Graphics Subsystem in an Embedded World
Integrating DirectFB into a UHAPI platform

Denis Oliver Kropp, DirectFB
Bas Engel, Philips Semiconductors

April 12 2006

Contents

- Changing perspectives
  - Graphics and digital services in an embedded world
  - DirectFB and UHAPI
  - System partitioning
- Integrating DirectFB and UHAPI
  - FrameBuffer vs non FrameBuffer
  - Input handling
  - HW acceleration
- Summary and Q&A

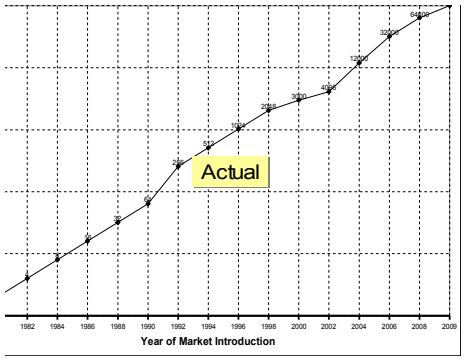
Graphics in an Embedded World

- With trend to digital content more emphasis on
  - The viewing experience
  - New services (VOD, content management, etc)
- Ease of use
  - Key factor in offering new services
- Increasing demand for high quality graphics
  - True Color with alpha channel (32 bit)
  - Slick animations and other eye-candy
  - HD resolutions
- Differentiation and scalability key aspect in future
  - Low end systems can have lower end graphics
  - CE companies require a versatile solution

Ongoing trend: more features, more SW

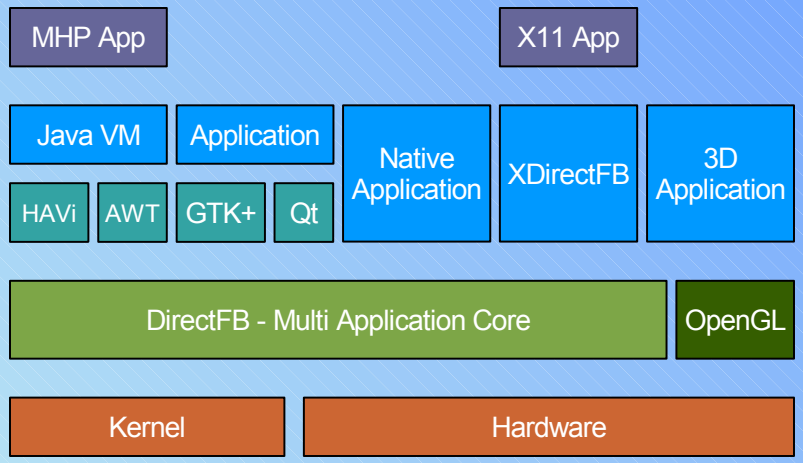
Increasing amount of digital services in future CE products

Consequently, we have an increasing amount of software in CE products



Integration and validation play a key role in fast TTM, so standardized APIs and building blocks are needed

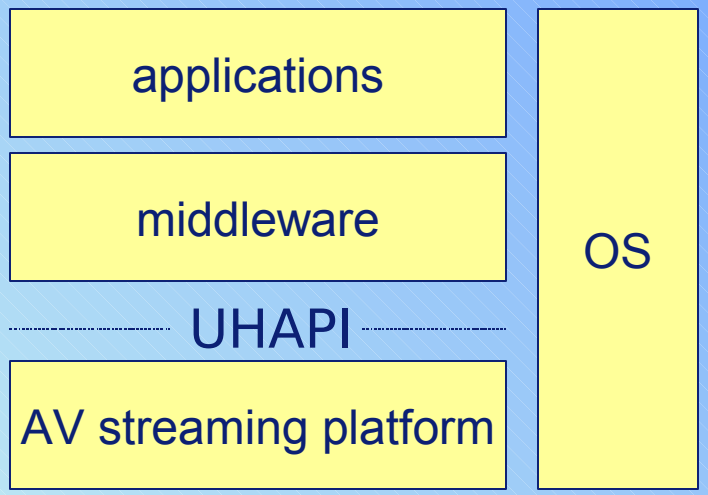
DirectFB



- It's a subsystem that provides
  - Accelerated graphics operations (blitting, scaling)
  - Multiple graphics and video layers
  - Input devices (remote control, local keyboard, etc)
  - Fast anti-aliased text rendering
  - Many pixel formats (ARGB, YUV+Planar)
  - Video memory management (on/off-screen)
- Established technology for embedded appliances
- Scalability allows state of the art desktop and embedded environments
- Adoption of OpenGL makes it competitive to the 3D market
- See www.directfb.org

