STM32F3 Technical Training

2013, CompelFest

Alexander Kvashin

Roman Popov

life.augmented

COMPELFEST

ARM Cortex M4 in few words

Cortex-M processors

- **Forget traditional 8/16/32-bit classifications**
  - Seamless architecture across all applications
  - Every product optimised for ultra low power and ease of use

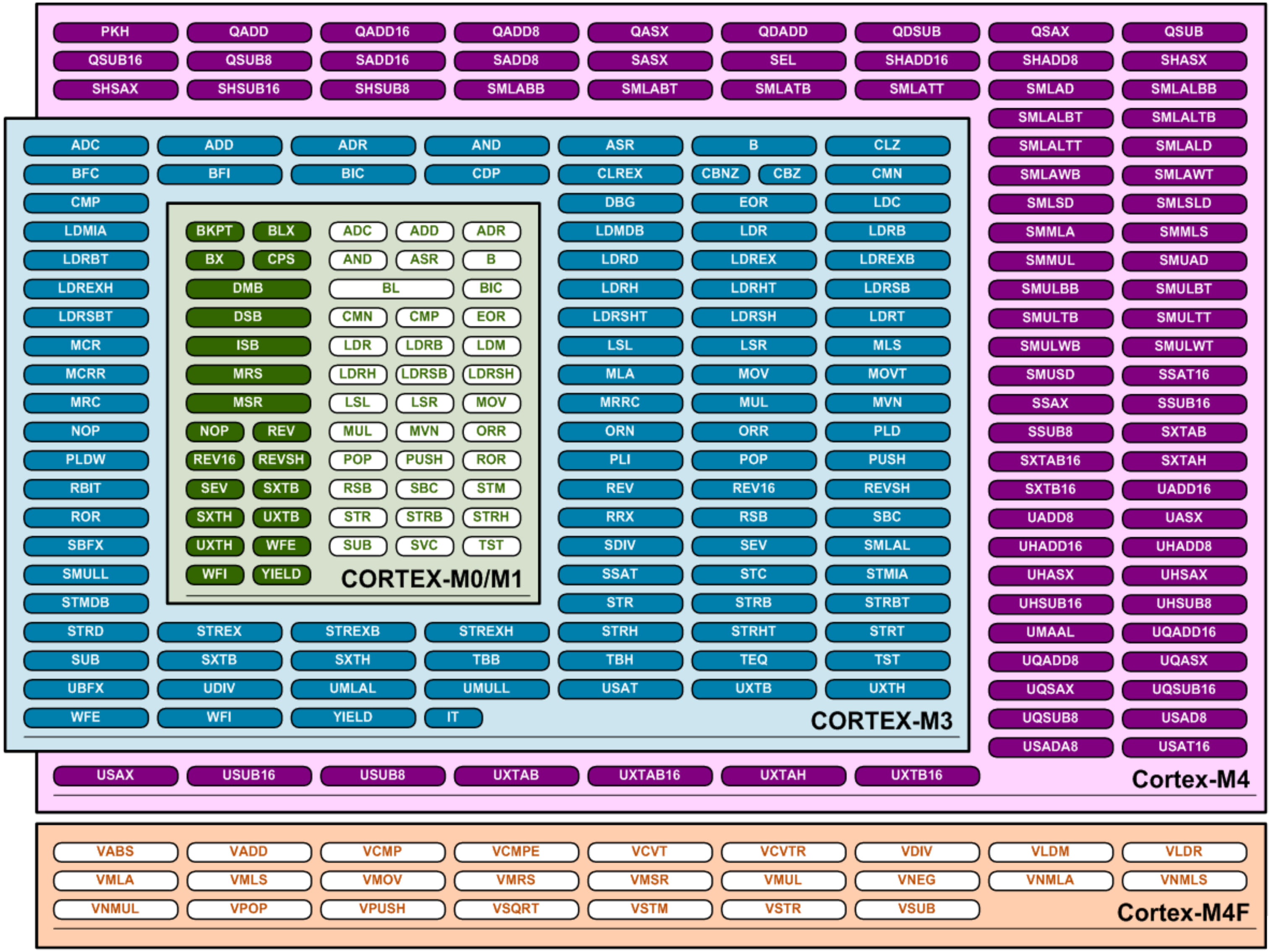
Cortex-M0	Cortex-M3	Cortex-M4
“8/16-bit” applications	“16/32-bit” applications	“32-bit/DSC” applications

Binary and tool compatible

life.augmented

COMPELFEST

Cortex-M processors binary compatible



PKH QADD QADD16 QADD8 QASX QDADD QDSUB QSAX QSUB
QSUB16 QSUB8 SADD16 SADD8 SASX SEL SHADD16 SHADD8 SHASX
SHSAX SHSUB16 SHSUB8 SMLABB SMLABT SMLATB SMLATT SMLAD SMLALBB
SMLALBT SMLALTB
SMLALTT SMLALD
SMLAWB SMLAWT
SMLSD SMLSLD
SMMLA SMMLS
SMMUL SMUAD
SMULBB SMULBT
SMULTB SMULTT
SMULWB SMULWT
SMUSD SSAT16
SSAX SSUB16
SSUB8 SXTAB
SXTAB16 SXTAH
SXTB16 UADD16
UADD8 UASX
UHADD16 UHADD8
UHASX UHSAX
UHSUB16 UHSUB8
UMAAL UQADD16
UQADD8 UQASX
UQSAX UQSUB16
UQSUB8 USAD8
USADA8 USAT16
USAX USUB16 USUB8 UXTAB UXTAB16 UXTAH UXTB16

Cortex-M4

ADC ADD ADR AND ASR B CLZ
BFC BFI BIC CDP CLREX CBNZ CBZ CMN
CMP DBG EOR LDC
LDMIA LDMDB LDR LDRB
LDRBT LDRD LDREX LDREXB
LDREXH LDRH LDRHT LDRSB
LDRSBT LDRSHT LDRSH LDRT
MCR LSL LSR MLS
MCRR MLA MOV MOVT
MRC MRRC MUL MVN
NOP ORN ORR PLD
PLDW PLI POP PUSH
RBIT REV REV16 REVSH
ROR RRX RSB SBC
SBFX SDIV SEV SMLAL
SMULL SSAT STC STMIA
STMDB STR STRB STRBT
STRD STREX STREXB STREXH STRH STRHT STRT
SUB SXTB SXTH TBB TBH TEQ TST
UBFX UDIV UMLAL UMULL USAT UXTB UXTH
WFE WFI YIELD IT

CORTEX-M3

BKPT BLX ADC ADD ADR
BX CPS AND ASR B
DMB BL BIC
DSB CMN CMP EOR
ISB LDR LDRB LDM
MRS LDRH LDRSB LDRSH
MSR LSL LSR MOV
NOP REV MUL MVN ORR
REV16 REVSH POP PUSH ROR
SEV SXTB RSB SBC STM
SXTH UXTB STR STRB STRH
UXTH WFE SUB SVC TST
WFI YIELD

CORTEX-M0/M1

VABS VADD VCMP VCMPE VCVT VCVTR VDIV VLDM VLDR
VMLA VMLS VMOV VMRS VMSR VMUL VNEG VNMLA VNMLS
VNMUL VPOP VPUSH VSQRT VSTM VSTR VSUB

Cortex-M4F

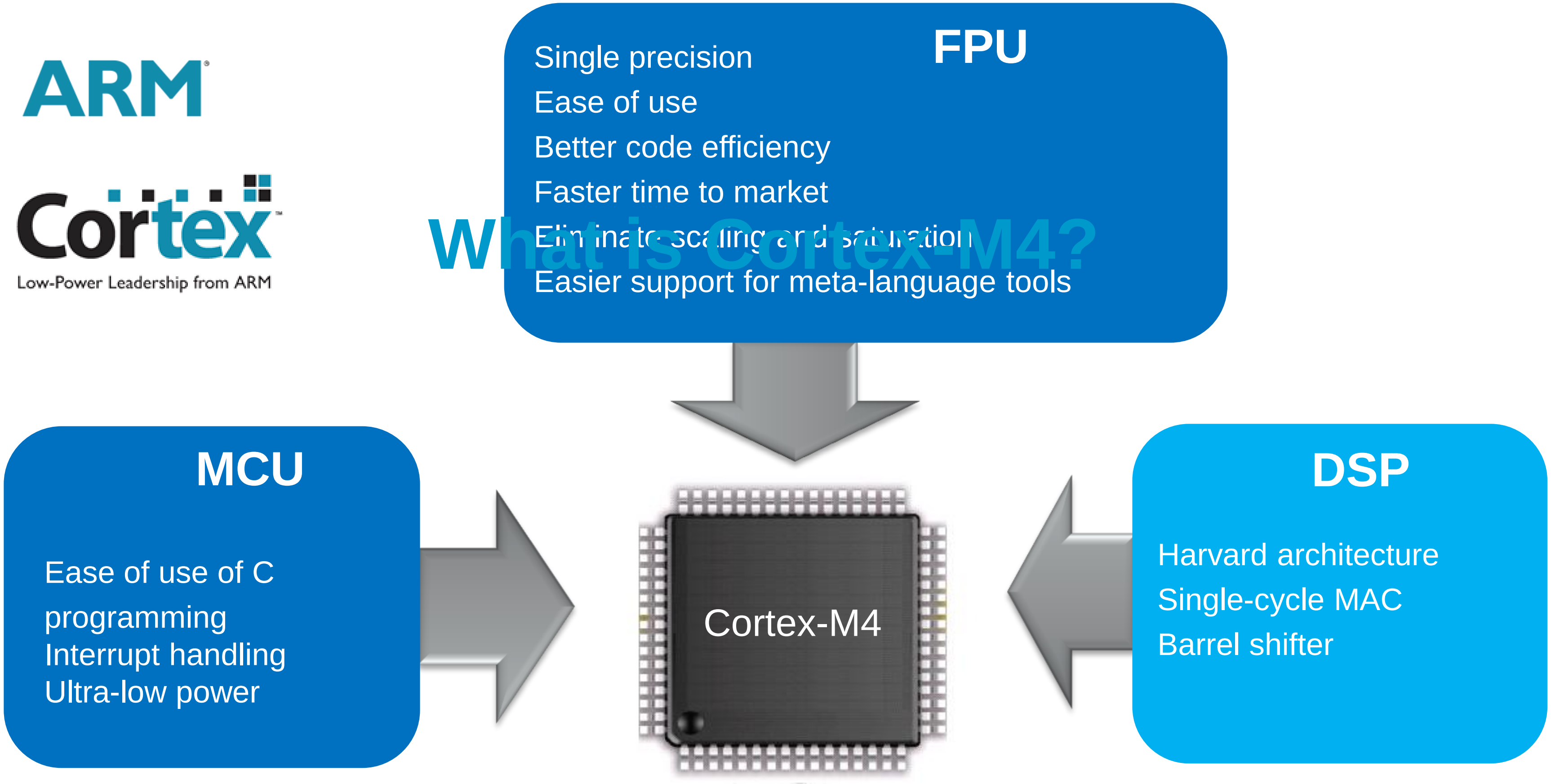
ARM Cortex M4 Core

ARM

Cortex

Low-Power Leadership from ARM

What is Cortex-M4?



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

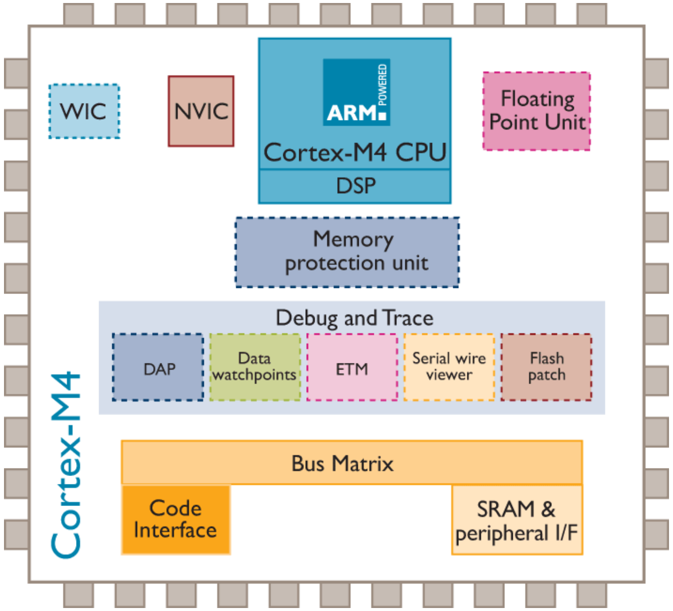
DSP

Harvard architecture
Single-cycle MAC
Barrel shifter

life.augmented

COMPELFEST

Cortex-M4 processor microarchitecure



- **ARMv7ME Architecture**
  - Thumb-2 Technology
  - DSP and SIMD extensions
  - Single cycle MAC (Up to 32 x 32 + 64 -> 64)
  - Optional single precision FPU
  - Integrated configurable NVIC
  - Compatible with Cortex-M3
- **Microarchitecture**
  - 3-stage pipeline with branch speculation
  - 3x AHB-Lite Bus Interfaces
- **Configurable for ultra low power**
  - Deep Sleep Mode, Wakeup Interrupt Controller
  - Power down features for Floating Point Unit
- **Flexible configurations for wider applicability**
  - Configurable Interrupt Controller (1-240 Interrupts and Priorities)
  - Optional Memory Protection Unit
  - Optional Debug & Trace

Cortex™
Low-Power Leadership from ARM

life.augmented

COMPELFEST

Cortex-M feature set comparison

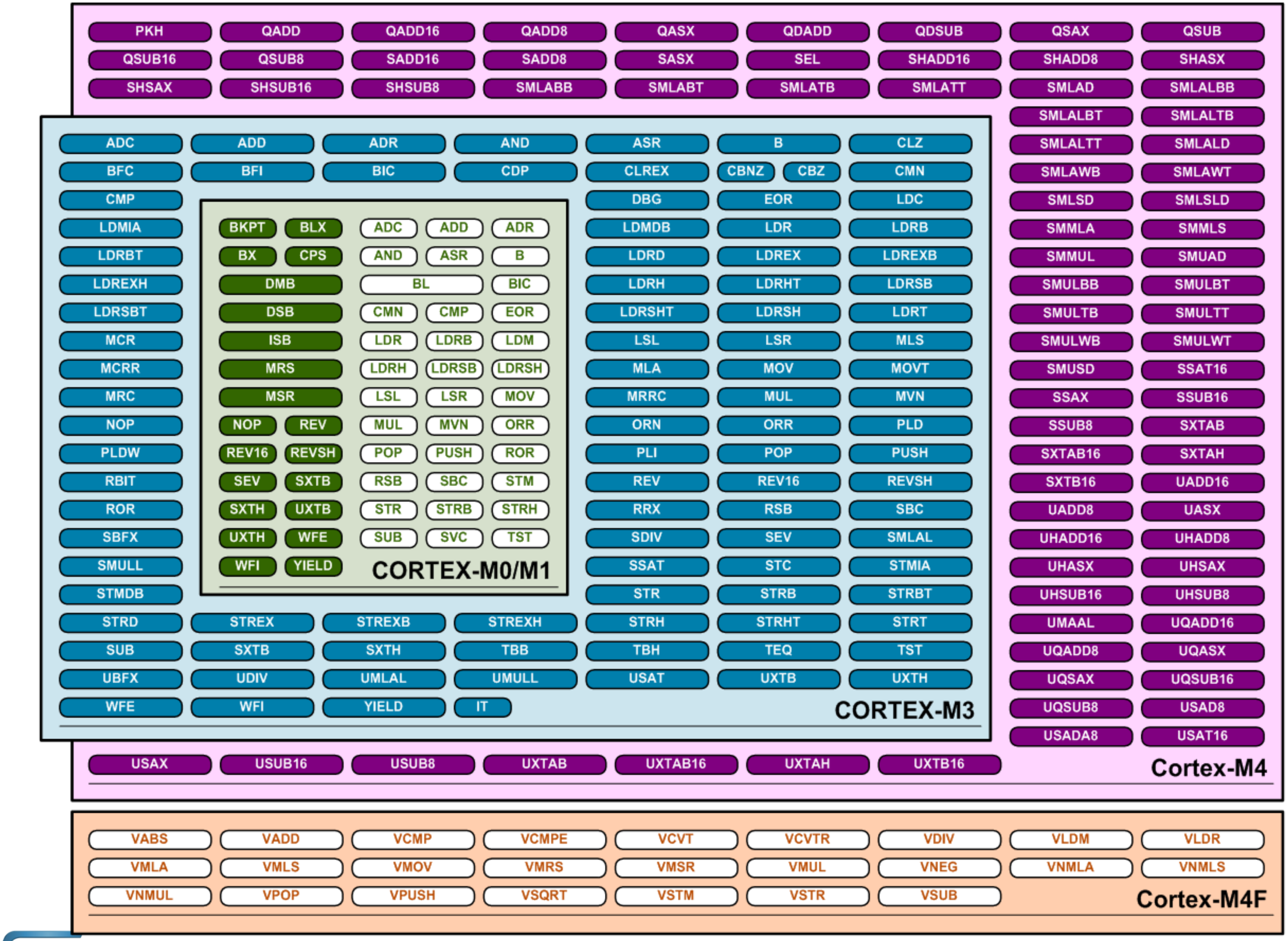
	Cortex-M0	Cortex-M3	Cortex-M4
Architecture Version	V6M	v7M	v7ME
Instruction set architecture	Thumb, Thumb-2 System Instructions	Thumb + Thumb-2	Thumb + Thumb-2, DSP, SIMD, FP
DMIPS/MHz	0.9	1.25	1.25
Bus interfaces	1	3	3
Integrated NVIC	Yes	Yes	Yes
Number interrupts	1-32 + NMI	1-240 + NMI	1-240 + NMI
Interrupt priorities	4	8-256	8-256
Breakpoints, Watchpoints	4/2/0, 2/1/0	8/4/0, 2/1/0	8/4/0, 2/1/0
Memory Protection Unit (MPU)	No	Yes (Option)	Yes (Option)
Integrated trace option (ETM)	No	Yes (Option)	Yes (Option)
Fault Robust Interface	No	Yes (Option)	No
Single Cycle Multiply	Yes (Option)	Yes	Yes
Hardware Divide	No	Yes	Yes
WIC Support	Yes	Yes	Yes
Bit banding support	No	Yes	Yes
Single cycle DSP/SIMD	No	No	**Yes**
Floating point hardware	No	No	**Yes**
Bus protocol	AHB Lite	AHB Lite, APB	AHB Lite, APB
CMSIS Support	Yes	Yes	Yes

Cortex M4 – DSP features

life.augmented

COMPELFEST

Cortex-M processors binary compatible



PKH QADD QADD16 QADD8 QASX QDADD QDSUB QSAX QSUB
QSUB16 QSUB8 SADD16 SADD8 SASX SEL SHADD16 SHADD8 SHASX
SHSAX SHSUB16 SHSUB8 SMLABB SMLABT SMLATB SMLATT SMLAD SMLALBB
SMLALBT SMLALTB
SMLALTT SMLALD
SMLAWB SMLAWT
SMLSD SMLSLD
SMMLA SMMLS
SMMUL SMUAD
SMULBB SMULBT
SMULTB SMULTT
SMULWB SMULWT
SMUSD SSAT16
SSAX SSUB16
SSUB8 SXTAB
SXTAB16 SXTAH
SXTB16 UADD16
UADD8 UASX
UHADD16 UHADD8
UHASX UHSAX
UHSUB16 UHSUB8
UMAAL UQADD16
UQADD8 UQASX
UQSAX UQSUB16
UQSUB8 USAD8
USADA8 USAT16
USAX USUB16 USUB8 UXTAB UXTAB16 UXTAH UXTB16

Cortex-M4

ADC ADD ADR AND ASR B CLZ
BFC BFI BIC CDP CLREX CBNZ CBZ CMN
CMP DBG EOR LDC
LDMIA LDMDB LDR LDRB
LDRBT LDRD LDREX LDREXB
LDREXH LDRH LDRHT LDRSB
LDRSBT LDRSHT LDRSH LDRT
MCR LSL LSR MLS
MCRR MLA MOV MOVT
MRC MRRC MUL MVN
NOP ORN ORR PLD
PLDW PLI POP PUSH
RBIT REV REV16 REVSH
ROR RRX RSB SBC
SBFX SDIV SEV SMLAL
SMULL SSAT STC STMIA
STMDB STR STRB STRBT
STRD STREX STREXB STREXH STRH STRHT STRT
SUB SXTB SXTH TBB TBH TEQ TST
UBFX UDIV UMLAL UMULL USAT UXTB UXTH
WFE WFI YIELD IT

CORTEX-M3

BKPT BLX ADC ADD ADR
BX CPS AND ASR B
DMB BL BIC
DSB CMN CMP EOR
ISB LDR LDRB LDM
MRS LDRH LDRSB LDRSH
MSR LSL LSR MOV
NOP REV MUL MVN ORR
REV16 REVSH POP PUSH ROR
SEV SXTB RSB SBC STM
SXTH UXTB STR STRB STRH
UXTH WFE SUB SVC TST
WFI YIELD

CORTEX-M0/M1

VABS VADD VCMP VCMPE VCVT VCVTR VDIV VLDM VLDR
VMLA VMLS VMOV VMRS VMSR VMUL VNEG VNMLA VNMLS
VNMUL VPOP VPUSH VSQRT VSTM VSTR VSUB

Cortex-M4F

Cortex
Low-Power Leadership from ARM

life.augmented

COMPELFEST

Cortex-M4 overview

- Main Cortex-M4 processor features
  - ARMv7-ME architecture revision
    - Fully compatible with Cortex-M3 instruction set
  - Single-cycle multiply-accumulate (MAC) unit
  - Optimized single instruction multiple data (SIMD) instructions
  - Saturating arithmetic instructions
  - Optional single precision Floating-Point Unit (FPU)
  - Hardware Divide (2-12 Cycles), same as Cortex-M3
  - Barrel shifter (same as Cortex-M3)

Single-cycle multiply-accumulate unit

- The multiplier unit allows any MUL or MAC instructions to be executed in a single cycle
  - Signed/Unsigned Multiply
  - Signed/Unsigned Multiply-Accumulate
  - Signed/Unsigned Multiply-Accumulate Long (64-bit)

- Benefits : Speed improvement vs. Cortex-M3
  - 4x for 16-bit MAC (dual 16-bit MAC)
  - 2x for 32-bit MAC
  - up to 7x for 64-bit MAC

Cortex-M4 extended single cycle MAC

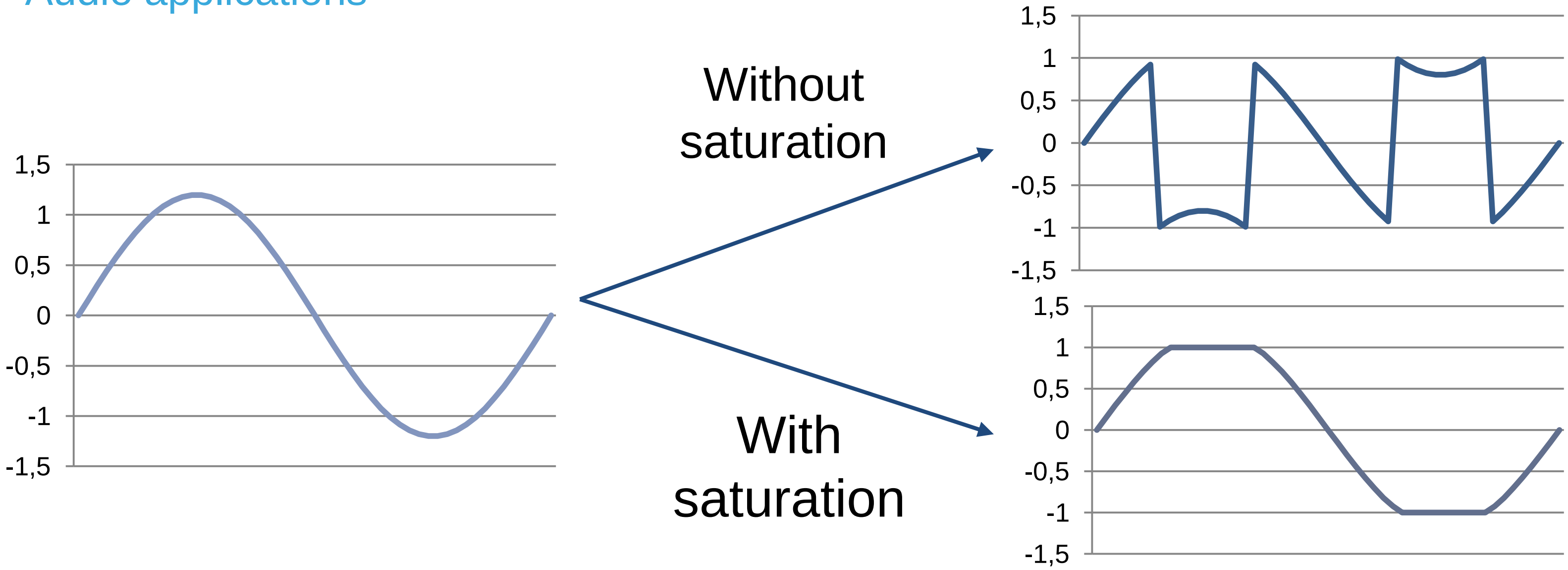
OPERATION	INSTRUCTIONS	CM3	CM4
16 x 16 = 32	SMULBB, SMULBT, SMULTB, SMULTT	n/a	1
16 x 16 + 32 = 32	SMLABB, SMLABT, SMLATB, SMLATT	n/a	1
16 x 16 + 64 = 64	SMLALBB, SMLALBT, SMLALTB, SMLALTT	n/a	1
16 x 32 = 32	SMULWB, SMULWT	n/a	1
(16 x 32) + 32 = 32	SMLAWB, SMLAWT	n/a	1
(16 x 16) ± (16 x 16) = 32	SMUAD, SMUADX, SMUSD, SMUSDX	n/a	1
(16 x 16) ± (16 x 16) + 32 = 32	SMLAD, SMLADX, SMLSD, SMLSDX	n/a	1
(16 x 16) ± (16 x 16) + 64 = 64	SMLALD, SMLALDX, SMLSLD, SMLSLDX	n/a	1
32 x 32 = 32	MUL	1	1
32 ± (32 x 32) = 32	MLA, MLS	2	1
32 x 32 = 64	SMULL, UMULL	5-7	1
(32 x 32) + 64 = 64	SMLAL, UMLAL	5-7	1
(32 x 32) + 32 + 32 = 64	UMAAL	n/a	1
32 ± (32 x 32) = 32 (upper)	SMMLA, SMMLAR, SMMLS, SMMLSR	n/a	1
(32 x 32) = 32 (upper)	SMMUL, SMMULR	n/a	1

All the above operations are single cycle on the Cortex-M4 processor

Saturated arithmetic

- Intrinsically prevents overflow of variable by clipping to min/max boundaries and remove CPU burden due to software range checks

- Benefits
  - Audio applications



Without saturation

With saturation

  - Control applications
    - The PID controllers’ integral term is continuously accumulated over time. The saturation automatically limits its value and saves several CPU cycles per regulators

Single-cycle SIMD instructions

- Stands for Single Instruction Multiple Data

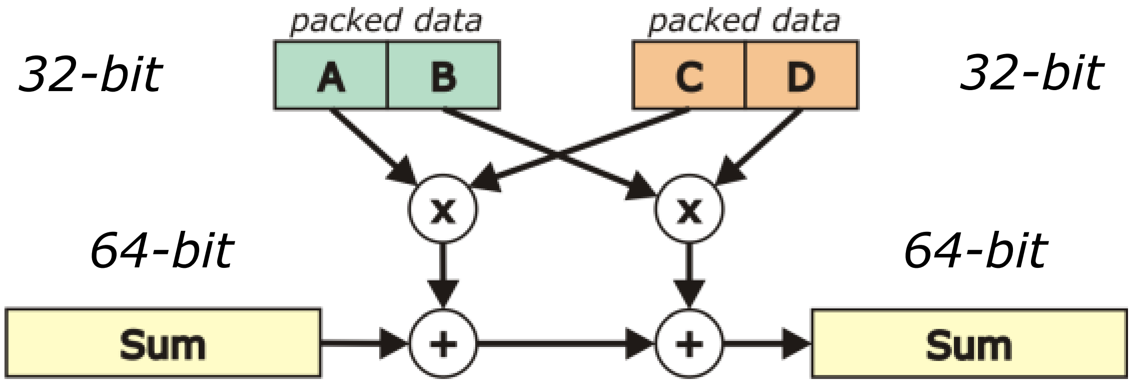
- Allows to do simultaneously several operations with 8-bit or 16-bit data format
  - Ex: dual 16-bit MAC (Result = 16x16 + 16x16 + 32)
  - Ex: Quad 8-bit SUB / ADD

- Benefits
  - Parallelizes operations (2x to 4x speed gain)
  - Minimizes the number of Load/Store instruction for exchanges between memory and register file (2 or 4 data transferred at once), if 32-bit is not necessary
  - Maximizes register file use (1 register holds 2 or 4 values)

SIMD operation example

- SIMD extensions perform multiple operations in one cycle

  Sum = Sum + (A x C) + (B x D)



- SIMD techniques operate with packed data

Cortex-M4 DSP instructions compared

		Cycle counts	
CLASS	**INSTRUCTION**	**CORTEX-M3**	**Cortex-M4**
Arithmetic	ALU operation (not PC)	1	1
	ALU operation to PC	3	3
	CLZ	1	1
	QADD, QDADD, QSUB, QDSUB	n/a	1
	QADD8, QADD16, QSUB8, QSUB16	n/a	1
	QDADD, QDSUB	n/a	1
	QASX, QSAX, SASX, SSAX	n/a	1
	SHASX, SHSAX, UHASX, UHSAX	n/a	1
	SADD8, SADD16, SSUB8, SSUB16	n/a	1
	SHADD8, SHADD16, SHSUB8, SHSUB16	n/a	1
	UQADD8, UQADD16, UQSUB8, UQSUB16	n/a	1
	UHADD8, UHADD16, UHSUB8, UHSUB16	n/a	1
	UADD8, UADD16, USUB8, USUB16	n/a	1
	UQASX, UQSAX, USAX, UASX	n/a	1
	UXTAB, UXTAB16, UXTAH	n/a	1
	USAD8, USADA8	n/a	1
Multiplication	MUL, MLA	1 - 2	1
	MULS, MLAS	1 - 2	1
	SMULL, UMULL, SMLAL, UMLAL	5 - 7	1
	SMULBB, SMULBT, SMULTB, SMULTT	n/a	1
	SMLABB, SMLBT, SMLATB, SMLATT	n/a	1
	SMULWB, SMULWT, SMLAWB, SMLAWT	n/a	1
	SMLALBB, SMLALBT, SMLALTB, SMLALTT	n/a	1
	SMLAD, SMLADX, SMLALD, SMLALDX	n/a	1
	SMLSD, SMLSDX	n/a	1
	SMLSLD, SMLSLD	n/a	1
	SMMLA, SMMLAR, SMMLS, SMMLSR	n/a	1
	SMMUL, SMMULR	n/a	1
	SMUAD, SMUADX, SMUSD, SMUSDX	n/a	1
	UMAAL	n/a	1
Division	SDIV, UDIV	2 - 12	2 – 12

Single cycle MAC

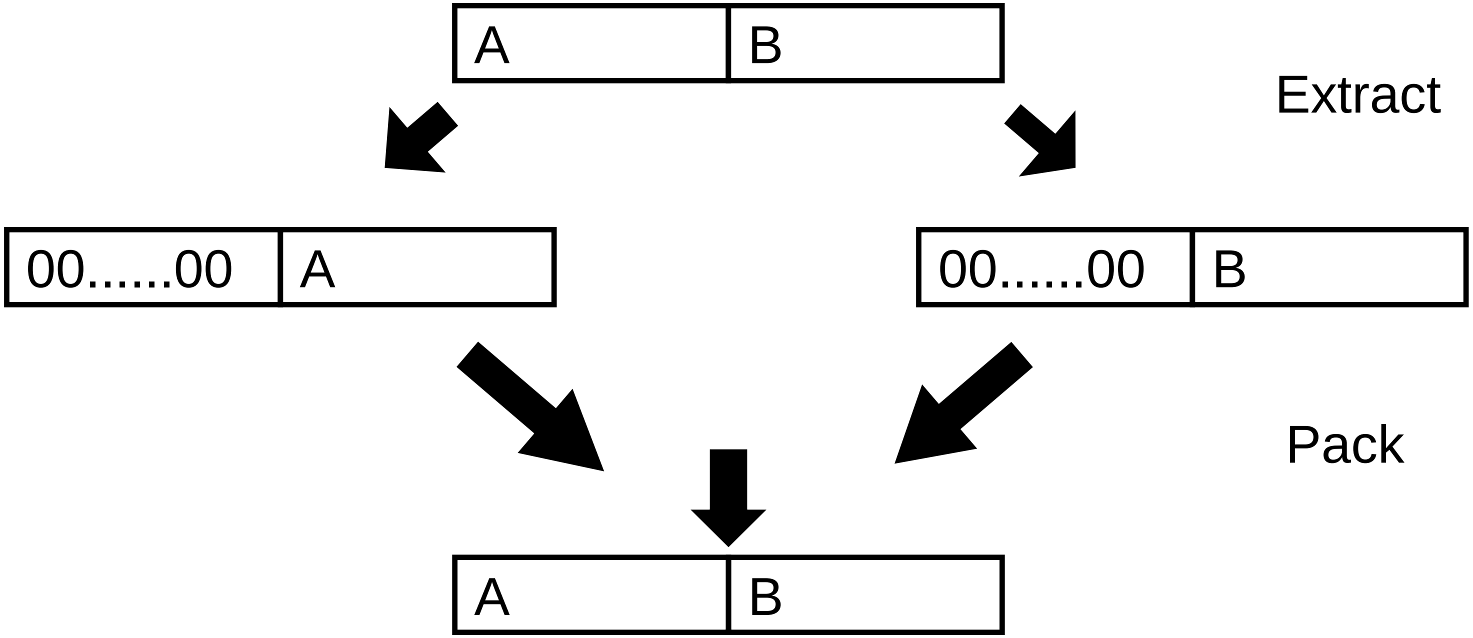
life.augmented

Cortex-M4 non–DSP instructions

		Cycle counts	
CLASS	**INSTRUCTION**	**CORTEX-M3**	**Cortex-M4**
Load/Store	Load single byte to R0-R14	1 - 3	1 - 3
	Load single halfword to R0-R14	1 - 3	1 - 3
	Load single word to R0-R14	1 - 3	1 - 3
	Load to PC	5	5
	Load double-word	3	3
	Store single word	1 - 2	1 - 2
	Store double word	3	3
	Load-multiple registers (not PC)	N+1	N+1
	Load-multiple registers plus PC	N+5	N+5
	Store-multiple registers	N+1	N+1
	Load/store exclusive	2	2
	SWP	n/a	n/a
Branch	B, BL, BX, BLX	2 - 3	2 - 3
	CBZ, CBNZ	3	3
	TBB, TBH	5	5
	IT	0 - 1	0 - 1
Special	MRS	1	1
	MSR	1	1
	CPS	1	1
Manipulation	BFI, BFC	1	1
	RBIT, REV, REV16, REVSH	1	1
	SBFX, UBFX	1	1
	UXTH, UXTB, SXTH, SXTB	1	1
	SSAT, USAT	1	1
	SEL	n/a	1
	SXTAB, SXTAB16, SXTAH	n/a	1
	UXTB16, SXTB16	n/a	1
	SSAT16, USAT16	n/a	1
	PKHTB, PKHBT	n/a	1

Packed data types

- Several instructions operate on “packed” data types
  - Byte or halfword quantities packed into words
  - Allows more efficient access to packed structure types
  - SIMD instructions can act on packed data
  - Instructions to extract and pack data



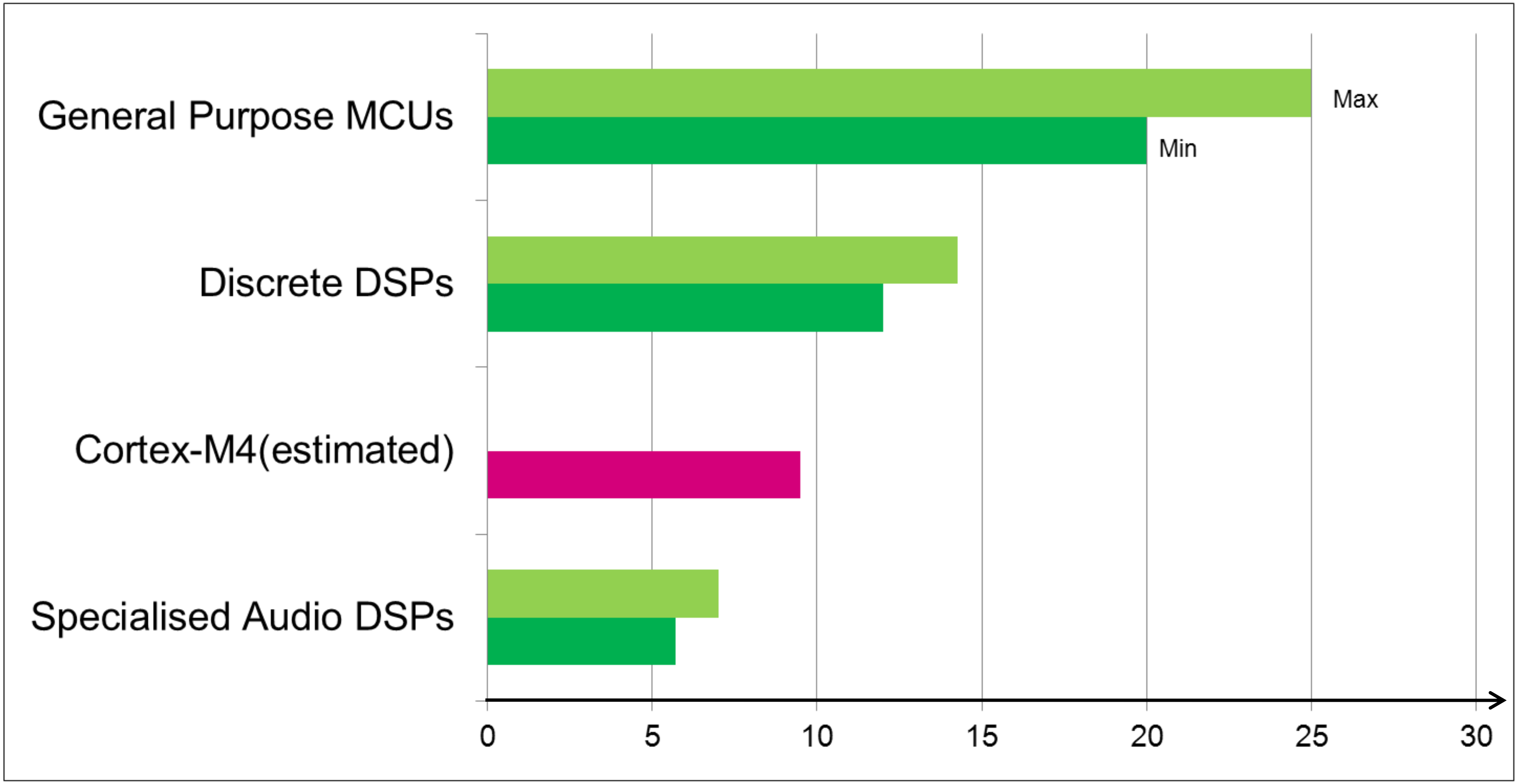
DSP performances for control application

- Example based on a complex formula used for sensorless motor drive
- Gain comes for load operations and SIMD instructions
- Total gain on this part is 25 to 35%

Cortex M3 (28-38 c.)	Cortex M4 (18-28 c.)
LDRSH R12,[R4, #+12]	LDR R10,[R4, #+12]
LDRSH R0,[SP, #+20]	(1 single 32-bit load replacing two 16-bit load with sign extension. **Gain: 2 cycles**
SXTH LR,R8	
MUL R8,LR,R0	
LDR R1,[R4, #+44]	
SDIV R0,R1,R7	
LDRSH R2,[R4, #+24]	
LDRSH R3,[R4, #+26]	LDR R2,[R4, #+22]
LDRSH R10,[R4, #+22]	(1 single 32-bit load replacing to 16-bit with sign extension. **Gain: 2 cycles**)
SXTH R6,R6	
MLS R5,R6,R10,R5	
MLA R5,R9,R12,R5	SMLSD R5, R10, R6, R5 (1 SIMD instruction replacing two multiply-accumulate. **Gain: 3 cycles**)
ASR R6,R8,#+15	
MLA R5,R6,R3,R5	
SXTH R0,R0	
MLS R5,R0,R2,R5	SMLSD R5, R0, R2 (1 SIMD instruction replacing two multiply-accumulate. **Gain: 3 cycles**)
STR R5,[SP, #+12]	

life.augmented

DSP application example: MP3 audio playback



MHz required for MP3 decode (smaller is better !)

DSP concept from ARM (*)

DSP lib provided for free by ARM

- The benefits of software libraries for Cortex-M4
  - Enables end user to develop applications faster
    - Keeps end user abstracted from low level programming
  - Benchmarking vehicle during system development
  - Clear competitive positioning against incumbent DSP/DSC offerings
  - Accelerate third party software development

- Keeping it easy to access for end user
  - Minimal entry barrier - very easy to access and use

- One standard library – no duplicated efforts
  - ARM channels effort/resources with software partner
  - Value add through another level of software – eg: filter config tools

Cortex
Low-Power Leadership from ARM

DSP lib function list snapshot

- Basic math – vector mathematics
- Fast math – sin, cos, sqrt etc
- Interpolation – linear, bilinear
- Complex math
- Statistics – max, min,RMS etc
- Filtering – IIR, FIR, LMS etc
- Transforms – FFT(real and complex) , Cosine transform etc
- Matrix functions

- PID Controller
- Support functions – copy/fill arrays, data type conversions etc

Cortex™
Low-Power Leadership from ARM

COMPELFEST

life.augmented

Tools

- Matlab / Simulink
  - Embedded coder for code generation
  - Mathworks
    - Demo being developed (availability end of year)
  - Aimagin (Rapidstm32)

- Filter design tools
  - Lot of tools available, most of them commercial product, some with low-cost offer, few free
    - http://www.dspguru.com/dsp/links/digital-filter-design-software

Floating Point Unit

life.augmented

COMPELFEST

Overview

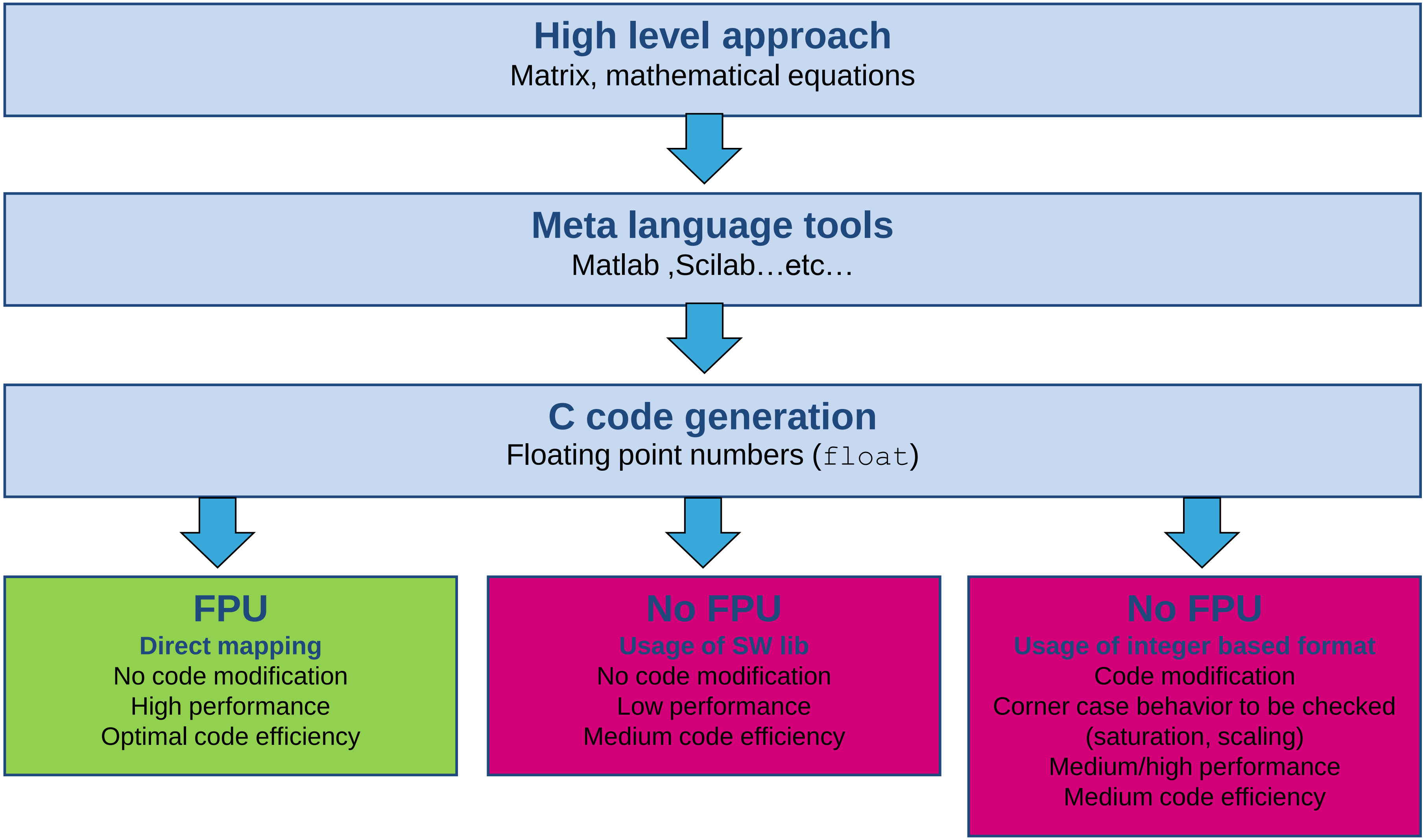
- **FPU : Floating Point Unit**
  - Handles “real” number computation
  - Standardized by **IEEE.754-2008**
    - Number format
    - Arithmetic operations
    - Number conversion
    - Special values
    - 4 rounding modes
    - 5 exceptions and their handling

- **ARM Cortex-M FPU ISA**
  - Supports
    - Add, subtract, multiply, divide
    - Multiply and accumulate
    - Square root operations

Floating Point Unit

- **Introduction**
  - FPU usage
  - Historical perspective
  - Benefit of floating point arithmetic
  - Example & performances
  - Rounding issues
- IEEE 754
- ARM FPv4-SP Single Precision FPU

FPU usage



High level approach

Matrix, mathematical equations

Meta language tools

Matlab ,Scilab…etc…

C code generation

Floating point numbers (`float`)

FPU

Direct mapping

No code modification

High performance

Optimal code efficiency

No FPU

Usage of SW lib

No code modification

Low performance

Medium code efficiency

No FPU

Usage of integer based format

Code modification

Corner case behavior to be checked

(saturation, scaling)

Medium/high performance

Medium code efficiency

Historical perspective

- Usage of floating point as always been a **need** for computers since the beginning (Konrad Zuse - 1935)
- But the **complexity of implementation** discarded their usage during decades (IBM 704 - 1956)
- Floating point unit where implemented in mainframes with various coding techniques depending of the manufacturer
- IBM PC where designed to have floating point capabilities through optional **arithmetic coprocessors** (80x87 series)
- The standardization of floating point coding was done in the 80's through the **IEEE 754** standard in 1985
- The Intel 80387 was the first intel coprocessor to implement the full IEEE 754 standard in 1987

Benefits of a Floating-Point Unit

- **FPU allows to handled “real” numbers (C float) without penalty**

- **If no FPU**
  - Need to emulate it by software
  - Need to rework all its algorithm and fixed point implementation to handle scaling and saturation issues

- **FPU eases usage of high-level design tools (MatLab/Simulink)**
  - Now part of microcontroller development flow for advanced applications.
  - Derivate code directly using native floating point leads to :
    - quicker time to market (faster development)
    - easy code maintenance
    - more reliable application code as no post modification are needed (no critical scaling operations to move to fixed point)

C language example

```
float function1(float number1, float number2)
{       float temp1, temp2;
        temp1 = number1 + number2;
        temp2 = number1/temp1;
        return temp2;
}
```

Code compiled on Cortex-M3

```
# float function1(…)
# { …
#    temp1 = number1 + number2;
     MOVS     R1,R4
     BL       __aeabi_fadd
     MOVS     R1,R0
#    temp2 = number1/temp1;
     MOVS     R0,R4
     BL       __aeabi_fdiv
#    return temp2;
     POP      {R4,PC}
# }
```

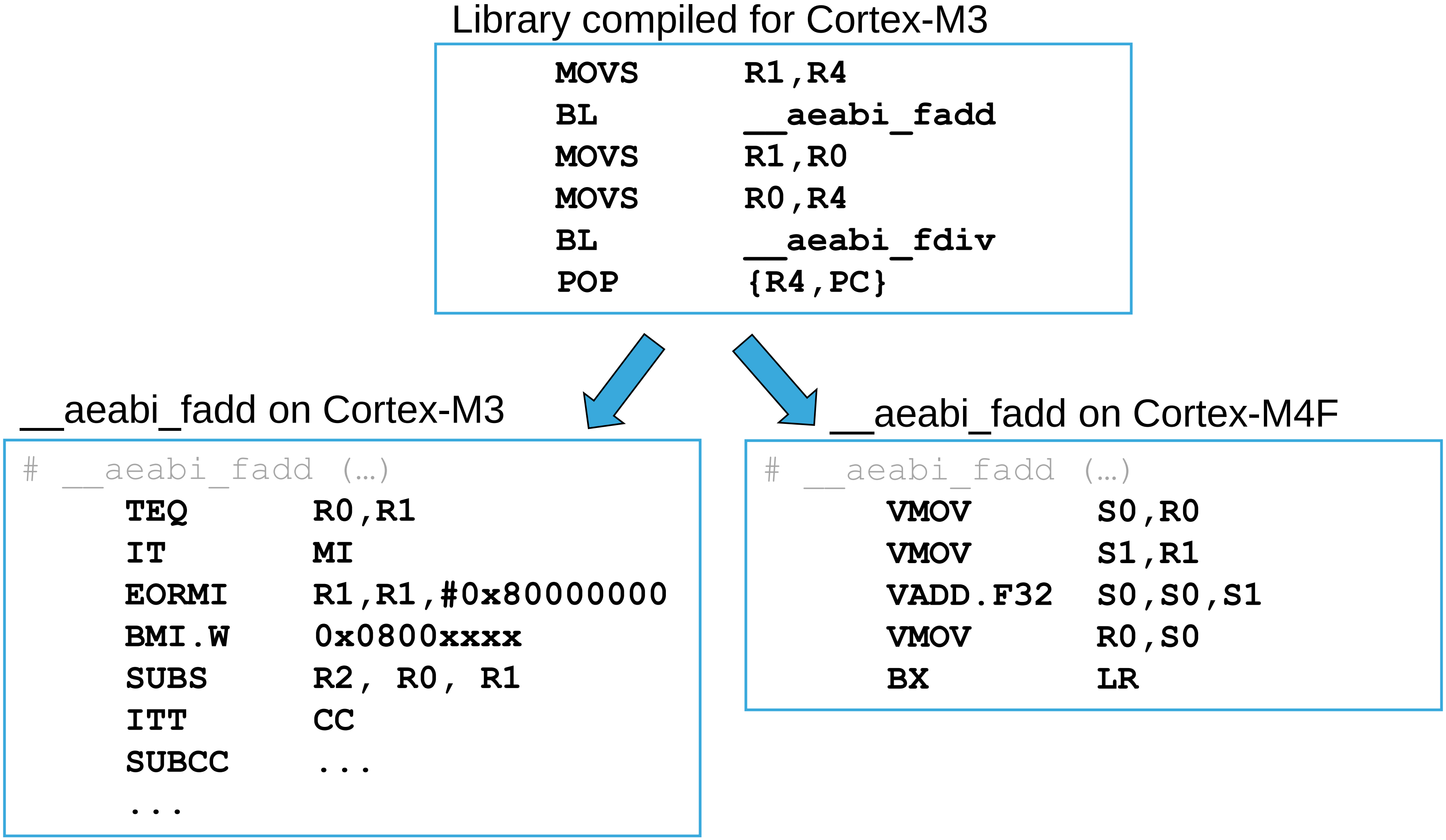
Same code compiled on Cortex-M4F

```
float function1(…)
# { …
#    temp1 = number1 + number2;
     VADD.F32 S1,S0,S1
#    temp2 = number1/temp1;
     VDIV.F32 S0,S0,S1
#
#    return temp2;
     BX       LR
# }
```

FPU assembly instructions

Call Soft-FPU (keil’s software library)

Binary library example



Library compiled for Cortex-M3

```
MOVS      R1,R4
BL        __aeabi_fadd
MOVS      R1,R0
MOVS      R0,R4
BL        __aeabi_fdiv
POP       {R4,PC}
```

__aeabi_fadd on Cortex-M3

```
# __aeabi_fadd (…)
     TEQ       R0,R1
     IT        MI
     EORMI     R1,R1,#0x80000000
     BMI.W     0x0800xxxx
     SUBS      R2, R0, R1
     ITT       CC
     SUBCC     ...
     ...
```

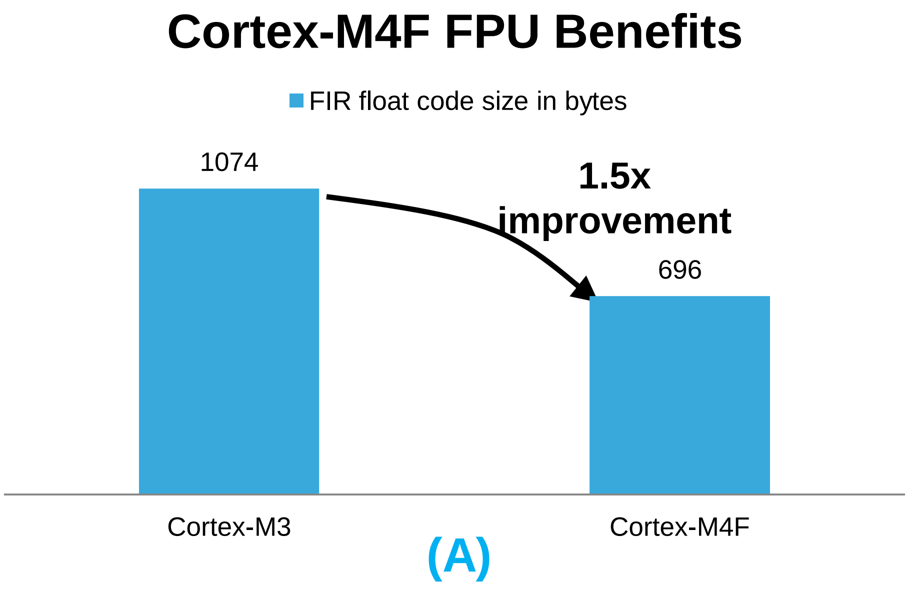
__aeabi_fadd on Cortex-M4F

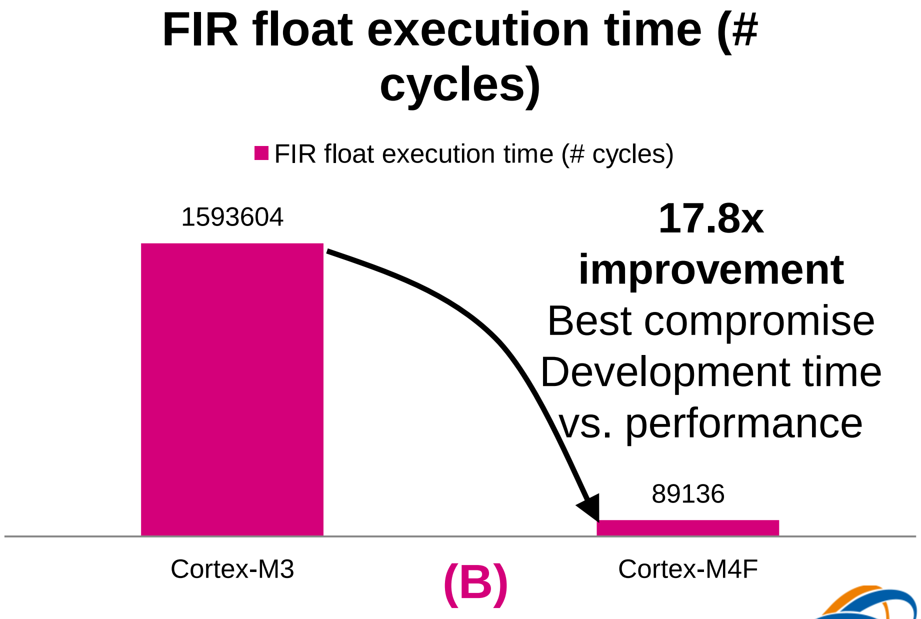
```
# __aeabi_fadd (…)
     VMOV        S0,R0
     VMOV        S1,R1
     VADD.F32    S0,S0,S1
     VMOV        R0,S0
     BX          LR
```

Reduced code size & Enhanced performances

Benefits of a Floating-Point Unit

- Comparison for a 166 coefficient FIR on float 32 with and without FPU (CMSIS library)
  - Improvement in code size (A)
  - Improvement in performance (B)





Best compromise Development time vs. performance

Cortex-M4 : Floating point unit Features

- **Single precision FPU**
- **Conversion between**
  - Integer numbers
  - Single precision floating point numbers
  - Half precision floating point numbers
- **Handling floating point exceptions** (Untrapped)
- **Dedicated registers**
  - 16 single precision registers (S0-S15) which can be viewed as 16 Doubleword registers for load/store operations (D0-D7)
  - FPSCR for status & configuration

Rounding issues

- **The precision has some limits**
  - Rounding errors can be accumulated along the various operations an may provide unaccurate results (do not do financial operations with floatings…)

- **Few examples**
  - If you are working on two numbers in different base, the hardware automatically « denormalize » on of the two number to make the calculation in the same base
  - If you are substracting two numbers very closed you are loosing the relative precision (also called cancellation error)

- **If you are « reorganizing » the various operations, you may not obtain the same result as because of the rounding errors…**
  - Value1 = ((2.0f - 1.99f) - 0.01f); /* Value1 = -9.313266E-9 */
  - Value2 = (2.0f - (1.99f + 0.01f)); /* Value2 = 0 */

IEEE 754

IEEE

life.augmented

COMPELFEST

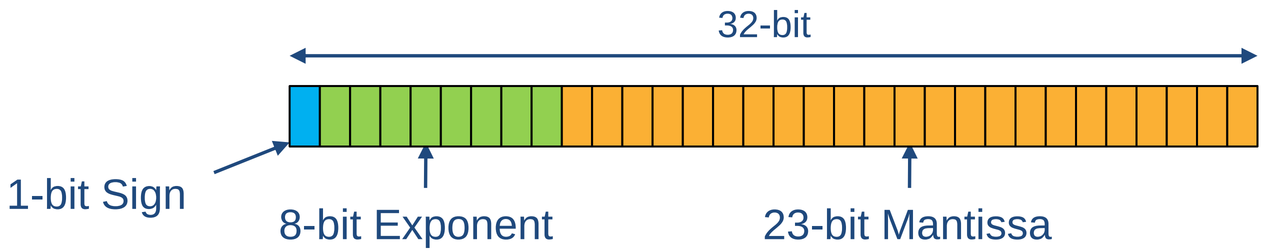
Floating Point Unit

- Introduction
- **IEEE 754**
  - Number format
  - Arithmetic operations
  - Number conversion
  - Special values
  - 4 rounding modes
  - 5 exceptions and their handling
- ARM FPv4-SP Single Precision FPU

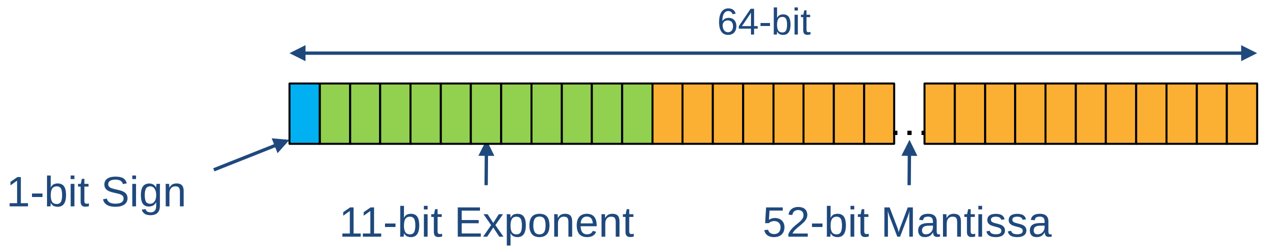
Number format

- **3 fields**
  - Sign
  - Biased exponent (sum of an exponent plus a constant bias)
  - Fractions (or mantissa)

- **Single precision : 32-bit coding**

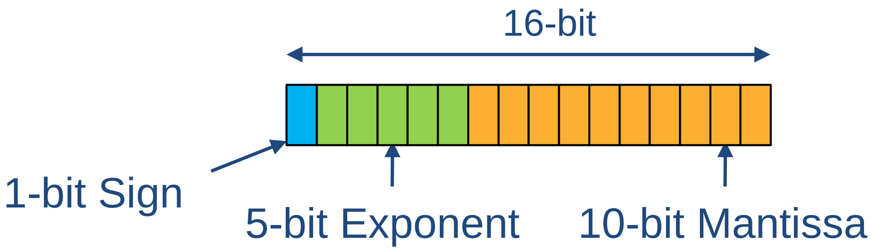


- **Double precision : 64-bit coding**



Number format

- **Half precision : 16-bit coding**



- Can also be used for storage in higher precision FPU
- ARM has an alternative coding for Half precision

Normalized number value

- **Normalized number**
  - Code a number as :

    A sign + Fixed point number between 1.0 and 2.0 multiplied by 2^N

- **Sign field (1-bit)**
  - 0 : positive
  - 1 : negative

- **Single precision exponent field (8-bit)**
  - **Exponent range** : 1 to 254 (0 and 255 reserved)
  - **Bias** : 127
  - **Exponent - bias range** : -126 to +127

- **Single precision fraction (or mantissa) (23-bit)**
  - **Fraction** : value between 0 and 1 : $\sum(N_i.2^{-i})$ with i in 1 to 24 range
  - The 23 N_i values are store in the fraction field

$$(-1)^s \times (1 + \sum(N_i.2^{-i})\ ) \times 2^{exp-bias}$$

Number value

- **Single precision coding of -7**
  - **Sign bit** = 1
  - **7** = 1.75 x 4 = (1 + ½ + ¼) x 4 = (1 + ½ + ¼) x 2^2
    = (1 + 2^{-1} + 2^{-2}) x 2^2
  - **Exponent** = 2 + bias = 2 + 127 = 129 = 0b10000001
  - **Mantissa** = 2^{-1} + 2^{-2} = 0b11000000000000000000000

- **Result**
  - Binary coding : 0b 1 10000001 11000000000000000000000
  - Hexadecimal value : **0xC0E00000**

