OpenPOWER® Kernel Features

IBM Liang ZHUANG/Jia HE
2015/10
OpenPOWER – Why, What, How

IBM Liang ZHUANG
2015/10
Rethink the Data Center
One Year. 113 Members.

Join the revolution at openpowerfoundation.org
#OPENPOWERSUMMIT
The Shared Vision

To create an open development
community dedicated to
transforming the approach to scale
d out server design based on the
POWER architecture.
Rethink the Data Center

THE END OF MOORE'S LAW
Cost-performance benefits are diminishing

IMPROVE INTEGRATION OF CPU WITH I/O AND ACCELERATOR SUBSYSTEMS
Maximum impact through collaboration with technology leaders

OPEN PROCESSOR ARCHITECTURE WITH SCALE-OUT PERFORMANCE
## Proposed Work Groups and Projects

<table>
<thead>
<tr>
<th>Work Group</th>
<th>Projects</th>
<th>Participants</th>
</tr>
</thead>
</table>
| System Software (Open Source)                  | • Linux LE  
• KVM  
• Firmware  
  – OpenPOWER FW interface  
• POWER LE ABI | Public          |
| Application Software (Open Source)             | • System Operating Environment  
  – OpenPOWER Software ecosystem enablement  
• Toolchain | Public          |
| Open Server Development Platform               | • Power 8 Developer Board  
• POWER 8 Reference Design | Member         |
| Hardware Architecture                           | • OpenPOWER profile of architecture  
  – Power8 ISA Book 1, 2, 3  
• Coherent Accelerator Interface Architecture (CAIA) | Member         |
| Compliance                                      | • Compliance                | Member         |
Proposed Ecosystem Enablement

Power Open Source Software Stack Components
- Cloud Software
- Standard Operating Environment (System Mgmt)
- Operating System / KVM
- Firmware
- Hardware

Existing Open Source Software Communities

New OSS Community

System Operating Environment Software Stack
A modern development environment is emerging based on tools and services

Multiple Options to Design with POWER Technology Within OpenPOWER

Hardware
- POWER8
- CAPI over PCIe

“Standard POWER Products” – 2014

Framework to Integrate System IP on Chip
- Customizable

“Custom POWER SoC” – Future

Industry IP License Model
Open Source Ecosystem

Available: Backbone, Bootstrap, Docker, Eigen lib, Erlang, Ganglia, GCC, gccgo, GDB, Jenkins, Jruby, LLVM, logstash, logstash-forwarder, Maven, Nagios, NGINX, node.js, OpenJDK, PHP, phpMyAdmin, Perl, Python, Python-Django, Python-Pip ecosystem, RR, RabbitMQ, rsyslog, Ruby, Ruby on Rails (rbenv), Ruby Gems, scala, snappy, Socket.io (npms) Supervisord, SpiderMonkey, SystemTap, Vagrant, V8, wireshark, Xerces

Port In Progress: Apache Gump, GoLang, kibana, Pubsub.io (3Q), Phantom.js
Evaluating: Cloudbees, OpenJDK - optimize, Ruby - optimize

Available: Accumulo (column), Cassandra, CouchDB (document), Derby, MariaDB (v10 optimized), Memcached (KV), MongoDB (document), MySQL, PostgreSQL, RabbitMQ, Redis (KV), Riak, SQLite, TokyoCabinet, Virtuoso (graph)

Port In Progress: Voldemort (KV), Neo4J (graph), MongoDB 3.0
Optimizing: PostgreSQL (1.86x), CouchDB
Evaluating: Couchbase (noSQL), InfiniSQL, MarkLogic (document, ISV), OrientDB

Available: Hadoop Core, Hive, HBase, Accumulo, Ambari, Avro, ElasticSearch, Falcon, Flume, Hue, Knox, Lucene-Solr, Mahout, Oozie, Parquet, Phoenix, Pig, Riak, Sqoop, Storm Tez, Zookeeper

Port In Progress: Spark
Optimizing: Hadoop (3Q15)
Evaluating: Clustertoolkit


Port In Progress: Glassfish


Port In Progress: CentOS
Evaluating: Cluster-Network, CoreOS (distro), MondoRescue, Open Identity Stack (forgerock.com)

Available: Open source application is ported and available on distro (Ubuntu or RHEL or SLES) (black), in community (purple), Lab7 (green) or Veristorm (orange). Does not mean it is optimized. Does not mean that a commercial ISV version is available.

