SILVER PROGRAMS RSTS OPTIMISATION NOTES 14-Aug-90

Peter Dick

Silver Programs Ltd

Latchmere Green

Little London

Basingstoke

Hampshire RG26 5EJ

Voice 0256-881658

Fax 0256-881778

_

IS THERE A PROBLEM?_

The system is slow...

- On line response time
- Perceived response time
- Batch elapsed time

Educate all Users

- Send them on courses
- Keep them informed of plans/actions
- Ask them to help you

Familiarity breeds fast keying rates

Encourage them to behave in a responsible manner

Do not leave Terminals logged in unnecessarily

STATUS counts them as Users

Security considerations

_

IDENTIFY THE CAUSE_

Only two main options:-

Is the system disk bound

Is it CPU bound

In theory...

Is there enough memory

Is the system bus bound

Too much terminal traffic

Productive processing or overheads

System overheads

Your own application code

Record / file handler bottlenecks

Discover raw speeds for each bit of hardware

Then calculate probable speeds for your system

SILVER PROGRAMS RSTS OPTIMISATION NOTES 14-Aug-90

_

PERFORMANCE ANALYSIS_

How do YOU know if things have got any better?

How do the USERS know if things have got any better?

Cheap tricks

Change the terminal speeds...

Display progress while updating a file

Make Prints show start and stop times (and cpu time?)

Rewrite the main menu (using TABS not SPACES)

Reschedule batch type jobs

DYNPRI

One day's thinking will save one month's tuning

_

TIMES CHANGE_

Remove all word processing to separate machines

Consider some new hardware

Disks

- Cache controllers
- Volume shadowing
- Multiple controllers

Tapes

- TK50
- Streamers v Start/Stop 800, 1600, 3200, 6250
- 3200 feet tapes
- Megatape, Exabyte, 4mm DAT

Line Printers

- LP25 -> LP26 -> LP27

Dual porting

- Disks
- Tapes
- Line Printers
- Terminals

Upgrade to Version 9.7

Reschedule processing

- Leave system running 24 hours per day
- Make use of the lunch hour
- Consider mid day BACKUPs
- Use Batch overnight to reduce daytime loadings
- Get more hours in day by redesigning DAYEND procedures

SILVER PROGRAMS RSTS OPTIMISATION NOTES 14-Aug-90

_

DISK INITIALISATION_

Avoid multiple public disks

Choose an efficient pack clustersize

Centre of used part of disk

- SATT.SYS (Storage Allocation Table) pre V9.1
- Don't centre SATT.SYS post V9.1
- SWAP.SYS
- UFDs - Contiguous - Pre-extended

Don't use New Files First without extreme justification

Dates...

- Last Accessed v
- Last Modified tradeoffs

_

SYSGEN OPTIONS_

Include STATS at SYSGEN time pre 9.6

Configure only actual requirements

Configure 'enough' Small Buffers

Use overlapped seek
  Unless single disk configured as multiple units

Don't use Data Caching without thinking pre 9.0

Make all monitor options resident (V8)

_
SYSTEM DEFAULTS_

Set XBUF to between 50K - 496K words

Set Virtual Disk as large as you can usefully use
  without causing undue SWAPping

Use BUFFER option for more Small Buffers

Install one swap file (SWAP0.SYS) on Virtual Disk

Load Run Time Systems / Libraries
  In the "right" order
  LOCKed at fixed addresses

Avoid fragmenting memory

Load all overlays ($LOAD OVERLAY/ALL) under V9

SILVER PROGRAMS RSTS OPTIMISATION NOTES 14-Aug-90

_

DURING TIMESHARING_

Run STATUS on a daily basis

Interval 1 hour?

Ensure files have correct Clustersize

Make as many large files as possible contiguous

Set limits on Data Caching v Directory Caching

Flag files for Caching with great care

Never set Cache/ALL during Timesharing

Consider Set Cache/ALL for Batch processing

Set Cache Clustersize to 8

Play around with /KEEP

Use LOAD/INDEX post 9.1

Balance disks:-

storage v

accesses v

transfers

Centre very heavily used files

But don't expect any real benefit!

