



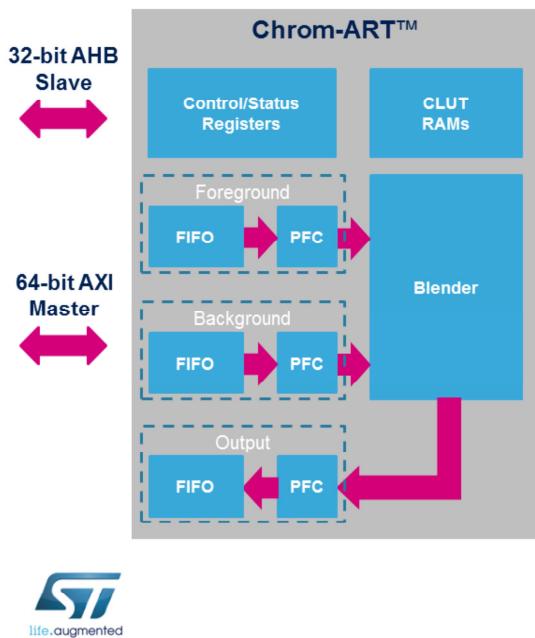
STM32H7 – Chrom-ART™

Chrom-ART Graphics Accelerator™

Revision 1.0



Hello, and welcome to this presentation of the STM32 Chrom-ART Accelerator™. It covers the features of this of this adaptive real-time accelerator block, which is widely used for graphic computing in the microcontroller.



- Provides hardware acceleration for graphical operations
 - Graphics-oriented 2D DMA
 - Planes blending & pixel format conversion
 - Specific modes for anti-aliased fonts

Application benefits

- Offloads CPU for graphical operations
- One pixel per cycle calculation
- Integrated pixel format converter & blender
- Simple integration through graphical stack

The Chrom-ART™ accelerator offers true hardware acceleration for graphical operations.

The Chrom-ART™ accelerator is built around a 2D DMA engine for fast data copy with specific functions to support pixel format conversion as well as blending operations between two planes. It also provides specific modes for managing anti-aliased fonts.

The Chrom-ART™ accelerator will offload the CPU for most of the graphical operations with a one pixel per cycle throughput, integrated pixel format conversion and blending.

The Chrom-ART™ accelerator is fully integrated in graphical stacks making its software integration transparent to the user.

- 2D DMA with graphical-oriented features with 4 operating modes
 - Register-to-memory
 - Memory-to-memory
 - Memory-to-memory with pixel format conversion
 - Memory-to-memory with pixel format conversion and blending
- User programmable parameters
 - Sources and destination addresses on the whole memory
 - Sources and destination size and offset
 - Source and destination color format
 - Up to 12 color formats supported from 4 up to 32 bits per pixel with indirect or direct color coding including YCbCr format
 - Internal memories for CLUT storage in indirect color mode with automatic loading
 - Alpha value can be modified (source value, fixed value or modulated value)



The Chrom-ART™ accelerator has four operating modes.

- Register-to-memory for rectangle filling operations,
- Memory-to-memory for 2D memory copy operations,
- Memory-to-memory with pixel format conversion for bitmap drawing with format conversion,
- Memory-to-memory with pixel format conversion and blending for bitmap or text drawing with transparency.

