Introduction to CBEA SDK

Veselin Dikov

1. Getting started

Executable format check utility:

 # file <executable>

Makefile headers (reside in SDK root directory)

make.header
make.footer
make.env

Makefile examples

```
# Makefile - ppu
#
# Subdirectories ###############
DIRS := spu

# Target #######################
PROGRAM_ppu:= simple

# Local Defines ################

IMPORTS := spu/lib_simple_spu.a \
                  -lspe
# imports the embedded simple_spu
# library allows consolidation of
# spu program into ppe binary

# make.footer ##################
include ../make.footer

# make.footer is in the top of
# the SDK
```

```
# Makefile - spu
#
# Target #######################
PROGRAMS_spu := simple_spu
# created embedded library
LIBRARY_embed:= lib_simple_spu.a

# Local Defines
################################
IMPORTS = $(SDKLIB_spu)/libc.a

# make.footer
################################
include ../../make.footer

# make.footer is in the top of
# the SDK
```

2. SPU Language Extensions

The architecture's extended instruction set is supported by the SPU Language extensions.

2.1. SIMD vectorization – vector data type

128 bit data structure. CBEA architecture has extended instruction set that operates on vector.
Example:

```
vector unsigned int vec =  (vector unsigned int)(1,2,3,4);
vector unsigned int v_ones =  (vector unsigned int)(1);
vector unsigned int vdest =  spu_add(vec, v_ones);
```

2.2. Memory Flow Control (MFC)

SPE units fetch data from main storage true DMA channels
Example:

```
mfc_get(&data,addr+16384*i,16384,20,0,0);
```

LS address

EA address

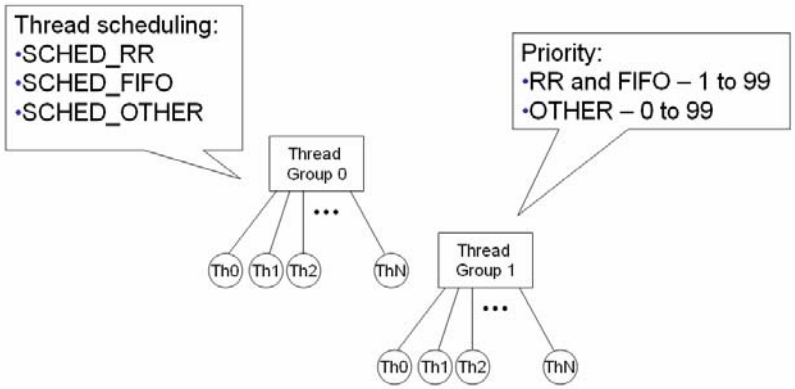
3. SPE library

Provides PPE functionality; two sets of functions – thread management and MFC access functions

Header: <libspe.h>

3.1. Thread management

Functions for creating and managing thread groups and threads, functions for accessing SPE thread attributes and communicating with the thread.



3.2. MFC mailboxes

Function for exchanging of messages and signals between PPE and SPE threads through DMA channels.

4. SDK libraries

Library name	*Short Description*	*PPE*	*SPE*
C Library	standard C99 functionality. POSIX.1 functions.	x	x
Audio Resample Library	audio resampling functionality for PPE and SPE	x	x
Curves and Surfaces Library	quadratic and cubic Bezier curves. Biquadric and bicubic Bezier surfaces, and curved point-normal triangles.	x	x
FFT Library	1-D FFT and kernel functions for 2-D FFT	x	x
Game Math Library	math routines applicable to game performance needs	x	x
Image Library	routines for processing images - convolutions and histograms	x	x
Large Matrix Library	basic linear algebra routines on large vectors and matrices		x

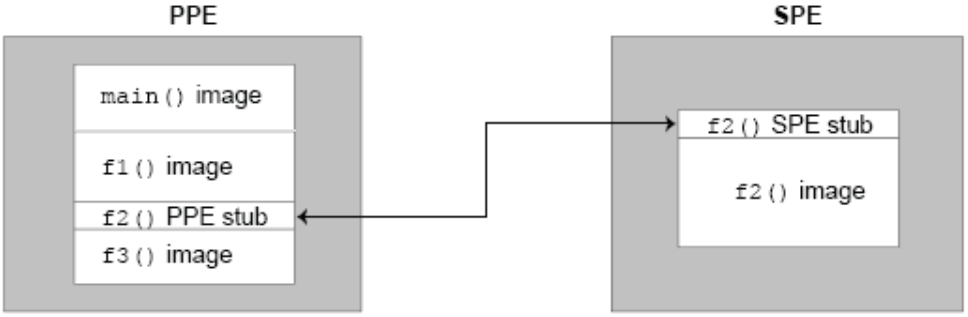
Math Library	general purpose math routines tuned to exploit SIMD	x	x
Matrix Library	routines for operations on 4x4 Matrices and quaternions	x	x
Misc Library	set of general purpose routines that don't logically fit within any other	x	x
Multi-Precision Math Library	operations on unsigned integer numbers with large number of bits		x
Noise LibraryPPE	1-,2-, 3-, 4-D noise, Lattice and non-lattice noise, Turbulance	x	x
Oscillator Libraries	definition of sound sources	x	x
Simulation Library	functionality related to the Full-Simulator	-	-
Sync Library	synchronization primitives, like atomic operations, mutex	x	x
Vector Library	a set of general purpose routines that operate on vectors.	x	x

