

---

Subject: Usage of linkimage

Posted by [oet](#) on Fri, 25 Jun 1993 15:35:02 GMT

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

---

I've just created my first shared library modul to link in  
a systime() function which allows to get GMT or Local Time  
and free formatted return-string following the rules of  
strftime(3). A second function links into IDL a simple  
strftime(3)-clone for IDL.

There is not to much documentation about linkimage in the  
IDL distribution. I've got some good stuff with examples  
for linkimage from the NSSDC-CDF-Implementation ( ncgl.gsfc.nasa.gov,  
cdf23-dist.tar.Z ).

(note: Since IDL-Rel. 3.0 there is also an internal implementation  
of CDF/NETCDF and HDF functions included).

Are there experts of linking external functions into IDL in the  
news group? I'm wondering if this is a good way to get these functions  
which we need for our real time meterological applications.

The included performance test batch gives good results, but  
I want to be sure to not go a wrong way writing own linkimage  
routines to IDL in this manner.

Any hints are gracefully welcome,

Thomas

P.S.:

To compile the library you under SunOS 4.1.2 and IDL 3.1 you need  
to uncomment the not needed line 1506 in \$IDL\_DIR/source/export.h  
otherwise there will be a fatal error.

| Thomas Oettli                  Internet: Thomas.Oettli@sma.ch |  
| Swiss Meteorological Institute      thomas.oettli@active.ch |  
| Kraehbuehlstr. 58                CServe: 100015.3543@compuserve.com |  
| 8044 Zuerich                      |

```
/* ----- cut here ----- */  
/* %Z% File         : %M%  
** %Z% Version     : %I%  
** %Z% Short description: SRDB time function library  
** %Z% Project     : SRDB Short Time Range Database  
** %Z% Author      : Thomas.Oettli@sma.ch  
** %Z% Created on   : 93/06/22  
** %Z% Last modified by : %Q%
```

```

** %Z% Last modified on : %E%
** %Z% Status      : testing
**
** -----
** Modification History :
**
** Notes: Compile syntax:
**
**      (Tested on SunOS 4.1.2 )
**
**      cc -I$(IDL_DIR)/source -c -pic -fsingle idl_timelib.c
**      ld -o idl_timelib.so -assert pure-text idl_timelib.o
**
** -----
**
** IDL linkimage syntax:
** (Put this line into your $IDL_STARTUP file)
**
**      linkimage, 'systime2','idl_timelib.so',1,max_args=2
**      linkimage, 'strftime2','idl_timelib.so',1,min_args=2,max_args=2
**
**
** IDL help:
** If you want to have included this functions in your
** library help file, put the dummy procedures in this
** directory into your library directory.
** Then create a new help file:
**
**      IDL> mk_library_help, '<your library directory>', !DIR+='/help/<mylib>.help'
**
**      Startup the IDL Online Help '?' and check if the functions are
**      available under your library topic.
**
***** ****

```

```

#include <stdio.h>
#include <time.h>
#include <export.h>

```

```

VPTR systime2(argc, argv, argk)
int argc;
VPTR argv[];
char *argk;
{
    ALLTYPES temp;
    time_t now=time(NULL);
    struct tm *tp;
    char ts[32];

```

```

static VARIABLE rcode;

if ( argc == 1 && argv[0]->value.i == 1 ) {
    temp.d=(double)now;
    store_scalar((VPTR) &rcode, TYP_DOUBLE, &temp);
    return((VPTR) &rcode);
}

if ( argc == 0 || argv[0]->value.i == 0 )
    tp=localtime(&now);
else
    tp=gmtime(&now);

if ( argc <= 1 ) {
    strftime(ts,32,"%a %h %d %H:%M:%S %Y",tp);
} else {
    strftime(ts,32,STRING_STR(&argv[1]->value.str),tp);
}
return (ret_str_as_STRING(ts));
}

```

```

VPTR strftime2(argc, argv, argk)
int argc;
VPTR argv[];
char *argk;
{
    struct tm *tp;
    char ts[32];
    long t_l;

    t_l=(long)argv[0]->value.d;
    tp=localtime(&t_l);
    strftime(ts,32,STRING_STR(&argv[1]->value.str),tp);

    return (ret_str_as_STRING(ts));
}