UHAPI

- It's an API specification for controlling for example
  - Tuner
  - Transport Stream Demultiplexer
  - ATSC Decoder
  - Video Mixer
  - PVR



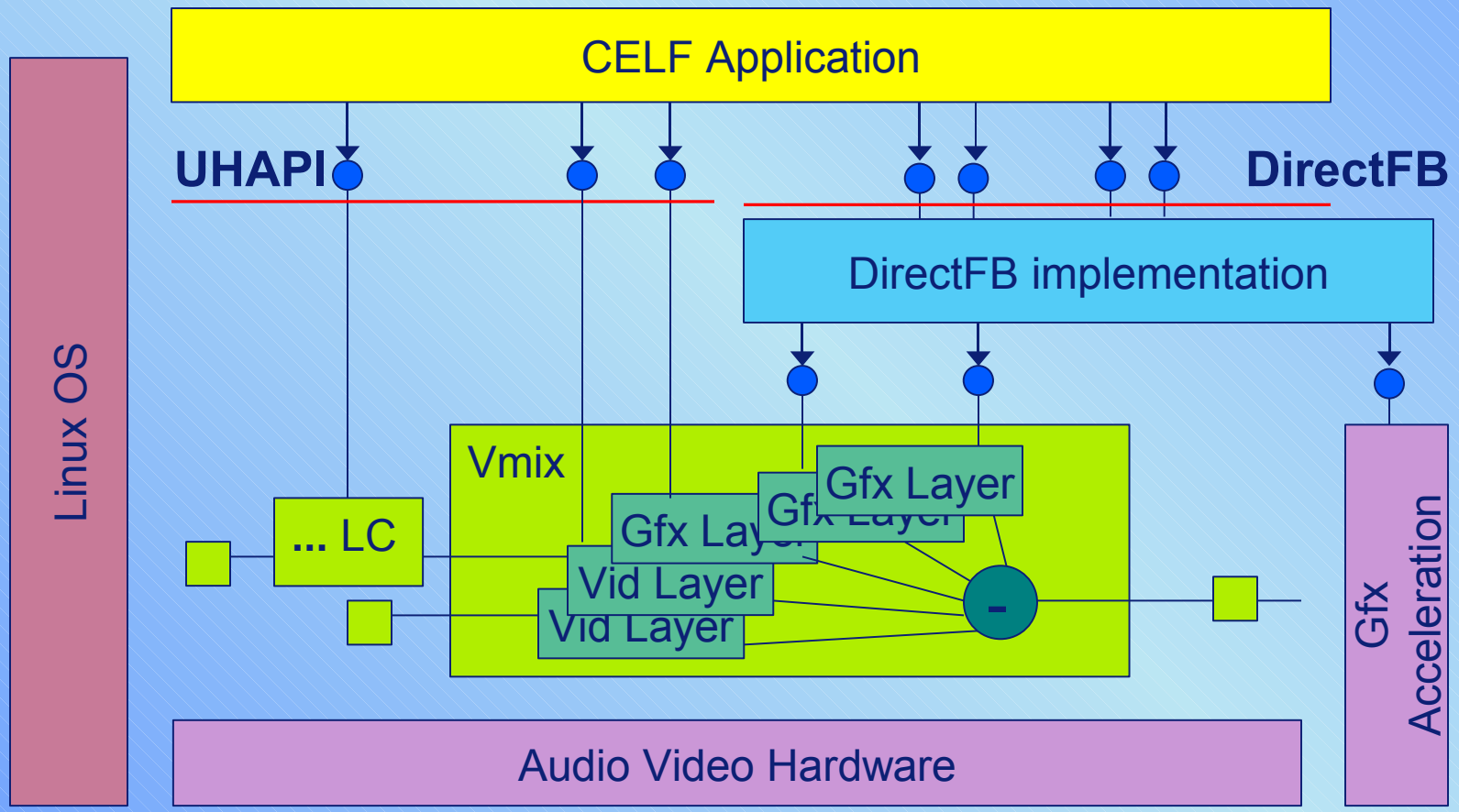
- Hardware and implementation technology independent
- For home based media appliances

