Subject: date question
Posted by Robert Moss on Tue, 24 Mar 1998 08:00:00 GMT
View Forum Message <> Reply to Message

I need to be able to convert a semi-standard date string, such as
 3/24/98 14:22:10.120
into a value that is equal to the number of seconds since 00:00 Sunday
the week of the date in question.

For example, 3/24/98 happens to be a Tuesday, so i need

secs since Sun 00:00 = 10.120 + (22 * 60) + (14 * 60 * 60) +
                (2 * 24 * 60 * 60)

where here (2 * 24 * 60 * 60) is the number of seconds between
00:00 Sunday and 00:00 Tuesday.

Of course, I'd like a general solution that is intelligent enough to
know what day of the week it is based on the date alone...

Anyone else out there have a "gps_time_week" function handy?

--
Robert M. Moss, Ph.D. - mossrm@texaco.com - FAX (713)954-6911
 ---------- -----

Subject: Re: date question
Posted by Richard D. Hunt on Mon, 30 Mar 1998 08:00:00 GMT
View Forum Message <> Reply to Message

Robert Moss wrote:
>
> I need to be able to convert a semi-standard date string, such as
> 3/24/98 14:22:10.120
> into a value that is equal to the number of seconds since 00:00 Sunday
> the week of the date in question.
>
> For example, 3/24/98 happens to be a Tuesday, so i need
>
> secs since Sun 00:00 = 10.120 + (22 * 60) + (14 * 60 * 60) +
>   (2 * 24 * 60 * 60)
>
> where here (2 * 24 * 60 * 60) is the number of seconds between
> 00:00 Sunday and 00:00 Tuesday.
>

> Of course, I'd like a general solution that is intelligent enough to
> know what day of the week it is based on the date alone...
>
> Anyone else out there have a "gps_time_week" function handy?
>
> --
> Robert M. Moss, Ph.D. - mossrm@texaco.com - FAX (713)954-6911
> ---------- -----
> This does not necessarily reflect the opinions of Texaco Inc.

I have created a modelling an simulation package to model certain sensors on the GPS satellite so I have written a number of routines including time routines to calculate their postion. The following routines will convert from GMT to theGPS time which is the number of seconds since the epoch (01/06/1980 00:00:00) and even allows you to adjust for the GPS leap seconds which is a time correction they update every now and then (currently +12 seconds). From these routines you can modulo the number of seconds in the week to get the wee number and seconds in the week. Check to make sure none of the lines of code get wrapped by the mailer.

