Optimizing the Memory Settings for SAP Systems on 32-Bit Windows

© Copyright 2008 SAP AG. All rights reserved.

No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP AG. The information contained herein may be changed without prior notice.

Some software products marketed by SAP AG and its distributors contain proprietary software components of other software vendors.

Microsoft, Windows, Outlook, and PowerPoint are registered trademarks of Microsoft Corporation.

IBM, DB2, DB2 Universal Database, OS/2, Parallel Sysplex, MVS/ESA, AIX, S/390, AS/400, OS/390, OS/400, iSeries, pSeries, xSeries, zSeries, z/OS, AFP, Intelligent Miner, WebSphere, Netfinity, Tivoli, and Informix are trademarks or registered trademarks of IBM Corporation in the United States and/or other countries.

Oracle is a registered trademark of Oracle Corporation.

UNIX, X/Open, OSF/1, and Motif are registered trademarks of the Open Group.

Citrix, ICA, Program Neighborhood, MetaFrame, WinFrame, VideoFrame, and MultiWin are trademarks or registered trademarks of Citrix Systems, Inc.

HTML, XML, XHTML and W3C are trademarks or registered trademarks of W3C®, World Wide Web Consortium, Massachusetts Institute of Technology.

Java is a registered trademark of Sun Microsystems, Inc.

JavaScript is a registered trademark of Sun Microsystems, Inc., used under license for technology invented and implemented by Netscape.

MaxDB is a trademark of MySQL AB, Sweden.

SAP, R/3, mySAP, mySAP.com, xApps, xApp, SAP NetWeaver, and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP AG in Germany and in several other countries all over the world. All other product and service names mentioned are the trademarks of their respective companies. Data contained in this document serves informational purposes only. National product specifications may vary.

These materials are subject to change without notice. These materials are provided by SAP AG and its affiliated companies ("SAP Group") for informational purposes only, without representation or warranty of any kind, and SAP Group shall not be liable for errors or omissions with respect to the materials. The only warranties for SAP Group products and services are those that are set forth in the express warranty statements accompanying such products and services, if any. Nothing herein should be construed as constituting an additional warranty.

SAP Library document classification: PUBLIC

Disclaimer
Some components of this product are based on Java™. Any code change in these components may cause unpredictable and severe malfunctions and is therefore expressively prohibited, as is any decompilation of these components.

Any Java™ Source Code delivered with this product is only to be used by SAP’s Support Services and may not be modified or altered in any way.
## Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Optimizing the Memory Settings for SAP Systems on 32-Bit</td>
<td>4</td>
</tr>
<tr>
<td>1 How is the Memory Addressed in the Process?</td>
<td>4</td>
</tr>
<tr>
<td>2 When is the Memory to be Analyzed?</td>
<td>8</td>
</tr>
<tr>
<td>3 How is the Virtual Address Space Analyzed?</td>
<td>9</td>
</tr>
<tr>
<td>4 How are the Results of the Memory Analysis Interpreted?</td>
<td>10</td>
</tr>
<tr>
<td>5 Post-Analysis Steps</td>
<td>13</td>
</tr>
</tbody>
</table>
Optimizing the Memory Settings for SAP Systems on 32-Bit

This document provides information on how to optimize the memory management for your SAP ABAP systems that run on 32-bit platforms. It provides answers to the following questions:

- How is the memory addressed in the process?
- How is the memory analyzed?
- When do you have to analyze the memory?
- How are the results of the memory analysis interpreted?

1 How is the Memory Addressed in the Process?

Every Windows process can address up to 4 GB ($2^{32}-1$) of virtual memory in its own address space. This memory consists of 2 parts: the user part (where the user program can allocate the memory) and the kernel part. Under normal circumstances, both parts are 2 GB.

The user part of the virtual address space can be increased by activating the /3GB option in boot.ini. This is called “4GT”. 4GT tuning is possible as of Windows 2000 Advanced Server and Windows Server 2003 (Standard, Enterprise and Datacenter Editions). To use 4GT, the option large address aware bit (LAB) must be set by the software vendor in the 32bit EXE file. For example, disp+work.exe has LAB activated by default, java.exe does not.

