LinuxCon Japan 2016

FUJITSU
shaping tomorrow with you

Address Range Memory Mirroring

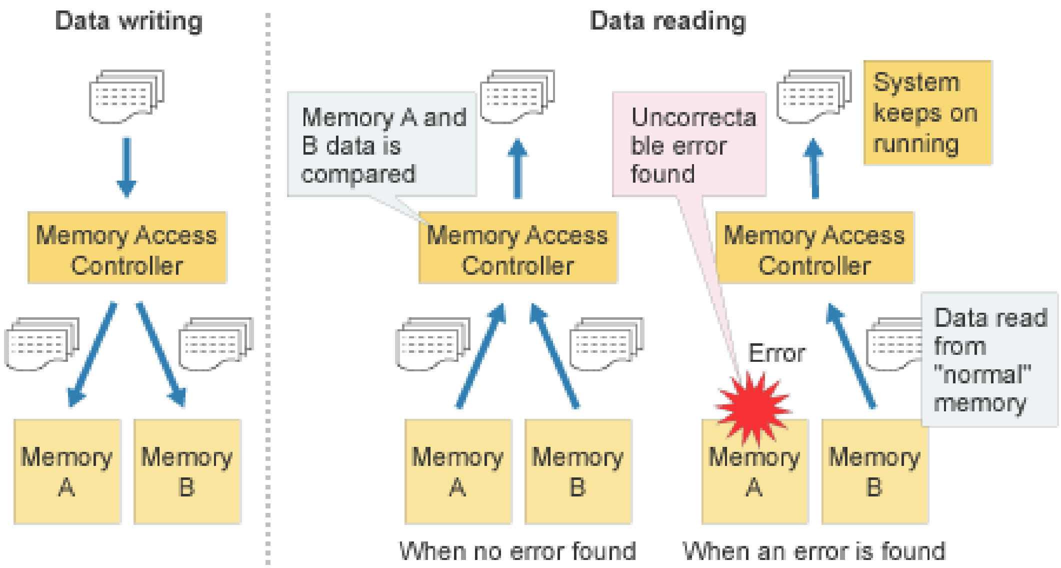
July 13, 2016
Taku Izumi
FUJITSU LIMITED

