 Charles W Sandmann (sandmann@clio.rice.edu)
1206 Braelinn, Sugar Land, TX 77479

CWSDPMI was written to provide DPMI services for V2 of DJGPP. It currently does not support 16-bit DPMI applications, or DPMI applications requiring a built in extender. It does support virtual memory and hardware interrupt reflection from real mode to protected mode. DJGPP V1.1x and RSX applications will also run using this server, which can be used to provide enhanced control over hardware interrupts. Some DPMI 1.0 extensions (0x506, 0x507, 0x508) have been implemented.

CWSDPR0.EXE is an alternate version which runs at ring 0 with virtual memory disabled. It may be used if access to ring-0 features are desired. It currently does not switch stacks on HW interrupts, so some DJGPP features such as SIGINT and SIGFPE are not supported and will generate a double fault or stack fault error (to be fixed someday).

Directions for use (server can be used in either of two different ways):

1) "cwsdpmi" alone with no parameters will terminate and stay resident FOR A SINGLE DPMI PROCESS. This means it unloads itself when your DPMI application exits. This mode is useful in software which needs DPMI services, since CWSDPMI can be exec'ed and then will unload on exit.

2) "cwsdpmi -p" will terminate and stay resident until you remove it. It can be loaded into UMBs with LH. "cwsdpmi -u" will unload the TSR.

3) The file used for virtual memory swapping, if desired, is controlled by the "-sc:\cwsdpmi.swp" syntax on the command line. You must specify either a file with full disk/directory syntax, or "-s-" which disables virtual memory. The environment variables GO32TMP, TMP, and TEMP are no longer used.

4) The default swap file name is now c:\cwsdpmi.swp, but this can be changed with the CWSPARAM image, as can some other parameters.

I would like to give special thanks to DJ Delorie who wrote the original GO32 code on which CWSDPMI is based. Morten Welinder also provided and improved much of the code in this program.

This section contains a list of the error messages you might see out of CWSDPMI and some details on what they mean.

Exceptions are only handled by CWSDPMI if the application does not establish an exception handler, exceptions nest 5 deep, or the error is particularly bad:

"Page fault" -
  1) an illegal page fault happens in a RMCB or HW interrupt, (lock all pages!)
  2) all available pages have been locked,
  3) the application is using non-committed pages for null pointer protection.
"Double Fault" - multiple exceptions occured

"Invalid TSS" - typically due to RMCB or HW interrupt being called after the
   selectors/memory have been deallocated (remember to reset the mouse)
"General Protection Fault" - bad parameter sent to a DPMI call

"80386 required."

Since 80286 and lesser processors don't have the hardware necessary to
run CWSDPMI. No workaround, upgrade.

"DOS 3 required."

A few interrupts are used which need DOS 3.0 or higher. I don't expect to
ever see this message, since 80386 machines were introduced after DOS 3.0
and that check is made first.

"CWSDPMI V0.90+ (r4) Copyright (C) 1997 CW Sandmann ABSOLUTELY NO WARRANTY"

An informational message displayed if the program is not run in one-pass mode.

"Protected mode not accessible."

This message should only be displayed if running CWSDPMI in a protected
environment with no access to protected mode. In this case, DPMI should
already be available and CWSDPMI would not be needed. I would like to
know if you see this message and DPMI is not available!

"Warning: cannot open swap file c:\cwsdpmi.swp"

Maybe you are out of file handles, or the swap file name is incorrectly
specified in the image (change the name with cwsparam).

"No swap space!"

This message means you tried to use more paging file than CWSDPMI is
compiled to handle (typically 256Mb worth). Since this is protected
against in the memory allocation code, you should never see this message.

"Swap disk full!"

This means the paging file could not be expanded when trying to page
memory out to disk. This would normally not be seen, unless you are
writing output to the same disk which holds the paging file. Decrease
the amount of memory your DPMI application is using or free up disk space.

"Interrupt 0x??"

Your application tried to call an interrupt from protected mode which
normally shouldn't be called (something like a data pointer). If the
request was allowed to continue it would likely hang your machine. If you
see this message and think the interrupt should be allowed to continue, let
me know.

"Error: Using XMS switched CPU into V86 mode."

This message might be seen if you have your memory manager in AUTO mode. The
only workaround in this case is to stop using AUTO mode.

"Error: could not allocate page table memory"

The page table memory (a minimum of 16Kb) is allocated from conventional
memory (either in the 640Kb region or UMBs). If CWSDPMI cannot allocate the
minimum necessary memory, you would see this message. Free up some
conventional memory. You may also see this message if a page directory needs
to be faulted in, and there are no available pages. This means too many pages

have been locked for the allocated page tables available. While CWSDPMI tries to dynamically allocate these if needed, this effort failed. You need to increase the number of page tables with CWSPARAM, or increase the amount of free conventional memory if it is low. If the application which calls CWSDPMI internally manages all the DOS memory, the page tables may need to be pre-allocated at DPMI startup time (if this is needed, try using the run option flag 2 in cwsparam).

"16-bit DPMI unsupported."

CWSDPMI is a 32-bit only DPMI server. Ideally, on the request to enter DPMI's PM with a 16-bit request, we would just fail the call setting the carry bit like the DPMI specification describes. Some buggy 16-bit compiler tools don't check the return status and will hang the machine in this case. So, I issue an error message and exit the image instead.

"Descriptors exhausted."

An attempt to nest a DPMI client failed in the setup phase due to insufficient free selectors in the LDT.

"CWSDPMI not removed"

When the -u parameter is specified, if DPMI is not detected this message is printed. Informational.