- Increasing industry adoption
- The (proposed) CELF AVG specification includes UHAPI
- See www.uhapi.org

Why DirectFB and UHAPI

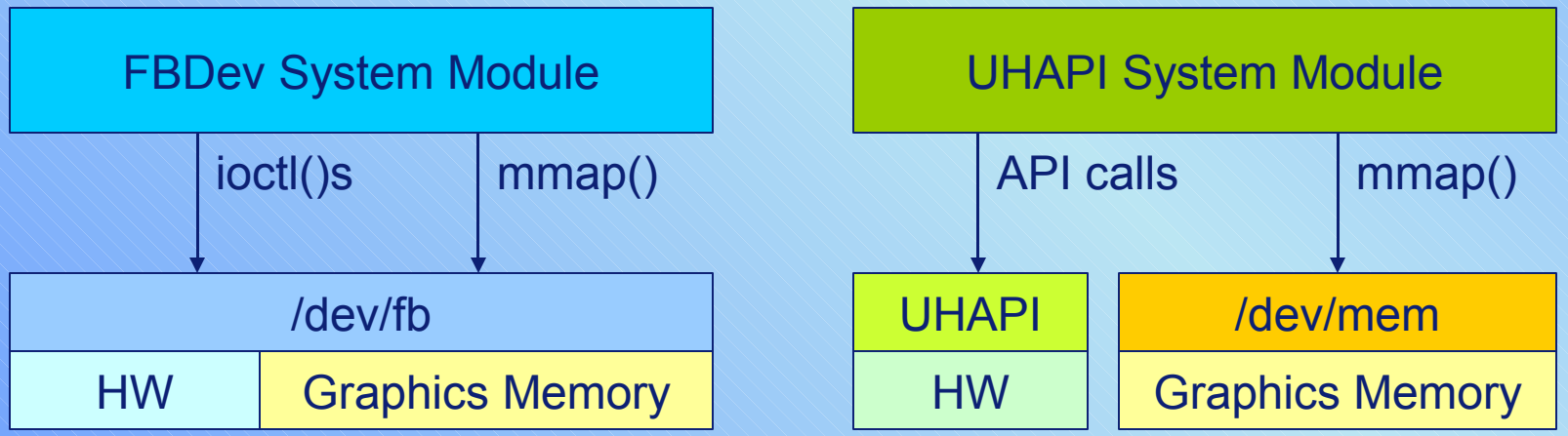
- Use the best of both worlds
  - UHAPI focus: Audio/Video control
  - DirectFB focus: Graphics control
- UHAPI and DirectFB are complementary
  - Except where AV streaming “meets” graphics
    - e.g. both have support for layer mixing and scaling
  - Roles have been defined and discussed at CELF June 2005 Yokohoma
- Optimize system
  - Performance
    - Proper APIs leverage hardware capabilities
  - Portability
    - Due to widely accepted standard interfaces
    - Simplicity in design