Evaluating: Needs to be vetted in new business development prioritization process. Some of these are available codes that need optimization to be competitive with x86.
Foundation Members (147 members over 22 countries, 2015/8)
Agenda

- OpenPOWER CPU (Power8)
  - Hardware Highlights
  - Linux Distro Support Status

- Kernel-related Software Enhancement
  - BE/LE
  - Hardware acceleration/crypto
  - Parallel programming
  - Energy management
  - Java Performance enhancements
OpenPOWER CPU (Power8) - Hardware Highlights

Cores
- 12 cores (SMT8) 96 threads per chip
- 8 dispatch, 10 issue
- 16 execution pipes

Caches
- 64K data cache, 32K instruction cache
- 512 KB L2 /core (SRAM)
- 96 MB L3 (eDRAM shared)
- Up to 128 MB L4 (eDRAM, off-chip)

Accelerators
- Crypto & memory expansion
- Transactional Memory
- Coherent Accelerator Processor Interface (CAPI)

Memory
- Up to 230 GB/s sustained bandwidth

New instructions
- Quadword new atomic instructions
- Vmx. vsx
## Linux Distro for OpenPOWER – Guest Status

<table>
<thead>
<tr>
<th>Linux</th>
<th>Release (start from)</th>
<th>Endian</th>
<th>KVM guest</th>
</tr>
</thead>
<tbody>
<tr>
<td>Redhat</td>
<td>7.0</td>
<td>Big</td>
<td>✔</td>
</tr>
<tr>
<td>Redhat</td>
<td>7.1</td>
<td>Little/Big</td>
<td>✔ ★</td>
</tr>
<tr>
<td>SUSE</td>
<td>12</td>
<td>Little/Big</td>
<td>✔</td>
</tr>
<tr>
<td>Ubuntu</td>
<td>14.04</td>
<td>Little/Big</td>
<td>✔</td>
</tr>
</tbody>
</table>

- ✔ Supported
- ✗ Not Supported
- ★ Firmware 8.30 or higher
# Linux Distro for OpenPOWER – Bare Metal & Host Status

<table>
<thead>
<tr>
<th>Linux</th>
<th>Release (start from)</th>
<th>Endian</th>
<th>Bare Metal</th>
<th>KVM Host</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Redhat</td>
<td>7.2</td>
<td>Little</td>
<td>✔</td>
<td>✔*</td>
<td>*Only in RHEV firstly</td>
</tr>
<tr>
<td>SUSE</td>
<td>12SP1</td>
<td>Little</td>
<td>✔</td>
<td>✔</td>
<td></td>
</tr>
<tr>
<td>Ubuntu</td>
<td>14.04</td>
<td>Little</td>
<td>✔</td>
<td>✔</td>
<td>First supported distro</td>
</tr>
<tr>
<td>PowerKVM*</td>
<td>2.1</td>
<td>BE</td>
<td>✔</td>
<td>✔</td>
<td>Guest BE/LE</td>
</tr>
<tr>
<td>PowerKVM</td>
<td>3.1</td>
<td>LE</td>
<td>✔</td>
<td>✔</td>
<td>Guest BE/LE</td>
</tr>
</tbody>
</table>

* PowerKVM™ is IBM’s hypervisor distro.

## Summary

All major distros are supporting in LE mode, whatever BML, Host or Guest.

（所有主流发行版，都支持LE mode工作在OpenPOWER CPU上）
Kernel-related Software Enhancement for OpenPOWER P8 CPU

- **High performance BE/LE support** (大小端问题)
- **Hardware Acceleration/Crypto** (硬件加速)
  - CAPI (Coherent Accelerator Processor Interface)
  - Hardware acceleration/Encryption
- **Parallel Programming Productivity** (并行计算)
  - SMT/Split Core
  - Hardware Transactional Memory
- **Energy Management** (能耗)
- **Java Performance Enhancements** (JAVA性能)
High performance BE/LE support

- **Why do we need to support both Big/Little Endian?**
  - A new eco-system
  - Easy for applications to migrate to powerpc platform
  - Driver codes integration between ppc64le and other le arch

- **How to?**
  - Firmware
  - Kernel 3.13 (ubuntu 14.04’s kernel)
  - Designed a new (ELF V2) ABI to simplify and improve performance
  - Advance Toolchain 7.0 for BE, 7.1 for LE
  - Gcc 4.8
  - Glibc 2.17
硬件加速 – CAPI Use Case

