FTF2016
TECHNOLOGY FORUM CHINA

LPC微控制器
增强产品的图形交互和数据互联的体验

FTF-DES-N2307

2016年9月

日程安排

- 目标
- LPC微控制器家族介绍
- 基于LPC4088开发板 和 LPC开发生态
- 图形解决方案
  - 项目加载，建立新工程并运行例程
  - 图形显示 应用
  - 如何选择一个 图形库
  - emWin技术详解

目标

- LPC微控制器具有LCD控制器，可用来显示图形界面
- 熟悉LPCXpresso IDE开发工具和开发生态
- 利用 emWin 建立并定制 带有触控功能的图形界面

NXP LPC 微控制器家族一览

基本型

简单易用
低功耗
管脚数较少

LPC800 系列

低功耗
基础控制
串行接口

- 30 MHz Cortex-M0+ 内核
- 基本串行接口
- 基础模拟外设
- 小封装低引脚数 – 包括 TSSOP，HVQFN 和XSON
- 是理想的替代8位、16位微处理器的产品

获取信息

LPC1100 系列

低功耗
选择范围广
工业标准
带有USB或CAN

- 50 MHz Cortex-M0+或M0 内核
- 串行外设: 全速USB集成PHY CAN集成收发器 UART，I2C，SPI
- 优化的模拟接口
- 封装全，家族广
- 与LPC1300系列引脚兼容

获取信息

LPC1200 系列

专注于工业极恶略的噪声环境下的应用

- 45 MHz Cortex-M0 内核
- 适用于恶略环境(IEC61697-1)
- 8 kV ESD 保护
- 基本模拟外设
- 实时日历时钟
- Fm I²C 接口，驱动能力10倍于正常值

获取信息

高性能

极优的能耗
高性能接口
灵活的外设

LPC1300 系列

基本连接外设
合理的性能

- 高达72 MHz Cortex-M3 内核
- 串行外设: 全速USB CAN总线 UART,I2C,SPI
- 管脚与LPC1100系列兼容

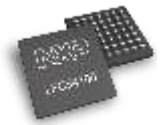
获取信息

LPC1500 系列

高精度运动控制

- 高达72 MHz Cortex-M3 内核
- 为霍尔或无霍尔传感器应用优化
- 提供FOC算法库
- 串行外设: USB, CAN。 UART,I2C,SPI
- 高性能模拟外设
- 多路SCT定时器和PWM

获取信息



LPC54100 系列

超低功耗系列
针对永久在线的传感器应用

- 高达100 MHz单核或双核微处理器: Cortex-M4F & M0+ (可选)
- 为传感器的监听收集，算法融合和交互做优化
- 超低功耗模式，在传感器监听状态下仅需要3uA
- 可动态调节的能耗模式

获取信息

LPC1700 系列

高性能，可选DSP功能。
多种数据交互方式 高性能外设

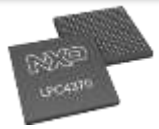
- 高达120 MHz Cortex-M3内核
- 高性能外设: USB, CAN, 以太网
- LCD控制器
- 引脚与LPC4000系列和ARM7 LPC2x00系列兼容

获取信息

LPC4000 系列

- 高达120 MHz Cortex-M4/M4F内核 带有DSP
- 高性能外设: USB, CAN, 以太网
- LCD控制器
- 模拟比较器
- 引脚兼容 LPC1700和 LPC2x00系列并拥有更好的性能

获取信息



LPC1800 系列

带有DSP和双核特色的超高性能系列
拥有多个高速外设接口
高性能外设

- 工业级别高性能Cortex-M3内核，主频高达 180 MHz
- LPC18Sxx系列包含安全加密特性
- 高性能接口: 两个高速USB, 两个CAN接口, 10/100以太网
- 高级灵活客赔的定时器
- 引脚与LPC4300系列兼容

LPC4300 系列

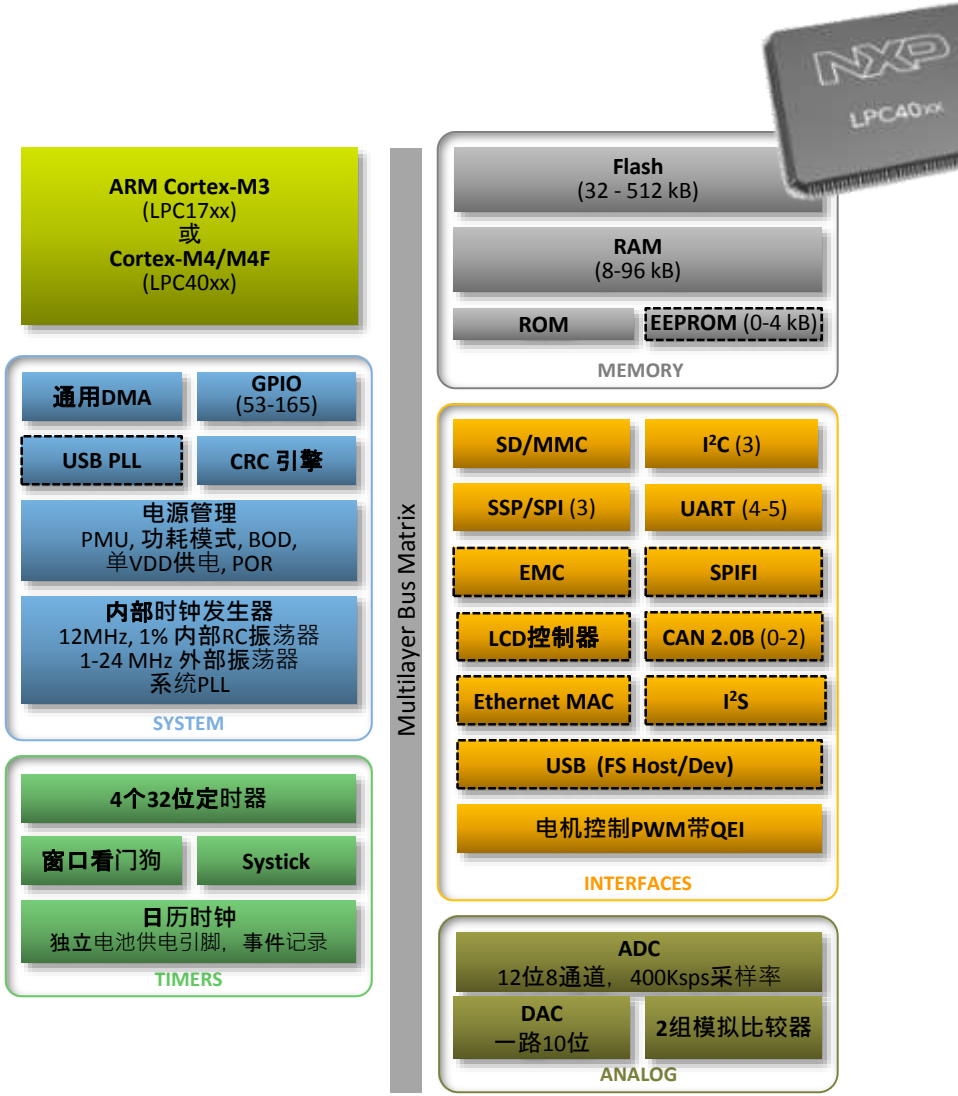
- 204 MHz Cortex-M4F 内核，带有DSP处理能力，以及多核Cortex-M0
- LPC43Sxx系列包含安全加密特性
- 任务分多核管理，优化性能
- 高性能接口: 两个高速USB, 两个CAN接口, 10/100M以太网, SGPIO可以灵活生成串行接口
- 内建高性能模拟外设，可达80Msps采样率的12位ADC

获取信息

LPC1700/4000 系列

高性能，支持多种连接手段，高性能的外设接口

- Cortex-M3/Cortex-M4F内核，主频可达120 MHz
  - 存储：512 kB Flash，96 KB RAM
  - SPIFI接口支持四线制的SPI Flash，具有XIP功能
  - 多种高性能外设覆盖多数应用
  - USB接口
    - 两个 全速 USB接口，均支持主机/从机模式。
    - 片上均集成PHY
    - 片上ROM集成有认证的USB从机驱动API
  - 图形LCD控制器，分辨率支持1024 x 768
  - CAN 2.0B
  - 10/100Mbps以太网接口
- 封装引脚兼容
  - LPC17xx 系列兼容 LPC2x00(ARM7TDMI) 和LPC40xx
  - LPC40xx 系列 与 LPC177x/8x 和 LPC2x00(ARM7TDMI)

