旷明智能

QuaGL 图形硬件加速
集成指导

文档编号:		**版本号:**	V1.1.0
日期:		**负责人:**	旷明多媒体团队
文档类型:			
版权声明:	本文件系旷明智能科技（“QuaMing”）的原创成果，旷明智能科技依法享有该文件的全部版权。对本文件的全部或部分复制，务必事先取得旷明智能科技的书面许可，并向版权所有者予以明确确认。任何侵犯本公司版权等知识产权的行为，本公司均保留依法追究其法律责任的权利。 在法律许可的范畴内，特此声明，使用前，请审慎阅读合同条款与条件以及相关说明，并严格遵循本文件中的各项说明。旷明智能科技对于不当行为所导致的后果（包括但不限于电压过高、超频或温度过高），不承担任何责任。 旷明智能科技所提供的信息仅作参考或典型应用之用，本文件中的所有声明、信息及建议，均不构成任何明示或暗示的担保。旷明智能科技保留随时变更电路设计和/或规格的权利，且无需另行通知。 客户应全权负责获取实施解决方案/产品可能所需的第三方许可，旷明智能科技不承担任何与第三方许可相关的许可费用或特许权使用费。对于任何由所需第三方许可证所涵盖的事项，旷明智能科技不承担任何保证、赔偿或其他义务。 凡以任何方式直接或间接使用本文件资料者，均视为自愿接受本文件声明的约束。		

修订记录

版本	更新人	修订日期	修订内容
V1.0.0	多媒体Team	2025/12/09	首次起草
V1.1.0	多媒体Team	2026/01/10	完善第3章集成指导

目录

1 概述

1.1 背景

旷明X-AIOS（简称XOS）是基于富瀚、眸芯和旷明芯片的AI操作系统解决方案，其图形渲染加速引擎，向上接入了GUI系统（支持的GUI有LVGL和AWTK），向下包含了芯片的硬件加速器和各种优化，包括图片解码、2D变换、图形绘制、图形显示等。

1.2 目的

本文档提供旷明XOS方案平台的图形图片加速引擎QuaGL的使用方式介绍，方便客户开发和优化GUI应用。同时，XOS里的QuaGL图形加速库，可以脱离GUI系统被客户应用程序直接调用，客户也可以将其集成到自己的GUI系统里，起到性能加速作用，以优化提升客户应用的UI交互体验。

