OpenVMS Technical Journal V16

OpenVMS RAD Support on Integrity Servers

Shyamalendu Sarkar
Durga Prasad P V
Narayanan A N

OpenVMS RAD Support on Integrity Servers 1
Introduction 2
Background 2
Socket Local Memory 2
Cell Local Memory 3
Interleaved Memory 3
How to configure memory for NUMA? 4
Resource Affinity Domains 4
Base RAD 4
Home RAD 4
OpenVMS Memory Manager and RAD 5
RAD Preference Array 5
Page Zeroing 6
Per-RAD Non-Paged Pool 6
OpenVMS Scheduler and RAD 7
Optimized Scheduling Algorithm 7
RAD—DCL commands 7
RAD—SDA commands 8
Importance of RAD 8
When to use RAD? 9
RAD configuration guidelines 9
RAD performance monitoring 9
References 10

Introduction

OpenVMS V8.4 introduced support for Resource Affinity Domain (RAD) for Integrity servers with Non-Uniform Memory Architecture (NUMA). Cell-based Integrity servers (rx7620, rx7640, rx8620, rx8640 and Superdomes) and Integrity i2 servers (BL860c i2, BL870c i2, BL890c i2, rx2800 i2) are all based on the NUMA architecture.

In OpenVMS, a software grouping of hardware processing resources and memory with similar access characteristics is called a RAD.

The RAD support functionality is same on the Alpha NUMA servers running OpenVMS V8.4.

This article discusses the key aspects of RAD support in OpenVMS V8.4 and its subsequent software updates for Integrity servers.

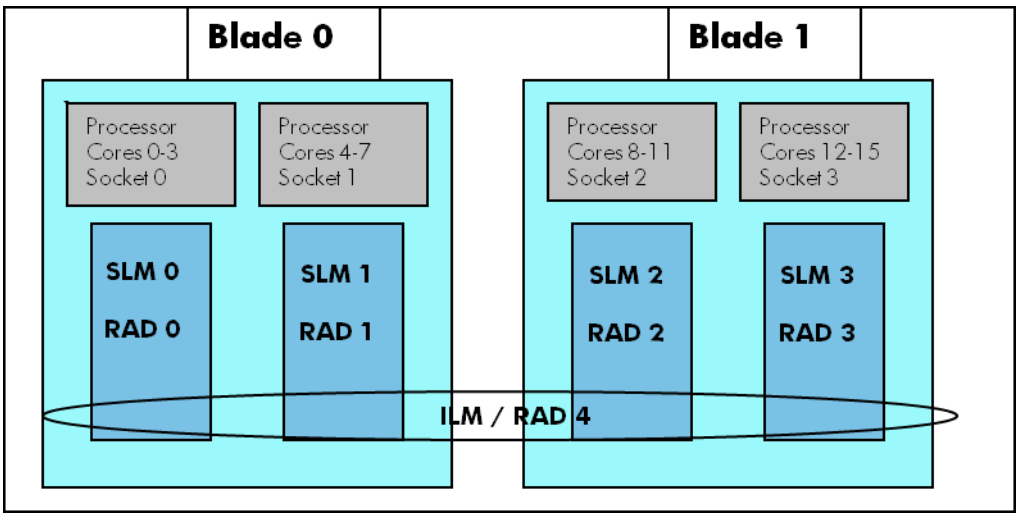
Background

Socket Local Memory

Integrity i2 servers are based on the quad-core or dual-core Tukwila processors. Each CPU socket is coupled with specific memory Dual Inline Memory Modules (DIMMs) through its integrated memory controllers. This memory is termed as Socket Local Memory (SLM).

Access to memory local to a socket is faster when compared to access to memory in a remote socket (other socket in the same blade or another blade). Figure 1 illustrates an i2 server (BL870c i2) configuration having two blades (blade 0 and blade 1). Each blade contains two quad-core sockets with their socket local memory. Memory access for socket 0 from SLM0 is faster when compared to access from SLM1, even though both the sockets are in the same blade. Similarly, memory access latency across the blades is costlier.