Rich

```
;+
;
; NAME:
;     GMTTOGPS
;
; PURPOSE:
;     Converts GMT to GPS time
;
; CALLING SEQUENCE:
;     gpstime = GMTTOGPS(gmttime)
;
; INPUTS:
;     gmttime = etime structure
;     leap_seconds = GPS leap seconds (default=0)
;
; KEYWORD PARAMETERS:
;
; OUTPUTS:
;     gpstime = time in seconds since EPOCH
```

```
;
; COMMON BLOCKS:
;
; NOTES:
;     The definition of a time structure is as follows:
;        ETime structure {
;           double   esec       ; seconds: 0.0-60.0
;           int      emin       ; min: 0-59
;           int      ehour      ; hours: 0-23
;           int      eday       ; days: 1-31
;           int      emonth     ; month: 1-12
;           int      eyear      ; year: 0-9999
;        }
;
; AUTHOR:
;     Richard Hunt
;
; MODIFICATION HISTORY:
;     4-10-96     Created.
;-
;
FUNCTION GMTToGPS, gmttime, leap_seconds=leap_seconds

   ; Set up some constants
   if (N_Elements(leap_seconds) EQ 0) then leap_seconds=0

   ; GPS epoch for ztime
   zepoch = {esec:0.0D0, emin:0, ehour:0, eday:6, emonth:1, eyear:1980}

   return, (TimDSec(gmttime, zepoch) + leap_seconds)
END




;+
;
; NAME:
;     GPSTOGMT
;
; PURPOSE:
;     Converts GPS to GMT time
;
; CALLING SEQUENCE:
;     gmttime = GPSTOGMT(gpstime)
;
; INPUTS:
```

```
;     gpstime = time in seconds since EPOCH
;     leap_seconds = GPS leap seconds (default=0)
;
; KEYWORD PARAMETERS:
;
; OUTPUTS:
;     gmttime = etime structure
;
; COMMON BLOCKS:
;
; NOTES:
;     The definition of a time structure is as follows:
;       ETime structure {
;         double   esec       ; seconds: 0.0-60.0
;         int      emin       ; min: 0-59
;         int      ehour      ; hours: 0-23
;         int      eday       ; days: 1-31
;         int      emonth     ; month: 1-12
;         int      eyear      ; year: 0-9999
;       }
;
; AUTHOR:
;     Richard Hunt
;
; MODIFICATION HISTORY:
;     4-10-96     Created.
;-
;
FUNCTION GPSToGMT, gpstime, leap_seconds=leap_seconds

   ; Set up some constants
   if (n_elements(leap_seconds) EQ 0) then leap_seconds=0

   ; GPS epoch for ztime
   gmttime = {esec:0.0D0, emin:0, ehour:0, eday:6, emonth:1, eyear:1980}

   ; Add GPS time to EPOCH time.
   TimASec, gmttime, (gpstime - leap_seconds)

   Return, gmttime
END




;+
;
; NAME:
```

```
;     TIMDSEC
;
; PURPOSE:
;     Return difference of two etime structures in seconds
;
; CALLING SEQUENCE:
;     sec = TIMDSEC(etime1, etime2)
;
; INPUTS:
;     etime1 = etime structure
;     etime2 = etime structure
;
; KEYWORD PARAMETERS:
;
; OUTPUTS:
;     sec = etime1 - etime2
;
; COMMON BLOCKS:
;
; NOTES:
;     The definition of a time structure is as follows:
;       ETime structure {
;         double   esec       ; seconds: 0.0-60.0
;         int      emin       ; min: 0-59
;         int      ehour      ; hours: 0-23
;         int      eday       ; days: 1-31
;         int      emonth     ; month: 1-12
;         int      eyear      ; year: 0-9999
;       }
;
; AUTHOR:
;     Richard Hunt
;
; MODIFICATION HISTORY:
;     4-10-96     Created.
;-
;
FUNCTION TimDSec, etime1, etime2

   ; Find the difference between the the Julian dates and convert that
to
   ; seconds. Can't convert the individual Julian dates to seconds
because
   ; of roundoff errors when the numbers get really big.
   d1 = JulDay(etime1.emonth, etime1.eday, etime1.eyear)
   d2 = JulDay(etime2.emonth, etime2.eday, etime2.eyear)
   js = (d1 - d2) * 86400.0D
```

```
   ; Convert the times to seconds.
   s1 = etime1.ehour*3600.0D + etime1.emin*60.0D  + etime1.esec
   s2 = etime2.ehour*3600.0D + etime2.emin*60.0D  + etime2.esec

   ; Return the difference
   Return, (js + s1 - s2)
END




;+
;
; NAME:
;     TIMASEC
;
; PURPOSE:
;     Add seconds to etime structure
;
; CALLING SEQUENCE:
;     TIMASEC, etime, sec
;
; INPUTS:
;     etime = etime structure
;     sec = seconds to be added
;
; KEYWORD PARAMETERS:
;
; OUTPUTS:
;     etime = etime structure with seconds added
;
; COMMON BLOCKS:
;
; NOTES:
;     The definition of a time structure is as follows:
;        ETime structure {
;           double   esec       ; seconds: 0.0-60.0
;           int      emin       ; min: 0-59
;           int      ehour      ; hours: 0-23
;           int      eday       ; days: 1-31
;           int      emonth     ; month: 1-12
;           int      eyear      ; year: 0-9999
;        }
;
; AUTHOR:
;     Richard Hunt
;
; MODIFICATION HISTORY:
```

```
;     4-10-96     Created.
;-
;
PRO TimASec, t, sec

   ; Convert the time portion of the etime structure to seconds in the
day.
   daysecs = t.ehour*3600.0D + t.emin*60.0D  + t.esec

   ; Add the seconds to increment by.
   daysecs = daysecs + sec

   ; Find out how many days are in this and get the remainder of
seconds.
   days = Long(daysecs / 86400.0D)
   remainder = daysecs - (days * 86400.0D)

   ; Convert the date to a Julian date.
   jd = JulDay(t.emonth, t.eday, t.eyear)

   ; Add the days to the Julian date.
   jd = jd + days

   ; Adjust if the remainder is less than 0.
   IF remainder LT 0.0 THEN BEGIN
      remainder = remainder + 86400.0D
      jd = jd - 1
   ENDIF

   ; Convert the Julian date back to a date.
   ;JD2YMD, jd, year, month, day
   CalDat, jd, month, day, year

   ; Convert the remainder of seconds to a time.
   hour = Long(remainder/3600)
   remainder = remainder - 3600*hour
   minutes = Long(remainder/60)
   remainder = remainder - 60*minutes
   seconds = remainder

   t.eyear=year
   t.emonth=month
   t.eday=day
   t.ehour=hour
   t.emin=minutes
   t.esec=seconds
END
```

--

Richard D. Hunt
SANDIA NATIONAL LABORATORIES
P.O. Box 5800 M/S 0965
Albuquerque, NM 87185-0965
Voice: (505)844-3193
Fax: (505)844-5993
E-Mail: rdhunt@sandia.gov