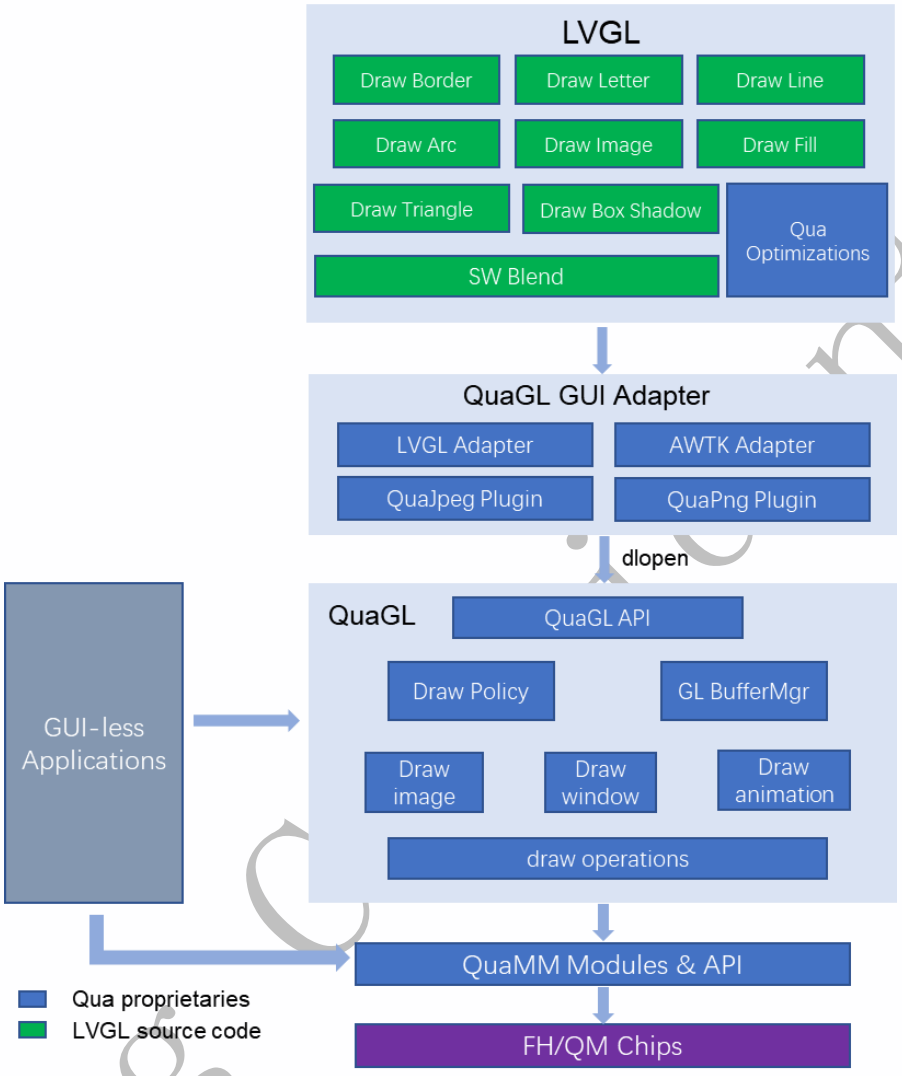
1.3 缩写

本文档涉及的缩写定义描述如下：

缩写名称	描述
XOS	X-AIOS，旷明自研的能适应未来芯片发展的、具备AI能力的、统一框架的智能操作系统，兼容运行于Android、Linux、RTOS、ROS等不同OS内核，拥有独立于客户应用场景的透明设计，为客户提供通用性和产品型SDK，赋能基于芯片构建服务和共创生态。
GUI	Graphic User Interface，图形用户界面
LVGL	Light and Versatile Graphic Library，一种轻量级多功能GUI图形库
AWTK	Toolkit AnyWhere，一种国产开源嵌入式GUI图形库
QuaMM	Qua Multimedia，旷明多媒体引擎框架和API接口，集多媒体播放、录制、音视频处理、图形渲染、手机投屏等功能，集成了旷明、富瀚、眸芯芯片的各种多媒体硬件加速和高度优化的软件扩展媒体库，赋能于芯片的能效发挥和XOS应用的开发优化。
QuaGL	Qua Graphic Library，旷明图形渲染加速引擎

2 总体介绍

2.1 总体架构图



图中的一些组件框图描述如下：

名称	描述
QuaMM Modules & API	QuaMM模块层组件及API接口，包括音频、视频、相机、显示、多媒体处理等模块，部分模块是QuaGL加速引擎的底层基础。
QuaGL	旷明图形渲染加速引擎
QuaGL GUI Adapter	基于QuaGL的GUI适配层，包括图片解码插件。
LVGL：Qua Optimizations	针对旷明XOS自带的LVGL的各种性能优化，直接对LVGL的源码做的修改。

GUI-less Applications	不带GUI的应用程序，可以直接调用QuaMM模块组件和QuaGL进行图形图片的渲染加速

2.2 使用描述

2.2.1 GUI-less Applications 使用

对于不带 GUI 的应用程序，应用程序可以直接调用 QuaMM 模块组件和 QuaGL 的接口实现图片解码和图片图形显示的加速。QuaMM 模块组件的接口使用请参考文档《旷明 QuaMM 多媒体接口使用说明》，QuaGL 的接口使用请参考《旷明图形加速引擎 QuaGL 接口使用说明》。

旷明也即将提供 QuaImagePlayer 引擎，将图片的解码和显示加速封装成 Player 接口供应用程序调用，简化了图片解码显示的整个流程，具体请参考《QuaImagePlayer 接口使用说明》。

2.2.2 GUI-based 的 Applications 使用

本节以基于 LVGL 的应用使用为例来介绍。

如果客户使用旷明整套 XOS SDK，SDK 中已经包含 LVGL9 的 GUI，默认已经开启了芯片支持的所有图形图片加速，且包含了其它系统级和 LVGL 源码级的有利于 GUI 应用运行的加速优化。以下以部分旷明、富瀚、眸芯的芯片为例，加速优化主要体现在：

加速优化项	芯片		
	QM10xH/MC6358/XC01	QM10xD/MC3302/LT00	QM10xV/MC6322/XC00
芯片级绘制加速	Y	Y	N
JPEG 解码加速	Y	Y	Y
PNG 解码加速	Y	N	Y
LVGL 绘制加速	Y	N	Y
LVGL 源码级优化	Y	Y	Y
XOS 系统级优化	Y	Y	Y

如果客户是自己移植的 LVGL，上面表格里的加速优化项默认不包含，需要参照第 3 章节来配置和集成。

3 集成指导

3.1 QuaGL引擎集成

集成前先获取到旷明提供的包含QuaMM和QuaGL组件的图形优化SDK包，组件列表如下：

组件类别	涉及组件的so库	涉及组件的主要头文件
QuaMM Module	quamm.system.<CHIP>_<OS>.so quamm.filter.<CHIP>_<OS>.so quamm.video.<CHIP>_<OS>.so libquamm_common.so libquamm_utils.so libsysutils.so libyuv.so	qua_type.h qua_sys_platform.h qua_mm_common.h qua_mm_system.h qua_mm_filter.h qua_mm_video.h qua_mm_video_common_type.h qua_mm_video_type.h
QuaGL	libquamm_gl.so libquamm_gladapter.so libquamm_utils.so	qua_type.h qua_gl.h qua_gl_buffer.h qua_gl_draw.h quagl_lvgl_adapter.h