Figure 1: BL870c i2 Configuration with SLM/ILM and RAD

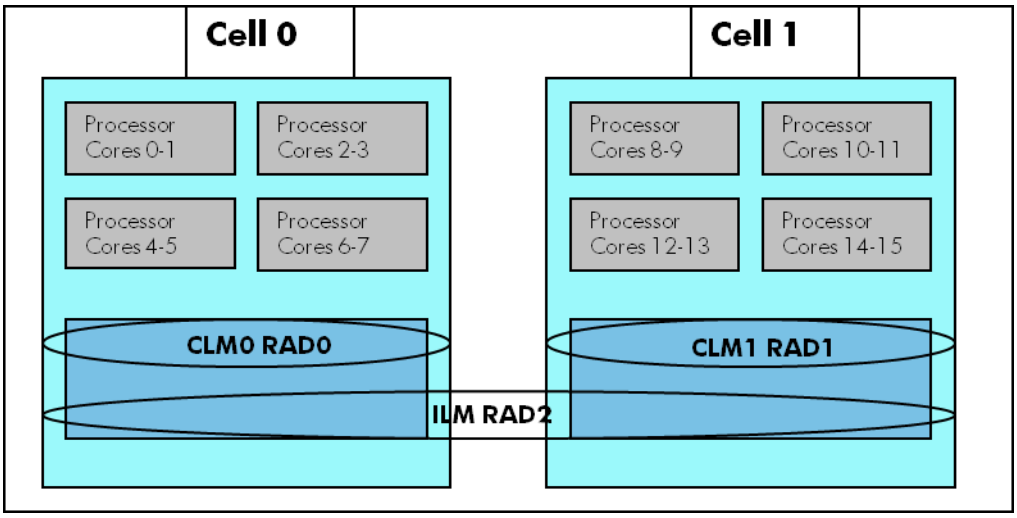


Cell Local Memory

On cell-based servers, processors and memory are grouped together into Cells. This memory is known as Cell Local memory (CLM). Access to memory local to a cell is faster when compared to access to memory outside a cell (remote cell).

Figure 2 illustrates an example for configuring two cell rx7640 system with CLM and ILM.

Figure 2: RAD Configuration on two cell rx7640 with CLM and ILM



Interleaved Memory

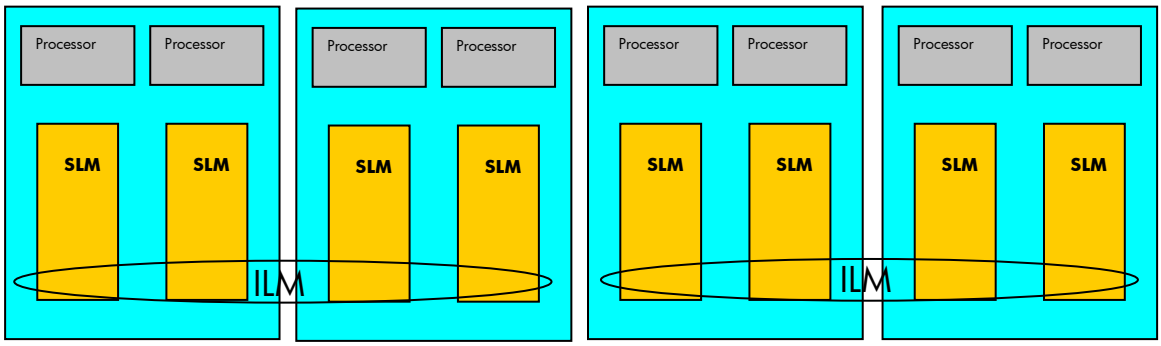
A portion of memory is taken from all the sockets or cells in a server, and it is striped in a round-robin fashion of cache-line sized chunks. It has the characteristic that memory accesses are on an average uniform. This memory is termed as the Interleaved Memory (ILM).

ILM provides consistency to memory access, irrespective of which processor in the system is accessing it.

With the exception of BL890c i2, all the other NUMA Integrity servers can have a single ILM. BL890c i2 can have two ILMs, with each ILM spanning across a set of two blades – ILM 0 across blade 0 and blade 1, ILM 1 across blade 2 and blade 3.

Figure 3 illustrates an example for configuring BL890c i2 system with SLM and ILM.

Figure 3: BL890c i2 system with SLM and ILM



How to configure memory for NUMA?

Memory configuration with only SLM/CLM or a combination of SLM/CLM and ILM is possible on Integrity NUMA servers. However on a cell-based system, OpenVMS requires a minimum of 2 GB of ILM.

