TOSHIBA
Leading Innovation

Common Software Models and Platform for Cell and SpursEngine

July 11, 2008

Seiji Maeda

Advanced SoC Development Center
Toshiba Corporation, Semiconductor Company

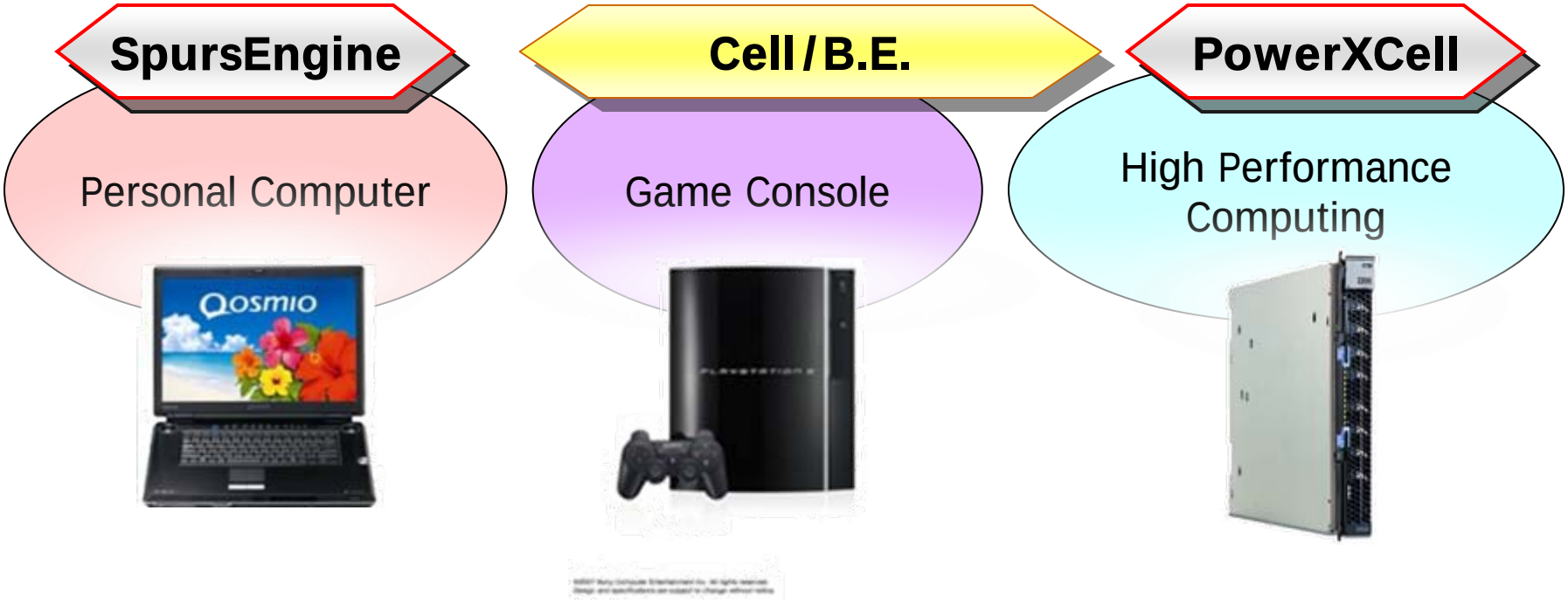
Outline

- Cell Family and New Topics
- Introduction of SpursEngine
- Emerging Use Cases
- Common Models and Platform
- Call for Participation

SpursEngine is a trademark of Toshiba Corp.
SpursEngine is also known as Toshiba Quad HD Processor.

Continuous Expansion of Cell World

- We already have 3 chips to encourage the Cell World
  - Cell/B.E. is shipped in PS3 by SCEI and QS2x by IBM
  - SpursEngine by Toshiba for PC market is announced
  - PowerXCell and QS22 by IBM are also announced

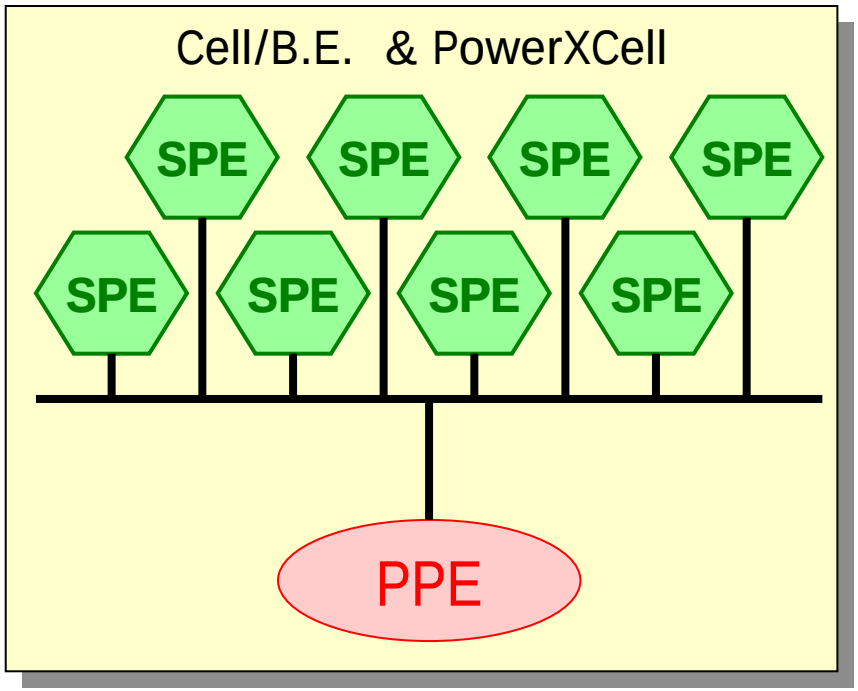


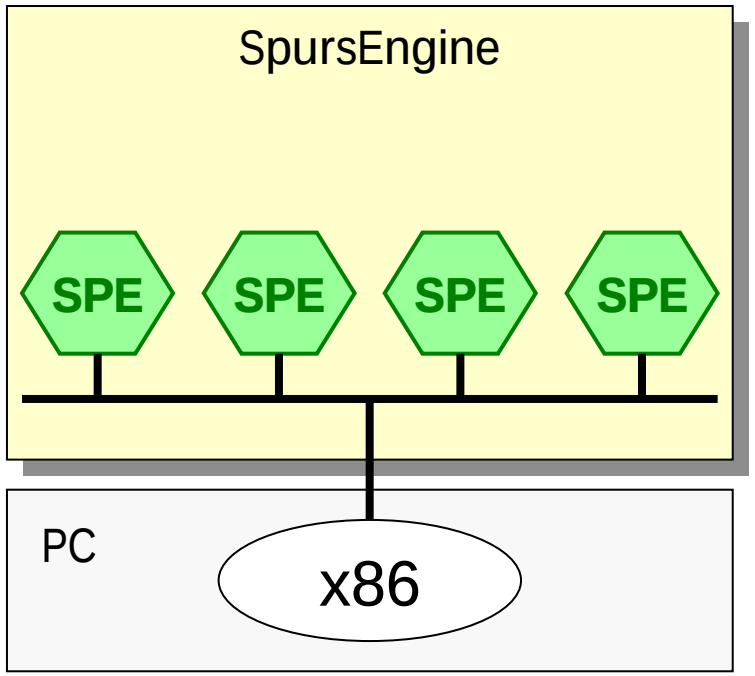
Cell/B.E. is a trademark of Sony Computer Entertainment Inc. PowerXCell is a trademark of IBM.