As of Windows Server 2003 it is also possible to differentiate between the user part and the kernel part of the virtual address space with the /USERVA option.

Figure 1: Virtual memory layout on 32-bit Windows (Source: Microsoft).

Figure 2: Virtual memory layout with /USERVA option
Example of boot.ini:

```plaintext
multi(0)disk(0)rdisk(0)partition(1)\WINDOWS="Windows Server 2003, Enterprise" /fastdetect /NoExecute=OptOut /3GB /USERVA=2800
```

**Caution**

The `/USERVA` option has the opposite effect of the `/3GB` option, that is it decreases the user mode part of the virtual address space. As the kernel part of the virtual address space is shared between all processes, it might be the case that 1 GB is not enough to accommodate all internal system structures. In this case `/USERVA` is very helpful as it differentiates between operating system needs and application needs. To find the optimal value for `/USERVA` start iterative process with the highest value for `/USERVA` and decrease the value until the desired result is reached. This allows the SAP process to address the maximum virtual memory and provides optimal size for the Windows kernel structures.

If the 32-bit process runs on 64-bit Windows, the user part of the virtual address space is 4 GB.

Figure 3: Virtual memory layout of a 32-bit process on a 32-bit or 64-bit operating system

The following figure shows the virtual address space of the SAP work process (WP) with a typical distribution of memory segments:

- Committed shared memory blocks (blue area)
- Loaded DLL or executable images (gray area)
- Reserved or committed memory for heap (light blue area)
- Extended memory (red area)
- Free space (green area).
The ABAP application can allocate free space in the virtual address space of the process (VAS) if it is not limited by the SAP kernel parameters (abap/heap_area_dia, abap/heap_area_nondia). In general, the amount of memory for the ABAP application is calculated as follows:

Extended Memory + Free Space in VAS in the SAP work process

Note that committed shared memory blocks and the extended memory block (EM) are shared between all virtual address spaces of all SAP work processes.

Increasing SAP shared memory blocks by modifying the SAP memory parameters, or reducing the user mode part of the virtual address space (by setting /USERVA) automatically limits the memory for the ABAP application on a 32-bit platform (see Figure 5).
Figure 5: “Low free space” problem

General considerations:

- Changing the SHM blocks on a 32-bit platform may have significant consequences, that is increasing the SHM block could lead to an error in the ABAP application if the ABAP application needs memory and not enough free space is available.
- A 64-bit platform can handle the growing SHM more efficiently. However if configuring too large an SHM block, the performance could become worse due to high paging rates. A 64-bit application server on a 64-bit platform requires less effort in optimizing the system.
- When analyzing and tuning the memory you have to decide if you want to achieve better performance (large SHM buffers lead to better performance) or get sufficient memory for the application (maximum heap, SHM buffers are set to a minimum).
- “Free space” means enough free address space in VAS to meet the ABAP program memory requirements. Another question is: How much memory does my application need? If there is not enough free space, the transaction crashes, and in ST22, for example, you get the error TSV_TNEW_PAGE_ALLOC_FAILED. On the other hand, too much free space has an impact on the SAP memory buffers (if they are too small) and the performance. Transaction ST02 (see the “History” section) helps you to identify the maximum memory usage (heap) for your transactions and the efficiency of allocated SAP memory buffers.
Figure 6 shows the gain of free space in the virtual address space of 32-bit process under various circumstances.

