

STM32G4 – System/Memory

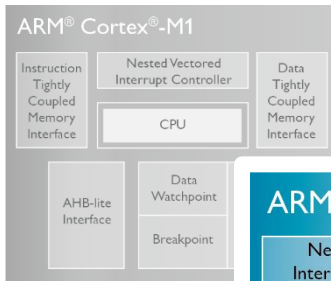


Cortex-Mx Core and Architecture

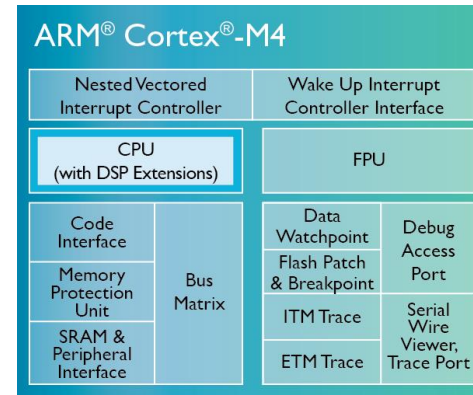
ARM Cortex-Mx evolution

3

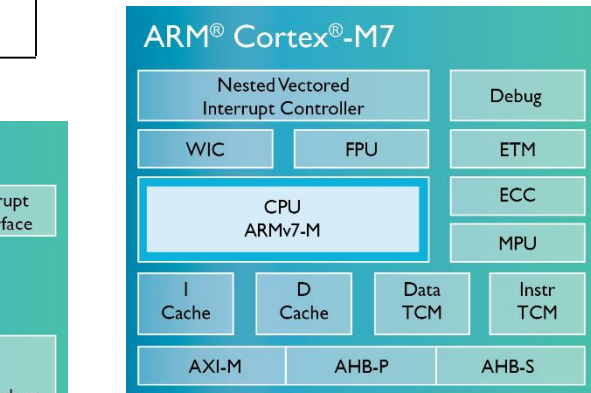
just for FPGAs



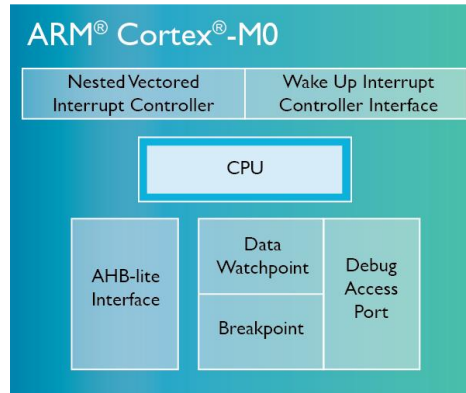
2007



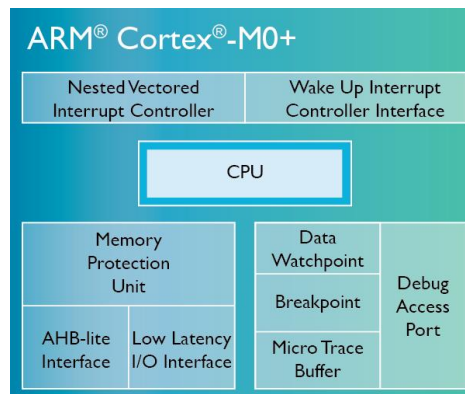
2010



2014 NEW!

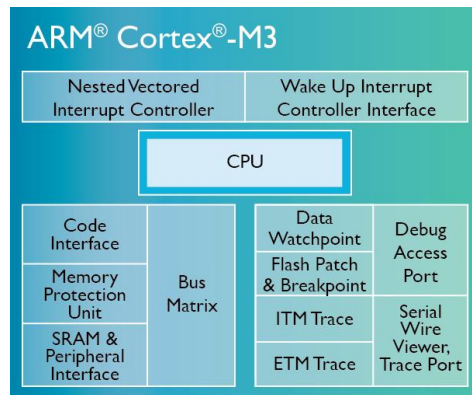


2009



2012

#1



2004



STM32G4 – ARM[®] Core

ARM[®] Cortex[®]-M4

Cortex-M processors

5

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

Cortex-M0 & M0+

“8/16-bit” applications

Cortex-M3

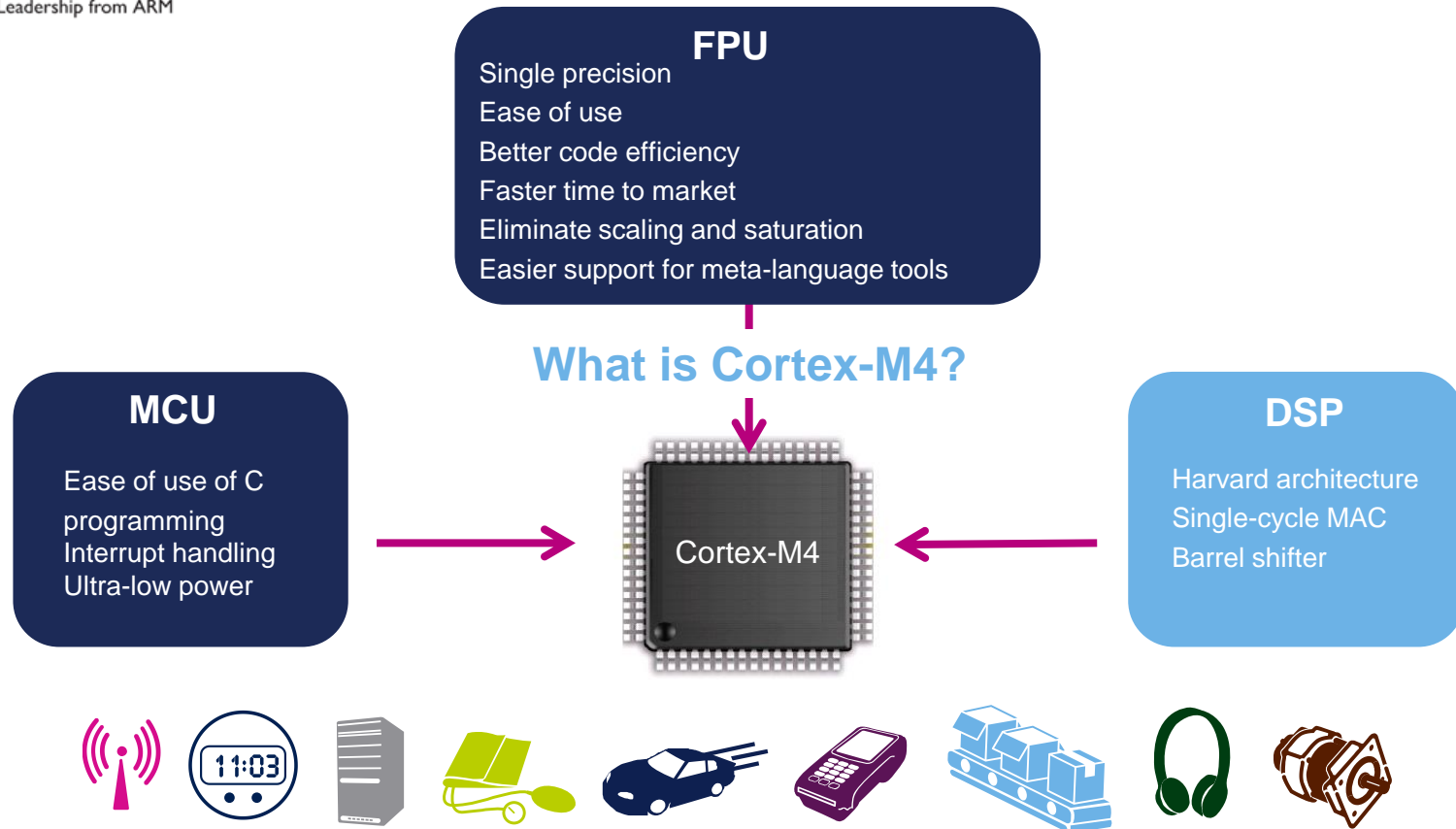
“16/32-bit” applications

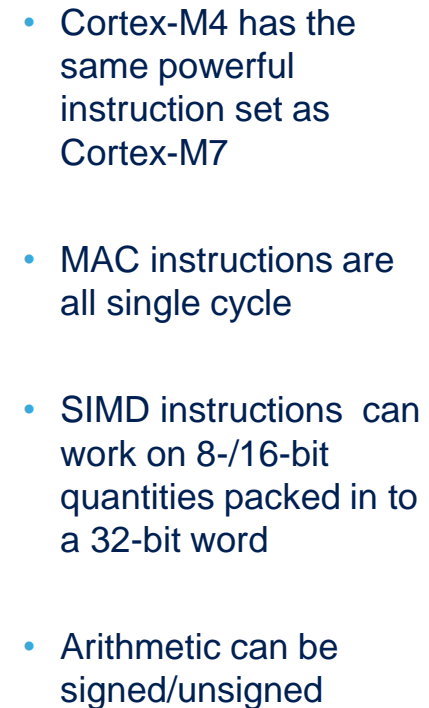
Cortex-M4

“32-bit/DSC” applications

Binary and tool compatible





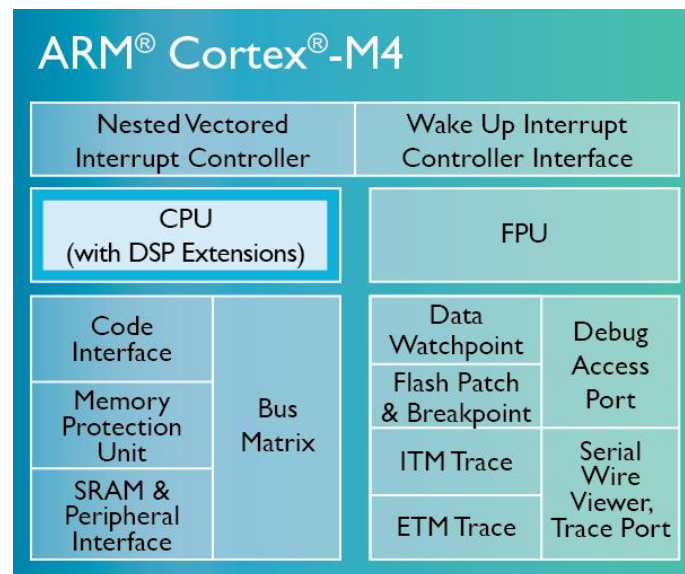


Source: ARM

Cortex-M4 Processor Overview

8

- **ARMv7E-M** Architecture
- **Harvard** architecture, **3-stage pipeline**
- **DIV in 12-cycles max, SIMD instructions**
- **Memory Protection Unit (MPU)**
- (Optional) Single Precision floating point (float)
 - ⇒ Included in current STM32 based on ARM Cortex-M4



12-cycles interrupt latency

Cortex-M performance benchmark

9

Benchmark	Dhrystone DMIPS/MHz(v2.1)- official	Dhrystone DMIPS/MHz(v2.1)- full optimization	Coremark/MHz (v1.0)
