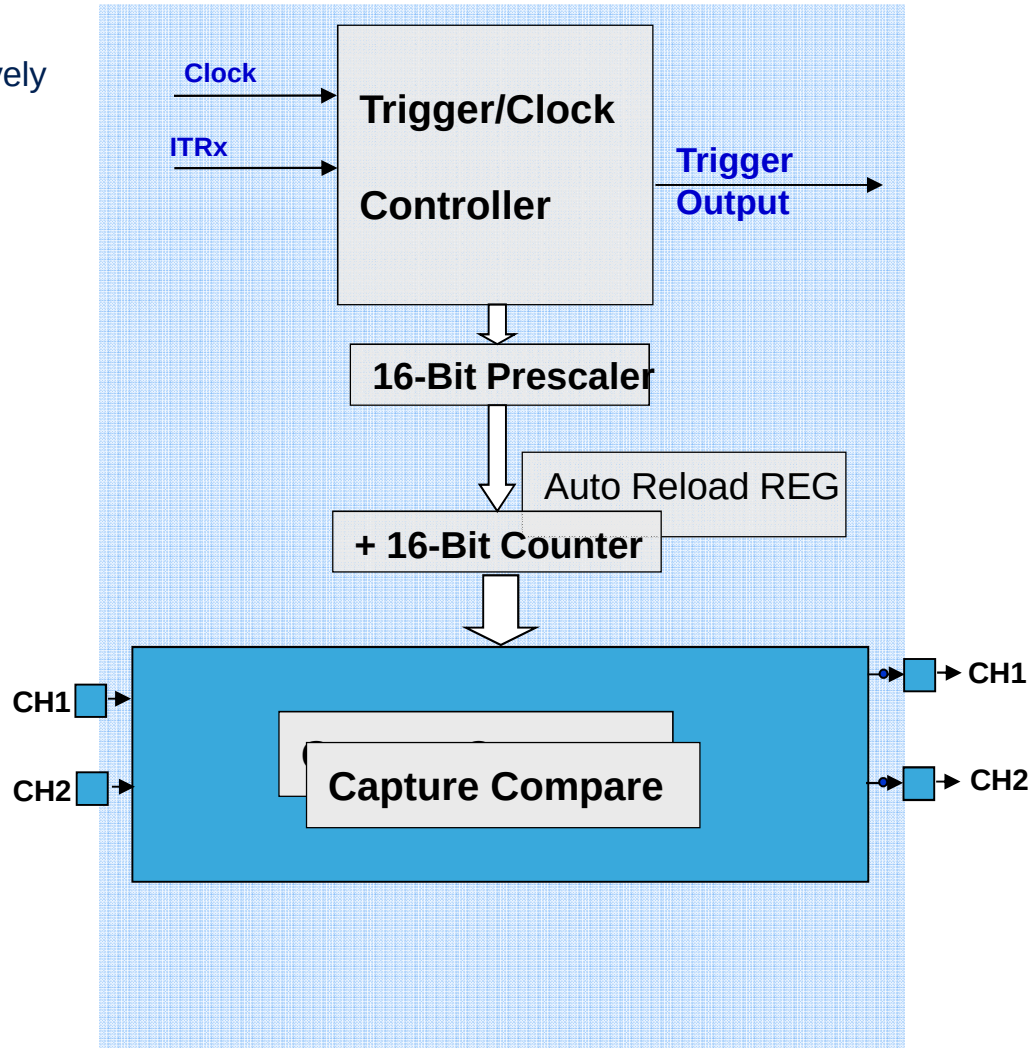
Advanced timer Features overview

- TIM1 and TIM8 on High Speed APB (APB2)
- Internal clock up to **168 MHz** (if AHB/APB2 prescaler distinct from 1)
- 16-bit Counter
  - Up, down and centered counting modes
  - Auto Reload
- 4 x 16 High resolution Capture Channels
  - Output Compare
  - PWM
  - Input Capture, PWM input Capture
  - One Pulse Mode
- 6 Complementary outputs: Channel1, 2 and 3
- Output Idle state selection independently for each output
- Polarity selection independently for each output
- Programmable PWM repetition counter
- Hall sensor interface
- Encoder interface
- 8 Independent IRQ/DMA Requests Generation
  - At each Update Event
  - At each Capture Compare Events
  - At each Trigger Input Event
  - At each Break Event
  - At each Capture Compare Update
- Embedded Safety features
  - Break input
  - Lockable unit configuration: 3 possible Lock level.



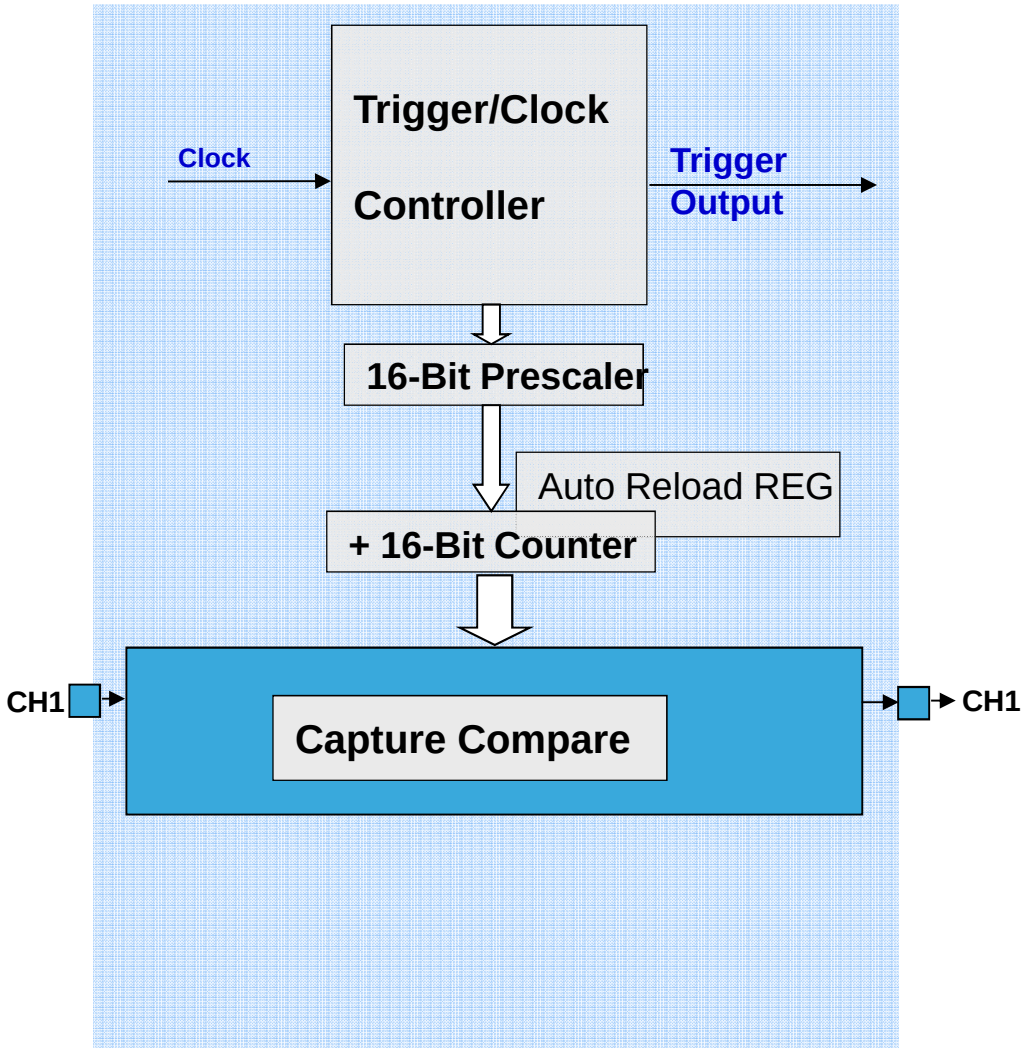
General Purpose 2 Channels timer (TIM9 & TIM12)
Features overview

- TIM9 on High speed APB (APB2) and TIM12 on Low Speed APB (APB1)
- Internal clock up to **168 MHz** and **84 MHz** respectively
- 16-bit Counter
  - Up counting mode
  - Auto Reload
- 2 x 16 High resolution Capture Compare Channels
  - Programmable direction of the channel: input/output
  - Output Compare
  - PWM
  - Input Capture, PWM Input Capture
  - One Pulse Mode
- Synchronization Timer Master/Slave
  - Synchronization with external trigger
  - Triggered or gated mode
- Independent IRQ Requests generation
  - At each Update Event
  - At each Capture Compare Events
  - At each Input Trigger



General Purpose 1 Channels timer (TIM10..11 & TIM13..14) Features overview

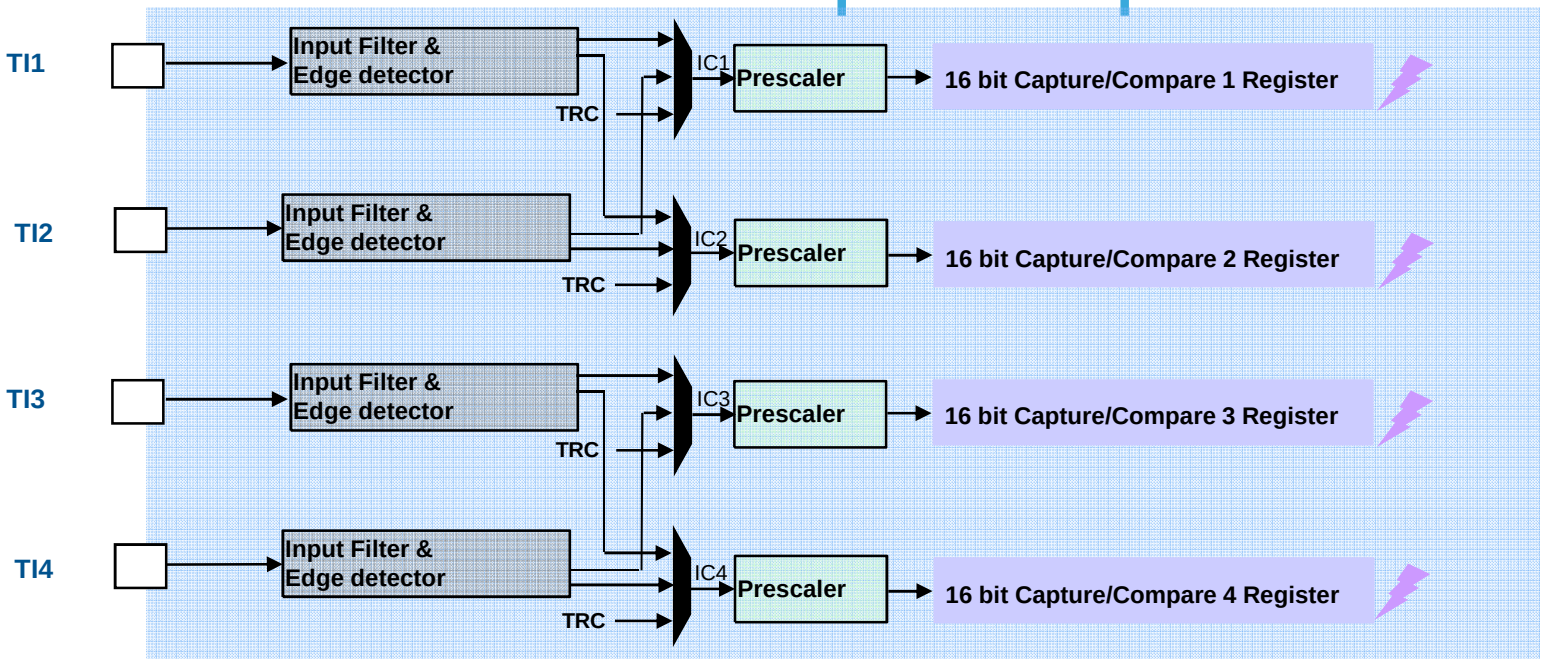
- TIM10..11 on High speed APB (APB2) and TIM13..14 on Low Speed APB (APB1)
- Internal clock up to **168 MHz for TIM10/11**
- Internal clock up to **84 MHz for TIM13/14**
- 16-bit Counter
  - Up counting mode
  - Auto Reload
- 2 x 16 High resolution Capture Compare Channels
  - Programmable direction of the channel: input/output
  - Output Compare
  - PWM
  - Input Capture
- Independent IRQ Requests generation
  - At each Update Event
  - At each Capture Compare Events



Capture Compare Array presentation

- The Capture Compare Array is composed of:
  - Capture Compare channels
    - 4 Identical for TIM2, 3, 4 and 5
    - 3 Channels with possible complementary signals generation + 1 no complementary Channel only for TIM1 and TIM8
    - 2 Identical for TIM9 and 12
    - Only One for TIM10, 11, 13 and 14
  - Break functionality only for TIM1 and TIM8
- Programmable direction of each channel: input/output use
- Each Channel is composed of:
  - Capture/Compare register
  - Input stage for capture:
    - 4 bits digital filter
    - Input Capture Prescaler:
      - Capture done at each an edge is detected
      - Capture done once every 2 events
      - Capture done once every 4 events
      - Capture done once every 8 events
  - Output stage for Compare:
    - Comparator
    - Output control
    - Dead Time generator for TIM1 and TIM8 only

Input Capture Mode



- IC1, IC2, IC3 and IC4 are specific as they can be independently mapped by software on TI1, TI2, TI3 or TI4.
- 4x16-bit capture compare registers are programmable to be used to latch the value of the counter after a transition detected by the corresponding Input Capture.
- When a capture occurs, the corresponding CCXIF flag is set and an interrupt or a DMA request can be sent if they are enabled.
- Possible set of an over-capture flag If a capture occurs while the CCxIF flag was already high

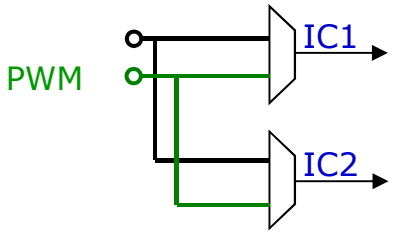
Note: The input capture circuit is sensitive to Rising edge, to Falling edge and to both rising and falling edges.

