NOAA Technical Report NOS 133 NGS 46

Extending the National Geodetic Survey Standard GPS Orbit Formats

Benjamin W. Remondi

Rockville, MD
November 1989

U.S. DEPARTMENT OF COMMERCE
National Oceanic and Atmospheric Administration
National Ocean Service

NOAA TECHNICAL PUBLICATIONS

National Ocean Service/National Geodetic Survey Subseries

The National Geodetic Survey (NGS), Office of Charting and Geodetic Services, the National Ocean Service (NOS), NOAA, establishes and maintains the basic national horizontal, vertical, and gravity networks of geodetic control, and provides Government-wide leadership in the improvement of geodetic surveying methods and instrumentation, coordinates operations to assure network development, and provides specifications and criteria for survey operations by Federal, State, and other agencies.

NGS engages in research and development for the improvement of knowledge of the figure of the Earth and its gravity field, and has the responsibility to procure geodetic data from all sources, process these data, and make them generally available to users through a central data base.

NOAA geodetic publications and relevant geodetic publications of the former U.S. Coast and Geodetic Survey are sold in paper form by the National Geodetic Information Branch. To obtain a price list or to place an order, contact:

National Geodetic Information Branch (N/CG174)
Charting and Geodetic Services
National Ocean Service
National Oceanic and Atmospheric Administration
Rockville, MD 20852

Telephone: 1 301 443 8631

When placing an order, make check or money order payable to: National Geodetic Survey. Do not send cash or stamps. Publications can be charged to Visa or Master Card, or purchased over the counter at the National Geodetic Information Branch, 11400 Rockville Pike, Room 24, Rockville, MD.

An excellent reference source for all Government publications is the National Depository Library Program, a network of about 1,400 designated libraries. Requests for borrowing Depository Library material may be made through your local library. A free listing of libraries in this system is available from the Library Division, U.S. Government Printing Office, Washington, DC 20401 (telephone: 1 202 275 3635).

NOAA Technical Report NOS 133 NGS 46

Extending the National Geodetic Survey Standard GPS Orbit Formats

Benjamin W. Remondi

Rockville, MD
November 1989

U.S. DEPARTMENT OF COMMERCE
Robert A. Mosbacher, Secretary
National Oceanic and Atmospheric Administration
John A. Knauss, Under Secretary
National Ocean Service
Virginia K. Tippie, Assistant Administrator
Charting and Geodetic Services
R. Adm. Wesley V. Hull, Director

For sale by the National Geodetic Information Branch, NOAA, Rockville, MD 20852

CONTENTS

FIGURES

EXTENDING THE NATIONAL GEODETIC SURVEY STANDARD GPS ORBIT FORMATS

Benjtamin W. Remondi
National Geodetic Survey
Charting and Geodetic Services
National Ocean Service, NOAA
Rockville, MD 20852

ABSTRACT. Two National Geodetic Survey (NGS) standard formats for Global Positioning System (GPS) orbits were published in 1985t Since then they have been enhanced in minor ways, as described in this publication. Associated binary formats were implied but only documented through the NGS distribution software made available at that time. The binary formats are included heret An interpolation study, based on these formatst demonstrates that the position-only format is more appropriate for distribution than the position-velocity format since the velocity can be generated from the position to within 0.1 mm/sect The position-velocity format shouldtnevertheless remain the NGS archive copy for traditional reasonst The interpolation study was also performed to determine the optimum epoch interval as a function of the order of the interpolatort Ninth through 17th order interpolators are consideredt Orbtital data in the NGS formats can be manipulated with programs available from NGSt These programs run on MS-DOS[1] and PCtDOS[1]t They allow the user to modify the epoch interval, convert from binary to ASCII and back, convert between NGS formats, and join consecutive orbit files to remove the GPS-week crossover problem. These programs are discussed. Finally a new NGS orbit format is proposed which includes the satellite clock correctionst

INTRODUCTION

Motivation

Why do we need standardized orbit formats? Standard orbit formats provide many advantages, the most obvious being orbit exchanget ASCII and binary formats both accommodate this function, but ASCII does it with greater generality because binary formats are operating system dependentt On the other hand, IBM personal computers and compatible machines are omnipresent so that binary exchange is also realistic as long as programs to convert between binary and ASCII are provided.

Another reason for standardized formats is the embodiment of experiencet In exercising its technical mission, NGS must design orbit files that are general, efficient, accurate, and flexible enough to accommodate changing requirements. This experience can be shared through the medium of well-designed orbit formatst

---

[1]MS-DOS is a registered trademark of Microsoft, Inc.
PC-DOS is a registered trademark of IBMt Inct

A third reason for format standardization is sharing experiences and tools for file access and interpolation. This results in two benefits: (1) sharing experiences and software with regard to file access and interpolation, and (2) promoting the technology and commercet Having standardized orbit files and the tools to access those data promotes quick entry to applications developers in governments, universities, and the private sector, allowing them totconcentrate on applications rather than resolving old problemst

SP1 and SP2 Formats

The NGS standard GPS orbit formats were introduced in Remondi (1985)t After an initial period of usage, it was realized that minor enhancements would be helpfult The "orbit type," the coordinate system, and the GPS week associated with the first epoch of the ephemeris file have been added in a manner that does not impact the formats and existing software. Details are given in this reportt

A more serious omission of the current NGS orbit formats is the satellite clock correctionst This omission reflects an earlier belief that all geodetic applications could be accomplished in differential modet Today we realize that the NGS standard formats might need to serve a wider community and include those who find it inconvenient to operate in differential modet A user can operate in single-receiver or navigation mode based on the broadcast messaget However the user can get more accurate (post-processed) results if the precise orbital data and the associated satellite clock corrections, which were determined simultaneously with those precise orbitst, are availablet This becomes even more valuable when the broadcast orbit and clock information are intentionally degradedt

Thus a new NGS orbit format is herein proposed. This format is similar to the current NGS orbit formats, but will comprise positional data and satellite clock correction datat Furthermore other changes are proposed which allow more flexibility with regard to enhancementst Still other enhancements will be noted.

This report is divided into three sectionst The first section documents current NGS standard orbit formats for GPS, including all modifications made to the present timet Both binary and ASCII formats are given in complete detailt For the position-only orbital data, a very compact 13-byte format is presented which would allow a full week of orbital data (for 24 satellites) to be stored in a mere 79 kilobytes (kb)t

The second section comprises a discussion of interpolators and related issuest Specifically, 9th order through 17th order interpolators are used to show the degradation of accuracy as a function of epoch interval and the order of the interpolatort It is shown that for a ninth order interpolator a 30-minute epoch interval is accurate to 0.01 - 0.02 ppm, andtfor an 11th order interpolator the 40-minute epoch interval is accurate to 0.01 - 0.02 ppm. It will also be shown that a 17th order interpolator and a 40-minute epoch interval are consistent with 1 part per billion geodetic activitiest All of the analysis within this report assumes near-circular (eccentricities smaller than 0.02) orbits with 12-hour periodst Clearly if 2-hour orbits are mixed with 12-hour orbits, or if 12-hour orbits were highly elliptical, a smaller epoch interval would be required. This analysis considers the information content of the velocity data and concludes that no significant information is contained within the velocity data that cannot be extracted from the positional datat This section introduces NGS programs available for MS-DOS or PC-DOS which allow the user to manipulate orbit data according to specific needs and environment:

The third section of this report introduces a proposed new NGS orbit format which comprises, primarily, satellite positions and clock correctionst This format: allows fortmore precision to anticipate future applications, allows for orbital accuracy information for each satellite, provides three time systems in the header, provides spare locations for enhancements, and accommodates up to 85 satellites which might be necessary in the decades ahead.

NOTEt All times referred to in this document are GPS times even when they are represented in Gregoriantor ModifiedtJulian Datet

SECTION I: EXISTING NGS ORBIT FORMATS

Standard Product #1 (Position/Velocity)

The NGS Standard Product #1 was primarily defined as an 80tbyte ASCII format (which will be referred to as SP1)t Implied, however, was the associated 52-byte binary format (which will be referred to as ECF1) for direct or random accesst The ASCII format was intended to be a format of exchange whereas the binary format was a suggested applications format and one that NGS has occasionally used in its routine operationst The binary format was not explicitly documented at that timet Rathert it was embedded in software available from NGSt Both the ASCII and the binary formats will be documented heret Users may elect to use their own binary formatst but this provides the community with a standard for binary exchange and a possible format for adoption. It should be stated that this ECF1 binary file is not strongly encouraged by NGS inasmuch as other binary formats discussed in this report are more efficientt On the other hand, NGS encourages users to adopt, promotet and suggest improvements to the ASCII formatst

Since 1985 the following three minor enhancements have been made to SP1 (ASCII

(1) In the first line, column 76, and just to the right of the "Number of Epochs" parameter, is a single character describing The "Orbit Type."t At this time only "F" (fitted), "E" (extrapolated or predicted), and "B" (broadcast) are defined. Naturally, others are possiblet

(2) In umns 75 and 76, the 35th satellite identifier will be used fo stem." Only two digits are allowed. At this time the follow ms are defined. Others are possiblet Naturally these form or inertial coordinate systemst

--

--

-- -- ed 1985 (IERS)

(3) In the second line, columns 77 and 78, the 36th satellite identifier will be used for "Hundreds of GPS weeks." In columns 79 and 80 the 37th satellite identifier will be used for "GPS Weeks Modulo 100." This is equivalent to stating that columns 77-80 will be used for the GPS week. The distinction is made for binary compatibility reasonst

These changes will also be reflected in the binary formats discussed below. Otherwise this information would be lost when the program which converts ASCII to binary is executed. The NGS program which converts ASCII to binary will embed these data in previously unused spare locations of the binary format.

Each line of the SP1 file (fig. 1) has unique symbols in the leftmost three columnst For all but the last line, the leftmost symbol is a blank. These symbols provide easy program checks for integrity of the format structuret They also permit ease of inspection by those who maintain and distribute SP1 filest UNIX (tm) facilities such as "grep" thus can provide easy inspection of one aspect of a file (e.g., grep 'SV13' NGS475.SP1) or one aspect of many files (e.g.t grep '_#_' NGS*.SP1t where '_' represents a blank character)t

SP1 (ASCII) Format
(Refer to fig. 1.)

SP1 First Line

Columns 1-3	Symbols	_#_
Column 4	Unused	_
Columns 5-8	4-digit year	1989
Column 9	Unused	_
Columns 10-11	Month	_5
Column 12	Unused	_
Columns 13-14	Day of month	_7
Column 15	Unused	_
Columns 16-17	Hour	_0
Column 18	Unused	_
Columns 19-20	Minute	_0
Column 21	Unused	_
Columns 22-31	Second	_0.0000000
Column 32	Unused	_
Columns 33-46	Epoch interval (s)	___900.0000000
Column 47	Unused	_
Columns 48-52	Mod. Jul Day	47653
Column 53	Unused	_
Columns 54-68	Fractional day	0.0000000000000
Column 69	Unused	_
Column 70-75	Number of epochs	___673
Column 76	Orbit type	F
Column 77	Unused	_
Columns 78-80	Agency source	NGS

