TouchGFX Presentation

AL Development – Introduction

Agenda

1 Introduction

2 Abstraction Layer Architecture

3 TouchGFX Generator

life.augmented

Introduction

Introduction

After this presentation you'll have a general overview- and basic understanding of the Abstraction Layer (AL) between TouchGFX and the hardware it runs on.

We'll be going through the following topics:

- Introduction
- Abstraction Layer Architecture
- TouchGFX Generator

Introduction

Further reading

- You will find a lot of help on the TouchGFX documentation site: https://support.touchgfx.com/
- Slides in this workshop will refer to relevant documentation pages. Links will be in the lower right corner of the slides
- A good place to start reading following this workshop is:

Introduction

TouchGFX Development

- Main Components:

- Main Activities:

- In this presentation we will focus on the component “TouchGFX AL” and thus the activity “TouchGFX AL Development”

Introduction

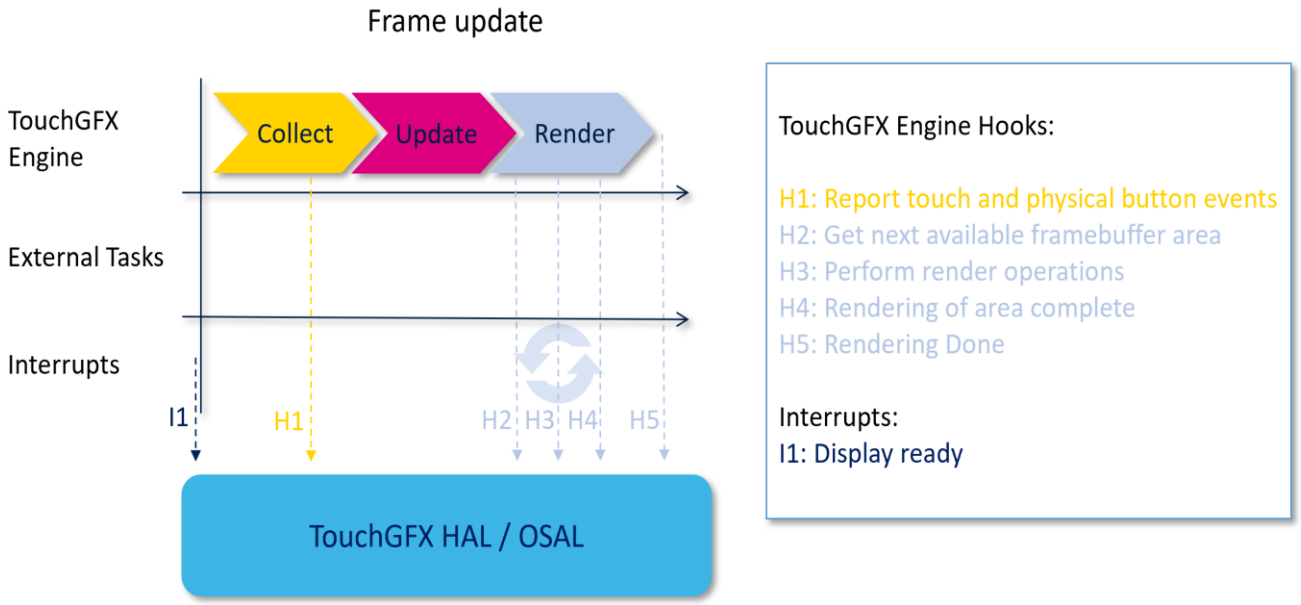
Introduction

A TouchGFX Abstraction Layer (AL) consists of:

- Hardware Abstraction Layer (HAL)
- Operating System Abstraction Layer (OSAL)

Responsibilities of the AL:

