19039 GFX2

Creating Graphical Applications Using MPLAB® Harmony

Class Objectives

When you walk out of this class you will....

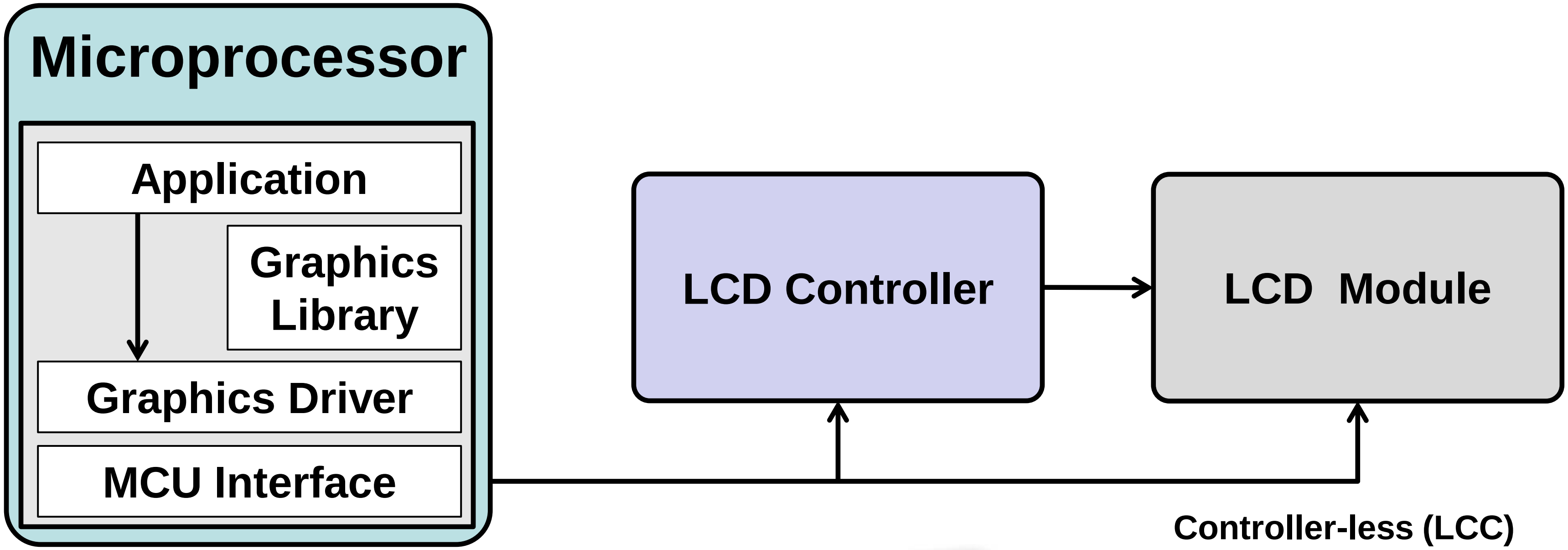
- Understand the MPLAB Harmony graphics development process
- Be able to quickly create graphical MPLAB Harmony applications using MPLAB Harmony Graphics Composer

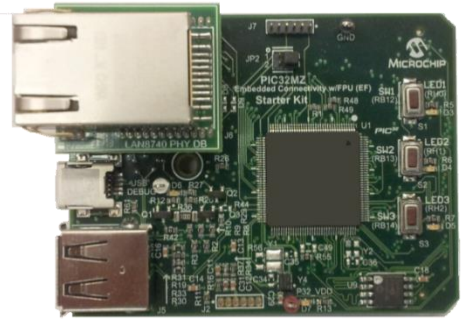
Class Agenda

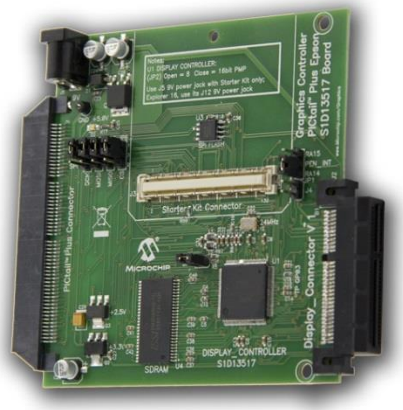
- Embedded Graphics Systems
- MPLAB Harmony Overview
- MPLAB Harmony Configurator
- Lab 1 - Lab Hardware Setup
- Graphics Theory
- MPLAB Harmony Graphics Library
- MPLAB Harmony Graphics Composer
- Lab 2 – Splash Screen
- Lab 3 – Basic Menu
- Lab 4 – LED Control Screen
- Lab 5 – Temperature Display Screen
- Summary

Embedded Graphics Systems

Embedded Graphics System Components



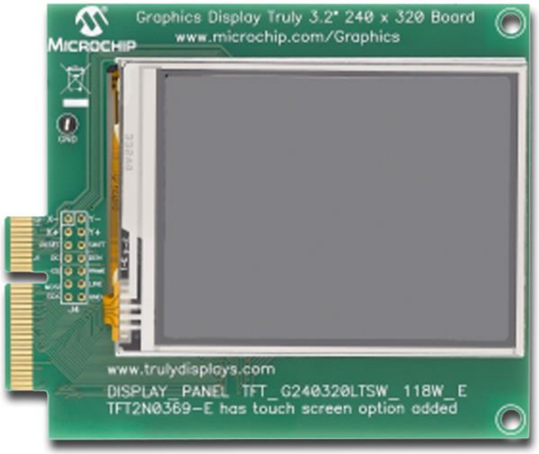


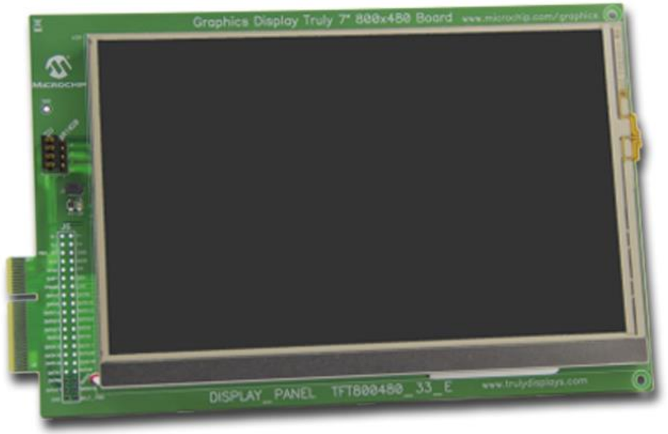


Controller-less (LCC)

Graphics Display Devices

Designation	Resolution	Pixel Count (width * height)
2-inch	176x220	38720
QVGA	320x240	76800
WQVGA	480x272	130560
VGA	640x480	307200
WVGA	800x480	384000

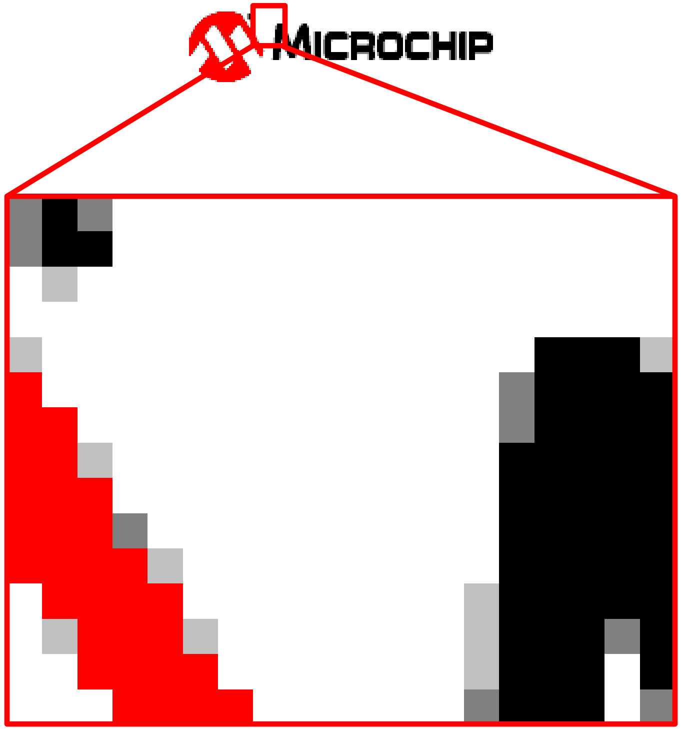

Graphics Display Truly 3.2" 240 x 320 Board
www.microchip.com/Graphics
MICROCHIP
www.trulydisplays.com
DISPLAY_PANEL TFT_G240320LTSW_118W_E
TFT2N0369-E has touch screen option added


Graphics Display Truly 7" 800x480 Board
MICROCHIP
DISPLAY_PANEL TFT800480_33_E
www.trulydisplays.com

What is a Pixel?

Definition

A pixel is a physical point in a raster image, or the smallest addressable element in an all points addressable display device.



Frame Buffers

Definition

A frame buffer is a block of memory used to store the data representation of a display image.

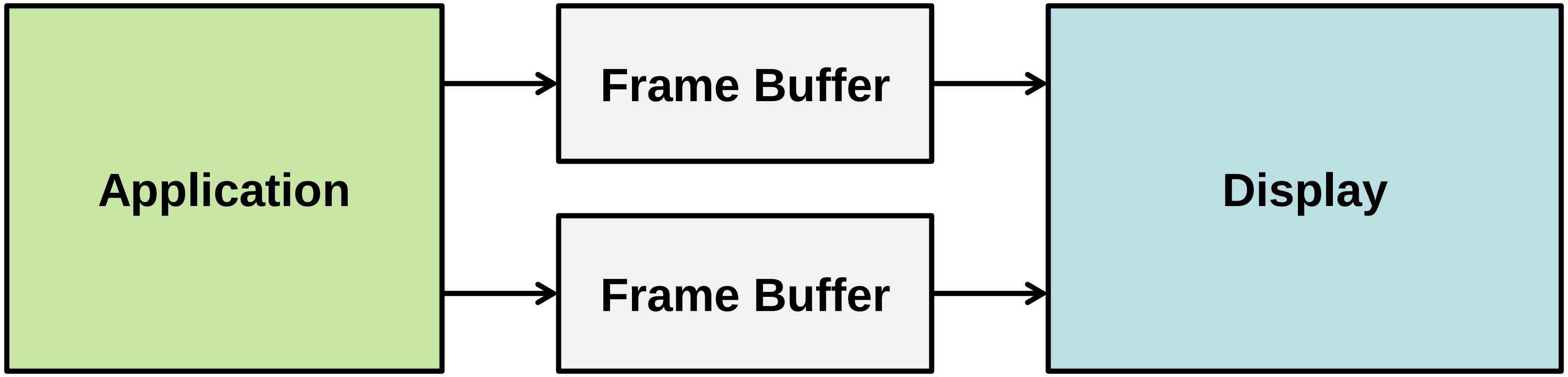
- **Size depends on display size and color depth**

How to calculate your frame buffer size (in bytes):

Size = (Bits Per Pixel / 8) * (Display Width) * (Display Height)

Multiple Frame Buffers

Graphic-intensive applications may need to use multiple frame buffers.



Application

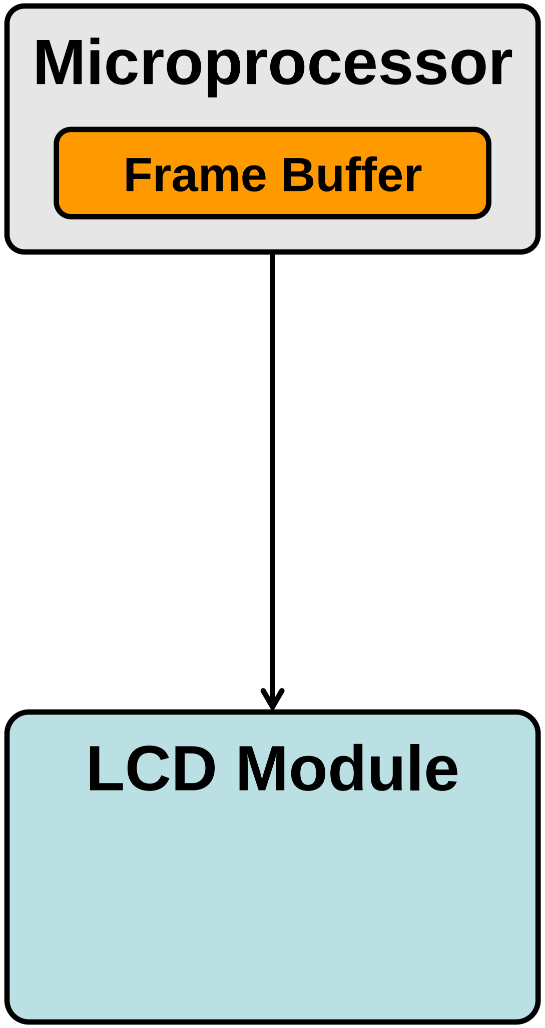
Frame Buffer

Frame Buffer

Display

Adding additional frame buffers increases the memory requirements exponentially.

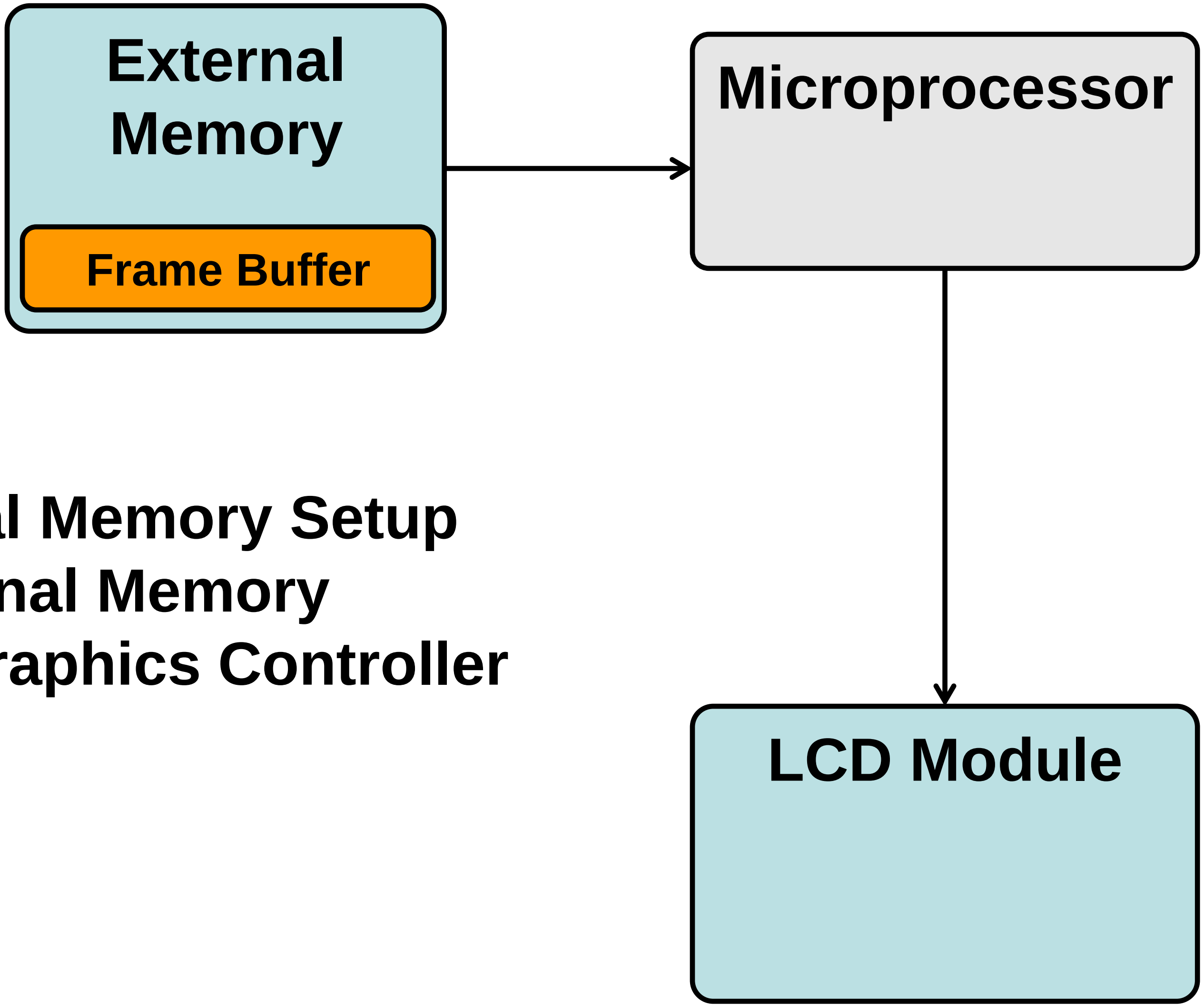
Graphical Setups



Basic Setup

- Internal Memory
- No Graphics Controller

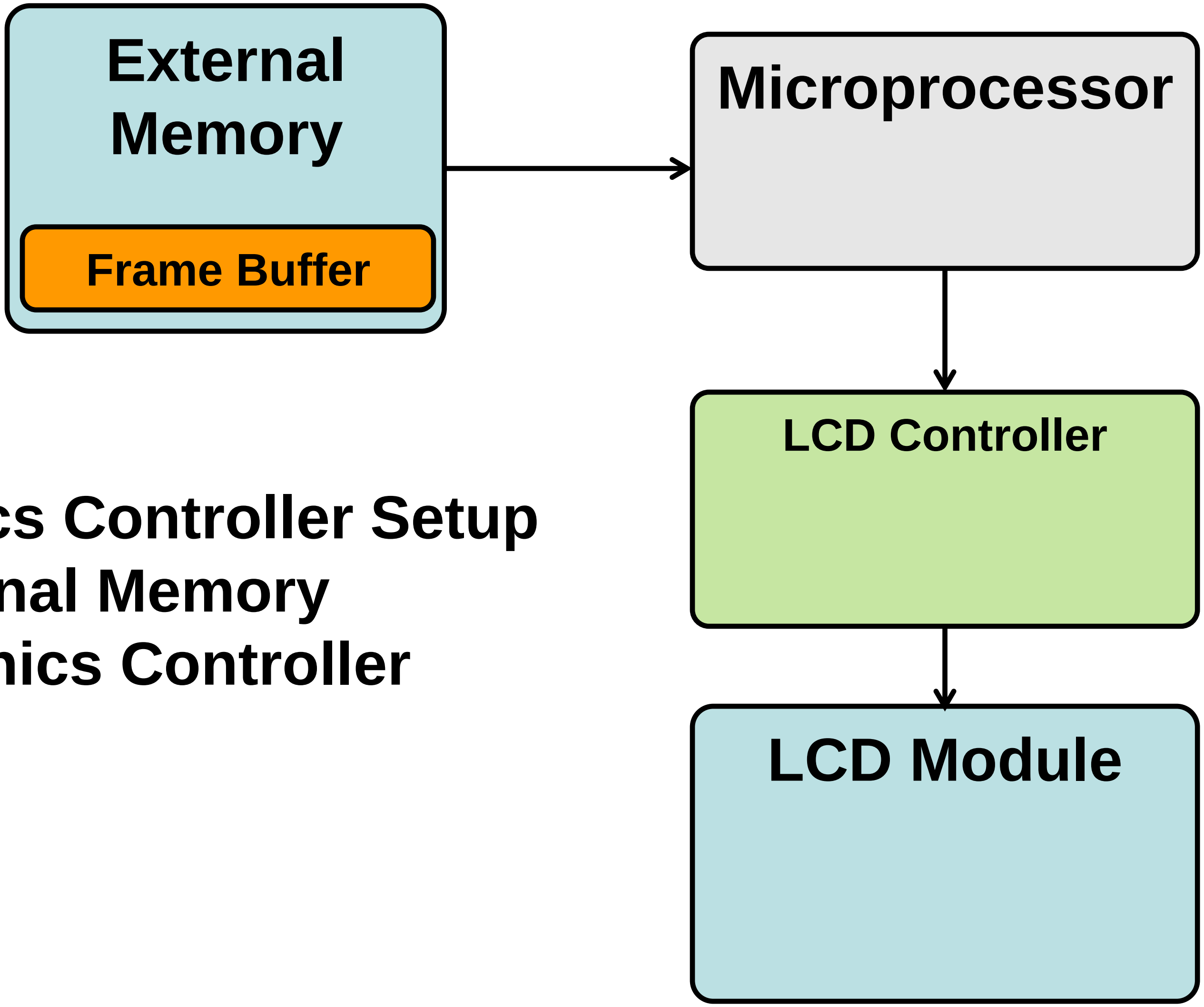
Graphical Setups



External Memory Setup

- External Memory
- No Graphics Controller

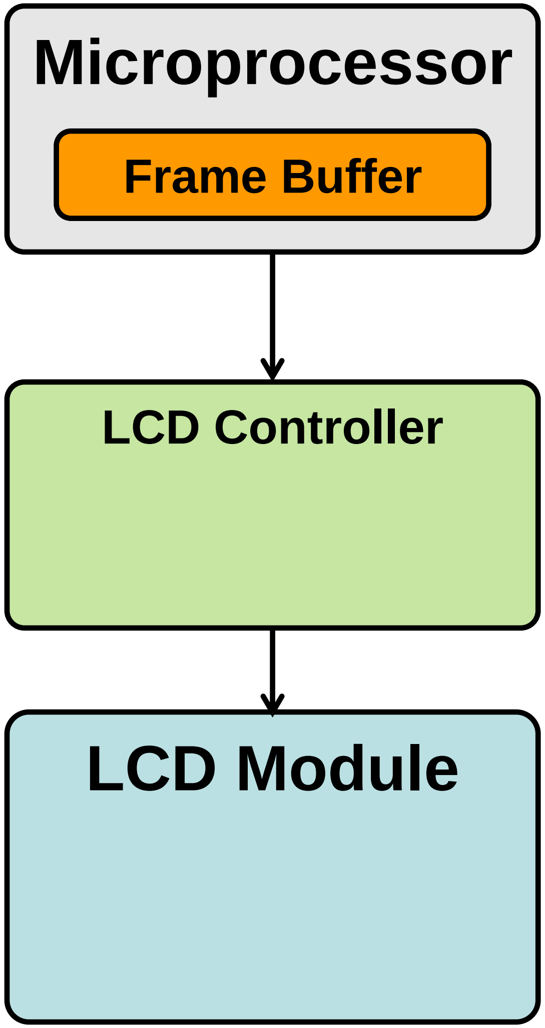
Graphical Setups



Graphics Controller Setup

- External Memory
- Graphics Controller

Graphical Setups



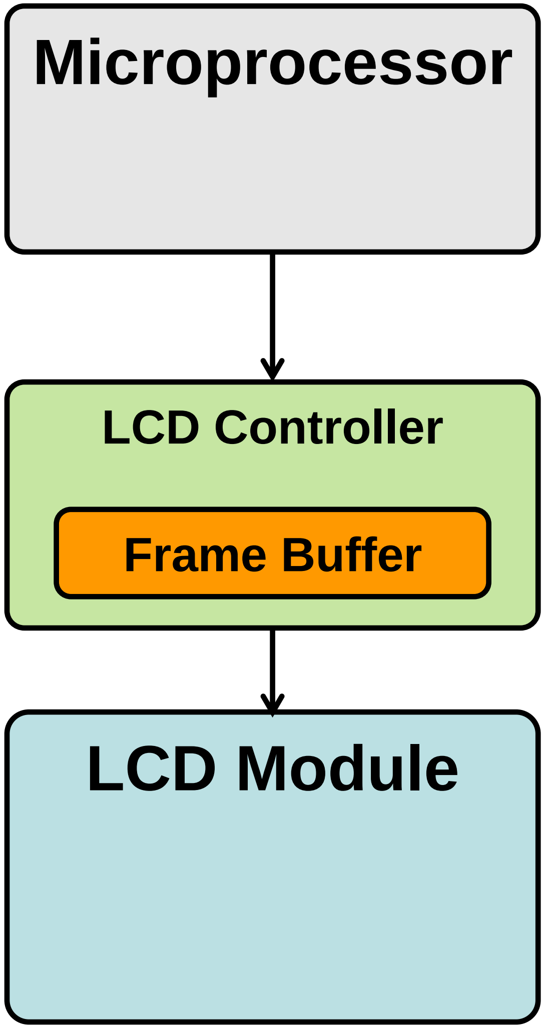
Graphics Controller Setup

- Internal Memory
- Graphics Controller

Graphical Setups

Graphics Controller Setup

- Controller Memory
- Graphics Controller



Calculate Frame Buffer Size

Calculate the frame buffer size requirement for a PIC32MZ LCC configuration.

