So you want AMODE 64?
But, are you sure you need it?

M. Carl Gehr, Jr. - Edge Information Group

SHARE: LANG Project
Boston, MA - 04-Aug-2010
Wednesday - 11:00AM

EDGE INFORMATION GROUP

Abstract

Are you experiencing storage constraints in your applications that you believe would be relieved by having AMODE 64 for your COBOL and PL/I applications? But, are you really using AMODE 31 effectively? Might you relieve some of the constraint by better tuning of your use of the LE storage runtime options to control HEAP and STACK usage? Are you sure your applications are all running in AMODE 31 with the ALL31(ON) run-time option?

This session will give you some tips and techniques for evaluating your storage usage to try to answer some of the questions above. And, after doing this analysis, you may also find that you can improve the performance of your applications by more effectively using the storage that you already have in AMODE 31.

Following this session, you should also attend the "Language Environment Futures Workshop" where IBM will discuss some of the possible ways 64-bit addressing might be exploited in COBOL and PL/I applications. This subject will be further discussed in a Birds of a Feather session, "Discussion: The Future of AMODE 64 for COBOL and PL/I," at 6:00PM.

EDGE INFORMATION GROUP

The presenter welcomes your comments and questions. Please feel free to contact us via E-Mail or phone:

Carl Gehr cgehr@edge-information.com
(513) 948-8906

Disclaimer: The information in this document represents information gathered by the presenter from various sources. We have done our best to provide accurate information. Any questions regarding the specific details on any product should be addressed to the vendor of that product. We have not independently verified each of the recommendations, and are not responsible for errors or omissions. Recommendations made are intended to be thought provoking and applicable to the general situation, and may not be appropriate for all users. It is your responsibility to evaluate the information and determine its applicability to your environment. If any errors are found, please let us know and we will make every effort to correct them in the future.

So you want AMODE 64?
But, are you sure you need it?

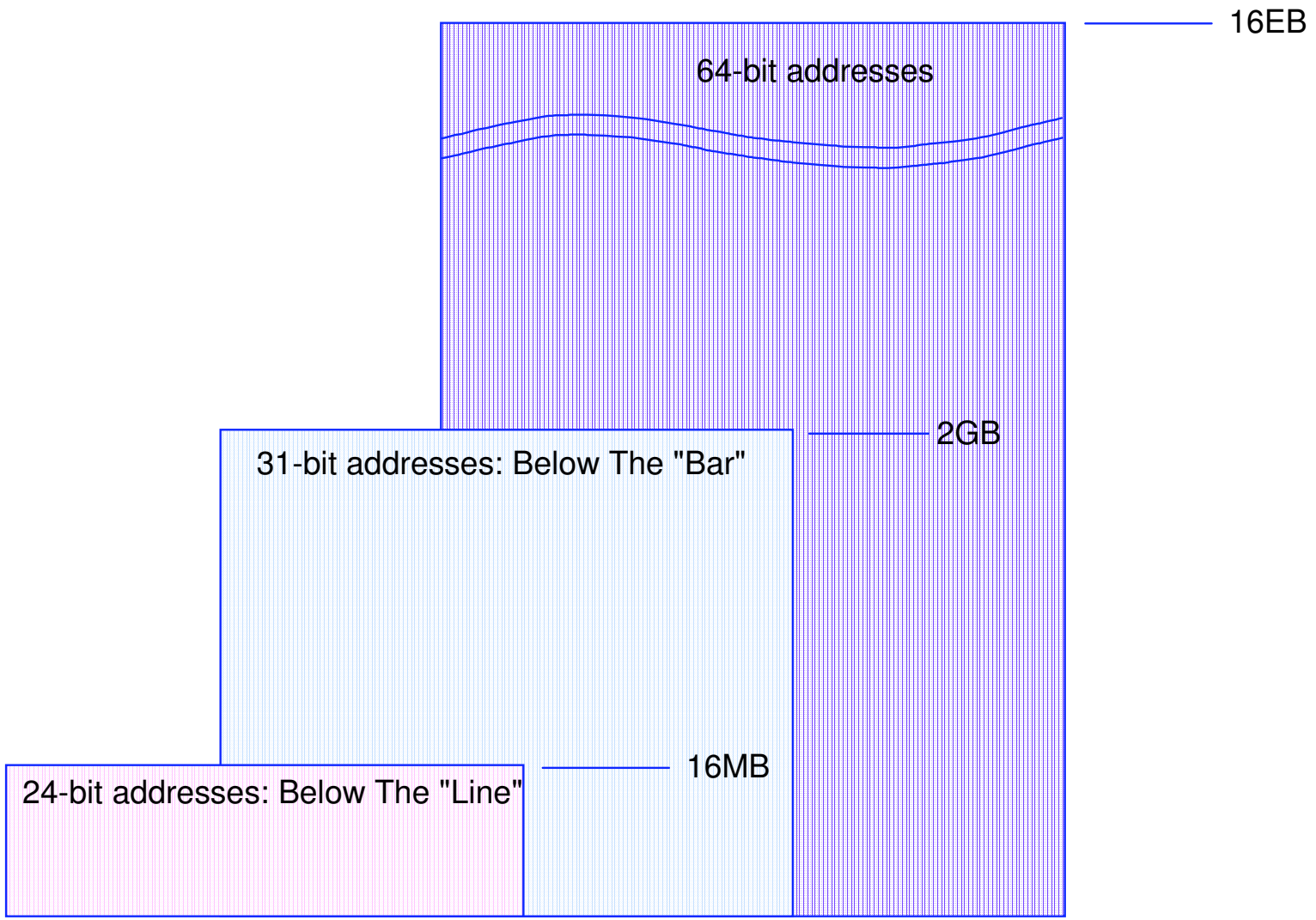
✓ Agenda

EDGE INFORMATION GROUP

What are we going to cover?

