```
==============
Memory Hotplug
==============

Created:          Jul 28 2007
Add description of notifier of memory hotplug Oct 11 2007

This document is about memory hotplug including how-to-use and current status.
Because Memory Hotplug is still under development, contents of this text will
be changed often.

1. Introduction
  1.1 purpose of memory hotplug
  1.2. Phases of memory hotplug
  1.3. Unit of Memory online/offline operation
2. Kernel Configuration
3. sysfs files for memory hotplug
4. Physical memory hot-add phase
  4.1 Hardware(Firmware) Support
  4.2 Notify memory hot-add event by hand
5. Logical Memory hot-add phase
  5.1. State of memory
  5.2. How to online memory
6. Logical memory remove
  6.1 Memory offline and ZONE_MOVABLE
  6.2. How to offline memory
7. Physical memory remove
8. Memory hotplug event notifier
9. Future Work List

Note(1): x86_64's has special implementation for memory hotplug.
         This text does not describe it.
Note(2): This text assumes that sysfs is mounted at /sys.



---------------
1. Introduction
---------------

1.1 purpose of memory hotplug
------------
```

```
Memory Hotplug allows users to increase/decrease the amount of memory.
Generally, there are two purposes.

(A) For changing the amount of memory.
    This is to allow a feature like capacity on demand.
(B) For installing/removing DIMMs or NUMA-nodes physically.
    This is to exchange DIMMs/NUMA-nodes, reduce power consumption, etc.

(A) is required by highly virtualized environments and (B) is required by
hardware which supports memory power management.

Linux memory hotplug is designed for both purpose.



1.2. Phases of memory hotplug
---------------
There are 2 phases in Memory Hotplug.
  1) Physical Memory Hotplug phase
  2) Logical Memory Hotplug phase.

The First phase is to communicate hardware/firmware and make/erase
environment for hotplugged memory. Basically, this phase is necessary
for the purpose (B), but this is good phase for communication between
highly virtualized environments too.

When memory is hotplugged, the kernel recognizes new memory, makes new memory
management tables, and makes sysfs files for new memory's operation.

If firmware supports notification of connection of new memory to OS,
this phase is triggered automatically. ACPI can notify this event. If not,
"probe" operation by system administration is used instead.
(see Section 4.).

Logical Memory Hotplug phase is to change memory state into
available/unavailable for users. Amount of memory from user's view is
changed by this phase. The kernel makes all memory in it as free pages
when a memory range is available.

In this document, this phase is described as online/offline.

Logical Memory Hotplug phase is triggered by write of sysfs file by system
```

```
administrator. For the hot-add case, it must be executed after Physical Hotplug
phase by hand.
(However, if you writes udev's hotplug scripts for memory hotplug, these
 phases can be execute in seamless way.)


1.3. Unit of Memory online/offline operation
------------
Memory hotplug uses SPARSEMEM memory model. SPARSEMEM divides the whole memory
into chunks of the same size. The chunk is called a "section". The size of
a section is architecture dependent. For example, power uses 16MiB, ia64 uses
1GiB. The unit of online/offline operation is "one section". (see Section 3.)

To determine the size of sections, please read this file:

/sys/devices/system/memory/block_size_bytes

This file shows the size of sections in byte.

-----------------------
2. Kernel Configuration
-----------------------
To use memory hotplug feature, kernel must be compiled with following
config options.

- For all memory hotplug
    Memory model -> Sparse Memory  (CONFIG_SPARSEMEM)
    Allow for memory hot-add       (CONFIG_MEMORY_HOTPLUG)

- To enable memory removal, the followings are also necessary
    Allow for memory hot remove    (CONFIG_MEMORY_HOTREMOVE)
    Page Migration                 (CONFIG_MIGRATION)

- For ACPI memory hotplug, the followings are also necessary
    Memory hotplug (under ACPI Support menu) (CONFIG_ACPI_HOTPLUG_MEMORY)
    This option can be kernel module.

- As a related configuration, if your box has a feature of NUMA-node hotplug
  via ACPI, then this option is necessary too.
    ACPI0004,PNP0A05 and PNP0A06 Container Driver (under ACPI Support menu)
    (CONFIG_ACPI_CONTAINER).
```



```
    This option can be kernel module too.

--------------------------------
4 sysfs files for memory hotplug
--------------------------------
All sections have their device information under /sys/devices/system/memory as

/sys/devices/system/memory/memoryXXX
(XXX is section id.)

Now, XXX is defined as start_address_of_section / section_size.

For example, assume 1GiB section size. A device for a memory starting at
0x100000000 is /sys/device/system/memory/memory4
(0x100000000 / 1Gib = 4)
This device covers address range [0x100000000 ... 0x140000000)

Under each section, you can see 4 files.

/sys/devices/system/memory/memoryXXX/phys_index
/sys/devices/system/memory/memoryXXX/phys_device
/sys/devices/system/memory/memoryXXX/state
/sys/devices/system/memory/memoryXXX/removable

'phys_index' : read-only and contains section id, same as XXX.
'state'      : read-write
               at read:  contains online/offline state of memory.
               at write: user can specify "online", "offline" command
'phys_device': read-only: designed to show the name of physical memory device.
               This is not well implemented now.
'removable'  : read-only: contains an integer value indicating
               whether the memory section is removable or not
               removable.  A value of 1 indicates that the memory
               section is removable and a value of 0 indicates that
               it is not removable.

NOTE:
  These directories/files appear after physical memory hotplug phase.

If CONFIG_NUMA is enabled the memoryXXX/ directories can also be accessed
via symbolic links located in the /sys/devices/system/node/node* directories.
```