SPE Accelerates General Purpose Cores

- PPE in Cell & PowerXCell is accelerated by 8 SPEs
  - 6 SPEs are available for user program in PS3
- x86 in PC is accelerated by 4 SPEs in SpursEngine
  - SpursEngine is mainly for an accelerator of PC platform

➔SPE is a key component of Cell family processors





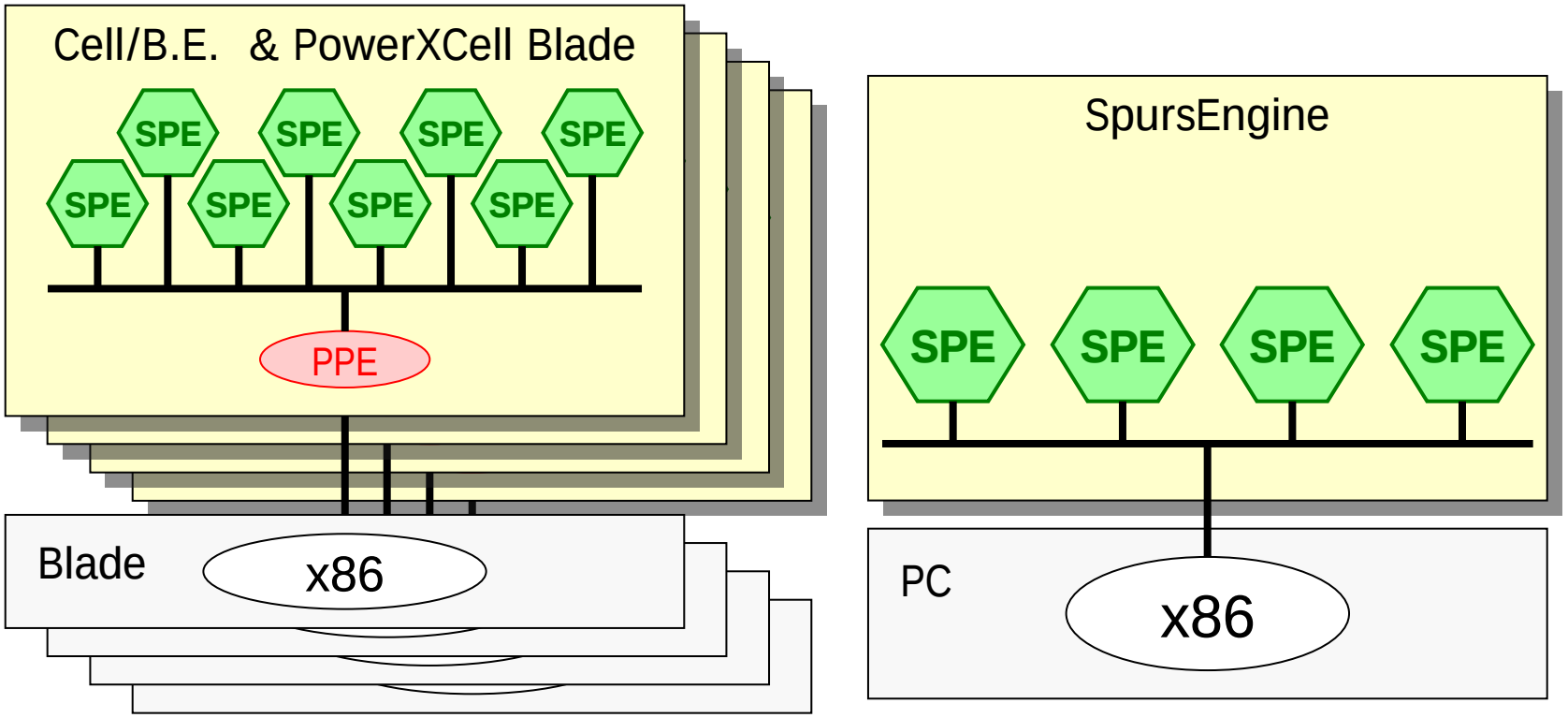
Connectivity of Other Systems

- Cell World must not become an isolated world
  - We should create natural path to connect Cell processors and existing other systems
- Considering general systems is very natural for both system suppliers and end users
  - In desktop computing market, PC is virtually the standard platform around the world
  - Even in HPC market, it is natural that users would like to connect super computers to general servers and clients
  - End users might have at least one or more PCs before getting Cell-accelerated systems

SPE Extends General Systems

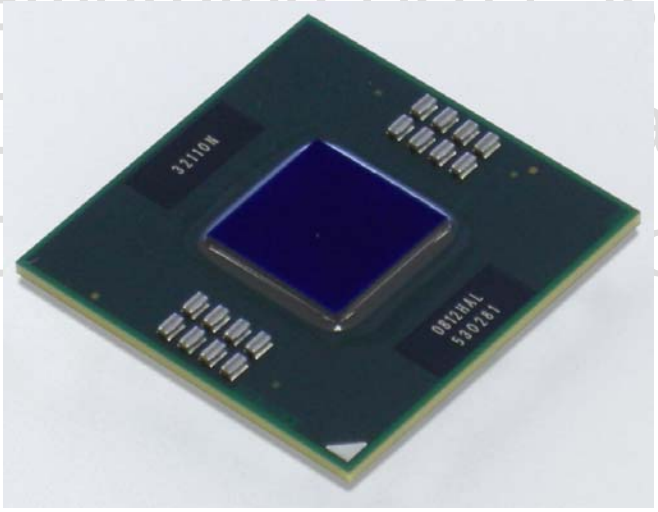
- Cell & PowerXCell Blades (QS2x) can be connected to x86-based blades
- SpursEngine can be connected to x86-based PC

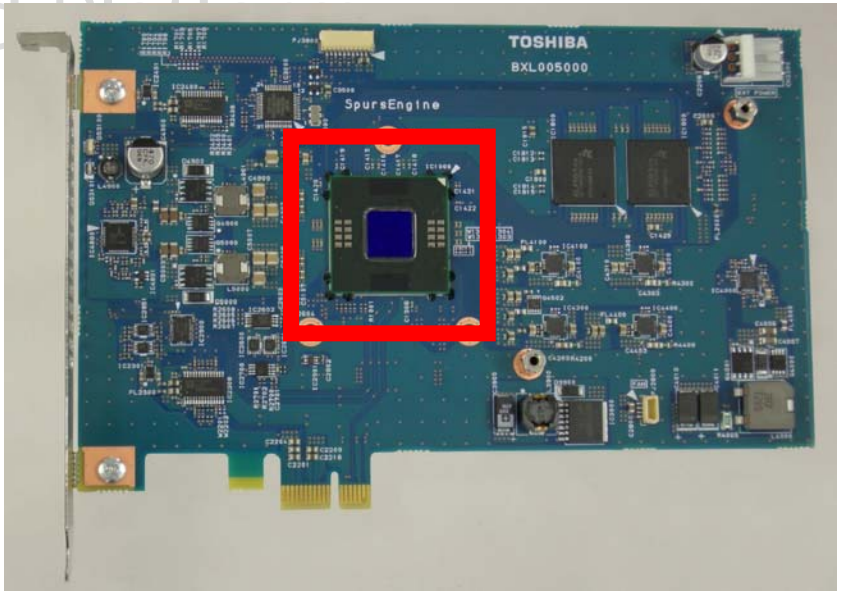
➔ Connectivity to general systems becomes more important



Outline

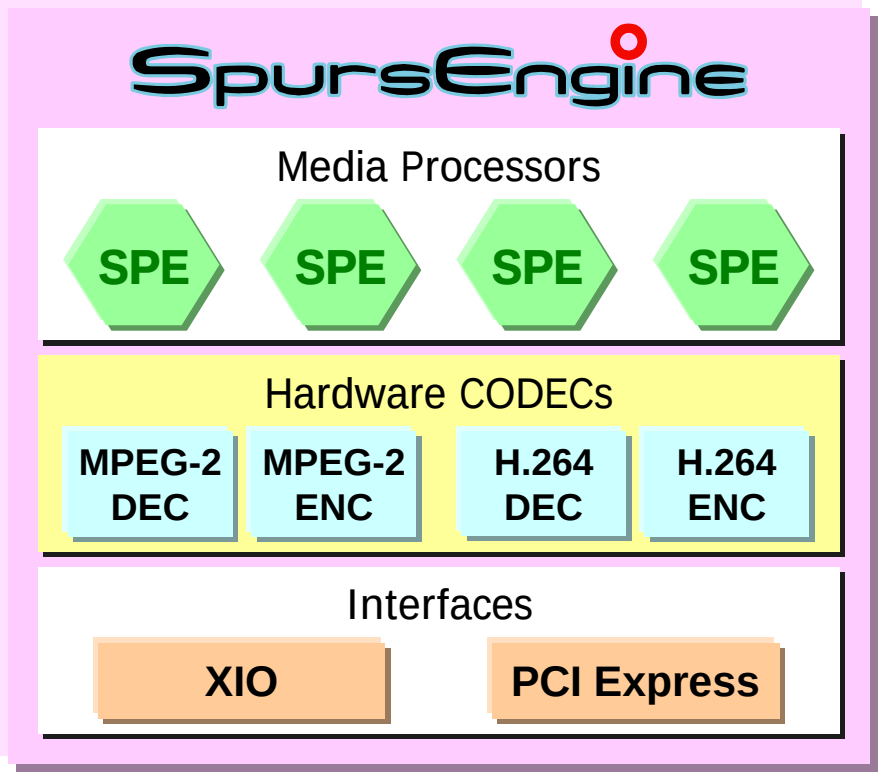
- Cell Family and New Topics
- Introduction of SpursEngine
- Emerging Use Cases
- C and Platf
- C on