- PIC32MZ LCC supports up to 512 KB RAM
- Use external SRAM for frame buffers > 512 KB

Display Resolution			Memory Requirement (bytes)				
			1 bpp (Mono)	2 bpp (4 shades)	4 bpp (16 shades)	8 bpp (256 colors)	16 bpp (65K colors)
WVGA	800x480	7”	48,000	96,000	192,000	384,000	768,000
VGA	640x480	5.7”	38,400	76,800	153,600	307,200	614,400
WQVGA	480x272	4.3”	16,320	32,640	65,280	130,560	261,120
QVGA	320x240	3.2”	9,600	19,200	38,400	76,800	153,600
OLED*	128x64	1”-2.7”	1,024	2,048	4,096	8,192	16,384

* Represents common values

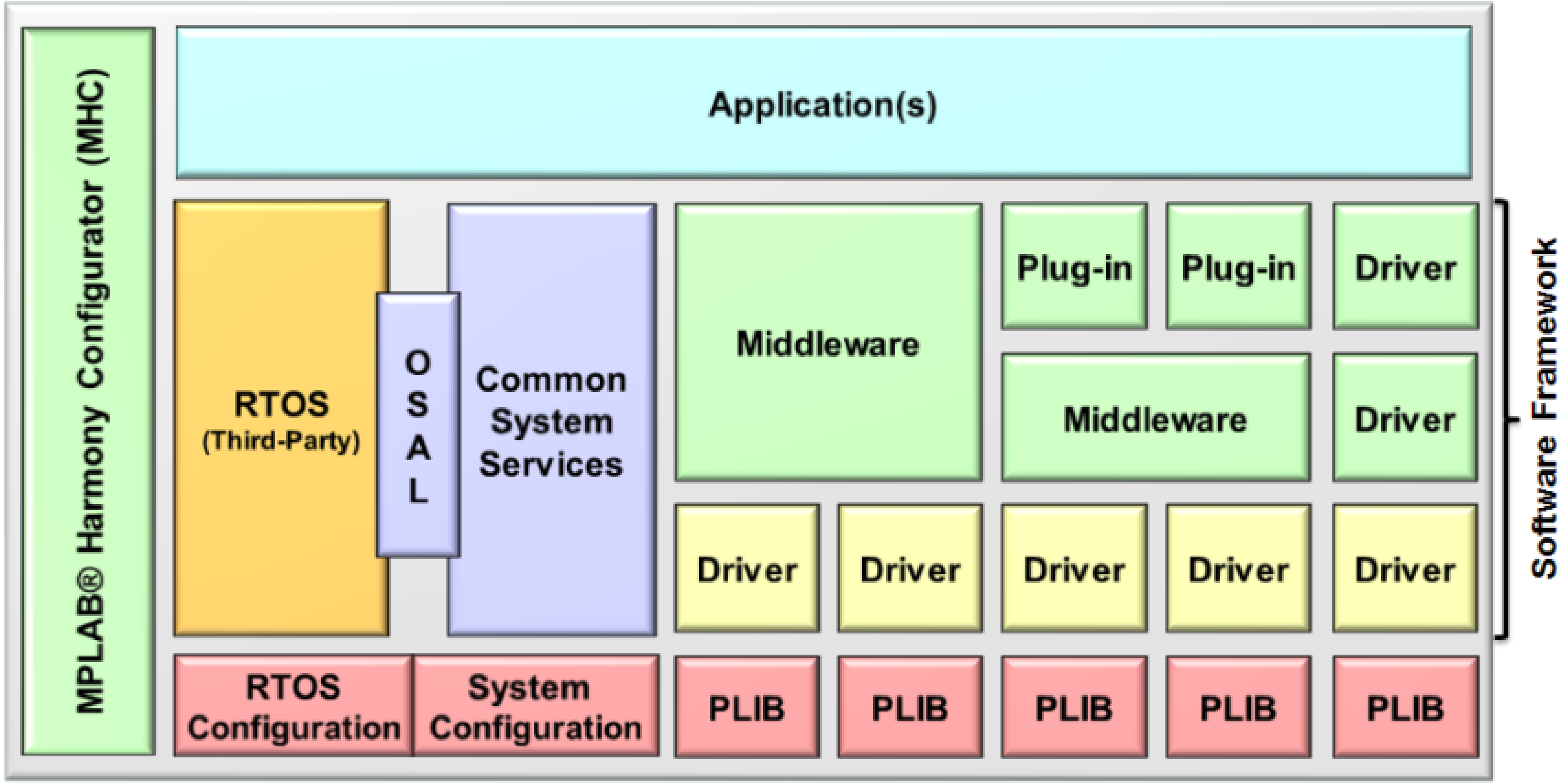
🟩 **Internal SRAM on PIC32MZ LCC**

🟦 **External SRAM**

- **Questions about:**
  # Embedded Graphics Systems?

MPLAB Harmony Overview

MPLAB Harmony Architecture



MPLAB Harmony Configurator (MHC)

What is MHC?

- Integrated Hardware Configuration Utility
- Visual configuration
- Fully integrated with MPLAB X IDE via plug-in
- Data driven architecture
- Template-based code generation

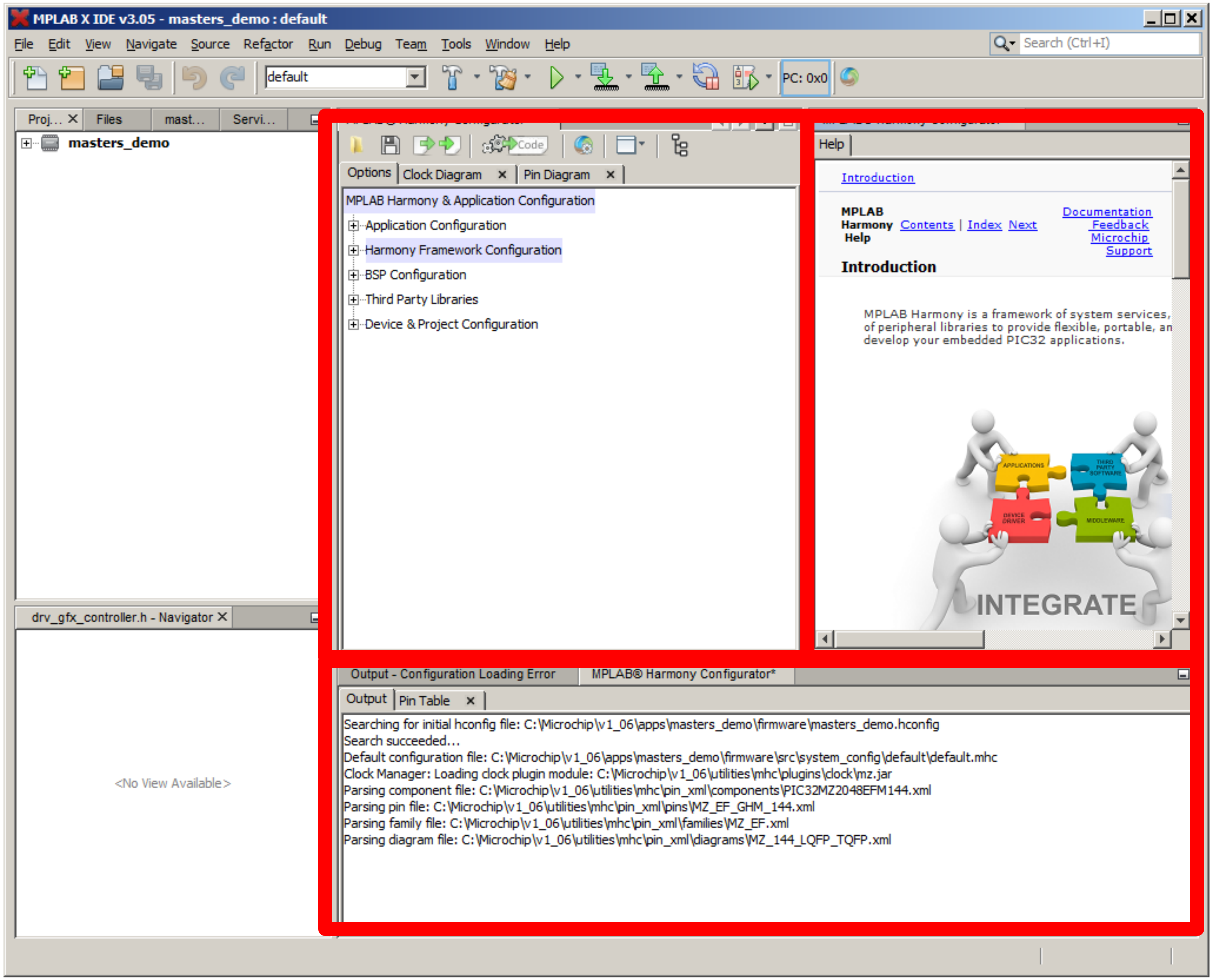
What does MHC do?

- Generates all hardware configuration code
- Generates all middleware framework related code
- Automatically updates MPLAB X IDE project with required files

MHC Layout

Main Areas

- Options
- Help
- Output



MPLAB X IDE v3.05 - masters_demo : default
File Edit View Navigate Source Refactor Run Debug Team Tools Window Help
Search (Ctrl+I)
default
PC: 0x0
Proj... Files mast... Servi...
masters_demo
drv_gfx_controller.h - Navigator
<No View Available>
Code
Options
Clock Diagram
Pin Diagram
Help
INTEGRATE
APPLICATIONS
THIRD PARTY SOFTWARE
DEVICE DRIVER
MIDDLEWARE
Output - Configuration Loading Error
MPLAB® Harmony Configurator*
Output
Pin Table

MPLAB Harmony & Application Configuration
- Application Configuration
- Harmony Framework Configuration
- BSP Configuration
- Third Party Libraries
- Device & Project Configuration

Introduction

MPLAB Harmony Help Contents | Index Next   Documentation Feedback Microchip Support

Introduction

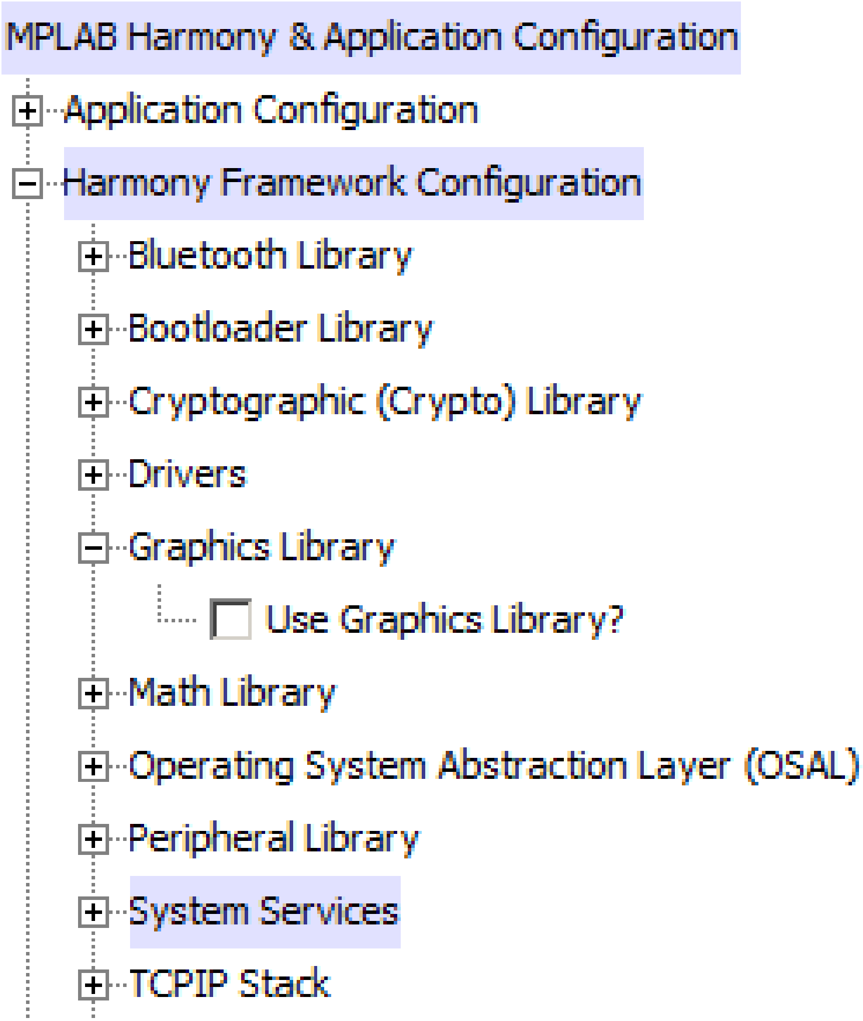
MPLAB Harmony is a framework of system services,
of peripheral libraries to provide flexible, portable, an
develop your embedded PIC32 applications.

```
Searching for initial hconfig file: C:\Microchip\v1_06\apps\masters_demo\firmware\masters_demo.hconfig
Search succeeded...
Default configuration file: C:\Microchip\v1_06\apps\masters_demo\firmware\src\system_config\default\default.mhc
Clock Manager: Loading clock plugin module: C:\Microchip\v1_06\utilities\mhc\plugins\clock\mz.jar
Parsing component file: C:\Microchip\v1_06\utilities\mhc\pin_xml\components\PIC32MZ2048EFM144.xml
Parsing pin file: C:\Microchip\v1_06\utilities\mhc\pin_xml\pins\MZ_EF_GHM_144.xml
Parsing family file: C:\Microchip\v1_06\utilities\mhc\pin_xml\families\MZ_EF.xml
Parsing diagram file: C:\Microchip\v1_06\utilities\mhc\pin_xml\diagrams\MZ_144_LQFP_TQFP.xml
```

MASTERs 2015

MHC Option Tab

- Main Configuration Interface
- Tree Structure
- Hardware Specific Options
- Visual Change Indicators
- Dynamic Options

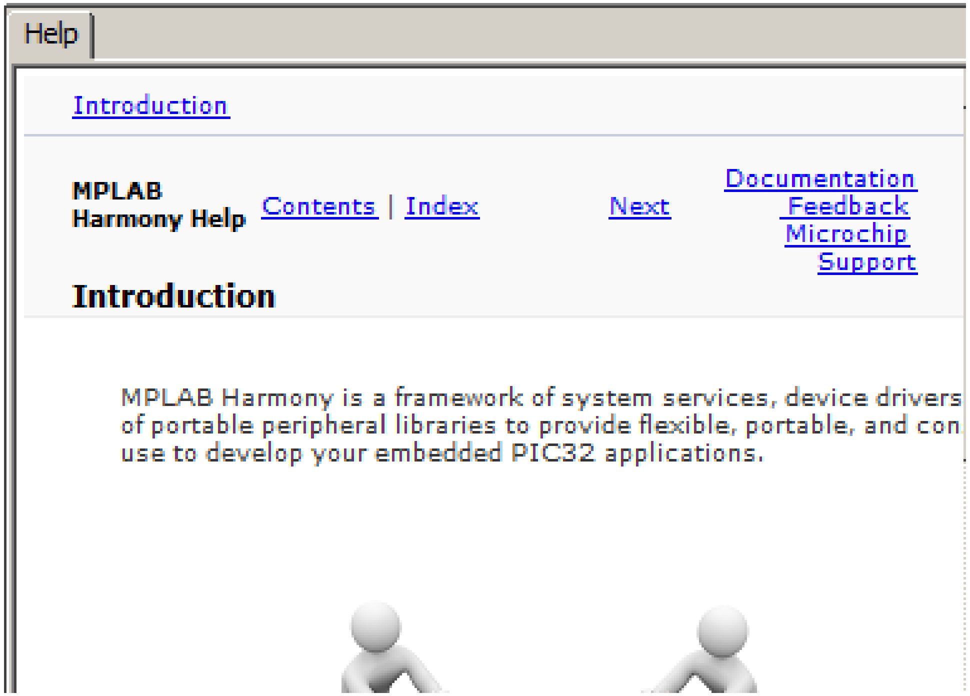


MPLAB Harmony & Application Configuration

- ⊞ Application Configuration
- ⊟ Harmony Framework Configuration
  - ⊞ Bluetooth Library
  - ⊞ Bootloader Library
  - ⊞ Cryptographic (Crypto) Library
  - ⊞ Drivers
  - ⊟ Graphics Library
    - □ Use Graphics Library?
  - ⊞ Math Library
  - ⊞ Operating System Abstraction Layer (OSAL)
  - ⊞ Peripheral Library
  - ⊞ System Services
  - ⊞ TCPIP Stack

MHC Help Tab

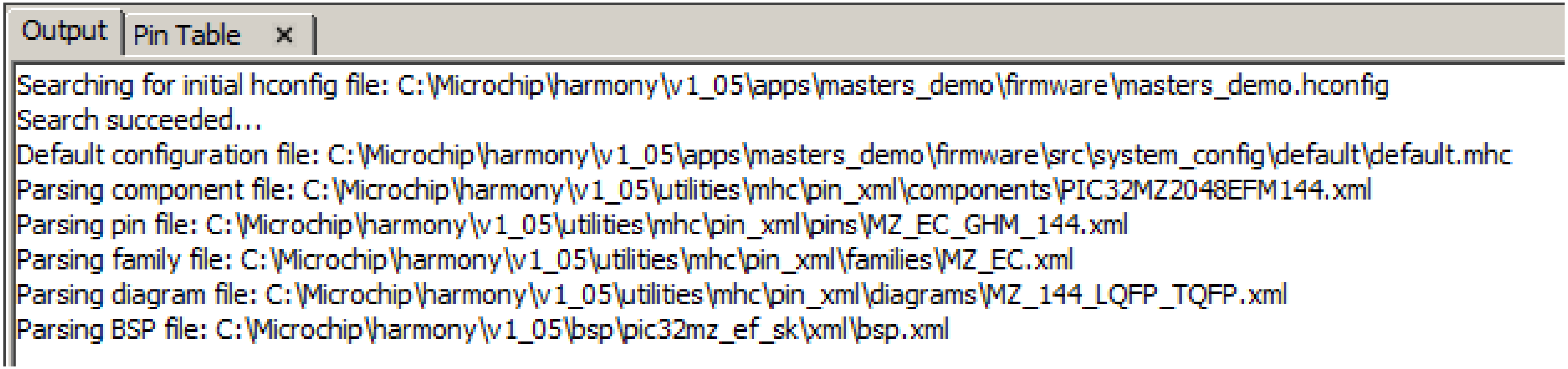
- Integrated MPLAB Harmony Documentation
- Interactive
- Context Sensitive



Introduction

MPLAB Harmony is a framework of system services, device drivers of portable peripheral libraries to provide flexible, portable, and con use to develop your embedded PIC32 applications.

MHC Output Tab



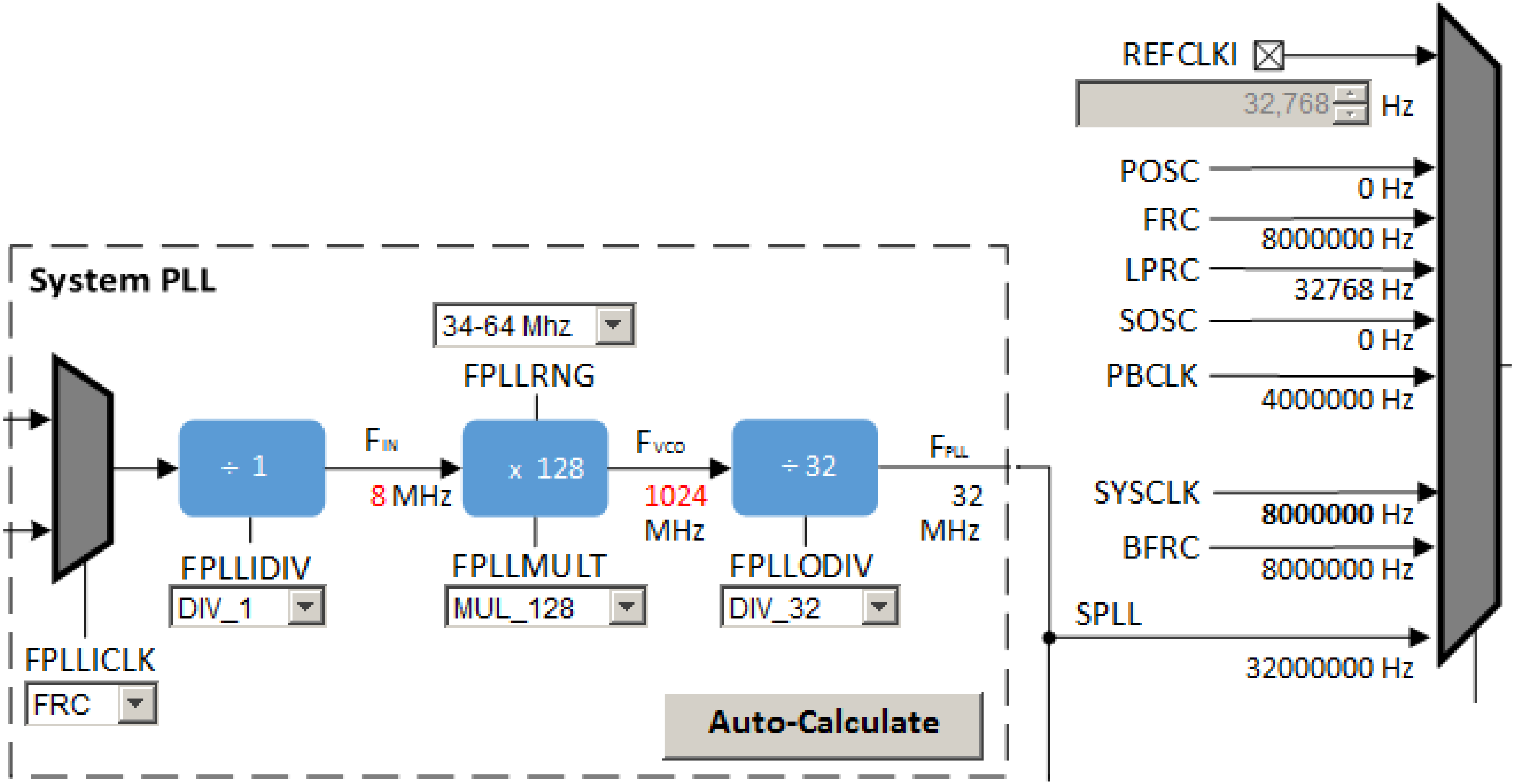
```
Searching for initial hconfig file: C:\Microchip\harmony\v1_05\apps\masters_demo\firmware\masters_demo.hconfig
Search succeeded...
Default configuration file: C:\Microchip\harmony\v1_05\apps\masters_demo\firmware\src\system_config\default\default.mhc
Parsing component file: C:\Microchip\harmony\v1_05\utilities\mhc\pin_xml\components\PIC32MZ2048EFM144.xml
Parsing pin file: C:\Microchip\harmony\v1_05\utilities\mhc\pin_xml\pins\MZ_EC_GHM_144.xml
Parsing family file: C:\Microchip\harmony\v1_05\utilities\mhc\pin_xml\families\MZ_EC.xml
Parsing diagram file: C:\Microchip\harmony\v1_05\utilities\mhc\pin_xml\diagrams\MZ_144_LQFP_TQFP.xml
Parsing BSP file: C:\Microchip\harmony\v1_05\bsp\pic32mz_ef_sk\xml\bsp.xml
```