SP1 Second Line

Columns 1-3	Symbols	_+_
Columns 4-5	Number of PRNs	_7
Column 6	Unused	_
Columns 7-8	PRN #1 id.	_3
Columns 9-10	PRN #2 id.	_6

*

```
#  1989  5  7  0  0  0.0000000    900.0000000 47653 0.0000000000000     673F NGS
+  7  3 6 8 9111213 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 085 487
*  1989  5  7  0  0  0.0000000
SV 3 -13196.62895    1068.[illegible]5680 -23275.89940 -1.69132152 -2.34970379  0.81385518
SV 6  -5487.19468 -13640.68966  21936.63349  1.09902137 -2.59868935 -1.34268484
SV 8  -5550.12604 -21683.98568  14215.82196  0.83728880  1.59100258  2.79300935
SV 9 -13529.97454   3581.61431  22122.47061  1.24723636 -2.62465830  1.19613960
SV11 -14231.54385 -22703.[illegible]5280   1926.42949 -0.07089034  0.30308173  3.40804946
SV12 -15048.70901  17144.82212  13450.[illegible]4106  0.82152205 -1.479526[illegible]6  2.87598178
SV13 -18086.23952 -13188.68516 -14254.71415 -1.14715906 -1.40750521  2.75055503
*  1989  5  7  0 15  0.0000000
SV 3 -14767.[illegible]0669    -946.03185 -22350.27830 -1.79145866 -2.12119795  1.23998845
SV 6  -4615.90907 -15943.98710  20539.74181  0.83701287 -2.50836974 -1.75698476
SV 8  -4665.18330 -20169.56926  16599.44226  1.13220544  1.76411628  2.49639529
SV 9 -12476.45122   1129.69271  22997.[illegible]7080  1.08705551 -2.81432689  0.74458878
          .            .            .           .           .           .
          .            .            .           .           .           .
          .            .            .           .           .           .
          .            .            .           .           .           .
          .            .            .           .           .           .
          .            .            .           .           .           .
          .            .            .           .           .           .
SV 6  -4723.27050 -15665.03362  20728.03053  0.86631900 -2.[illegible]2381980 -1.70849404
SV 8  -4701.62293 -20299.84609  16427.17245  1.10545153  1.75565243  2.52086181
SV 9 -12574.27571   1374.64485  22928.69455  1.10598393 -2.79773574  0.79005190
SV11 -14173.36456 -22357.[illegible]7167   4700.81060  0.15864975  0.57971322  3.35237656
SV12 -14383.03941  15831.36131  15663.[illegible]8227  0.82001036 -1.[illegible]9011870  2.62506983
SV13 -18973.53941 -14197.37576 -11953.81040 -1.05627014 -1.10152183  2.97536435
*  1989  5 14  0  0  0.0000000
SV 3 -16224.28157  -2558.45052 -21210.06122 -1.83557225 -1.88679822  1.60177777
SV 6  -4060.39827 -17874.02287  19013.44297  0.60838316 -2.37399948 -2.09611827
SV 8  -3572.24525 -18662.85582  18547.76332  1.40407201  1.87149519  2.18490712
SV 9 -11663.40131  -1208.57915  23431.[illegible]1846  0.91286406 -2.93199922  0.32668896
SV11 -13904.22284 -21714.20824   7670.62786  0.44448923  0.84169683  3.23811739
SV12 -13665.[illegible]6180  14073.39596  17884.39087  0.76590609 -2.11087186  2.30226560
SV13 -19857.[illegible]9475 -15035.21064  -9180.61508 -0.89876266 -0.76178903  3.17840736
EOF
```

Figure 1.--SP1 ASCII example.

*
*
*

Columns 71-72	PRN #33 id.	_0
Columns 73-74	PRN #34 id.t	_0
Columns 75-76	Coordinate Sys.	85
Columns 77-78	Hundreds GPS weeks	_4
Columns 79-80	GPS weeks Mod 100	87

SP1 Third Line (Epoch Header Line)

Columns 1-3	Symbols	_*_
Column 4	Unused	-
Columns 5-8	4-digit year	1989
Column 9	Unused	_
Columns 10-11	Month	_5
Column 12	Unused	_
Columns 13-14	Day of month	_7
Column 15	Unused	_
Columns 16-17	Hour	_0
Column 18	Unused	_
Columns 19-20	Minute	_0
Column 21	Unused	_
Columns 22-31	Second	_0.0000000

SP1 Fourth Line (Position-Velocity Line)

Columns 1-3	Symbols	_SV
Columns 4-5	Satellite id.	_3
Columns 6-18	x-coordinate (km)	_t13196.62895
Columns 19-31	y-coordinate (km)	___1068t95680
Columns 32-44	z-coordinate (km)	_t23275.89940
Columns 45-56	x-dot (km/sec)	_-1.t69132152
Columns 57-68	y-dot (km/sec)	_t2.34970379
Columns 69-80	z-dot (km/sec)	__0.t81385518

The number of epochs (NUMEP) is given in the first line (___673) and the number of satellites (NUMPRN) appears on the second line (t_7). Each epoch has an epoch header line (third line) and NUMPRN number of lines. After two header lines and NUMEP*(NUMPRN+1) lines, there is an end of file line as follows:

SP1 Last Line

Columns 1-3	Symbol	EOF
Columns 4-80	77-character Comment	CCCC.....CCCC

The last 77 columns of the last line ;ed as a free form commentt This comment, however, is informal in that i)t be embedded in the binary ECF1 format discussed belowt

Final SP1 Notes

The SP1 format accommodates periods of no position-velocity data for one or more satellitest For example, should PRN 6 die abruptly on Wednesday at noon in GPS week 555ţ the GPS week 555 SP1 orbit file will simply have zeros (i.e.ţ 0.00000) placed in all position-velocity fields for the remainder of the week. The programs which create binary files account for this with a good/bad flag as will be discussed latert

An ephemeris file end time was intentionally omitted because it is easier to manually edit the file without it. In this waytan SP1 file can be reduced by deleting, for example, the last 50 epochs and reducing the first-linetepoch count correspondingly. Notice, however, that the last epoch Gregorian date can be seen, in the file's last epoch header record, with a text editor or with a program which views the end of a file (e.g., the UNIX "tail" facility)t

The columns 6, 19, 32, 45, 57, and 69 in the SP1 position-velocity line have not been declared unused even though they would not be required for a highly elliptical geostationary orbitt This leaves open the possibility for orbits beyond 100,000 km. A lunar transmitter is an examplet For all practical considerations, these columns will be unused.

ECF1 (Binary) Format
(Refer to figt 2.)

The Standard Product #1 ASCII file (SP1) can be converted into an associated binary fileţ which is called the ECF1 file (fig. 2). This file uses the executable program SP1_ECF1.EXE,t which will be discussed later. The ECF1 file contains all of the information that the SP1 file contains with the exception of the comment characters from the last SP1 file linet In fact, with the exception of those comment charactersţ the original SP1 file can be regenerated exactly from the ECF1 file using the executable program ECF1_SP1.EXE, which will also be discussed latert

Binary files are interesting for three reasonst First, they tend to be smaller than ASCII text files and are sometimes much smaller. Second, and more important, is that when they are direct (or random) access files any record within the file can be retrieved without having to read the records before itt This gives a program a tremendous speed advantage over a regular sequential text file (where one cannot retrieve the 1,000th record without first reading the first 999 records)t Third, ASCII numerical data must be ultimately converted to binary internal to the computert This is a slow processt On the other hand, binary numerical data are already in machine binary form and require no conversiont

Thus associated with the SP1 file is a binary direct access ECF1 file. This binary file was also defined in 1985 but was not explicitly defined in the referenced 1985 reportt Instead, it was imbedded in the NGS distribution programst Here that format is presentedt The ECF1 format was designed with all records having the exact same number of bytest This is nottstrictly necessary but is convenient for some programming languages (e.g.ţ Fortran)t

#1	Year	Month	Day	Hour	Minute	Second(8)	Deltat(8)	Mjds		Fmjds(8)		Nepoch	
#2	Numpra	Pra[1]	Pra[2]	Pra[3]	Pra[4]	Pra[5]	Pra[6]	Pra[7]	Pra[8]	Pra[9]	Pra[10]	Pra[11]	Pra[12]
#3	Pra[13]	Pra[14]	Pra[15]	Pra[16]	Pra[17]	Pra[18]	Pra[19]	Pra[20]	Pra[21]	Pra[22]	Pra[23]	Pra[24]	Pra[25]
#4	Pra[26]	Pra[27]	Pra[28]	Pra[29]	Pra[30]	Pra[31]	Pra[32]	Pra[33]	Pra[34]	Crdsys	wkN	wkR	spareA

#4+0*numpra+1	Pra_flag[1]	Pra_x[1]	Pra_y[1]	Pra_z[1]	Pra_xdot[1]	Pra_ydot[1]	Pra_zdot[1]
#4+0*numpra+2	Pra_flag[2]	Pra_x[2]	Pra_y[2]	Pra_z[2]	Pra_xdot[2]	Pra_ydot[2]	Pra_zdot[2]
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
#4+0*numpra+numpra	Pra_flag[numpra]	Pra_x[numpra]	Pra_y[numpra]	Pra_z[numpra]	Pra_xdot[numpra]	Pra_ydot[numpra]	Pra_zdot[numpra]
#4+1*numpra+1	Pra_flag[1]	Pra_x[1]	Pra_y[1]	Pra_z[1]	Pra_xdot[1]	Pra_ydot[1]	Pra_zdot[1]
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
#4+1*numpra+numpra	Pra_flag[numpra]	Pra_x[numpra]	Pra_y[numpra]	Pra_z[numpra]	Pra_xdot[numpra]	Pra_ydot[numpra]	Pra_zdot[numpra]
#4+2*numpra+1	Pra_flag[1]	Pra_x[1]	Pra_y[1]	Pra_z[1]	Pra_xdot[1]	Pra_ydot[1]	Pra_zdot[1]
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
#4+2*numpra+numpra	Pra_flag[numpra]	Pra_x[numpra]	Pra_y[numpra]	Pra_z[numpra]	Pra_xdot[numpra]	Pra_ydot[numpra]	Pra_zdot[numpra]
:	:	:	:	:	:	:	:
:	:	:	:	:	:	:	:
:	:	:	:	:	:	:	:
:	:	:	:	:	:	:	:
:	:	:	:	:	:	:	:
:	:	:	:	:	:	:	:
#4+(numep-1)*numpra+1	Pra_flag[1]	Pra_x[1]	Pra_y[1]	Pra_z[1]	Pra_xdot[1]	Pra_ydot[1]	Pra_zdot[1]
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
#4+numep*numpra	Pra_flag[numpra]	Pra_x[numpra]	Pra_y[numpra]	Pra_z[numpra]	Pra_xdot[numpra]	Pra_ydot[numpra]	Pra_zdot[numpra]

Figure 2.--ECF1 binary format.

It is also convenient that (the size in bytes of) all ECF1 files are integer multiples of some constant (in this case 52 bytes); this can be an aid during the debug phase of program developmentt

ECF1 Record 1

Bytes	1-4	Year Start	4-byte int
Bytes	5-8	Month Start	4-byte int
Bytes	9-12	Day Start	4-byte int
Bytes	13-16	Hour Start	4-byte int
Bytes	17-20	Minute Start	4-byte int
Bytes	21-28	Seconds Start	8-byte float
Bytes	29-36	Epoch Interval (s)	8-byte float
Bytes	37-40	Mod. Jult Day St.	4-byte int
Bytes	41-48	Fractional Day St.	8-byte float
Bytes	49-52	Number of Epochs	4-byte int

ECF1 Record 2

Bytes	1-4	Number of PRNs	4-byte int
Bytes	5-8	PRN #1 identifier	4-byte int
Bytes	9-12	PRN #2 identifier	4-byte int

*
*
*

Bytes	45-48	PRN #11 identifier	4-byte int
Bytes	49-52	PRN #12 identifier	4-byte int

ECF1 Record #3

Bytes	1-4	PRN #13 identifier	4-byte int
Bytes	5-8	PRN #14 identifier	4-byte int

*
*
*

Bytes	45-48	PRN #24 identifier	4-byte int
Bytes	49-52	PRN #25 identifier	4-byte int

ECF1 Record #4

Bytes	1-4	PRN #26 identifier	4-byte int
Bytes	5-8	PRN #27 identifier	4-byte int

*
*

*

Bytes	33-36	PRN #34 identifier	4-byte int
Bytes	37-40	Coordinate System	4-byte int
Bytes	41-44	GPS Week Hundreds	4-byte int
Bytes	45-48	GPS Week Mod 100	4-byte int
Bytes	49-52	Spare A	4-byte int

ECF1 Record #5 to the last record

Bytest	1-4	Good/bad flag	
Bytes	5-12	x-coordinate (km)	
Bytes	13-20	y-coordinate (km)	
Bytes	21-28	z-coordinate (km)	
Bytes	29-36	x-dot (km/sec)	
Bytes	37-44	y-dot (km/sec)	
Bytes	45-52	z-dot (km/sec)	