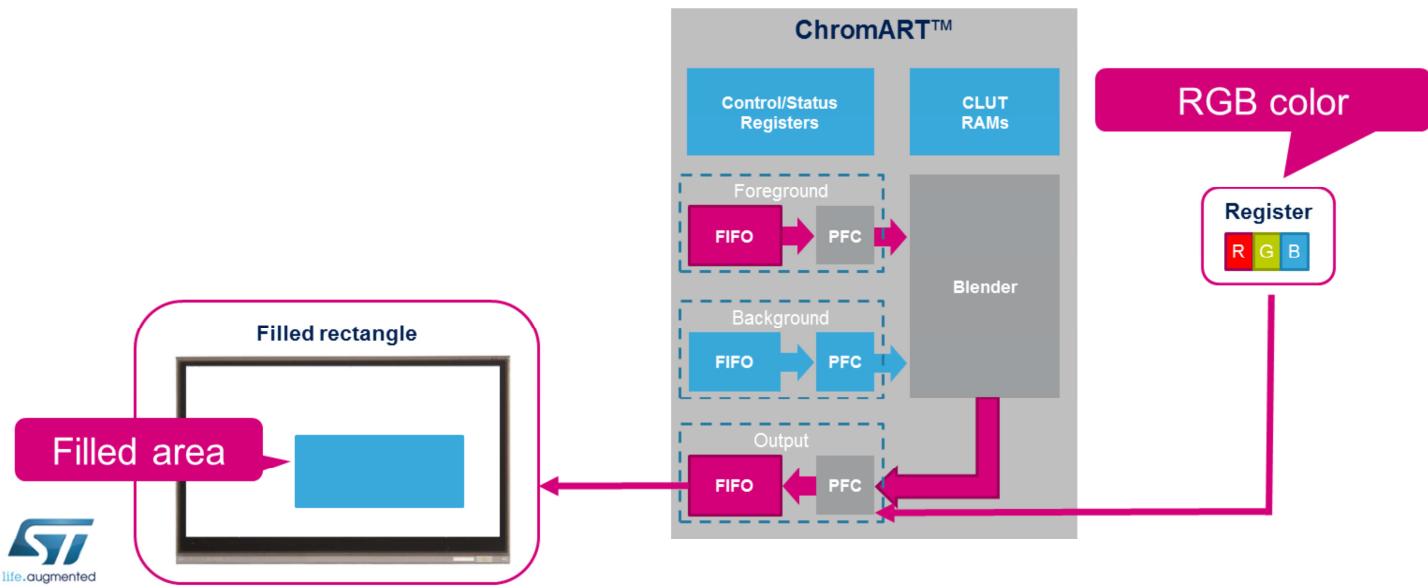
The user can program independently all the parameters for the source and the destination:

- The address of the layer including its size and position
- The color format
- The way transparency is managed.

Register to memory

4

Filling a part or the whole of a destination image with a specific color



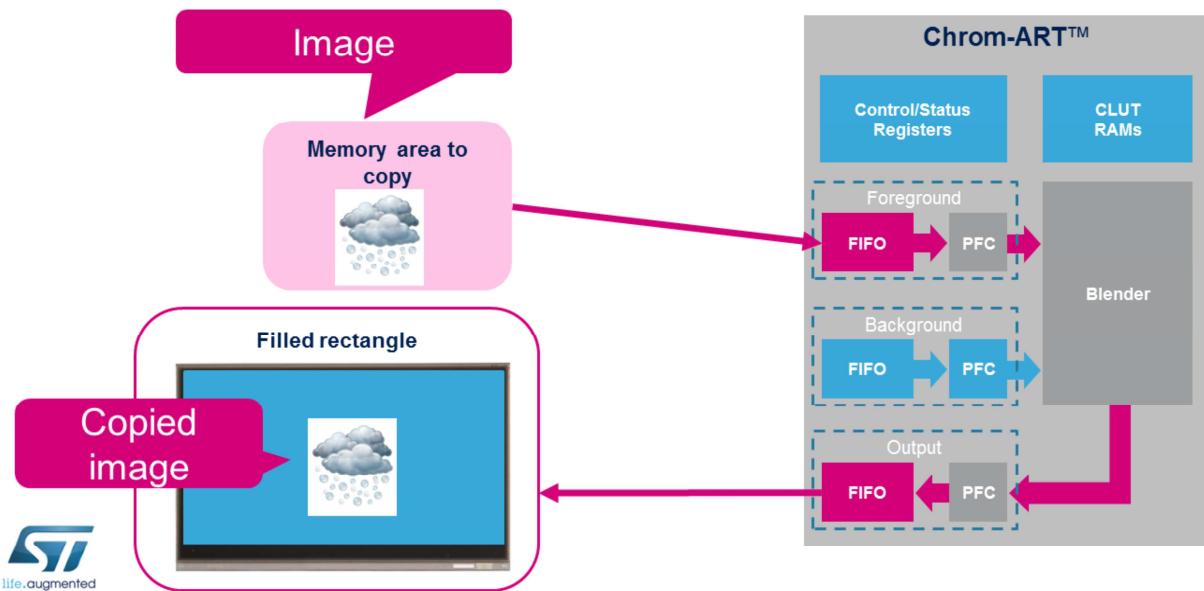
Register-to-memory mode is used to fill a part or whole destination image with a specific color.

The color value is set in a register of the output PFC.

