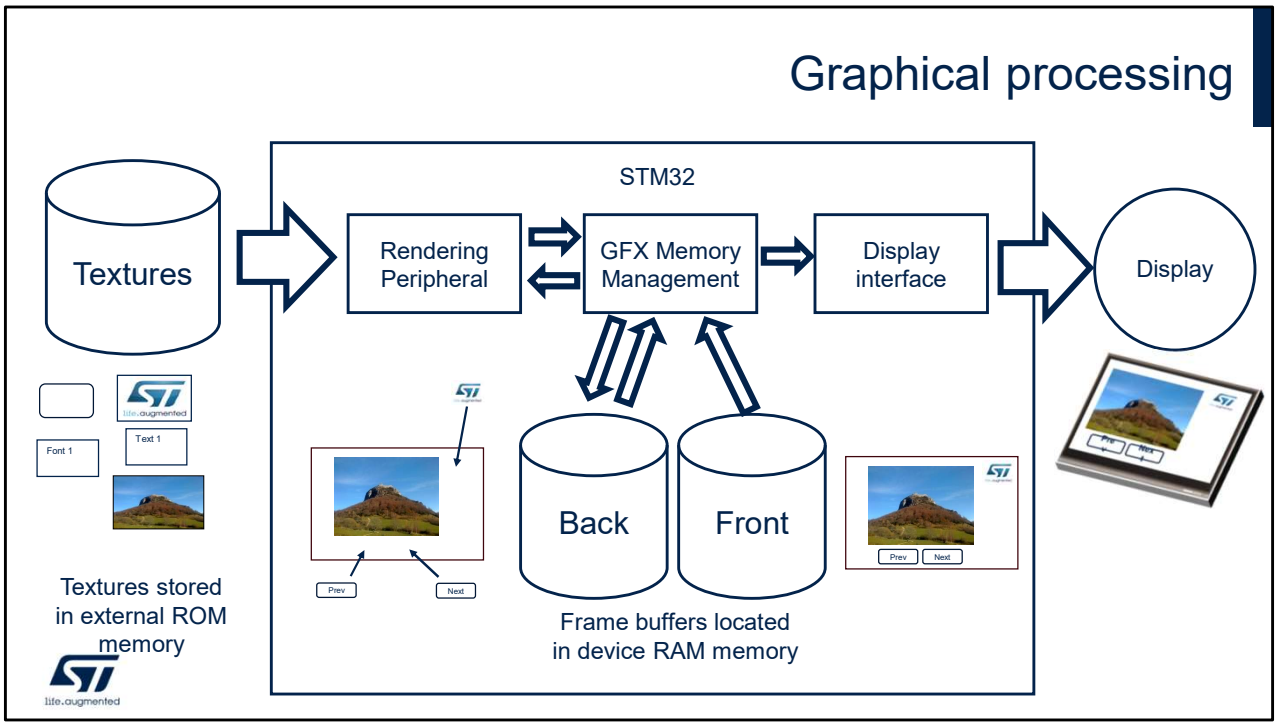
Hello, and welcome to this presentation, which introduces the graphic architecture present in the STM32U5.

The different graphical peripherals of an STM32

- Rendering bloc:
  - **GPU2D** *Neo-Chrom GPU*: Advanced image construction instructions
  - **DMA2D** *Chrom-ART*: Blending Copying filling part of images or reformatting RGB images
  - **CPU**: Can process images with adequate libraries
  - **JPEGCODEC**: Compress or Decompress 2D assets
- Memory optimization:
  - **GFXMMU** *Chrom GRC*: Optimize memory usage according to the display shape
  - **DCACHE2** : Cache GPU2D needed textures to external memories
- Display:
  - **LTDC**: LCD-TFT Display Controller, offers interface for RGB and H and V signals synchro.
  - **DSI**: Mipi compliant Display Serial Interface

The STM32U5 Series can serve graphical applications thanks to a variety of peripherals with different roles or features. Let's introduce them briefly here:
Rendering blocks have the mission to prepare the frame to be displayed on screen. They can rotate an image for example, or zoom in and out thanks to the GPU2D, superpose images or fill areas as DMA2D or change format as JPEGCODEC can do. The CPU then offers custom solutions but at the cost of a longer processing time.
Memory Optimization blocks have the role of limiting the memory accesses. It can be done by optimizing the address range thanks to the memory management unit, or by caching frequently accessed memory area, thanks to DCACHE2.
Finally, the displaying peripherals will handle the transmission of

the created buffer, from internal memory to the user display, thanks to LCD-TFT Display Controller or more advanced DSI interface.
For more details on each peripheral or their availability within a given STM32U5 product, please refer to the dedicated OnLine Training block or to the product Datasheet.