Table of Contents

Address Range Mirroring overview

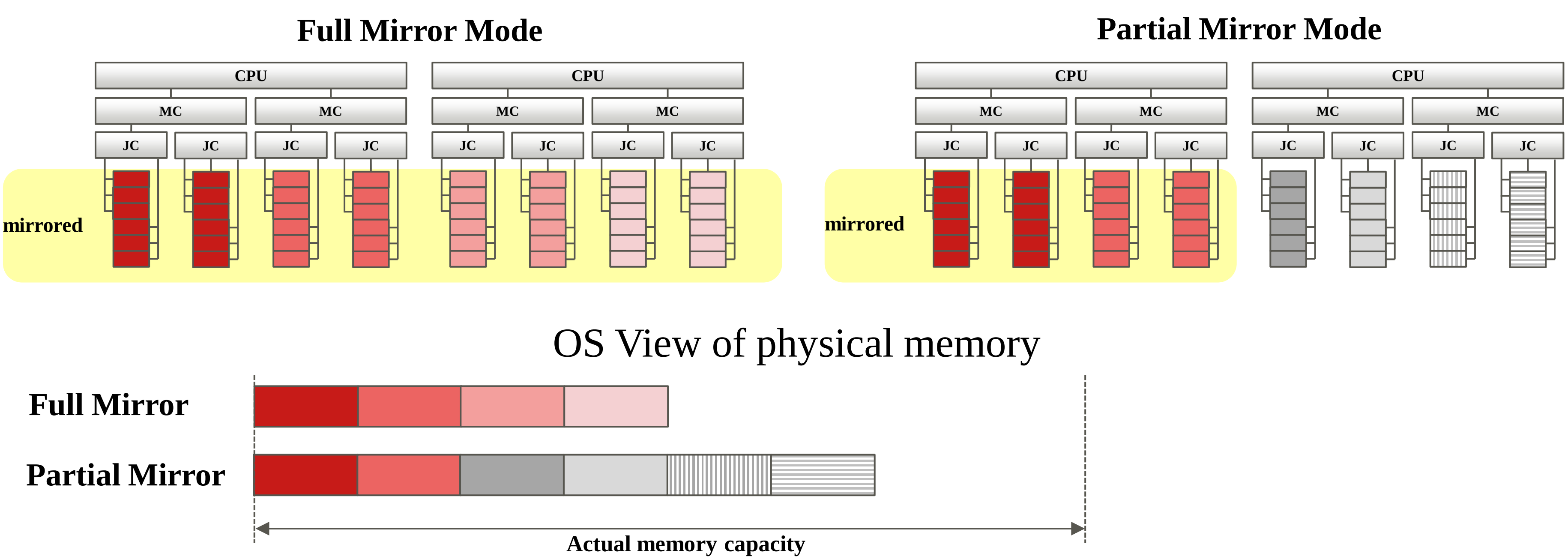
Memory Mirroring

- Memory RAS feature on Xeon family-based systems
- Provides memory redundancy
  - Data writing
    - written to both sides of the memory mirror at the same time (Memory A & B)
  - Data reading
    - Memory Controller reconfirms data validity by comparing data
    - If an uncorrectable error is detected in Memory A, data in Memory B is used for the read operation [tolerance for UCE]



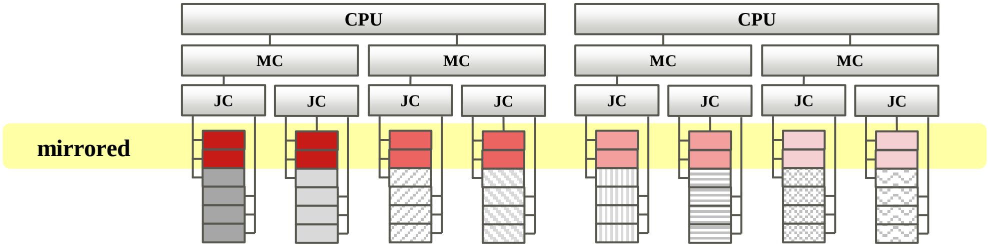
Traditional Memory Mirroring

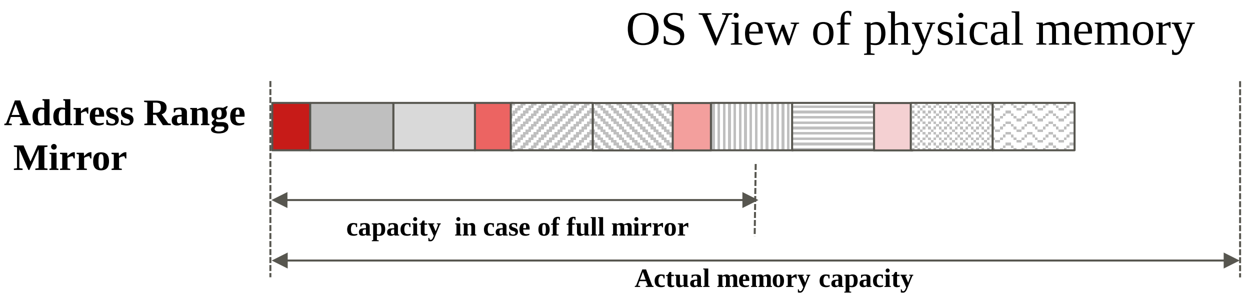
- Full Mirror Mode
  - Pros: Transparent to OS
  - Cons: Halves memory capacity available to OS
- Partial Mirror Mode
  - Pros: Transparent to OS and Keep More capacity than Full Mirror Mode
  - Cons: Mirrored range is one-sided



Address Range Mirroring

- New memory RAS feature on Haswell EX based systems
- Allows high granularity of mirroring
  - Configurable the amount of mirrored memory size
    - optimize total available memory while keeping highly reliable memory range
- Distributes mirrored memory range on each NUMA node
  - Keeps NUMA performance





Address Range Mirroring cont.