PWM Input Mode

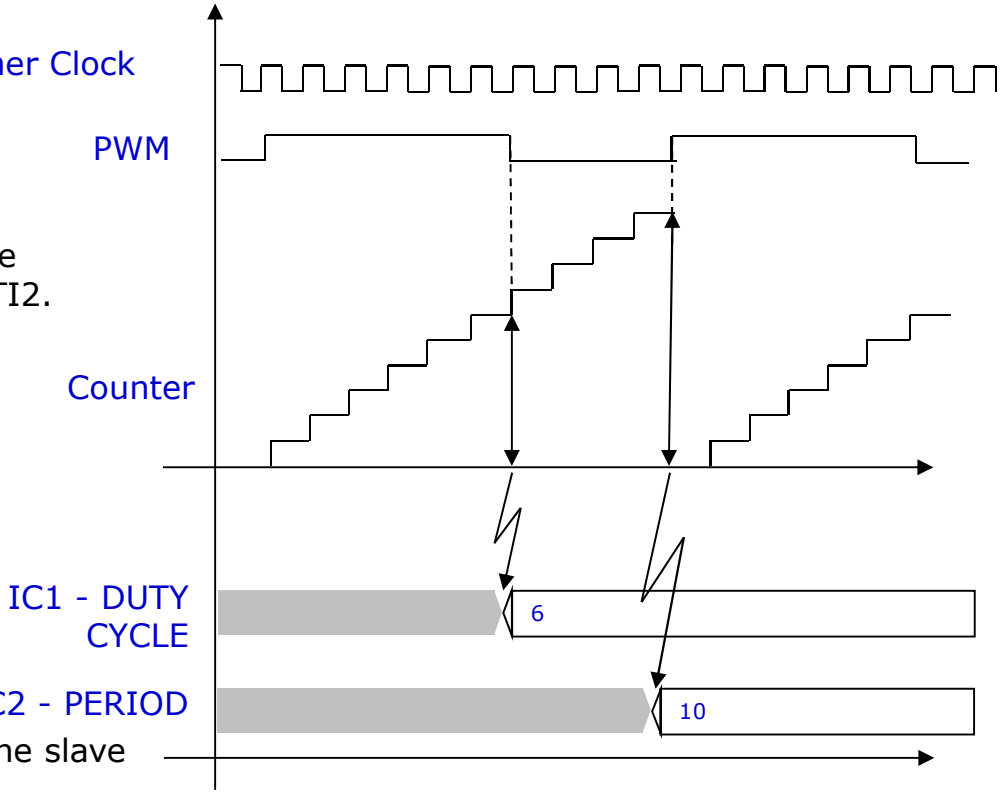
PWMI Configuration tips:

- IC1 and IC2 must be configured to be connected together to the PWM signal:

⇨ IC1 and IC2 are redirected internally to be mapped to the same external pin TI1 or TI2.



- IC1 and IC2 active edges must have opposite polarity.



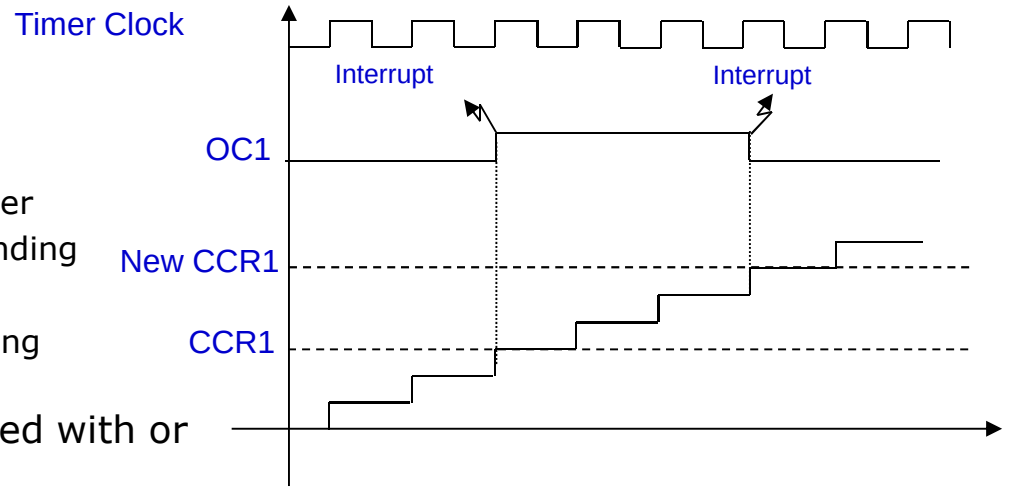
- IC1 or IC2 is selected as trigger input and the slave mode controller is configured in reset mode.

The PWM Input functionality enables the measurement of the period and the pulse width of an external waveform.

Output Compare Mode

The Output Compare is used to control an output waveform or indicate when a period of time has elapsed.

- When a match is found between the capture/compare register and the counter:
  - The corresponding output pin is assigned to the programmable Mode, it can be:
    - Set
    - Reset
    - Toggle
    - Remain unchanged
  - Set a flag in the interrupt status register
  - Generates an interrupt if the corresponding interrupt mask is set
  - Send a DMA request if the corresponding enable bit is set
- The CCRx registers can be programmed with or without preload registers

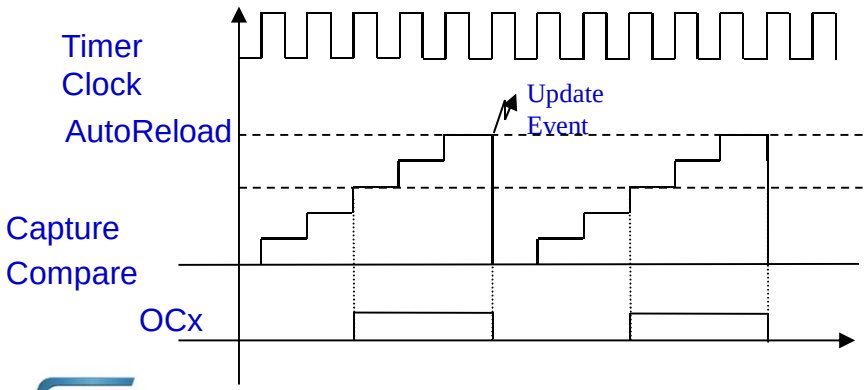


PWM Mode

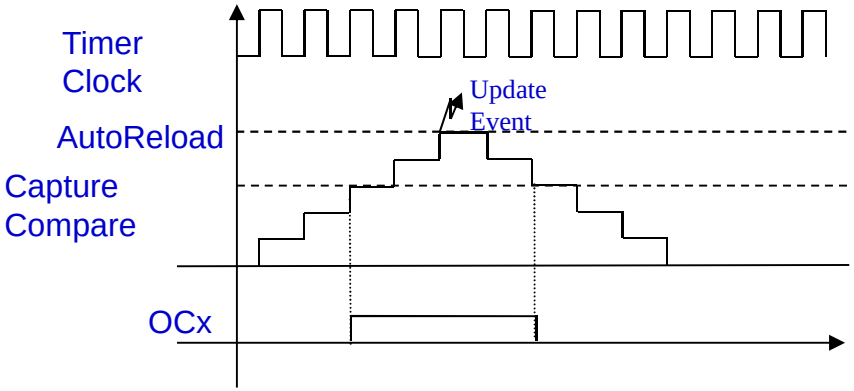
- The PWM mode allows to generate:
  - 7 independent signals for TIM1 and TIM8
  - 4 independent signals for TIM2, 3, 4 and 5
  - 2 independent signals for TIM9 and 12
  - 1 signals for TIM10, 11, 13 and 14
  - The frequency and a duty cycle determined as follow:
    - One auto-reload register to defined the PWM period.
    - Each PWM channel has a Capture Compare register to define the duty cycle.

  ➔ Example: to generate a 40 KHz PWM signal w/ duty cycle of 50% on TIM1 clock at 72MHz:
    - Load Prescaler register with 0 (counter clocked by TIM1CLK/(0+1)), Auto Reload register with 1799 and CCRx register with 899
- There are two configurable PWM modes:
  - Edge-aligned Mode
  - Center-aligned Mode

Edge-aligned Mode



Center-aligned Mode



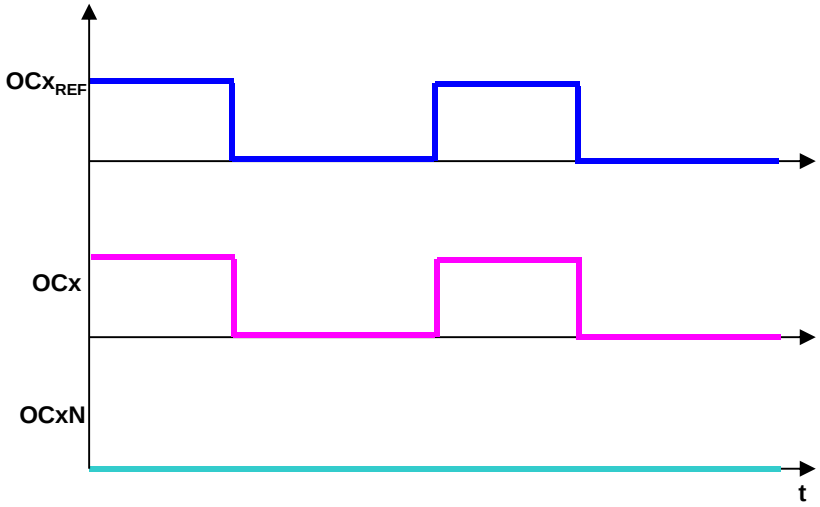
life.augmented

Advanced Control timer TIM1 and TIM8
Complementary PWM outputs for motor control

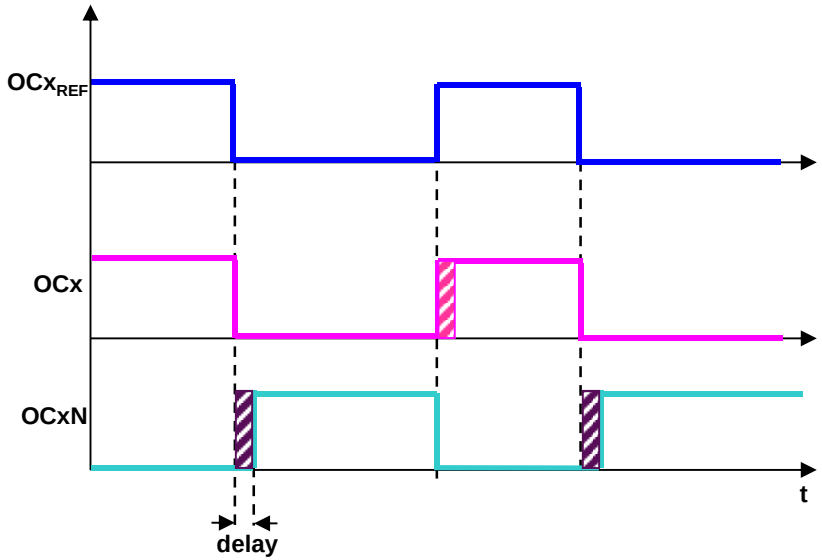
- This mode allows the TIM1 and TIM8 to:
  - Output two complementary signals for each three channels.
  - Output two independent signals for each three channels.
  - Manage the dead-time between the switching-off and the switching-on instants of the outputs.
- One reference waveform OCx_{REF} to generate 2 outputs OCx and OCxN for the three channels.
- Full modulation capability (0 and 100% duty cycle), edge or center-aligned patterns
- Dedicated interrupt and DMA requests for TIM1 period and duty cycles updating.
- Three programmable write protection levels
  - Level1: Dead Time and Emergency enable are locked.
  - Level2: Level1 + Polarities and Off-state selection for run and Idle state are locked.
  - Level3: Level2 + Output Compare Control and Preload are locked.

Examples of OCx waveform in Complementary PWM mode

Dead-time disabled

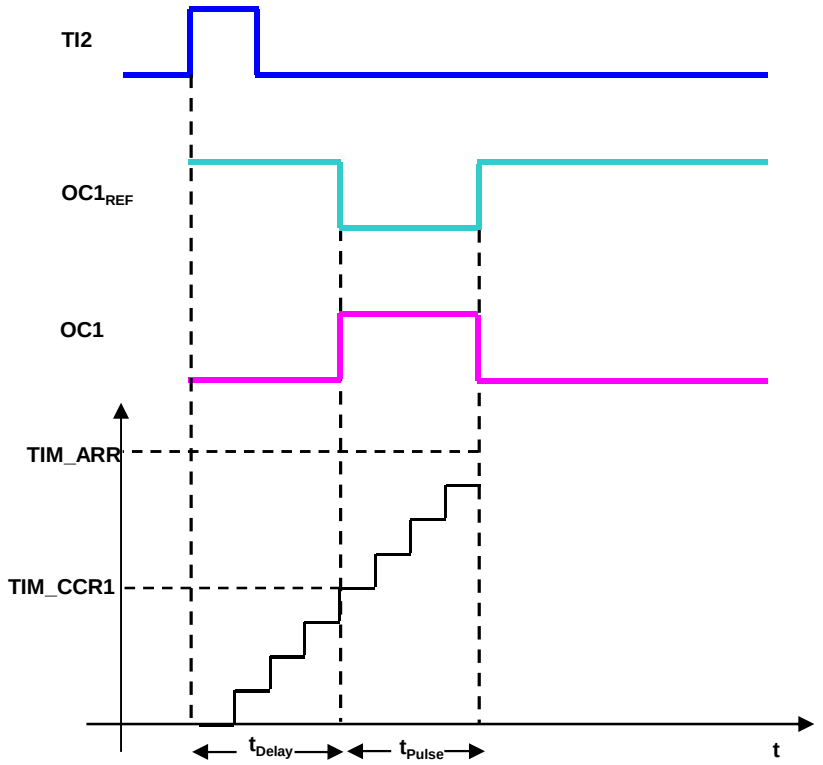


Dead-time enabled



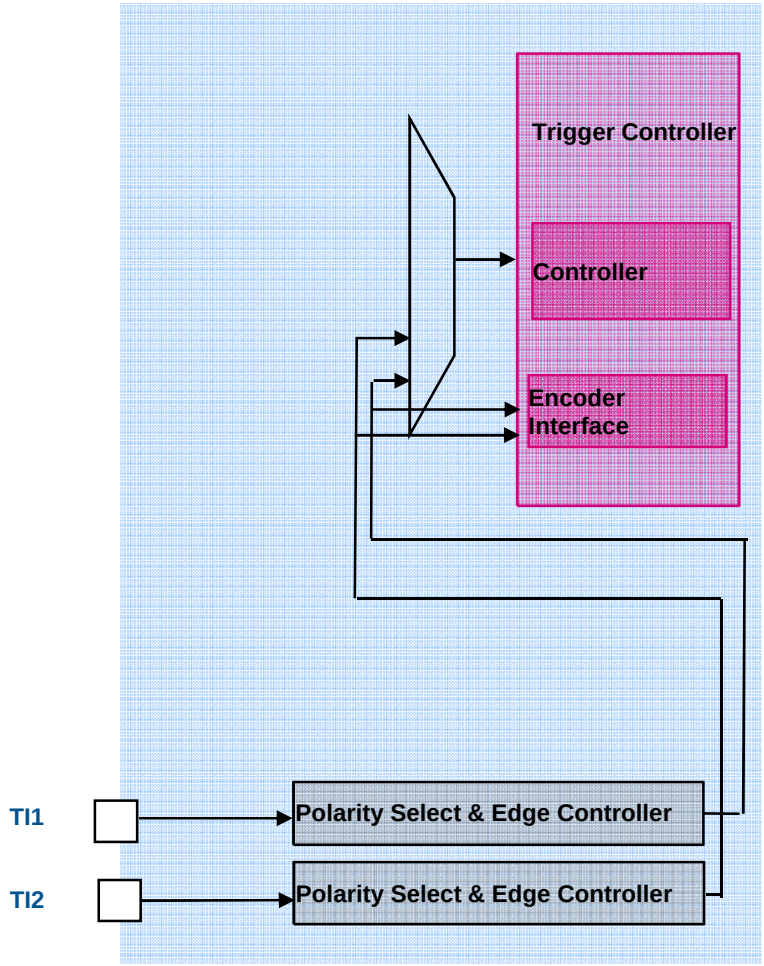
One Pulse Mode

- One Pulse Mode (OPM) is a particular case of the previous modes: Ouput Compare and Input Capture.
- It allows the counter to be started in response to a stimulus and to generate a pulse with a programmable length after a programmable delay.
- There are two One Pulse Mode waveforms selectable by software:
  - Single Pulse
  - Repetitive Pulse