Cortex-M0	0.84	1.21	2.33
Cortex-M0+	0.94	1.31	2.42
Cortex-M3	1.25	1.89	3.32
Cortex-M4	1.25	1.95	3.40
Cortex-M7	2.14	2.55	5.01
Cortex-M23	0.98	-	2.5
Cortex-M33	1.5	-	3.86

Source : CoreMark.org website and ARM website

Cortex-M feature set comparison

10

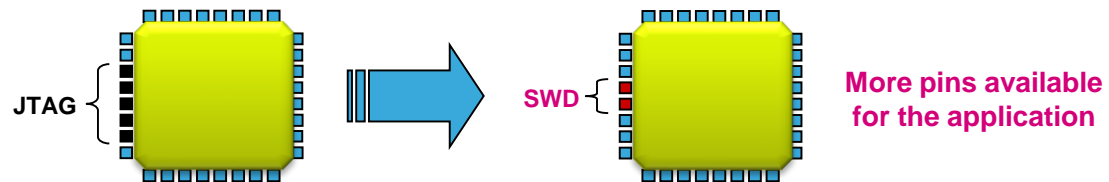
	Cortex-M0	Cortex-M0+	Cortex-M3	Cortex-M4
Architecture Version	V6M	V6M	v7M	v7ME
Instruction set architecture	Thumb, Thumb-2 system Instructions	Thumb, Thumb-2 System Instructions	Thumb + Thumb-2	Thumb + Thumb-2, DSP, SIMD, FP
DMIPS/MHz	0.95	0.95	1.25	1.25
Bus interfaces	1	1+I/O port	3	3
Integrated NVIC	Yes	Yes	Yes	Yes
Number interrupts available	1-32 + NMI	1-32 + NMI	1-240 + NMI	1-240 + NMI
Interrupt priorities available	4	4	8-256	8-256
Breakpoints, Watch points	4/2/0, 2/1/0	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)	Yes (Option)
Integrated trace option (ETM)	No	MTB (Option)	Yes (Option)	Yes (Option)
Fault Robust Interface	No	No	Yes (Option)	No
Single Cycle Multiply	Yes (Option)	Yes (Option)	Yes	Yes
Hardware Divide	No	No	Yes	Yes
WIC Support	Yes (Option)	Yes	Yes	Yes
Bit banding support	No	No	Yes	Yes
Single cycle DSP/SIMD	No	No	No	Yes
Floating point hardware	No	No	No	Yes (Option)
Bus protocol	AHB Lite	AHB Lite, I/O	AHB Lite, APB	AHB Lite, APB
CMSIS Support	Yes	Yes	Yes	Yes