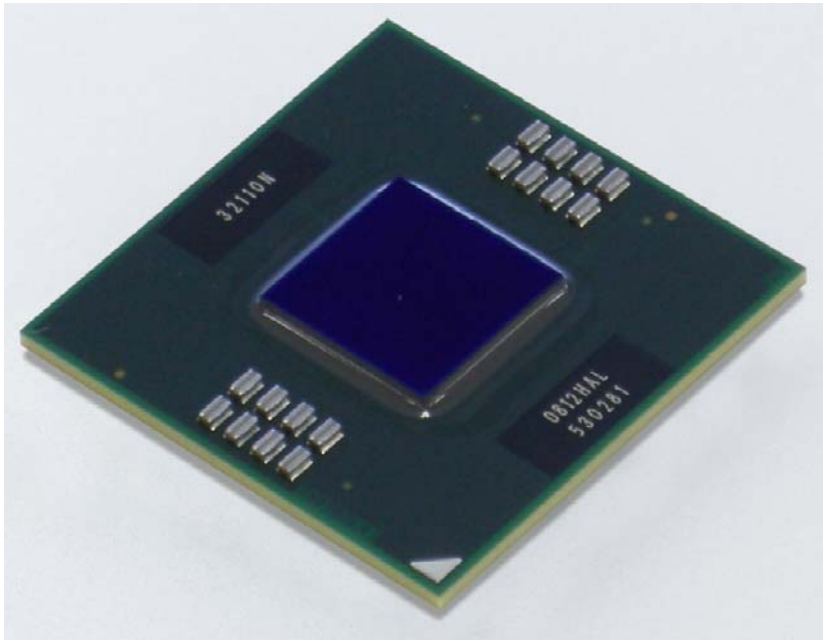




SpursEngine Concept

- Media processing accelerator derived from Cell/B.E.
  - 4 SPEs in SpursEngine are fully compatible to those of Cell/B.E.
- Supports Full-HD digital life efficiently
  - Hardware CODECs of standard formats are embedded in the same chip

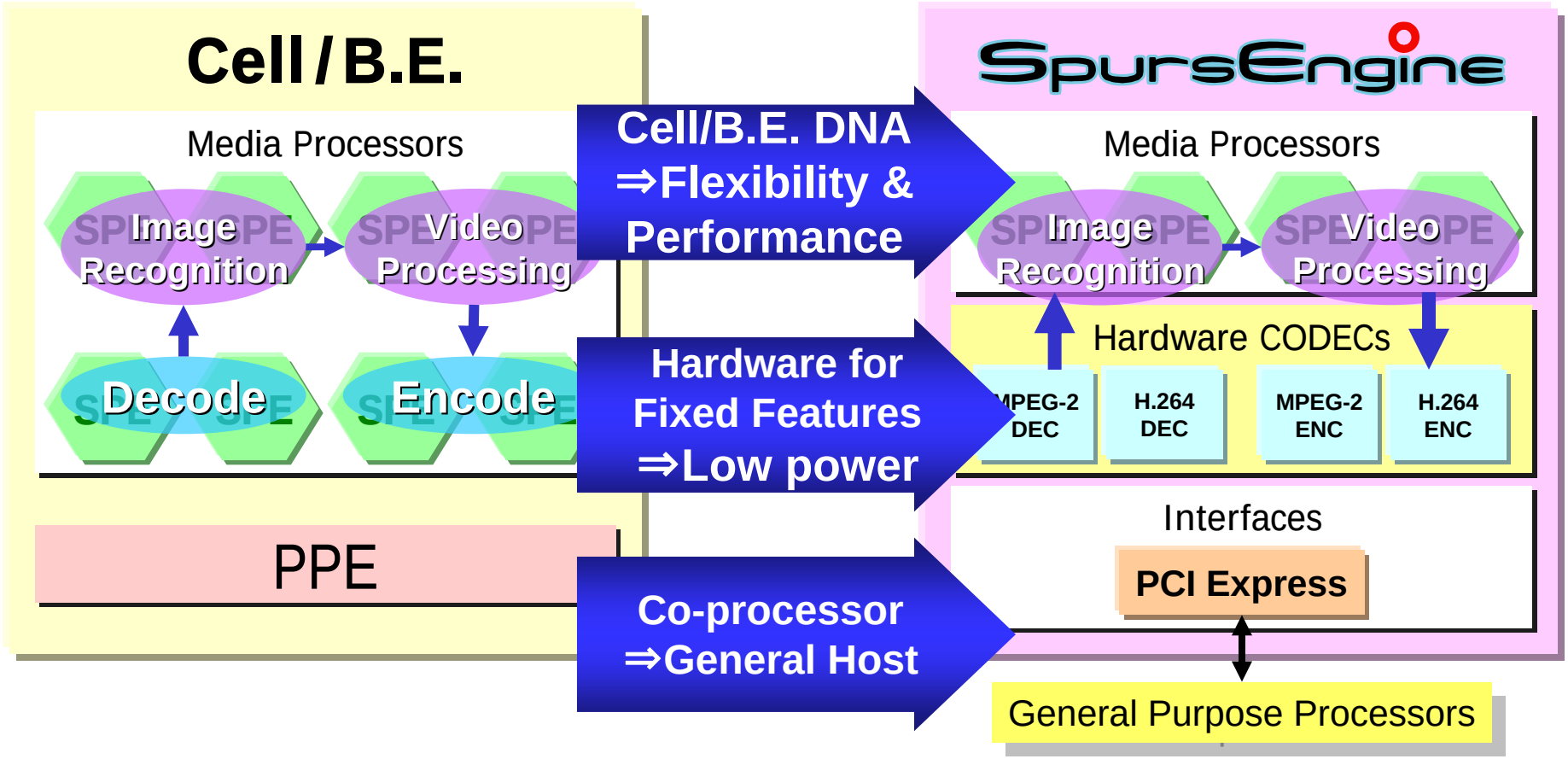




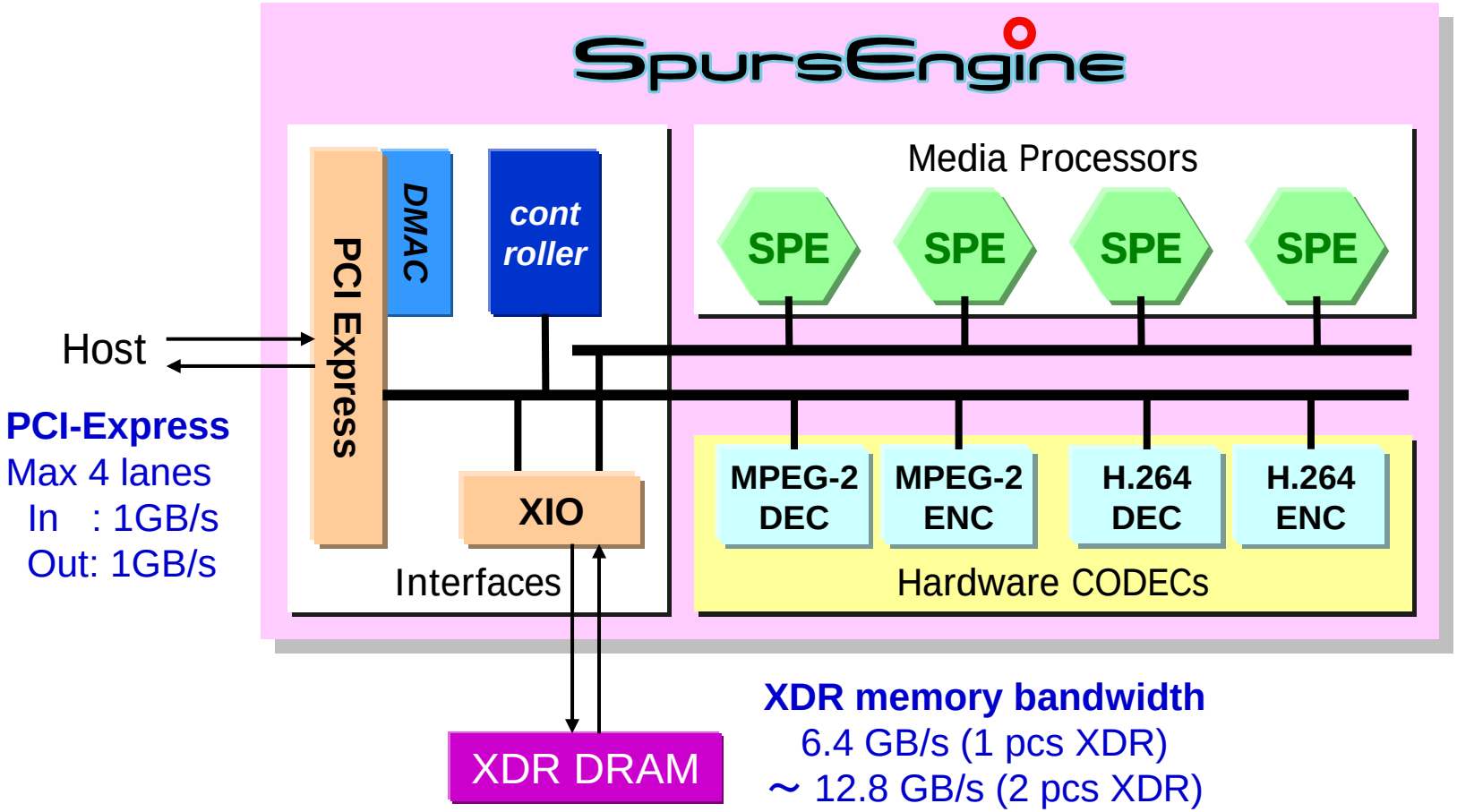
XIO is a trademark of Rambus Inc.