Place very heavily used files at top of directory

  Consider a Sleeper to keep 12 files open all day

Keep directories small

Run REORDR frequently

  Use DAILY.COM command file

Pre extend new production files

Do not run all keyboards at 9600 baud unnecessarily

  Do printers print any faster at faster BAUD Rates?

Advocate decentralised printing

  Advocate 'Exception' Reporting

  Do not suggest printer silencers

VT220's use SET TERM EDIT in EDT

Use WRITE option of EDT to print out parts of files

Avoid the following when the system is 'busy':-

  RUN $MONEY

  PIP /ZE with files that have bit 128 set

  Initialising Disks and Tapes

  Full directories

  Version 9 BACKUP with /BUFFERSIZE=MAX

  BASIC+2 Compilations and TASK building

SILVER PROGRAMS RSTS OPTIMISATION NOTES 14-Aug-90

_

VERSION 9_

It has been out for five years

There have been eight releases

With one exception (LOGIN) it is faster than V8

It has good features

Use /POSITION:1 on BASIC PLUS RTS if your programs are .BACs

Modify standard START.COM

Teach users to use V9 - do not SW BASIC in LOGIN.COM

Use $ in all command procedures

Remember CCLs can be used to call command procedures

Don't add unused CCLs

Use Symbols as commands

SET PROMPT to show current account

```
$ SP=="-SET PROMPT F$CHR(72)+F$CHR(55)+F$CHR(51)
      +F$CHR(36)+F$CHR(32)+F$USER()+F$CHR(32)"
$ SP
```

Don't waste resources with QUOTAs: set all to unlimited

Use Captive A/C to keep users in controlled environment

Virtual disk

- Improves life if currently disk bound
- Reduces performance if CPU bound during Timesharing
- Always good news for single batch stream
- Put BP2IC2 and $TKB on DV0: if single user system
- Small control files
- DCLWRK$ under V9.7

Consider using BACKUP to reorganize disks regularly

- Flag critical files as Placed
- Delete all files no longer required
- BACKUP whole disk - at least twice with /VERIFY
- Initialize Disk
- RESTORE disk will give placed files best position

Consider using PIP to reorganize disks regularly

- /CL:0 will create optimum clustersize
- /SE:n will select on filesize (V9.3)

Use I+D space option on system build if possible

When you upgrade use it as an excuse to tidy up system

Rebuild BASIC+2 to take advantage of Version 9 Features

SILVER PROGRAMS RSTS OPTIMISATION NOTES 14-Aug-90

_

APPLICATION PROGRAMS_

Record / file handler bottleneck

RECORD I/O v

Virtual Arrays v

formatted ASCII

Use Logical names to allow files to be moved around

Consider use of hash PPN to reduce files in Owners PPN

Avoid excessive file Open and Closes

Keep Production files separate from test files

Batch "scan" programs

Use huge RECORDSIZEs

Use Read regardless

Making programs more robust will save time

But do not over use extend mode

BPCREF all production programs

Make programs smaller

Remove all unused code

Make programs bigger

Compile at actual run size +2K words

Reduce Garbage Collector overheads

Use BELL patch

Reduce CHAINing overheads

Consider SEND/RECEIVE

Consider SWAP CONSOLE

Consider Multi TTY Service

Use Multi TTY Service if you are a genius

Eliminate VALs within loops

CHANGE v ASCII(MID(W$,W%,1%)) FOR W%=1% TO LEN(W$)

Don't FIELD whole record - only the required elements

Initial FIELD of whole buffer

v FIELD per record

Avoid OPEN/KILL -> use a Null Device

Character movement

LSET / RSET with FIELD v

LEFT / RIGHT

INSTR much faster than LEFT / MID / RIGHT

Use XLATE to strip out unwanted characters

Think about what statement modifiers do

Beware hidden I/O

Extend work files to maximum size

Use Clustersize 256 for spool files?