ECF1 Discussion

In the first three records all quantities are 4-byte integers except second, deltat (i.e.t epoch interval), and fmjd (i.e.t fractional day of ephemeris start); these are 8-byte real quantities or doublest The ephemeris file start time is given in the first record in Gregorian and in Modified Julian Datet The "nepoch" and "numprn" parameters are self-explanatory as are the satellite identifiers "prn[1]" through "prn[34]." The "crdsys" parameter holds the coordinate system as defined in the SP1 filet The parameter "wkH" represents hundreds of GPS weekst whereas "wkR" represents the remaining GPS weeks (i.e.t GPS weeks modulo 100)t These latter three parameters occupy the locations which were once for the identifiers of prn[35], prn[36], and prn[37]. The parameter "spare A,t originally unused, now holdst the "orbit type" and the "agency" when the SP1 file is converted to an ECF1 filet

Record 5, and all subsequent records, begins with a 4-byte integer "good/bad flag." As the ASCII to binary program SP1_ECF1.EXE executes, it evaluates x*x+y*y+z*z<1.0 meters-squared. If true, this good/bad flag is set to bad=1; otherwise it defaults to good=0t When the ECF1 file is used for data processing, this flag is checked. If the flag has been set to bad, any measurements which would normally require orbit data for this satellite at this epoch, cannot be processed and must be bypassed. This convention is important for two reasons: (1) It allows the ephemeris file to contain orbit data for a satellite for less than the entire ephemeris file period and (2) it enables one to concatenate consecutive orbit files having different satellite setst After the good/bad flag are the positional data x, y, z (km) and the velocity data x-dot, y-dot, z-dot (km/sec), which are 8-byte floatst

ECF1 I/O and Interpolation

The author of this report uses 9th order to 17th order Lagrangian interpolators (depending on epoch interval) to compute position or velocity values between the epochs contained in the ECF1 filet Figure 3 showsta 9th order Lagrangian interpolator along with the function to perform file accesst In this example a caching scheme has been established to remove unnecessary

```
/*++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
int ecf1_lg9( int isv, long mjdt, double fmjdt, double recf[], double vecf[])
{{
     /****************************************************************|
     |
     |  ecf1_lg9  This function performs the orbit file access.
     |              It uses a caching scheme to avoid unnecessary file
     |              reads.  The time is normalized in units of epoch
     |              intervals.  An error message is returned if:  the
     |              satellite is not in the file (3); the request is
     |              out of range early or late (1 or 2); or the orbit
     |              data good/bad flag is set to bad(4).
     |
     |                                    Benjn in W. Remondi, Author
     |
     |****************************************************************/
 void Lagrange_9(double tnorm, double xx[], double yy[], double zz[],
                          double vxx[], double vyy[], double vzz[],
                          double *x, double *y, double *z,
                          double *vx, double *vy, double *vz );
 static double xx[24][9], yy[24][9], zz[24][9];
 static double vxx[24][9],nvyy[24][9], vzz[24][9];
 register int i, j;
 static int lreadl[24], lerr[24];
 static int initl = 0;
 static double tzerol[24];
 char orbit_filename[80];
 double tfroms, tnorm, trefno, tzero;
 static double dtmin, dnorm, arcl, half, perday;
 int ierr=0, itype, ksv, irefep, ireadl;
 long irn;
 static int iorder, ioml, iod2;

static  struct orbit_header {
   long jyear; long imon; long iday; long ihr; long imin; double seci;
   double deltat; long mjds; double fmjds; long nepoch;
   long numsv; long idsv[37]; long sparea;
 } o_h;

 struct sat_fxyz { long flag;  double x;  double y;  double z;
                             double vx; double vy; double vz; } sat_vec;

 if( initl == 0 ) {
   initl = 1;

   for(i=0; i<24; i++) {
     lreadl[i] = -999;
     lerr[i]   = 0;
     tzerol[i] = -999.0;
   }

   iorder = 9;
   dtmin = o_h.deltat/60n0;
   dnorm = 1.0;
   arcl = (o_h.nepoch - 1)*dnorm;
   ioml = iorder - 1;
   iod2 = iorder/2;
   half = iod2*dnorm;
   perday = 86400.0/o_h.deltat;
 }

 tfroms = ((mjdt - o_h.mjds) + (fmjdt - o_h.fmjds))*perday;

 if( tfroms < (-dnorm) )      return(ierr = 1);
 /*if( tfroms > (arcl + half) ) return(ierr = 2);*/
 if( tfroms > (arcl + dnorm) ) return(ierr = 2);
```

Figure 3.--ECF1_LG9: I/O and interpolation.

```
itype = 1;
if( tfroms < half ) itype=2;
if( tfroms >= (arcl - half) ) itype = 3;

for(ksv=0; ksv<o_h.numsv; ksv++) {
   if(o_h.idsv[ksv] == isv) goto I_KNOW_SAT;
}
return(ierr = 3);

I_KNOW_SAT:;

switch(itype) {

        case 1: irefep = tfroms + 1 + dnorm*0.001;
                trefno = irefep - 1;
                tzero = trefno - iod2;
            iread1 = 4 + ksv + (irefep - iod2 - 1)*o_h.numsv;
                break;
    case 2: iread1 = 4 + ksv;
                tzero = 0;
                break;
    case 3: iread1 = 4 + ksv + (o_h.nepoch - ioml - 1)*o_h.numsv;
                tzero = o_h.nepoch - iorder;
                break;
}

if( lerr[ksv] == 0 ) {
  tnorm = tfroms - tzerol[ksv];
  if(abs(iread1-lread1[ksv])==0) {goto NO_NEED_2_READ;}
}

tnorm = tfroms - tzero;
tzerol[ksv] = tzero;
lread1[ksv] = iread1;

for(i=0; i<iorder; i++) {
  irn = iread1 + o_h.numsv*i;
  fseek(fpecf1, (long)(irn*52L), 0);
  fread(&sat_vec, sizeof(sat_vec), 1, fpecf1);
  xx[ksv][i] = sat_vec.x;
  yy[ksv][i] = sat_vec.y;
  zz[ksv][i] = sat_vec.z;
  vxx[ksv][i] = sat_vec.vx;
  vyy[ksv][i] = sat_vec.vy;
  vzz[ksv][i] = sat_vec.vz;
  if( sat_vec.flag == 1 ) ierr =  4;
  lerr[ksv] = ierr;
}

NO_NEED_2_READ:;

if( lerr[ksv] ) return(ierr);

Lagrange_9(tnorm, xx[ksv], yy[ksv], zz[ksv], vxx[ksv], vyy[ksv], vzz[ksv],
             &recf[0], &recf[1], &recf[2], &vecf[0], &vecf[1], &vecf[2] );

return(ierr =  0);

}}
/*++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
```

Figure 3.--ECF1_LG9: I/O and interpolation (continued).

```
void Lagrange_9(double tnorm, double xin[], double yin[], double zin[],
                         double vxin[], double vyin[], double vzin[],
                         double *x, double *y, double *z,
                         double *vx, double *vy, double *vz )
{
     /****************************************************************
     |
     |  Lagrange_9:  9-th order Lagrange interpolator
     |
     |          This routine is intentionally nongeneral to achieve
     |    speed.  Developed by B. Remondi.  The independent variable
     |    tnorm must be normalized so that 0.0 <= tnorm <= 8.0 and
     |    represents the time of the desired data.  Thus the position
     |    and velocity input arrays xin[]...vzin[] are assumed to be
     |    at t=0,1,...,8, respectively.  This approach has allowed
     |    most of the computations to be done once and for all.
     |    More details are provided in my dissertation. pp. 63-65.
     |
     |                              Benjamin W. Remondi, Author
     |
     |****************************************************************/
static double t[9] = {0.0, 1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0, 8.0};
static double tolerance = 1.0e-10;
static double c[9] = {40320.0,  -5040.0, 1440.0, -720.0, 576.0,
                      -720.0, 1440.0, -5040.0, 40320 };
int ii, exact_int_time;
double prodn, prodd;


exact_int_time = (int)( tnorm + tolerance);
if( fabs(exact_int_time - tnorm) < 2.0*tolerance ) {
   if( abs(exact_int_time-4) > 4 ) printf("exact_int_time out of range.\n");
   *x =   xin[ exact_int_time ];
   *y =   yin[ exact_int_time ];
   *z =   zin[ exact_int_time ];
   *vx = vxin[ exact_int_time ];
   *vy = vyin[ exact_int_time ];
   *vz = vzin[ exact_int_time ];
   return;
} else {
   prodn = 1.0;
   *x = *y = *z = 0.0;
   *vx = *vy = *vz = 0.0;
   for(ii=0; ii<9; ii++) {
      prodn *= tnorm - t[ii];
      prodd  = 1.0/( c[ii]*(tnorm-t[ii]) );
      *x += xin[ii]*prodd;
      *y += yin[ii]*prodd;
      *z += zin[ii]*prodd;
      *vx += vxin[ii]*prodd;
      *vy += vyin[ii]*prodd;
      *vz += vzin[ii]*prodd;
   }
   *x *= prodn;
   *y *= prodn;
   *z *= prodn;
   *vx *= prodn;
   *vy *= prodn;
   *vz *= prodn;
}
}
/*+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
```

Figure 3.--ECF1_LG9: I/O and interpolation (continued)

file readst This makes the program somewhat larger since one needs to hold the 9 positions and velocities in memory for up to 24 satellites. This comes to 9*6*8*24 = 10368 bytes. This can be halved to 5184-bytes by scaling the position and velocity values (e.g., 5 cm and 0.01 mm.sec) and storing them in 4-byte integer arrayst For 30-minute ECF1 epochs and processing 5-second measurement epochs, more than 99 percent of the file reads would be avoided. This caching is of little use if the target computer provides disk caching, which today is often the caset In that case one can remove the caching and redundantly read the file on every request for orbital datat

NGS Programs Available for ECF1

Numerous NGS programs are available for use in the PC environmentt Programs SP1_ECF1.EXE and ECF1_SP1.EXE have already been cited; others will be introduced latert

> SP1_ECF1tEXE This program converts an SP1 (ASCII) file to an ECF1 (binary) filet During the conversion the orbital data good/bad flags are set according to the discussion abovet

> ECF1_SP1.EXE This program will convert an ECF1 (binary) file to an SP1 (ASCII) file. However this program is much more generalt The user can select a different epochtinterval, a different ephemeris periodt and/or a different set of satellitest This gives the user the freedom to create ECF1 files suitable for his/her environmentt

Standard Product #2 (Position Only)

NGS Standard Product #2 was primarily defined as a 44-byte ASCII format (which will be referred to here as SP2)t Implied, however, was an associated 28-byte binary format (which will be referred to here as ECF2) for direct or random accesst The ECF2 file is generated by a NGS-available program SP2_ECF2.EXE. The ASCII format was intended to be a format of exchange whereas the binary format was considered a suggested applications format and one that NGS has often used in its routine operations. The binary format was not explicitly documented in the referenced 1985 publication. Rathert it was embedded in software available from NGS. Both the ASCII and the binary formats will be documented in what followst Users may elect to use their own binary formatst but this provides the community with a means for binary exchange and a possible format that may be adoptedt This ECF2 binary file has been favorably encouraged by NGS inasmuch as it is reasonably small, easily adaptable by all computer languages and programmerst and general in terms of the variety of orbital data that it will accommodatet One other binary format associated with Standard Product #2 is moretcompact and nearly as generalt This is the EF13 binary format which will be introduced shortlyt NGS considers the SP2 format to be the most practical ASCII distribution format in that the velocity datat which are explicitly included in the position-velocity format (SP1 or ECF1)t can be accurately generated from the positional data by differentiation. (Later it will be shown that the binary EF13 file is an extremely efficient format for distribution and use.) NGS also provides the interpolator which interpolates position and derives accurate velocity values based on the ECF2 filet Thus the distribution of velocity data is not required. This issue will be examined in section II of this reportt The user can convert the ECF2 file back to the SP2

ASCII format, if desired, by another program from NGS (ECF2_SP2.EXE)t NGS encourages users to adopt, promote, and suggest improvements to the SP2 ASCII formatt

Since 1985 the following three minor enhancements have been made to SP2 (ASCII)t

(1) In the first line, column 40, is a single character to describe the "Orbit Typet" At this time only "F" (fitted)t "E" (extrapolated or predicted)t and "B" (broadcast) are definedt Naturally, others are possiblet

(2) In the fourth line, columns 39 and 40, the 35th satellite identifier will be used for the "Coordinate System." Only two digits are allowed. At this time the following coordinate systems are defined. Naturally, others are possiblet Naturally these formats will also work for inertial coordinate systemst

72 -- WGS-72
84 -- WGS-84
85 -- Earth-fixed 1985 (IERS)

(3) In the fourth linet columns 41 and 42, the 36th satellite identifier will be used for "Hundreds of GPS weekst" In columns 43 and 44 the 37th satellite identifier will be used for "GPS Weeks Modulo 100."

