# Hands-on – SPU to SPU DMA Transfer

Cell Programming Workshop
Cell Ecosystem Solutions Enablement

# Class Objectives

- **Objective: Learn how to do basic DMA transfers**
- **'Hello World' already uses DMA to send data to an SPU**
- **We will modify 'Hello world' example to:**
  - Send data from an SPU back to the PPU
  - Send data between SPUs

/opt/cell_class/Hands-on-30/DMA/spu-spu/libspe2

# ‘Hello World’ Review

- **PPU creates NUM_THREADS threads**
- **Each pthread starts a copy of the SPU program**
- **SPU program does a DMA transfer to receive a string**
- **SPU program prints the string and exits**

```
. . .
  /* Here is the actual DMA call */
  /* the first parameter is the address in local store to
place the data */
  /* the second parameter holds the main memory address
*/
  /* the third parameter holds the number of bytes to DMA
*/
  /* the fourth parameter identifies a "tag" to associate
with this DMA  */
  /* (this should be a number between 0 and 31, inclusive)
*/
  /* the last two parameters are only useful if you've
implemented your  */
  /* own cache replacement management policy.  Otherwise set
them to 0.  */

  mfc_get(parameter_area, argp.ull, 128, 31, 0, 0);

  /* Now, we set the "tag bit" into the correct channel on
the hardware  */
  /* this is always 1 left-shifted by the tag specified with
the DMA     */
  /* for whose completion you wish to wait.
*/
  mfc_write_tag_mask(1<<31);

  /* Wait for the data array DMA to complete. */
  mfc_read_tag_status_all();

  printf("SPE: Data received is: %s", parameter_area );

. . .
```

# Next Step: Data Transfer back to PPU

- **DMA Puts look exactly like DMA Gets**

```
. . .

  mfc_get(parameter_area, argp.ull, 128, 31, 0, 0);

  /* Now, we set the "tag bit" into the correct channel on the hardware  */
  /* this is always 1 left-shifted by the tag specified with the DMA     */
  /* for whose completion you wish to wait.                              */
  mfc_write_tag_mask(1<<31);

  /* Wait for the data array DMA to complete. */
  mfc_read_tag_status_all();

  printf("SPE: Data received is: %s", parameter_area );

  sprintf( parameter_area, "%llx: Back at you\n", speid );

  // Now send it back.

  mfc_put(parameter_area, argp.ull, 128, 31, 0, 0);
  mfc_write_tag_mask(1<<31);
  mfc_read_tag_status_all();

. . .
```

# SPU to SPU Transfer

- **Passes data from PPU to SPU, SPU to SPU, then back to the PPU.**
- **New concepts/constructs:**
  - Mailboxes
  - Using SPE local store addresses
- **PPU Flow**
  - Receive a mailbox message from each SPU. The message contains an offset into local store where the buffer for the SPU lives.
  - Send mailbox messages to SPUs. The message has a control block pointer.
  - Send a mailbox message to each SPU to tell them when to execute.
- **SPU Flow**
  - Send a mailbox message to the PPU with the offset to the local buffer.
  - Read a mailbox to get a control block address.
  - Wait on mailbox for a signal to do a data movement to another SPU

# PPU Code

```
#include <stdio.h>
#include <libspe2.h>
#include <libmisc.h>
#include <pthread.h>
#include <string.h>
#include "control.h"

extern spe_program_handle_t hello_spu;
char buffer[128] __attribute__ ((aligned(128)));

#define ACTIVE_SPUS 6

void *ppu_pthread_function(void *arg)
{
        unsigned int entry = SPE_DEFAULT_ENTRY;

        spe_context_run(*((spe_context_ptr_t *)arg), &entry, 0, NULL, NULL, NULL) ;
        pthread_exit(NULL);
}

int main()
{

        int i;
        spe_context_ptr_t ctxs[ACTIVE_SPUS];
        pthread_t threads[ACTIVE_SPUS];

        control_block * cb = (control_block *)malloc_align(128,7);
        cb->first = 0;
        cb->last = ACTIVE_SPUS - 1;
        cb->memory = buffer;
        strcpy (buffer, "Zao shang hao!");
```

**control.h**

```
typedef struct {
  char * memory;
  int first;
  int last;
  void * lstore[8];
  char padding[84];
} control_block;
```