Cortex-M3/-M4 Debug Capabilities

11



- Full JTAG and also Serial Wire Debug interface



- **Breakpoint and Watchpoint units**
 - 4 hardware breakpoints (besides BKPT instruction)
 - 2 hardware watchpoints
- **Serial Wire Viewer** for targeted low bandwidth data trace
 - Triggered by embedded break and watch points
 - Uses SWO signal
- **ETM capability for better real time debugging**
 - Instruction trace only
 - External signal triggering capability
 - Can be used in parallel with data watchpoint

- For more details, please refer to ARM website at the following link:
 - <http://www.arm.com/products/processors/cortex-m/cortex-m4-processor.php>



Cortex M4 – DSP features

ARM® Cortex® -M4

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

15

- 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

16

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

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

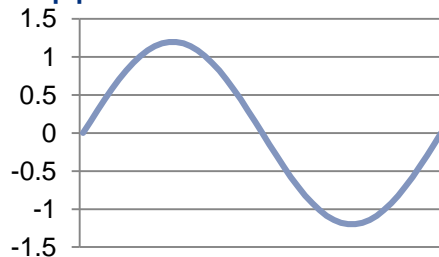
Saturated arithmetic

17

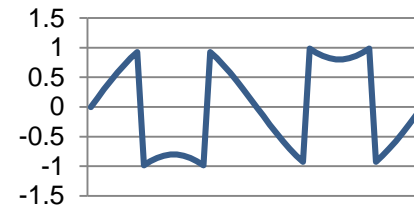
- Intrinsically prevents overflow of variable by clipping to min/max boundaries and remove CPU burden due to software range checks

- Benefits

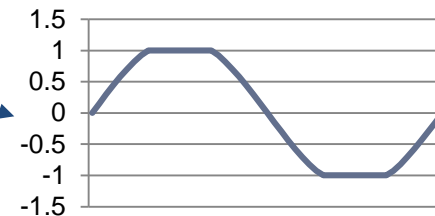
- Audio applications



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

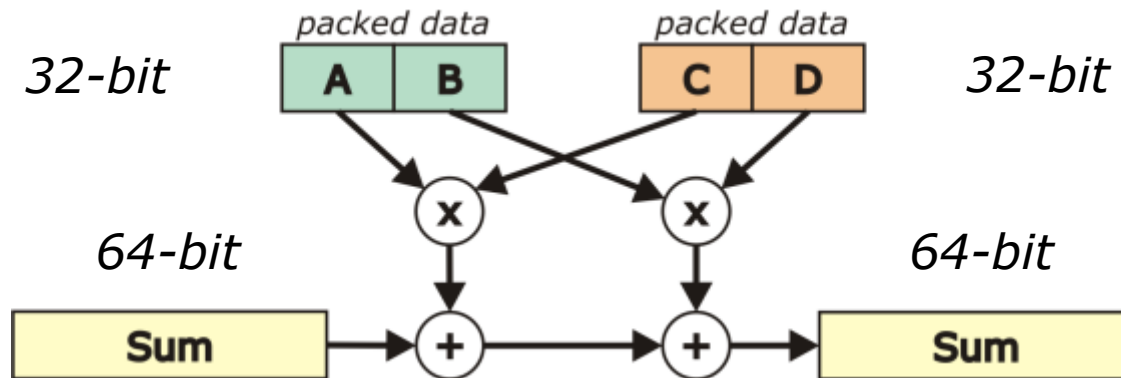
18

- 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 ($\text{Result} = 16 \times 16 + 16 \times 16 + 32$)
 - 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

19

- SIMD extensions perform multiple operations in one cycle
 $\text{Sum} = \text{Sum} + (A \times C) + (B \times D)$



- SIMD techniques operate with packed data

Cortex-M4 DSP instructions compared

20

CLASS	INSTRUCTION	Cycle counts	
		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

Cortex-M4 non-DSP instructions

21


CLASS	INSTRUCTION	Cycle counts	
		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

DSP lib provided for free by ARM

22

- 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

23

- 

CMSIS-DSP

Version 1.7.0

CMSIS DSP Software Library

General	CMSIS-Core(A)	CMSIS-Core(M)	Driver	DSP	NN	RTOS v1	RTOS v2	Pack	SVD	DAP	Zone
Main Page Usage and Descriptions Reference											

CMSIS-DSP

CMSIS DSP Software Library

Revision History of CMSIS-DSP

Deprecated List

Reference

Basic Math Functions

Fast Math Functions

Complex Math Functions

FILTERING Functions

High Precision Q31 Biquad Cascade Filter

Biquad Cascade IIR Filters Using Direct Form

Biquad Cascade IIR Filters Using a Direct Form II

Convolution

Partial Convolution

Correlation

Finite Impulse Response (FIR) Decimator

Finite Impulse Response (FIR) Filters

Finite Impulse Response (FIR) Lattice Filter

Finite Impulse Response (FIR) Spanner Filter

Infinite Impulse Response (IIR) Lattice Filter

Least Mean Square (LMS) Filters

Normalized LMS Filters

Finite Impulse Response (FIR) Interpolator

Matrix Functions

Transform Functions

Controller Functions

Statistics Functions

Support Functions

Interpolation Functions

Examples

Data Structures

Data Fields

CMSIS DSP Software Library

Introduction

This user manual describes the CMSIS DSP software library, a suite of common signal processing functions for use on Cortex-M processor based devices.

The library is divided into a number of functions each covering a specific category:

 - Basic math functions
 - Fast math functions
 - Complex math functions
 - Filters
 - Matrix functions
 - Transform functions
 - Motor control functions
 - Statistical functions
 - Support functions
 - Interpolation functions

The library has separate functions for operating on 8-bit integers, 16-bit integers, 32-bit integer and 32-bit floating-point values.