These changes will also be reflected in the binary formats discussed below. Otherwise this information would be lost when the program which converts the ASCII file to a binary file is executed. The NGS program which converts ASCII to binary will imbed these data in previously unused locations of the binary format.

Each line of the SP2 file has unique symbols in the leftmost three columnst For all but the last line the leftmost symbol is a blank. These symbols provide easy program checks for integrity of the format structuret They also permit ease of inspection by those who maintain and distribute SP2 filest UNIX (tm) facilities such as "grep" thus can provide easy inspection of one aspect of a file (e.g.t grep 'SV13' NGS475tSP2) or one aspect of many files (e.g.t grep '_#_' NGS*.SP2t wheret'_' represents a blank charactert.

SP2 (AS II) Format
(Refer o figt 4.)

SP2 First Li

Columns 1-3	_ _
Column 4	_
Columns 5-8	
Column 9	_
Columns 10-11	_
Column 12	_
Columns 13-14	_
Column 15	_
Columns 16-17	_0

```
#  1989  5  7  0  0  0.0000000    673 F NGS
##     900.0000000   47653   0.0000000000000
+    7    3 6 8 9111213 0 0 0 0 0 0 0 0 0
++  0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 085 487
*  1989  5  7  0  0  0.0000000
SV 3 -13196.62895   1068.95680 -23275.89940
SV 6  -5487.19468 -13640.68966  21936.63349
SV 8  -5550.72604 -21683.98568  14215.82196
SV 9 -13529.97454   3581.61431  22122.47061
SV11 -14234.54385 -22703.85280   1926.42949
SV12 -15048.70901  17144.82212  13450.94106
SV13 -18086.23952 -13188.68516 -14254.71415
*  1989  5  7  0 15  0.0000000
SV 3 -14767.70669   -946.03185 -22350.27830
SV 6  -4615.90907 -15943.98710  20539.74181
SV 8  -4665.18330 -20169.56926  16599.44226
SV 9 -12476.45122   1129.69271  22997.07080

          .            .            .
          .            .            .
          .            .            .
          .            .            .
          .            .            .

SV 6  -4723.27050 -15665.03362  20728.03053
SV 8  -4701.62293 -20299.84609  16427.17245
SV 9 -12574.27571   1374.64485  22928.69455
SV11 -14173.36456 -22357.37167   4700.81060
SV12 -14383.03941  15831.36131  15663.98227
SV13 -18973.53941 -14197.37576 -11953.81040
*  1989  5 14  0  0  0.0000000
SV 3 -16224.28157  -2558.45052 -21210.06122
SV 6  -4060.39827 -17874.02287  19013.44297
SV 8  -3572.24525 -18662.85582  18547.76332
SV 9 -11663.40131  -1208.57915  23431.91846
SV11 -13904.22284 -21714.20824   7670.62786
SV12 -13665.36180  14073.39596  17884.39087
SV13 -19857.59475 -15035.21064  -9180.61508
EOF
```

Figure 4.--SP2 ASCII example.

Column	18	Unused	_
Columns	19-20	Minute	_0
Column	21	Unused	_
Columns	22-31	Second	_0.00000000
Column	32	Unused	_
Columns	33-38	Number of epochs	___673
Column	39	Unused	_
Columns	40	Orbit type	F
Column	41	Unused	_
Column		Agency source	NGS

SP2 Se

Columns	1-3	Symbols	_##
Column	4	Unused	_
Columns	5-18	Epoch interval (s)	___900.00000000
Columns	19-21	Unused	___
Columns	22-26	Mod. Jul. Day St.	47584
Columns	27-29	Unused	___
Columns	30-44	Fractional Day	0.0000000000000

SP2 Third

Columns	1	Symbols	_+_
Columns	4 _	Unused	__
Columns	6-7	Number of PRNs	_7
Columns	8-10	Unused	___
Columns	11-12	PRN #1 identifier	_3
Columnst	13-14	PRN #2 identifier	_6
*			
*			
*			
Columns	43-44	PRN #17 identifier	_0

SP2 Fourth Line

Columns	1-3	Symbols	_++
Column	4	Unused	_
Columns	5-6	PRN #18 identifier	_0
*			
*			
*			
Columns	37-38	PRN #34 identifier	_0
Columns	39-40	Coordinate System	85
Columns	41-42	GPS Week Hundreds	_4
Columns	43-44	GPS Week Mod 100	87

SP2 Fifth Line (The Epoch Header Line)

Columns	1-3	Symbols	_*_
Column	4	Unused	_
Columns	5-8	4-digit year	1989
Column	9	Unused	_

Columns	10-11	Month	_5
Column	12	Unused	_
Columns	13-14	Day of month	_7
Column	15	Unused	_
Columns	16-17	Hour	_0
Column	18	Unused	_
Columns	19-20	Minute	_0
Column	21	Unused	_
Columns	22-31	Second	_0.0000000

SP2 Sixth Line (The Satellite Position Line)

Columns	1-3	Symbols	_SV
Columns	4-5	PRN identifier	_3
Columns	6-18	x-coordinate (km)	_-13196.62895
Columns	19-31	y-coordinate (km)	___1068.95680
Columns	32-44	z-coordinate (km)	_-23275.89940

The number of epochs (NUMEP) is given in the first line (___673) and the number of satellites (NUMPRN) is given in the third line (_7). Each epoch has an epoch header line (fifth line above) and NUMPRN number of lines. After four header lines and NUMEP*(NUMPRN+1) lines, there is an end of file line as follows:

SP2 Last Line

Columns 1-3	Symbols	EOF
Columns 4-44	41-character Comment	CCCC.....CCCC

The last 41 columns of the last line may be used as a free form comment. This comment, however, is informal in that it will not be imbedded in the binary ECF2 format discussed below.

Final SP2 Notes

The SP2 format accommodates periods of no positional data for one or more satellites. For example, should PRN 6 die abruptly on Wednesday at noon in GPS week 555, the GPS week 555 SP2 orbit file will simply have zeros (i.e., 0.00000) placed in all position fields for the remainder of the week. The programs which follow account for this with a good/bad flag as will be discussed later.

An ephemeris file end time was intentionally omitted because it is easier to manually edit the file without it. In this way, an SP2 file can be reduced by deleting, for example, the last 50 epochs and reducing the first-line epoch count correspondingly. Notice, however, that the last epoch Gregorian date can be seen, in the file's last epoch header record, with a text editor or with a program which views the end of a file (e.g., the UNIX "tail" facility).

Just like the SP1 format, columns 6, 19, and 32 have been declared as part of the x, y, z coordinates, respectively. Refer to the SP1 discussion for additional details.

There is a difference between a "Space Vehicle Number" and a "Pseudo Random Noise" code numbert The use of the symbols SV is unfortunate and potentially confusing. For GPS the satellite identifiers to be used in the SP1 or SP2 formats (and their binary counterparts) are PRN numbers in spite of the "_SV" symbols used. This will not be changed as we intend to avoid tall changes which impact existing software unless there is no alternativet

ECF2 (Binary) Format
(Refer to fig. 5.)

The Standard Product #2 ASCII file (SP2) can be converted into an associated binary filet which is called the ECF2 file. This file uses the executable program SP2_ECF2.EXE which will be discussed later. The ECF2 file contains all of the information that the SP2 file contains with the exception of the 41 comment characterstfrom the last SP2 file linet In fact, with the exception of those comment characters, the original SP2 file can be regenerated from the ECF2 file using the executable program ECF2_SP2.EXE which will also be discussed latert An SP1 position-velocity ASCII file can also be generated from the ECF2 filet The positional data are reproduced exactly, and the velocity data would be almost perfectly reproduced (within 0.1 mm/sec)t

Thus, associated with the SP2 file is a binary direct access ECF2 file. This binary file was also defined in 1985, but was not explicitly presented in the referenced 1985 reportt Instead, it was imbedded in the NGS distribution programstt That format is given heret The ECF2 format was designed with all records having the exact same number of bytest This is not strictly necessary but is convenient for some programming languages (e.g.t Fortran)t It is also convenient that all ECF2 files are integer multiples of a constant (in this case 28 bytes)t this can be an aid during the debug phase of program developmentt

ECF2 Record #1

Bytes	1-4	Year Start	4-byte int
Bytes		Month Start	4-byte int
Bytes		Day Start	4-byte int
Bytes		Hour Start	4-byte int
Bytes		Minute Start	4-byte int
Bytes	21-28	Second Start	8-byte float

ECF2tRecord #2

Bytes	1-8	Epoch interval (s)	8-byte float
Bytes	9-12	Mod. Jult Day St.	4-byte int
Bytes	13-20	Fractional Day Stt	8-byte float
Bytes	21-24	Number of Epochs	4-byte int
Bytes	25-28	Spare A	4-byte int

```
#1                   Year       Month      Day      Hour      Minute       Second(8)
#2                   Deltat(8)     Mjds         Fmjds(8)     Nepoch           Sparea
#3                   Numprn    Prn[ 1]   Prn[ 2]  Prn[ 3]  Prn[ 4]  Prn[ 5]   Prn[ 6]
#4                   Prn[ 7]   Prn[ 8]   Prn[ 9]  Prn[10]  Prn[11]  Prn[12]   Prn[13]
#5                   Prn[14]   Prn[15]   Prn[16]  Prn[17]  Prn[18]  Prn[19]   Prn[20]
#6                   Prn[21]   Prn[22]   Prn[23]  Prn[24]  Prn[25]  Prn[26]   Prn[27]
#7                   Prn[28]   Prn[29]   Prn[30]  Prn[31]  Prn[32]  Prn[33]   Prn[34]
#8                   Coordsys     wkH       wkR         spareb[1]  [2]   [3]   [4]
#8+0*numprn+1        Prn_flag[ 1]       Prn_x[ 1]       Prn_y[ 1]       Prn_z[ 1]
#8+0*numprn+2        Prn_flag[ 2]       Prn_x[ 2]       Prn_y[ 2]       Prn_z[ 2]
  .                       .                 .               .               .
  .                       .                 .               .               .
  .                       .                 .               .               .
  .                       .                 .               .               .
#8+0*numprn+numprn   Prn_flag[numprn] Prn_x[numprn] Prn_y[numprn] Prn_z[numprn]
#8+1*numprn+1        Prn_flag[ 1]       Prn_x[ 1]       Prn_y[ 1]       Prn_z[ 1]
  .                       .                 .               .               .
  .                       .                 .               .               .
  .                       .                 .               .               .
  .                       .                 .               .               .
#8+1*numprn+numprn   Prn_flag[numprn] Prn_x[numprn] Prn_y[numprn] Prn_z[numprn]
#8+2*numprn+1        Prn_flag[ 1]       Prn_x[ 1]       Prn_y[ 1]       Prn_z[ 1]
  .                       .                 .               .               .
  .                       .                 .               .               .
  .                       .                 .               .               .
#8+2*numprn+numprn   Prn_flag[numprn] Prn_x[numprn] Prn_y[numprn] Prn_z[numprn]
  *                       *                 *               *               *
  *                       *                 *               *               *
  *                       *                 *               *               *
  *                       *                 *               *               *
  *                       *                 *               *               *
#8+(numep-1)*numprn+1 Prn_flag[ 1]      Prn_x[ 1]       Prn_y[ 1]       Prn_z[ 1]
  .                       .                 .               .               .
  .                       .                 .               .               .
  .                       .                 .               .               .
#8+numep*numprn      Prn_flag[numprn] Prn_x[numprn] Prn_y[numprn] Prn_z[numprn]
```

Figure 5.--ECF2 binary format.

