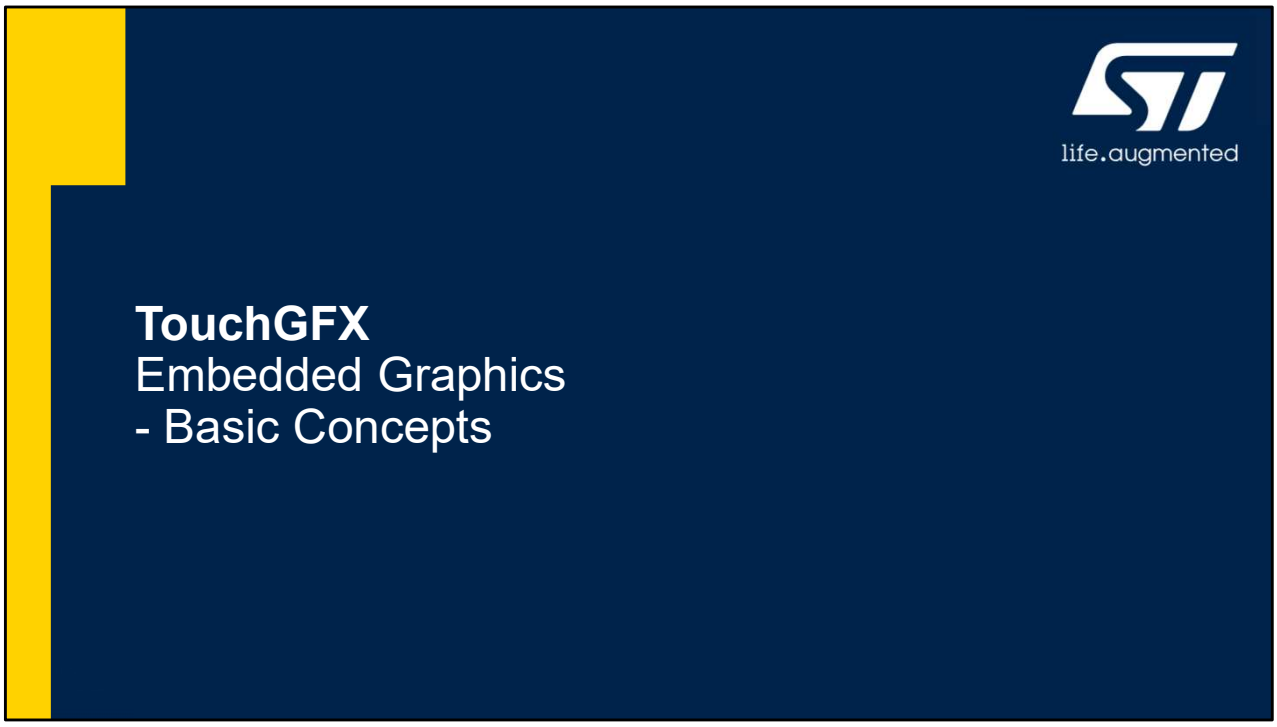


The presentation is aiming at some of the basic key concepts that is important to know when developing an embedded UI, with focus on TouchGFX.
Starting out by introducing some general topics, before talking more about more TouchGFX orientated subjects.

The main chapter covered are
(https://support.touchgfx.com/docs/basic-concepts/embedded-graphics)

This presentation wil cover some background knowledge for embedded UI development, and there won't look at how to use TouchGFX.

Presentations for getting started with UI development and TouchGFX can either be the presentation:

https://support.touchgfx.com/docs/resources/presentations#ui-development---fundamentals)
Or the workshop for a step-by-step guide

https://support.touchgfx.com/docs/resources/presentations#ui-development---

getting-started)

This presentation takes approximately ~ 1 hour

Agenda

1. Introduction
2. Hardware
3. Color Formats
4. Framebuffer
5. Graphics Engine
6. Main Loop
7. Main Loop & Framebuffer
8. Performance
9. Operating Systems

life.augmented

The agende for the presentation is a short introduction.