**Figure 6: Virtual address space limitations for 32-bit process**

## 2 When is the Memory to be Analyzed?

In general, we recommend you to analyze the memory in the following cases:

- Before changing any memory parameters check if there is enough free space to apply the changes and for the ABAP programs to run.
- If the application gets memory errors `TSV_TNEW_PAGE_ALLOC_FAILED`, `PXA_NO_SHARED_MEMORY`, and so on.
- To detect fragmenting DLLs.
- If the SAP system does not start after applying the operating system security patches or changes in memory parameters, you can see, for example, the following error in the dew_wX trace:

```c
A PXA_INITIALIZATION
I *** ERROR => [MapOsShm] Can't find free space for Shared Memory
(Size=1023801648)
[shmnt.c 1780]
I *** ERROR => [CreateOsShm] MapViewOfFile(6,00000398) failed with
Err=0
[shmnt.c 1780]
I *** ERROR => ShmCreate: Create (6,1023801520,1) failed [shmnt.c 460]
```
Note that there are cases, where the maximum of free virtual address space is allocated for a transaction but where the application still does not run and crashes with TSV_TNEW_PAGE_ALLOC_FAILED. If this is the case, you must check the memory setting. If this does not help, you must revise the transaction code or install the 64-bit application server.

### 3 How is the Virtual Address Space Analyzed?

To analyze the virtual address space as described in this section, the SAP instance must be running. The analysis can take place any time without any impacts on running productive systems.

You can use the following tools to check the virtual address space of the SAP work process:

- **SAP Address Space Viewer:**
  - Download and install the SAP Address Space Viewer (see SAP Note 129813).
  - Run the SAP Address Space Viewer
  - In the “Process List”, select any SAP dialog work process and choose **Analyze**.
  
  ![Figure 4](image1)

- **\$(DIR_EXECUTABLE)\sapntkill.exe:**
  - Run sapntkill:

    ```
    sapntkill.exe -MEM <process ID of any dialog process>
    ```

  - The result is written to the following text file:

    `<drive>:\usr\sap\<SID>\D<...>\work\adrspc_<process nr.>`

  - To view the results, open the file in any text editor or run the **SAP Address Space Viewer**, choose **Import File** and specify the file name.
4 How are the Results of the Memory Analysis Interpreted?

This section explains how to interpret the results of the memory analysis. For this purpose, there are some sample screenshots of the virtual address space that may be useful for further analysis.

Example 1, represents a real case example. Some space is occupied by EM (red), SHM (dark blue) and some free space (green) for later heap allocation. No fragmenting DLLs (gray) in the middle. The virtual address space size is 2 GB. Sum of free space in VAS including fragmentation (green segments) is ~600MB.

Figure 7: Example 1

Example 2, represents a real case example. Almost all space is occupied by EM* (red) and SHM* (dark blue). Almost no free space (green) is available for later heap allocation. The virtual address space size is 2 GB.
Figure 8: Example 2

Sum of free space in VAS including fragmentation (green segments) is ~240 MB.

Example 2, represents a system with the focus on optimizing performance; some SHM segments are set to too high values and there is almost no space for HEAP. This is not a problem until memory-intensive transactions start.

Possible solutions for example 2:

- Activate the /3GB option in boot.ini that allows you to increase the user part of the virtual address space. As a result, more free space in the virtual address space for heap or for increasing SAP SHM is available.

- Reduce some SAP SHM. To do this, revise at least the following segments:
  
  PXA: abap/buffersize
  
  ROLL: rdisp/ROLL_SHM
  
  PAGE: rdisp/PG_SHM

Example 3. Despite the /3GB option, either one or several large contiguous SHM segments cannot be allocated, or the system does not start after applying the newest Microsoft security patches.

Pay special attention to fragmenting DLLs that are loaded within the virtual address space range 0x10000000 to 0x70000000. Such DLLs might prevent the SAP work process from allocating one contiguous large SHM segment. To investigate this case, the SAP system must be running. You have to reduce at least one or possibly both of the following SAP shared memory buffers:

- abap/buffersize = 250000
- em/address_space_MB = 256

In other words, the memory size must be significantly reduced so that the memory block can be allocated in the virtual address space. After that, see section 3 How is the Virtual Address Space Analyzed to get a snapshot of the virtual address space of the process.
To identify the fragmenting DLL, choose any SAP work process in the list, and choose the Clean Address Space button in the SAP Address Space Viewer:

When identifying the fragmenting DLLs, pay attention to the Loaded Base address and the DLL Base address. The DLL Base address is the address to which the DLL is to be loaded. However, the DLL loading mechanism loads the DLLs in order of their dependencies, which means that DLL Base address can be occupied by another DLL that was loaded earlier. In this situation the DLL is loaded to the next free address which normally does not fragment the VAS. Therefore, check the DLLs where the Loaded Base address is the same as the DLL Base address.
SAMLIB.DLL is loaded with base address 0x5CCF0000 (good examples for rebasing are DLLs in the range 0x10000000 to 0x70000000). Note: DLLs marked as (X) are system DLLs and cannot be rebased in the conventional way with the SAP Address Space Viewer.

⚠️ Caution

Never use the SAP Address Space Viewer to rebase the Windows system DLLs. SAP Note 664607 contains a list of the patches for fragmenting system DLLs.

Solutions for eliminating “fragmenting” DLLs:
- For DLLs under Windows File Protection (%windir%\system32) see SAP Note 664607 “Hot Fixes or patch sets with fragmenting DLLs”. Never use the SAP Address Space Viewer to rebase the system DLLs as this might damage your system.
- Do not use rebasing for other DLLs. Instead of rebasing, only use the “preloading” mechanisms (SAP Note 853696, available as of SAP startup framework 6.40).
- There is always a set of system DLLs loaded in the range 0x70000000 to 0x77000000 that are not meant for rebasing.

5 Post-Analysis Steps

After analyzing the free virtual address space and deciding whether you want to achieve good performance or increase the maximum size of free memory space, you can use transaction ST02, ST02 → History (search for max used), ST02 → Current Parameters to check your SAP SHM segments.

Segments to be revised are:
- Program buffer (abap/buffersize)
- Export/import area (rsdb/obj/buffersize)
- Roll area (rdisp/ROLL_SHM)
- Page area (rdisp/PG_SHM)

Check the history of maximum buffer usage in ST02. The hit ratio should be more than 99% for program buffer, and increase or reduce the corresponding parameters according to your needs.
### Configuration Information

#### Attachment: Table of main SAP SHM segments and their approximate sizes.

<table>
<thead>
<tr>
<th>SAP Address Space Viewer definition</th>
<th>SH M</th>
<th>Parameter</th>
<th>Formula (proportionally depends on)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>4.6x</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>6.40</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>&gt;=7.00</td>
</tr>
</tbody>
</table>

**Administration and special buffers**

<table>
<thead>
<tr>
<th>Segment Type</th>
<th>SHM</th>
<th>Parameters</th>
<th>Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dispatcher Administration Tables</td>
<td>2</td>
<td>WpNo</td>
<td>rdisp/wp_ca_bnk_no</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>rdisp/appc_ca_bnk_no</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>rdisp/max_comm_entries</td>
</tr>
<tr>
<td>Dispatcher Communication Areas</td>
<td>3</td>
<td></td>
<td>WpNo</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>rdisp/appc_ca_bnk_no</td>
</tr>
<tr>
<td>Statistic Area</td>
<td>4</td>
<td></td>
<td>WpNo</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>stat/bufsize</td>
</tr>
<tr>
<td>Alert Area</td>
<td>13</td>
<td></td>
<td>WpNo</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>alert/MONI_SEGM_SIZE</td>
</tr>
</tbody>
</table>