- Information Output Window
- Adjustable Level of Detail

MASTERs 2015

MHC Clock Diagram Tab

Provides visual interface for configuring oscillators



System PLL

FPLLRNG: 34-64 Mhz

FPLLIDIV: DIV_1

FPLLMULT: MUL_128

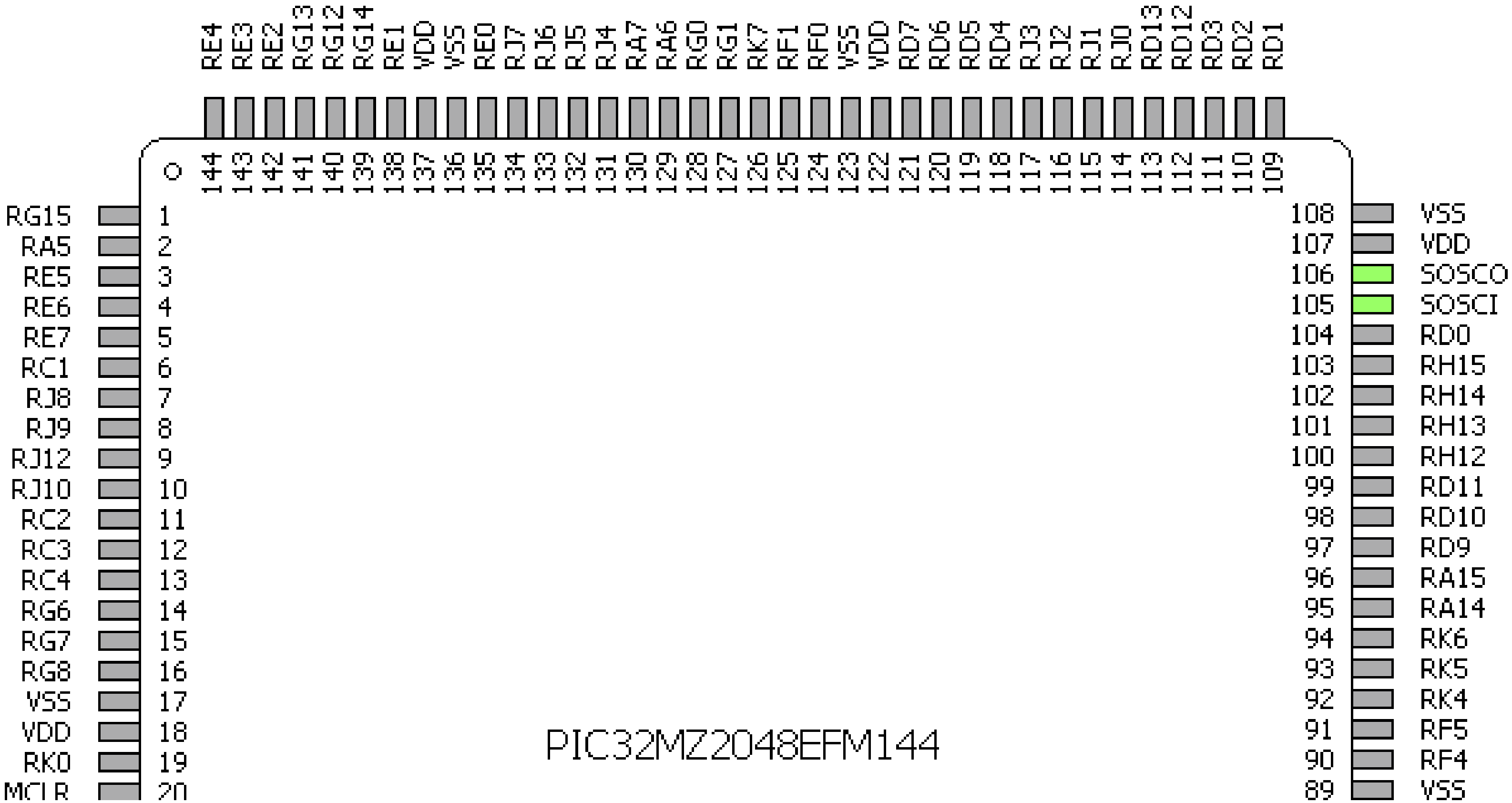
FPLLODIV: DIV_32

FPLLICLK: FRC

REFCLKI: 32,768 Hz

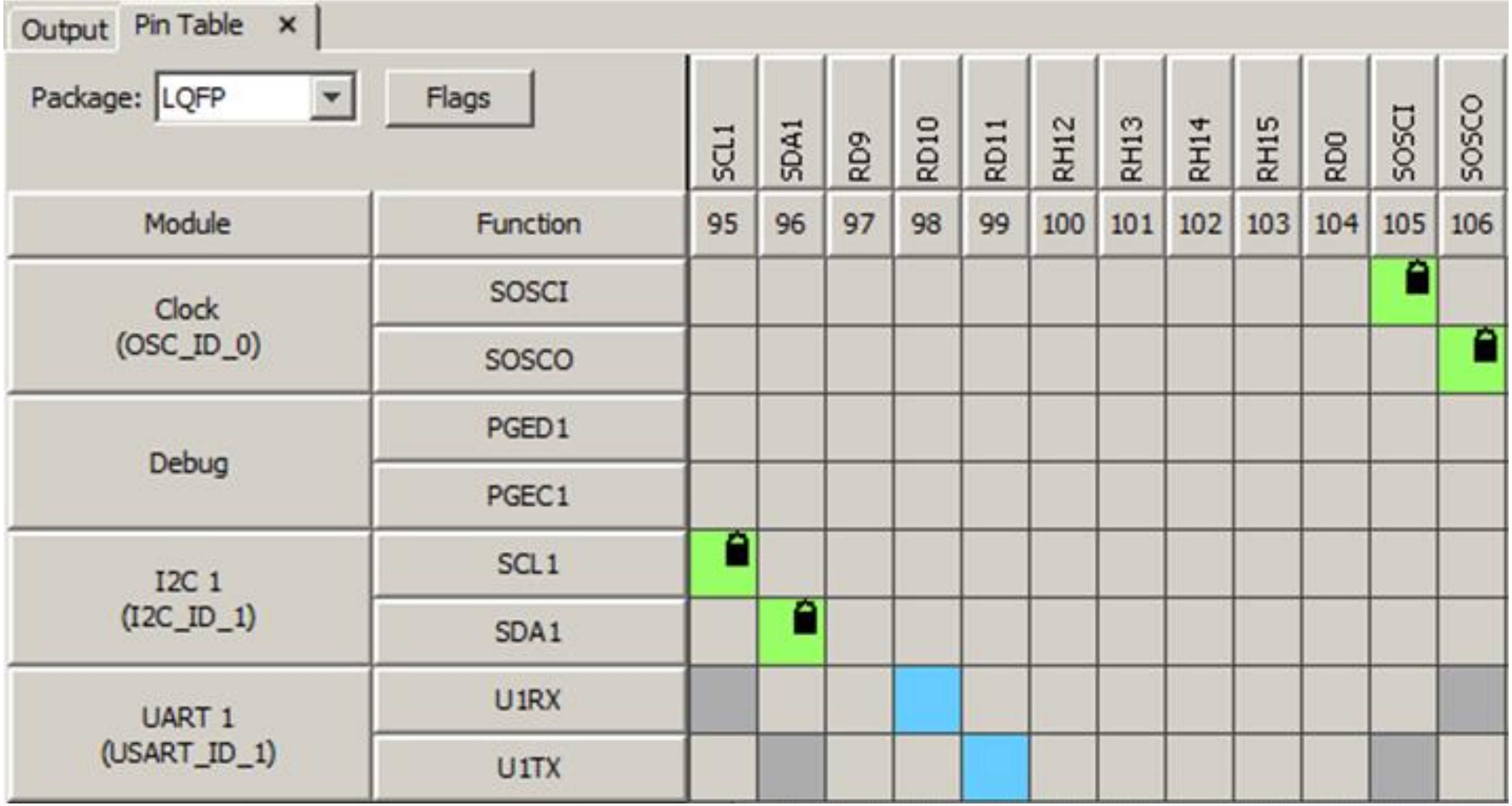
MHC Pin Diagram Tab

Provides visual representation of selected processor



MHC Pin Table Tab

Provides table-based method for configuring selected processor pins



Package: LQFP

		SCL1	SDA1	RD9	RD10	RD11	RH12	RH13	RH14	RH15	RD0	SOSCI	SOSCO
Module	Function	95	96	97	98	99	100	101	102	103	104	105	106
Clock (OSC_ID_0)	SOSCI											🔒	
	SOSCO												🔒
Debug	PGED1												
	PGEC1												
I2C 1 (I2C_ID_1)	SCL1	🔒											
	SDA1		🔒										
UART 1 (USART_ID_1)	U1RX	▨			🟦								▨
	U1TX		▨			🟦						▨	

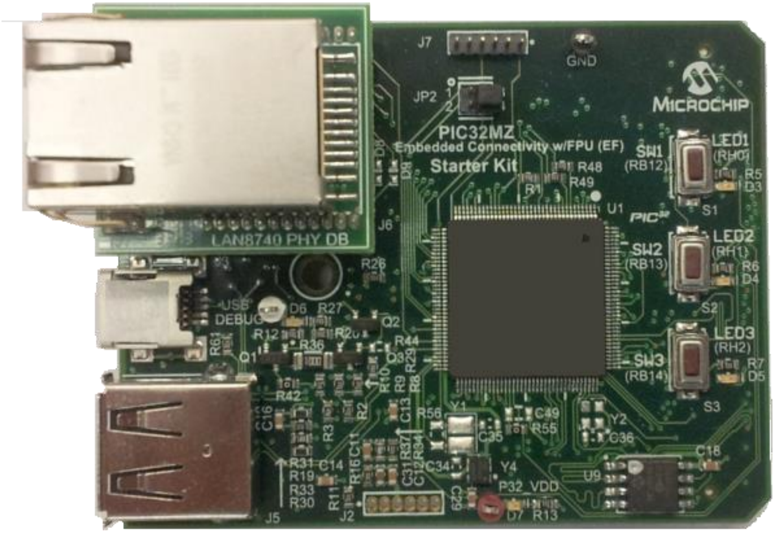
MHC Development Process

- Create New MPLAB Harmony Project
- Configure Clock Settings
- Select Board Support Package (optional)
- Configure MPLAB Harmony Framework
- Configure Pin Settings
- Generate Code
- Create Application Code
- Compile and Flash

- **Questions about:**

  MPLAB Harmony or MPLAB Harmony Configurator?

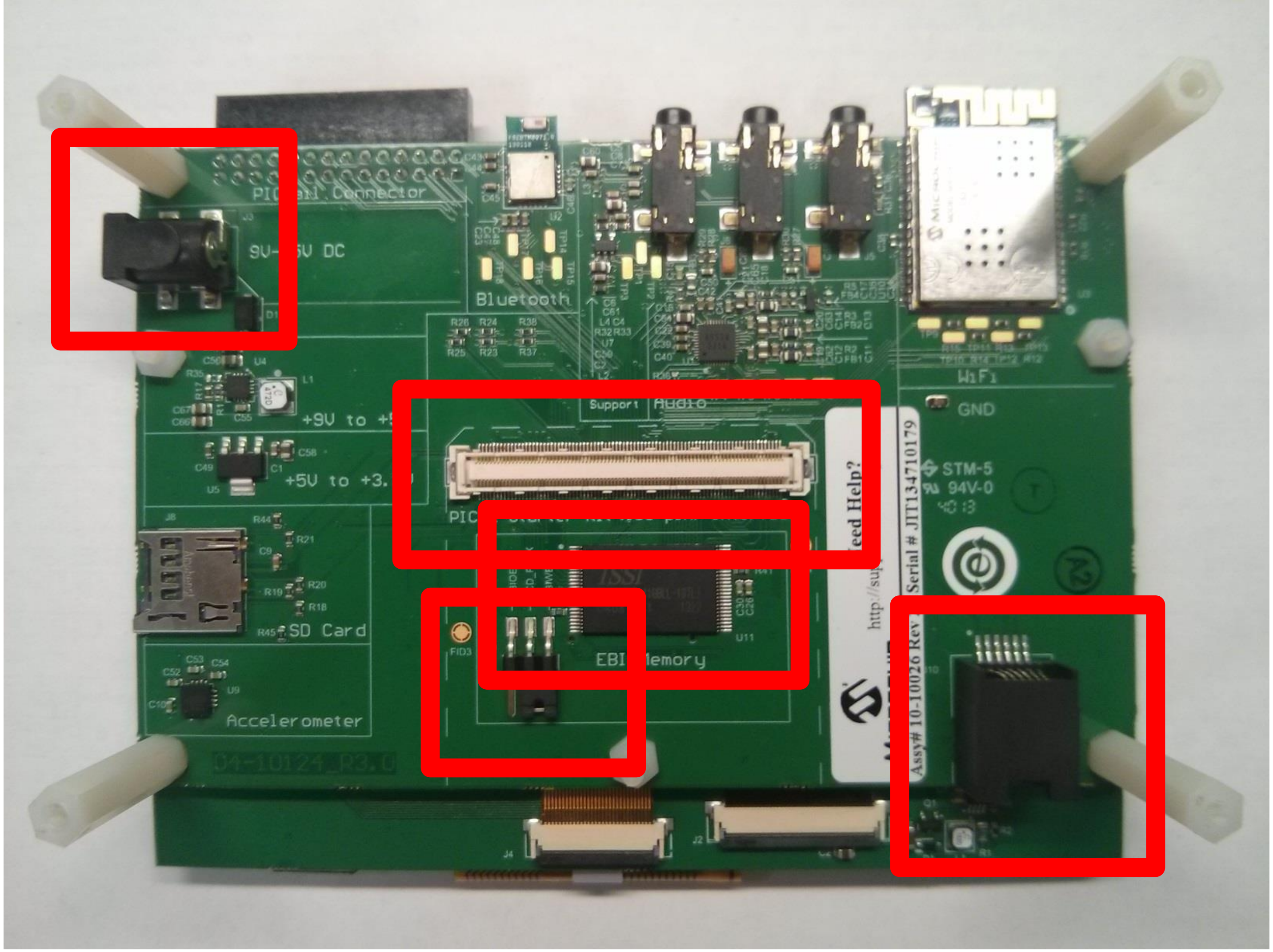
Hardware for Labs



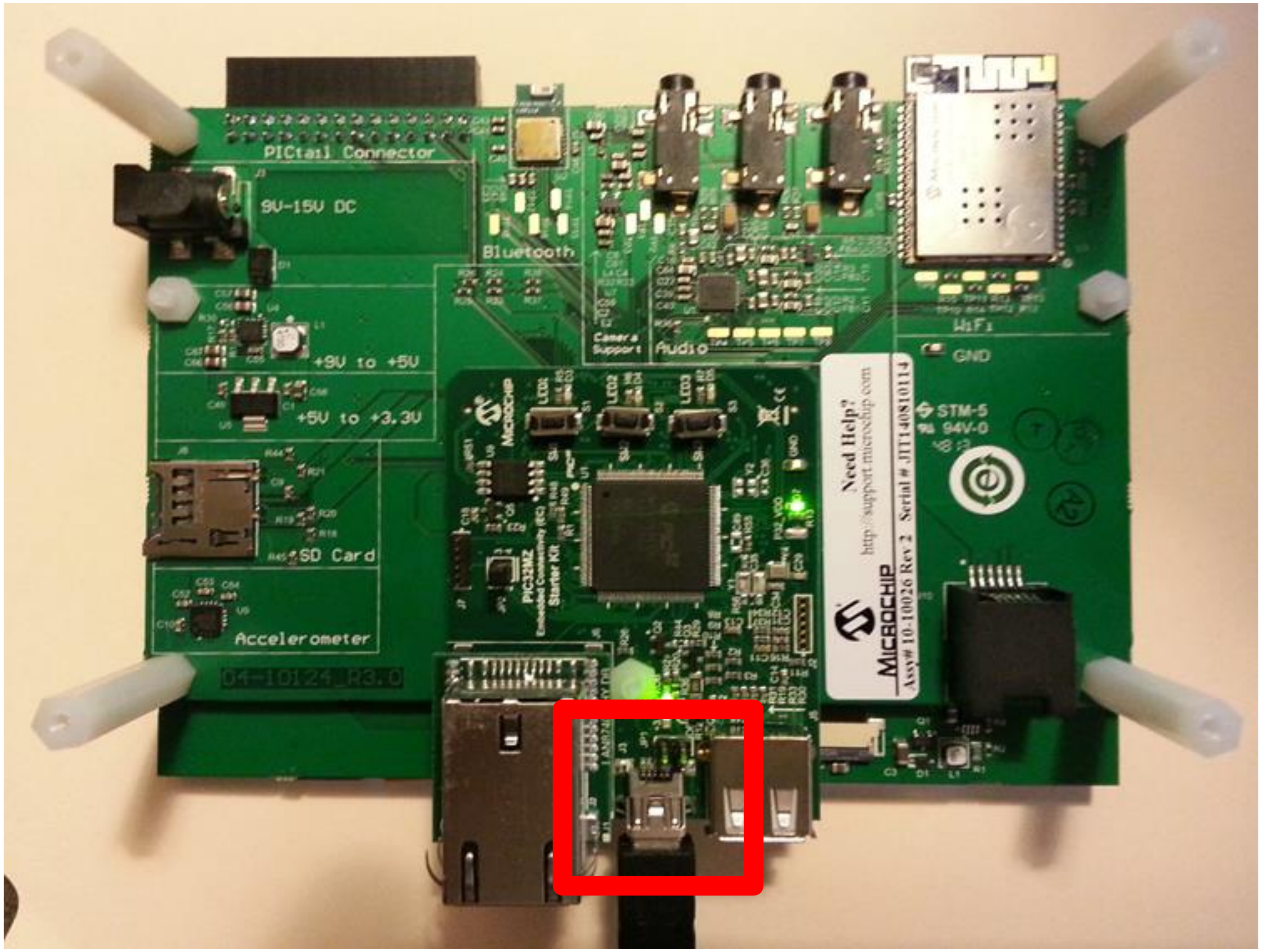
PIC32MZ EF Starter Kit (DM320007)

Multimedia Expansion Board II (DM320005-2)

MEB II Hardware Setup



MEB II Hardware Setup



Lab 1
Getting Started

Lab 1 – Getting Started

Objective:

Using the MHC and the MHGC, you will create a brand new project that will display a background color on the MEB II LCD display.

Lab 1 – Getting Started

Conclusion:

Having gotten your feet wet with the MHC and the MHGC, we are now ready to use the MHGC to add graphics to the project.

Graphics Theory

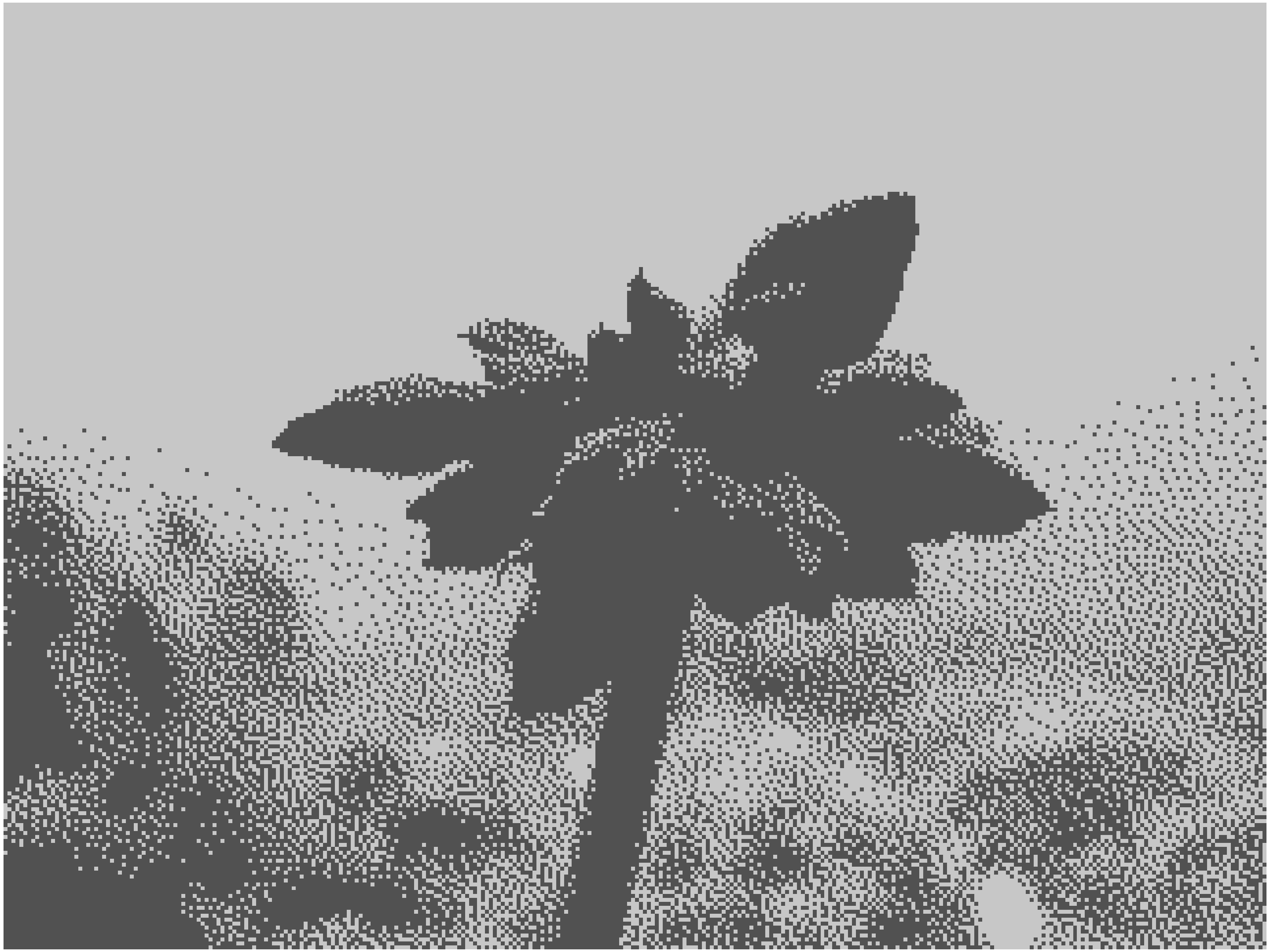
- **Colors**
- Images
- Transparency
- Fonts

Color Depth

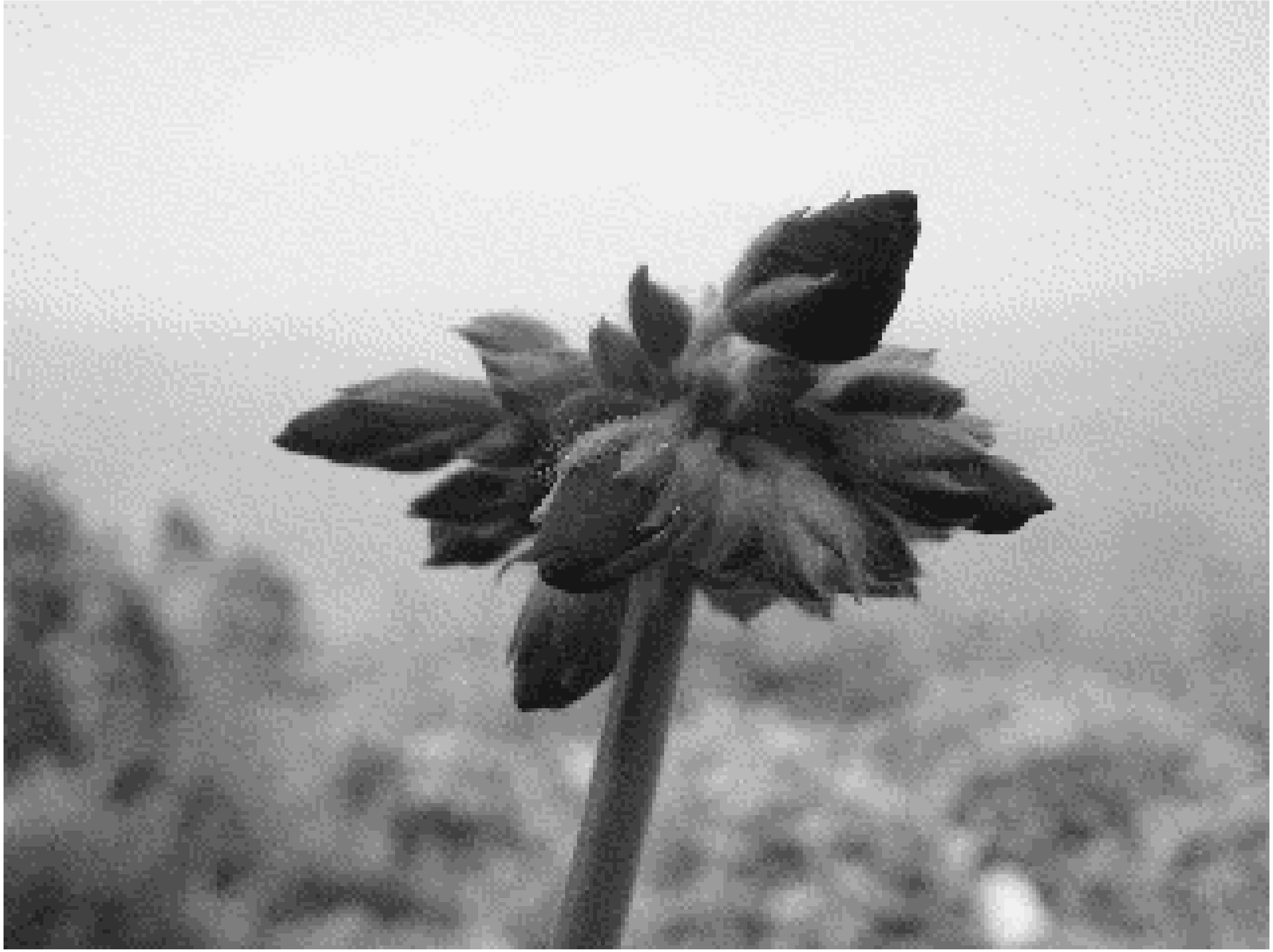
Definition

Color Depth is the number of bits used to represent the color of a single pixel in an image or in a frame buffer. Color Depth is usually specified using a bpp (bits per pixel) notation. A higher color depth gives a broader range of distinct colors, but also requires more memory to store the image or frame.

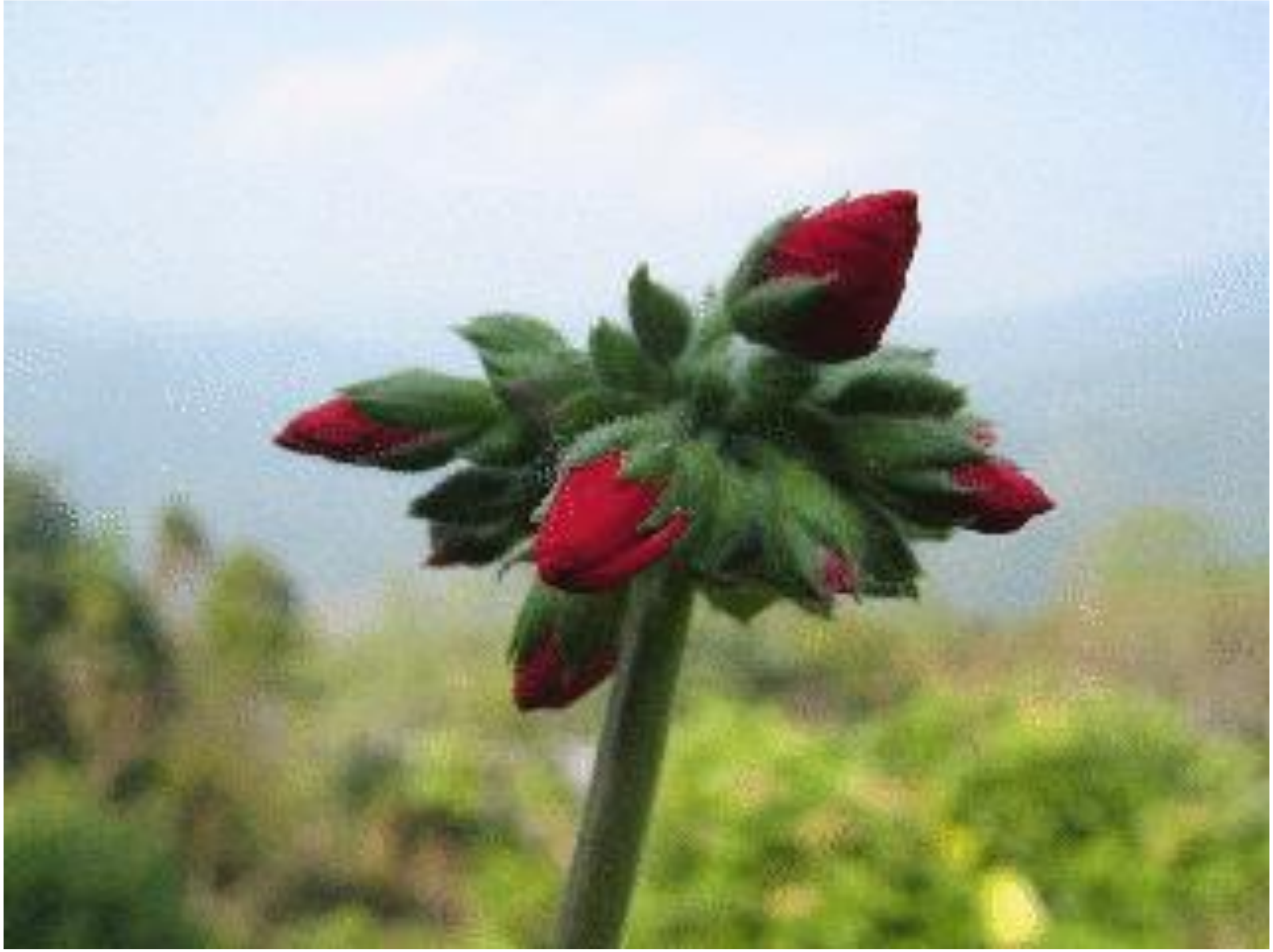
Monochrome (1 BPP)



16 Colors (4 BPP)



256 Colors (8 BPP)

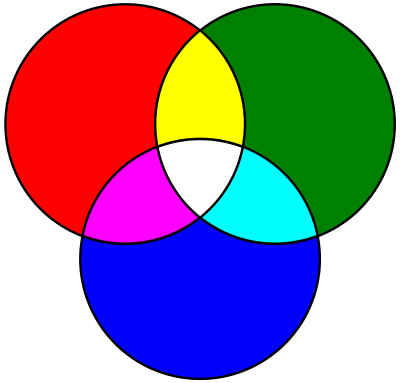


High Color (16 BPP)

RGB Color Model

Definition

The RGB Color Model is an additive color model in which red, green, and blue light are added together in various ways to reproduce a broad array of colors. Colors are expressed as a triplet (RGB) **Source: Wikipedia**



Common Color Formats						
BPP	Color Count	Red	Green	Blue	Alpha	Designation
16	65536	5	6	5	0	RGB_565
18	262144	6	6	6	0	RGB_666
24	16777215	8	8	8	0	RGB_888
32	16777215	8	8	8	8	ARGB_8888

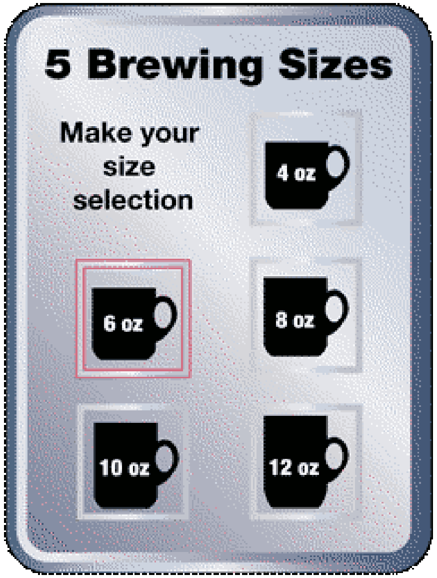
RGB_565 Data Layout

Color:	Red	Green	Blue
	0000 0	000 000	0 0000
Shades:	32	64	32

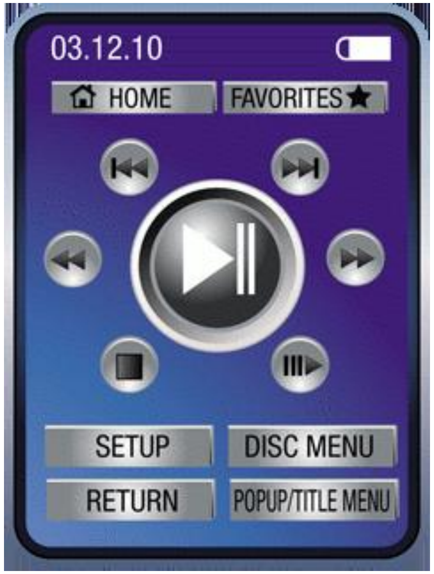
Trivia Question: Why is the extra bit given to the green spectrum?