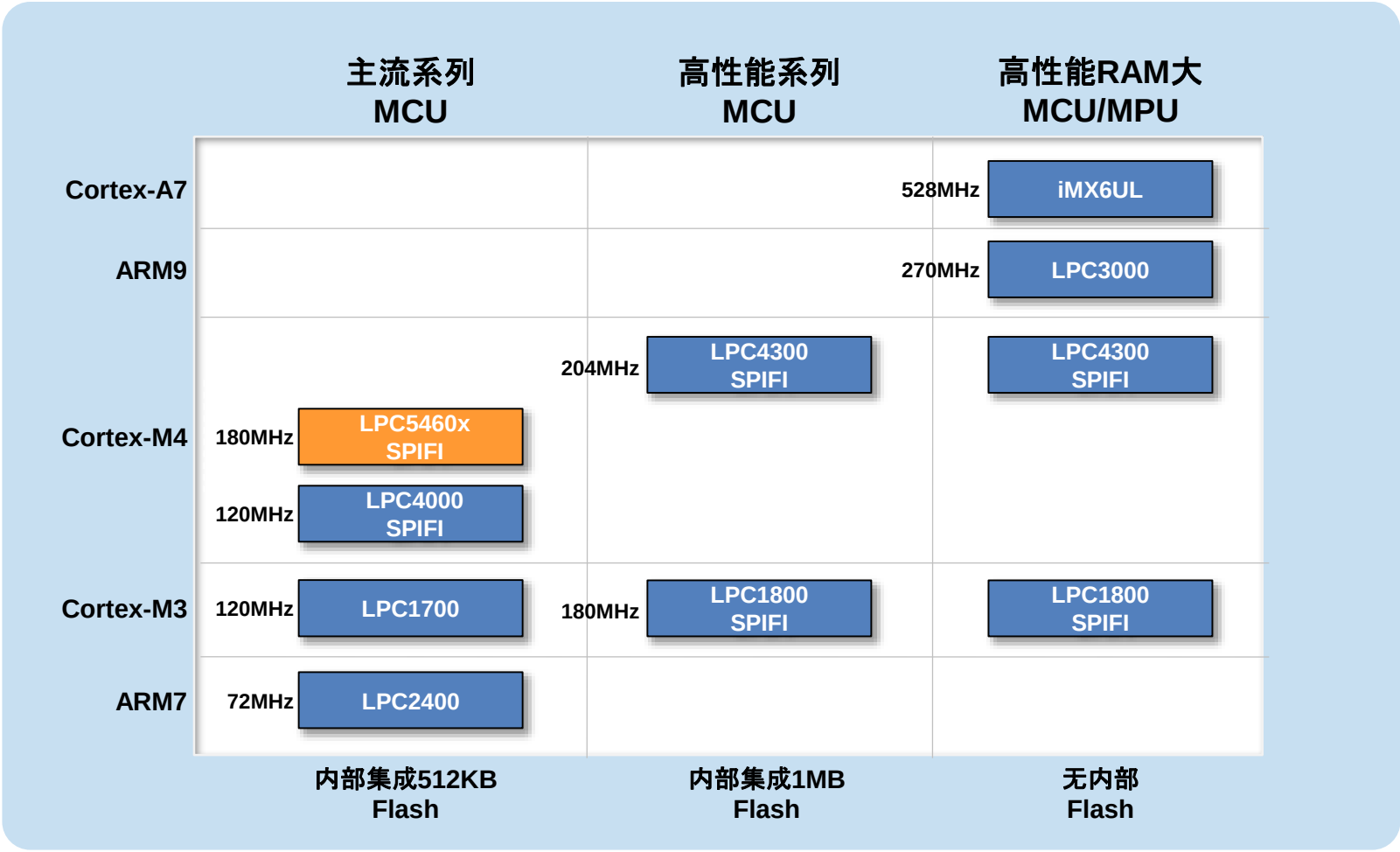


图形LCD控制器

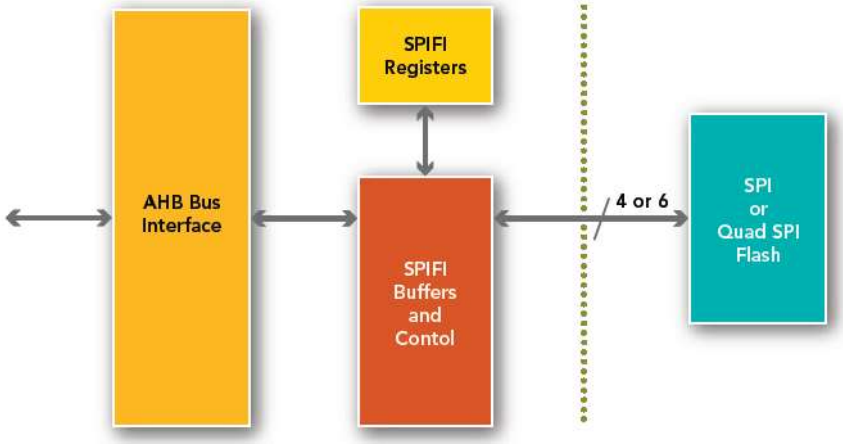
- LPC LCD控制器的特性
  - 支持STN和TFT屏幕
  - **支持最大1024x768的分辨率**
  - 24位LCD数据接口，**支持**24bpp(1600万色）
  - **支持**调色板，可以显示256到64K颜色
  - **可以配置**LCD总线位数，适配不同的屏幕总线
  - 专用LCD DMA**控制器**
  - **支持硬件光**标
- **丰富的**图形软件库资源
  - **免**费使用Segger**的**emWin图形库
  - **同**时也支持如TouchGFX**和**GUIX**的**图形库
- LPCOpen软件例程库 **和 多种板**级驱动支持包
  - **非常有效的**节约用户移植软件的工作量
  - **有专用的文档介**绍如何支持一块非标准的LCD显示器

具有LCD控制器的MCU/MPU家族

NXP具有超过40个器件带有LCD控制器，并在该市场领域超过10年的经验



存储器
SPIFI (支持四线制SPI Flash接口)



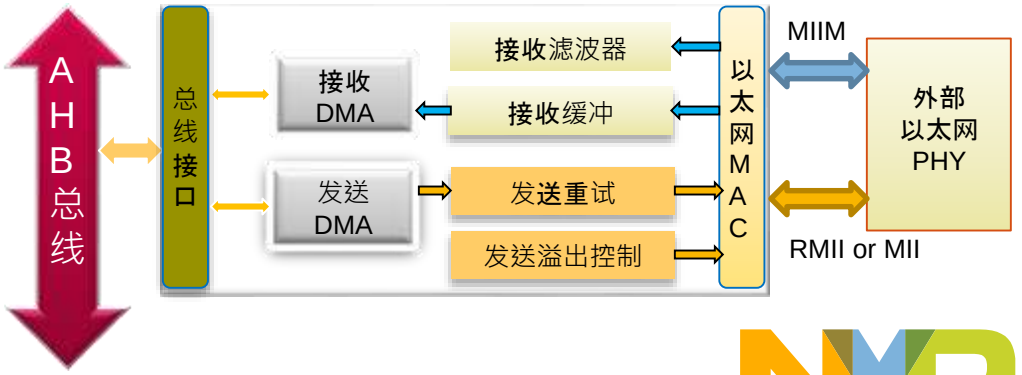
- 像内部存储一样，让外部SPI Flash出现下存储器地址映射
  - 获得执行指令的能力
- 使用SPIFI接口的好处：
  - 降低成本。体积小，价格低(相于并行Flash)
  - 并不损失太多的性能。大约为内部Flash执行性能的~70%。
  - 节约空间，节约使用的管脚数（相对于并行Flash）。
  - 为应用提供更多运行和存储空间。SP对I Flash是理想的镜像或者数据存储器件，可以让内部Flash空间更自由的给用户程序使用。
- SPIFI接口 支持
  - 多个主流供应商的四线制SPI Flash
  - 代码指令执行，数据访问以及从SPIFI接口启动
  - DMA访问

注重连接的外设 - USB

- 特色Features
  - 支持USB 2.0主机/从机/OTG模式
  - 支持USB低速,全速和高速协议，片上集成PHY
  - 支持所有的端点(Endpoint)类型，如 控制，批量，中断和同步传输
  - USB具有独立的2级PLL锁相环2nd
    - 内核和USB可以运行在不同的频率下，互不干扰
  - 主机控制器兼容OHCI/EHCI
  - 所有具有USB的器件均通过USB-IF认证
  - USB具有专用的DMA
  - 片上ROM集成有USB从机外设驱动
- NXP 为用户提供的 VID/PID 使用计划

注重连接的外设 – 以太网

- 10/100 Mbps IEEE 802.3 以太网MAC
- IEEE 1588-2008时间戳模块
- **支持全双工或半双工操作**
- **支持**DMA访问，为了提高性能具有数据包专用的RAM块
- **支持**MII和RMII**接口**连接外置PHY
- LPCOpen软件库 **支持** LWIP协议栈
- 发**送和接收具有独立的**缓冲区，可以提供更好数**据吞吐性能**



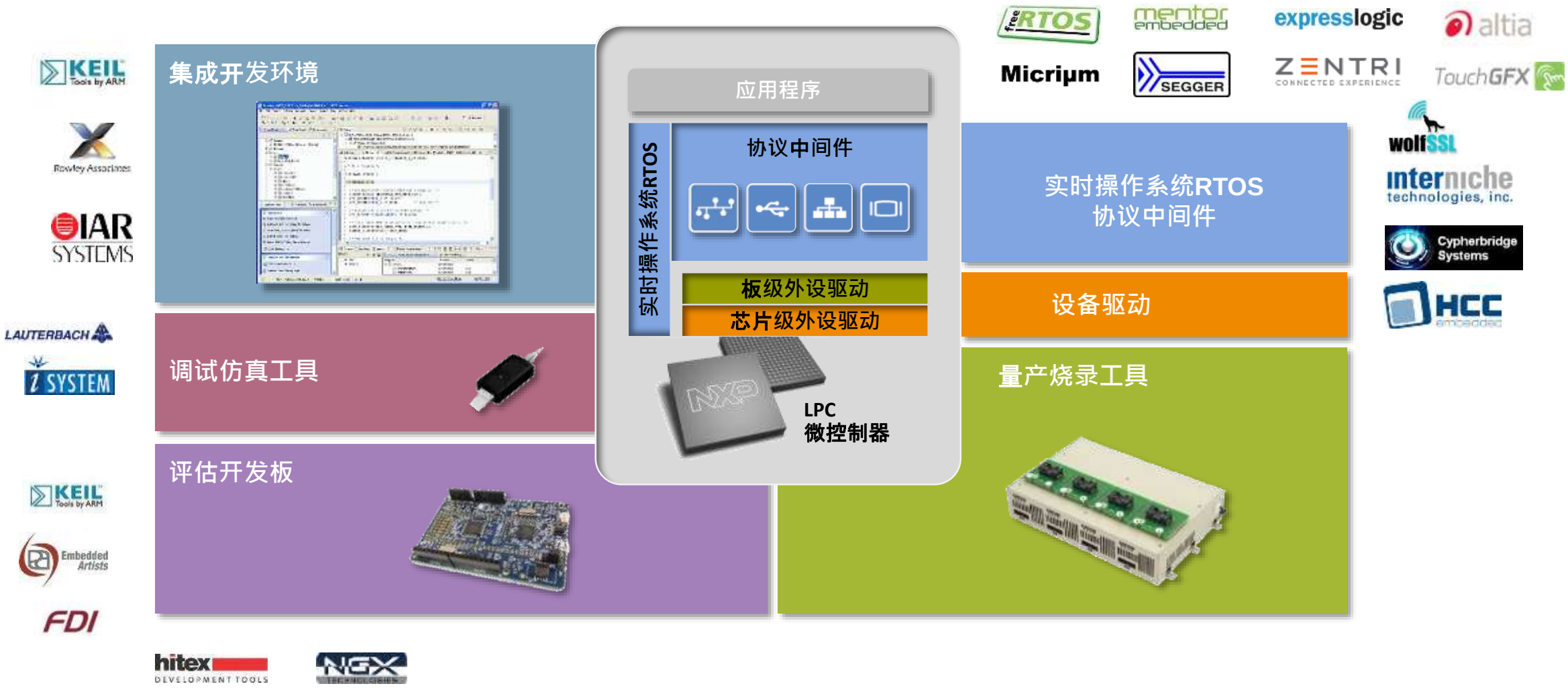
评估板 Embedded Artists LPC4088 LCD评估板

- 基于LPC4088微处理器的显示评估板
  - 经过CE认证，生产符合ISO9001
  - 工作温度范围 从-20 到 60摄氏度 (温度受限于LCD屏幕)
- 板上集成16MB四线制SPI Flash 和 32MB SDRAM
- 4.3”寸TFT LCD屏幕，带有透射式的电容触摸
- 板上集成支持CMSIS-DAP的仿真器，也支持外部仿真器接口
- USB主机和从机接口
- MicroSD/TF 卡存储器 卡槽接口
- XBee 无线模块兼容的接口
- mbed兼容，带有多种示例程序
- 专有 LPCOpen软件库 外设驱动/例程 包

开发板大图

- 获取资料

LPC微控制器开发生态



图形交互
应用和实现

段码式和点阵式 LCD显示器

7seg 15seg

段码LCD

段式驱动LCD显示器的特点：
低速率带宽，
低通讯速率
低功耗,
可显**示内容少**,
需要定制

图形显示LCD

适用于 急需要字符也需要图形显示的应用场合

TFT显示器的应用举例

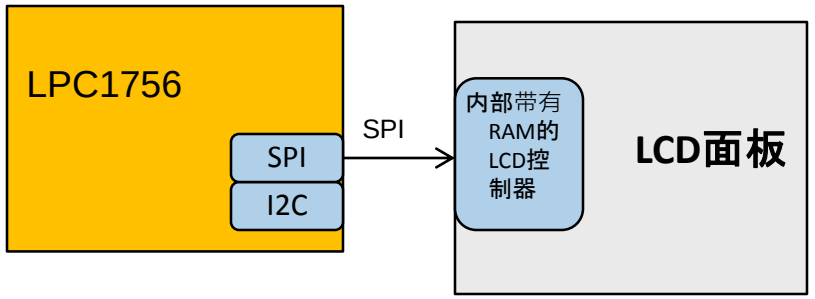



- TFT LCD显示器的嵌入式应用
  - **智能家居和安防**
    - **温控器，安全控制板，内部通**话系统
  - **保**证交易的安全
    - POS**机**，访客系统，售票机
  - **白色家**电
    - **高端**产品，具有显示和人机交互界面
  - **工**业人机交互/PLC**控制**单元
    - 转速监控器，温度监控器，报警器
  - **医**疗设备
    - **便携**仪表，大型监控设备
- TFT LCD显示器典型的分辨率需求从CGA(320x200)**到**XGA(1024x768)，**刷新率小于**15帧每秒