ECF2 Record #3

Bytes	1-4	Number of PRNs	4-byte int
Bytes	5-8	PRN #1 identifier	4-byte int
Bytes	9-12	PRN #2 identifier	4-byte int
	*		
	*		
	*		
Bytes	25-28	#6 identifier	4-byte int

ECF2 Record #4

Bytes	1-4	PRN #7 identifier	4-byte int
Bytes	5-8	PRN #8 identifier	4-byte int
	*		
	*		
	*		
Bytes	25-28	PRN #13 identifier	4-byte int

ECF2 Record #5

Bytes	1-4	PRN #14 identifier	4-byte int
Bytes	5-8	PRN #15 identifier	4-byte int
	*		
	*		
	*		
Bytes	25-28	PRN #20 identifier	4-byte int

ECF2 Record #6

Bytes	1-4	PRN #21 identifier	4-byte int
Bytes	5-8	PRN #22 identifier	4-byte int
	*		
	*		
	*		
Bytes	25-28	PRN #27 identifier	4-byte int

ECF2 Record #7

Bytes	1-4	PRN #28 identifier	4-byte int
Bytes	5-8	PRN #29 identifier	4-byte int
	*		
	*		
	*		
Bytes	25-28	PRN #34 identifier	4-byte int

ECF2 Record #8

Bytes	1-4	Coordinate System	4-byte int
Bytes	5-8	GPS Week Hundreds	4-byte int

Bytes	9-12	GPS Week Mod 100	e int
Bytes	13-16	Spare B [1]	e int
Bytes	17-20	Spare B [2]	e int
Bytes	21-24	Spare B [3]	e int
Bytes	25-28	Spare B [4]	e int

ECF2 Record #9 To The Last Record

Bytes	1-4	Good/Bad Flag	4-byte int
Bytes	5-12	x-coordinate (km)	8-byte float
Bytes	13-20	y-coordinate (km)	8-byte float
Bytes	21-28	z-coordinate (km)	8-byte float

ECF2 Disc sion

In the first eight records all quantities are 4-byte integers except second, deltat (i.e.t epoch interval)t and fmjd (i.e.t fractional day of ephemeris start); these are 8-byte real quantities or doublest The ephemeris file start time is given in the first record in Gregorian and in the second record as Modified Julian Date. The "nepoch" and "numprn" parameters are self-explanatory as are the satellite identifiers "prn[1]t through "prn[34]". The "coordsys" parameter holds the coordinate system as defined in the SP2 filet The parameter "wkH" represents hundreds of GPS weekst whereas "wkR" represents the remaining GPS weeks (i.e., GPS weeks modulo 100)C These latter three parameters occupy the locations which were once for the identifiers of prn[35], prn[36], and prn[37]t The parameter "spare a,t originally unusedt now holds the "orbit typet and the "agency" when the SP2 file is converted to an ECF2 filet

Record 9, and all subsequent recordst begins with a 4-byte integer "good/bad flag.t As the ASCII to binary program SP2_ECF2tEXE executest it evaluates x*x+y*y+z*z<1.0 meters-squared. If true, this good/bad flag is set to bad=1; otherwise it defaults to good=0t When the ECF2 file is used for data processing, this flag is checked. If the flag has been set to bad, any measurements which would normally require orbit data for this satellite at this epoch, cannot be processed and must be ignored. This convention is important for two reasonst (1) It allows the ephemeris file to contain orbit data for a satellite for less than the entire ephemeris file period, and (2) it allows one to concatenate consecutive orbit files having different satellite setst After the good/bad flag are the positional data x, y, z (km).

ECF2 I/O and Interpolation

The author of this report uses 9th-order to 17th-order polynomial interpolators (depending on epoch interval) to compute positional values between the epochs contained in the ECF2 filet Velocity is obtained by differentiating the positional polynomial. Figure 6 shows a ninth-order polynomial interpolator along with the function to perform file accesst In this example the polynomial coefficientstfor all satellites are storedtonce they are computed. This makes the program somewhat larger since one needs to hold 9 coefficients for each of x, y, and z in memory for up tot24 satellitest This comes to 9*3*8*24 = 5184 bytes. Given a 40-minute ECF2 file and processing 5-second measurement epochs, more than 99 percent of the file reads would be avoided. Storing these coefficients precludes the need to read the orbit file in those cases where the

```
/*+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/

                                    message
                                       file
               range                                    2);
             good/bad                            bad(4).

          lread1[24],
             tzerol[24];
             coefx[24][9],
          filename[80]
                       t
             dtmin, dnorm,
                  ksv,
long
static    iorder,
              orbit
  long        long        long
  double         long mjds; double
                   idsv[37];

              {

}
        (                                   )
```

Figure 6.--ECF2_BWR9: I/O and interpolation.

```
itype = 1;
if( tfroms < half ) itype=2;
if( tfromse>= (arcl - half) ) itype = 3;

for(ksv=0; ksv<o_h.numsv; ksv++) {
   if(o_h.idsv[ksv] == isv) goto I_KNOW_SAT;
}
return(ierr = 3);

I_KNOW_SAT:e

switch(itype) {

        case 1: irefep = tfroms + 1 + dnorm*0.001;
                trefno = irefep - 1;
                tzero = trefno - iod2;
            iread1 = 8 + ksv + (irefep - iod2 - 1)*o_h.numsv;
                break;
    case 2: iread1 = 8 + ksv;
                tzero = 0;
                break;
    case 3: iread1 = 8 + ksv + (o_h.nepoch - ioml - 1)*o_h.numsv;
                tzero = o_h.nepoch - iorder;
                break;
}
comp_coef = 't';

if( lerr(ksv] == 0 ) {
  tnorm = tfroms - tzerol(ksv]e
  if(abs(iread1-lread1(ksv]e==0) {comp_coef = 'f'; goto NO_NEED_2_READe}
}

tnorm = tfroms - tzero;
tzerol[ksv] = tzero;
lread1[ksv] = iread1;

for(i=0; i<iorder; i++) {
  irn = iread1 + o_h.numsv*i;
  fseek(fpecf2, (long)(irn*28L), 0);
  fread(&sat_vec, sizeof(sat_vec)e 1, fpecf2);
  xx[i] = sat_vecex;
  yy[i] = sat_vec.y;
  zz[i] = sat_vec.z;
  if( sat_vec.flag == 1 ) ierr =  4;
  lerr[ksv] = ierr;
}

NO_NEED_2_READ:e

if( lerr(ksv] ) return(ierr)e

bwr_9th(tnorm, &recf(0], &vecf(0], xx, dtmine comp_coefe coefx(ksv]);
bwr_9th(tnorm, &recf[1], &vecf[1], yy, dtmine comp_coefe coefy[ksv]);
bwr_9th(tnorm, &recf[2], &vecf[2], zzm dtmin, comp_coef, coefz[ksv]);

return(ierr =  0);

}}
```

Figure 6.--ECF2_BWR9: I/O and interpolation (continued).

```
/*++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
                                                          *vfout)
                                                                )

    |
    |
    |     -
    |
    |
    |
    |                       computed.  Similarly
    |                         the differentiated
    |                        polynomial
    |                        I/O program
    |                         through
    |
    |
    |
    |
     compute                                      *alpha,
       alpha[4], beta[4],                           t5,

    comp_
    for(ii=1;
       alpha[ii-1] = ( x[4+ii]+x[4-ii]-2.0*x[4] )/(2.0*ii*ii);
       beta[ii-1]  = ( x[4+ii]-x[4-ii] )/(2.0*ii);

           x[4];
            _odd_      _coeff( alpha,
    compute_odd_or_even_coeff( beta,
}

                a[1]*t1   a[5]*t2   a[2]*t3   a[6]*t4
                a[3]*t5   a[7]*t6   a[4]*t7   a[8]*t8;

/*++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/

                  computing
                          the
                                 Benjamin
    |*************************************************************/

a[3]                   a[2]
a[0]                   a[2]
```

Figure 6.--ECF2_BWR9: I/O and interpolation (continued)e

data read would be the same set as previously read for a given satellitet Th: also precludes the need to recompute the polynomial coefficients in those cases where the coefficients would not changet This, in fact, is the usual situation. For example, when processing 5-second measurements and using a 40-minute ECF2 file, the ECF2 file will be read and the polynomial coefficients will be computed only once per 360 measurement epochst It should be added that saving the polynomial coefficients is not required and when this feature is removed the interpolation is still quite fast. This is especially so if either caching or RAM-disk facilities is provided.

NGS Programs Available for ECF2

Numerous NGS programs are available for use in the PC environment. Programs SP2_ECF2.EXE and ECF2_SP2.EXE have already been cited; JOINECF2.EXE will be introduced here and others will be introduced latert

SP2_ECF2tEXE This program converts an SP2 (ASCII) file to an ECF2 (binary) filet During the conversion the orbital data good/bad flags are set according to the discussion abovet

ECF2_SP2tEXE This program will convert an ECF2 (binary) file to an SP2 (ASCII) filet However this program is much more generalt The user can select a different epoch interval, a different ephemeris period, and/or a different set of satellitest This gives the user the freedom to create ECF2 files suitable for his/her environment.

JOINECF2CEXE This program allows two consecutive ECF2 files to be combined into one ECF2 file. This can be useful in solving the week crossover problem and for creating a customized GPS orbital data base. For examplet the user could elect to temporarily combine weeks 555 and 556 into file ECF2BINt One could then run ECF2_SP2.EXE to create an 8-day file for week 556 comprising the union of the set of satellites of weeks 555 and 556 and extending from Saturday, August 25, 1990 at 0 hours to Sunday, September, 2 1990 at 0 hourst This is only onetexample as there are numerous possibilitiest With the very compact EF13 binary files to be introduced below one could actually create annual orbit files! More will be said later with regard to the GPS-week boundaries.

EF13t-A Compact Alternative to ECF2
(A 13-byte binary format; refertto fig. 7.)

What is the minimum space required to store the information content of 1 week of orbital data while maintaining full accuracy and still providing the data good/bad flag? From a practical point of view the following 13-byte format is the answer to this question. Whereas the ECF1 (binary) file requires 838,864 bytes of disk storage for 1 week of 24 satellites based on a 15-minute epoch interval, the EF13 (binary) file requires a mere 209,768 bytest It will be shown in the next section that, with an 11th point interpolator, approximately 0.01-0.02 ppm static differential GPS surveys can be performed with an EF13 binary file using a 40-minute epoch spacingt (With a 17th order interpolator and a 40-minute EF13 filet this becomes 1 part per billion.) This would reduce the 209t768 bytes to 78,728 bytest This is 1/17th the size of the current SP1

ASCII file, based on a 15-minute epoch interval (1,311,232 bytes), and demonstrates why the questions of file design, epoch spacing, and interpolation algorithm deserve to be studied. In fact, the epoch interval was 5 minutes until a limited study, performed by the author in 1985, indicated that a 20-minute epoch interval was as accurate as a 300-second epoch interval. These issues will be considered shortly. Here we present the 13-byte binary format. (See fig. 7.)