Introduction

Introduction

Goal of this presentation

- General knowledge about embedded graphics
  - Basic hardware
  - Color formats
  - Framebuffer
- Basic knowledge about the TouchGFX Engine
  - The Graphics Engine
  - The Main Loop
  - Performance
  - Operating System

life.augmented

Introduction

Further reading

- You will find a lot of help afterwards in the TouchGFX documentation site: https://support.touchgfx.com/
- Slides in this presentation will refer to relevant documentation pages. Links will be in the lower right-hand corner of the slides
- The presentation is based on the TouchGFX documentation: https://support.touchgfx.com/docs/basic-concepts/embedded-graphics

life.augmented

Introduction

TouchGFX Development

- Main Components:

- Main Activities

- In this presentation, we will not focus on a specific component or activity as it is an introduction to the basic elements. We will instead focus on important know-how elements when working with components and activities related to TouchGFX Development

life.augmented

Documentation Link: Development Introduction 6

Ref technical introduction

Introduction

- **What is Embedded Graphics?**
  - *What it can be*
    - All embedded devices with a graphical display
    - Modern smartphone, with high resolution and 3D animations
    - Old 8 bit MCU and a character og segment displays
  - *What it is not*
    - A personal computer and tablets
  - *What it is to TouchGFX*
    - UIs on STM32 microcontroller (32 bit MCUs).
    - Interactive applications in 2-2.5D.
    - User interface running at 30-60 frames per second.

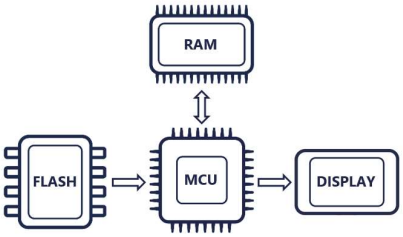
life.augmented

Documentation Link: Embedded Graphics

Hardware

Hardware

- The basic hardware can be divided into 4 parts
  - MCU
    - Heavy Lifting
  - RAM
    - Framebuffer
  - Flash
    - Images, Fonts, Texts
  - Display



- More information can be found in the documentation:

  Hardware Selection

life.augmented

Documentation Link: Embedded Graphics 9

The Main parts

Flash

Static data, stores the image, Fonts and texts

Ram

Mainly for framebuffer, which is where the calculated image is stored, and transferred from and to the display

Display

Image from Ram (sizes, types, resolution)

MCU

Handles the heavy lifting, calculates the color, renders transfer images

Muliple set up with internal ram, flash, displays

Example: STM32H750B-DK, external flash and external ram are used for TouchGFX and the display is a LTDC display
https://www.st.com/en/evaluation-tools/stm32h750b-dk.html

Color Formats

Color Formats

- Digital images can be divided into small single components called "pixels"
- Pixels
  - Single color
  - RGB
  - 0 to 255
  - Opacity (Alpha - RGBA)

life.augmented

Documentation Link: Color Formats 11

Color Formats

- Color Depths
  - Amount of bits to describe a pixel
    - Bits per pixel (BPP)
    - Range 1 – 32
  - Affects visual quality at the cost of memory
  - The higher BPP, the more memory is needed
  - Low BPP can be compensated with Dithering

24 bpp

8 BPP – with dithering

8 BPP – without dithering

Documentation Link: Color Depth

Amount of info/bits to describe pixel.
1 - 2 colors
8 bpp - 256 colors
24 bpp - 16777216 colors
32 bpp – 16,777,216 colors and corresponding opacity values

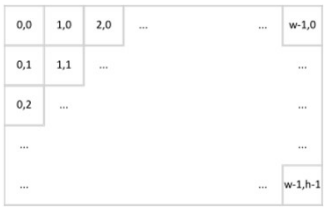
Range 16 bpp – 5 Red, 6 green, 5 blue
8/6 bits 2 pr. RGB

Dithering = Adding noise

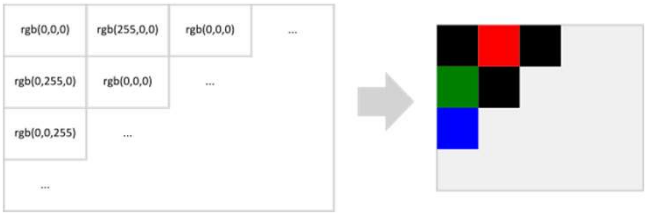
Framebuffer

Framebuffer

- What is a Framebuffer?
  - A framebuffer contains pixel information
  - A 2D memory block with the image to display
  - The framebuffer size is determined by the screen resolution of the display and the pixel BPP



