Persistent Memory in Linux

Tom Coughlan (with Thanks to Jeff Moyer), Red Hat
Since our last meeting...

- Fedora 23 shipped with pmem support
  - (min. kernel version 4.4)
- RHEL 7.3 shipped with
  - Full support for pmem as a block device
  - Technical Preview for pmem in Direct Access (DAX) mode
Major Kernel Subsystems

- System Call Interface
- Process Control
- VFS
  - ext2
  - ext4
  - xfs
  - ...
- Virtual Memory
- Block Layer
- Network Core
- Platform Support (ACPI, etc)
- Device Drivers
- Architecture Support
Modified Kernel Subsystems

System Call Interface

Process Control

VFS
- ext2
- ext4
- xfs
- ...

Virtual Memory

Block Layer

Network Core

Platform Support (ACPI, etc)

Device Drivers

Architecture Support
### For example, four 8 GiB NVDIMMs, configured in the firmware as two 16 GiB interleave sets:

```bash
# ndctl list
[
  {
    "dev":"namespace1.0",
    "mode":"raw",
    "size":17179869184,
    "blockdev":"pmem1"
  },
  {
    "dev":"namespace0.0",
    "mode":"raw",
    "size":17179869184,
    "blockdev":"pmem0"
  }
]
```
PMEM Namespace Configurations

- Default, but don't use it!
## PMEM Namespace Configurations

<table>
<thead>
<tr>
<th>RAW</th>
<th>SECTOR</th>
<th>MEMORY</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default, but don't use it!</td>
<td>Atomic Sector Updates (provided by the btt)</td>
<td>Configurable Sector Size (includes DIF/DIX)</td>
</tr>
</tbody>
</table>

© 2017 SNIA Persistent Memory Summit. All Rights Reserved.
# PMEM Namespace Configurations

<table>
<thead>
<tr>
<th>RAW</th>
<th>SECTOR</th>
<th>MEMORY</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default, but don't use it!</td>
<td>Atomic Sector Updates (provided by the btt)</td>
<td>DAX Support</td>
</tr>
<tr>
<td></td>
<td>Configurable Sector Size (includes DIF/DIX)</td>
<td>Requires space for kernel page structures</td>
</tr>
</tbody>
</table>
“Memory” Namespaces

- Need to reserve space for kernel page structures
  - 64 bytes per 4 KiB page
- If the pmem space is small (and expensive), store the page structures in DRAM
  - e.g. 32 GiB pmem => 512 MiB
- If the pmem space is large, store the page structures in pmem
  - e.g. 1 TiB pmem => 16 GiB
Configuring DAX

```
# ndctl list
[
{
"dev":"namespace0.0",
"mode":"raw",
"size":17179869184,
"blockdev":"pmem0"
}
]

# fdisk -l /dev/pmem0

Disk /dev/pmem0: 17.2 GB, 17179869184 bytes, 67 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 4096 bytes
I/O size (minimum/optimal): 4096 bytes / 4096 bytes
```
Configuring DAX using DRAM to host struct pages

```
# ndctl create-namespace -f -e namespace0.0 --mode=memory --map=mem
{
  "dev":"namespace0.0",
  "mode":"memory",
  "size":17177772032,
  "uuid":"3c88e67f-8b25-4661-adf9-f0ed390cbd6a",
  "blockdev":"pmem0"
}

# fdisk -l /dev/pmem0

Disk /dev/pmem0: 17.2 GB, 17177772032 bytes, 33550336 sectors
   Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 4096 bytes
   I/O size (minimum/optimal): 4096 bytes / 4096 bytes
```
Configuring DAX using DRAM to host struct pages

# ndctl create-namespace -f -e namespace0.0 --mode=memory --map=mem

```json
{
  "dev": "namespace0.0",
  "mode": "memory",
  "size": 17177772032,
  "uuid": "3c88e67f-8b25-4661-adf9-f0ed390cbd6a",
  "blockdev": "pmem0"
}
```