SpursEngine and Cell/B.E.

- SpursEngine is optimized to HD contents and PC market inheriting Cell/B.E. DNA



SpursEngine Block Diagram

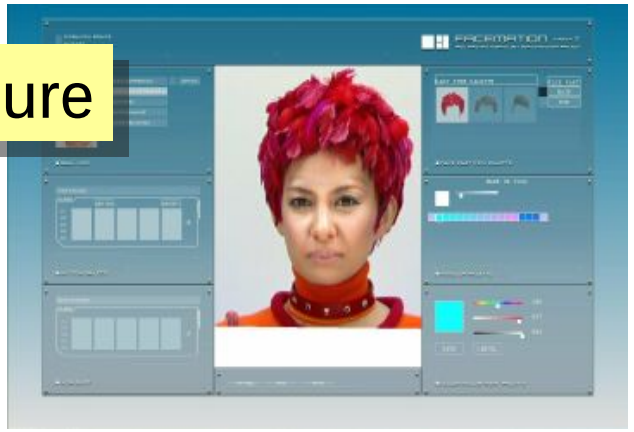


XIO and XDR are trademarks of Rambus Inc.

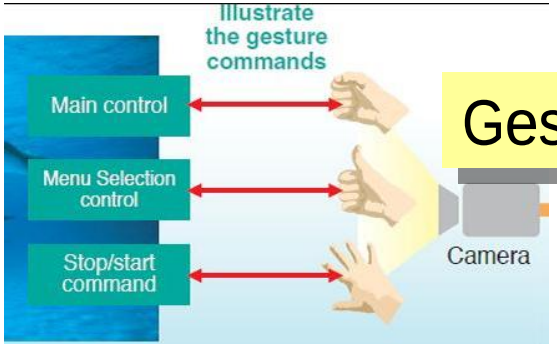
Existing Applications

- SpursEngine realizes these applications

Face Motion Capture

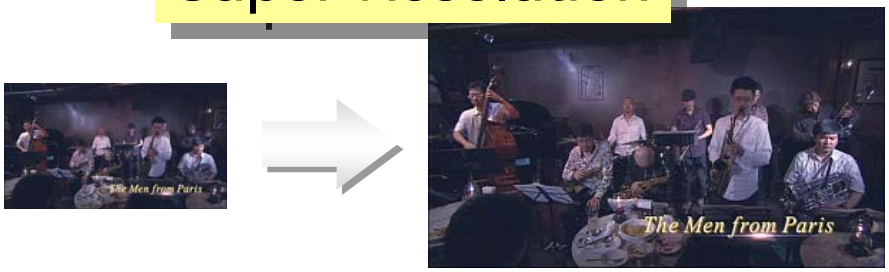


Gesture Recognition



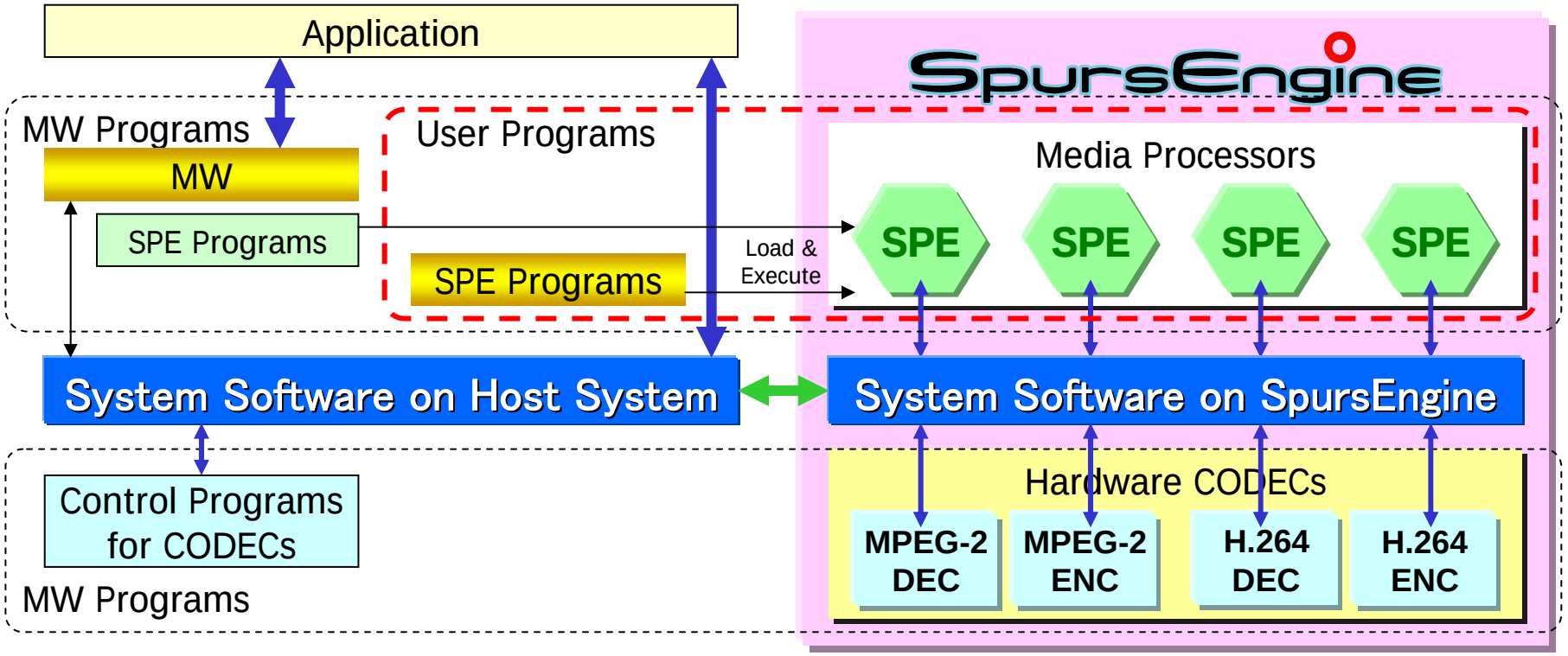
Face Detection

Super Resolution



Software Platform

- For easy to use, application can use middleware
  - Middleware internally utilizes SPEs and HW CODECs
- For programmability, application can execute own programs
  - Using system software, SPEs can be controlled directly



Compatibility of SpursEngine

- SPU ISA is completely identical
  - Both Cell/B.E. and SpursEngine support SPU ISA 1.0
  - SPU MFC (DMA engine) is also compatible
  - SPE compilers can be used for SpursEngine
- Performance is different
  - SPEs on Cell/B.E. run at 3.2 GHz, but those of SpursEngine run at 1.0 - 1.6 GHz
  - Cell/B.E. has 8 SPEs, but SpursEngine has 4 SPEs
- General purpose processor might be different
  - It is natural for us to consider x86-based systems in addition to PowerPC-based systems
- SPE programs might be better when they are scalable and independent from general purpose processors