The 2D framebuffer, indexable by x, y



From framebuffer pixel values to colors on a display

Framebuffer

- The amount of framebuffers can be divided into 3 strategies
  - More than one (usually two)
  - One
  - Less than one
- Strategy impact on performance
  - More than one gives better performance
  - But cost more memory
- Calculating framebuffer cost
  - The cost is $width \times height \times bpp \times nr.of\ frambuffers \div 8\ bits$
  - Screen resolution at 800 x 480, color depth of 24 BPP and 2 framebuffers
  - $800 * 480 * 24\ bpp * 2\ framebuffers \div 8\ bits = 2.304\ Mb$

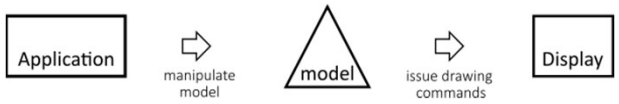
life.augmented

Documentation Link: Framebuffer 15

Graphics Engine

Graphics Engine

- TouchGFX is a retained mode graphics engine
  - User manipulates an abstract model
  - The engine translates the model to drawing operations



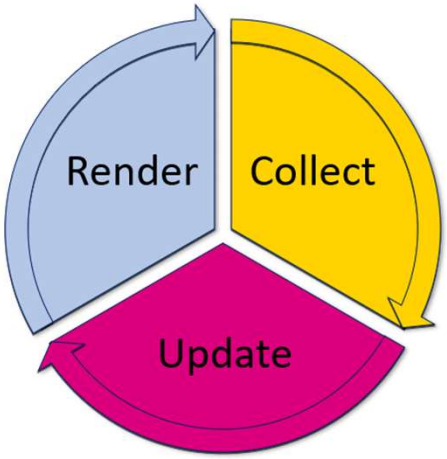
- The components in the model is referred to as "widgets"
  - Graphics displayed by interacting with the widgets
  - Interaction is done in the Designer or C++ code
  - Handling of the drawing

life.augmented

Main Loop

Main Loop

- TouchGFX runs as an “infinite” loop
  1. Collecting events
     - Touchscreen and buttons
     - Backend (Ticks)
  2. Updating
     - Changes the “state” of the widgets based on the events collected
  3. Rendering
     - TouchGFX draws relevant updates in the framebuffer
  - After the loop, TouchGFX waits for a signal and runs the loop again



life.augmented

Documentation Link: Main Loop 19

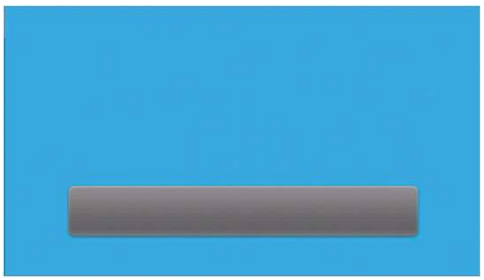
TouchGFX three main acitvies

Ticks is called any time before a new frame is transferd therefor before a loop starts

To ensure that the rendering is synchronized with the display
Frames are rendered at a fixed rate, which makes it easier to setup an animation fx.
1 sec animation at 60 Hz = 60 steps in the animation

Main Loop

- Example



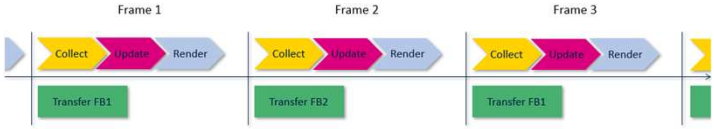
Documentation Link: Main Loop 20

First press is collected and sent to button,
button uses the information that it is pressed to change it's picture
and tells the engine to invalidate the area covering the button.
The engine then update the framebuffer with the new picture.
The release is then collected and sent to the button widget.
Other than updating it's own picture the button tells the screen it's been clicked which is connected to a show text, and the text is added to the list of elements to update.
Then that picture is sent to the framebuffer to display on the screen.

Framebuffer & Main Loop

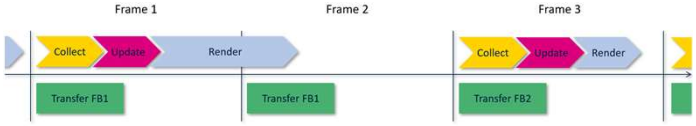
Framebuffer & Main Loop

- Effect of the framebuffer strategy
  - More than one framebuffer
    - When a loop is done, the framebuffer is switched



