Subject: [Q]: SPAWN fails. Why? (resource temporarily unavailable)
Posted by on Tue, 28 Mar 1995 10:46:49 GMT
View Forum Message <> Reply to Message

Hi again, I'm back with more questions!

My application is getting pretty huge now, and I've run into a problem.

The applivation is event driven, and on a given event an
RGB image = bytarr(520, 20108,3) is created.
(Yes, I know it's huge!)

When the image is created it is stored as a raw image file to disk and the memory occupied by the image is freed (to the IDL pool) by setting the variable to a scalar.

Then a new routine is called which reads the image from disk, does some manipulation on it and writes it by using the WRITE_JPEG idl routine.

The next routine to be called simply performs a SPAWN of a tar command to put all related product files (including the jpeg file) into a single tar archive file.

This works fine! The first time, that is. :-(
The second time IDL fails at the SPAWN command:

```
% At MAKE_TAR_FILE  .
%       Called from CREATE_BROWSE .
%       Called from GENERATE_CB .
%       Called from CANVAS_CB .
%       Called from B_MAINLOOP .
%       Called from XMANAGER .
%       Called from BROWSE_TASK .
%       Called from $MAIN$ .
% SPAWN: Error managing child process.
  Resource temporarily unavailable
```

HELP,/mem
and the first line of HELP gives

```
  heap memory in use:     683241, calls to MALLOC:     514084, FREE:     513879
  Code area used: 100% (372/372),  Symbol area used: 90% (452/500)
```

respectively.
Hey, I have an Indogo 2 with 192 MB memory, so it cannot be
a memory proble - or can it?
What kind of resource is IDL refering to then?

Does anyone have a clue?

I have tried the following (In increasing metaphysical order)
- Not use spawn, but use CALL_EXTERNAL to a c-library
  (Thanks for previous answers on CALL_EXTERNAL on an SGI!)
  This shows that is is the fork() that fails.
- Since fork fails, I doubled the max-limit number of concurrently
  executing processes (systune(1M)) and also the max-limit
  of processes per user.
- I tried running each routine "by hand" from the main level,
  completely bypassing XMANAGER and other event handling.
- I reorganized the program to reduce the call stack depth
- Increased code and data area by .SIZE 60000 60000
  before compiling the modules

You can see from my attempts that I'm getting desperate.
ALL suggestions are appreciated. If you think of
something I may not have tried - give me a hint.

TIA,
Frank

PS:
 Still haven't got any hints on how to store linenumbers for
 use by "HELP, calls=calls" in RESTORED routines/functions
DS