Special values

- **Denormalized (Exponent field all “0”, Mantisa non 0)**
  - Too small to be normalized (but some can be normalized afterward)
  - $(-1)^s \times (\sum(N_i.2^{-i}) \times 2^{-bias}$
- **Infinity (Exponent field “all 1”, Mantissa “all 0”)**
  - Signed
  - Created by an overflow or a division by 0
  - Can not be an operand
- **Not a Number : NaN (Exponent filed “all1”, Mantisa non 0)**
  - Quiet NaN : propagated through the next operations (ex: 0/0)
  - Signalled NaN : generate an error
- **Signed zero**
  - Signed because of saturation

Summary of IEEE 754 number coding

Sign	Exponent	Mantissa	Number
0	0	0	+0
1	0	0	-0
0	Max	0	+oo
1	Max	0	-oo
-	Max	!=0 MSB=1	QNaN
-	Max	!=0 MSB=0	SNaN
-	0	!=0	Denormalized number
-	[1, Max-1]	-	Normalized number

Floating-point rounding

- **Round to nearest**
  - Default rounding mode
  - If the two nearest are equally near : select the one with the LSB equal to 0
- **Directed rounding**
  - 3 user-selectable directed rounding modes
  - Round toward +oo, -oo or 0
- **Usage**
  - Program through FPU configuration registers

Floating-point operations

- **Add**
- **Subtract**
- **Multiply**
- **Divide**
- **Remainder**
- **Square root**

Floating-point format conversion

- **Floating-point and Integer**
- **Round-floating point number to integer value**
- **Binary-Decimal**
- **Comparison**

Exceptions

- **Invalid operation**
  - Resulting in a NaN
- **Division by zero**
- **Overflow**
  - The result depend of the rounding mode and can produce a +/-oo or the +/-Max value to be written in the destination register
- **Underflow**
  - Write the denormalize number in the destination register
- **Inexact result**
  - Caused by rounding

Exception handling

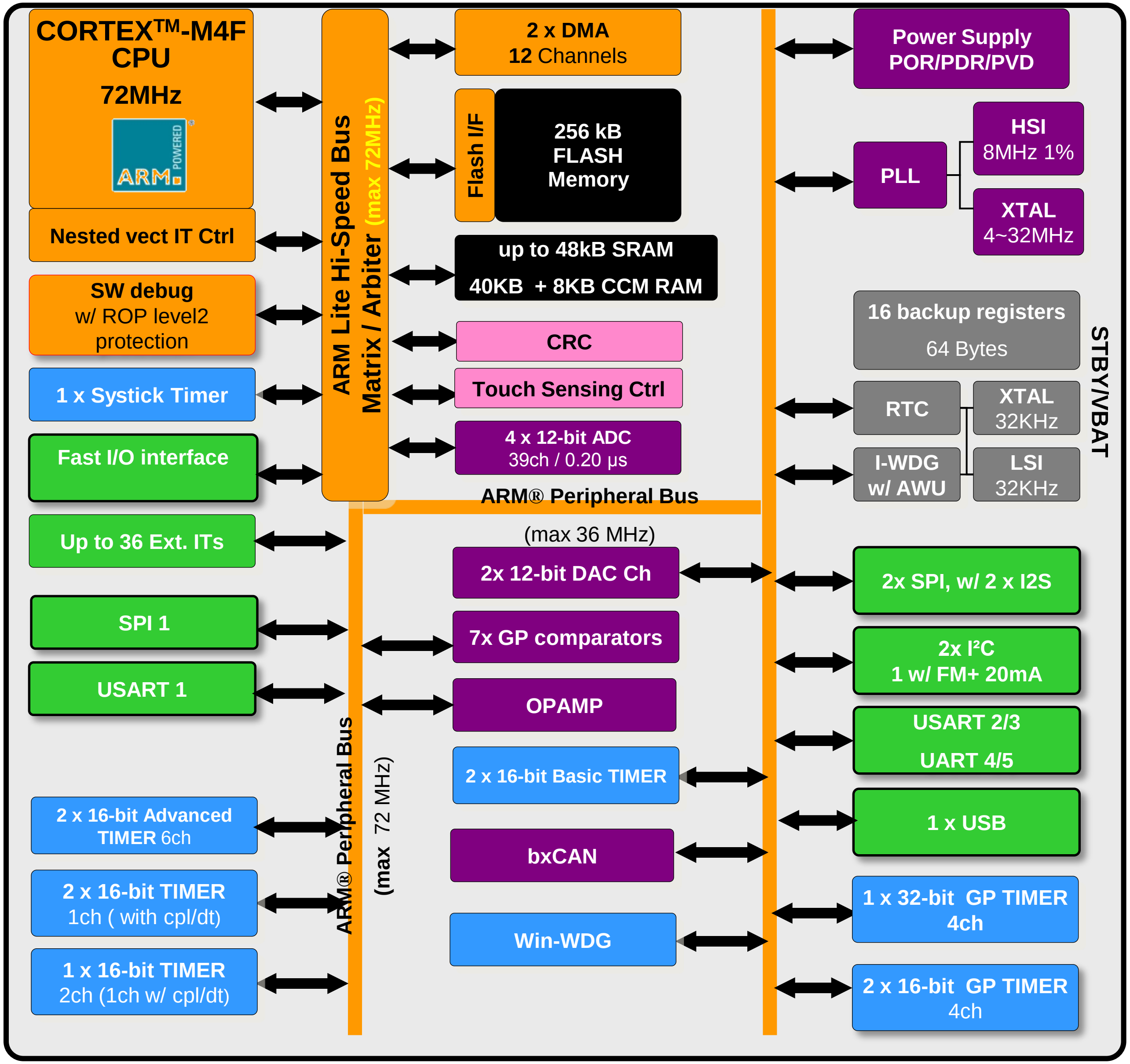
- A **TRAP** can be requested by the user for any of the 5 exception with a specific handler

- The **TRAP handler** can return a **value to be used instead** of the exceptional operation result

Block Diagrams

STM32F30x Series

- **ARM 32-bit Cortex-M4F CPU**
- **Operating Voltage:**
  - VDD = VDD = 2.0 V to 3.6 V or 1.8V +/8%
  - VBAT = 1.8 V to 3.6 V
- **Safe Reset System (Integrated Power On Reset (POR)/Power Down Reset (PDR) + Programmable voltage detector (PVD))**
- **Embedded Memories:**
  - FLASH: up 256 Kbytes
  - SRAM: up 48 Kbytes
- **CRC calculation unit**
- **2 x DMA: 12 channels .**
- **Power Supply with software configurable internal regulator and low power modes.**
- **Low Power Modes with Auto Wake-up**
- **Low power calendar RTC with 64 bytes of backup registers**
- **Up to 72 MHz frequency managed & monitored by the Clock Control w/ Clock Security System**
- **Rich set of peripherals & IOs**
  - 2× 12-bit DAC channels with output buffer
  - 7 general purpose comparators (Window mode and wakeup from low-power mode)
  - 4 operational amplifiers
  - Dual Watchdog Architecture
  - 13 Timers w/ advanced control features (including 1 Cortex SysTick timer and 2 WDGs timers)
  - 14 communication Interfaces
  - Up to 88 fast I/Os all mappable on external interrupts/event
  - 4x12-bits 5Msps ADC w/ up to 39 external channels + Temperature sensor/ voltage reference/VBAT measurement



CORTEX^{TM}-M4F CPU
72MHz

Nested vect IT Ctrl

SW debug
w/ ROP level2 protection

1 x Systick Timer

Fast I/O interface

Up to 36 Ext. ITs

SPI 1

USART 1

2 x 16-bit Advanced TIMER 6ch

2 x 16-bit TIMER
1ch (with cpl/dt)

1 x 16-bit TIMER
2ch (1ch w/ cpl/dt)

ARM Lite Hi-Speed Bus Matrix / Arbiter (max 72MHz)

2 x DMA
12 Channels

Flash I/F

256 kB FLASH Memory

up to 48kB SRAM
40KB + 8KB CCM RAM

CRC

Touch Sensing Ctrl

4 x 12-bit ADC
39ch / 0.20 µs

ARM® Peripheral Bus
(max 36 MHz)

ARM® Peripheral Bus
(max 72 MHz)

2x 12-bit DAC Ch

7x GP comparators

OPAMP

2 x 16-bit Basic TIMER

bxCAN

Win-WDG

Power Supply
POR/PDR/PVD

PLL

HSI
8MHz 1%

XTAL
4~32MHz

16 backup registers
64 Bytes

RTC

XTAL
32KHz

I-WDG
w/ AWU

LSI
32KHz

STBY/VBAT

2x SPI, w/ 2 x I2S

2x I²C
1 w/ FM+ 20mA

USART 2/3
UART 4/5

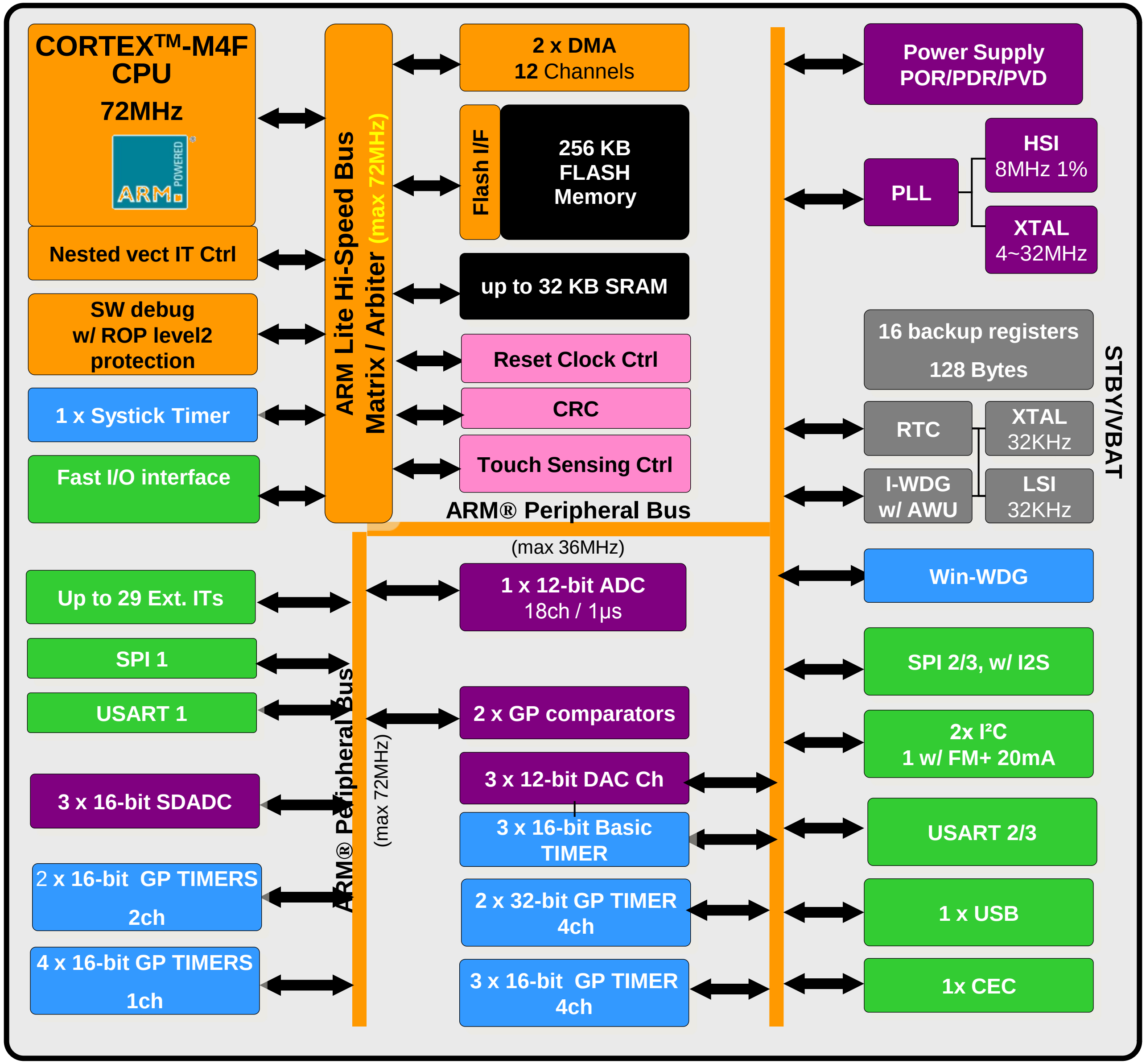
1 x USB

1 x 32-bit GP TIMER
4ch

2 x 16-bit GP TIMER
4ch

STM32F37x Series

- ARM 32-bit Cortex-M4F CPU
- Operating Voltage:
  - VDD = 2.0 V to 3.6 V or 1.8V +/8%
  - VBAT = 1.8 V to 3.6 V
- Safe Reset System (Integrated Power On Reset (POR)/Power Down Reset (PDR) + Programmable voltage detector (PVD))
- Embedded Memories:
  - FLASH: up 256 Kbytes
  - SRAM: up 32Kbytes
- CRC calculation unit
- 2 x DMA: 12 Channels
- Power Supply with software configurable internal regulator and low power modes.
- Low Power Modes with Auto Wake-up
- Low power calendar RTC with 128 bytes of backup registers
- Up to 72 MHz frequency managed & monitored by the Clock Control w/ Clock Security System
- Rich set of peripherals & IOs
  - 3 × 12-bit DAC channels with output buffer
  - 2 general purpose comparators (Window mode and wakeup from low-power mode)
  - Dual Watchdog Architecture
  - 17 Timers (including Cortex SysTick and WDGs)
  - 14 communication Interfaces
  - Up to 84 fast I/Os all mappable on external interrupts/event
  - 1 x12-bits SAR ADC w/ up to 18 external channels .
  - 3 x 16-bit Sigma-Delta ADC with conversion speed up to 50 ksps and up to 19 single/ 10 diff channels



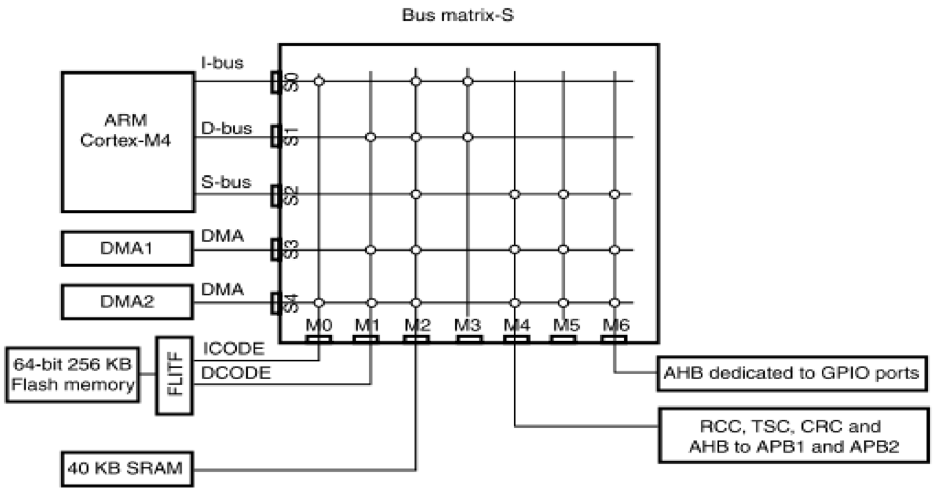
CORTEX™-M4F CPU
72MHz
Nested vect IT Ctrl
SW debug w/ ROP level2 protection
1 x Systick Timer
Fast I/O interface
ARM Lite Hi-Speed Bus Matrix / Arbiter (max 72MHz)
2 x DMA 12 Channels
Flash I/F
256 KB FLASH Memory
up to 32 KB SRAM
Reset Clock Ctrl
CRC
Touch Sensing Ctrl
ARM® Peripheral Bus (max 36MHz)
Power Supply POR/PDR/PVD
PLL
HSI 8MHz 1%
XTAL 4~32MHz
16 backup registers 128 Bytes
RTC
XTAL 32KHz
I-WDG w/ AWU
LSI 32KHz
STBY/VBAT
Up to 29 Ext. ITs
SPI 1
USART 1
3 x 16-bit SDADC
2 x 16-bit GP TIMERS 2ch
4 x 16-bit GP TIMERS 1ch
ARM® Peripheral Bus (max 72MHz)
1 x 12-bit ADC 18ch / 1µs
2 x GP comparators
3 x 12-bit DAC Ch
3 x 16-bit Basic TIMER
2 x 32-bit GP TIMER 4ch
3 x 16-bit GP TIMER 4ch
Win-WDG
SPI 2/3, w/ I2S
2x I²C 1 w/ FM+ 20mA
USART 2/3
1 x USB
1x CEC

Memory and System Architecture

System Architecture

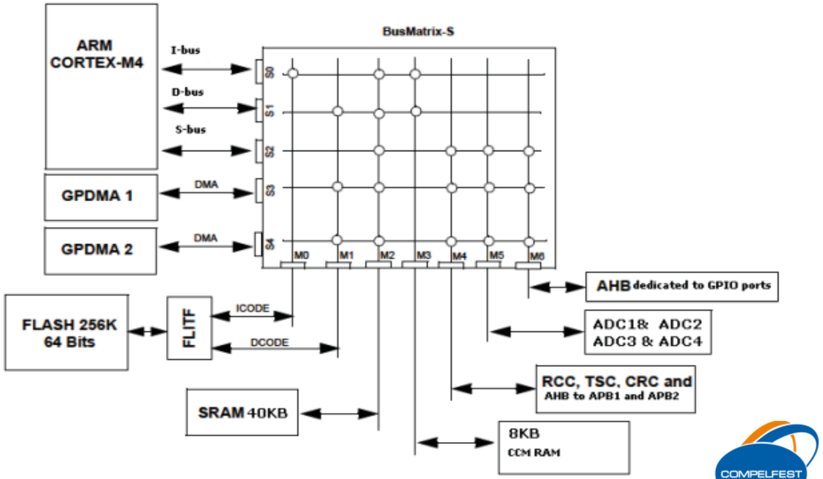
In STM32F37x

- Five masters:
  - Cortex-M4F core I-bus
  - Cortex-M4F core D-bus
  - Cortex-M4F core S-bus
  - GP-DMA1 and GP-DMA2 (general-purpose DMAs)
- Five slaves:
  - Internal SRAM
  - Internal Flash memory
  - AHB to APBx (APB1 or APB2), which connect all the APB peripherals
  - AHB dedicated to GPIO ports



In STM32F30x

- Five masters:
  - Cortex-M4 core I-bus
  - Cortex-M4 core D-bus
  - Cortex-M4 core S-bus
  - GP-DMA1 and GP-DMA2 (general-purpose DMAs)
- Seven slaves:
  - Internal SRAM on the Dcode
  - Internal SRAM on the ICode (CCM RAM)
  - Internal Flash memory
  - AHB to APBx (APB1 or APB2), which connec all the APB peripherals
  - AHB dedicated to GPIO ports
  - ADCs 1,2,3 and 4.

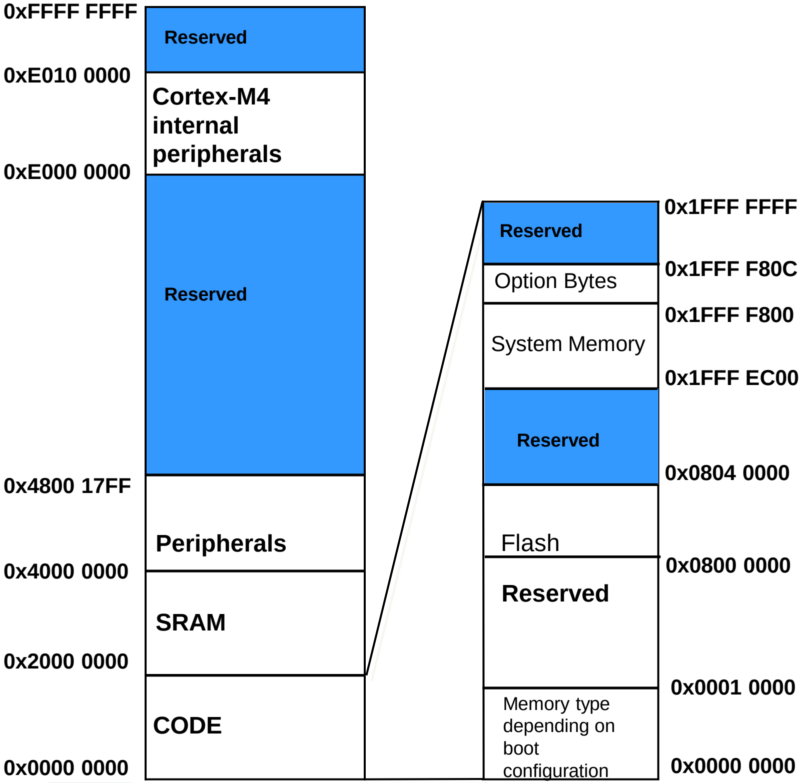


Memory Mapping and Boot Modes

- Addressable memory space of 4 Gbytes
- FLASH : up to 256 Kbytes
- RAM:
  - Up to 40 (F30x) and 32 (F37x) Kbytes SRAM with HW parity check
  - Up to 8 Kbytes CCM RAM with HW parity check (STM32F30xonly)
  - 4 bits per word for parity check

- **Boot modes**
  - Depending on the Boot configuration, Embedded Flash memory, System memory or Embedded SRAM memory is aliased at @0x00 thanks to memory remapping bits in SYSCFG registers.
  - Even when aliased, these memories are still accessible from their original memory space.

- The boot configuration is defined with **BOOT0 pin** and **BOOT1 bit** in USER Option Byte.



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

- **System memory** : contains the Bootloader used to re-program the FLASH through USART or USB .

- **Boot from SRAM :** In the application initialization code you have to Relocate the Vector Table in SRAM using the NVIC Exception Table and Offset register.

Embedded Flash Memory

life.augmented

COMPELFEST

Flash Features Overview

- Flash general features:
  - Up to 256 KBytes
  - 128 pages of 2KBytes size
  - Access time: 35ns
  - Half word (16-bit) program time: 52.5µs (Typ)
  - Page erase time and Mass erase time: 20ms (Min), 40ms (Max)
- Flash interface features:
  - Read Interface with pre-fetch buffer
  - Option Bytes loader
  - Flash program/erase operations
  - Types of Protection:
    - Readout protection: Level 0, Level 1 and Level 2
    - Write Protection

Flash Memory Organization

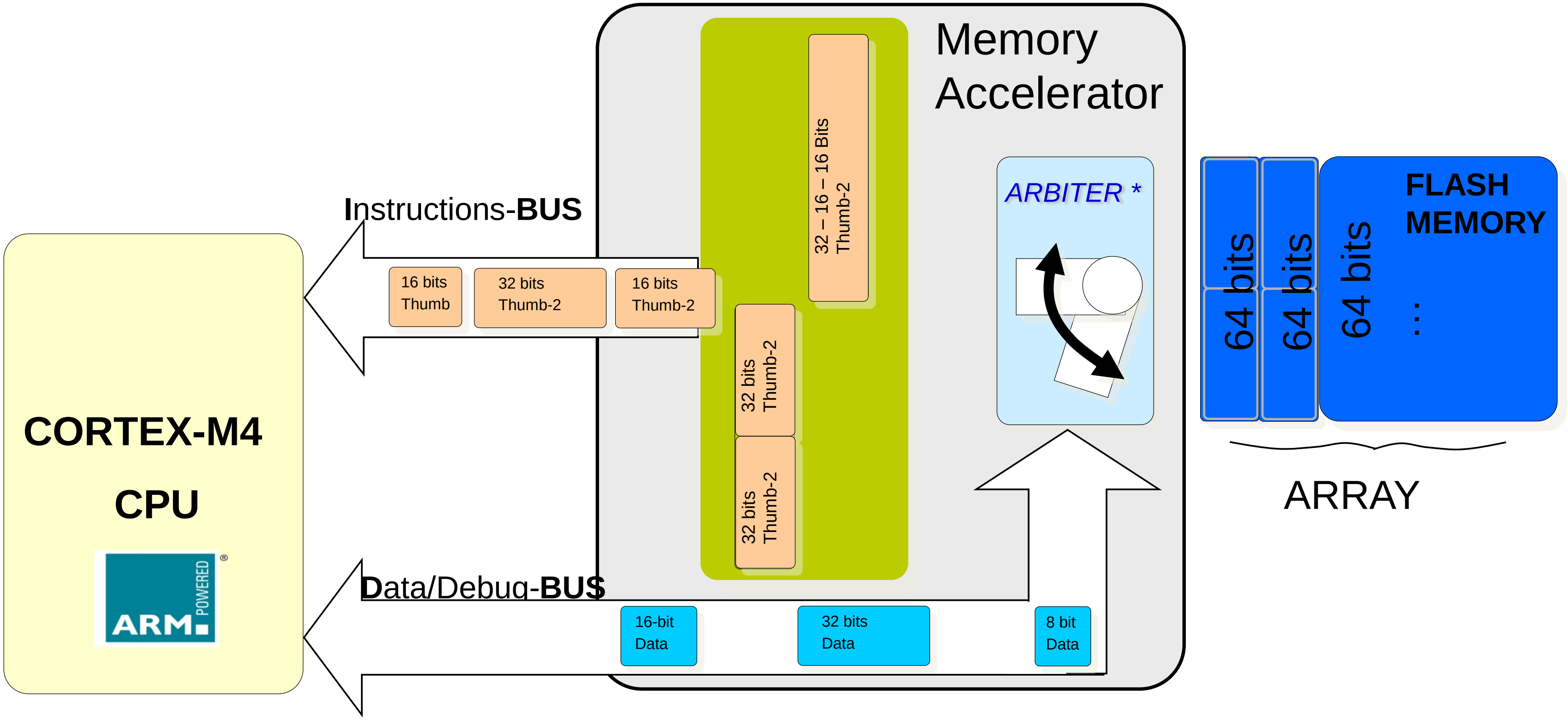
- Main memory block containing 128 pages of 2Kbyte each.

- Information block contains the system memory and option bytes, is divided into two parts:
  - System Memory
    - 8 KB size
    - contains the bootloader which is used to reprogram the Flash memory through USART1, USART2 or USB.
    - used to boot the device in System memory boot mode.
    - programmed by ST when the device is manufactured, and protected against unwanted write/erase operations.
  - 8 Option bytes : Can be read from the memory location starting from 0x1FFFF800 or from the Option byte register (FLASH_OBR) in the Flash memory interface register area.
    - 4 for write protection
    - 1 for read protection
    - 1 for device configuration:
      - IWDG HW/SW mode
      - Reset when entering STANDBY mode
      - Reset when entering STOP mode
      - VDDA supervisor
      - BOOT1
      - SRAM parity check
    - 2 For User Data (To store Security IDs, etc.)

Flash memory prefetch controller

- Mission: Support 72 MHz operation directly from Flash memory

- 64-bits wide Flash with Prefetch (2 x 64bits buffers).



Flash protections (1/6)

- Two kinds of protections are available:
  - **Write protection** to avoid unwanted writings
  - **Readout protection** to avoid piracy: Level 0, Level 1 and Level 2 (No debug)
  - Both are activated by setting option bytes

- **Write protection**
  - The write protection is implemented with a choice of protecting 2 pages (4K) at a time
  - **4 options bytes** are used to protect all the 256KBytes main Flash program memory
  - Any programming or erase of a protected page is discarded and the Flash will return protection error flag in the FLASH_SR status register
  - **Un-protection**
    - Erase the corresponding bit on WRPx option bytes, x = 0..3.
    - Reset the device (POR Reset) or set the FORCE_OPTLOAD bit to re-load the options bytes for disabling any write protection.
  - The write protection bit values are visible also through FLASH_WRPR write protection register.

Flash Protections (2/6)

- **Read protection**
  - The read protection is activated by setting the **RDP option byte** and then, by applying POR reset or using FORCE_OPTLOAD bit from FLASH_CR register to reload the new RDP option byte.
  - Three levels of protection from no protection **(Level 0)** to maximum protection **(Level 2 or No debug)**

RDP byte value	RDP complement value	Read protection level
0xAA	0x55	Level 0
Any value but 0xAA or 0xCC	Any value (not necessarily complement) but 0x55 and 0x33	Level 1
0xCC	0x33	Level 2 (No debug)

- **Readout protection Level 0**
  - No read protection
    - All operations (if no write protection is set) from/to the Flash, option byte or the RTC Backup registers are possible in all boot configurations (Flash user boot, boot RAM, boot loader or debug).

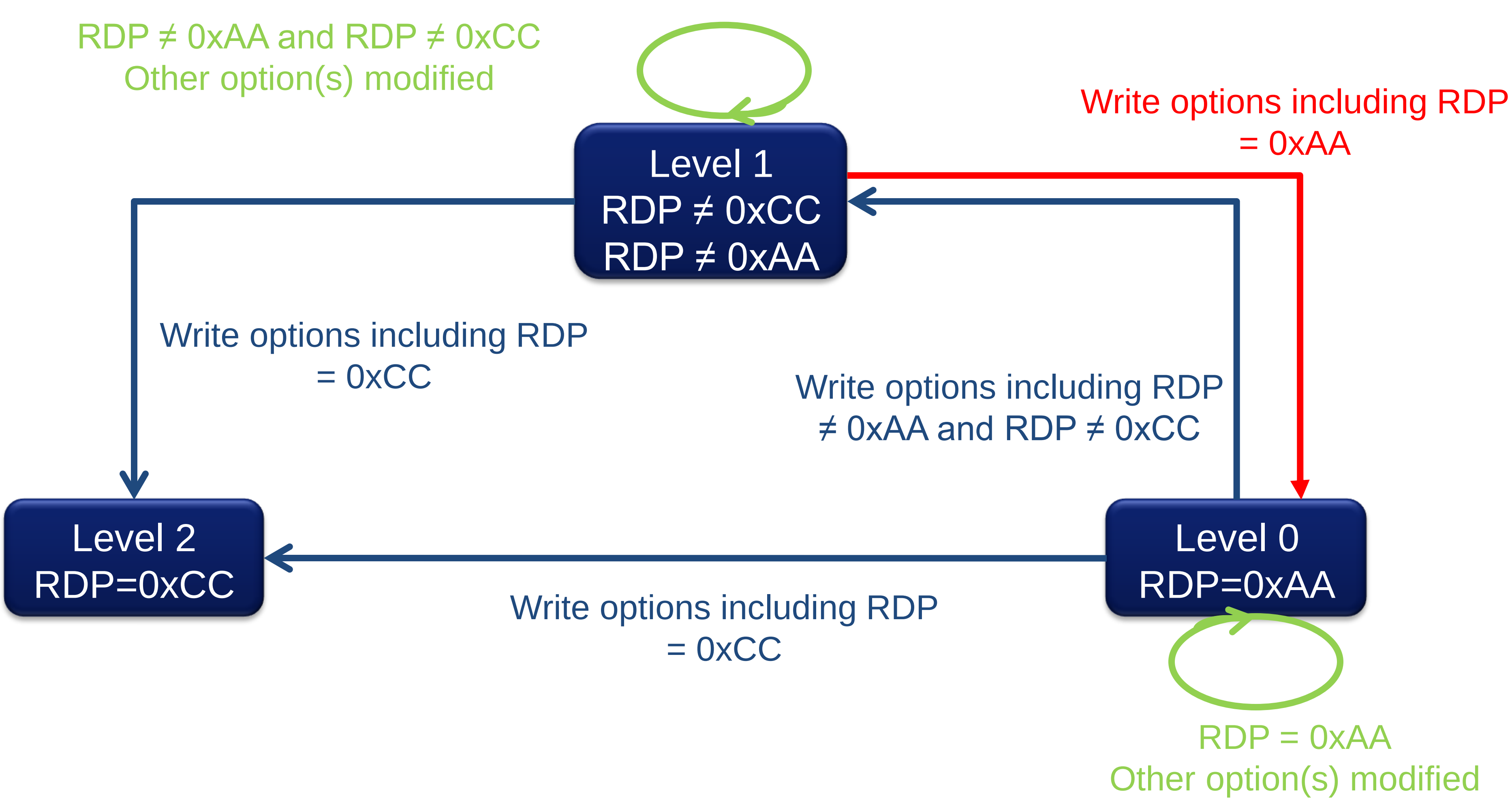
Flash Protections (3/6)

- **Readout protection Level 1**
  - When this protection is enabled :
    - **User mode:** Code executing in user mode can access main Flash memory and option bytes with all operations.
    - **Debug, boot RAM and boot loader modes:** The main Flash memory and backup registers (RTC_BKPxR in RTC) are totally inaccessible in these modes, a simple read access generates a bus error and a Hard Fault interrupt. Any attempted program/erase operations sets the PGERR flag.
  - **Un-protection:**
    - When the RPD is reprogrammed to the value 0xAA to move back to Level 0, a Mass erase of the main Flash memory is performed and the backup registers (RTC_BKPxR in RTC) are reset.

Flash Protections (4/6)

- **Readout protection Level 2 (No debug)**
  - When This protection is enabled :
    - All protections provided by **Level 1** are **active**.
    - **Boot from RAM, boot from system memory and all debug features (serial-wire) are disabled.**
    - Option bytes can no longer be changed except in user mode but not totally ; RDP option byte cannot be programmed/erased and other option bytes can only be programmed (not erased).
  - **Un-protection:**
    - Not possible :level 2 cannot be removed at all: it is an **irreversible operation.**

Flash Protections (5/6)



RDP ≠ 0xAA and RDP ≠ 0xCC
Other option(s) modified

Level 1
RDP ≠ 0xCC
RDP ≠ 0xAA

Write options including RDP
= 0xAA

Write options including RDP
= 0xCC

Write options including RDP
≠ 0xAA and RDP ≠ 0xCC

Level 2
RDP=0xCC

Level 0
RDP=0xAA

Write options including RDP
= 0xCC

RDP = 0xAA
Other option(s) modified

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

Flash Protections (6/6)

- **Access status versus protection level and execution modes :**

Area	Protection level	User execution			Debug, boot from RAM or boot from system memory (loader)		
		Read	Write	Erase	Read	Write	Erase
Main memory	1	Yes	Yes	Yes	No	No	No
	2	Yes	Yes	Yes	N/A	N/A	N/A
System memory	1	Yes	No	No	Yes	No	No
	2	Yes	No	No	N/A	N/A	N/A
Option bytes	1	Yes	Yes	Yes	Yes	Yes	Yes
	2	Yes	Yes	No	N/A	N/A	N/A
Backup registers	1	Yes	Yes	N/A	No	No	N/A
	2	Yes	Yes	N/A	N/A	N/A	N/A

life.augmented

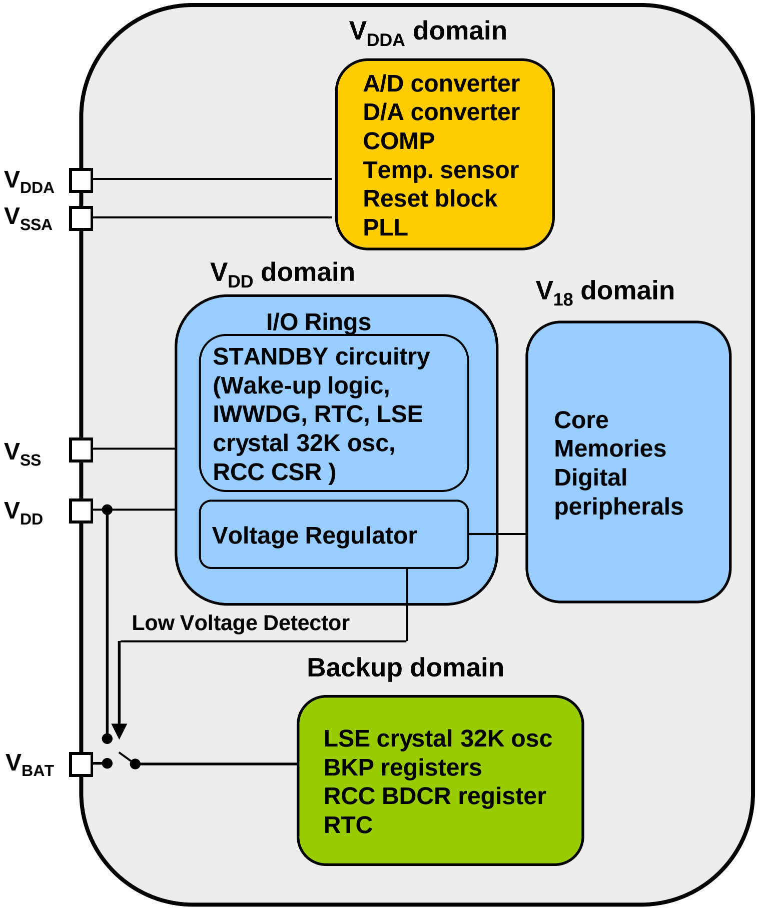
COMPELFEST

Power Control (PWR)

SRM32F30x Power Supply

- **Power Supply Schemes**
  - **VDD = 2.0 to 3.6 V : External Power Supply for I/Os (or VDD = 1.8 +/- 8%: %: External Power Supply for I/Os with internal regulator is OFF.)**
  - **VDDA = 2.0 to 3.6 V: External Analog Power supplies for ADC,DAC, Reset blocks, RCs and PLL.**

    ➔ DAC working only if VDDA >=2.4 V
  - **VBAT = 1.65V to 3.6 V: For Backup domain when VDD is not present.**
  - **Power pins connection:**
    - VDD and VDDA can be provided by a separated power supply source.
    - VSS and VSSA must be tight to ground

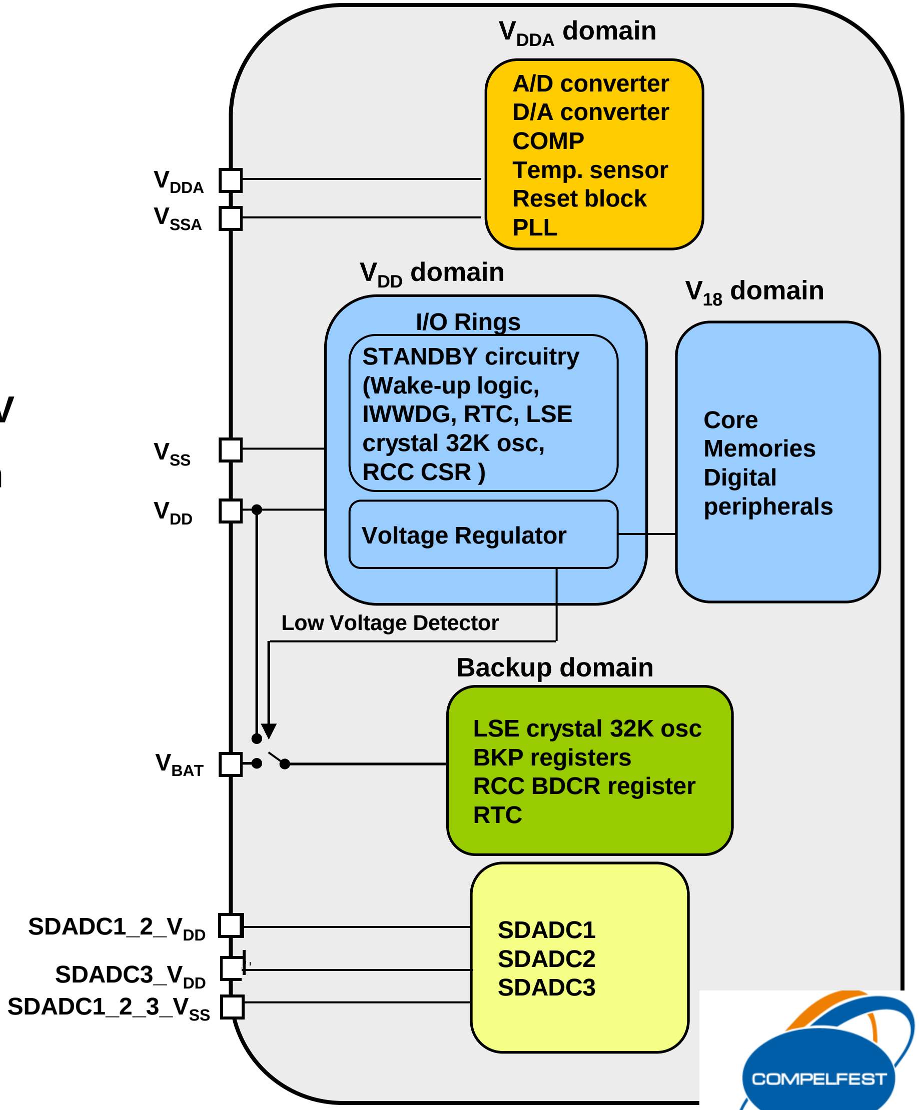


SRM32F37x Power Supply

Power Supply Schemes

- VDD = 2.0 to 3.6 V : External Power Supply for I/Os (or VDD = 1.8 +/- 8%: %: External Power Supply for I/Os with internal regulator is OFF.)
- VDDA = 2.0 to 3.6 V: External Analog Power supplies for ADC,DAC, Reset blocks, RCs and PLL.

  ➔ ADC and DAC working only if VDDA >=2.4 V
- VBAT = 1.65V to 3.6 V: For Backup domain when VDD is not present.
- SDADCx_VDD = 2.2 to 3.6V : External Analog Power supplies for SDADCs with:
- Power pins connection:
  - VDD and VDDA can be provided by a separated power supply source.
  - VSS, VSSA and SDADCx_VSS must be tight to ground
  - The SD1_SD2_VDD and SD3_VDD can be different from VDD , VDDA and from one another.



life.augmented

Power Sequence

- When VDD power supply source is different from VDDA power supply source (VDD < VDDA)
  - The VDDA voltage **level must be always greater or equal** to the VDD voltage
  - During power-on, the VDDA must be provided first (before VDD)
  - During power-off, it is allowed to have temporarily VDD > VDDA, but the voltage difference must be <0.4V
    - could be maintained by an external Schottky diode
- When SDADCx power supply is different from VDDA, VDD power supply and from one another:
  - SDADCx_VDD <= VDDA
  - SDADC1_VDD/SDADC2_VDD <= SDADC3_VDD
  - SDADC3_VDD must start before or at the same time as SD12_VDD

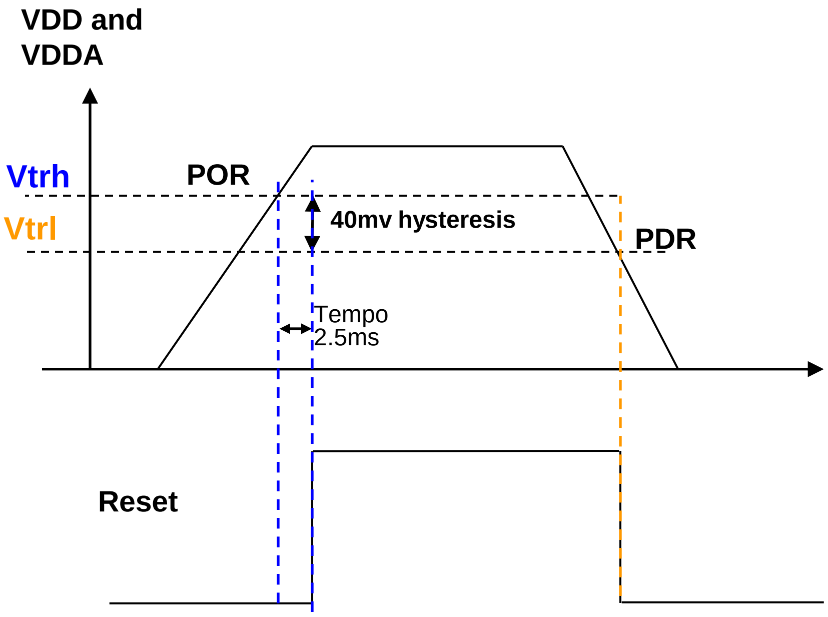
Supply monitoring and Reset circuitry

- The STM32F3xx POR / PDR circuitries are always active and monitor two supply voltages: VDD and VDDA.
- The POR supervisor circuit monitors only **VDD**
- The PDR supervisor circuit monitors **VDD** and **VDDA**
  - The PDR supervisor on VDDA can be disabled by programming Option byte.
- Programmable Voltage Detection (PVD) - Can be ON/OFF.
- The PVD enable/disable is controlled by software via a dedicated bit (PVDE).

Power On Reset / Power Down Reset

- Two Integrated POR / PDR circuitries guarantees proper product reset when voltage is not in the product guaranteed voltage range (2V to 3.6V)
  - **No need for external reset circuit**

- POR and PDR have a typical hysteresis of 40mV

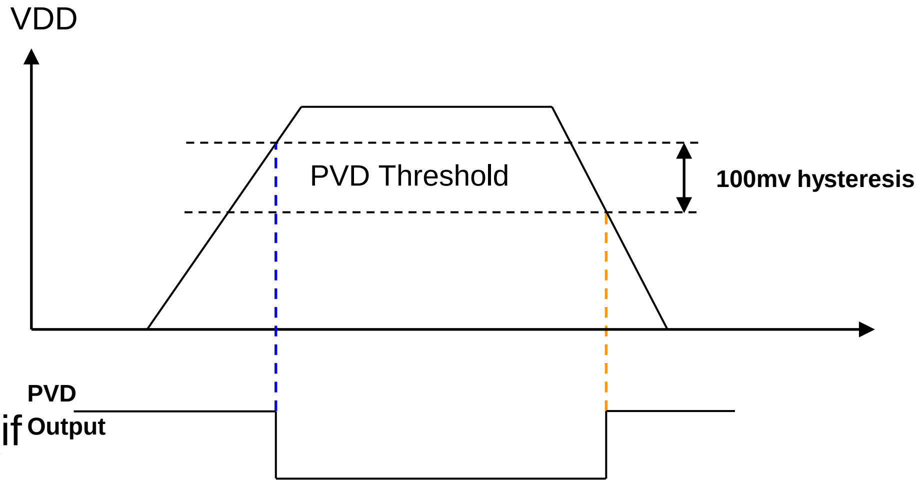


Vtrl min 1.8V / Vtrh max 2V

- The PDR detector monitors VDD and also VDDA (if kept enabled in the option bytes). The POR detector monitors only VDD.

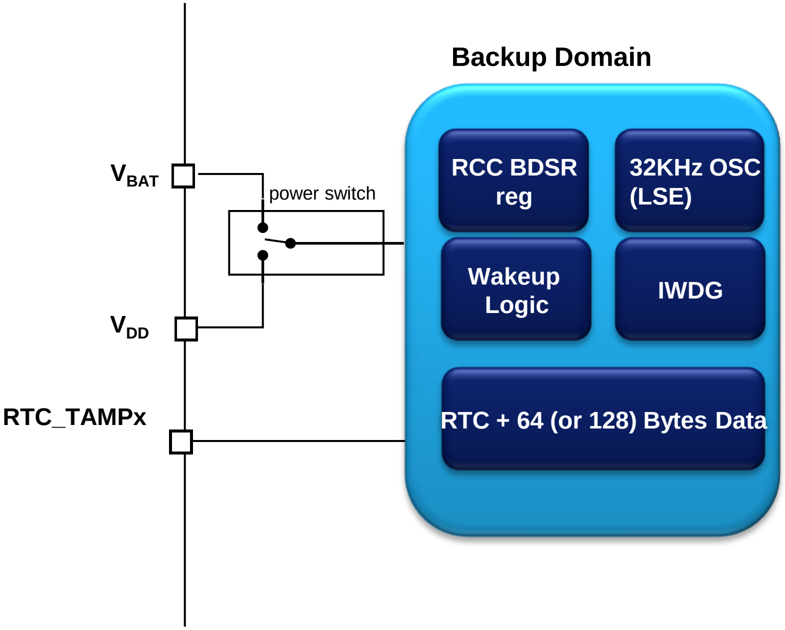
Programmable Voltage Detector (PVD)

- Programmable Voltage Detector
  - Enabled by software
  - Monitors the VDD power supply by comparing it to a threshold
  - Threshold configurable from 2.1V to 2.9V by step of 90mV
  - Generates 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

- Backup Domain contains
  - Low power calendar RTC (Alarm, periodic wakeup from Stop/Standby)
  - 64 and 128 Bytes Data RTC registers in STM32F30x and STM32F37x respectively.
  - Separate 32kHz Osc (LSE) for RTC
  - RCC BDSR register: RTC clock source selection and enable + LSE config
  - ➔ Reset only by RTC domain RESET
- VBAT independent voltage supply
  - Automatic switch-over to VBAT when VDD goes lower than PDR level
  - No current sunk on VBAT when VDD present.
- 2 x Tamper events detection: resets all user backup registers
- TimeStamp event detection.



Low Power Modes (1/4)

- **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 (2/4)

- **STOP Mode**: all peripherals' clocks, PLL, HSI and HSE are disabled, SRAM and registers 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
  - 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 output, RTC alarm, **COMPx, I2Cx, USARTx or the CEC (*).**
    - The **I2Cx**, **USARTx, CEC (*)** can be configured to enable the HSI RC oscillator for processing incoming data. If this is used, the voltage regulator should not be put in the low-power mode but kept in normal mode.

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

(*): CEC is available in STM32F37x only.

Low Power Modes (3/4)

- **STANDBY Mode**: Voltage Regulator off, the entire V18 domain is powered off.
  - SRAM and register contents are lost except registers in the Backup domain and STANDBY circuitry
  - PLL, the HSI RC and the HSE crystal oscillators are also switched off.
  - RTC and IWDG are kept running in STANDBY (if enabled)
  - In STANDBY mode all IO pins are high impedance and non-active except:
    - Reset pad (still available)
    - RTC pins (if configured)
    - PC14 & PC15 could be forced to output high/low in RTC registers
    - WKUPx pins (if enabled)
  - Wake-up sources:
    - WKUPx pins rising edge
    - RTC alarm and tamper events
    - 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.

STM32F3xx Low Power modes

Mode name	Entry	Wakeup	Effect on 1.8V domain clocks	Effect on VDD domain clocks	Voltage regulator	IO state	Wakeup latency
SLEEP, SLEEP now or SLEEP on-exit	WFI	Any interrupt	CPU CLK OFF no effect on other clocks or analog clock sources	None	ON	All I/O pins keep the same state as in the Run mode	None
	WFE	Wake-up event					
STOP	PDDS, LPSDSR bits + SLEEPDEEP bit + WFI or WFE	Any EXTI line (configured in the EXTI registers, **internal** and **external** lines)	All 1.8V domain clocks OFF	HSI and HSE and oscillators OFF	ON, in low power mode (depending on PWR_CR)		HSI RC wakeup time + regulator wakeup time from Low-power mode
STANDBY	PDDS bit + SLEEPDEEP bit + WFI or WFE	WKUP pin rising edge, RTC alarm, RTC tamper event, external reset in NRST pin, IWDG reset			OFF	all I/O pins are high impedance (*)	Reset phase

(*): Standby mode: all I/O pins are high impedance except:

- Reset pad (still available)
- RTC pins PC14 and PC15 if configured in the RTC registers.
- WKUP pin 1 (PA0) and WKUP pin 2(PC13), if enabled.

Direct memory access controller (DMA)

life.augmented

COMPELFEST

DMA Features

- 12 independently configurable channels: hardware requests or software trigger on each channel.
  - DMA1: 7 Channels
  - DMA2: 5 Channels
- Software programmable priorities: Very high, High, Medium or Low. (Hardware priority in case of equality).
- Programmable and Independent source and destination transfer data size: Byte, Halfword or Word.
- 3 event flags for each channel: DMA Half Transfer, DMA Transfer complete and DMA Transfer Error.
- Memory-to-memory, peripheral-to-memory and memory-to-peripheral transfers and peripheral-to-peripheral transfers.
- Faulty channel is automatically hardware disabled in case of bus access error.
- Programmable number of data to be transferred: up to 65535.
- Support for circular buffer management.

DMA1 Request Mapping (1/2)

Periphera l	Channel 1	Channel 2	Channel 3	Channel 4	Channel 5	Channel 6	Channel 7
ADC	ADC1						
SPI		SPI1_RX	SPI1_TX	SPI2_RX	SPI2_TX		
USART		USART3_TX	USART3_RX	USART1_TX	USART1_RX	USART2_RX	USART2_TX
I2C				I2C2_TX	I2C2_RX	I2C1_TX	I2C1_RX
TIM1 (*)		TIM1_CH1	TIM1_CH2	TIM1_CH4 TIM1_TRIG TIM1_COM	TIM1_UP	TIM1_CH3	
TIM2	TIM2_CH3	TIM2_UP				TIM2_CH1	TIM2_CH2 TIM2_CH4
TIM3		TIM3_CH3	TIM3_CH4 TIM3_UP			TIM3_CH1 TIM3_TRIG	
TIM4	TIM4_CH1			TIM4_CH2	TIM4_CH3		TIM4_UP
TIM6 / DAC (*)			TIM6_UP DAC_CH1 (1)				

- (*) Available on STM32F30x only.
- (1) DMA request mapped on this DMA channel only if the corresponding remapping bit is set in the SYSCFG_CFGR1 register

DMA1 Request Mapping (2/2)

Peripherals	Channel 1	Channel 2	Channel 3	Channel 4	Channel 5	Channel 6	Channel 7
TIM7 / DAC (*)				TIM7_UP DAC_CH2 (1)			
TIM16			TIM16_CH1 TIM16_UP			TIM16_CH1 TIM16_UP (*) (1)	
TIM17	TIM17_CH1 TIM17_UP						TIM17_CH1 TIM17_UP (*) (1)
TIM18 / DAC channel 3 (**)					TIM18_UP DAC_CH3		
TIM19 (**)	TIM19_CH3 TIM19_CH4	TIM19_CH1	TIM19_CH2	TIM19_UP			

- (*) Available on STM32F30x only
- (**) Available on STM32F37x only.
- (1) DMA request mapped on this DMA channel only if the corresponding remapping bit is set in the SYSCFG_CFGR1 register

DMA2 Request Mapping

Peripherals	Channel1	Channel2	Channel3	Channel4	Channel5
ADC	ADC2	ADC4	ADC2 (1) SDADC1	ADC4 (1) SDADC2	ADC3 SDADC3
SPI3	SPI3_RX	SPI3_TX			
UART4(*)			UART4_RX		UART4_TX
TIM6 / DAC channel 1			TIM6_UP DAC_CH1		
TIM7 / DAC channel 2				TIM7_UP DAC_CH2	
TIM8 / DAC (**)	TIM8_CH3 TIM8_UP	TIM8_CH4 TIM8_TRIG TIM8_COM	TIM8_CH1		TIM8_CH2
TIM18 / DAC channel 3 (**)					TIM18_UP DAC_CH3

- (*) Available on STM32F30x only
- (**) Available on STM32F37x only.

General Purpose IOs

GPIO features

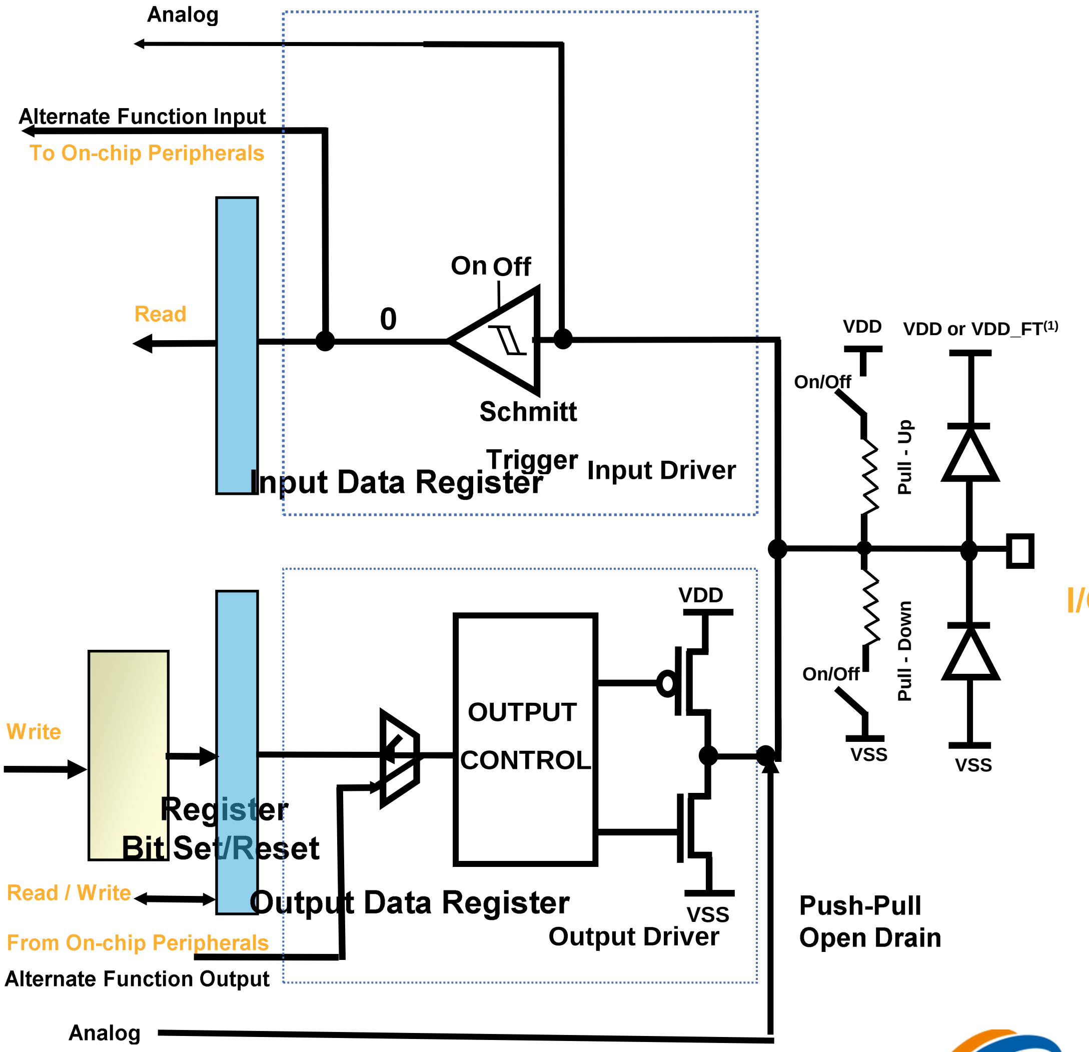
- Up to 84 (in STM32F37x) and 88 (in STM32F30x) multifunction bi-directional I/O ports available on biggest package 100 pin.
- Several I/Os are 5V tolerant (ADC, opamp, comparators pins are not).
- All Standard I/Os are shared in 6 ports: GPIOA, GPIOB, GPIOC, GPIOD, GPIOE, GPIOF.
- Atomic Bit Set and Bit Reset using BSRR and BRR registers
- GPIO connected to AHB bus, max toggling frequency 18 MHz
- Configurable Output slew rate speed up to 50MHz
- Locking mechanism (GPIOx_LCKR) provided to freeze the I/O configuration
  - When the LOCK sequence has been applied on a port bit, it is no longer possible to modify the configuration of the port bit until the next reset (no write access to the CRL and CRH registers corresponding bit).
- Up to 84 (in STM32F37x) and 88 (in STM32F30x) GPIOs can be set-up as external interrupt (up to 16 lines at time) able to wake-up the MCU from low power modes.

life.augmented

GPIO Configuration Modes

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

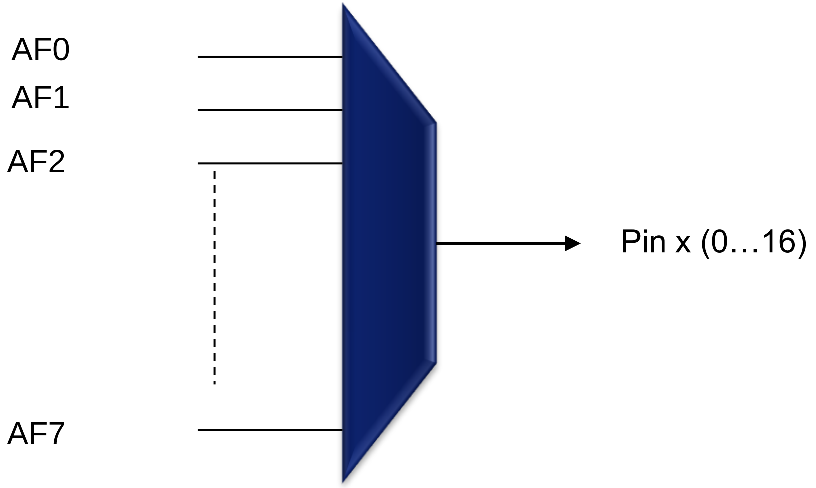
* In output mode, the I/O speed is configurable through OSPEEDR register: 2MHz, 10MHz or 50MHz



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

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.

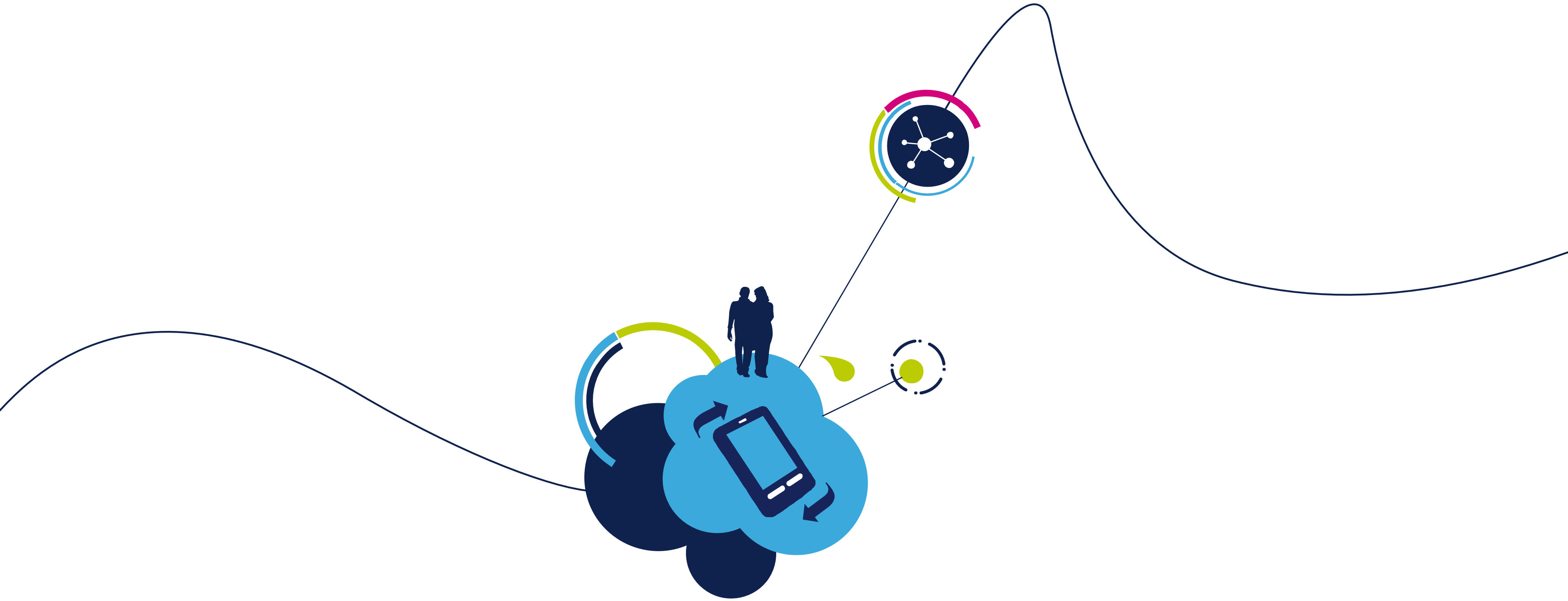


I/Os special considerations

- During and just after reset, the alternate functions are not active and the I/O ports are configured in input floating mode. But, the debug pins (JTAG/SWD) are in AF pull-up/pull-down after reset:
  - PA13: JTMS/SWDIO in pull-up
  - PA14: JTCK/SWCLK in pull-down
  - PA15: JTDI
  - PB3: JTDO
  - PB4: NJTRST

- Using the HSE or LSE oscillator pins as GPIOs
  - When the HSE or LSE oscillator is switched OFF (default state after reset), the related oscillator pins can be used as general purpose IOs.
  - When the oscillator is configured in a user external clock mode, only the OSC_IN or OSC32_IN pin is reserved for clock input and the OSC_OUT or OSC32_OUT pin can still be used as general purpose IOs.