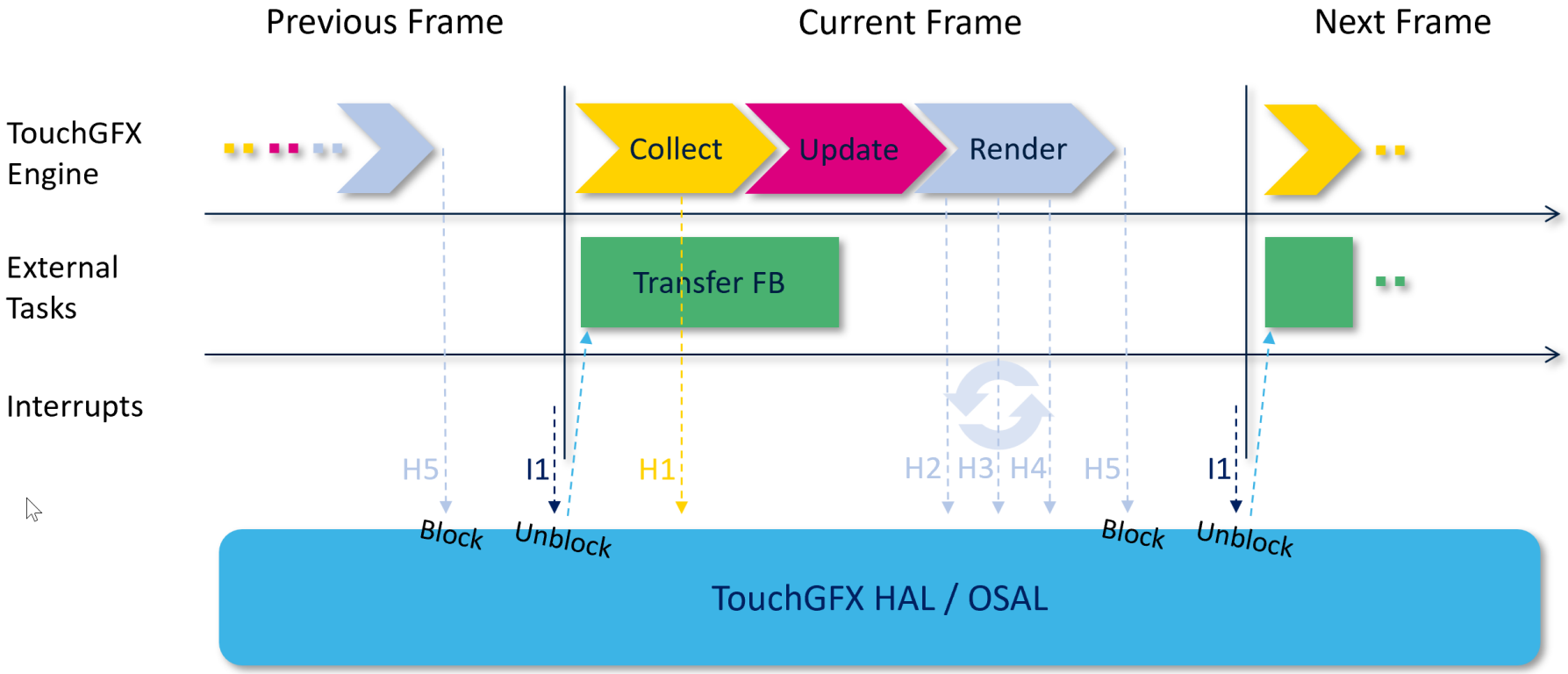
- Synchronize TouchGFX Engine main loop with display transfer
- Report touch and physical button events
- Synchronize framebuffer access
- Report the next available framebuffer area
- Perform render operations
- Handle framebuffer transfer to display



Documentation Link: AL Development Introduction

Introduction

It is up to the AL developer to fulfil these responsibilities in the AL (HAL + OSAL). Below is an example for Parallel RGB (LTDC) display interface:



TouchGFX Engine Hooks:

H1: Report touch and physical button events
H2: Get next available framebuffer area
H3: Perform render operations
H4: Rendering of area complete
H5: Rendering Done