    - Nothing has changed and the framebuffer is retransmitted

    - If the rendering is to slow, the the old framebuffer is retransmitted

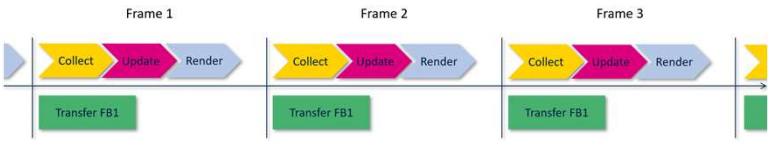


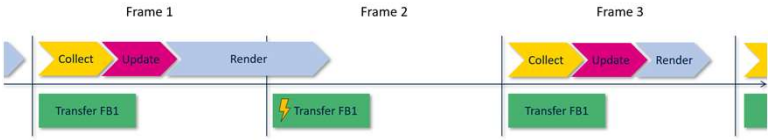
life.augmented

Documentation Link: Handling the framebuffers

Framebuffer & Main Loop

- One framebuffer
  - We have to draw in the same framebuffer that is being transmitted
  - Creates risk for transmitting part of the old frame “tearing”
  - Solutions are only drawn when transfer is done or in the already transferred part of the framebuffer



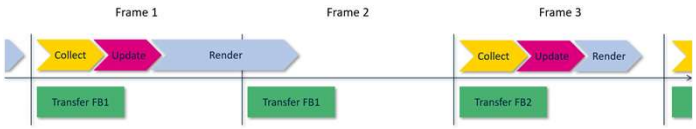


Documentation Link: Handling the framebuffers

Performance

Performance

- Good performance
  - Desired graphics and animations
  - High frame rate
- Usually the frame rate is around 60 Hz
  - A loop therefor has 1 s / 60 = 0.01667 s = 16.67 ms
- Using more time results in a lower frame rate



life.augmented

Documentation Link: Performance 25

Performance

- Example

life.augmented

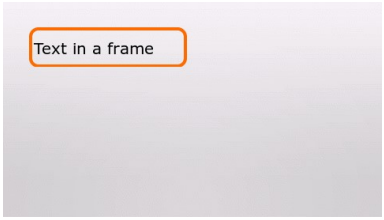
Documentation Link: Performance 26

Left: STM32F746
Right: STM32H735

Notice the performance different

Performance

- What affects the rendering time?
  - The amount of the updated screen
    - Larger area = more computation
  - Layers in graphics
    - Each layer requires rendering
  - Complexity of the rendered pixel
    - Some widgets are heavier to render than others
    - Transparency adds to complexity
  - Hardware support for rendering
    - Offloads the MCU
    - Chrom-ART

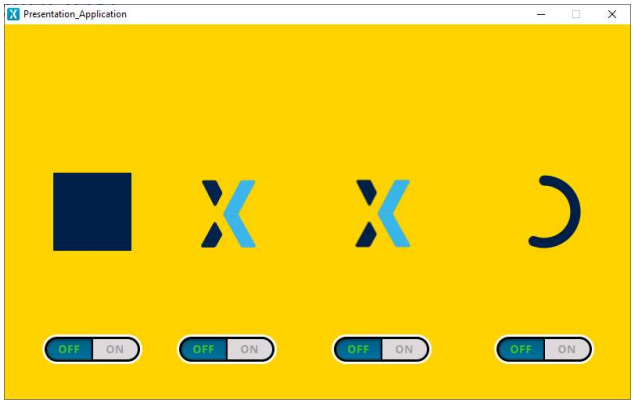


life.augmented

Documentation Link: What Affects the Rendering Time? 27

Performance

- Example



life.augmented

Documentation Link: What Affects the Rendering Time? 28

From the left

1: Is a box widget, fast to display since it is just a full pixel color and therefore no calculation
2: Is an image widget, fast to display, since we are coping pixel information from the flash. The amount of non-opaque pixel, in the image, can have an impact on the performance, since they need to be blended with the framebuffer
3: Is a texture mapper widget, slow to display, since the new pixels needs to be calculated before being drawn into the framebuffer.
4: Is a circle widget, slow to display, since the new pixels needs to be calculated before being drawn into the framebuffer.

Operation Systems

Operation Systems

- Embedded devices handling more than the UI
- RTOS
  - Interleaving tasks
    - Ensure that the UI is not blocked by another proces or the like
  - Communication between tasks
    - Utilizing RTOS message queue
  - FreeRTOS
- No RTOS
  - Small setups with low complexity

More Info

More info

- To learn more about the hardware and what to select, read the Hardware Selection Introduction
- To get started with the UI development, read the UI Development Introduction
- Or watch the presentation or workshop

  UI Development – Fundamentals

  UI Development - Getting Started

life.augmented

Thank you

life.augmented