```

/\* ----- cut here ----- \*/

```

; %Z% %M% Rel. %I%, by oet@sma.ch last modified on %E% by %Q%
;
;+
; NAME:
;     SYSTIME2
;
; PURPOSE:
;
```

; The SYSTIME2 function returns the current system time  
; as either a string that contains the current day, date and  
; time, or as the number of seconds elapsed since January 1,  
; 1970. The difference to the IDL internal systime() is that  
; systime2 allows the use of a time format string following the  
; rules for the C function strftime as second argument.  
;  
; CATEGORY:  
; Date/Time  
;  
; CALLING SEQUENCE:  
; Result = SYSTIME2(Arg [,<fmt>])  
;  
; Arguments:  
; If Arg is present and nonzero, the number of  
; seconds elapsed since January 1, 1970 is returned as  
; a double-precision, floating-point value.  
;  
; Otherwise, the default scalar string containing the cur-  
; rent date/time in standard 24-character system for-  
; mat is returned. This format is:  
;  
; DOW MON DD HH:MM:SS YEAR  
;  
; where DOW is the day of the week, MON is the  
; month, DD is the day of the month, HH is the hour, MM  
; is the minute, SS is the second, and YEAR is the  
; year.  
;  
; If a format string is defined as second argument  
; systime2 works as follows:  
;  
; systime2(0,<fmt>) : scalar string formatted output  
; of localtime  
;  
; systime2(1,<fmt>) : scalar string formatted output  
; of gmtime  
;  
; The rules for the format string follows the rules for  
; the C function strftime(3):  
;  
;  
; <fmt> is a character string that consists of field  
; descriptors and text characters, reminiscent of printf(3V).  
; Each field descriptor consists of a % character followd  
; by another character that specifies the replacement for the  
; field descriptor. All other characters  
; are copied from fmt into the result. The following field

; descriptors are supported:

;     %% same as %

;     %a day of week, using locale's abbreviated weekday names

;     %A day of week, using locale's full weekday names

;     %b

;     %h month, using locale's abbreviated month names

;     %B month, using locale's full month names

;     %c date and time as %x %X

;     %C date and time, in locale's long-format date and time representation

;     %d day of month (01-31)

;     %D date as %m/%d/%y

;     %e day of month (1-31; single digits are preceded by a blank)

;     %H hour (00-23)

;     %I hour (00-12)

;     %j day number of year (001-366)

;     %k hour (0-23; single digits are preceded by a blank)

;     %l hour (1-12; single digits are preceded by a blank)

;     %m month number (01-12)

;     %M minute (00-59)