Interrupts:
I1: Display ready

Documentation Link: AL Development Introduction

Abstraction Layer Architecture

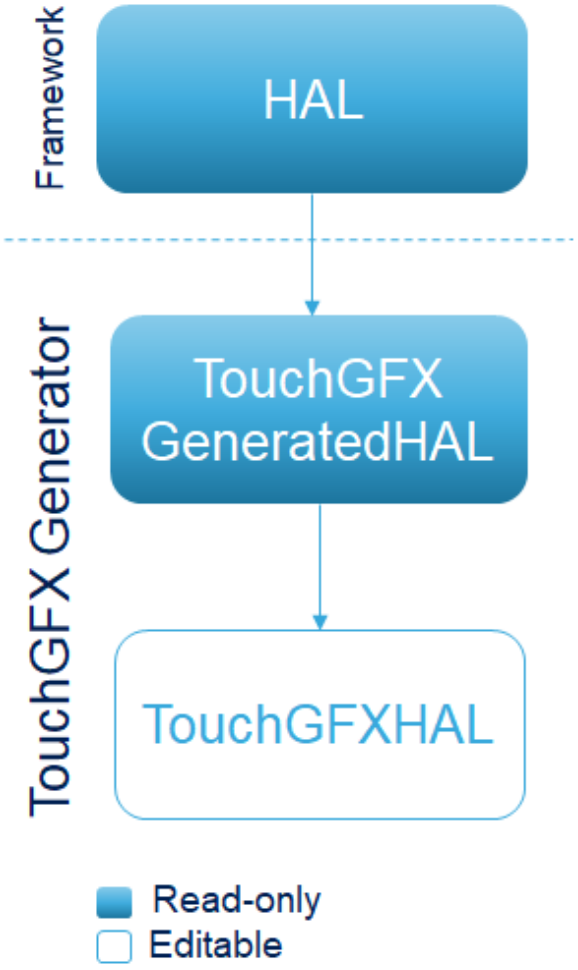
Abstraction Layer Architecture

Responsibility	Area	Description
Synchronize TouchGFX Engine Main Loop with display transfer	HAL/OSAL (required)	TouchGFX Engine Main Loop blocks after rendering. Display signals TouchGFX Engine when ready to process new frames.
Report Touch and Physical Events	HAL (optional)	If available, touch- and physical events (e.g. buttons) can be reported to TouchGFX Engine.
Synchronize Framebuffer access	OSAL (required)	Ensures that only one actor accesses a framebuffer.
Report next available Framebuffer area	HAL (required)	Report where TouchGFX Engine should render next. Depends on the framebuffer strategy.
Perform Render Operations	HAL (optional)	Implementation of additional rendering capabilities, e.g. DMA2D.
Handle Framebuffer transfer to display	HAL (required)	Depends on display interface type. For e.g. SPI/FMC developers initiate the transfer manually when asked to.

Abstraction Layer Architecture

The TouchGFX AL is generated by TouchGFX Generator in CubeMX.

- HAL responsibilities are implemented in sub-classes of HAL based on CubeMX MCU configuration
- The OSAL is automatically generated based on CubeMX Middleware configuration: CMSIS RTOS V1 or V2.. For other RTOS' developers are allowed to implement the OSAL themselves