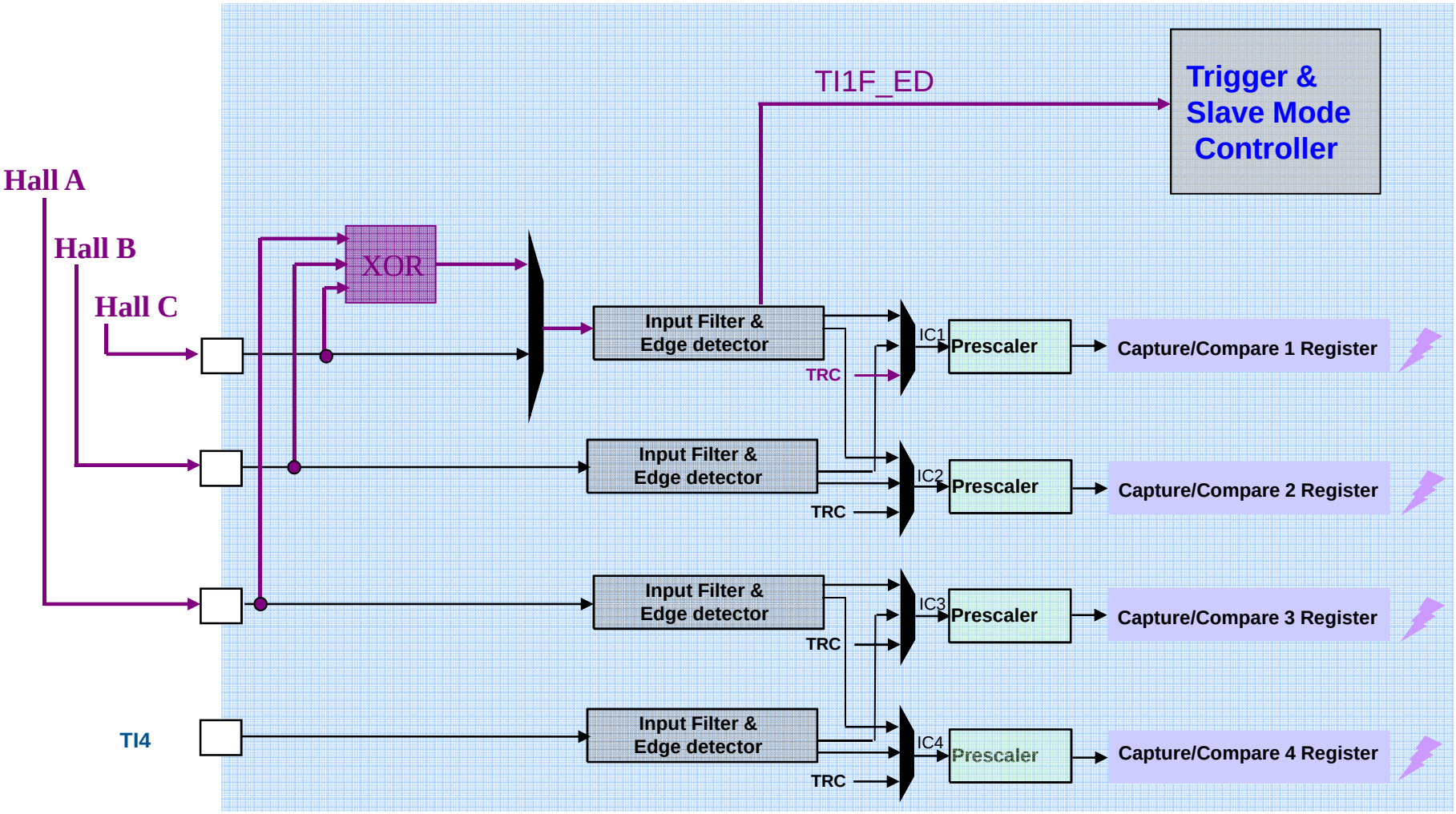


Encoder Interface

- Encoders are used to measure position and speed of motion systems (either linear or angular)
- The encoder interface mode acts as an external clock with direction selection
- The counter provides information on the current position (for instance angular position of an electric motor's rotor)
- To obtain dynamic information (speed, acceleration) on must measure the number of counts between two periodic events, generated by another timer
- Encoders and Microcontroller connection example:
  - An external incremental encoder can be connected directly to the MCU without external interface logic.
  - The third encoder output which indicates the mechanical zero position, may be connected to an external interrupt and trigger a counter reset.



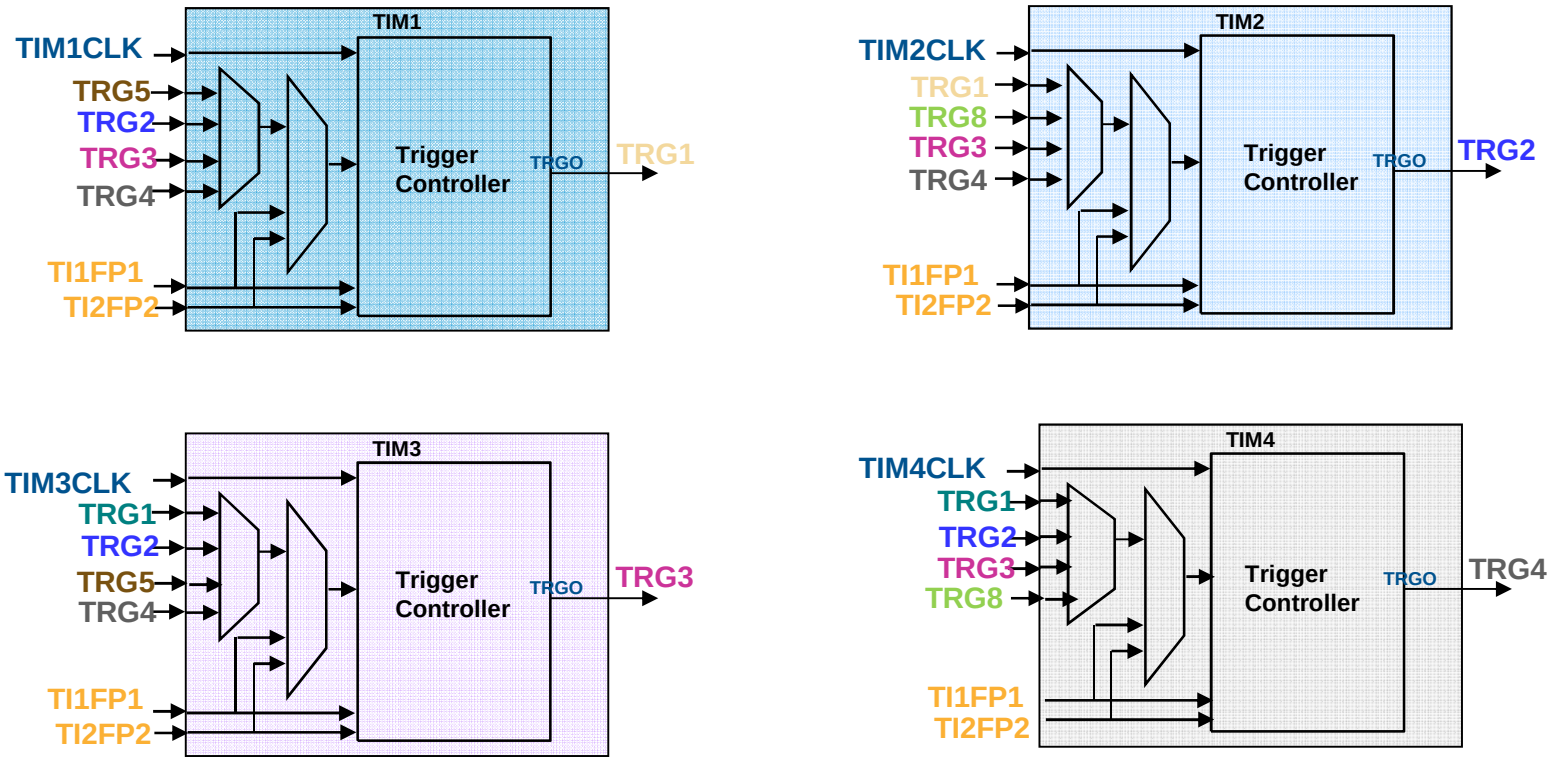
Hall sensor Interface



TIMs Synchronization(1/2)

- The eight Timers are linked together for timer synchronization or chaining.

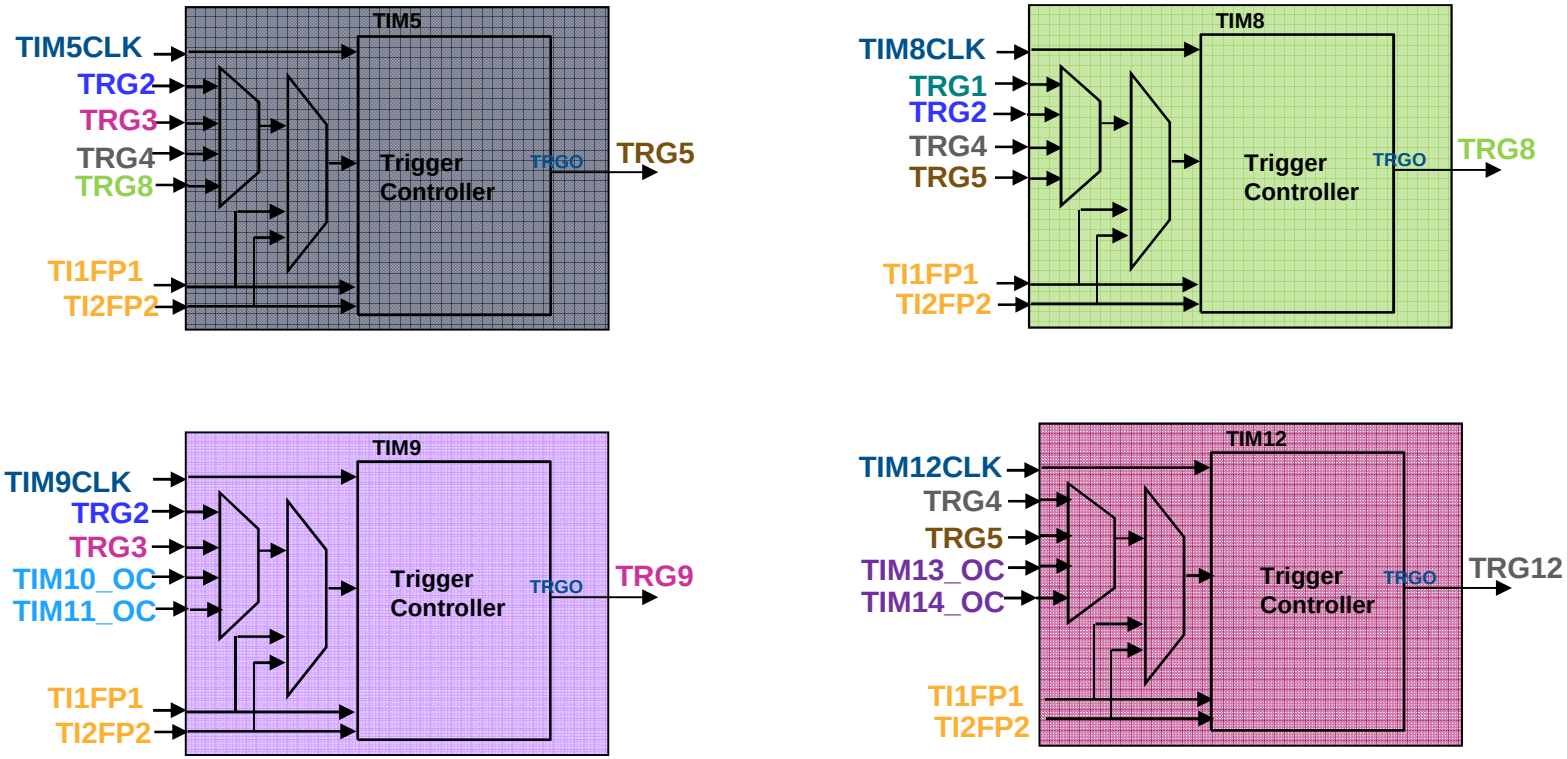
Timer Link System



TIMs Synchronization(2/2)

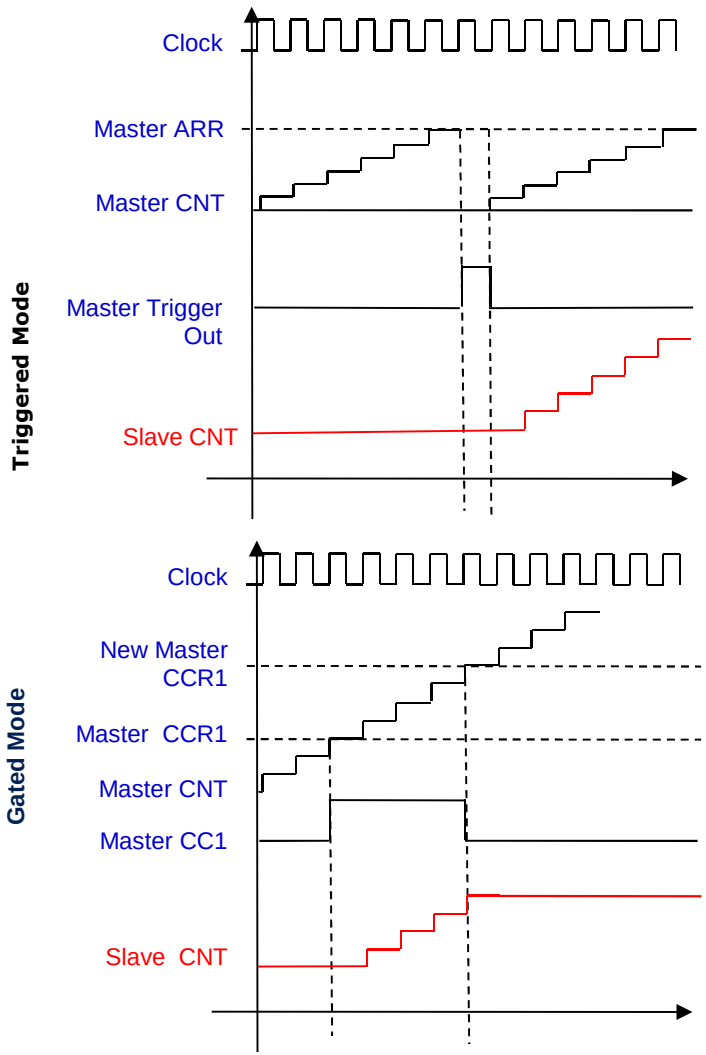
- The eight Timers are linked together for timer synchronization or chaining.

