Programma di Controllo dell'Antenna di Noto

Ver. 3.0

G. Tuccari, S. Buttaccio

Rapporto Tecnico IRA 254/98

Il programma di guida dell'antenna di Noto e` alla terza versione. Abbiamo sentito l'utilita` di fissare in un nuovo documento le caratteristiche del sistema e le variazioni introdotte negli anni trascorsi dalla versione 2, ovvero dal 1992.

La prima versione del programma e` stata sviluppata nel 1989, e pertanto il sistema si avvicina a dieci anni di vita. Anni in cui e` stato intensamente utilizzato e in cui sono stati resi evidenti le caratteristiche e le carenze. Tra le funzioni mancanti e` stata evidente la impossibilita` di comandare posizioni fisse di cielo, interagendo con il Field System. Questa funzione puo` risultare utile e in fase osservativa e in fase di controllo strumentale, qualora si voglia far stazionare l'antenna per un periodo determinato di tempo in una posizione fissa. A tale scopo e` stata inserita la funzione AZEL.

Inoltre e` stata implementata una funzione di scansione in azimut, che consente di far muovere a velocita` desiderata (ovviamente entro i limiti consentiti dall'antenna) la parabola, attorno ad una posizione puntata e per un offset fissato. Tale funzione viene attivata da SOURCE=SCANAZ.

La funzione TRACK fornisce informazioni sul puntamento reale, ovvero alla data e orario indicato, restituisce le posizioni comandate e reali.

In questo documento si presentano le caratteristiche generali del programma, e si sottolineano le variazioni apportate in questa versione.

1. Descrizione del Programma di Tracking

La struttura del programma non e` stata modificata rispetto alla precedente versione e mantiene pertanto la stessa organizzazione, per cui ogni operazione e` schedulata e viene eseguita in intervalli di tempo prestabiliti. Una modifica apportata riguarda piuttosto l'hardware, ovvero il computer su cui il programma viene eseguito. Si e` passati da un 80286 al 80486, che si presenta piu` che sufficiente ad eseguire i compiti che gli sono affidati. Il sistema operativo e` ancora il DOS, e l'interazione con l'esterno e` ottimizzata per avvenire attraverso il Field System. Rimangono comunque possibili operazioni manuali, quali il puntamento di una sorgente o il movimento manuale dell'antenna, impostazione di offset, etc.

All'avvio del programma deve essere controllata l'esattezza della data che eventualmente va corretta. Il menu` di partenza e` autoesplicativo e qualora non venga dato alcun comando entro 30 secondi, il programma esce in DOS. Seguiremo l'opzione del tracking (T); il sistema a tal punto legge il rubidium repeater ed imposta l'orario che incrementera` ad ogni secondo al transito del 1PPS di stazione. Da quel momento il computer di puntamento e` sincronizzato al maser ed e` in grado di rimanere indefinitamente in tale stato fino all'uscita dal menu` di tracking.

Il subreflector viene puntato ogni 10 secondi, in funzione dei parametri presenti sul file di inizializzazione tp, che riporta i parametri di puntamento, il ricevitore a cui si riferiscono e i parametri suddetti. Viene utilizzato un fit polinomiale per il settaggio degli assi Z1, Z2 e Z3, mentre X e Y vengono comandati a posizioni fisse, ma dipendenti dal ricevitore.

Nell'elenco che segue vengoro riportate le funzioni svolte nei decimi di secondo indicati, per ogni secondo di esecuzione del programma.

Inizializzazione delle variabili	(iniz,main)
lettura parametri di puntamento e correzione per la rifrazione	(iniz)
controllo presenza stampante	(printer)
prima sincronizzazione con lettura rubidium-repeater e 1PPS	(rubid)
stampa su monitor della maschera	(mask)
decimo 0 puntamento	(ever)
visualizzazione ora e nome della sorgente	(utname)
controllo tastiera	(keyb)
controllo varazioni nel tracking	(joy)
decimo 1 puntamento	(ever)
controllo stato antenna	(status)
calcolo e invio posizione subreflector (ogni 10 sec)	(scuwrite)

decimo 2 puntamento	(ever)
lettura da ACU posizione comandata	(readc)
decimo 3 puntamento	(ever)
lettura da ACU posizione reale	(readp)
controllo quadrante antenna e on-source	(main)
decimo 4 puntamento	(ever)
conversione coordinate attuali in equatoriali	(equ)
decimo 5 puntamento	(ever)
lettura da computer remoto richiesta di puntamento	(hpread)
decimo 6 puntamento	(ever)
visualizzazione informazioni parte 1	(show1)
decimo 7 puntamento	(ever)
aggiornamento grafico	(graph)
decimo 8 puntamento	(ever)
invio informazioni a computer remoto	(hpwrite)
decimo 9 puntamento	(ever)
visualizzazione informazioni parte 2	(show2)
risincronizzazione con 1PPS	(sync)

La routine ever, che viene eseguita durante il puntamento, utilizza altre subroutines per il calcolo, la conversione e l'avvio all'acu delle posizioni da puntare (midn, rifra, altaz, x); inoltre ever controlla le eventuali modifiche al tracking tramite joy.

2. Variazioni introdotte rispetto alle Ver. 2

Le variazioni introdotte nel programma intersessano i seguenti punti:

a) implementazione su computer 486 con clock di sistema 40 Mhz;

b) modifica del file dei parametri di puntamento;

c) implementazione del comando per la scansione nel tracking;

d) implementazione del comando di puntamento di una posizione fissa;

e) implementazione del comando TRACK di Field System.

Il punto a) riguarda l'hardware che viene utilizzato per comandare l'antenna. Un computer veloce semplifica la gestione dei tempi del sistema, che a tutti gli effetti, entro il decimo di secondo e` un sistema real-time. Cio` significa che piu` funzioni possono essere realizzate, anche se ancora non implementate. Il computer da usare per la versione 3 del programma e` cosi` caratterizzato:

processore 80486 con clock 40 MHz

2 porte seriali RS232

1 switch seriale a 4 vie + 1 porta per 1PPS

1 porta joystick digitale per la cloche di puntamento manuale.

Le linee seriali sono entrambe operanti a 9600 baud, dati a 8 bit, parita` pari, 1 bit di stop. La porta 1 e` connessa direttamente alla seriale dell'ACU. La porta due e` connessa attraverso lo switch seriale ai dispositivi:

porta 2_1 PC di Field System

porta 2_2 SCU

porta 2_3 Rubidium repeater

porta 2_4 non usata

La scheda di controllo della seriale e` adibita anche alla sincronizzazione con il maser del clock del PC di guida.

Un ritardo hardware e` utilizzato nelle routine readp, readc e status ed e` caratterizzato dalla label 123. Tale ritardo e` dipendente dalla velocita` di calcolo del computer usato. Il parametro da ottimizzare e` il numero di cicli da eseguire della operazione dummy di label 123. Il ritardo deve essere grande abbastanza da ottenere il completo trasferimento delle informazioni da parte dell'antenna, ma non tanto da scavalcare il decimo.

La modifica del punto b) riguarda l'automatizzazione del posizionamento del subriflettore. Viene ancora utilizzato il file TP che contiene:

otto parametri di puntamento

la definizione del ricevitore relativo

offset di Z1, offset di Z2, offset di Z3, costante di proporzionalita` del primo ordine, costante di proporzionalita` del secondo ordine, costante di proporzionalita` del terzo ordine, offset di X, offset di Y.

La scansione del tracking in azimut, punto c), viene realizzata attraverso la decodifica nel decimo 5 del nome di sorgente "SCANAZ". Il comando di FS per realizzarla e` il seguente:

SOURCE=SCANAZ,DDPPSS,+(-)ddppss,-1

ove DDPPSS e` la velocita` di scansione in gradi, primi, secondi al secondo

e ddppss e` l'offset finale in gradi, primi, secondi per cui la scansione si ferma.

Un qualunque puntamento di sorgente blocca la scansione ed azzera gli offset.

Il punto d) descrive il puntamento in posizione fissa, ovvero la possibilita` di inviare l'antenna in una posizione di cielo desiderata, disabilitando il tracking stesso. La sintassi e`:

SOURCE=AZEL,az,el

con az, el espressi in gradi.

Il punto e) consiste nella possibilta` di ricavare tramite FS le condizioni di puntamento dell'antenna secondo la seguente definizione:

TRACK=YY/DDD.HH:MM:SS.d AZP ELP AZC ELC

con YY anno

DDD giorno dell'anno

HH:MM:SS ora, minuti, secondi a cui le posizioni si riferiscono

AZP,ELP azimut ed elevazione puntate

AZC,ELC azimut ed elevazione comandate.

3. Variazioni manuali al tracking

Per effettuare variazioni al tracking si deve selezionare il tasto F4, cosi` da potere successivamente utilizzare i tasti funzione da F7 a F10 che controllano la direzione, o in maniera piu` semplice attraverso la cloche che consente il movimento diagonale. La velocita` di scan puo` essere impostata come "slow" a 30'/min o "fast" come 60'/min. Con il tasto F5 si puo` variare il modo di scansione tra AR-DEC e AZ-EL.

I tasti F11 e F12 introducono ed eliminano relativamente un offset di circa 3 primi in AR, utile qualora si voglia rapidamente muoversi attorno ad un punto.