A change in the memory interleaving configuration on i2 servers is accomplished using the `memconfig` command. The system requires a reset for the change to take effect.

```
Shell>memconfig –mi
```

OpenVMS supports the following memory interleaving configurations:

- MaxUMA (0% SLM:100% ILM)
- MostlyUMA (12.5% SLM:87.5% ILM)
- Balanced (50% SLM: 50% ILM)
- MostlyNUMA (87.5% SLM:12.5% ILM)
- MaxNUMA (100% SLM: 0% ILM)

Note:
MostlyNUMA is the default memory configuration.

Integrity cell-based systems need to be configured to have CLM; otherwise the entire memory is treated as interleaved (ILM).

HP recommends using Partition Manager (`parmgr`) for configuring memory on the Integrity cell-based systems. The `parmgr` command provides a graphical user interface and is available for free from HP. For more information on the `parmgr` command, see the Technical documentation website at:
http://bizsupport2.austin.hp.com/bc/docs/support/SupportManual/c02058520/c02058520.pdf

Resource Affinity Domains

In OpenVMS, a software grouping of hardware processing resources and a memory with similar access characteristics is called RAD. Each RAD is a logical grouping of processors and memory local to a socket or a cell. OpenVMS treats the ILM as a separate RAD; also all the processors in the system are considered as members of this ILM RAD.

Base RAD

The ILM is considered as the BASE RAD. The BASE RAD contains all the system data that is not specific to a process or to a CPU, including read-only code and read/write data. If the ILM is not configured in the system, RAD 0 is treated as the BASE RAD.

A BL890c i2 server blade can have two ILMs, each of which corresponds to a RAD. The first ILM RAD is designated as the Base RAD.

Home RAD

Each process, when created, is assigned to a particular RAD and this RAD is considered as the HOME RAD for that process. OpenVMS automatically assigns a Home RAD for every process if a

specific RAD is not explicitly mentioned for that process. By default, OpenVMS assigns one of the RADs other than the Base RAD as a process Home RAD.

OpenVMS Memory Manager and RAD

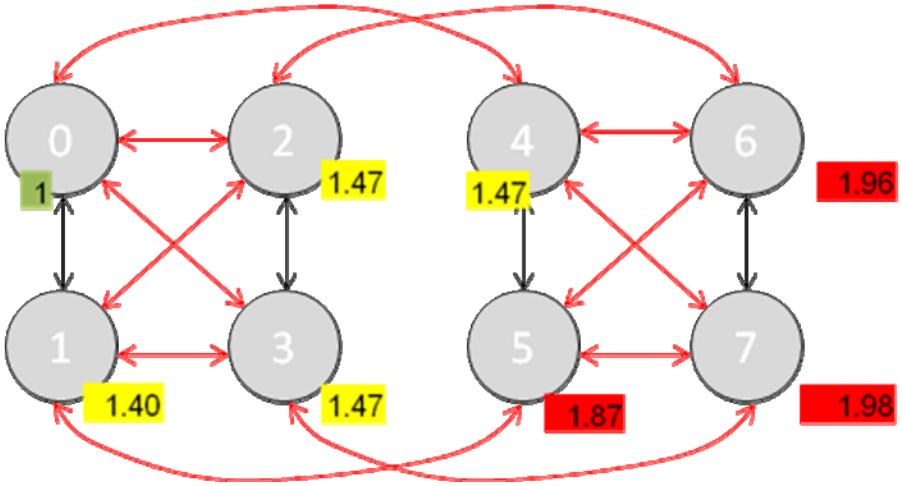
The OpenVMS Memory Manager allocates the required memory for a process from the Home RAD of that process. If this process runs on a CPU from its Home RAD, the number of local memory references will far exceed the number of remote memory references. This helps to reduce the CPU wait cycles for memory references, thereby increasing the system performance.

RAD Preference Array

On Integrity servers, OpenVMS maintains a RAD Preference Array for each RAD. Entries in this array are ordered in terms of increasing access cost. During memory allocation, this array is used to choose the optimum RAD(s) to satisfy the memory requirement. If the memory is not available in a particular RAD, the RAD Preference Array is used to determine the next best RAD with minimum access cost from where memory can be allocated.

Figure 4 illustrates the relative memory latencies for a CPU in the Socket 0 across various SLMs in a BL890c i2 server blade – access to SLM 0 being the fastest and access to SLM 7 being the slowest.

Figure 4: Relative memory latencies across Sockets on BL890c i2



- Slowest latencies are marked in red
- Second slowest latencies are marked in yellow
- Fastest latencies are marked in green

The following table lists the sample RAD Preference Array for each RAD on a BL890c i2 system with SLMs and ILMs.