- Requires OS support to fully utilize Address Range Mirroring
  - Necessary to be aware of mirrored region

- Provides Firmware-OS interface
  - UEFI Variables
    - A method to request the amount of mirrored memory
  - UEFI Memory map
    - Presents mirrored memory range on the platform

Mirrored memory size configuration

FUJITSU

- UEFI variable is used to configure mirroring
  - MirrorRequest
    - Written by the OS to request a new mirror configuration on the next boot
  - MirrorCurrent
    - Written by the firmware and read by the OS to communicate the current status of Address Range Mirroring

```
Shell> dmpstore MirrorRequest
Dump Variable MirrorRequest
Variable NV+RT+BS '7B9BE2E0-E28A-4197-AD3E-32F062F9462C:MirrorRequest' DataSize = 5
  00000000: 01 01 C4 09 00                                          *.....*
Shell> dmpstore MirrorCurrent
Dump Variable MirrorCurrent
Variable NV+RT+BS '7B9BE2E0-E28A-4197-AD3E-32F062F9462C:MirrorCurrent' DataSize = 5
  00000000: 01 01 C4 09 00                                          *.....*
```

```
typedef struct  {
   UINT8         MirrorVersion;
   BOOLEAN       MirrorMemoryBellow4GB;      // set to true to mirror memory below 4 GB
   UINT16        MirroredAmountAbove4GB;     // percentage of memory to mirror above 4GB
   UINT8         MirrorStatus;
} ADDRESS_RANGE_MIRROR_VARIABLE_DATA
```

Mirrored memory size configuration cont.

- latest efibootmgr(8) supports UEFI variables for Address Range Mirroring
  - -m: set ‘t’ to mirror memory below 4GB
  - -M: percentage memory to mirror above 4GB

```
# efibootmgr -m t -M 25.00
```

- Confirm current settings of mirroring

```
# efibootmgr
BootCurrent: 0002
Timeout: 10 seconds
BootOrder: 0002,0001,0000
Boot0000* EFI SCSI Device
Boot0001* EFI Internal Shell
Boot0002* opensuse-secureboot
MirroredPercentageAbove4G: 25.00
MirrorMemoryBelow4GB: true
```

Presentation method of mirrored range

- The information which address range is mirrored is passed via EFI memory map

```
Shell> memmap
Type       Start             End                # Pages          Attributes
BS_data    0000000000000000-0000000000000FFF  0000000000000001 000000000001000F
available  0000000000001000-000000000003FFFF  000000000000003F 000000000001000F
BS_data    0000000000040000-000000000009FFFF  0000000000000060 000000000001000F
BS_data    0000000000100000-0000000000FFFFFF  0000000000000F00 000000000001000F
available  0000000001000000-0000000035FC5FFF  0000000000034FC6 000000000001000F
RT_data    0000000035FC6000-0000000035FCCFFF  0000000000000007 800000000001000F
...
BS_data    0000000047ED2000-0000000048013FFF  0000000000000142 000000000001000F
available  0000000100000000-000000203FFFFFFF  0000000001F40000 000000000000000F
available  0000002040000000-000000302FFFFFFF  0000000000FF0000 000000000001000F
available  0000003030000000-000000304FFFFFFF  0000000000020000 000000000000000F
reserved   000000005FC00000-000000005FFFFFFF  0000000000000400 0000000000000001
reserved   0000000060000000-000000008FFFFFFF  0000000000030000 8000000000000001
```

EFI_MEMORY_MORE_RELIABLE attribute: 0x10000

- EFI_MEMORY_MORE_RELIABLE attribute in EFI Memory descriptor indicates mirrored range
  - Defined in UEFI spec 2.5

Motivation

- **Background**
  - Linux can process memory errors in user space memory
    - Just kill the affected processes, even recover if the broken page can be replaced by reading from disk
    - Avoid broken page in the future
  - Linux has no recovery path for errors encountered during kernel code execution
    - Uncorrectable Error in kernel memory would crash the system
  - Full memory mirroring is a good approach, however, as system memory capacity grows, the amount of memory lost for redundancy also grows

- **Motivation**
  - Improve high availability by avoiding uncorrectable errors in kernel memory
    - Allocate all kernel memory from mirrored memory