Memory-to-memory

5

Copying a part or whole source image into a part or whole destination image

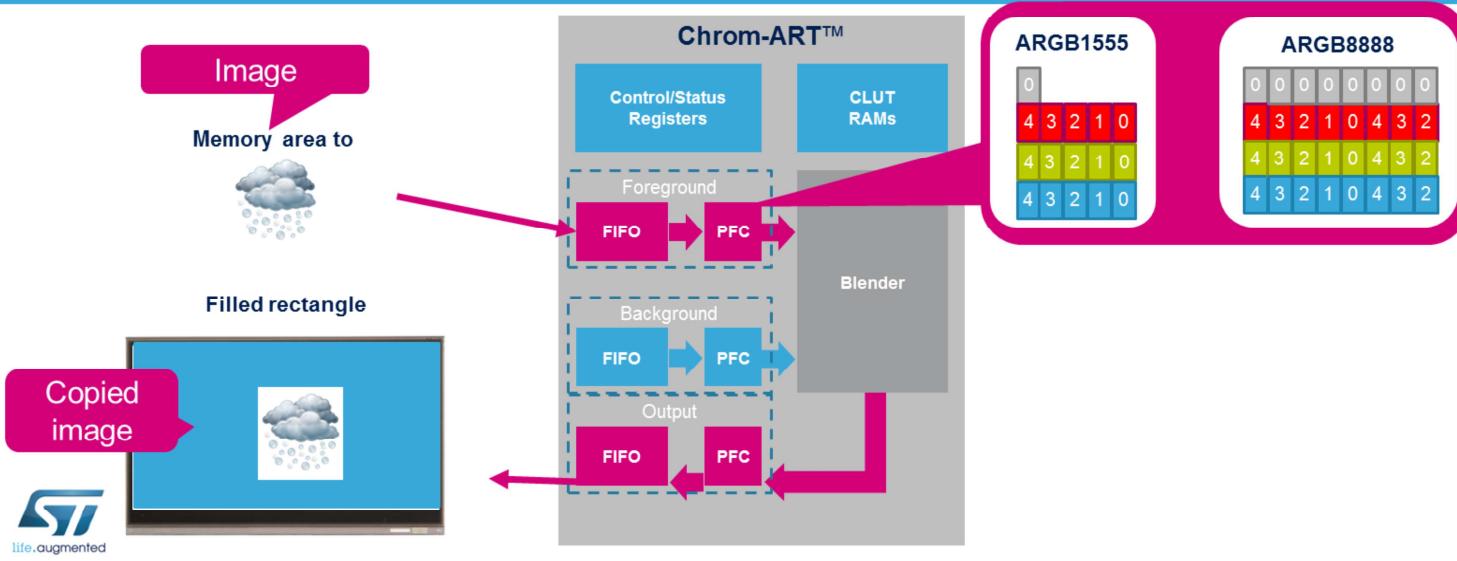


Memory-to-memory mode is used to copy a part or whole source image into a part or whole destination image without changing the color format.

Memory-to-memory with pixel format conversion

6

Copying a part or whole source image into a part or whole destination image with a pixel format conversion