Integrating DirectFB and UHAPI

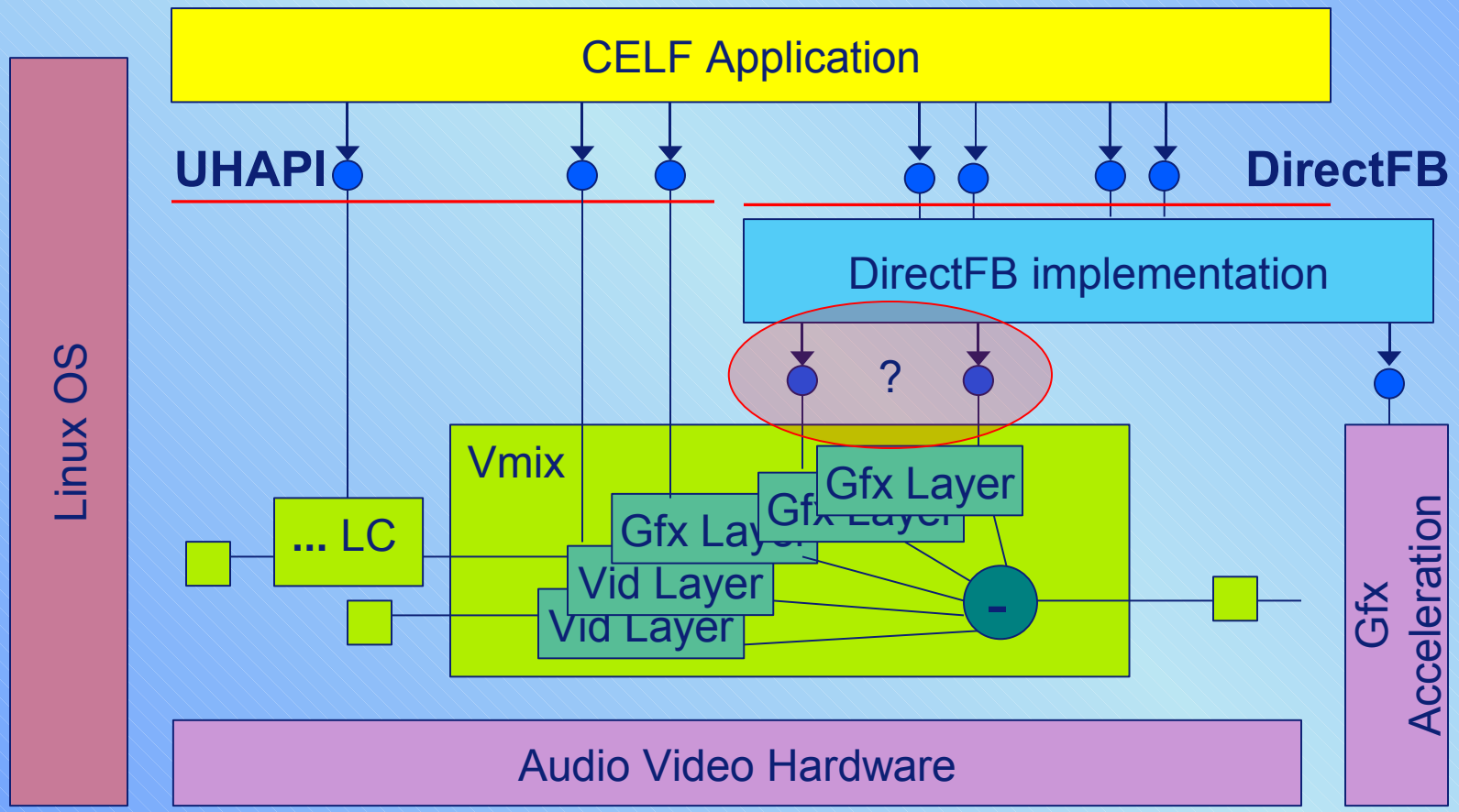


Main challenges

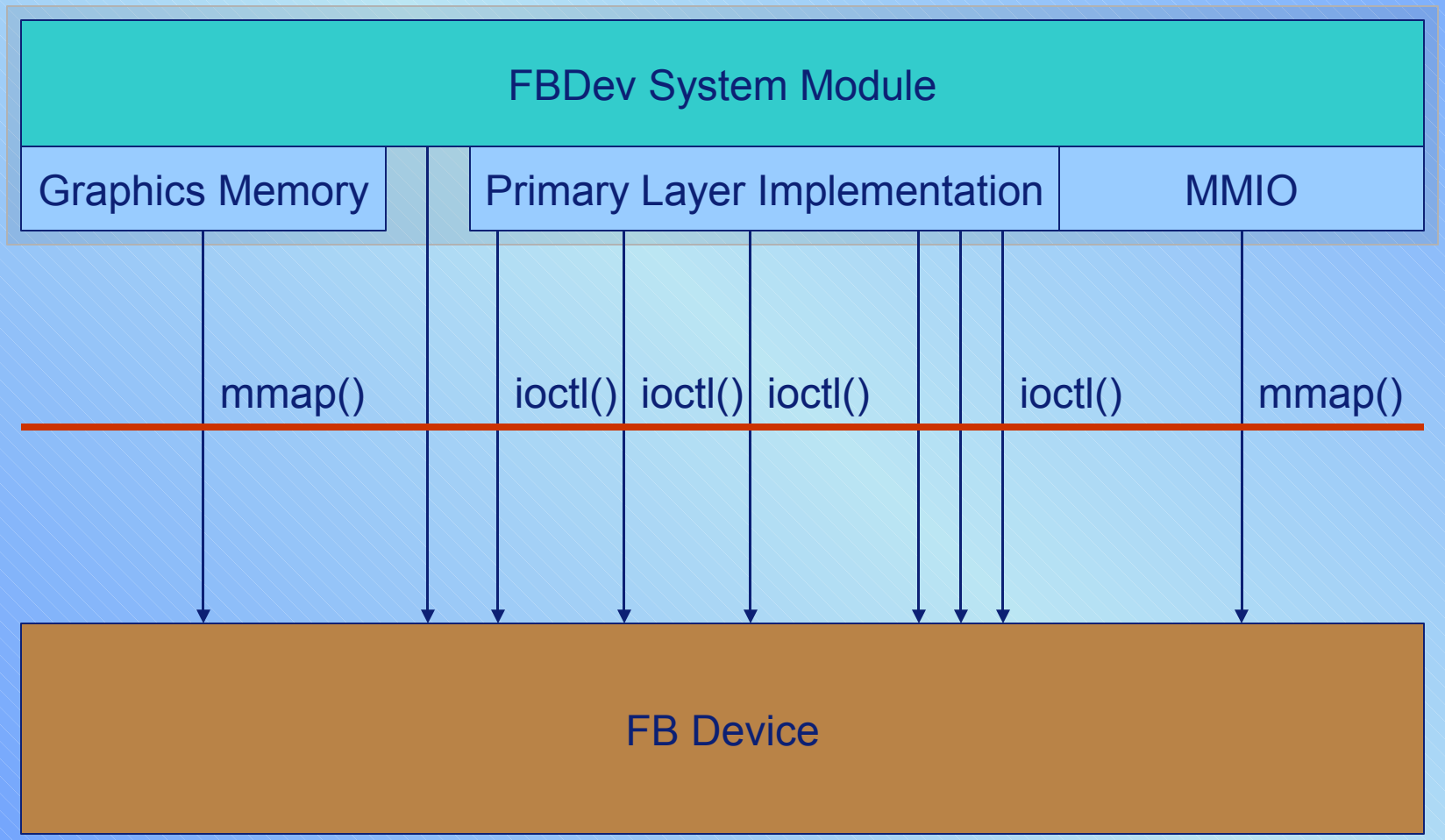
- Writing DirectFB „system module“
- Implementing layer driver on top of uhIVmixLayer
- Allocating and accessing graphics memory
- Adding the accelerated driver