Current Status of linux

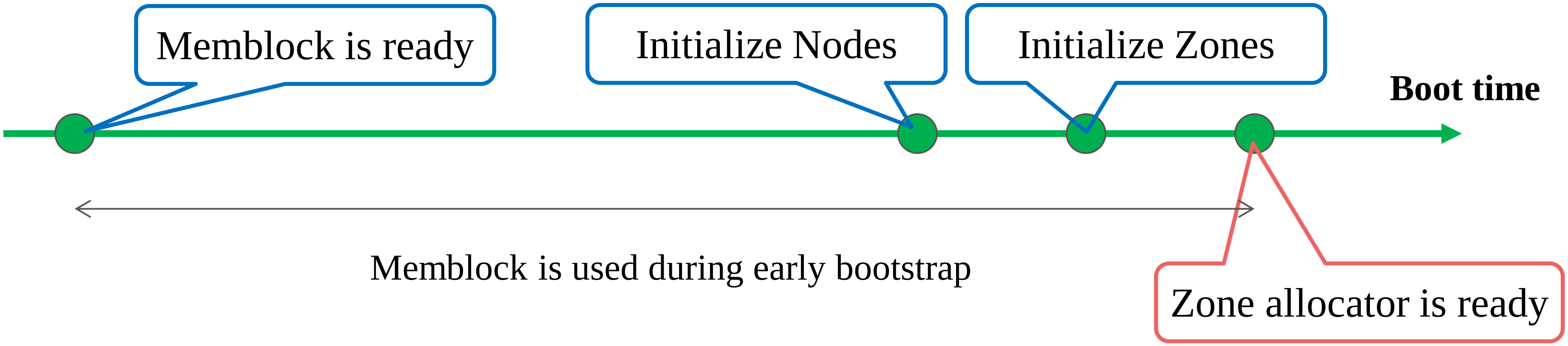
Physical memory management

Memblock

- Manage memory blocks during early bootstrap period
- Discarded after initialization and take over to Zone allocator