- What does 64-Bit really mean?
- Why do people often decide that they need 64-bit capability in their applications?
- What can you do to evaluate an application that appears to be exceeding the 2GB limit of 31-bit?
- What are some possible actions you can take while you are waiting for full 64-bit application implementation?
- What other SHARE sessions should you attend to get more information on 64-bit and also let IBM know your requirements?

Basics: Virtual Storage Layout



Addressing Ranges

- For 24-bit addresses:
  - Access locations up to 16MB
  - Below "The Line" addresses
- For 31-bit addresses:
  - Access locations up to 2GB
  - Below "The Bar" addresses
- For 64-bit addresses:
  - Access locations up to 16 Exabytes
  - "The Bar" - Addresses between 2GB and 4GB; effectively not an addressable area.

EDGE INFORMATION GROUP

What is where?

- AMODE({24 | 31 |64 | ANY | MIN})
  - Determines the addressing capability assigned to the entry point of the module.
  - Instructions can address:
    - Instructions
    - Data
  - AMODE(ANY) means either 24 or 31 is OK.
  - AMODE(MIN) means select most restrictive.
- RMODE({24 | 31})
  - Determines where the program is loaded.
  - Important: No RMODE(64)
  - Thus only data resides in storage above the Bar.

Assumptions for this discussion

- You are using a z/OS that supports AMODE(64)
- Your applications are not written in C/C++ that already has 64-bit support.
- You are using current levels of Enterprise COBOL and/or Enterprise PL/I.
  - If 64-bit support is provided, it will be for these compilers or their successors.
  - If you are using earlier compilers, you should consider migrating to the new levels.

What is your storage problem?

- Key questions:
  - Is your constraint due to a large amount of application code?
  - Is your constraint due to large data requirements?
- If the problem is large code requirements, 64-bit support is not going to help you.
- If the problem is large data requirements:
  - Do you know where the data usage is?
    - STACK?
    - HEAP?
  - Have you tried to tune your data usage?

Storage Usage by COBOL

- Code: In load module
- HEAP Storage:
  - WORKING-STORAGE variables
  - NORENT programs have Working-Storage in the module.
- STACK Storage:
  - Intrinsic functions
  - Library routines
  - LOCAL-STORAGE variables

Storage Usage By PL/I

- Code: In load module
- STACK Storage
  - Automatic variables
  - Library routines
- HEAP Storage
  - BASED variables
  - CONTROLLED variables
  - AREA variables

EDGE INFORMATION GROUP

Constrained by Code?

- Even if you are constrained by large code requirements, you may have options without 64-bit .
- Is your code in large monolithic application module(s)?
  - Fewer options for easy tuning.
  - Most likely will require re-engineering or redesign.
  - May contribute to large data use?
- Is your code modular?
  - Static or dynamic linkage?
  - Do you understand logic flow?

EDGE INFORMATION GROUP

Code: Large Monolithic

- Redesigning to modular with dynamic linkages will allow:
  - Sharing of storage for nonconcurrent routines.
  - Reduced STACK requirements.
    - Minimal help for COBOL
    - Possibly major help for large, single PL/I
  - Reduced HEAP requirement.
    - Potentially significant for COBOL
    - Likely less beneficial for PL/I, but depends on design and storage usage
- May benefit from dynamic storage allocation.

EDGE INFORMATION GROUP

Code: Modular

- Static linkage: Not much can be done.
  - Change static to dynamic CALLs.
    - Share nonconcurrent code storage.
    - Exception routines may never use storage.
    - Unlikely to change STACK or HEAP use.
  - COBOL LOCAL storage may reduce HEAP.
- Dynamic Linkage:
  - May already have most of the above.
  - Tuning is still possible.
  - Use COBOL *CANCEL* statement to free W-S and space used by code.

EDGE INFORMATION GROUP

Data Constraints

- Have you done data storage tuning?
  - Most PL/I shops have been doing this.
  - Few COBOL shops have bothered to do tuning unless they have had a specific problem.
- PL/I Tuning
  - Generally concentrate on STACK
  - HEAP may be an issue if ALLOCATE used
- COBOL Tuning
  - Generally concentrate on HEAP
  - STACK is usually of minor concern
  - Use of COBOL LOCAL-STORAGE will reduce HEAP, but increase STACK. But STACK is LIFO.

EDGE INFORMATION GROUP

Note: For this presentation...

No intent to teach how to do storage tuning.

Rather, just to show the kinds of information that are available for you to do tuning.