;     %n same as \n

;     %p locale's equivalent of AM or PM, whichever is appropriate

;     %r time as %I:%M:%S %p

;     %R time as %H:%M

```
; %S seconds (00-59)
;
; %t same as \t
;
; %T time as %H:%M:%S
;
; %U week number of year (01-52), Sunday is the first
;     day of the week
;
; %w day of week; Sunday is day 0
;
; %W week number of year (01-52), Monday is the first
;     day of the week
;
; %x date, using locale's date format
;
; %X time, using locale's time format
;
; %y year within century (00-99)
;
; %Y year, including century (for example, 1988)
;
; %Z time zone abbreviation
;
; The difference between %U and %W lies in which day is
; counted as the first day of the week. Week number 01 is the
; first week with four or more January days in it.
;
; The time zone for the localtime can be changed by setting
; the environment variable TZ to another available value
; from /usr/share/lib/zoneinfo (SunOS 4.x), for example
; if you define
;
;     setenv TZ Hongkong
;
; before starting IDL will, the local time showed with
;
;     print, systime2(0,"%a %h %d %H:%M:%S %Y")
;
; will you provide with a formatted string of the actual
; local time in Hongkong. (Unfortunately using setenv, 'TZ=Hongkong'
; from within IDL doesn't have any effect for this function).
;
; Examples
;
; String date GMT
```

```
; IDL> print, systime2(1,'%d %m %Y %H:%M')
; 25 06 1993 12:09
;
; Day of year:
;
; IDL> print, systime2(0,'%j')
; 176
;
; WMO header date YYGGgg  GMT
;
; IDL> print, systime2(1,'%d%H%M')
; 251214
;
; RESTRICTIONS:
; Tested on SunOS 4.1.2 only
;
; MODIFICATION HISTORY:
; Written by: Thomas.Oettli@sma.ch, 22-june-1993.
;
;
;
;
; dummy procedure for linkimage routine to generate help file entry with
; mk_library_help
;
; ----- CUT HERE -----
;
; %Z% %M% Rel. %I%, by oet@sma.ch last modified on %E% by %Q%
;
;+
; NAME:
;     STRFTIME2
;
; PURPOSE:
;     strftime2(<sec>,<fmt>) returns a time value <sec> -
;     (double)seconds since 1-jan-1970 converted to a character
;     string in a format specified by <fmt>.
;
;     <fmt> is a character string that consists of field
;     descriptors and text characters, reminiscent of the C
;     function printf(3V). Each field descriptor consists of a %
;     character followed by another character that specifies the
;     replacement for the field descriptor. All other characters
;     are copied from fmt into the result. The following field
;     descriptors are supported:
;
;
; CATEGORY:
```

```
; Date/Time  
;  
; CALLING SEQUENCE:  
;   Result = SYSTIME2(Arg [,<fmt>])  
;  
; Arguments:  
;   If Arg is present and nonzero, the number of  
;   seconds elapsed since January 1, 1970 is returned as  
;   a double-precision, floating-point value.  
;  
;   Otherwise, the default scalar string containing the cur-  
;   rent date/time in standard 24-character system for-  
;   mat is returned. This format is:  
;  
;   DOW MON DD HH:MM:SS YEAR  
;  
; where DOW is the day of the week, MON is the  
; month, DD is the day of the month, HH is the hour, MM  
; is the minute, SS is the second, and YEAR is the  
; year.  
;  
; If a format string is defined as second argument  
; systime2 works as follows:  
;  
; systime2(0,<fmt>) : scalar string formatted output  
;                      of localtime  
;  
; systime2(1,<fmt>) : scalar string formatted output  
;                      of gmtime  
;  
; The rules for the format string follows the rules for  
; the C function strftime(3):  
;  
;  
; <fmt> is a character string that consists of field  
; descriptors and text characters, reminiscent of printf(3V).  
; Each field descriptor consists of a % character followd  
; by another character that specifies the replacement for the  
; field descriptor. All other characters  
; are copied from fmt into the result. The following field  
; descriptors are supported:  
;  
; %% same as %  
;  
; %a day of week, using locale's abbreviated weekday  
;      names  
;  
; %A day of week, using locale's full weekday names
```

```
; %b
; %h month, using locale's abbreviated month names
;
; %B month, using locale's full month names
;
; %c date and time as %x %X
;
; %C date and time, in locale's long-format date and
;   time representation
;
; %d day of month (01-31)
;
; %D date as %m/%d/%y
;
; %e day of month (1-31; single digits are preceded by
;   a blank)
;
; %H hour (00-23)
;
; %I hour (00-12)
;
; %j day number of year (001-366)
;
; %k hour (0-23; single digits are preceded by a blank)
;
; %l hour (1-12; single digits are preceded by a blank)
;
; %m month number (01-12)
;
; %M minute (00-59)
;
; %n same as \n
;
; %p locale's equivalent of AM or PM, whichever is
;   appropriate
;
; %r time as %l:%M:%S %p
;
; %R time as %H:%M
;
; %S seconds (00-59)
;
; %t same as \t
;
; %T time as %H:%M:%S
;
; %U week number of year (01-52), Sunday is the first
```

```
; day of the week
;
; %w  day of week; Sunday is day 0
;
; %W  week number of year (01-52), Monday is the first
;      day of the week
;
; %x  date, using locale's date format
;
; %X  time, using locale's time format
;
; %y  year within century (00-99)
;
; %Y  year, including century (for example, 1988)
;
; %Z  time zone abbreviation
;
; The difference between %U and %W lies in which day is
; counted as the first day of the week. Week number 01 is the
; first week with four or more January days in it.
;
; The time zone for the localtime can be changed by setting
; the environment variable TZ to another available value
; from /usr/share/lib/zoneinfo (SunOS 4.x), for example
; if you define
;
;      setenv TZ Hongkong
;
; before starting IDL will, the local time showed with
;
;      print, systime2(0,"%a %h %d %H:%M:%S %Y")
;
; will you provide with a formatted string of the actual
; local time in Hongkong.
;
; Examples
;
; RESTRICTIONS:
; Tested on SunOS 4.1.2 only
;
; MODIFICATION HISTORY:
; Written by: Thomas.Oettli@sma.ch 22-June 1993
;
;
;
;----- CUT HERE -----
```

```

; Performance test batch for the idl_timelib.so functions
; systime2() and strftime2()
;
linkimage, 'systime2','idl_timelib.so',1,max_args=2
linkimage, 'strftime2','idl_timelib.so',1,min_args=1,max_args=2

; Test batch for the systime2 function in the shared library
; archive idl_timelib.so in comparision with the IDL internal
; routine systime

print, *****
print, '*          SYSTIME2 - TEST           *'
print, *****

print, *****
print, '* Test for 10000 calls to the idl internal systime(0) function *'
print, *****

help, /mem
t=systime(1)
for i=0,10000 do tvar=systime(0)
print, 'time required in seconds: ', systime(1) - t
help, /mem
print, ""

print, *****
print, '* Test for 10000 calls to the idl internal systime(1) function *'
print, *****

help, /mem
t=systime(1)
for i=0,10000 do tvar=systime(1)
print, 'time required in seconds: ', systime(1) - t
help, /mem
print, ""

print, *****
print, '* Test for 10000 calls to the idl internal systime(0) function *'
print, '* with a string concatenating to get an output in the form of   *'
print, '*          dd mmm yyyy           *'
print, '* using                         *'
print, '* tvar=SYSTIME(0)               *'
print, '* tvar=STRMID(tvar,8,3) + STRMID(tvar,4,4) + STRMID(tvar,20,40)*'
print, *****

help, /mem
t=systime(1)

