SPERRY UNIVAC

basic systems
MARKETING SUPPORT SERVICES

Bulletin

NUMBER 56

November, 1982

TUNING TECHNIQUES FOR ASSEMBLIES AND COMPILES ON OS/3 - SYSTEM 80 AND SERIES 90.

ABSTRACT

This Bulletin provides guidelines for maximizing the performance on OS/3, System 80 and Series 90, when a large number of compiles and assemblies are required to be run concurrently

COMPANY CONFIDENTIAL "C"

FOR FURTHER INFORMATION CONTACT:
BASIC SYSTEMS PERFORMANCE MEASUREMENT
P.O. BOX 500
BLUE BELL, PA 19424
ATTN: PAT PINELLI
M.S. B103M
(215) 542-4890 SPERRYNET: 423-4890

ID CONTACT:

Ralf Thimell, London, 258-3738

TABLE OF CONTENTS

1. INTRODUCTION

The purpose of this bulletin is to show how to prepare disk packs, where to place files, and how to write Job Control for maximum performance when a large number of assemblies and compiles are required to run concurrently. The techniques discussed in this bulletin will increase overall system throughput not only for assemblies/compiles but for other applications as well.

2. DISK PREPARATION

Preparation of a disk prior to assigning space for files is extremely important. Disk preparation can be broken down into two specific steps:

A. DISK PREP

1) Prep the disk using the following options:

```
PREPT = C
RETRY = 01
```

These two options give the most accurate prep.

2) Place the VTOC in the center of the pack when possible. This is extremely important in reducing head movement when files are being opened and closed frequently.

You can indicate where you want the VTOC to reside by specifying six hexadecimal numbers representing the starting address in cylinder/head format (CCCCHH) on the VTOCB keyword. You can specify the hexadecimal numbers to indicate the cylinder and head address of the ending track of the VTOC with the VTOCE keyword.

VTOCB = CCCCHH, VTOCE = CCCCHH

To place VTOC's in the center of a disk use the hexadecimal numbers representing VTOC beginning and ending addresses for various disks as follows:

DISK TYPE	VTOCB	VTOCE
8417	011300	01130D
8419	019400	019406
8416	00CA00	00CA06
8418	00CA00	00CA06
8418 II	019400	019406
8430	00CA00	00CA12
8433	019400	019412

B. ELIMINATE ALTERNATE TRACKS

After the Disk Prep terminates normally, review the output listing, especially the alternate track assignments. Eliminate all alternates by assigning dummy files to those cylinders containing alternates. With the "ADDR" parameter of the // EXT Job Control statement, you can specify the absolute cylinder address in hexadecimal at which the file is to begin. For more information see the OS/3 Job Control User Guide, UP8065, Section 4.

Eliminating alternates, by assigning dummy files, may seem like a waste of space; however, it is worth it from a performance standpoint. By eliminating alternate tracks within files, unnecessary head movement is eliminated during processing. If an alternate is encountered in a highly used area (directory, index), performance degradation will occur.

3. FILE PLACEMENT

The placement of the compiler and assembler related libraries along with work files, is the single most important thing that you can do to improve the performance of your system. It will be necessary to implement at least some of the procedures described in this section to achieve compiler and assembler performance increases.

1. Copy system files YMAC and YOBJ to all available disk packs being utilized for compilations and assemblies.

2. Copy all compilers and assemblers into an alternate load library and execute them from these alternate load libraries.

3. Use SYSRES as little as possible and spread files used by compilers/assemblers across many disk packs. This approach is to eliminate head movement and contention as much as possible.

4. Work files should be hard assigned to specific devices.

   The work files (WORK1, WORK2, WORK3) should be placed on separate packs.

Example of file layout and disk usage for 1 assembly and 1 compile in a multiprogramming environment:

	100	101	102	103
VOL =	REL080	PACK01	PACK02	PACK03
		RUN	SPOOL	
	USE FOR	YOBJ	YOBJ	YOBJ
	SYSTEM	YMAC	YMAC	YMAC
	FUNCTIONS	ALTLOD	ALTLOD	ALTLOD
	ONLY	EXCLOD	EXCLOD	EXCLOD
		ALTSRC	ALTSRC	ALTSRC
JOB NAME				
	ASM1	YMAC	WORK1	WORK2
		ALTSRC		ALTLOD
	COBOL1	WORK2	YOBJ	WORK1
		ALTLOD	ALTSRC	
			WORK3	

5. File Considerations:

   A. If a file is opened and closed often, place it near the VTOC.

   B. Place heavily used files next to each other to eliminate as much head movement as possible. Heavily used files should be placed on separate packs when possible.

   C. Allocate enough file space initially to eliminate multiple extents when space is exhausted.

   D. A good scheme to use in file placement is as follows:

      Place VTOC in the center of the pack; alternate on either side of the VTOC starting with the most heavily used files and trailing off with the least used files.

      Example:

DISK
L
LH
H
VTOC
H
LH
L

L = Least Used Files
LH = Less Heavily Used Files
H = Most Used Files

6. This approach, as well as any other, requires regular maintenance. File fragmentation and multiple extents will cause performance degradation.

4. JOB CONTROL

Job Control required to enhance performance of assemblies and compiles is minimal. The main technique is to create several job streams for each assembly or compile. Each Job Stream will be the same except for disk assignments (see file placement). Assign a specific job stream to each user/group of users. Some analysis will be required to get the proper mix for daily operations; however, the results obtained from this procedure are well worth the effort.

Some rules and examples to follow in creating job streams for assemblies and compiles are:

1. Assign all work Procs (WORK1, WORK2, WORK3) to specific disk drives. If possible replace work Procs with actual job control statements. This will reduce Proc expansion time at job initiation.