以上组件的so库需放到客户板级文件系统的链接寻找路径（$LD_LIBRARY_PATH），LVGL应用编译时只需链接libquamm_common.so、libquamm_gladapter.so和libquamm_utils.so即可。

注意：以上CHIP和OS相关的so组件，需在so动态库的链接寻找路径下新建quamm目录存放，比如存放在/usr/lib/quamm之下。

3.2 初始化

分为QuaMM初始化和QuaGL初始化，需在LVGL的进程启动初始化阶段调用。

3.2.1 QuaMM 初始化

QuaMM初始化使用函数：

```
QUA_S32 qua_mm_init(QUA_BOOL primary_user, QUA_CONST_CHAR *platform,
qua_mm_system_t **system);
```

参数介绍：

- primary_user:
  - true：应用的多媒体功能采用QuaMM多媒体SDK开发

■ false：应用的多媒体功能不采用QuaMM多媒体SDK开发，如果仅使用图形加速引擎，传该值。

- platform: 平台参数，由CHIP和OS组成，如“qm10xh_linux”,具体定义可参考《旷明QuaMM多媒体接口使用说明》第2章节介绍。
- system: 输出参数，包含了QuaMM SDK版本

3.2.2 QuaGL 初始化

QuaGL初始化使用函数：

QUA_S32 **lv_quagl_adpt_init**(QUA_S32 scr_width, QUA_S32 scr_height, QUA_ULONG fb_phy_addr, QUA_VOID_PTR fb_virt_addr, quagl_descriptor_t *descriptor);

参数介绍：

- scr_width: 屏幕宽度
- scr_height: 屏幕高度
- fb_phy_addr：屏幕framebuffer的物理地址
- fb_virt_addr: 屏幕framebuffer的虚拟地址
- descriptor: 图形引擎描述符，需联系旷明获取设置方式

3.3 图片解码配置

需要向LVGL注册加速优化的解码器。在lv_init(void)函数的尾部加入以下代码：

```
#if LV_USE_QUAJPEG
    lv_quajpeg_init();
#endif
#if LV_USE_QUAPNG
    lv_quapng_init();
#endif
```

这样，如果LVGL编译时打开了LV_USE_QUAJPEG和LV_USE_QUAPNG控制开关，应用通过LVGL的image控件显示图片时，会优先使用上面注册的解码器进行图片解码。

相应地，也需在lv_deinit(void)函数里加入各解码器插件的deinit函数。

另外，需要将解码器插件的入口代码加入LVGL编译，包括SDK中提供的quajpeg和quapng代码，放入LVGL源码的编译路径是：

<LVGL根目录>/src/libs/quajpeg

<LVGL根目录>/src/libs/quapng

3.4 绘制接口集成

QuaGL的顶层绘制接口包括：

```
QUA_S32 lv_quagl_adpt_draw_image(const lv_draw_image_dsc_t *draw_dsc,
  const lv_image_decoder_dsc_t *decoder_dsc, lv_draw_unit_t *draw_unit,
  lv_draw_sw_blend_dsc_t *blend_dsc, const lv_area_t *clipped_img_area);
QUA_S32 lv_quagl_adpt_draw_window(lv_draw_unit_t *draw_unit, const
  lv_draw_sw_blend_dsc_t *blend_dsc);
```

这两个绘制函数需要加入到LVGL的绘制代码流程中，参考SDK中提供的patch,将修改移植到以下路径的对应.c文件里：

<LVGL根目录>/src/draw/sw/lv_draw_sw_fill.diff

<LVGL根目录>/src/draw/sw/lv_draw_sw_img.diff

另外，还有一些头文件的修改patch：

<LVGL根目录>/src/draw/lv_draw_buf.h.diff

<LVGL根目录>/src/draw/lv_image_dsc.h.diff

<LVGL根目录>/src/draw/sw/blend/lv_draw_sw_blend.h.diff

3.5 LVGL源码优化

开启NEON优化：

```
#define  LV_USE_DRAW_SW_ASM     LV_DRAW_SW_ASM_NEON
```

图片缓存设置：

```
#define LV_CACHE_DEF_SIZE <合理值>
#define LV_IMAGE_HEADER_CACHE_DEF_CNT <合理值>
```

以上<合理值>表示需要根据系统可用内存空间大小和图片的使用场景综合考虑。

其它LVGL源码级优化旷明会以patch的方式提供，请客户根据具体场景优化的需求向旷明提Case索取。

3.6 XOS系统级优化

XOS系统级优化默认包含在旷明发布的XOS SDK中，如果客户使用的是富瀚或眸芯发布的SDK，请客户根据具体问题的解决需求向旷明提Case索取。