Outline

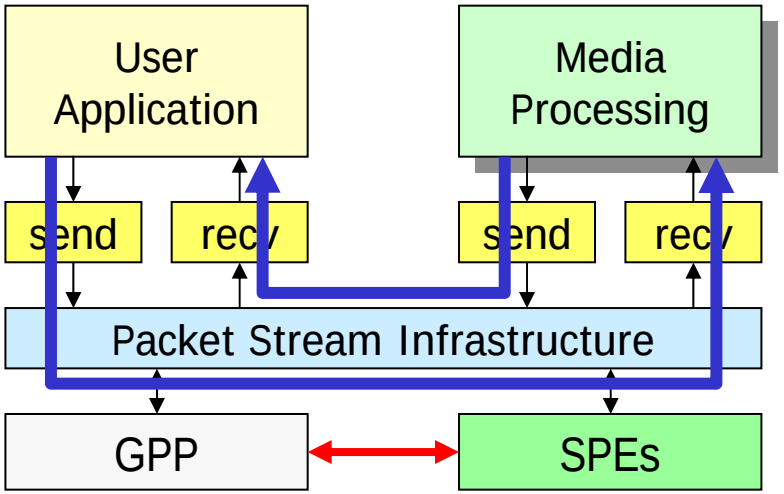
- Cell Family and New Topics
- Introduction of SpursEngine
- Emerging Use Cases
- Common Models and Platform
- Call for Participation

Expansion of Possible Platforms

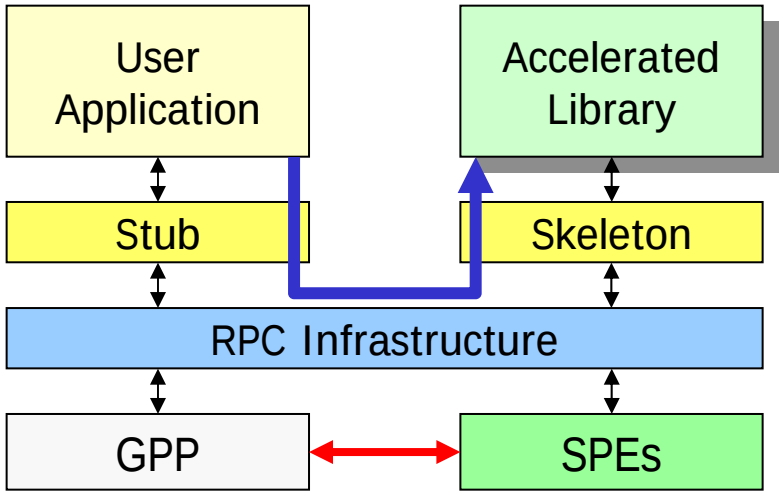
- According to the extension of Cell family, we should regard the following platforms as possible platforms in addition to Cell/B.E. standalone
  - General Purpose Processor(GPP) + Cell/B.E.
  - GPP + SpursEngine
- Possible OSes might include Linux/x86 for servers and Windows for PCs
  - Linux/PPC is no longer the only platform for Cell
  - SPE programs should be host-OS independent
- Possible interconnects between GPP and Cell derivatives might be narrower than internal buses
  - Ethernet might be possible in addition to PCI-express
  - Programmer should consider the narrow path

Use Case #1: SPEs as Accelerators

- Heavy media processing such as HD CODECs and pattern recognition might be implemented by powerful SPEs
  - Data packets streams between GPP and SPEs
- Heavy library might be off-loaded to accelerated library powered by SPEs
  - User application calls accelerated library with RPC fashion



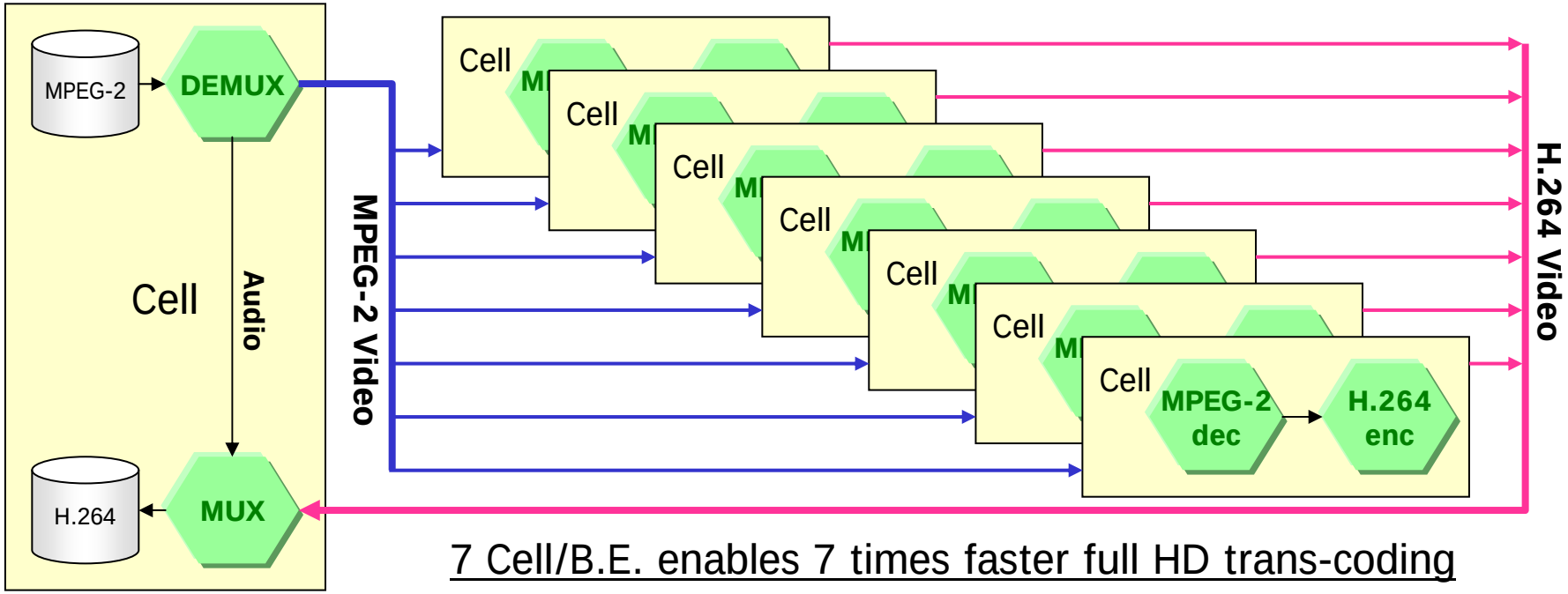
Packet Stream



RPC

Example: High Speed Trans-coding

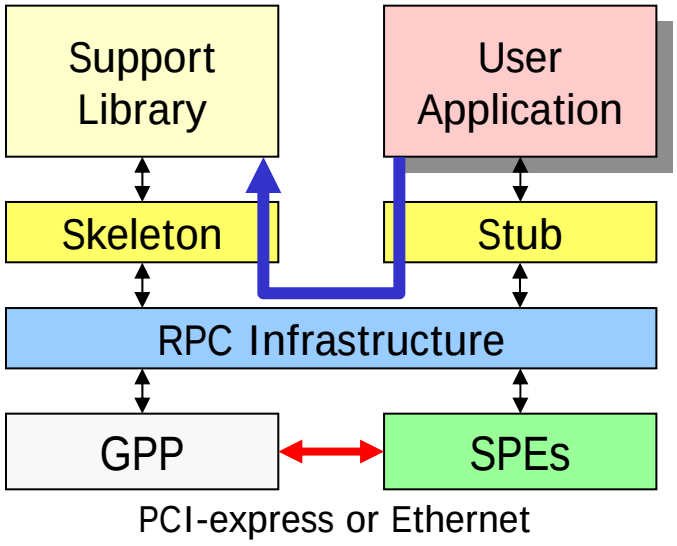
- Accelerate trans-coding using multiple Cells
  - Client sends MPEG-2 video packets to multiple Cell servers and gather H.264 video packets from the servers
  - Cell servers trans-code video from MPEG-2 to H.264
- All video data are transmitted via Ethernet