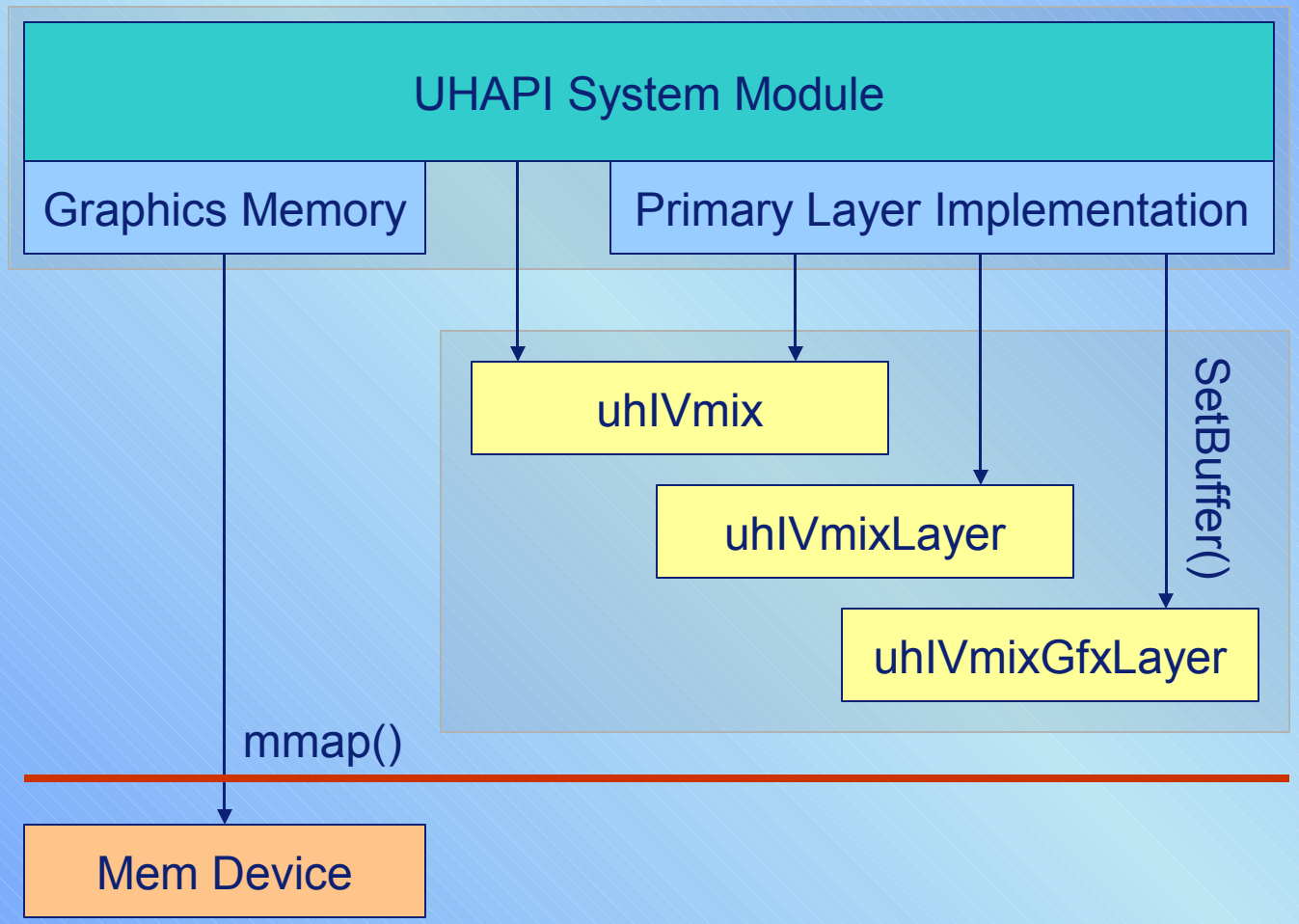
Put another layer in between?



Frame Buffer Device System Module



UHAPI System Module



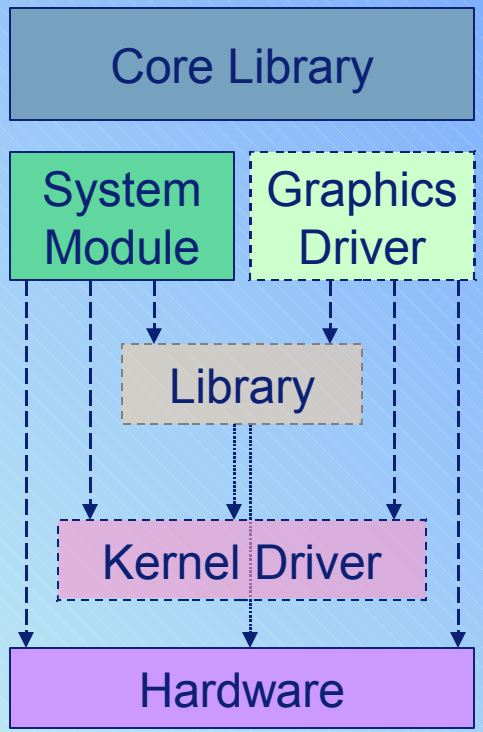
Advantages of UHAPI System Module

Simplified architecture and implementation

- Avoids going through very limited FBDev API layer
  - Full feature set of DirectFB and HW without private extensions
- Flexible resource management
  - Easy match with the memory model of UHAPI and DirectFB
- Much improved interoperability
  - No kernel in between, notifications easily usable
- Modular approach
  - Different subsystems can be exchanged by others