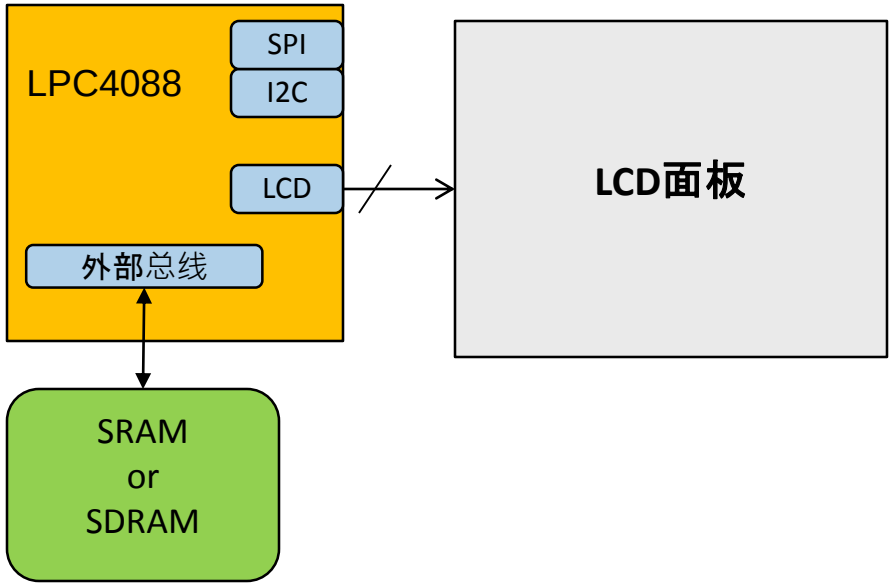
如何让微控制器连接LCD

微控制器连接LCD的两种方式

– **通**过串行总线连接LCD（**如I2C或SPI**）
  - **只能支持低分辨率的屏幕，原因是串行**总线的数据带宽有限
  - **但是不需要很多的管脚就可以**连接控制LCD



– **通**过并行RGB总线连接LCD
  - **能**够支持中端分辨率的LCD

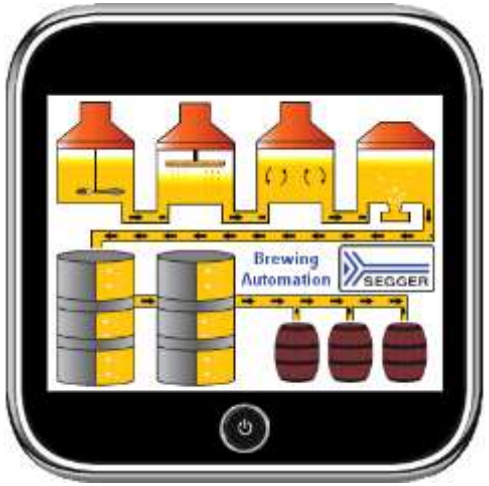


图形显示开发工具包

应用带来的挑战

- 用户体验的提升需要
- 产品经理对项目的定义
- 用户界面UI开发的时间消耗
- 开发者不熟悉UI或图形界面开发方法
- 如何根据应用选择 合适的TFT显示器
  - 以往的产品设计并没有用过显示屏
  - 白色家电，洗衣机，烤箱等
- 产品质量和设计上的挑战
  - 消费者注重产品的外观和屏幕显示效果
  - 应对人机交互设计

使用emWin做人机界面的效果



客户用LPC+emWin做了什么…

- 点钞机 (LPC4300 + 3.1寸，320x240 LCD)
- ATM机 (LPC1800 + 14寸，1024x768 LCD)
- 工业触摸屏 (LPC1788 + 10.1寸，640x480 LCD)
- 洗衣机(LPC3000)
- 电梯信息显示屏 (LPC1788)
- 高精度衡器显示 (LPC1788)
- 安全输入触摸板 (LPC2132)

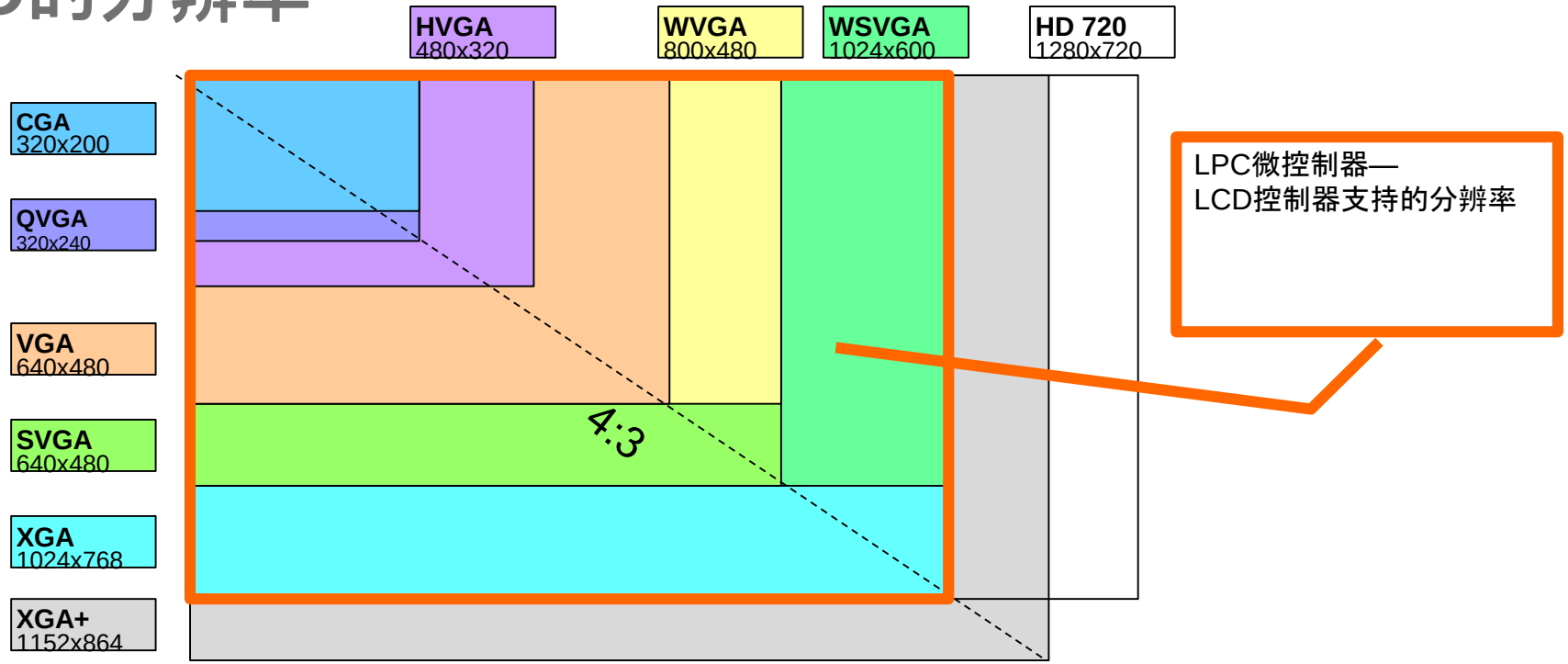
使用法律条款

- 针对NXP现有ARM系列如Cortex-M0/M3/M4内核的产品的用户是免费的
- 如果使用NXP的微控制器则无需额外支付版权费或者许可费用
- 以库的方式提供，不提供源代码
- 可以和SEGGER达成协议获取源代码

- The full license agreement is included in every installer, but there are essentially no limitations on the use of emWin with NXP MCUs
- The only restriction is that the emWin library is provided solely in object code ("library") format. Customers may use these libraries on NXP MCUs free of charge (without royalty or additional license fees), for both personal and commercial development
- As part of the licensing agreement with Segger, the source code for emWin can not be provided, but if you require the original source code for your own project, Segger offers special pricing for NXP customer's when upgrading from the NXP emWin library

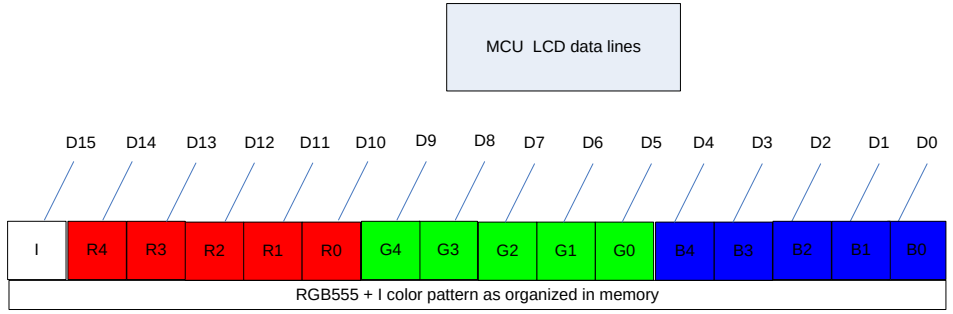
使用TFT LCD显示
技术细节

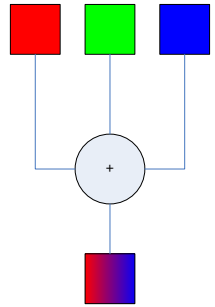
选择LCD的分辨率



选择LCD的色深

- 色深 或者 像素位数 (bpp)





分辨率和色深

- 分辨率和显示器的尺寸是两个概念!
  - QVGA 320 X 240
  - VGA 640 x 480
  - SVGA 800 X 600
  - 扫描的方向
- 色深 或 像素位数 (bpp)

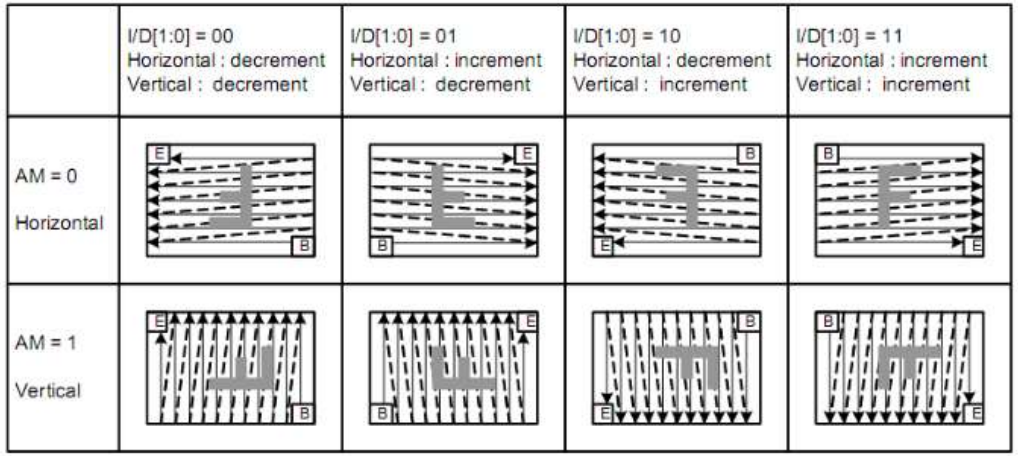
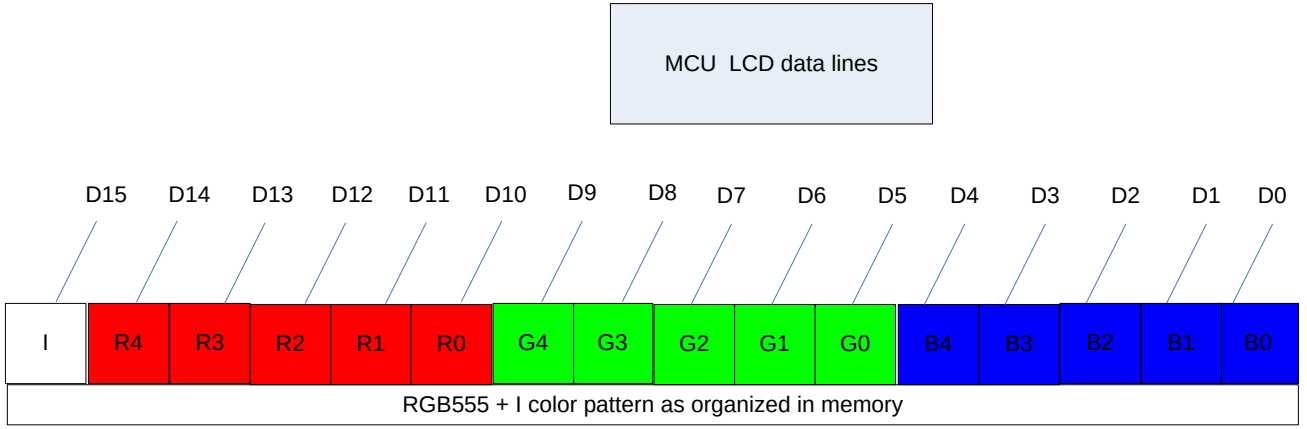
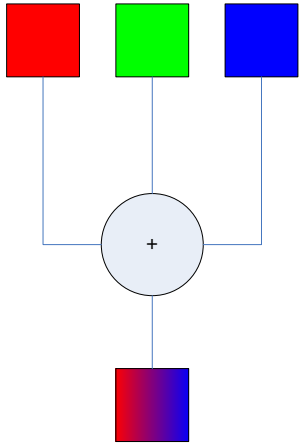


Figure25 GRAM Access Direction Setting





显示缓冲区?