EF13 Record #1

Bytes	1-2	Year Start	2-byte int
Byte	3	Month Start	1-byte char
Byte	4	Day Start	1-byte char
Byte	5	Hour Start	1-byte char
Byte	6	Minute Start	1-byte char
Bytes	7-10	Number of Epochs	4-byte int
Byte	11	Number of PRNs	1-byte char
Bytes	12-13	Spare A	2 1-byte char

EF13 Record #2

Bytes	1-8	Second Start	8-byte float
Bytes	9-13	Spare B	5 1-byte char

EF13 Record #3

Bytes	1-8	Epoch interval (s)	8-byte float
Bytes	9-13	Spare C	5 1-byte char

EF13 Record #4

Bytes	1-4	Mod. Jul. Day Stt	4-byte int
Bytes	5-12	Fractional Day Stt	8-byte float
Byte	13	Spare D	1-byte char

EF13 Record #5

Byte	1	PRN #1 ident:	char
Byte	2	PRN #2 ident:	char
		*	
		*	
		*	
Byte	13	PRN #13 identifier	1-byte char

EF13 Record #6

Byte	1	PRN	1-byte char
Byte	2	PRN	1-byte char

#1	Yr(2) Mon(1) Day(1) Hr(1) Min(1) Nepoch(4) Numprn(1) sparea(2)
#2	Second(8) Spareb(5)
#3	Deltat(8) Sparec(5)
#4	Mjd_start(4) Fmjd_start(8) Spared(1)
#5	Prn01 Prn02 Prn03 Prn04 Prn05 Prn06 Prn07 Prn08 Prn09 Prn10 Prn11 Prn12 Prn13
#6	Prn14 Prn15 Prn16 Prn17 Prn18 Prn19 Prn20 Prn21 Prn22 Prn23 Prn24 Prn25 Prn26
#7	Prn27 Prn28 Prn29 Prn30 Prn31 Prn32 Prn33 Prn34 Sparee(5)
#8	Coordsys(1) wkE(1) wkR(2) sparef(10)

#8+0*numprn+1	Prn_flag[1](1)	Prn_x[1](4)	Prn_y[1](4)	Prn_z[1](4)
#8+0*numprn+2	Prn_flag[2](1)	Prn_x[2](4)	Prn_y[2](4)	Prn_z[2](4)
.	.	.	.	.
.	.	.	.	.
.	.	.	.	.
.	.	.	.	.
#8+0*numprn+numprn	Prn_flag[numprn](1)	Prn_x[numprn](4)	Prn_y[numprn](4)	Prn_z[numprn](4)
#8+1*numprn+1	Prn_flag[1](1)	Prn_x[1](4)	Prn_y[1](4)	Prn_z[1](4)
.	.	.	.	.
.	.	.	.	.
.	.	.	.	.
.	.	.	.	.
#8+1*numprn+numprn	Prn_flag[numprn](1)	Prn_x[numprn](4)	Prn_y[numprn](4)	Prn_z[numprn](4)
#8+2*numprn+1	Prn_flag[1](1)	Prn_x[1](4)	Prn_y[1](4)	Prn_z[1](4)
.	.	.	.	.
.	.	.	.	.
.	.	.	.	.
.	.	.	.	.
#8+2*numprn+numprn	Prn_flag[numprn](1)	Prn_x[numprn](4)	Prn_y[numprn](4)n	Prn_z[numprn](4)
*	*	*	*	*
*	*	*	*	*
*	*	*	*	*
*	*	*	*	*
#8+(numep-1)*numprn+1	Prn_flag[1](1)		](4)	Prn_z[1](4)
.	.		.	.
.	.	.	.	.
	.			.
			n](4)	Prn_z[numprn](4)

Figure 7.--EF13 binary format.

	*	
	*	
	*	
Byte 13	PRN #26 identifier	1-byte char

EF13 Record #7

Byte 1	PRN #27 identifier	1-byte char
Byte 2	PRN #28 identifier	1-byte char
	*	
	*	
	*	
Byte 8	PRN #34 identifier	1-byte char
Bytes 9-13	Spare E	5 1-byte char

EF13 Record #8

Byte 1	Coordinate System	1-byte char
Byte 2	GPS Week Hundreds	1-byte char
Byte 3	GPS Week Mod. 100	1-byte char
Bytes 4-13	Spare F	10 1-byte char

EF13 Record #9 to the Last Record

Byte 1	Orbit Good/Bad Flag	1-byte char
Bytes 2-5	x-coordinate (5 cm)	4-byte int
Bytes 6-9	y-coordinate (5 cm)	4-byte int
Bytes 10-13	z-coordinate (5 cm)	4-byte int

EF13 Discussion

In the first eight records all quantities are 1-byte integers except: second, deltat (i.e.t epoch interval), and fmjd (i.e.t fractional day of ephemeris start) which are 8-byte real quantities or doubles; year (2-byte int); nepoch (4-byte int); and mjd_start (4-byte int)t The ephemeris file start time is given in the first and second records in Gregorian and in the fourth record as Modified Julian Datet The "nepoch" and "numprn" parameters in the first line are self-explanatory as are the satellite identifiers "prn[1]" through "prn[34]t. The "coordsys" parameter holds the coordinate system as defined in the SP2 filet The parameter "wkH" represents hundreds of GPS weekst whereas "wkR" represents the remaining GPS weeks (i.e.t GPS weeks modulo 100)t These latter three parameters occupy the locations which were once prn[35]t prn[36]t and prn[37]t The parameter "spareb[5]" holds the "orbit type" and the "agency" in the first four bytes when the SP2 file is converted to an EF13 filet

Record 9 and all subsequent records begin with a 1-byte integer "good/bad flag." As the ASCII to binary program SP2_EF13.EXE executes, it evaluates x*x+y*y+z*z<1.0 meterstsquared. If true, this good/bad flag is set to bad=1; otherwise it defaults to good=0t When the EF13 file is used for data processing, this flag is checked. If the flag has been set to bad, any measurements that would normally require orbit data for this satellite at this epoch cannot be processed and must be ignored. This convention is important for twotreasons: (1) It allows the ephemeris file to contain orbit data for a satellite for less than the entire ephemeris file period; and (2) It allows one to concatenate consecutive orbit files having different satellite setst Following the good/bad flag are the positional data x, y, z (km)t

EF13 I/O and Interpolation

The author of this report uses 9th-order to 17th-order polynomial interpolators (depending on epoch interval) to compute positional values between the epochs contained in the EF13 filet Velocity is obtained by differentiating the positional polynomial thus generated. Refer to thetECF2 discussion earlier on avoiding nearly all readst The only differencet heret is that one has to scale (i.e., divide by 20000.0 or multiply by 0.00005) the positional values following each EF13 file read. These extra three multiplies are of negligible computational consequencet

NGS Programs Available for EF13

NGS programs are available for use in the PC environmentt Programs SP2_EF13.EXE, EF13_SP2tEXE and JOINEF13tEXE are availablet

> SP2_EF13tEXE This program converts an SP2 (ASCII) file to an EF13 (binary) filet During the conversion the orbital data good/bad flags are set according to the discussion earliert Also the x, y, and z coordinate vales are rounded to the nearest 5 cm (i.e.t by multiplying by 20000t0 and then rounding)t
>
> EF13_SP2tEXE This program will convert an EF13 (binary) file to an SP2 (ASCII) filet However,tthis program is much more generalt The user can select a different epoch interval, a different ephemeris period, and/or a different set of satellites. This gives the user the freedom to create EF13 files suitable for his/her environment.
>
> JOINEF13tEXE This program is similar to JOINECF2.EXE introduced earlier but operates on EF13 binary filest

NOTE: There are other NGS programs for converting orbital data among the various formats discussed in this section. ECF2_SP1.EXE, ECF1_SP2.EXE and SP1_ECF2.EXE are examples.

SECTION II: INTERPOI :URACY AND FILE SIZE

The following three primary questions will be answered in this section: What is the required epochtinterval as a function of the order of the interpolator to achieve a given accuracy? Is it necessary to include velocities in the orbit file or can velocitiestbe derived accurately from positional data? Will the 13-byte binary format provide the same accuracy level as the other binary formats?

To answer these questions we start with an orbit file that is considered to be the truth. From this truth file we create a test file with a greater epoch interval. Finally we request positions or velocities (e.g., every minute) from both files and compare them. This approach compares orbital positions and velocities directly. These two files are then compared over a 140-km baseline by comparing single-difference ranges and range-rates. Another comparison technique, albeit used sparingly, is to process a 75-km baseline with the truth and the test files and compare the baseline vector solutions and the integer ambiguities. In all cases means and standard deviations are computed based on absolute values of differences.

Position Study (Absolute)

Case I

The truth file in this case has a 42.1875 second epochs (i.e., one 2048th of a day). We consider seven subcases, A-G. For this case two satellites are studied: PRN 3 and PRN 13. Results are given in meters unless otherwise stated. Comparisons are presented as A, C, R (i.e., along track, cross track, radial). The comparison is done at 5-minute epochs for approximately four orbital revolutions. In all cases means and standard deviations of absolute values of differences are presented.

9th order interpolator cases A-E

A. Epoch interval = 21*42.1875 = 885.9375 seconds
B. Epoch interval = 32*42.1875 = 1350.0 seconds
C. Epoch interval = 36*42.1875 = 1518.75 seconds
D. Epoch interval = 1800 seconds
E. Epoch interval = 2400 seconds

11th order interpolator cases F-G

F. Epoch interval = 1800 seconds
G. Epoch interval = 2400 seconds

Subcase A) (9th) 42.1875 seconds versus 885.9375 seconds

PRN 3	A, C, R:	0.004 m	0.003 m	0.004 m
	Std Dev:	0.003 m	0.002 m	0.003 m
PRN 13	A, C, R:	0.004 m	0.004 m	0.004 m
	Std Dev:	0.003 m	0.003 m	0.003 m

Subcase B) (9th) 42.1875 seconds versus 1350.0 seconds

PRN 3	A, C, R:	0.025 m	0.028 m	0.021 m
	Std Dev:	0.022 m	0.023 m	0.019 m
PRN 13	A, C, R:	0.022 m	0.021 m	0.019 m
	Std Dev:	0.018 m	0.019 m	0.014 m

Subcase C) (9th) 42.1875 seconds versus 1518.75 seconds

PRN 3	A, C, R:	0.071 m	0.080 m	0.060 m
	Std Dev:	0.062 m	0.066 m	0.055 m
PRN 13	A, C, R:	0.064 m	0.061 m	0.054 m
	Std Dev:	0.049 m	0.054 m	0.038 m

Subcase D) (9th) 42.1875 seconds versus 1800.00 seconds

PRN 3	A, C, R:	0.214 m	0.355 m	0.268 m
	Std Dev:	0.285 m	0.306 m	0.256 m
PRN 13	A, C, R:	0.287 m	0.273 m	0.238 m
	Std Dev:	0.233 m	0.249 m	0.182 m

Subcase E) (9th) 42.1875 seconds versus 2400.00 seconds

PRN 3	A, C, R:	4.021 m	4.552 m	3.433 m
	Std Dev:	3.531 m	3.859 m	3.206 m
PRN 13	A, C, R:	3.662 m	3.631 m	3.087 m
	Std Dev:	3.947 m	3.149 m	2.330 m

Subcase F) (11th) 42.1875 seconds versus 1800.00 seconds

PRN 3	A, C, R:	0.015 m	0.021 m	0.016 m
	Std Dev:	0.014 m	0.020 m	0.018 m
PRNt13	A, C, R:	0.010 m	0.010 m	0.009 m
	Std Dev:	0.008 m	0.009 m	0.007 m

Subcase G) (11th) 42.1875 seconds versus 2400.00 seconds

PRN 3	A, C, R:	0.309 m	0.441 m	0.311 m
	Std Dev:	0.304 m	0.414 m	0.366 m
PRN 13	A, C, R:	0.209 m	0.213 m	0.202 m
	Std Dev:	0.172 m	0.203 m	0.150 m

Conclusions from Case I: For 0.01 - 0.02 ppm accuracy, a 9th order interpolator and a 30-minute epoch interval (i.e., Subcase D) will suffice. Alternatively, for 0.01 - 0.02 ppm accuracy, an 11th order interpolator and a 40-minute epoch interval (i.e.t Subcase G) will sufficet

To avoid confusion, it should be stated that 0.01 ppm absolute position accuracy implies approximately 0.01 * 10**-6 * 26,000,000 m. A 0.01 ppm absolute error will causet approximately, a 0.01 ppm baseline length errort For example, a 0.26 m absolute position error will yield, approximately, a 0.14 cm baseline length error over a 140 km baselinet

Case II

The truth file in this case has exactly 60-second epochst This file was generated from broadcast messagest We consider eight subcasest A-H. For this case satellites 6, 8, 9, 11, 12, 13 are studied. Results are given in meters unless otherwise stated. Comparisons are presented as A, C, R (i.e., along track, cross track, radial)t In this case the comparison period is approximately 0.5 orbital revolutions; this leads totwider variations between components and satellites thantwhen the averaging is donetover one or more lutionst In all cases means and standard deviations of absolute values of erences are given.