Timer Link System



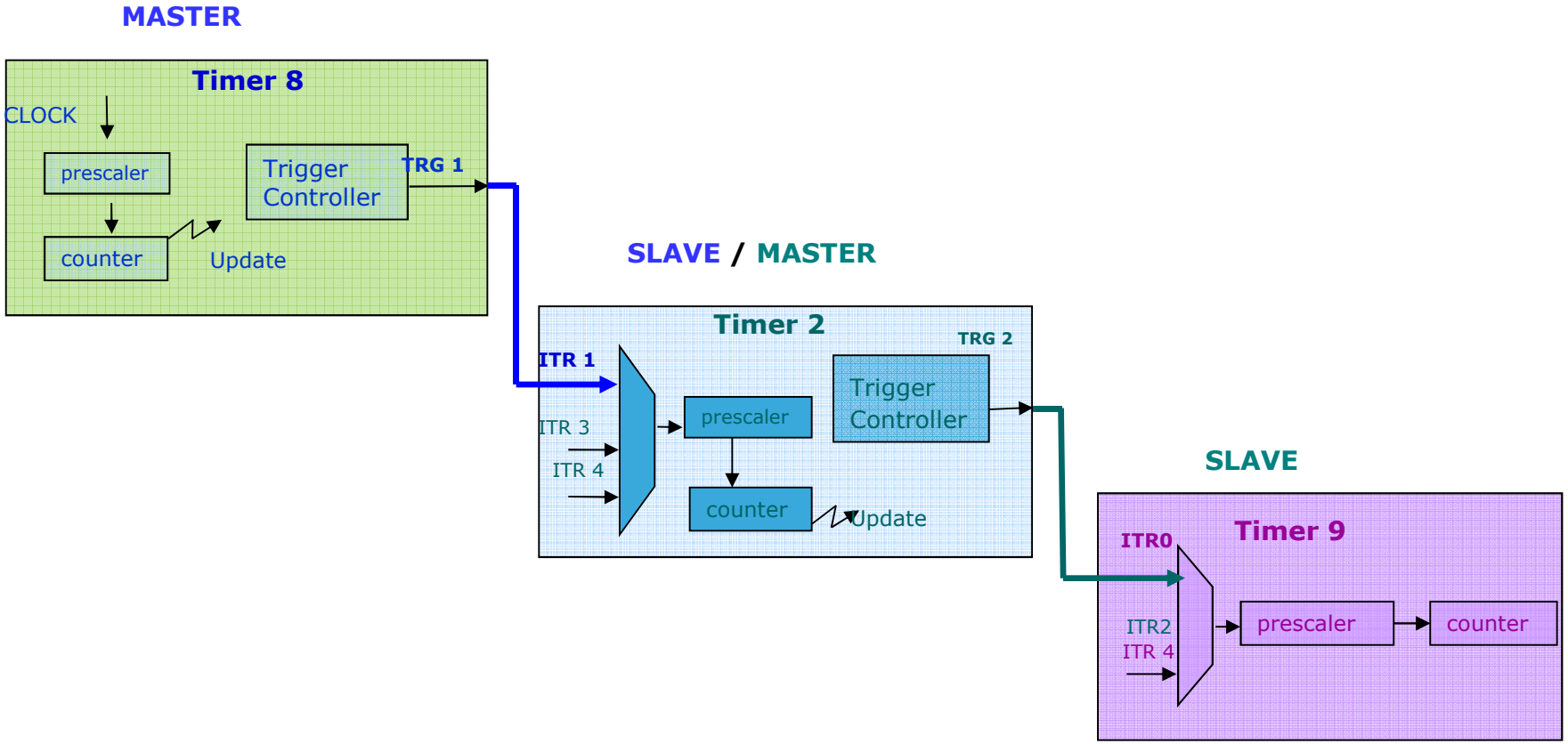
Synchronization Mode Configuration

- The Trigger Output can be controlled on:
  - Counter reset
  - Counter enable
  - Update event
  - OC1 / OC1Ref / OC2Ref / OC3Ref / OC4Ref signals
- The slave timer can be controlled in two modes:
  - Triggered mode : only the start of the counter is controlled.
  - Gated Mode: Both start and stop of the counter are controlled.



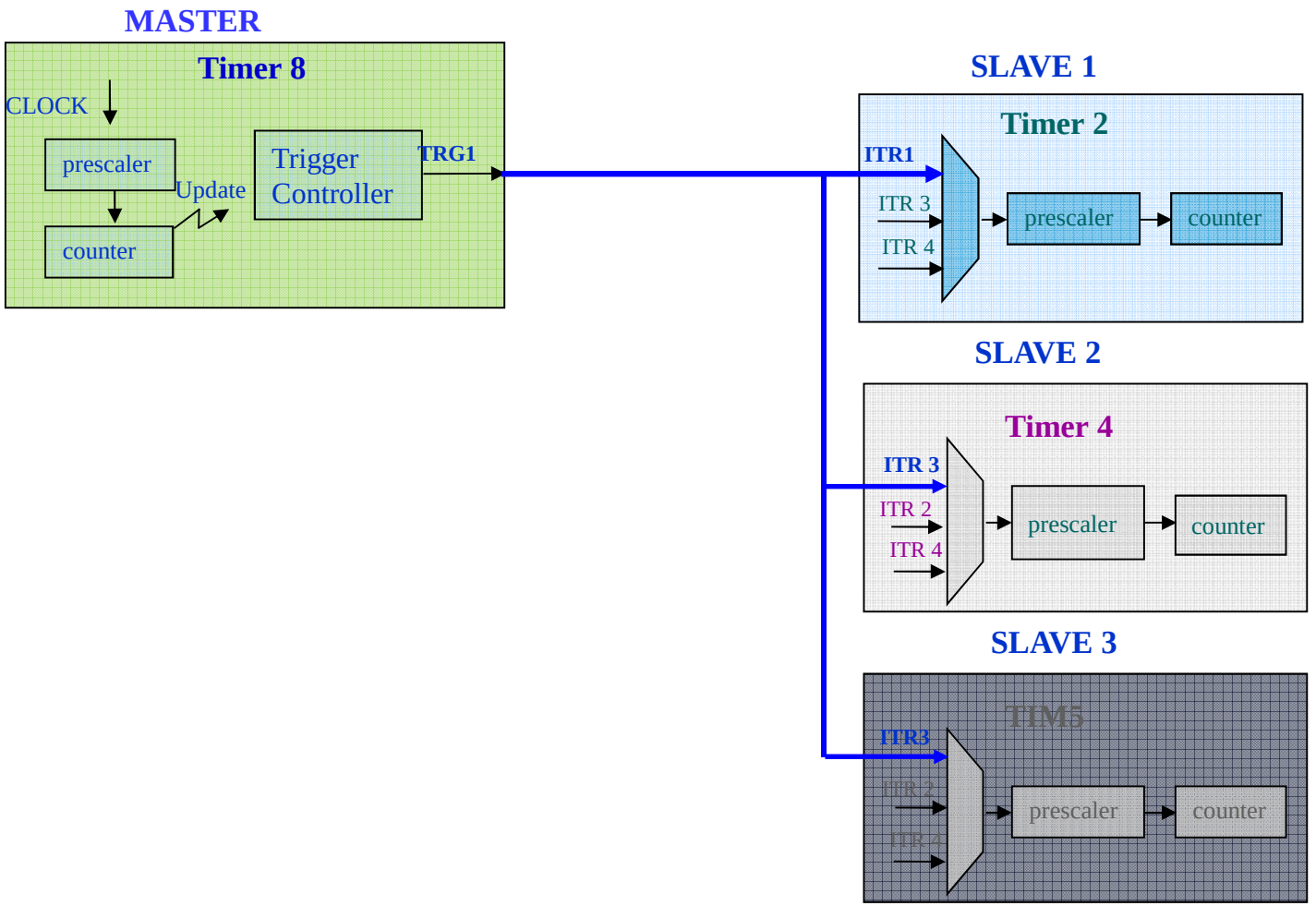
Synchronization – Configuration examples (1/3)

- Cascade mode:
  - TIM8 used as master timer for TIM2
  - TIM2 configured as TIM8 slave, and master for TIM9.



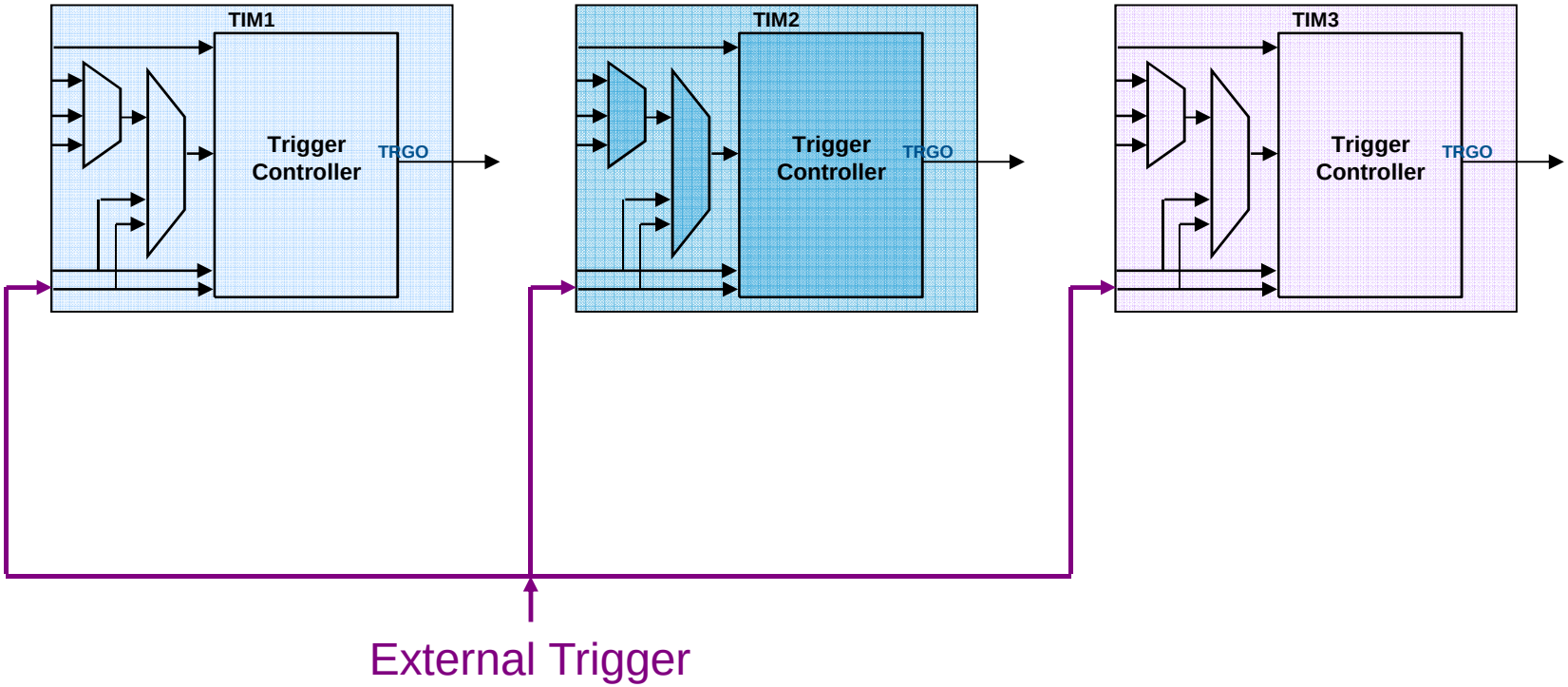
Synchronization – Configuration examples (2/3)

- One Master several slaves: TIM8 used as master for TIM2, TIM4 and TIM5.



Synchronization – Configuration examples (3/3)

- Timers and external trigger synchronization
  - TIM1, TIM2 and TIM3 are slaves for an external signal connected to respective Timers inputs

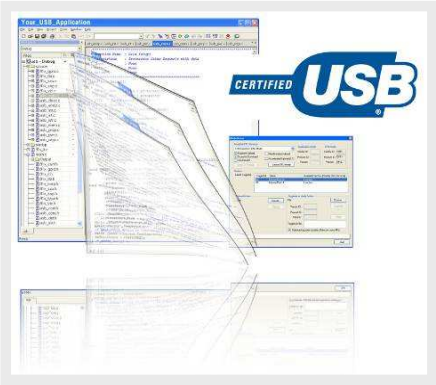


STM32 FIRMWARE RESOURCE

life.augmented

Software Libraries – Speed Time to Market

- **ST software libraries free at www.st.com/mcu**
- C source code for easy implementation of all STM32 peripherals in any application
  - **Standard library** – source code for implementation of all standard peripherals. Code implemented in demos for STM32 evaluation board
  - **USB software library** – software kit for easy implementation of any USB transfer type. Is already www.usb.org certified
  - **Motor Control library –** Sensorless Vector Control for 3-phase brushless motors



ST engineered, tested, documented and free

- **Get the most out of STM32 with an RTOS**
- Royalty-free, real-time operating systems (RTOS) for embedded applications
- Wide range of choices from leading RTOS providers
  - CMX, FreeRTOS, IAR, Keil, Micrium, Segger

life.augmented

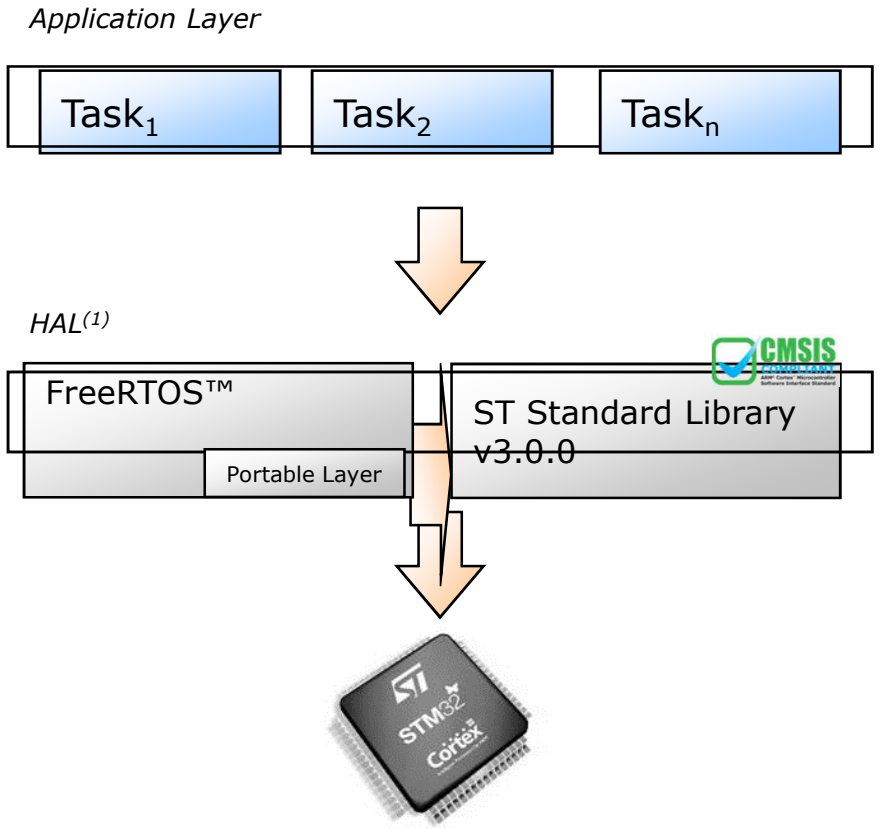
FreeRTOS™ and STM32

Key Technologies

- Real Time Operating System
  - Support both preemptive and cooperative multitasking
  - Provide Execution trace functionality and Stack overflow detection options
  - Provide Communication and Synchronization between tasks, or between tasks and interrupts
- Compliant with ARM CMSIS standard
- Supported Products: STM32 family (low, medium and high density)