Open work files on Virtual Disk

Use OPEN AS rather than OPEN FOR OUTPUT for work files

Read the programming manual

Use SPEC% rather than opening file on two channels

Use Filename stringscan and STATUS

(rather than pages of lexical analysis code)

OPEN MODES

MODE 5 - guarded update - instead of MODE 1

MODE 8 - Echo control - for all keyboard input

MODE 128 - No superses - on OPEN for OUTPUT

MODE 8192 - read only - for reading VIRTUAL ARRAYS (if you must)

Unset contiguous flag with sys call when extending file

Cancel type ahead in error traps

SLEEP

On disk block interlock

When using Pseudo Keyboards

When detached

SLEEP 0 if using big Run Burst

Spawn jobs rather than detaching them

(and then set them to run at reduced priority)

Use cursor positioning not TAB()

Avoid Repainting Screens

Use Clear to end of line/screen rather than SPACE$

Avoid Spot the cursor input

Split report programs

INPUT all parameters

Detach/Spawn

CHAIN to Report module

Give maximum core to I/O

Get print programs to use printer features

Get Peripherals to do the work if they can

Sorting

Don't bubble

Use QUIK techniques

SILVER PROGRAMS RSTS OPTIMISATION NOTES 14-Aug-90

_

GENERAL INDEXING METHODS_

Use as little as "Sensible"

Use small keys where possible (Minimum Unique Keys)

Avoid designing systems which continually change Keys

The less keys the faster the Updates

How many 'real time' keys do you need?

Can you rebuild some indices overnight?

Never use temporary indices

Consider using Sorts for reports on transaction files

Remember ALL ISAM file handling methods use a lot of I/O

Always Consider Direct access methods

Consider adding extra keys to static files

(rather than transaction files)

Always write an index print program

Always write an index rebuild program

Original IAM functions can be made to go much faster

_

RMS_

Reorganize regularly (RMSDES+RMSIFL)

Avoid multiple duplicate keys

Use NULL key feature rather than changing keys

Use GETRFA() rather than doing two lookups by KEY

Remember 2*Maximum bucketsize program overhead per file

Keep recordsizes as small as possible

Sequential access by primary key faster than secondary

Avoid loading file by RMS PUTS use RMSIFL

Always use RMS file design to pre-extend file

Always have primary key at start of record

Remember one LSET can generate hundreds of disk I/O's

Remember that the GET PUT and UPDATE statements do not
  generate only one disk I/O + remember SPEC% overheads

SILVER PROGRAMS RSTS OPTIMISATION NOTES 14-Aug-90

_

UTILITIES_

ACCESS

Reorganize regularly (ACCREO)

Use "FI" rather than "PA" to test if key exists

Use Multisequential ACCESS in all reports

Consider using Sorts and Multiple Adds for file loading

Use Large Bucket Size

Use New ACCESC - ACCESS caching version to give up to

30% improvements on random access (Not on 18bit)

BASIC LANGUAGE

BASIC PLUS

BAS24K

BASIC PLUS 2

BP3

Develop in Basic Plus:

Time to compile a program

Run under BP2 / BP3

Processor bound programs will run much faster

BP2 threaded code v BP3 pure code

BP2 can use RMS files and can CALL external subrtns

BP2 overlays v BP3 paging

DECOMPILERS

$PPCODE

PART1

UNBAC

The LINK

Allows upto 5 directly connected processors

Increases maximum number of Users to over 100

Speeds up all aspects of a system

Solves CPU bound problems

Does not solve disk bound problems

Supports all languages (except PRO IV)

Supports all disk modes - including RMS

Requires Version 9.2 or later

MENU-11

WORD PROCESSING

EDIT11

TECO/VT

EDT

LEX

DECWORD

WORD11

WPS+

QTEXT11

_

UTILITIES_

PROTOCOL CONVERSION

- 2780
- Black Boxes
- Serial -> Parallel

PERFORMANCE MEASUREMENT

- QSTATS
- STATUS
- EMT logging
- RPM

SORTING

- SQWIK & MQWIK
- SORT.TSK
- SORT1
- FSORT3

DIRECTORY OVERHEADS

- REORDR.BAC
- REORDR.TSK
- DISKIT
- SDDOPT
- SUMMIT

SPOOLING

  OPSER SPOOL

  PBS

BACKUP PACKAGES

  BACKUP (V8)