Using the Library

The library installer contains prebuilt versions of the libraries in the Lib folder:

 - arm_cortexM7fdp_math.lib (Cortex-M7, Little endian, Double Precision Floating Point Unit)
 - arm_cortexM7fdp_math.lib (Cortex-M7, Big endian, Double Precision Floating Point Unit)
 - arm_cortexM7fdp_math.lib (Cortex-M7, Little endian, Single Precision Floating Point Unit)
 - arm_cortexM7fbdp_math.lib (Cortex-M7, Big endian and Single Precision Floating Point Unit on)
 - arm_cortexM7fbdp_math.lib (Cortex-M7, Little endian)
 - arm_cortexM7fbdp_math.lib (Cortex-M7, Big endian)
 - arm_cortexM4f_math.lib (Cortex-M4, Little endian, Floating Point Unit)
 - arm_cortexM4bf_math.lib (Cortex-M4, Big endian, Floating Point Unit)
 - arm_cortexM4f_math.lib (Cortex-M4, Little endian)
 - arm_cortexM4b_math.lib (Cortex-M4, Big endian)
 - arm_cortexM3f_math.lib (Cortex-M3, Little endian)
 - arm_cortexM3b_math.lib (Cortex-M3, Big endian)
 - arm_cortexM0f_math.lib (Cortex-M0 / Cortex-M0+, Little endian)
 - arm_cortexM0b_math.lib (Cortex-M0 / Cortex-M0+, Big endian)
 - arm_armv8MBL_math.lib (Armv8-M Baseline, Little endian)
 - arm_armv8MBL_math.lib (Armv8-M Mainline, Little endian)
 - arm_armv8MBL_math.lib (Armv8-M Mainline, Little endian, Single Precision Floating Point Unit)
 - arm_armv8MBL_math.lib (Armv8-M Mainline, Little endian, DSP instructions)
 - arm_armv8MBL_math.lib (Armv8-M Mainline, Little endian, DSP instructions, Single Precision Floating Point Unit)

Generated on Wed Jul 10 2019 15:20:40 for CMSIS-DSP

- 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>

IIR – single cycle MAC benefit

25

	<u>Cortex-M3</u> <u>cycle count</u>	<u>Cortex-M4</u> <u>cycle count</u>
xN = *x++;	2	2
yN = xN * b0;	3-7	1
yN += xNm1 * b1;	3-7	1
yN += xNm2 * b2;	3-7	1
yN -= yNm1 * a1;	3-7	1
yN -= yNm2 * a2;	3-7	1
*y++ = yN;	2	2
xNm2 = xNm1;	1	1
xNm1 = xN;	1	1
yNm2 = yNm1;	1	1
yNm1 = yN;	1	1
Decrement loop counter	1	1
Branch	2	2

$$y[n] = b_0x[n] + b_1x[n-1] + b_2x[n-2] - a_1y[n-1] - a_2y[n-2]$$

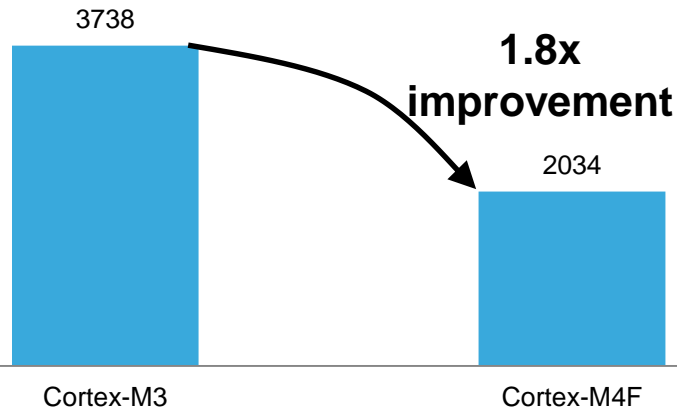
- Only looking at the inner loop, making these assumptions
 - Function operates on a block of samples
 - Coefficients b0, b1, b2, a1, and a2 are in registers
 - Previous states, x[n-1], x[n-2], y[n-1], and y[n-2] are in registers
- Inner loop on Cortex-M3 takes 27-47 cycles per sample
- Inner loop on Cortex-M4 takes 16 cycles per sample

- Cortex-M4F benefits Vs. Cortex-M3

- Improvement in code size (A)
- Improvement in performance (B)

Complex FFT 64 points (CFFT-64)

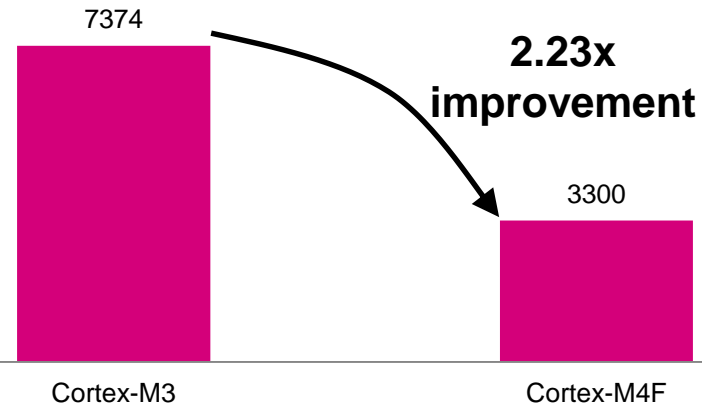
■ CFFT-64 (Q15 data) code size in bytes



(A)

Complex FFT 64 points (CFFT-64)

■ CFFT-64 Q15 execution time (# cycles)