- **连续的 存**储器缓冲区，大小至少为一帧图像的数据
- **包括**图像上每个像素的颜色数据
- 颜色数据通常定义为一下几种格式：
  - 1 位 (1 bpp): 单色
  - 2 位 (2 bpp): 灰阶 (4色）基于调色板
  - 4 位 (4 bpp): 色阶 (16色）基于调色板，控制器查找自己的灰阶色表
  - 8 位 (8 bpp): 色阶 (256色）基于调色板，控制器查找自己的灰阶色表
  - 16 位 (16 bpp): 高色彩格式 (5:5:5 - 32,768种色彩; 5:6:5 - 65,536种色彩)
  - 24 位 (24 bpp): 真彩色格式 (16,777,216种色彩)

存储器大小=分辨率 X 颜色深度

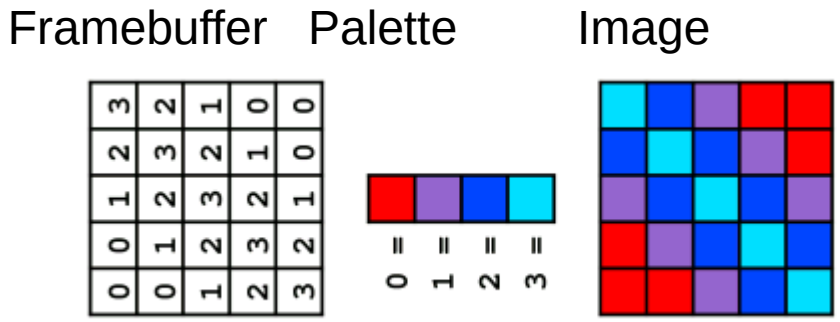
- 分辨率X颜色深度= 显示需要的数据位数 (除以8得到字节数)
- 帧缓冲区 = 一帧（或多帧）图像(bitmap)的数据总共占用的存储器的大小

举例: 480 x 272 x 16bpp / 8bits(byte) = 261,120 字节/帧图像

分辨率		1 位 / 像素点	2 位 / 像素点	4 位 / 像素点	8 位 / 像素点	16 位 / 像素点	24 位 / 像素点
XGA	1024x768	98,304	196,608	393,216	786,432	1,572,864	2,359,296
WVGA	800x480	48,000	96,000	192,000	384,000	768,000	1,152,000
VGA	640x480	38,400	76,800	153,600	307,200	614,400	921,600
WQVGA	480x272	16,320	32,640	65,280	130,560	261,120	391,680
QVGA	320x240	9,600	19,200	38,400	76,800	153,600	230,400
CGA	320x200	8,000	16,000	32,000	64,000	128,000	192,000

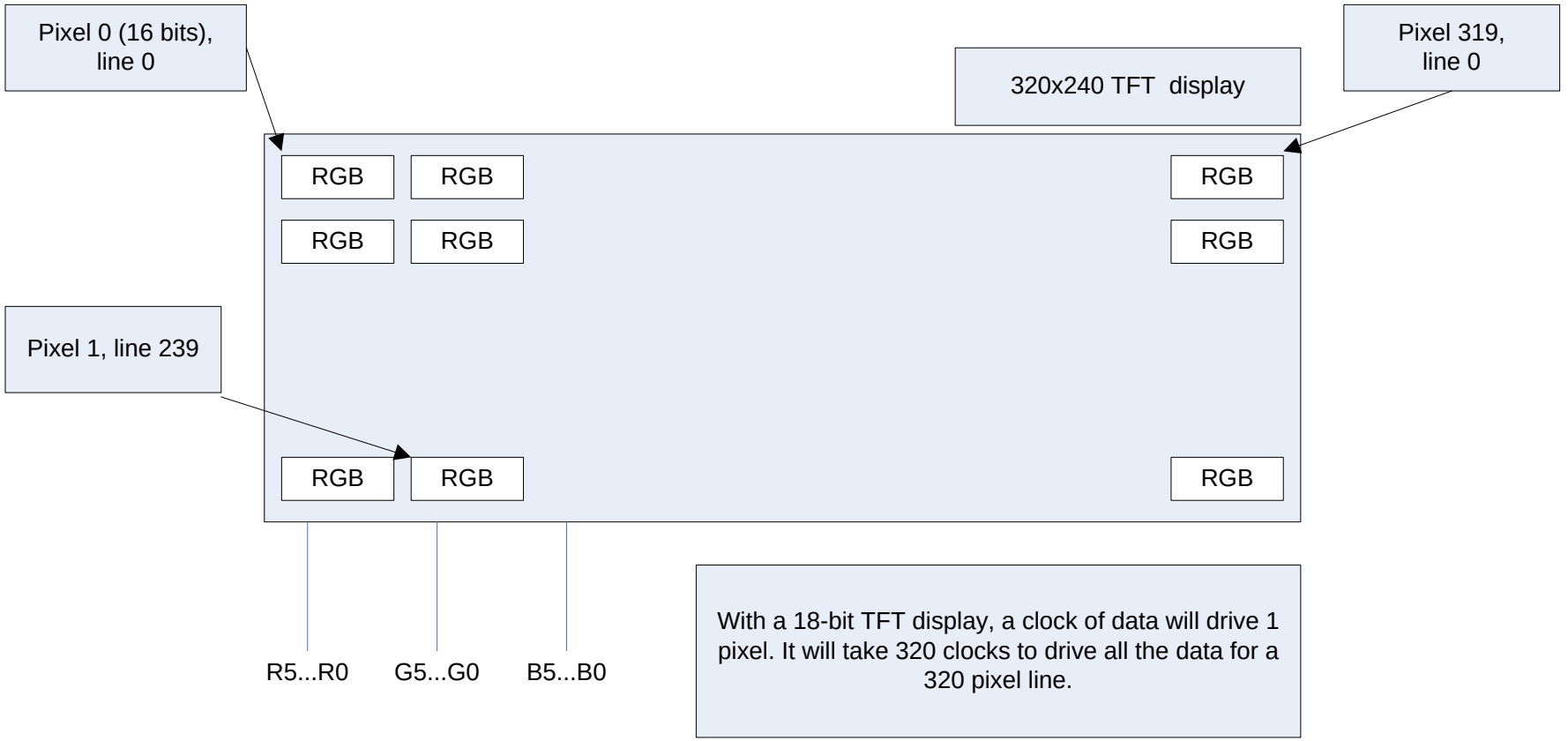
基于调色板的显示缓冲区

- 显示缓冲区包含每个像素点的数据表
- 调色板的RAM被预先设置为16位的颜色数据表中的代号

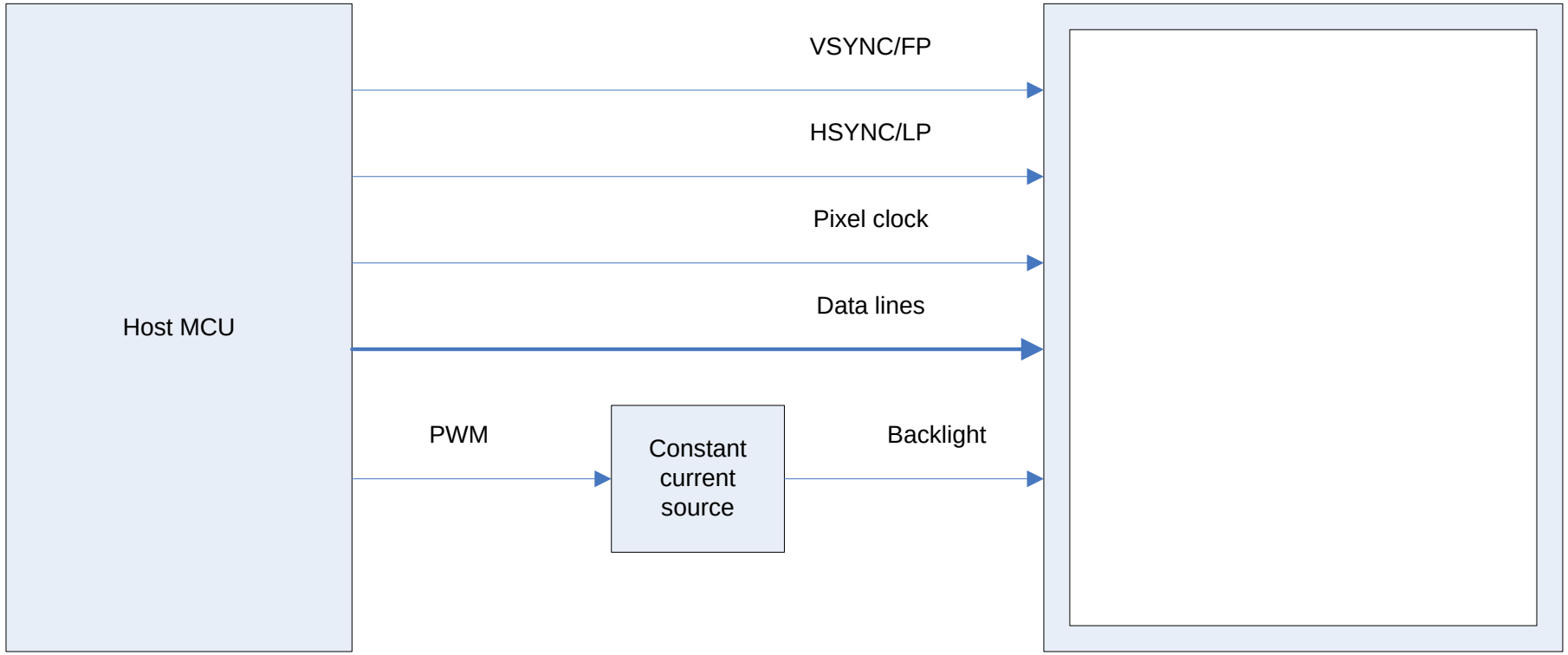


- NXP微控制器支持256阶颜色显示
  - 针对彩色的STN或者TFT支持 1, 2, 4, 或8 bpp 色阶 颜色显示
  - 针对单色STN屏幕支持1, 2,或4位bpp 灰阶 显示

需要时钟驱动的LCD


Pixel 0 (16 bits), line 0
320x240 TFT display
Pixel 319, line 0
RGB
RGB
RGB
RGB
RGB
RGB
Pixel 1, line 239
RGB
RGB
RGB
R5...R0
G5...G0
B5...B0
With a 18-bit TFT display, a clock of data will drive 1 pixel. It will take 320 clocks to drive all the data for a 320 pixel line.