  BACKUP (V9)

  SAVRES

  PIP

  SAVER

INTER PROCESSOR COMMUNICATION

  KERMIT

  FNTALK

  DECNET

  ETHERNET

NEWS

  DEC

  DECUS

  DECPROFESSIONAL

  DECUSER

SILVER PROGRAMS RSTS OPTIMISATION NOTES 14-Aug-90

_

HARDWARE OPTIONS_

Upgrade to 11/93 or 11/94

Buy a 11/70 with FPP and MASSBUS disks if lots of space

Consider PEP-70

Upgrade to a disk cache controller

Buy more disks

Buy faster disks

Buy small RAM disks if small amount of data

Buy big RAM disks if rich

Upgrade to a cache tape drive (TU81+?)

Swap DZ11s for DHU11s

Swap LP25 -> LP26 -> LP27

Ethernet now available

_

SUMMARY_

Identification of problems

- CPU bound
- Disk bound
- Memory
- Tapes
- Printers
- Keyboards

- Quantify the damage
  - How much will it cost in money
  - How much good will it do
  - How long will it take to effect
  - How long will it last

- Supplier
  - Digital
  - Others

_

WHEN ALL ELSE FAILS_

- Duplicate - don't (e)migrate
  - If you really must move, reprogram don't convert

- Ask Digital for an 11/93+

- Give up
  - Simply change job

```
_
RUN UNSUPP$:STATUS_

Output to <KB:STATUS.LST [/VT05, /VT5x, /VT100]>?
```

Respond with the name of the output device or file. Append /VT05, /VT50, /VT52 or /VT100 if the display is written on one of these terminals. STATUS then asks:

```
Interval <60>?
```

Respond with the interval between displays. Using this value as a guide, STATUS chooses a sleep time so as to present the display at the requested interval. If you respond with an explicit zero value, one display (only) is output. STATUS then asks:

```
Percent mode (Yes/No) <No>?
```

Respond with "Yes" if you want directory, user, and swap statistics to be presented in terms of their percentual contribution, or with "No" if you want a presentation in terms of accesses (or blocks) per second. You can switch back and forth between the two; see below. STATUS then asks:

```
Physical I/O only (Yes/No) <No>?
```

Respond with "Yes" if you want to see only I/O requests that caused a physical disk operation, or with "No" to see all requests, including those satisfied from the cache.

If the display interval is not zero, STATUS asks:

```
Detach (Yes/No) <No>?
```

Respond with "Yes" to detach the program. Processing begins.

If STATUS is printing on a terminal, it checks for input from that terminal. The following may be typed (followed by a delimiter):

C	Clear the screen
%	Display in percent mode
V	Display in access/second mode
P	Display physical I/O only
R	Display all requests
N	Display non-mounted disks as well as mounted disks
M	Display only mounted disks (the default upon startup)
Dn	Display disks starting from disk unit n

If you type CTRL/C at the terminal, STATUS closes its output file and asks if it should detach. Type "Y" if you want detaching.

The display is divided into two parts: the upper has monitor and directory cache information, while the lower has disk data.

System status

The first line of the display contains a status line, including date and time of the display, the system identification, and the number of errors logged (excluding hung terminals).

Jobs

The number of jobs logged in, and the maximum number permitted.

User running

The percent of time spent running user programs.

Fip needed

The percent of time during which at least one job was waiting for the file manipulation service (to open, close, or extend a file, for example).

Seconds

The interval over which this sample was taken.

Idle

The percent of time during which the system was idle.

SYS Charged

The percent of time during which the system was executing monitor code, while some job was running.

Fip in use

The file processor was executing. This time is also part of SYS charged (or uncharged).