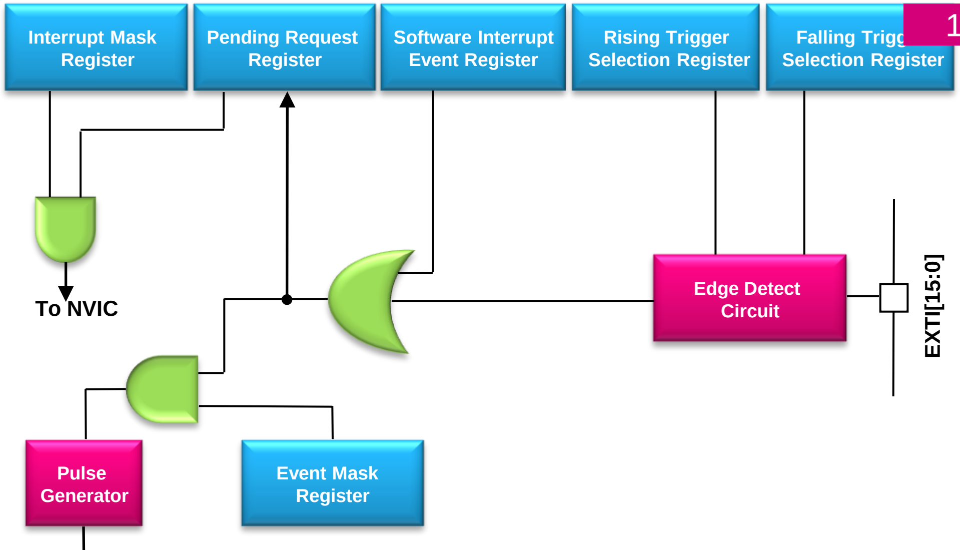
- Using the GPIO pins in the backup supply domain
  - The PC13/PC14/PC15 GPIO functionality is lost when the device enters Standby mode. In this case, if their GPIO configuration is not bypassed by the RTC configuration, these pins are set in an analog input mode.



Extended interrupts and events controller (EXTI)

EXTI Features

- Manages the external and **internal asynchronous** events/interrupts and generates the event request to the CPU/Interrupt Controller and a wake-up request to the Power Manager
- Some communication peripherals (UART, I2C, CEC (*), comparators) are able to generate events when the system is in run/sleep mode and also when the **system is in stop mode** allowing to wake up the system from stop mode.
- These peripherals are able to generate both a synchronized (to the system APB clock) and an asynchronous version of the event.
- All others features are same as STM32F1xx series
- Up to 36 (F30x) 29(F37x) Interrupt/Events requests : Up to 88 (in STM32F30x) and 84 (in STM32F37x) GPIOs can be used as EXTI line(0..15)
- (*) The CEC is available on STM32F37x only.



EXTI line 16 is connected to the PVD output
EXTI line 17 is connected to the RTC Alarm event
EXTI line 18 is connected to USB Device FS wakeup event
EXTI line 19 is connected to RTC tamper and Timestamps
EXTI line 20 is connected to RTC wakeup
EXTI line 21 is connected to Comparator 1 output
EXTI line 22 is connected to Comparator 2 output
EXTI line 23 is connected to I2C1 wakeup
EXTI line 24 is connected to I2C2 wakeup
EXTI line 25 is connected to USART1 wakeup
EXTI line 26 is connected to USART2 wakeup.
EXTI line 27 is connected to CEC wakeup. (STM32F37x only)
EXTI line 28 is connected to USART3 wakeup
EXTI line 29 is connected to Comparator 3 output (F30x only)
EXTI line 30 is connected to Comparator 4 output (F30x only)
EXTI line 31 is connected to Comparator 5 output. (F30x only)
EXTI line 32 is connected to I Comparator 6 output (F30x only)
EXTI line 33 is connected to Comparator 7 output (F30x only)
EXTI line 34 is connected to UART4 wakeup. (F30x only)
EXTI line 35 is connected to UART5 wakeup. (F30x only)

life.augmented

COMPELFEST

Reset and Clock control RCC

RCC introduction

- Reset:
  - Initialize the device
  - Wakeup device
  - Safety functions (watchdog)

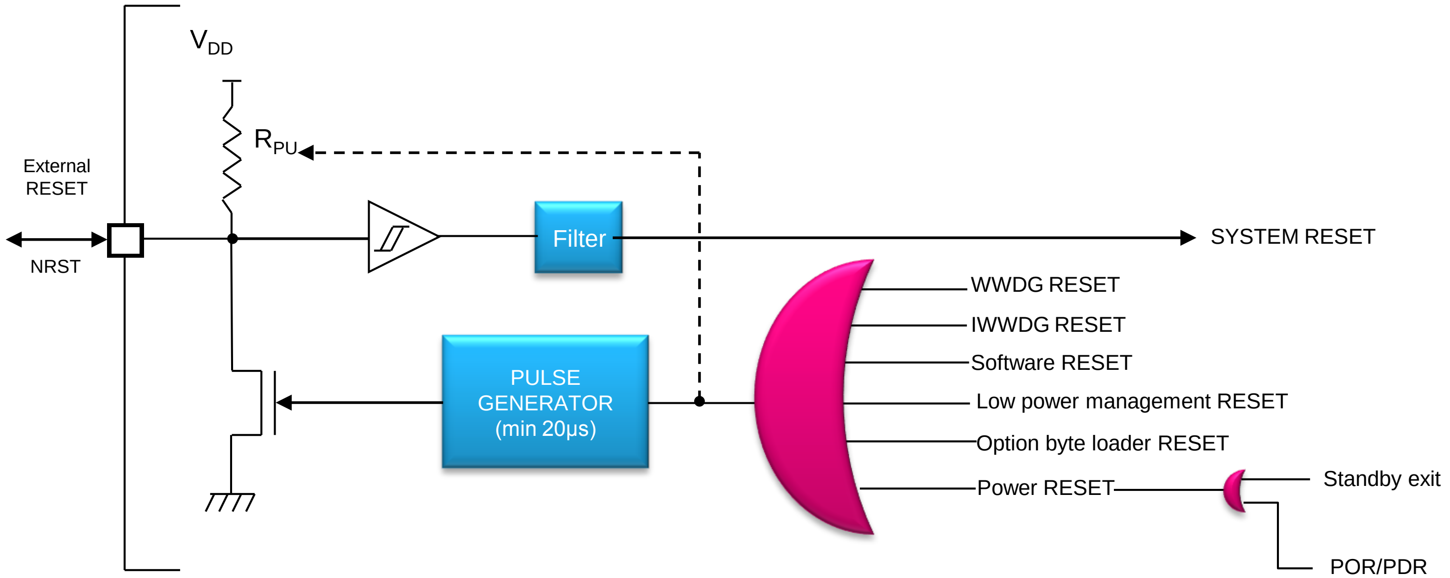
- Clocks:
  - Select appropriate clock source:
    - Internal
    - External
  - Select appropriate speed:
    - High speed
    - Low speed
    - Speed regulation
  - Modify clock parameters for:
    - Core
    - Peripherals
  - Security functions:
    - In case of clock source malfunction

Reset sources

- System RESET
  - Resets all registers except some RCC registers and Backup domain
  - Sources:
    - Low level on the NRST pin (External Reset)
    - WWDG & IWWDG end of count condition
    - A software reset (through NVIC)
    - Low power management reset
    - Option byte loader reset (FORCE_OBL bit)
- Power RESET
  - Resets all registers except the Backup domain
  - Sources:
    - Power On/Power down Reset (POR/PDR)
    - 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

Reset block diagram

- Reset sources in STM32F3 family and their relation to RESET pin:



Clock features (1/2)

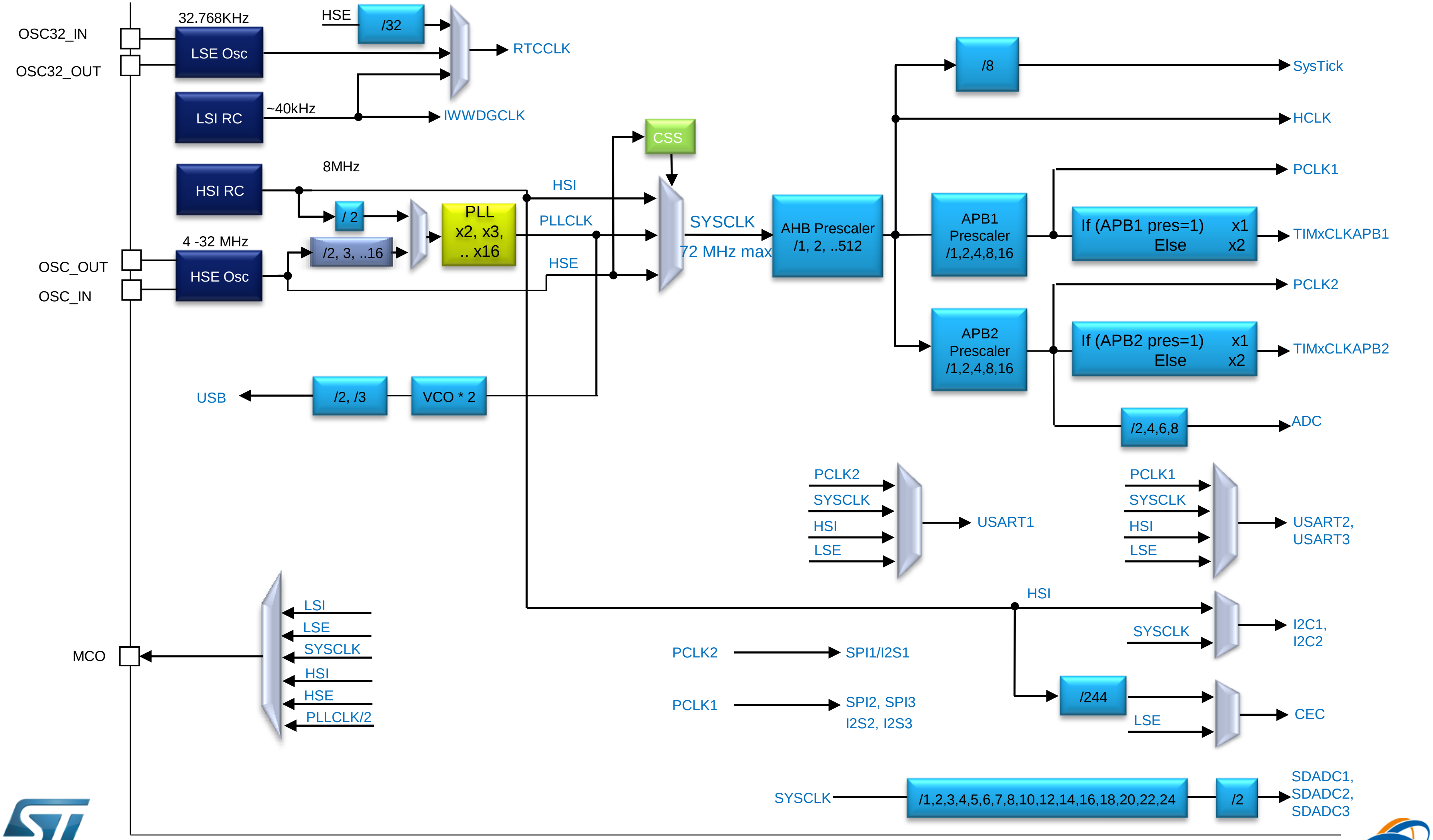
- System Clock (SYSCLK) sources:
  - HSE (High Speed External oscillator or crystal)
    - 4MHz to 32MHz,
    - can be bypassed by user clock
  - HSI (High Speed Internal RC):
    - factory trimmed internal RC oscillator 8MHz +/- 1%
  - PLL x2, x3, .. x16
    - From HSE or HSI/2
    - 16MHz – 72MHz output

- Additional clock sources:
  - LSI (Low Speed Internal RC):
    - ~40kHz internal RC
  - LSE (Low Speed External oscillator):
    - 32.768kHz
    - can be bypassed by user clock
    - Configurable driving strength (power/robustness compromise)

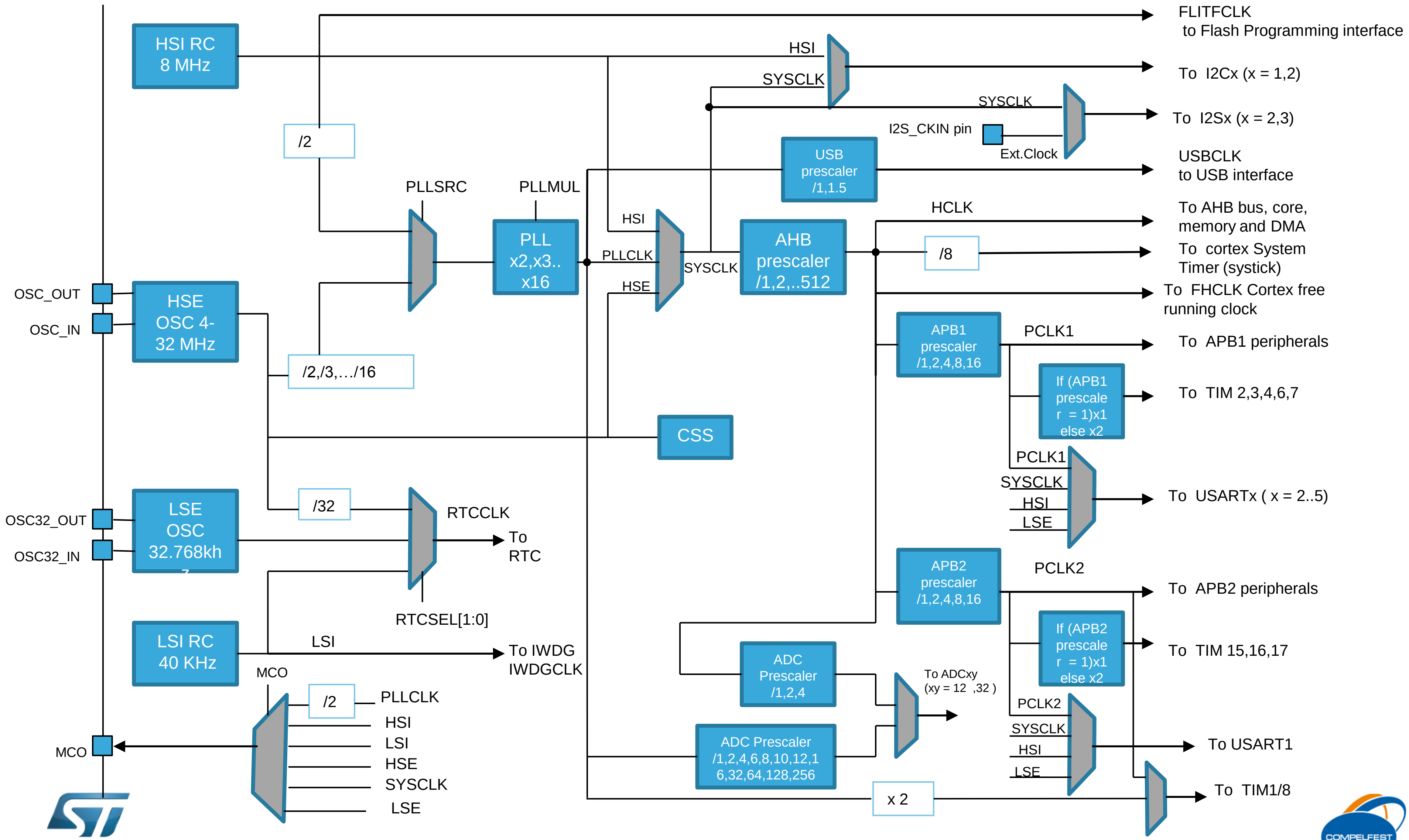
Clock features (2/2)

- Clock-out capability on the MCO:
  - LSI, LSE, SYSCLK, HSI, HSE, PLL/2
- Clock Security System (CSS) to switch to backup clock:
  - In case of HSE clock failure
  - Enabled by SW w/ interrupt capability linked to NMI
  - Could generate BREAK for Timers
- RTC Clock sources:
  - LSE, LSI and HSE/32
- USART, I2C & CEC have multiple possible clock sources:
  - Possibility to wakeup device if there is no system clock:
    - For USART: HSI, LSE
    - For I2C: HSI
    - For HDMI-CEC: LSE, HSI

Clock scheme STM32F37x

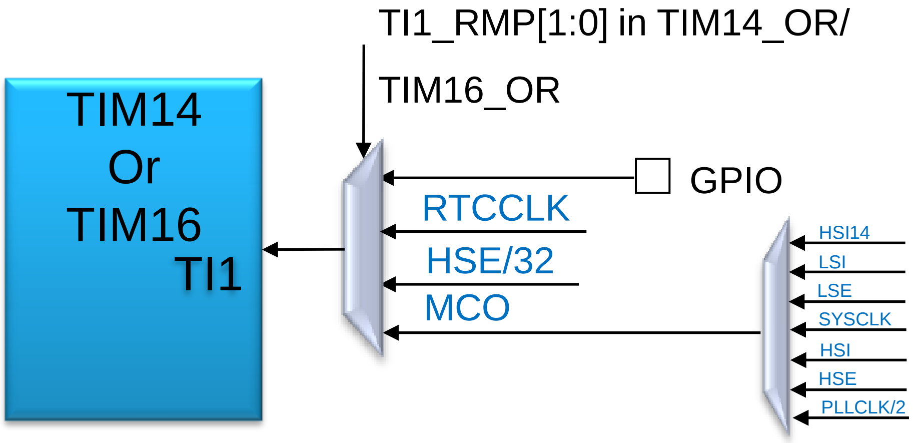


Clock scheme STM32F30x



HSI/LSI/ext. clock measurement

- TIM14 (in F37x) and TIM16 (in F30x) input capture can be triggered by:
  - GPIO pin
  - RTCCLK
  - HSE/32
  - MCO output



- Purposes:
  - **Measure HSI frequency** using the precise LSE clock. HSI is used as system clock. Knowing the (more precise) LSE frequency we can determine the HSI frequency.
  - **Measure the LSI frequency** using HSE or HSI. To fine tune IWWDG and/or RTC timing (if LSI used as RTC clock).
  - Have rough indication of the **frequency of external crystal** – by comparing HSI and HSE/32

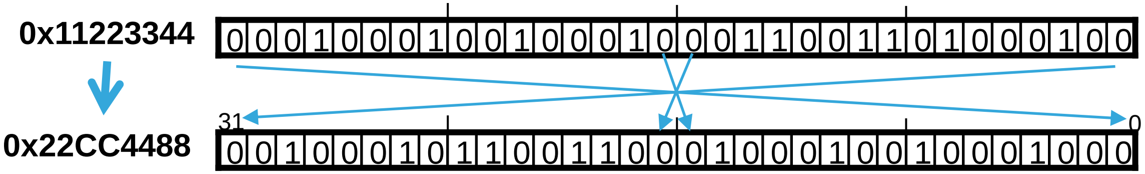
CRC calculation unit

CRC Introduction 1/2

- CRC-based techniques are used to verify data integrity (communications)
- In functional safety standards (such as EN/IEC 60335-1), CRC peripheral offers a means of verifying the embedded Flash memory integrity
- Single input/output 32-bit data register, but handles 8,16, 32-bits input data size
- CRC computation done in 4 AHB clock cycles (HCLK) maximum
- General-purpose 8-bit register (can be used for temporary storage)

CRC Introduction 2/2

- New features:
  - Programmable parameters:
    - Programmable polynomial:
      - By default uses CRC-32 (Ethernet) polynomial: 0x4C11DB7
      - Alternatively uses a fully programmable polynomial with programmable size (7, 8, 16, 32 bit)
    - Programmable polynomial size (7, 8, 16, 32 bits)
    - Programmable CRC initial value (default = 0xFFFF_FFFF)
  - Reversibility option on I/O data
    - Input data can be reversed by 8, 16, 32 bit
    - Example if input data 0x1A2B3C4D is used for CRC calculation as:
      - 0x58D43CB2 with bit-reversal done by byte
      - 0xD458B23C with bit-reversal done by half-word
      - 0xB23CD458 with bit-reversal done on the full word
    - Output data can be reversed in 32-bit (output register)
      - Example on output data 0x11223344:



0x11223344

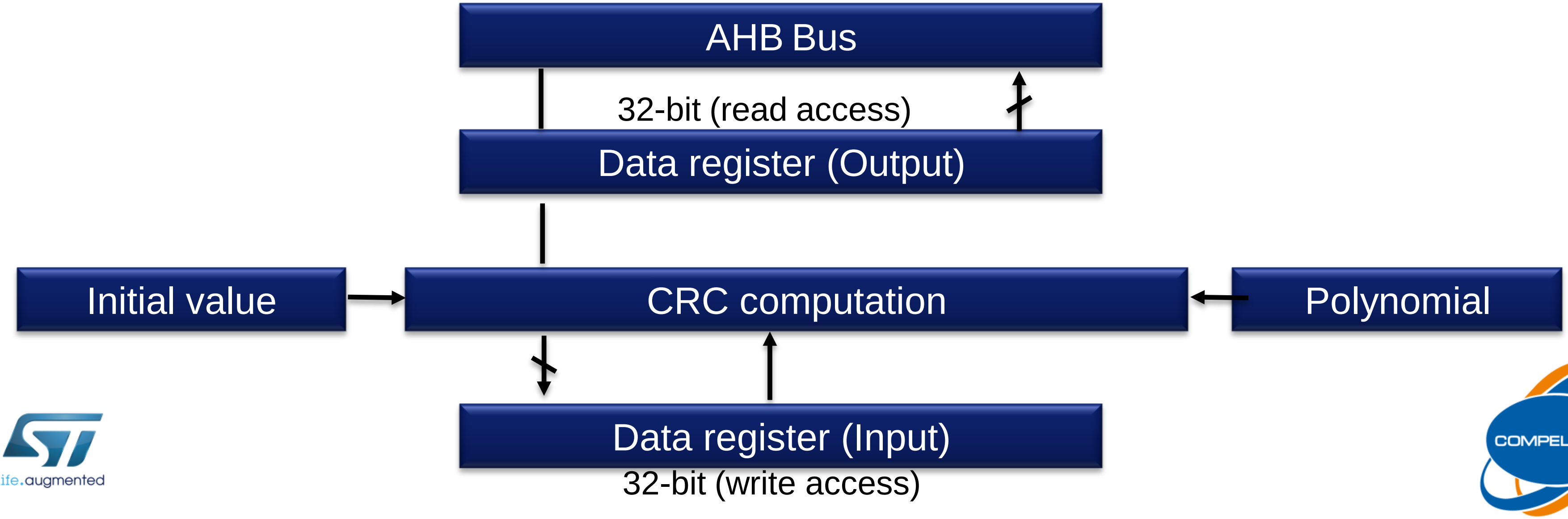
0 0 0 1 0 0 0 1 0 0 1 0 0 0 1 0 0 0 1 1 0 0 1 1 0 1 0 0 0 1 0 0

0x22CC4488

0 0 1 0 0 0 1 0 1 1 0 0 1 1 0 0 0 1 0 0 0 1 0 0 1 0 0 0 1 0 0 0

CRC Operation

- Operation:
  - Each write operation to the data register creates a combination of the previous CRC value (stored in CRC_DR) and the new one. CRC computation is done on the whole 32-bit data word or byte by byte depending on the format of the data being written.
  - The duration of the CRC computation depends on input data width:
    - 4 AHB clock cycles for 32-bit
    - 2 AHB clock cycles for 16-bit
    - 1 AHB clock cycles for 8-bit
  - Polynomial can be changed after finishing current CRC calculation (or after CRC reset)
  - The input and output data can be bit reversed, to manage the various endianness schemes (REV_IN [1:0], REV_OUT bits).



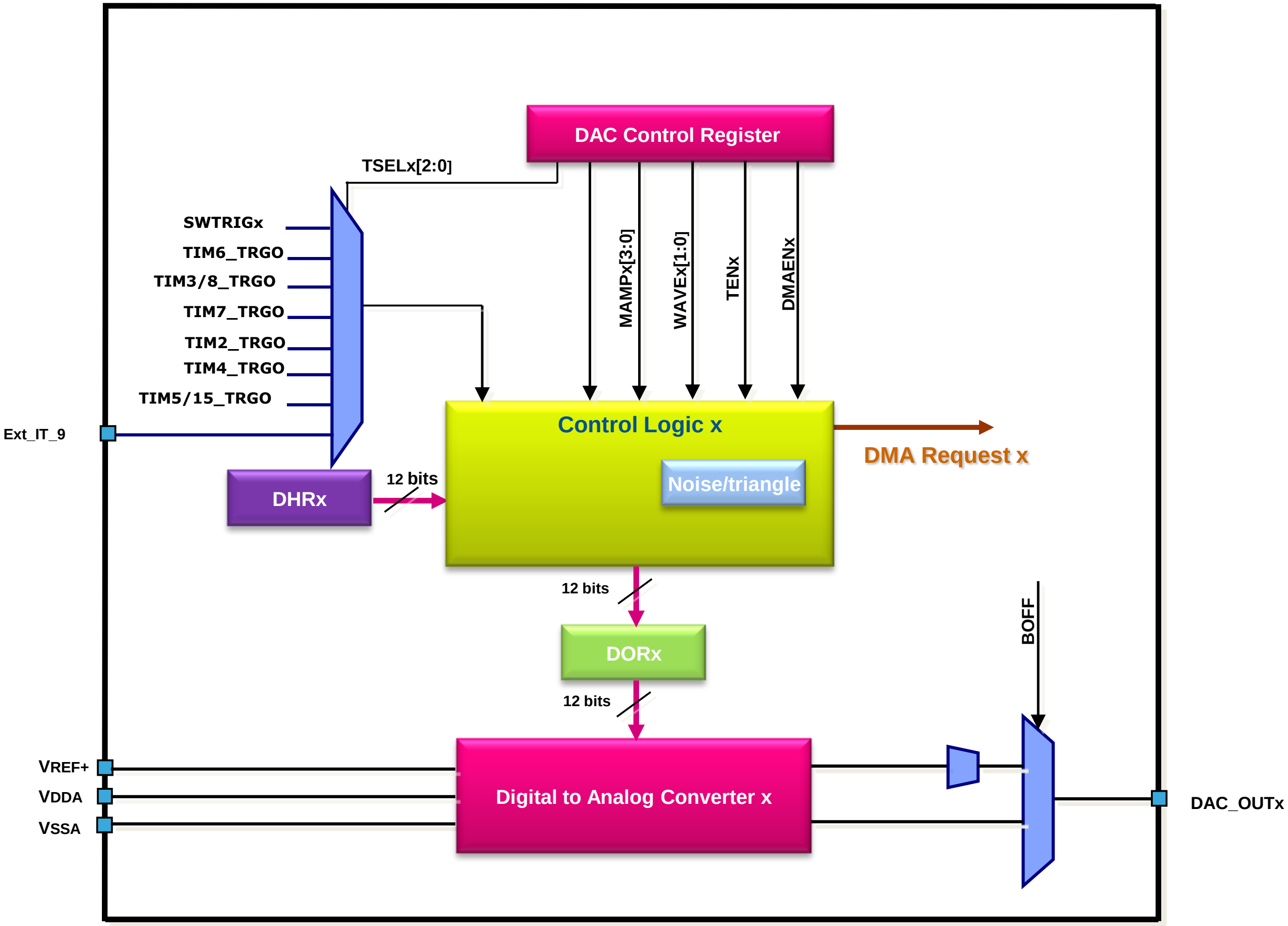
Digital to Analog Converter DAC

DAC introduction

- Interfaces:
  - Two 12-bit DAC converters inside STM32F37x:
    - DAC1 with 2 DAC output channels
    - DAC2 with 1 output channel
  - One 12-bit DAC converter inside STM32F30x:
    - DAC1 with 2 DAC output channels

- Features and differences:
  - 8-bit or 12-bit mode (left or right data alignment in 12-bit mode)
  - Synchronized update capability
  - Noise-wave or Triangular-wave generation
  - DMA capability for each channel (with DMA underrun error detection)
  - External triggers for conversion (Timers, ext. pin, SW trigger)
  - Programmable output buffer to drive more current
  - Input voltage reference VREF+
  - DAC supply requirement: VDDA = 2.4V to 3.6 V
  - DAC outputs range: 0 ≤ DAC_OUTx ≤ VREF+
  - Dual DAC channel mode supported by DAC1 only:
    - Two channels can be used independently or simultaneously when both channels are grouped together for synchronous update operations (dual mode).

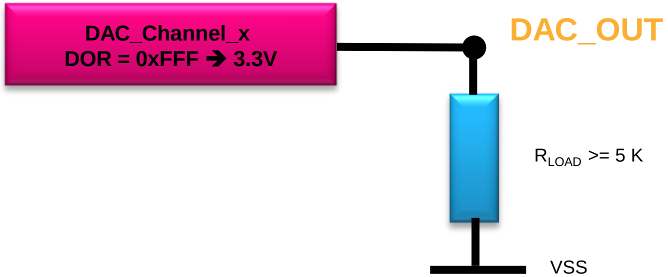
DACx channel block diagram



DAC analog output

- Output voltage:
  - Analog output voltage is given by formula:
    - DAC Output = V_{REF+} * (DOR / 4095)
      - V_{REF+} ……. reference voltage (shared with ADC, input pin or shared with VDDA)
      - DOR ……. Data output register

- Output current:
  - Optional output analog buffer (booster) to improve current capability (BOFF bit)
  - Without output analog buffer (BOFF bit = 1):
    - Rail to rail output: Vout = (V_{REF+} + 1LSB) – (V_{REF+} - 1LSB)
    - Output impedance: 15kΩ
      - Min. load for 1% error: >1.5MΩ
  - With output analog buffer (BOFF bit = 0):
    - Limited output near edges: Vout = (200mV) – (V_{DDA} - 200mV)
    - Min. load for 1LSB error: >5kΩ

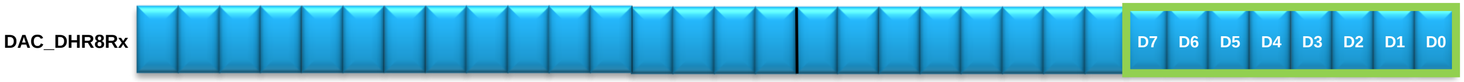


DAC data format

- 8-bit mode:
  - Always right alignment (in register DAC_DHR8Rx)
  - Also in dual channel mode (register DAC_DHR8RD)

- 12-bit mode:
  - Right alignment (in register DAC_DHR12Rx)
  - Left alignment (in register DAC_DHR12Lx)
  - Also in dual channel mode (registers DAC_DHR12RD, DAC_DHR12LD)

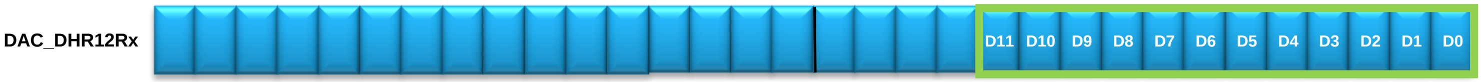
8 bits Right alignment: Load DAC_DHR8Rx [7:0]



DAC_DHR8Rx

D7 D6 D5 D4 D3 D2 D1 D0

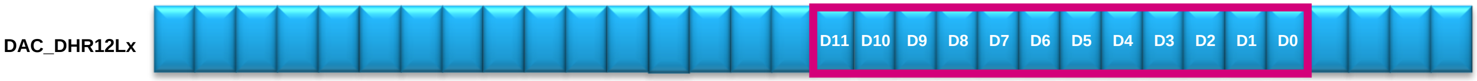
12 bits Right alignment : Load DAC_DHR12Rx [11:0]



DAC_DHR12Rx

D11 D10 D9 D8 D7 D6 D5 D4 D3 D2 D1 D0

12 bits Left alignment : Load DAC_DHR12Lx [15:4]



DAC_DHR12Lx

D11 D10 D9 D8 D7 D6 D5 D4 D3 D2 D1 D0

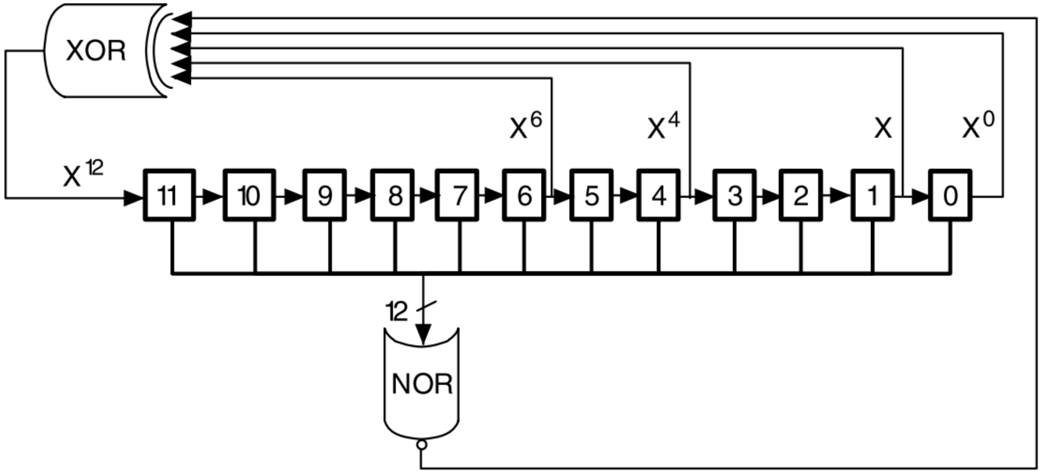
DAC conversion triggers

- Conversion started (load data to the DORx register) by:
  - Automatically (if external trigger disabled TENx) :
    - From DAC_DHRx after one APB1 clock cycle
  - Triggered conversion (if external trigger enabled TENx) :
    - After three APB1 clock cycles after trigger generated by external trigger (except SWTRIG)

- Triggers:
  - Timers:
    - Timer 6 TRGO event
    - Timer 3 TRGO event (or Timer 8 as option for STM32F30x)
    - Timer 7 TRGO event
    - Product dependent :
      - For STM32F37x: Timer 5 (for DAC1) / Timer 18 (for DAC2) TRGO event
      - For STM32F30x: Timer 15
    - Timer 2 TRGO event
    - Timer 4 TRGO event
  - External pin
    - EXTI line9
  - Software
    - SWTRIG bit

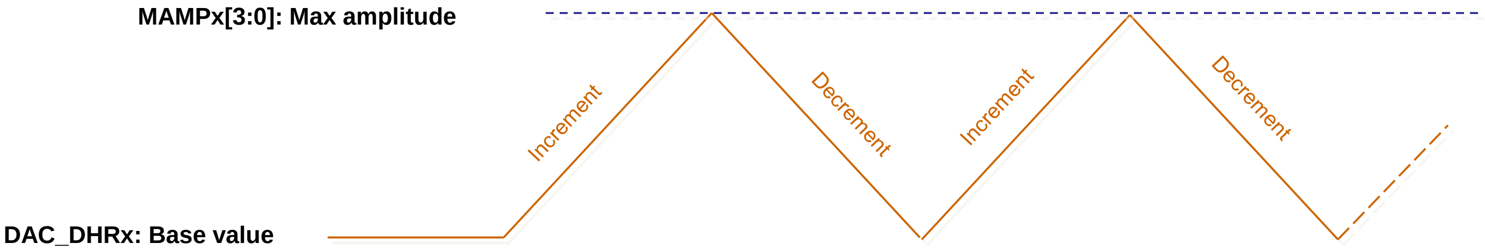
DAC noise wave generation

- Noise generation:
  - Based on LFSR (linear feedback shift register):
    - Initial value = 0xAAAA
  - The LFSR 12bits value can be masked partially or totally
  - Anti lock-up mechanism: if LFSR equal to 0 then a 1 is injected on it
  - Calculated noise value (updated through external trigger) is added to the DAC_DHRx content without overflow



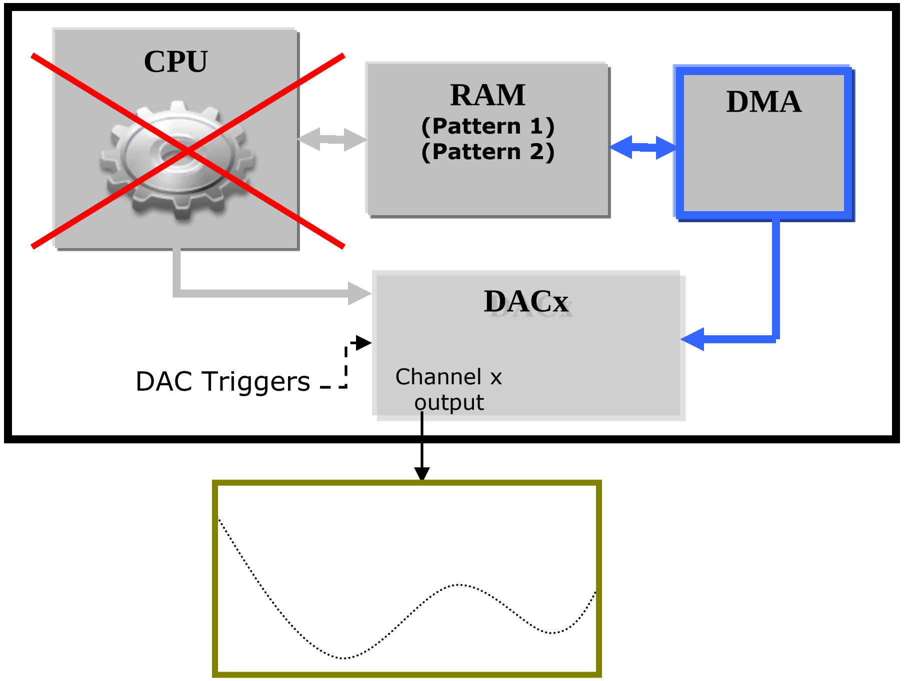
DAC triangle wave generation

- Triangle generation:
  - Add a small-amplitude triangular waveform on a DC or slowly varying signal: used as basic waveform generator for example
  - Calculated triangle value (updated through external trigger) is added to the DAC_DHRx content without overflow to reach the configurable max amplitude
  - Up-Down triangle counter:
    - Incremented to reach defined max amplitude value
    - Decremented to return to the initial base value
  - Triangle max. amplitude are configurable: (2^N–1) with N=[1..12]



DAC with DMA

- A DAC DMA request is generated when an external trigger occurs:
  - The value of the DAC_DHR register is then transferred to the DAC_DOR register.
- DMA underrun error detection with interrupt capability



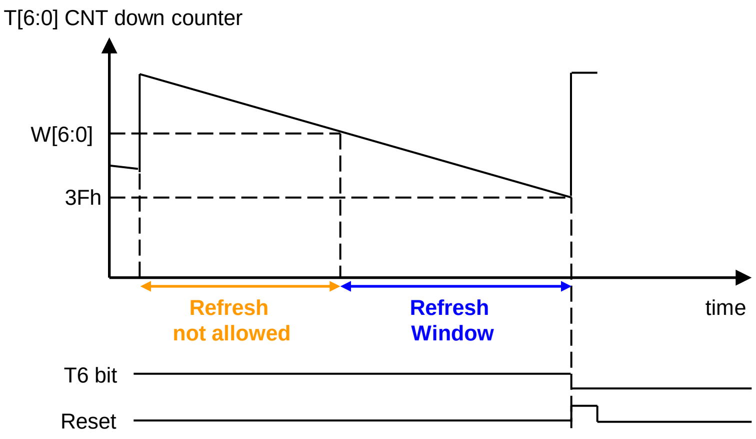
Independent and System Watchdogs

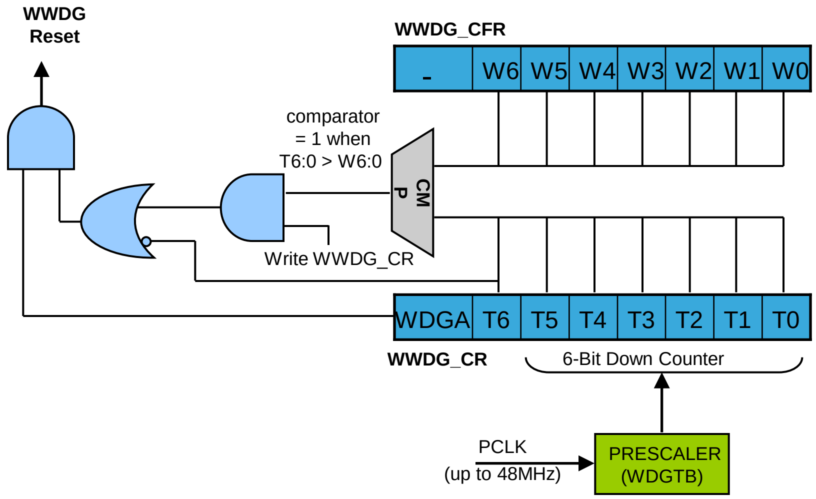
System window watchdog (WWDG) features (1/2)

- Configurable time-window – can be programmed to detect abnormally late or early application behavior

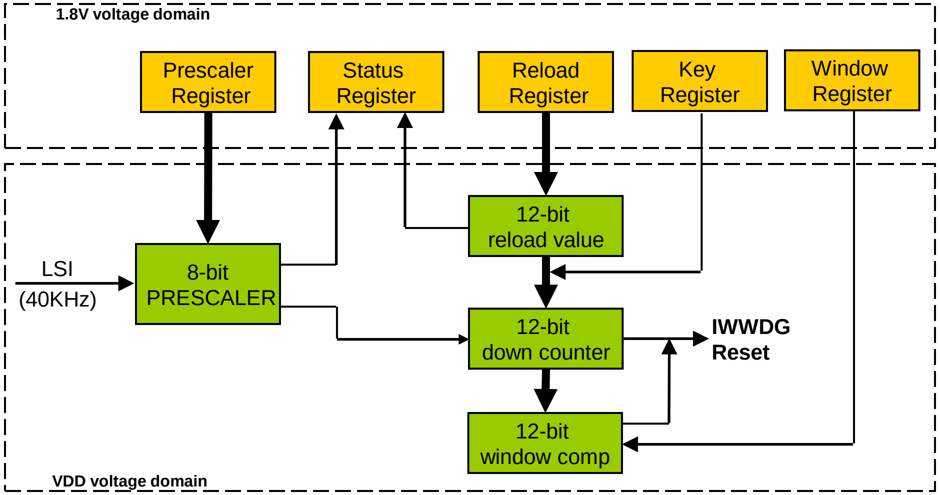
- Conditional reset
  - Reset (if watchdog activated) when the down counter value becomes less than 40h (T6=0)
  - Reset (if watchdog activated) if the down counter is reloaded outside the time-window

- To prevent WWDG reset:
  - write T[6:0] bits (with T6 equal to 1) at regular intervals while the counter value is lower than the time-window value (W[6:0])





Independent window watchdog (IWDG) features (1/2)



Best suited to applications which require the watchdog to run as a totally independent process outside the main application

- Selectable HW/SW start through option byte
- Clocked from an independent RC oscillator (LSI)
  - can operate in Standby and Stop modes
- Once enabled the IWWDG can’t be disabled (LSI can’t be disabled too)
- Implemented in the VDD voltage domain:
  - Is still functional in STOP and STANDBY mode

Serial peripheral interface SPI

SPI Features (1/3)

- Full duplex synchronous transfers (3 lines)
- Half duplex/Simplex synchronous transfers (2 lines, bi-directional data line at half duplex)
- Programmable clock polarity & phase, data MSB/LSB first
- Master/multi Master/Slave operation
- Dynamic software/hardware NSS management (Master/Slave)
- Hardware CRC feature (8-bit & 16-bit data frames checking)
- Flags with IT capability (TxE, RxNE, MODF, OVR, CRCERR)
- Programmable bit rate: **up to $f_{PCLK}/2$**
- BSY flag (ongoing communication check)
- DMA capability (separated Rx/Tx channels, automatic CRC & Tx/Rx access/threshold handling)

Up to 18 MHz bit rate

SPI Features (2/3)

NEW!

- **New enhanced NSS control:**
  - NSS pulse mode (NSSP)
  - TI mode
- **Programmable data frame from 4-bit to 16-bit**
- **Two 32-bit Tx/Rx FIFO buffers with DMA capability**
- **Data packed mode control**

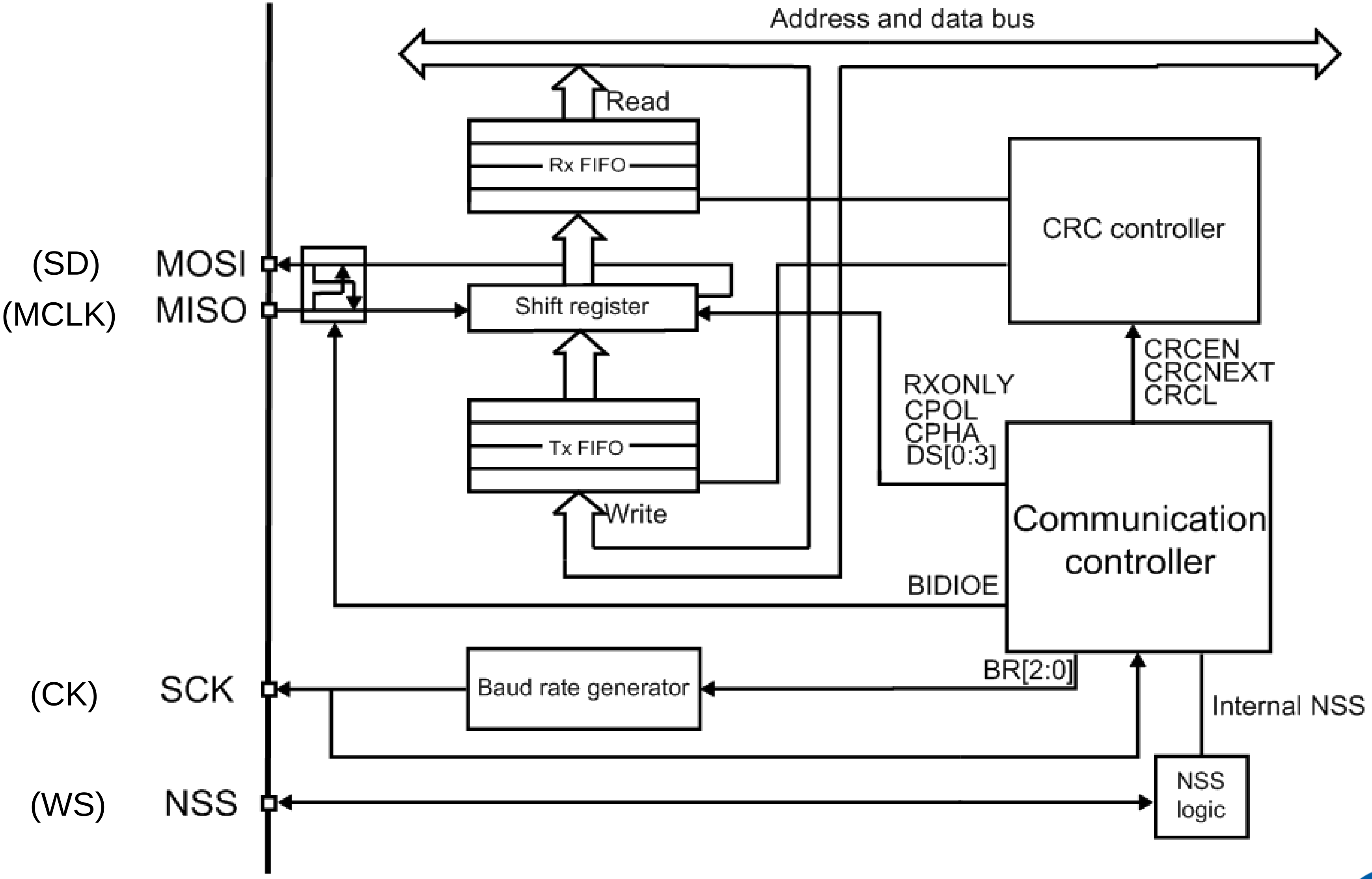
SPI Features (3/3)

STM32F0xx	SPI1	SP2	SPI3
SPI	X	X	-
I2S	X	-	-

STM32F37xxx	SPI1	SP2	SPI3
SPI	X	X	X
I2S	X	X	X

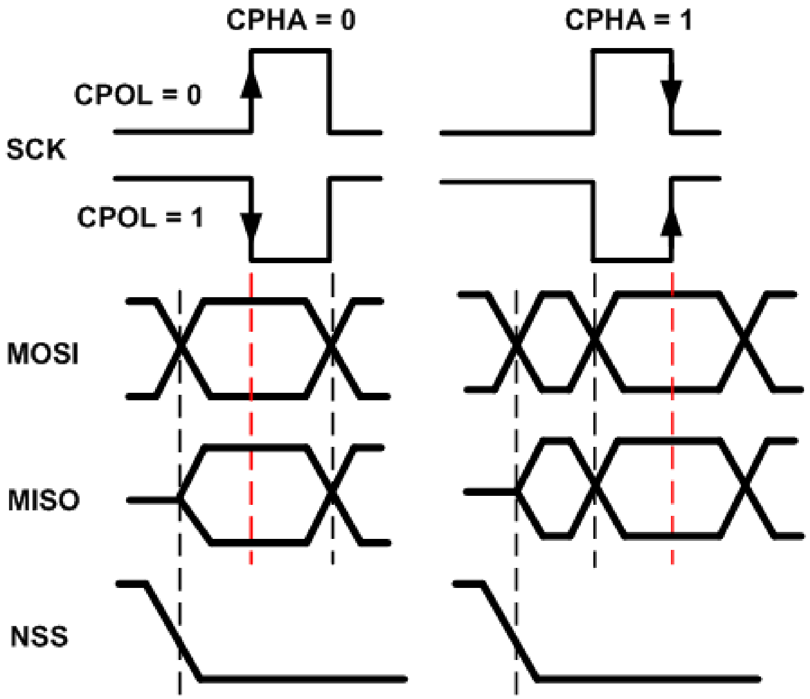
STM32F30xxx	SPI1	SP2	SPI3
SPI	X	X	X
I2S	-	X	X

SPI block scheme



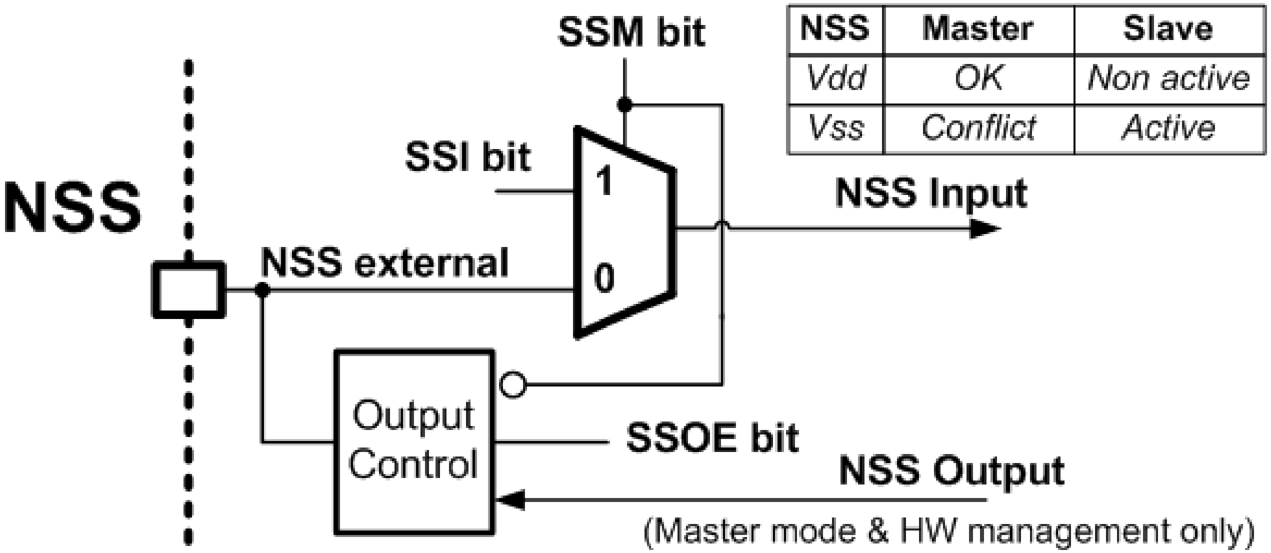
Data Frame Format (Motorola mode)

- Data frame format :
  - Programmable size of transfer data frame format from 4-bit up to 16-bit
  - Programmable clock polarity & phase, data bit order MSB/LSB first



NSS management

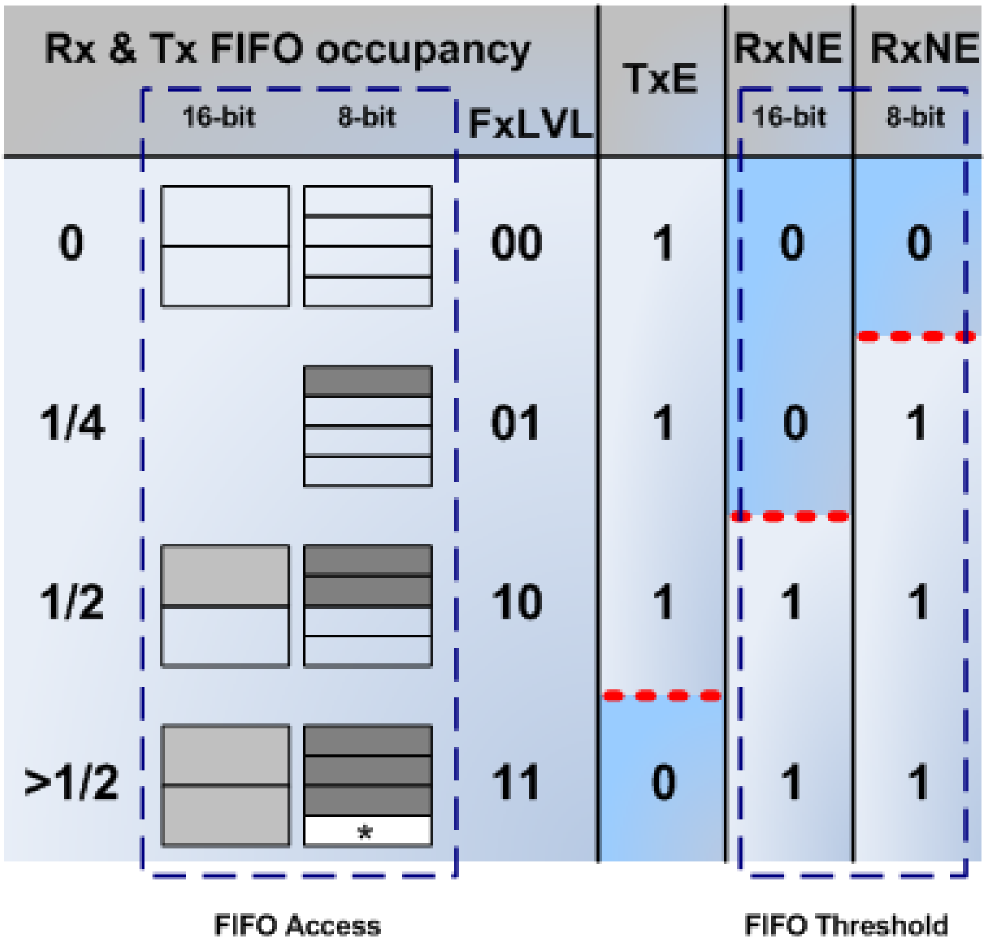
- **NSS input** – SSM selects HW control (NSS pin) or SW control (SSI bit):
  - Slave mode - select slave for communication (optionally can be used for synchronization of a transaction begin)
  - Master mode - signalize conflict between masters
- **NSS output** – HW control at master mode only (SSM=0, SSOE=1 or at specific modes - TI, NSSP)



NSS	Master	Slave
Vdd	*OK*	*Non active*
Vss	*Conflict*	*Active*

32-bit Rx and Tx FIFOs

- Two separated 32-bit FIFOs for receive and transmission
- 8-bit or 16-bit read/write access to FIFOs.
- Occupancy level flags FTLVL[1:0], FRLVL[1:0], TxE, RxNE
- Different capability of Tx and Rx FIFOs



Rx & Tx FIFO occupancy				TxE	RxNE	RxNE
	16-bit	8-bit	FxLVL		16-bit	8-bit
0			00	1	0	0
1/4			01	1	0	1
1/2			10	1	1	1
>1/2		*	11	0	1	1

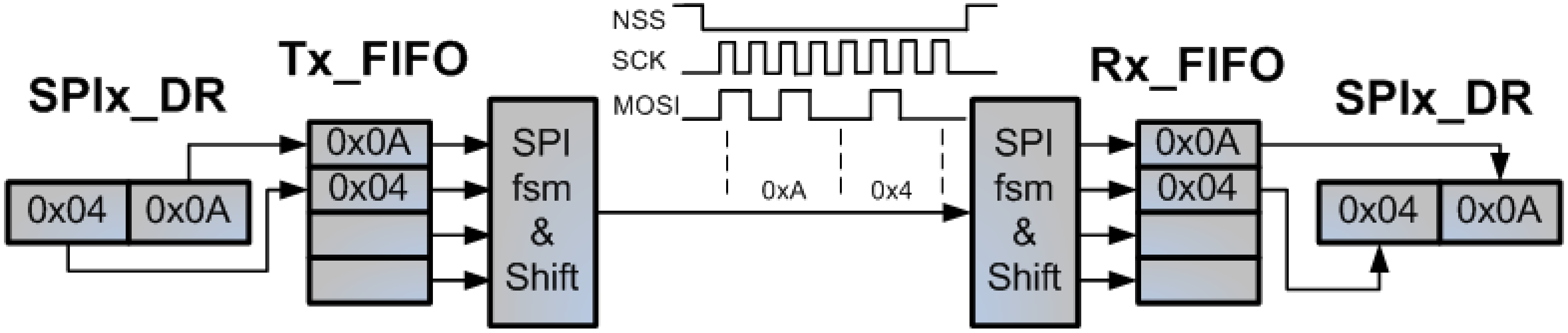
*) *Max 3x 8-bit for TxFIFO, 4x 8-bit for RxFIFO*

Packed mode & FIFOs access

- When data frame fits into one byte (from 4 up to 8 bits) two patterns can be accessed in parallel by single write/read FIFOs.