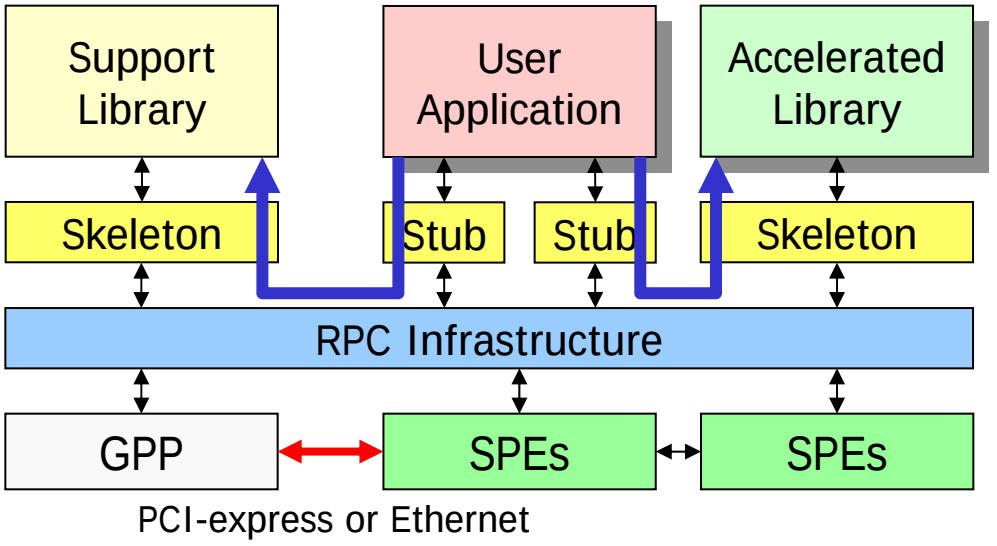
7 Cell/B.E. enables 7 times faster full HD trans-coding

Use Case #2 : SPEs as Main Processors

- Applications might be designed and implemented on SPEs and use GPP for system services
  - Applications might be ported from GPP and some library might remain on GPP for efficiency
- Hybrid system might offer library on both GPP and SPEs
  - Some libraries can be accelerated by SPEs



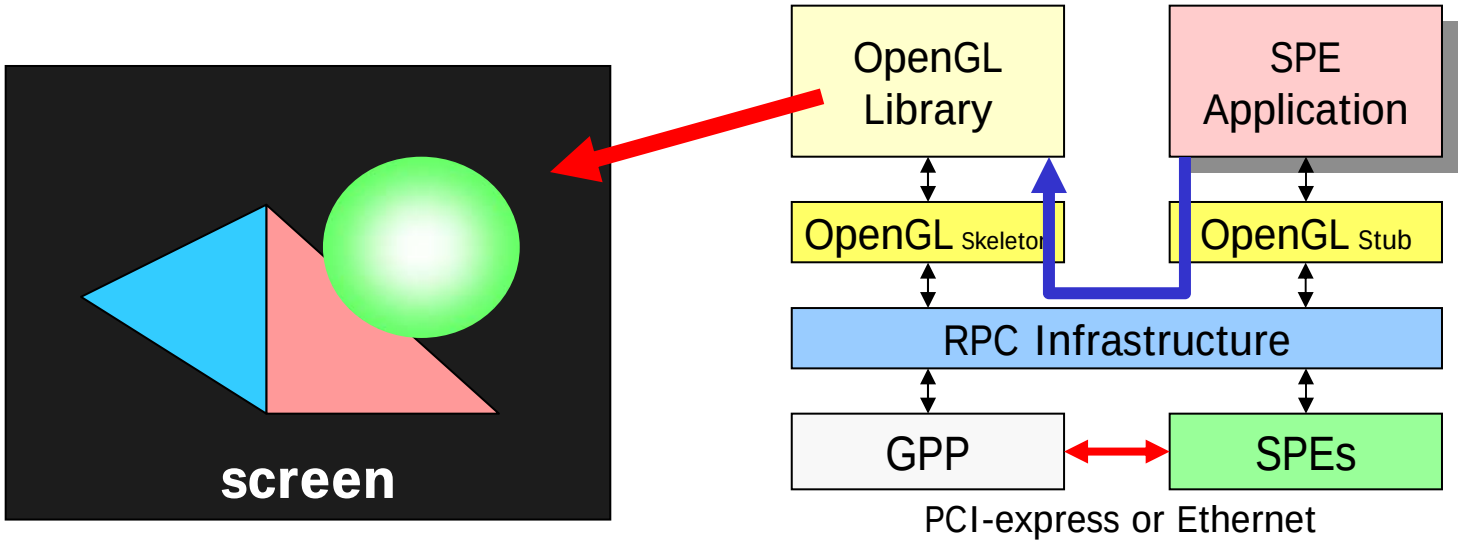
User Application on SPEs



Hybrid

Example: SPE Application Using OpenGL

- SPE Application directly call OpenGL API
  - GPP handles the requests and draw image on screen
- It is possible that OpenGL is implemented on other SPEs
  - GPP only handles I/O requests to frame buffer



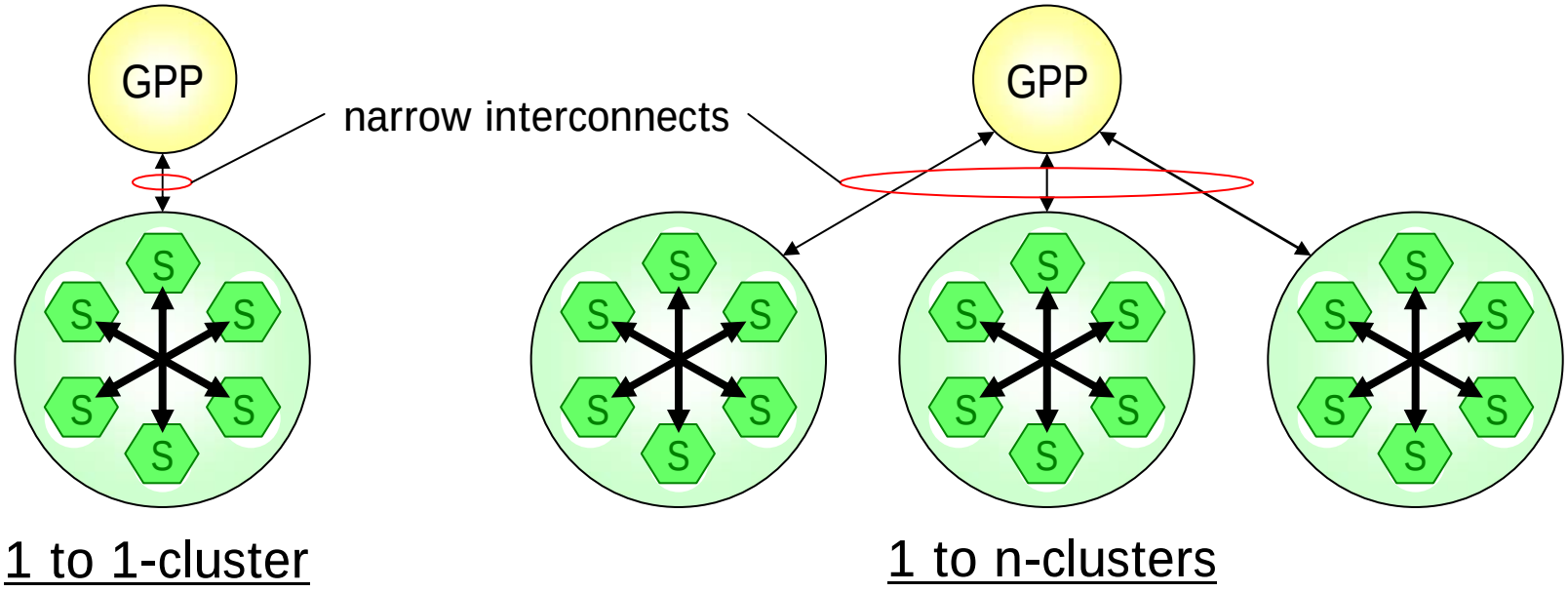
OpenGL Application on SPEs

Outline

- Cell Family and New Topics
- Introduction of SpursEngine
- Emerging Use Cases
- Common Models and Platform
- Call for Participation

