PVR2D Driver

Software Architecture Specification

Filename	:	PVR2D Driver.Software Architecture Specification.doc
Version	:	1.0.16b Not Issued - Live Document
Issue Date	:	30/06/2009
Author	:	PowerVR

Contents

List of Figures

1. Introduction

1.1. Scope

This document provides a top-level view of the PVR2D software architecture with a brief description of the major architectural components. It is assumed that the reader is already familiar with the hardware functionality of MBX, SGX and its derivatives.

2. Product Overview

The components of the PVR2D project are entirely software. The PVR2D project itself is intended to deliver a reference PVR2D driver for MBX, SGX, and their variants, but as this cannot be developed in isolation of its environment the project will also require the presence of other components for service functionality and use by other 3D components.

2.1. Goals and Objectives

The major high-level design goals for this project's software architecture are detailed below.

- The design of the API is such that it reflects the 2D HW cores from Imagination Technologies.
- The API is designed to be used on both MBX and SGX as the have very similar 2D HW cores. Any differences are clearly documented.
- The API includes access to surface and memory allocations as well.
- Platform portability. The code must be easily ported to different operating systems potentially including Linux, RTLinux, Symbian, WinCE, others.
- Does not impede performance.

2.2. Product Environment

The product is designed to support:

- MBX hardware and variants including MBXLite, MBX with VGP, MBX with MMU.
- SGX 535 and its 2D HW core.
- SGX variants with the PTLA 2D Core (SGX 543/544/545).
- SGX family members without a 2D HW Core. Here the API is implemented using 3D HW.

The OpenGL-ES DDK will support at least one available standalone SoC system incorporating an MBX/SGX variant, and a PC Linux environment with a PCI FPGA system.

2.3. Assumptions, Dependencies and Constraints

2.3.1. Assumptions

- None.

2.3.2. Constraints

PVR2D is limited by what the 2D HW can achieve. There are also a number of HW constraints that are taken as requirement for use with the API.

3. Architecture Specification

3.1. Overview

The following diagram 'Figure 1 Architecture Overview' details a top-level overview of the OpenGL-ES reference driver's architecture.