Example:

- 4-bit data frame length, MSB first, 16-bit threshold is set for RxFIFO,
  both FIFOs can be accessed by single 16-bit read or write
- 1x TxE event at transmitter - 1x RxNE event at receiver



16-bit access when write to data register
SPI_DR= 0x040A
when TxE=1

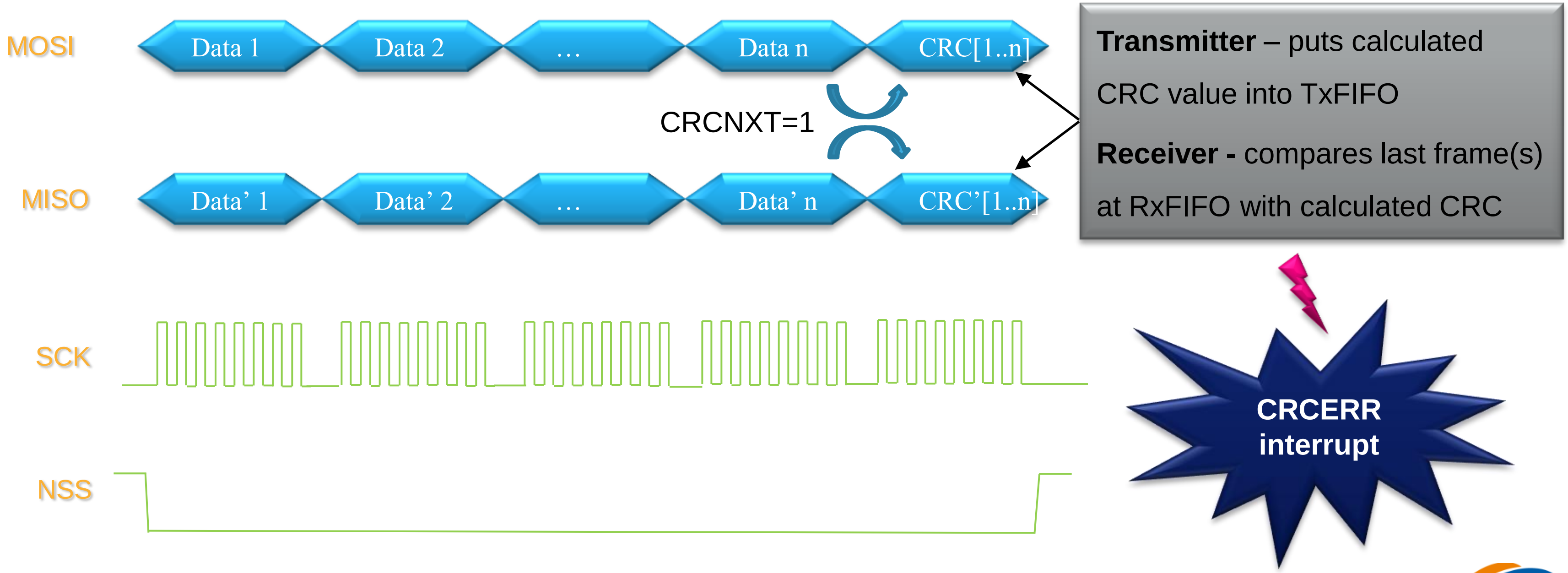
16-bit access when read from data register
u16_var= SPI_DR; /* ~u16_var= 0x040A */
when RxNE=1

CRC calculation (1/3)

- Hardware CRC feature for reliable communication:
  - Separated CRC calculators implemented for transmitted and received data
  - CRC value is transmitted as a last transaction(s)
  - CRC error checking for last received byte and interrupt generation on error
  - Programmable CRC polynomial calculation (odd polynomials only)
  - Available for 8-bit or 16-bit data patterns only
  - Two possible CRC calculations: CRC8, CRC16-CCITT standard

CRC calculation (2/3)

- Example of n data transfer between two SPIs followed by the CRC transmission of each one in Full-duplex mode
- SPI_TXCRCR and SPI_RXCRCR – separated registers for CRC calculation

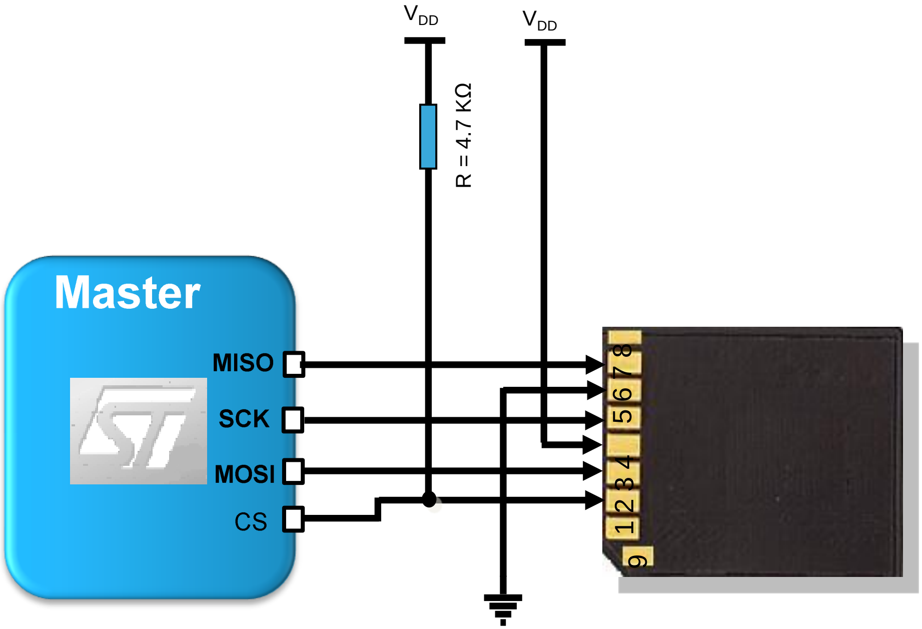


Transmitter – puts calculated CRC value into TxFIFO

Receiver - compares last frame(s) at RxFIFO with calculated CRC

CRC calculation (3/3)

- Basic SD/MMC support (SPI protocol):
  - Performance: speed up to 18MHz
  - Error checking: hardware CRC calculation



Universal Synchronous Asynchronous Receiver Transmitter (USART)

USART Features (1/3)

- Fully-programmable serial interface characteristics:
  - Data can be 8 or 9 bits
  - Even, odd or no-parity bit generation and detection
  - 1, 1.5 or 2 stop bit generation
  - Programmable baud rate generator
    - Integer part (12 bits)
    - Fractional part (4 bits)
  - Configurable oversampling method by 16 or by 8
  - Up to 9Mbps when the clock frequency is 72 MHz and oversampling by 8 is selected.
- Programmable data order with MSB or LSB first.
- Swappable Tx/Rx pin configuration
- Tx/Rx pins active level inversion & Binary data inversion
- Support hardware flow control (CTS and RTS)
- Dual clock domain allowing
  - UART functionality and wakeup from Stop mode
  - Convenient baud rate programming independent from the PCLK reprogramming
- Dedicated transmission and reception flags (TxE and RxNE) with interrupt capability

USART Features (2/3)

- Support for DMA
  - Receive DMA request
  - Transmit DMA request
- LIN Master/Slave compatible
- Synchronous Mode: Master mode only
- IrDA SIR Encoder Decoder
- Smartcard Capability T = 0, T = 1 (using the Address/character match, End of block, receiver timeout etc…)
- Basic support for Modbus communication (using Address/character match and Receiver timeout features).
- Single wire Half Duplex Communication

USART Features (3/3)

- Multi-Processor communication
  - USART can enter Mute mode
  - Mute mode: disable receive interrupts
  - Wake up from mute mode (by idle line detection or address mark detection)
- Auto-baudrate detection using various character patterns.
- Driver enable (for RS485) signal sharing the same pin as nRTS.
- 14 interrupt sources

STM32F30x USART Implementation

USART features	USARTs1/2/3	UART4/5
Hardware Flow Control	YES	NO
Continous communication using DMA	YES	YES
Multiprocessor communication	YES	YES
Synchronous mode	YES	NO
Smartcard mode	YES	NO
Single wire half duplex mode	YES	YES
IrDA	YES	YES
LIN	YES	YES
Dual clock domain and wake up from STOP mode	YES	YES
Receiver timeout	YES	YES
Modbus Communication	YES	YES
Autobaudrate detection	YES	NO
Driver enable	YES	NO

STM32F37x USART Implementation

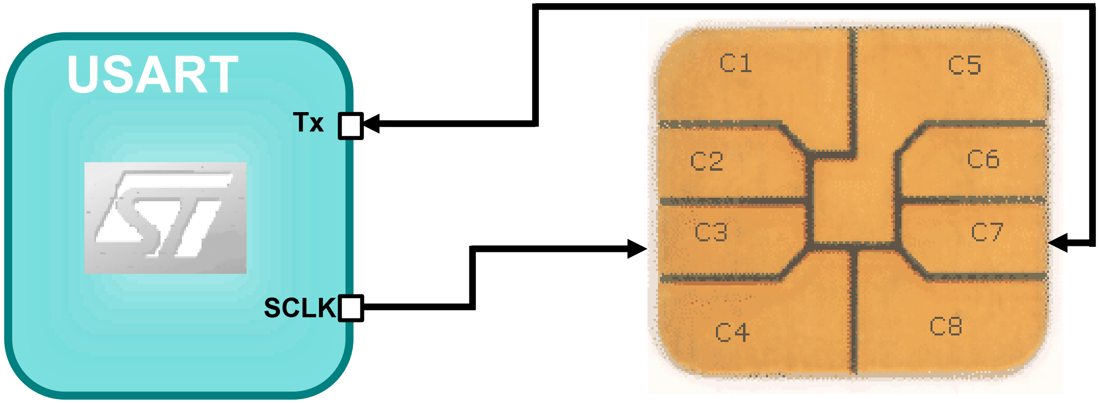
USART features	USART1	USART2	USART3
Hardware Flow Control	YES	YES	YES
Continous communication using DMA	YES	YES	YES
Multiprocessor communication	YES	YES	YES
Synchronous mode	YES	YES	YES
Smartcard mode	YES	YES	YES
Single wire half duplex mode	YES	YES	YES
IrDA	YES	YES	YES
LIN	YES	YES	YES
Dual clock domain and wake up from STOP mode	YES	YES	YES
Receiver timeout	YES	YES	YES
Modbus Communication	YES	YES	YES
Autobaudrate detection	YES	YES	YES
Driver enable	YES	YES	YES

Wakeup from STOP

- When USART_CLK clock is HSI or LSE, the USART is able to wakeup the MCU from STOP.
- Wakeup from STOP is enabled by setting UESM bit in the USART_CR1.
- The sources of wake up from STOP mode can be the standard RXNE interrupt (the RXNEIE bit must be set before entering Stop mode). Or, a specific interrupt may be selected through the WUS bit fields in the USART_CR3:
    - Wake up on address match
    - Wake up on Start bit detection
    - Wake up on RXNE

Smart Card mode (1/2)

- USART supports Smart Card Emulation ISO 7816-3
  - Half-Duplex, Clock Output (SCLK)
  - 9Bits data, 1.5 Stop Bits in transmit and receive.
  - T=0, T = 1 support
  - Programmable Clock Prescaler to guarantee a wide range clock input

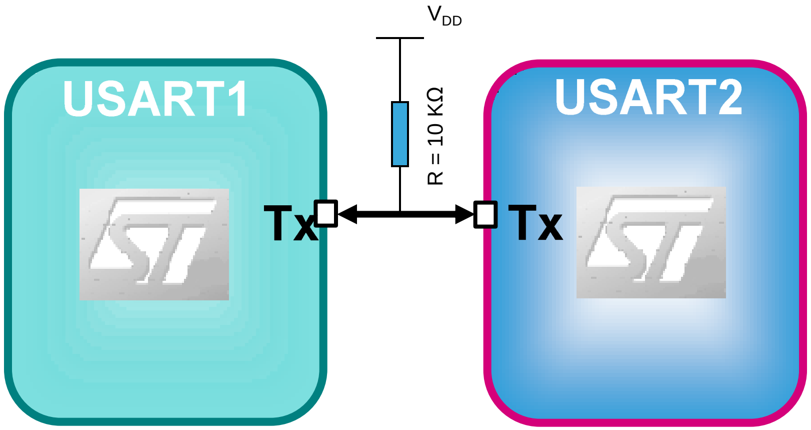


Smart Card mode (2/2)
STM32F3 vs STM32F1xx

Features		STM32F3xx	STM32F1xx
Maximum USART baudrate in Smartcard mode		3Mbits/s	4.5Mbits/s
T = 0	In case of transmission error, according to the protocol specification, the USART should send the character.	The USART can handle automatic re-sending of data. The number of retries is programmable (8 max).	The data retry should be done by software.
	In case of reception error , according to the protocol specification, the smartcard must resend the same character.	The number of maximum retries is programmable (8 max). If the received character is still erroneous after the programmed number of retries, the USART will stop transmitting the NACK and will signal the error as a parity error.	
	A programmable guardtime is automaticaly inserted between two consecutive characters in transmission.	Yes	No
New **T = 1**	Character Wait Time (CWT)	Implemented using the new timeout feature.	All T = 1 feature should be implemented by software.
	Block Wait Time (BWT)		
	Block length and end of block detection		
	Direct/Inverse convention	Implemented using some new features: MSB/LSBFIRST, Binary data inversion etc…	

Single Wire Half Duplex mode

- USART supports Half duplex synchronous communication mode
  - Only Tx pin is used (Rx is no longer used)
- Used to follow a single wire Half duplex protocol.



Half Duplex

IrDA SIR Encoder Decoder

- USART supports the IrDA Specifications
  - Half-duplex, NRZ modulation,
  - Max bit rate 115200 bps
  - The pulse width is 3/16 bit duration in normal mode
  - Low power mode: 1.42MHz <PSC < 2.12MHz

Modbus communication

- Basic support for the implementation of Modbus/RTU and Modbus/ASCII protocols.
  - Modbus/RTU: In this mode, the end of one block is recognized by a “silence” (idle line) for more than 2 character times. This function is implemented through the programmable timeout function.
  - Modbus/ASCII: In this mode, the end of a block is recognized by a specific (CR/LF) character sequence. The USART manages this mechanism using the character match function.

Auto-baudrate detection

- 2 patterns for auto-baudrate detection:
  - Any character starting with a bit at 1 → the USART will measure the duration of the START bit (Falling edge to rising edge).
  - Any character starting with a 10xx bit pattern → the USART will measure the duration of the START bit and of the first data bit (Falling edge to Falling edge).
- Once the automatic baudrate detection is activated, the USART will wait for the first character on the RX line. The auto-baudrate completion is indicated by the setting of ABRF flag.
- The automatic baud rate detection range: bit duration should be between 16 to 65536 USART clock periods.
- Only oversampling by 16 is used with auto-baudrate detection feature.
- If the line is noisy, the correct baudrate detection is not guaranteed.

USART Interrupts

Interrupt event	Interrupt flag
Transmit Data Register Empty	TXE
Transmission Complete	TC
CTS	CTSIF
Receive Data Register Not Emptyy	RXNE
Overrun Error	ORE
Idle line detection	IDLE
Parity Error	PE
LIN break	LBDF
Noise Flag, Overrun error and Framing Error in multibuffer communication.	NE, ORE, FE
Character Match	CMF
Receiver timeout error	RTOF
End of Block	EOBF
Wakeup from STOP mode	WUF

STM32F1/F2/L USART vs STM32F3 USART Main features (1/2)

Feature	STM32F3	STM32F1/2/L
Programmable data length (8 or 9 bits)	Yes	Yes
Configurable stop bits	1, 1.5, 2	0.5, 1, 1.5, 2
Synchronous mode (Master only)	Yes	Yes
Single wire Half duplex	Yes	Yes
Programmable parity	Yes	Yes
Hardware flow control (nCTS/nRTS)	Yes	Yes
Driver Enable (for RS485)	Yes	No
Swappable Tx/Rx pin	Yes	No
IrDA	Yes	Yes
Basic support for Modbus	Yes	No
LIN	Yes	Yes
Smartcard	Yes (T = 0, T=1)	Yes (T = 0)
Dual Clock domain and wake up from STOP mode	Yes	No
Programmable data order with MSB first or LSB first	Yes	No

STM32F1/F2/L USART vs STM32F3 USART Main features (2/2)

Feature	STM32F3	STM32F1/2/L
Receiver timeout	Yes	No
Auto-baudrate detection	Yes	No
Continous communication using DMA	Yes	Yes
Address/character match interrupt	Yes	No
End of Block interrupt	Yes	No
Multiprocessor communication	Yes	Yes

Inter-Integrated Circuit (I2C)

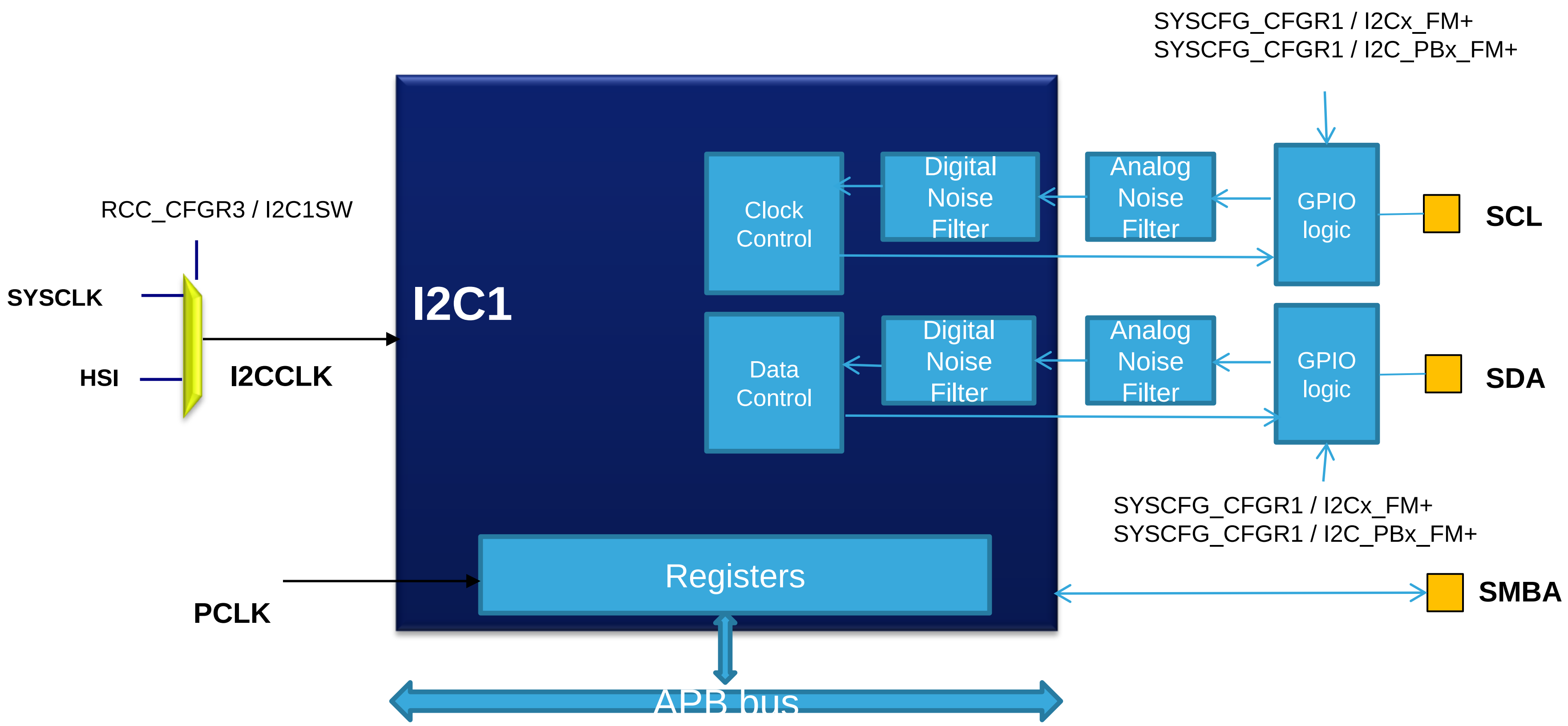
I2C Features (1/2)

- I2C specification rev03 compatibility
- SMBus 2.0 HW support
- PMBus 1.1 Compatibility
- Multi Master and slave capability
- Controls all I²C bus specific sequencing, protocol, arbitration and timing
- Standard, fast and **fast mode +** I²C mode (**up to 1MHz**)
- **20mA output drive** capability for FM+ mode

I2C Features (2/2)

- 7-bit and 10-bit addressing modes
- **Multiple** 7-bit Addressing Capability with **configurable mask**
- **Programmable setup and hold time**
- **Easy to use event management**
- **Programmable analog and digital noise filter**
- **Wakeup from STOP mode on address match**
- Optional clock stretching
- **Independent clock**
- 1-byte buffer with DMA capability

I2C Block Diagram



I2C SDA and SCL noise filter

251

- Analog noise filter in SDA and SCL I/O
  - Can filter spikes with a length up to 50ns
  - This filter can be enabled or disabled by SW (enabled by default)

- Digital noise filter for SDA and SCL
  - Suppress spikes with a programmable length from **0 to 15 I2CCLK periods**.

- Only analog filter can be enabled when Wakeup from STOP feature is enable.

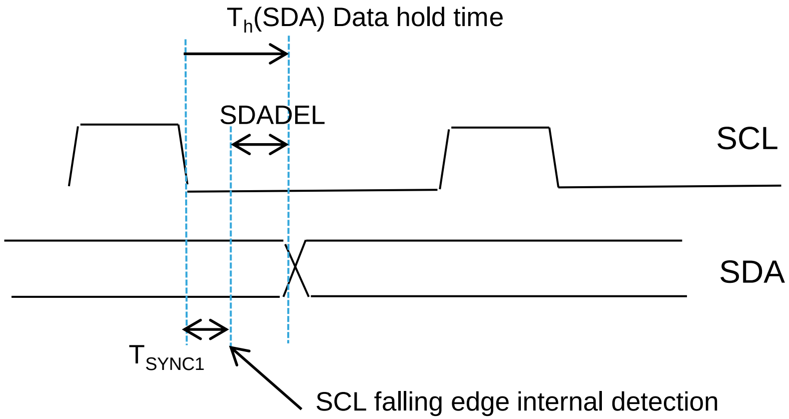
- Filters configuration must be programmed when the I2C is disable.

I2C Programmable timings

252

- Setup and Hold timings between SDA and SCL in transmission are programmable by SW with PRESC, SDADEL and SCLDEL fields in I2C Timing Register (I2Cx_TIMINGR).
  - SDADEL is used to generate Data Hold time. T_{SDADEL} = SDADEL * (PRESC+1) * T_{I2CCLK}
  - SCLDEL is used to generate Data Setup time. T_{SCLDEL} = (SCLDEL+1) * (PRESC+1) * T_{I2CCLK}

  - **Example Data Hold Time** :



T_h(SDA) Data hold time
SDADEL
SCL
SDA
T_{SYNC1}
SCL falling edge internal detection

- The Setup and Hold configuration must be programmed when the I2C is disable.

- I2C_Timing_Config_Tool is available in FWLib to calculate I2C_TIMINGR value for your application.

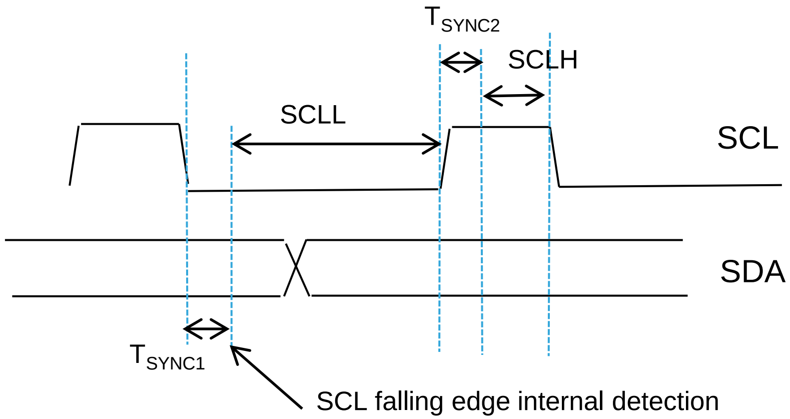
I2C Master clock generation

253

- SCL Low and High duration are programmable by SW with PRESC, SCLL and SCLH fields in I2C Timing Register (I2Cx_TIMINGR).
  - SCL Low counter is (SCLL+1) * (PRESC+1) * $T_{I2CCLK.}$. It starts counting after SCL falling edge internal detection. After counting, SCL is released.
  - SCL High counter is (SCLH+1) * (PRESC+1) * T_{I2CCLK} . It starts counting after SCL rising edge internal detection. After counting SCL is driven low.

- The total SCL period is :
  - TSYNC1 + TSYNC2 + [(SCLL+1) + (SCLH+1)] * (PRESC+1) * T_{I2CCLK}

  - **SCL Period**:



- The SCLL and SCLH configuration must be programmed when the I2C is disable.

- I2C_Timing_Config_Tool is available in FWLib to calculate I2C_TIMINGR value for your application.

Slave Addressing Mode

- I2C can acknowledge several slave addresses. 2 address registers :
  - I2Cx_OAR1 : 7-bit or 10-bit mode.
  - I2Cx_OAR2 : 7-bit mode only. OA2MSK[2:0] allow to mask from 0 to 7 LSB of OAR2 :

OA2MSK[2:0]	Address match condition
000	address[7:1] = OA1[7:1]
001	address[7:2] = OA1[7:2] (Bit 1 is don’t care)
010	address[7:3] = OA1[7:3] (Bit 2:1 are don’t care)
...	
111	All addresses are acknowledged except I2C reserved addresses.

life.augmented

Wakeup from STOP on address match

255

- When I2CCLK clock is HSI, the I2C is able to wakeup MCU from STOP when it receives its slave address. All addressing mode are supported.
  - During STOP mode and no address reception : HSI is switched off.
  - On START detection, I2C enables HSI, used for address reception.
- Wakeup from STOP is enabled by setting WUPEN in I2C1_CR1.
- Clock stretching must be enabled to ensure proper operation: NOSTRETCH=0.

Easy Master mode management

256

- For payload <= 255 bytes : **only 1 write action needed** !! (apart data rd/wr)

I2Cx_CR2 is written w/ :

- START=1
- SADD : slave address
- RD_WRN : transfer direction
- NBYTES = N : number of bytes to be transferred
- **AUTOEND** =1 : STOP automatically sent after N data.

AUTOEND	
0 : Software end mode	End of transfer SW control after NBYTES data transfer : • TC flag is set. Interrupt if TCIE=1. • TC is cleared when START or STOP is set by SW ➢ If START=1 : RESTART condition is sent
1 : Automatic end mode	STOP condition sent after NBYTES data transfer

- Data transfer managed by Interrupts (TXIS / RXNE) or DMA

Easy to use event management

- For payload > 255 : in addition, **RELOAD** must be set in I2Cx_CR2.

RELOAD	
0 : No reload	NBYTES data transfer is followed by STOP or ReSTART
1 : Reload mode	NBYTES is reloaded after NBYTES data transfer (data transfer will continue) : • TCR flag is set. Interrupt if TCIE=1. • TCR is cleared when I2Cx_CR2 is written w/ NBYTES≠0

- AUTOEND = 0 has no effect when RELOAD is set

Slave mode

258

- By default : I2C slave uses clock stretching .
  - ➢This can be disabled by setting NOSTRETCH=1

- Reception : Acknowledge control can be done on selected bytes in **Slave Byte Control (SBC)** mode with RELOAD=1
  - SBC = 1 enables the NBYTES counter in slave mode (Tx and Rx modes).
  - SBC = 1 is allowed only when NOSTRETCH=0.

SBC	
0 : Slave Byte Control disable	All received bytes are acknowledged.
1 : Slave Byte Control enable	If RELOAD=1, after NBYTES data are transferred : • TCR set & SCL stretched before ACK pulse in reception. • TCR is cleared when I2Cx_CR2 is written w/ NBYTES≠0 ➢ if I2Cx_CR2/NACK = 1: received byte is NOT Acknowledged

I2C events

Interrupt event	Interrupt flag
Receive Buffer Not Empty	RXNE
Transmit buffer Interrupt Status	TXIS
Stop detection interrupt flag	STOPF
Transfer Complete Reload	TCR
Transfer Complete	TC
Address matched	ADDR
NACK reception	NACKF

SMBUS

- **ARP** (Address resolution protocol) support : Device default address, Arbitration in slave mode
- **Host Notify** protocol support : host address
- **Alert** support : Alert pin and Alert Response support
- Configurable **Timeout** detection : Clock low timeout, Cumulative clock low extend time
- Configurable **bus idle** detection
- Command and data **acknowledge control** in SBC mode
- **PEC** HW calculation

I2S Peripheral

life.augmented

SPI / I2S mode switch

- The I2S protocol is used for audio data communication between a microcontroller/DSP and an audio Codec/DAC.

- I2S interface is implemented as a mode in the SPI peripheral.

- To switch from SPI to I2S mode:
  - Disable SPI peripheral (reset SPE bit in SPI_CR1 register)
  - Select I2S mode (set I2SMOD bit in SPI_I2SCFGR register)

I2S Features (1/2)

- Two I2Ss: Available on SPI2 and SPI3 peripherals.
- Two I2Ss extension added for Full-Duplex communication.
- Simplex/or Full duplex communication (transmitter and receiver)
- I2S2 and I2S3 operate in master or slave configuration.
- 8-bit programmable linear prescaler to support all standard audio sample frequencies from 8 kHz up to 192 kHz.
- Audio-frequency precision same as **high-density and XL-density devices.**
- Programmable data format (16-, 24- or 32-bit data formats).

I2S Features (1/2)

- Underrun flag in slave transmit mode, Overrun flag in receive mode and **new de-synchronization flag** in slave transmit/receive mode.

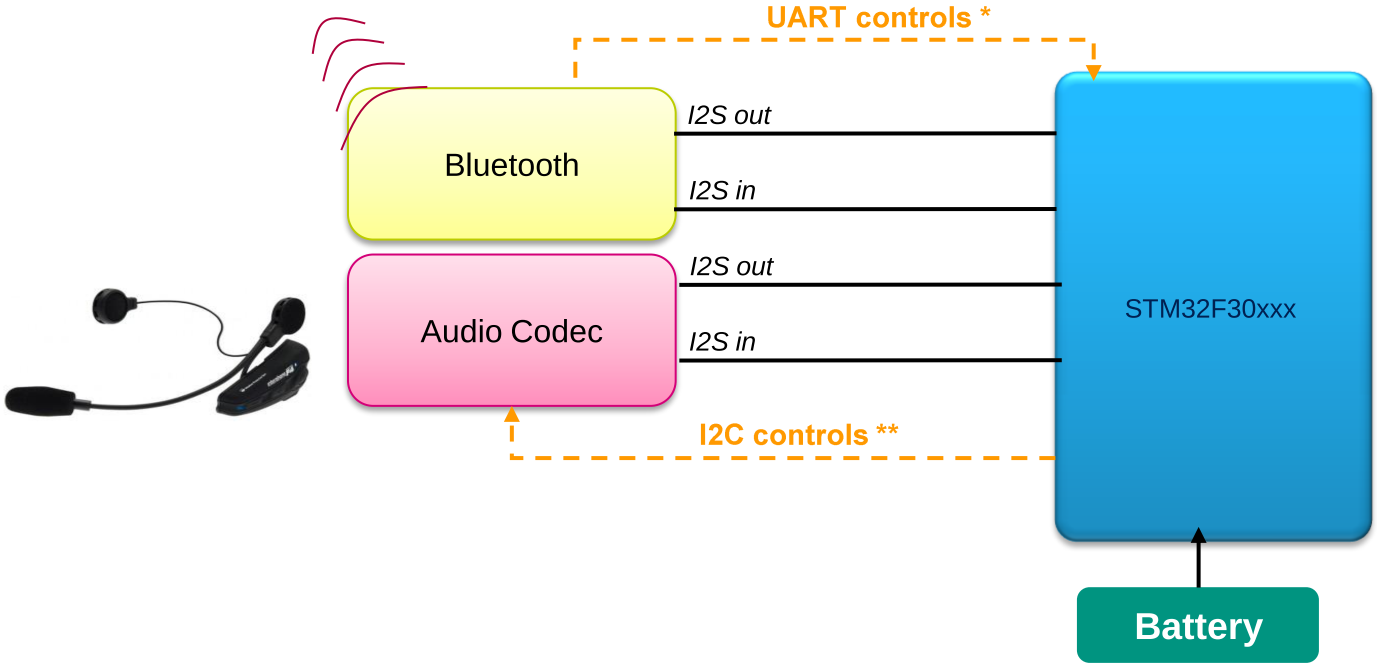
- Support for DMA: New DMA requests for I2S2_ext/I2S3_ext allows full duplex transfers.

- I2S protocols supported:
  - I2S Phillips standard.
  - MSB Justified standard (Left Justified).
  - LSB Justified standard (Right Justified).
  - PCM standard (with short and long frame synchronization on 16-bit channel frame or 16-bit data frame extended to 32-bit channel frame)

- **Master clock** may be output to drive an external audio component. Ratio is fixed at **256xFs** (where Fs is the audio sampling frequency).

The choice of the standard strongly depends on the external device and the audio data to be transmitted

2 x Full-Duplex Communication



Bluetooth Headset

* Depends on the Codec control method
** Depends on the Bluetooth control method

STM32F37xxx vs STM32F30xxx

Features	STM32F37xxx	STM32F30xxx
Instance	3 (I2S1, I2S2,I2S3)	2 (I2S2, I2S3)
Communication mode	Simplex	Simplex/full-duplex
External clock	No	Yes

Real-Time Clock (RTC)

life.augmented

COMPELFEST

RTC Features (1/2)

- Ultra-low power battery supply current.
- Calendar with Sub seconds, seconds, minutes, hours, week day, date, month, year.
- Daylight saving compensation programmable by software
- Two programmable alarms with interrupt function. The alarms can be triggered by any combination of the calendar fields.
- A periodic flag triggering an automatic wakeup interrupt. This flag is issued by a 16-bit auto-reload timer with programmable resolution. This timer is also called 'wakeup timer'.
- A second clock source (50 or 60Hz) can be used to update the calendar.
- Maskable interrupts/events:
  - Alarm A, Alarm B, Wakeup interrupt, Time-stamp, Tamper detection
- Digital calibration circuit (periodic counter correction) to achieve 0.95 ppm accuracy
- Time-stamp function for event saving with **sub second precision** (1 event)
- Backup registers which are reset when an tamper detection event occurs.
  - 64 bytes for STM32F30x
  - 128 bytes for STM32F37x

RTC Features (2/2)

- Alternate function outputs:
  - RTC_CALIB: 512 Hz or 1Hz clock output (with an LSE frequency of 32.768 kHz). It is routed to the device RTC_OUT output.
  - RTC_ALARM: Alarm A, B flag output. It is routed to the device RTC_OUT output.

- Alternate function inputs:
  - RTC_TAMP1: tamper1 event detection.
  - RTC_TAMP2: tamper2 event detection.
  - RTC_TAMP3: tamper3 event detection.
  - RTC_TS: timestamp event detection.
  - RTC_REFIN: reference clock input.

- The RTC clock source could be any of the following three:
  - LSE oscillator clock.
  - LSI oscillator clock.
  - HSE divided by 32 in clock controller.

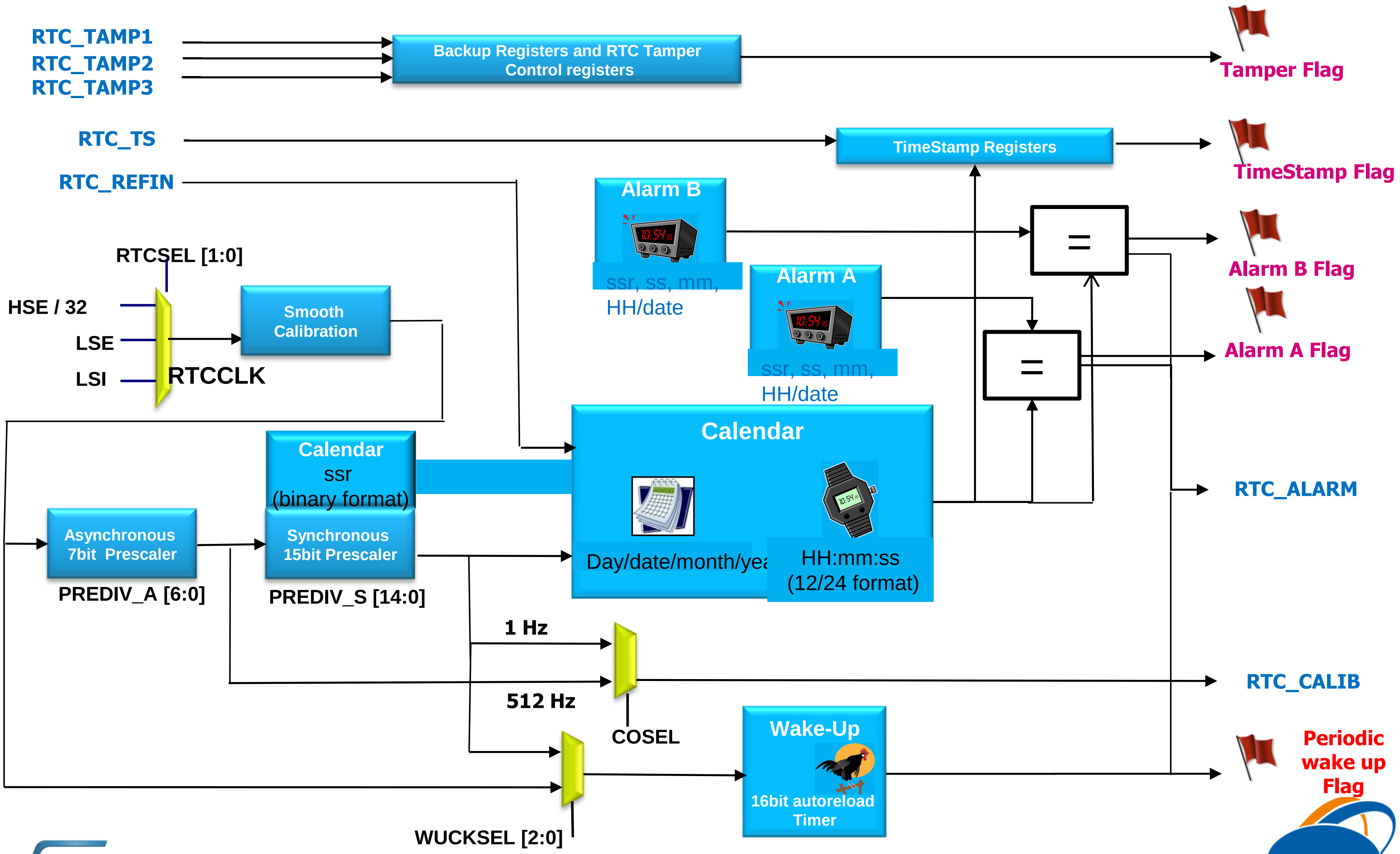
RTC overview across families (1/2)

	STM32F2x	STM32F0x	STM32F4x	**STM32F30x**	**STM32F37x**
RTC in VBAT	**YES**				
Calendar in BCD	**YES**				
Calendar Sub seconds access	NO	**YES** **Resolution down to RTCCLK**			
Calendar synchronization on the fly	NO	**YES**			
Alarm on calendar	2 wo/ subseconds	1 w/ subseconds	**2 w/ subseconds**		
Calendar Calibration	Calib window : **64min** Calibration step: -2ppm/ +4ppm Range [-63ppm+126ppm]	Calib window : **8s/16s/32s** Calibration step: 3.81ppm/1.91ppm/**0.95 ppm** Range [-480ppm +480ppm]			

RTC overview across families (2/2)

	STM32F2x	STM32F0x	STM32F4x	**STM32F30x**	**STM32F37x**
Synchronization on mains	**YES**				
Periodic wakeup	YES	NO	**YES**		
Timestamp	YES Sec, Min, Hour, Date	**YES** **Sec, Min, Hour, Date, Sub seconds**			
Tamper	YES 2 pins/1 event Edge detection only	YES 2 pins/2 event Level Detection Configurable filtering	**YES** **3 pins/ 3 events** **Level Detection** **with Configurable filtering**		
32-bit Backup registers	20	5	20	**16**	**32**
PC13-14-15 output state kept in Standby (if not used by RTC/LSE)	NO	YES	NO	**YES**	

RTC Block Diagram



RTC registers write protection

- By default and after reset, the RTC registers are write protected to avoid possible parasitic write accesses.
  - DBP bit must be set in PWR_CR to enable RTC write access
  - A Key must be written in RTC_WPR register.

- To unlock write protection on all RTC registers
  - 1. Write ‘0xCA’ into the RTC_WPR register
  - 2. Write ‘0x53’ into the RTC_WPR register

*** Except for the clear of Alarm and Wakeup timer interrupt flags**

➔**Writing a wrong key reactivates the write protection.**

RTC Clock Sources

- The RTC has two clock sources:
  - ***RTCCLK*** used for RTC timer/counter, can be either the HSE/32, LSE or LSI clocks.
  - ***PCLK1*** used for RTC register read/write access.

- Before to start using the RTC you have to program the clock controller :
  - Configure and Enable the **RTCCLK** source in the RCC_BDCR register

RTC in Low Power Modes and in Reset

281

- The RTC remains active what ever the low power mode
  - Sleep, STOP, STANDBY

- When enabled, 5 events can exit the device from low power modes:
  - Alarm A
  - Alarm B
  - Wakeup
  - Tamper 1/ 2 / 3
  - TimeStamp

- The RTC remains active in VBAT mode (VDD off) when clocked by LSE

- The RTC remains active under Reset except at Power-on Reset
  - The RTC configuration registers including prescaler programming are not affected by system Reset else than Power-on Reset.
  - When clocked by LSE, the RTC clock is not stopped under Reset, except power-on reset.

RTC Alternate function configuration (1/2)

282

RTC pin (PC13) :

Pin configuration and function	RTC_ALARM enabled	RTC_CALIB enabled	Tamper enabled	Time stamp enabled	PC13MODE	PC13VALUE
Alarm out output OD	**1**	**Don’t care**	**Don’t care**	**Don’t care**	**Don’t care**	**0**
Alarm out output PP	**1**	**Don’t care**	**Don’t care**	**Don’t care**	**Don’t care**	**1**
Calibration out output PP	**0**	**1**	**Don’t care**	**Don’t care**	**Don’t care**	**Don’t care**
TAMPER input floating	**0**	**0**	**1**	**0**	**Don’t care**	**Don’t care**
TIMESTAMP and TAMPER input floating	**0**	**0**	**1**	**1**	**Don’t care**	**Don’t care**
TIMESTAMP input floating	**0**	**0**	**0**	**1**	**Don’t care**	**Don’t care**
Output PP forced	**0**	**0**	**0**	**0**	**1**	**PC13 output data value**
Standard GPIO	**0**	**0**	**0**	**0**	**0**	**Don’t care**

OD: open drain; PP: push-pull.

PC13 is available in VBAT mode

RTC Alternate function configuration (2/2)

283

LSE pin PC14 configuration (1) :

Pin configuration and function	LSEON	LSEBYP	PC14MODE	PC14VALUE
LSE oscillator	1	0	Don’t care	Don’t care
LSE BYPASS	1	1	Don’t care	Don’t care
Output PP forced	0	Don’t care	1	PC14 output data value
Standard GPIO	0	Don’t care	0	Don’t care

1. OD: open drain; PP: push-pull.

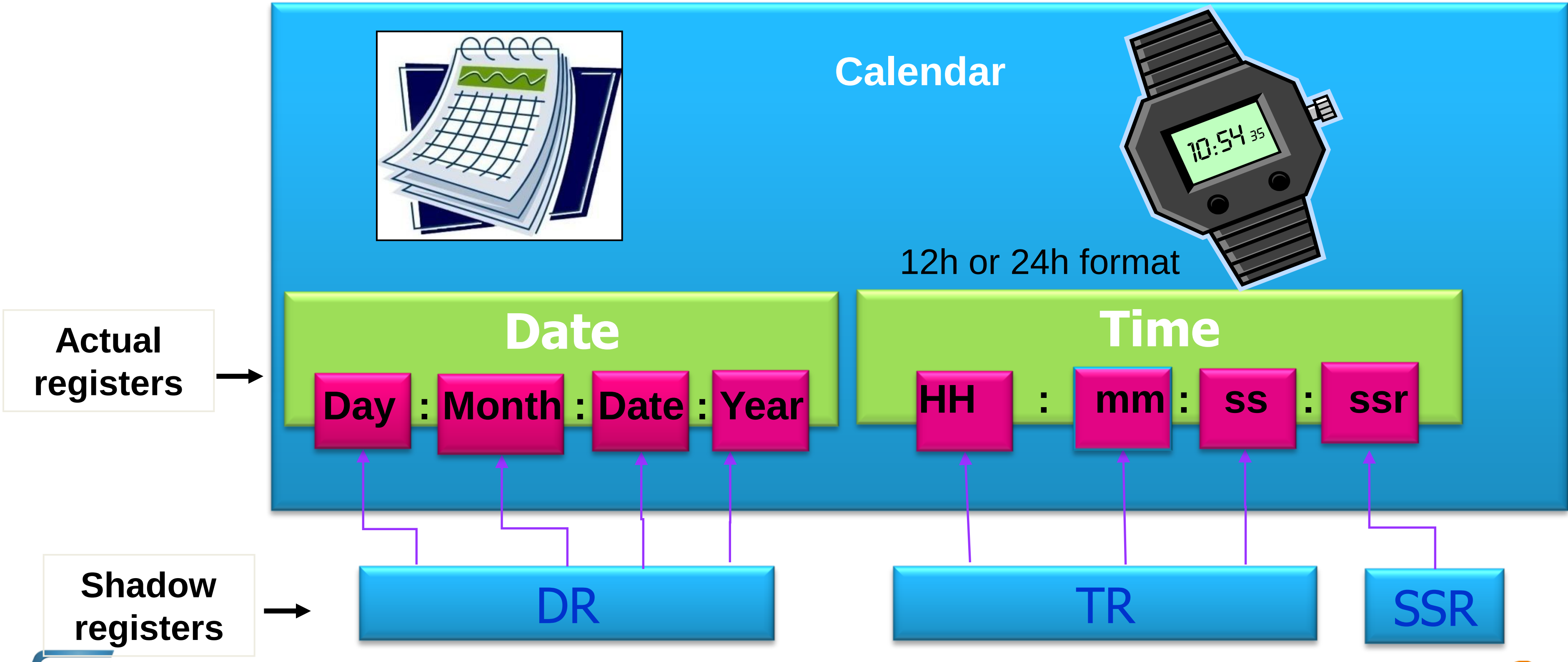
LSE pin PC15 configuration (1) :

Pin configuration and function	LSE ON	LSEBYP	PC15MODE	PC15VALUE
LSE oscillator	1	0	Don’t care	Don’t care
Output PP forced	1	1	1	PC15 output data value
	0	Don’t care		
Standard GPIO	0	Don’t care	0	Don’t care

1. OD: open drain; PP: push-pull.

RTC Calendar (1/4)

- The initialization or the reading of the calendar value is done through 3 shadow registers, SSR, TR and DR. The RTC TR and DR registers are in BCD format.
- SSR register represents the RTC Sub seconds register



Calendar

12h or 24h format

Actual registers

Date

Day : Month : Date : Year

Time

HH : mm : ss : ssr

Shadow registers

DR

TR

SSR

RTC calendar (2/4)

- RTC initialization :
  - **Enter in initialization phase mode by setting the INIT bit in ISR register**
    - This mode is confirmed with the INITF flag also in ISR register
  - Program the prescaler register (PRER) according to the clock source to get 1Hz clock to the calendar.
  - **Load the initial date values in the 2 shadow registers (TR, DR).**
    - And other configuration registers like RTC_CR (hour format, …)
  - **Exit the initialization phase clearing INIT bit.**
    - The actual calendar register are then automatically loaded and the counting restarts after few RTCCLK clock periods.

- After reset the check of the INITS flag in ISR register indicates if the calendar is already initialized (year not at zero) or not (like after Power-on).

- To manage the “daylight saving” there are 3 bits in CR:
  - SUB1H or ADD1H to subtract or add one hour to the calendar
  - BCK to memorize above action

RTC calendar (3/4)

- The shadow registers are automatically updated each time the RTCCLK clock is synchronized with System Clock.

- The calendar read can be done in 2 different modes :
  - BYPSHAD=0 : Read shadow registers
    - RSF flag in ISR register is used to ensure that the calendar value from shadow register is the up-to-date one.
    - Update of DR is frozen **after reading TR** , and unfrozen when **DR is read**.
    - Update of TR and DR is frozen **after reading SSR** , and unfrozen when **DR is read**.
  - BYPSHAD=1 : Bypass shadow registers
    - Reading calendar makes direct access to the calendar counters
    - Software must read all calendar registers twice and compare the results to ensure that the data are coherent and correct.

RTC calendar (4/4)

- Calendar can be synchronized up to 1s on the fly by adding/subtracting an offset with the sub second resolution.
  - Allow synchronization to remote clock
- Reference Clock detection: A more precise second source clock (like mains 50 or 60 Hz) can be used to enhance the long-term precision of the calendar:
  - The second source clock is automatically detected and used to update the calendar
  - The LSE clock is automatically used to update the calendar whenever the second source clock becomes unavailable
- Timestamp : Calendar value (including sub-seconds) is saved in Timestamp registers on external I/O event

RTC Programmable Alarm

- 2 Full programmable Alarms
- Able to **exit** the device from **STOP/STANDBY modes**.
- **Alarms event** can also be routed to the **specific output pin RTC_OUT** with configurable polarity.
- The **Alarm flags** are set if the calendar sub seconds, seconds, minutes, hours or date match the value programmed in the alarm registers (ALRMASSR & ALRMAR, ALRMBSSR & ALRMBR).
- Calendar sub second**,** seconds, minutes, hours or date fields can be independently selected (maskable or not maskable).

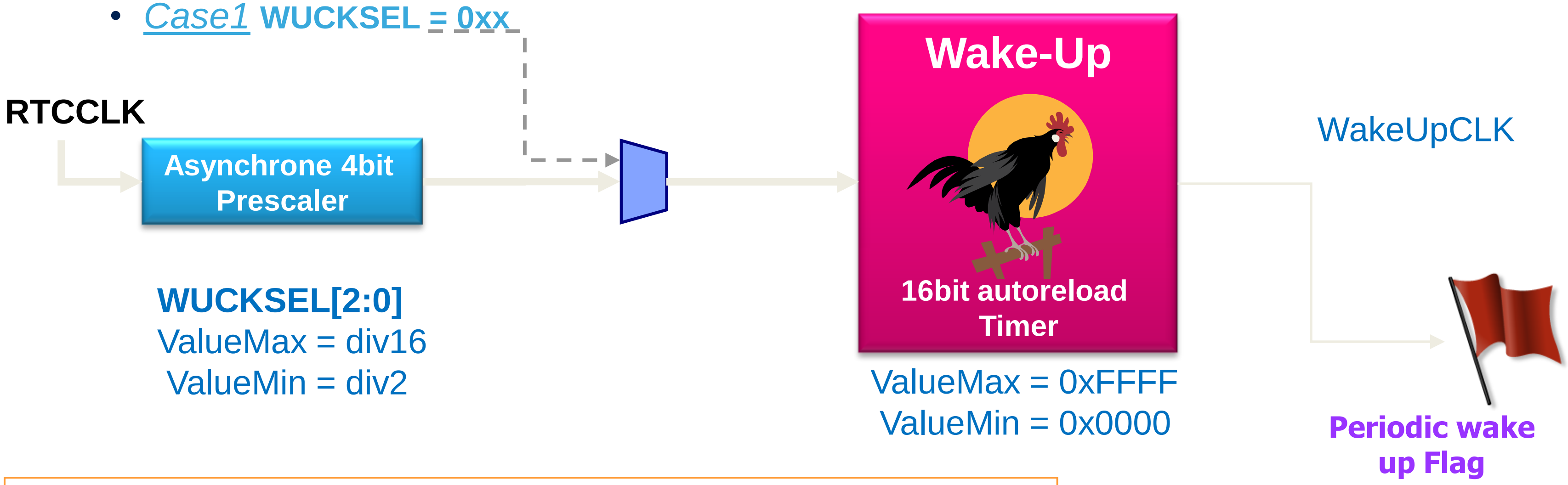
WakeUp configuration (1/3)

- The periodic wakeup flag is generated by a 16-bit programmable binary auto-reload down counter (WUTR registers)

- Able to **exit** the device from **STANDBY modes**.

- The wakeup clock source selection is done via WUCKSEL [2:0] bits in control register RTC_CR (to program these bits the auto wakeup must be deactivated, WUTE=0).

- **3 possible cases are possible:**
  - *Case1* **WUCKSEL = 0xx**



RTCCLK

Asynchrone 4bit Prescaler

WUCKSEL[2:0]
ValueMax = div16
ValueMin = div2

Wake-Up

16bit autoreload Timer

ValueMax = 0xFFFF
ValueMin = 0x0000

WakeUpCLK

Periodic wake up Flag

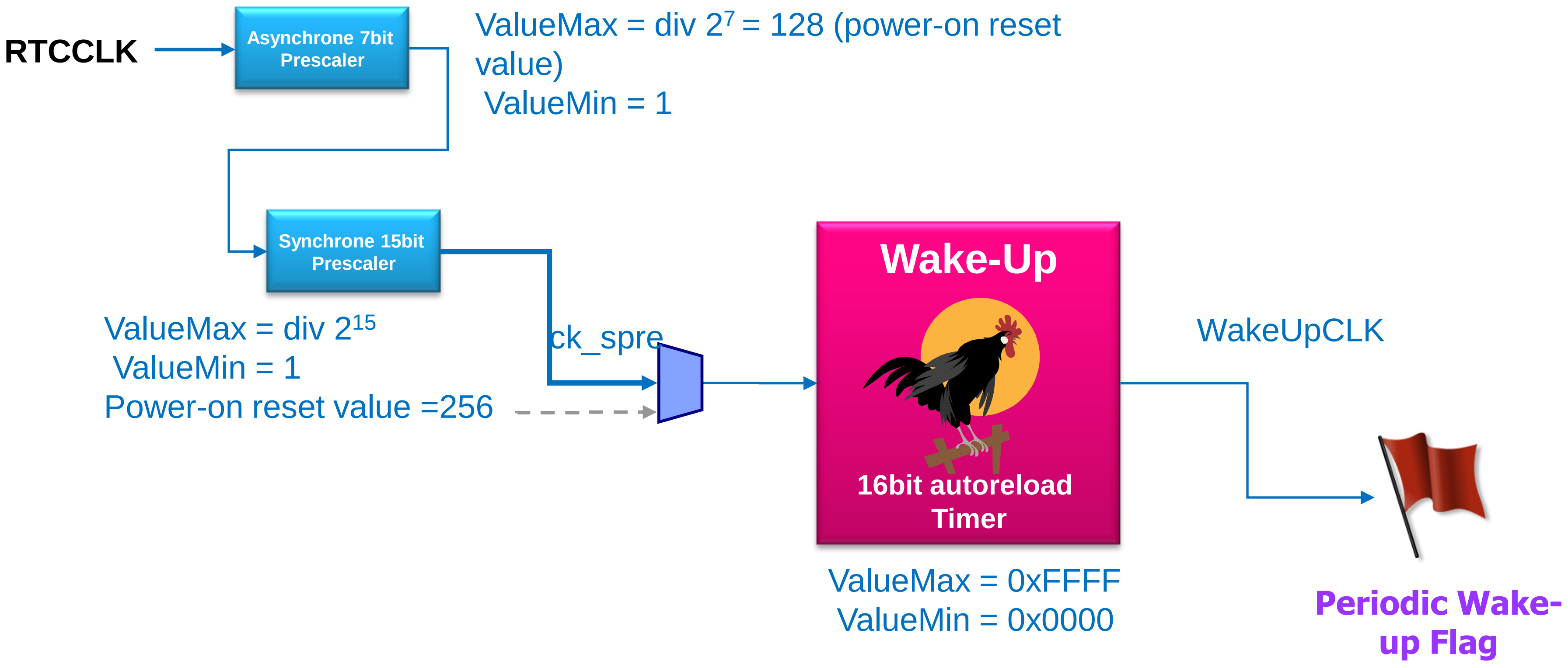
WakeUpCLKmin = RTCCLK/(2 x (0x0001 + 1)) => 122µs

WakeUpCLKmax = RTCCLK/(16 x (0xFFFF + 1)) => 32s

RTCCLK = 32.768KHz
Resolution min=2xRTCCL

WakeUp configuration (2/3)

- *Case2* **WUCKSEL = 10x**



RTCCLK

Asynchrone 7bit Prescaler

ValueMax = div 2^7 = 128 (power-on reset value)
ValueMin = 1

Synchrone 15bit Prescaler

ValueMax = div 2^{15}
ValueMin = 1
Power-on reset value =256

ck_spre

Wake-Up

16bit autoreload Timer

ValueMax = 0xFFFF
ValueMin = 0x0000

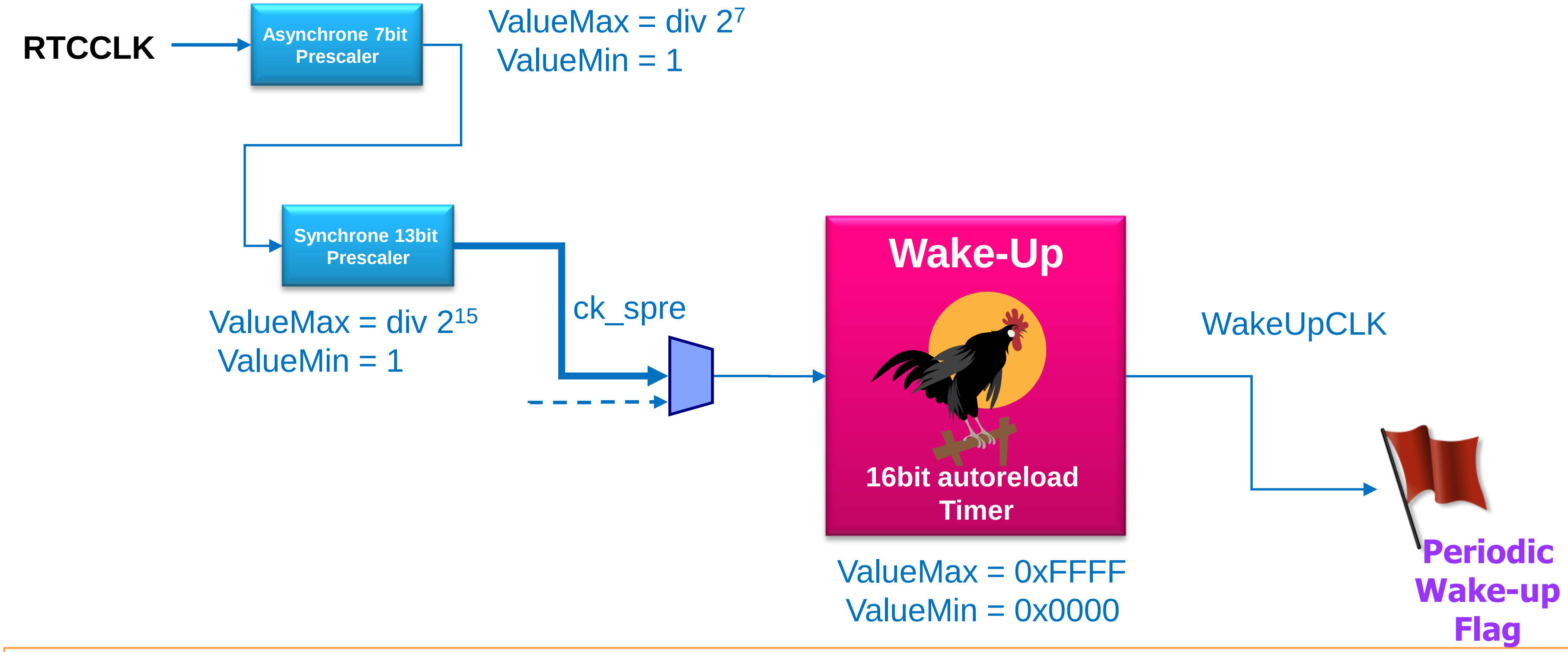
WakeUpCLK

Periodic Wake-up Flag

WakeUpCLKmin = RTCCLK/(1 x (0x0000 + 1))
WakeUpCLKmax = RTCCLK/(2^{22} x (0xFFFF + 1))
If ck_spre is 1Hz (when used for calendar): 1s <= WakeUpCLK <= 18.2h (1s resolution)

WakeUp configuration (3/3)

- *Case3* **WUCKSEL = 11x**



WakeUpCLKmin = RTCCLK/(1 x (0x10000 + 1))

WakeUpCLKmax = RTCCLK/(2^{22} x (0x1FFFF + 1))

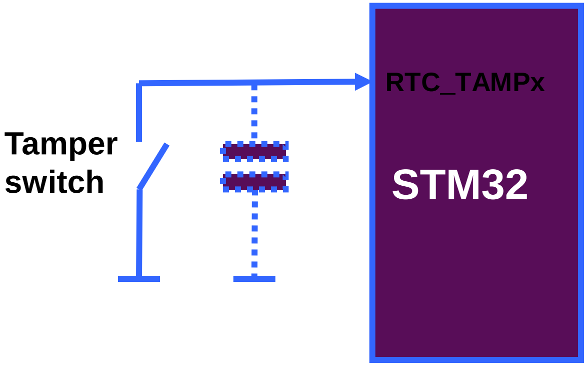
If ck_spre is 1Hz (when used for calendar): 18.2s <= WakeUpCLK <= 36.4h (1s resolution)

Smooth digital calibration

- Consists in masking/adding N (configurable) 32KHz clock pulses, fairly well distributed in a configurable window.
- A 1Hz output is provided to measure the quartz frequency and the calibration result.