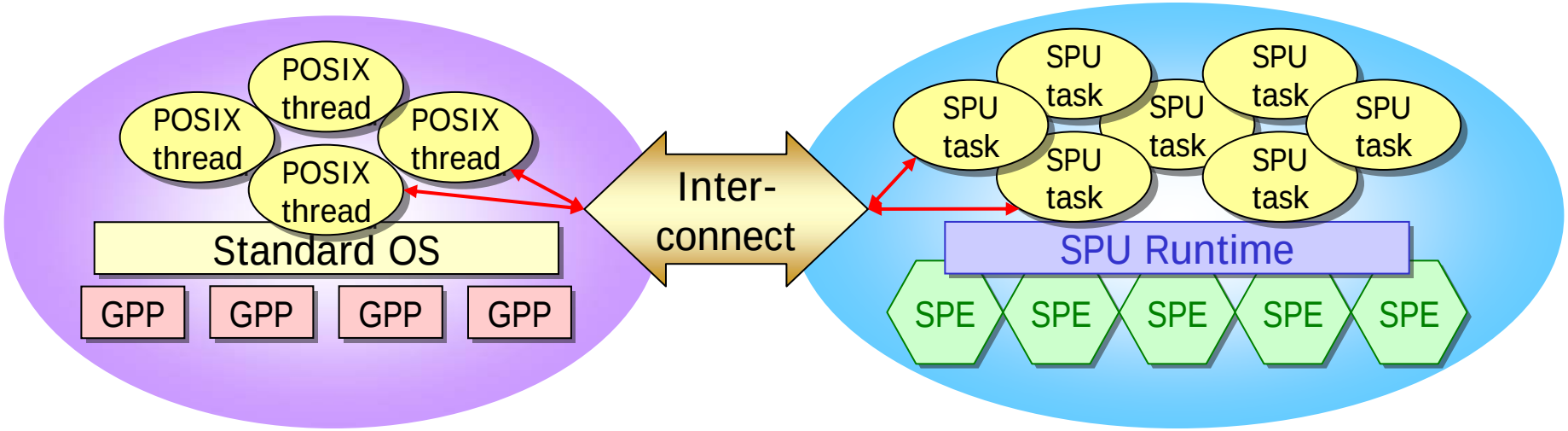
Possible Cell Software Architectures

- Architecture shouldn't be limited but the following architecture is efficient for current implementations
  - GPP might communicate with one or more SPE clusters using narrow interconnects such as Ethernet
  - In each cluster, all SPEs might be connected by high speed interconnect and shared memory



Simple Views for Programmers

- Complicated interactions between GPP and SPE shouldn't be always required
  - SPU programmers implement programs using SPU programming environment such as SPU task model and runtime
  - GPP programmers implement programs using general environment such as POSIX thread and standard OSes
  - Interconnect transfers data between GPP and SPU programs



GPP Programmers' View

SPU Programmers' View

Common Models and Platforms

- Considering use cases and software architectures, the following models and platform should be standardized for software compatibility

Common Models:

- GPP-Cluster communication model
- SPU-centric programming model

Common Platform:

- Common communication infrastructure
- Common SPU runtime

GPP-Cluster Communication Models

Stream Processing Model

- Sending and receiving packetized data continuously
- Suitable for multi-media processing
- SPE fits this model very well

Remote Procedure Call Model

- Sending request and receiving result
- Suitable to implement accelerated libraries
- Libraries can be implements on suitable processors

SPU-centric Programming Models

Accelerated Library Model

- Only library routines are running on SPEs
- Applications call libraries via RPC infrastructure

SPE Application Model

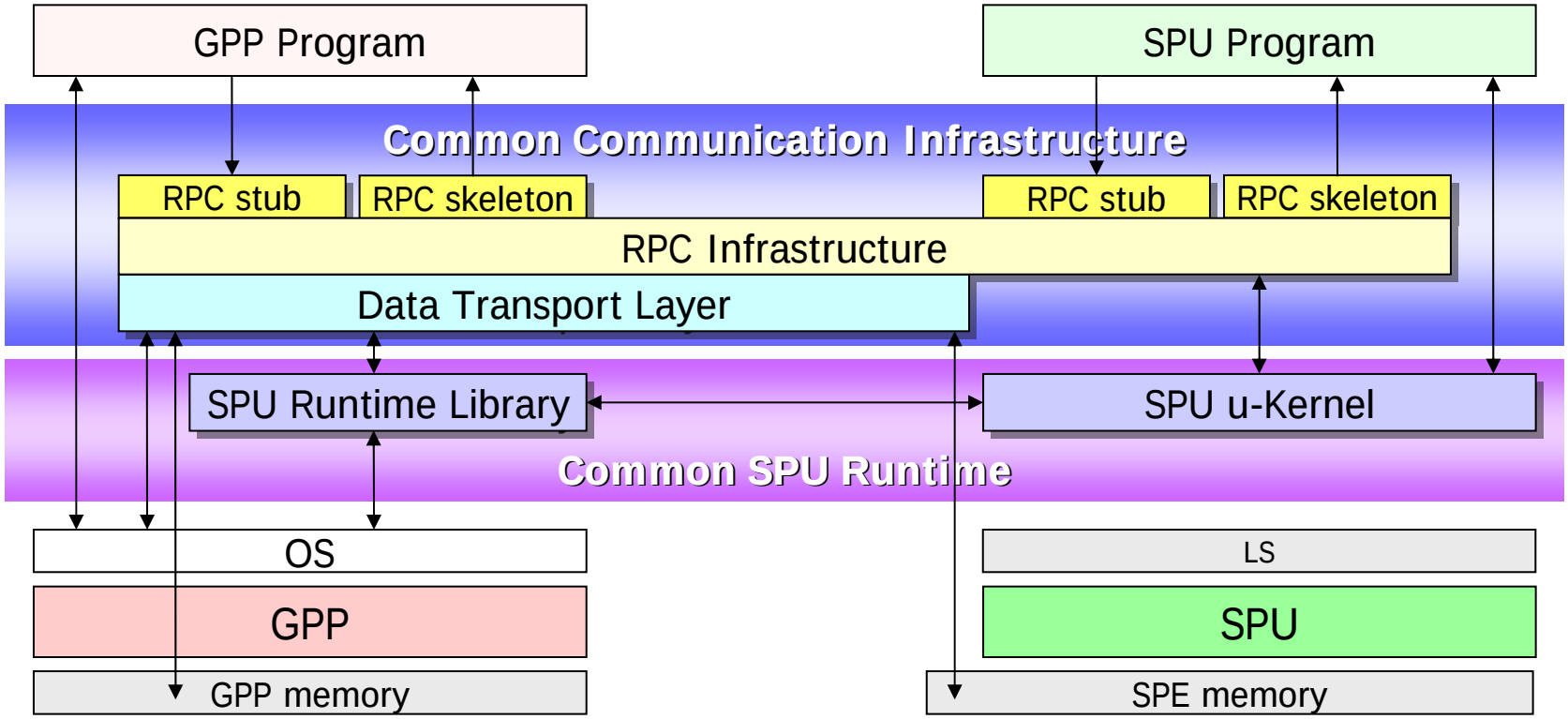
- Main routines are running on SPEs
- System services especially I/O are handled by GPP

Hybrid Model

- Main routines are running on SPEs
- Some libraries are accelerated by other SPEs
- Remained services are handled by GPP

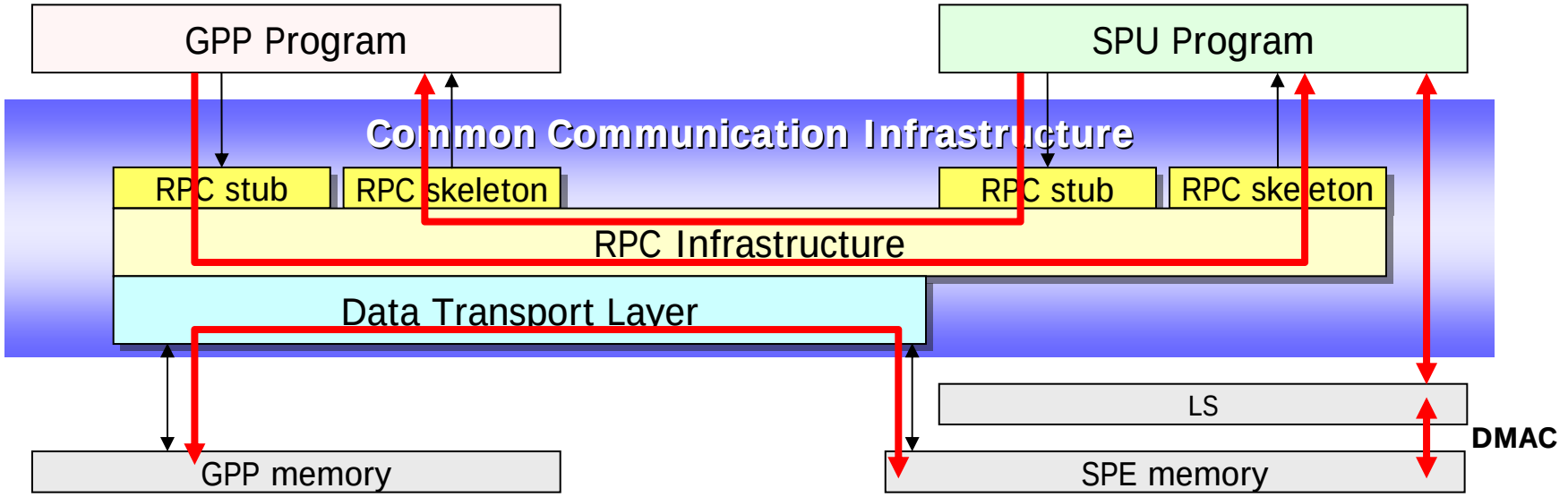
Reference Software Stack of Common Platform