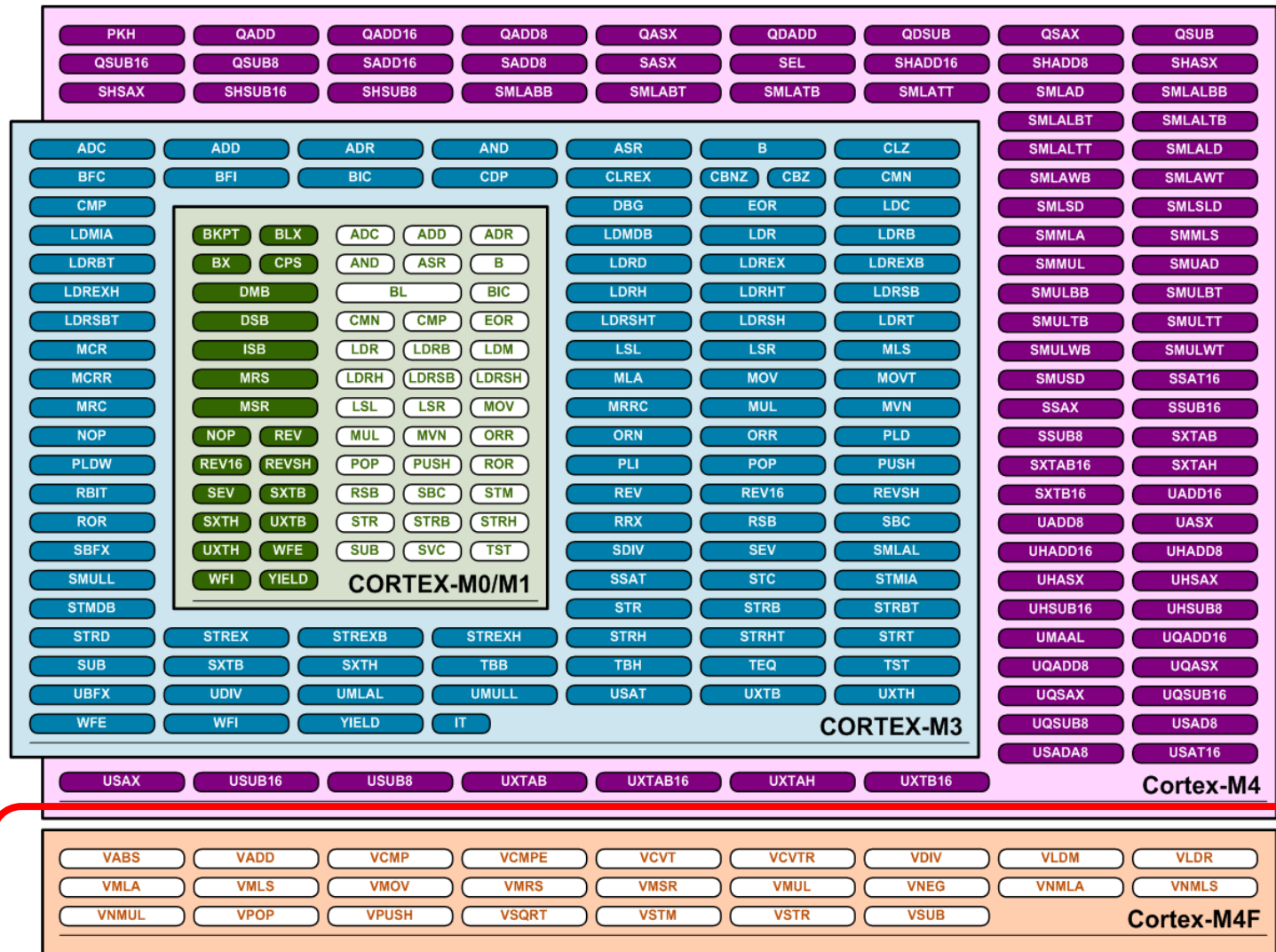


(B)