Calibration window	Accuracy	Total range
8 s	**±1.91 ppm**	**[0 ±480ppm]**
16s	**±0.95 ppm**	**[0 ±480ppm]**
32s	**±0.48 ppm**	**[0 ±480ppm]**

Tamper detection



Capacitor is optional (filtering can be done by software)

Biasing is done using the I/O's Pull-up resistor

- 3 tamper pins and events
- Configurable active level for each event
- Configurable use of I/Os pull-up resistors
- Configurable pre-charging pulse to support different capacitance values
  - 1, 2, 4 or 8 cycles
- Configurable filter:
  - Sampling rate : 128Hz, 64Hz, 32Hz, 16Hz, 8Hz, 4Hz, 2Hz, 1Hz
  - Number of consecutive identical events before issuing an interrupt to wake-up the MCU : 1, 2, 4, 8

- RTC_TAMP1 available in VBAT mode.
- Reset of backup registers when tamper event detected
- Tamper event can generate a timestamp event

General Purpose Timers' section

(TIM2/3/4/5 - TIM12/13/14 - TIM15/16/17 - TIM6/7/18)

STM32F30x Timer features overview

	Counter resolution	Counter Type	Prescaler factor	DMA	Capture Compare channels	Synchronization	
						Master config	Slave config
General purpose **TIM2**	32 bit	Up, Down and Up/Down	1...65536	YES	4	YES	YES
General purpose **TIM3 and TIM4**	16 bit	Up, Down and Up/Down	1…65536	YES	4	YES	YES
Basic **TIM6 and TIM7**	16 bit	Up	1…65536	YES	0	YES	NO
1 channel, 1 complementary output **TIM16 and TIM17**	16 bit	Up	1…65536	NO	1	YES[1]	NO
2 channels, 1 complementary output **TIM15**	16 bit	Up	1…65536	NO	2	YES	YES

[1] TIM16 and TIM17 have no TRGO output, instead OC output is used

TIM2/5 | TIM3/4/19 | TIM12 | TIM15 | TIM13/14 | TIM16/17 | TIM6/7/18

life.augmented

COMPELFEST

STM32F37x Timer features overview

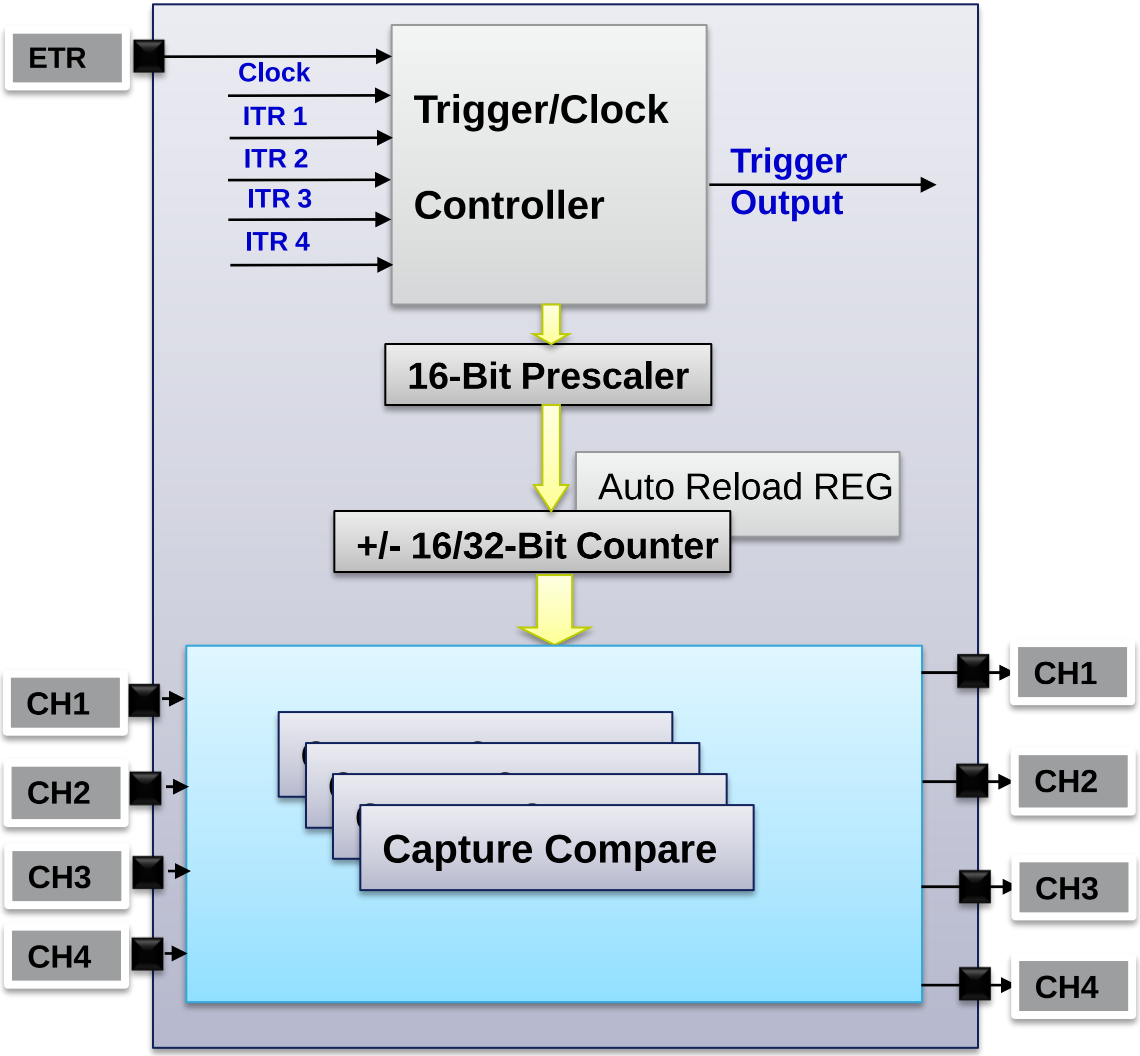
	Counter resolution	Counter Type	Prescaler factor	DMA	Capture Compare channels	Synchronization	
						Master config	Slave config
General purpose **TIM2 and TIM5**	32 bit	Up, Down and Up/Down	1...65536	YES	4	YES	YES
General purpose **TIM3, TIM4 and TIM19**	16 bit	Up, Down and Up/Down	1…65536	YES	4	YES	YES
Basic **TIM6, TIM7 and TIM18**	16 bit	Up	1…65536	YES	0	YES	NO
1 channel, 1 complementary output **TIM16 and TIM17**	16 bit	Up	1…65536	NO	1	YES[(1)]	NO
2 channels, 1 complementary output **TIM15**	16 bit	Up	1…65536	NO	2	YES	YES
1 channel **TIM13 and TIM14**	16 bit	Up	1…65536	NO	1	YES[(2)]	NO
2 channels **TIM12**	16 bit	Up	1…65536	NO	2	NO	YES

[(1)] TIM16 and TIM17 have no TRGO output, instead OC output is used
[(2)] TIM13 and TIM14 have no TRGO output, instead OC output is used

TIM2/5 TIM3/4/19 TIM12 TIM15 TIM13/14 TIM16/17 TIM6/7/18

Features overview (1/3)

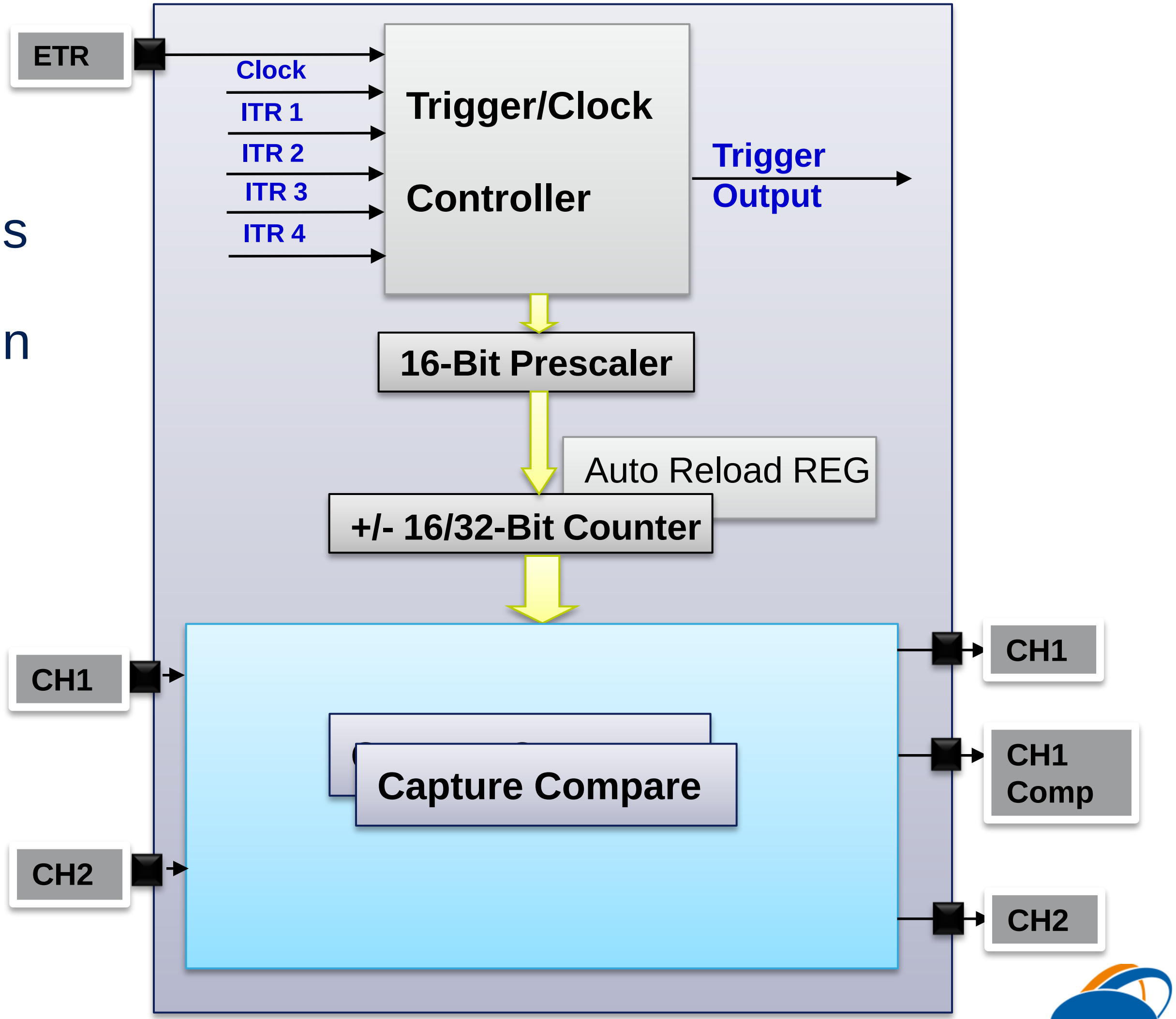
- Up to 4 16-bit resolution Capture Compare channels (TIM3/4/19)
- Up to 4 32-bit resolution Capture Compare channels (TIM2/5)
- Inter-timers synchronization
- Up to 6 IT/DMA Requests
- Encoder Interface
- Hall sensor Interface



TIM2/5 TIM3/4/19

Features overview (2/3)

- Up to 2 16-bit resolution Capture Compare channels
- Inter-timers synchronization
- Encoder Interface
- Only TIM15 has complementery output on channel1



TIM12 TIM15

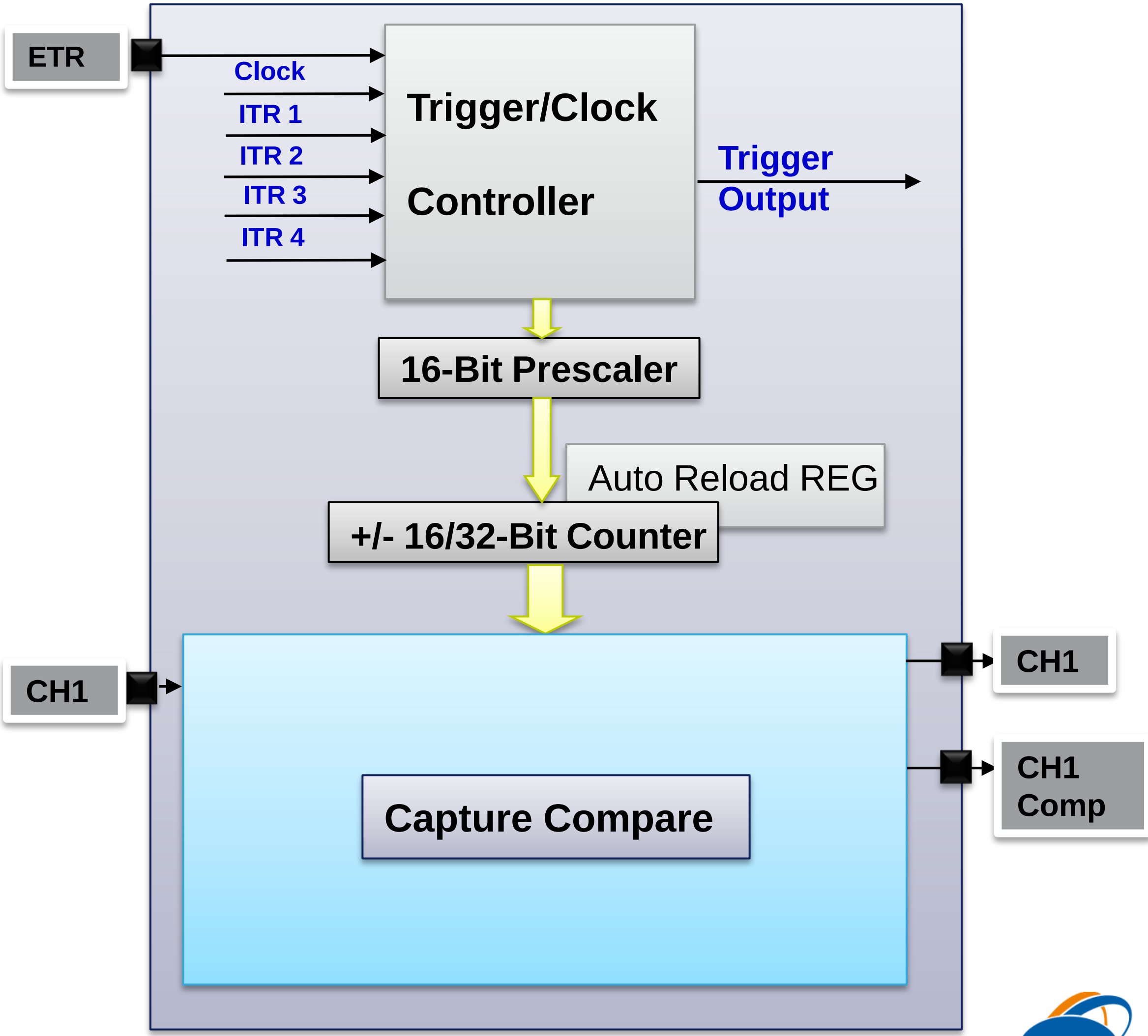
Features overview (3/3)

- One 16-bit resolution Capture Compare channels
- Only TIM16/17 has complementary output on channel 1

TIM2/5 TIM3/4/19 TIM12

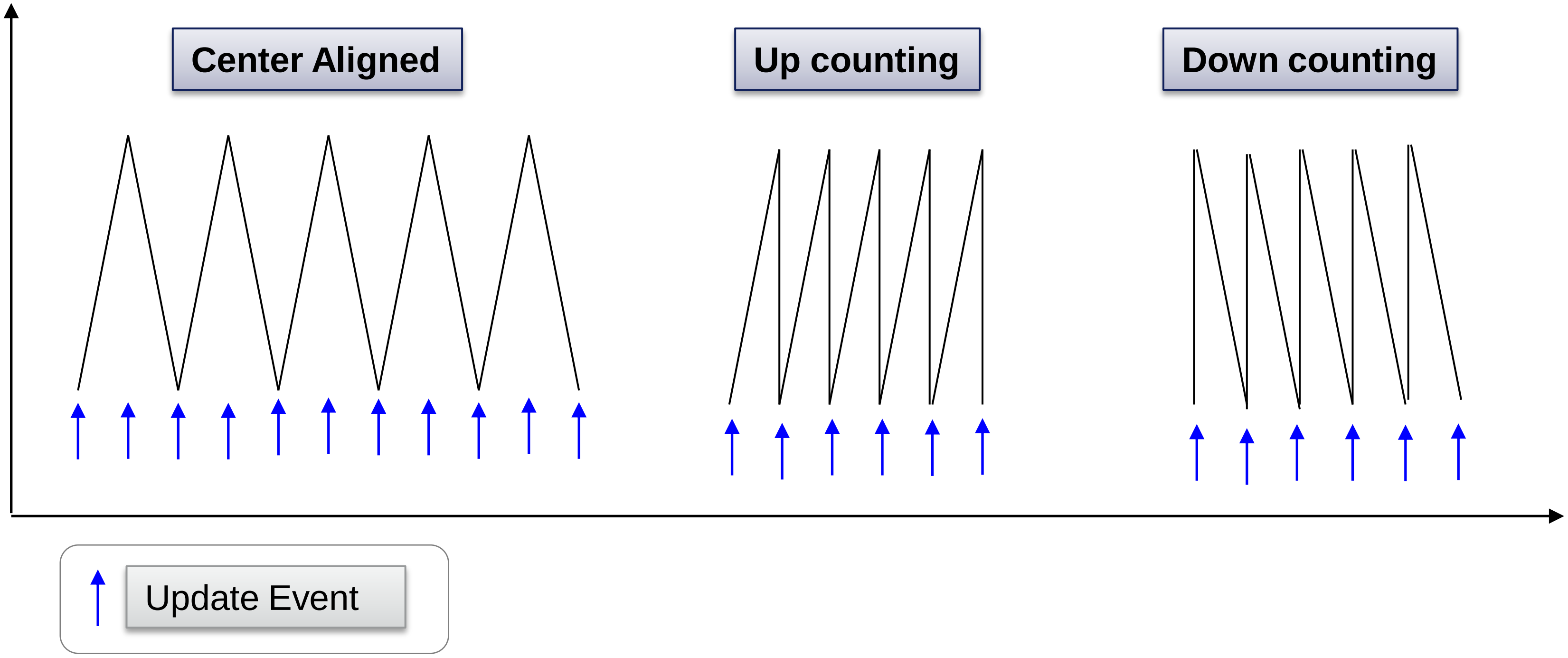
TIM15 TIM13/14 TIM16/17

TIM6/7/18



Counting Modes (1/2)

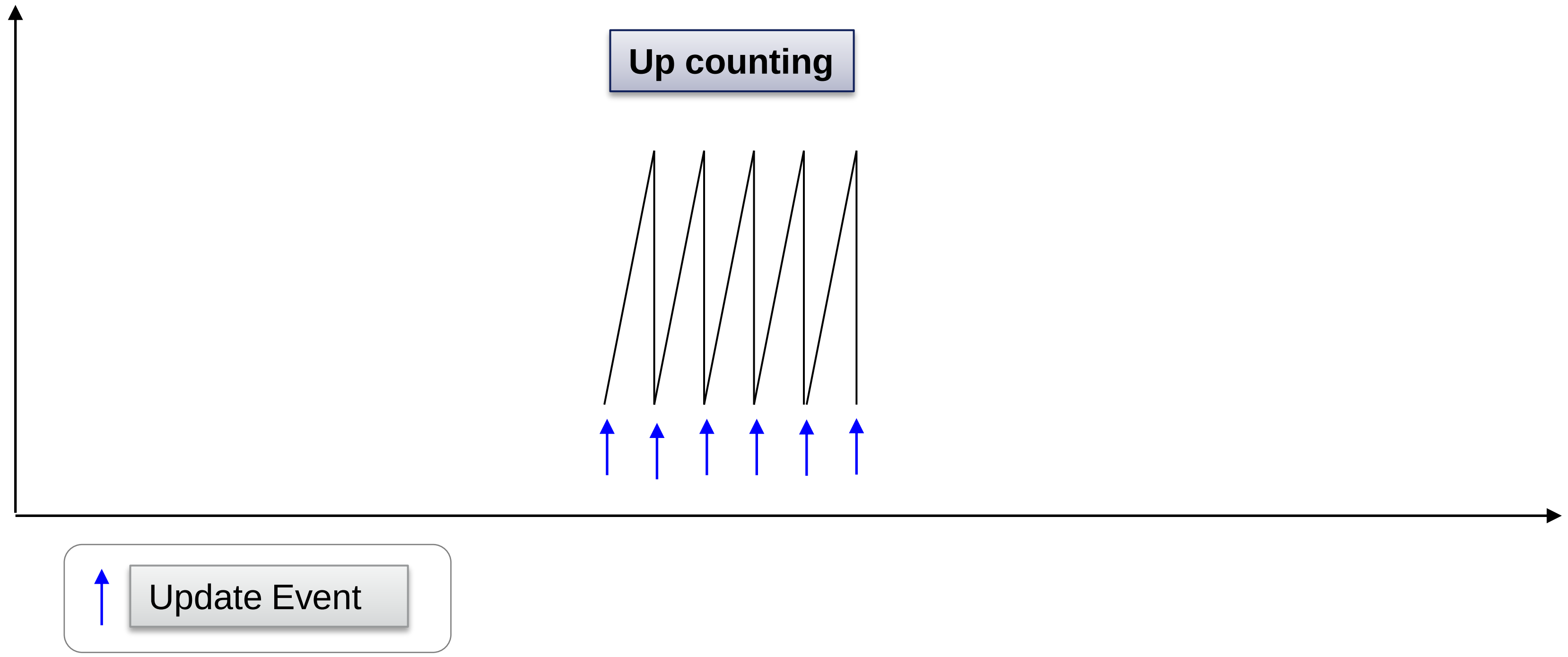
- There are three counter modes:
  - Up counting mode
  - Down counting mode
  - Center-aligned mode



TIM2/5 TIM3/4/19

Counting Modes (2/2)

- There is only one counting mode:
  - Up counting mode



TIM12 TIM15 TIM13/14 TIM16/17 TIM6/7/18

Update Event

- The content of the preload register is transferred into the shadow register
  - depends on the *Auto-reload Preload* feature if enabled or not
    - If enabled, at each Update Event the transfer occurs
    - If not enabled, the transfer occurs Immediately

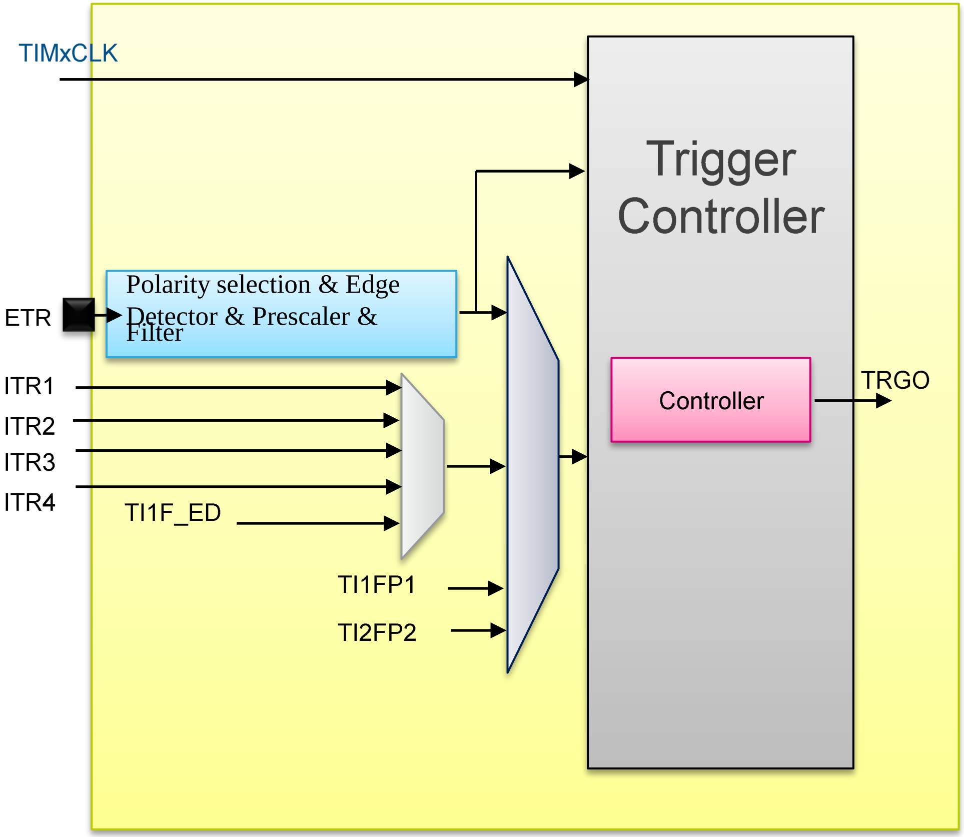
- The Update Event is generated
  - For each counter overflow/underflow
  - Through software, by setting the UG bit (Update Generation)

- The Update Event (UEV) request source can be configured to be
  - Next to counter overflow/underflow event
  - Nest to Counter overflow/underflow event plus the following events
    - Setting the UG bit by software
    - Trigger active edge detection (through the slave mode controller)

TIM2/5 TIM3/4/19 TIM12 TIM15 TIM13/14 TIM16/17

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



TIM2/5 TIM3/4/19 TIM12 TIM15

Capture Compare Array presentation

- Up to 4 channels
  - TIM2/3/4/5/19 have 4 channels
  - TIM12/15 have 2 channels
  - TIM13/14/16/17 have one channel
  - TIM6/7/18 have no channels

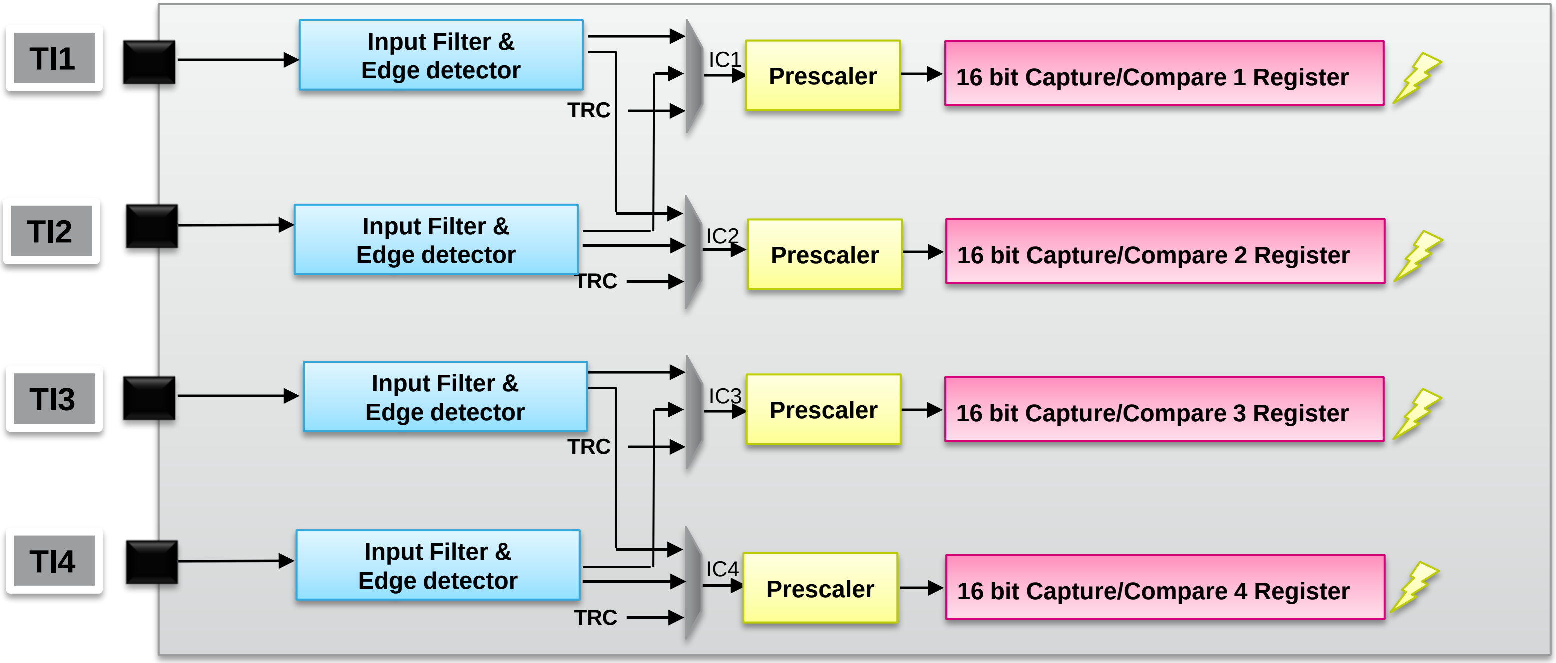
- Programmable bidirectional channels
  - Input direction: channel configured in **Capture** mode
  - Output direction: Channel configured in **Compare** mode

- Channel’s main functional blocs
  - Capture/Compare register
  - Input stage for capture
    - 4-bit digital filter
    - Input Capture Prescaler:
  - Output stage for Compare
    - Output control bloc

TIM2/5 TIM3/4/19 TIM12 TIM15 TIM13/14 TIM16/17

Input Capture Mode (1/2)

- Capture stage architecture


TI1
Input Filter & Edge detector
TRC
IC1
Prescaler
16 bit Capture/Compare 1 Register
TI2
Input Filter & Edge detector
TRC
IC2
Prescaler
16 bit Capture/Compare 2 Register
TI3
Input Filter & Edge detector
TRC
IC3
Prescaler
16 bit Capture/Compare 3 Register
TI4
Input Filter & Edge detector
TRC
IC4
Prescaler
16 bit Capture/Compare 4 Register

TIM2/5 TIM3/4/19

Input Capture Mode (2/2)

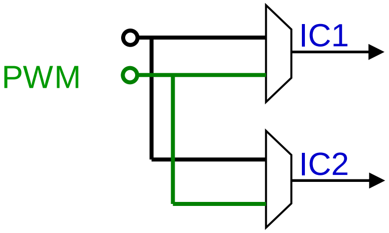
- Flexible mapping of TIx inputs to channels' inputs ICx
  - {TI1->IC1}, {TI1->IC2}, {TI2->IC1} and {TI2->IC2} are possible
- When an active Edge is detected on ICx input, the counter value is latched in the corresponding CCR register.
- When a Capture Event occurs, the corresponding CCXIF flag is set and an interrupt or a DMA request can be sent if they are enabled.
- An over-capture flag for over-capture signaling
  - Takes place when a Capture Event occurs while the CCxIF flag was already high

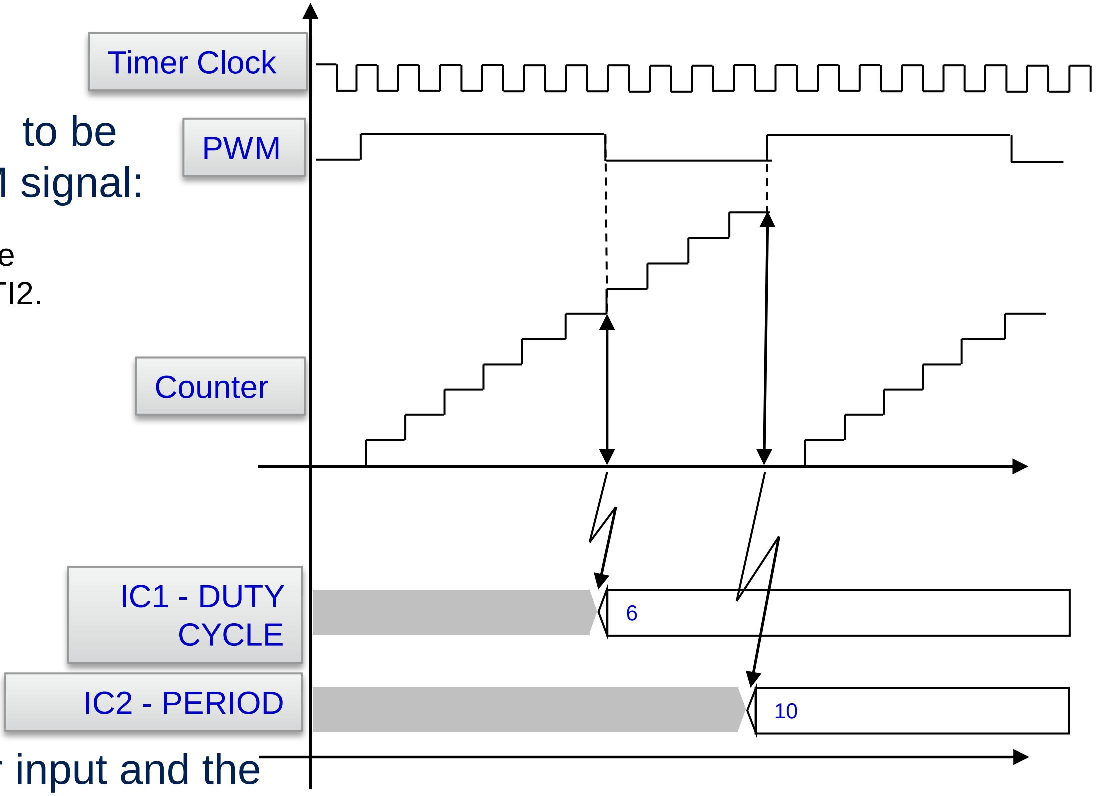
TIM2/5 TIM3/4/19

PWM Input Mode

- 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.





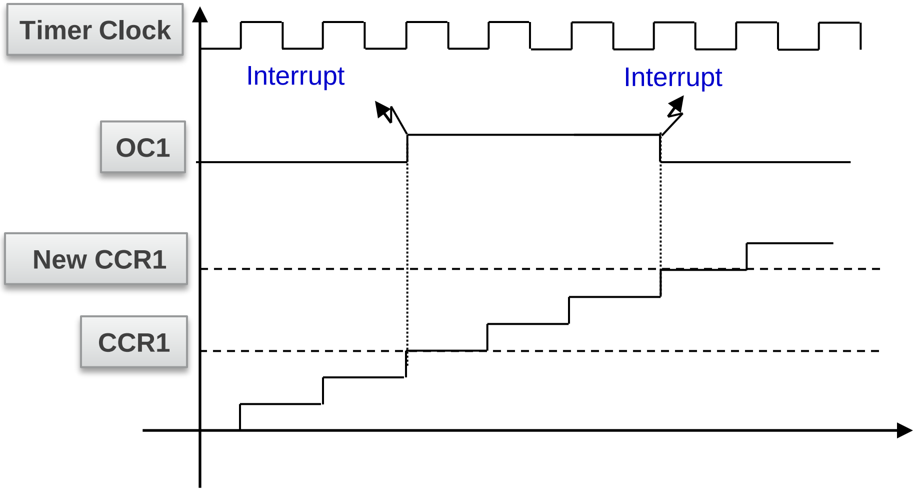
- IC1 and IC2 active edges must have opposite polarity.
- IC1 or IC2 is selected as trigger input and the slave mode controller is configured in reset mode.
- The PWM Input functionality enables the measurement of the period and the pulse width of an external waveform.

TIM2/5 TIM3/4/19 TIM12 TIM15

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

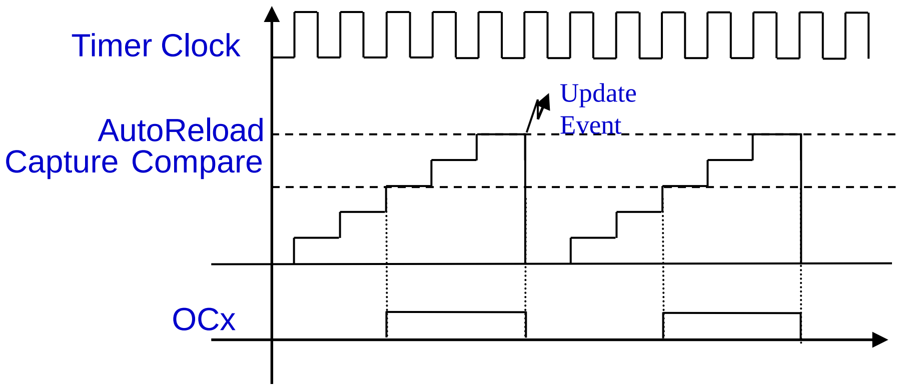


- The CCRx registers can be programmed with or without preload registers

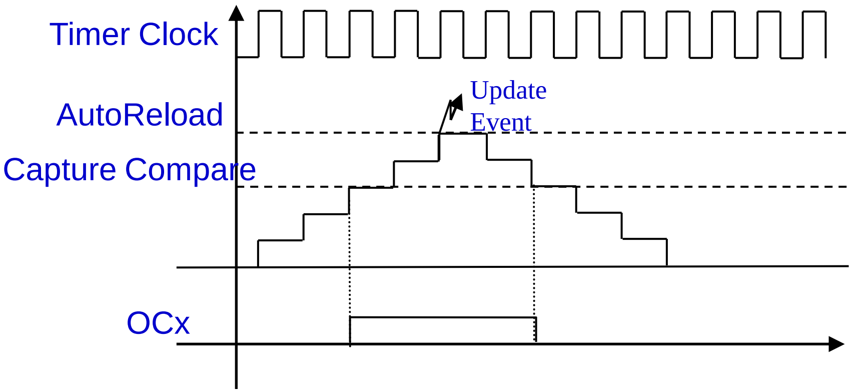
PWM Mode

- Available on all channels
- Two PWM mode available
  - PWM mode 1
  - PWM mode 2
  - Each PWM mode behavior (waveform shape) depends on the counting direction

Edge-aligned Mode



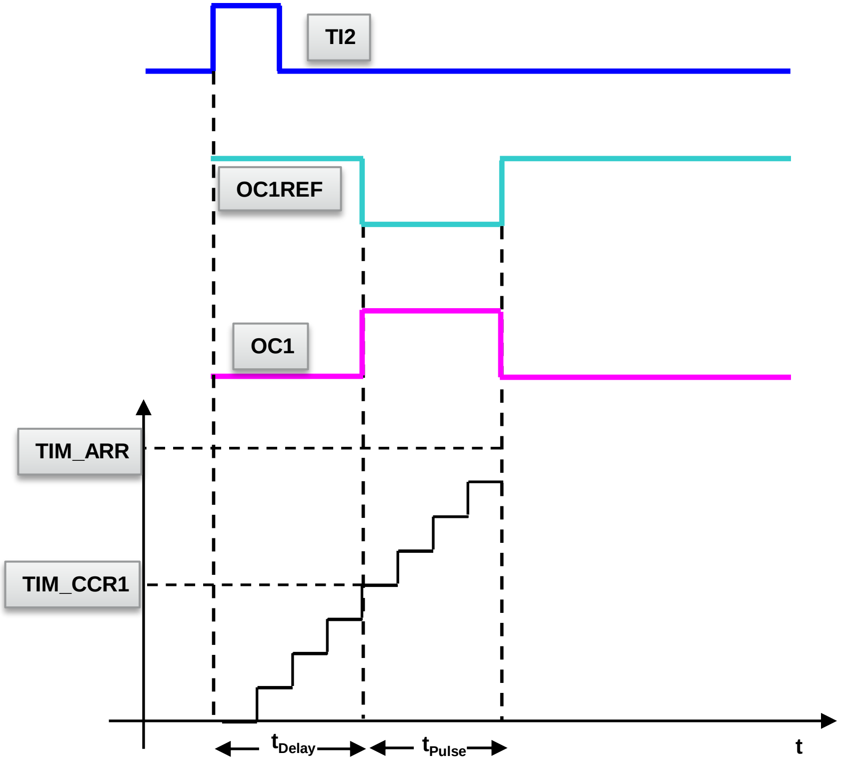
Center-aligned Mode



TIM2/5 | TIM3/4/19 | TIM12 | TIM15 | TIM13/14 | TIM16/17

One Pulse Mode (1/2)

- One Pulse Mode (OPM) is a particular case of Output Compare mode
- 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



TIM2/5 TIM3/4/19 TIM12 TIM15 TIM13/14 TIM16/17

One Pulse Mode (2/2)

Exercise:

How to configure One Pulse Mode to generate a repetitive Pulse in response to a stimulus ?

One Pulse Mode configuration steps

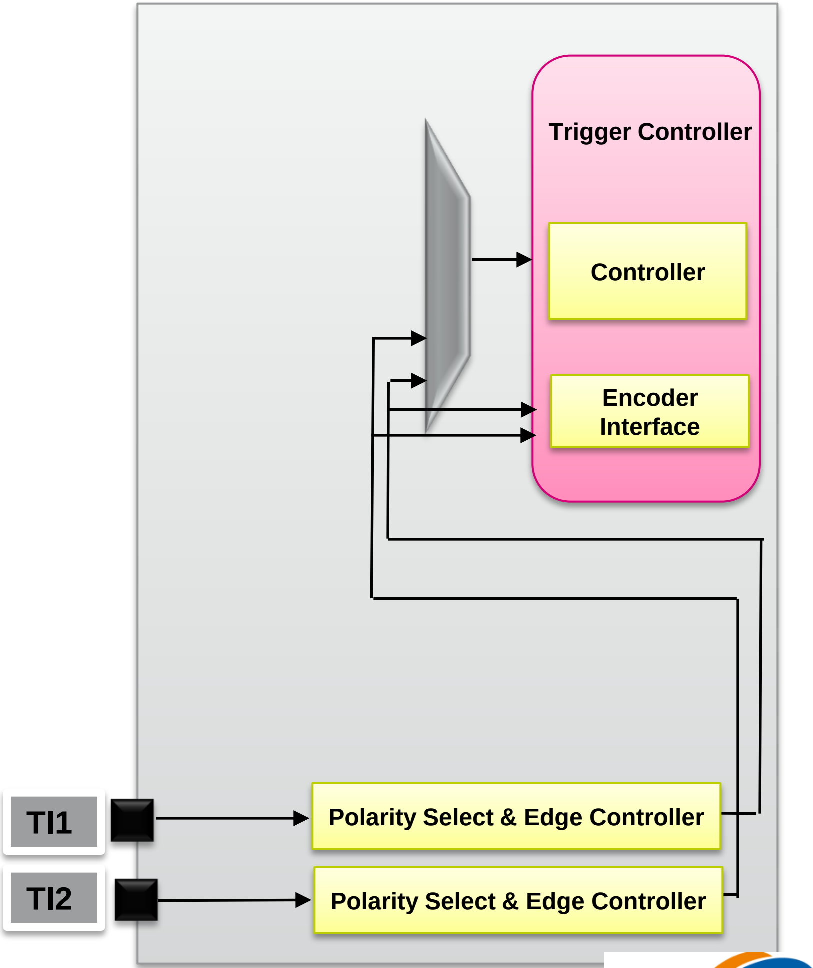
1. Input Capture Module Configuration:
   - i. Map TIxFPx on the corresponding TIx.
   - ii. TIxFPx Polarity configuration.
   - iii. TIxFPx Configuration as trigger input.
   - iv. TIxFPx configuration to start the counter (Trigger mode)

2. Output Compare Module Configuration:
   - i. OCx configuration to generate the corresponding waveform.
   - ii. OCx Polarity configuration.
   - iii. t_{Delay} and t_{Pulse} definition.

3. **One Pulse Module Selection: Set or Reset the corresponding bit (OPM) in the Configuration register (CR1).**

Encoder Interface (1/2)

- Encoders are used to measure position and speed of mobile systems (either linear or angular)
- The encoder interface mode acts as an external clock with direction selection
- Encoders and Microcontroller connection example:
  - A 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.
- Encoder enhancement
  - A copy of the Update Interrupt Flag (UIF) is copied into bit 31 of the counter register
  - Simultaneous read of the Counter value and the UIF flag : Simplify the position determination


Trigger Controller
Controller
Encoder Interface
TI1
TI2
Polarity Select & Edge Controller
Polarity Select & Edge Controller

TIM2/5 TIM3/4/19 TIM12 TIM15

Encoder Interface (2/2)

Exercise:

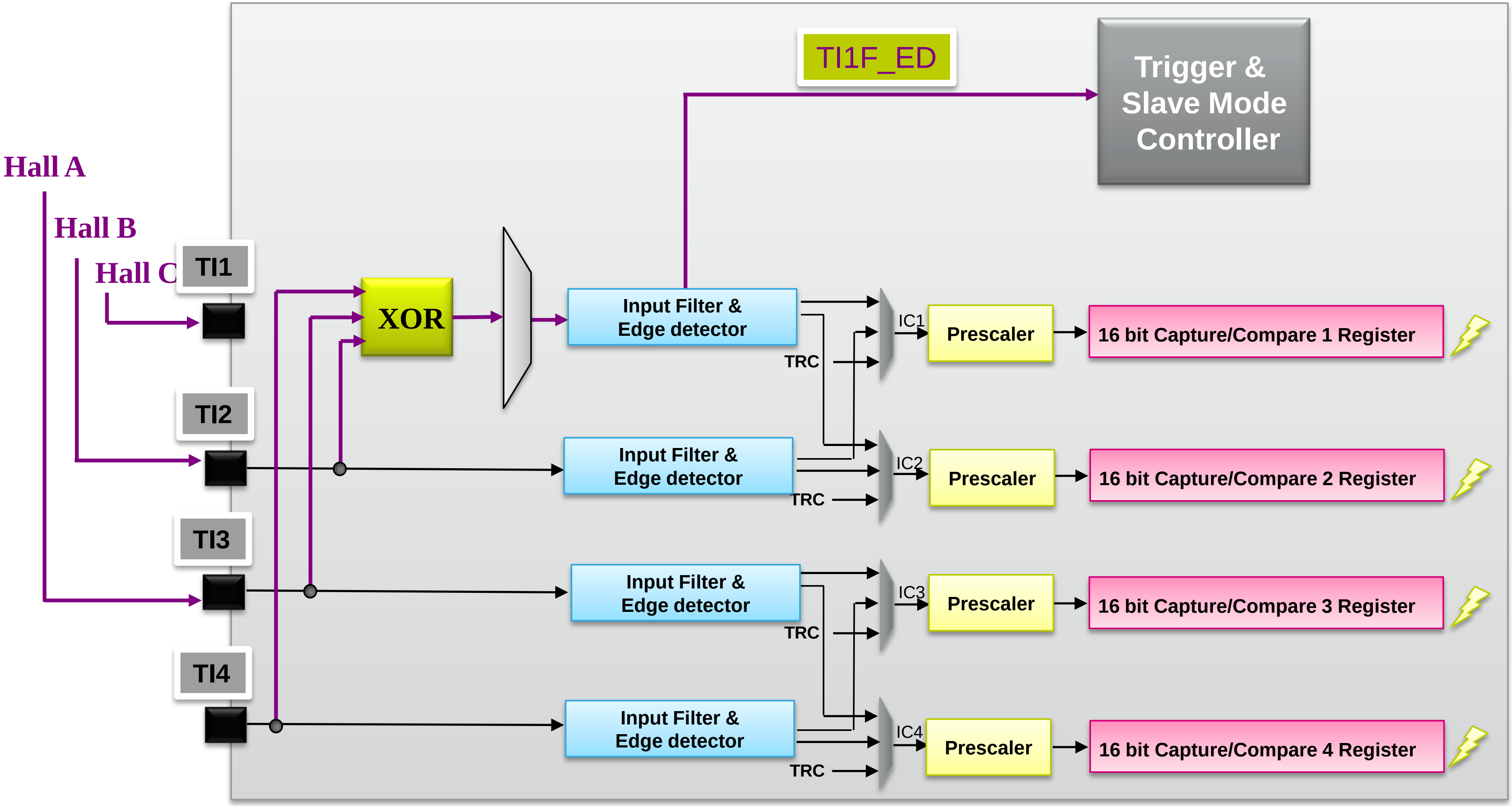
How to configure the Encoder interface to detect the rotation direction of a motion system?

Encoder interface configuration steps:

1. Select the active edges: example counting on TI1 and TI2.
2. Select the polarity of each input: example TI1 and TI2 polarity not inverted.
3. Select the corresponding Encoder Mode.
4. Enable the counter.

TIM2/5 TIM3/4/19 TIM12 TIM15

Hall sensor Interface (1/2)


TI1F_ED
Trigger & Slave Mode Controller
Hall A
Hall B
Hall C
TI1
TI2
TI3
TI4
XOR
Input Filter & Edge detector
Input Filter & Edge detector
Input Filter & Edge detector
Input Filter & Edge detector
TRC
TRC
TRC
TRC
IC1
IC2
IC3
IC4
Prescaler
Prescaler
Prescaler
Prescaler
16 bit Capture/Compare 1 Register
16 bit Capture/Compare 2 Register
16 bit Capture/Compare 3 Register
16 bit Capture/Compare 4 Register

TIM2/5 TIM3/4/19

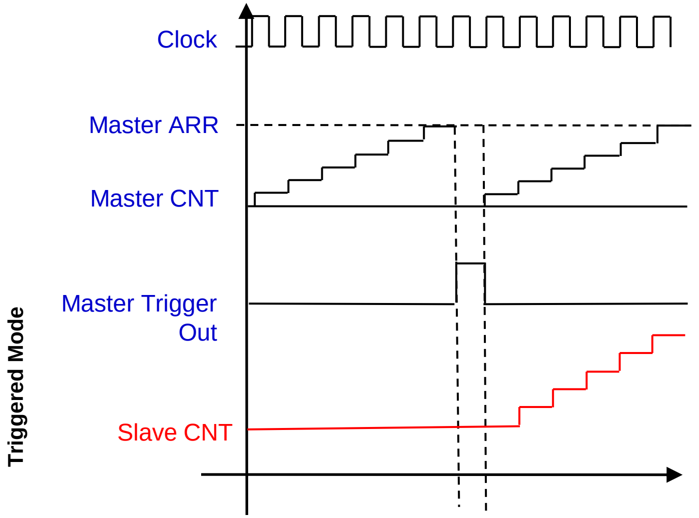
Hall sensor Interface (2/2)

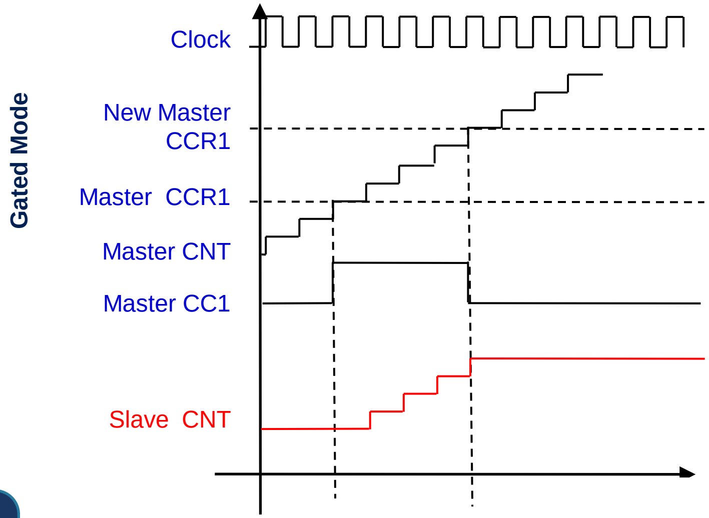
- Hall sensors are used for:
  - Speed detection
  - Position sensor
  - Brushless DC Motor Sensor
- How to configure the TIM to interface with a Hall sensor?
  - Select the hall inputs for TI1: TI1S bit in the CR2 register
  - The slave mode controller is configured in reset mode
  - TI1F_ED is used as input trigger
- To measure a motor speed:
  - Use the Capture/Compare Channel 1 in Input Capture Mode
  - The Capture Signal is the TRC signal
  - The captured value which correspond to the time elapsed between 2 changes on the inputs, gives an information about the motor speed

TIM2/5 TIM3/4/19

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
  - Reset Mode - Rising edge of the selected trigger input (TRGI) reinitializes the counter

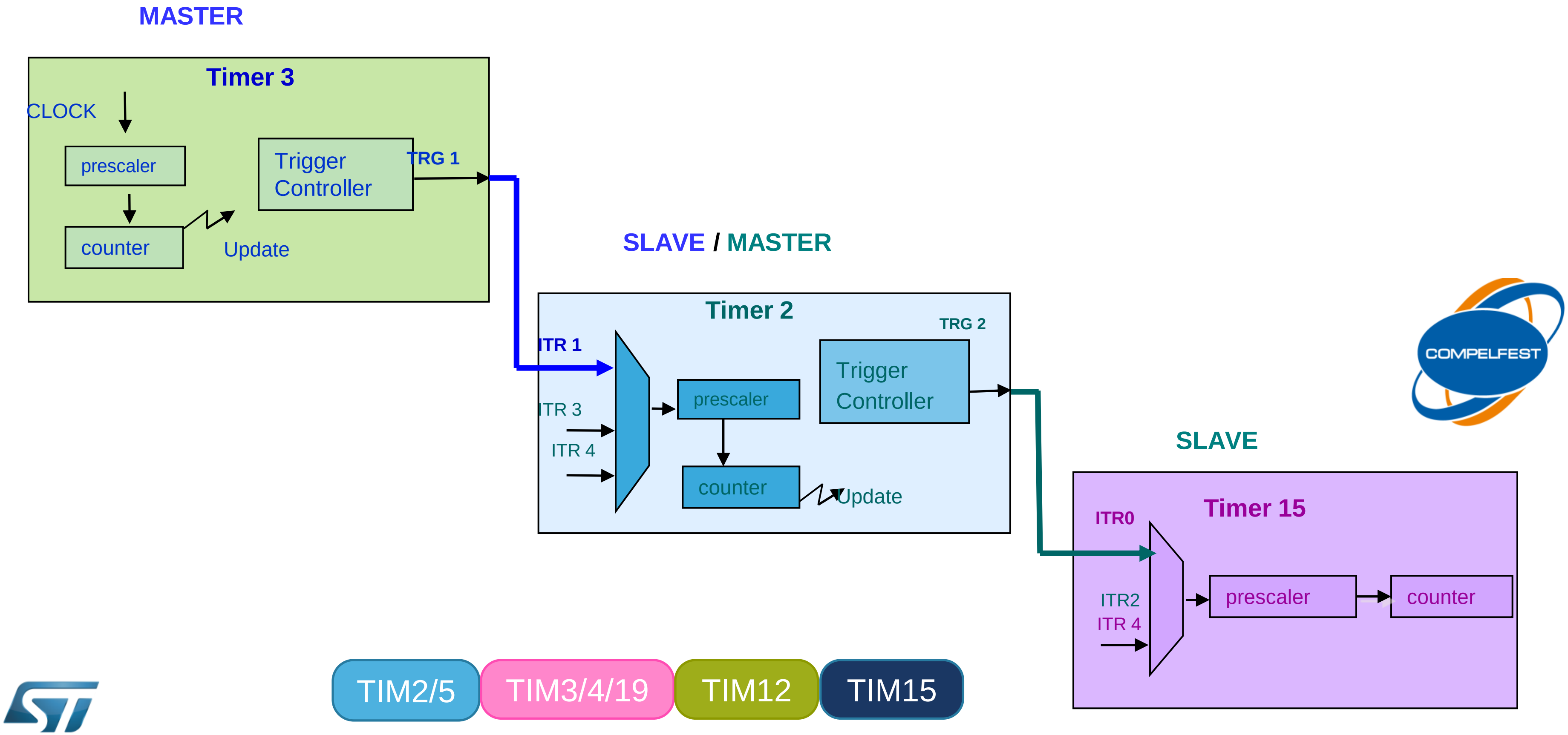




TIM2/5 TIM3/4/19 TIM12 TIM15

Synchronization: Configuration examples (1/3)

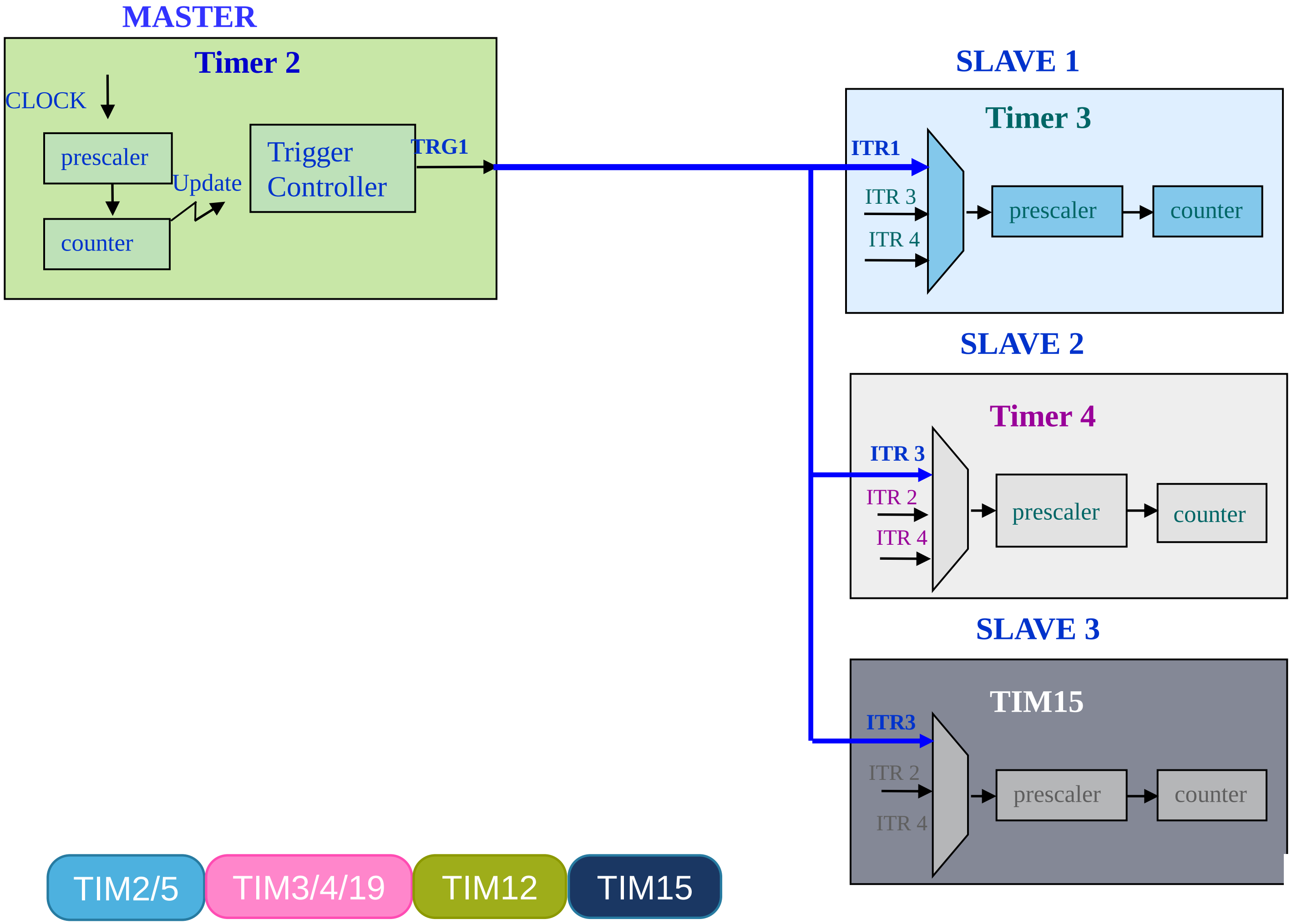
- Cascade mode:
  - TIM3 used as master timer for TIM2
  - TIM2 configured as TIM3 slave, and master for TIM15



TIM2/5 TIM3/4/19 TIM12 TIM15

Synchronization: Configuration examples (2/3)

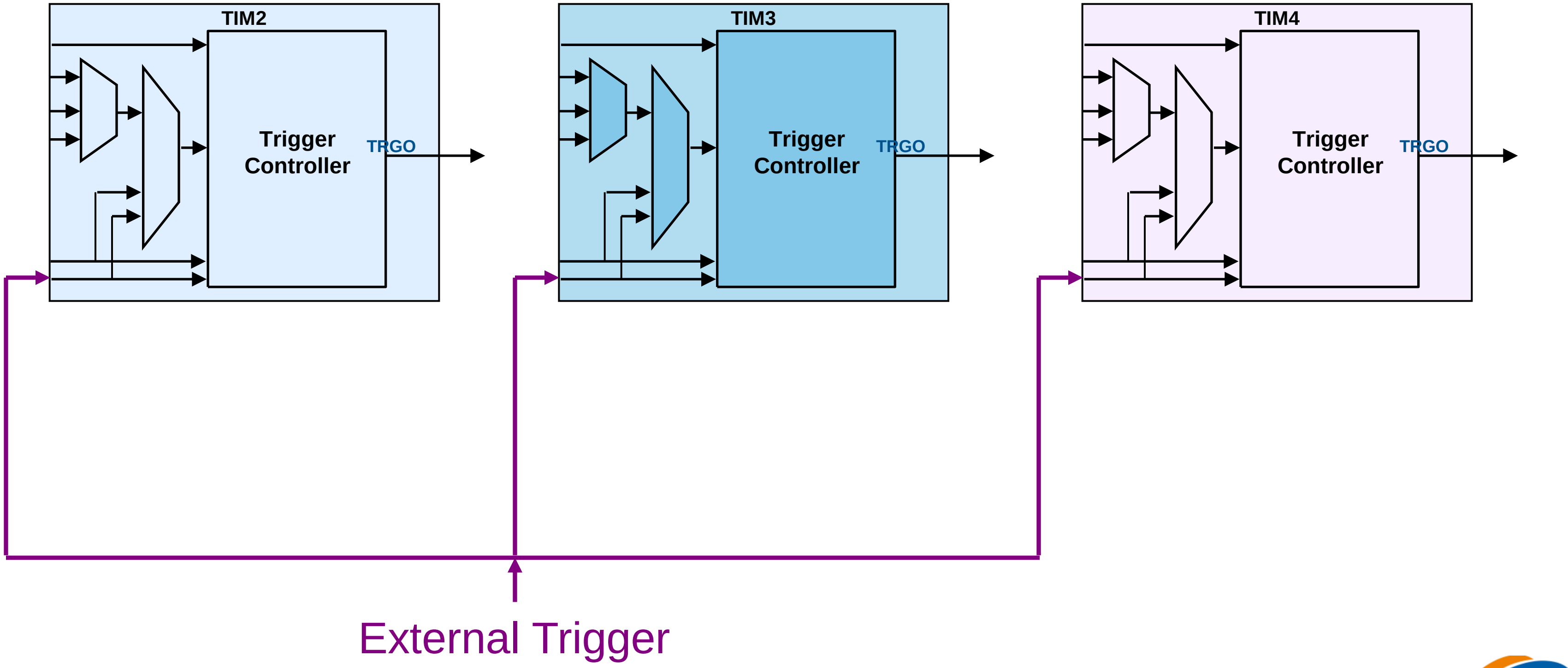
- One Master several slaves: TIM2 used as master for TIM3, TIM4 and TIM15



TIM2/5 TIM3/4/19 TIM12 TIM15

Synchronization: Configuration examples (3/3)

- Timers and external trigger synchronization
  - TIM2, TIM3 and TIM4 are slaves for an external signal connected to respective Timers inputs



Universal Serial Bus interface (USB Device)

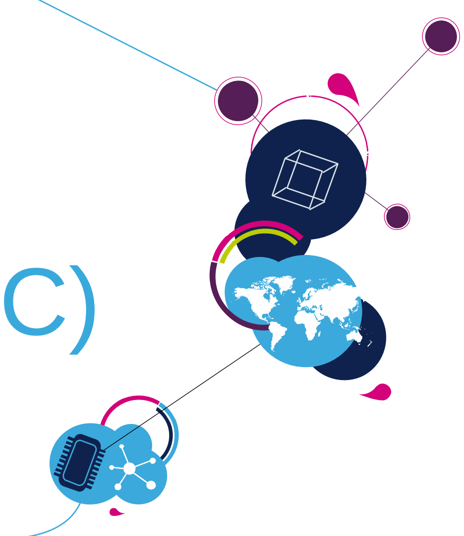
life.augmented

COMPELFEST

USB (same as STM32F10x)

- Full speed USB 2.0 transfer.
- Configurable endpoints transfer mode type: control, bulk, interrupt and Isochronous.
- Configurable number of endpoints: up to 8 bidirectional endpoints and 16 mono-directional endpoints.
- USB suspend/resume support.
- Dedicated SRAM Area (**P**acket **M**emory **A**rea) up to 512bytes **(!No longer shared with CAN).**
- Dynamic buffer allocation according to the user needs.
- Special double buffer support for Isochronous and Bu transfers.