- Common communication infrastructure consists of RPC system and data transport layer
- Common SPU runtime consists of SPU micro-kernel and SPU runtime library



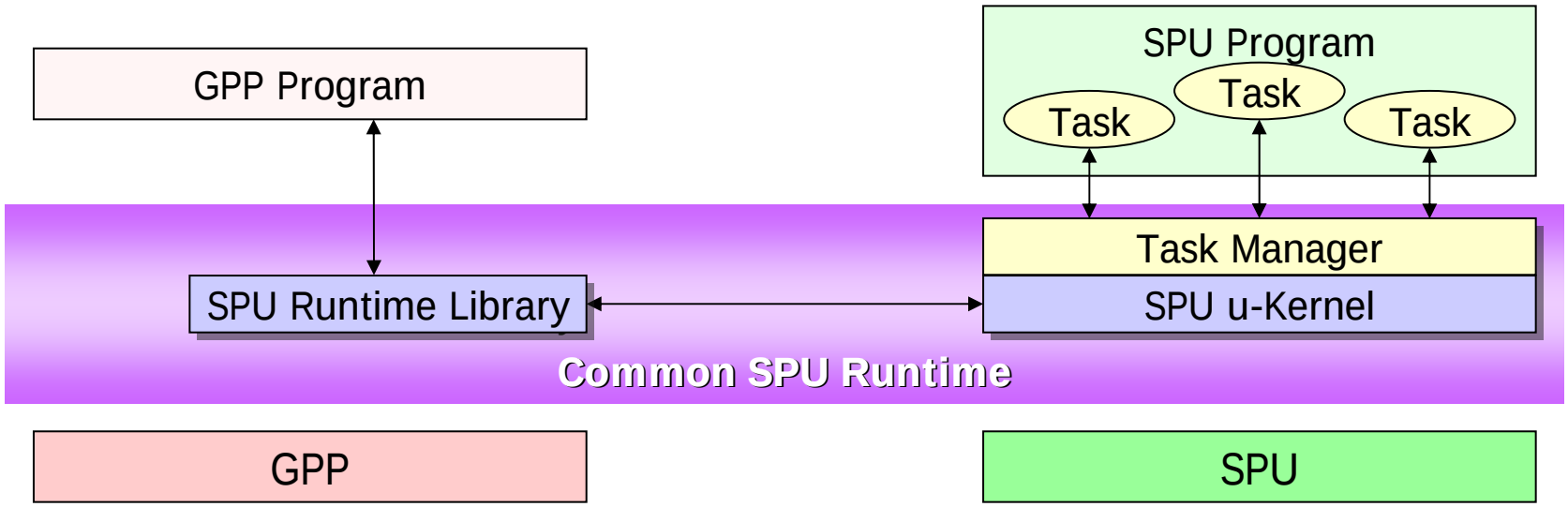
Common Communication Infrastructure

- Data transport layer transfers data between GPP memory and SPE memory (a.k.a. XDR DRAM)
  - Data are transferred between SPE memory and LS by DMAC
- RPC infrastructure delegates function calls to servers
  - RPC stub marshals arguments and RPC skeleton de-marshals the arguments
  - Data pointed by the arguments in addition to the marshaled arguments are transferred by data transport layer



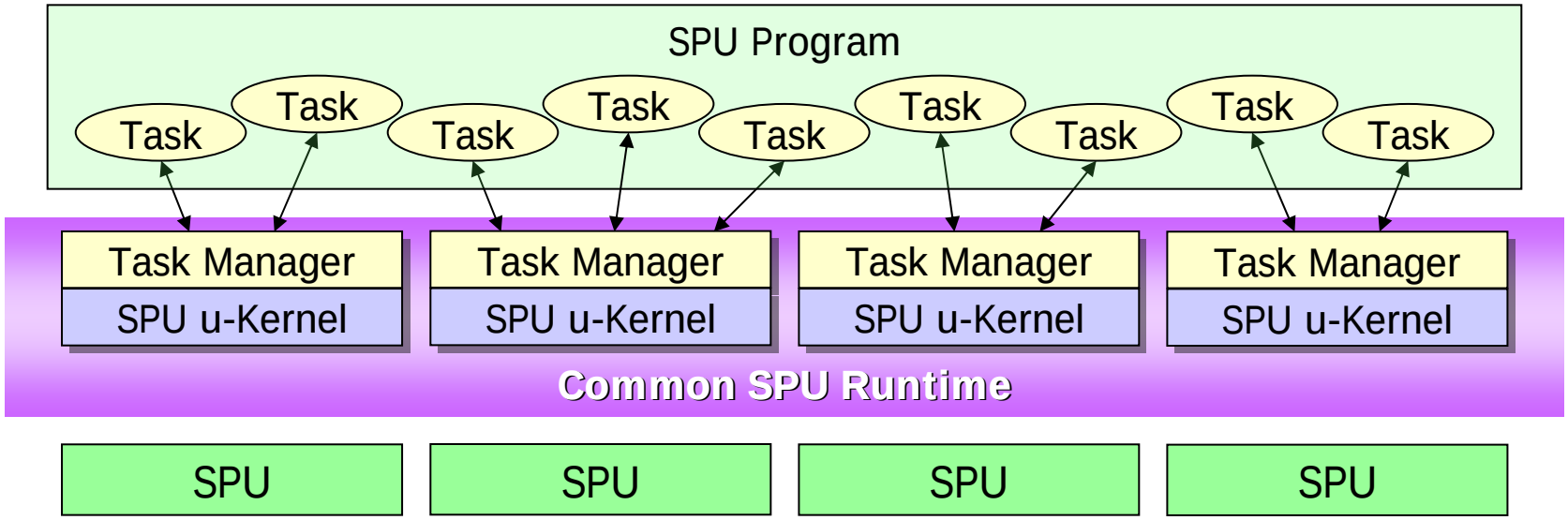
Common SPU Runtime

- GPP program can manage SPU program using SPU runtime library on GPP
  - GPP program might be application or SPU program loader
- SPU micro-kernel on SPU can be extended by policy managers such as task manager
  - Task manager enables multi-tasking model on SPU



Scalability of SPU Programs

- Common SPU runtime might offer scalability of SPU program
  - Multiple tasks should be executed by multiple SPEs
  - All tasks should be scheduled without interactions to GPP
- MARS by Sony is the best candidate
  - Suits the requirements of common SPU runtime
  - Please listen the next presentation!



Benefits of Common Platform

- Your can make SPU programs GPP independent
  - Common communication infrastructure standardizes interactions between GPP and SPE
- You can select one or more preferring programming models for SPU programs
  - Common SPU runtime can be extended to support suitable programming models
- You can reuse your application and library running on SPE across all Cell family processors
  - All your efforts on one platform are preserved on other platforms

Call for Participation

- Please quit “reinvention of the wheel”
  - Please join us to create common environment
  - Please stop developing environment, but feedback to common environment instead
  - Please focus on your actual applications
- Please look forward to enjoying compatibility
  - Firstly, write your code using PS3 or CRS(Toshiba’s Cell Reference Set)
  - Then, scale up to high performance computing world using QS22 or later blades without any modification
  - *And, make available to PC users using SpursEngine!*

Questions?

Common platform:

- cbe-oss-dev@ozlabs.org

This presentation:

- seiji.maeda@toshiba.co.jp

TOSHIBA

Leading Innovation