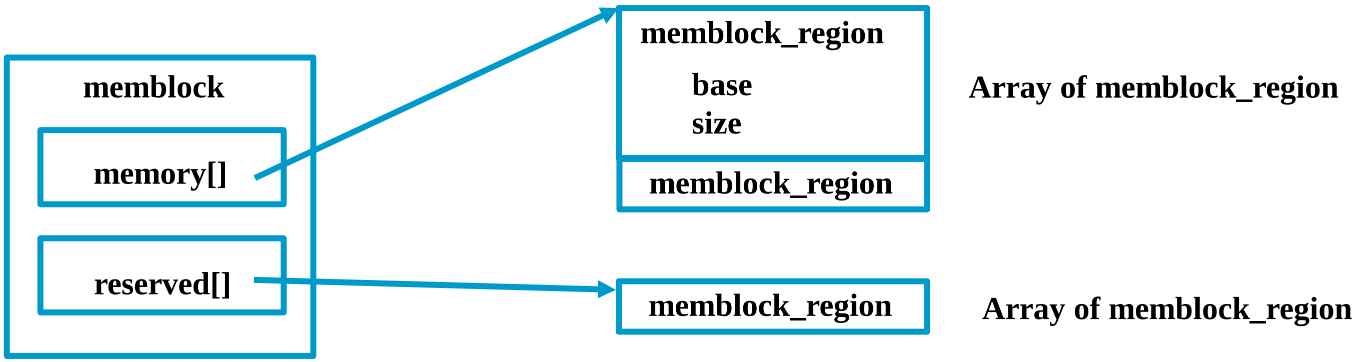
Zone Allocator

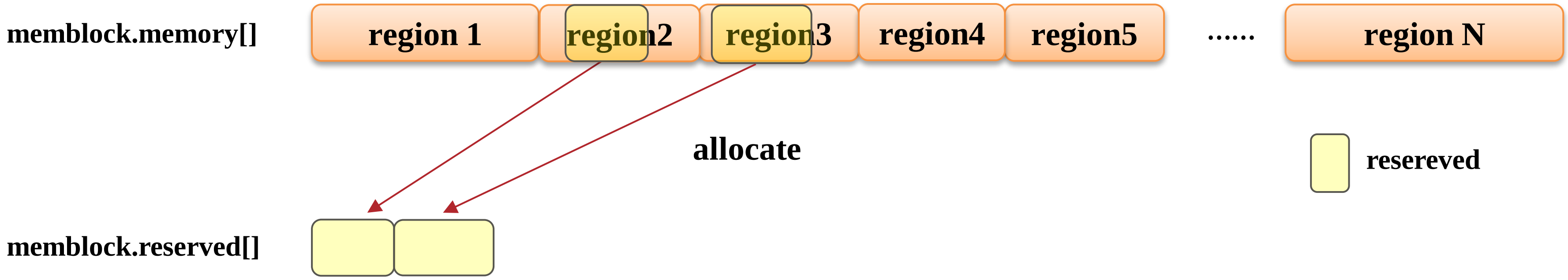
- Usual kernel memory allocator



Memblock

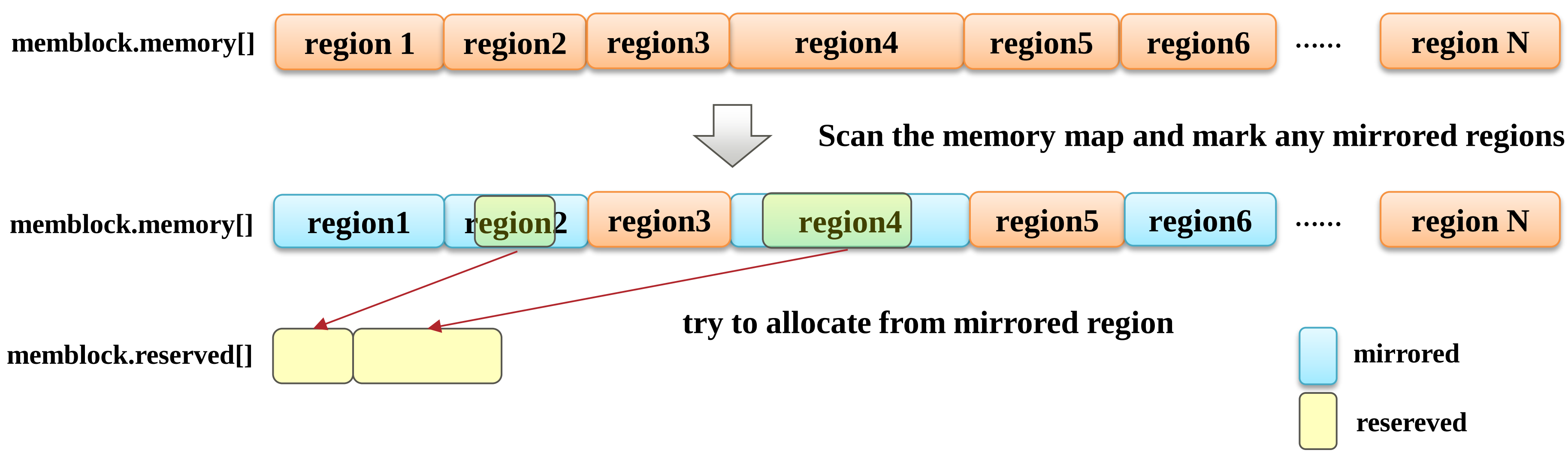
- Simply manages memory blocks
  - Consists of two arrays
    - memory: All the present memory in the system
    - reserved: Allocated memory ranges
  - Allocate by finding regions in memory && !reserved