Floating Point Unit

FPU instructions



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



C language example

30

```
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

31

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

32

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

Cortex-M4F FPU Benefits

■ FIR float code size in bytes

1074

1.5x
improvement

696

Cortex-M3

Cortex-M4F

(A)

FIR float execution time (# cycles)

■ FIR float execution time (# cycles)

1593604

17.8x
improvement

Best compromise
Development time
vs. performance

89136

Cortex-M3

Cortex-M4F

(B)



System Overview

STM32G474 Memory Mapping

34

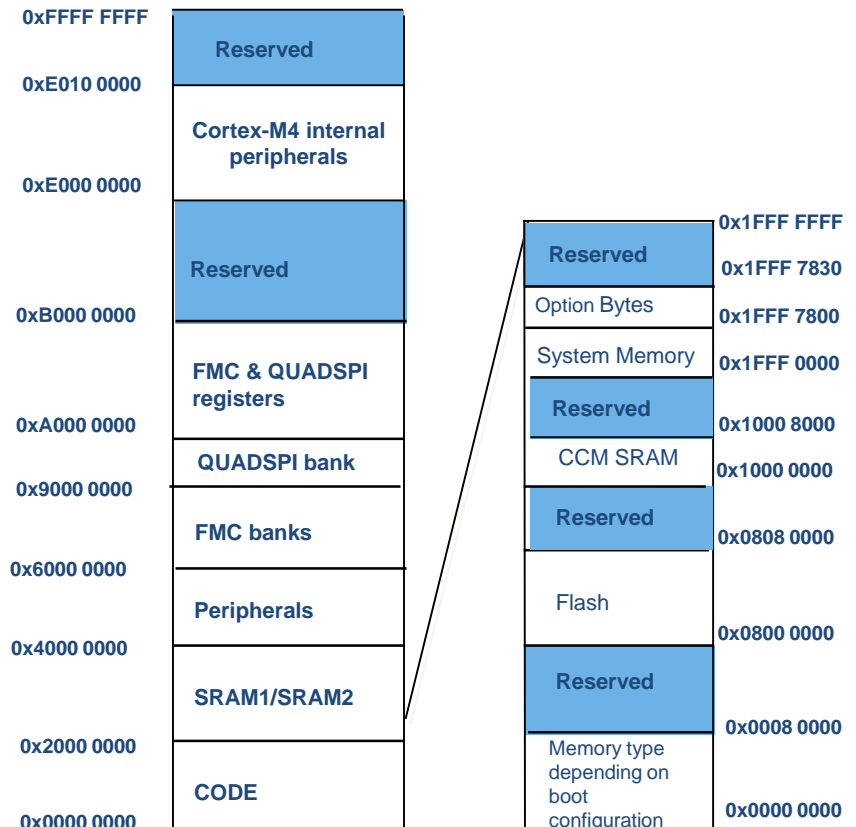
- Flash memory: up to 512Kbyte, dual bank

- FB_MODE = 0 in SYSCFG_MEMRMP:
 - Bank 1 @ 0x0800 0000 (alias 0x0000 0000)
 - Bank 2 @ 0x0804 0000
- FB_MODE = 1 in SYSCFG_MEMRMP
 - Bank 2 @ 0x0800 0000 (alias 0x0000 0000)
 - Bank 1 @ 0x0804 0000

- SRAM: 128 Kbytes split in 3 parts:

- SRAM1: 80 Kbytes @ 0x2000 0000
- SRAM2: 16Kbytes @ 0x2001 8000
- CCM SRAM: 32 Kbytes @ 0x1000 0000
 - Access through D-code and I-code

- FMC/QUADSPI mapping same as L4.

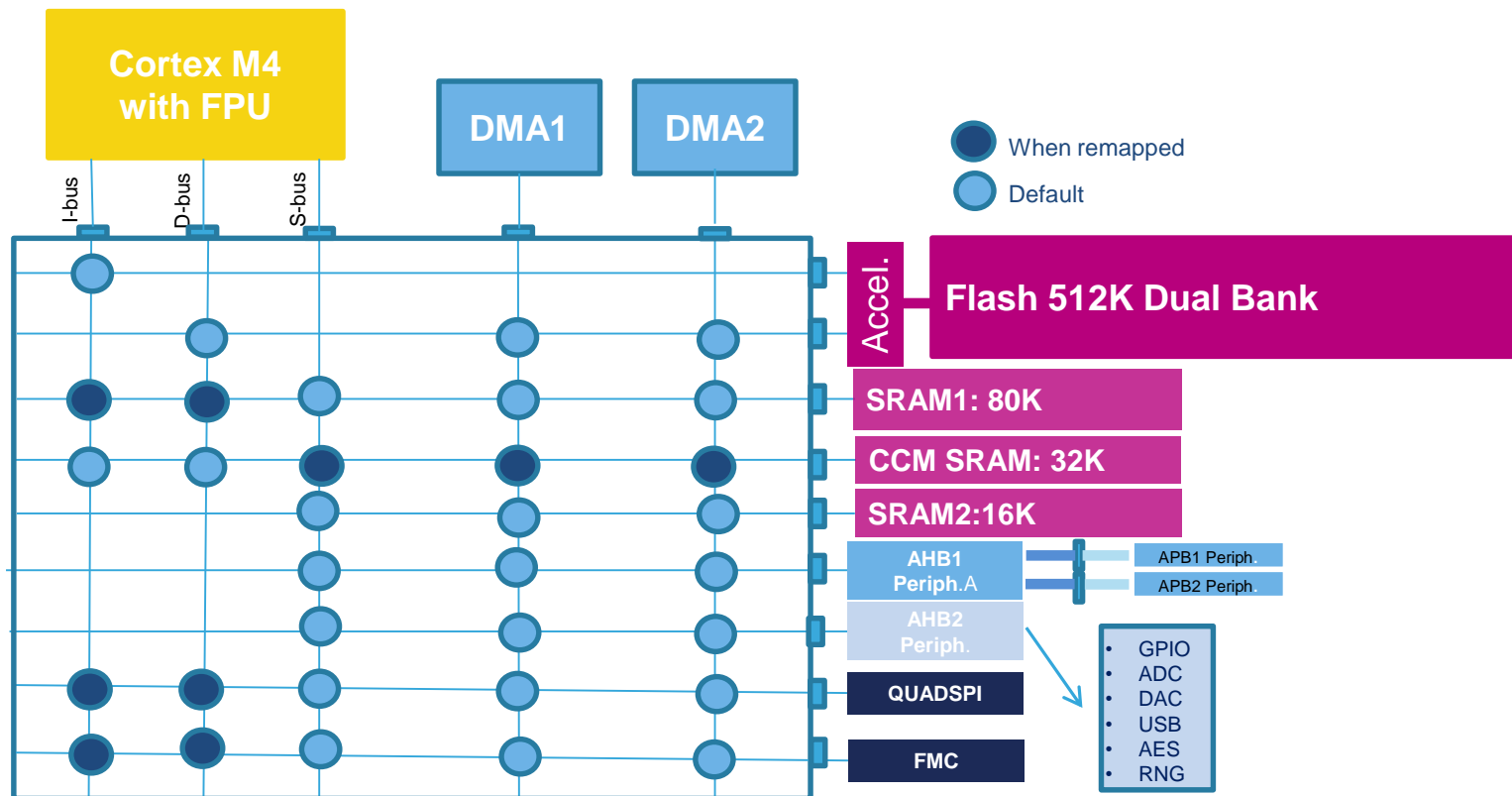


Memory remap

Performance booster!

- Address 0x0000 0000 remapping options
 - Main Flash memory
 - System Flash memory (Bootloader)
 - FMC bank 1 (NOR/PSRAM modes)
 - SRAM1
 - QUADSPI
 - Boosts performance thanks to I-Code/D-Code accesses instead of System Bus
- FB_MODE in SYSCFG_MEMRMP
 - Swap Flash memory banks 1 & 2

STM32G474x Bus matrix



Boot modes

37

Boot mode configuration					Selected boot area
BOOT_LOCK bit	nBOOT1 bit	BOOT0 pin	nSWBOOT0 bit	nBOOT0 bit	
0	x	0	0	x	Main Flash memory
0	1	1	0	x	System memory
0	0	1	0	x	Embedded SRAM
0	x	x	1	1	Main Flash memory
0	1	x	1	0	System memory
0	0	x	1	0	Embedded SRAM
1	x	x	x	x	Main Flash memory forced

- **BOOT_LOCK Forcing boot from Flash memory**
 - It is possible to force booting from Main Flash memory regardless the other option bits.

Bootloader

Protocol	I/Os and Comments
USART	USART1 on pins PA9/PA10 USART2 on pins PA2/PA3 USART3 on pins PC10/PC11
USB	USB DFU interface on pins PA11/PA12
SPI	SPI1 on pins PA4/PA5/PA6/PA7 SPI2 on pins PB12/PB13/PB14/PB15
I2C	I2C2 on pins PC4/PA8 I2C3 on pins PC8/PC9 I2C4 on PC6/PC7

CCM SRAM features

Performance, integrity and safety (Class B, SIL)

- 32 Kbytes of CCM SRAM with access through D-code and I-code:
 - **Code execution maximum performance without remap**
- **HW parity check:** 4 bits per word
 - Enabled with **SRAM_PE** in user options bytes
 - NMI generated on parity error
 - Optional Break to Timers, system fault input of HRTIM
 - Disabled by default

Note: The parity check is implemented also on the first 32Kbytes of SRAM1 in the STM32G474.

CCM SRAM features

Secured SRAM

- **Write protection** with 1-Kbyte granularity
 - **SYSCFG_SWPR** write protection register
- **Read/Write protection** with RDP
 - When protected, the CCM SRAM cannot be read or written by the JTAG or serial wire debug port, and when the boot in System flash or boot in SRAM is selected.
 - Erased when RDP changed from Level 1 to Level 0
- **Software reset** and optional **Hardware reset** when system reset
 - Erased when setting **CCMER** bit
 - Erased with system reset with **CCMSRAM_RST** in user option bytes

Performance

41

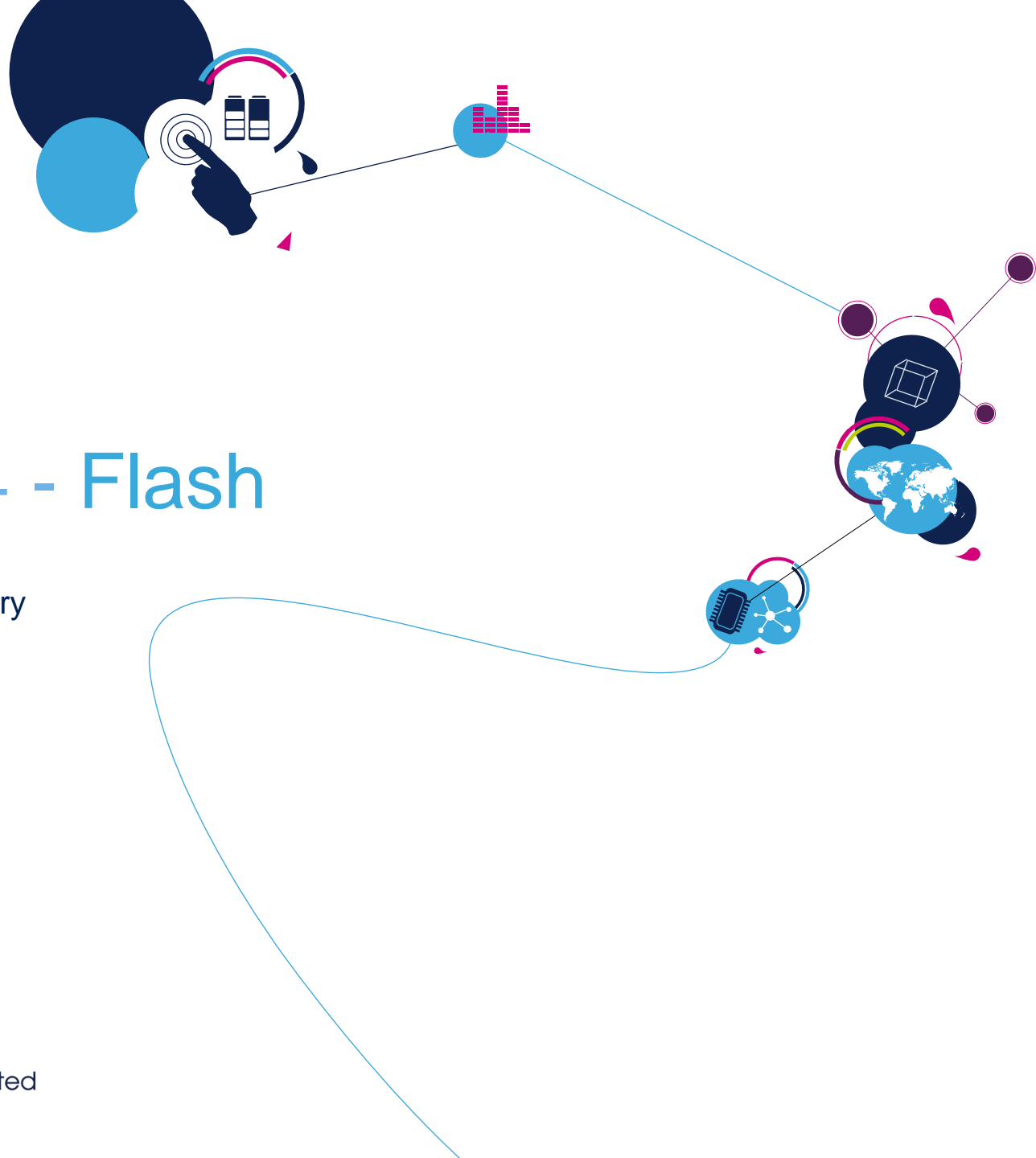
	Execution in Flash memory						Execution in SRAM	
	ART ON I-Cache ON D-Cache ON Prefetch ON		ART ON I-Cache ON D-Cache ON Prefetch OFF		ART OFF		Code & Data in SRAM1	Code in CCM SRAM, Data in SRAM1
CoreMark / MHz @ 150 MHz	Dual Bank	Single Bank	Dual Bank	Single Bank	Dual Bank	Single Bank	2.37	3.42
	3.26	3.36	3.23	3.32	1.05	1.47		

Differences with STM32G431 devices

- Multilayer AHB matrix
 - No QUADSPI slave
 - No FMC slave.
- SRAM size
 - SRAM1: 16KB, parity check on the whole SRAM1.
 - SRAM2: 6KB
 - CCM SRAM: 10KB, parity check on the whole CCM SRAM.