Documentation Link: AL Architecture

TouchGFX Generator

TouchGFX Generator

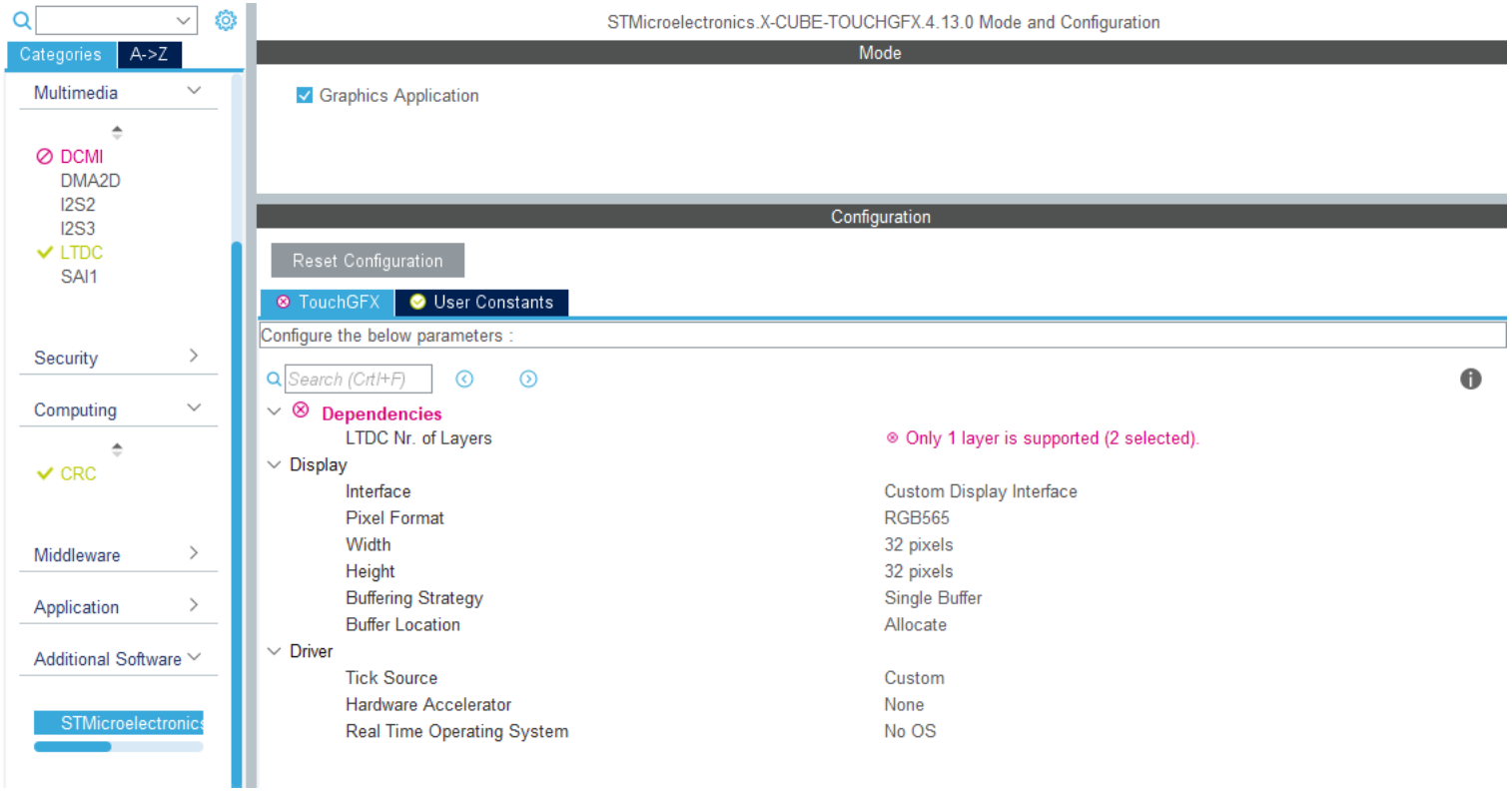
The TouchGFX Abstraction Layer is generated by TouchGFX Generator.

- TouchGFX Generator is part of X-CUBE-TouchGFX
- When enabled, TouchGFX Generator creates a TouchGFX Abstraction Layer according to TouchGFX Generator and CubeMX configuration