Major Benefit

- The HAL(1) is ready to use.
- Modular and reusable firmware development enforced by the the tasks oriented approach.
- Reduced cost of firmware development
- Improved time to market

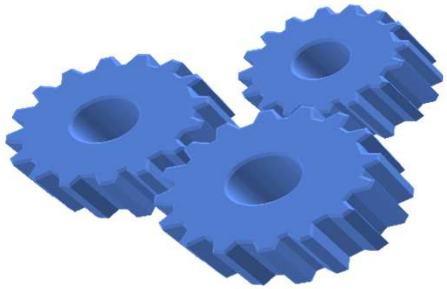


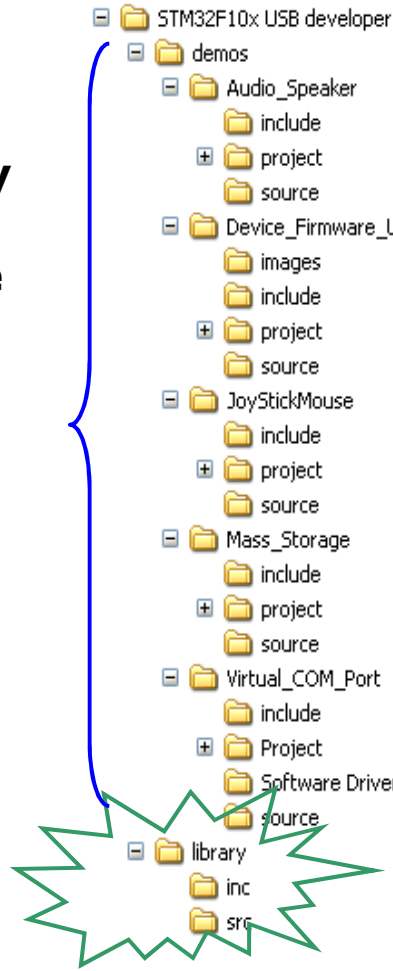
(1) HAL: Hardware Abstraction Layer

STM32F10x USB Developer kit

STM32F10x USB Library

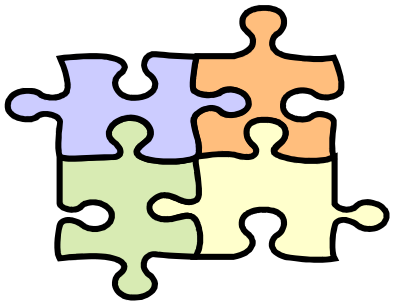
- **USB 2.0 full speed certified**
- **All in Strict ANSI-C**
- **Independent from any SW tool chain**
- **Independent from the Firmware library**
- **Self documented**
- **Supporting many USB classes:**





STM32F10x USB Developer Kit demos

- **Cover all USB transfer types**
- **Independent from any SW tool chain**
- **Running on STMicroelectronics STM32F10x-EVAL board and can be easily tailored to any other hardware**



life.augmented

Open Source TCP/IP Stacks

- uip
  - Protocols supported
    - Transmission Control Protocol (TCP)
    - User Datagram Protocol (UDP)
    - Internet Protocol (IP)
    - Internet Control Message Protocol (ICMP)
    - Address Resolution Protocol (ARP)
  - Memory requirements
    - Typical code size on the order of a few kilobytes
    - RAM usage can be as low as a few hundred bytes.
    - Memory conserved by limiting to one outstanding transmit packet
  - Website
    - http://www.sics.se/~adam/uip

- **uip and lwip licens**es
  - No restriction in shipping in real products
  - Redistribution of stack source or binaries (such as in our kit) must carry copyright

- lwip
  - Protocols supported
    - Internet Protocol (IP) including packet forwarding over multiple network interfaces
    - Internet Control Message Protocol (ICMP) for network maintenance and debugging
    - User Datagram Protocol (UDP) including experimental UDP-lite extensions
    - Transmission Control Protocol (TCP) with congestion control, RTT estimations, and fast recovery/transmit
    - Dynamic Host Configuration Protocol (DHCP)
    - Point-to-Point Protocol (PPP)
    - Address Resolution Protocol (ARP) for Ethernet
    - Specialized raw API for enhanced performance
    - Optional Berkeley-like socket API
  - Memory Requirements
    - Typical code size is on the order of 25 to 40 kilobytes
    - RAM requirements are approximately 15 to a few tens of kilobytes
  - Website
    - http://www.sics.se/~adam/lwip
    - http://savannah.nongnu.org/projects/lwip

Slide 132

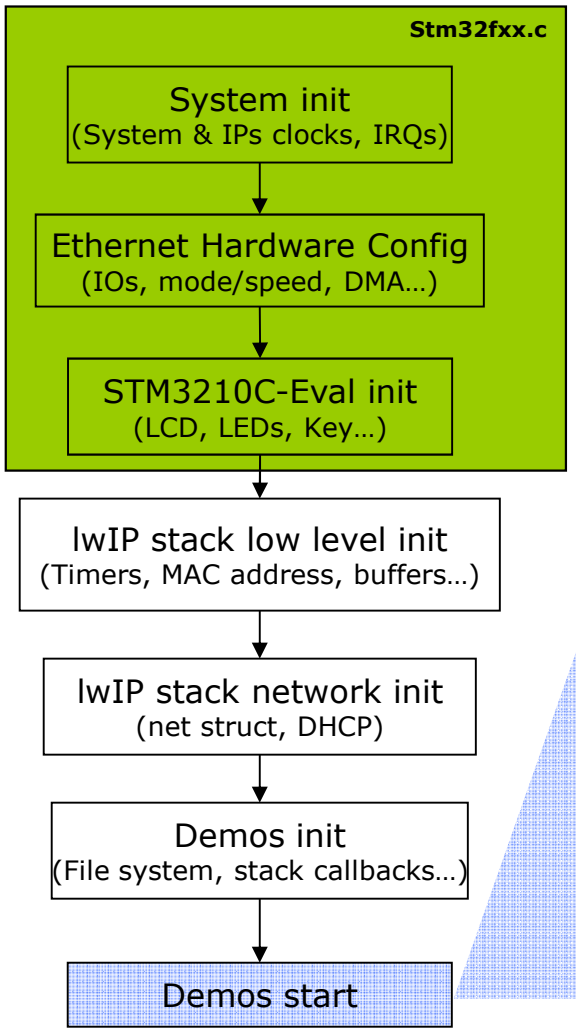
MSOffice3 should be "Limted to one outstanding transmit packet"
, 8/14/2007

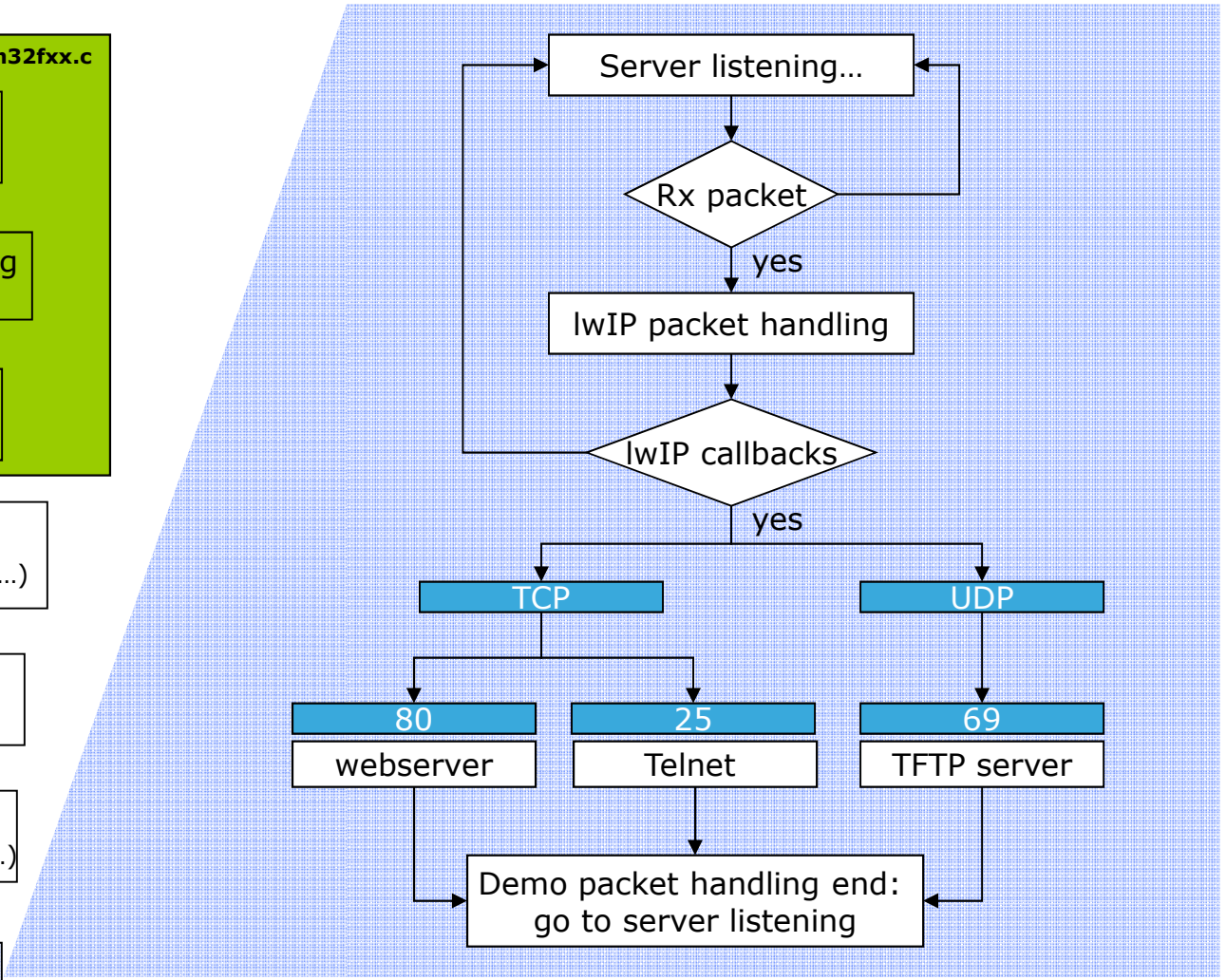
MSOffice4 Throughput with lwIP is more like 2 to 3 Mbps. uIP provides a raw throughtput of ~30Mbps for "ping" packets, but is much lower for http traffic. lwIP has a lower ping throughput, but a higher http throughput than uIP due to better buffer management and allowing multiple outstanding packets.
, 8/14/2007

Ethernet software Open Source solutions

- lwIP and uIP are both ported on STM32 connectivity line.
- GCC-Eclipse, IAR IDE and CrossStudio IDE are supported
- WEB server demo based on lwIP (v 1.3) is now available

LwIP Demonstration

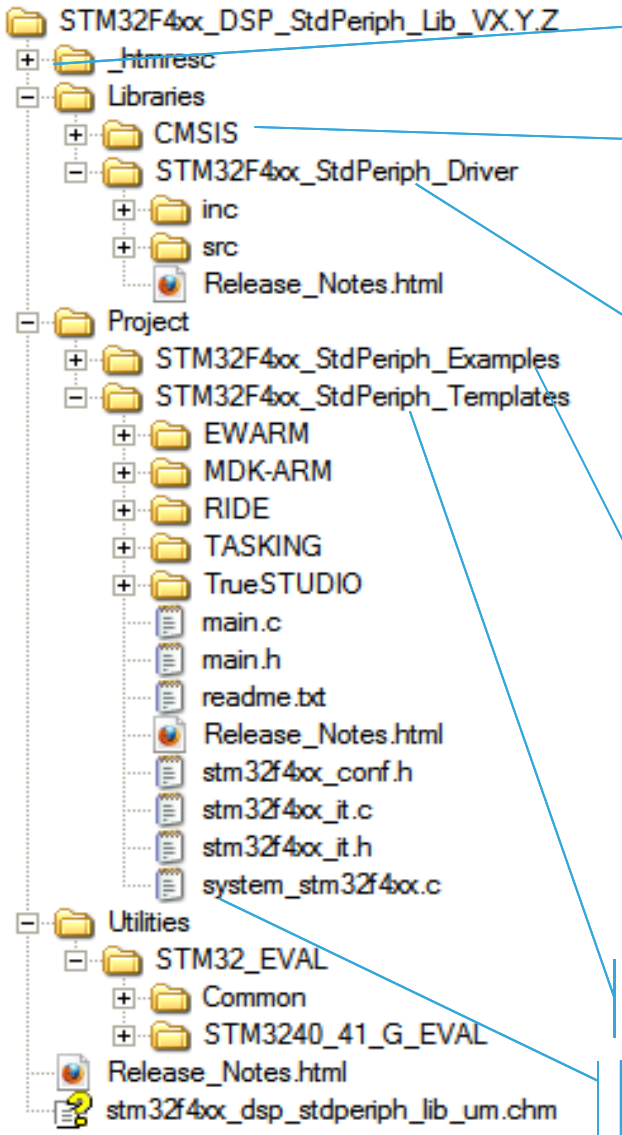




life.augmented

STM32 Standard Peripherals Firmware Library

Package organization



This Folder contains all package html page resources

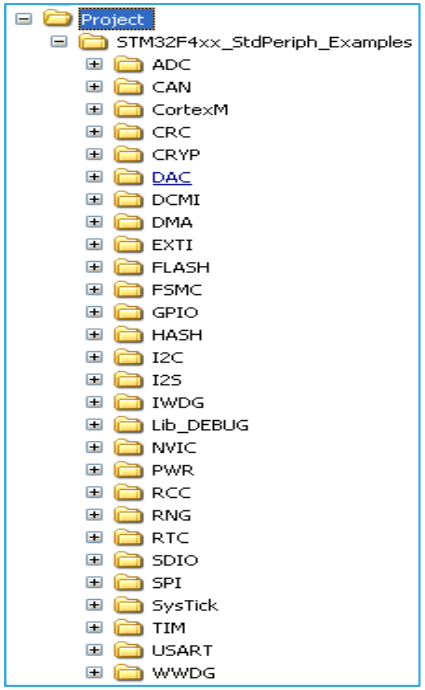
This folder contains the Cortex-M4F CMSIS files:
Core Peripheral Access Layer: contains name definitions, address definitions and helper functions to access Cortex-M4F core registers and peripherals…
CMSIS DSP Software Library: features a suite of common signal processing functions for use on Cortex-M processor based devices. The library is completely written in C.

