Subject: Re: Incredible interaction between graphics and DLM: SOS!
Posted by Allan Whiteford on Tue, 05 Aug 2008 16:40:01 GMT
View Forum Message <> Reply to Message

Fabio,

IDL will be sending signals back and forward with all sorts of other stuff to handle graphics. Note that the man page for sleep says:

"sleep() may be implemented using SIGALRM; mixing calls to alarm() and sleep() is a bad idea.

Using longjmp() from a signal handler or modifying the handling of SIGALRM while sleeping will cause undefined results."

The IDL command line will have signals to deal with the graphics subsystem which can/will cause sleep(20) to return early. There may also be "longjmp"s involved.

This shouldn't affect most other things you want to do, for instance if instead of sleep(20) you called a function which happened to take 20 seconds to do something then it should not return early.

Pyro is presumably also using signals and the like to handle communication; I don't know anything about it.

Not really a solution to your problem but maybe it'll help you to look in the right place. Mess with the signal handlers at your own risk though!

Thanks,

Allan

fabio.tosetti@gmail.com wrote:
> I have a very simple file set (see below), with only one function,
> idl_test(), written just to isolate the problem.
>
> The following instruction simply sleep for 20 seconds and correctly
> returns:
>
> IDL> print, idl_test()
>
>
> The following sequence cause the idl_test() to return BEFORE the
> expected time:
>
> IDL> err = dialog_message("A dialog")
> IDL> print, idl_test()

```
>
> The idl_test() returns after about 3-4 seconds.
> The same happens if I plot a graphic before calling idl_test().
>
> Of course in my application I don't want to sleep(), but to call some
> Python code that do remote communication (Pyro), but the result is
> exactly the same, and it's very frustrating :-(...
>
> THE EXAMPLE FILES ARE:
>
> 1) idl_test.dlm:
>
> MODULE idl_test
> DESCRIPTION idl test library
> VERSION 1.0
> SOURCE LB
> BUILD_DATE AUG 05 2008
>
> FUNCTION IDL_TEST 0 0
>
> 2) idl_test.h:
>
> #ifndef IDL_TEST_H_INCLUDE
> #define IDL_TEST_H_INCLUDE
>
> extern "C"{
>  int IDL_Load(void);
> }
>
> #endif /*IDL_TEST_H_INCLUDE*/
>
>
> 2) idl_test.cpp file:
>
> #include "idl_test.h"
>
> #include <stdio.h>
> #include <string.h>
> #include <errno.h>
> #include <stdlib.h>
> #include <sys/types.h>
> #include <unistd.h>
>
> extern "C"{
>  #include "idl_export.h"
> }
>
> IDL_VPTR idl_test(int lArgc, IDL_VPTR Argv[]) {
```

```
>
>   // TEST
>   int sec = 20;
>   printf("Sleeping for %d seconds... will timeout earlier!\n", sec);
>   sleep(sec);
>   return IDL_StrToSTRING("idl_test returned before 20 seconds!!!");
>  }
>
>
>  int IDL_Load(void) {
>
>     // These tables contain information on the functions and procedures
>   // that make up the TESTMODULE DLM. The information contained in
>  these
>   // tables must be identical to that contained in testmodule.dlm.
>   //
>      static IDL_SYSFUN_DEF2 function_addr[] = {
>
>       { {(IDL_SYSRTN_GENERIC) idl_test}, "IDL_TEST", 0, 0, 0, 0},
>
>      };
>
>      // Register my routines: the routines must be specified exactly
>  the same
>      // as in .dlm.
>      return IDL_SysRtnAdd(function_addr, TRUE,
>  IDL_CARRAY_ELTS(function_addr));
>  }
>
>  4) Makefile:
>
>
>  all: idl_test.so
>
>  # ITT had the great idea of changing the name of
>  # the installation directory since idl6.3
>  # So if you have idl < 6.4 use
>  #IDLDIR = /usr/local/rsi/idl/external/include
>  # else use
>  IDLDIR = /usr/local/itt/idl/external/include
>
>  CPPFLAGS = -Wall -W -Wreturn-type -Wunused -D_GNU_SOURCE
>  MORE_INCLUDE = -I$(IDLDIR) -I/usr/include/python
>
>  %.o:%.h
>
>  .cpp.o:
>   g++ $(CPPFLAGS) $(MORE_INCLUDE) -c $< -o $@
```

```
>
> idl_test.so: idl_test.o $(IDLDIR)/idl_export.h
>   g++ $(CPPFLAGS) -shared -o idl_test.so idl_test.o -lstdc++
>
> clean:
>   rm -f *.o *.so
```