Gli offset introdotti manualmente o attraverso il field system possono essere annullati rapidamente atrraverso la digitazione di CNTL-O e CNTL-P relativamente ai modi AR-DEC o AZ-EL.

```
      PRO.FOR
      logical s
      character kk
      character*15 tr,oldname
      character*31 prp
      integer prflag
      dimension ii(10)
$include:'comm.for'
      data prp /'COMMAND POSITION MODE EXITED "'/
      call screen(7,1,80,0,a1)
c        call printer(prflag)
c        if(prflag.eq.1) then
c                open(3,file='prn')
c        else
           open(3,file='nul')
c        endif
c        open(5,file='d:tem')
c       open(9,file='schedule')
c       call iopcom(1,8,2,8,1,0,0,0,0,0,a1)
      call iniz
      ra=0.0
      de=90.0*drad
      name='POLAR STAR'
52      call cls
      call putcur(10,25,0,a1)
      write(tr,'(a12)') 'T - TRACKING'
      call wchars(0,tr,12,14,0,1,a1)
      call putcur(16,25,0,a1)
      write(tr,'(a15)') 'E - EXIT TO DOS'
      call wchars(0,tr,15,14,0,1,a1)
      call putcur(14,25,0,a1)
      write(tr,'(a9)') 'S - SETUP'
      call wchars(0,tr,9,14,0,1,a1)
      call putcur(12,25,0,a1)
      write(tr,'(a12)') 'A - AUTOSTOW'
      call wchars(0,tr,12,14,0,1,a1)
      call putcur(18,25,0,a1)
      write(tr,'(a13)') 'C - CHANGE TP'
      call wchars(0,tr,13,14,0,1,a1)
      kk=' '
      c30=0
53      call inkey(kk,h,c)
      c30=c30+0.01
      if((kk.eq.'e').or.(kk.eq.'E').or.(c30.ge.10000.0)) then
            call clscom(1,a1)
            call clscom(2,a1)
            close(3)
            stop' '
      endif
      if((kk.eq.'t').or.(kk.eq.'T')) goto 66
      if((kk.eq.'s').or.(kk.eq.'S')) then
            call iopcom(1,8,2,8,1,0,0,0,0,0,a1)
            call setup (*52)
            call clscom(1,a1)
      endif
      if((kk.eq.'a').or.(kk.eq.'A')) then
            call iopcom(1,8,2,8,1,0,0,0,0,0,a1)
            call astow (*52)
            call clscom(1,a1)
      endif
      if((kk.eq.'c').or.(kk.eq.'C')) call parm (*52)
      goto 53
66      call cls
      call iopcom(1,8,2,8,1,0,0,0,0,0,a1)
      call iopcom(2,8,2,8,1,0,0,0,0,0,a1)
c        call putcur(10,10,0,a1)
c        write(*,*) '    DO YOU WANT TO USE WITH FIELD SYSTEM ? [y/n]'
```

```
c67       call inkey(kk,h,c)
c         if((kk.ne.'y').and.(kk.ne.'n')) goto 67
c         if(kk.eq.'y') hh=.true.
c         if(kk.eq.'n') hh=.false.
c         kk=' '
c         call cls
c         call putcur(10,10,0,a1)
c         write(*,*) '    DO YOU WANT TO USE DIRECTION CONTROL ? [y/n]'
c57       call inkey(kk,h,c)
c         if((kk.ne.'y').and.(kk.ne.'n')) goto 57
c         if(kk.eq.'y') corr=.true.
c         if(kk.eq.'n') corr=.false.
c         call cls
      hh=.true.
      corr=.false.
      fl=0
      jjj=0
      call sync
c         call clscom(2,a1)
c         call iopcom(2,8,1,7,1,0,0,0,0,0,a1)
      call mask
      goto 1
1000    jjj=jjj+1
      if (jjj.ne.2) goto 1
      write(3,7) iday,ihr,imin,isec,'PC<-->ACU COMMUNICATION ERROR "'
1       iosec=isec
      call gettim(ihr,imin,isec,i100)
      i10th=int(i100/10)
      if(i10th.eq.0) then
c                write(5,*) i10th
1111            if(ii(1).eq.0) then
c                       write(5,*) i10th
                  call utname
                  call ever
      if (name.eq."autostow") then
            ra=0.0
            de=90.0*drad
            name='POLAR STAR'
            call astow (*52)
      endif
c         if (recev.eq." 1.3 cm") then
      if (isec.eq.0.or.isec.eq.10.or.isec.eq.20.or.isec.eq.30
     +  .or.isec.eq.40.or.isec.eq.50) then
                  call scuwrite
                  endif
c          endif
                  call keyb
                  ib=0
                  if(lo)then
                        lo=.false.
                        call clscom(1,a1)
                        call clscom(2,a1)
                        goto 52
                  endif
                  ii(1)=1
            endif
            goto 1
      endif
      if(i10th.eq.1) then
c               write(5,*) i10th
            if(ii(2).eq.0) then
c                       write(5,*) i10th
                  call ever
                  ii(2)=1
                  call status(2,0,s,*1000,*321)
                  if(.not.s) then
                        write(3,7)iday,ihr,imin,isec,prp
```

```
                        call outcom(1,'¬S',2,il,a1)
                    endif
321                       continue
              endif
              goto 1
        endif
        if(i10th.eq.2) then
c                 write(5,*) i10th
              if(ii(3).eq.0) then
c                          write(5,*) i10th
                    call ever
                    call readc
                    ii(3)=1
              endif
              goto 1
        endif
        if(i10th.eq.3) then
c                 write(5,*) i10th
              if(ii(4).eq.0) then
c                 write(5,*) i10th
              if(stp) goto 44
              ons=1
              call readp
              if((abs(paz-azp).ge.0.005).or.(abs(pel-elp).ge.0.005))then
                    ons=0
                    call beep2(2000,5,a1)
              endif
              call ever
              ii(4)=1
44                call status(1,0,s,*1000,*322)
              call putcur(4,4,0,a1)
              if(s) write(tr,'(a3)') 'CW '
              if(.not.s) write(tr,'(a3)') 'CCW'
              call wchars(0,tr,3,2,0,0,a1)
322               continue
              endif
              goto 1
        endif
        if(i10th.eq.4) then
c                 write(5,*) i10th
              if(ii(5).eq.0) then
c                          write(5,*) i10th
                    call ever
                    call equ
                    ii(5)=1
              endif
              goto 1
        endif
        if(i10th.eq.5) then
c                 write(5,*) i10th
              if(ii(6).eq.0) then
c                          write(5,*) i10th
                    call ever
                    ii(6)=1
                    oldname=name
                    oldra=ra
                    oldde=de
                    oldazof=azof
                    oldelof=elof
c                          oldraof=raof
c                          olddecof=decof
                    if(hh) call hpread(*2000)
                    if (name.eq."scanaz") then
                    speed=ra
                    offin=de
c                        write(*,*) offin,speed
                    ntimes=1
```

```
                  if(azof.lt.offin) wscan=1
                  if(azof.gt.offin) wscan=-1
                  name="SCANNING"
                  ra=oldra
                  de=oldde
                  azof=oldazof
                  elof=oldelof
c                        raof=oldraof
c                        decof=olddecof
                  endif
                  if(name.eq."azel") then
                  paz=ra*180.0/3.14158
                  pel=de*180.0/3.14158
                  name="AZEL"
                  newstp=.true.
                  endif
            endif
            goto 1
      endif
      if(i10th.eq.6) then
c               write(5,*) i10th
            if(ii(7).eq.0) then
c                        write(5,*) i10th
                  call show1
                  call ever
                  ii(7)=1
            endif
            goto 1
      endif
      if(i10th.eq.7) then
c                 write(5,*) i10th
            if(ii(8).eq.0) then
c                        write(5,*) i10th
                  call ever
                  call graph
                  azpo=azp
                  elpo=elp
                  ii(8)=1
            endif
            goto 1
      endif
      if(i10th.eq.8) then
c                 write(5,*) i10th
            if(ii(9).eq.0) then
c                        write(5,*) i10th
                  call ever
                  if(hh) call hpwrite
                  ii(9)=1
            endif
            goto 1
      endif
      if(i10th.eq.9) then
c                 write(5,*) i10th
            if(ii(10).eq.0) then
c                        write(5,*) i10th
                  call ever
                  call show2
                  ii(10)=1
76                      call portin(768,0,isy)
                  if(isy.ne.247) goto 76
                  isec=isec+1
                  if(isec.eq.60) then
                        isec=0
                        imin=imin+1
                        if(imin.eq.60) then
                              imin=0
                              ihr=ihr+1
```

```
                              if(ihr.eq.24) ihr=0
                        endif
                  endif
                  call settim(ihr,imin,isec,10)
                  do 33 j=1,10
                  ii(j)=0
33                      continue
                  jjj=0
            endif
            goto 1111
      endif
7       format(' ',i3,'-',i2,':',i2,':',i2,'     ',a31)
2000    write(3,7) iday,ihr,imin,isec,
     +  'PC_A<-->PC_FS COMMUNICATION ERROR "'
      goto 1
      end
      subroutine ever
$include:'comm.for'
      if(.not.stp) then
            call gettim(ihr,imin,isec,i100th)
            i3=int(i100th/10)
            if((ihr+imin+isec.eq.0).and.(i3.le.3)) then
            call midn
            else
            ztim=(isec+imin*60+ihr*3600+i100th*0.01)*7.272205216e-5
            endif
            if(newstp) goto 1010
            call altaz
            pazo=500
            if(corr) then
                  if((azp.gt.90).and.(azp.lt.270)) then
                        if((azp.lt.180.).and.(paz.ge.270.)) then
                              if((paz-azp).gt.180.) then
                              pazo=paz
                              paz=45.
                              endif
                        endif
                        if((azp.ge.180.).and.(paz.le.90.)) then
                              if((azp-paz).gt.180) then
                              pazo=paz
                              paz=300
                              endif
                        endif
                  endif
            endif
      endif
1010    call x
      call joy
11      return
      end
```