This folder contains all the subdirectories and files that make up the core of the library:
inc sub-folder contains the Peripheral's Drivers header files
misc.h: all the functions prototypes for the miscellaneous firmware library functions.
stm32f4xx_ppp.h (one header file per peripheral): Function prototypes, data structures…
src sub-folder contains the Peripheral's Drivers source files.
misc.c: all the miscellaneous firmware functions (add-on to CMSIS functions)..
stm32f4xx_ppp.c (one source file per peripheral): Function bodies of each peripheral.

readme.txt: brief text file describing the example and how to make it work.
stm32f4xx_conf.h: header file allowing to enable/disable the peripheral's drivers header files inclusion.
stm32f4xx_it.c: source file containing the interrupt handlers (the function bodies may be emptied if not used).
stm32f4xx_it.h: header file including all interrupt handler prototypes.
main.c: example of code.
system_stm32f4xx.c: this file provide functions to setup the STM32 system: Configures the System clock frequency, AHB/APBx prescalers and Flash settings.

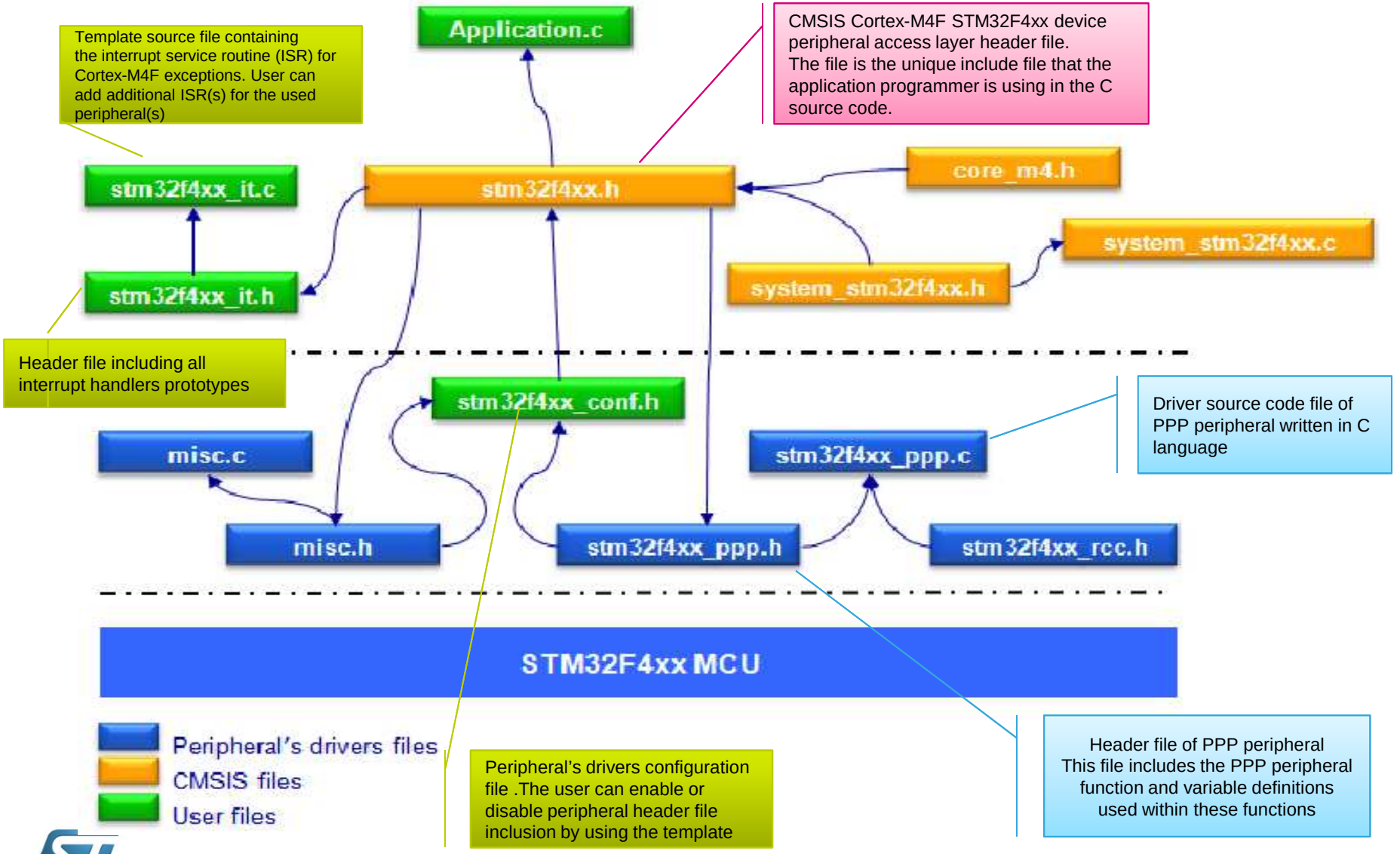
This folder contains standard template projects for EWARM, MDK-ARM, RIDE, TASKING and TrueSTUDIO

STM32_EVAL Implement an abstraction layer to interact with the Human Interface resources; buttons, LEDs, LCD, (USARTs)…



life.augmented

STM32F4x standard peripheral library architecture



life.augmented

Coding conventions

- All firmware is coded in ANSI-C
  - Strict ANSI-C for all library peripheral files
    - Relaxed ANSI-C for projects & Examples files.
- *PPP* is used to reference any peripheral acronym, e.g. *TIM* for Timer.
- Registers & Structures
  - STM32F10x registers are mapped in the microcontroller address space
    - FW library registers have the same names as in STM32F10x Datasheet & reference manual.
  - All registers hardware accesses are performed through a C structures :
    - Work with only one base address and indirect addressing
    - Improve code re-use : e.g. the same structure to handle and initialize 3 USARTs.

Using the Library (1/2)

- In the main file , you have to declare a *PPP_InitTypeDef* structure, e.g: PPP_InitTypeDef PPP_InitStructure;
  - The **PPP_InitStructure** is a working variable located in data memory that allows you to initialize one or more instance of PPPs.
- You have to fill the ***PPP_InitStructure*** variable with the allowed values of the structure member.
  - Configuration of the whole structure:
    - PPP_InitStructure.member1 = val1;
    - PPP_InitStructure.member2 = val2;
    - ...
    - PPP_InitStructure.memberN = valN;

Note : The previous initialization could be merged in only one line like the following :

- PPP_InitTypeDef PPP_InitStructure = { val1, val2, …, valn}

  This reduces and optimises code size.

- Configuration of few structure's members:
  - PPP_StructInit(&PPP_InitStructure);
  - PPP_InitStructure.memberX = valX;
  - PPP_InitStructure.memberY = valY;

life.augmented

Using the Library (2/2)

- You have to initialize the PPP peripheral by calling the *PPP_Init(..)* function :
  - PPP_Init(PPPx, &PPP_InitStructure);
- At this stage the PPP peripheral is initialized and can be enabled by making a call to *PPP_Cmd(..)* function.
  - PPP_Cmd(PPPx, ENABLE);
- To access the functionality of the PPP peripheral, the user can use a set of dedicated functions. These functions are specific to the peripheral and for more details refer to STM32F10x Firmware Library User Manual.

Notes :

1) Before configuring a peripheral, you have to enable its clock by calling one of the following functions:

- RCC_AHBPeriphClockCmd(RCC_AHBPeriph_PPPx , ENABLE);
- RCC_APB2PeriphClockCmd(RCC_APB2Periph_PPPx , ENABLE);
- RCC_APB1PeriphClockCmd(RCC_APB1Periph_PPPx , ENABLE);

2) PPP_DeInit(..) function can be used to set all PPP's peripheral registers to their reset values:

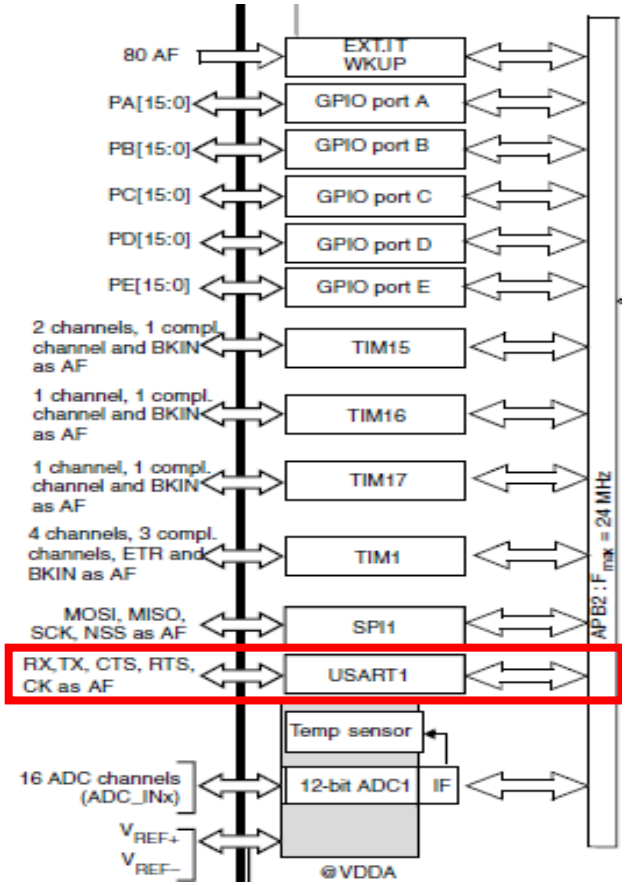
- PPP_DeInit(PPPx);

3) If after peripheral configuration, the user wants to modify one or more peripheral settings he should proceed as following:

- PPP_InitStucture.memberX = valX;
- PPP_InitStructure.memberY = valY;
- PPP_Init(PPPx, &PPP_InitStructure);

UART1 configuration example


```
/* Enable USART1 Clock */
RCC_APB2PeriphClockCmd(RCC_APB2Periph_USART1, ENABLE );
/* set all UART1's peripheral registers to their reset values */
USART_DeInit( USART1 ) ;
/* USART1 configured as follow:
      - BaudRate = 19200 baud
      - Word Length = 8 Bits
      - One Stop Bit
      - Even parity
      - Hardware flow control disabled (RTS and CTS signals)
      - Receive and transmit enabled */
USART_InitStructure.USART_BaudRate = 19200;
USART_InitStructure.USART_WordLength = USART_WordLength_8b;
USART_InitStructure.USART_StopBits = USART_StopBits_1;
USART_InitStructure.USART_Parity = USART_Parity_Even;
USART_InitStructure.USART_HardwareFlowControl = USART_HardwareFlowControl_None;
USART_InitStructure.USART_Mode = USART_Mode_Rx | USART_Mode_Tx;
 /* Configure USART1 */
USART_Init( USART1, &USART_InitStructure);
/* Enable USART1 */
USART_Cmd( USART1, ENABLE );
```



USART 1 is ready now …

STM32 USB Library

life.augmented

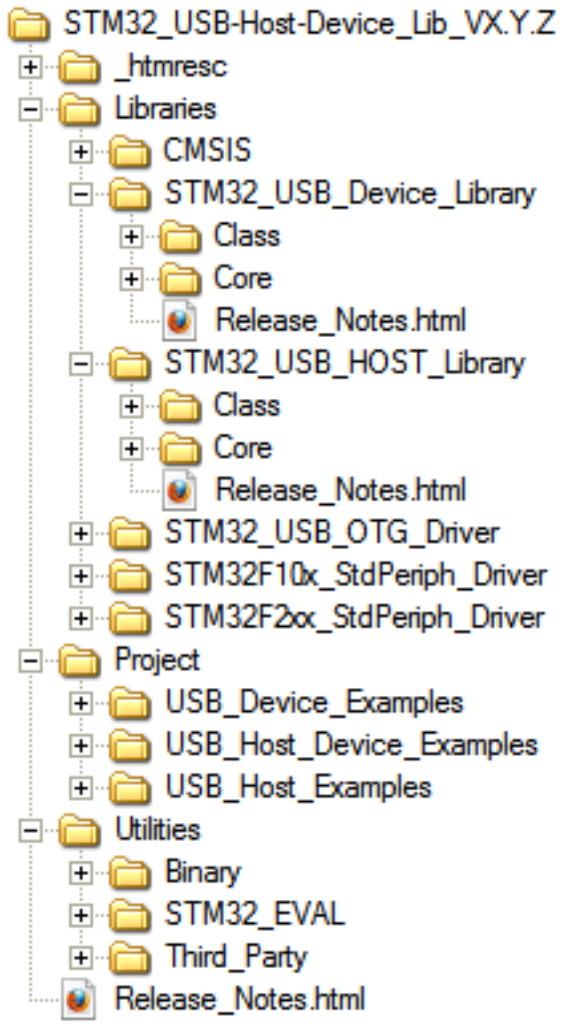
STM32 USB Host and Device Library

- Compatible with the *STM32F105x, STM32F107x and STM32F2xx devices in HS and FS USB modes*
- Fully compatible USB library with “the *Universal Serial Bus Specification, revision 2.0” spec.*
- Optimized to work with the *USB OTG IPs (high speed and full speed).*
- Uses all the features offered by the *USB OTG IPs (high speed and full speed)..*
- Built with a reduced footprint, high transfer performance, robustness and a high code and documentation package quality.
- Can be easily extended to support the OTG feature.
- Built following a generic and easy to use architecture allowing adding more specific vendor classes and supporting multi interface application (composite devices).
- Supports multi USB OTG cores allowing using several cores with the same library.

life.augmented

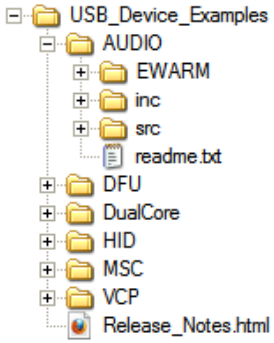
USB Host and Device Library folder structure

- composed of three main directories and organized as follows :
  - ***Libraries:*** contains the STM32 USB OTG low level driver, the standard peripherals libraries, the Host and the Device libraries.
  - ***Project:*** contains the workspaces and the sources files for the examples given with the package.
  - ***Utilities:*** contains the stm32 drivers relative to the used boards (LCD, SD card, buttons, joystick..Etc). This folder contains also the free file system FatFS used for the Host demos.