```

```

for i=0,10000 do $
    tvar=SYSTIME(0) & $
    tvar=STRMID(tvar,8,3) + STRMID(tvar,4,4) + STRMID(tvar,20,40)
print, 'time required in seconds: ', systime(1) - t
help, /mem
print, ""

print, '*' * * * * * * * * * * * * * * * * * * * * * * * *
print, ""

print, '*****'
print, '* Test for 10000 calls to the shared lib systime2(0) function *'
print, '*****'
help, /mem
t=systime(1)
for i=0,10000 do tvar=systime2(0)
print, 'time required in seconds: ', systime(1) - t
help, /mem
print, ""

print, '*****'
print, '* Test for 10000 calls to the shared lib systime2(1) function *'
print, '*****'
help, /mem
t=systime(1)
for i=0,10000 do tvar=systime2(1)
print, 'time required in seconds: ', systime(1) - t
help, /mem

print, ""

print, '*****'
print, '* Test for 10000 calls to the shared lib systime2 function      *'
print, '* with a format string the get the output in the format          *'
print, '*           dd mmm yyyy           *'
print, '* using                      *'
print, '* tvar=SYSTIME2(0,'%d %h %Y')           *'
print, '*****'

help, /mem
t=systime(1)
for i=0,10000 do tvar=systime2(0,'%d %h %Y')
print, 'time required in seconds: ', systime(1) - t
help, /mem

```

```
print, "
print, *****
print, "
print, *****
print, *          STRFTIME2 - TEST          *
print, *****
help, /mem
t=systime(1)
for i=0,10000 do tvar=strftime2(t,"%a %h %d %H:%M:%S %Y")
help, /mem
print, 'time required in seconds: ', systime(1) - t

print, *****
```

---