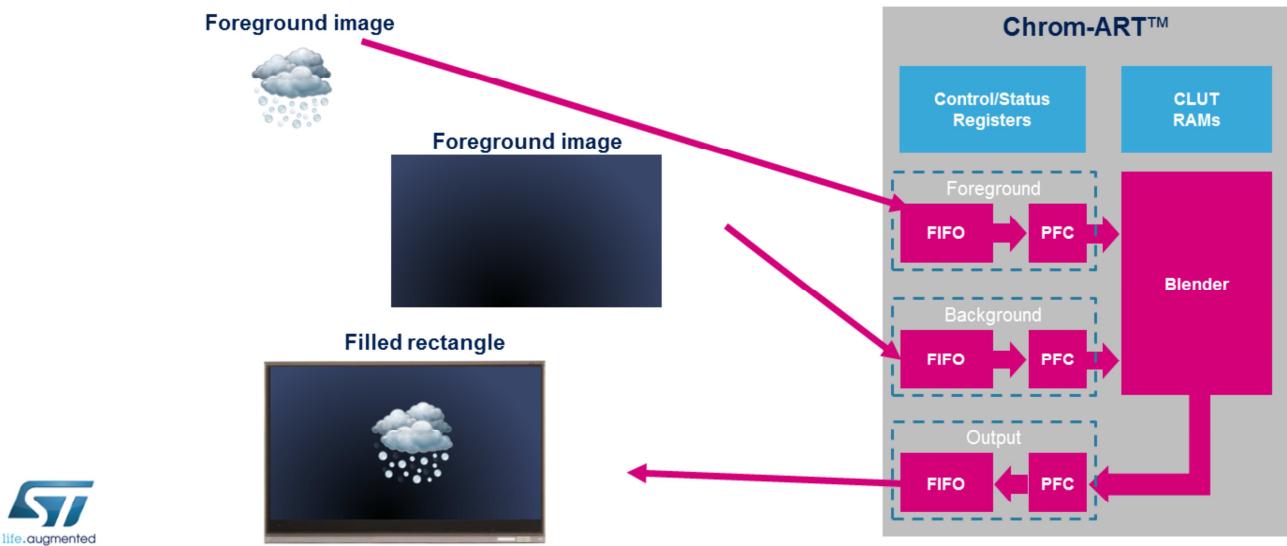


Memory-to-memory mode with pixel format conversion is used to do the same type of copy as Memory-to memory mode but with a pixel format conversion. It can copy any input format to any output format. For example, an ARGB565 image into an ARGB888 image without having to use the CPU.

Memory-to-memory with pixel format conversion and blending

7

Copying a part or whole source image into a part or whole destination image with a pixel format conversion and blending



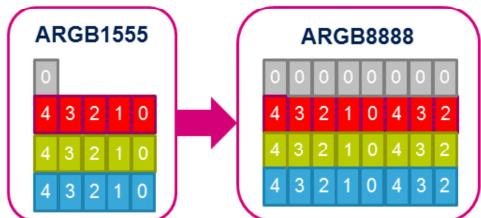
Memory-to-memory mode with pixel format conversion and blending is used to blend a part or whole source image with a part or whole destination image with a different pixel format. This is widely used to draw bitmap icons having transparency or fonts.



Input pixel format conversion

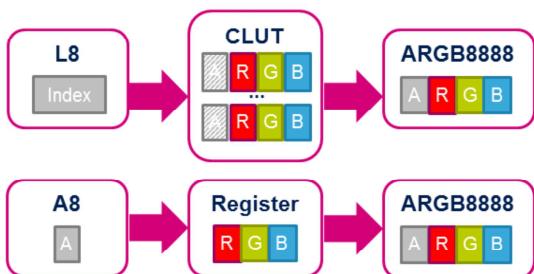
8

Independently supports different input color depths



- Direct mode

- ARGB8888, ARGB444444 or ARGB1555
- RGB888 or RGB565 (with alpha in register)
- Alpha can be replaced or modulated
- Block based (8x8) YCbCr 4:4:4, 4:2:2 and 4:2:0



- Indirect mode

- L8 or L4 and a Color Look-up Table (**CLUT**)
- A8 or A4 and a color register (for fonts)
- Mixed AL88 or AL44

- Internal operations are done in ARGB8888



For each foreground and background layer, the format can be programmed independently.

Direct mode fetches the RGB or ARGB content directly from the memory.

Indirect mode uses an intermediate color look-up table to determine the color to be used during the copy or blending operation.

All the input color modes are transformed internally into ARGB8888 format to perform the blending operation.

YCbCr color space converter

9

Color space converter for JPEG decoding

- YCbCr mode
 - Available on the foreground plane
 - The Chrom-ART™ accelerator fetches 8x8-pixel blocks
 - Supports standard chroma sub-sampling
 - 4:4:4
 - 4:2:2
 - 4:2:0
 - Complete CPU offload for color space conversion to display picture or video
 - The Chrom-ART™ color space converter is 10 times faster than the CPU at equivalent speed



The YCbCr mode enables a specific color space converter hardware to convert YCbCr data into RGB data. This block is available on the foreground plane.

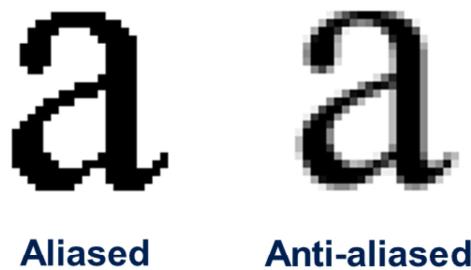
This color space converter is used to copy or blend uncompressed data from the JPEG decoder into the framebuffer.