Cache hits

The value printed is the percent of directory blocks that were found in the XBUF cache.

Lost

Lost time is the percent of time no job in memory could run, but some job on the disk was selected for running. If high, you need more memory.

SYS uncharged

This is the percent of time monitor code was run which was not chargeable to some specific job.

I/O service

This is the percent of time the system serviced interrupts at priority level 4 (terminals and other slow devices), and priority level 5 (disk and magtape).

Cache CPU

This is the percent of time the Directory cache code was running.

Char/sec out

This is the number of bytes per second output to all terminals.

Char/sec in

This is the number of bytes per second input from all terminals.

Min sml buff

This is the minimum number of small buffers not in use.

Access per Sec

This group of numbers presents the number of transfers per second

for each drive and unit.

Blocks per Sec

This group of numbers presents the number of disk blocks transferred per second.

Total

This column specifies each drive's contribution to the total system load.

Reads

This column specifies the percentage of accesses which are reads.

Direct

This column describes directory read and write accesses.

User

This column describes user data read and write accesses.

SwpSys

This column describes swap and other system accesses, including overlay code and DECtape buffer accesses.

SILVER PROGRAMS RSTS OPTIMISATION NOTES 14-Aug-90

While the STATUS program is often used to monitor the throughput of the system, there would appear to be some mis-understandings about the~ Blocks per Second~figures.

A 512 Block contiguous file is read sequentially one block at a time using a standard 512 byte buffer. Data caching is enabled with a cache clustersize set to 8 and the file is marked for sequential caching.

In this first table, "Y" was entered to the question "Physical I/O only?", ie only physical I/O is displayed:-~

```
14-Jan-90  14:42:07  RSTS V9.7-08 Silver Ghost 73         3 Errors
  5/12 Jobs   23% User running     0% Fip needed        8.6 Seconds
   20% Idle   26% SYS charged      0% Fip in use        88% Cache hits
    0% Lost   30% SYS uncharged    7% I/O service       13% Cache CPU
             97.3 Char/sec out     .1 Char/sec in       214 Min sml buff
Access                                    Blocks
per sec Total Reads Direct User SwpSys  per sec Total Direct User SwpSys

    7.4   98%  100%         7.4            59.5  100%    .0  59.5     .0~
```

This shows 7.4 Accesses per second for 8.6 seconds = 63.64 accesses in all. This figure is correct - since it took 64 accesses, each of 8 disk blocks, to fill the data cache. (64 * 8 = 512)

This table also shows 59.5 Blocks per second for 8.6 seconds = 511.7 blocks read. This figure is also correct - the file is 512 blocks long.

In the next table, the default "NO" option was used in reply to the question "Physical I/O only?", ie all I/O is displayed:-~

```
14-Jan-90  14:43:30  RSTS V9.7-08 Silver Ghost 73         3 Errors
  5/12 Jobs    22% User running     0% Fip needed        8.8 Seconds
   21% Idle    28% SYS charged      0% Fip in use        88% Cache hits
    1% Lost    28% SYS uncharged    7% I/O service       11% Cache CPU
            101.8 Char/sec out      .1 Char/sec in       214 Min sml buff
Access                                    Blocks
per sec Total Reads Direct User SwpSys  per sec Total Direct User SwpSys

   58.1  100%  100%        58.1           108.8  100%    .0 108.8     .0~
```

This shows 58.1 Accesses per second for 8.8 seconds = 511.28 accesses in all. This figure is correct - the file is still 512 blocks long.

This table also shows 108.8 Blocks per second for 8.8 seconds = 957.44 blocks read. This figure is misleading...

STATUS counts the 64 accesses that read the data from disk into cache in 8 block transfers (ie 512 disks blocks were read from disk) PLUS the 7 out of each 8 disk accesses that were found in the cache (ie 448 disk blocks were read from cache). This makes a total of 960 blocks read...