2. Eliminate all compile-and-go job control streams and replace them with the specific statements. Only link-and-execute when compiles or assemblies have been thoroughly "desk checked". Eliminating linking on every assembly and compile will increase overall throughput.

3. Test the UPSI byte for clean compiles/assemblies before linking. Force the user through job control variables to specify when a link is necessary. Depending on the size of an assembler program, the link could take longer than the assembly.

4. Job streams for 1 assembly and 1 compile running concurrently:

JOB CONTROL

```
    // JOB ASM1,,(SIZE-SEE SECTION 5)
*1  // GBL L=N, N=SOURCE, O=N, LST=NC
    // DVC 20  // LFD PRNTR
*2  // DVC 51  // VOL PACK02
    // EXT ST,,5,,(256, 8000)
    // LBL $SCR1  // LFD $SCR1,16
*2  // DVC 52  // VOL PACK03
    // EXT ST,,5,,(256,8000)
    // LBL $SCR2  // LFD $SCR2,16
*2  // DVC 50  // VOL PACK01
    // LBL $Y$MAC  // LFD ALTMAC
*2  // DVC 50  // VOL PACK01
    // LBL ALTSRC  // LFD ALTSRC
    // DVC 52  // VOL PACK03
    // LBL ALTLOD  // LFD ALTLOD
    // OPTION SCAN,SUB
*3  // EXEC ASM,ALTLOD
*4  // PARAM COPY=(N)
    // PARAM LIN=ALTMAC/(N)
    // PARAM IN=&N/ALTSRC
    // IF ('&O' NE 'N')OBJ
    // PARAM OUT=(N)
    //OBJ NOP
    // IF ('&LST' NE 'NC')XREF
    // PARAM LST=(NC)
    //XREF NOP
    // IF ('&L' EQ 'N')NOLINK
    // IF ('&O' EQ 'N')NOLINK
*5  // SKIP NOLINK,1100
*2  // DVC 51  // VOL PACK02
    // EXT ST,,5,,(256,8000)
    // LBL $SCR1  // LFD $SCR1,16
    // OPTION SCAN,SUB
*6  // EXEC LNKEDT,ALTLOD
    /$
            LINKOP  (PARAMS)
            LOADM &N
            INCLUDE &N,$Y$RUN
    /*
    // GO ENDJ
    //NOLINK OPR 'Program Was Not Linked'
    //ENDJ NOP
    /&
    // FIN
```

NOTES

*1 Variables which must be keyed in at run time.

L=N	Force user to specify link.
N=SOURCE	User must specify Program Name.
O=N	Don't generate object module until program is correct.
LST=NC	Don't generate a cross reference unless necessary.

*2 Specify actual disk packs used.

*3 Execute Assembler.

*4 Define Parameters.

*5 Test UPSI for a clean assembly.

*6 Execute linker from a specific load library.

*7 The module name in the source library must be the same as the name on the assembler start card.

JOB CONTROL

```
    // JOB COBOL1,,(SIZE-SEE SECTION 5)
*1  // GBL N=COBOL1,L=N
    // DVC 20  // LFD PRNTR
    // DVC 52  // VOL PACK03
    // EXT ST,,5,,(256,8000)
    // LBL $SCR  // LFD $SCR
*2  // DVC 50  // VOL PACK01
    // EXT ST,,5,,(256,8000)
    // LBL $SCR2  // LFD $SCR2
*2  // DVC 51  // VOL PACK02
    // EXT ST,,5,,(256,8000)
    // LBL $SCR3  // LFD $SCR3
*2  // DVC 50  // VOL PACK01
    // LBL ALTLOD  // LFD ALTLOD
*2  // DVC 51  // VOL PACK02
    // LBL $Y$OBJ  // LFD ALTOBJ
*2  // DVC 51  // VOL PACK02
    // LBL ALTSRC  // LFD ALTSRC
    // OPTIONS SCAN,SUB
*3  // EXEC COBOL74,ALTLOD
*4  // PARAM LIN=ALTSRC
    // PARAM IN=&N/ALTSRC
    // PARAM OBJ=ALTOBJ
    // PARAM ALTLOD=ALTLOD
    // IF ('&L' EQ 'N')NOLINK
*5  // SKIP NOLINK,1100
*2  // DVC 52  // VOL PACK03
    // EXT ST,,5,,(256,8000)
    // LBL $SCR1  / LFD $SCR1
    // OPTION SCAN,SUB
*6  // EXEC LNKEDT,ALTLOD
    /$
          LINKOP (PARAMS)
          LOADM &N
          INCLUDE &N, $Y$RUN
    /*
    // GO ENDJ
    //NOLINK OPR 'Program was not Linked'
    //ENDJ
    /&
    // FIN
```

NOTES

*1 Variables must be keyed in at run time.

L=N Force the user to specify link.

N=COBOL1 User must specify program name.

*2 Specify actual disk packs used.

*3 Execute Cobol from specific load library.

*4 Define Parameters.

*5 Test UPSI for clean compile.

*6 Execute linker from a specific load library.

5. RECOMMENDED SIZES FOR COMPILERS AND ASSEMBLER FOR MAXIMUM PERFORMANCE

		DECIMAL
ASSEMBLER	= X'10000'	ASSEMBLER = 65536
COBOL	= X'12000'	COBOL = 73728
FORTRAN	= X'19000'	FORTRAN = 102400
RPG	= X'12000'	RPG = 73728

These sizes are recommended for performance and may have to be increased based on program size and complexity.

6. MINIMUM SIZES FOR COMPILERS AND ASSEMBLER AS PUBLISHED IN THE "SRD"

ASSEMBLER = 20480
COBOL = 57344
FORTRAN = 67584
RPG = 29626

*Note: Minimum sizes published in the SRD are in decimal.