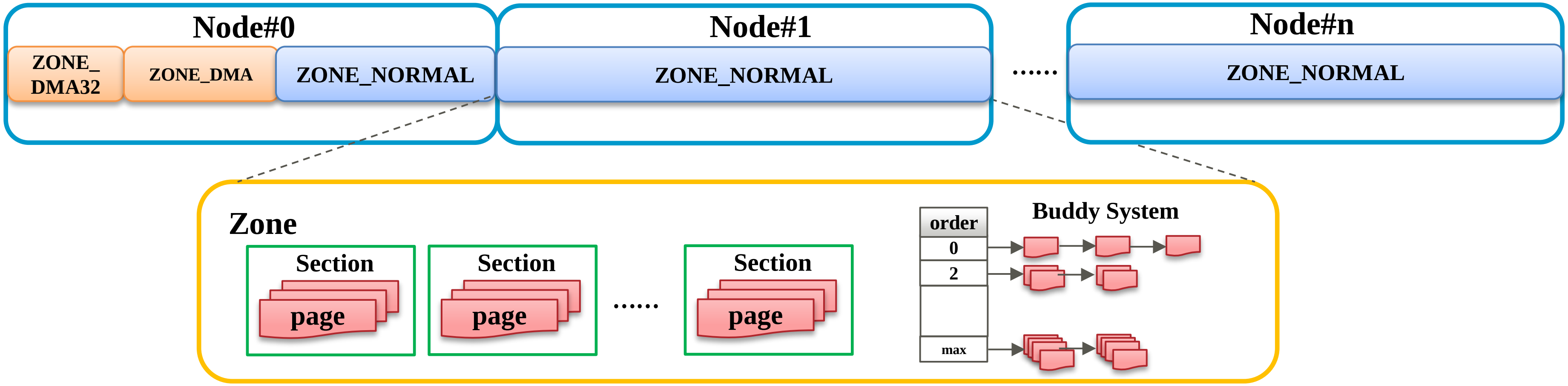
Memblock cont.

- Mirror support of Memblock has been merged into linux-4.3
  - Find mirrored region from EFI memory map information
    - Mark as MEMBLOCK_MIRROR
  - Try to allocate from mirrored region
    - If run out of mirrored memory, fall back to use non-mirrored memory



Zone allocator

- Manages memory areas called zones
  - All pages are managed by Zone
    - ZONE_DMA, ZONE_DMA32
      - used for DMA
    - ZONE_NORMAL
      - memory directly mapped, used by kernel and user space
  - Find zones suitable for memory allocation and allocate memory
- As of linux-4.3, no mirror support for zone allocator
  - Works without any regard to mirrored region