- **Use Case**
  - Hadoop needs 3 data copy
  - More disks = more costs in server
  - Erasure code algorithm = cpu usage 99.9%

- **Solution: CAPI FPGA accelerator over PCIe**
  - Coherent Accelerator Processor Interface
  - No changes on server hardware configuration
  - No additional rack space, no more AC power support,

- **Benefit**
  - Disk number decreased 50%, server cost decreased 30%
  - 1 CAPI card equals 10-20 CPU cores’ compute capacity
  - Cost performance improved 300-400%
CAPI – Why it rocks!

传统方式

Application
Read/Write
Syscall

FileSystem
strategy()  iodone()

LVM
strategy()  iodone()

Disk & Adapter DD
Pin buffers,
Translate,
Map DMA,
Start I/O
Interrupt,
unmap,
unpin, iodone
scheduling

CAPI方式

Application
Posix Async
I/O Style API
aio_read()
aio_write()

User Library
Shared
Memory
Work Queue

< 500
Instructions

20K
Instructions

Attach flash memory to POWER8 via CAPI coherent Attach

© 2014 IBM Corporation
CAPI – HW Components

- **CAPP** - Coherent Accelerator Processor Proxy
  - Maintains directory of cache lines held by Accelerator
  - Snoops PowerBus on behalf of Accelerator

- **PSL** - Power Service Layer
  - Performs Address Translations
  - Maintains Cache coherence

- **AFU** - FPGA chip self defined by vendors

**Benefits**
- Accelerator can work with same memory addresses that the processors use
- Pointers de-referenced same as the host application
- Removes OS & device driver overhead
CAPI – Software Components

- **CAPI support in kernel**
  - Regard CAPI device as a PCIe device, match/init/configure
  - Maintenance, system protection, and communication functions.

- **Application library**
  - Libcxl
Hardware Acceleration/Encryption

- **Nest Accelerator unit (NX)**
  - Comprises cryptographic and memory compression/decompression engines (co-processors) with support hardware.
    - AES (Advanced Encryption Standard) engine
    - SHA (Secure Hash Algorithm) engine
    - RNG (Random Number Generator)
    - 842 Compression/Decompression
      - IBM-proprietary algorithm
      - Performance shows >300% performance improvement with eCryptfs

- **Open for OpenPower**

- **Kernel support**
  - Driver for nx842
  - crypto API
    - zswap, dm-crypt, eCryptfs, IPsec

- **Benefits for applications**
  - userspace support in libica.
  - OpenSSL
  - Java, new crypto instructions
SMT (Simultaneous Multithreading)/Split core

- Virtualization benefits from smt/ split core
- SMT = 8 (8 threads/core) is enough?
  Need context switch if
guests number > core number, because mmu is shared in one core.
- Split core = 2, 4
  => 2 or 4 guests can run in 1 core at the same time
- Dynamic Split Core
HTM (Hardware Transactional Memory)

- **Motivation: to scale out**
  - Hardware managed atomicity to shared data
  - Light weight locking
  - Parallel thread concurrency

- **Use Cases**
  - Optimistic Execution of Lock-Based Applications
  - Transactional Programming in High-Level Languages

- **begin, end, abort, suspend, resume**

- **New in power8 for HTM**
  - New instructions mark beginning and end of transaction.
  - Hardware ensures atomicity
HTM

- How to determine potential benefits?
  - If the transactions reference a large amount of data, TM is not helpful.
  - If read-only transactions typically do not reference the same data as concurrent transactions that write data, TM may help.

- Limitations
  - No syscall
  - No reset stack/context
  - getcontext(), setcontext(), makecontext(), swapcontext(), setjmp(), and longjmp()
HTM examples

To enter a critical section (pthread_mutex_lock):

```c
if (__TM_begin(tm_buff) == 0) {
    long val = mutex->mt_lock;
    if (val == UL_FREE) {
        /* Free */
        /* Enter critical section using TM */
        return 0;
    }
    /* Busy */
    __TM_abort();
}
/* Busy */
__TM_abort();
```

else {
    /* Not in a transaction */
    ...
    /* Giving up - Not using TM - Need to acquire lock */
    ... <acquire lock> ...
    /* Enter critical section holding lock - Not using TM */
}