- Memory mapping:
 - CCM SRAM is aliased @0x2000 5800 to allow continuous address space with SRAM1/SRAM2.
- FLASH :
 - 128K, single bank.

STM32G4 - Flash

Embedded Flash memory



G474 Flash Key features

44

- Up to 512Kbyte of dual-bank Flash memory with read-while-write capability
 - Single bank mode DBANK=0.
 - Dual bank mode DBANK=1
- Flash memory read operations with two data width modes :
 - Single bank mode DBANK=0: read access of 128 bits
 - Dual bank mode DBANK=1: read access of 64 bits
- Page erase, bank erase and mass erase
- 2 programming modes :
 - Standard (for main memory and OTP)
 - Fast (main memory only). Programs 32 double-words without verifying the Flash location.

G474 Flash Key features

45

- ART accelerator™ (Instruction cache, Data cache and prefetch buffer) allowing linear performance in relation to frequency
- Protections:
 - Single Bank mode: 4 WPR areas, 2 PCROP areas, 1 memory securable area
 - Dual Bank mode: 2 WPR areas per bank, 1 PCROP area per bank, 1 memory securable area per bank
- Error Code Correction (ECC): 8 bits for 64-bit double-word
 - Dual Bank mode: 72 bits
 - Single Bank mode: 144 bits
 - Single error detection and correction /Double error detection without correction.

G474 Flash organization

- Main memory block:
 - Dual bank mode (DBANK = 1): the Flash is divided in 2 banks of 256 KB, and each bank is organized as follows:
 - The main memory block containing 128 pages of 2 Kbytes
 - Each page is composed of 8 rows of 256 bytes
 - Single bank mode (DBANK = 0) : the main memory block is organized as one single bank of 512 KB as follows:
 - The main memory block containing 128 pages of 4 Kbytes
 - Each page is composed of 8 rows of 512 bytes
- An Information block containing:
 - System memory which is reserved for use by ST and contains the **bootloader**.
 - 1-KByte (128 double-words) **OTP (one-time programmable)** area for user data (in Bank 1 only).
 - Option bytes for user configuration.

G474 Flash Memory Organization

47

DBANK Mode	Dual Bank				Single Bank			
Flash area	Flash memory addresses		Size (bytes)	Page#	Flash memory addresses		Size (bytes)	Page#
Main memory	Bank 1	0x0800 0000 – 0x0800 07FF	2 K	Page 0	Single Bank	0x0800 0000 – 0x0800 0FFF	4 K	Page 0
		0x0800 0800 – 0x0800 0FFF	2 K	Page 1		0x0800 1000 – 0x0800 1FFF	4 K	Page 1
		-	-	-		-	-	-
		0x0803 F000 – 0x0803 FFFF	2K	Page 127		-	-	-
	Bank 2	0x0804 000 – 0x0804 07FF	2 K	Page 0		-	-	-
		0x0804 0800 – 0x0804 0FFF	2K	Page 1		-	-	-
		-	-	-		-	-	-
		0x0807 F000 – 0x0807 FFFF	2 K	Page 127		0x0807 F000 – 0x0807 FFFF	4 K	Page 127

Flash dual bank

Read-while write and Dual-bank boot capability

- Option **DBANK** in user option bytes selects dual bank mode
- Dual bank Flash memory with **dual-bank boot** capability.
 - Option **BFB2** in user option bytes
 - BFB2 = 1, device boots either in Bank 2 or in Bank 1 depending on valid bank.
 - BFB2 = 0, device boots in Bank 1 only.
- Read-while-write
 - With its dual-bank capability, it is possible to read from one bank while programming/erasing the other bank => code execution is not stopped when the Flash memory is being programmed

Flash read access

210 DMIPS at 170 MHz

Adaptive real-time memory accelerator (ART Accelerator™) allowing linear performance versus frequency, regardless of Flash memory access time.

Wait states (WS) (FLASH latency)	HCLK (MHz)	
	V _{CORE} Range 1	V _{CORE} Range 2
0 WS	≤ 20	≤ 8
1 WS	≤ 40	≤ 16
2 WS	≤ 60	≤ 26
3 WS	≤ 80	-
4 WS	≤ 100	-
5 WS	≤ 120	-
6 WS	≤ 140	-
7 WS	≤ 160	-
8 WS	≤ 170	-

Flash memory protection

Flexible Flash protections according to application needs

- **Readout protection (RDP)**

Prohibits any access to Flash/CCM SRAM/Backup registers by debug interface (JTAG/SWD) or when booting from SRAM1 or when the Bootloader is selected.

- **Proprietary Code Protection (PCROP):**

- Used to protect specific code area from any read or write access. The code can only be executed.