STM32 USB host device Library (1/2)

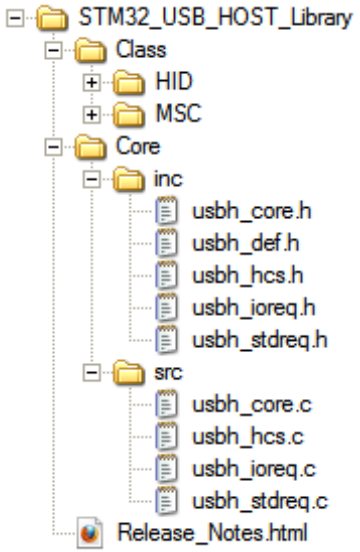
- Including the following examples for device :
  - **Mass storage demonstration** based on the *microSD card available on the EVAL boards.*
  - **HID Joystick demonstration** based on the embedded joystick on the EVAL boards.
  - **Virtual com demonstration**.
  - **DFU based application**.
  - **Audio (OUT) demonstration**.
  - **Dual Core devices demonstration** based on Mass storage and Human interface device examples (available only for the STM322xG-EVAL)



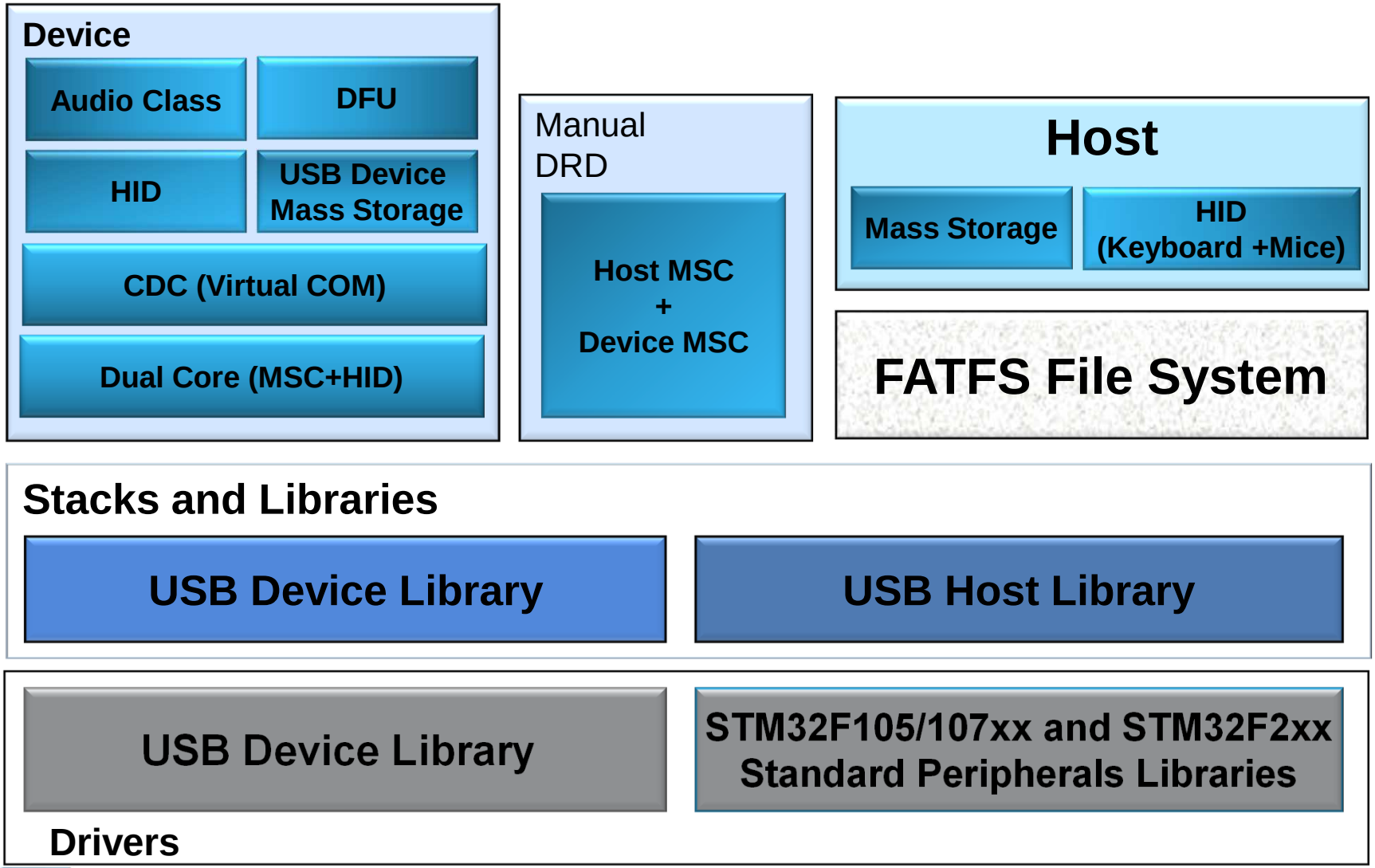
life.augmented

STM32 USB host device Library (2/2)

- And the following examples for Host:
  - Mass storage demonstration with file explorer, write files and slide show.
  - HID demonstration supporting dynamically mice and keyboards.
  - Dual Core devices demonstration based on Mass storage the High speed port and Human interface device (keyboards or mice) on the full speed port.



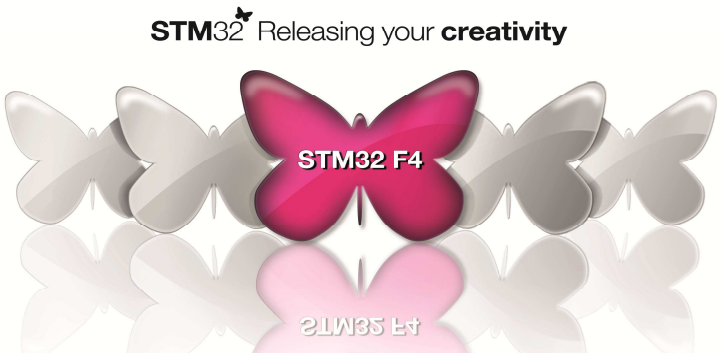
STM32 USB Host and Device Library



life.augmented

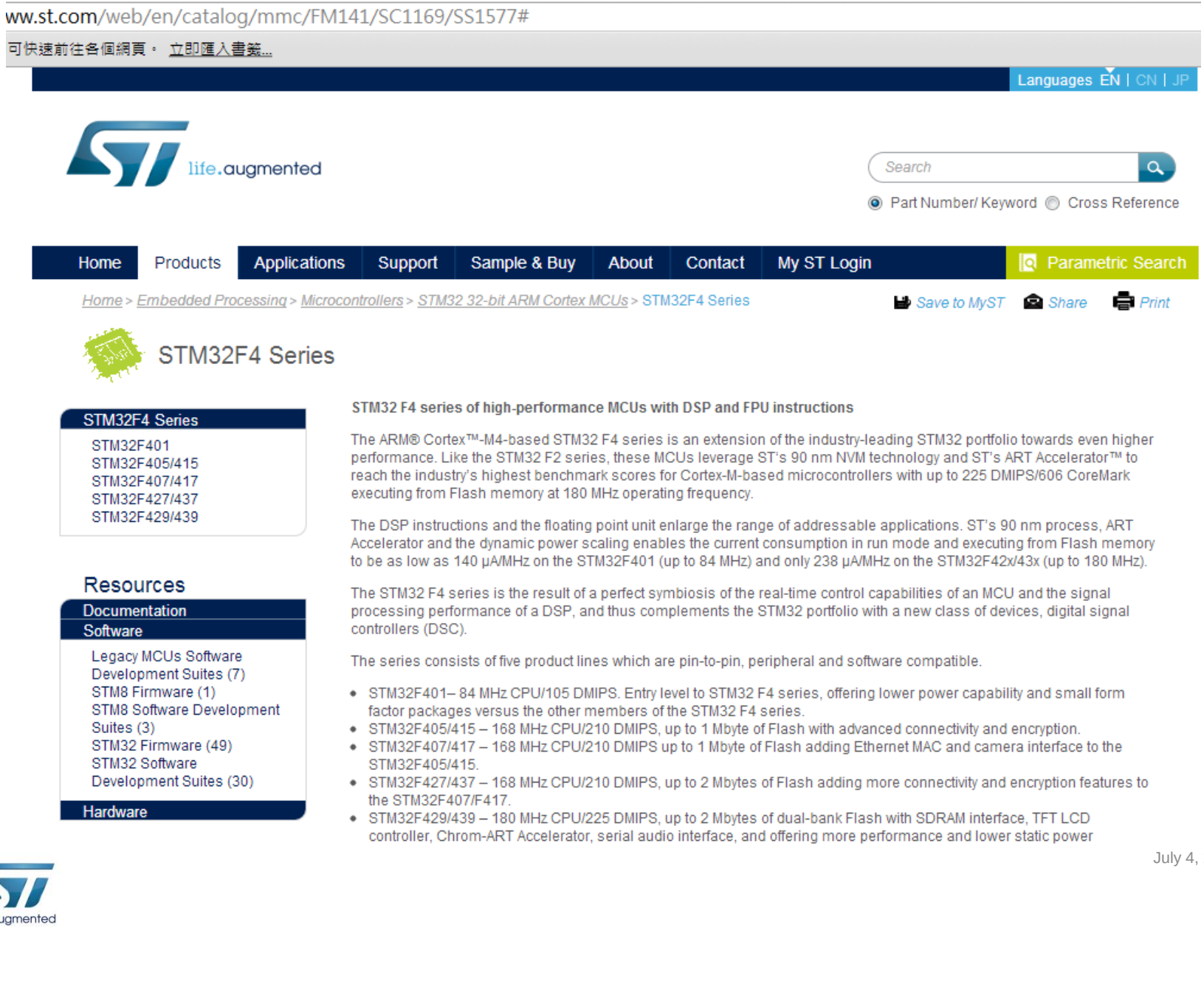
Document and firmware Library available on internet Web.

www.st.com/stm32f4



life.augmented

www.st.com/stm32f4



life.augmented

www.st.com/stm32f4

STM32F4 Datasheet :
http://www.st.com/st-web-ui/static/active/en/resource/technical/document/datasheet/DM00035129.pdf

STM32F4xx Reference Manual
http://www.st.com/web/en/resource/technical/document/reference_manual/DM00031020.pdf

STM32F4 Cortex-M4 Programming Manual :
http://www.st.com/web/en/resource/technical/document/programming_manual/DM00046982.pdf

STM32F4 Flash Programming Manual :
http://www.st.com/web/en/resource/technical/document/programming_manual/DM00023388.pdf

STM32F4x Clock configuration tool :
Excel tool : http://www.st.com/st-web-ui/static/active/en/st_prod_software_internet/resource/technical/software/utility/stsw-stm32091.zip
User Manual : http://www.st.com/web/en/resource/technical/document/application_note/DM00039457.pdf

life.augmented

www.st.com/stm32f4

- STM32F4x Standard Peripherals Library :
  http://www.st.com/st-web-ui/static/active/en/st_prod_software_internet/resource/technical/software/firmware/stm32f4_dsp_stdperiph_lib.zip

- LwIP TCP/IP stack demonstration for STM32F4x7 microcontrollers :
  Firmware Package : http://www.st.com/st-web-ui/static/active/en/st_prod_software_internet/resource/technical/software/firmware/stsw-stm32070.zip
  User Manual : http://www.st.com/internet/com/TECHNICAL_RESOURCES/TECHNICAL_LITERATURE/APPLICATION_NOTE/DM00026013.pdf

- *STM32F4xx USB On-The-Go host and device library :*
  http://www.st.com/st-web-ui/static/active/en/st_prod_software_internet/resource/technical/software/firmware/stm32_f105-07_f2_f4_usb-host-device_lib.zip
  User Manual : http://www.st.com/st-web-ui/static/active/en/resource/technical/document/user_manual/CD00289278.pdf

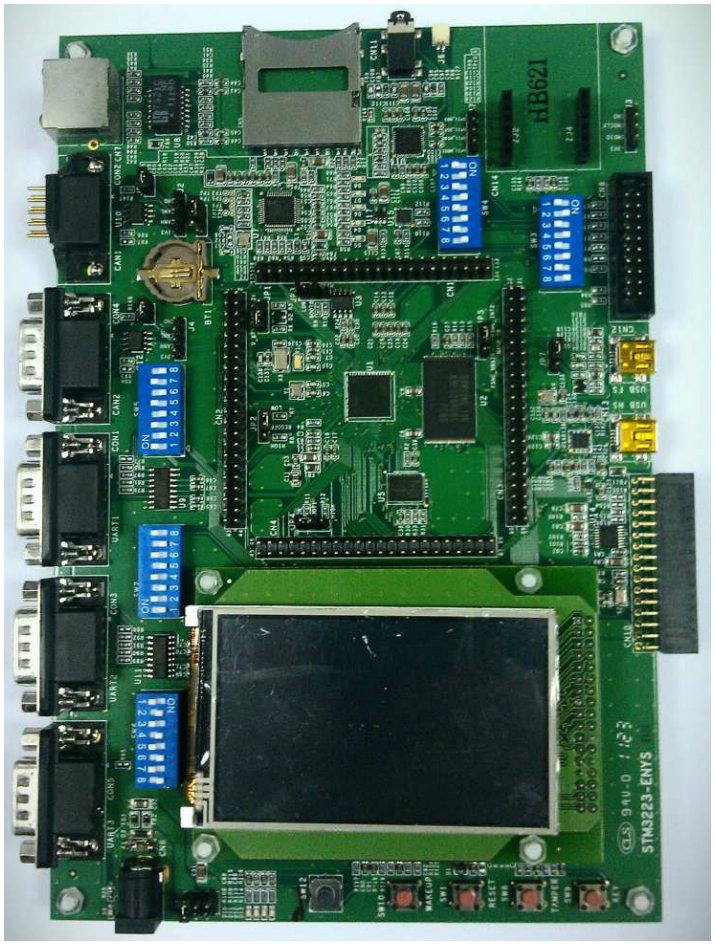
- ST-Link Utility :
  http://www.st.com/web/catalog/tools/FM146/CL1984/SC720/SS1450/PF251168?s_searchtype=partnumber

STM32F4 EVALUATION BOARD

life.augmented

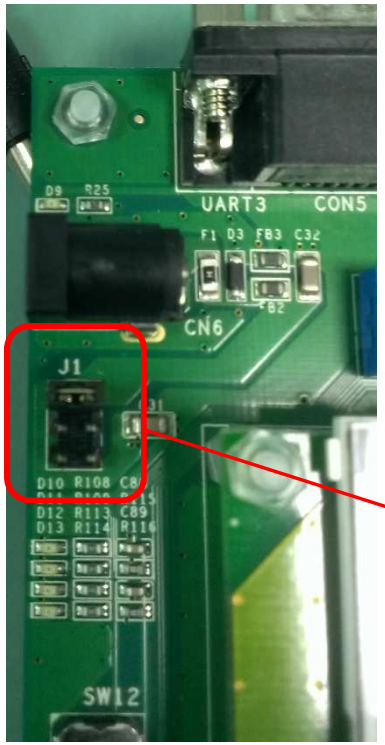
STM32F4 Evaluation Board.