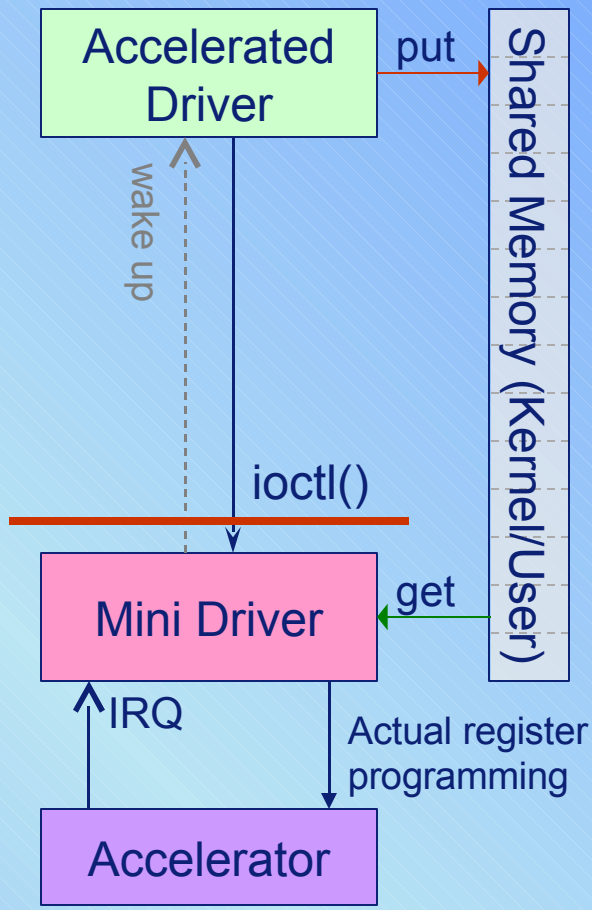
Accelerated Graphics Driver

- DirectFB focusses on hardware acceleration
  - However, optional due to software fallbacks
- Good reasons for acceleration
  - Performance boost
  - Parallel rendering and code execution
  - Keep CPU resources free for other work
  - Reduce latencies, e.g. in the user interface
  - Enable high end user interfaces
- Independent and reusable component
  - Could be tied to a system module though
- Well defined APIs for modules and drivers
  - Freedom of implementation
  - Can reuse existing driver libraries



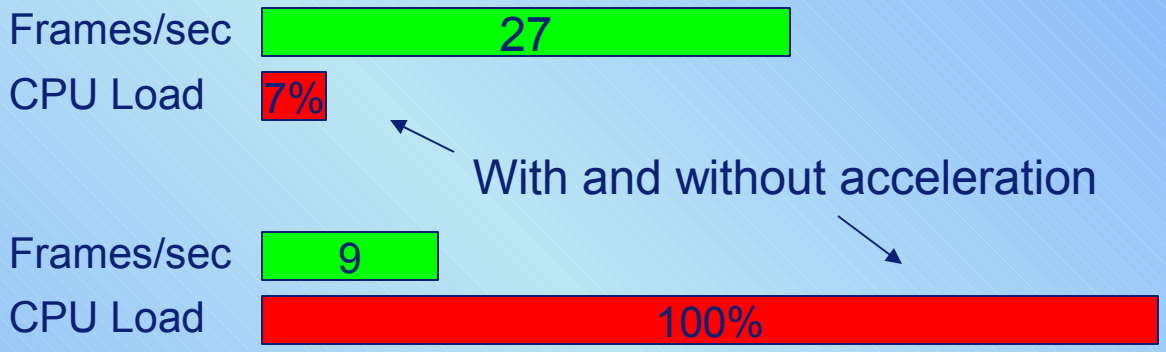
HW acceleration on embedded systems

- Most PC based DirectFB drivers have
  - Direct register programming via MMIO
  - Busy loop polling the state (FIFO, Idle)
- Embedded hardware
  - Busy loop requires too many cycles
    - Use IRQs to signal finished HW operations
  - Command buffer mechanism
    - User space puts packets into the ring buffer
    - Kernel driver does the actual programming
    - ISR processes next command packet
    - Trigger execution of first packet via ioctl when the hardware is idle
    - Idle wait in user space when buffer is full
    - Can wait for finishing commands using ioctl and serial number of packet

