Apr 4, 1985

Thank you for your attendance at the Fortune Worldwide Sales Conference and in particular for attending Workshop #3. As we agreed at the workshop I am now putting on paper the issues that we discussed. I am sorry it took so long to do so but it seems there is never enough time in a day. During the course of that workshop we discussed many things related to configuring the Fortune 32:16 for maximum performance. The discussions mainly revolved around the following issues.

1. How the Fortune 32:16 allocates memory at startup time.
2. Utilities available to monitor performance and configuration.
3. Things that can be done to enhance performance.

The following material is documented in one way or another in either the Introduction to For:Pro text or the Fortune Programmer's manual distributed with Development Utilities. Additionally the Fortune Field Service Manual has information concerning system configuration.

The implications and deductions drawn from this information is solely my own and do not necessarily reflect that of **Fortune Systems Corporation.**

The underlying philosophy of this paper is that a multiuser system performs best when all tasks are resident in RAM. This of course is an ideal that is not always possible. With this in mind we will attempt to do what we can to avoid swapping but if swapping becomes necessary we will attempt to do so as efficiently as possible.

I. 32:16 Resource Allocation

The following is a representation of the settings in EAROM on the Fortune 32:16. This menu can be displayed and/or modified in two ways. First, from the unix prompt as superuser it can be displayed by entering uconf. If etc is not in your path enter /etc/uconf. This allows you to view and change these setting while the OS is up. Keep in mind, however, that any changes will not be reflected until the next powerup. The other method of accessing this information is from a cold start. Turn the power to the 32:16 on while holding down the Cancel/Del key. A menu showing powerup actions should then display. Press the F7 function key to change the boot. Enter hd02/sa/reconf and press return and then execute. This should take you into the configuration menu.

Fortune Systems Configuration Menu

Power up action = BOOT		Daylight savings = YES	
Boot device = SLOT E		Line frequency = 60	
Boot drive # = 00		Language = ENGLISH	
Boot Program # = 00		Floating point? = YES	
Boot file = hd02/unix		Hex number = 0000	
Flex drive #1 = SHUGART		Number buffers = 200	89
Flex drive #2 = SHUGART		Number inodes = 200	17/ 100
Flex drive #3 = SHUGART		Number files = 127	13/ 100
Flex drive #4 = SHUGART		Number texts = 015	4/ 37
Root device = hd02		Number clists = 050	0/ 48
Swap device = hd11	344/5120	Number processes = 064	12/ 37
TTY00 port speed= 2400		Max process size = 350	
TTY01 port speed= 1200		Set params auto? = YES	
Console location= CRT		Appx. # of users = 3	
Timezone = PACIFIC			
	Used/Total	Used/Total	

Total primary storage = 1024K; programs are using 290K of 788K available
EAROM has been changed 37 times

F1 STORE SCREEN DATA IN EAROM **F2** READ CURRENT EAROM SETTINGS
F3 EXIT WITHOUT CHANGING EAROM

Although all of the fields on the menu are important we are concerned here with mainly the bottom right portion. The first field that we are concerned with is the Set params auto. This can be set to yes or no. By default it is set to yes. When it is set to yes the contents for the number of buffers, inodes, files, texts, clists, and processes are calculated by using the following formula against the number of users field.

```
# processes = (# users * 10) + 7          /*17,27,37....*/
# text = # proc                           /*17,27,37....*/
# inodes = ((#users-1)*25) +50            /*50,75,100...*/
# files = # inodes                        /*50,75,100...*/
# callout= #users + 8                     /*9,10,11.....*/
# flocks = #proc *2                       /*34,54,74....*/
# buffers = (core/10)-13                  /*38,63,89,100*/
```

PRIMARY STORAGE

For most environments all you have to do is set auto params to yes and store the actual number of users in response to the Appx. # of users field. The number entered should reflect the maximum number of users that would be using the system at one time. This would include active terminals as well as any background processing such as ghost tasking in the Business Basic Interpreter.

If you do not feel that you are getting the optimum allocation for your environment you can then use some of the utilities in the next sections to determine what you really need and are using. Once this has been determined you can modify these settings.