COMM.FOR

```
real*8 datjul
real*4 stimo
real azp,elp,azc,elc,azpo,elpo,paz,pel,ra,de
real tlst
real ha,raof,decof,azof
real elof,ztim
real key,tp(8),offz1,offz2,offz3,clin,csqr,ccub,xs,ys
real sinlat,coslat,elev,pazo
real fl
character*10 name
character*6 fss,fve,fsc,fof
character*16 recev
integer wscan,new,ons
logical hpf,lo,stp,vel,scan,ofs,hh,corr,newstp
common /FAR/ azp,elp,azc,elc,azpo,elpo,paz,pel
common /FAR/ ra,de,new,name,tlst,wscan
common /FAR/ ihr,imin,isec,i100th,ons,ha,raof
common /FAR/ decof,azof,elof,stimo,ztim
common /FAR/ key,tp,datjul,iday
common /FAR/ sinlat,coslat,elev,ib,ick(3),apo
common /FAR/ epo,corr,pazo,speed,offin
common /FAR/ iraph,irapm,iraps,ideph,idepm
common /FAR/ ideps,irahh,irahm,irahs
common /FAR/ idehh,idehm,idehs,idrah,idram,idras
common /FAR/ iddeh,iddem,iddes
common /FAR/ iraofh,iraofm,iraofs,ideofh,ideofm
common /FAR/ ideofs,hpf,lo,stp,vel,ntimes,newstp
common /FAR/ scan,ofs,hh,fss,fve,fsc,fof,sig,fl,recev
common /FAR/ offz1,offz2,offz3,clin,csqr,ccub,xs,ys
data pi /3.141592653589/,drad /0.0174532925/,step /5.81764e-5/
```

```
      subroutine altaz
$include:'comm.for'
      data grad /57.29577951/,pi2 /1.5707963267949e0/
      data ppi /6.2831853071796e0/
      call sit1
      ha=tlst-ra+raof
c       write(*,*) ha*24/ppi,tlst*24/ppi
      cosd=cos(de+decof)
      sind=sin(de+decof)
      cosh=cos(ha)
      sinh=sin(ha)
      x=coslat*sind-sinlat*cosd*cosh
      y=-cosd*sinh
      z=coslat*cosd*cosh+sinlat*sind
      if((x.ne.0.0).or.(y.ne.0.0)) goto 10
      azim=0.0
      elev=sign(pi2,z)
      goto 20
10      pdist=sqrt(x*x+y*y)
      azim=atan2(y,x)
      if(azim.lt.0.0) azim=azim+ppi
      elev=atan2(z,pdist)
20      azim=azim+azof
      elin=elev+elof
      call rifra(elin,elout)
      elev=elout
      cose=cos(elev)
      sine=sin(elev)
      cosa=cos(azim)
      sina=sin(azim)
      sk=-tp(4)+tp(3)*sine-tp(6)*sine*cosa+tp(5)*sine*sina
      azim=azim+sk/cose+tp(1)
      elev=elev+tp(8)*cose+tp(6)*sina+tp(5)*cosa+tp(7)
      paz=grad*azim
      pel=grad*elev
      if(paz.ge.360.) paz=paz-360.
      if(paz.lt.0) paz=paz+360.
      if(pel.lt.5.) pel=4.999
      end
      subroutine sit1
$include:'comm.for'
      data dck2 /1.002737811906/,cu /6.2831853071795/
      data cpi /3.1415926535897/
      tlst=stimo+dck2*ztim
      tlst=amod(tlst,cu)
      if(tlst.ge.0.0) goto 1
      tlst=tlst+cu
1       continue
      end
```

```
subroutine astow (*)
character*2 as
data as /'¬D'/
call outlen(1,nlo,nlof,a1)
call outcom(1,as,2,il,a1)
return 1
end
•
```

```
subroutine conv(prad,ipd,ipm,ips,ick)
srad=prad*180.0/3.141592653589
if(srad.lt.0.0) srad=srad+360
if(srad.gt.180) srad=srad-360
ick=0
if(srad.lt.0) ick=1
srad=abs(srad)
ipd=srad
srad=srad-ipd
ipm=srad*60.
srad=srad*60.-ipm
pps=srad*60.
ips=int(pps)
pts=pps-ips
if(pts.lt.0.5) ips=aint(pps)
if(pts.ge.0.5) ips=anint(pps)
if(ips.eq.60) then
      ips=0
      ipm=ipm+1
      if(ipm.eq.60) then
            ipm=0
            ipd=ipd+1
            if(ipd.eq.91) then
                  ipd=89
                  ick=abs(ick-1)
            endif
      endif
endif
end
```

```
subroutine conver(rrad,iih,iim,iis)
srad=rrad*12.0/3.141592653589
if(srad.lt.0.0) srad=24.0+srad
if(srad.ge.24.0) srad=srad-24.0
iih=srad
srad=srad-iih
iim=srad*60.0
srad=srad*60.0-iim
sss=srad*60.0
iis=int(sss)
sts=sss-iis
if(sts.lt.0.5) iis=aint(sss)
if(sts.ge.0.5) iis=anint(sss)
if(iis.eq.60) then
      iis=00
      iim=iim+1
      if(iim.eq.60) then
            iim=00
            iih=iih+1
            if(iih.eq.24) iih=0
      endif
endif
end
```

```
      subroutine equ
      double precision decdp,arp,dsind,dcosh,dsinh
$include:'comm.for'
      data pi2 /6.2831853071795/
      e1=elp*drad-elof
      a1=azp*drad-azof
      ce=cos(e1)
      se=sin(e1)
      ca=cos(a1)
      sa=sin(a1)
      sk=-tp(4)+tp(3)*se-tp(6)*se*ca+tp(5)*se*sa
      az1=a1-sk/ce-tp(1)
      el1=e1-tp(7)-tp(8)*ce-tp(6)*sa-tp(5)*ca
      call rifra(el1,eout)
      elev=eout
      el1=2*el1-elev
      call cuv1(az1,el1,x,y,z)
      x1=coslat*x+sinlat*z
      y1=-y
      z1=-sinlat*x+coslat*z
      dsind=dble(x1)
      decdp=dasin(dsind)
      if(dsind.eq.1) then
            arp=0
      else
            dcosd=dcos(decdp)
            dsinh=y1/dcosd
            dcosh=z1/dcosd
            arp=datan2(dsinh,dcosh)
      endif
      dep=sngl(decdp)
      rap=tlst-sngl(arp)
      if(rap.lt.0.0) rap=rap+pi2
      call conver(rap,iraph,irapm,iraps)
      call conv(dep,ideph,idepm,ideps,ick(1))
      call conver(raof,iraofh,iraofm,iraofs)
      call conv(decof,ideofh,ideofm,ideofs,ick(2))
      call conver((ra-raof),irahh,irahm,irahs)
      call conv((de-decof),idehh,idehm,idehs,ick(3))
      call conver(abs(ra-raof-rap),idrah,idram,idras)
      call conv(abs(dep-de-decof),iddeh,iddem,iddes,ites)
      end
      subroutine cuv1(al,be,x,y,z)
      cosbet=cos(be)
      x=cos(al)*cosbet
      y=sin(al)*cosbet
      z=sin(be)
      end
```

```
      subroutine graph
      integer apo,epo,ap,ep,ac,ec
$include:'comm.for'
      apo=int(azpo)
      epo=int(elpo)
      ap=int(azp)
      ep=int(elp)
      ac=int(azc)
      ec=int(elc)
      if(fl.eq.0) then
            fl=1
            goto 1
      endif
      call circa(0,apo+140,340-epo,3,15,1,1000,a1)
      if(ac.gt.359) return
      if(ap.gt.359) return
      if(ec.gt.125) return
      if(ep.gt.125) return
      call wdot(0,340-ec,ac+140,0,0,a1)
1       call circa(0,ap+140,340-ep,3,15,1,1000,a1)
      end
```

```
      subroutine hpread(*)
      character*1 a,b,f,c,d,g,l,asi,bsi
      character*4 err
      character*2 dummy,sign
      character*7 rad,dec,azod,elod,raod,decod
      character*18 hhhh
      character*10 vname
      real vra,vde,vraof,vdecof,vazof,velof,h,k,ko
      real ho,koo,hoo
      integer*4 erc,ert,o
      logical lsign
      dimension o(10)
      dimension k(7),h(7),ko(7),ho(7),koo(7),hoo(7)
$include:'comm.for'
      call portot(768,0,0)
      icc=0
33      call comlen(2,ik,ll,a1)
      if(ik.lt.59) return
c        if(ik.lt.59) then
c                xxx=sin(1)**(cos(1)*tan(1))
c                icc=icc+1
c                if(icc.ge.1500) then
c                        call comlen(2,ik,ll,a1)
c                        call incom(2,f,ik,il,a1)
c                        write(*,*) ik
c                        return 1
c                endif
c                goto 33
c        endif
c        call putcur(2,2,0,a1)
      call incom(2,f,1,il,a1)
c        write(*,*) il
      if(f.ne.'z'.and.f.ne.'w') then
             call comlen(2,ik,ll,a1)
             call incom(2,f,ik,il,a1)
c               return 1
      endif
      new=1
      ert=ichar(f)-87
      if (ert.eq.32) ert=37
      call incom(2,rad,7,il,a1)
      call incom(2,dec,7,il,a1)
      call incom(2,vname,10,il,a1)
      call incom(2,raod,7,il,a1)
      call incom(2,decod,7,il,a1)