Touch Sensing Controller (TSC)



Application -Microcontroller Division

MMS Group

life.augmented

COMPELFEST

TSC Features (1/2)

- Proven and robust surface charge transfer acquisition principle available on **STM32F05x**, **STM32F30x** and **STM32F37x** families
- Supports up to **24** capacitive sensing channels split over **8** analog I/O groups
  - Number of channels and analog I/O groups depend on the device used
- Up to **8** capacitive sensing channels can be acquired in parallel offering a very good response time
  - 1 counter per analog I/O group to store the current acquisition result
- One sampling capacitor for up to 3 capacitive sensing channels to reduce the system components
- Full hardware management of the charge transfer acquisition sequence
  - No CPU load during acquisition
- Spread spectrum feature to improve system robustness in noisy environ

TSC Features (2/2)

- Programmable charge transfer frequency
- Programmable sampling capacitor I/O pin
  - Any GPIO of an analog IO group can be used for the sampling capacitor
- Programmable channel I/O pin
  - Any GPIO of an analog IO group can be used for the channel
- Programmable max count value to avoid long acquisition when a channel is faulty
- Dedicated end of acquisition and max count error flags with interrupt capability
- Compatible with proximity, touchkey, linear and rotary touch sensors
- Designed to operate with STMTouch touch sensing firmware library

STM32F051x TSC Overview

- Supports up to **18** capacitive sensing channels split over **6** analog I/O groups
- **6.8 MHz** maximum charge transfer frequency

Analog I/O group	Number of capacitive sensing channels		
	STM32F051Rx	STM32F051Cx	STM32F051Kx
G1	3	3	3
G2	3	3	3
G3	3	2	2
G4	3	3	3
G5	3	3	3
G6	3	3	0
Number of capacitive sensing channels	**18**	**17**	**14**

STM32F302/303 TSC Overview

- Supports up to **24** capacitive sensing channels split over **8** analog I/O groups
- **10.2 MHz** maximum charge transfer frequency

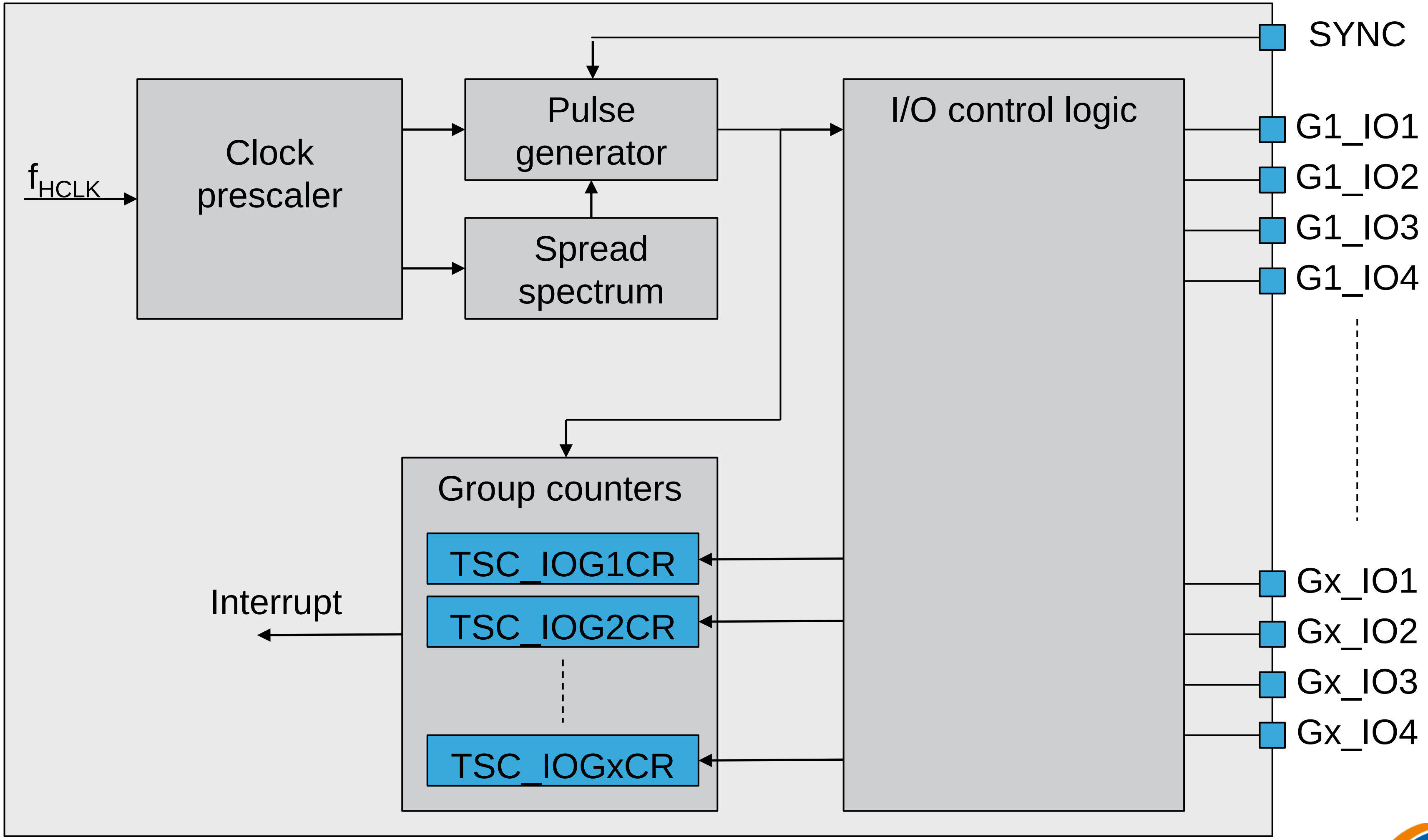
Analog I/O group	Number of capacitive sensing channels		
	STM32F30xVx	STM32F30xRx	STM32F30xCx
G1	3	3	3
G2	3	3	3
G3	3	3	2
G4	3	3	3
G5	3	3	3
G6	3	3	3
G7	3	0	0
G8	3	0	0
Number of capacitive sensing channels	**24**	**18**	**17**

STM32F372/373 TSC Overview

- Supports up to **24** capacitive sensing channels split over **8** analog I/O groups
- **10.2 MHz** maximum charge transfer frequency

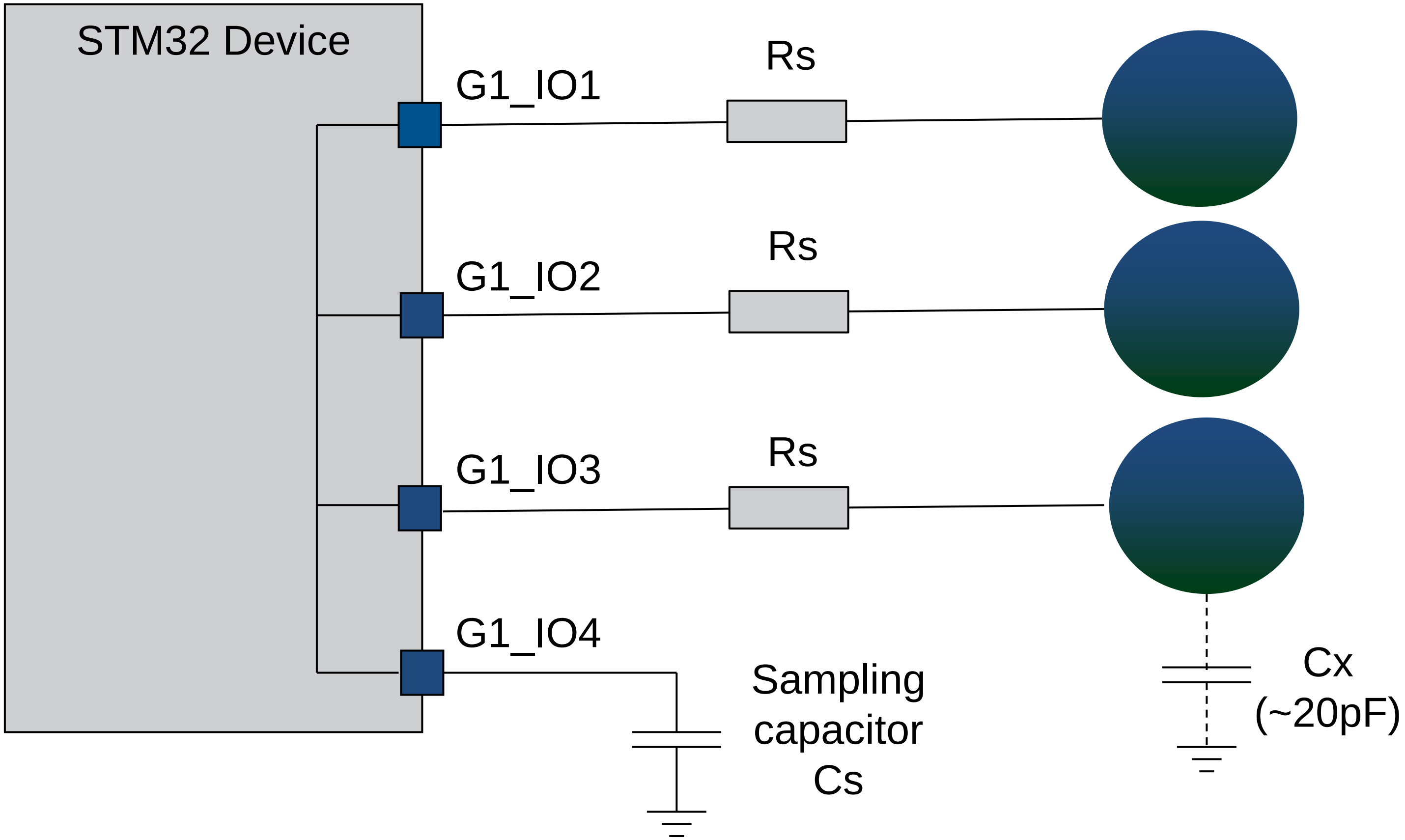
Analog I/O group	Number of capacitive sensing channels		
	STM32F37xVx	STM32F37xRx	STM32F37xCx
G1	3	3	3
G2	3	3	2
G3	3	3	1
G4	3	3	3
G5	3	3	3
G6	3	2	2
G7	3	0	0
G8	3	0	0
Number of capacitive sensing channels	**24**	**17**	**14**

TSC Block Diagram



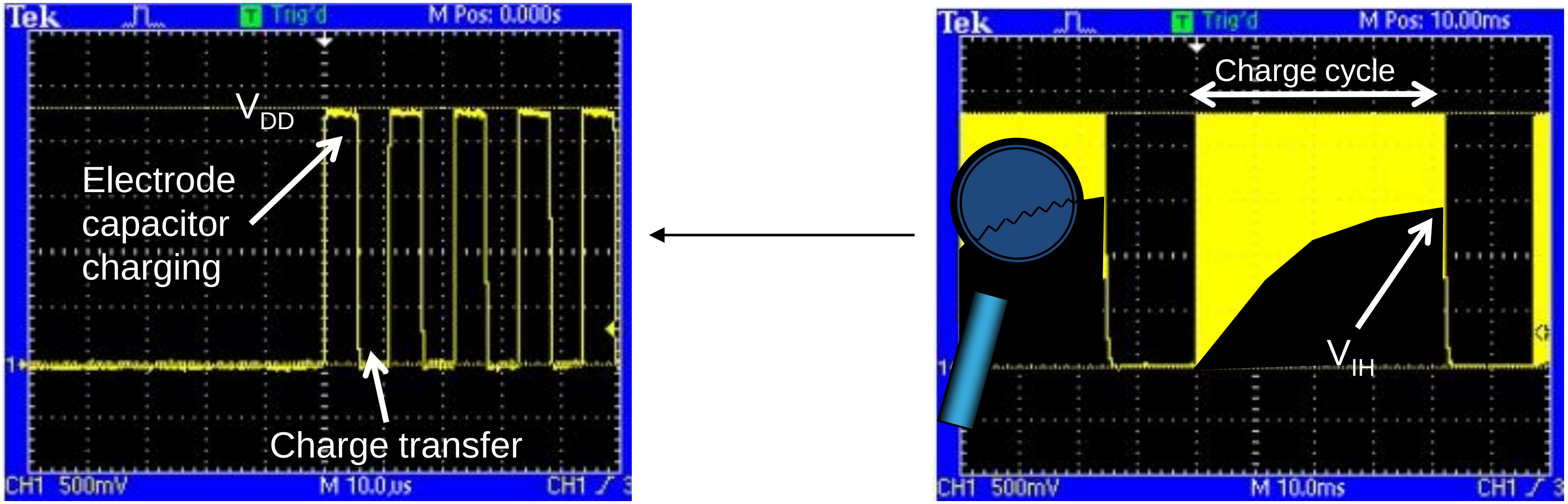
Charge Transfer Measuring Circuit

- Rs is used to improve ESD robustness (typically 10K)
- Cs sampling capacitor value depends on the required channels sensitivity
  - Higher Cs value is, higher the sensitivity but longer the acquisition time is

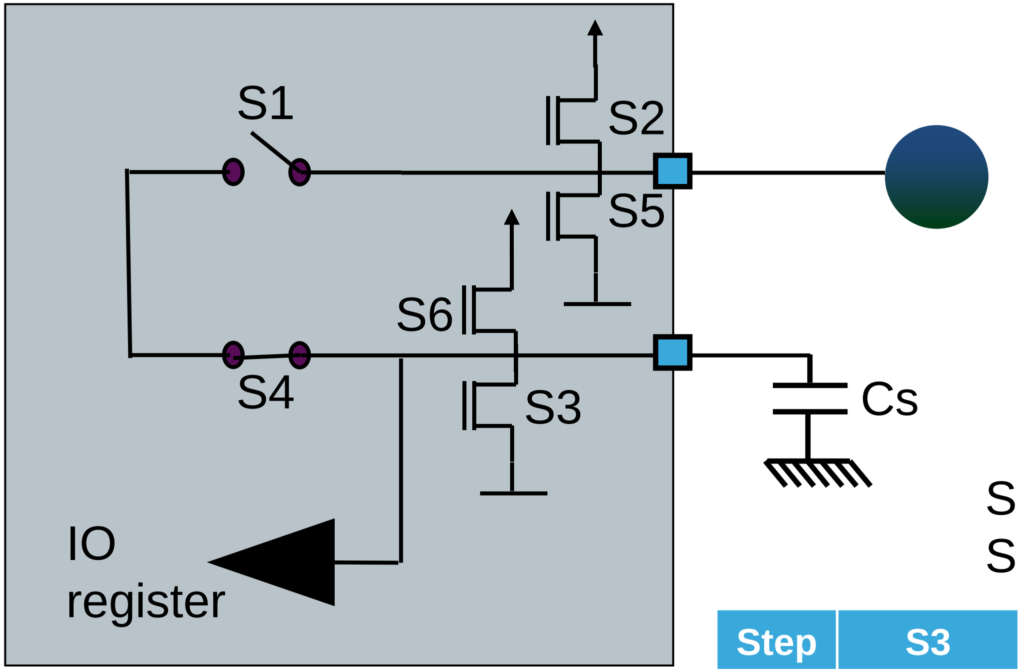


Charge Transfer Acquisition Overview

- Charge transfer uses the electrical properties of the capacitor charge Q
- It uses a **sampling capacitor (C_S)** in which the **electrode (C_X)** charges are transferred to
- Charge Transfer is performed through analog switches directly embedded into the GPIO
- The charge transfer cycle is repeated N times until the voltage on the sampling capacitor reaches the **V_{IH} threshold** of the GPIO it is connected to
- The number N of transfer cycles required to reach the threshold represents the size of Cx
  - The number of transfer decreases when the electrode is touched.



Charge Transfer Acquisition Sequence



S4 closed for the whole acquisition
S5 & S6 opened for the whole acquisition

Step	S3	S2	S1	Description
1	Closed	Opened	Closed	Cs discharge
2	Opened	Opened	Opened	Deadtime
3	Opened	Closed	Opened	Charge cycle (Cx charge)
4	Opened	Opened	Opened	Deadtime
5	Opened	Opened	Closed	Transfer cycle (charge transferred to Cs)
6	Opened	Opened	Opened	Deadtime
7	Closed	Opened	Closed	Cx discharge

Repeat until Vcs is read as a logical ‘1’

STMTouch Touch Sensing Library

- Complete free C source code library with firmware examples
- Multifunction capability to combine capacitive sensing functions with traditional MCU features
- Enhanced processing features for optimized sensitivity and immunity
  - Calibration, environment control system (ECS), debounce filtering , detection exclusion system (DxS), …
- Complete and simple API for status reporting and application configuration
- Touchkey, proximity, linear and rotary touch sensors support
- Compliant with MISRA
- Compliant with all STM32 C compilers
- STM32F051 support planned for end Q2 2012

GPIO Analog Switch and Hysteresis Control

- In addition to the management of charge transfer acquisition, the touch sensing controller provides a manual control of both the embedded analog switches and hysteresis of the GPIOs belonging to the analog I/O groups.

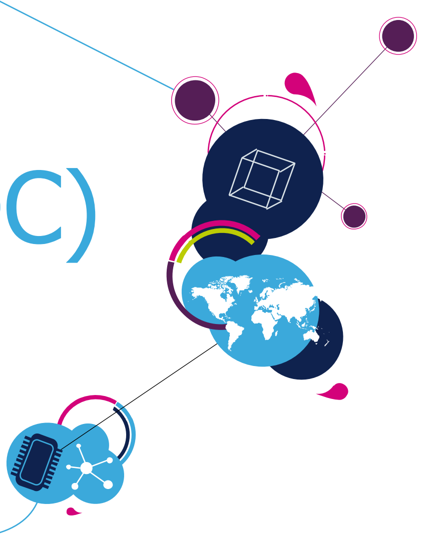
- This could be useful to implement a different capacitive sensing acquisition principle of for others purpose (ie: analog multiplexor).

More Information on Touch Sensing Solutions

- For further information on touch sensing solutions from MCD:
  - Visit the intranet: http://mcd.rou.st.com/modules.php?name=mcu&file=familiesdocs&FAM=118
  - Visit the sharepoint: http://gnbproject7mms.gnb.st.com/mcdappli/touchsensing/default.aspx
  - Attend to a dedicated training (please contact Thierry GUILHOT)

STM32F30x Specific features/peripherals

Analog-to-digital converter (ADC) 5MSPS



ADC Features (1/2)

- Up to 4 ADCs:
  - ADC1 & ADC2 are tightly coupled and can operate in dual mode (ADC1 is master)
  - ADC3 & ADC4 are tightly coupled and can operate in dual mode (ADC3 is master)
- Programmable Conversion resolution : 12, 10, 8 or 6 bit
- External Analog Input Channels for each of the 4 ADCs:
  - 5 fast channels from dedicated GPIOs pads
  - Up to 11 slow channels from dedicated GPIOs pads
- ADC conversion time:
  - Fast channels : up to 5.1Ms/s with 12 bit resolution in single mode
  - Slow channels: up to 4,8Ms/s with 12 bit resolution in single mode
- AHB Slave Bus interface
- Channel-wise programmable sampling time
- Self-calibration
- Configurable regular and injected channels
- Hardware assistant to prepare the context of the injected channels to allow fast context switching
- Can manage Single-ended or differential inputs

ADC Features (2/2)

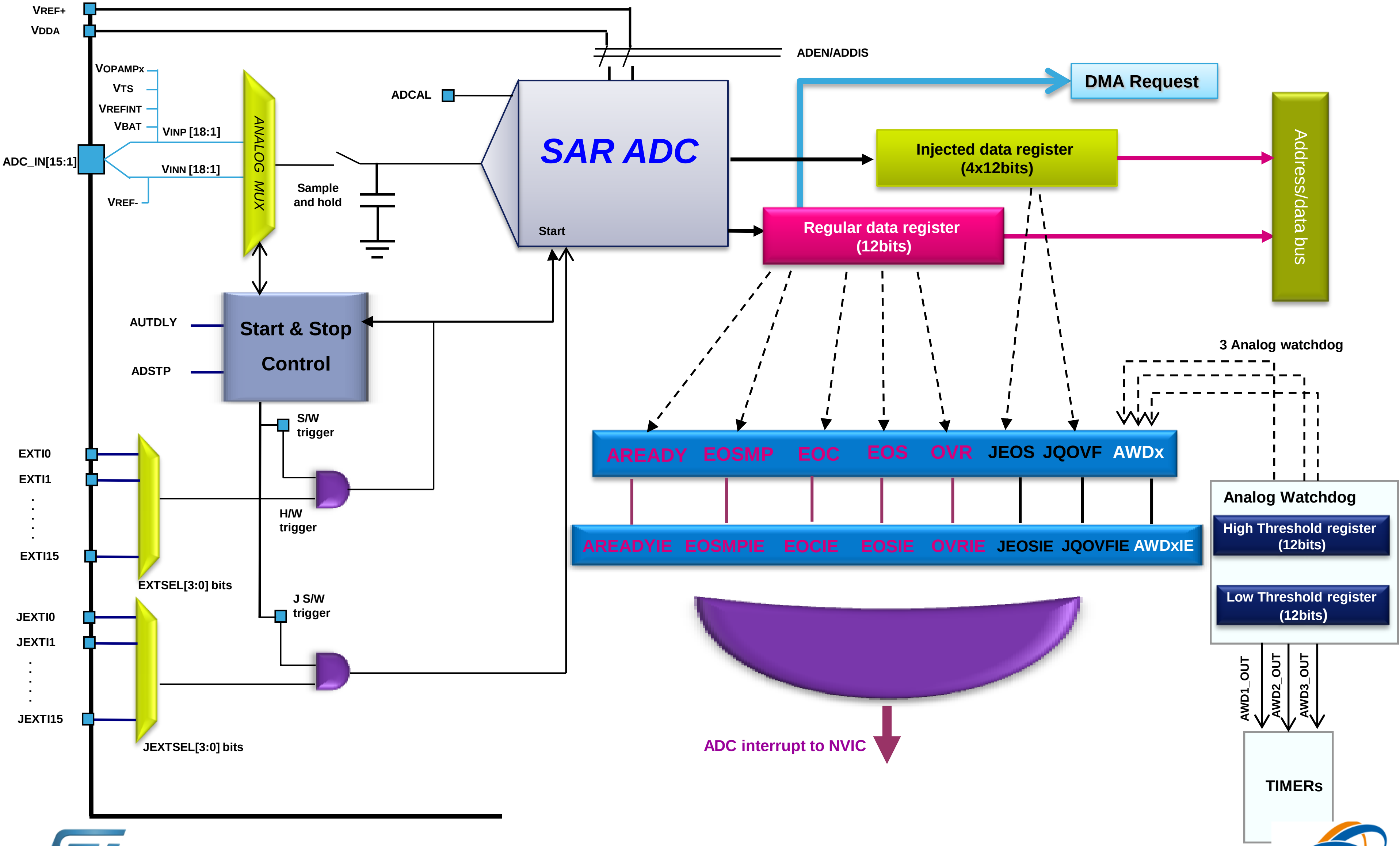
- 3 internal channels connected to :
  - Temperature sensor Vsense connected to ADC1
  - Internal voltage reference VREFINT connected to all ADCs
  - VBAT/2 power supply connected to ADC1
- Programmable sampling time
- Single, continuous and discontinuous conversion modes
- Dual ADC mode
- Left or right Data alignment with inbuilt data coherency
- Software or Hardware start of conversion
- 3 Analog Watchdog per ADC
- DMA capability
- Auto Delay insertion between conversions
- Interrupt generation

ADC Pins

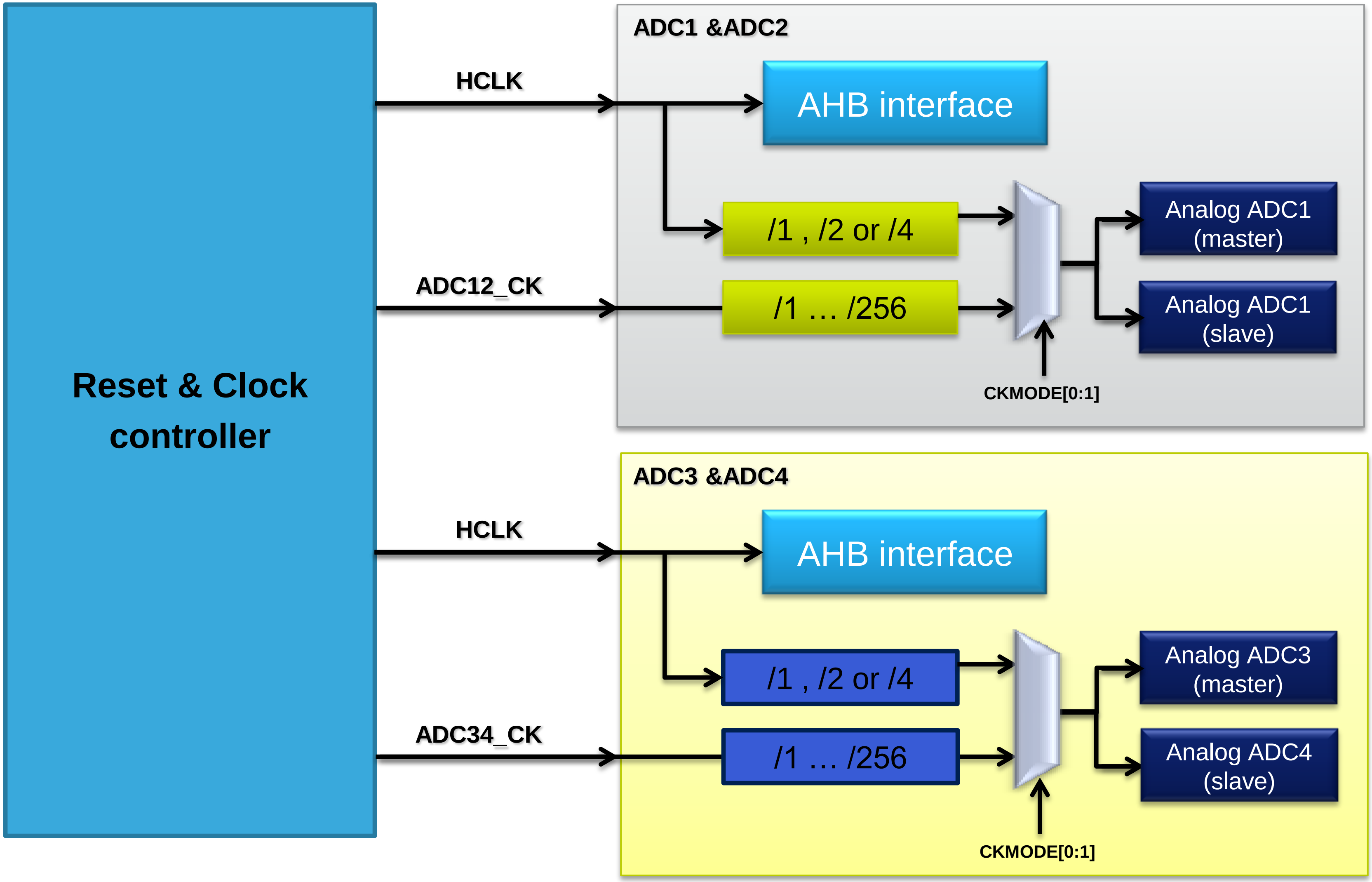
Name	Signal Type	Remarks
V_{REF+}	Input, analog reference positive	The higher/positive reference voltage for the ADC, 1.8 V ≤ V_{REF+} ≤ V_{DDA}
V_{DDA}	Input, analog supply	Analog power supply equal to V_{DD} and 1.8 V ≤ V_{DDA} ≤ V_{DD} (3.6 V)
V_{REF-}	Input, analog reference negative	The lower/negative reference voltage for the ADC, V_{REF-} = V_{SSA}
V_{SSA}	Input, analog supply ground	Ground for analog power supply equal to V_{SS}
V_{INP}[18:1]	Positive input analog channels for each ADC	Connected either to external channels: ADC_INi or internal channels.
V_{INN}[18:1]	Negative input analog channels for each ADC	Connected to V_{REF-} or external channels: ADC_INi-1
ADCx_IN16:1	External analog input signals	Up to 16 analog input channels (x=ADC number = 1,2,3 or 4): • 5 fast channels • 11 slow channels

life.augmented

ADC Block Diagram



ADC Clocks

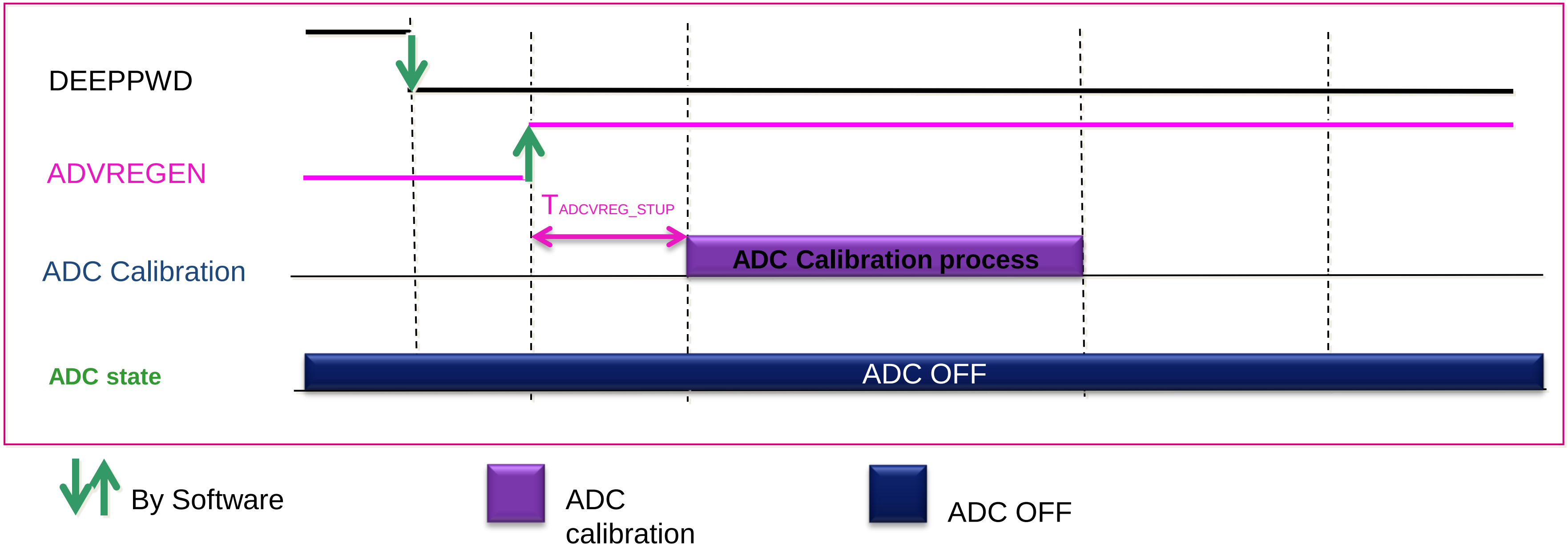

Reset & Clock controller
HCLK
ADC12_CK
HCLK
ADC34_CK
ADC1 &ADC2
AHB interface
/1 , /2 or /4
/1 … /256
CKMODE[0:1]
Analog ADC1 (master)
Analog ADC1 (slave)
ADC3 &ADC4
AHB interface
/1 , /2 or /4
/1 … /256
CKMODE[0:1]
Analog ADC3 (master)
Analog ADC4 (slave)

How to choose ADC Clock

ADC clock source	ADCxy_CK	AHB div 1, 2 or 4
Benefits	▪ Independent and asynchronous ADC clock versus AHB clock	▪ Bypassing the clock domain resynchronizations: deterministic latency between the trigger event and the start of conversion
Drawbacks	▪ Uncertainty of the trig instant is added by the resynchronizations between the two clock domains	▪ ADC clock depends on the AHB clock
Clock constraints when using injected channels	▪ FHCLK >= FADC/ 4 if the resolution of all channels are 12-bit or 10-bit ▪ FHCLK >= FADC/ 3 if there are some channels with 8 bits resolution ▪ FHCLK >= FADC/ 2 if there are some channels with 6 bits resolution	

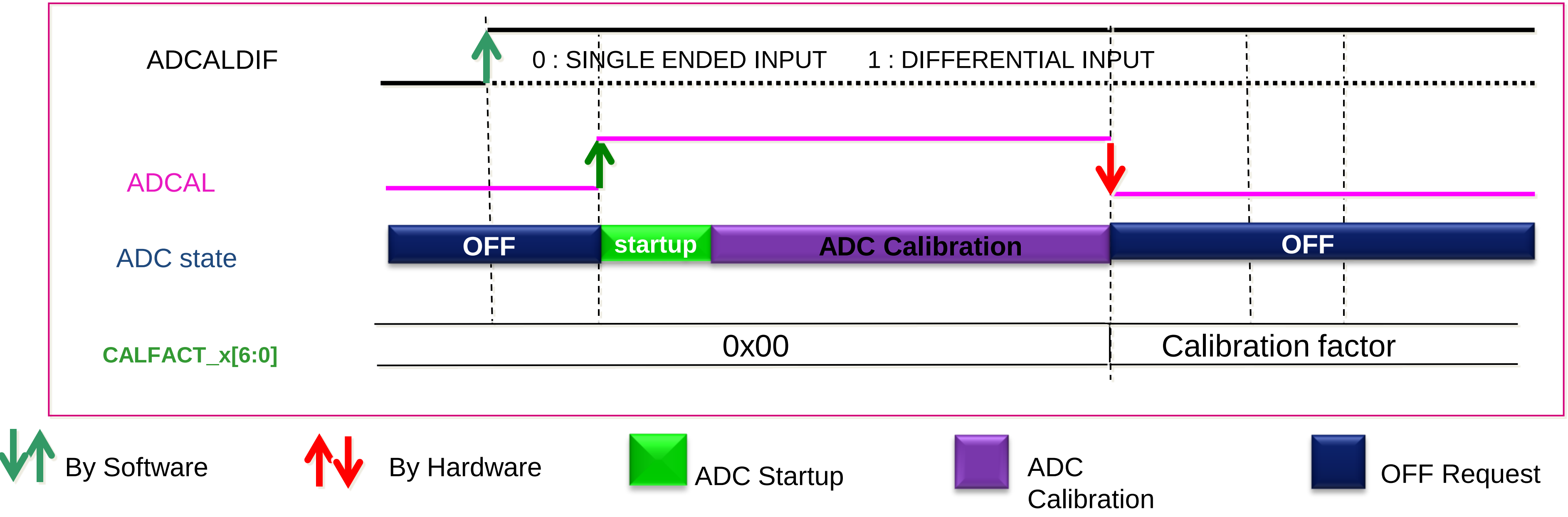
ADC Deep-Power-Down Mode

- By default, the ADC is placed in deep-power-down mode where its supply is internally switched off to reduce the leakage currents,
- To start ADC operations the following sequence should be applied:



ADC Calibration

- The calibration factor to be applied for single-ended input conversions is different from the factor to be applied for differential input conversions:
  - If ADCALDIF=0, calibration applied for single conversion and value stored in CALFACT_S
  - If ADCALDIF=1, calibration applied for differential conversion and value stored in CALFACT_D



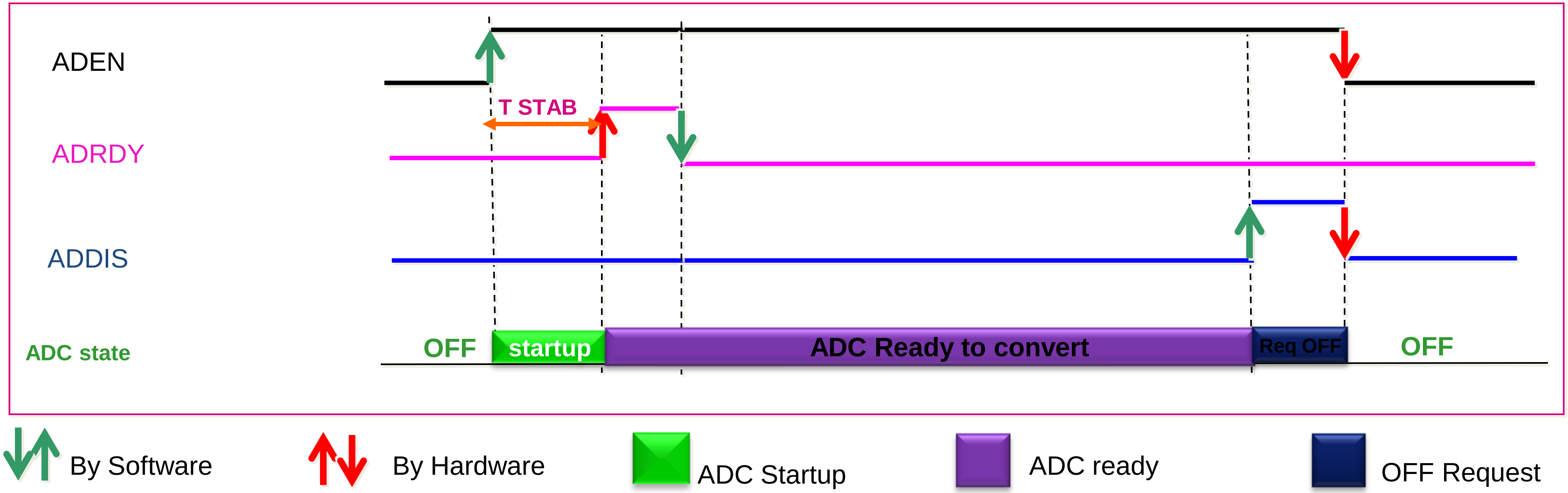
Note: The calibration factor is lost when entering Standby, Vbat mode or when the ADC enter deep power down mode. In this case it is possible to re-write the calibration factor into the ADC_CALFACT register without recalibrating.

ADC ON OFF control

- To enable ADC: Set ADEN=1 then wait till ADRDY flag will be equal to 1,

- What ever is the digital and the analog clock of the ADC, ADRDY signal guarantees that ADC data will be transmitted from one domain to the other.

- ADC cannot be re-programmed unless it is stopped (ADSTART = 0).



ADC Control bits constraints

- When ADEN is equal to 0, the software is allowed to write:
  - RCC control bits to configure and enable the ADC clock,
  - The control bits DIFSEL in the ADC_DIFSEL register,
  - The control bits ADCAL and ADEN in the ADC_CR register,

- When ADEN is equal to 1 and ADDIS to 0, the software is allowed to write:
  - ADSTART, JADSTART and ADDIS of the ADC_CR,
  - ADC_JSQR register

- When ADEN=1 and ADSTART = 0, the software is allowed to write:
  - All control bits related to configuration of regular conversions,

- When ADEN=1 and JADSTART = 0, the software is allowed to write:
  - All control bits related to configuration of injected conversions,

- When ADSTART=JARDSTART=1 and ADDIS=0, The software is allowed to write
  - ADSTP or JADSTP of the ADC_CR register.

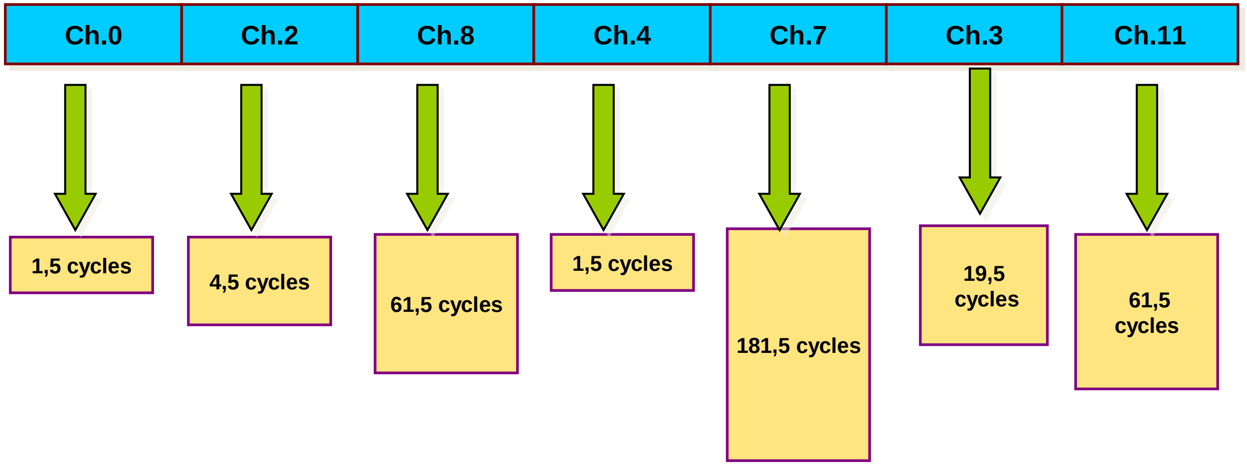
Note: There is no hardware protection to prevent these forbidden write accesses and ADC behavior may become in an unknown state. To recover from this situation, the ADC must be disabled (clear all ADC_CR register bits).

ADC Channel selection

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

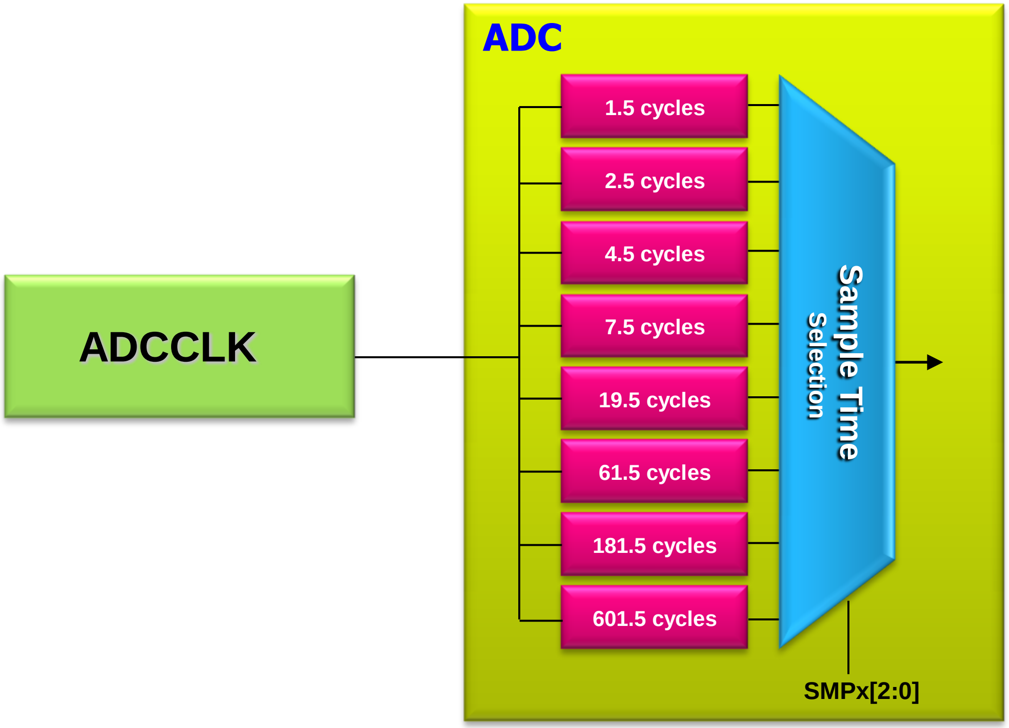
Example: - Conversion of channels: 0, 2, 8, 4, 7, 3 and 11

- Different sampling time.



ADC Sampling Time ($T_{Sampling}$)

- Three bits programmable sampling time channel by channel programmable:
  - 1.5 cycles
  - 2.5 cycles
  - 4.5 cycles
  - 7.5 cycles
  - 19.5 cycles
  - 61.5 cycles
  - 181.5 cycles
  - 601.5 cycles



Note: The sampling time value depends on the type of channel (fast or slow), the resolution and output impedance of the external signal source to be convert

Total Conversion Time

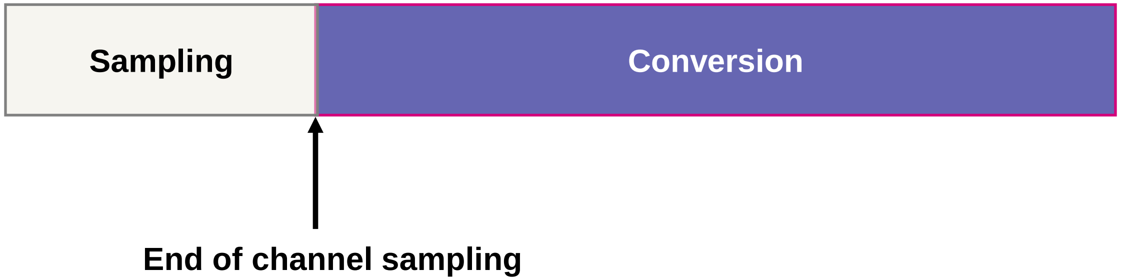
- Total conversion Time = $T_{Sampling}$ + $T_{Conversion}$

Resolution	$T_{Conversion}$
12 bits	12,5 Cycles
10 bits	10,5 Cycles
8 bits	8,5 Cycles
6 bits	6,5 Cycles

Resolution	Total conversion Time (When FADC = 72MHz)	
12 bits	12,5 + 1,5 = 14cycles	19.4 us → 5,1 Msps
10 bits	10,5 + 1,5 = 12 cycles	16,6 us → 6 Msps
8 bits	8,5 + 1,5 = 10 cycles	13,8 us → 7,2 Msps
6 bits	6,5 + 1,5 = 8 cycles	11,1 us → 9 Msps

End of sampling

- The ADC indicates the end of sampling phase by setting the EOSMP flag only for regular conversion.
- The EOSMP flag is cleared by software by writing1 to it.
- An interrupt can be generated if the EOSMPIE bit is set in the ADC_IER register.



➔As soon as the sampling is completed it is possible to prepare next conversion (for instance switching I/Os) during the conversion phase.

Single-ended & Differential input channels

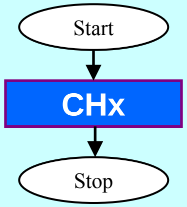
- Channels can be configured to be either single-ended or differential input by writing ADC_DIFSEL register:
  - In single ended input mode, the analog voltage to be converted for channel “i” is the difference between the external voltage ADC_INi (positive input) and VREF- (negative input)
  - In differential input mode, the analog voltage to be converted for channel “i” is the difference between the external voltage ADC_INi (positive input) and ADC_Ini+1 (negative input)

Note 1: When configuring the channel “i” in differential input mode, channel “i+1” is no longer usable in single-ended mode or in differential mode and must never be configured to be converted.

ADC conversion modes

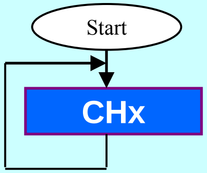
Single channel

single conversion mode



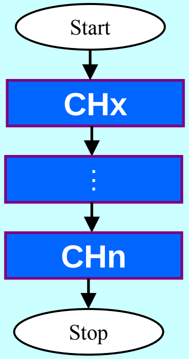
Single channel

Continuous conversion mode



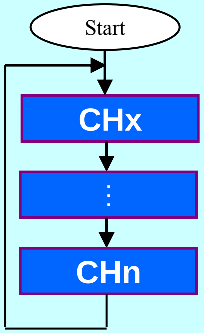
Multi-channels (Scan)

single conversion mode

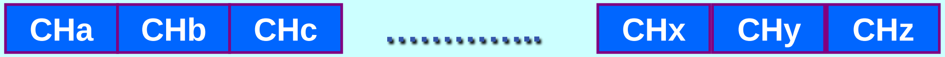


Multi-channels (Scan)

continuous conversion mode

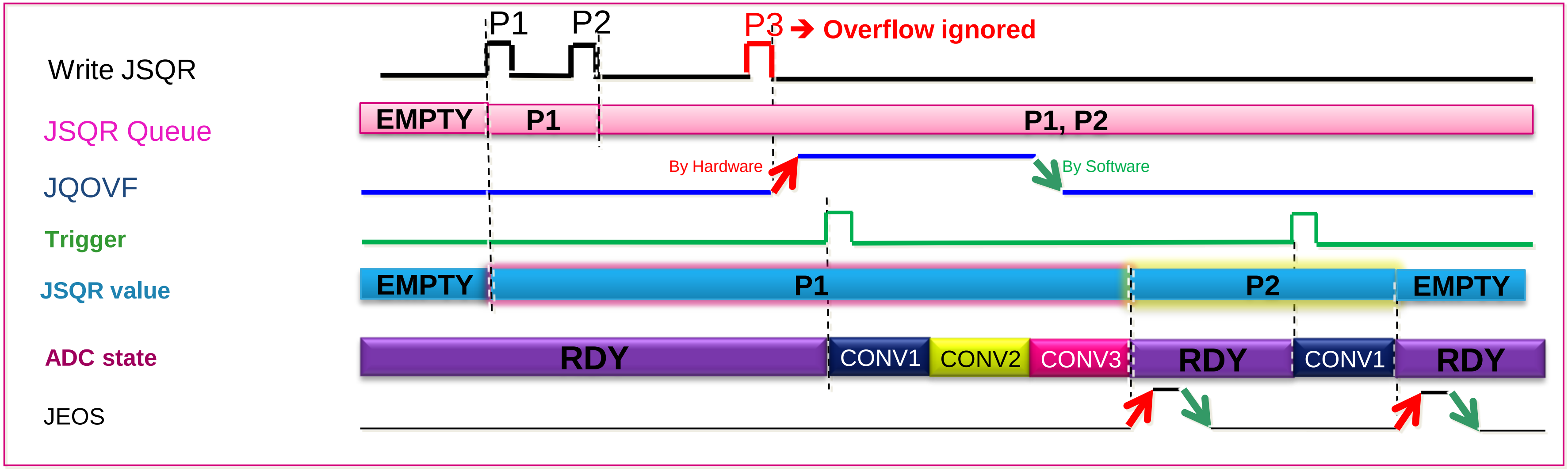


Discontinuons conversion mode



Queue overflow

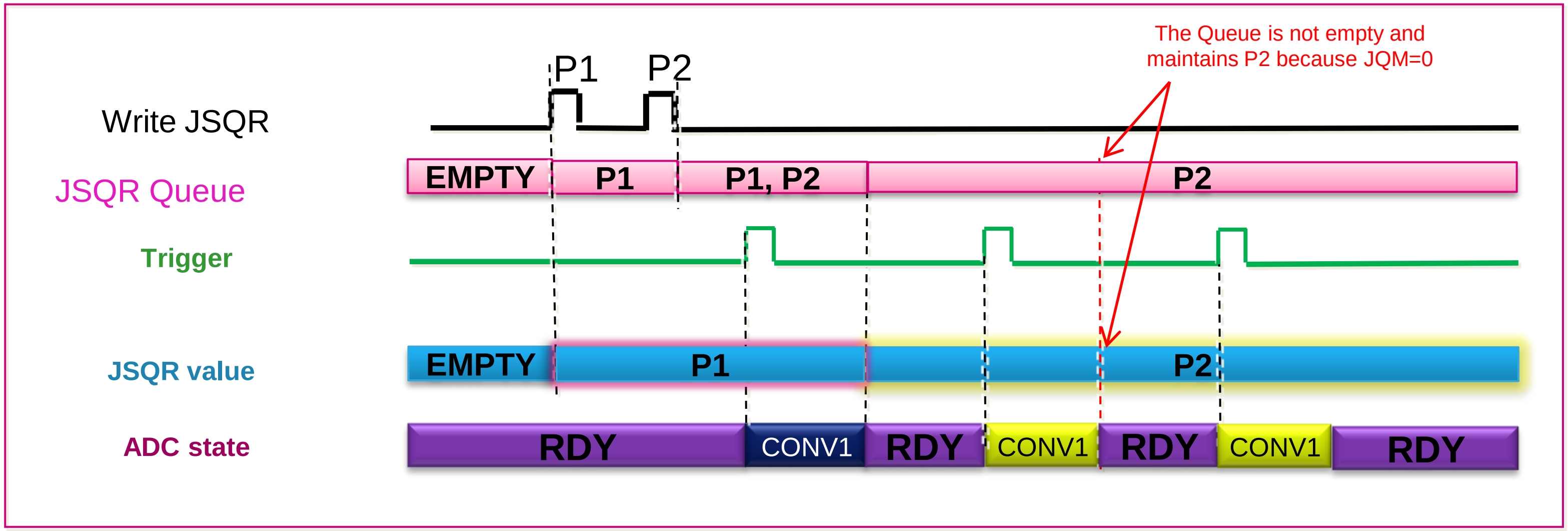
- A Queue overflow occurs when writing into JSQR register while the Queue is full,
- This overflow is signaled by the assertion of the JQOVF flag,
- When an overflow occurs, the write access of JSQR register which has created the overflow is ignored and the queue of context is unchanged,
- An interrupt can be generated if bit JQOVFIE is set.



P1: sequence of 3 conversions
P2: sequence of 1 conversion
P3: sequence of 2 conversions

Queue empty, JQM=0

- When the Queue become empty:
  - If JQM=0 ➔ The Queue is maintained with the last active context,

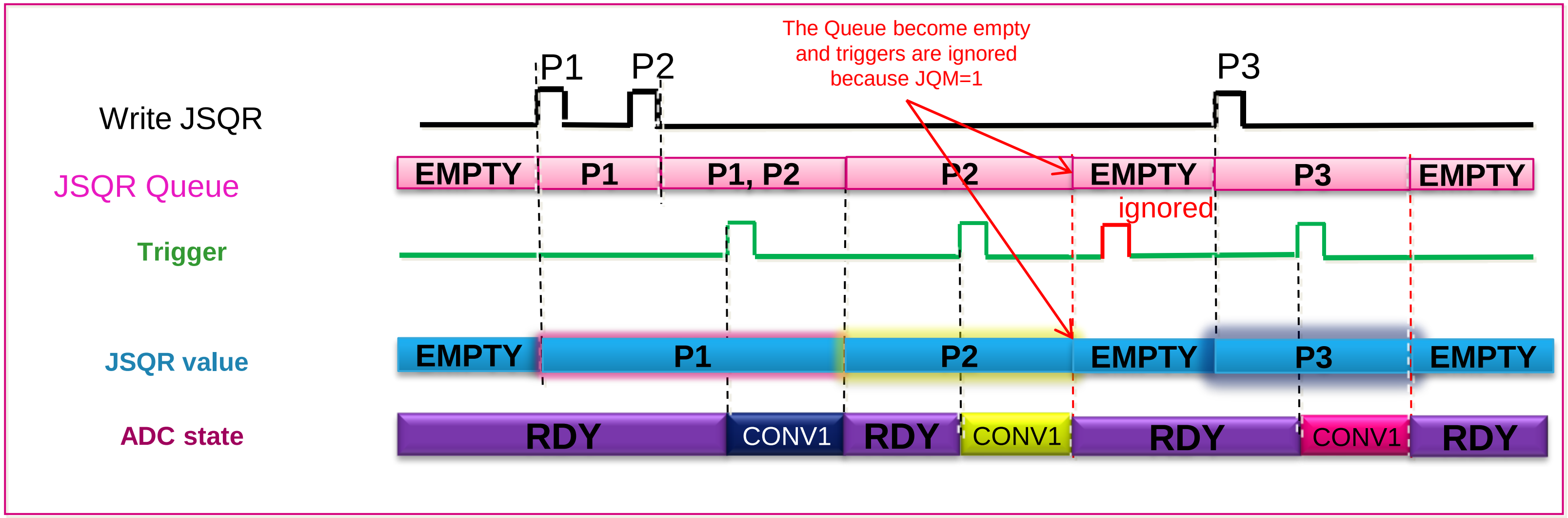


P1 sequence of 1 conversion

P2 sequence of 1 conversion

Queue empty, JQM=1

- When the Queue become empty:
  - If JQM=1 ➔ The Queue become empty and triggers are ignored,



The Queue become empty and triggers are ignored because JQM=1

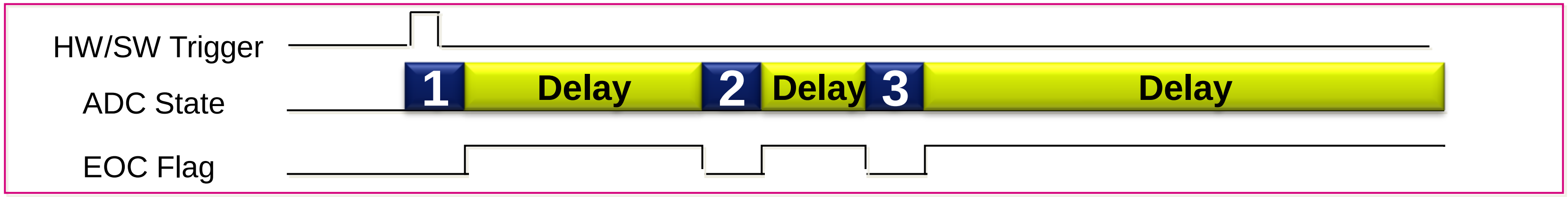
P1 sequence of 1 conversion

P2 sequence of 1 conversion

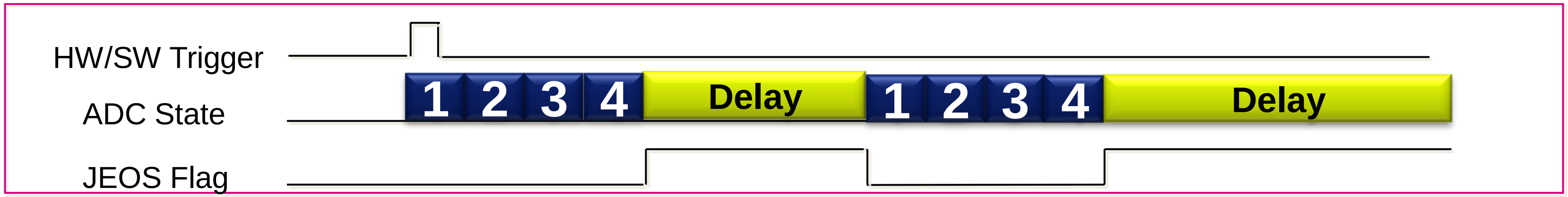
P3 sequence of 1 conversion

Auto delayed conversion (1/2)

- **Auto Delay Mode:** when AUTDLYbits = 1, a new conversion can start only if the previous data has been treated:
  - For regular conversions: once the ADC_DR register has been read or if the EOC bit has been cleared.



  - For injected conversions: when the JEOS bit has been cleared,



i Regular channel conversion

Note : A trigger event (**for the same group of conversions**) occurring during an already ongoing sequence or during this delay is ignored.

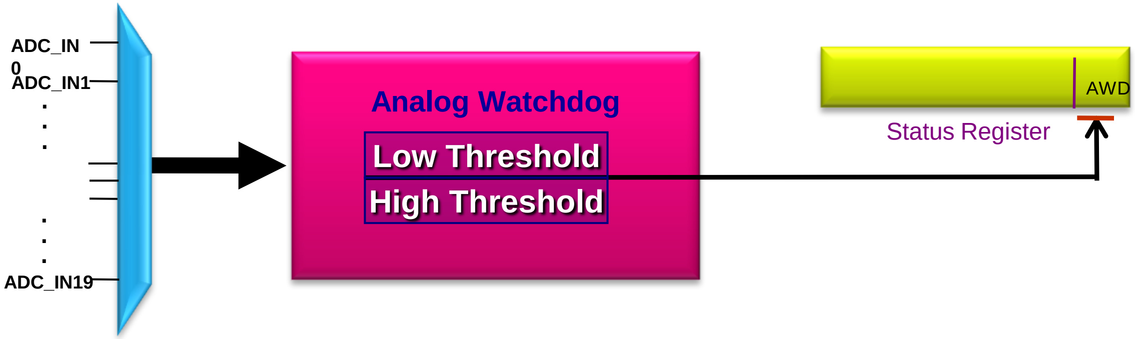
➔ This is a way to automatically adapt the speed of the ADC to the speed of the system that reads the data.

Auto delayed conversion (2/2)

- No delay inserted between each conversions of different groups (a regular conversion followed by an injected conversion or conversely)
  - If an injected trigger occur during the automatic delay of a regular conversion, the injected conversion starts immediately,
  - Once the injected sequence is complete, ADC waits the delay of the previous regular conversion before lunching new regular conversion,
- In auto-injected mode (JAUTO=1) a new regular conversion can start only when the automatic delay of the previous injected sequence of conversion has ended (when JEOS has been cleared),

ADC Analog Watchdogs

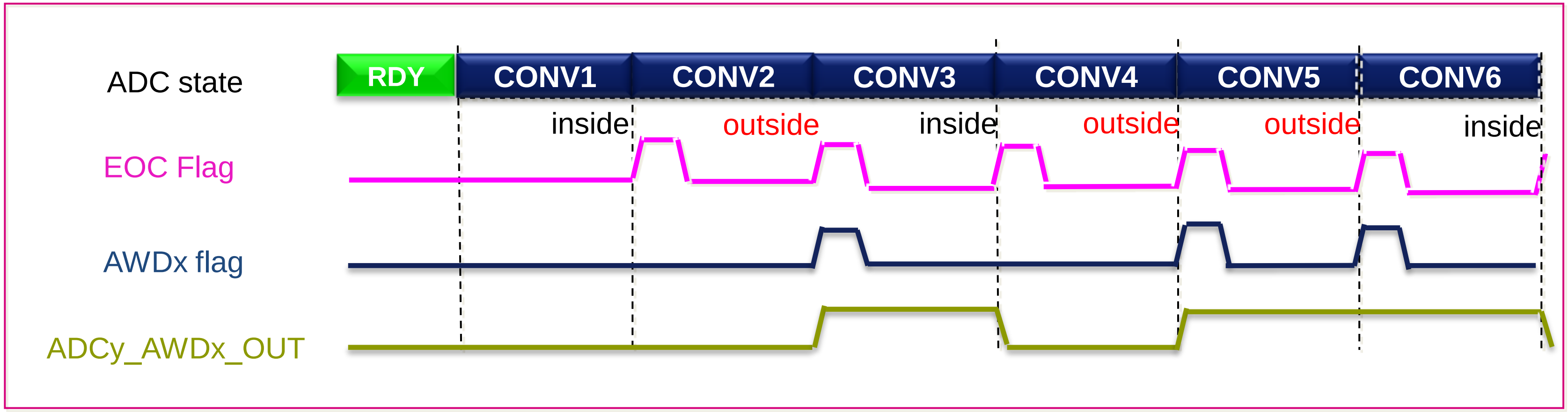
- ADC Analog Watchdog 1
  - 12-bit programmable analog watchdog low and high thresholds
  - Enabled on one or all converted channels
  - Interrupt generation on low or high thresholds detection
- ADC Analog Watchdog 2&3
  - Enabled on some selected channels by programming bits in AWDCHx[19:0],
  - Resolution Limited to 8 bits and only the 8 MSBs of the thresholds can be programmed into HTx[7:0] and LTx[7:0]



Note: *The watchdog comparison is performed on the raw converted data before any alignment calculation and before applying any offsets.*

Analog WDG signal generation

- Each analog watchdog is associated to an internal hardware signal ADCy_AWDx_OUT which is connected to an output timer,



***Note**: AWDx flag has no influence on the generation of ADCy_AWDx_OUT (ex: ADCy_AWDx_OUT can toggle while AWDx flag remains at 1 if the software did not clear the flag).*

DMACFG bit management

- DMA can be used to manage the regular channels conversions (ADCx DR register),
  - **DMA one shot mode (DMACFG = 0):**

    In this mode the ADC stops generating DMA requests once the DMA has reached the last DMA transfer even if conversion has been started again,

  - **DMA circular mode (DMACFG=1):**

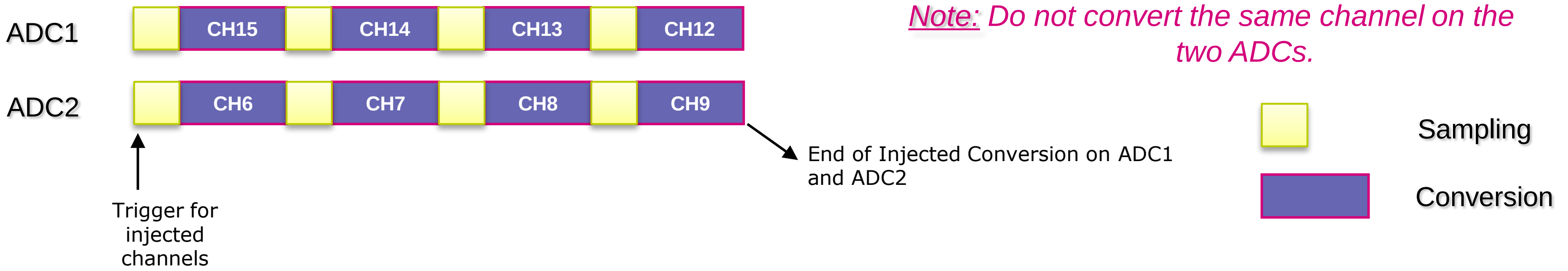
    In this mode, the ADC generates a DMA transfer request each time a new conversion data is available in the data register, even if the DMA has reached the last DMA transfer.

ADC Dual mode

- ADC1 and ADC2 can be used together in Dual mode (ADC1 is the master),
- ADC3 and ADC4 can be used together in Dual mode (ADC3 is the master),
- Six possible modes are implemented:
  - Injected simultaneous mode,
  - Regular simultaneous mode,
  - Interleaved mode,
  - Alternate trigger mode,
  - Injected simultaneous + Regular simultaneous mode,
  - Regular simultaneous + Alternate trigger mode,

Injected simultaneous mode

- Converts an injected channel group,
- The external trigger source comes from the injected group multiplexer of the master ADC,
- An JEOC is generated at the end of all channels conversion,
- Results stored on injected data registers of each ADC.