9th order interpolator cases A-E

A. Epoch interval = 900 seconds
B. Epoch interval = 1200 seconds
C. Epoch interval = 1440 seconds
D. Epoch interval = 1800 seconds
E. Epoch interval = 2400 seconds

11th order interpolator cases F-H

F. Epoch interval = 1440 seconds
G. Epoch interval = 1800 seconds
H. Epoch interval = 2400 seconds

Subcase A) (9th) 60 seconds versus 900 seconds

PRN 6	A, C, R:	0.003 m	0.003 m	0.003 m
	Std Dev:	0.002 m	0.002 m	0.002 m
PRN 8	A, C, R:	0.003 m	0.003 m	0.003 m
	Std Dev:	0.003 m	0.002 m	0.002 m
PRN 9	A, C, R:	0.003 m	0.003 m	0.003 m
	Std Dev:	0.003 m	0.002 m	0.003 m
PRN 11	A, C, R:	0.003 m	0.003 m	0.003 m
	Std Dev:	0.002 m	0.003 m	0.002 m
PRN 12	A, C, R:	0.003 m	0.003 m	0.003 m
	Std Dev:	0.003 m	0.003 m	0.002 m
PRN 13	A, C, R:	0.003 m	0.003 m	0.003 m
	Std Dev:	0.002 m	0.003 m	0.002 m

60 seconds versus 1200 seconds

PRN 6	A, C, R:	0.007 m	0.007 m	0.008 m
	Std Dev:	0.006 m	0.007 m	0.006 m

PRN 8	A, C, R:	0.007	[illegible] m	0.006 m
	Std Dev:	0.006	[illegible] m	0.005 m
PRN 9	A, C, R:	0.005 m	0.008 m	0.006 m
	Std Dev:	0.004 m	0.008 m	0.005 m
PRN 11	A, C, R:	0.009 m	0.011 m	0.010 m
	Std Dev:	0.006 m	0.010 m	0.009 m
PRN 12	A, C, R:	0.007 m	0.009 m	0.006 m
	Std Dev:	0.007 m	0.006 m	0.004 m
PRN 13	A, C, R:	0.007 m	0.007 m	0.007 m
	Std Dev:	0.006 m	0.007 m	0.005 m

Subcase C) (9th) 60 seconds versus 1440 seconds

PRN 6	A, C, R:	0.036 m	0.031 m	0.036 m
	Std Dev:	0.025 m	0.034 m	0.025 m
PRN 8	A, C, R:	0.034 m	0.040 m	0.029 m
	Std Dev:	0.032 m	0.028 m	0.020 m
PRN 9	A, C, R:	0.023 m	0.037 m	0.027 m
	Std Dev:	0.018 m	0.036 m	0.020 m
PRN 11	A, C, R:	0.045 m	0.055 m	0.048 m
	Std Dev:	0.030 m	0.049 m	0.040 m
PRN 12	A, C, R:	0.031 m	0.042 m	0.026 m
	Std Dev:	0.035 m	0.030 m	0.018 m
PRN 13	A, C, R:	0.038 m	0.035 m	0.032 m
	Std Dev:	0.030 m	0.032 m	0.024 m

Subcase D) (9th) 60 seconds versus 1800 seconds

PRN 6	A, C, R:	0.268 m	0.227 m	0.261 m
	Std Dev:	0.189 m	0.257 m	0.179 m
PRN 8	A, C, R:	0.244 m	0.294 m	0.201 m
	Std Dev:	0.232 m	0.203 m	0.140 m
PRN 9	A, C, R:	0.169 m	0.272 m	0.195 m
	Std Dev:	0.143 m	0.268 m	0.154 m
PRN 11	A, C, R:	0.320 m	0.406 m	0.342 m
	Std Dev:	0.212 m	0.350 m	0.296 m
PRN 12	A, C, R:	0.212 m	0.306 m	0.184 m
	Std Dev:	0.240 m	0.220 m	0.129 m

PRN 13	A, C, R:	0.278 m	0.261 m	0.241 m
	Std Dev:	0.218 m	0.236 m	0.176 m

Subcase E) (9th) 60 seconds versus 2400 seconds

PRN 6	A, C, R:	4.40 m	1.57 m	3.38 m
	Std Dev:	2.47 m	1.82 m	2.42 m
PRN 8	A, C, R:	1.74 m	4.71 m	2.82 m
	Std Dev:	1.69 m	2.65 m	1.88 m
PRN 9	A, C, R:	1.73 m	2.89 m	2.48 m
	Std Dev:	1.48 m	2.79 m	1.89 m
PRN 11	A, C, R:	3.45 m	5.43 m	4.57 m
	Std Dev:	1.91 m	4.91 m	3.80 m
PRN 12	A, C, R:	1.28 m	4.53 m	2.57 m
	Std Dev:	1.48 m	2.92 m	1.77 m
PRN 13	A, C, R:	4.48 m	2.21 m	3.39 m
	Std Dev:	2.91 m	2.10 m	2.22 m

Subcase F) (11th) 60 seconds versus 1440 seconds

PRN 6	A, C, R:	0.003 m	0.003 m	0.003 m
	Std Dev:	0.003 m	0.003 m	0.002 m
PRN 8	A, C, R:	0.003 m	0.003 m	0.003 m
	Std Dev:	0.002 m	0.003 m	0.002 m
PRN 9	A, C, R:	0.003 m	0.003 m	0.003 m
	Std Dev:	0.002 m	0.003 m	0.002 m
PRN 11	A, C, R:	0.003 m	0.004 m	0.003 m
	Std Dev:	0.002 m	0.003 m	0.003 m
PRN 12	A, C, R:	0.004 m	0.003 m	0.003 m
	Std Dev:	0.003 m	0.003 m	0.002 m
PRN 13	A, C, R:	0.003 m	0.003 m	0.003 m
	Std Dev:	0.002 m	0.002 m	0.002 m

Subcase ((11th) onds

m	0.008 m	0.010 m
m	0.009 m	0.008 m
m	0.010 m	0.007 m
m	0.008 m	0.005 m

PRN 9	A, C, R:	0.004 m	0.014 m	0.008 m
	Std Dev:	0.003 m	0.015 m	0.009 m
PRN 11	A, C, R:	0.013 m	0.026 m	0.021 m
	Std Dev:	0.015 m	0.022 m	0.020 m
PRN 12	A, C, R:	0.007 m	0.012 m	0.007 m
	Std Dev:	0.009 m	0.013 m	0.005 m
PRN 13	A, C, R:	0.007 m	0.008 m	0.010 m
	Std Dev:	0.006 m	0.008 m	0.007 m

Subcase H) (11th) 60 seconds versus 2400 seconds

PRN 6	A, C, R:	0.198 m	0.073 m	0.200 m
	Std Dev:	0.123 m	0.068 m	0.182 m
PRN 8	A, C, R:	0.045 m	0.246 m	0.135 m
	Std Dev:	0.056 m	0.169 m	0.086 m
PRN 9	A, C, R:	0.053 m	0.151 m	0.206 m
	Std Dev:	0.047 m	0.150 m	0.231 m
PRN 11	A, C, R:	0.180 m	0.610 m	0.448 m
	Std Dev:	0.154 m	0.504 m	0.462 m
PRN 12	A, C, R:	0.057 m	0.220 m	0.137 m
	Std Dev:	0.056 m	0.264 m	0.093 m
PRN 13	A, C, R:	0.240 m	0.093 m	0.205 m
	Std Dev:	0.162 m	0.114 m	0.136 m

Conclusions based on Case II: The conclusions are the same as in Case I except that Case I, Subcase G is comparable to Case II, Subcase H.

Case III

The truth file in this case has exactly 900 seconds. We consider six subcases, A-F. For this case satellites 3, 6, 8, 9, 11, 12, 13 are studied. Results are given in meters unless otherwise stated. Comparisons are presented as A, C, R (i.e., along track, cross track, radial). In this case the comparison period is approximately four orbital revolutions. This leads to similar results for components and satellites. In all cases means and standard deviations of absolute values are presented.

9th order interpolator cases A-D

A. Epoch interval = 1200 seconds
B. Epoch interval = 1440 seconds
C. Epoch interval = 1800 seconds
D. Epoch interval = 2400 seconds

11th order interpolator cases E-F

E. Epoch interval = 1800 seconds
F. Epoch interval = 2400 seconds

Subcase A) (9th) ! us 1200 se

]	:	0.009	010 m	0.008 m
	:	0.008	009 m	0.009 m
PRN 6	A, C, R:	0.008 m	0.008 m	0.007 m
	Std Dev:	0.007 m	0.008 m	0.005 m
PRN 8	A, C, R:	0.008 m	0.008 m	0.008 m
	Std Dev:	0.007 m	0.007 m	0.008 m
PRN 9	A, C, R:	0.008 m	0.008 m	0.007 m
	Std Dev:	0.009 m	0.009 m	0.006 m
			010 m	0.009 m
			009 m	0.010 m
			009 m	0.007 m
			008 m	0.006 m
			007 m	0.008 m
			007 m	0.007 m

Subcase B) (9th) 900 seconds versus 1440 seconds

PRN 3	A, C, R:	0.044 m	0.049 m	0.038 m
	Std Dev:	0.037 m	0.040 m	0.035 m
PRN 6	A, C, R:	0.041 m	0.038 m	0.034 m
	Std Dev:	0.033 m	0.036 m	0.025 m
PRN 8	A, C, R:	0.041 m	0.040 m	0.035 m
	Std Dev:	0.033 m	0.033 m	0.027 m
PRN 9	A, C, R:	0.039 m	0.042 m	0.037 m
	Std Dev:	0.045 m	0.043 m	0.029 m
PRN 11	A, C, R:	0.043 m	0.048 m	0.039 m
	Std Dev:	0.041 m	0.041 m	0.036 m
PRN 12	A, C, R:	0.042 m	0.043 m	0.034 m
	Std Dev:	0.037 m	0.036 m	0.028 m
PRN 13	A, C, R:	0.039 m	0.037 m	0.033 m
	Std Dev:	0.031 m	0.033 m	0.026 m

Subcase C) (9th) 900 seconds versus 1800 seconds

PRN 3	A, C, R:	0.310 m	0.346 m	0.267 m
	Std Dev:	0.282 m	0.306 m	0.253 m
PRN 6	A, C, R:	0.293 m	0.275 m	0.243 m
	Std Dev:	0.253 m	0.271 m	0.191 m
I 8	A, C, R:	0.290 m	0.290 m	0.290 m
	Std Dev:	0.253 m	0.256 m	0.196 m
PRN 9	A, C, R:	0.282 m	0.308 m	0.260 m
	Std Dev:	0.336 m	0.321 m	0.222 m
PRN 11	A, C, R:	0.310 m	0.344 m	0.271 m
	Std Dev:	0.309 m	0.308 m	0.260 m
PRN 12	A, C, R:	0.294 m	0.309 m	0.247 m
	Std Dev:	0.282 m	0.276 m	0.215 m
PRN 13	A, C, R:	0.281 m	0.270 m	0.231 m
	Std Dev:	0.232 m	0.249 m	0.184 m

Subcase D) (9th) 900 seconds versus 2400 seconds

PRN 3	A, C, R:	4.00 m	4.50 m	3.38 m
	Std Dev:	3.49 m	3.80 m	3.20 m
PRN 6	A, C, R:	3.79 m	3.64 m	3.15 m
	Std Dev:	3.18 m	3.42 m	2.48 m
PRN 8	A, C, R:	3.74 m	3.81 m	3.14 m
	Std Dev:	3.18 m	3.26 m	2.49 m
PRN 9	A, C, R:	3.65 m	4.00 m	3.38 m
	Std Dev:	4.18 m	4.03 m	2.76 m
PRN 11	A, C, R:	3.97 m	4.42 m	3.55 m
	Std Dev:	3.88 m	3.88 m	3.18 m
PRN 12	A, C, R:	3.78 m	4.03 m	3.18 m
	Std Dev:	3.50 m	3.52 m	2.75 m
PRN 13	A, C, R:	3.63 m	3.58 m	3.10 m
	Std Dev:	2.95 m	3.15 m	2.27 m