      call incom(2,azod,7,il,a1)
      call incom(2,elod,7,il,a1)
      call incom(2,sign,2,il,a1)
      call incom(2,err,4,il,a1)
      if(ik.eq.60) call incom(2,dummy,1,il,a1)
      if(ik.eq.61) call incom(2,dummy,2,il,a1)
      call midchr(sign,asi,1,1)
      call midchr(sign,bsi,2,1)
      do 20 i=1,7
      call midchr(rad,a,i,1)
      call midchr(dec,b,i,1)
      call midchr(raod,g,i,1)
      call midchr(decod,l,i,1)
      call midchr(azod,c,i,1)
      call midchr(elod,d,i,1)
      k(i)=ichar(a)-48
      h(i)=ichar(b)-48
      ko(i)=ichar(c)-48
      ho(i)=ichar(d)-48
      koo(i)=ichar(g)-48
      hoo(i)=ichar(l)-48
20      continue
```

```
      vra=(k(1)*10**6+k(2)*10**5+k(3)*10**4+k(4)*10**3+k(5)*100+k(6)*10+
     +  k(7))/10**6
      vde=(h(1)*10**6+h(2)*10**5+h(3)*10**4+h(4)*10**3+h(5)*100+h(6)*10+
     +  h(7))/10**6
      vazof=(ko(1)*10**6+ko(2)*10**5+ko(3)*10**4+ko(4)*10**3+ko(5)*100+
     +  ko(6)*10+ko(7))/10**6
      velof=(ho(1)*10**6+ho(2)*10**5+ho(3)*10**4+ho(4)*10**3+ho(5)*100+
     +  ho(6)*10+ho(7))/10**6
      vraof=(koo(1)*10**6+koo(2)*10**5+koo(3)*10**4+koo(4)*10**3+
     +  koo(5)*100+koo(6)*10+koo(7))/10**6
      vdecof=(hoo(1)*10**6+hoo(2)*10**5+hoo(3)*10**4+hoo(4)*10**3+
     +  hoo(5)*100+hoo(6)*10+hoo(7))/10**6
      isigna=ichar(asi)-48
      isignb=ichar(bsi)-48
      isign=isigna*10+isignb
      ert=ert+isigna+isignb
      do 444 i=1,7
      ert=ert+k(i)+h(i)+ko(i)+ho(i)+koo(i)+hoo(i)
444     continue
      do 555 i=1,10
      call midchr(vname,a,i,1)
      o(i)=ichar(a)
      if((o(i).le.57).and.(o(i).ge.48)) then
            o(i)=o(i)-48
            ert=ert+o(i)
      endif
      if((o(i).le.122).and.(o(i).ge.97)) then
            o(i)=o(i)-87
            if (o(i).eq.32) o(i)=37
            ert=ert+o(i)
      endif
      if(o(i).eq.32) ert=ert+36
555     continue
      do 90 i=1,4
      call midchr(err,a,i,1)
      o(i)=ichar(a)-48
90      continue
      erc=(o(1)*1000)+(o(2)*100)+(o(3)*10)+o(4)
      if(erc.ne.ert) then
c                 write(*,*) erc,ert
            call outcom(2,21,1,il1,a1)
            return
      endif
      newstp=.false.
      wscan=0
      name=vname
      ra=vra
      de=vde
      raof=vraof
      decof=vdecof
      azof=vazof
      elof=velof
      if(f.eq.'w') de=-de
      lsign=btest(isign,0)
      if(lsign) decof=-decof
      lsign=btest(isign,1)
      if(lsign) raof=-raof
      lsign=btest(isign,2)
      if(lsign) elof=-elof
      lsign=btest(isign,3)
      if(lsign) azof=-azof
      call outcom(2,6,1,il1,a1)
      end
```

```
      subroutine hpwrite
      integer k,m
      character ch,jj,i3,i2,iyc,ihc,imc,isc
      character*15 cht
      dimension k(12)
$include:'comm.for'
      call portot(768,0,0)
      jj=70
      if(ons.eq.1) jj=79
      if(stp) jj=80
      if(lo) jj=70
      call concat(cht,jj,1,1)

      call getdat(iyear,imon,iiday)
      iyea=iyear-1900
      iyc=iyea
      i3p=iday/100
      i3=i3p
      i2p=(iday-i3p*100)
      i2=i2p
      ihc=ihr
      imc=imin
      isc=isec
      call concat(cht,iyc,2,1)
      call concat(cht,i3,3,1)
      call concat(cht,i2,4,1)
      call concat(cht,ihc,5,1)
      call concat(cht,imc,6,1)
      call concat(cht,isc,7,1)
      call outcom(2,cht,7,il,a1)
      l=0
      iz=azof*1000.0*180.0/PI
10      k(l+1)=1
      if (iz .lt. 0) then
          iz = -iz
          k(l+1)=-1
      endif
      do 30 i=1,5
          k(l+7-i)=mod(iz,10)
          iz=iz/10
30      continue
      if(l.eq.0) then
            l=6
            iz=elof*1000.0*180.0/PI
            goto 10
      endif
      do 40 i=1,12
      ch=k(i)
40      call concat(cht,ch,i,1)
      call outcom(2,cht,12,il,a1)
      l=0
      iz=azc*1000
80      do 100 i=1,6
          k(l+7-i)=mod(iz,10)
          iz=iz/10
100     continue
      if(l.eq.0) then
            l=6
            iz=elc*1000
            goto 80
      endif
      do 110 i=1,12
      ch=k(i)
110     call concat(cht,ch,i,1)
      call outcom(2,cht,12,il,a1)
      call outcom(2,'x',1,il2,a1)
      end
```

```
      subroutine iniz
$include:'comm.for'
      alat=(36.+52./60.+36./3600.)*drad
      coslat=cos(alat)
      sinlat=sin(alat)
      call midn
      stp=.false.
      vel=.false.
      scan=.false.
      ofs=.false.
      lo=.false.
      write(fss,'(a6)') ' STOP'
      write(fsc,'(a5)') ' SCAN'
      raof=0.
      decof=0.
      azof=0.
      elof=0.
c        open(40,file='rifraz')
c        read(40,*) mu,h,zs,zt,d,lg,rg1,val1
c        close(40)
      open(40,file='tp')
      read(40,*) tp
      read(40,'(a15)') recev
      read(40,*) offz1,offz2,offz3,clin,csqr,ccub,xs,ys
      close(40)
      do 87 i=1,8
      tp(i)=tp(i)*drad
87      continue
      end
```

```
      subroutine joy
$include:'comm.for'
      call portin(512,0,i)
      call inkey(a1,ik,ic)
      if(ic.ne.0) ib=ik
      if(stp) goto 100
      if(wscan.eq.1) then
      if(azof.le.offin) then
            azof=azof+speed/150
      else
            wscan=0
      endif
      endif
      if(wscan.eq.-1) then
      if(azof.ge.offin) then
            azof=azof-speed/150
      else
            wscan=0
      endif
      endif
      if(scan) then
      speed1=0.25
      if(.not.vel) speed1=2.5
      if((ib.eq.66).or.(i.eq.239).or.(i.eq.175).or.(i.eq.111)) then
            if(ofs) elof=elof+speed1*step
            if(.not.ofs) decof=decof+speed1*step
      endif
      if((ib.eq.67).or.(i.eq.223).or.(i.eq.159).or.(i.eq.95)) then
            if(ofs) then
                  elof=elof-speed1*step
            else
                  decof=decof-speed1*step
            endif
      endif
      if((ib.eq.68).or.(i.eq.127).or.(i.eq.111).or.(i.eq.95)) then
            if(ofs) then
                  azof=azof-speed1*step
            else
                  raof=raof+speed1*step
            endif
      endif
      if((ib.eq.65).or.(i.eq.191).or.(i.eq.175).or.(i.eq.159)) then
            if(ofs) then
                  azof=azof+speed1*step
            else
                  raof=raof-speed1*step
            endif
      endif
      if(ib.eq.24) then
            azof=0
            elof=0
      endif
      if(ib.eq.19) then
            raof=0
            decof=0
      endif
      endif
100     end
```

```
      subroutine keyb
      character*1 sig
$include:'comm.for'
      ia=ib
      if(ia.eq.59) then
            lo=.true.
            call portot(768,0,0)
      endif
      if(ia.eq.60) then
            if(stp) then
                  stp=.false.
                  write(fss,'(a5)') ' STOP'
            else
                  stp=.true.
                  write(fss,'(a6)') ' START'
                  paz=azp
                  pel=elp
            endif
      endif
      if(scan) then
            if(ia.eq.61) then
                  if(vel) then
                        vel=.false.
                        write(fve,'(a5)') ' SLOW'
                  else