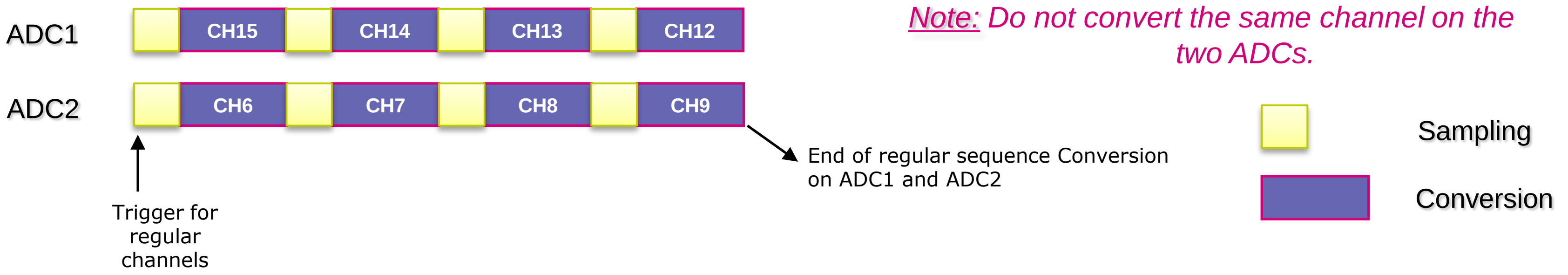

ADC1
CH15
CH14
CH13
CH12
ADC2
CH6
CH7
CH8
CH9
Trigger for injected channels
End of Injected Conversion on ADC1 and ADC2
Sampling
Conversion

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

- This mode can be combined with auto-delayed mode,
- Once SW set JADSTART or JADSTP bits of the master ADC, the corresponding bits of the slave ADC are also automatically set,

Regular simultaneous mode

- Converts an regular channel group,
- The external trigger source comes from the regular group multiplexer of the master ADC,
- An EOC is generated at the end of each channel conversion,
- Results stored on the common data register ADC_CDR and on the each ADCx_DR,


ADC1
CH15
CH14
CH13
CH12
ADC2
CH6
CH7
CH8
CH9
Trigger for regular channels
End of regular sequence Conversion on ADC1 and ADC2
Sampling
Conversion

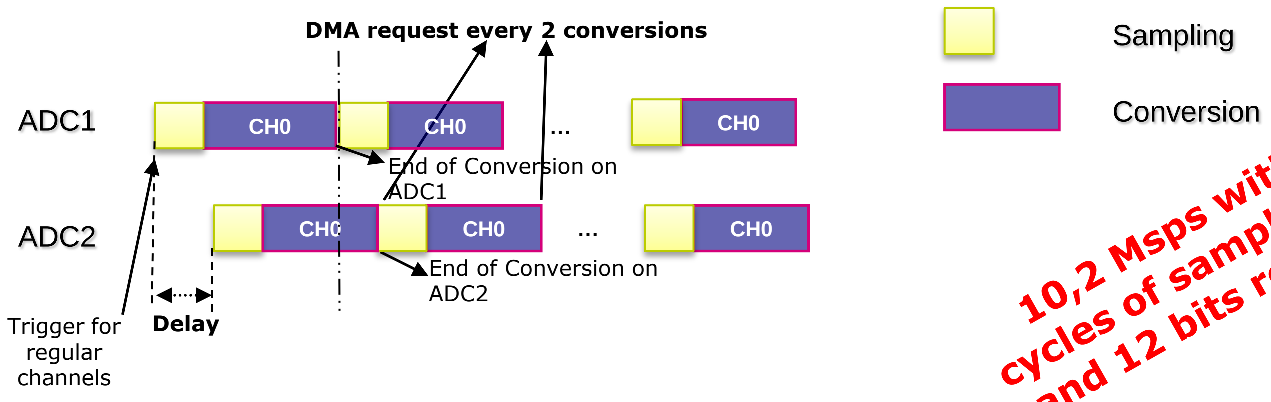
Note: Do not convert the same channel on the two ADCs.

- This mode can be combined with auto-delayed mode,
- Once SW set ADSTART or ADSTP bits of the master ADC, the corresponding bits of the slave ADC are also automatically set,

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 configurable delay,
- An EOC is generated at the end of each channel conversion,
- Results stored on the common data register ADC_CDR and on the each ADCx_DR,

18 Msps with 1,5 cycles of sampling time and 6 bits resolution

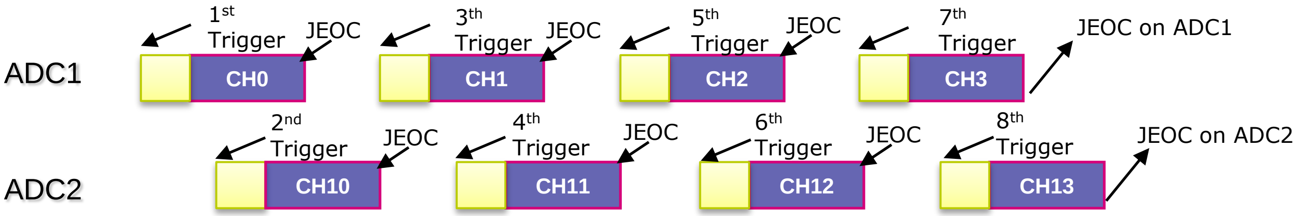

DMA request every 2 conversions
Sampling
Conversion
ADC1
CH0
CH0
...
CH0
End of Conversion on ADC1
ADC2
CH0
CH0
...
CH0
End of Conversion on ADC2
Trigger for regular channels
Delay

10,2 Msps with 1,5 cycles of sampling time and 12 bits resolution

- This mode can not be combined with auto-delayed mode,
- Once SW set ADSTART or ADSTP bits of the master ADC, the corresponding bits of the slave ADC are also automatically set,

Alternate trigger mode

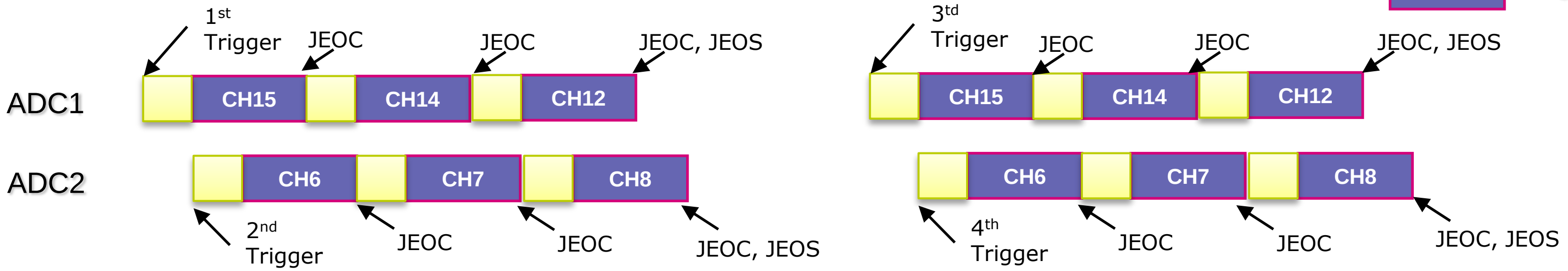
- Converts an injected channel group,
- The external trigger source comes from the injected group multiplexer of the master ADC,
- If discontinuous mode is enabled:



🟨 Sampling

🟪 Conversion

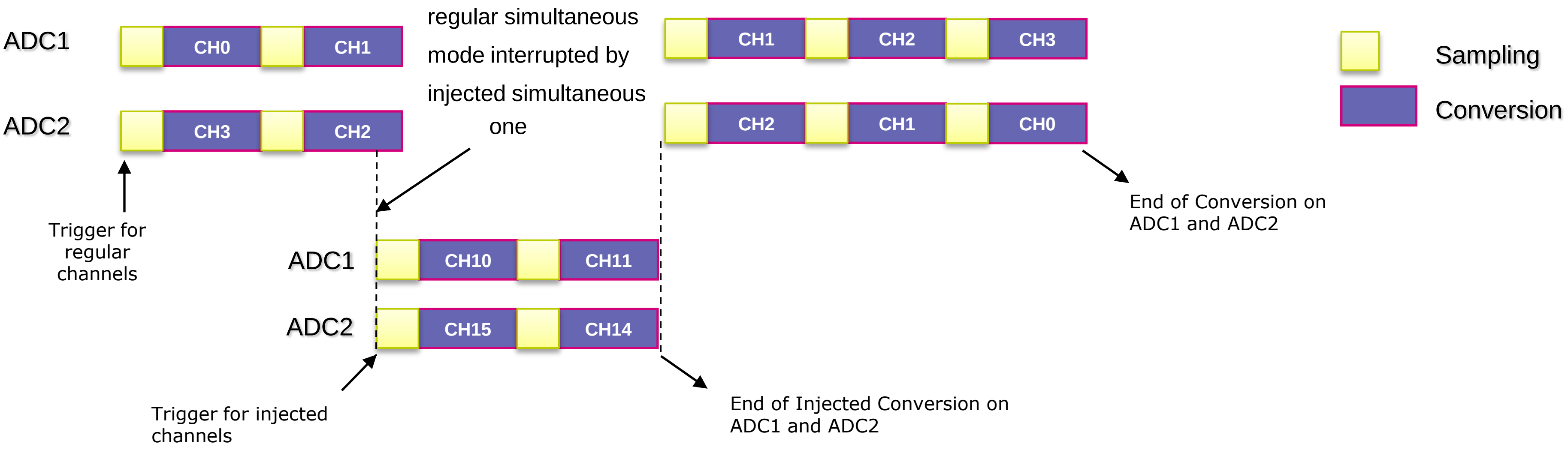
- If discontinuous mode is disabled:



- This mode can not be combined with auto-delayed mode,
- Once SW set JADSTART or JADSTP bits of the master ADC, the corresponding bits of the slave ADC are also automatically set,

Regular simultaneous + Injected simultaneous

- Converts an injected and regular channel groups,
- The external trigger source comes from the master ADC,
- Results of injected channels stored on ADCx_JDRy registers, and regular channels on each ADCx_DR register and on ADC_CDR register.

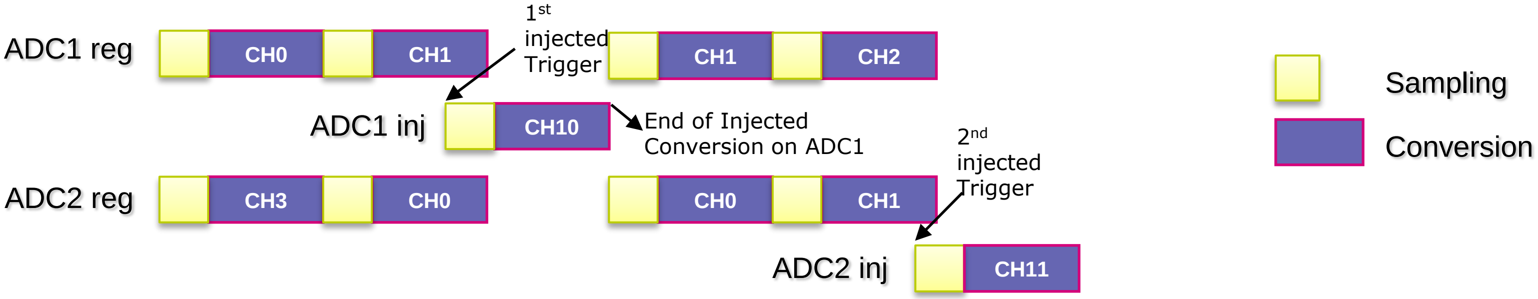

ADC1
CH0
CH1
ADC2
CH3
CH2
regular simultaneous mode interrupted by injected simultaneous one
CH1
CH2
CH3
CH2
CH1
CH0
Sampling
Conversion
End of Conversion on ADC1 and ADC2
Trigger for regular channels
ADC1
CH10
CH11
ADC2
CH15
CH14
Trigger for injected channels
End of Injected Conversion on ADC1 and ADC2

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

- This mode can be combined with auto-delayed mode,

Regular simultaneous + Alternate trigger

- Converts an injected and regular channel groups,
- The external trigger source comes from the master ADC,
- Results of injected channels stored on ADCx_JDRy registers, and regular channels on each ADCx_DR register and on ADC_CDR register.



- This mode can not be combined with auto-delayed mode,

DMA requests in dual ADC mode

- **MDMA=0b00:**
  - One DMA channel should be configured for each ADC to transfer the data available on ADCx_DR register,

- **MDMA=0b10:**
  - A single DMA request is generated each time both master and slave EOC events have occurred,
  - Used in interleaved and in regular simultaneous mode when ADC resolution is 10 or 12 bits

1st DMA request ADC_CDR[31:0] = SLV_ADC_DR[15:0] | MST_ADC_DR[15:0]

2nd DMA request ADC_CDR[31:0] = SLV_ADC_DR[15:0] | MST_ADC_DR[15:0]

- **MDMA=0b11:**
  - A single DMA request is generated each time both master and slave EOC events have occurred,
  - Used in interleaved and in regular simultaneous mode when ADC resolution is 6 or 8 bits

1st DMA request ADC_CDR[15:0] = SLV_ADC_DR[7:0] | MST_ADC_DR[7:0]

2nd DMA request ADC_CDR[15:0] = SLV_ADC_DR[7:0] | MST_ADC_DR[7:0]

ADC Flags and interrupts

ADRDY: « ADC ready »

EOC : « Regular End Of Conversion »

EOS : « **Regular** End Of Sequence »

JEOC : « Injected End Of Conversion »

JEOS : « **Injected** End Of Sequence »

JQOVF : «**Injected** Injected context queue overflows»

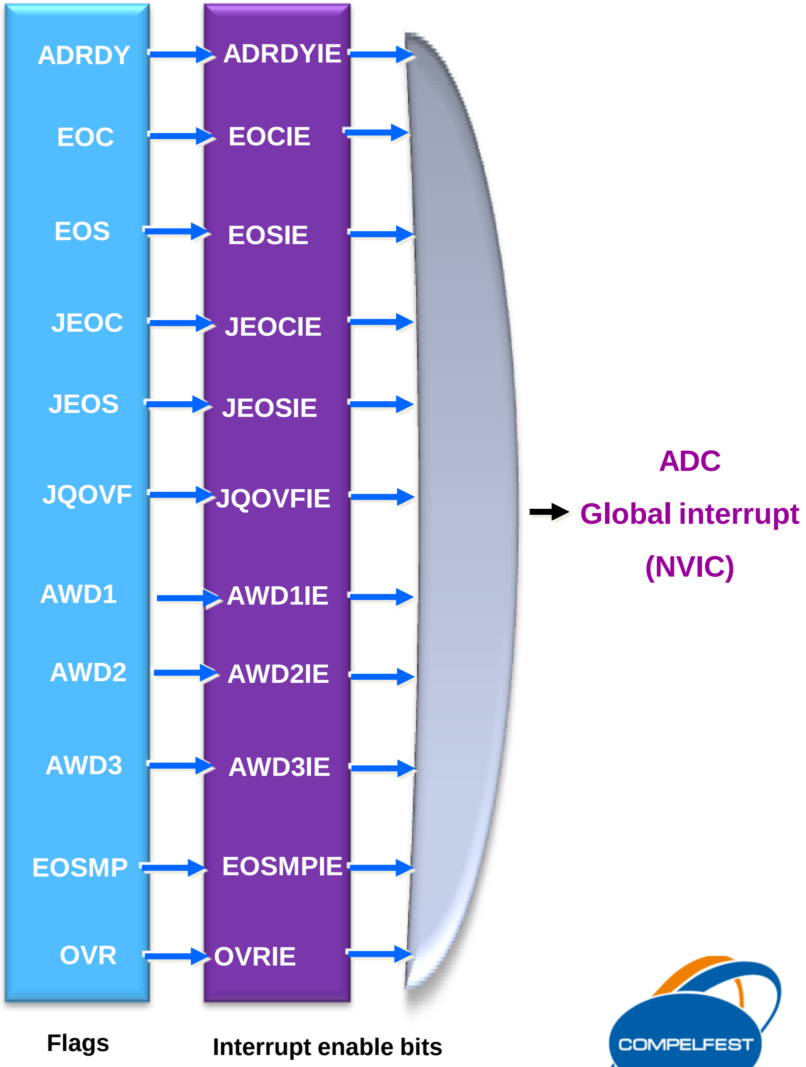
AWD1 : « Analog watchdog 1»

AWD2 : « Analog watchdog 2»

AWD3 : « Analog watchdog 3»

EOSMP: End Of Sampling

OVR: Overrun



Flags	Interrupt enable bits
ADRDY	ADRDYIE
EOC	EOCIE
EOS	EOSIE
JEOC	JEOCIE
JEOS	JEOSIE
JQOVF	JQOVFIE
AWD1	AWD1IE
AWD2	AWD2IE
AWD3	AWD3IE
EOSMP	EOSMPIE
OVR	OVRIE

ADC
Global interrupt
(NVIC)

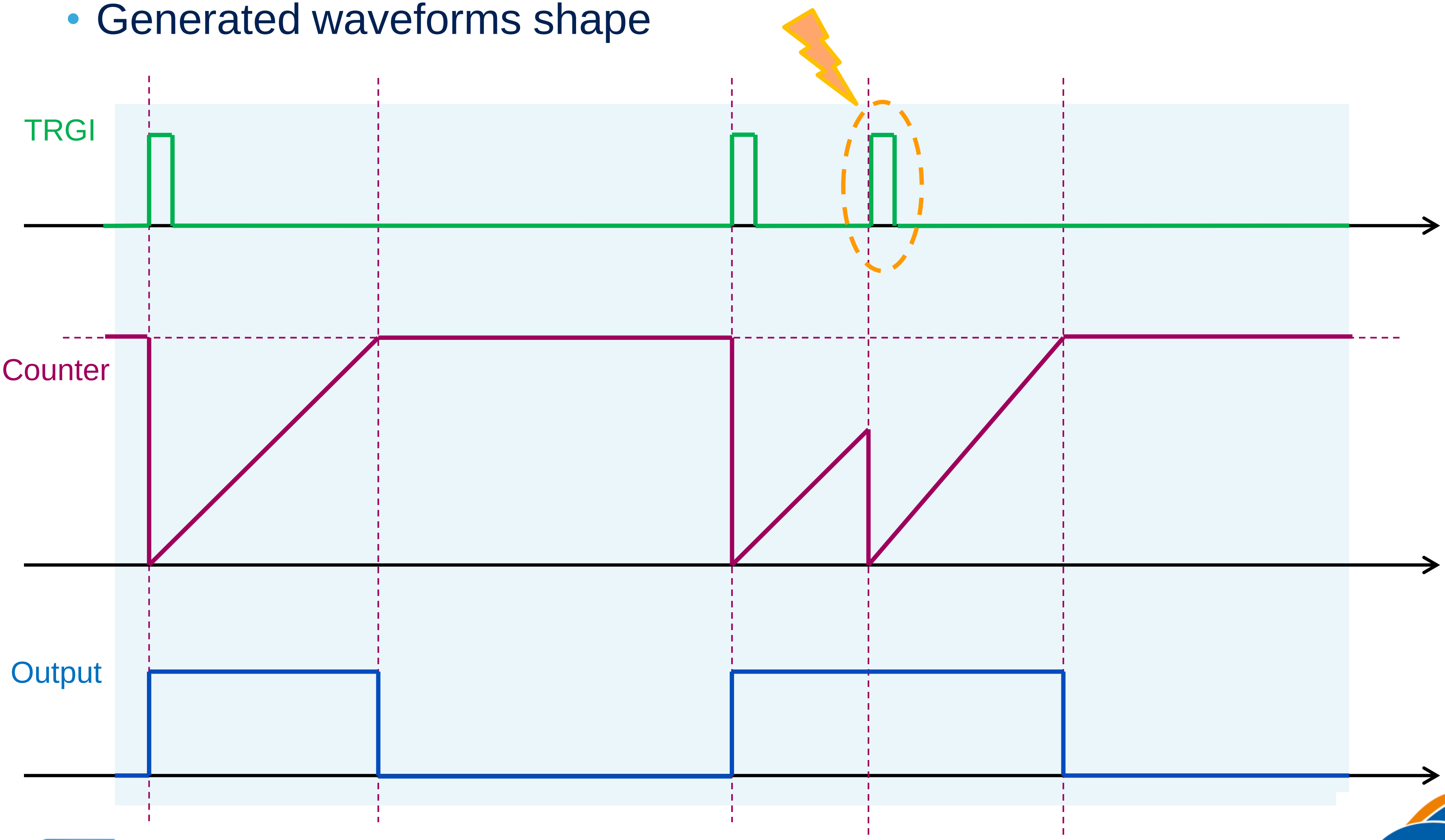
TIM1 and TIM8 enhancements in STM32F30X

Channel-level enhancements

- Up to 6 channels on Advanced control timers:
  - Up to 4 channels with input/output stages
    - Channels remain compatible with those on existing products' timers
    - New features:
      - More complex waveforms generation
      - Enhanced triggering capability
      - More channel's modes
      - Multitude of coupling scenarios between channels
  - 2 extra channels
    - Internal channels
      - Not wired to GPIOs
      - Used within the Timer itself for complex waveform generation
      - Routed to the ADC triggering logic (via Timer's TRGO output)
    - Compare-and-PWM-modes-only channels
      - No capture modes
      - No DMA channels nor Interrupt request lines
    - High coupling with channels 1,2,3 and4
      - Enhance waveform generation on those channel: More complex waveforms can be generated
      - Enhanced triggering mechanism: ADC oriented triggering mechanism
      - Designed to meet many Motor Control applications' requirements

Retriggerable One Pulse Mode (1/2)

- Generated waveforms shape



Retriggerable One Pulse Mode (2/2)

- Available on Channel 1, 2, 3 and 4
- Different from the existing One Pulse mode:
  - The outputted pulse starts as soon as a trigger active edge is detected
  - The pulse length is extended if a new active edge is detected
- Pulse length is set using the ARR register
  - For Up-counting mode, CCRx register has to be set to zero
  - For Down-counting mode, CCRx register has to be set to ARR value
- Configuration sequence
  - Set the timer to **slave** mode: the ***Combined Reset+Trigger*** mode **shall** be used
  - Select the Retriggerable One Pulse mode through the OCxM[3:0] bit field
    - Retriggerable OPM mode 1
    - Retriggerable OPM mode 2

Channels Coupling (1/2)

- Two coupling schemes:
  - Adjacent channels coupling:
    - Channel1 and Channel2 coupling
    - Channel3 and channel4 coupling
  - Enhanced channels coupling (feature used by Motor Control applications)
    - Channel5 and Channel1
    - Channel5 and Channel2
    - Channel5 and Channel3
- Flexible coupling mechanism on adjacent channels
  - Channels coupling output can be directed to one channel or to both of them
- Generated Waveforms' shape
  - Frequency control through TIMx_ARR register value
  - Phase-shift (delay) control through one of the two channels' TIMx_CCR register
  - Pulse-length (duty-cycle) control through the second channels' TIMx_CCR register

Channels Coupling (2/2)

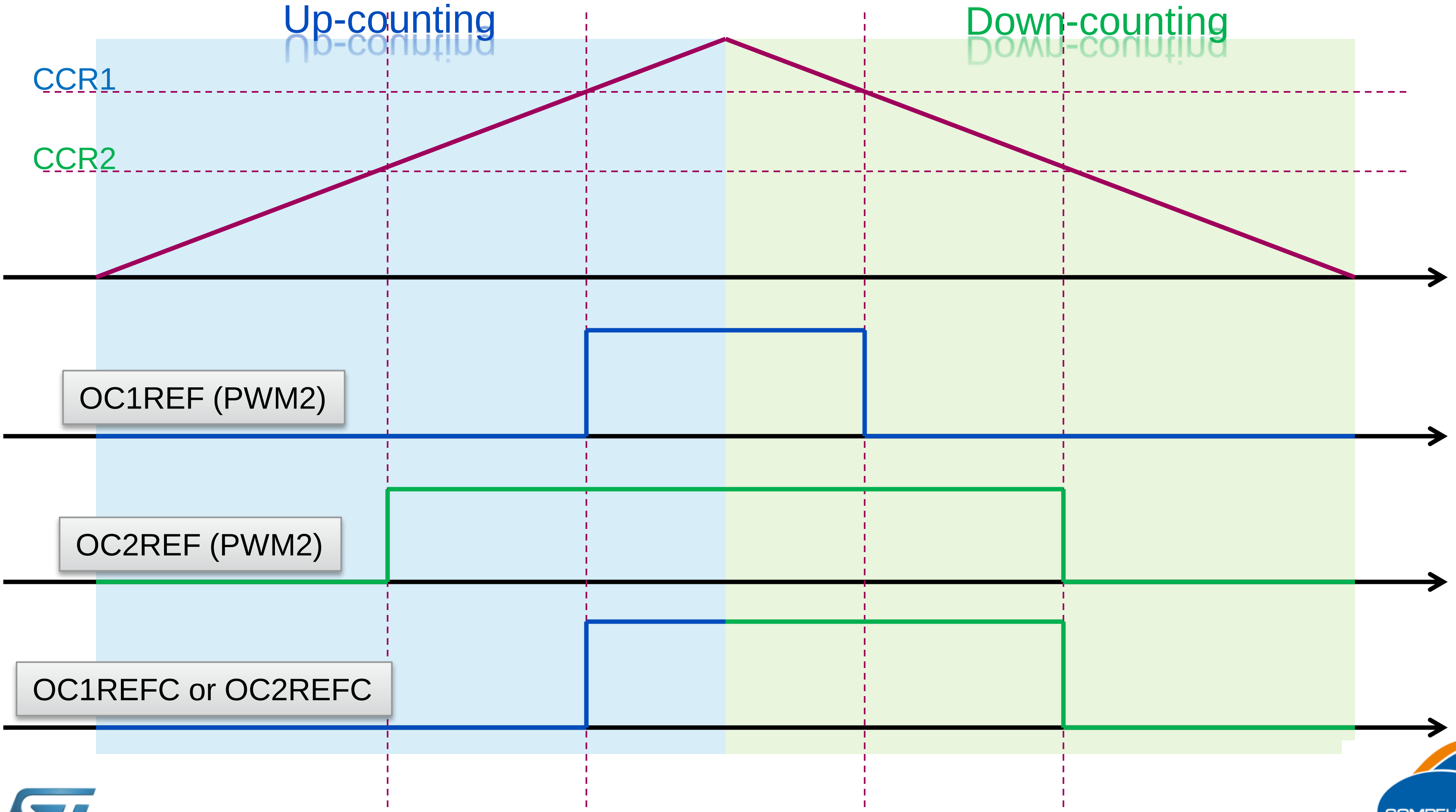
- Available PWM modes
  - Each channel among the first four channels can be configured in one of the following PWM modes
  - Asymmetric and Combined PWM modes are applicable on coupled channels only

	PWM mode 1	PWM mode 2
Independent	OCxM[3:0] = 4b’0110’	OCxM[3:0] = 4b’0111’
Asymmetric	OCxM[3:0] = 4b’1110’	OCxM[3:0] = 4b’1111’
Combined	OCxM[3:0] = 4b’1100’	OCxM[3:0] = 4b’1101’

Coupling between channels is activated

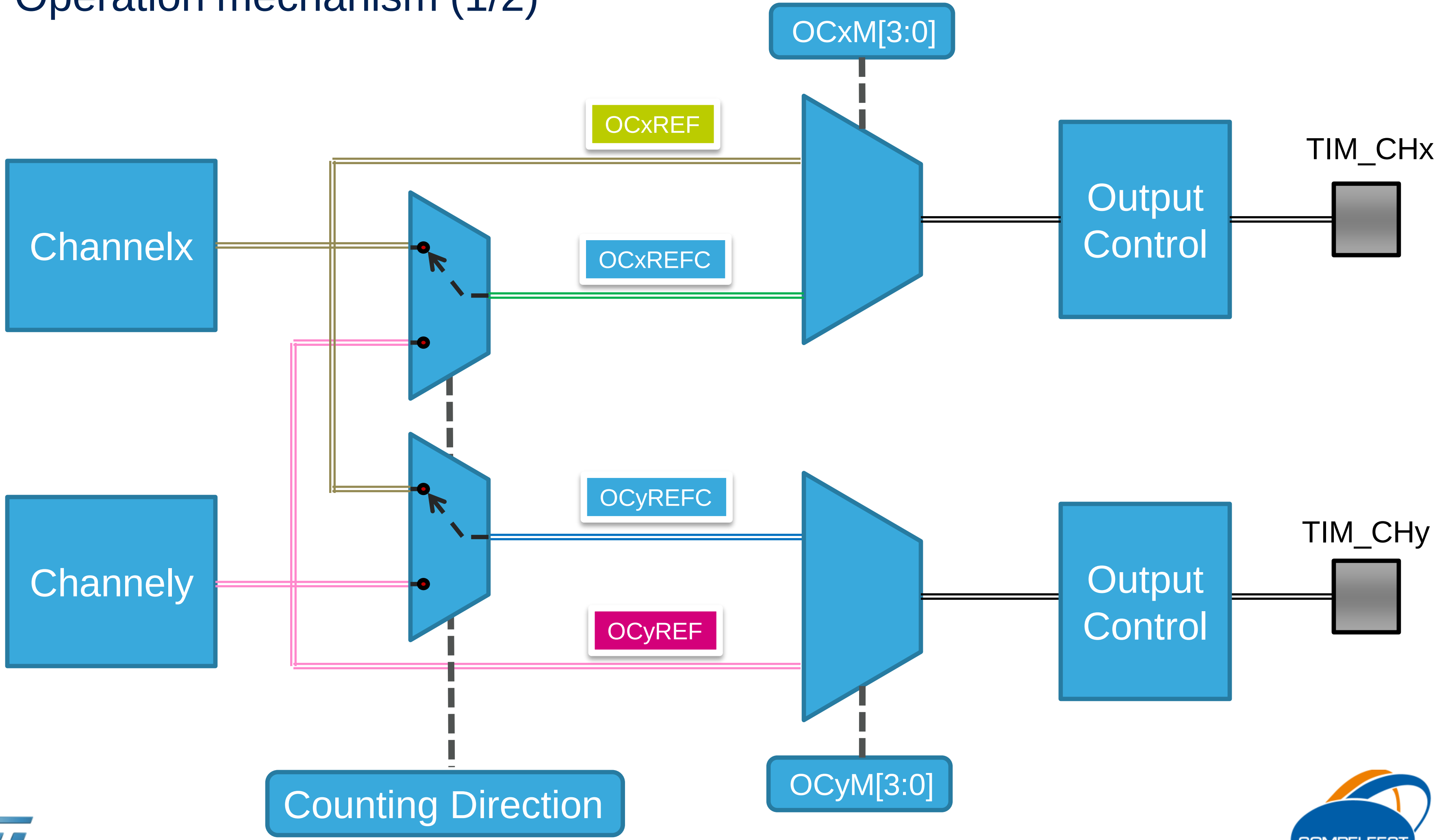
Asymmetric PWM mode (1/3)

- Output waveform shape



Asymmetric PWM mode (2/3)

- Operation mechanism (1/2)

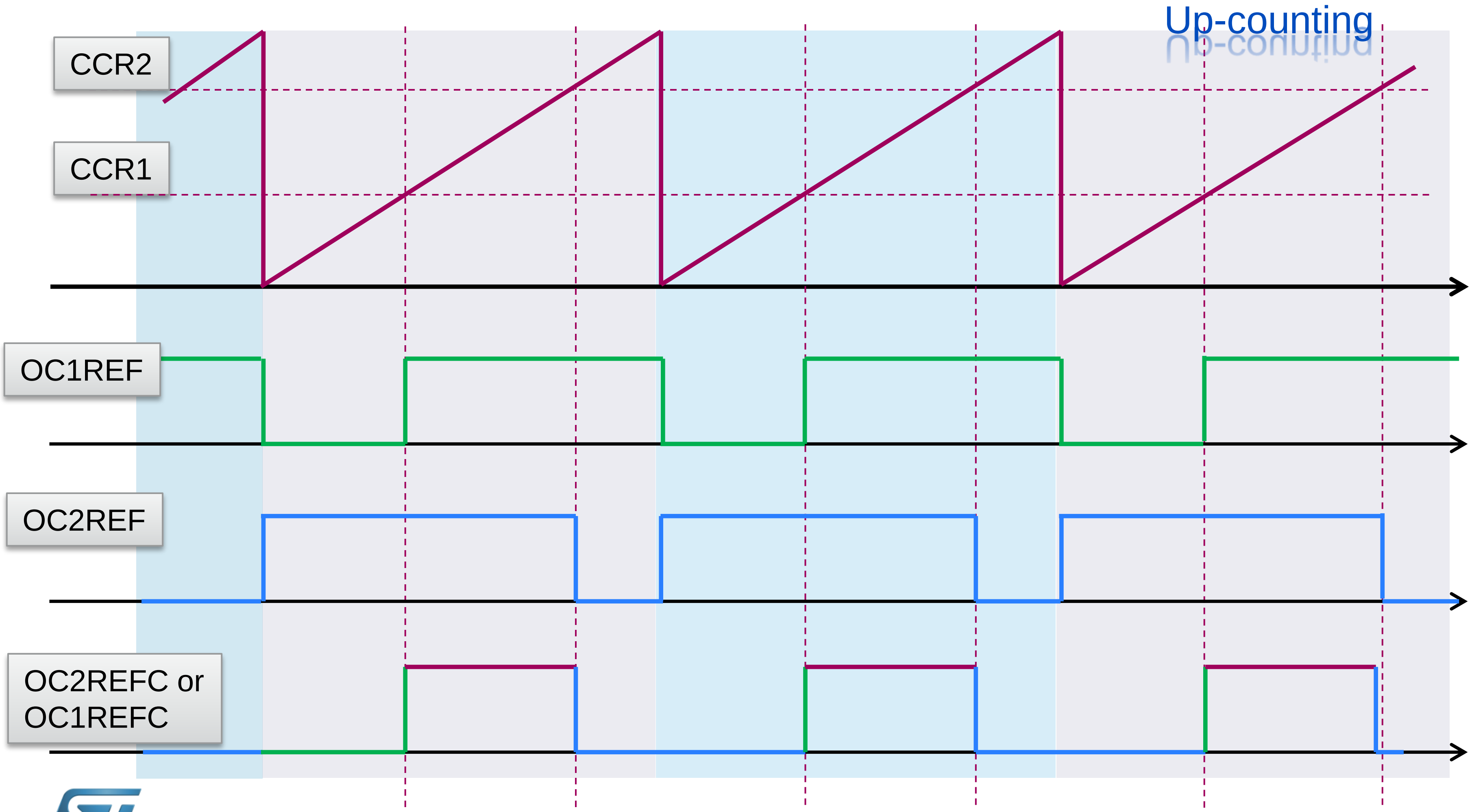


Asymmetric PWM mode (3/3)

- Operation mechanism (2/2)
  - The counting direction selects which channel output to be directed to OCxREFC
  - Coupled channel has to be configured in the same PWM mode
- Center-aligned counting mode required
  - Asymmetric mode is **effective** only when the timer is configured to count in **center-aligned** mode
- Available on the following channel couples:
  - (Channel1, Channel2)
  - (Channel3, Channel4)
- Two Asymmetric PWM mode are available
  - Asymmetric PWM1 mode
  - Asymmetric PWM2 mode

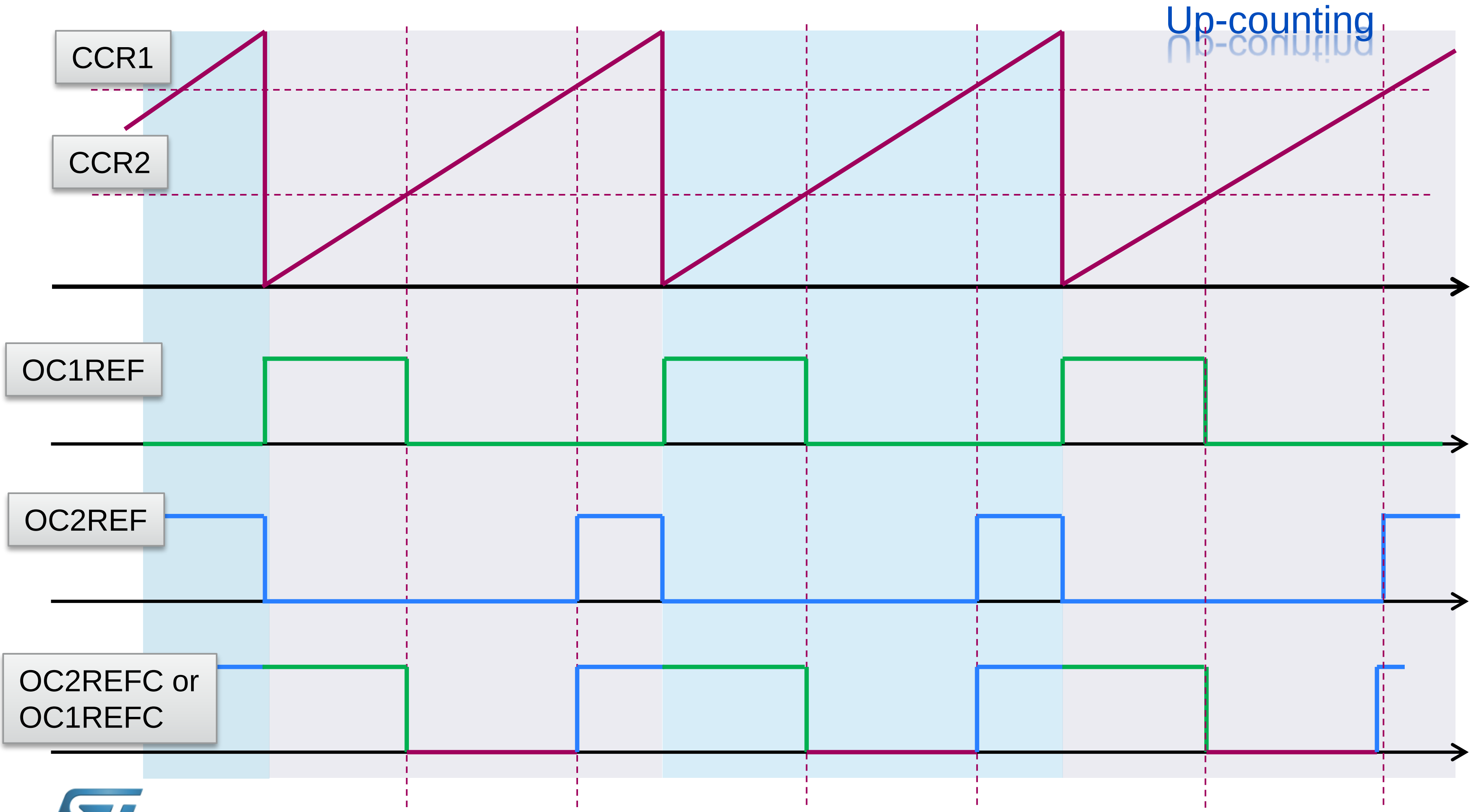
Combined PWM mode (1/5)

- Output waveform shape (Logical And)



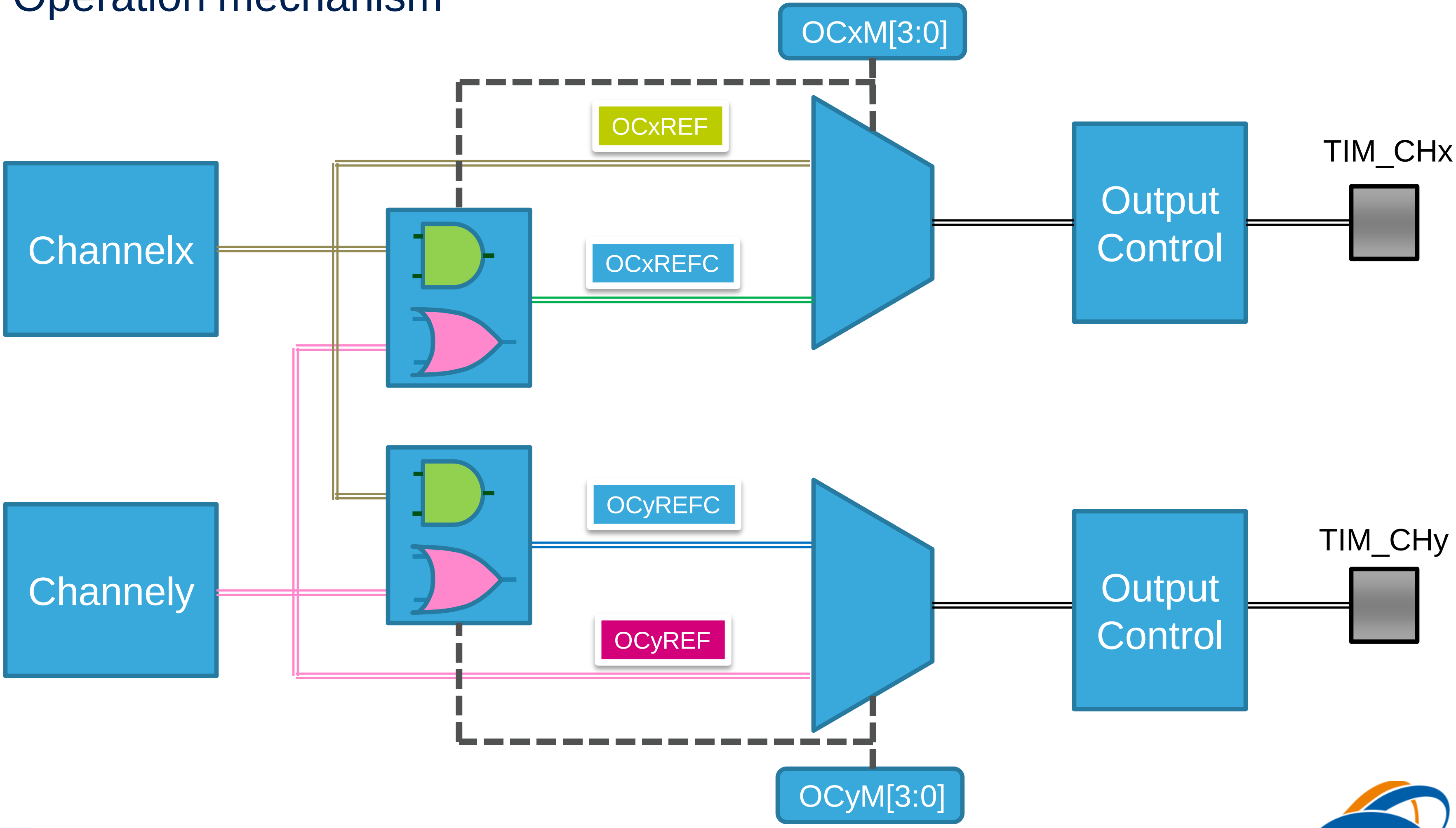
Combined PWM mode (2/5)

- Output waveform shape (Logical Or)



Combined PWM mode (3/5)

- Operation mechanism



life.augmented

COMPELFEST

Combined PWM mode (4/5)

- Two logical operators coupling modes:
  - Logical **And**
  - Logical **Or**

- Two Combined PWM mode are available
  - Combined PWM1 mode
  - Combined PWM2 mode

- Different PWM mode on each channel
  - In order to get the desired output, the two coupled channels has to be configured with different PWM modes: PWM1 and PWM2
  - If the same PWM mode is configured on both channels, the output signal waveform is similar to one of the two channels waveforms depending on the Logical Operator applied

Combined PWM mode (5/5)

- Configuration sequence
  - Configure the two coupled channels on different PWM modes
  - Configure one channel or both coupled channels to output a logical combination of the channels' waveforms

- Counting mode independent:
  - Acts on Edge-aligned counting mode
  - Acts on Center-aligned counting mode

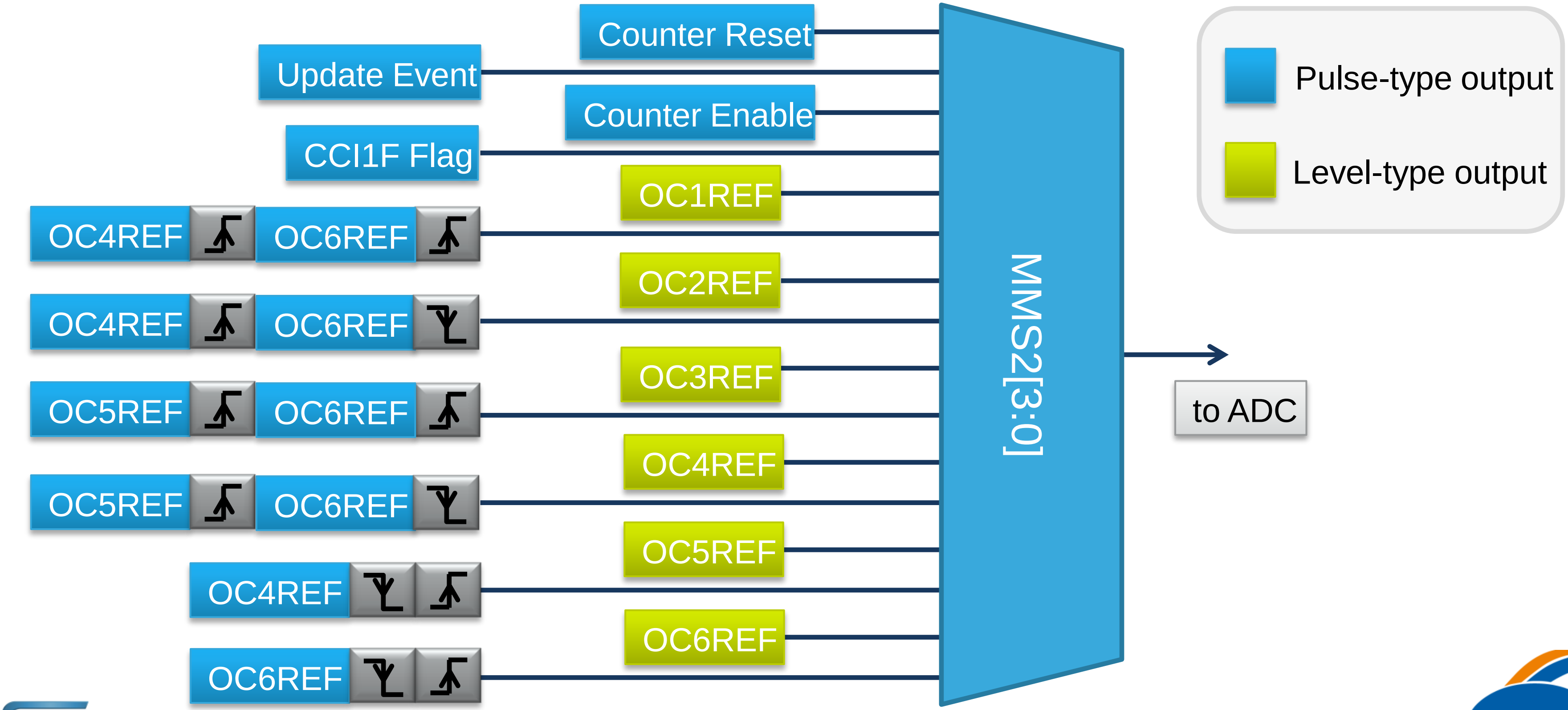
- Available on the following channel couples:
  - (Channel1, Channel2)
  - (Channel3, Channel4)

Channel 5&6 features

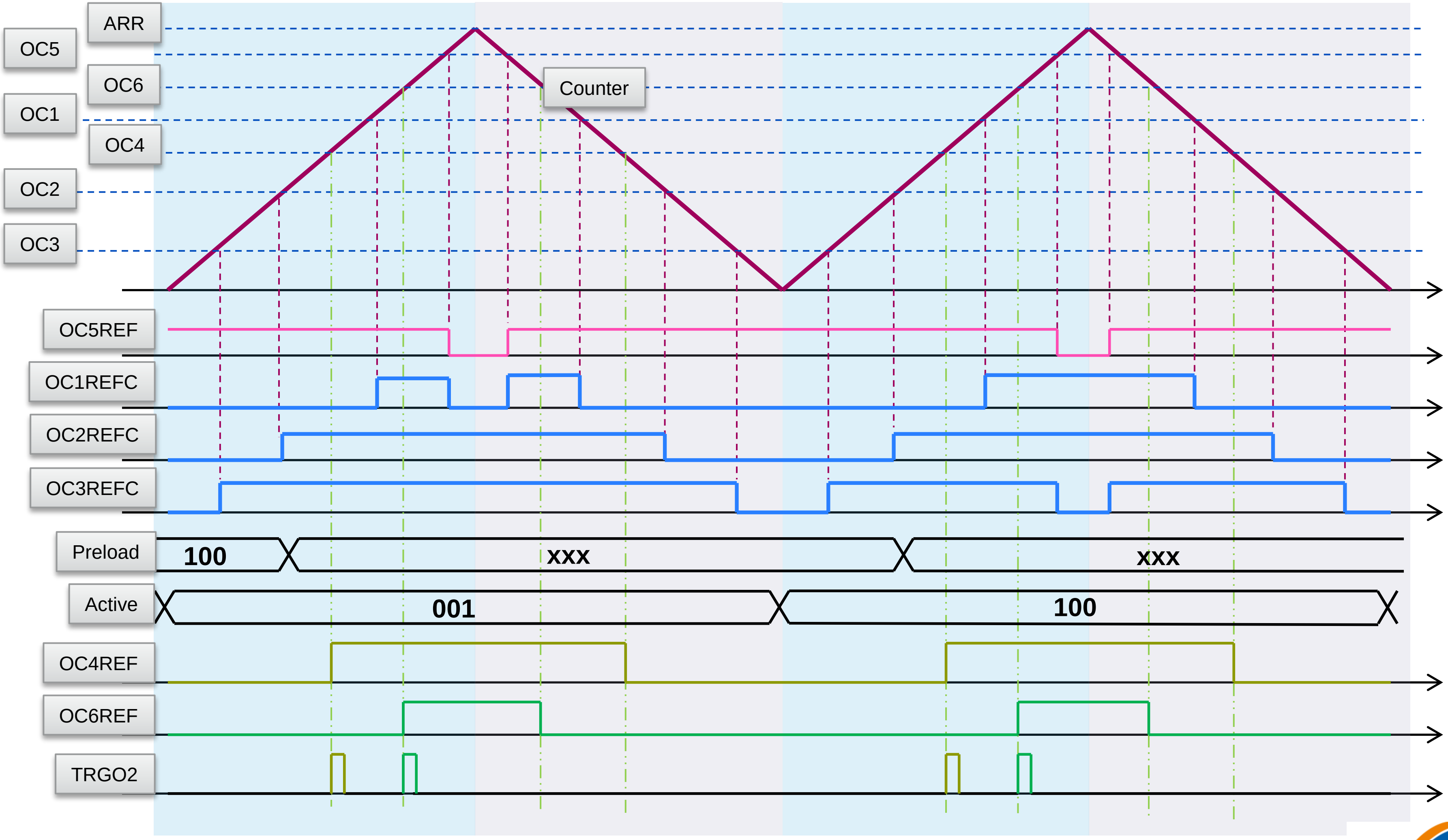
- Channels 5&6 characteristics:
  - Only available on advanced control Timers: TIM1 & TIM8
  - Compare-and-PWM-modes-only channels
  - Internal channels (no external output)
- Channel 5&6 use cases:
  - Can be used to generate more complex waveforms when combined with other channels (applicable for Channel5 only)
  - Can be used to trigger ADC conversion (many triggering scenarios)
- Compatible with the first four channels' implementation
  - Same control registers (for implemented features)
  - Same control bit-fields' structure (for implemented features)
- Typical use case
  - Used by single-shunt current measurement applications

Enhanced Triggering mechanism

- Additional set of triggers dedicated for ADC
  - Outputted on the new (second) trigger output TRGO2
  - Controlled through the new bit-field MMS2[3:0]

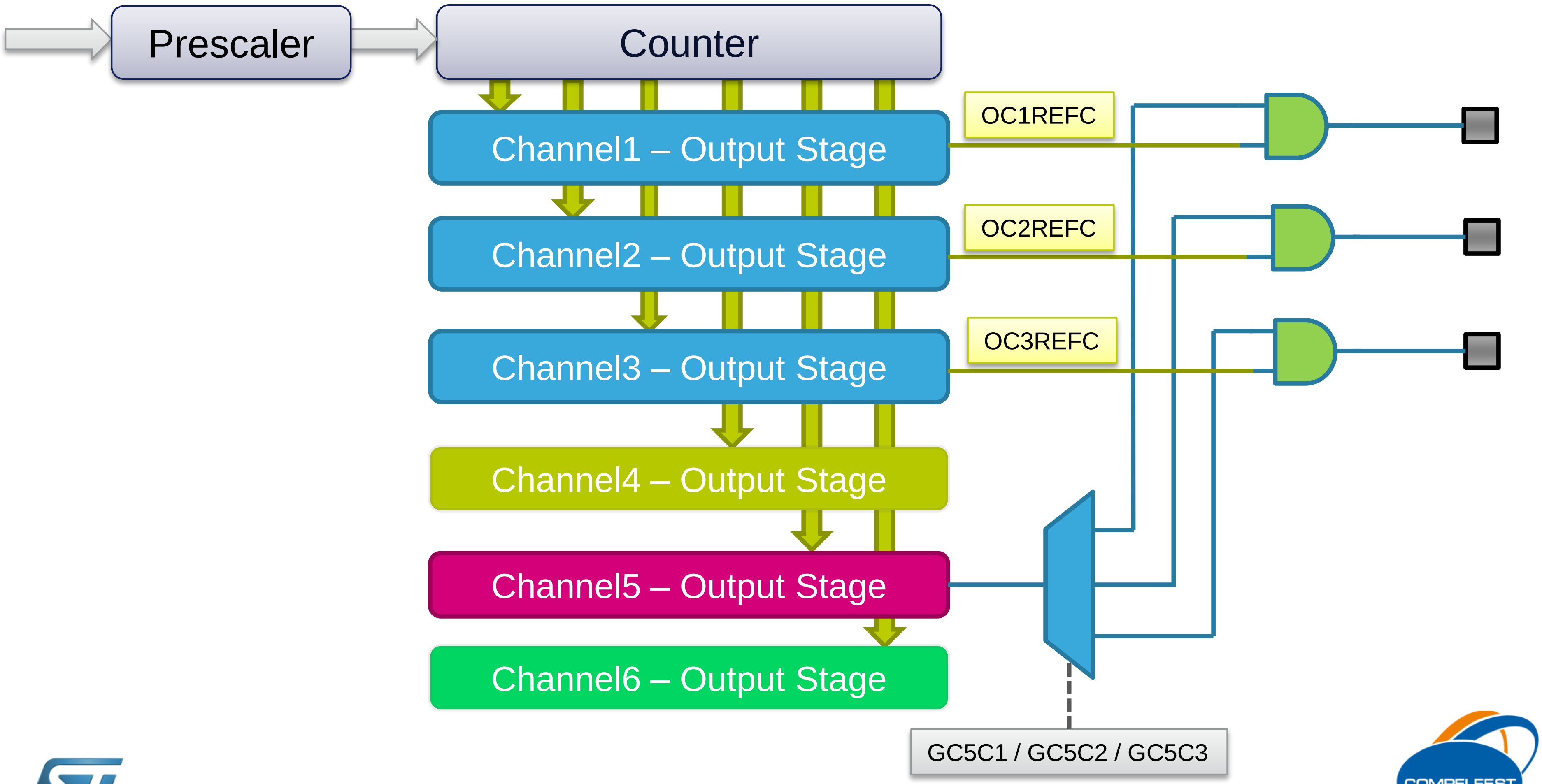


Combined 3-phase PWM mode (1/3)



Combined 3-phase PWM mode (2/3)

- Operation mechansim



Combined 3-phase PWM mode (3/3)

- Waveforms generation on up to three channels
  - Based on coupling Channel5’s output with others channels
    - Channel1
    - Channel2
    - Channel3
- Dedicated for Motor Control application
  - Used by ST’s patented Single-shunt current reading application
  - Can reduce CPU load by 5-10% compared to current implementation on F1/F2/F4 families
  - Frees many MCU resources (DMA channels, Interrupt request lines)

Miscellaneous enhancements (1/2)

- Repetition counter width is up to 16 bit
  - Gives about 650ms between updates for 100KHz PWM frequency
- Two OCxREF clearing sources:
  - External OCxREF clearing input: ETRF input
  - Internal OCxREF clearing input
    - Connected internally to comparator output: a pseudo-cell for Cycle-by-cycle current control
- Timers' synchronization enhancement
  - Introduction of a new synchronization mode: ***Combined Reset+Trigger*** Mode
    - When a trigger active edge is detected, the counter content is Reset and the counting is started
    - For configuring the *Retriggerable One-Pulse* mode, the timer has to be configured in slave mode: *Combined Reset+Trigger* mode shall be used

Miscellaneous enhancements (2/2)

- Up to two break input sources
  - Break input 1 (legacy one)
    - Idle State programming
    - Has the highest priority over Break inputs
    - Multiplexed with internal break signals:
      - Clock failure event from CSS block
      - SRAM parity error
      - Comparators outputs
      - PVD interrupt
      - Cortex M0 lockup (hard fault) output
    - Built with a digital filter with a flexible set of sampling periods
    - Asynchronous functioning (unless the filter is enabled)
    - Typical use case: Over-voltage protection handling
  - Break input 2 (new one)
    - No Idle State programming
    - Lower priority compared to Break input 1 (legacy one)
    - Built with a digital filter with a flexible set of sampling periods
    - Typical use case: Over-current protection handling

Product-level enhancements (1/2)

- Two clock sources for Advanced Control Timers (TIM1/TIM8)
  - APB clock (as for TIM15,16,17)
  - PLL output
    - Advanced Control Timers operate with clock frequency up to **144MHz**
    - To reach 144MHz operation frequency the following conditions shall be fulfilled:
      - SYSCK/AHB prescaler must be set to 1
      - AHB/APB prescaler must be set to 1
- Clock source selection
  - Please refer toTIM1SW bit description within RCC_CFGR3 register description (RCC chapter)
  - The TIM1SW control bit can set/reset by software
    - In case where one of the above conditions is not fulfilled, the TIM1SW control bit is reset by hardware

Product-level enhancements (2/2)

- Encoder Mode enhancement
  - Two Timers can share the same Quadrature Encoder output signals
    - TIM2 IC1 (*respectively TIM2 IC2*) is connected to TIM15 IC1 (*respectively TIM15 IC2*)
    - TIM3 IC1 (*respectively TIM3 IC2*) is connected to TIM15 IC1 (*respectively TIM15 IC2*)
    - TIM4 IC1 (*respectively TIM4 IC2*) is connected to TIM15 IC1 (*respectively TIM15 IC2*)
- Configuration
  - Using ENCODER_MODE bit field within the SYSCFG_CFGR1 register (for more details refer to SYSCFGR chapter)
- Use case
  - Used with M/T technique for estimating Velocity and Acceleration for wide-range of velocity values (especially for low velocity values)

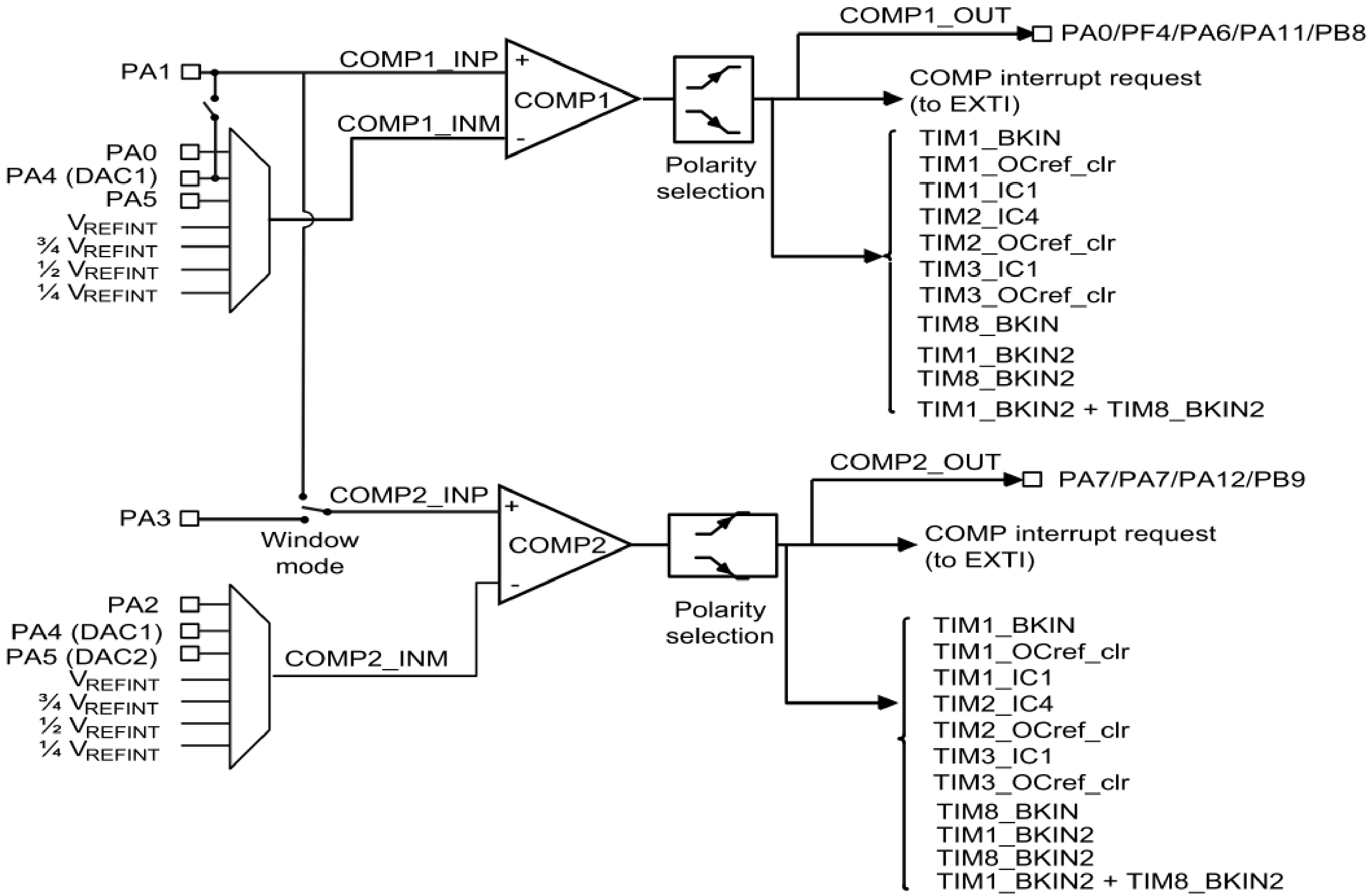
Comparators (COMP)

COMPARATORS (COMP)

- 3 comparator pairs COMP1/COMP2, COMP3/COMP4 and COMP5/COMP6 and single COMP7
  - Rail-to-rail inputs
  - Programmable speed / consumption: 4 modes
  - Programmable hysteresis: 4 levels
  - Inputs and outputs available externally - can be used as a standalone device without MCU interaction
  - Comparator pairs can be combined into a window comparators
  - Multiple choices for input thresholds and output redirection
  - Comparator blanking – The blanking time period is defined by TIM OC – multiple timer OC events available
    - to avoid reaction of the regulation loop on the current spikes at the beginning of the PWM period caused by the recovery current in power switches

- Can be used for:
  - Exiting low power modes on an analog event
  - Signal conditioning
  - Cycle-by-cycle current control with blanking (w/ DAC and TIM)

Block diagram for STM32F30x



BKIN: PWM's "Emergency stop" input
OCRefClear: PWM clear for cycle-by-cycle current controller

COMP features

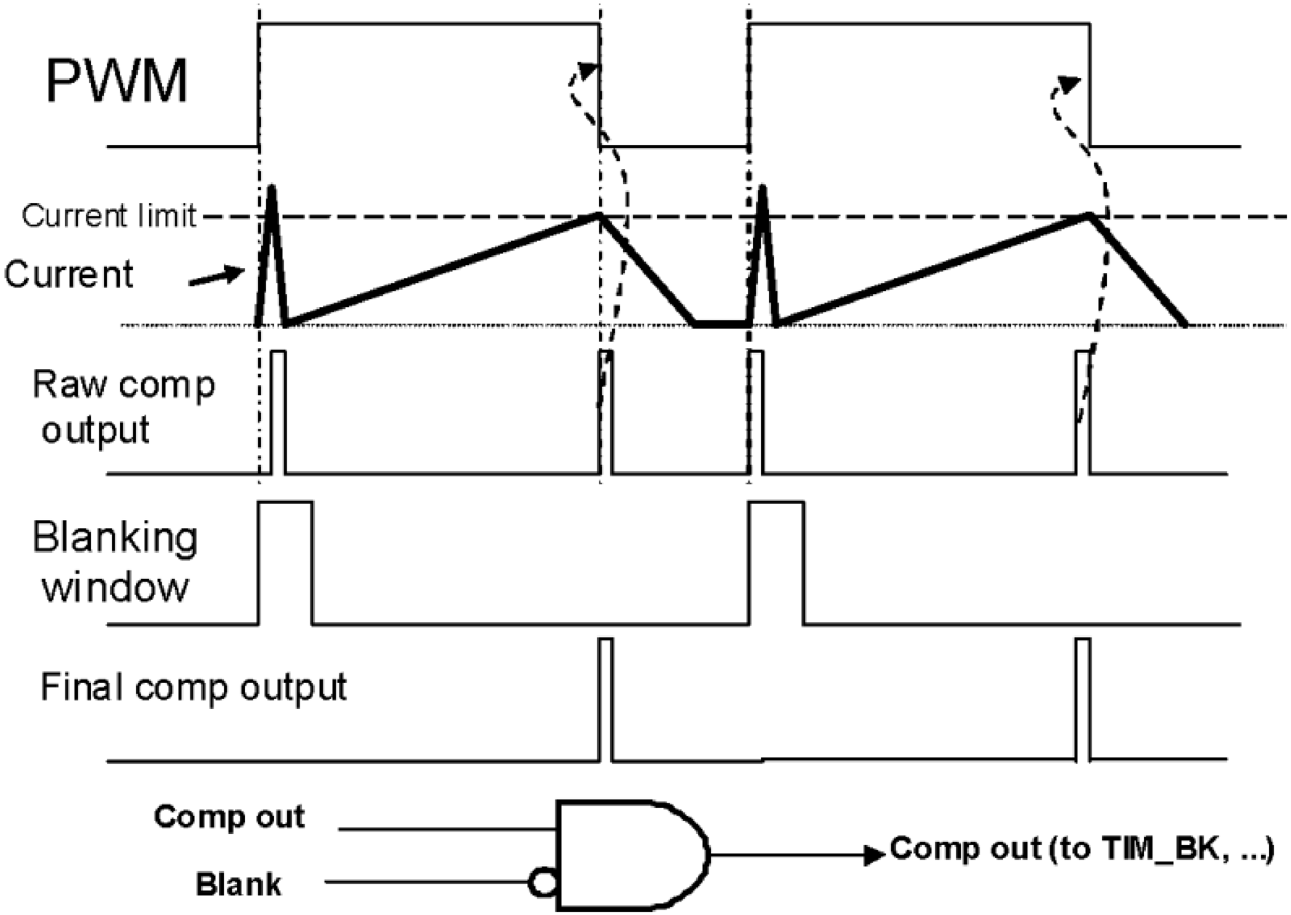
- Comparator characteristics at a glance
  - Full operating voltage range 2V < Vdda < 3.6V
  - Propagation time vs consumption
    (characterized typ. 2.7 < Vdd < 3.6V, for 100 mV step with 10 mV overdrive, TBD)
    - High speed / full power
    - Medium speed / medium power
    - Low speed / Low power
    - Very low speed / Ultra-low power
    - Input offset: +/-4mV typ, +/- 20mV max
  - Programmable hysteresis: 0, 8, 15, 31 mV

- Fully asynchronous operation
  - Comparators working in STOP mode
  - No clock related propagation delay

- Functional safety (Class B)
  - The comparator configuration can be locked with a write-once bit

Application example

- Cycle-by cycle current control with blanking



Operational Amplifier

life.augmented

COMPELFEST

Features (1/2)

- Up to 4 operational amplifiers
- Rail to Rail input/output
- Low Offset voltage
- Access to all terminals
- Input multiplexer on inverting and non inverting inputs
- Input multiplexer can be triggered by a timer and synchronized with a PWM signal.
- 4 operating modes:
  - Standalone mode: External gain setting
  - Follower mode
  - PGA mode: internal gain setting (x2, x4, x8, x16)
  - PGA mode: internal gain setting (x2, x4, x8, x16) with inverting input used for filtering.