提供LCD总线的时序



刷新频率

- 刷新率 (Hz) = 像素时钟频率/[(垂直分辨率+垂直预同步+垂直尾同步）*（一行像素时钟数+水平预同步+水平尾同步）)]

  pixel_clock_rate / [(vertical_resolution + vertical_front_porch + vertical_back_porch) * (pixel_clocks_per_data_line + horizontal_front_porch + horizontal_back_porch))]

- 举例:
  - 6.5MHz 像素时钟
  - 垂直分辨率 = 240 条线
  - 垂直预同步 = 5条线
  - 垂直尾同步 = 1跳线
  - 一行像素时钟 = 320 个像素时钟
  - 水平预同步 = 20个时钟
  - 水平尾同步 = 10个时钟

  - 刷新率 = 6,500,000 / [(240 + 5 + 1) * (320 + 20 + 10)] = 75.5Hz

LCD 时序 信号管脚

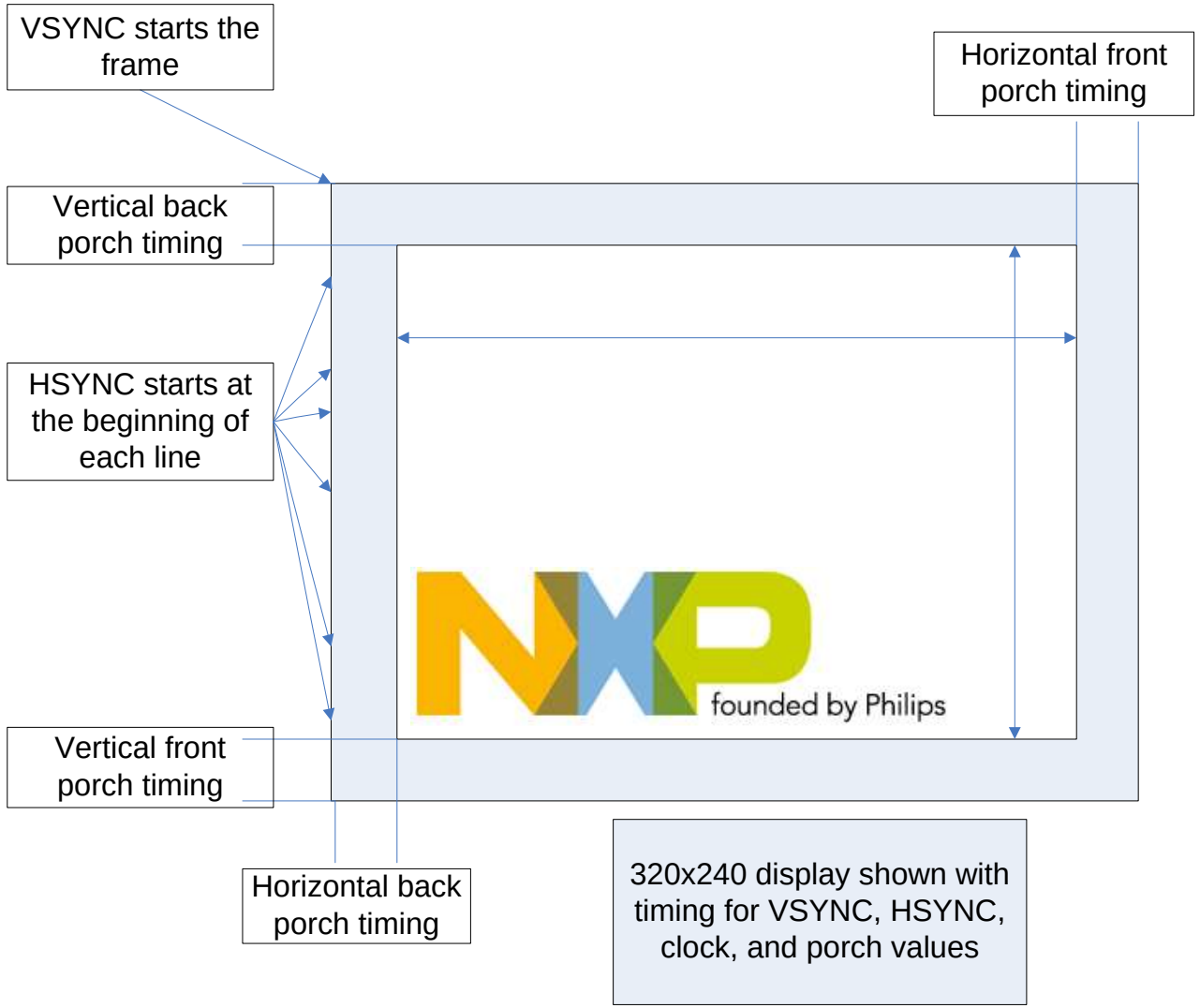
- LCD控制器最多需要31个引脚，若是驱动STN屏幕则至少需要10个引脚。

Pin name	Type	Function
LCD_PWR	output	LCD panel power enable.
LCD_DCLK	output	LCD panel clock.
LCD_ENAB_M	output	STN AC bias drive or TFT data enable output.
LCD_FP	output	Frame pulse (STN). Vertical synchronization pulse (TFT)
LCD_LE	output	Line end signal
LCD_LP	output	Line synchronization pulse (STN). Horizontal synchronization pulse (TFT)
LCD_VD[23:0]	output	LCD panel data. Bits used depend on the panel configuration.
LCD_CLKIN	input	Optional clock input.

LCD TFT 面板的信号引脚

Pin name	12-bit, 4:4:4 mode (18 pins)	16-bit, 5:6:5 mode (22 pins)	16-bit, 1:5:5:5 mode (24 pins)	24-bit (30 pins)
LCD_PWR	Y	Y	Y	Y
LCD_DCLK	Y	Y	Y	Y
LCD_ENAB_M	Y	Y	Y	Y
LCD_FP	Y	Y	Y	Y
LCD_LE	Y	Y	Y	Y
LCD_LP	Y	Y	Y	Y
LCD_VD[1:0]	-	-	-	RED[1:0]
LCD_VD[2]	-	-	Intensity	RED[2]
LCD_VD[3]	-	RED[0]	RED[0]	RED[3]
LCD_VD[7:4]	RED[3:0]	RED[4:1]	RED[4:1]	RED[7:4]
LCD_VD[9:8]	-	-	-	GREEN[1:0]
LCD_VD[10]	-	GREEN[0]	Intensity	GREEN[2]
LCD_VD[11]	-	GREEN[1]	GREEN[0]	GREEN[3]
LCD_VD[15:12]	GREEN[3:0]	GREEN[5:2]	GREEN[4:1]	GREEN[7:4]
LCD_VD[17:16]	-	-	-	BLUE[1:0]
LCD_VD[18]	-	-	Intensity	BLUE[2]
LCD_VD[19]	-	BLUE[0]	BLUE[0]	BLUE[3]
LCD_VD[23:20]	BLUE[3:0]	BLUE[4:1]	BLUE[4:1]	BLUE[7:4]

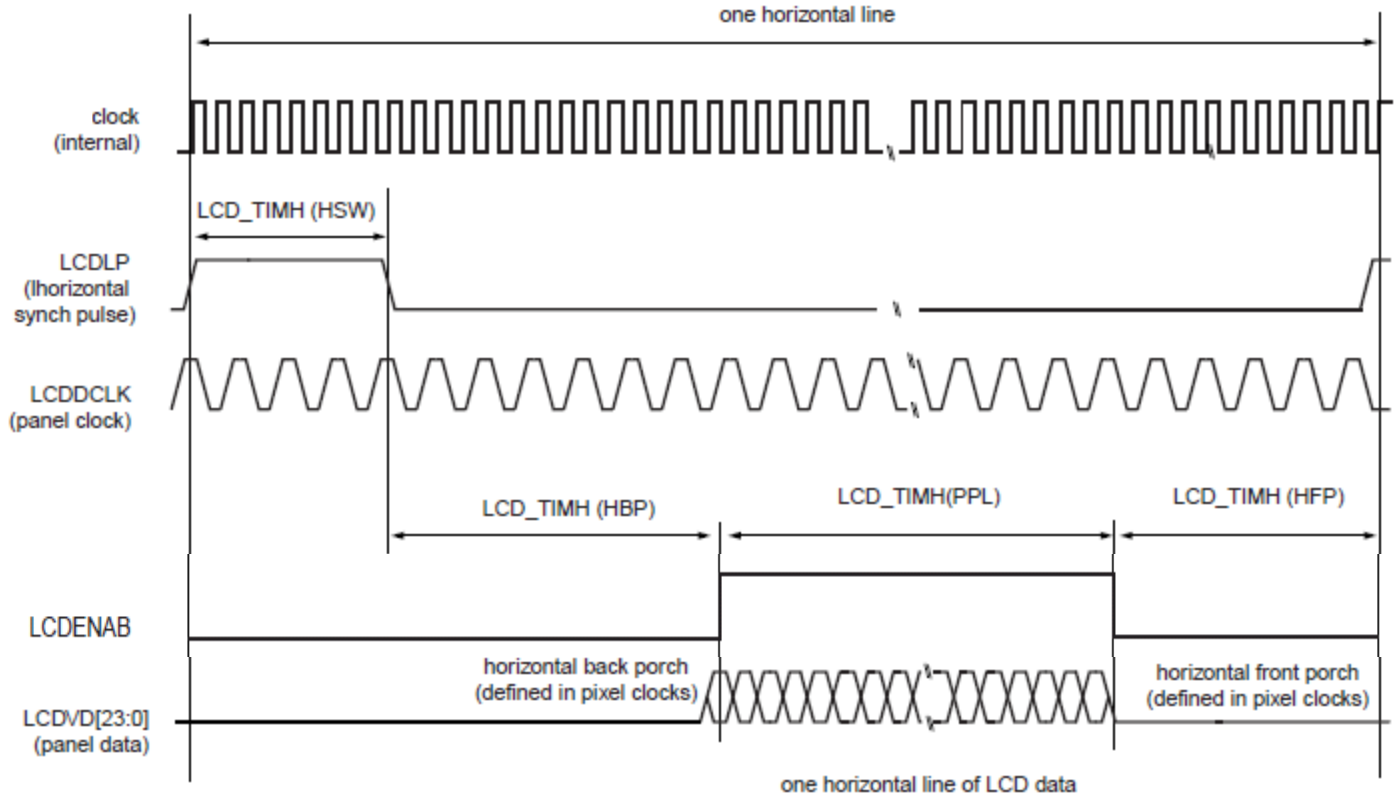
驱动 LCD 需要的时序


VSYNC starts the frame
Horizontal front porch timing
Vertical back porch timing
HSYNC starts at the beginning of each line
Vertical front porch timing
Horizontal back porch timing
320x240 display shown with timing for VSYNC, HSYNC, clock, and porch values

举例: 信利 240 x 320 TFT RGB666

Characteristics	Symbol	Min	Typ	Max	Unit
DOTCLK Frequency	f_{DOTCLK}	-	5.5	8.22	MHz
DOTCLK Period	t_{DOTCLK}	122	182	-	nSec
Horizontal Frequency (Line)	f_H	-	19.6	29.3	kHz
Vertical Frequency (Refresh)	f_V	-	60	90	Hz
Horizontal Back Porch	t_{HBP}	-	30	-	t_{DOTCLK}
Horizontal Front Porch	t_{HFP}	-	10	-	t_{DOTCLK}
Horizontal Data Start Point	t_{HBP}	-	30	-	t_{DOTCLK}
Horizontal Blanking Period	$t_{HBP} + t_{HFP}$	-	40	-	t_{DOTCLK}
Horizontal Display Area	HDISP	-	240	-	t_{DOTCLK}
Horizontal Cycle	H_{cycle}	-	280	-	t_{DOTCLK}
Vertical Back Porch	t_{VBP}	-	4	-	Line
Vertical Front Porch	t_{VFP}	-	2	-	Line
Vertical Data Start Point	t_{VBP}	-	4	-	Line
Vertical Blanking Period	$t_{VBP} + t_{VFP}$	-	6	-	Line
Vertical Display Area	VDISP	-	320	-	Line
Vertical Cycle	V_{cycle}	-	326	-	Line