- Single Bank: 2 PCROP areas
- Dual Bank: 1 PCROP area per bank

- **Write Protection (WRP) :**

- Used to protect a specific code area from unwanted write access
- Single Bank: 4 WPR areas
- Dual Bank: 2 WPR areas per bank

Flash memory protection

Flexible Flash protections according to application needs

- Securable memory area
 - Area of code which can be executed only once at boot and never again, unless a new reset occurs.
 - Single Bank: 1 memory securable area
 - Dual Bank: 1 memory securable area per bank
 - When activated, any access to securable memory area (fetch, read, programming, erase) is rejected, generating a bus error.
- Disabling core debug access
 - Temporal disable of debug access when running code in Securable memory area

Option bytes

52

Options	Description	Comment
BOR_LEV[2:0]	Brown-out reset threshold level	New in STM32G4
nRST_STOP; nRST_STDBY; nRST_SHDW	Reset/No reset generated when entering STOP/STANDBY/SHUTDOWN mode	Same as STM32F3.
WWDG_SW; IDWG_SW IWDG_STOP; IWDG_STDBY	HW/SW window watchdog / independent watchdog Independent watchdog counter is frozen / not frozen in STOP/STANDBY mode	Same as STM32F3.
BFB2	Dual-bank boot enable/disable	New in STM32G474.
DBANK	Selection between single bank mode with 128-bit data read width and dual bank mode with 64 bits data read width	New in STM32G474
nBOOT1	Boot configuration (together with BOOT0 pin)	Same as STM32F3.
nSWBOOT0	BOOT0 taken from the option bit nBOOT0 or taken from PB8/BOOT0 pin	New in STM32G4.
nBOOT0		New in STM32G4.
CCM SRAM_RST	CCM SRAM erase when system reset	New in STM32G4
SRAM_PE	SRAM1 and CCM SRAM parity check enable	Same as STM32F3.

Option bytes

53

Options	Description	Comment
RDP[7:0]	Readout protection level	Same as F3
PCROP1_STRT[14:0] PCROP1_END[14:0] PCROP2_STRT[14:0] PCROP2_END[14:0]	Bank 1 PCROP area start offset address Bank 1 PCROP area end offset address Bank 2 PCROP area start offset address Bank 2 PCROP area end offset address	New in STM32G4
PCROP_RDP	PCROP area preserved when RDP level decreased	New in STM32G4
WRP1A_STRT[6:0] WRP1A_END[6:0] WRP1B_STRT[6:0] WRP1B_END[6:0] WRP2A_STRT[6:0] WRP2A_END[6:0] WRP2B_STRT[6:0] WRP2B_END[6:0]	Bank 1 Write protection area A start offset address Bank 1 Write protection area A end offset address Bank 1 Write protection area B start offset address Bank 1 Write protection area B end offset address Bank 2 Write protection area A start offset address Bank 2 Write protection area A end offset address Bank 2 Write protection area B start offset address Bank 2 Write protection area B end offset address	In STM32F3, the write protection is implemented with a granularity of 2 pages, with one option bit for every two pages.

Option bytes

54

Options	Description	Comment
BOOT_LOCK	When set, it forces boot from main flash memory.	New in STM32G4.
SEC_SIZE1[7:0] SEC_SIZE2[7:0]	Bank 1 securable memory area size Bank 2 securable memory area size	New in STM32G4 No SEC_SIZE2 in STM32G431.
IRHEN	Internal reset holder enable bit	New in STM32G474
NRST_MODE	PG10/NRST function selection	New in STM32G474

Low-power modes

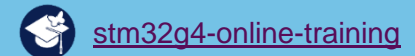
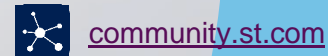
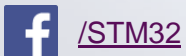
Consumption optimization when execution from SRAM

- Flash clock can be gated off in Run/Low-power run and/or in Sleep/Low-power sleep modes
 - Flash clock is configured in the Reset and Clock Controller (RCC)
 - Flash clock is enabled by default
- Flash can be configured in Power-down mode during Sleep/Low-power sleep modes
- Flash can be configured in Power-down mode during Run/Low-power run modes

Differences with STM32G431 devices

- Flash 128Kbytes, single bank.
- Flash organization:
 - 64 pages, each page is 2 Kbytes.
- Protections:
 - 2 WRP areas
 - 1 PCROP area
 - 1 securable memory area

Releasing Your Creativity



 www.st.com/STM32G4