- STM32F407IG (Cortex™-M4 CPU with FPU, ART Accelerator™, Embedded 1024KBytes Flash, 192+4 Kbytes RAM).
- On board external 4Mbits PSRAM, 1GbitsNAND through FSMC.
- Option for supply power from USB or external 5V power supply..
- USB OTG HS and FS with Micro-AB connector.
- I2S audio DAC, stereo audio jack.
- IEEE-802.3.2002 compliant Ethernet.
- 3.2” 240x400 TFT Color LCD with touch screen.
- Joystick with 4 direction control and selector.
- Camera Interface.
- RTC with backup battery.
- SD Card interface.
- 4 x color LEDs.
- Easy evaluate STM32 peripheral on evaluation board.



life.augmented

Power supply config



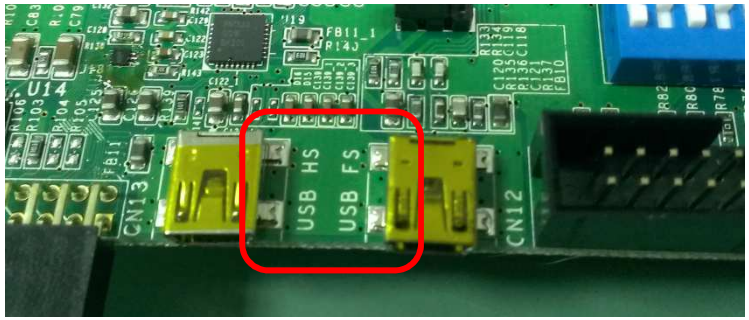
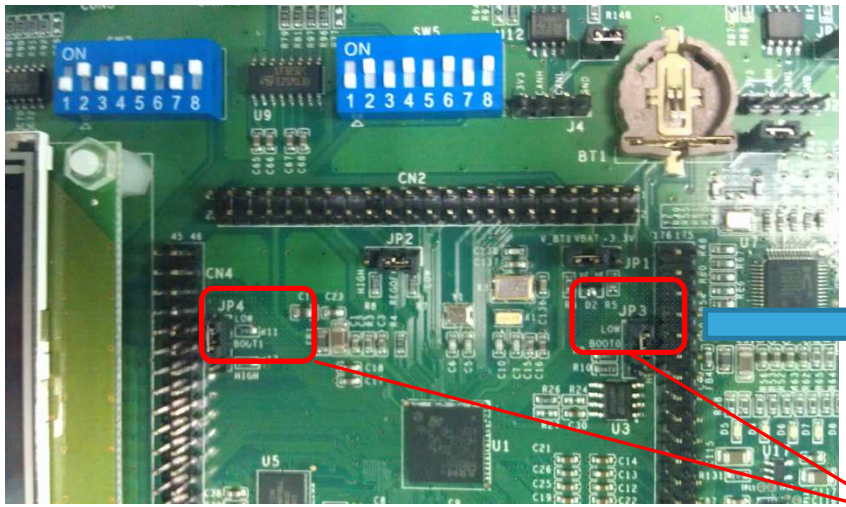


Table 1 **Power related Jumpers**

JUMPER	Description
JP1	JP1 is used to select one of the four possible power supply resources. For **power supply jack**(CN6) to the STM32F2/4-EVB only, JP1 is set as shown to the right: (Default Setting) J1 **PSU** FS HS
	For power supply from USB OTG FS (CN12) to STM32F2/4-EVB only, JP1 is set as shown to the right: J1 PSU **FS** HS
	For power supply from USB OTG HS (CN13) to STM32F2/4-EVB only, JP1 is set as shown to the right: J1 PSU FS **HS**

life.augmented

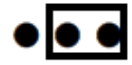
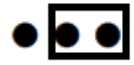
Boot configuration



Low

High

Table 2 Boot related Switch

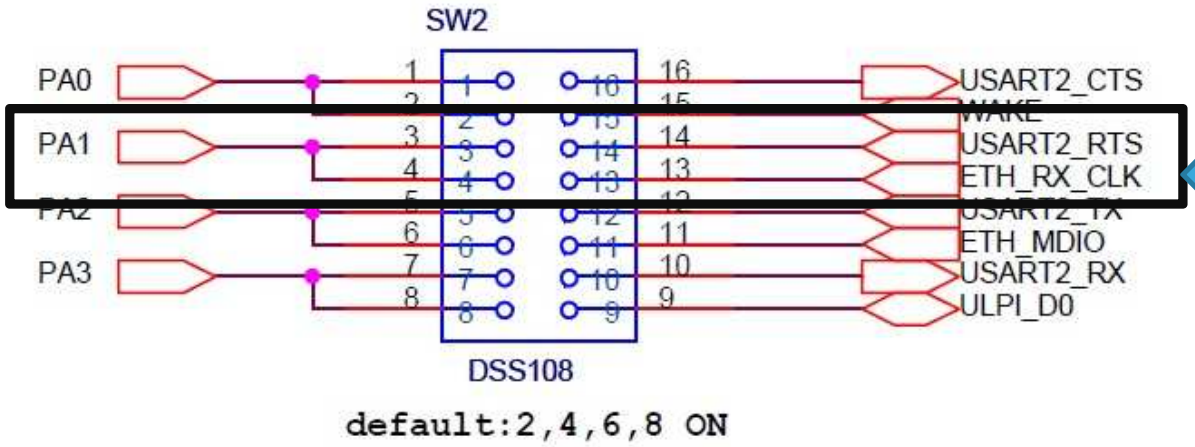
BOOT0	BOOT1	Boot from	JP3 BOOT0	JP4 BOOT1
0	X	STM32F2/4-EVB boot from **User Flash** when BOOT0 is set as shown to the right. BOOT1 is don't care in this configuration. (Default setting)		
1	0	STM32F2/4-EVB boot from **System Memory** when BOOT0 and BOOT1 are set as shown to the right.		
1	1	STM32F2/4-EVB boot from **Embedded SRAM** when BOOT0 and BOOT1 are set as shown to the right.		

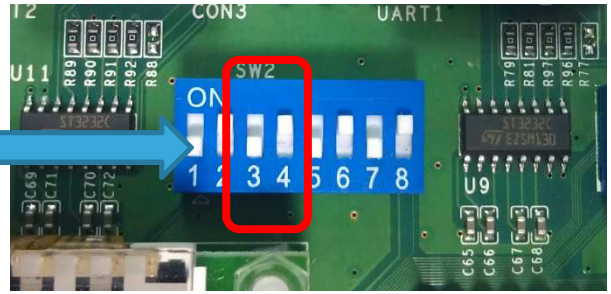
life.augmented

I/O Configuration

Table 7. STM32F41x pin and ball definitions (continued)

Pin number						Pin name (function after reset)[1]	Pin type	I / O structure	Notes	Alternate functions	Additional functions
LQFP64	WLCSP90	LQFP100	LQFP144	UFBGA176	LQFP176						
15	F8	24	35	N2	41	PA1	I/O	FT	(4)	USART2_RTS / UART4_RX/ ETH_RMII_REF_CLK / ETH_MII_RX_CLK / TIM5_CH2 / TIM2_CH2/ EVENTOUT	ADC123_IN1





LAB Getting Start & ST-Link Utility

LAB – Getting Started

Development tool Installation

- Install RVMDK.
- Active RVMDK license.

ST-Link Utility install

- Install ST-Link Utility.
- Connect to STM32F4 evaluation board.

life.augmented

LAB : ST-Link Utility Installation

- step1:

- Unzip STM32 ST-LINK Utility_V3.0.zip

- step2:

- Run the stm32_st-link_utility.exe in unzip folder to install ST-Link utility.

- step3:

- Connect to STM32F4-Discovery board.

- step4:

- Execute the STM32 ST-LINK Utility

- step5:

- =>ST-Link=> Firmware update. (Upgrade ST-Klink firmware if need)

- step6:

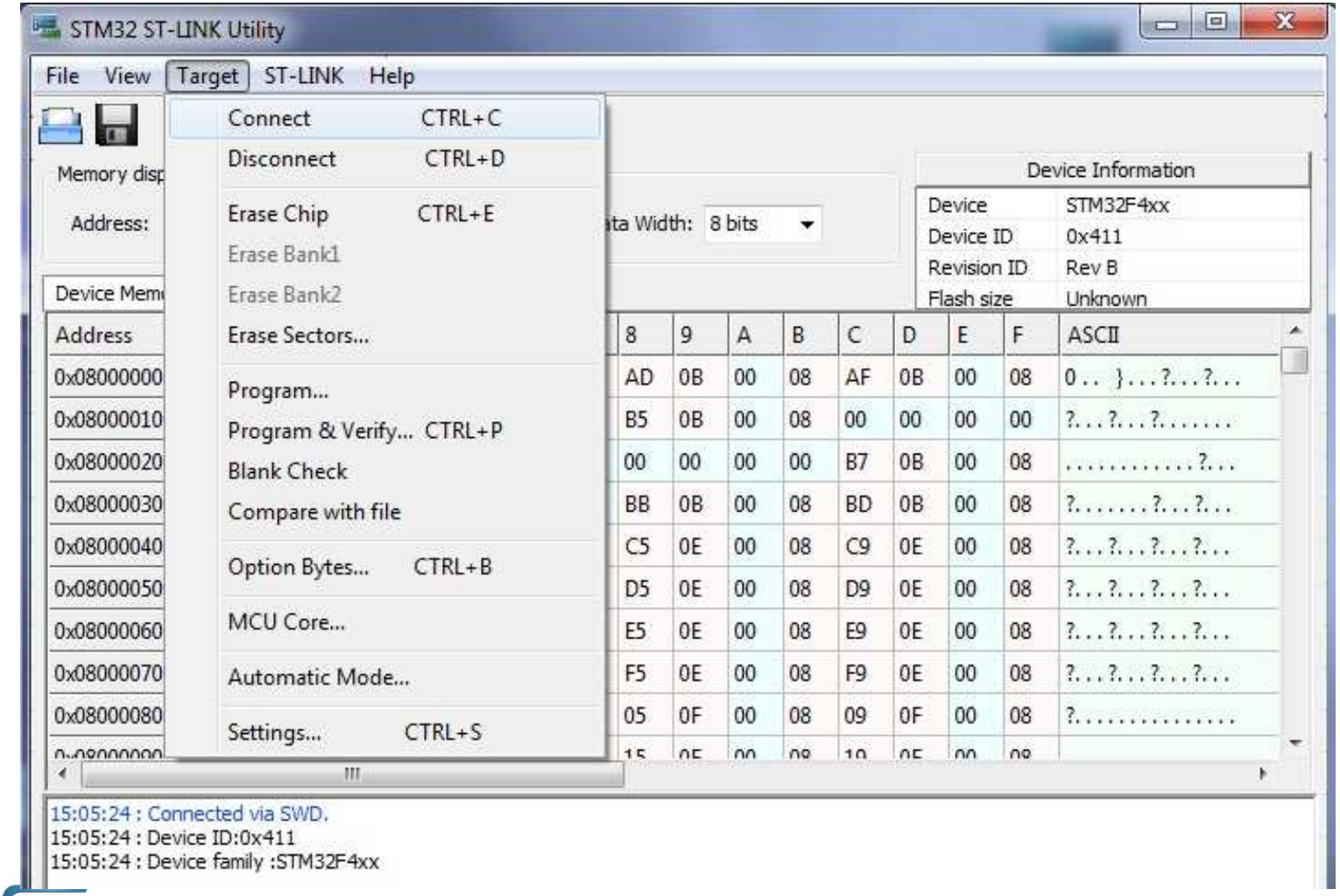
- Option to use SWD =>Target=>settings=>Connection Protocol => SWD.

- step7:

- =>Target =>Connect.

ST Link Utility

Target => Connect=>



life.augmented

Install ST-Link driver in Windows 8

- step1:

Press "Windows-I" to load the Charms Bar

- step2:

Click on "Power", hold down the "Shift-key", then click on Restart to restart the PC. (Keep hold the "Shift-Key").

- step3:

And then you should see the troubleshooting page. Click on "Troubleshoot" , and the next page on "Advanced options".

- step4:

Click on the "later to change the startup behavior of the Windows 8 operating system" and click on Restart again on the next page

- step5:

After reboot system, you should see an advanced menu with nine different startup options. Press 7 or F7 to "disable the driver signature enforcement". Windows 8 should restart the system and the drivers that you have installed should be working from that moment on.

LAB Demo Project

life.augmented

LAB : Demo Project

- step1:

- Unzip "ARM_Design_Contest_STM32F4_Eval_V1.0.rar".

- step2:

- Open RVMDK project from : "ARM_Design_Contest_STM32F4_Eval_V1.0\STM32F4_EVB\Project\STM32F4_EVB_Demo\MDK-ARM\Project.uvproj "

- step3:

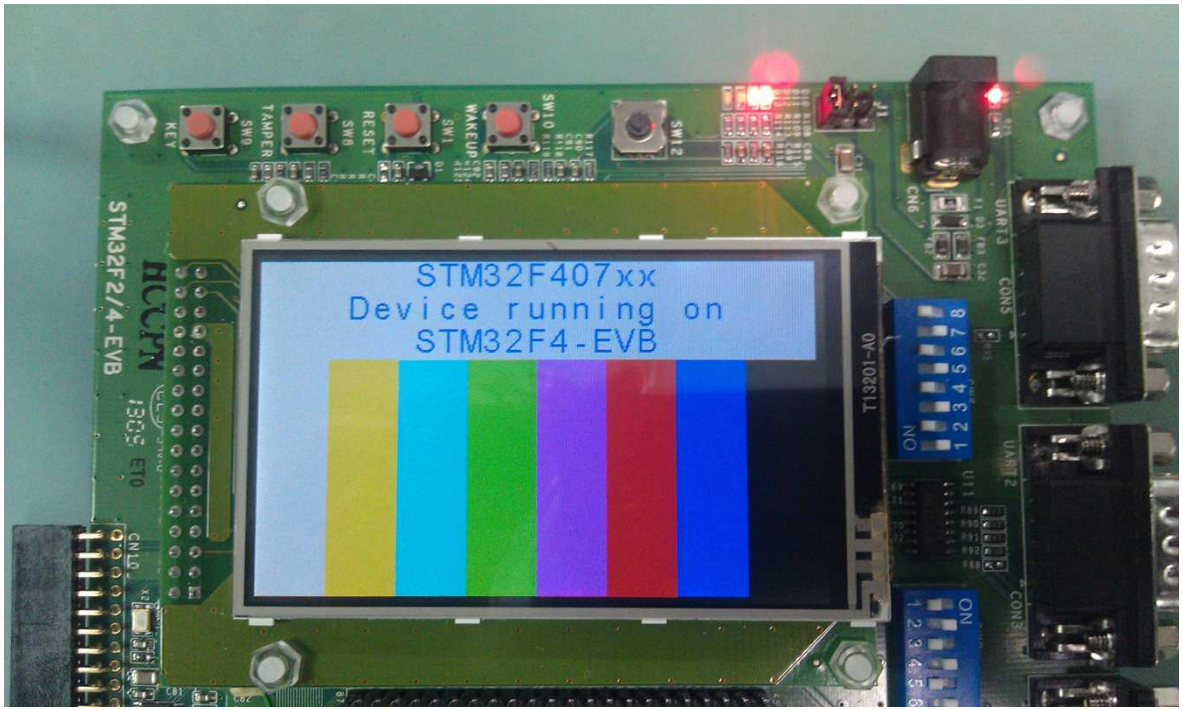
- Build Target.

- step4:

- Download to flash and entering debug session.

LAB : Demo Project

- **You should be able to see LED blinking and LCD display as below :**



life.augmented

LAB Board Test Project

life.augmented

LAB : Board Test Project

- step1:

- Unzip "ARM_Design_Contest_STM32F4_Eval_V1.0.rar".

- step2:

- Open RVMDK project from : "ARM_Design_Contest_STM32F4_Eval_V1.0\STM32F4_EVB\Project\ STM32F4_EVB_BoardTest\MDK-ARM\BoardTest.uvproj "

- step3:

- Build Target.

- step4:

- Download to flash and entering debug session.

LAB : Board Test Project

- LED (D10) toggling.
- Button press with LED (D12) toggling.
- Ethernet - LwIP.
- USB FS Virtual com port.
- Touch Screen - I2C.
- LCD Display.

life.augmented

Thank You !


STM32
STM32 Releasing your creativity


STM32


STM8 S


life.augmented


life.augmented