LCD – TFT 水平时序

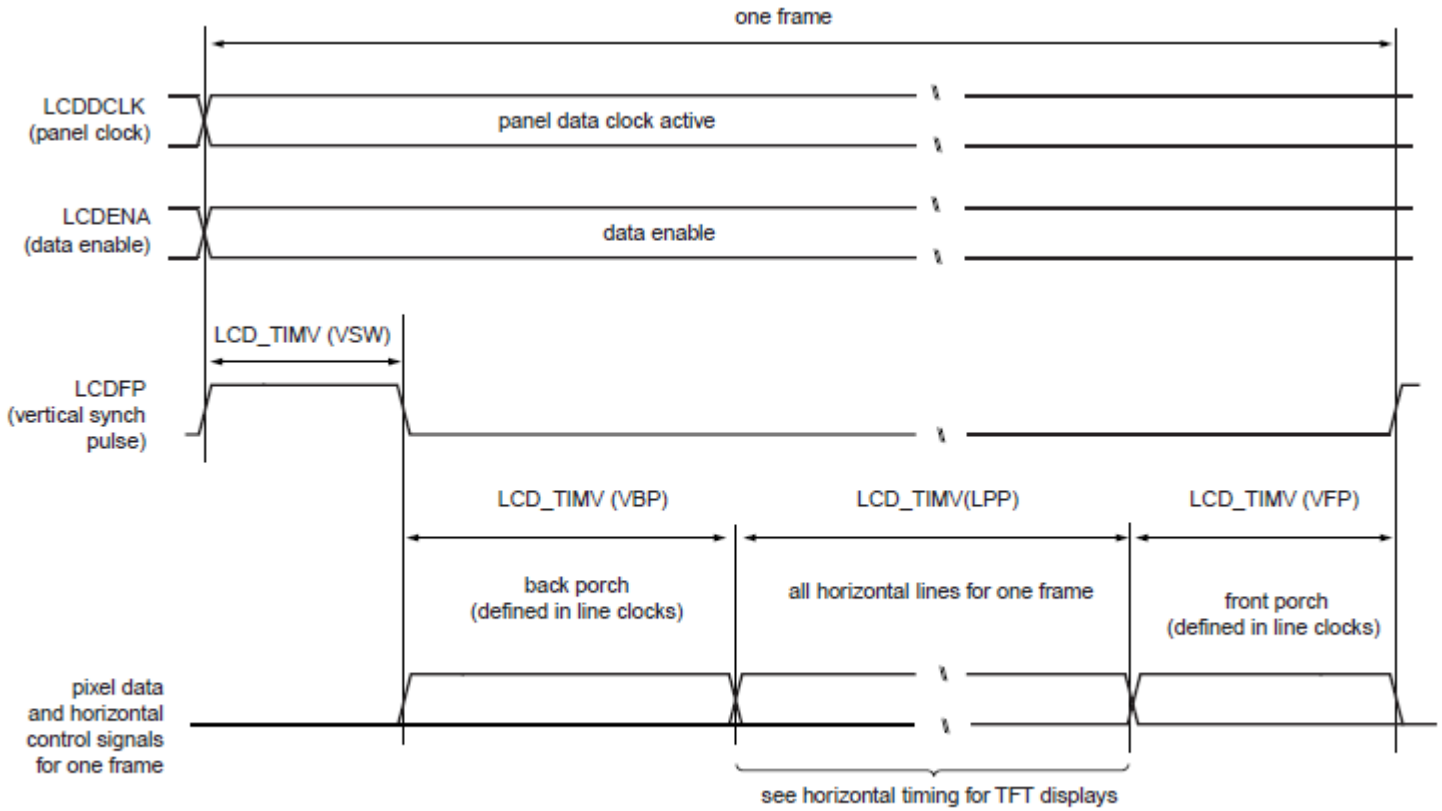


(1) The active data lines will vary with the type of TFT panel.

(2) The LCD panel clock is selected and scaled by the LCD controller and used to produce LCDCLK.

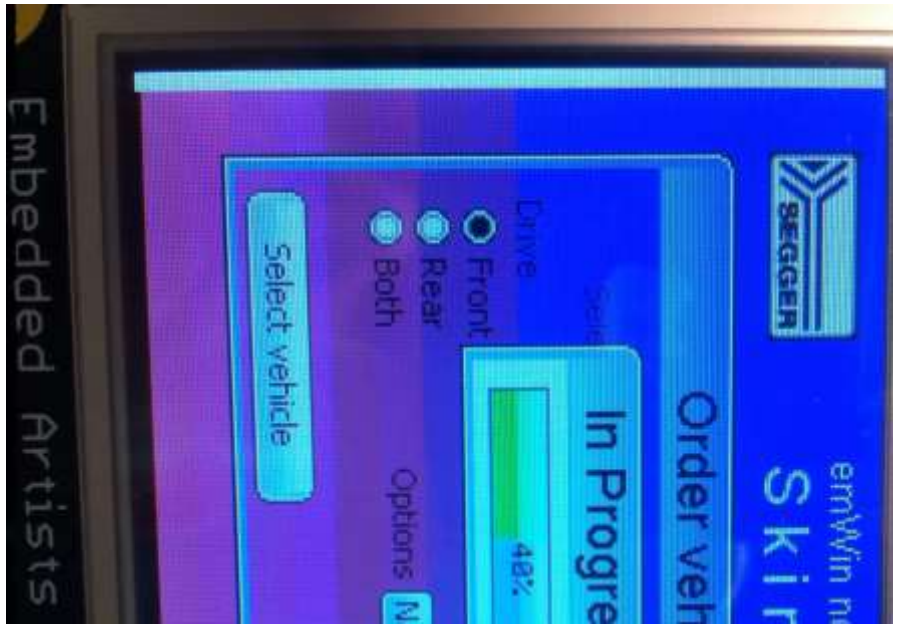
(3) The duration of the LCD_LP is controlled by the HSW field in the LCD_TIMH register.

LCD – TFT 垂直时序



(1) Polarities may vary for some displays.

错误设置LCD时序后的效果截图




```
static const LCD_PARAM_T truly_g240320ltsw =
{
   30, //28,       /* Horizontal back porch */
   10,       /* Horizontal front porch */
    2,       /* HSYNC pulse width */
  240,       /* Pixels per line */
    8,       /* Vertical back porch */
    2,       /* Vertical front porch */
    2,       /* VSYNC pulse width */
  320,       /* Lines per panel */
```

Vertical Back Porch	t_{VBP}	-	4

LCD 闪烁 问题

- 闪烁的现象:

- **原因是LCD的DMA无法及**时为**LCD的FIFO搬运数据**
  - **可以利用FIFO下溢功能**监测
- **解决方法**
  - **提高LCD 在AHB**总线上的优先级（之后我们详细介绍）
  - **降低**帧刷新速率，既在允许的像素时钟频率范围内合理的减小
  - **使用32位**宽的外部SDRAM，**提高**总线的吞吐量
  - **提高SDRAM的**时钟频率，或者使用更快速率的SRAM产品
  - **将固定代**码或者执行频率较高的代码放在SRAM中

总线带宽计算工具

LPC178x Bus Bandwidth on Various LCD Resolutions and Color Depths at Various Refresh Rate	
Bus Clock (MHz):	80
Static External Memory Configuration -	
Bus Width:	32
Read Delay, WAITRD:	1
Dynamic External Memory Configuration -	
Bus Width:	32
Precharge Command Period, t_{RP}:	2
RAS Latency (Active to Read/Write Delay), RAS (t_{RCD}):	2
CAS Latency, CAS:	2
LCD Resolution -	
Horizontal (Pixels):	640
Vertical (Pixels):	480
Refresh Rate -	
Refresh Rate (Hz):	60
LCD Color Depths -	
Color Depth (bpp):	16
Frame Buffer -	
Frame Buffer (KB):	600
LCD Data Rate -	
Data Rate (Mpixels/s):	18.432
Data Rate (MWords/s):	9.216
Data Rate (Mbursts/s):	2.304
Static External Memory Burst -	
Burst (clocks):	13
Dynamic External Memory Burst -	
Burst (clocks):	15
Bus Bandwidth Needed by LCD:	
Static External Memory (%):	37.44
Dynamic External Memory (%):	43.2

- http://www.lpcware.com/content/nxpfile/lcd-bus-bandwidth-calculator-lpc177x8x

LPC4088 LCD AHB 总线优先级

- AHB总线仲裁寄存器 Matrix_Arb （AHB Matrix Arbitration register： Matrix_Arb - 0x400F C188）
- 该寄存器支持多种优先级，如 3 = 最高优先级, 0 = 最低优先级

Bit	Symbol	Description	Reset value
1:0	PRI_ICODE	I-Code bus priority. Should be lower than PRI_DCODE for proper operation.	0x1
3:2	PRI_DCODE	D-Code bus priority.	0x3
5:4	PRI_SYS	System bus priority.	0
7:6	PRI_GPDMA	General Purpose DMA controller priority.	0
9:8	PRI_ETH	Ethernet DMA priority.	0
11:10	PRI_LCD	LCD DMA priority.	0
13:12	PRI_USB	USB DMA priority.	0
15:14	-	Reserved. Read value is undefined, only zero should be written.	NA
16	ROM_LAT	ROM latency select. Should always be 0.	0
31:17	-	Reserved. Read value is undefined, only zero should be written.	NA

- 通过把该寄存器设置0x0000 0C09来提高LCD_DMA的仲裁优先级

EMWIN功能

工具简介

Bitmap converter – 将图像文件转成**C数据文件或者**bitmap**格式**

emWinView – 在PC**上**拟显示效果

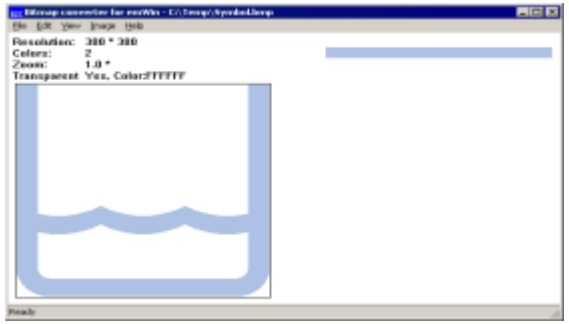
Bin2C – C将各种文件转成数组格式，快速应用到**C文件中**

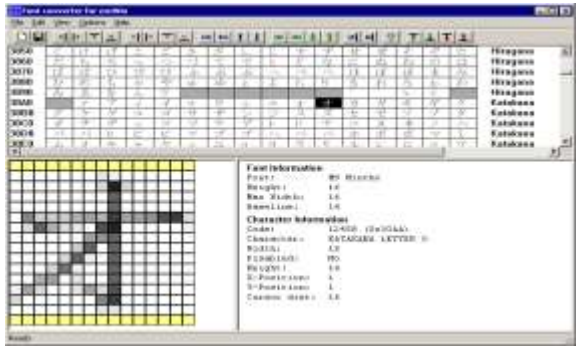
U2C - 将UTF8**格式的文字**转成**C代码**

可选工具:

Font converter – 用于将主机系统下的字体转换为兼容emWin**的格式**.

GUIBuilder – 无需写代码即可基于用用生成应用界面.