For example:
/sys/devices/system/node/node0/memory9 -> ../../memory/memory9

A backlink will also be created:
/sys/devices/system/memory/memory9/node0 -> ../../node/node0

----
4. Physical memory hot-add phase

4.1 Hardware(Firmware) Support

On x86_64/ia64 platform, memory hotplug by ACPI is supported.

In general, the firmware (ACPI) which supports memory hotplug defines memory class object of _HID "PNP0C80". When a notify is asserted to PNP0C80, Linux's ACPI handler does hot-add memory to the system and calls a hotplug udev script. This will be done automatically.

But scripts for memory hotplug are not contained in generic udev package(now). You may have to write it by yourself or online/offline memory by hand. Please see "How to online memory", "How to offline memory" in this text.

If firmware supports NUMA-node hotplug, and defines an object _HID "ACPI0004", "PNP0A05", or "PNP0A06", notification is asserted to it, and ACPI handler calls hotplug code for all of objects which are defined in it. If memory device is found, memory hotplug code will be called.

4.2 Notify memory hot-add event by hand

In some environments, especially virtualized environment, firmware will not notify memory hotplug event to the kernel. For such environment, "probe" interface is supported. This interface depends on CONFIG_ARCH_MEMORY_PROBE.

Now, CONFIG_ARCH_MEMORY_PROBE is supported only by powerpc but it does not contain highly architecture codes. Please add config if you need "probe" interface.

Probe interface is located at

```
/sys/devices/system/memory/probe

You can tell the physical address of new memory to the kernel by

% echo start_address_of_new_memory > /sys/devices/system/memory/probe

Then, [start_address_of_new_memory, start_address_of_new_memory + section_size)
memory range is hot-added. In this case, hotplug script is not called (in
current implementation). You'll have to online memory by yourself.
Please see "How to online memory" in this text.



------------------------------
5. Logical Memory hot-add phase
------------------------------

5.1. State of memory
------------
To see (online/offline) state of memory section, read 'state' file.

% cat /sys/device/system/memory/memoryXXX/state


If the memory section is online, you'll read "online".
If the memory section is offline, you'll read "offline".


5.2. How to online memory
------------
Even if the memory is hot-added, it is not at ready-to-use state.
For using newly added memory, you have to "online" the memory section.

For onlining, you have to write "online" to the section's state file as:

% echo online > /sys/devices/system/memory/memoryXXX/state

After this, section memoryXXX's state will be 'online' and the amount of
available memory will be increased.

Currently, newly added memory is added as ZONE_NORMAL (for powerpc, ZONE_DMA).
```

This may be changed in future.

----
6. Logical memory remove

6.1 Memory offline and ZONE_MOVABLE

Memory offlining is more complicated than memory online. Because memory offline has to make the whole memory section be unused, memory offline can fail if the section includes memory which cannot be freed.

In general, memory offline can use 2 techniques.

(1) reclaim and free all memory in the section.
(2) migrate all pages in the section.