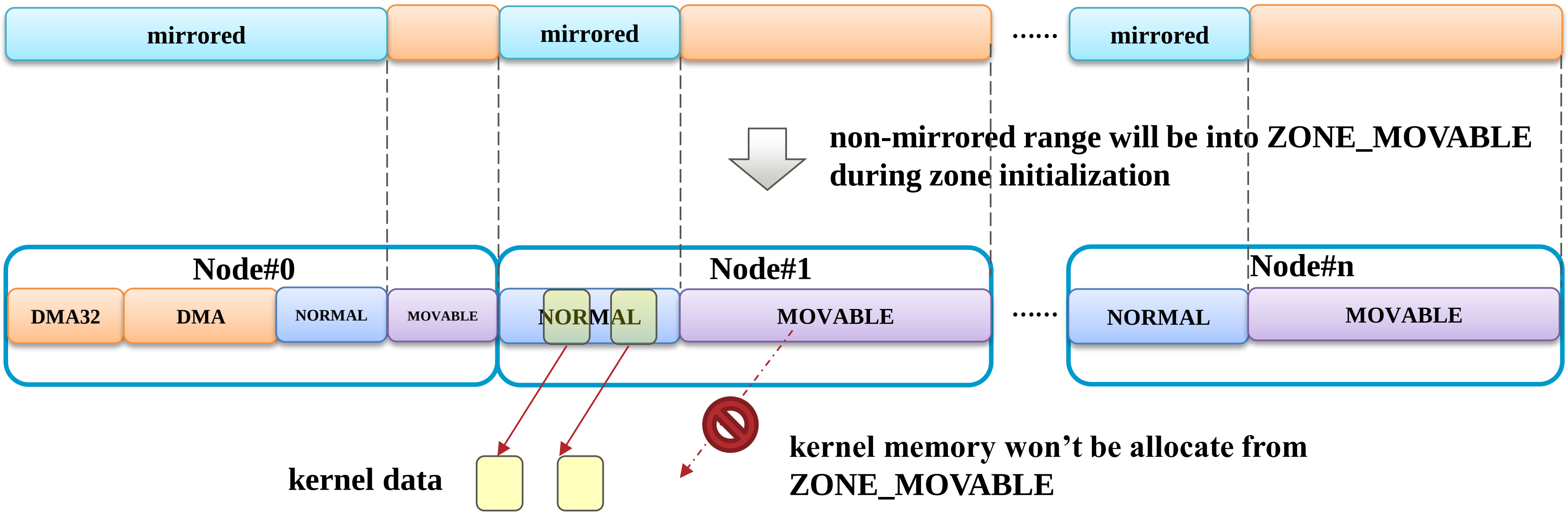


Solution for mirror support of Zone allocator

- Requirement
  - Allocate kernel memory from mirrored region
  - Allocate user memory from non-mirrored region
- Clue
  - ZONE_MOVABLE
    - Not exist by default
      - need to boot with “kernelcore” or “movablecore” specified
    - migratable memory can be allocated = users page only
      - kernel pages won’t be allocated; kernel page is NOT migratable
      - go well together memory hot-remove
- Idea
  - Arrange non-mirrored range into ZONE_MOVABLE

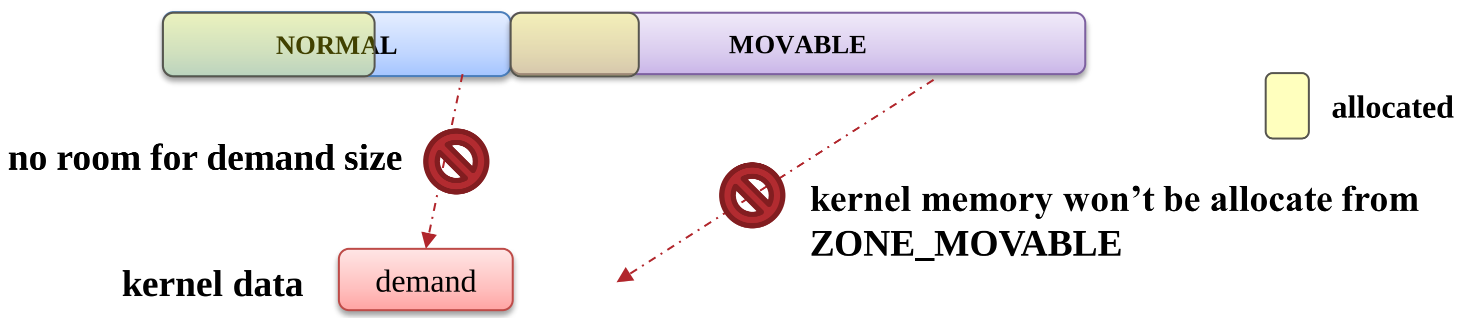
kernelcore=mirror boot option

- New in linux-4.6 for Address Range Mirroring
- By specifying “kernelcore = mirror” boot option,
  - Non-mirrored region will be arranged into ZONE_MOVABLE
  - kernel memory won’t be allocated from ZONE_MOVABLE, so it will be allocated from mirrored region



Drawback of kernelcore=mirror approach

- When running out of mirrored memory, never fall back to use non-mirrored memory
  - Possible that kernel memory is exhausted, though there is non-mirrored memory available
  - This behavior is as planned to protect kernel memory surely
  - Need sizing of total kernel memory



Future Plan

Future plan

- Handling of user’s memory
  - Prevent involuntarily user’s memory allocation from mirrored range
    - Currently user’s memory can be allocated from ZONE_NORMAL
    - Add a new __GFP_NONMIRROR allocation flag to be part of GFP_HIGHUSER_MOVABLE ?
  - Add the method that any user apps’ memory can be allocated from mirrored memory
    - Add a new MADV_MIRROR flag to the madvice(2) ?
- Handling of mirrored memory exhaustion case
  - Add fallback to non-mirrored memory option
    - In my opinion, we should not fallback. Change mirrored size to expand instead