SILVER PROGRAMS RSTS OPTIMISATION NOTES 14-Aug-90

_

UNDOCUMENTED FEATURES_

The RSTS Operating System containeds many undocumented features. Although these features in recent years have tended to become humorous, the early examples were often left in by the programmer for his/her own benefit.

1. Our first feature, which has been around since the very early 1970s, is the Basic Plus command ~NAME "TEST.DAT/SI:-1" AS "TEST.DAT"~which will always work even if an existing TEST.DAT file would normally have failed with a ?Name or account now exists message. It was written to save space in the original EDIT/EDITCH Basic Plus editor programs.

2. Another feature that has been available since the middle 1970s is~ SET SCOPE.~ At the standalone OPTION prompt:-~SET SCOPE~will allow rub out characters entered at KB0 to erase the previous data rather than be treated as a hard copy device with an \a\ action. This command causes this characteristic to be stored in INIT.SYS forever - or until you reset it with a~SET HARDCOPY~command.

3. Next, a Basic Plus feature that was coded in about 1976. All PEEK

commands need privileges - except one. ~PEEK(-136%)~examines the switch register. This was programmed into RSTS to allow the TECO Run Time System to be debugged using the hardware switch register from a non privileged account. Talking of TECO...

4. If you have MAKE as a CCL for TECO, try ~MAKE LOVE~ as a command.

5. The TECO.INI file contains several useful time dependent messages. To experience these messages, copy the TECO.INI file into your account, make TECO a CCL and then type~TECO.~

   Esta muy loco? You should be in bed.
   The early bird only gets worms.
   I am very tired. Please let me sleep.

6. A hidden option that will dramatically increase performance of DH-11 interfaces has existed since the days of Version 6.

   The effect of this feature is to reduce the cpu overhead for character I/O - by a factor of 10 for strings of over 200 bytes. The code was written for the field service remote diagnostic systems and, as far as I know, is still in use today!

   The option is too complex for today - See DEC PROFESSIONAL February 1987 page 54 for full description.

_

UNDOCUMENTED FEATURES_

7. It would appear that most $TKB Users don't know about the OV: system wide logical. This can be set to the Virtual Disk and will greatly reduce task build times. (NB it is worth while having $TKB and BASIC2 on the Virtual disk as both of these are very overlaid.)

8. Ever since the introduction of Version 9 (May 1985) there have been three date formats:-~

```
SET SYSTEM/DATE-FORMAT=ALPHABETIC
SET SYSTEM/DATE-FORMAT=NUMERIC
SET SYSTEM/DATE-FORMAT=STAR~
```

The first date produces DD-Mmm-YY, the second MM-DD-YY and the third DDDD.dd is based on STARTREK dates. This means that the DDDD is increased by 1 for each day since RSTS was "launched" and the .dd refers to the decimal number of hours.

For example, ~PRINT DATE$(20016%)~generates 16-Jan-90 for Alphabetic format and the same command displays 7160.75 if run at 18:00 or 7160.50 if run at 12 noon.

9. Also introduced with Version 9.0 were two commands left in DCL to help with its development.~

   SET VERIFY/WATCH~causes the CCL commands generated by DCL to be displayed. For example, PIP /DI:W:HD:PR:SI=:*.* is displayed if you simply type DIR.~

   SET VERIFY/NOP~causes the DCL commands to be ignored - NOP standing for No OPeration.

   These commands have two good uses:- as a migration aid for Users changing from Version 8 to Version 9~SET VERIFY/WATCH/NOP~shows what DCL is going to do without it actually doing it. In addition, you should consider inserting ~SET VERIFY/NOP~ into [0,1]LOGIN.COM for April Fools Day.

   To turn the operation mode back on, simply enter the obvious ~SET VERIFY/NONOP~ ie NO No-OPeration.

10. Lastly, RSTS 9.6 saw the introduction of the latest undocumented feature - but first some background information...

    You need to know that the PC SIG in the States have a mouse as a mascot, the VMS SIG a cheshire cat and the RSTS SIG a bull dog called Spike. Under RSTS 9.6 and 9.7 type~HELP SPIKE.~