**What is Charm++?**

Charm++ is a message-driven paradigm developed by the Parallel Programming Lab (PPL) which uses messages to communicate between entities called Chares (C++ objects). The Chares are then spread across the available processors. The runtime system automatically handles mapping of objects to processors, routing messages, load-balancing, automatic check-pointing, has associated tools for collecting performance data, etc.

**Charm++ on Cell so Far**

We have created an **Offload API** that will allow Charm++ applications to take advantage of the Cell processor. Using the Offload API, the application can send **work requests** to the SPEs. Each of the SPEs has a simple scheduler running on it that will coordinate the execution of work requests along with moving the data needed by those work requests.

**Some Attributes of Cell:**

- SPEs run at same clock speed as PPE
- All SPE main memory accesses are done explicitly using DMA transactions
- SPE loads and stores can only access the local store (256KB, 6-cycle latency)
- Each SPE has 25.6 GFlop/s peak performance (single-precision @ 3.2 GHz)
- SPEs have two in-order pipelines: up to two instructions per clock
- 243M transistors (221mm²)
- Element Interconnect Bus (EIB): high-bandwidth from 74GB/s (worst-case) to 200GB/s (best-case)
- PPE is a 2-way SMT
- Both the PPE and SPE can initiate DMA transactions

**Future Work**

- **Application Development:** NAMD, Cosmology, etc.
- **Development of Load-Balancers** (considering several metrics)
  - Greatly varying communication costs depending on sending and receiving processing elements (over EIB, within node, and over the interconnect)
  - Varying capabilities of processing elements including memory size, ISA characteristics, etc.
- **Portability / Ease-of-Programming**
  - Modify Charmixi to allow offloadable keyword
  - Auto-generate code needed by Offload API based on user’s code
- **Performance Analysis**
  - generate Projections data for SPEs
- **Special Purpose Hardware**: FPGAs, GPUs, etc.

**What is Cell?**

Jointly developed by IBM, Sony, and Toshiba, the Cell processor has 9 cores. One of the cores is a fairly standard PowerPC core. This “main core” is the called the Power Processing Element (PPE). The other 8 cores are called Synergistic Processing Elements (SPEs). A brief summary of the Cell’s characteristics is below.

**Charm++ Complements Cell**

- Ease of Programming
  - The nature of Cell makes it difficult to program
  - Charm++ programming model fits Cell well
- Intelligent Message Scheduling
  - Messages arriving for Chares are queued for execution
  - Schedulerpeek ahead in message queue to initiate DMA transfers of data/code that will be used by future messages while the current messages are still being processed
- Load-Balancing
  - Processing elements have different characteristics (SPE vs PPE)
  - Network-topology aware load-balancers (different communication costs between interconnect and EIB)
  - Application can use different load-balancers for different platforms
- Portability
  - Charm++ applications can already run on a variety of platforms
  - Will allow users to take advantage of Cell’s power with little to no modification of existing code
  - Chares Encapsulate Data/Code (Locality)
  - Arriving message in self-contained
  - Other data used by entry method is usually contained within the associated Chare

**Some Attributes of Cell:**

- SPEs run at same clock speed as PPE
- All SPE main memory accesses are done explicitly using DMA transactions
- SPE loads and stores can only access the local store (256KB, 6-cycle latency)
- Each SPE has 25.6 GFlop/s peak performance (single-precision @ 3.2 GHz)
- SPEs have two in-order pipelines: up to two instructions per clock
- 243M transistors (221mm²)
- Element Interconnect Bus (EIB): high-bandwidth from 74GB/s (worst-case) to 200GB/s (best-case)
- PPE is a 2-way SMT
- Both the PPE and SPE can initiate DMA transactions

**Applications:**

- **NAMD**
  - A molecular dynamics code, first Charm++ application that will be adapted to utilize the Cell. NAMD was developed by the Theoretical and Computational Biophysics Group (TCBG) at UIUC’s Beckman Institute in collaboration with the Parallel Programming Lab. In 2002 it won the Gordon Bell Award and is currently being used around the world.
  - **NAMD Highlights:**
    - 2002 Gordon Bell Award
    - Scaled to 8000 Processors (BlueGene/L)
    - File-Convertible with AMBER, CHARMM, X-PLOR
    - Used by DOE National Labs

- **NAMD Highlights:**
  - 2002 Gordon Bell Award
  - Scaled to 8000 Processors (BlueGene/L)
  - File-Convertible with AMBER, CHARMM, X-PLOR
  - Used by DOE National Labs

**Ongoing PPL Research Includes:**

- Development of network-topology aware load-balancers
- Fault tolerance
- Dynamically changing the number of physical processors
- Multi-Cluster computing
- ParFUM: Parallel Framework for Unstructured Meshes
- Adaptive AMPI (AMPI)

**Recently in the News...**

- **NAMD:**
  - Using NAMD, researchers in TCBG recently simulated an entire life form at the atomic level for the first time. They simulated the satellite tobacco mosaic virus. They used 256 processors on NCSA’s SGI Altix to perform the 1,000,000 atom virus simulation for a total of 30s (at 1.1ms per day).

**Future Work**

- **Application Development**
  - NAMD, Cosmology, etc.
- **Development of Load-Balancers** (considering several metrics)
  - Greatly varying communication costs depending on sending and receiving processing elements (over EIB, within node, and over the interconnect)
  - Varying capabilities of processing elements including memory size, ISA characteristics, etc.
- **Portability / Ease-of-Programming**
  - Modify Charmixi to allow offloadable keyword
  - Auto-generate code needed by Offload API based on user’s code
- **Performance Analysis**
  - generate Projections data for SPEs
- **Special Purpose Hardware**
  - FPGAs, GPUs, etc.