The recommended way of doing so is to write down the current allocations. These can be found in "Total" Fields on the far right of the uconf screen.(the total number of 512 character blocks allocated for swapping is in the middle of the page). Once you have written down these current totals you may change auto params to no and plug in these numbers as a starting place. NOTE: even if you set auto params equal to no you must maintain the approximate number of user field as it is used elsewhere by the operating system. What we are actually attempting to do here is to make as much memory available to users as possible. Changing some fields such as the number of clist for example really do not make that much difference so it may be best to leave them alone. However, the number of buffers allocated reflect 1K buffers. If we look at the above example there are 89 1K buffers reserved on my system. If I determined that I never used more than 30 buffers for example and I changed that field to 39 just for safety I would have reclaimed 50K of users use. That 50K might be just enough to keep me from swapping in certain applications.

The other important field is the maximum process size. This field represents the maximum amount of memory that any one process can allocate. The system will not allow any more than that amount. Additionally if there is not enough available after allocations and the kernal it will reduce this field. Many applications and languages instruct the user to modify this field as part of the installation process. Other time people are instructed via word of mouth to increase this field for efficiency in certain applications such as word processing. The real danger in setting this field to high lays in the possibility that programs and utilities my not adequately call for resources and will hence lock on to this amount.

The Business Basic Interpreter and applications written with it handles the maximum process size a little differently. They have ipl files located in /b/ipl that are used when a program is loaded. There are usually front end shell scripts that determine which ipl will be used for an application. The ipls allow for a memory "partition" size. On most systems this will be set to 20K. This field can be increased for larger applications. The reason it is set to 20K is due to the BAS applications which are predominantly written in 7 to 8 K modules. The system will use this partition size parameter to lock on to memory space. For example if set to 30K each user that uses that ipl will be allocated 30K. This is true even if the program and data they are actually using is 10K. So we can see by this that there is a danger in setting this field to high.

System Utilities

1. Pstat — This command is documented in the For:Pro Manuals. It can be used to take a snapshot of system utilization. Two of it's arguments not too widely known allow for the number of times and at what interval the command will run. For example the following command

```
pstat -m 2 5
```

This will have the effect of checking memory availability every two seconds until it reaches a count of five. Additionally as with most unix commands the output can be redirected as in the following

```
pstat -m 360 10 >> junk
```

The above should check memory availability every hour up to 10 times and append that output to a file called junk in the current directory. This all gives us the ability to check on resources at prescribed time at save the output that could be printed at a latter date for evaluation.

2. vmstat — This command can also be used to check on the use of system resources by programs. Check the For:Pro manual for options and descriptions of output. This utility is probably of most use when you are developing programs as opposed to checking on applications supplied by Fortune or other vendors.

3. size — The size command will allow you to determine the size of an object program for both text and data. With this you can determine the overhead of a program on both a one time and per user basis.

4. ps — This command in addition to being useful for determining what processes are active is useful in finding programs that may not be shared. The SZ parameter if very large is probably a good indication that the code is not shared.

Performance Enhancements

1. Menu System

The Fortune Global Menu system is of course a very user friendly interface to the Fortune System and software. As is the case with everything that is nice there is a cost associated with it. For the following example I am using this document as a control using Fortune:Word for editing. I am going to present the memory usage in creating this document using different modes of accessing the word processing software. First I will enter via the global menu system. I use the pstat -m command on my console while using terminal 02 to do my work. The following is the memory available to other tasks and users along the way.

	Step	Memory available
437	At login prompt	610
321 302	At global menu	531
	At wp menu	405
	In edit	263
	In edit via index	191

Now I set up my login to go directly into the word processing menu.

Step	Memory available
At wp menu	502
In edit	360

Now I set up my login to go into the borne shell from there I will run wp2

Step	Memory available
At shell	609
At wp menu	483
In edit	341

Now I set up my login to execute a shell script that execs /m/wp2/wped "filename" This in essence takes me directly into the editing session.

Step	Memory available
In edit	486

What the above really shows is the difference between worse case and best case. The worst case is where I went through the menu system, into Fortune:Word, into the index, and then pressed the goto key. In this case I had only 191K available to other users on the system. In the best case where I went directly into the editing session there was 486K available. This represents a difference of 295K..

Of course the above methods may not always be possible, however tests with Multiplan and Business Basic show the same kind of results. In word processing in particular if there is anyway of implementing the above great savings can be made. In addition to saving memory word processors can be saved from going through multiple menus and security is enhanced as when finished in the application the user is taken back to the login screen.

Most all of Fortune distributed applications can be directly called. When in doubt as to how to do so run the application via the menu system on one terminal and use the ps command on another terminal to see what is actually being run.