# fdisk -l /dev/pmem0

```
Disk /dev/pmem0: 17.2 GB, 17177772032 bytes, 33550336 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 4096 bytes
I/O size (minimum/optimal): 4096 bytes / 4096 bytes
```
Configuring DAX using the NVDIMM to host struct pages

```
# ndctl create-namespace -f -e namespace0.0 --mode=memory --map=dev

    {
        "dev":"namespace0.0",
        "mode":"memory",
        "size":16909336576,
        "uuid":"b5c852b2-75c2-4e8b-94b2-06694d6ff243",
        "blockdev":"pmem0"
    }

# fdisk -l /dev/pmem0

Disk /dev/pmem0: 16.9 GB, 16909336576 bytes, 33026048 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 4096 bytes
I/O size (minimum/optimal): 4096 bytes / 4096 bytes
```
Convert to a BTT Namespace...

```
# ndctl list
[
  {
    "dev":"namespace0.0",
    "mode":"memory",
    "size":17179869184,
    "blockdev":"pmem0"
  }
]
```
Configuring a BTT Namespace

```bash
# ndctl create-namespace -f -e namespace0.0 -m sector
{
    "dev":"namespace0.0",
    "mode":"sector",
    "uuid":"9e24b27a-bb46-44ad-b7fb-81ebfee0a3d6",
    "sector_size":4096,
    "blockdev":"pmem0s"
}

# fdisk -l /dev/pmem0s

Disk /dev/pmem0s: 17.2 GB, 17162027008 bytes, 4189948 sectors
Units = sectors of 1 * 4096 = 4096 bytes
Sector size (logical/physical): 4096 bytes / 4096 bytes
I/O size (minimum/optimal): 4096 bytes / 4096 bytes
```
File System Setup for DAX

# mkfs -t xfs -d su=1g,sw=1 /dev/pmem0
# mount -t xfs -o dax /dev/pmem0 /mnt/dax

# mkfs -t ext4 /dev/pmem0
# mount -t ext4 -o dax /dev/pmem0 /mnt/dax

NOTES:

- Partitions on DAX devices must be aligned on page boundaries
- XFS requires atomic sector writes for its journal
  - recommend leaving XFS CRC enabled when DAX is in use
- Inconsistent Behavior:
  - ext4 fails if DAX unavailable
  - XFS logs a message
Next Steps:

- If you don’t have pmem hardware yet...

  - kernel parameter `memmap=XG!YG` specifies a range of RAM to emulate pmem
    - e.g., `memmap=192G!1024G` will reserve 192G starting at 1024G
    - on boot, one pmem device will appear for each range specified
  
  - more info: nvdimm kernel wiki,
    https://nvdimm.wiki.kernel.org/how_to_choose_the_correct_memmap_kernel_parameter_for_pmem_on_your_system

- To get started on application devel...

  - Use NVML, a suite of libraries for pmem programming
    - `libpmem` and `libmemobj` in particular
  
  - for more information: nvml blog, http://pmem.io/blog/
Future Work

- Make DAX fully supported for XFS and ext4
- Install / Boot
  - Substantial work: UEFI, potentially ACPI, Anaconda.
  - Not planned for RHEL 7.4
- Support (2 MiB) huge pages
  - this is in the upstream kernel
  - investigating the feasibility for RHEL 7
  - (support for 1 GiB huge pages will require substantial work upstream - not likely for RHEL 7)
- Allow virtual machines to have access to /dev/pmem0 devices
- More refined error handling
- Port ext4 and XFS perf. improvements (iomap) to ext4-dax and xfs-dax
- Additional performance work...
Summary

- Persistent Memory products available today
  - Capacities about to explode
- Linux is prepared
  - pmem driver stack, DAX, ext4, xfs, etc.
- RHEL is prepared
  - ndctl & other tools, validation
References

- ProgModel - http://www.snia.org/tech_activities/standards/curr_standards/npm
- SNIA_NVDIMM - http://www.snia.org/forums/ssi/NVDIMM
- Managing pmem – http://events.linuxfoundation.org/sites/events/files/slides/Managing%20Persistent%20Memory_0.pdf
- WIKI – https://nvdimm.wiki.kernel.org/