Answer: The human eye can see more of the green color spectrum than the red and blue spectrums. Therefore, more detail can be gained by increasing the green range.

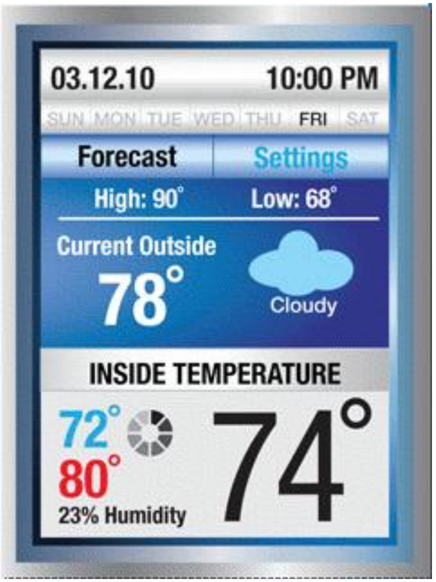
Guess How Many Colors?



16 Shades – 4bpp
320x240 Resolution
37.5K byte per frame



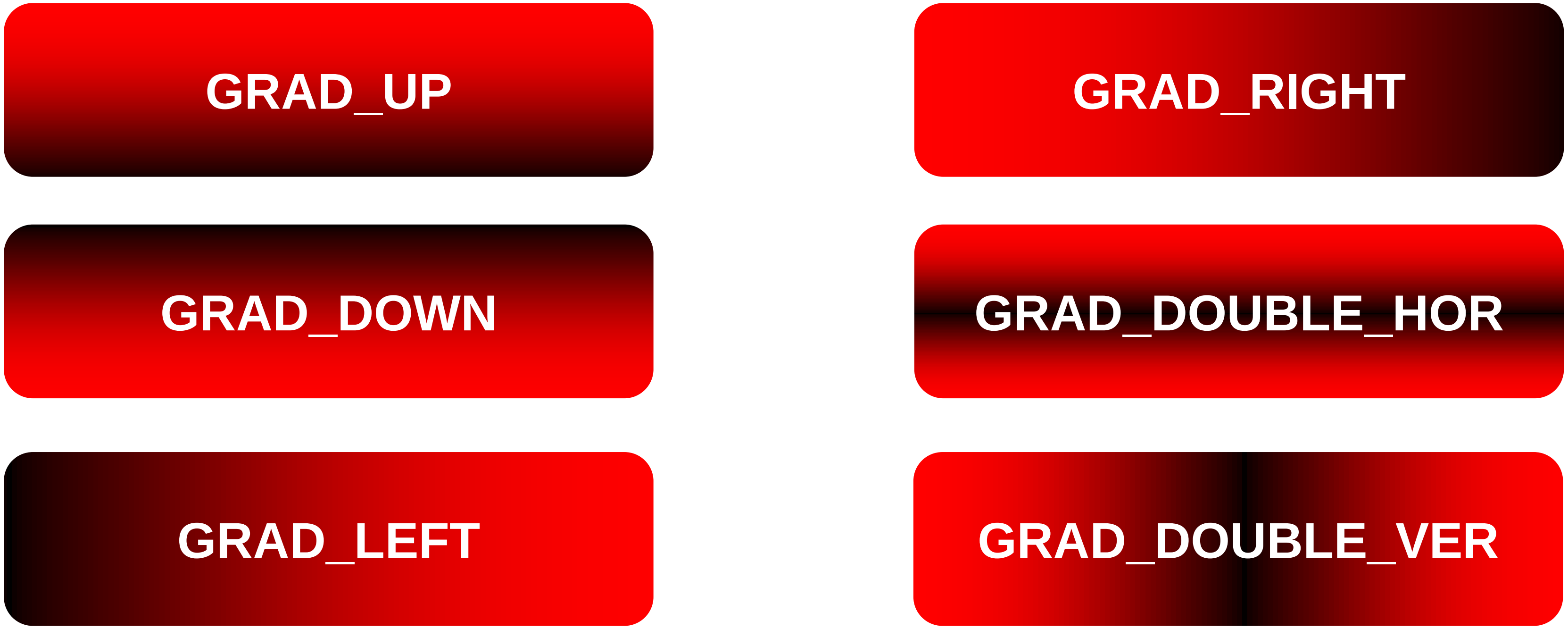
256 Colors – 8bpp
320x240 Resolution
75K byte per frame



256 Colors – 8bpp
320x240 Resolution
75K byte per frame

Gradients

- Smooth transition from one color to another
- Implemented in the primitive layer of the library
- Vertical or horizontal are supported



GRAD_UP

GRAD_RIGHT

GRAD_DOWN

GRAD_DOUBLE_HOR

GRAD_LEFT

GRAD_DOUBLE_VER

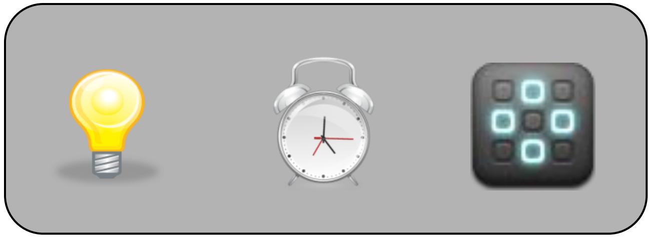
Graphics Theory

- Colors
- **Images**
- Transparency
- Fonts

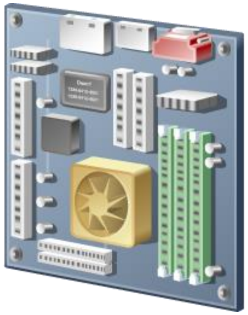
How are images used?

- Company Logos

- Icons



- Indicators



How are images used?

- **Create screen backgrounds**

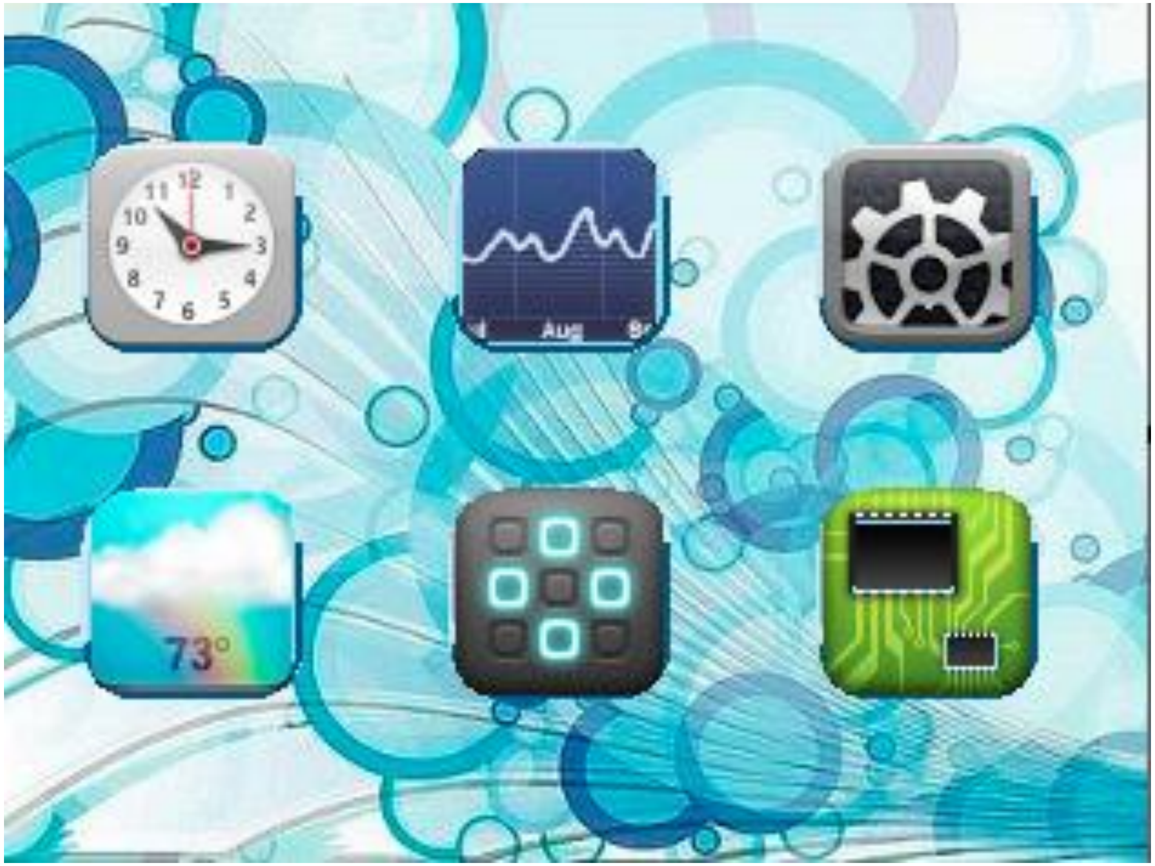


Image Support

- MPLAB Harmony Graphics Library supports both BMP and JPEG image formats
  - Color depth up to 24 BPP
- Images are converted using MPLAB Harmony Graphics Configurator
  - May be compressed using run length encoding (RLE)

Design Considerations

Image size color depth affects memory needs.

Image Size: 59x60 pixels
Color Depth: 16 BPP

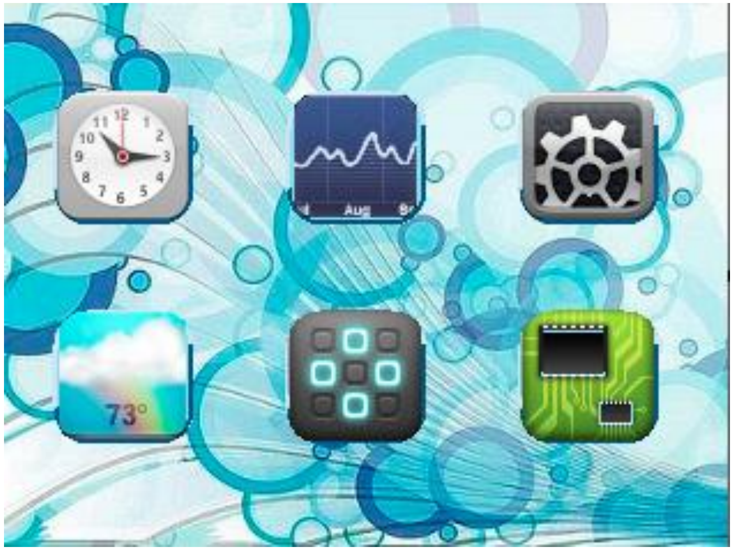
Requires:
(59x60x16)/8 = 7K bytes

Image Size: 59x60 pixels
Color Depth: 32 BPP

Requires:
(59x60x32)/8 = 14K bytes

Design Considerations

Guess how much memory needed for this screen?



Resolution: 320x240
Bits per Pixel: 16

Frame Buffer Count: 1
Images: 1 full screen
6 60x60 icons

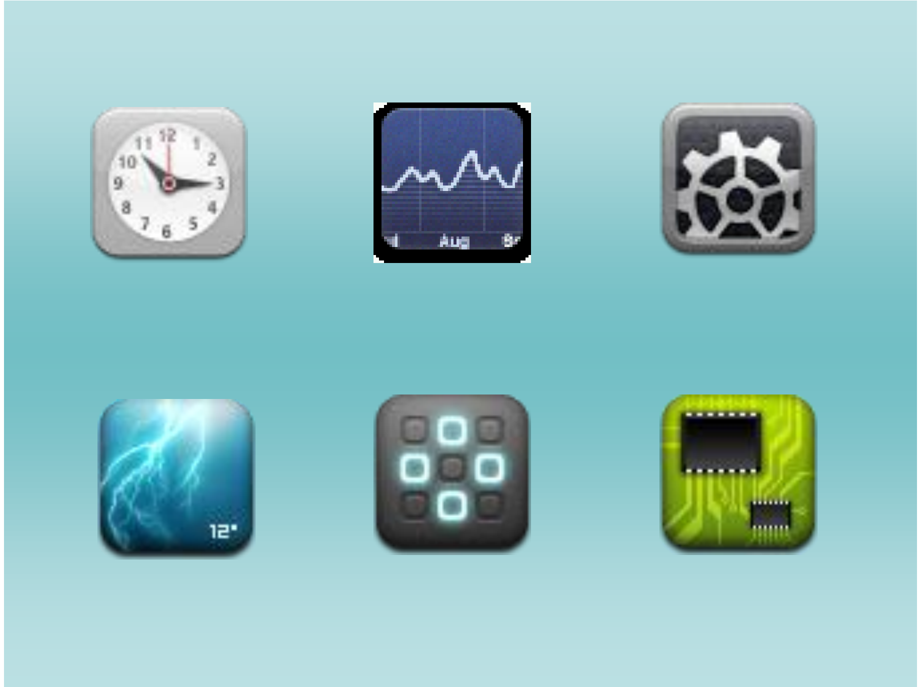
Pixel Count = 320 * 240 = 76800
Bits per Pixel: 16

Frame Buffer Size: (((76800 * 16) / 8) / 1024) = 150K
Full Screen Image: (((76800 * 16) / 8) / 1024) = 150K
Icons = ((((60 * 60) * 16) / 8) / 1024) * 6) = 42K

ANSWER: As drawn, this screen requires ~342 KBytes

Memory Savings Tips

- Gradient Background
  - Doesn’t consume extra memory space
  - Consumes extra CPU cycles to draw.

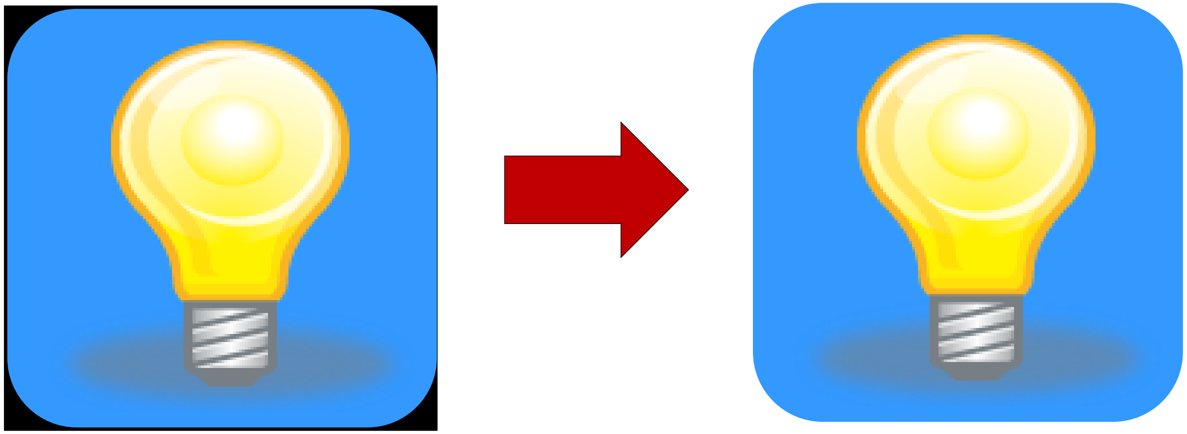


Graphics Theory

- Colors
- Images
- **Transparency**
- Fonts

Transparency

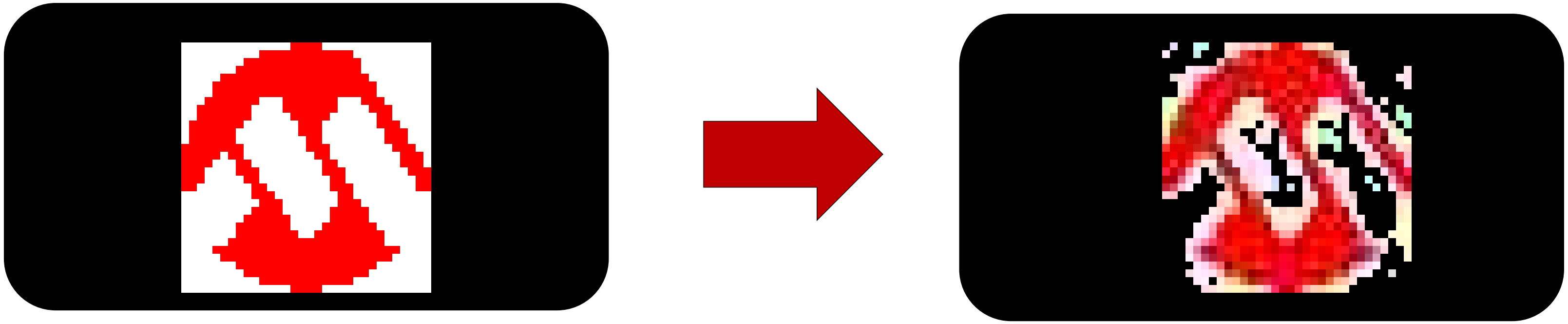
- **Masks colors from being drawn**
  - Hardware accelerated (if available)
  - Primitive Layer Software Implementation
- **Commonly used to hide icon edges**



Design Consideration

- **Masked colors must be an EXACT match**
- **Image compression may introduce masking artifacts**

- **Example:**
  - Attempting to mask white background.



Alpha Blending

- Graphical technique which blends pixels together instead of overwriting one with another.

Example: Blend two pixels together using a 50% blend technique

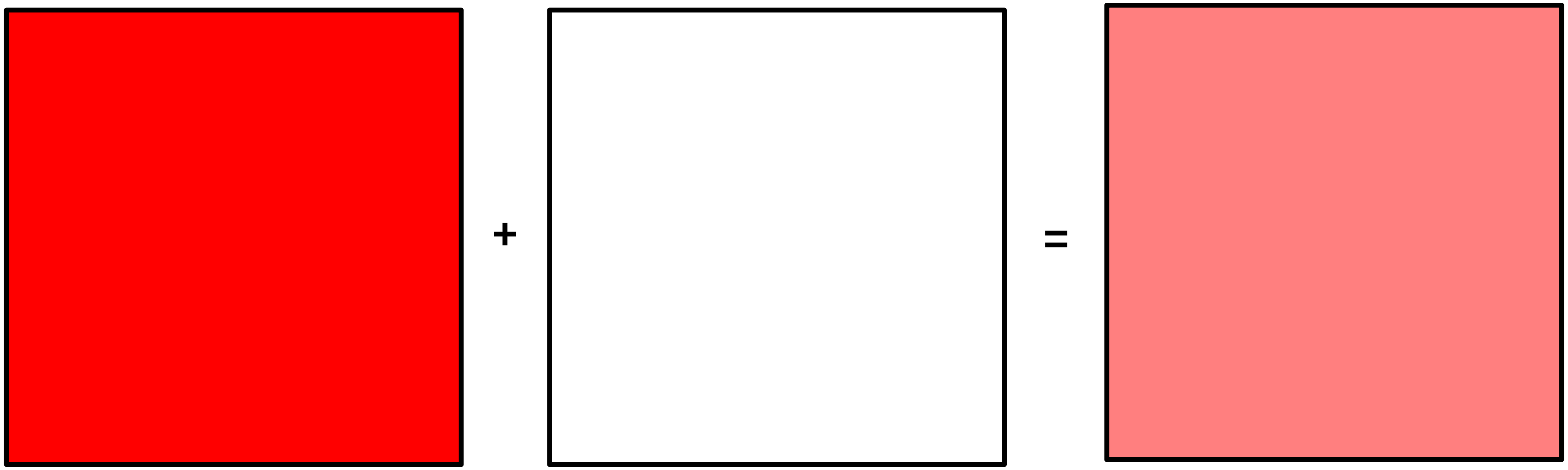


Image Blending

Alpha blending can be used to transition from one image to another.

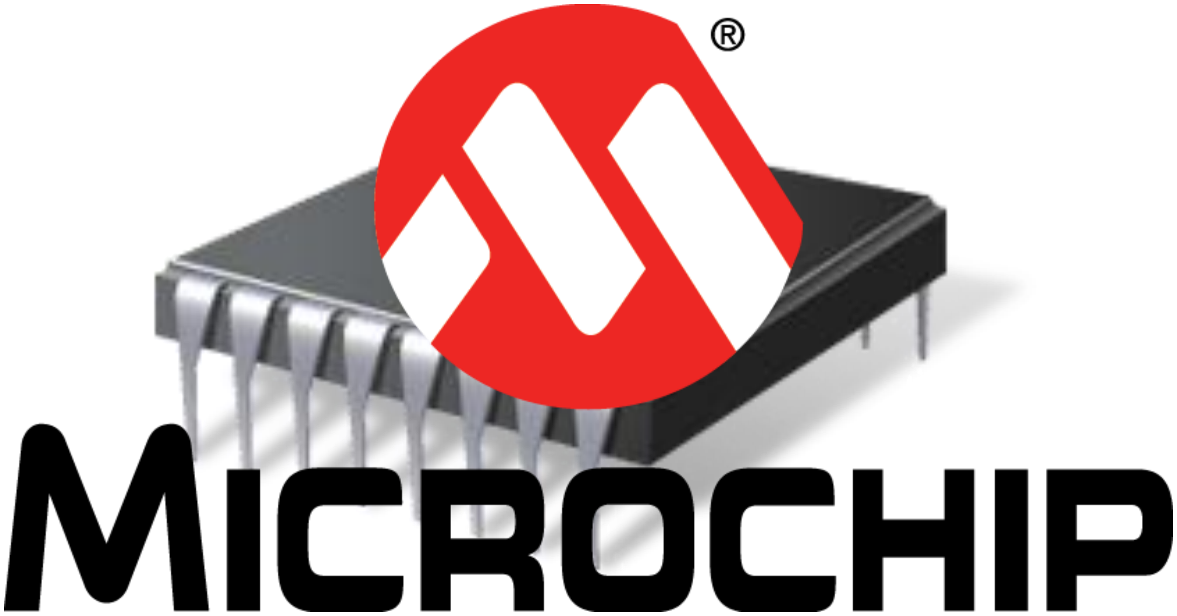
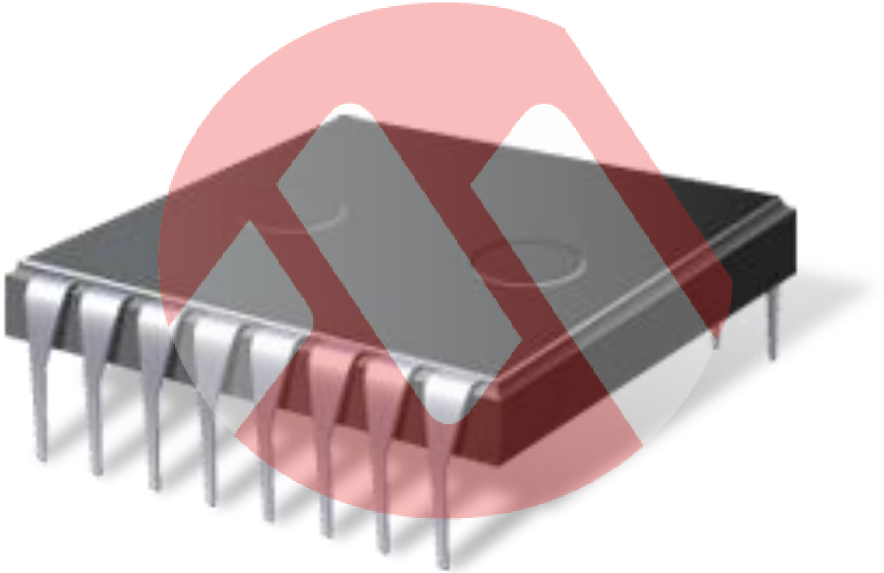


Image Watermarking

Alpha blending can be used to overlay one image onto another.



Graphics Theory

- Colors
- Images
- Transparency
- **Fonts**

Fonts

Definition:

<u>Fonts</u> are electronic data files containing a set of glyphs, characters, and symbols. Fonts are created with font editors and are often considered works of art. Precreated fonts are available from many sources, but may be licensed. Often times they are copyrighted. Please read all licensing agreements before use.