                        vel=.true.
                        write(fve,'(a5)') ' FAST'
                  endif
            endif
      endif
      if(ia.eq.62) then
            if(scan) then
                  scan=.false.
                  write(fsc,'(a5)') ' SCAN'
                  write(fve,'(a5)') '     '
                  write(fof,'(a6)') '      '
                  vel=.false.
                  ofs=.false.
            else
                  scan=.true.
                  write(fsc,'(a6)') 'NOSCAN'
                  write(fve,'(a5)') ' SLOW'
                  write(fof,'(a6)') ' AZ-EL'
            endif
      endif
      if(scan) then
            if(ia.eq.63) then
                  if(ofs) then
                        ofs=.false.
                        write(fof,'(a6)') ' AZ-EL'
                  else
                        ofs=.true.
                        write(fof,'(a6)') 'RA-DEC'
                  endif
            endif
      endif
      if(ia.eq.64) call nsor
      call putcur(2,10,0,a1)
      call wchars(0,fss,6,4,0,0,a1)
      call putcur(2,18,0,a1)
      call wchars(0,fve,5,9,0,0,a1)
      call putcur(2,26,0,a1)
      call wchars(0,fsc,6,9,0,0,a1)
      call putcur(2,34,0,a1)
      call wchars(0,fof,6,9,0,0,a1)
      end
```

```
      subroutine nsor
      character*20 st,nam
      character*6 ts
      character*35 tts
      dimension st(3)
$include:'comm.for'
      data st(1) /'SOURCE NAME'/,st(2) /'RA (hh mm ss.d)'/
      data st(3) /'DE (ñdd mm ss.d)'/
5       do 10 i=1,3
      write(ts,'(a6)') 'ENTER '
      call putcur(14,15,0,a1)
      call wchars(0,ts,6,15,0,0,a1)
      call wchars(0,st(i),17,15,0,0,a1)
      if(i.eq.1) read(*,'(a10)',err=5) nam
      if(nam.eq.'*') goto 9
      name=nam
      if(i.eq.2) read(*,'(i2,1x,i2,1x,f4.1)',err=5) irh,irm,rs
      if(i.eq.3) read(*,'(a1,i2,1x,i2,1x,f4.1)',err=5)sig,idh,idm,ds
9       call putcur(14,15,0,a1)
      write(tts,'(a35)') '                                          '
      call wchars(0,tts,35,1,0,0,a1)
      call putcur(15,1,0,a1)
      call wchars(0,tts,10,1,0,0,a1)
      if(nam.eq.'*') return
10      continue
      ra=(abs(irh)*3600.+abs(irm)*60.+abs(rs))*pi/(12.*3600.)
      de=(abs(idh)*3600.+abs(idm)*60.+abs(ds))*pi/(180.*3600.)
      if(sig.eq.'-') de=-de
      end
      •
```

```
      subroutine mask
      character*3 f
      character*7 st
      character*16 sp
      character*5 tt
      dimension f(10),st(16)
$include:'comm.for'
      data f(1) /'F1'/,f(2) /'F2'/,f(3) /'F3'/,f(4) /'F4'/,f(5) /'F5'/
      data f(6) /'F6'/,f(7) /'F7'/,f(8) /'F8'/,f(9) /'F9'/,f(10) /'F10'/
      data st(1) /'AZ  COM'/,st(2) /'EL  COM'/,st(3) /'RA  COM'/
      data st(4) /'DE COM'/,st(5) /'AZ  CUR'/,st(6) /'EL  CUR'/
      data st(7) /'RA  CUR'/,st(8) /'DE CUR'/,st(9) /'DIF  AZ'/
      data st(10) /'DIF  EL'/,st(11) /'DIF  RA'/,st(12) /'DIF DE'/
      data st(13) /'OFF  AZ'/,st(14) /'OFF  EL'/,st(15) /'OFF  RA'/
      data st(16) /'OFF DE'/
      call screen(7,1,80,0,a1)
      call linea(0,130,205,510,350,3,2,0,0,0,a1)
      call linea(0,130,205,510,350,15,1,0,0,0,a1)
c        call painta(0,320,280,9,15,0,0,a1,a1)
      do 10 i=6,582,64
      j=52+i
10      call linea(0,i,13,j,28,15,1,0,0,0,a1)
      i=0
1       call putcur(1,8*i+4,0,a1)
      call wchars(0,f(i+1),3,7,0,1,a1)
      i=i+1
      if(i.le.9) goto 1
      do 20 i=1,4
      do 30 j=0,12,4
      call putcur(j/2+6,6+(i-1)*18,0,a1)
      call wchars(0,st(i+j),7,7,0,1,a1)
30      continue
20      continue
      call putcur(4,34,0,a1)
      write(sp,'(a9)') 'RECEIVER '
      write(sp(9:),'(a7)') recev
      call wchars(0,sp,16,14,0,1,a1)
      call putcur(18,69,0,a1)
      write(sp,88) 'UT TIME'
      call wchars(0,sp,7,7,0,1,a1)
      call putcur(18,2,0,a1)
      write(sp,99) 'SOURCE NAME'
      call wchars(0,sp,11,7,0,1,a1)
99      format(a11)
      call linea(0,36,64,610,172,15,1,0,0,0,a1)
      call linea(0,177,65,177,171,15,0,0,0,0,a1)
      call linea(0,37,90,609,90,15,0,0,0,0,a1)
      call linea(0,321,65,321,171,15,0,0,0,0,a1)
      call linea(0,37,118,609,118,15,0,0,0,0,a1)
      call linea(0,467,65,467,171,15,0,0,0,0,a1)
      call linea(0,37,146,609,146,15,0,0,0,0,a1)
      call putcur(16,14,0,a1)
      write(sp,77) 'EL'
      call wchars(0,sp,2,7,0,1,a1)
77      format(a2)
88      format(a7)
      call putcur(25,66,0,a1)
      write(sp,77) 'AZ'
      call wchars(0,sp,2,7,0,1,a1)
      call putcur(2,3,0,a1)
      write(tt,'(a4)') 'EXIT'
      call wchars(0,tt,4,4,0,1,a1)
      call putcur(2,43,0,a1)
      write(tt,'(a5)') 'NEW S'
      call wchars(0,tt,5,9,0,1,a1)
      call putcur(2,51,0,a1)
      write(tt,'(a5)') ' EST '
```

```
call wchars(0,tt,5,9,0,1,a1)
call putcur(2,59,0,a1)
write(tt,'(a5)') ' NORD'
call wchars(0,tt,5,9,0,1,a1)
call putcur(2,67,0,a1)
write(tt,'(a5)') ' SUD '
call wchars(0,tt,5,9,0,1,a1)
call putcur(2,75,0,a1)
write(tt,'(a5)') 'OVEST'
call wchars(0,tt,5,9,0,1,a1)
end
```

```
      subroutine midn
$include:'comm.for'
      wlong=-14.9886*drad
      ztlong=0.*drad
      call tim(iyear)
      call sit0(iyear,iday,wlong,ztlong)
      ztim=(isec+imin*60.+ihr*3600.+i100th*0.01)*7.272205216e-5
      datjul=datjul+ztim*0.159154943
      end
      subroutine sit0(iy,id,oblong,ztlong)
      real*8 djul0,djl,t,ga,dp2,dck2,red
$include:'comm.for'
      data dck2 /1.002737811906d0/, dp2 /6.2831853071795d0/
      data cu /6.2831853071795d0/
      iy1=iy-1
      djl=iy1/4-iy1/100+iy1/400+id
      djul0=djl+iy*365.0d0+1721059.5d0
      t=(djul0-2415020.0d0)/36525.0d0
      ga=(.276919398d0+100.002135902777d0*t+1.0752d-6*t*t)*dp2
      alpha=ga-oblong+dck2*ztlong
      alpha=dmod(alpha,cu)
      if(alpha.ge.0.0) goto 1
      alpha=alpha+cu
1     stimo=alpha
      datjul=djul0+ztlong/dp2
      end
      subroutine tim(iyear)
      dimension iy(12)
$include:'comm.for'
      data iy /0,31,59,90,120,151,181,212,243,273,304,334/
      call gettim(ihr,imin,isec,i100th)
      call getdat(iyear,imon,iday)
      iday=iy(imon)+iday
      if(imon.le.2) return
      if((iyear.eq.1992).or.(iyear.eq.1996).or.(iyear.eq.2000))then
            iday=iday+1
      endif
      end
```

```
      subroutine parm(*)
      character*25 ms,rb
      character*1 a
      dimension pp(8)
$include:'comm.for'
      dgrad=57.29577951
      open(22,file='tp')
      write(rb,'(a25)') '                         '
5       call cls
      do 10 i=1,8
      pp(i)=tp(i)*dgrad
      write(*,100) i,pp(i)
100     format(' tp(',i1,')=',f11.6)
10      continue
      write(ms,'(a25)') ' Are these values ok? Y/N'
      call putcur(10,3,0,a1)
      call wchars(0,ms,25,4,0,0,a1)
7       call inkey(a,ia,ic)
      if(ic.eq.0) goto 7
      call putcur(10,3,0,a1)
      call wchars(0,rb,25,1,0,0,a1)
      if((a.eq.'Y').or.(a.eq.'y')) return 1
20      write(ms,'(a19)') 'Enter tp# to change'
      call putcur(10,3,0,a1)