Storage Tuning at a Glance

- Primary tool is LE Run-Time option:
  RPTSTG(ON | OFF) or RePorT SToraGe
  - Reports the storage in various categories. In particular:
    - STACK
    - HEAP
  - Reports allocated versus used storage.
  - Provides a global view of storage usage
- More granular view and tuning requires an understanding of individual program use of STACK and HEAP.

EDGE INFORMATION GROUP

RPTSTG for Storage Tuning

- General tuning strategy:
  - Is initial allocation of STACK and HEAP is large enough to satisfy all storage requests.
  - Is initial allocation of STACK and HEAP more than necessary?
  - Objective: Allocate no more than needed.
  - Side benefit may minimize number of GETMAIN/FREEMAIN requests.
- For our purpose, emphasis is on storage utilization rather than performance
  - Performance may improve.
  - Should not get worse.

What does RPTSTG show for STACK?

EDGE INFORMATION GROUP

Storage Report for Enclave COBNLEP 08/02/10 2:30:00 PM
Language Environment V01 R11.00

STACK statistics:

Initial size:	131072
Increment size:	131072
Maximum used by all concurrent threads:	22976
Largest used by any thread:	22976
Number of segments allocated:	1
Number of segments freed:	0

What does RPTSTG show for HEAP?

EDGE INFORMATION GROUP

Storage Report for Enclave COBNLEP 08/02/10 2:30:00 PM
Language Environment V01 R11.00

HEAP statistics:

Initial size:	32768
Increment size:	32768
Total heap storage used (sugg. initial size):	32
Successful Get Heap requests:	0
Successful Free Heap requests:	0
Number of segments allocated:	1
Number of segments freed:	0

What does RPTSTG show for HEAP24?

EDGE INFORMATION GROUP

Storage Report for Enclave COBNLEP 08/02/10 2:30:00 PM
Language Environment V01 R11.00

HEAP24 statistics:

Initial size:	8192
Increment size:	4096
Total heap storage used (sugg. initial size):	0
Successful Get Heap requests:	0
Successful Free Heap requests:	0
Number of segments allocated:	0
Number of segments freed:	0

What does RPTSTG show for ANYHEAP?

EDGE INFORMATION GROUP

Storage Report for Enclave COBNLEP 08/02/10 2:30:00 PM
Language Environment V01 R11.00

ANYHEAP statistics:

Initial size:	16384
Increment size:	8192
Total heap storage used (sugg. initial size):	9256
Successful Get Heap requests:	14
Successful Free Heap requests:	1
Number of segments allocated:	1
Number of segments freed:	0

What does RPTSTG show for BELOWHEAP?

EDGE INFORMATION GROUP

Storage Report for Enclave COBNLEP 08/02/10 2:30:00 PM
Language Environment V01 R11.00

BELOWHEAP statistics:

Initial size:	8192
Increment size:	4096
Total heap storage used (sugg. initial size):	2728
Successful Get Heap requests:	7
Successful Free Heap requests:	4
Number of segments allocated:	1
Number of segments freed:	0

EDGE INFORMATION GROUP

Detailed Look at Components

- Evaluate each program component:
  - What is the STACK request at initialization?
    - COBOL: Primarily for Register Save Areas; and LOCAL-Storage, if used
    - PL/I: RSAs, plus all AUTOMATIC variables
  - What is the HEAP usage?
    - COBOL: Working-Storage
    - PL/I: Dynamic allocations
- To really understand the above, you need to understand flow through module components.
  - STACK is LIFO
  - HEAP is language, logic dependent

So you want AMODE 64?
But, are you sure you need it?

✓ **Summary**

EDGE INFORMATION GROUP

Do you need 64-Bit?

- Where are you address space constrained?
  - If yes, what can you do until you get AMODE(64)?
- Code constrained? No help with 64-bit.
- Data constrained?
  - Have you tuned current storage usage?
  - Evaluate your applications to determine if you can improve the way is storage used.
- Let IBM know, with details, why you need AMODE(64).

EDGE INFORMATION GROUP

Your next steps . . .

- You are urged to attend two follow up sessions later today.

- Language Environment Futures Workshop
  - Wed - 1:30PM - Room 203
  - Discussion: The Future of AMODE(64) for COBOL and PL/I
  - Abstract: This regular offering of the LANG Project provides IBM the opportunity to discuss items they are considering for LE and language support. In Boston, IBM will discuss issues related to AMODE(64) in COBOL and PL/I applications.

- BOF: Discussion of Requirements for COBOL and PL/I Support of AMODE(64) [NOT in the Agenda book]
  - Wed - 6:00PM - Room 104
  - Abstract: Users will be able to indicate how they see the need for AMODE(64) support in COBOL and PL/I. C/C++ already has this support, but COBOL and PL/I applications may have additional requirements. This is your opportunity to tell IBM what you need and react to what you hear in the Futures session.

So you want AMODE 64? But, are you sure you need it?

Thank you for attending. . .
Any Questions?

SHARE: LANG Project
Boston, MA - 04-Aug-2010
Wednesday - 11:00AM

LE, COBOL, PL/I, C/C++ Project