Font Terminology

- **Body**
  - Imaginary area that encompasses each glyph in a font

Font body impacts spacing

Font body impacts spacing

Font body impacts spacing

Font body impacts spacing

- **Point Size**
  - The height of the glyph body

24 pt Arial

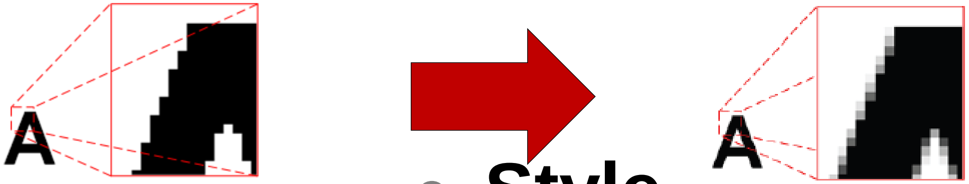
24 pt Calibri

24 pt Cochin

24 pt Gabriola

Font Terminology

- **Anti-Aliasing**
  - Blurring the edges to soften the look
  - Desirable for larger point sizes



- **Style**
  - *Italics*, **Bold,** Anti-Aliased**,** etc

- **Extended Glyphs**
  - Used render languages that use more than one byte to represent a single त + े = ते

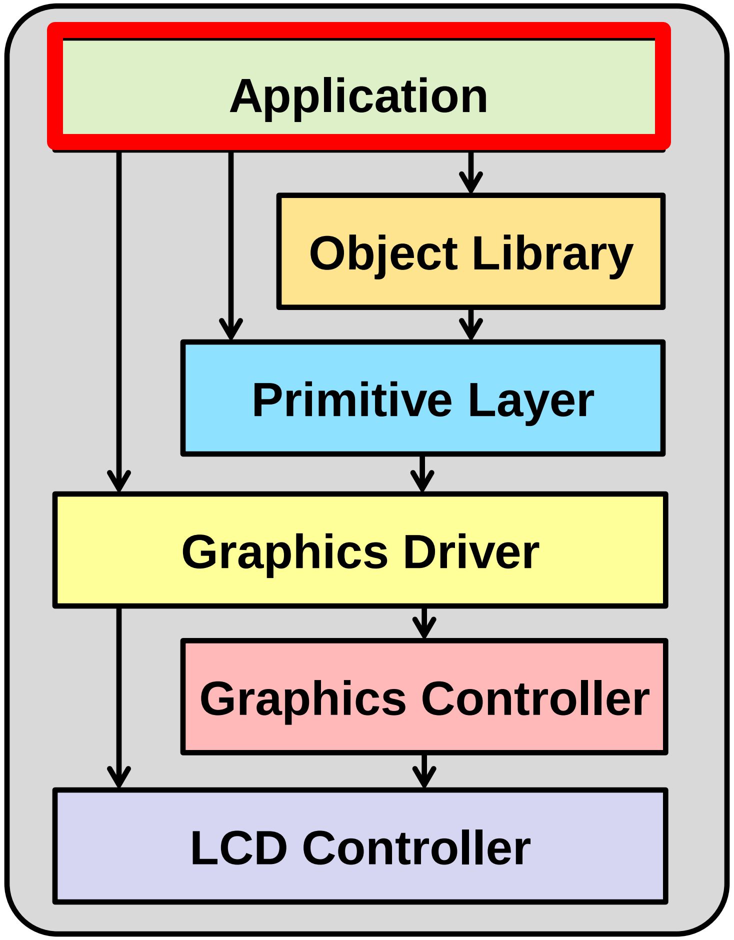
How to get fonts

- Purchase fonts
  - Thousands of sources available online
  - Read license terms!
- Many websites offer “free” fonts
  - Look for open source fonts (http://www.openfontlibrary.org)
  - Download Google Web Fonts (http://code.google.com/p/googlefontdirectory)
- Create your own fonts using a font editor
  - Fony – freeware
  - FontForge – freeware
  - Other editors range in price
- Convert from Open Type
  - Results vary depending on the converter

- **Questions about:**
  # Graphics Theory?

MPLAB Harmony Graphics Library

MPLAB Harmony Graphics Stack Overview

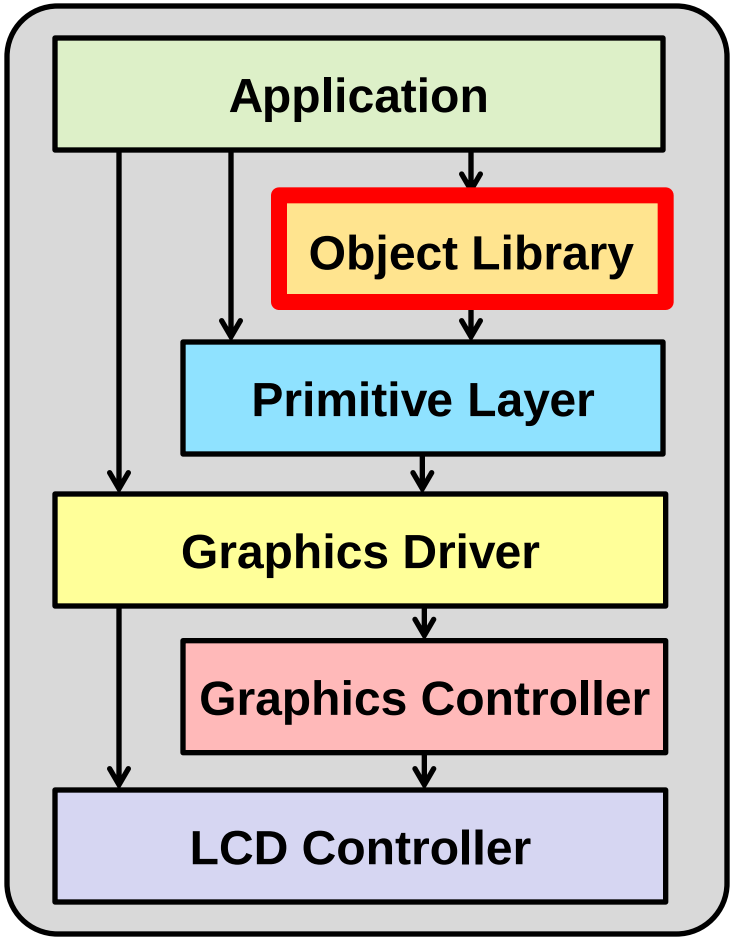


Application Layer

- User Code Location
- Interfaces with:
  - Object Library
  - Primitive Layer
  - Graphics Driver

MASTERs 2015

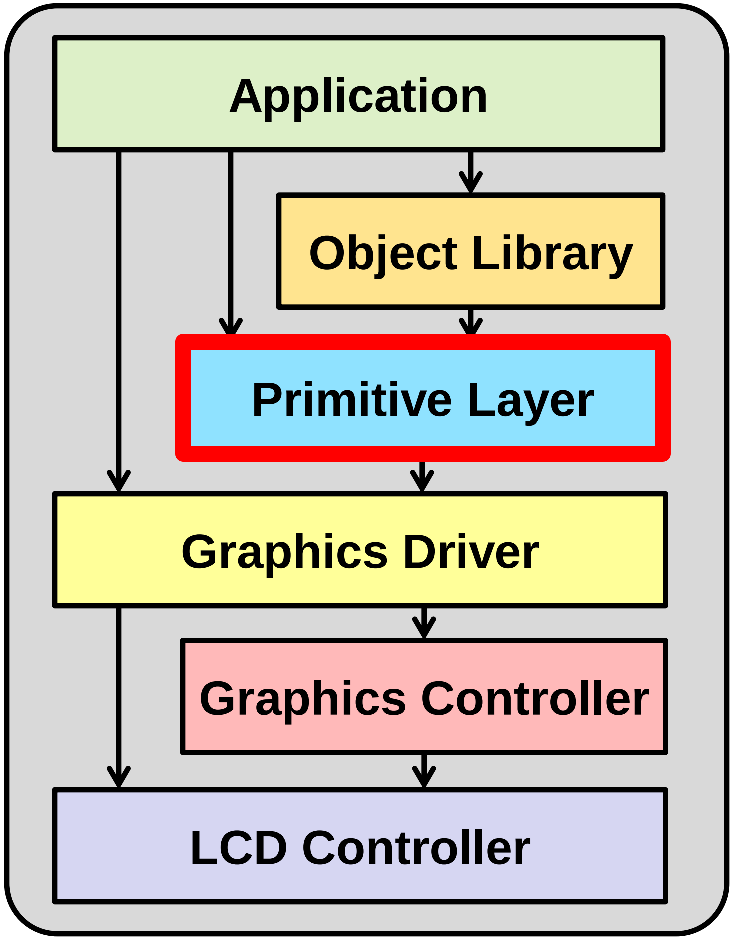
MPLAB Harmony Graphics Stack Overview



Object Layer

- Graphical Widget Library
- Provides buttons, check boxes, sliders, text boxes, radio buttons, etc.

MPLAB Harmony Graphics Stack Overview

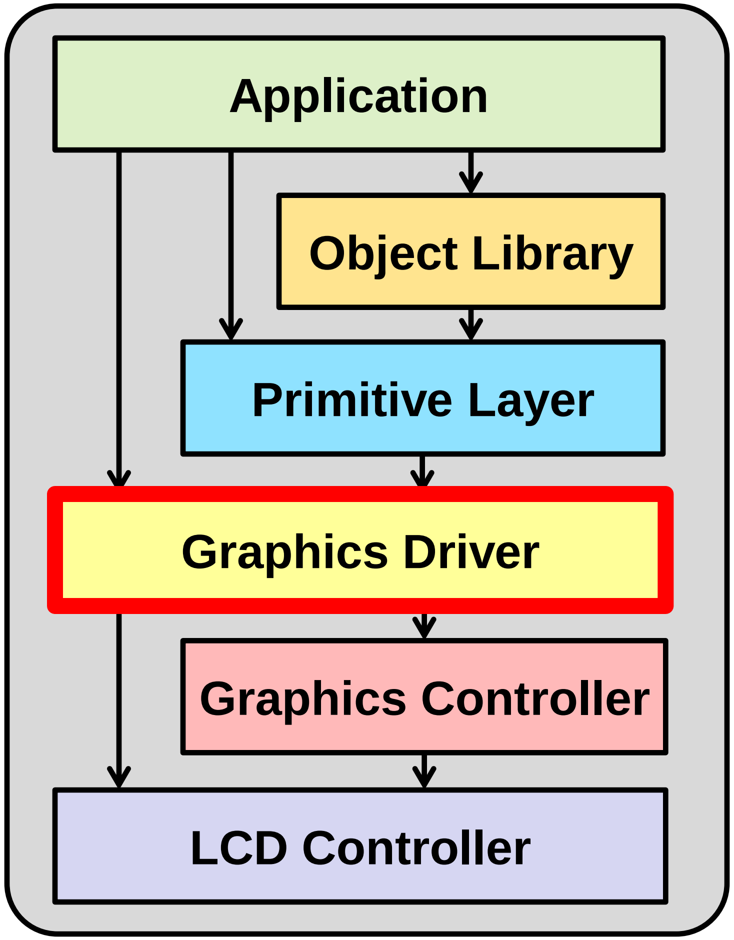


Primitive Layer

- Basic shape rendering
- Draws lines, rectangles, circles, text, images, polygons, etc.

MASTERs 2015

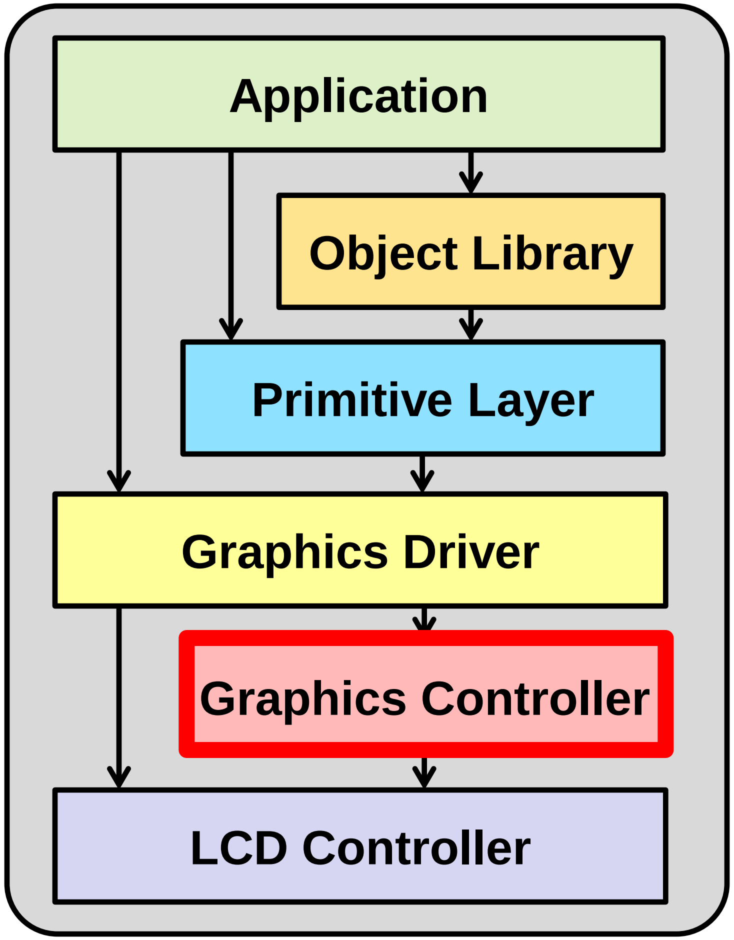
MPLAB Harmony Graphics Stack Overview



Graphics Driver

- Hardware abstraction layer.

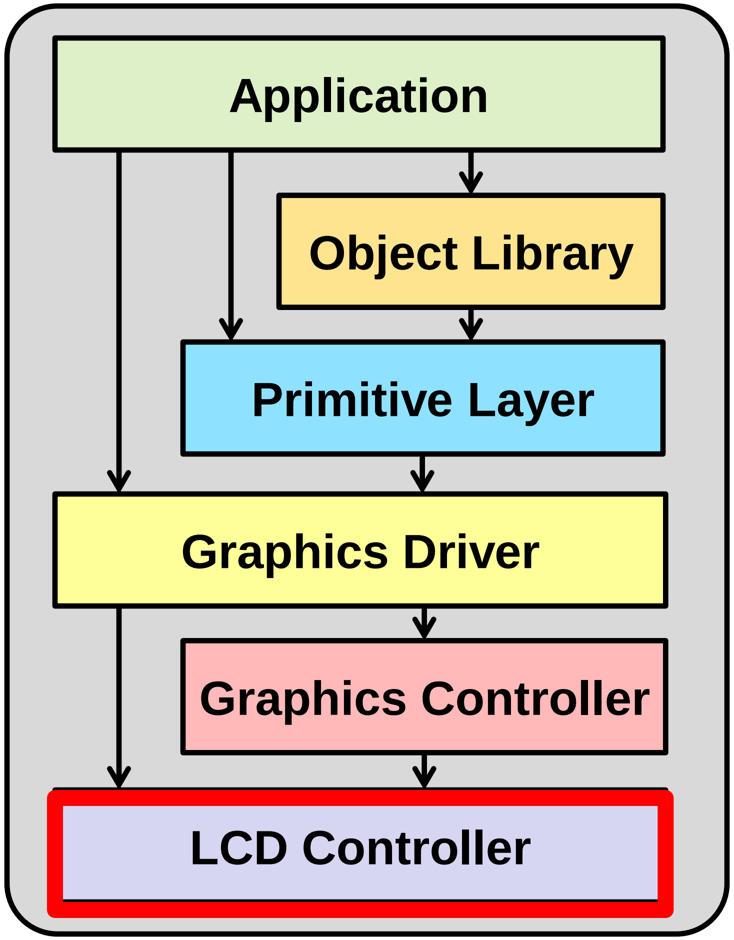
MPLAB Harmony Graphics Stack Overview



Graphics Controller

- Discreet hardware for managing LCD controller.
- May provide hardware graphics acceleration or frame buffer management.

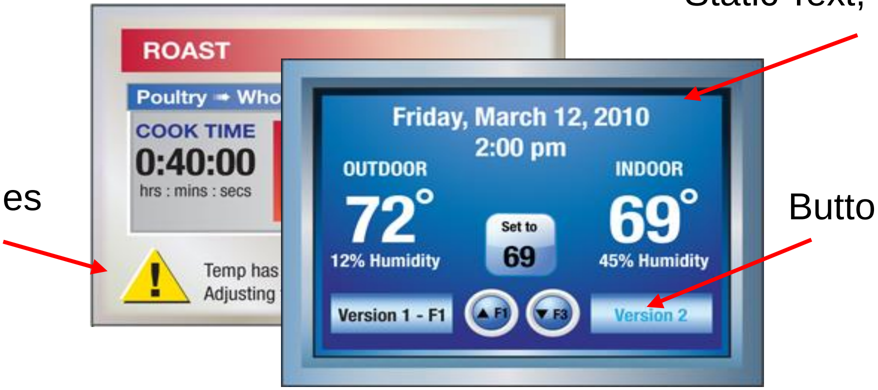
MPLAB Harmony Graphics Stack Overview



LCD Controller

- Display Hardware

MPLAB Harmony Graphics Library



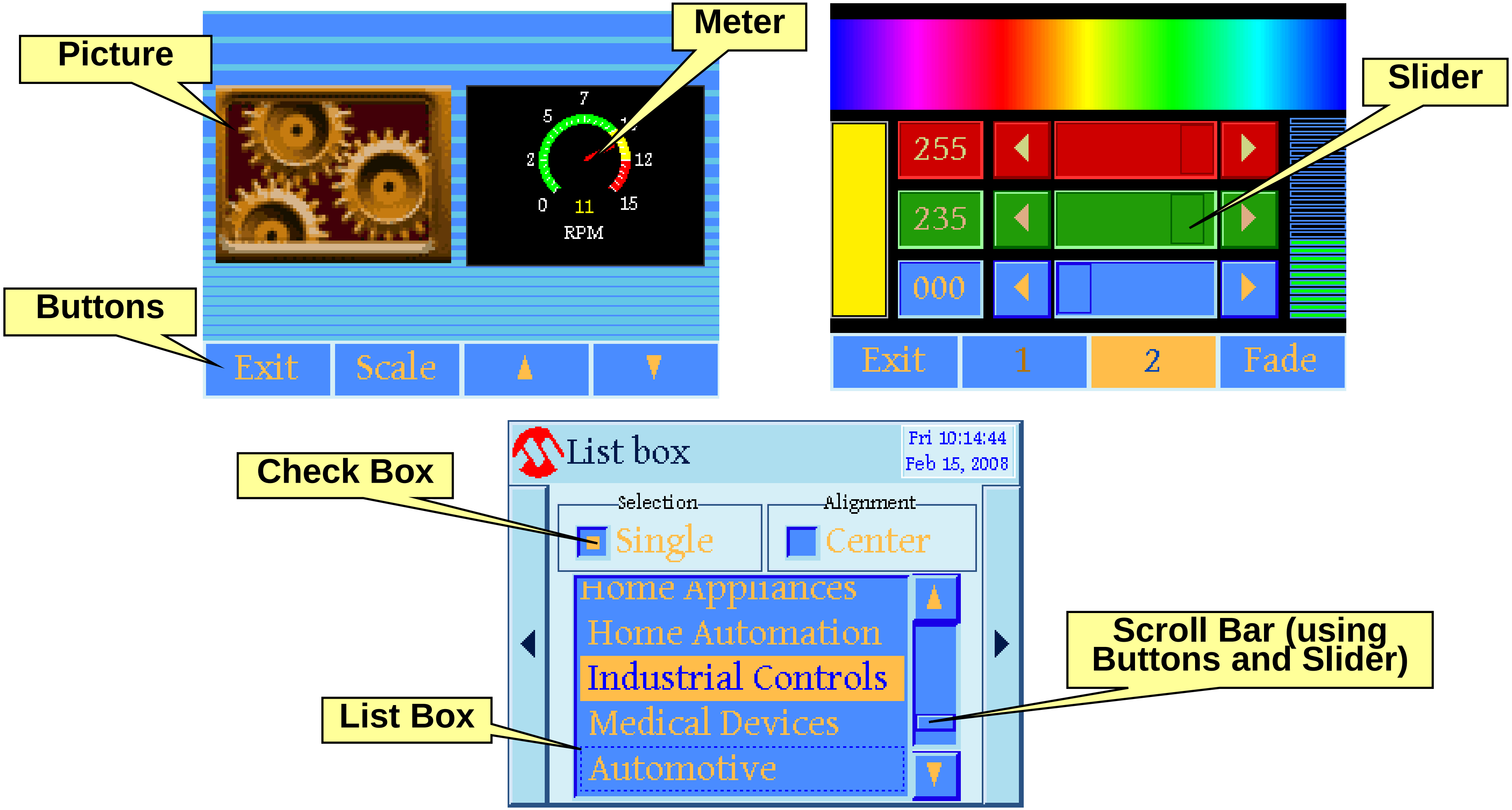
Static Text, Multiple Fonts

Images

Button

- Modular design – compile only what you need!
- Supports up to 24 bpp color depth
- Supports gradients, transparency, and alpha blending
- Provides color schemes for easy look-and-feel configuration
- Demo source provided for our low-cost, full featured development tools
- Free to Microchip customers
  - Source code included
  - Multiple display controller drivers included

Library Widgets



Picture

Meter

Buttons

Slider

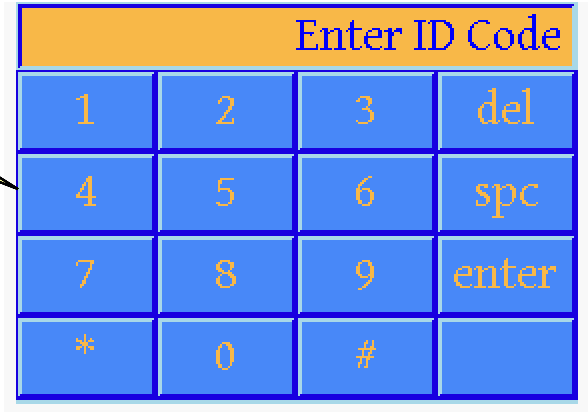
Check Box

List Box

Scroll Bar (using Buttons and Slider)

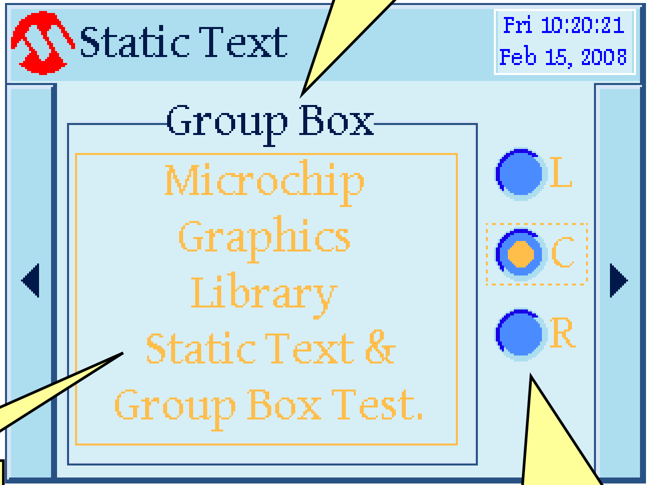
Library Widgets

Text Entry



Enter ID Code

Group Box



Static Text

Fri 10:20:21
Feb 15, 2008

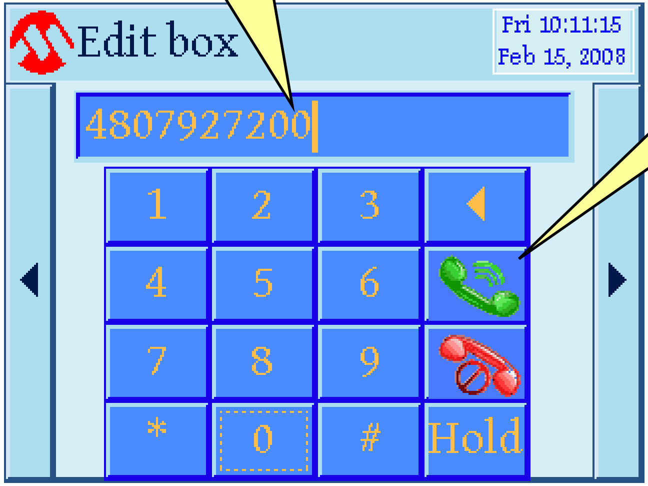
Group Box

Microchip Graphics Library Static Text & Group Box Test.