      call wchars(0,ms,19,1,0,0,a1)
      ic=0
8       call inkey(a,ia,ic)
      if(ic.eq.0) goto 8
      call putcur(10,3,0,a1)
      call wchars(0,rb,25,1,0,0,a1)
      if((ia.lt.2).or.(ia.gt.9)) goto 20
      write(ms,'(a15)') 'Enter new value'
      call putcur(10,3,0,a1)
      call wchars(0,ms,15,1,0,0,a1)
      call putcur(10,20,0,a1)
      read(*,*,err=20) v
      tp(ia-1)=v*drad
      goto 5
      end
```

```
      subroutine printer (prflag)
      character*35 str
      character*1 kk
      integer prst,a1,prflag
      call prnasm(0,prst)
      if(prst.ne.16) then
          prflag=0
          str='PRINTER NOT READY'
          call putcur(4,10,15,0,a1)
          call wchars(0,str,35,4,0,0,a1)
10      call inkey(kk,i,ii)
      if(ii.eq.0) goto 10
      else
          prflag=1
      endif
      end
```

```
      subroutine readc
      real*8 xxx
      character*7 b
      character*1 d
      character*2 a
      dimension k(12)
$include:'comm.for'
      data a /'¬V'/
      i=0
      j=2
      icount=0
      call comlen(1,lk,ll,a1)
      call incom(1,b,lk,il,a1)
      call outcom(1,a,2,il,a1)
1       call comlen(1,lk,ll,a1)
      if(lk.lt.7) then
            do 123 jk=1,1000
123             xxx=sin(1)**(cos(1)*tan(1))
            call comlen(1,lk,ll,a1)
            if(lk.lt.7) return
      endif
      call incom(1,b,7,il,a1)
      call midchr(b,d,1,1)
      if(d.eq.'¬') goto 300
10      call midchr(b,d,j,1)
      k(i+j-1)=iand(ichar(d),240)
      k(i+j-1)=ishl(k(i+j-1),-4)
      k(i+j)=iand(ichar(d),15)
      i=i+1
      j=j+1
      if (j.le.7) goto 10
      azc=100*k(1)+10*k(2)+k(3)+0.1*k(4)+0.01*k(5)+0.001*k(6)
      elc=100*k(7)+10*k(8)+k(9)+0.1*k(10)+0.01*k(11)+0.001*k(12)
      if(azc.ge.360.) azc=azc-360.
      if(azc.lt.0.) azc=azc+360.
      return
300     call incom(1,b,1,il,a1)
      end
```

```
      subroutine readp
      character*7 b
      character*1 d
      character*2 a
      dimension k(12)
$include:'comm.for'
      data a /'¬T'/
      i=0
      j=2
      icount=0
      call comlen(1,lk,ll,a1)
      call incom(1,b,lk,il,a1)
      call outcom(1,a,2,il,a1)
1       call comlen(1,lk,ll,a1)
      if(lk.lt.7) then
            do 123 jk=1,1000
123             xxx=sin(1)**(cos(1)*tan(1))
            call comlen(1,lk,ll,a1)
            if(lk.lt.7) return
      endif
      call incom(1,b,7,il,a1)
      call midchr(b,d,1,1)
      if(d.eq.'¬') goto 300
10      call midchr(b,d,j,1)
      k(i+j-1)=iand(ichar(d),240)
      k(i+j-1)=ishl(k(i+j-1),-4)
      k(i+j)=iand(ichar(d),15)
      i=i+1
      j=j+1
      if (j.le.7) goto 10
      azp=100*k(1)+10*k(2)+k(3)+0.1*k(4)+0.01*k(5)+0.001*k(6)
      elp=100*k(7)+10*k(8)+k(9)+0.1*k(10)+0.01*k(11)+0.001*k(12)
      if(azp.ge.360.) azp=azp-360
      if(azp.lt.0.) azp=azp+360
      return
300     call incom(1,b,1,il,a1)
      end
```

```
      subroutine rifra(e,eo)
      real mu,rg1(22)
$include:'comm.for'
      data lg/19/
      data rg1 /8.61235e-2,9.1747e-2,9.73551e-2,.10295,.10853,
     +  .11967,.13077,.14185,.1529,.16394,.17496,.18597,.20796,
     +  .22991,.25185,.27376,.29567,.33944,.38318,-.96873,
     +  -.96873,-.96873/
      mu=1.00032
      if(e.ge.rg1(lg)) goto 1
      call lag(e,eo)
c        write(*,*) eo
      return
1       ta=1./tan(e)
      del=(((mu-1.)/mu)*ta)
      eo=e+del
c        write(*,*) mu
c        eo=e
      end
      subroutine lag(e,el)
      real rg1(22),val1(22)
$include:'comm.for'
      data h /10./, zs /68./, zt /85.01/,d /2.5/,lg /19/
      data rg1 /8.61235e-2,9.1747e-2,9.73551e-2,.10295,.10853,
     +  .11967,.13077,.14185,.1529,.16394,.17496,.18597,.20796,
     +  .22991,.25185,.27376,.29567,.33944,.38318,-.96873,
     +  -.96873,-.96873/
      data val1 /3.32411e-3,3.15455e-3,3.00044e-3,
     +  2.85986e-3,2.73117e-3,
     +  2.50409e-3,2.3103e-3,2.14313e-3,1.99756e-3,
     +  1.86971e-3,1.75656e-3,1.65575e-3,1.48388e-3,
     +  1.34279e-3,1.22489e-3,
     +  1.12483e-3,1.03882e-3,8.98374e-4,7.88299e-4,
     +  -.96873,-.96873,-.96873/
      el=0
      l=lg
      if(e.ge.rg1(1)) goto 3
33      elev=el
      return
3       do 1 i=1,l
      k=i
      if((rg1(i).le.e).and.(e.le.rg1(i+1))) goto 2
1       continue
      goto 33
2       rif=(val1(k)*(e-rg1(k+1))-val1(k+1)*(e-rg1(k)))
      rrif=rg1(k)-rg1(k+1)
      rif=rif/rrif
      el=e+rif
      end
```

```
      subroutine rubid(iih,iim,iis)
      integer ic(10)
      real*8 xxx
      character*1 cc
      character*3 cx
      character*10 c
      call iopcom(2,8,2,7,2,0,0,0,0,0,ierr)
      call portot(768,0,2)
      call outcom(2,'@',1,1,ierr)
      call outcom(2,'UUUUUUUUUUUUUUUUUUUU',20,20,ierr)
      call outcom(2,'#21:02103E00FFFFB2',18,18,ierr)
22      call comlen(2,ii,ll,ierr)
      if (ii.lt.3) then
            do 123 k=1,5000
123             xxx=sin(1)**(cos(1)*tan(1))
            goto 22
      endif
      call incom (2,cx,3,3,ierr)
      call outcom(2,'#21?',4,4,ierr)
32      call comlen(2,ii,ll,ierr)
      if (ii.lt.10) then
            do 124 k=1,2500
124             xxx=sin(1)**(cos(1)*tan(1))
            goto 32
      endif
      call incom (2,c,10,10,ierr)
      do 10 i=1,8
      call midchr(c,cc,i,1)
      ix=ichar(cc)
      if(ix.ge.48.and.ix.le.57) ic(i)=ix-48
      if(ix.ge.65.and.ix.le.70) ic(i)=ix-55
10      continue
      iis=10*ic(7)+ic(8)
      iim=10*ic(5)+ic(6)
      ih10=iand(ic(3),3)
      iih=10*ih10+ic(4)
      call iopcom(2,8,2,8,1,0,0,0,0,0,ierr)
c        write (*,*) iih,iim,iis
1       end
```

```
      subroutine scuwrite
      character ch(6)
      character*2 head1,head2,head3,headx,heady
      character*3 ppp1,ppp2,ppp3,pppx,pppy
      character*25 pppt
      integer k1,k2,k3,kx,ky
      dimension k1(6),k2(6),k3(6),kx(6),ky(6)
$include:'comm.for'
      head1='¬h'
      head2='¬i'
      head3='¬j'
      headx='¬f'
      heady='¬g'
      call portot(768,0,1)
c        call clscom(2,a1)
c        call iopcom(2,8,2,8,1,0,0,0,0,0,a1)
      z=(clin*pel)+((csqr)*(pel**2))
      z=z+((ccub)*(pel**3))
      iz1=(z+offz1)*1000
      iz2=(z+offz2)*1000
      iz3=(z+offz3)*1000
      izx=xs*1000
      izy=ys*1000



      do 30 i=1,6
            s=0
            do 20 j=1,i-1
                  s=s+k1(j)*10**(6-j)
20              continue
            k1(i)=(iz1-s)/(10**(6-i))
30      continue
      do 40 i=1,5,2
            k1(i)=ishl(k1(i),4)
            k1(i)=iand(k1(i),240)
            k1(i)=k1(i)+k1(i+1)
            ch(i)=char(k1(i))
40      continue
      j=1
      do 100 i=1,5,2
            call concat(ppp1,ch(i),j,1)
            j=j+1
100     continue



      do 301 i=1,6
            s=0
            do 201 j=1,i-1
                  s=s+k2(j)*10**(6-j)
201             continue
            k2(i)=(iz2-s)/(10**(6-i))
301     continue
      do 401 i=1,5,2
            k2(i)=ishl(k2(i),4)
            k2(i)=iand(k2(i),240)
            k2(i)=k2(i)+k2(i+1)