It allows to convert directly YCbCr data arranged in 8x8 pixel blocks into a linear RGB format. All the standard YCbCr chroma sub-sampling formats are supported from 4:4:4 down to 4:2:0.

This new feature offloads the CPU from this costly operation and makes it 10 times faster.

Efficient support for anti aliased bitmap fonts

- Using A8 or A4 coding
 - Only the Alpha channel is stored in the memory
 - The Chrom-ART™ accelerator adds a programmed color



Specific modes can be used to efficiently manage texts and fonts.

Only the transparency value is stored in memory for rendering anti-aliased fonts.

The color is added during the pixel format conversion process and can be programmed by the user.

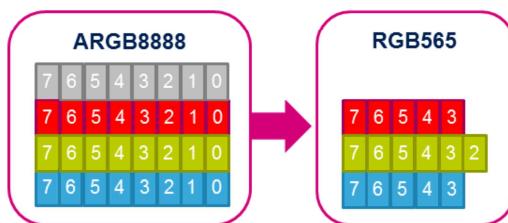
These modes are very efficient for storing high-quality bitmap fonts.

Output pixel format conversion

11

Support several output color depth

- Direct mode
 - ARGB8888, ARGB44444 or ARGB1555
 - RGB888 or RGB565
- Memory-to-memory without Pixel Format Conversion (PFC) is agnostic of the pixel format

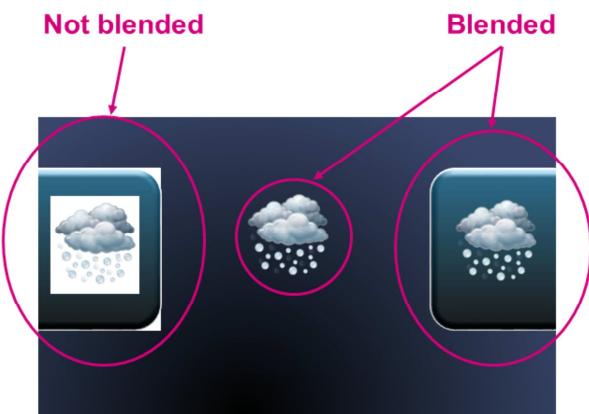


The output pixel format converter generates the color for the destination independently from the source.

There is no indirect mode in output as this would imply to calculate a color look-up table (CLUT).

Nevertheless, memory-to-memory operations without Pixel Format Conversion (PFC) can copy data independently of their formats.

Fully hardware blending process



• Hardware Blending

- Blend Foreground & Background pixels
- 1 pixel generated per cycle
- Native ARGB8888 operation
- Input data are converted into ARGB8888 by their respective PFC
- Output data has also its own PFC



The fully hardware blender allows to blend a foreground image and a background image with transparency.

This can be used to draw bitmap images of any shape with a perfect rendering.

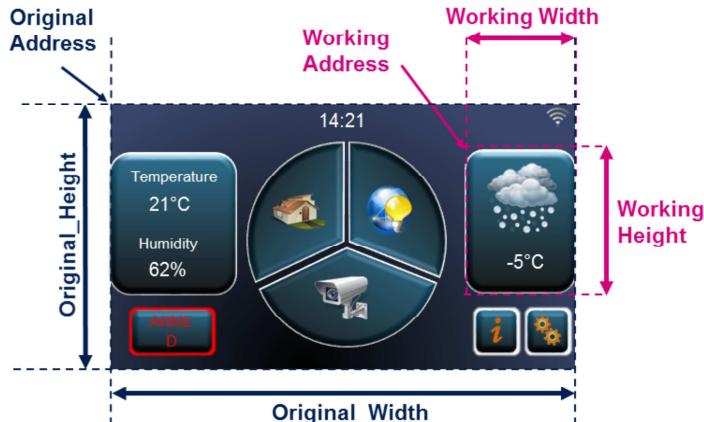
1 pixel is generated per cycle making this complex operation much more efficient than if it was done by the CPU.

The resulting pixel can be coded independently from the source thanks to the output pixel format converter.

Output configuration

13

Output configuration defines the area of work & output color coding



- Area of work
 - Width
 - Height
- Output configuration
 - Address
 - LineOffset
 - Color Format (BPP)

$$\text{Working_Address} = \text{Original_Address} + (X + \text{Original_Width} * Y) * \text{BPP}$$

$$\text{LineOffset} = \text{Original_Width} - \text{Working_Width}$$



The output configuration defines the working area for the Chrom-ART™ operation.