Static Text

Radio Buttons

Edit Box

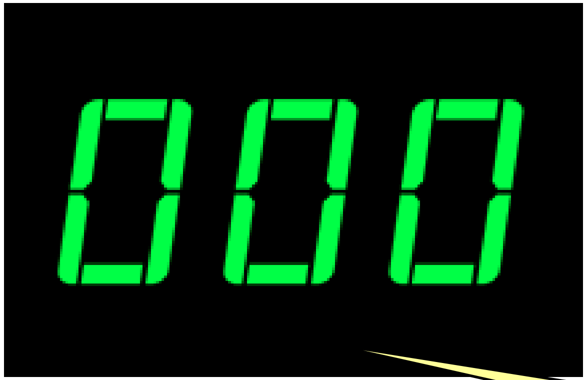


Edit box

Fri 10:11:15
Feb 15, 2008

4807927200

Button with Image



Digital Meter

Primitive Layer

- **Software based or accelerated drawing**
  - Bars, circles, rectangles, lines, arcs (bevel), etc.
  - Image rendering
  - Text rendering (font support)
- **Additional features**
  - Alpha Blending (controller dependent)
  - Transparency
  - Gradients

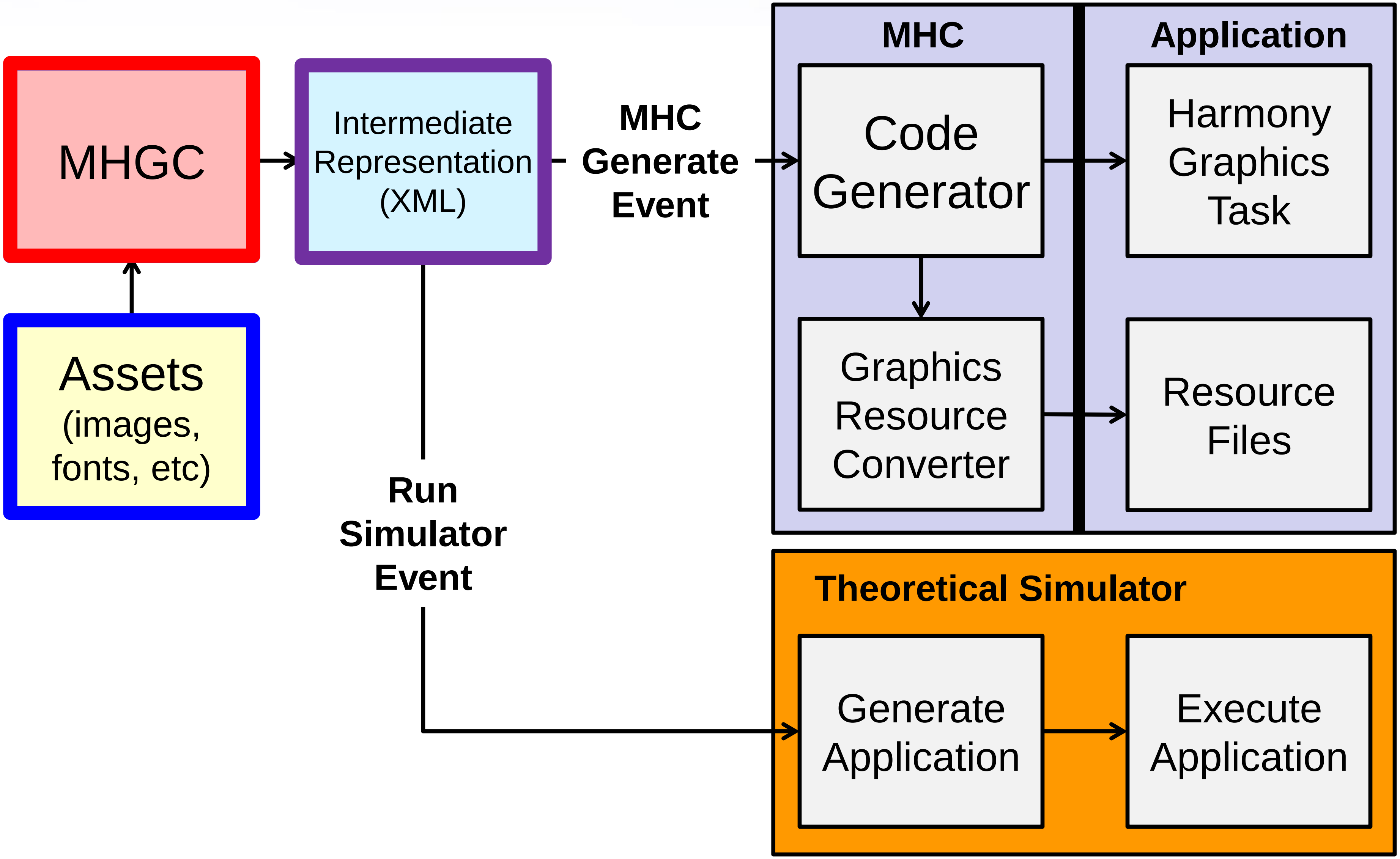
- **Questions about:**
  MPLAB Harmony Graphics Library?

MPLAB Harmony Graphics Composer

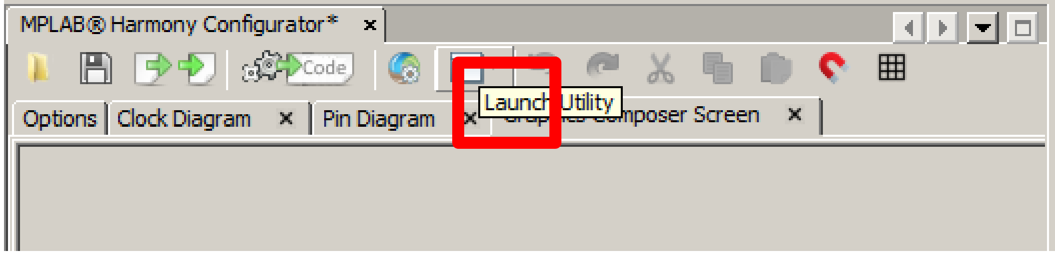
What is MPLAB Harmony Graphics Composer?

- Visual Design Tool
- Fully Integrated with MHC
- Supports MPLAB Harmony Graphics Library
- Graphics Library Event Management
- Template-based Code Generation
- Pipeline-based Code Generation Path

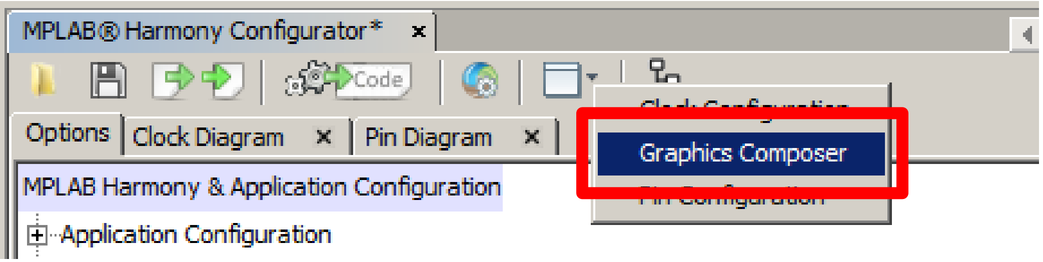
Graphics Composer Pipeline



Launching Graphics Composer

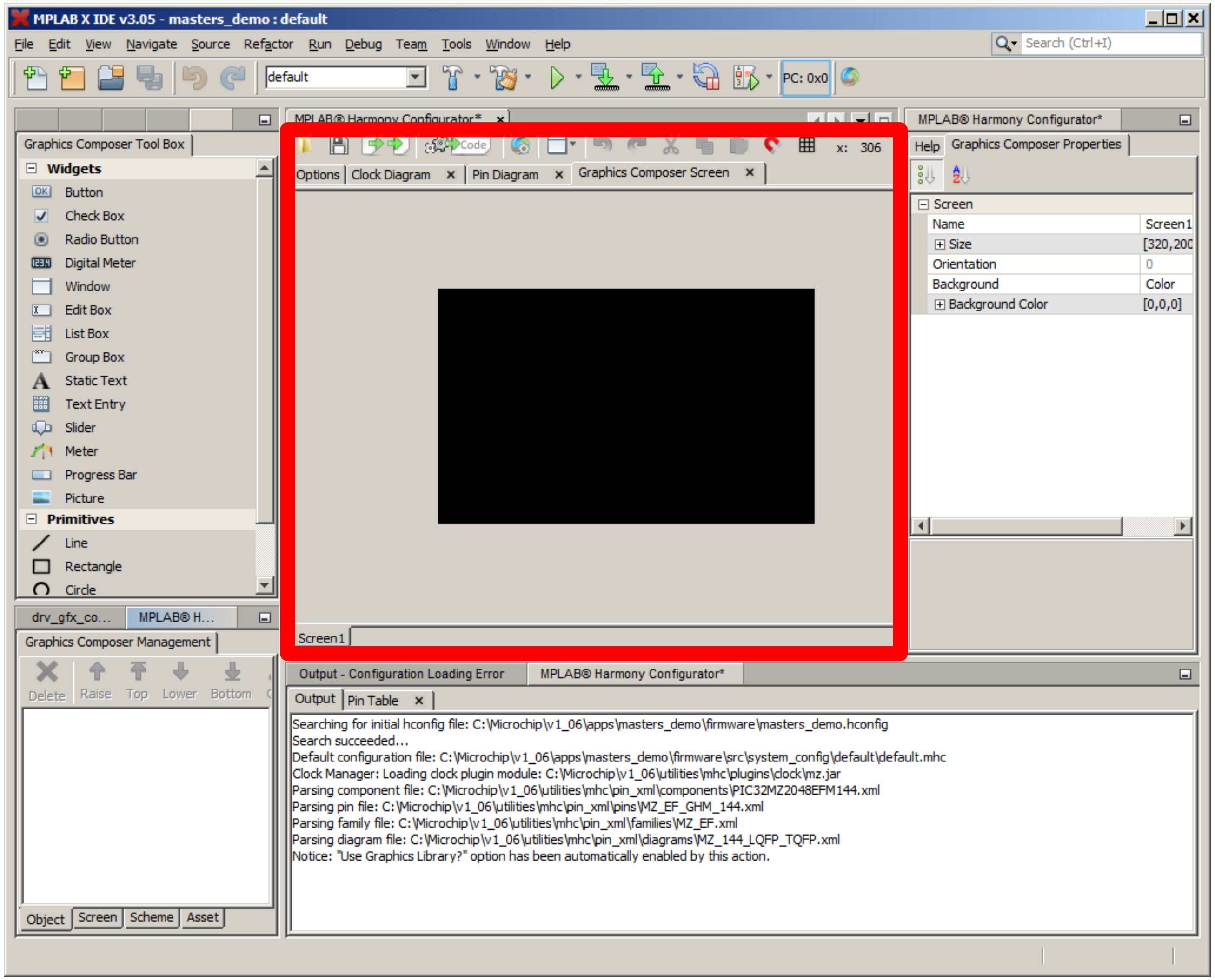


1. Click "Launch Utility" toolbar item.


MPLAB® Harmony Configurator*
Code
Options
Clock Diagram
Pin Diagram
Clock Configuration
Graphics Composer
Pin Configuration
MPLAB Harmony & Application Configuration
Application Configuration

2. Click "Graphics Composer" from the drop-down menu.

Graphics Composer User Interface



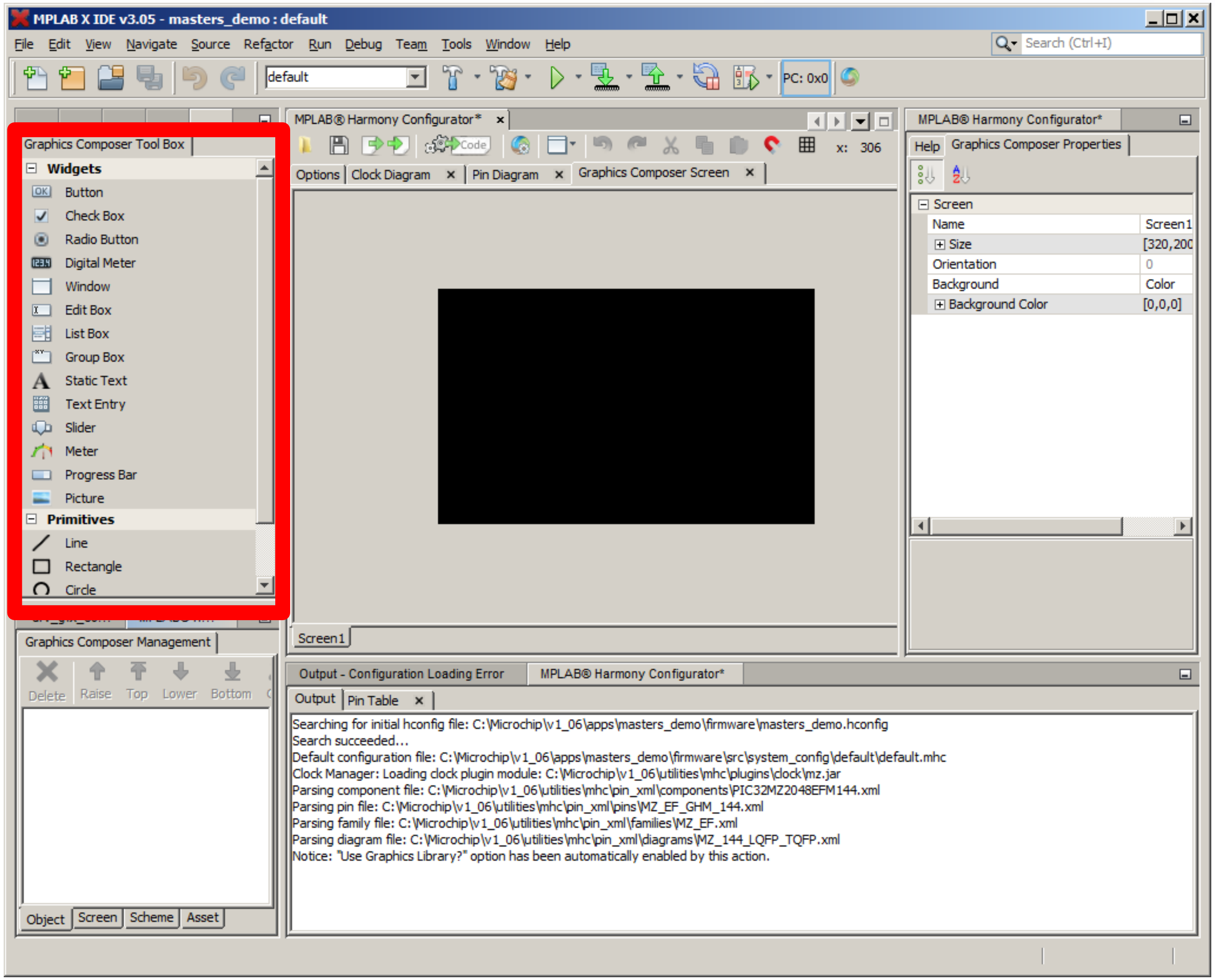
Screen	
Name	Screen1
⊞ Size	[320,200
Orientation	0
Background	Color
⊞ Background Color	[0,0,0]

```
Searching for initial hconfig file: C:\Microchip\v1_06\apps\masters_demo\firmware\masters_demo.hconfig
Search succeeded...
Default configuration file: C:\Microchip\v1_06\apps\masters_demo\firmware\src\system_config\default\default.mhc
Clock Manager: Loading clock plugin module: C:\Microchip\v1_06\utilities\mhc\plugins\clock\mz.jar
Parsing component file: C:\Microchip\v1_06\utilities\mhc\pin_xml\components\PIC32MZ2048EFM144.xml
Parsing pin file: C:\Microchip\v1_06\utilities\mhc\pin_xml\pins\MZ_EF_GHM_144.xml
Parsing family file: C:\Microchip\v1_06\utilities\mhc\pin_xml\families\MZ_EF.xml
Parsing diagram file: C:\Microchip\v1_06\utilities\mhc\pin_xml\diagrams\MZ_144_LQFP_TQFP.xml
Notice: "Use Graphics Library?" option has been automatically enabled by this action.
```

Screen Tab

- Visual representation of the device screen

Graphics Composer User Interface



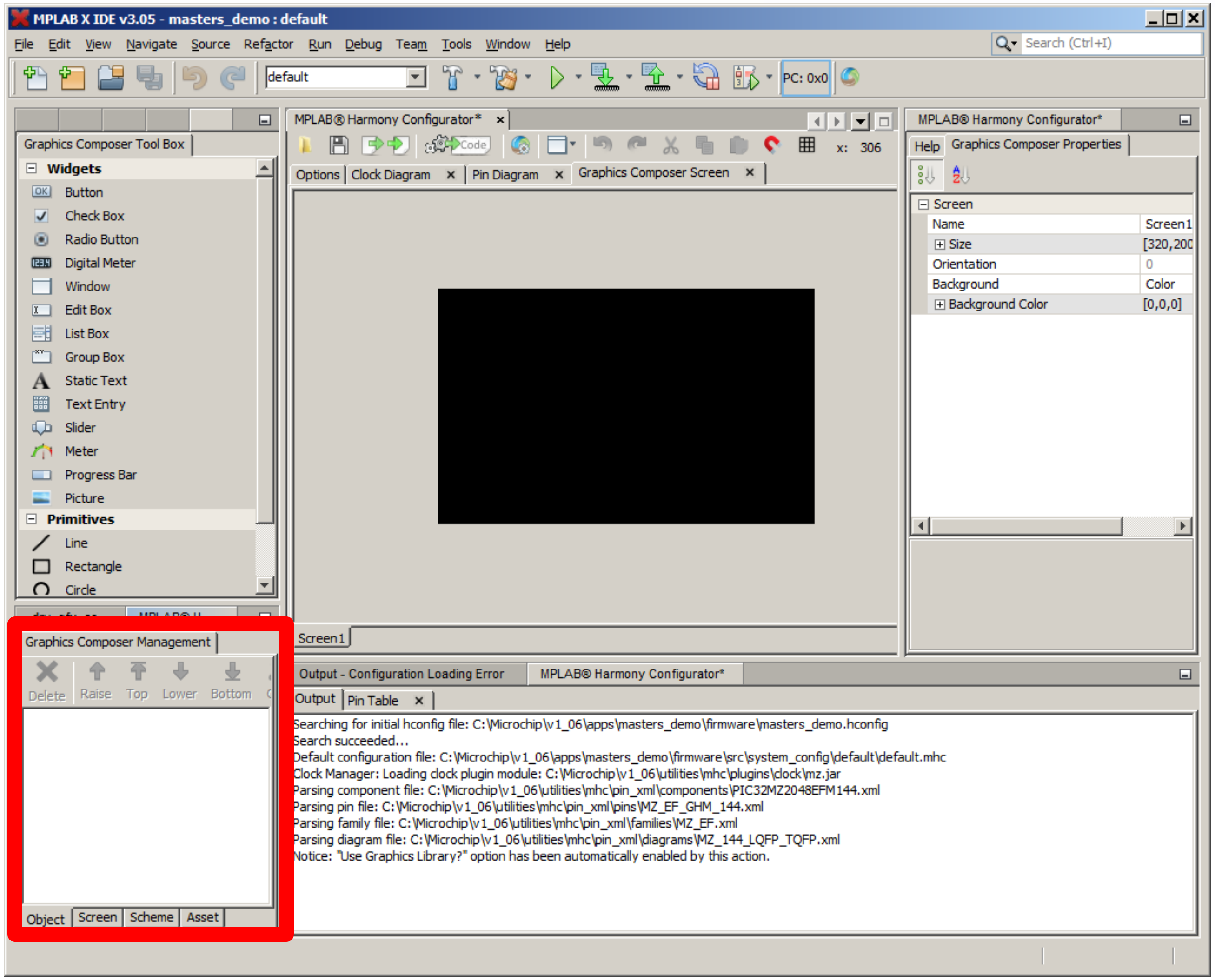
Screen	
Name	Screen1
Size	[320,200
Orientation	0
Background	Color
Background Color	[0,0,0]

```
Searching for initial hconfig file: C:\Microchip\v1_06\apps\masters_demo\firmware\masters_demo.hconfig
Search succeeded...
Default configuration file: C:\Microchip\v1_06\apps\masters_demo\firmware\src\system_config\default\default.mhc
Clock Manager: Loading clock plugin module: C:\Microchip\v1_06\utilities\mhc\plugins\clock\mz.jar
Parsing component file: C:\Microchip\v1_06\utilities\mhc\pin_xml\components\PIC32MZ2048EFM144.xml
Parsing pin file: C:\Microchip\v1_06\utilities\mhc\pin_xml\pins\MZ_EF_GHM_144.xml
Parsing family file: C:\Microchip\v1_06\utilities\mhc\pin_xml\families\MZ_EF.xml
Parsing diagram file: C:\Microchip\v1_06\utilities\mhc\pin_xml\diagrams\MZ_144_LQFP_TQFP.xml
Notice: "Use Graphics Library?" option has been automatically enabled by this action.
```

Tool Box Tab

- Lists widgets and primitives available for use

Graphics Composer User Interface



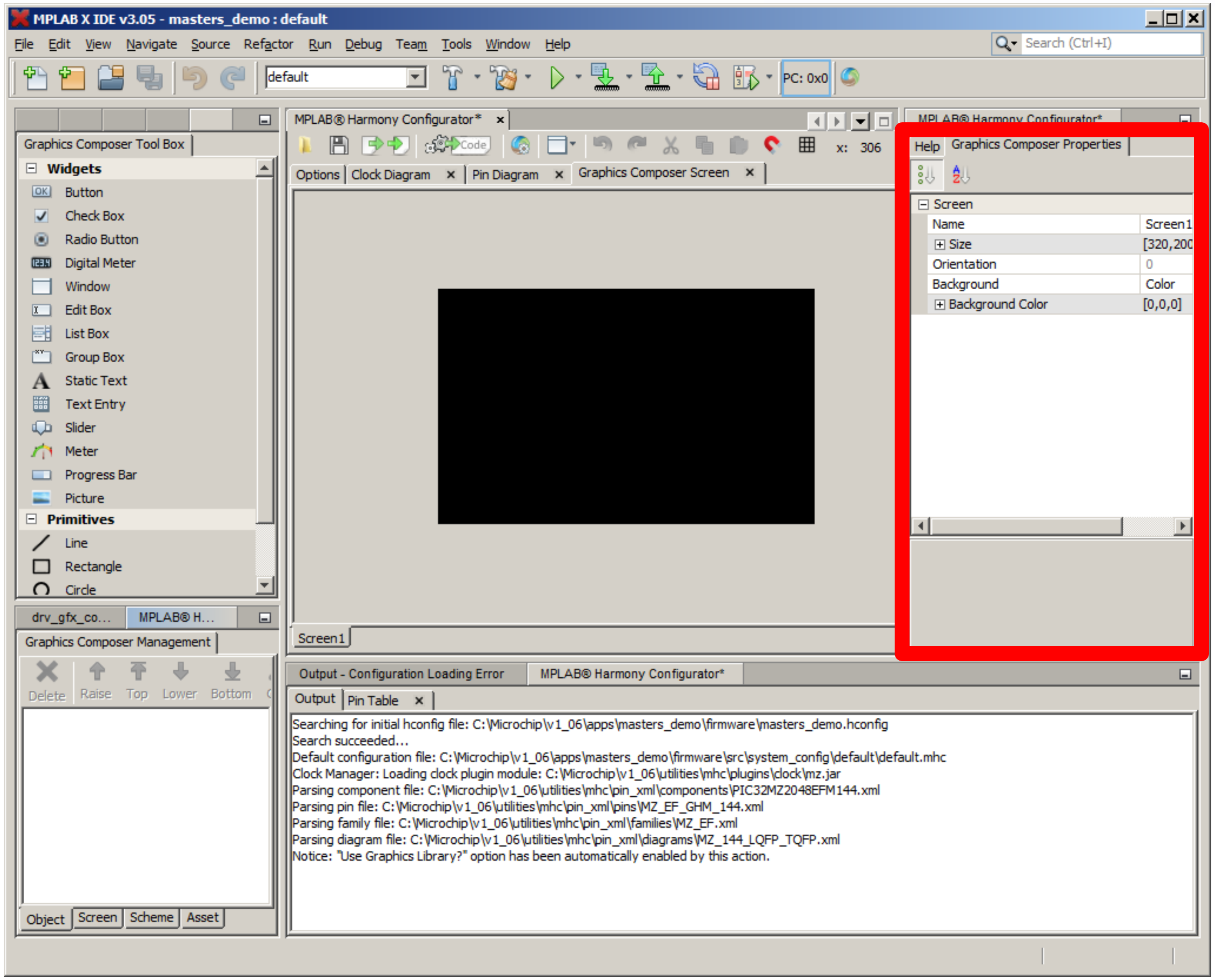
Screen	
Name	Screen1
⊞ Size	[320,200
Orientation	0
Background	Color
⊞ Background Color	[0,0,0]

```
Searching for initial hconfig file: C:\Microchip\v1_06\apps\masters_demo\firmware\masters_demo.hconfig
Search succeeded...
Default configuration file: C:\Microchip\v1_06\apps\masters_demo\firmware\src\system_config\default\default.mhc
Clock Manager: Loading clock plugin module: C:\Microchip\v1_06\utilities\mhc\plugins\clock\mz.jar
Parsing component file: C:\Microchip\v1_06\utilities\mhc\pin_xml\components\PIC32MZ2048EFM144.xml
Parsing pin file: C:\Microchip\v1_06\utilities\mhc\pin_xml\pins\MZ_EF_GHM_144.xml
Parsing family file: C:\Microchip\v1_06\utilities\mhc\pin_xml\families\MZ_EF.xml
Parsing diagram file: C:\Microchip\v1_06\utilities\mhc\pin_xml\diagrams\MZ_144_LQFP_TQFP.xml
Notice: "Use Graphics Library?" option has been automatically enabled by this action.
```