            ch(i)=char(k2(i))
401     continue
      j=1
      do 1001 i=1,5,2
            call concat(ppp2,ch(i),j,1)
            j=j+1
1001    continue
```

```
      do 302 i=1,6
            s=0
            do 202 j=1,i-1
                  s=s+k3(j)*10**(6-j)
202               continue
            k3(i)=(iz3-s)/(10**(6-i))
302      continue
      do 402 i=1,5,2
            k3(i)=ishl(k3(i),4)
            k3(i)=iand(k3(i),240)
            k3(i)=k3(i)+k3(i+1)
            ch(i)=char(k3(i))
402      continue
      j=1
      do 1002 i=1,5,2
            call concat(ppp3,ch(i),j,1)
            j=j+1
1002     continue


      do 303 i=1,6
            s=0
            do 203 j=1,i-1
                  s=s+kx(j)*10**(6-j)
203               continue
            kx(i)=(izx-s)/(10**(6-i))
303      continue
      do 403 i=1,5,2
            kx(i)=ishl(kx(i),4)
            kx(i)=iand(kx(i),240)
            kx(i)=kx(i)+kx(i+1)
            ch(i)=char(kx(i))
403      continue
      j=1
      do 1003 i=1,5,2
            call concat(pppx,ch(i),j,1)
            j=j+1
1003     continue

      do 304 i=1,6
            s=0
            do 204 j=1,i-1
                  s=s+ky(j)*10**(6-j)
204               continue
            ky(i)=(izy-s)/(10**(6-i))
304      continue
      do 404 i=1,5,2
            ky(i)=ishl(ky(i),4)
            ky(i)=iand(ky(i),240)
            ky(i)=ky(i)+ky(i+1)
            ch(i)=char(ky(i))
404      continue
      j=1
      do 1004 i=1,5,2
            call concat(pppy,ch(i),j,1)
            j=j+1
1004     continue


      pppt=head1//ppp1//head2//ppp2//head3//ppp3//heady//pppy
     +//headx//pppx
      call outcom(2,pppt,25,il,a1)
      end
```

```
      subroutine setup (*)
      logical so,si,su
      character*3 us,cp
      character*5 jj
      character*45 ms
      data us /'¬AC'/,cp /'¬S'/
      call cls
      write(jj,'(a5)') 'SETUP'
      call putcur(20,32,0,a1)
      call wchars(0,jj,5,4,0,1,a1)
      call status(2,7,so,*110,*110)
      if(so) then
            call outcom(1,us,3,il,a1)
      else
            goto 100
      endif
11      call gettim(ihr,imin,isec,i100th)
      if(imin.eq.59) goto 11
22      call gettim(ihr,im,is,i100th)
      if(im.ne.imin+1) goto 22
      call outcom(1,cp,2,il,a1)
      return1
110     write(ms,'(a45)') 'PC <--> ACU COMMUNICATION ERROR'
      goto 115
100     write(ms,'(a45)')'CONTROL:COMPUTER ENABLE & DRIVE POWER SWITCH'
115     call cls
      call putcur(10,15,0,a1)
      call wchars (0,ms,45,1,0,1,a1)
5       call inkey(a,a,i)
      if(i.eq.0) goto 5
      return 1
      end
      •
```

```
      subroutine show1
      character*8 st
$include:'comm.for'
      data grad /57.29577951/
      if((pazo.ne.500).and.(.not.stp)) azc=pazo
      do 20 i=1,2
      do 30 j=0,12,4
      if((i.eq.1).and.(j.eq.0)) write(st,1) azc
      if((i.eq.1).and.(j.eq.4)) write(st,1) azp
      if((i.eq.1).and.(j.eq.8)) write(st,1) abs(azc-azp)
      if((i.eq.1).and.(j.eq.12)) write(st,1) (azof*grad)
      if((i.eq.2).and.(j.eq.0)) write(st,1) elc
      if((i.eq.2).and.(j.eq.4)) write(st,1) elp
      if((i.eq.2).and.(j.eq.8)) write(st,1) abs(elc-elp)
      if((i.eq.2).and.(j.eq.12)) write(st,1) (elof*grad)
1       format(f7.3)
      call putcur(j/2+6,15+(i-1)*18,0,a1)
      call wchars(0,st,8,14,0,0,a1)
30      continue
20      continue
      end
```

```
      subroutine show2
      character*10 st
      character*1 sg1,sg2,sg3
$include:'comm.for'
      sg2=' '
      if(ick(1).eq.1) sg2='-'
      sg3=' '
      if(ick(2).eq.1) sg3='-'
      sg1=' '
      if(ick(3).eq.1) sg1='-'
      do 20 i=3,4
      do 30 j=0,12,4
      if((i.eq.3).and.(j.eq.0)) write(st,2) irahh,irahm,irahs
      if((i.eq.3).and.(j.eq.4)) write(st,2) iraph,irapm,iraps
      if((i.eq.3).and.(j.eq.8)) write(st,2) idrah,idram,idras
      if((i.eq.3).and.(j.eq.12)) write(st,2) iraofh,iraofm,iraofs
      if((i.eq.4).and.(j.eq.0)) write(st,4) sg1,idehh,idehm,idehs
      if((i.eq.4).and.(j.eq.4)) write(st,4) sg2,ideph,idepm,ideps
      if((i.eq.4).and.(j.eq.8)) write(st,3) iddeh,iddem,iddes
      if((i.eq.4).and.(j.eq.12)) write(st,4) sg3,ideofh,ideofm,ideofs
4       format(a1,i3,':',i2,':',i2)
2       format(i4,':',i2,':',i2)
3       format(tr1,i3,':',i2,':',i2)
      call putcur(j/2+6,13+(i-1)*18,0,a1)
      call wchars(0,st,10,14,0,0,a1)
30      continue
20      continue
      end
```

```
      subroutine status (byte,bit,stat,*,*)
      integer a1,byte,bit
      logical stat
      real*8 xxx
      character*6 b
      character*1 d
      character*2 a
      data a /'¬W'/
      i=0
      icount=0
      call comlen(1,lk,ll,a1)
      call incom(1,b,lk,il,a1)
      call outcom(1,a,2,il,a1)
      call comlen(1,lk,ll,a1)
      if(lk.lt.6) then
             do 123 k=1,800
123              xxx=sin(1)**(cos(1)*tan(1))
             call comlen(1,lk,ll,a1)
             if(lk.eq.0) return1
      endif
      call incom(1,b,6,il,a1)
      call midchr(b,d,1,1)
      if (d.eq.'¬') then
            call incom(1,b,1,il,a1)
            return2
      endif
      call midchr(b,d,byte+1,1)
      stat=btest(ichar(d),bit)
      end
```

```
      subroutine sync
      character*27 ms
      character*1 a
      call cls
      call portot(768,0,2)
      call rubid(ihh,imm,iss)
      call portot(768,0,0)
2     call portin(768,0,isy)
      if(isy.ne.247) goto 2
      iss=iss+1
      if(iss.eq.60) then
            iss=0
            imm=imm+1
            if(imm.eq.60) then
                  imin=0
                  ihh=ihh+1
                  if(ihh.eq.24) ihh=0
            endif
      endif
      call settim (ihh,imm,iss,0)
      return
      end
```

```
      subroutine utname
      character*12 st
$include:'comm.for'
      call putcur(20,3,0,a1)
      call wchars(0,name,10,14,0,0,a1)
      call putcur(20,66,0,a1)
      write(st,5)iday,ihr,imin,isec
      call wchars(0,st,12,14,0,0,a1)
5       format(i3,1x,i2,':',i2,':',i2)
      call putcur(4,70,0,a1)
      if(stp) ons=0
      if(ons.eq.1) then
      write(st,'(a9)') 'ON SOURCE'
      else
      write(st,'(a9)') '         '
      endif
      call wchars(0,st,9,2,0,0,a1)
      end
```

```
      subroutine x
      integer k,m
      character ch
      character*8 cht
      dimension k(12)
$include:'comm.for'
      cht='¬X'
      l=0
      iz=paz*1000
10    do 30 i=1,6
            s=0
            do 20 j=1,i-1
                  s=s+k(j+l)*10**(6-j)
20          continue
            k(i+l)=(iz-s)/(10**(6-i))
30    continue
      if(l.eq.0) then
            l=6
            iz=pel*1000
            goto 10
      endif
      l=-2
      do 40 i=1,11,2
      k(i)=ishl(k(i),4)
      k(i)=iand(k(i),240)
      k(i)=k(i)+k(i+1)
      ch=char(k(i))
      m=i-l
      goto 50
60    l=l+1
40    continue
      goto 70
50      call concat(cht,ch,m,1)
      goto 60
70    call outcom(1,cht,8,il,a1)
      end
```

