
Subject: call external, callable idl or linkimage ?

Posted by [Karl Young](#) on Tue, 11 Feb 1997 08:00:00 GMT

[View Forum Message](#) <> [Reply to Message](#)

I am trying to figure out how to write a DLL (dynamically linked library) for NT 4.0 containing a Fortran routine (built using Fortran Powerstation) that can be called from IDL. The problem is, is that I need this routine to call another IDL routine and am confused about which style of linking with IDL to use (i.e. call external, callable idl or linkimage) I can't figure out any obvious way to combine call external and callable idl to accomplish this and can't figure out how to do this using linkimage (which I guess is the obvious choice here) There doesn't seem to be any documentation or examples re. how to do this.

If anybody has done anything remotely like this and could send along some advice (or dare I hope, code fragments and compilation flags...) I would be eternally in your debt. I'm not averse to wrapping the Fortran routine in a Visual C++ wrapper if anyone knows how to do it that way. (Rewriting either the IDL or Fortran parts is not really an option as all the routines are rather large and complex)

Thanks in advance for any tips or comments,

-- KY

Subject: Re: Call External

Posted by [jbob](#) on Mon, 04 Aug 1997 07:00:00 GMT

[View Forum Message](#) <> [Reply to Message](#)

In article <33E52715.587F@xrsrv1.med.ge.com> Ken Kump <kump@xrsrv1.med.ge.com> writes:

From: Ken Kump <kump@xrsrv1.med.ge.com>
Newsgroups: comp.lang.idl-pvwave
Date: Sun, 03 Aug 1997 19:49:25 -0500
Organization: GE Medical Systems
Path: gemsw3s1.med.ge.com!not-for-mail
Lines: 13
NNTP-Posting-Host: 3.57.88.46
Mime-Version: 1.0
Content-Type: text/plain; charset=us-ascii
Content-Transfer-Encoding: 7bit
X-Mailer: Mozilla 3.01Gold (X11; I; SunOS 5.5 sun4m)
Xref: gemsw3s1.med.ge.com comp.lang.idl-pvwave:9693

Where can I find the C compilation flags for IDL 3.6 (or any version) for gcc and/or g++ for use in a call_external?

Thanks,

--

Ken Kump
GE Medical Systems
3000 Grandview Ave, W622
Waukesha, WI 53188
USA
(414) 548-4549

Hello fellow GEMS person.

I've used the attached makefile successfully to produce a "csvd.so" to be used in a CALL_EXTERNAL as:

```
foo = CALL_EXTERNAL(!xdir+"/csvd.so","csvd",c,mmax,nmax,m,n,ip,nu,nv,s,u,v)
```

This uses file "csvd.cc" as a C++ wrapper for the main code in FORTRAN file "dsa_csvd.f".

We are using fairly old SW, but it works:
SunOS 4.1.2 and its bundled loader/linker
GNU C 2.6.3
GNU libg++ 2.6.2
GNU make 3.73
Sun SPARCompiler Fortran 2.0.1
IDL Version 4.0.1

J. Bob Brown jbob@snap.med.ge.com
Consulting for GE Medical Systems
For ID only -- standard disclaimers apply.

"Of course that's just my opinion. I could be wrong."
-Dennis Miller

```
#=====
=====
#                    makefile
# 5/19/95                    11:24:44
# /home/snap2/sageidl/scm/extern_src/csvd/SCCS/s.makefile version 1.2
```

```

# created on 5/19/95 at 11:24:43
# GE Medical Fremont
# Copyright (c) 1993 by General Electric Company. All rights reserved.
#=====
=====

# The FORTRAN is setup here for Sun SPARCompiler Fortran 2.0.1 installed
# on SunOS 4.1.2. -JBob
#
# I could NOT get the GNU loader to work, even with the "-shared"
# option. So this uses the Sun loader, which seems to be adversely
# affected by the "-g" option. -JBob

CXX      = g++
CXXFLAGS = -fpic
FFLAGS   = -pic
LDFLAGS  = -L/usr/lang/SC2.0.1
LDLIBS   = -lF77 -lM77 -lpfc -lm
INSTALLDIR = ../../lib/external

# Use ".SUFFIXES" to limit implicit rules.
.SUFFIXES:
.SUFFIXES: .cc .f .o

all : csvd.so $(INSTALLDIR)/csvd.so

csvd.so : dsa_csvd.o csvd.o makefile
$(LD) -o csvd.so csvd.o dsa_csvd.o \
  /usr/lang/SC2.0.1/libF77.a \
  /usr/lang/SC2.0.1/libM77.a \
  /usr/lang/SC2.0.1/libpfc.a \
  /usr/lang/SC2.0.1/libm.a

install $(INSTALLDIR)/csvd.so : csvd.so
install csvd.so $(INSTALLDIR)

# For development, "make fnames" is an easy way to verify the FORTRAN
# function names to be used in "csvd.cc" and the "csvd" function name to
# be used in IDL. This is done by requesting assembler output and
# looking at the function names in the resulting ".s" files.

fnames :
$(FC) -S dsa_csvd.f
$(CXX) -S csvd.cc

# For development, "make dummy" is used to determine if we have included
# all of the necessary library routines by using static linking to see
# if there are any loader errors. The ".o" files are removed at the end

```

so that they won't confuse any implicit rules when making "csvd.so".

dummy : FORCE

g++ -g -c dummy.cc

g++ -g -c csvd.cc

\$(FC) -c dsa_csvd.f

\$(LD) -Bstatic \$(LDFLAGS) \$(LDLIBS) -o dummy dummy.o csvd.o dsa_csvd.o

rm *.o

clean : FORCE

rm -f *.o *.s dummy core

veryclean : FORCE

rm -f *.o *.s dummy core *.so

FORCE is used to force certain commands without regard for rules.

FORCE:

Subject: Re: Call External

Posted by [Karl Krieger](#) on Mon, 04 Aug 1997 07:00:00 GMT

[View Forum Message](#) <> [Reply to Message](#)

On Sun, 3 Aug 1997, Ken Kump wrote:

> Where can I find the C compilation flags for IDL 3.6 (or any version)
> for gcc and/or g++ for use in a call_external?

I am using

CFLAGS = -shared -fPIC

for gcc on a Sun under Solaris 2.5 (IDL4.0.1/5.0). It works for external calls using linkimage, so I guess it will be o.k. for call_external as well.

Karl Krieger

--

IPP, PO Box 1533 | Phone: +49-89-3299-1655 | E-Mail:

D-85740 Garching | FAX : +49-89-3299-1149 | krieger@ipp.mpg.de