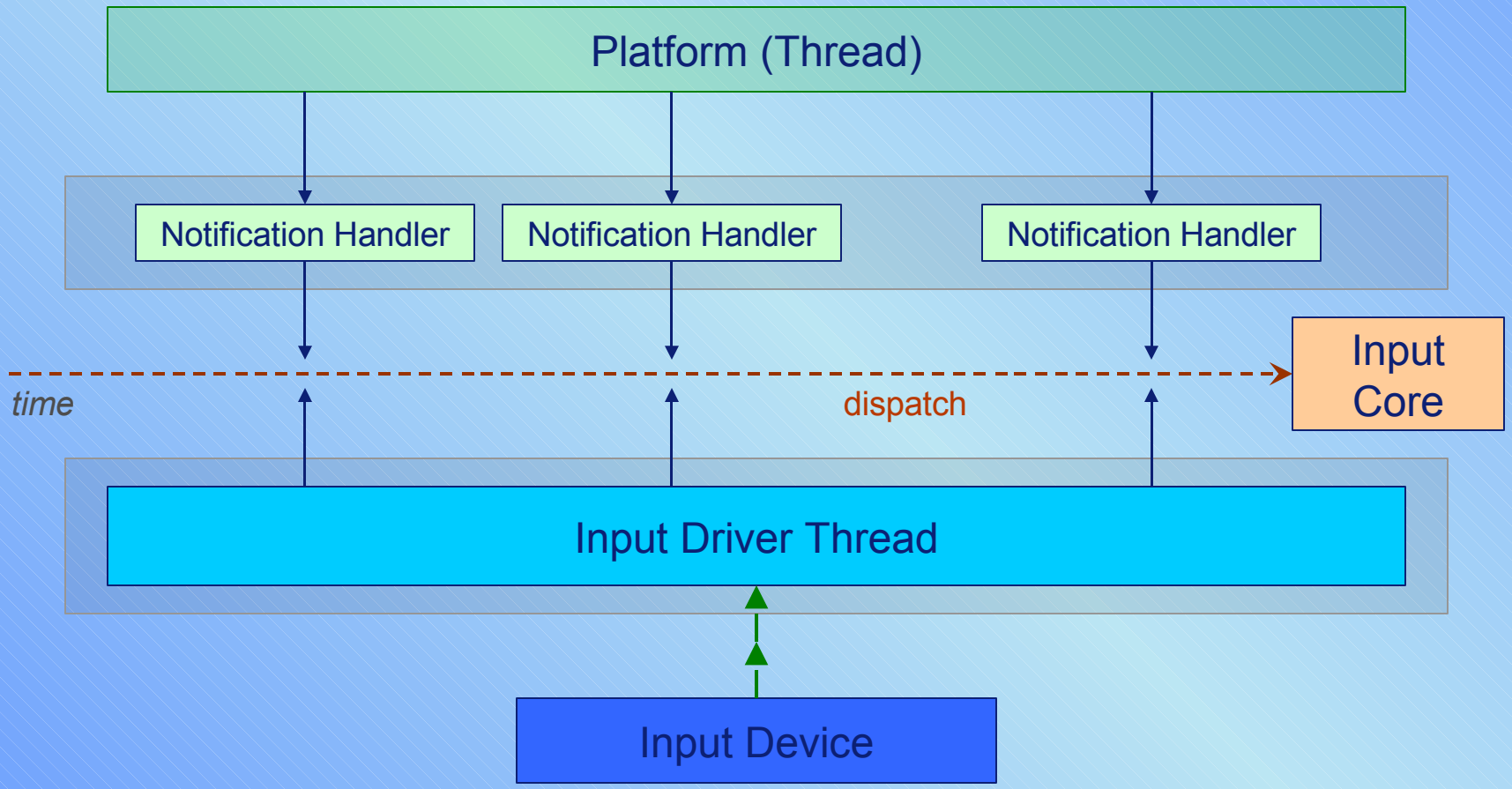


Benchmarking

- Application benchmark
  - Realistic values for specific application scenarios, e.g. sprite animation
- Synthetic benchmark
  - Raw numbers, e.g. reflecting memory bandwidth or other bottle necks



Input handling



Summary

- DirectFB and UHAPI are a powerful combination
  - To leverage the value of state of the art digital processing and new viewing experiences
- Not restricted by existing solutions
  - Simplified architecture and implementation
- Modular approach
  - Solution not specific to DirectFB and UHAPI
- Improved application integration
  - Due to standard infrastructure and proper hardware abstraction

Any questions