To exit critical section (pthread_mutex_unlock):

```c
if (__TM_end() == 0) {
    /* Was inside transaction - No need to do anything */
    return 0;
}
else {
    /* Must have acquired lock instead of using TM */
    ... <release lock> ...
}
```
HTM examples

num_retries = 10;
while (1) {

    if (__TM_begin (TM_buff) == 0) { /* Transaction State Initiated. */
        if (shmlock.isLocked()) {
            num_retries=0; /*resort to locks*/
            ___TM_abort();
        }
        sum = a + b; //add transaction code here.
        ___TM_end ();
        break;
    }

    else {
        /* Transaction Failed. Use locks if the transaction failure is “persistent” or tried too many times. */
        if (num_retries-- <= 0 || ___TM_is_failure_persistent (TM_buff)) {
            /*resort to conventional lock*/
            while (shmlock.readLock() != 0);
            sum = a + b;
            shmlock.unLock();
            break;
        }
    }

    .......
}

© 2014 IBM Corporation
# Energy Management – Idle states

- **Idle states**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Power Savings</th>
<th>Exit Overhead</th>
</tr>
</thead>
<tbody>
<tr>
<td>Snooze</td>
<td>Software defined polling state</td>
<td>Low</td>
<td>Low</td>
</tr>
<tr>
<td>Nap</td>
<td>Core is clockgated</td>
<td>High</td>
<td>Low</td>
</tr>
<tr>
<td>Fastsleep</td>
<td>Voltage to Core and L2 cache is brought to minimum</td>
<td>Higher</td>
<td>High</td>
</tr>
<tr>
<td>Winkle</td>
<td>Voltage to Core, L2 and L3 cache is turned off. Enabled only for offline CPUs</td>
<td>Highest</td>
<td>Highest</td>
</tr>
</tbody>
</table>

- `/sys/devices/system/cpu/cpuX/cpuidle/stateY`
- `cpupower idle-set -d <state_number>`
- `cpupower idle-set -e <state_number>`
- `cpupower idle-info`
Energy Management - DVFS

- Dynamic Voltage and Frequency Scaling/DVFS

<table>
<thead>
<tr>
<th>Name</th>
<th>Pstate at low load</th>
<th>Pstate at high load</th>
</tr>
</thead>
<tbody>
<tr>
<td>OnDemand</td>
<td>Lowest</td>
<td>Highest</td>
</tr>
<tr>
<td>Userspace</td>
<td>Default:Nominal</td>
<td>Default: Nominal</td>
</tr>
<tr>
<td>Powersave</td>
<td>Lowest</td>
<td>Lowest</td>
</tr>
<tr>
<td>Performance</td>
<td>Highest</td>
<td>Highest</td>
</tr>
</tbody>
</table>

- The higher the pstates, the more performance the CPU gets, but at cost of higher power consumption.

- `/sys/devices/system/cpu/cpuX/cpufreq`
Energy Management – tuned-adm

- CPU Power Management using tuned-adm

- tuned-adm
  - Balanced profile for workloads which require a fine balance between performance and power savings
  - Latency-performance profile for latency sensitive workloads
  - Throughput-performance profile for workloads that expect steady performance
  - Powers avings profile for workloads that care mostly about power savings
Java Performance enhancements

- IBM Java will transparently provide support for EBB/BHRB profiling during JVM startup to improve JIT code optimization.

- **PMU** (Performance Monitor Unit): New types of counters

- **EBB** (Event-Based Branching)
  - Generates event-based exceptions when a certain event criteria is met. Following an EBB exception, the BESCR register tells which kind of event triggered the exception.
  - Asynchronous userspace interrupt based on events.
  - PMU EBB event, work together with PMU.
  - Signal handler like

- **BHRB** (Branch History Rolling Buffer)
  - Rolling list of recent branches
  - Can be used as a call trace leading up to Performance Monitor interrupt
  - Can be used to detect branch prediction problems

- **DSCR** (Data Stream Control Register)
  - DSCR is a register which controls the prefetching data between RAM and caches
  - Dynamic degree of aggressiveness, per process
Summary

- POWER8 is the first OpenPOWER CPU

**Open**
- BE/LE kvm
- CAPI
- Hardware accelerator (Encrypt/Compression)

**Power**
- Parallel programming: smt/split core, HTM
- Java Performance enhancement
- Energy management
Thanks
Q&A