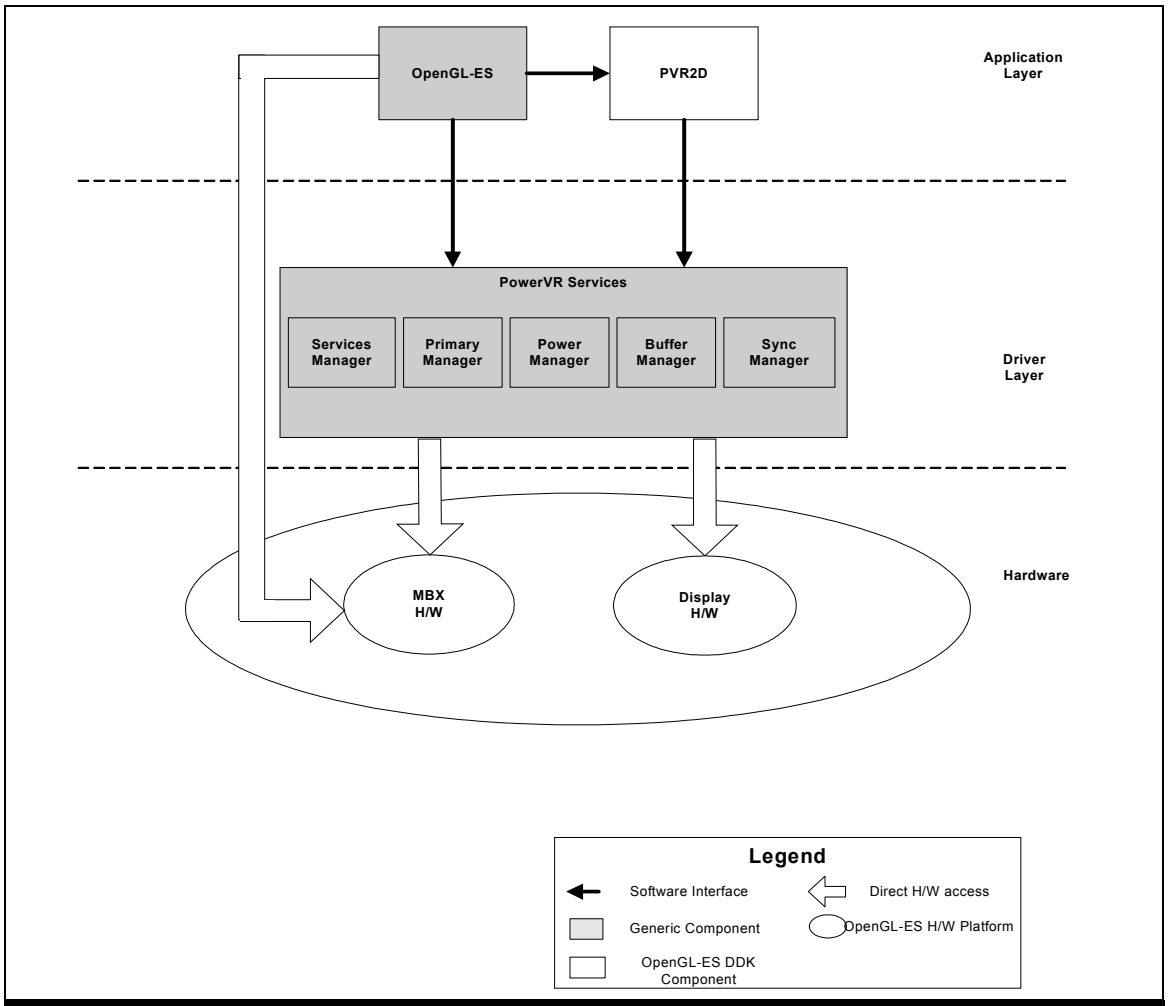

Figure 1 Architecture Overview

The architecture chosen represents a classical layered approach. Elements in grey are provisioned components which are not specifically developed for the PVR2D project.

The architecture chosen meets the design goals defined in section 2.1 (see below for details).

- Platform portable. Platform specific code abstracted to services components to simplify porting to another platform.

3.2. Detailed Component Description

PVR2D	
Component Type	Library

PVR2D	
Purpose	Provides simple abstracted user space API to set display modes, flip and blit.
Functionality	Provides mode setup and blitting functionality.
External Interfaces	This component provides an API as described in PVR2D.Sofware Functional Specification.doc
Internal Interfaces	None
Dependencies and Inter-relationships with other major components	This component uses the PowerVR services component. The 'PowerVR services' component is used for memory management, power management, synchronisation, device initialisation and display setup.

OpenGL-ES	
Component Type	Library
Purpose	Provide the OpenGL-ES Common API interface to 3d hardware
Functionality	Provides 3D acceleration of OpenGL-ES common profile. (See "OpenGL-ES DDK.Software Functional Specification.doc" for details)
External Interfaces	This component provides a 'C' interface as defined by the OpenGL-ES common profile.
Internal Interfaces	None
Dependencies and Inter-relationships with other major components	This component uses the PowerVR services component. The 'PowerVR services' component is used for memory management, power management, synchronisation, device initialisation and display setup.

PowerVR Services	
Component Type	Kernel Module
Purpose	The primary aim of this component is to provide a degree of insulation from platform specific detail and manage shared resources, and provide multi-context and multi-process synchronisation without undue SW serialisation.
Functionality	Services are provided to initialise the device, to manage memory allocations for all device addressable memory, to synchronisation access to shared resources, to synchronise activities on different HW modules, provide power management, resource management, and initialise the device
External Interfaces	IOCTL interface defined in Consumer Services.Software Functional Specification.doc
Internal Interfaces	None
Dependencies and Inter-relationships with other major components	None