RAD Preference Array for a BL890c i2 system with SLMs and ILMs

RAD #	RAD Preference Array
0 (SLM 0)	0; 8(1st ILM);1;2;3;4;5;9(2nd ILM);6;7
1 (SLM 1)	1; 8(1st ILM);0;2;3;5;4;9(2nd ILM);6;7
2 (SLM 2)	2; 8(1st ILM);3;0;1;6;7;9(2nd ILM);4;5
3 (SLM 3)	3; 8(1st ILM);2;0;1;7;6;9(2nd ILM);4;5
4 (SLM 4)	4; 9(2nd ILM);5;0;6;7;1;8(1st ILM);2;3
5 (SLM 5)	5; 9(2nd ILM);4;1;6;7;0;8(1st ILM);2;3
6 (SLM 6)	6; 9(2nd ILM);7;2;4;5;3;8(1st ILM);0;1
7 (SLM 7)	7; 9(2nd ILM);6;3;4;5;2;8(1st ILM);0;1
8 (ILM 0)	8(1st ILM);0;1;2;3;9(2nd ILM);4;5;6;7
9 (ILM 1)	9(2ndILM);4;5;6;7; 8(1st ILM);0;1;2;3

Page Zeroing

In a RAD-enabled system, page lists corresponding to free pages and demand-zero pages are maintained individually for each RAD. The physical pages in these lists correspond to the SLM/CLM and ILM of the associated RADs.

CPU, when in its idle loop, zeroes free pages of physical memory. This meets the need for future demand-zero page allocations. For Page Zeroing, the OpenVMS Memory Manager chooses pages from the Home RAD of the idle CPU till sufficient number of pages are zeroed. This ensures local memory references, thereby keeping the latency to a minimum.

Additionally, once a sufficient number of demand-zero pages are available in a RAD, a CPU executing in the idle loop in that RAD assists in zeroing the pages for the ILM RAD. In BL890c i2 server blade that has two ILM RADs, the CPU does the Page Zeroing for the ILM RAD closer to it.

Per-RAD Non-Paged Pool

On a RAD-enabled system, the Non-Paged Pool can be configured for each RAD. By default, this feature is turned off in the Integrity NUMA servers and the Non-Paged Pool comes from the Base RAD. In the Alpha servers with RAD configured, this feature is turned on by default.

If the sixth bit in the RAD_SUPPORT system parameter is set, the Per-RAD Non-Paged Pool feature is enabled and the NPAGERAD system parameter value is used to determine the total memory for the Non-Paged Pool among all the RADs other than the Base RAD. If the value of the NPAGERAD is 0 (default), OpenVMS computes a suitable value based on the distribution of memory between the SLMs/CLMs and the ILM.

For more information on NPAGERAD system parameter, see References.

The NPAGEDYN system parameter defines the total amount of Non-Paged Pool memory on the system. The Non-Paged Pool memory size from the Base RAD is (NPAGEDYN – NPAGERAD).

OpenVMS Scheduler and RAD

With RAD soft-affinity, the OpenVMS Scheduler attempts to schedule a process on a CPU from the Home RAD of that process. This is a complementary mechanism to ensure that when a process is scheduled on a CPU, the memory references for that process become more local. Any CPU hard-affinity set for a particular process to particular CPU(s) takes precedence over the RAD soft-affinity.

Optimized Scheduling Algorithm

When a process is ready to be scheduled, if the scheduler does not find an idle CPU from the Home RAD of that process, it skips the scheduling of that process for 'skip count' iterations; beyond that the Scheduler chooses a CPU from a remote RAD.

For more information on RAD_SUPPORT system parameter, see References.

RAD—DCL commands

OpenVMS provides various DCL commands and Lexicals to view and modify the Home RAD of a particular process in the system. The following examples illustrate the same:

Command to display the Home RAD of the current process

```
$ show proc/rad

18-APR-2010 02:17:31.49   User: SYSTEM      Process ID:   0000042F
Node: SKD12                                 Process name: "SYSTEM"

Home RAD: 1
```

Command to display the Home RAD of a particular process

```
$ show proc/rad/id=00000421

18-APR-2010 02:18:45.52   User: INTERnet    Process ID:   00000421
Node: SKD12                                 Process name: "TCPIP$INETACP"

Home RAD: 1
```

Command to change a process' Home RAD

```
$SET PROCESS/RAD=HOME=n
```

Note:
A process' Home-RAD is changed implicitly when its affinity is set.