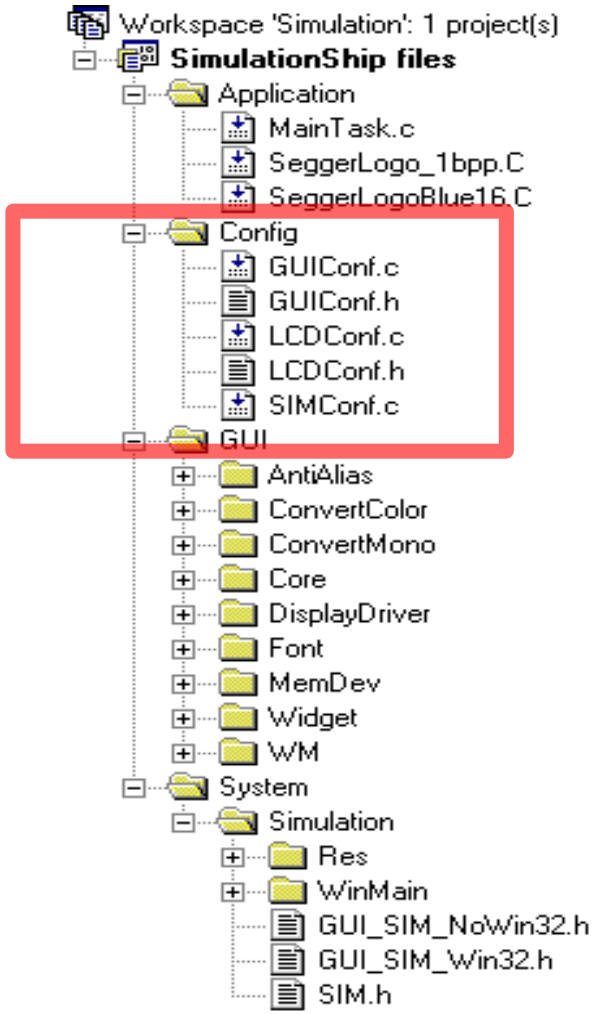
驱动配置文件

编译配置文件 (.h 文件):

- **GUIConf.h –** 选择需要的功能
- **LCDConf.h** - 显示屏幕的驱动配置

Runtime configuration (.c 文件):

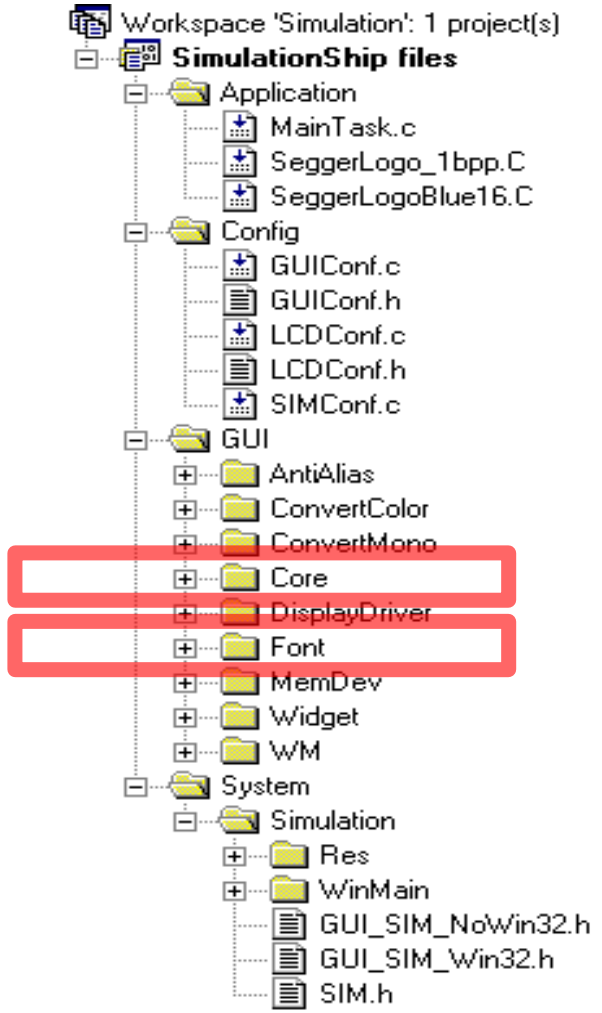
- **GUIConf.c –** 配置动态缓冲区
- **LCDConf.c –** 显示器配置和初始化
- **SIMConf.c –** 模拟器配置
- **GUI_X.c / GUI_X_embOS.c**



核心功能

基本emWin包提供一下功能:

- **支持BMP,GIF,PNG和JPEG格式的数据**
- **支持绘图数据从无地址存储器中获取**
- **支持LTR, RTL 和 双向（ bidirectional ） 文字布局**
- **软件支持透明混合(alpha blending)**
- **支持高亮，光标和动画效果**
- **基本的画 线，椭圆，矩形，弧线和圆形**
- **支持无抗锯齿功能的字体格式**
- **支持不同类型的数据绘图 (十进制，bin格式，16进制和浮点)**
- **多级图像数据缓冲区**
- **虚拟屏幕**
- **支持触摸屏**
- **支持多任务处理**
- **标准字体包(ASCII和ISO 8859-1)**



存储设备管理

存储设备管理器如何工作?

- 存储设备管理器是将绘图数据传入存储设备从而替代传递给LCD屏幕。存储设备是一个独立的存储区域专用于绘图操作.

存储设备管理器怎么用?

- **防止**图像闪烁
- **解**压缩图像的容器
- 缩**放和旋**转图像
- 渐变色处理
- **窗口**动画效果
- **透明效果**

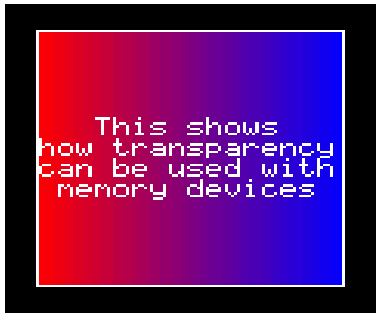
“透明”效果的含义?

存储设备管理器可以感知当前访问的像素是否需要透明处理。这是通过在存储的像素数据上增加额外的标志位达成的，通常存储设备支持1,8和16bpp。

存储设备支持alpha blending么?

是的，32bpp存储设备支持alpha blending, 但不支持“透明”效果处理.

```
#include "GUI.h"


void MainTask(void) {
  GUI_MEMDEV_Handle hMem;
  GUI_RECT Rect = { 10, 10, 109, 109 };


  GUI_Init();
  hMem = GUI_MEMDEV_Create(Rect.x0, Rect.y0, Rect.x1 - Rect.x0 + 1, Rect.y1 -
Rect.y0 + 1);
  GUI_DrawGradientH(Rect.x0, Rect.y0, Rect.x1, Rect.y1, GUI_RED, GUI_BLUE);
  GUI_MEMDEV_Select(hMem);
  GUI_DrawRectEx(&Rect);
  GUI_SetTextMode(GUI_TM_TRANS);
  GUI_DispStringInRect("This shows\n"
                       "how transparency\n"
                       "can be used with\n"
                       "memory devices"
                       , &Rect
                       , GUI_TA_HCENTER | GUI_TA_VCENTER);
  GUI_MEMDEV_Select(0);
  GUI_MEMDEV_Write(hMem);
  while (1) {
    GUI_Delay(100);
  }
}
```

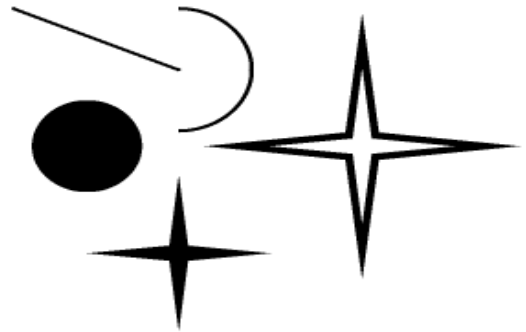


抗锯齿

- 抗锯齿效果可以使曲线或者斜线显示的更加平滑，其原理是讲背景色与显示图形的颜色“混合”让整体的颜色显的更加协调一致.
- emWin支持以下抗锯齿显示
  - **文本 Text**
    需要字体转换工具生成带抗锯齿的字体.
  - **弧形 Arcs**
    `GUI_AA_DrawArc()`
  - **圆 Circles**
    `GUI_AA_FillCircle()`
  - **线 Lines**
    `GUI_AA_DrawLine()`
  - **多边形 Polygons**
    `GUI_AA_DrawPolyOutline()`
    `GUI_AA_FillPolygon()`

注意: 相对于不带抗锯齿效果的字体，抗锯齿的字体显示性能会明显降低

```
#include "GUI.h"

static GUI_POINT _aPoint[] = {
  {  -5,  -5 },  {   0, -50 },  {   5,  -5 },  {  50,   0 },
  {   5,   5 },  {   0,  50 },  {  -5,   5 },  { -50,   0 },
};

void MainTask(void) {
  GUI_Init();
  GUI_SetBkColor(GUI_WHITE);
  GUI_SetColor(GUI_BLACK);
  GUI_Clear();
  GUI_SetPenSize(2);
  GUI_AA_DrawLine(10, 10, 100, 50);
  GUI_AA_DrawArc(100, 50, 40, 40, 270, 450);
  GUI_AA_FillCircle(50, 100, 30);
  GUI_AA_DrawPolyOutline(_aPoint, GUI_COUNTOF(_aPoint), 4, 200, 100);
  GUI_AA_FillPolygon(_aPoint, GUI_COUNTOF(_aPoint), 100, 170);
  while (1) {
    GUI_Delay(100);
  }
}
```



窗口管理器

什么是 **窗口管理器**?

- **按照窗口等级划分管理显示系统**
  每一层都有自己的桌面窗口。每个桌面窗口拥有自己的子窗口的等级分层树.
- **基于系统的回调机制**
  事件产生驱动回调机制，完成窗口的交互.

  所有的绘图操作均有WM_PAINT时间完成。
- **是控件库使用的基础**

所有的控件都是基于窗口管理器（WM）调用的.

基本功能:

- **自动剪裁**
- **自动使用多缓冲区**
- **自动使用存储器设备**
- **自动使用显示驱动的缓冲区**
- **支持窗口的变化**

```
#include "WM.h"

void _cbWin(WM_MESSAGE * pMsg) {
  int xSize, ySize;

  switch (pMsg->MsgId) {
  case WM_PAINT:
    xSize = WM_GetWindowSizeX(pMsg->hWin);
    ySize = WM_GetWindowSizeY(pMsg->hWin);
    GUI_Clear();
    GUI_DrawRect(0, 0, xSize - 1, ySize - 1);
    GUI_DispStringHCenterAt("Window", xSize / 2, 10);
    break;
  default:
    WM_DefaultProc(pMsg);
  }
}

void _cbBk(WM_MESSAGE * pMsg) {
  switch (pMsg->MsgId) {
  case WM_PAINT:
    GUI_DrawGradientV(0, 0, 319, 239, GUI_BLUE, GUI_MAGENTA);
    break;
  default:
    WM_DefaultProc(pMsg);
    break;
  }
}

void MainTask(void) {
  WM_HWIN hWin;

  GUI_Init();
  WM_SetCallback(WM_HBKWIN, _cbBk);
  hWin = WM_CreateWindowAsChild(10, 10, 100, 100, WM_HBKWIN, WM_CF_SHOW, _cbWin,
0);
  while (1) {
    GUI_Delay(100);
  }
}
```

窗口控件(Widget)工具库

窗口控件(Widget) = 窗口**Wi**ndow + 控件Ga**dget**

目前emWin支持的控件包括:

- Buttons, CheckBox, Dropdown, Edit, Framewin, Graph, Header, Iconview, Image, Listbox, Listview, Listwheel, Menu, Multiedit, Progbar, Radio, Scrollbar, Slider, Text, Treeview

仅需一行代码就可以创建一个控件。

创建一个控件有两种方式：

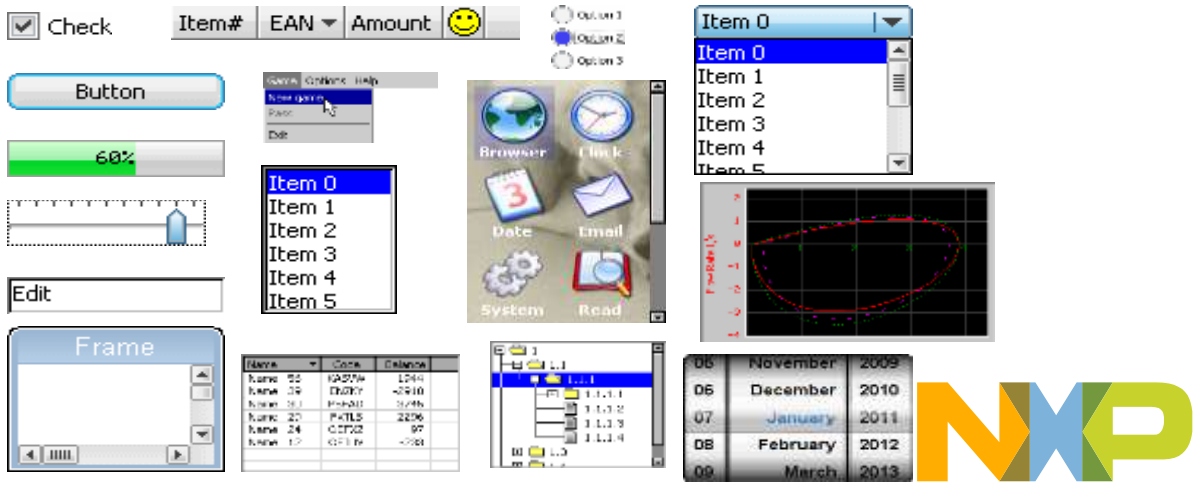
- **直接创建**
  每个控件都已有创建的函数API:
  - `<WIDGET>_CreateEx()`
    创建一个无用户数据的控件.
  - `<WIDGET>_CreateUser()`
    创建一个新的用户数据.
- **间接创建**
  间接创建意味着使用对话框创建函数和 `GUI_WIDGET_CREATE_INFO` 间接创建一个常规的带有指针句柄的控件
  - `<WIDGET>_CreateIndirect()`
    通过对话框功能来间接创建.