antcn.c

```
/* Input */
/* IP(1) = mode
        0 = initialize LU
        1 = pointing (from SOURCE command)
        2 = offset (from RADECOFF, AZELOFF, or XYOFF commands)
        3 = on/off source status (from ONSOURCE command)
        4 = direct communications (from ANTENNA command)
        5 = on/off source status for pointing programs
        6 = reserved for future focus control
        7 = log tracking data (from TRACK command)
  8 - 99 = reserved for future use
100 - 32767 = for site specific use

   IP(2) = class number (mode 4 only)
   IP(3) = number of records in class (mode 4 only)
   IP(4) - not used
   IP(5) - not used
*/

/* Output */
/*  IP(1) = class with returned message
      (2) = number of records in class
      (3) = error number
          0 - ok
         -1 - illegal mode
         -2 - timeout
         -3 - wrong number of characters in response
         -4 - interface not set to remote
         -5 - error return from antenna
         -6 - error in pointing model initialization
          others as defined locally
      (4) = 2HAN for above errors, found in FSERR.CTL
        = 2HST for site defined errors, found in STERR.CTL
      (5) = not used
*/

/* Defined variables */
#define MINMODE 0  /* min,max modes for our operation */
#define MAXMODE 7

/* Include files */
#include <stdio.h>
#include "../../fs/include/params.h" /* FS parameters             */
#include "../../fs/include/fs_types.h" /* FS header files         */
#include "../../fs/include/fscom.h"  /* FS shared mem. structure */
#include "../../fs/include/shm_addr.h" /* FS shared mem. pointer */
#include <ctype.h>

struct fscom *fs;

/* Subroutines called */
void setup_ids();
void putpname();
void skd_run(), cls_clr();
int nsem_test();
void logit();

int portopen_(), portread_(), portwrite_();
#define SIZE 256

/* antcn main program starts here */
main()
{
  int ierr, nrec, nrecr,flag;
  int dum = 0,iij,ih;
```

```
  int r1, r2,it[6],iii;
  int imode,i,nchar;
  long ip[5], class, clasr, mio, er,erf;
  char buf[80], buf2[100];

  static char sDev[]="/dev/ttyd2";
  long lBaud=9600;
  int iBits=8;
  int iStop=1;
  int iParity=2;
  int iPort;
  int iLen;
  int iErr;
  char sBuff[SIZE];
  char bif[2];
  int iMaxC=SIZE-1;
  int iMx=2;
  int iCount;
  int iTermCh='x';
  int iTo=150;

  iLen=strlen(sDev);
  iErr=portopen_(&iPort,sDev,&iLen,&lBaud,&iParity,&iBits,&iStop);

/* Set up IDs for shared memory, then assign the pointer to
   "fs", for readability.
*/
  setup_ids();
  fs = shm_addr;

/* Put our program name where logit can find it. */

  putpname("antcn");

/* Return to this point to wait until we are called again */

Continue:
  skd_wait("antcn",ip,(unsigned)0);

  imode = ip[0];
  class = ip[1];
  nrec = ip[2];

  if (imode < MINMODE || imode > MAXMODE) {
    ierr = -1;
    goto End;
  }

/* Handle each mode in a separate section */

  switch (imode) {

    case 0:              /* initialize */
      ierr = 0;
      strcpy(buf,"Initializing antenna interface");
      logit(buf,0,NULL);
      fs->ionsor = 0;
      break;

    case 1:              /* source= command */
      ierr = 0;
      strcpy(buf,"Commanding to a new source ");
      logit(buf,0,NULL);
      rep:rte_time(it,&it[5]);
      if((it[1]==0)||(it[1]==10)||(it[1]==20)||(it[1]==30)||(it[1]==40)||
      (it[1]==50)){rte_sleep(150);}
      fs->ionsor = 0;
```

```
  fs->RAOFF=fs->DECOFF=fs->AZOFF=fs->ELOFF=0;
  if (0 != (iErr=portflush_(&iPort))) {
   strcpy(buf,"flush error");
   logit(buf,0,NULL);}
  if ((fs->decdat) < 0 ) {strcpy(sBuff,"w");goto a;}
  strcpy(sBuff,"z");
  a: mio=((fs->radat))*1000000;
  int2str(sBuff,mio,-7,1);
  mio =((fs->decdat))*1000000;
  mio =abs(mio);
  int2str(sBuff,mio,-7,1);
  strncat(sBuff,(fs->lsorna),10);
  mio=(fs->RAOFF)*1000000;
  int2str(sBuff,mio,-7,1);
  mio=(fs->DECOFF)*1000000;
  int2str(sBuff,mio,-7,1);
  mio=(fs->AZOFF)*1000000;
  int2str(sBuff,mio,-7,1);
  mio=(fs->ELOFF)*1000000;
  int2str(sBuff,mio,-7,1);
  mio=0;
  int2str(sBuff,mio,-2,1);
  er=0;
  iii=0;
  do{
  if (isdigit(sBuff[iii])) er=er+toascii(sBuff[iii])-48;
  if (isalpha(sBuff[iii])){
  erf=toascii(sBuff[iii])-87;
  if (erf==32) erf=37;
  er=er+erf;}
  if (isspace(sBuff[iii])) er=er+36;
  }while (iii++<55);
  int2str(sBuff,er,-4,1);
  iLen=strlen(sBuff);
  if (0 != (iErr=portwrite_(&iPort, sBuff, &iLen))) {
   strcpy(buf,"write error");
   logit(buf,0,NULL);}
  strcpy(sBuff," ");
  iErr=portread_(&iPort,sBuff,&iCount,
       &iMaxC,&iTermCh,&iTo);
  if(sBuff[0] == ' ') {
   strcpy(buf,"Antenna Communications ERROR");
   logit(buf,0,NULL);}
  else if (toascii(sBuff[0]) != 6) {
   goto rep;}
  break;

case 2:              /* offsets          */
  ierr = 0;
  strcpy(buf,"Commanding new offsets");
  logit(buf,0,NULL);
  repo:rte_time(it,&it[5]);
  if((it[1]==0)||(it[1]==10)||(it[1]==20)||(it[1]==30)||(it[1]==40)||
  (it[1]==50)){rte_sleep(150);}
  fs->ionsor = 0;
  if (0 != (iErr=portflush_(&iPort))) {
   strcpy(buf,"flush error");
   logit(buf,0,NULL);}
  if ((fs->decdat) < 0 ) {strcpy(sBuff,"w");goto b;}
  strcpy(sBuff,"z");
  b: mio=(fs->radat)*1000000;
  int2str(sBuff,mio,-7,1);
  mio =(fs->decdat)*1000000;
  mio =abs(mio);
  int2str(sBuff,mio,-7,1);
  strncat(sBuff,(fs->lsorna),10);
  mio=(fs->RAOFF)*1000000;
```

```
   mio=abs(mio);
   int2str(sBuff,mio,-7,1);
   mio =(fs->DECOFF)*1000000;
   mio=abs(mio);
   int2str(sBuff,mio,-7,1);
   mio=(fs->AZOFF)*1000000;
   mio=abs(mio);
   int2str(sBuff,mio,-7,1);
   mio =(fs->ELOFF)*1000000;
   mio=abs(mio);
   int2str(sBuff,mio,-7,1);
   mio=0;
   if((fs->DECOFF)<0) {mio=mio+1;}
   if((fs->RAOFF)<0) {mio=mio+2;}
   if((fs->ELOFF)<0) {mio=mio+4;}
   if((fs->AZOFF)<0) {mio=mio+8;}
   int2str(sBuff,mio,-2,1);
   er=0;
   iii=0;
   do{
   if (isdigit(sBuff[iii])) er=er+toascii(sBuff[iii])-48;
   if (isalpha(sBuff[iii])){
   erf=toascii(sBuff[iii])-87;
   if (erf==32) erf=37;
   er=er+erf;}
   if (isspace(sBuff[iii])) er=er+36;
   }while (iii++<55);
   int2str(sBuff,er,-4,1);
   iLen=strlen(sBuff);
   if (0 != (iErr=portwrite_(&iPort, sBuff, &iLen))) {
    strcpy(buf,"write error");
    logit(buf,0,NULL);}
   strcpy(sBuff," ");
   iErr=portread_(&iPort,sBuff,&iCount,
        &iMaxC,&iTermCh,&iTo);
   if(sBuff[0] == ' ') {
    strcpy(buf,"Antenna Communications ERROR");
    logit(buf,0,NULL);}
   else if (toascii(sBuff[0]) != 6) {
    goto repo;}
   break;

 case 3:        /* onsource command with error message */
   ierr = 0;
   iTo=150;
   fs->ionsor = 0;
   if(0 != (iErr=portflush_(&iPort))){
         strcpy(buf,"flush error");
         logit(buf,0,NULL);}
   strcpy(bif," ");
   iErr=portread_(&iPort,sBuff,&iCount,
        &iMaxC,&iTermCh,&iTo);
   iij=0;
/*   do{printf("%d ",(toascii(sBuff[iij])));iij++;}while(iij<31);
   printf("\n"); */
   if(toascii(sBuff[0]) == 32) {
    strcpy(buf,"read error");
    logit(buf,0,NULL);}
   if (toascii(sBuff[0]) == 70) {
    fs->ionsor=0;}
   if (toascii(sBuff[0]) == 79) {
    fs->ionsor =1;}
   if (toascii(sBuff[0]) == 80) {
    fs->ionsor=0;
    strcpy(buf,"WARNING: Antenna stopped");
    logit(buf,0,NULL);}
   break;
```

```
    case 4:             /* direct antenna= command */
      ierr = 0;
      strcpy(buf,"No Available Command");
      logit(buf,0,NULL);
      break;

    case 5:     /* onsource command with no error logging */
      ierr = 0;
      iTo=150;
      fs->ionsor = 0;
      if(0 != (iErr=portflush_(&iPort))){
            strcpy(buf,"flush error");
            logit(buf,0,NULL);}
      strcpy(bif," ");
      iErr=portread_(&iPort,sBuff,&iCount,
           &iMaxC,&iTermCh,&iTo);
      if(toascii(sBuff[0]) == 32) {
       strcpy(buf,"read error");
       logit(buf,0,NULL);}
      if (toascii(sBuff[0]) == 70) {
       fs->ionsor=0;}
      if (toascii(sBuff[0]) == 79) {
       fs->ionsor =1;}
      if (toascii(sBuff[0]) == 80) {
       fs->ionsor=0;
       strcpy(buf,"WARNING: Antenna stopped");
       logit(buf,0,NULL);}
      break;

    case 6:             /* reserved */
      ierr = -1;
      goto End;
      break;

    case 7:     /* track command with additional info  */
      ierr = 0;
      iTo=150;
      fs->ionsor = 0;
      if(0 != (iErr=portflush_(&iPort))){
            strcpy(buf,"flush error");
            logit(buf,0,NULL);}
      strcpy(bif," ");
      if(0 != (iErr=portread_(&iPort,sBuff,&iCount,
           &iMaxC,&iTermCh,&iTo))){goto trackexit;}
      if(toascii(sBuff[0])!=79){
      if(toascii(sBuff[0])!=70){
       if(toascii(sBuff[0])!=80){goto trackexit;}}}
      iij=1;
/*      do{printf("%d ",(toascii(sBuff[iij])));iij++;}while(iij<44);
      printf("\n");*/
      mio=toascii(sBuff[1]);
      strcpy(buf," ");
      int2str(buf,mio,-2,1);
      strncat(buf,"/",1);
      mio=toascii(sBuff[2]);
      int2str(buf,mio,-1,1);
      mio=toascii(sBuff[3]);
      int2str(buf,mio,-2,1);
      strncat(buf,".",1);
      mio=toascii(sBuff[4]);
      int2str(buf,mio,-2,1);
      strncat(buf,":",1);
      mio=toascii(sBuff[5]);
      int2str(buf,mio,-2,1);
      strncat(buf,":",1);
      mio=toascii(sBuff[6]);
```

```
      int2str(buf,mio,-2,1);
      strncat(buf,".3 ",3);

      for (ih=7;ih<=9;ih++){mio=toascii(sBuff[ih]);
      int2str(buf,mio,-1,1);}
      strncat(buf,".",1);
      for (ih=10;ih<=12;ih++){mio=toascii(sBuff[ih]);
      int2str(buf,mio,-1,1);}
      strncat(buf," ",1);

      for (ih=13;ih<=15;ih++){mio=toascii(sBuff[ih]);
      int2str(buf,mio,-1,1);}
      strncat(buf,".",1);
      for (ih=16;ih<=18;ih++){mio=toascii(sBuff[ih]);
      int2str(buf,mio,-1,1);}
      strncat(buf," ",1);

      for (ih=19;ih<=21;ih++){mio=toascii(sBuff[ih]);
      int2str(buf,mio,-1,1);}
      strncat(buf,".",1);
      for (ih=22;ih<=24;ih++){mio=toascii(sBuff[ih]);
      int2str(buf,mio,-1,1);}
      strncat(buf," ",1);

      for (ih=25;ih<=27;ih++){mio=toascii(sBuff[ih]);
      int2str(buf,mio,-1,1);}
      strncat(buf,".",1);
      for (ih=28;ih<=30;ih++){mio=toascii(sBuff[ih]);
      int2str(buf,mio,-1,1);}
      strncat(buf," ",1);

      logit(buf,0,NULL);

      if(toascii(sBuff[0]) == 32) {
       strcpy(buf,"read error");
       logit(buf,0,NULL);}
      if (toascii(sBuff[0]) == 70) {
       fs->ionsor=0;}
      if (toascii(sBuff[0]) == 79) {
       fs->ionsor =1;}
      if (toascii(sBuff[0]) == 80) {
       fs->ionsor=0;
       strcpy(buf,"WARNING: Antenna stopped");
       logit(buf,0,NULL);}
/*
      strcpy(buf,"No Available Command");
      logit(buf,0,NULL);  */
      trackexit:;
      if(0 != (iErr=portflush_(&iPort))){
            strcpy(buf,"flush error");
            logit(buf,0,NULL);}
  }  /* end of switch */

End:
  ip[0] = clasr;
  ip[1] = nrecr;
  ip[2] = ierr;
  memcpy(ip+3,"AN",2);
  ip[4] = 0;
  goto Continue;
}
```

File tp.1_6 per il ricevitore 18cm

```
0.5947,0.0,0.0008,-0.0189,-0.0074,0.0075,0.4191,-0.1017
 18 cm
25.0,       25.0,       25.0,       0.0,  0.0,  0.0,  48.8, 48.1
z1off,      z2off,      z3off,      c1,   c2,   c3,   x,    y
```