For example:

```
$ show process/rad

18-APR-2010 19:48:30.75   User: SYSTEM     Process ID:   0000042F
Node: SKD12                                Process name: "SYSTEM"

Home RAD: 1
```

If this process' affinity is set to any processor belonging to RAD 0, its Home RAD is changed to RAD 0.

CPU 1 is from RAD 0.

```
$ set process/affinity/set=1

$ show process/rad

18-APR-2010 19:54:43.21   User: SYSTEM    Process ID:   0000042F
Node: SKD12                               Process name: "SYSTEM"

Home RAD: 0
```

Here the process Home RAD is changed from RAD 1 to RAD 0.

Note:
HP does not recommend frequent changes of RAD and CPU affinity as it can degrade the performance.

Utility to understand the distribution of memory and CPUs across RADs

```
$ @SYS$COMMON:[SYSHLP.EXAMPLES]RAD.COM;

Node: OVMS Version: V8.4      System: HP rx7640 (1.60GHz/12.0MB)

RAD   Memory (GB)   CPUs
===   ===========   ===============
  0       27.49     0-7, 16-23
  1       35.49     8-15, 24-31
  2        8.99     0-31
```

RAD—SDA commands

The SDA commands related to RAD are as follows:

- Command to get the details of a system RAD configuration.
  `SDA> show rad [number|/all]`
- Command to get the page distribution across RADs in terms of free pages and zero pages.
  `SDA>show pfn/rad`

Importance of RAD

The NUMA architecture provides mechanisms to reduce the average memory latency. The magnitude of the reduction of this memory latency for a process depends upon the memory access pattern of the process and the memory configuration in each RAD. Local memory access is much faster when compared to a remote access or access to the ILM. The best possible RAD configuration – proper percentage of SLM/CLM and ILM distribution – is one of the key factors for performance improvement. Accessing the local memory reduces the processor cycles spent on memory references, thereby reducing the processor utilization and increasing the throughput.

The impact of RAD becomes more when we deal with a large system, for instance BL890c i2.

Figure 4 illustrates relative latencies on BL890c i2. From the Socket 0, accessing memory in the Socket 7 is much more expensive when compared to accessing memory in the Socket 1 (in the same blade).

When to use RAD?

Most of the enterprise-application software exhibits high amount of local memory reference. Few applications are designed to access more global data and have scattered access to memory. RAD is best suited for the first category of applications. For the latter, usage of Interleaved Memory is a better choice than Local Memory.

In some cases, the number of applications deployed on the system and the layout of the process distribution has a higher bearing on the RAD. For instance, if a system is hosting multiple applications with small working sets compared to the available physical memory, RAD is the obvious choice. With a small working set, application processes will have a better locality of memory reference and performance gain. On the other hand, if a single application with a very large working set is deployed on a system, impact of RAD will be less.

RAD configuration guidelines

Applications with more local memory reference pattern and small working set will gain good amount of performance, if most of the memory is configured as Socket Local Memory or Cell Local Memory.

On the BL8x0c i2 server, in our in-house tests with Oracle, the MostlyNUMA memory configuration exhibits the best performance.

On a cell-based system, HP recommends a composition of 7/8 of total memory as CLM and 1/8 of total memory as ILM. However, if the application exhibits a performance degradation, either the application is suitable for fully Interleaved Memory or the system requires further tuning in the RAD configuration.

RAD performance monitoring

On RAD configured systems, the `RADCHECK` utility of OpenVMS provides information on the locality of memory in terms of page distribution across RAD(s) for a process.

This utility can control the RAD scheduling algorithm and provide the scheduling statistics.

```
$ radcheck :== $sys$test:radcheck
```

To get more information on the usage of `RADCHECK`, use the help option.

```
$ radcheck  -help
```

References

RAD Support in OpenVMS V8.4
http://h71000.www7.hp.com/doc/84final/6679/6679pro_002.html#index_x_35

HP OpenVMS System Management Utilities Reference Manual: (M–Z) –
http://h71000.www7.hp.com/doc/84final/6048/ba555_90009.pdf

Memory subsystem information for HP Integrity Server Blades (BL860c i2, BL870c i2, and BL890c i2)
http://h20195.www2.hp.com/V2/GetPDF.aspx/4AA1-1126ENW.pdf