# PPU Code (continued)

```
 for (i=0;i<ACTIVE_SPUS;i++){
                ctxs[i] = spe_context_create (0, NULL);
                spe_program_load (ctxs[i], &hello_spu);
                pthread_create (&threads[i], NULL, &ppu_pthread_function, &ctxs[i]);
        }
        for (i=0;i<ACTIVE_SPUS;i++){
                cb->lstore[i] = spe_ls_area_get(ctxs[i]);
                while (!spe_out_mbox_status(ctxs[i])) {}
                unsigned int temp;
                spe_out_mbox_read(ctxs[i], &temp, 1);
                cb->lstore[i] += temp;

        }
        for (i=0;i<ACTIVE_SPUS;i++){
                unsigned int data;
                data = (unsigned int)(cb);
                spe_in_mbox_write(ctxs[i], &data, 1, SPE_MBOX_ANY_NONBLOCKING);
        }

        for (i=0;i<ACTIVE_SPUS;i++){
                spe_in_mbox_write(ctxs[i], (unsigned int *)&i, 1, SPE_MBOX_ANY_NONBLOCKING);
                pthread_join (threads[i], NULL);
        }

        printf("PPE says %s\n", buffer);
        return 0;
}
```

# SPU Code

```
#include <stdio.h>
#include <spu_mfcio.h>
#include <malloc_align.h>
#include <string.h>
#include "../control.h"

int main()
{
  int myId;
  char * buffer = malloc_align(128,7);
  control_block * ls_cb = malloc_align(sizeof(control_block),7);
  control_block * cbPtr;

  // Send buffer offset
  spu_write_out_mbox((unsigned int)buffer);

  // Get control block address and transfer it to local storage
  cbPtr = (control_block *) spu_read_in_mbox();
  mfc_get (ls_cb, (unsigned int)cbPtr, sizeof(control_block), 1, 0, 0);
  mfc_write_tag_mask(1<<1);
  mfc_read_tag_status_all();

  // Wait to execute until it is our turn
  myId = spu_read_in_mbox();

  // First SPU in the chain reads from PPU memory
  if (myId == ls_cb->first) {
    mfc_get(buffer, (unsigned int)ls_cb->memory, 128, 1, 0, 0);
    mfc_write_tag_mask(1<<1);
    mfc_read_tag_status_all();
  }

  printf ("SPE %d says %s\n", myId, buffer);
```

# SPU Code (continued)

```
 switch ( myId ) {
    case 0: { strcpy(buffer,"Dobry dien!"); break; }
    case 1: { strcpy(buffer,"Bon giorno!"); break; }
    case 2: { strcpy(buffer,"Buenas dias!"); break; }
    case 3: { strcpy(buffer,"Ohayo gozaimasu!"); break; }
    case 4: { strcpy(buffer,"Boker tov!"); break; }
    case 5: { strcpy(buffer,"Dobre jitro!"); break; }
    case 6: { strcpy(buffer,"Bon jour!"); break; }
    case 7: { strcpy(buffer,"Chao buoi sang!"); break; }
  }

  // Send message to next SPU (or PPU if we are the last SPU)
  if (myId == ls_cb->last) {
    mfc_put(buffer, (unsigned int)ls_cb->memory, 128, 1, 0, 0);
  }
  else {
    mfc_put(buffer, (unsigned int)ls_cb->lstore[myId+1], 128, 1, 0, 0);
  }

  mfc_write_tag_mask(1<<1);
  mfc_read_tag_status_all();

  return 0;
}
```

# Special Notices - Trademarks

This document was developed for IBM offerings in the United States as of the date of publication. IBM may not make these offerings available in other countries, and the information is subject to change without notice. Consult your local IBM business contact for information on the IBM offerings available in your area. In no event will IBM be liable for damages arising directly or indirectly from any use of the information contained in this document.

Information in this document concerning non-IBM products was obtained from the suppliers of these products or other public sources. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. Send license inquires, in writing, to IBM Director of Licensing, IBM Corporation, New Castle Drive, Armonk, NY 10504-1785 USA.

All statements regarding IBM future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only.

The information contained in this document has not been submitted to any formal IBM test and is provided "AS IS" with no warranties or guarantees either expressed or implied.

All examples cited or described in this document are presented as illustrations of the manner in which some IBM products can be used and the results that may be achieved. Actual environmental costs and performance characteristics will vary depending on individual client configurations and conditions.

IBM Global Financing offerings are provided through IBM Credit Corporation in the United States and other IBM subsidiaries and divisions worldwide to qualified commercial and government clients. Rates are based on a client's credit rating, financing terms, offering type, equipment type and options, and may vary by country. Other restrictions may apply. Rates and offerings are subject to change, extension or withdrawal without notice.

IBM is not responsible for printing errors in this document that result in pricing or information inaccuracies.