The examples above really represent a utopian environment in that it is only considering one user. Of course most applications are sharable so the savings would not be as great if for example you had one user going directly into Fortune:Word for example and another accessing Fortune:Word via the global menu.

Before leaving Fortune:Word an additional note. As we saw above the ability to enter a document from the index creates extra overload. The use of the insert mode in document creations also seems to improve the throughput to the user. A major consideration is using page breaks when creating documents. A simple test will show this. Take a 10 page document that has been properly paginated. Go in and edit that document. On another terminal do a ps and pay close attention to the SZ entry for wped. Now make a copy of that same document. Edit the copied document and remove the page breaks. Save the copy and then go back into edit on the modified document. Again run ps on another terminal. You should see a change on the SZ entry for wped. This would appear to show that unpaginated documents will take more and more memory until reaching the maximum process size.

Remember the earlier discussions of the Business Basic ipl files. The memory partition field is going to lock in memory for each user that accesses it. Lets assume a multi module application where most modules are under 20K but one big one is 50K. If the ipl is set to 50K that is what is allocated. If there is anyway to put the entrance into the 50K module from a different ipl and disallow users of a 20K ipl access to that module substantial memory can be saved. A good example of this would be a ghost process.

Most often ipl files and the Basic Interpreter are accessed through front end shell scripts such as DBASIC, BASIC, TBASIC, SBASIC and so on. The preceding four in order we the Basic interpreter, the BAS applications, the training data base for BAS, and Business Surveys. All of these shell scripts provide the same function. They allocate an ipl based on a round robin approach, set terminal characteristics, start the interpreter, and wait for completion to reset terminal characteristics. Since this shell waits for completion is can be using 20K of memory. An alternative method would be to have the shell script exec basic.psd instead of just running it. This would cause the shell script not to wait. Of course the terminal would be in high intensity but if you are returning to the login screen that should be no problem. Additionally decision logic could be added to the shell script to allocate the ipl based on the tty number or login name as opposed to the round robin approach. This would allow for more customized ipls.

Performance Enhancements (cont)

2. Swap Allocation

By default Fortune formats hard and flexible disks with up to 3 partitions. Partition 0 is usually boot, partition 1 swap, and partition 2 is the file system. On a single hard disk system hd01 is used for swap. If you have a system with an expansion disk you may find performance improvement by placing the swap on the expansion disk. This seems to make an improvement in performance in that head movement is reduced for data acquisition and swapping. Of course to be able to do this you must manually format and set up the expansion disk for swap use. The menu driven software distributed with the expansion disk sets up little if any room in partition 1. So you would have to manually setup the expansion disk for more swap space. To do so refer to the format, mkconf, mkfs, and lost+found entries in the For:Pro manual. Once this has been set up it will be necessary to modify the uconf menu to place swap on hd11 instead of hd01 for example. It is then necessary to use the setnswap command so that the kernal will know the proper device and partition. The above assumes two disks of like type and access speeds. If you have system with an internal stepper motor 20 and an Expansion with a voice coil 20,30, or 45 you may want to put the os on the expansion and use the internal for swap.

As stated before we usually set up 3 partitions. Actually 8 are allowed (0-7). You may find it desirable to set up more partitions on a device and mount them at startup time. The benefit of this is that you can have smaller file systems. It may also be desirable in case of file system damage to have the other partitions for backup.

Over time the possibility of disk fragmentation exists. This is especially true on systems where a lot of archiving and installation and de-installation of products takes place. It is possible for files to be segmented all over the disk based on the free list. Currently the only correction for this is to back up applications and data; cold boot the system and reinstall.

3. Sticky bits

Fortune as a unix based system allows for the ability to set stick bits on programs. To see how to do this refer to the chmod command in the For:Pro manual. What this does is to allow that program or application to use contiguous swap space. This may require that a greater amount of swap space is set aside. This should, however, allow for faster initial and swap loaded of those applications.

4. Spooling

Printer spooling can also have an impact on system performance. Some improvement can be derived by attaching faster printers to the system. Example of these are the laser printers. Since the spooler finishes faster it is not a burden on the system for as long. Additionally the current spooler has default options set for priority and number of pages to save in memory. You may wish to modify those using the lpdun command. For example by default the number of pages default to 2. This is so you can stop and restart print jobs. If you really don't care it can be sent to zero.