5. Remote Procedure Calls (RPC)

Communication between PPE and SPE threads via function stubs

Enables the implementation of Function-Offload Model, where:

- SPE threads work as services – provide some functionality via IDL interfaces
- PPE communicates with them thought RPC calls



Interface Description Language (IDL) is an industry standard for definition of RPC interfaces:

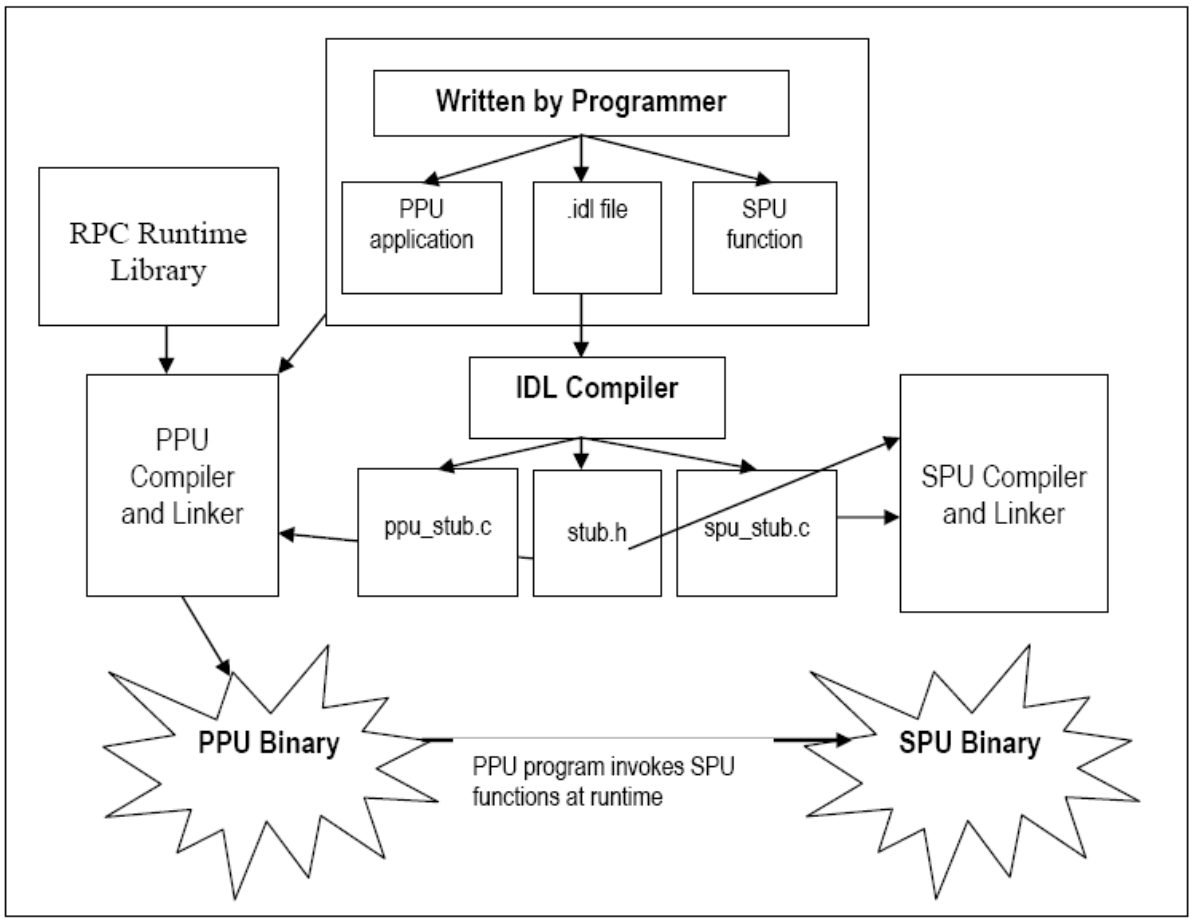
idl files

```
interface add
{
      import "../stub.h";
      const int ARRAY_SIZE = 1000;
      [sync] idl_id_t do_inv ([in] int array_size,
            [in, size_is(array_size)] int array_a[],
            [out, size_is(array_size)] int array_res[]);
      …
}
```

idl compiler

```
# idl -p ppe_sample.c -s spe_sample.c sample.idl
```

IDL “mechanism”:



6. References

- CBEA-Tutorial.pdf, SDK documentation
- idl.pdf, SDK documentation
- libraries_SDK.pdf, SDK Documentation
- libspe_v1.0.pdf, SDK Documentation
- SPU_language_extensions_v21.pdf, Sony online resources
  http://cell.scei.co.jp/pdf/SPU_language_extensions_v21.pdf, 15.03.2006