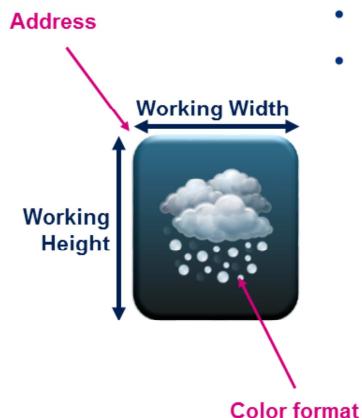
The address and the line offset parameters are used to select which sub-area of the output is concerned.

FG & BG configuration

14

FG & BG configuration define address and color coding

- Foreground parameters
 - Address
 - LineOffset (as per output)
 - Color format



- Background parameters
 - Address (as per output)
 - LineOffset (as per output)
 - Color format

Most of the time, background and output have the same parameters



The background and foreground layers have their own configuration for address, line offset and color format. This defines which area of the foreground and background layers are targeted by the Chrom-ART™ operations.

Interrupt event	Description
Configuration error	Error of configuration detected when starting the Chrom-ART™ (not allowed or wrong configuration)
CLUT transfer complete	CLUT copy from a system memory area to the internal Chrom-ART™ memory is complete
CLUT access error	CPU accesses the CLUT while the CLUT is being automatically copied from a system memory to the internal Chrom-ART™ memory
Transfer watermark	Indicates the last pixel of the watermarked line has been transferred
Transfer complete	Chrom-ART™ transfer operation is complete
Transfer error	An error occurred during Chrom-ART™ transfer



The Chrom-ART™ accelerator has 6 interrupt sources to signal:

- Configuration errors
- CLUT transfer complete
- CLUT access error
- Watermark reached during a transfer
- Transfer complete
- Transfer error

No DMA trigger is used as the Chrom-ART™ accelerator embeds its own DMA.

Mode	Description
Run	Active.
Sleep	Active. Peripheral interrupts cause the device to exit Sleep mode.
Stop	Frozen. Peripheral registers content is kept.
Standby	Powered-down. The peripheral must be reinitialized after exiting Standby mode.

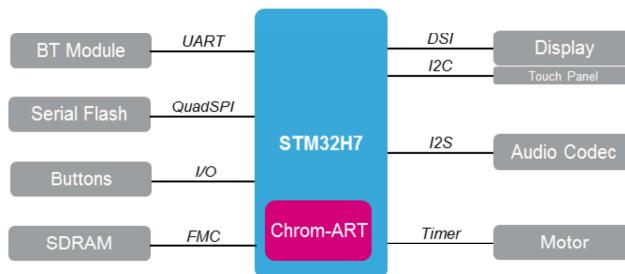


The Chrom-ART™ accelerator is active in Run and Sleep modes. A Chrom-ART™ interrupt can cause the device to exit Sleep mode. In Stop mode, the Chrom-ART™ accelerator is frozen and its registers content is kept. In Standby mode, the Chrom-ART™ accelerator is powered-down and it must be reinitialized afterwards.

Application examples

17

- Home alliances including connectivity and HMI:



- Chrom-ART™ can handle HMI building
 - Composition of the scene with transparency and animations
 - Text management



The Chrom-ART™ accelerator is widely used in any graphical application to compute the frame buffer without any CPU load and with a very efficient throughput. It can compose the whole scene with transparency and facilitate the management of animations. Text rendering is also accelerated, making it easy and efficient to manage anti-aliased fonts.

Related peripherals

18

- Refer to these trainings related to this peripheral:
 - RCC (Chrom-ART™ clock control and Chrom-ART™ enable/reset)
 - Interrupts (Chrom-ART™ interrupt mapping)



You can refer to the trainings related to the RCC and interrupts for additional information.

Comparison across STM32F/H series

19

Chrom-ART™ features	STM32F0	STM32F1	STM32F2	STM32F3	STM32F4	STM32F7	STM32H7
Number of instances	0	0	0	0	1	1	1
Maximum speed	N/A	N/A	N/A	N/A	180MHz	216MHz	200MHz(*)

(*) 64-bit wide data bus



The Chrom-ART™ accelerator is implemented in the STM32F4 and STM32F7 series up to 216MHz with a 32-bit wide bus and up to 200MHz on the H7 series with a 64-bit wide bus.