直接创建

```
void MainTask(void) {
  WM_HWIN hWin;

  GUI_Init();
  hWin = FRAMEWIN_CreateEx(10, 10, 100, 50, WM_HBKWIN, WM_CF_SHOW, 0, 0, "Window", NULL);
  while (1) {
    GUI_Delay(100);
  }
}
```

间接创建

```
static const GUI_WIDGET_CREATE_INFO _aDialogCreate[] = {
  { FRAMEWIN_CreateIndirect, "Window", 0, 10, 10, 100, 50 }
};

void MainTask(void) {
  WM_HWIN hWin;

  GUI_Init();
  hWin = GUI_CreateDialogBox(_aDialogCreate, GUI_COUNTOF(_aDialogCreate), NULL, 0, 0, 0);
  while (1) {
    GUI_Delay(100);
  }
}
```



皮肤

- 皮肤可以提升控件的外观感受

目前emWin提供两种皮肤:

- **默认皮肤**
  经典类型. 外形可以调用API改变，这些功能API在用户手册的'控件'一章里有介绍.
- **灵活版的皮肤**
  用户轻松设计为自己套路的皮肤外观.

每个控件可以由_SetDefaultSkin配置为默认皮肤:

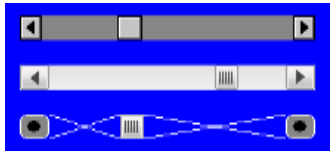
```
<WIDGET>_SetDefaultSkin()
```

每个控件的皮肤样式可以由_SetSkin定义:

```
<WIDGET>_SetSkin()
```

灵活版的皮肤可以通过其属性获取或者配置:

```
<WIDGET>_GetSkinFlexProps()
<WIDGET>_SetSkinFlexProps()
```



```
#include "DIALOG.h"


static int _ScrollbarSkinCust(const WIDGET_ITEM_DRAW_INFO * pDrawItemInfo) {
  switch (pDrawItemInfo->Cmd) {
  case WIDGET_ITEM_CREATE:
    WM_SetHasTrans(pDrawItemInfo->hWin);
    break;
  case WIDGET_ITEM_DRAW_BUTTON_L:
  case WIDGET_ITEM_DRAW_BUTTON_R:
    GUI_SetColor(GUI_GRAY);
    GUI_FillRoundedRect(pDrawItemInfo->x0, pDrawItemInfo->y0,
                        pDrawItemInfo->x1, pDrawItemInfo->y1, 4);
    GUI_SetColor(GUI_WHITE);
    GUI_DrawRoundedRect(pDrawItemInfo->x0, pDrawItemInfo->y0,
                        pDrawItemInfo->x1, pDrawItemInfo->y1, 4);
    GUI_SetColor(GUI_BLACK);
    GUI_FillCircle((pDrawItemInfo->x1 + pDrawItemInfo->x0) / 2,
                   (pDrawItemInfo->y1 + pDrawItemInfo->y0) / 2, 4);
    break;
  case WIDGET_ITEM_DRAW_SHAFT_L:
  case WIDGET_ITEM_DRAW_SHAFT_R:
    GUI_SetColor(GUI_WHITE);
    GUI_DrawLine(pDrawItemInfo->x0, pDrawItemInfo->y0, pDrawItemInfo->x1, pDrawItemInfo->y1);
    GUI_DrawLine(pDrawItemInfo->x0, pDrawItemInfo->y1, pDrawItemInfo->x1, pDrawItemInfo->y0);
    break;
  default:
    return SCROLLBAR_DrawSkinFlex(pDrawItemInfo);
  }
  return 0;
}
```

SAMPLE: Skinning\MainTask_ScrollBarSkin.c

图形编辑预览GUI-Builder

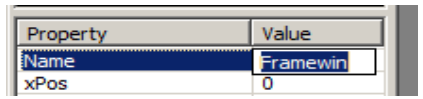
UI-Builder为开发者提供一个无需C语言开发图形界面的工具.

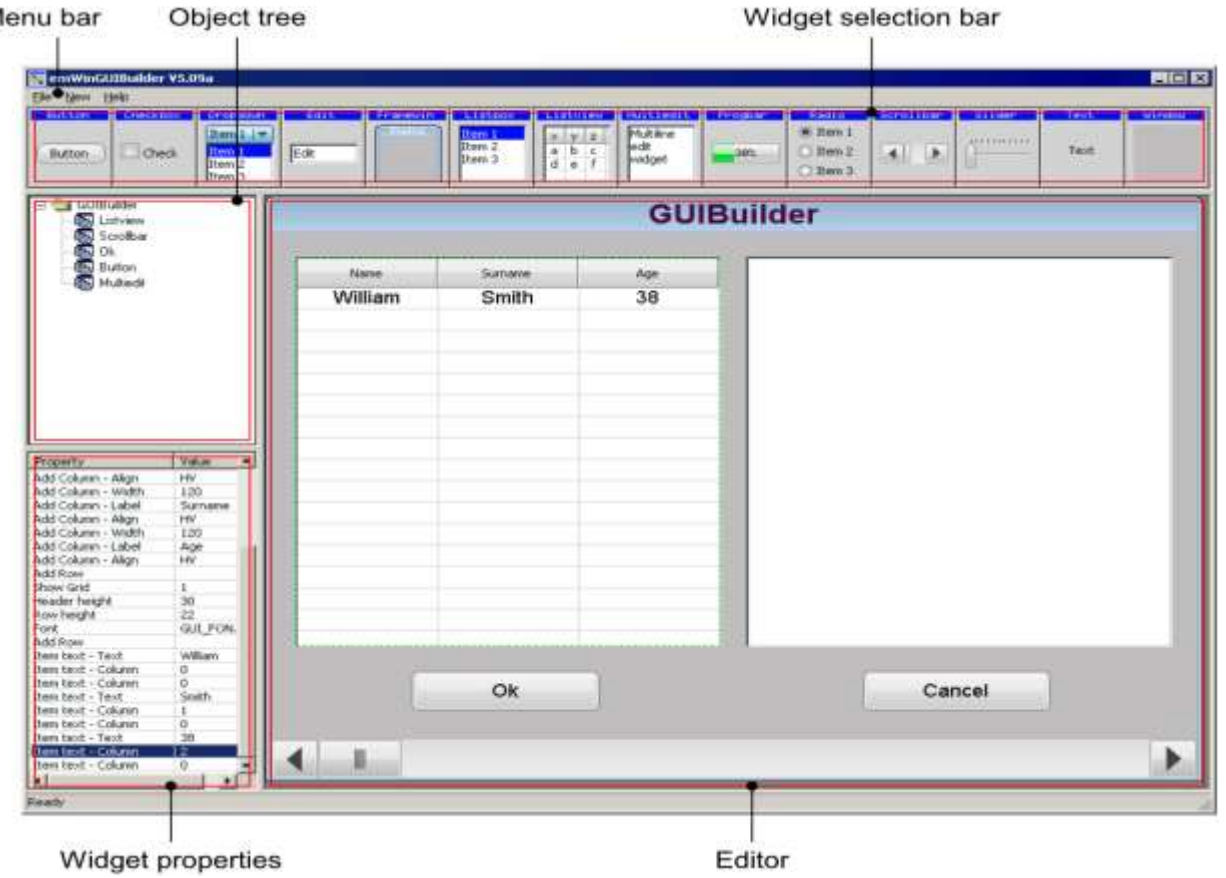
GUI-Builder**使用方法**:

- **检查在** `GUIBuilder.ini`中的**项目路径**
  该文件可以在应用对应的文件夹中找到.
- **启动GUI-Builder**
- **启动**`FRAMEWIN` **或** `WINDOW` **控件**
- **只有**这些控件可以被父级窗口中的对话框所调用.
- **将控件拖拽到父级窗口中**
  控件可以被重新放置或者调整大小，只需要鼠标拖动或者通过属性窗口修改即可.
- **配置控件**
  导航目录菜单提供了可以选择功能.
- **保存对话框界面**
  每个对话框独立保存为一个文件.

文件名称自动产生，由窗口的名称和父级窗口组成，如下：

```
<WindowName>Dlg.c
```





功能:

- **拖拽或者下拉菜单**可以打开一个文件
- 可以模拟**多级对话框**
- 每个对话框保存为一个独立的文件
- **文件名自动生成**

常用对话框

常用对话框包括：

- 消息提示Message boxes
- 颜色选择Color selection
- 文件浏览Exploring a file system

消息提示框MESSAGEBOX

- 仅需要一行代码就可以设置.

选择文件

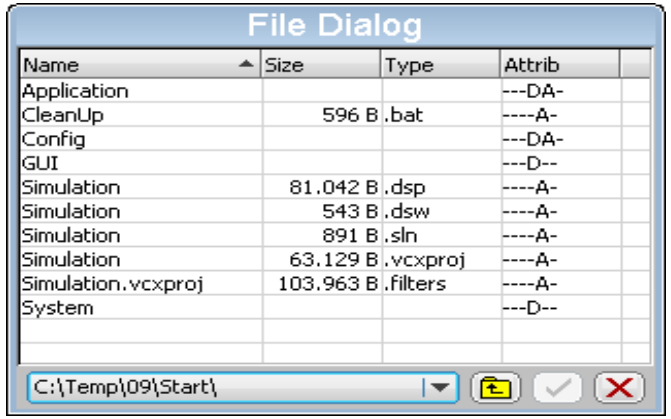
- 嵌入式文件系统浏览器.
- 简介的回调机制来获取文件数据.
- 使用emFile访问文件的例程

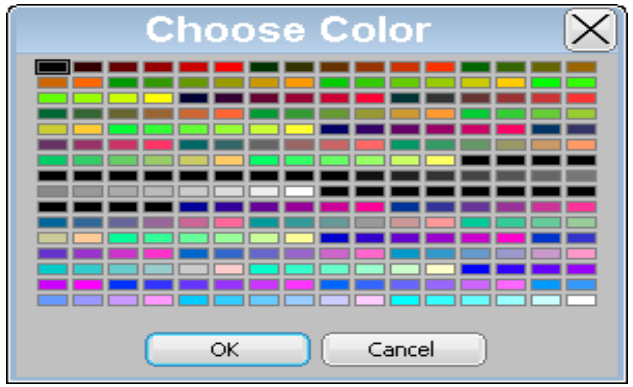
选择颜色

- 应用可以从一个表格中选择颜色.

日历

- 从公历日期表中选择一个日期.





支持NXP的哪些硬件？

- 板级支持包（BSP）支持以下评估板:
  - Embedded Artists LPC4088 显示评估板
  - Keil MCB1700
  - Embedded Artists LPC1788
  - IAR 1788-SK
  - 支持的编译器Keil µVision / IAR EWARM / LPCXpresso / MS Visual C++
- NXP提供的库支持 ARM926/ARM7/Cortex-M0/M3/M4 等内核
- 用户手册/文档:
  - 入门文档 – 如何在NXP微控制器上使用emWin
  - 移植介绍 – 根据硬件和LCD的参数移植一份驱动
  - emWin用户手册 – 超过1000页来介绍emWin功能
- http://www.nxp.com/zh-Hans/pages/emwin-graphics-library:EMWIN-GRAPHICS-LIBRARY?uc=true&lang_cd=zh-Hans
- http://www.nxp.com/zh-Hans/video/introduction-to-nxp-segger-emwin-free-graphics-library:NXP-SEGGER-EMWIN-GRAPHICS-LIB

SECURE CONNECTIONS
FOR A SMARTER WORLD