Features (2/2)

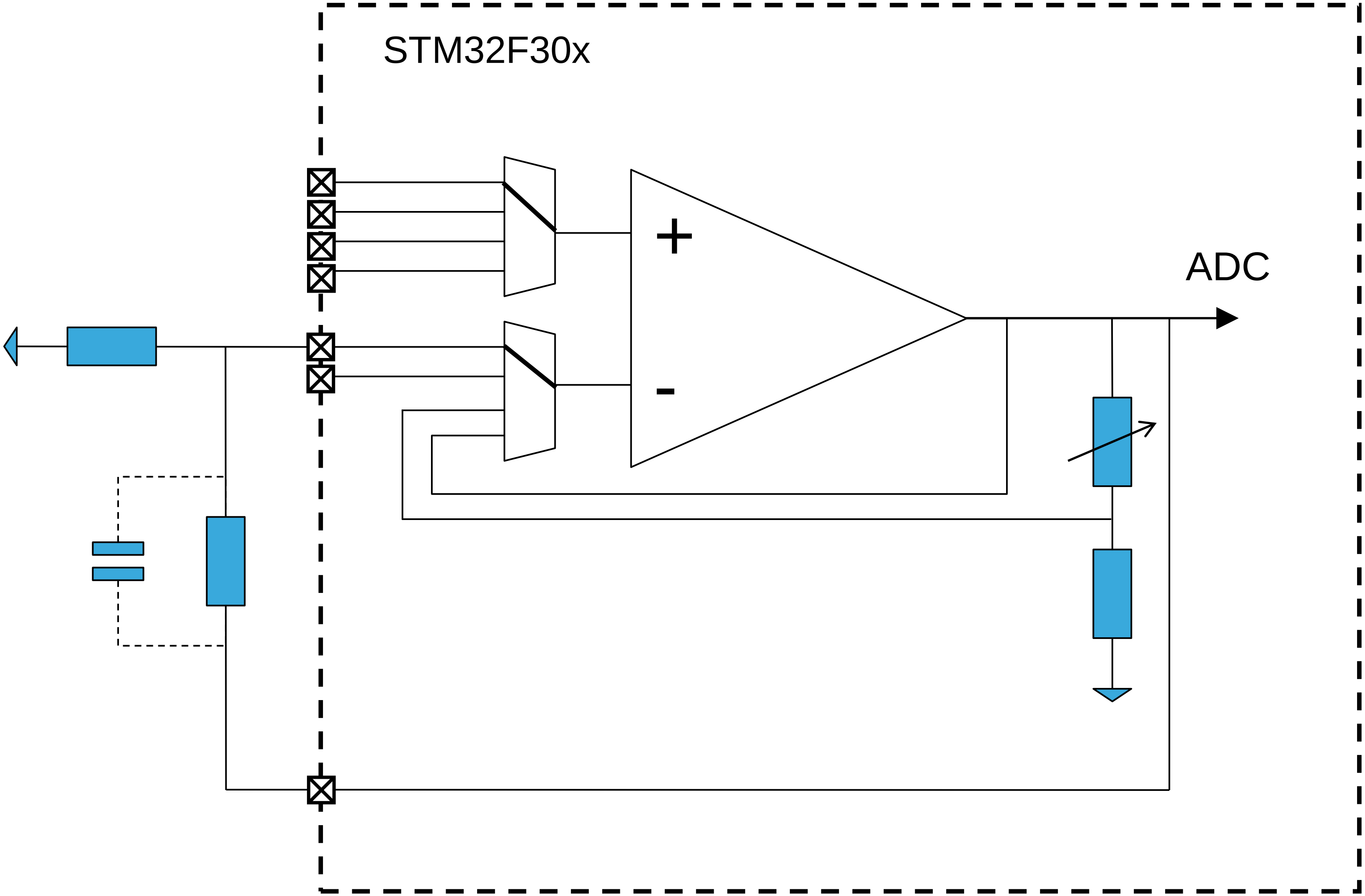
- Operating conditions
  - 2.0V < VDDA < 3.6V
  - -40 C < Temp < 105 C

- Input stage
  - Input: rail to rail
  - Offset: 10mV max
  - Ibias < +/-1µA max (mostly I/O leakage)

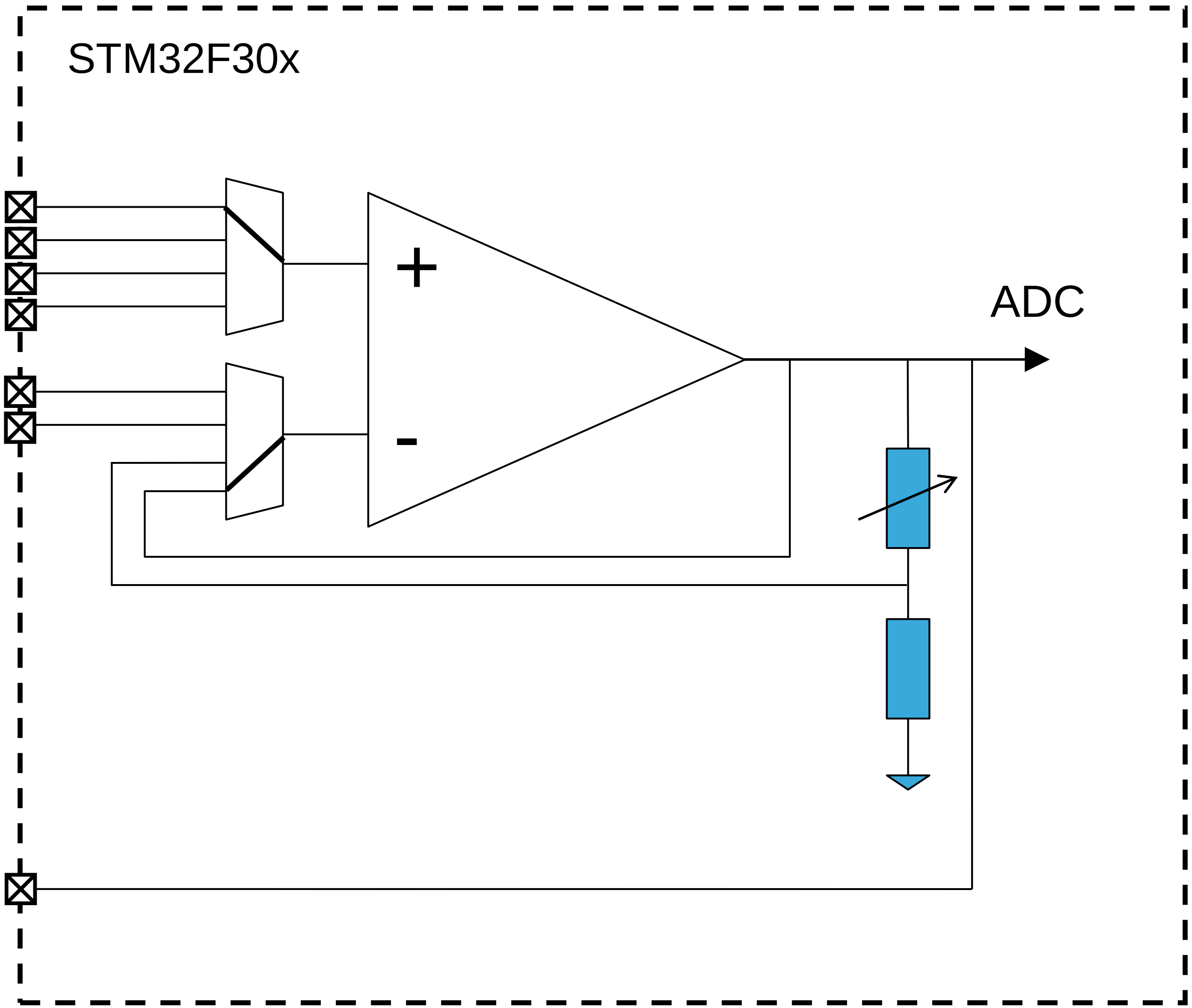
- Output stage
  - Iload < 500µA (sink and source)
  - Capacitive load < 50pF (stable when connected internally on ADC input)
  - GNDA + 100mV < Vout < VDDA – 100mV (Max)
    - 20mV @ 100µA I out

- Speed
  - GBW: 8MHz
  - Slew rate 4.5V/µs
  - unity gain stable

Standalone mode, External Gain setting



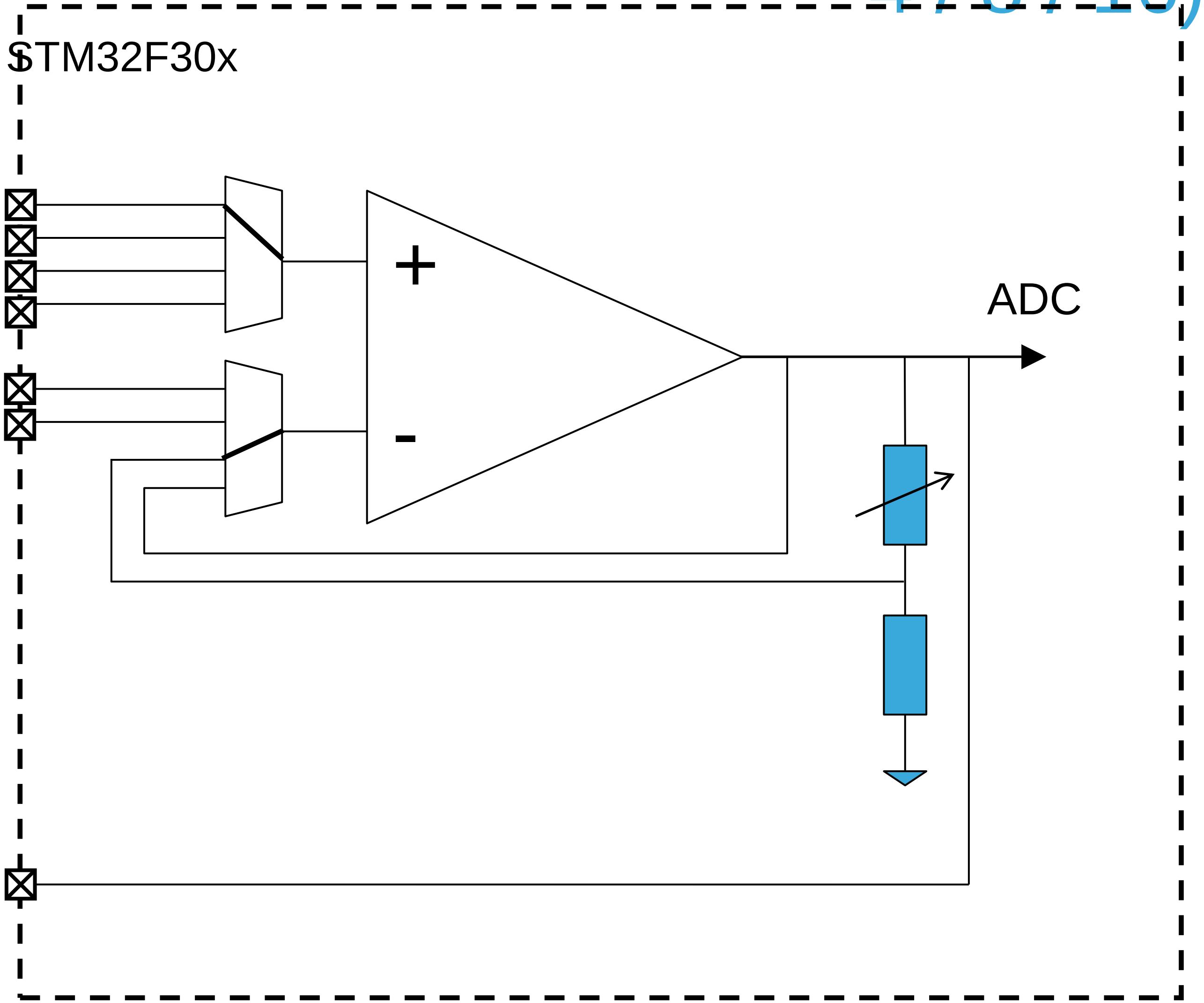
Follower mode



These I/Os are available

Always connected to OpAmp ouput

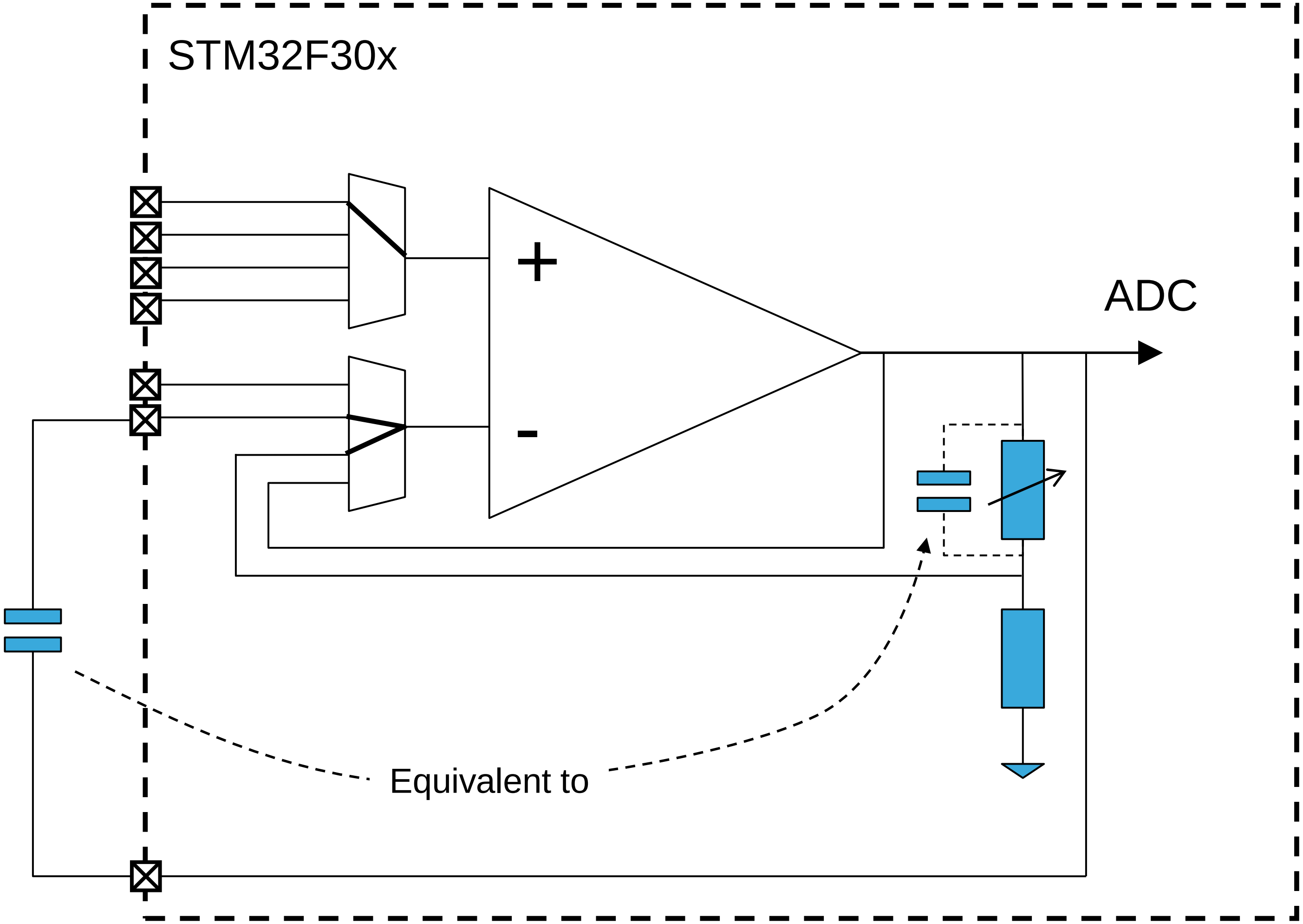
PGA Mode, Internal Gain setting (Gain = 2 / 4 / 8 / 16)



These I/Os are available

Always connected to OpAmp ouput.

PGA Mode, Internal Gain setting (Gain = 2 / 4 / 8 / 16) with Inverting input used for filtering.



Allows optional low-pass filtering

NB: gain dependant cut-off frequency

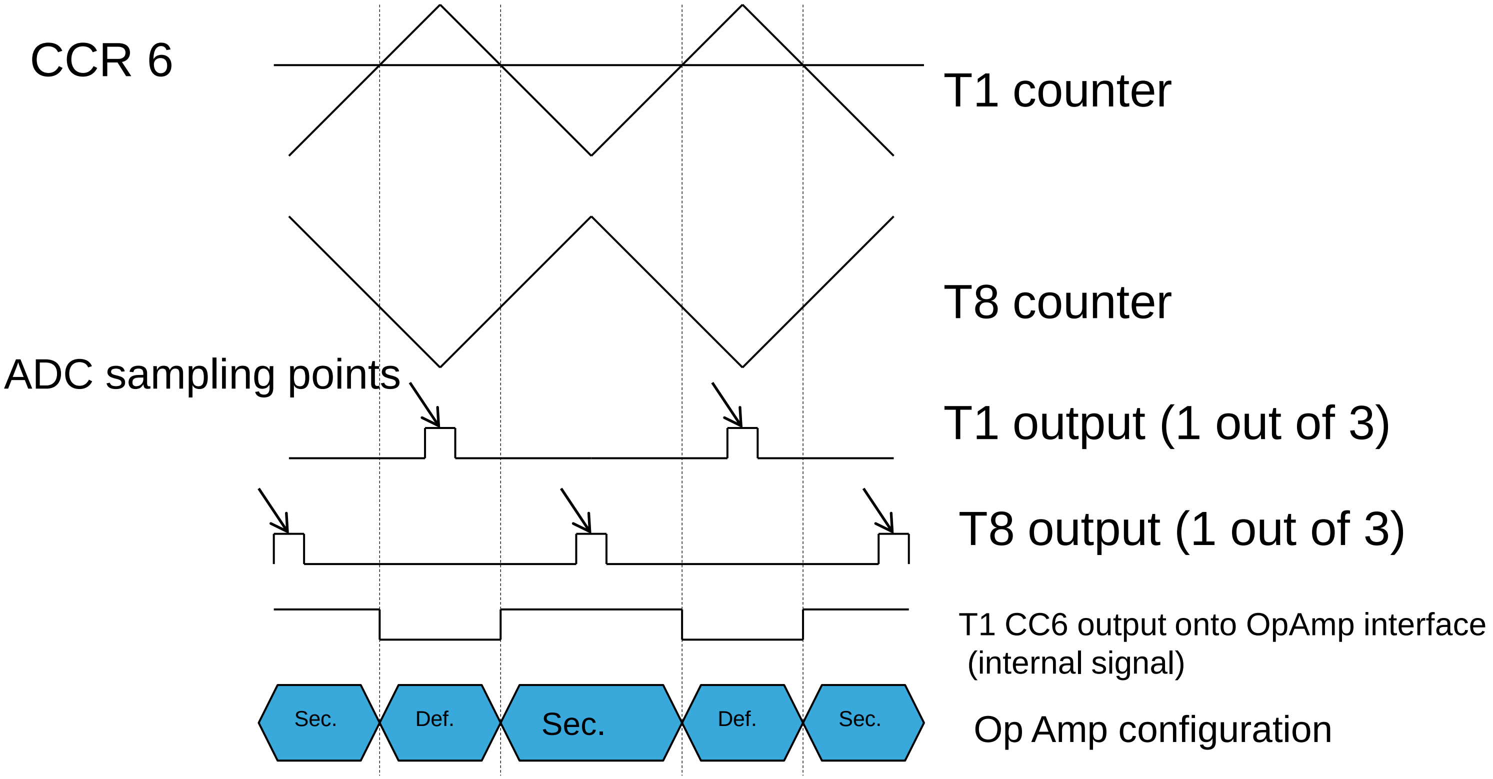
PGA network: 5.4K/5.4K (G = 2), 16.2K/5.4K (G = 4), 37.8K/5.4K (G = 8) 40.5K/2.7K (G = 16)

Timer Controlled Multiplexer mode (1/2)

414

- This mode allows switching automatically from one inverting (or non inverting) input to another inverting (or non inverting) input.
  - Benefit: useful in dual motor control with a need to measure the currents on the 3 phases on a first motor and then on the second motor.
- The automatic switch is triggered by TIM1 CC6 output arriving on the OPAMP input multiplexers.
- The Timer Controlled Multiplexer mode is enabled by setting TCM_EN bit.
- If TCM_EN bit is set, inverting and non inverting input selection is done using VPS_SEL and VMS_SEL bits.
- If TCM_EN bit is reset, inverting and non inverting input selection is done using VP_SEL and VM_SEL bits.

Timer Controlled Multiplexer mode (2/2)



OPAMP calibration

416

- It is possible to do the trimming of every opamp offset.

- At startup, trimmed offset values are initialized with the preset ‘factory’ trimming value

- The user can switch from the ‘factory’ values to the ‘user’ trimmed values using the USER_TRIM bit in the OPAMP control register.

- The offset of each operational amplifier can be trimmed by programming the TRIMOFFSETN and TRIMOFFSETP bits in the OPAMP control register.

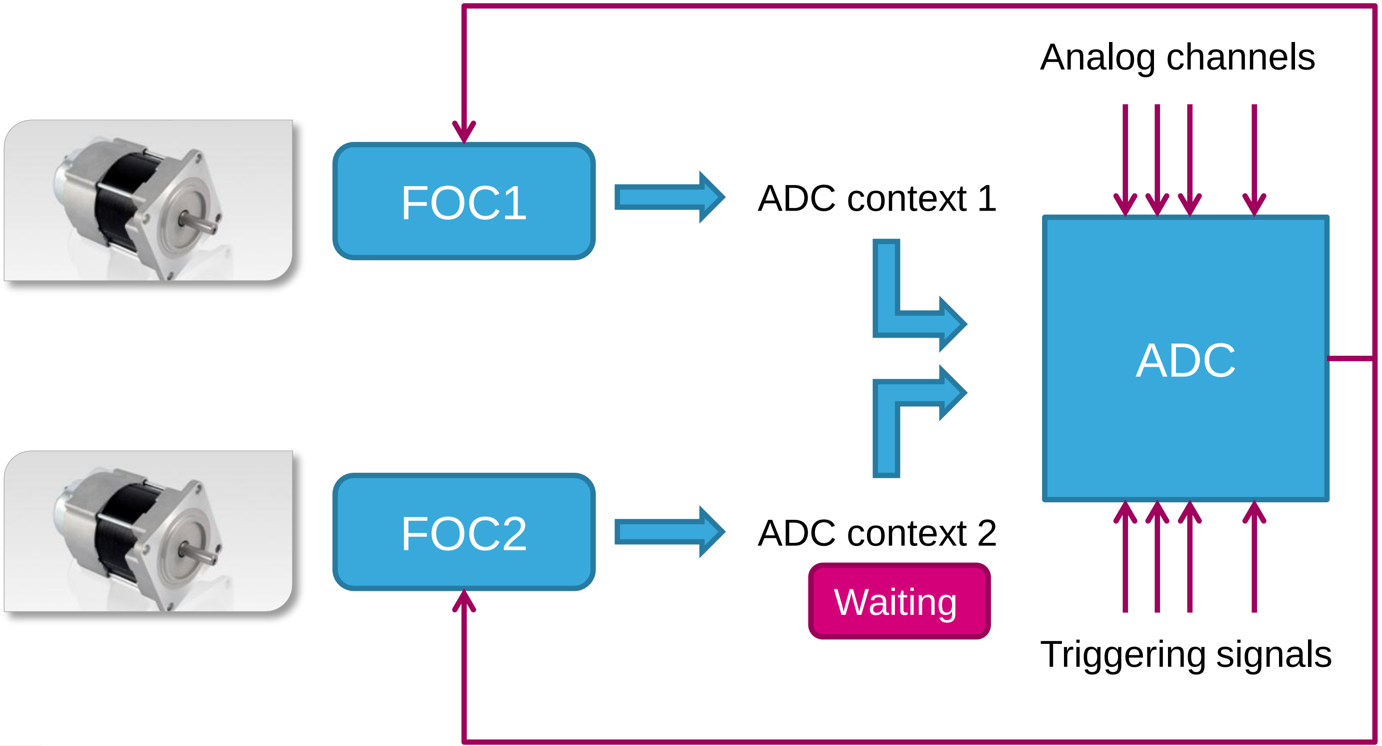
ADC context FIFO for dual three shunt current sampling in dual motor control (ADC sharing)

Patent pending

- Concept
  - Using two/three shunts topologies is required the simultaneous sampling of two analog quantities. This is actually implemented using two different ADC peripheral.
  - Dual simultaneous motor driving (2/3 shunt topologies) can be achieved using just two ADC peripheral if the sampling of each motor current is done in different times (ADC sharing).
  - The FOC algorithm of each motor can request the ADC conversions while the previous one is not already performed.
  - To perform automatically (saving CPU load) this mechanism the ADC context FIFO has been implemented.

ADC context FIFO

- FOC1 requires a conversion of channel x triggered by signal y (ADC context 1)
- FOC2 requires a conversion of channel n triggered by signal m (ADC context 2) but the ADC has been reserved so the context is stored in the FIFO
- Signal y triggers the conversion and the result is sent to FOC1. The FIFO go ahead programming the context 2
- Signal m triggers the conversion and the result is sent to FOC2.



For simplicity only one ADC is used in this Example

STM32F37x Specific Features/ peripherals

Sigma delta analog to digital converter (SDADC)

SDADC introduction (1/2)

- Sigma delta principle inside STM32:
  - High precision (new applications: medical, metering, gaming)
  - Excellent linearity (simplifies calibration)
  - No sample & hold

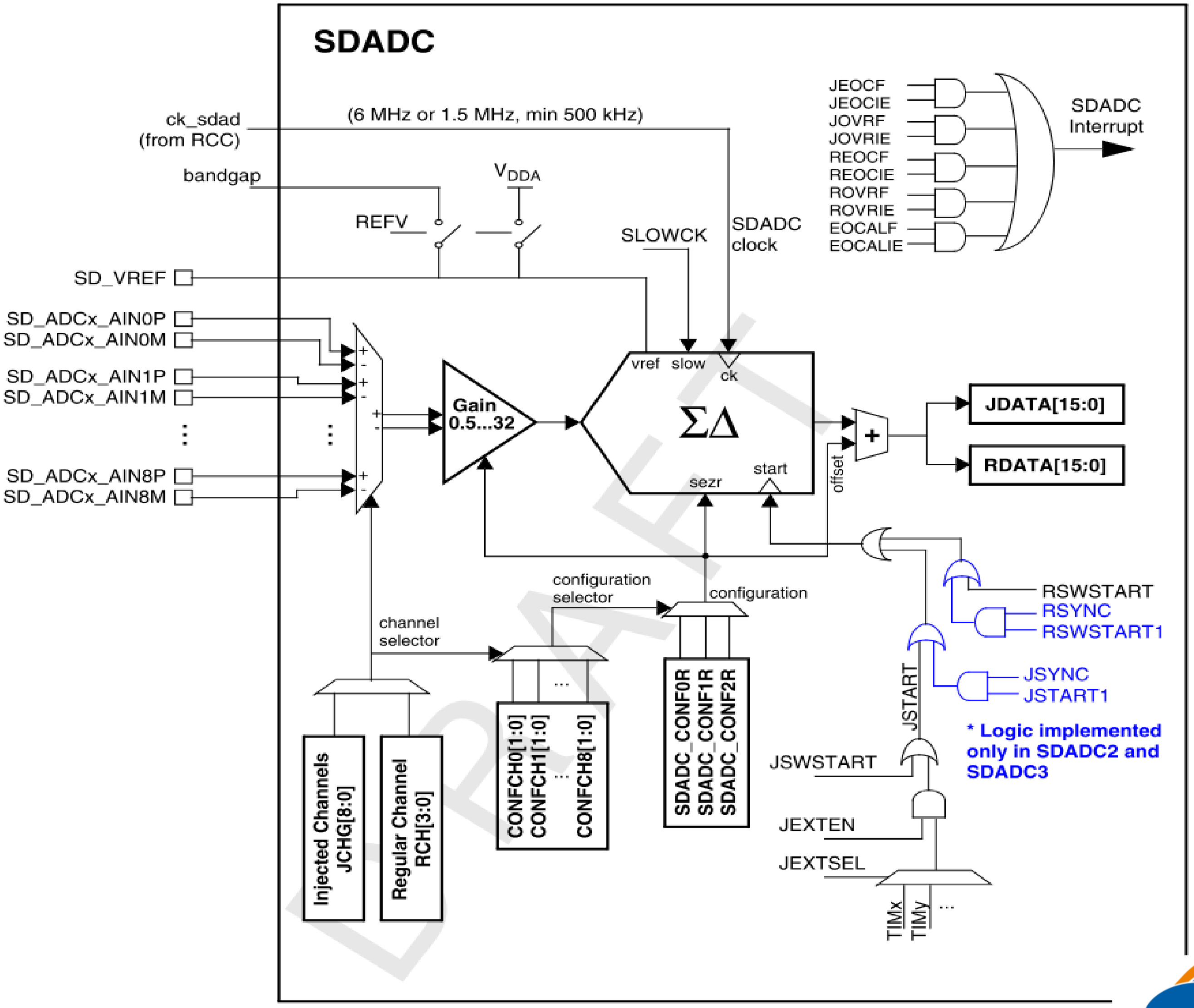
- Main properties:
  - 3 Σ-Δ ADCs in all packages (19 single ended and 10 differential inputs max.)
  - 16-bit resolution, ENOB = 14 bits (SNR = 89dB)
  - Low power modes:
    - Slow (speed reduced 4x): up to 600uA (instead of 1200uA in run mode)
    - Standby: up to 200uA, wakeup time 50us
    - Power down: up to 10uA, wake up time 100us
  - Internal or external reference voltage usage
  - Independent power supply pins: SDADCx_VDD
  - Conversion rates:
    - Up to 50ksps in fast mode (single channel)
    - Up to 16.6ksps in normal mode (multiple channels)
  - 7 programmable gains: ½, 1, 2, 4, 8, 16*, 32* (* = digital gains)

SDADC introduction (2/2)

- Next features:
  - 9 single ended inputs or 5 differential inputs per one SDADC (or combination)
  - DMA capability to transfer data to RAM (conversion when CPU in sleep mode)
  - Triggers:
    - Software
    - Timer
    - External pin
    - Synchronization to first SDADC (SDADC1)
  - Signed output data format (16-bit signed number)
  - Zero offset calibration
  - 3 measuring modes – per analog channel selection:
    - Single ended referenced to zero
    - Single ended offset mode
    - Differential mode
  - Interrupts and flags:
    - Interrupts: EOCAL, REOC, JEOC, ROVR, JOVR
    - Flags: STABIP, CALIBIP, RCIP, JCIP

SDADC block diagram

- One SDADC configuration:



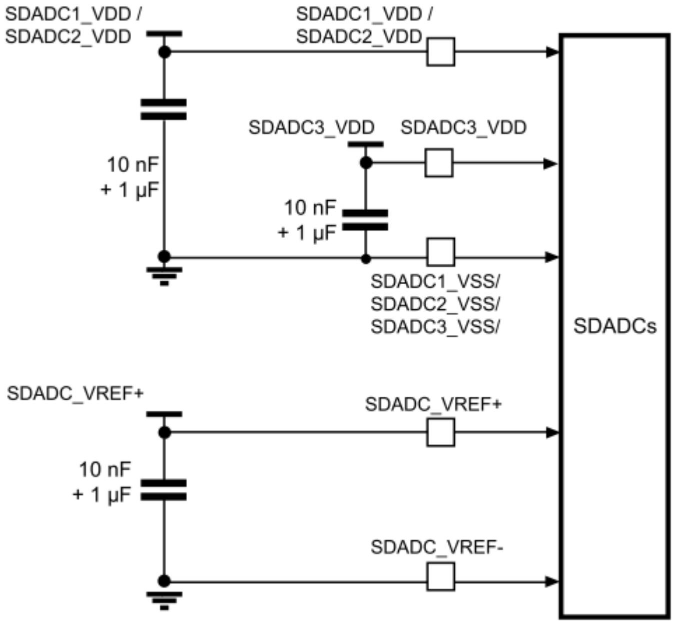
*** Logic implemented only in SDADC2 and SDADC3**

SDADC pins

Name	Signal type	Remarks
SDADCx_VDD	Input, analog Supply	Analog power supply. Must be greater than 2.4 V (or 2.2 V in Slow mode) and less than 3.6 V.
SDADCx_VSS	Input, analog supply ground	Analog ground power supply.
SDADCx_AIN[8:0]P	Analog input	Positive differential analog inputs for the 9 channels
SDADCx_AIN[8:0]M	Analog input	Negative differential analog inputs for the 9 channels.
SD_VREF+	Input or In/Out, positive analog Reference	When the external reference is selected (REFV=00), this pin must be driven externally to a voltage between 1.1 V and SDADCxVDD (minimum for x=1..3).When an internal reference is selected (REFV is 01, 10, or 11), this pin must have an external capacitance connected to SD_VREF-
SD_VREF-	Input, negative analog reference	This pin, when present, must be driven to the same voltage level as SDADCxVSS.

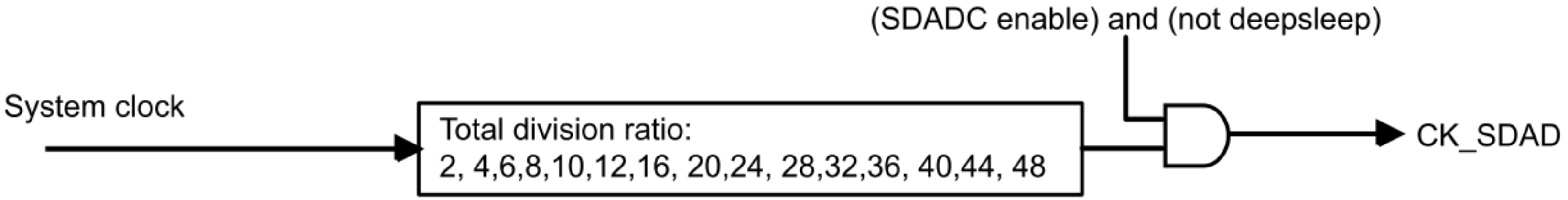
SDADC power supply and reference voltages

- Power supply:
  - Independent power supplies:
    - SDADC1/2 _VDD– for SDADC1 and SDADC2
    - SDADC3_VDD – for SDADC3
    - SDADCx_VSS – common for all SDADCs
  - Voltage range:
    - Full speed mode operation: 2.4V – 3.6V
    - Slow mode operation: 2.2V – 3.6V



- Reference voltage selection:
  - Internal:
    - Internal bandgap voltage: 1.2V
    - Internal bandgap voltage amplified by 1.5x : 1.8V
    - VDDA power supply
  - External
    - Dedicated SDADC_VREF+ , SDADC_VREF- pins
    - Voltage range 1.1V – SDADCx_VDD

SDADC clock



- Clock management:
  - System clock divided by divider (from 2 to 48, 50% duty cycle)
  - Clock range:
    - max. 6MHz – standard conversion clock
    - max. slow mode clock 1.5MHz – reduced speed, reduced power, lower voltage operation
    - min. clock speed = 500kHz

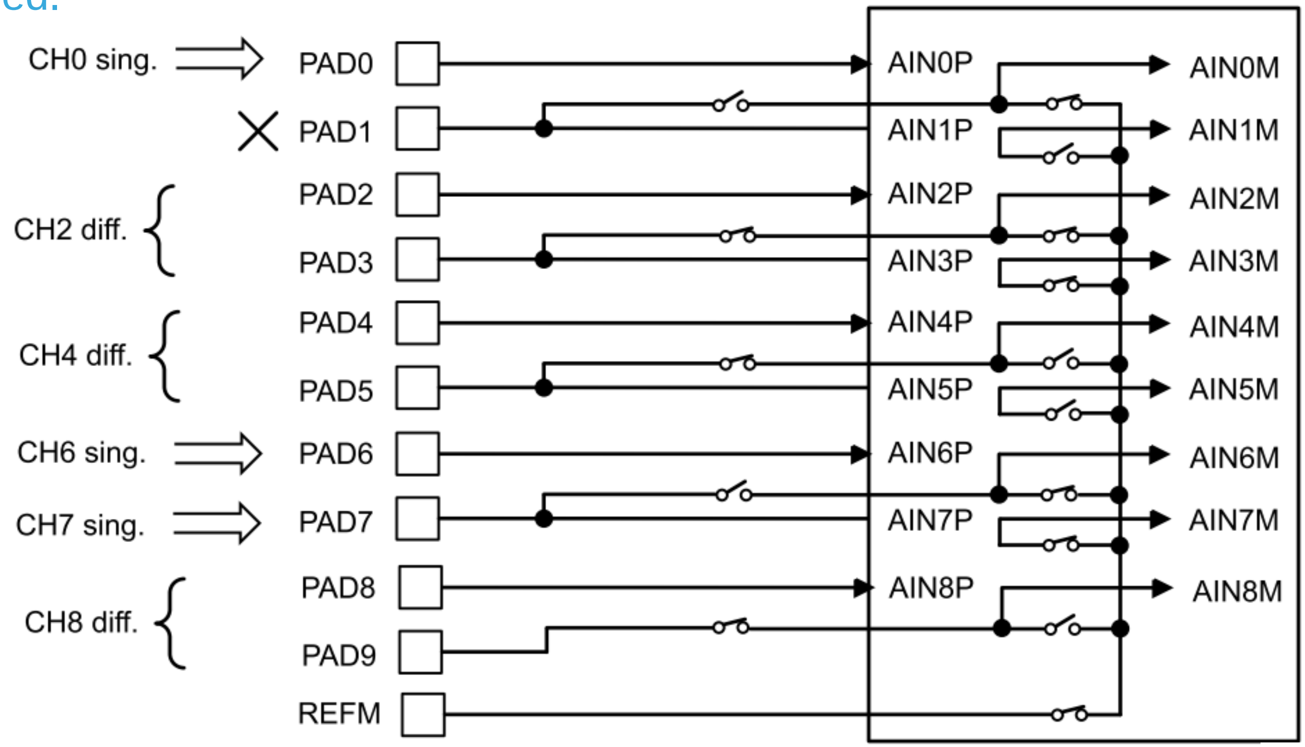
Input channel configurations

- Measurement modes:
  - Differential mode:
    - Used both SDADC analog channel inputs: SDADCx_AINxP and SDADCx_AINxM
    - Signed result: 0x8000 – 0x7FFF (-32768 – 32767)
  - Single ended modes:
    - Offset mode: as differential mode with minus input internally grounded (reduced dynamic range of SDADC – only positive range: 0x0000 – 0x7FFF)
    - Referenced to zero: minus input internally grounded but offset injected to have full dynamic range (zero voltage corresponds to code -32768)

- Three SDADC configuration registers (SDADC_CONFxR, x = 0..2) => 3 possible configurations:
    - In each register is channel configuration:
      - Measurement mode (differential or single ended)
      - Gain (½ , 1, 2, 4, 8, 16, 32)
      - Offset calibration value (stored here after offset calibration)
      - Common voltage used during offset calibration (VSSA, VDDA, VDDA/2)
  - Each SDADC analog channel is assigned to one configuration register
    - Example: 3 analog channels in application
      - Channel 0 uses SDADC_CONF0R
      - Channel 1 and channel 2 use SDADC_CONF1R (same gain and measuring mode)

Channels configuration example

- Mixed configurations – example of input pins connection:
  - CH2, CH4 and CH8 are used as differential.
  - CH0, CH6 and CH7 are used in single-ended mode.
  - REFM is used – VSSA.
  - PAD 1 is not used.



Regular and injected conversions

- Injected conversions
  - Injected group is defined as bitfield in register – each one bit corresponds to one channel
  - Selected channels in the injected group are always converted sequentially (from lowest selected channel) – scan mode
  - Triggers:
    - Software (writing ‘1’ to the JSWSTART bit)
    - External pin
    - Timers
    - Synchronous with SDADC1

- Regular conversions
  - Channel selection is defined as channel number in register
  - Cannot run in scan mode
  - Triggers:
    - Software (writing ‘1’ to the RSWSTART bit)
    - Synchronous with SDADC1

Standard, slow, low power conversion modes

- Standard mode:
  - Normal:
    - Multiplexing more channels
    - One conversion takes 360 cycles (16.6ksps @ 6MHz)
  - Fast continuous (FAST = 1):
    - On one channel only in continuous mode – regular channel or one injected channel selected
    - One conversion takes 120 cycles (50ksps @ 6MHz)

use this

- Slow mode (SLOWCK = 1):
  - Reduced power consumption (~600uA consumption), operation from 2.2V
  - Limited clock speed – up to 1.5MHz (so 4x reduced also conversion rate)

- Standby when idle (SBI = 1):
  - SDADC goes to standby when no conversion (~200uA consumption)
  - Needed time for wakeup from power down – 50us

- Power down when idle (PDI = 1):
  - SDADC goes to power down when no conversion (~10uA consumption)
  - Needed time for wakeup from power down – 100us

Request precedence

- Priority order of SDADC operations:
  1. Calibration sequence
  2. Injected conversions
  3. Regular conversions

- But:
  - Conversion which is already in progress is never interrupted by the request for another action (current conversion is finished first)
  - Request is ignored if a like action is already pending or in progress
  - No action can start before stabilization has finished (wakeup from power down or standby mode)

SDADC calibration

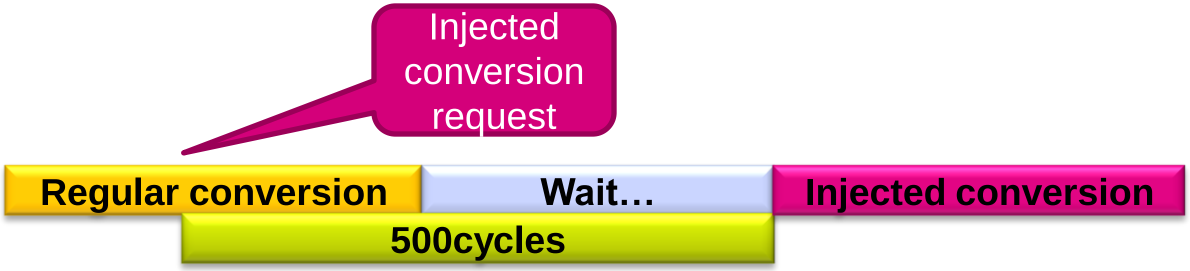
- General properties for sigma delta converters:
  - Perfect linearity (due to 1-bit converter and oversampling)
  - Resolution increases with decreasing data rate
  - But large offset and gain error (need calibration)

- Offset calibration:
  - Principle:
    - Short internally both channel inputs (positive and negative)
    - Perform conversion and store result to configuration register(s)
    - During standard conversion subtract from result the calibrated value
  - Implementation in STM32F37x:
    - Set in configuration registers:
      - required gain (1/2 .. 32)
      - common mode for calibration (VSSA, VDDA, VDDA/2)
    - Set how many configurations to calibrate (CALIBCNT[1:0] bits)
    - Start calibration by setting bit STARTCALIB
    - Calibration sequence then executes on given gain(s) :
      - Calibration values are stored into configuration registers (OFFSETx[11:0] bits)
      - 30720 cycles (5.12 ms at 6 MHz) for one configuration register
    - Calibration data are automatically subtracted from each conversion data

Deterministic timing

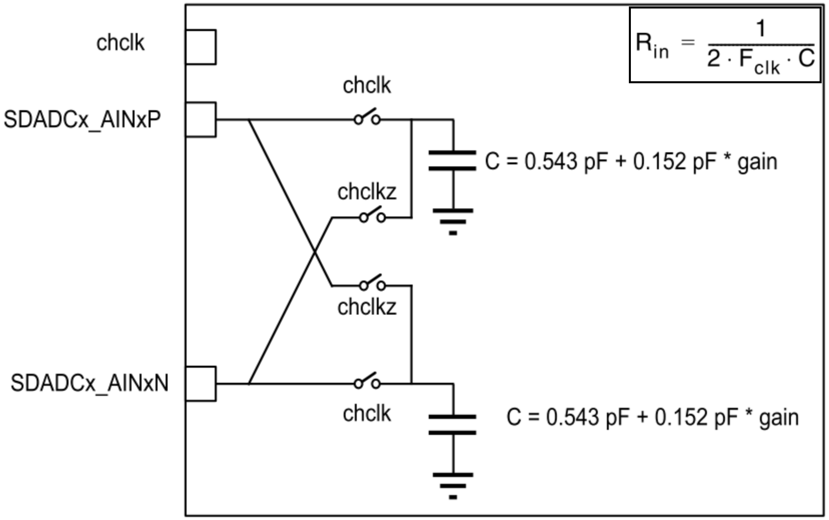
- Application requirements:
  - Launching conversion in precise intervals (e.g. FFT sampling by timer trigger)
  - Problem: waiting for some ongoing (regular) conversion

- Solution in SDADC:
  - Start of each injected conversion with delay during which cannot be started regular conversion
  - When bit JDS = 1 (Injected Delay Start) the start of each injected conversion is delayed:
    - by 500 cycles if PDI = 0 (power down when idle)
    - by 600 cycles if PDI = 1, SLOWCK = 0 (because wakeup from power down takes 600 cycles)

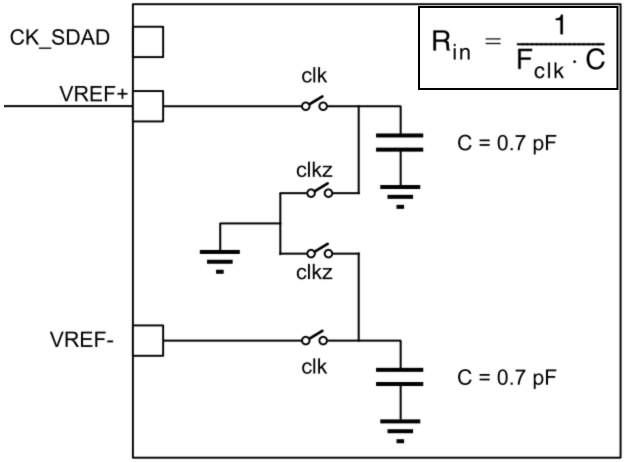


Inputs impedances

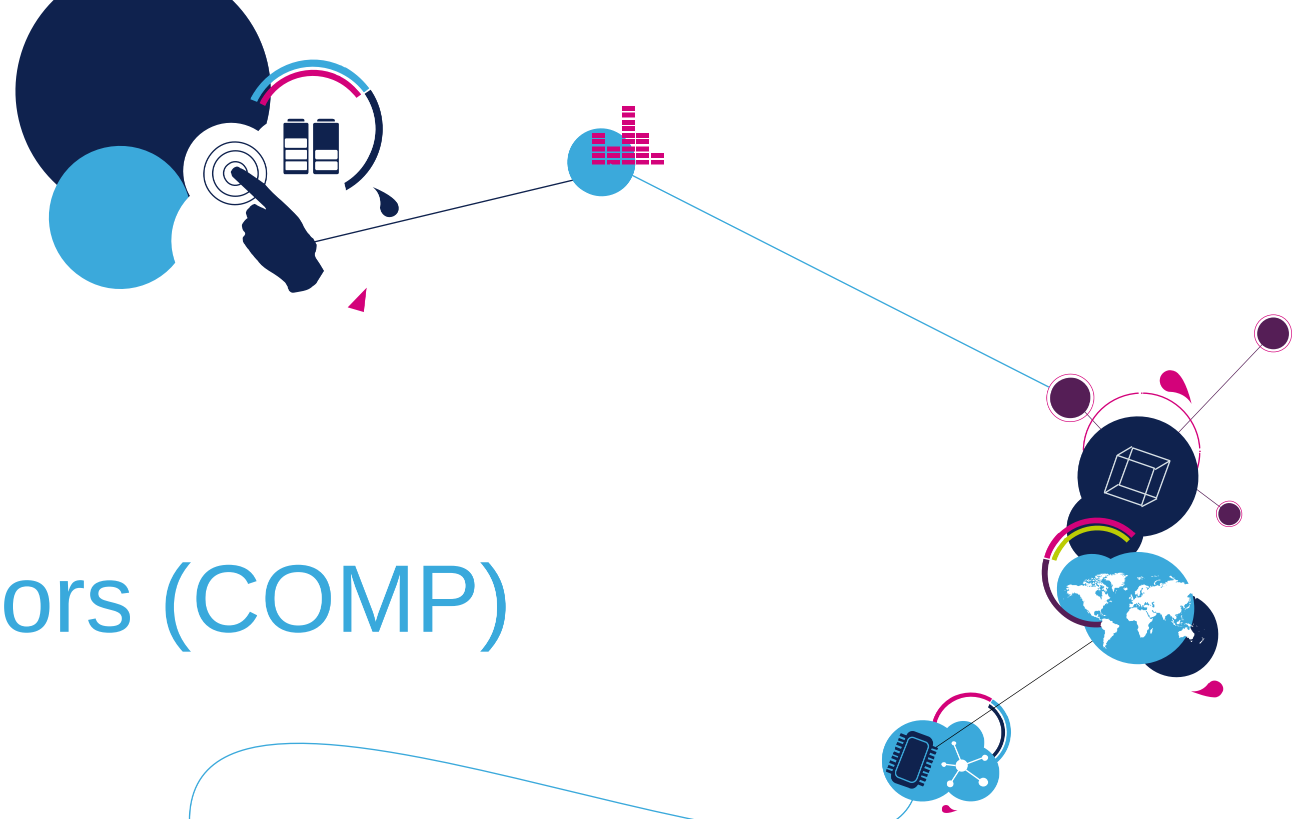
- Analog inputs impedance:
  - Depends from:
    - selected SDADC clock
    - analog gain (0.5 – 8)
    - conversion is in progress
  - Switching capacitance character
  - Range (examples):
    - 540kΩ @ 1.5MHz, gain = 0.5
    - 135kΩ @ 6MHz, gain = 1
    - 47kΩ @ 6MHz, gain = 8


chclk
SDADCx_AINxP
SDADCx_AINxN
chclk
chclkz
chclkz
chclk
$R_{in} = \frac{1}{2 \cdot F_{clk} \cdot C}$
C = 0.543 pF + 0.152 pF * gain
C = 0.543 pF + 0.152 pF * gain

- Reference voltage input impedance:
  - Depends only from selected SDADC clock
  - Switching capacitance character
  - Range (6MHz – 1.5MHz):
    - ~ 230kΩ – 1000kΩ



Comparators (COMP)

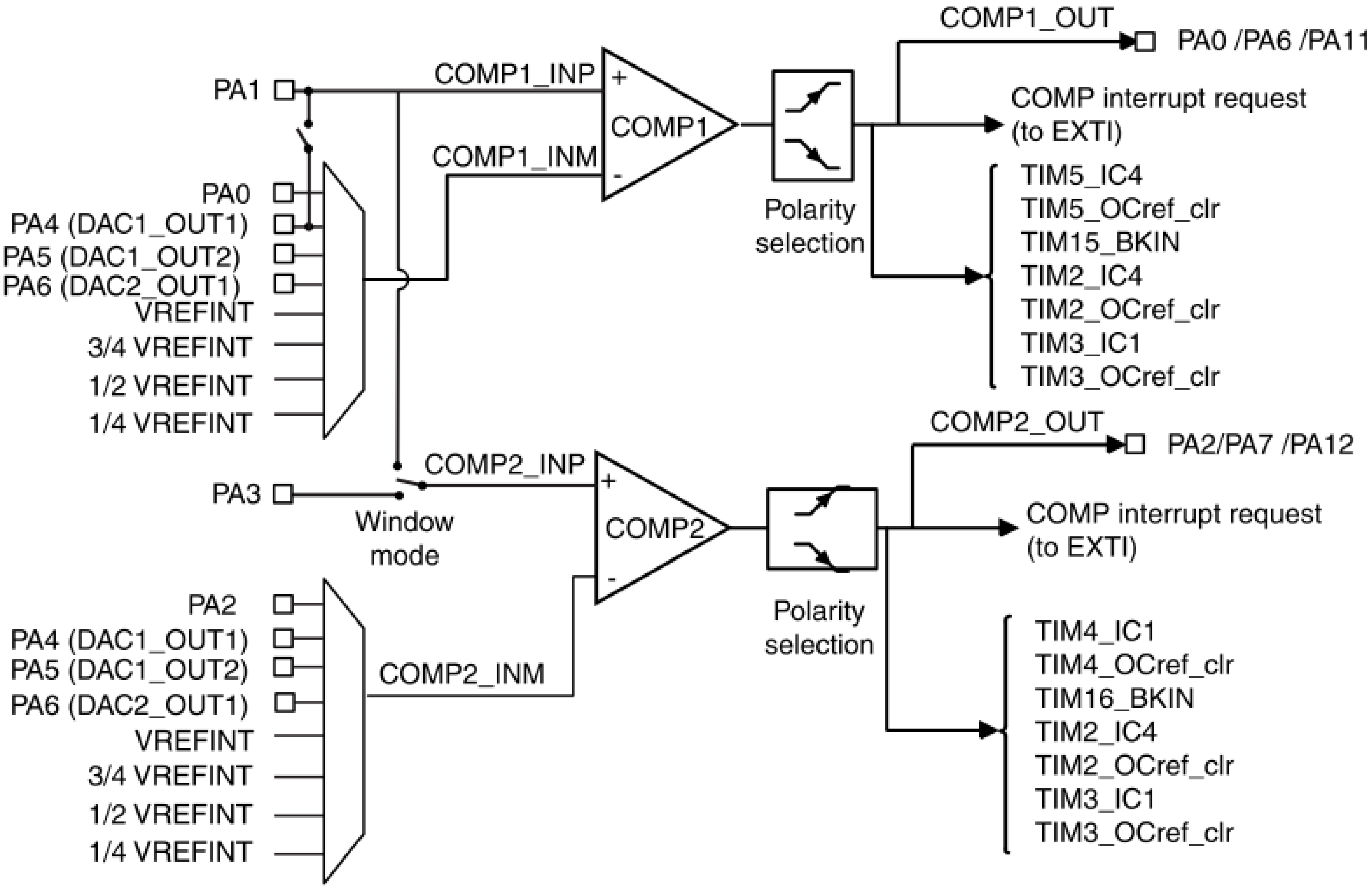


COMPARATORS (COMP)

- 2 general purpose comparators COMP1 and COMP2
  - Rail-to-rail inputs
  - Programmable speed / consumption: 4 modes
  - Programmable hysteresis: 4 levels
  - Inputs and outputs available externally - can be used as a standalone device without MCU interaction
  - Can be combined into a window comparator
  - Multiple choices for input thresholds and output redirection

- Can be used for:
  - Exiting low power modes on an analog event
  - Signal conditioning
  - Cycle-by-cycle current control (w/ DAC and TIM)
  - …

Block diagram for STM32F37x



BKIN: PWM's "Emergency stop" input
OCRefClear: PWM clear for cycle-by-cycle current controlle

COMP features

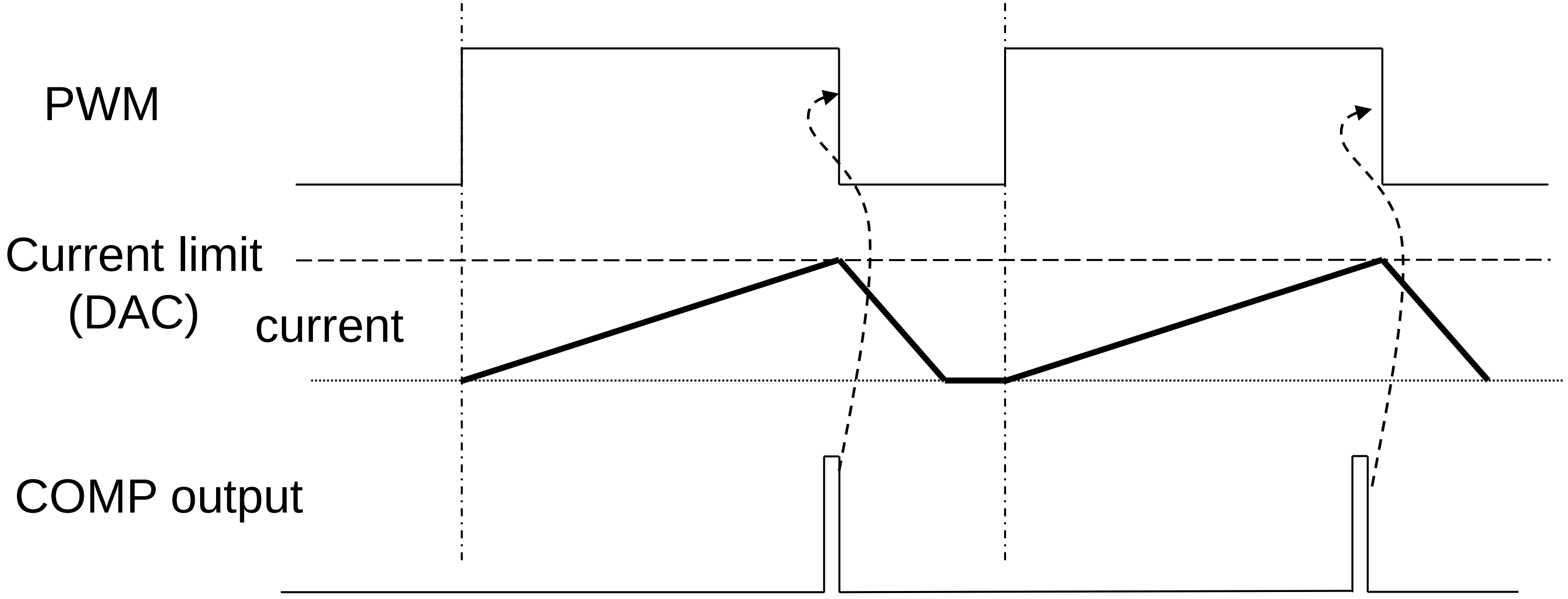
- Comparator Characteristics at a glance
  - Full operating voltage range 2V < Vdda < 3.6V
  - Propagation time vs consumption
    (characterized typ. 2.7 < Vdd < 3.6V, for 100 mV step with 10 mV overdrive, TBD)
    - High speed / full power
    - Medium speed / medium power
    - Low speed / Low power
    - Very low speed / Ultra-low power
    - Input offset: +/-4mV typ, +/- 20mV max
  - Programmable hysteresis: 0, 8, 15, 31 mV

- Fully asynchronous operation
  - Comparators working in STOP mode
  - No clock related propagation delay

- Functional safety (Class B)
  - The comparator configuration can be locked with a write-once bit

Application example

- Cycle-by cycle current control
  - Uses DAC for threshold and Ocref_clr timer input



STM32 Releasing your **creativity**

Thank you !

life.augmented

COMPELFEST