**Other Buffers**

<table>
<thead>
<tr>
<th>Segment Type</th>
<th>SHM</th>
<th>Parameters</th>
<th>Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>EM</td>
<td>-</td>
<td>em/address_space_MB</td>
<td>-</td>
</tr>
<tr>
<td>Abap Program Buffer</td>
<td>6</td>
<td>abap/buffersize</td>
<td>-</td>
</tr>
<tr>
<td>Shared Paging Buffer</td>
<td>8</td>
<td>rdisp/PG_SHM</td>
<td>-</td>
</tr>
<tr>
<td>Shared Roll Buffer</td>
<td>9</td>
<td>rdisp/ROLL_SHM</td>
<td>-</td>
</tr>
<tr>
<td>Calendar Buffer</td>
<td>11</td>
<td>zcsa/calendar_area</td>
<td>-</td>
</tr>
<tr>
<td>TemSe Char-Code Convert Buffer</td>
<td>12</td>
<td>rsts/ccc/cachesize</td>
<td>-</td>
</tr>
<tr>
<td>Presentation Buffer</td>
<td>14</td>
<td>zcsa/presentation_buffer_area</td>
<td>-</td>
</tr>
<tr>
<td>Shared Roll Administration</td>
<td>17</td>
<td>rdisp/ROLL_MAXFS</td>
<td>rdisp/ROLL_MAXFS</td>
</tr>
<tr>
<td>Shared Paging Administration</td>
<td>18</td>
<td>rdisp/PG_MAXFS</td>
<td>rdisp/PG_MAXFS</td>
</tr>
<tr>
<td>Tale Buffer</td>
<td>19</td>
<td>zcsa/table_buffer_area</td>
<td>-</td>
</tr>
<tr>
<td>Dispatcher Request Queue</td>
<td>31</td>
<td>rdisp/elemt_per_queue</td>
<td>rdisp/elemt_per_queue</td>
</tr>
<tr>
<td>Tale Buffer, Part Buffering</td>
<td>32</td>
<td>rtbb/buffer_length * 1024</td>
<td>rtbb/buffer_length * 1024</td>
</tr>
<tr>
<td>Enqueue Table</td>
<td>34</td>
<td>enque/table_size * 1024</td>
<td>enque/table_size * 1024</td>
</tr>
<tr>
<td>Database Statistics Buffer</td>
<td>41</td>
<td>rsdb/staton</td>
<td>-</td>
</tr>
<tr>
<td>Database TTAB Buffer</td>
<td>42</td>
<td>rsdb/ntab/entrycount</td>
<td>rsdb/ntab/entrycount</td>
</tr>
<tr>
<td>Database FTAB Buffer</td>
<td>43</td>
<td>rsdb/ntab/tabsize</td>
<td>rsdb/ntab/tabsize</td>
</tr>
<tr>
<td>Database IREC Buffer</td>
<td>44</td>
<td>rsdb/ntab/itbsize</td>
<td>rsdb/ntab/itbsize</td>
</tr>
<tr>
<td>Database Short Nametab Buffer</td>
<td>45</td>
<td>rsdb/ntab/ntabsize</td>
<td>rsdb/ntab/ntabsize</td>
</tr>
<tr>
<td>Database CUA Buffer</td>
<td>47</td>
<td>rsdb/cua/buffersize</td>
<td>-</td>
</tr>
<tr>
<td>Number Range Buffer</td>
<td>48</td>
<td>nobuf/max_no_buffer_entries</td>
<td>nobuf/max_no_buffer_entries</td>
</tr>
<tr>
<td>Configuration Information</td>
<td></td>
<td>rdisp/ROLL_SHM</td>
<td>rdisp/ROLL_SHM</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>-------</td>
<td>---------------------------------</td>
<td>---------------------------------</td>
</tr>
<tr>
<td>Spool Administration</td>
<td>49</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Extended Memory Administration</td>
<td>51</td>
<td>3.2 MB (hard-coded)</td>
<td>3.2 MB (hard-coded)</td>
</tr>
<tr>
<td>Export/Import Buffer</td>
<td>54</td>
<td>rsdb/obj/buffersize</td>
<td>rsdb/obj/buffersize</td>
</tr>
<tr>
<td>Memory Pipes</td>
<td>62</td>
<td>mpi/total_size_MB</td>
<td>mpi/total_size_MB</td>
</tr>
<tr>
<td>Online Text Repository Buffer</td>
<td>64</td>
<td>-</td>
<td>rsdb/otr/buffersize_kb</td>
</tr>
<tr>
<td>Export/Import Shared Memory</td>
<td>65</td>
<td>-</td>
<td>rsdb/esm/buffersize_kb</td>
</tr>
</tbody>
</table>