In the current implementation, Linux's memory offline uses method (2), freeing all pages in the section by page migration. But not all pages are migratable. Under current Linux, migratable pages are anonymous pages and page caches. For offlining a section by migration, the kernel has to guarantee that the section contains only migratable pages.

Now, a boot option for making a section which consists of migratable pages is supported. By specifying "kernelcore=" or "movablecore=" boot option, you can create ZONE_MOVABLE...a zone which is just used for movable pages.
(See also Documentation/kernel-parameters.txt)

Assume the system has "TOTAL" amount of memory at boot time, this boot option creates ZONE_MOVABLE as following.

```
1) When kernelcore=YYYY boot option is used,
  Size of memory not for movable pages (not for offline) is YYYY.
  Size of memory for movable pages (for offline) is TOTAL-YYYY.

2) When movablecore=ZZZZ boot option is used,
  Size of memory not for movable pages (not for offline) is TOTAL - ZZZZ.
  Size of memory for movable pages (for offline) is ZZZZ.
```

```
Note) Unfortunately, there is no information to show which section belongs
to ZONE_MOVABLE. This is TBD.


6.2. How to offline memory
------------
You can offline a section by using the same sysfs interface that was used in
memory onlining.

% echo offline > /sys/devices/system/memory/memoryXXX/state

If offline succeeds, the state of the memory section is changed to be "offline".
If it fails, some error core (like -EBUSY) will be returned by the kernel.
Even if a section does not belong to ZONE_MOVABLE, you can try to offline it.
If it doesn't contain 'unmovable' memory, you'll get success.

A section under ZONE_MOVABLE is considered to be able to be offlined easily.
But under some busy state, it may return -EBUSY. Even if a memory section
cannot be offlined due to -EBUSY, you can retry offlining it and may be able to
offline it (or not).
(For example, a page is referred to by some kernel internal call and released
 soon.)

Consideration:
Memory hotplug's design direction is to make the possibility of memory offlining
higher and to guarantee unplugging memory under any situation. But it needs
more work. Returning -EBUSY under some situation may be good because the user
can decide to retry more or not by himself. Currently, memory offlining code
does some amount of retry with 120 seconds timeout.

-------------------------
7. Physical memory remove
-------------------------
Need more implementation yet....
 - Notification completion of remove works by OS to firmware.
 - Guard from remove if not yet.

--------------------------------
8. Memory hotplug event notifier
--------------------------------
```

```
Memory hotplug has event notifer. There are 6 types of notification.

MEMORY_GOING_ONLINE
  Generated before new memory becomes available in order to be able to
  prepare subsystems to handle memory. The page allocator is still unable
  to allocate from the new memory.

MEMORY_CANCEL_ONLINE
  Generated if MEMORY_GOING_ONLINE fails.

MEMORY_ONLINE
  Generated when memory has successfully brought online. The callback may
  allocate pages from the new memory.

MEMORY_GOING_OFFLINE
  Generated to begin the process of offlining memory. Allocations are no
  longer possible from the memory but some of the memory to be offlined
  is still in use. The callback can be used to free memory known to a
  subsystem from the indicated memory section.

MEMORY_CANCEL_OFFLINE
  Generated if MEMORY_GOING_OFFLINE fails. Memory is available again from
  the section that we attempted to offline.

MEMORY_OFFLINE
  Generated after offlining memory is complete.

A callback routine can be registered by
  hotplug_memory_notifier(callback_func, priority)

The second argument of callback function (action) is event types of above.
The third argument is passed by pointer of struct memory_notify.

struct memory_notify {
       unsigned long start_pfn;
       unsigned long nr_pages;
       int status_change_nid;
}

start_pfn is start_pfn of online/offline memory.
nr_pages is # of pages of online/offline memory.
```

```
status_change_nid is set node id when N_HIGH_MEMORY of nodemask is (will be)
set/clear. It means a new(memoryless) node gets new memory by online and a
node loses all memory. If this is -1, then nodemask status is not changed.
If status_changed_nid >= 0, callback should create/discard structures for the
node if necessary.

--------------
9. Future Work
--------------
  - allowing memory hot-add to ZONE_MOVABLE. maybe we need some switch like
    sysctl or new control file.
  - showing memory section and physical device relationship.
  - showing memory section is under ZONE_MOVABLE or not
  - test and make it better memory offlining.
  - support HugeTLB page migration and offlining.
  - memmap removing at memory offline.
  - physical remove memory.
```