All prices shown are IBM's United States suggested list prices and are subject to change without notice; reseller prices may vary.

IBM hardware products are manufactured from new parts, or new and serviceable used parts. Regardless, our warranty terms apply.

Many of the features described in this document are operating system dependent and may not be available on Linux. For more information, please check: http://www.ibm.com/systems/p/software/whitepapers/linux_overview.html

Any performance data contained in this document was determined in a controlled environment. Actual results may vary significantly and are dependent on many factors including system hardware configuration and software design and configuration. Some measurements quoted in this document may have been made on development-level systems. There is no guarantee these measurements will be the same on generally-available systems. Some measurements quoted in this document may have been estimated through extrapolation. Users of this document should verify the applicable data for their specific environment.

Revised July 23, 2006

# Special Notices - Trademarks (continued)

The following terms are trademarks of International Business Machines Corporation in the United States and/or other countries: alphaWorks, BladeCenter, Blue Gene, ClusterProven, developerWorks, e business(logo), e(logo)business, e(logo)server, IBM, IBM(logo), ibm.com, IBM Business Partner (logo), IntelliStation, MediaStreamer, Micro Channel, NUMA-Q, PartnerWorld, PowerPC, PowerPC(logo), pSeries, TotalStorage, xSeries; Advanced Micro-Partitioning, eServer, Micro-Partitioning, NUMACenter, On Demand Business logo, OpenPower, POWER, Power Architecture, Power Everywhere, Power Family, Power PC, PowerPC Architecture, POWER5, POWER5+, POWER6, POWER6+, Redbooks, System p, System p5, System Storage, VideoCharger, Virtualization Engine.

A full list of U.S. trademarks owned by IBM may be found at: http://www.ibm.com/legal/copytrade.shtml.

Cell Broadband Engine and Cell Broadband Engine Architecture are trademarks of Sony Computer Entertainment, Inc. in the United States, other countries, or both.

Rambus is a registered trademark of Rambus, Inc.

XDR and FlexIO are trademarks of Rambus, Inc.

UNIX is a registered trademark in the United States, other countries or both.

Linux is a trademark of Linus Torvalds in the United States, other countries or both.

Fedora is a trademark of Redhat, Inc.

Microsoft, Windows, Windows NT and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries or both.

Intel, Intel Xeon, Itanium and Pentium are trademarks or registered trademarks of Intel Corporation in the United States and/or other countries.

AMD Opteron is a trademark of Advanced Micro Devices, Inc.

Java and all Java-based trademarks and logos are trademarks of Sun Microsystems, Inc. in the United States and/or other countries.

TPC-C and TPC-H are trademarks of the Transaction Performance Processing Council (TPPC).

SPECint, SPECfp, SPECjbb, SPECweb, SPECjAppServer, SPEC OMP, SPECviewperf, SPECapc, SPEChpc, SPECjvm, SPECmail, SPECimap and SPECsfs are trademarks of the Standard Performance Evaluation Corp (SPEC).

AltiVec is a trademark of Freescale Semiconductor, Inc.

PCI-X and PCI Express are registered trademarks of PCI SIG.

InfiniBand™ is a trademark the InfiniBand® Trade Association

Other company, product and service names may be trademarks or service marks of others.

Revised July 23, 2006

# Special Notices - Copyrights



The following are trademarks of International Business Machines Corporation in the United States, or other countries, or both.

IBM IBM Logo Power Architecture

Other company, product and service names may be trademarks or service marks of others.

All information contained in this document is subject to change without notice. The products described in this document are NOT intended for use in applications such as implantation, life support, or other hazardous uses where malfunction could result in death, bodily injury, or catastrophic property damage. The information contained in this document does not affect or change IBM product specifications or warranties. Nothing in this document shall operate as an express or implied license or indemnity under the intellectual property rights of IBM or third parties. All information contained in this document was obtained in specific environments, and is presented as an illustration. The results obtained in other operating environments may vary.

While the information contained herein is believed to be accurate, such information is preliminary, and should not be relied upon for accuracy or completeness, and no representations or warranties of accuracy or completeness are made.

THE INFORMATION CONTAINED IN THIS DOCUMENT IS PROVIDED ON AN "AS IS" BASIS. In no event will IBM be liable for damages arising directly or indirectly from any use of the information contained in this document.

IBM Microelectronics Division
1580 Route 52, Bldg. 504
Hopewell Junction, NY 12533-6351

The IBM home page is http://www.ibm.com
The IBM Microelectronics Division home page is
http://www.chips.ibm.com

 