Management Tab

- Manages objects, screens, schemes, and assets

Graphics Composer User Interface



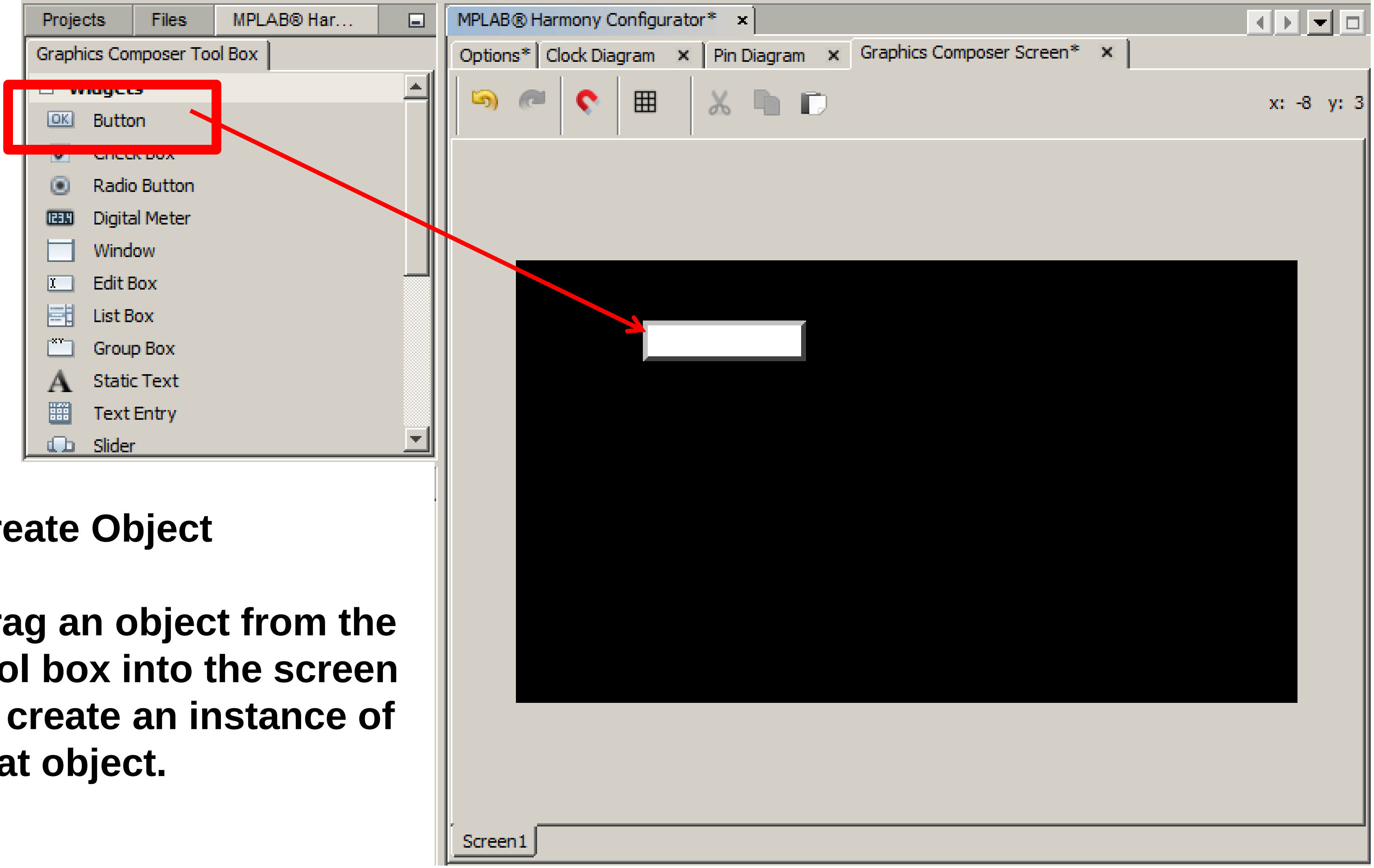
Screen	
Name	Screen1
⊞ Size	[320,200
Orientation	0
Background	Color
⊞ Background Color	[0,0,0]

```
Searching for initial hconfig file: C:\Microchip\v1_06\apps\masters_demo\firmware\masters_demo.hconfig
Search succeeded...
Default configuration file: C:\Microchip\v1_06\apps\masters_demo\firmware\src\system_config\default\default.mhc
Clock Manager: Loading clock plugin module: C:\Microchip\v1_06\utilities\mhc\plugins\clock\mz.jar
Parsing component file: C:\Microchip\v1_06\utilities\mhc\pin_xml\components\PIC32MZ2048EFM144.xml
Parsing pin file: C:\Microchip\v1_06\utilities\mhc\pin_xml\pins\MZ_EF_GHM_144.xml
Parsing family file: C:\Microchip\v1_06\utilities\mhc\pin_xml\families\MZ_EF.xml
Parsing diagram file: C:\Microchip\v1_06\utilities\mhc\pin_xml\diagrams\MZ_144_LQFP_TQFP.xml
Notice: "Use Graphics Library?" option has been automatically enabled by this action.
```

Property Tab

- Manages properties for screens and objects

Graphics Composer Tool Box Tab



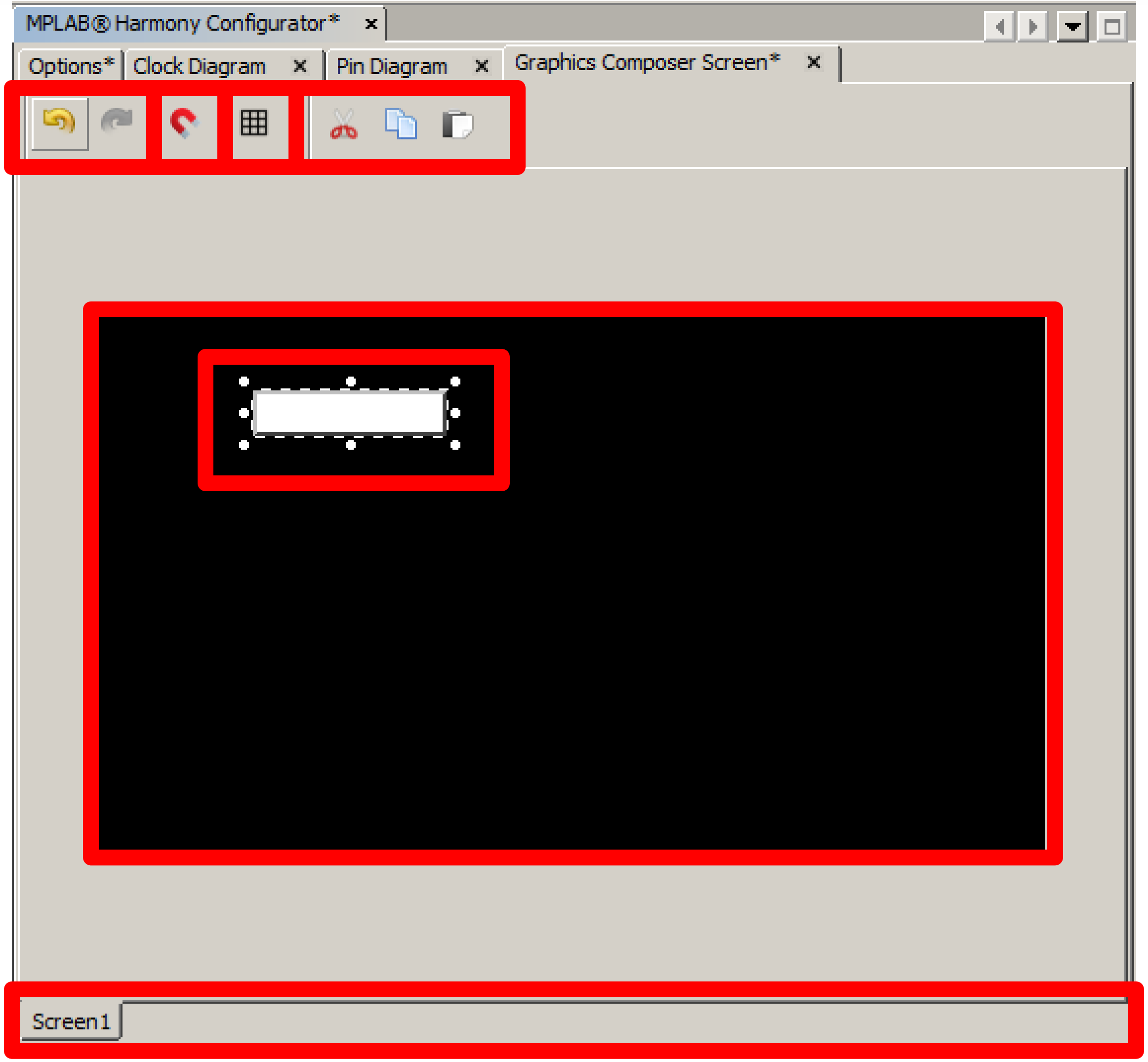
Create Object

Drag an object from the tool box into the screen to create an instance of that object.

Graphics Composer Screen Tab

Screen Tab

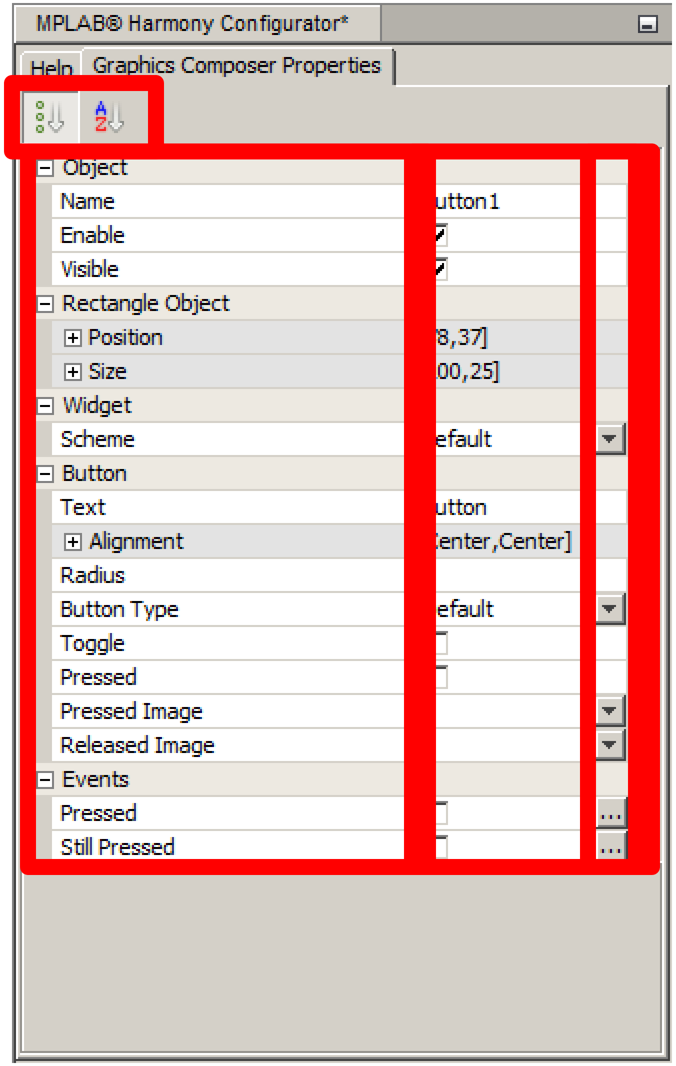
- Screen Area
- Screen Tabs
- Tool Bar
  - Undo/Redo
  - Snapping
  - Grid
  - Cut/Copy/Paste
- Selected Object



Graphics Composer Properties Tab

Properties Tab

- Property Name
- Property Value
- Property Action
- Sort Options



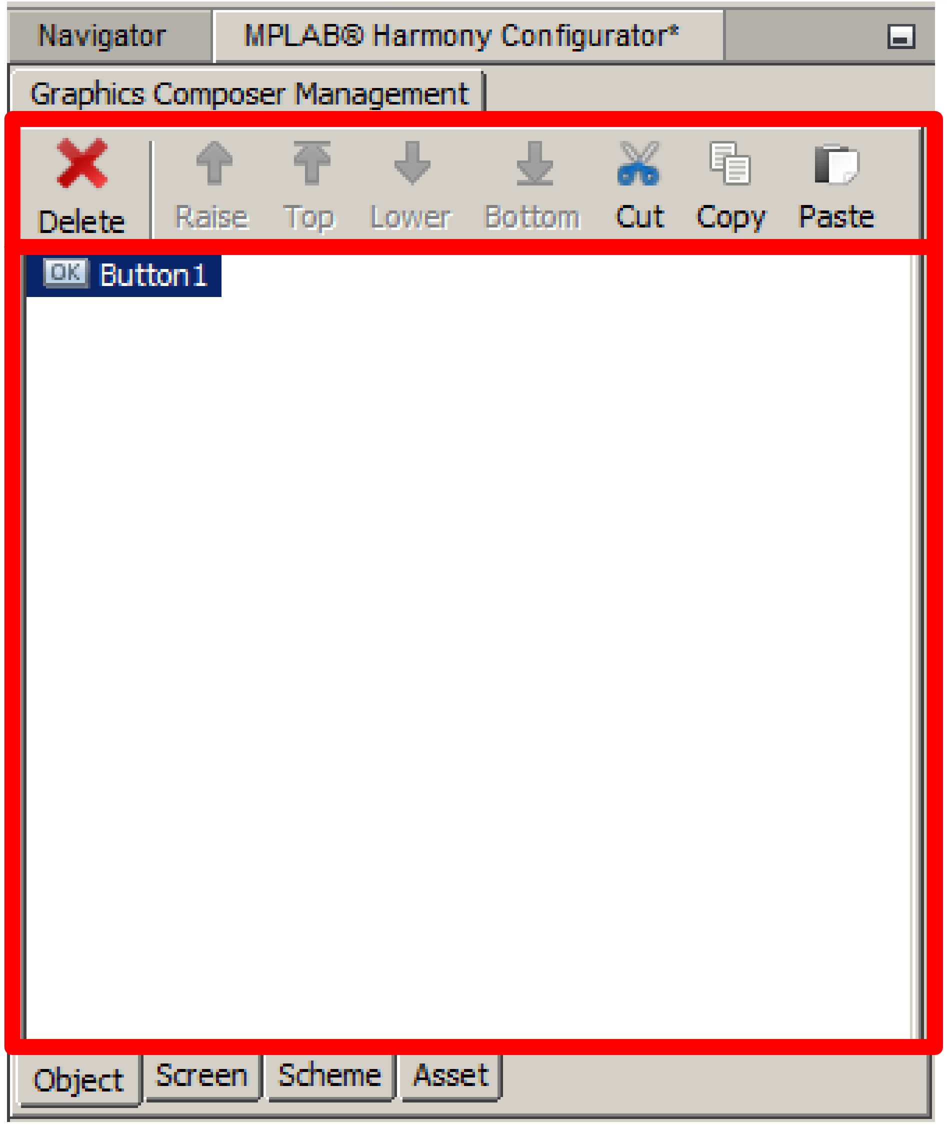
MPLAB® Harmony Configurator*

⊟ Object	
Name	utton1
Enable	✓
Visible	✓
⊟ Rectangle Object	
⊞ Position	8,37]
⊞ Size	00,25]
⊟ Widget	
Scheme	efault ▾
⊟ Button	
Text	utton
⊞ Alignment	enter,Center]
Radius	
Button Type	efault ▾
Toggle	□
Pressed	□
Pressed Image	▾
Released Image	▾
⊟ Events	
Pressed	□ ...
Still Pressed	□ ...

Graphics Composer Object Tab

Object Tab

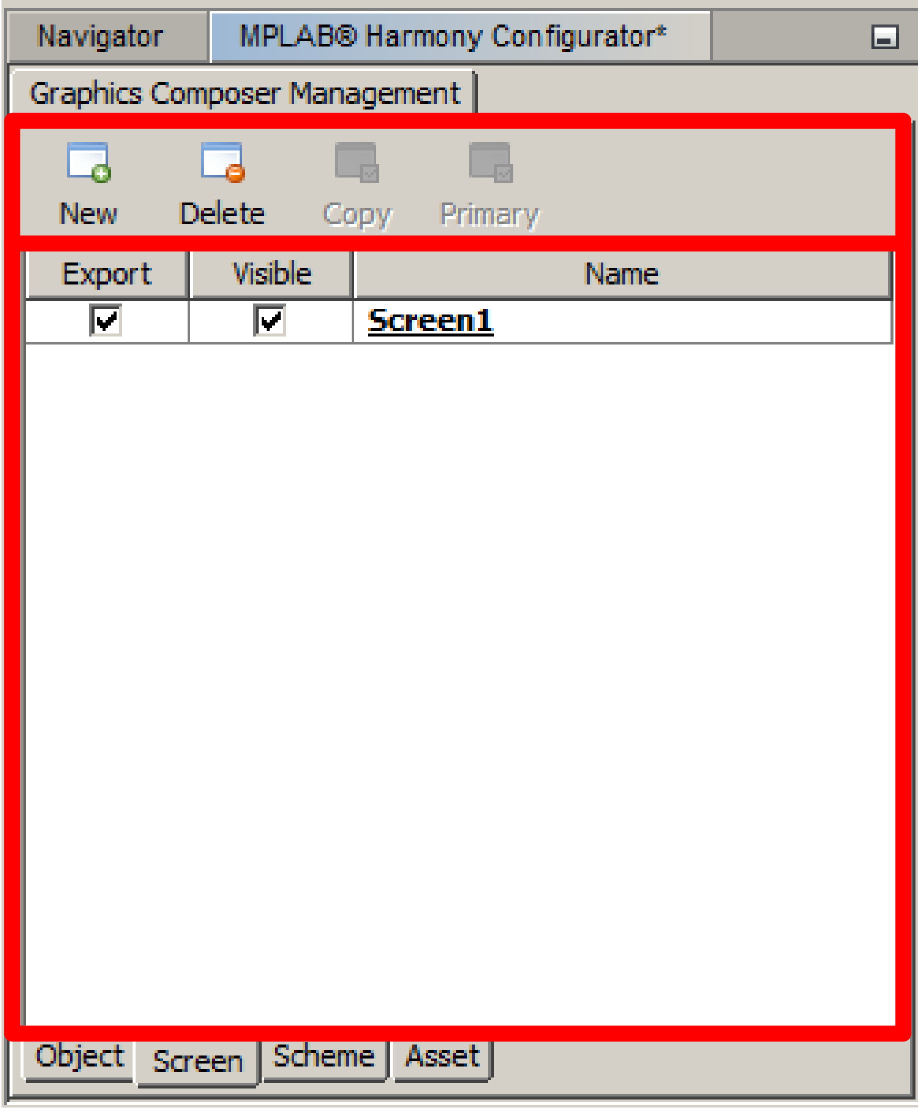
- Object List
- Tool Bar
  - Delete
  - Height Order
  - Cut/Copy/Paste



Graphics Composer Screen Tab

Screen Tab

- Screen List
  - Export
  - Visible
- Tool Bar
  - New
  - Delete
  - Copy
  - Primary



Navigator | MPLAB® Harmony Configurator*

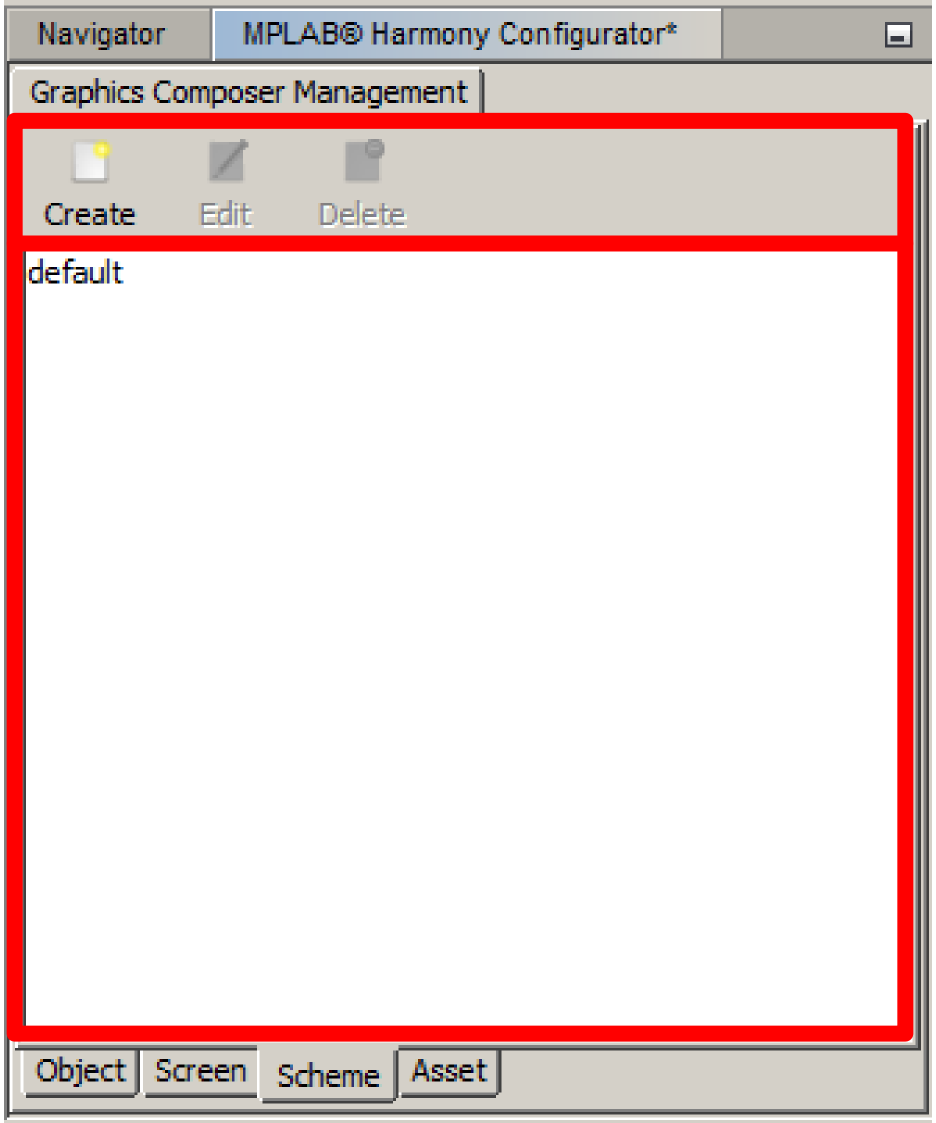
Graphics Composer Management

Export	Visible	Name
☑	☑	Screen1

Graphics Composer Scheme Tab

Scheme Tab

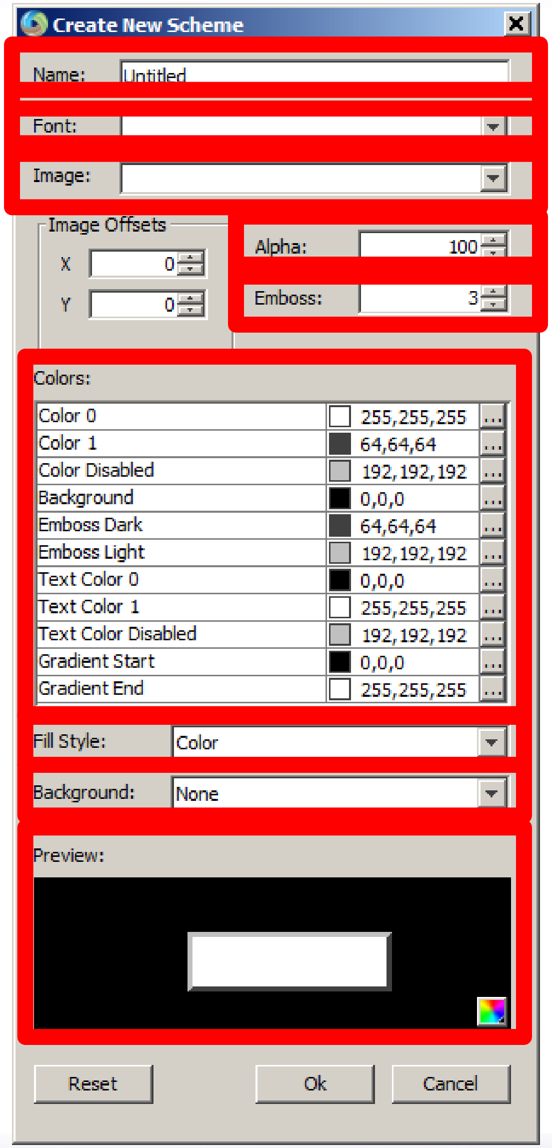
- Scheme List
- Tool Bar
  - Create
  - Edit
  - Delete



What is a Scheme?

A Scheme is a set of common colors used to standardize the look and feel of a user interface.

- Name
- Font
- Background Image
- Blending
- Emboss
- Color Settings
- Fill Style
- Background Fill Type
- Preview



Create New Scheme

Name: Untitled

Font:

Image:

Image Offsets
X 0
Y 0

Alpha: 100

Emboss: 3

Colors:

Color 0	□	255,255,255
Color 1	■	64,64,64
Color Disabled	▨	192,192,192
Background	■	0,0,0
Emboss Dark	■	64,64,64
Emboss Light	▨	192,192,192
Text Color 0	■	0,0,0
Text Color 1	□	255,255,255
Text Color Disabled	▨	192,192,192
Gradient Start	■	0,0,0
Gradient End	□	255,255,255

Fill Style: Color

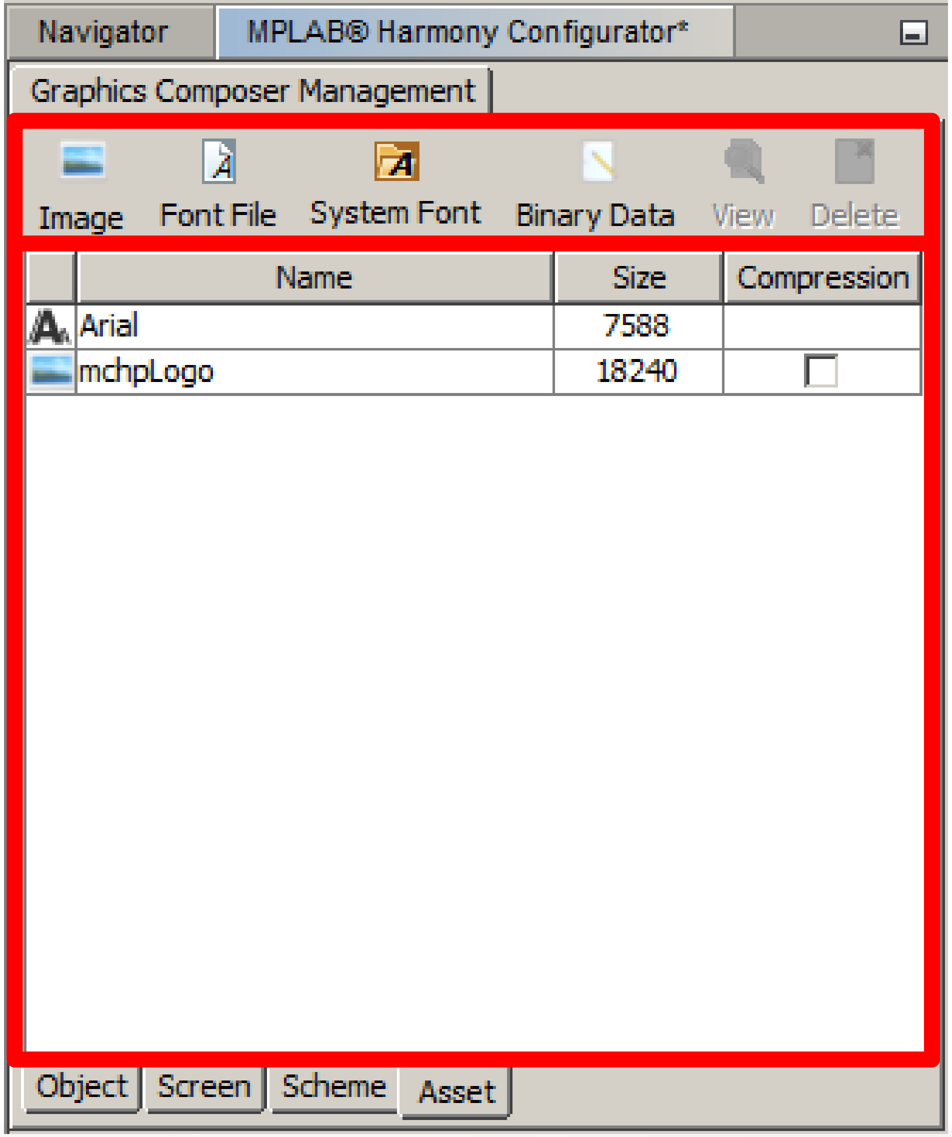
Background: None

Preview:

Graphics Composer Asset Tab

Asset Tab

- Asset List
  - Type
  - Name
  - Size
  - Compression
- Tool Bar
  - Image
  - Font File
  - System Font
  - Binary Data
  - View
  - Delete



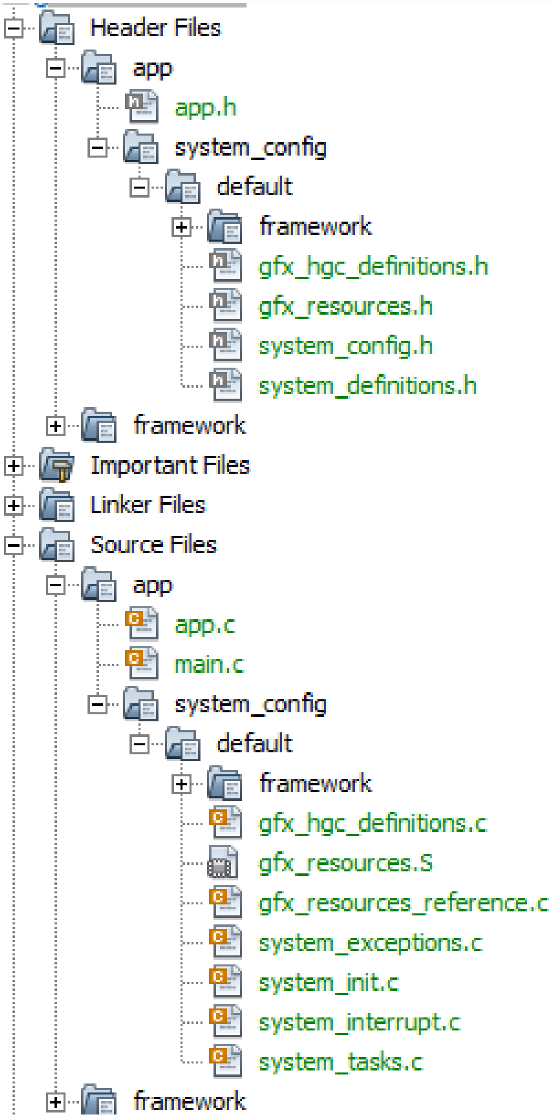
	Name	Size	Compression
	Arial	7588	
	mchpLogo	18240	□

Graphics Composer Output

How does Composer modify my project?