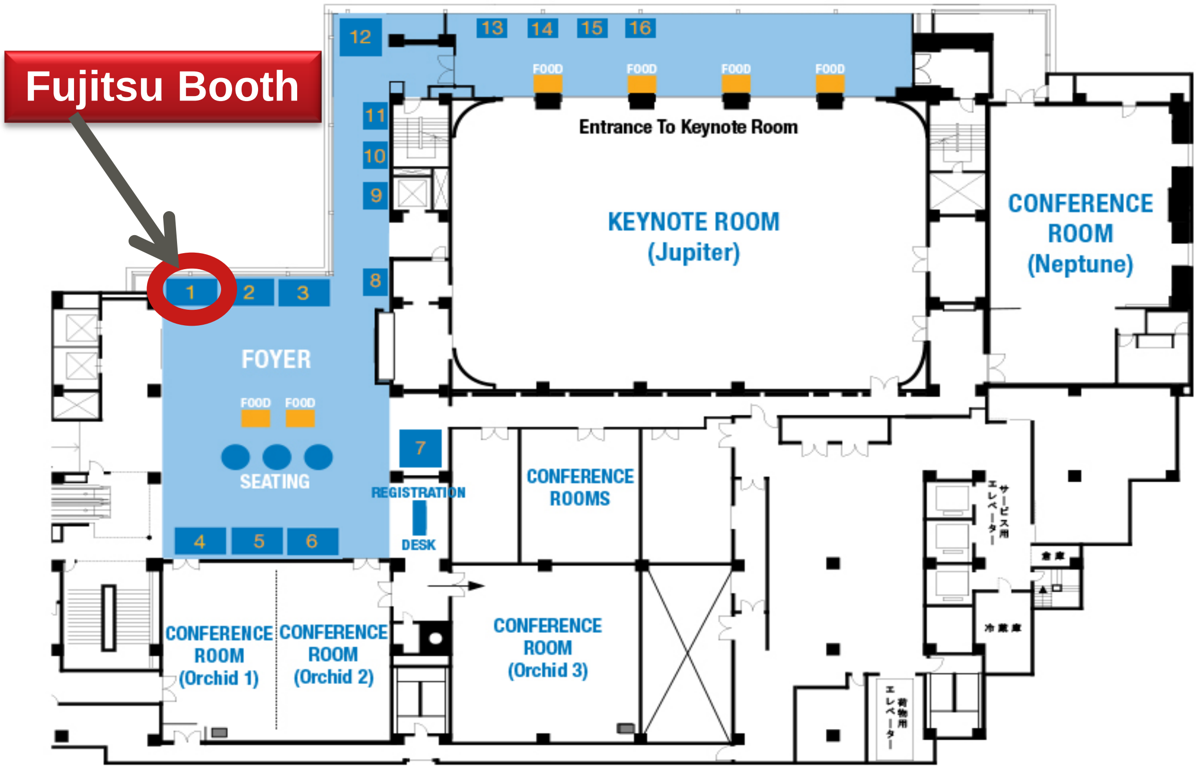
Feedback from MM summit 2016

- Got negative feedback for putting user-space program into mirrored range
  - madvice(2) is wrong interface
    - placement in mirrored memory would be mandatory
    - mirrored memory could be an opt-out resource rather than opt-in
      - But nobody would volunteer to opt-out…
  - A little messy so everything has to go there including shared libraries
    - Need restart apps?
  - Sizing for ZONE_NORMAL becomes difficult
    - second coming of low-memory problem
  - Assuming user-space program can figure the right thing to choose what needs to be mirrored is not safe
    - Security issues: some program can force the exhaustion of mirrored memory
  - Partial mirroring is simply the wrong approach
    - Simply mirroring the entire address space is easy

Summary

- Address Range Mirroring overview
  - What is Address Range mirroring
- Current status of linux
  - Current implementation for Address Range mirroring
- Future plan
  - Feedback from MM summit 2016

[PR] Fujitsu Booth



Fujitsu shares the emerging technology and trends.
Please come and experience the future innovation of technology with us.
Find out what Fujitsu delivers you today.

Cloud Monitoring software for OpenStack (based on Monasca)
- Fujitsu Software ServerView Cloud Monitoring Manager

Cloud Service Management software (Open Source Software)
- Fujitsu Software Enterprise Service Catalog Manager

Fujitsu booth is at corner of the foyer.
We are looking forward to see you in our booth .

FUJITSU

shaping tomorrow with you