Documentation Link: Generator

TouchGFX Generator

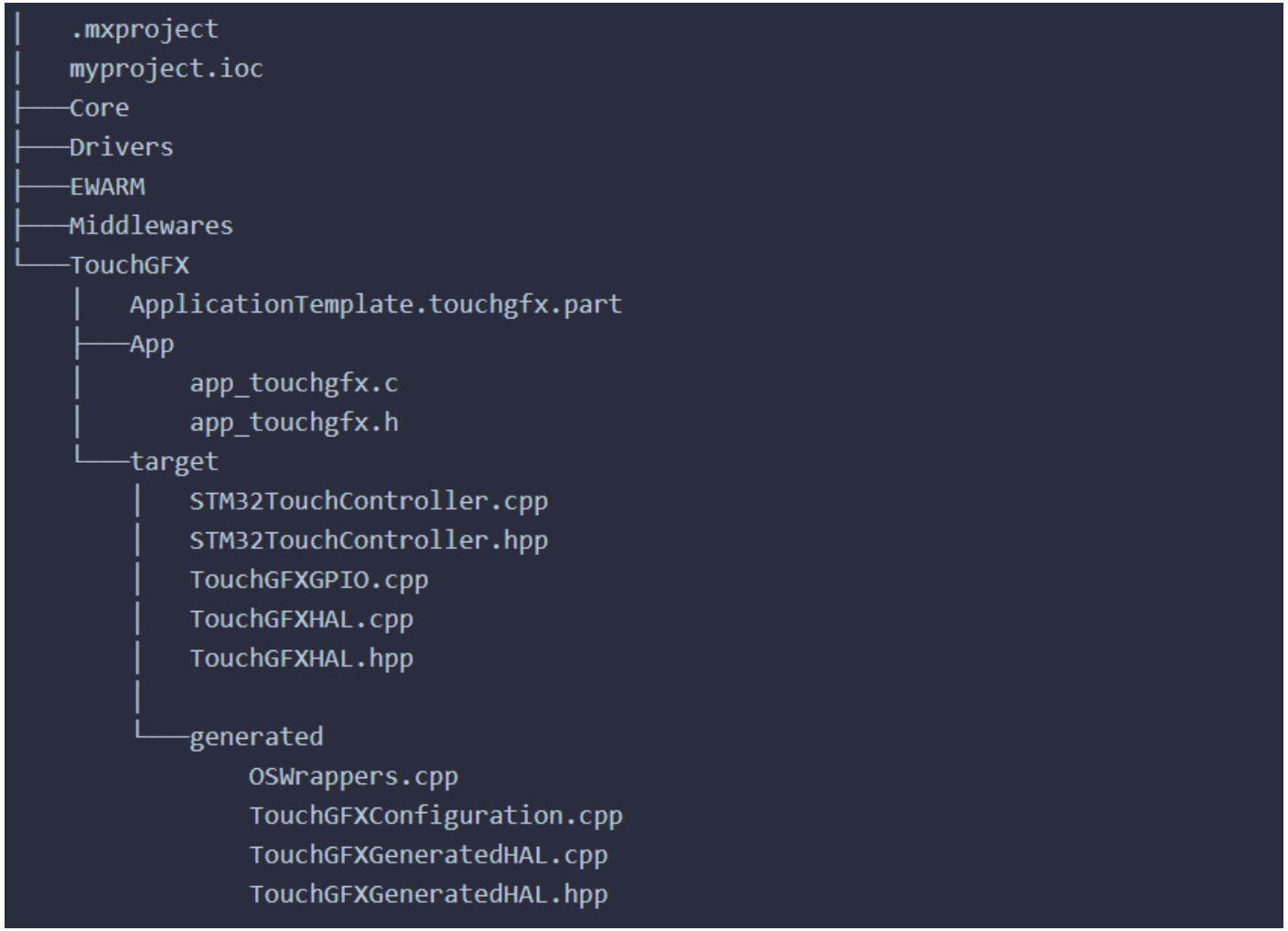
- The *Dependencies* group list warnings or information based on CubeMX configuration.
- The *Display* group has settings related to interface, format, dimensions and strategy.
- The *Driver* group has settings related to driving and accelerating TouchGFX as well as selecting the OSAL.



Documentation Link: Generator

TouchGFX Generator

The generated AL is created inside the *TouchGFX* folder when generating code from CubeMX.



Documentation Link: Generator

TouchGFX Generator

- After project generation TouchGFX GUI is developed in TouchGFX Designer
- Updates to TouchGFX Generator configuration in CubeMX are reflected in TouchGFX Designer (pixel format, screen dimensions)
- TouchGFX Designer ensures that target project is updated with new files
- Target binary is programmed using STM32CubeProgrammer

Documentation Link: Generator

Thank you

life.augmented