- Composer State Machine
  - gfx_hgc_definitions.h
  - gfx_hgc_definitions.c

- Composer Resource Files
  - gfx_resources.h
  - gfx_resources.S
  - gfx_resources_reference.c



```
Header Files
  app
    app.h
    system_config
      default
        framework
        gfx_hgc_definitions.h
        gfx_resources.h
        system_config.h
        system_definitions.h
  framework
Important Files
Linker Files
Source Files
  app
    app.c
    main.c
    system_config
      default
        framework
        gfx_hgc_definitions.c
        gfx_resources.S
        gfx_resources_reference.c
        system_exceptions.c
        system_init.c
        system_interrupt.c
        system_tasks.c
  framework
```

- Questions about:

  MPLAB Harmony Graphics Composer?

Break Time

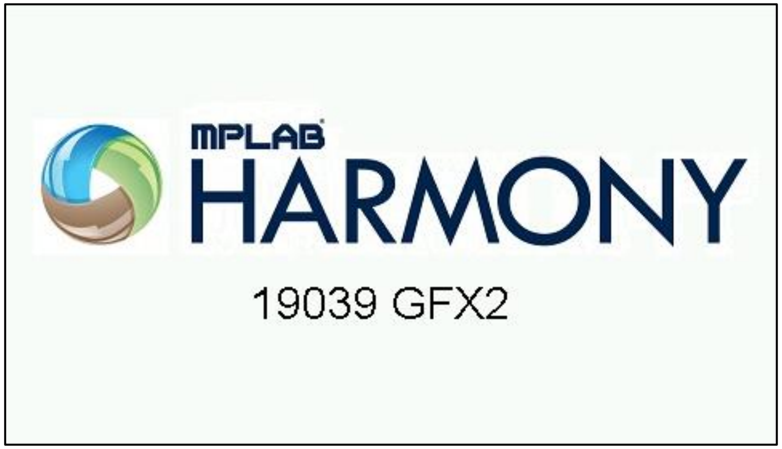
Lab 2
Splash Screen

Lab 2 – Splash Screen

Objective:

Using the MHC, you will create a splash screen with images and font.

Lab 2 – Splash Screen



MPLAB HARMONY

19039 GFX2

Splash Screen Visual Target

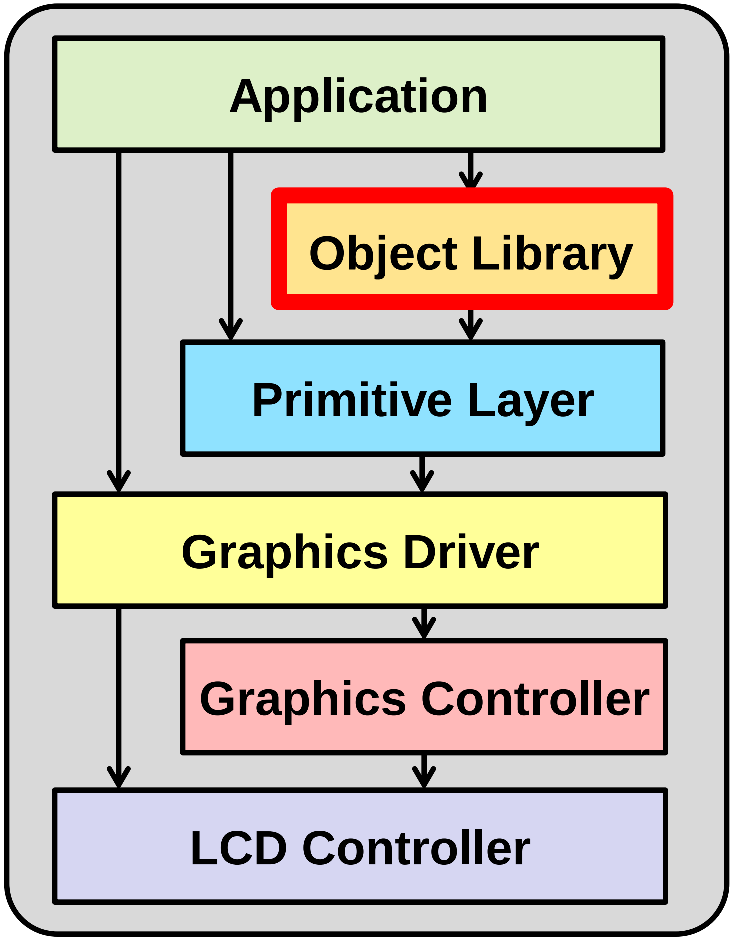
Lab 2 – Splash Screen

Conclusion:

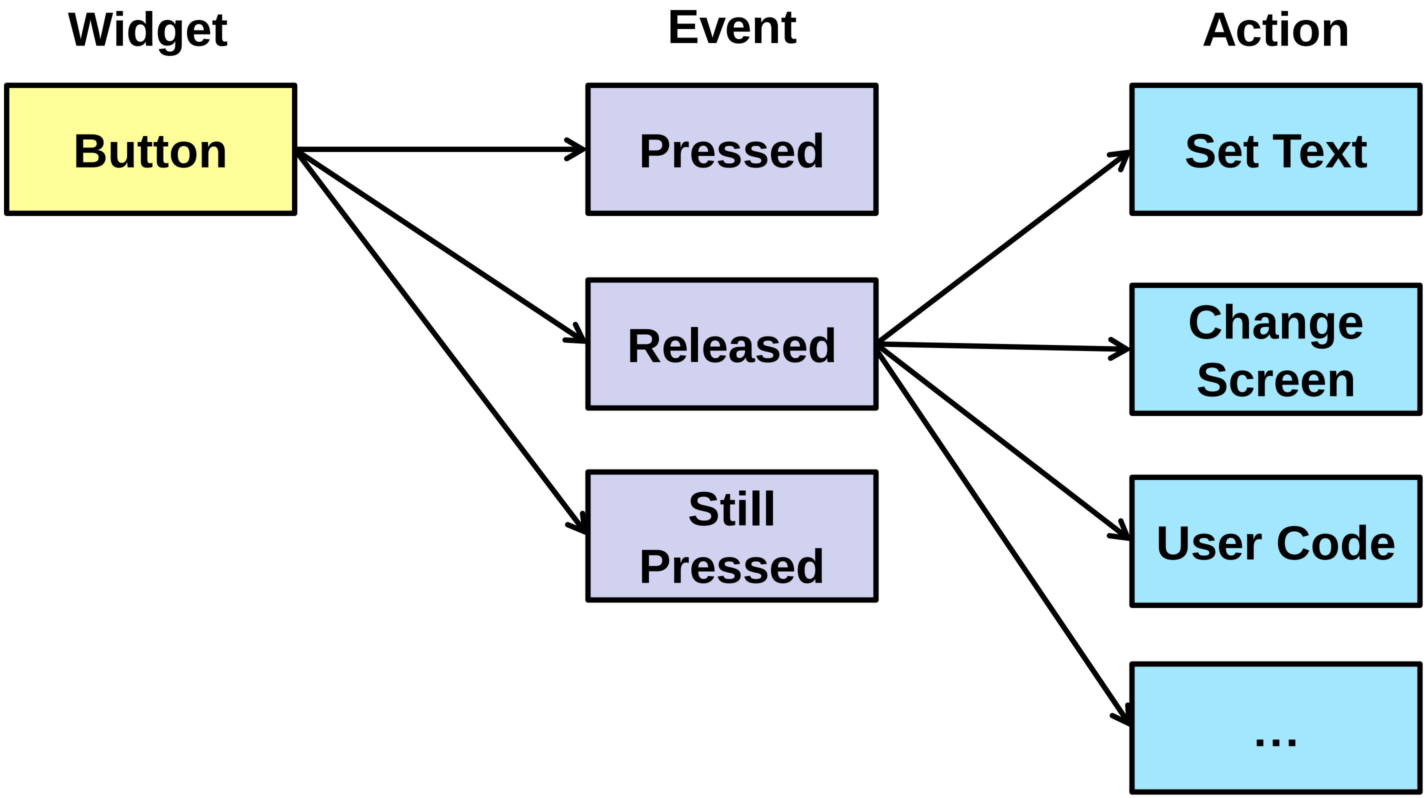
Now that we have added simple graphics and fonts. We will enable touch functionality, add widgets and events in the next lab.

MPLAB Harmony Graphics Composer Widget Events

Object Library Events



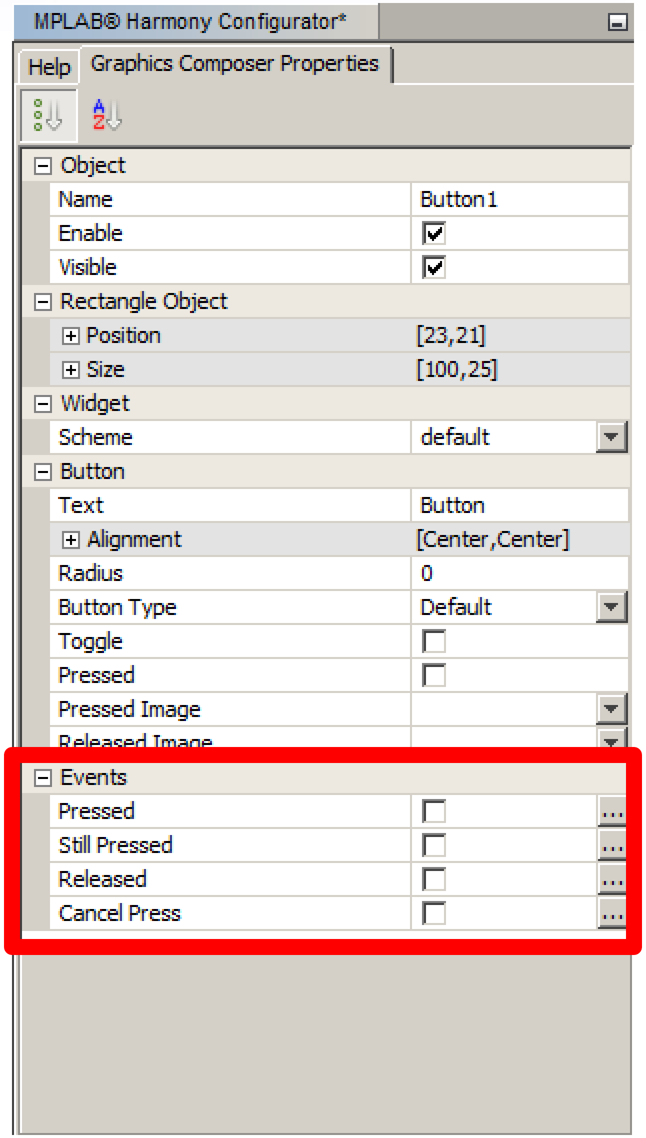
Object Library Events



Widget Event Editor

Event Properties

- Select Object
- Enable Event
- Press Action Button



MPLAB® Harmony Configurator*

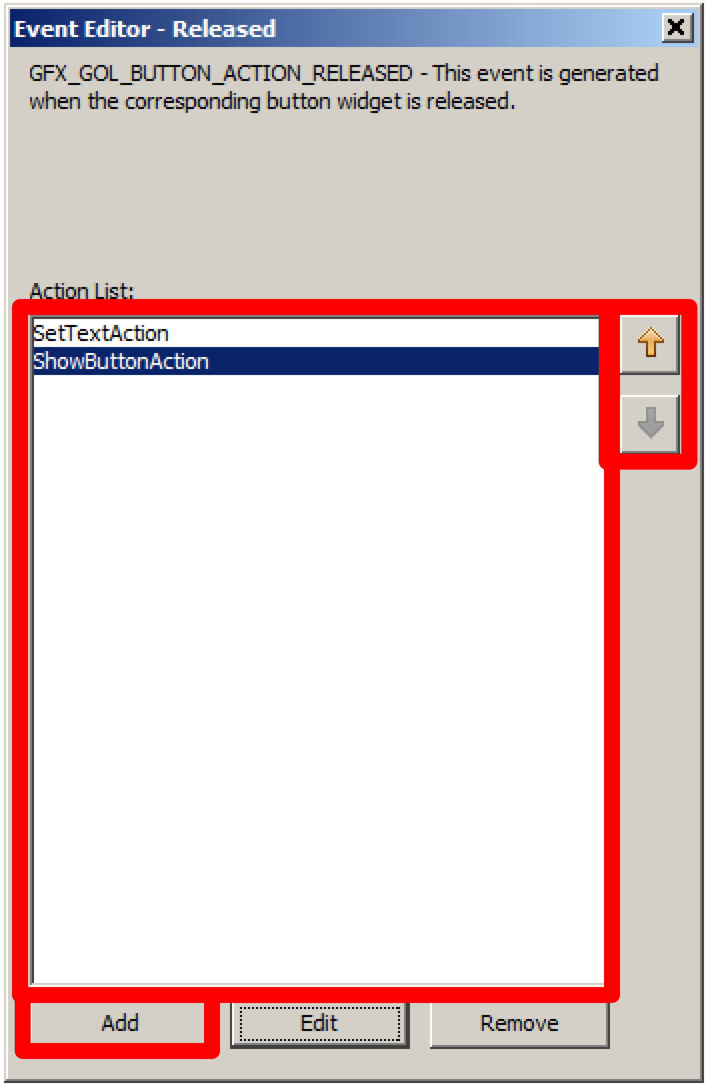
Help | Graphics Composer Properties

⊟ Object	
Name	Button1
Enable	☑
Visible	☑
⊟ Rectangle Object	
⊞ Position	[23,21]
⊞ Size	[100,25]
⊟ Widget	
Scheme	default
⊟ Button	
Text	Button
⊞ Alignment	[Center,Center]
Radius	0
Button Type	Default
Toggle	□
Pressed	□
Pressed Image	
Released Image	
⊟ Events	
Pressed	□ ...
Still Pressed	□ ...
Released	□ ...
Cancel Press	□ ...

Widget Events

Event Properties

- Action List
- Add Button
- Order Buttons



Event Editor - Released

GFX_GOL_BUTTON_ACTION_RELEASED - This event is generated when the corresponding button widget is released.

Action List:

SetTextAction
ShowButtonAction

- Questions about:

MPLAB Harmony Graphics Composer Events?

MASTERs 2015

Lab 3
Interactive Menu Screen

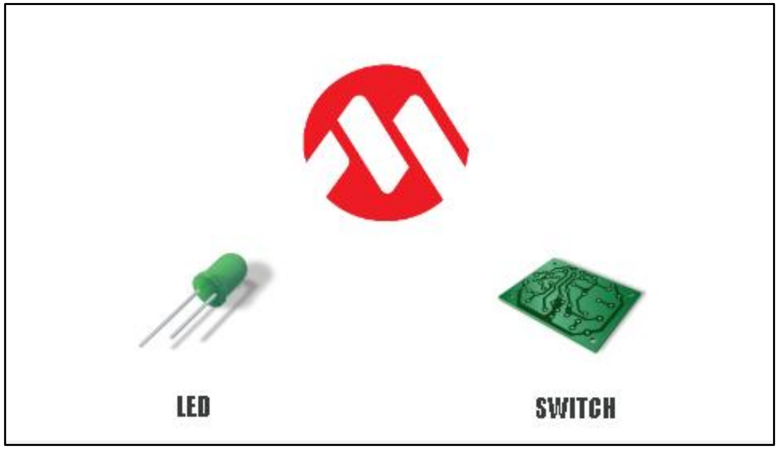
Lab 3 – Interactive Menu Screen

Objective:

In this lab, you will enable touch to your project. reate a menu screen with widgets

- **Add events to the widgets**

Lab 3 – Interactive Menu Screen



Menu Screen Visual Target

Lab 3 – Interactive Menu Screen

Conclusion:

Having seen how easy it is to add widgets and touch capability to your project, we can now explore how to add peripheral I/O to the graphics project in the following labs.

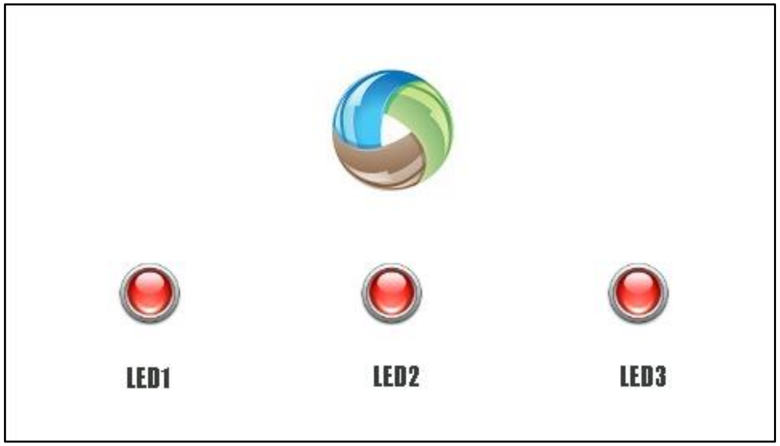
Lab 4
LED Control Screen

Lab 4 – LED Control Screen

Objective:

Use the MHGC to create an LED Control screen. It will contain widgets to control the LEDs on the MEB II development board.

Lab 4 – LED Control Screen



LED Screen Visual Target

Lab 4 – LED Control Screen

Conclusion:

Having learned how to send output from widgets to a peripheral, we will next learn how to receive an input and display it.

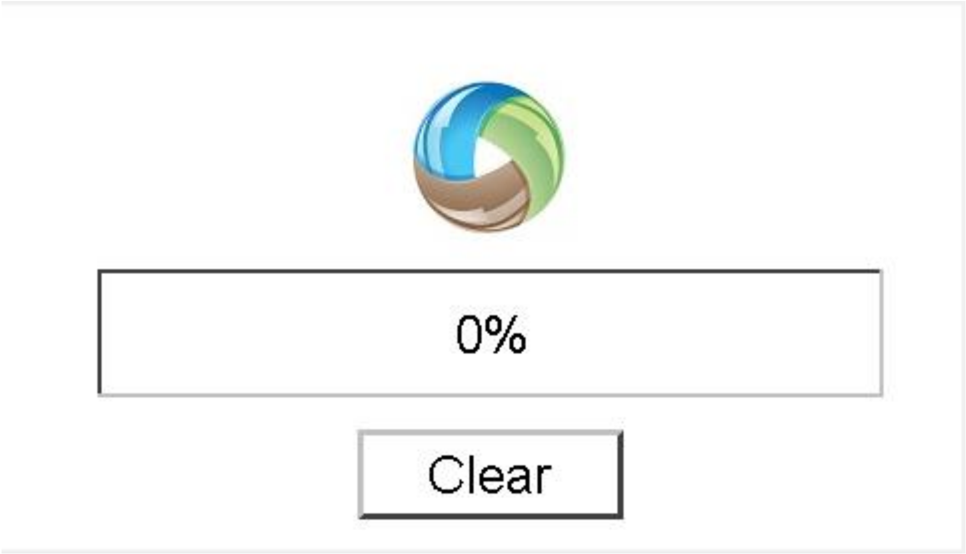
Lab 5
Switch Input Screen

Lab 5 – Switch Input Screen

Objective:

Use the MHGC to create a new screen. This screen will contain widgets that will respond to switch inputs from the MEB II.

Lab 5 – Switch Input Screen



Switch Input Screen Visual Target

Lab 5 – Switch Input Screen

Conclusion:

Now you have learned how to integrate peripheral I/O to a graphics project using MHGC.

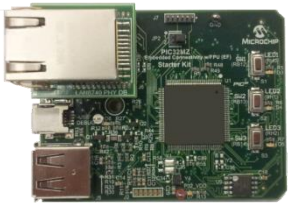
Class Summary

- **Today we covered:**
  - Basic Graphics Theory
  - MPLAB Harmony Overview
  - MPLAB Harmony Configurator
  - MPLAB Harmony Graphics Composer

- Questions about:

  # MPLAB Harmony Graphics Development?

Dev Tools For This Class



- PIC32MZ EF Starter Kit (DM320007)
- Multimedia Expansion Board II (MEB II) (DM320005-2)
- MPLAB Harmony v1.06

Suggested Reading

Programming 32-bit Microcontrollers in C

by Lucio Di Jasio

ISBN-10: 0750687096

ISBN-13: 978-0750687096

Thank You!

LEGAL NOTICE

SOFTWARE:

You may use Microchip software exclusively with Microchip products. Further, use of Microchip software is subject to the copyright notices, disclaimers, and any license terms accompanying such software, whether set forth at the install of each program or posted in a header or text file.

Notwithstanding the above, certain components of software offered by Microchip and 3rd parties may be covered by "open source" software licenses – which include licenses that require that the distributor make the software available in source code format. To the extent required by such open source software licenses, the terms of such license will govern.

NOTICE & DISCLAIMER:

These materials and accompanying information (including, for example, any software, and references to 3rd party companies and 3rd party websites) are for informational purposes only and provided "AS IS." Microchip assumes no responsibility for statements made by 3rd party companies, or materials or information that such 3rd parties may provide.

MICROCHIP DISCLAIMS ALL WARRANTIES, WHETHER EXPRESS, IMPLIED, OR STATUTORY, INCLUDING ANY IMPLIED WARRANTIES OF NONINFRINGEMENT, MERCHANTABILITY, AND FITNESS FOR A PARTICULAR PURPOSE. IN NO EVENT WILL MICROCHIP BE LIABLE FOR ANY DIRECT OR INDIRECT, SPECIAL, PUNITIVE, INCIDENTAL, OR CONSEQUENTIAL LOSS, DAMAGE, COST, OR EXPENSE OF ANY KIND RELATED TO THESE MATERIALS OR ACCOMPANYING INFORMATION PROVIDED TO YOU BY MICROCHIP OR OTHER THIRD PARTIES, EVEN IF MICROCHIP HAS BEEN ADVISED OF THE POSSIBLITY OF SUCH DAMAGES OR THE DAMAGES ARE FORESEEABLE.

TRADEMARKS:

The Microchip name and logo, the Microchip logo, dsPIC, FlashFlex, flexPWR, JukeBlox, KEELOQ, KEELOQ logo, Kleer, LANCheck, MediaLB, MOST, MOST logo, MPLAB , OptoLyzer, PIC, PICSTART, PIC^{32} logo, RightTouch, SpyNIC, SST, SST Logo, SuperFlash and UNI/O are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

The Embedded Control Solutions Company is a registered trademark of Microchip Technology Incorporated in the U.S.A.

Analog-for-the-Digital Age, BodyCom, chipKIT, chipKIT logo, CodeGuard, dsPICDEM, dsPICDEM.net, ECAN, In-Circuit Serial Programming, ICSP, Inter-Chip Connectivity, KleerNet, KleerNet logo, MiWi, MPASM, MPF, MPLAB Certified logo, MPLIB, MPLINK, mTouch, MultiTRAK, NetDetach, Omniscient Code Generation, PICDEM, PICDEM.net, PICkit, PICtail, RightTouch logo, REAL ICE, SQI, Serial Quad I/O, Total Endurance, TSHARC, USBCheck, VariSense, ViewSpan, WiperLock, Wireless DNA, and ZENA are trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

SQTP is a service mark of Microchip Technology Incorporated in the U.S.A.

GestIC is a registered trademarks of Microchip Technology Germany II GmbH & Co. KG, a subsidiary of Microchip Technology Inc., in other countries.

All other trademarks mentioned herein are property of their respective companies.