The blocks presented in the previous slide will handle the graphical processing parts. It is important however to predefine your memory topology regarding both the screen characteristics and type of processing that the application will perform. Indeed, the bigger the resolution of your display, the longer the processing time and the more input material will be needed.

To be more concrete, we will first define the textures memory:

These graphical assets are voluminous and consequently must be stored in external memory. The highest data rate external memory in STM32U5 Series is the HSPI, offering high bandwidth for rendering peripheral to prepare frames.

The Frames are the images that will be visible at the end of the process on the screen, they are stored in frame buffers that are

located usually in internal SRAM to achieve highest throughput. We have a minimum of 2 frame buffers within a graphical processing chain.

The back frame Buffer: it is built using the different texture assets and rendering peripherals. The amount of frames that MCU can build up is usually expressed in fps.

The front frame Buffer: it is also a back frame but with a completed rendering, ready for display. The amount of front frame sent to the display is expressed in Hz.

Periodically, the operating system changes a back frame ready in front frame and restarts building the next frame on a new back frame buffer.

Displaying the front frame buffer

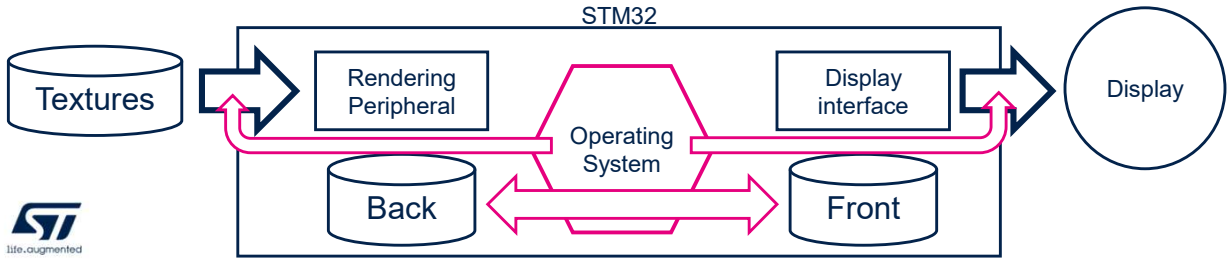
- At display side 3 main parameters impact data throughput:
  - Resolution : Amount of pixels composing your display
  - Colordepth : Width of bitfields coding a RGB pixel
  - Refresh rate : Amount of frame sent to your display (usually 60 Hz)
- The embedded **LTDC**: LCD-TFT Display Controller can handle Source and Gate drivers of your display and allow user to populate less or cheaper external components.
- The **DSI** on top of it, offer a higher datarate protocol allowing to reduce parallelism of the lines on PCB.
- The **FSMC** can handle certain type of display and might be an alternative.

life.augmented

Displaying the front frame buffer

Highly sequential environnement

- These environment is highly time constrained:
  - In first steps to obtain the graphical assets and achieve the desired rendering computation steps. The DCACHE2 can help optimise GPU2D performance in this context.
  - In a second step to adequatly handover the back frame buffer ready becoming next front frame buffer thanks to semaphores.
  - Last to adequately load the selected display interface to achieve the desired refresh rate and keep a good quality display.



Sequential environment

Different hardware implementation possible

- The force of this ecosystem reside in its versatility, allowing to treat with same tools and library application facing the constraints of wearable markets (high integration, high quality) or industrial markets (low cost, easy manufacturing)

	Wearable application	Home appliance
GFX Assets location	NAND Flash copied to a fast PSRAM at boot time	NOR Flash
GFX Assets com.	OSPI for NAND and HSPI for PSRAM	8 bits mode on HSPI interface
Display interface	MIPI DSI	LTDC or FSMC
FPS / Refreshment rates		
Packages	WLCSP208 for size constraint	LQFP144 for ease of manufacturing
PCB techno	Via-in-pad, narrow tracks, etc…	More relaxed constraint, cheap PCB

life.augmented

Find the different hardware implementations

GFX ecosystem

- TouchGFX GUI Tool composed of:
  - TouchGFX Designer, a PC GUI-builder and simulator
  - TouchGFX Generator, to configure and generate a TouchGFX project
  - TouchGFX Engine, which is an optimized, hardware-accelerated graphical library
- StemWIN library
  - STemWin is a simple graphical software framework optimized for STM32 microcontrollers.

Here you have an overview of the graphics ecosystem of the STM32U5

Thank you

life.augmented