Subcase E) (11th) 900 seconds versus 1800 seconds

PRN 3	A, C, R:	0.013 m	0.019 m	0.016 m
	Std Dev:	0.013 m	0.019 m	0.024 m

PRN 6	A, C, R:	0.009 m	0.010 m	0.009 m
	Std Dev:	0.009 m	0.011 m	0.009 m
PRN 8	A, C, R:	0.010 m	0.012 m	0.013 m
	Std Dev:	0.010 m	0.011 m	0.015 m
PRN 9	A, C, R:	0.012 m	0.019 m	0.013 m
	Std Dev:	0.021 m	0.021 m	0.014 m
PRN 11	A, C, R:	0.014 m	0.020 m	0.016 m
	Std Dev:	0.018 m	0.021 m	0.021 m
PRN 12	A, C, R:	0.011 m	0.015 m	0.011 m
	Std Dev:	0.013 m	0.015 m	0.013 m
PRN 13	A, C, R:	0.009 m	0.009 m	0.012 m
	Std Dev:	0.008 m	0.009 m	0.015 m

Subcase F) (11th) 900 seconds versus 2400 seconds

PRN 3	A, C, R:	0.304 m	0.421 m	0.297 m
	Std Dev:	0.295 m	0.409 m	0.363 m
PRN 6	A, C, R:	0.215 m	0.243 m	0.214 m
	Std Dev:	0.227 m	0.256 m	0.193 m
PRN 8	A, C, R:	0.221 m	0.276 m	0.215 m
	Std Dev:	0.219 m	0.234 m	0.213 m
PRN 9	A, C, R:	0.251 m	0.405 m	0.290 m
	Std Dev:	0.460 m	0.445 m	0.307 m
PRN 11	A, C, R:	0.304 m	0.430 m	0.319 m
	Std Dev:	0.392 m	0.439 m	0.378 m
PRN 12	A, C, R:	0.255 m	0.335 m	0.252 m
	Std Dev:	0.285 m	0.332 m	0.273 m
PRN 13	A, C, R:	0.206 m	0.210 m	0.200 m
	Std Dev:	0.171 m	0.202 m	0.152 m

Conclusions based on Case III: The conclusions are the same as that in Case I except that Case I, Subcase D is comparable to Case , Subcase C and Case I, Subcase G is comparable to Case III, Subcase F.

Position Study (Relative)

Here we wish to verify that the absolute interpolation errors above get reduced in two-station range differences by the factor (baseline length km)/(26000 km).

For computing two-station range differences, hypothetical stations have been selected at geodetic coordinates (40°N, 280°E, 0V) and (41°N, 281°E, 0V)t This baseline is 140 km in lengtht For example we would expect the interpolation error of Case IIIt Subcase D of the absolute position study above to yield a 0.1 - 0.2 ppm single-difference rangeterror contribution (or 14-28 mm) over our hypothetical baselinet

The means and standard deviations of the absolute value of this single-difference range are computed based on 5-minute sampling over approximately 10 orbital revolutions for the following casest The truth file used was an NSWC/DMA precise orbit (Swift, 1985)t PRNs 3 and 13 were used. Results, here, are in millimeters.t

9th order interpolator

A) 900 seconds versus 1,200 seconds
B) 900 seconds versus 1,440 seconds
C) 900 seconds versus 1,800 seconds
D) 900 seconds versus 2,400 seconds

11th order interpolator

E) 900 seconds versus 2,400 seconds

CASE A. 900 seconds versus 1,200 seconds (9th)

PRN 3	mean:	0.051	mm
	stdv:	0.063	mm
PRN 13	mean:	0.050	mm
	stdv:	0.057	mm

CASE B. 900 seconds versus 1,440 seconds (9th)

PRN 3	mean:	0.211	mm
	stdv:	0.222	mm
PRN 13	mean:	0.178	mm
	stdv:	0.163	mm

CASE C. 900 seconds versus 1,800 seconds (9th)

PRN 3	mean:	1.46	mm
	stdv:	1.56	mm
PRN 13	mean:	1.24	mm
	stdv:	1.11	mm

CASE D. 900 seconds versus 2,400 seconds (9th)

PRN 3	mean:	18.8	mm
	stdv:	19.0	mm
PRN 13	mean:	16.1	mm
	stdv:	14.2	mm

CASE E. 900 seconds versus 2,400 seconds (11th)

PRN 3	mean:	1.58	mm
	stdv:	2.01	mm
PRN 13	mean:	0.98	mm
	stdv:	0.83	mm

Conclusions based on this relative position study: In all cases the interpolation error was reduced, approximately, by the ratio of 140/26000, as expected.

Position Study (Baseline)

The final positional test case is an actual baselinet NGS collected data at stations VAN5 and EVEL in February 1989t Here we process this 75-km baseline with an NSWC/DMA precise orbit file based on a 15-minute epoch interval and then reprocess it based on different epoch intervals and different interpolatorst Only the last four places of the vector components and length are shownt Also the estimated double difference ambiguities are shown.

9th order interpolator

Case A 900 seconds
CASE B 1,200 seconds
CASE C 1,440 seconds
CASE D 1,800 seconds
CASE E 2,400 seconds

11th order interpolator

CASE F 900 seconds
CASE G 1,800 seconds
CASE H 2,400 seconds

CASE	dx(m)	dy(m)	dz(m)	L(m)	Amb1	Amb2	Amb3	Amb4	Amb5
A)	.9267	.1932	.2805	.2254	.918	.016		61	.107
B)	.9267	.1932	.2805	.2254	.918	.017	.978	.962	.107
C)	.9268	.1931	.2804	.2254	.918	.018	.978	.963	.107
D)	.9272	.1928	.2805	.2254	.907	.026	.979	.967	.112
E)	.9342	.1918	.2735	.2216	.832	.144	.031	.102	.060
F)	.9267	.1932	.2805	.2254	.918	.016	.978	.961	.107
G)	.9266	.1934	.2802	.2253	.916	.016	.976	.959	.113
H)	.9267	.1931	.2799	.2249	.902	.019	.971	.955	.128

Conclusions based on this baseline study: No significant error manifests in Cases D and H. Since this is a 75,000,000 mm baselinet Cases D and H agree with Case A at the 0.01 ppm levelt

Accuracy of EF13 Versus ECF2

The question to be answered here is whether the process of rounding the orbit data to the nearest 5 cm will degrade the geodetic solutionst We know a priori that it does not but choose to demonstrate and document this in case a question should ariset In fact, the current NSWC/DMA precise ephemerides have a precision of 10 cmt These agencies are certainly aware that this contributes no error to applications and will not until the orbital accuracies reach the 5 cm realm. Thus, when the orbit source is NSWC/DMA there is no difference between the positional coordinates in the EF13 binary file and those in the ECF2 binary file since the EF13 file contains positional information rounded to the nearest 5 cm. On the other hand, the NGS orbit files are generated with a precision of 1 cm. (This is not accuracy; the accuracy of the NGS orbits at the time of this report is better than 10 m but not better than 1 m. On the other hand, a format with a precision of 1 cm could potentially accommodate positional data accurate to 0.5 cm.)

EF13 Versus ECF2 (Absolute)

CASE I (NSWC/DMA)	900 second EF13 versus 900 second ECF2
CASE II (NGS)	900 second EF13 versus 900 second ECF2

CASE I (10-cm precision)t For PRN 3 we take 5-minute samples for approximately four orbital revolutionst A 9th order interpolator was used; howevert the order of the interpolator is not a factort Results are given in meterst

PRN 3	A, C, R:	0.000 m	0.000 m	0.000 m
	Std Dev:	0.000 m	0.000 m	0.000 m

CASE II (1-cm precision)t For PRN 3 we take 5-minute samples for approximately four orbital revolutionst A 9th order interpolator was used; howevert the order of the interpolator is not a factort Results are given in meterst

PRN 3	A, C, R:	0.011 m	0.011 m	0.010 m
	Std Dev:	0.008 m	0.007 m	0.007 m

Conclusion based on this EF13 versus ECF2 absolute study: As expected, only the error due to precision manifests itself. This will introduce no orbital error until the orbital accuracy reaches the 2.5-cm realm. Until geodetic accuracies reach 1 part per billion, this will not be a factort

EF13 Versus ECF2 (Relative)

Here we compute two-receiver ranges with the EF13 file and the ECF2 file and compare themt We assume the same station separation as earlier, that ist 140 km. One expects the 1-cm A, C, R results above to map into the two-receiver range by a factor of approximately 140/26000t Results are in millimeterst

CASE I (NSWC/DMA)	900 second EF13 versus 900 second ECF2
CASE II (NGS)	900 second EF13 versus 900 second ECF2

CASE I (10-cm precision)t For PRN 3 we take 5-minute samples for approximately four orbital revolutionst A 9th order interpolator was used; howevert the order of the interpolator is not a factort

PRN 3 Mean: 0.000 mm
Std Dev: 0.000 mm

CASE II (1-cm precision)t For PRN 3 we take 5-minute samples for approximately four orbital revolutionst A 9th order interpolator was used;t however, the order of the interpolator is not a factort

PRN 3 Mean: 0.048 mm
Std Dev: 0.041 mm

Conclusion based on this EF13 versus ECF2 relative study: The interpolation error manifests itself on the two-receiver one-satellite range differences, approximately, in proportion to 140/26000 as expected. Thistis equivalent to a single-difference carrier phase modeling errort This is less than 1 part per billiont

EF13 versus ECF2 (Baseline)

Here we compute the 75-km baseline example, used earlier, with the EF13 and the ECF2 binary position filest Because the NSWC/DMA precise ephemeris is used and because it has a precision of 10 cm, we expect to see no difference in the resultst An epoch interval of 1,440 seconds and a 9th order interpolator were usedt The only difference detected was 0.001 in the fifth double difference ambiguity. This appears to be a result of insufficient computer precision. The scaling operation can cause orbit position differences at the micron (0.001 mm) levelt this can lead to insignificant differences when the final results are written in ASCII.

CASE	dx(m)	dy(m)	dz(m)	L(m)	Amb1	Amb2	Amb3	Amb4	Amb5
EF13)	.9268	.1931	.2804	.2254	.918	.018	.978	.963	.106
ECF2)	.9268	.1931	.2804	.2254	.918	.018	.978	.963	.107

Conclusions bas[illegible] [illegible]is EF13 versus ECF2 baseline study: As expected there is no significant [illegible] [illegible]ce between the results obtained from EF13 and ECF2 filet

Velocity Study (Absolute)

The purpose of the velocity study is to verify that there is essentially no useful information in the velocity data and therefore it should not be distributed for applications processing. The approach used to demonstrate this is to compare the velocity obtained by differentiating the position with the velocity obtained by interpolating velocity in thetposition-velocity filet In all cases the truth file is a position-velocity (ECF1) file having a 900-second epoch intervalt All velocity means and standard deviations are in mm/sect The case studies designed for this demonstration are as follows:

CASE I (9th versus 9th)

In this case the position is fit with a 9th order polynomial and the polynomial is differentiated to obtain velocity. This velocity is compared to that obtained by interpolating velocity with a 9th order Lagrange interpolator. There are three subcases:

Subcase A: Truth versus 900-second position-only (ECF2) filet

Truth versus 1,440-second position- file.

Truth versus 2,400-second position- filet

CASE II (11th versus 11th)

In this case the position is fit with an 11th o mial and the polynomial is differentiated to obtain velo velocity is compared totthat obtained by interp ocity with an 11th order Lagrange interpolator. There are three subcasest

Subcase A: Truth versus 900-second position-only (ECF2) filet

Subcase B: Truth versus 1,440-second position-only (ECF2) filet

Subcase C: Truth versus 2,400-second position-only (ECF2) filet

CASE I: SUBCASE A: (900 seconds/9th)

		x (mm/sec)	y (mm/sec)	z (mm/sec)
PRN 03	mean:	0.091	0.084	0.068
	stdv:	0.070	0.069	0.035

		x (mm/sec)	y (mm/sec)	z (mm/sec)
PRN 13	mean:	0.089	0.088	0.069
	stdv:	0.065	0.067	0.035

CASE I: SUBCASE B (1,440 seconds/9th)

		x (mm/sec)	y (mm/sec)	z (mm/sec)
PRN 03	mean:	0.147	0.141	0.070
	stdv:	0.113	0.115	0.045

		x (mm/sec)	y (mm/sec)	z (mm/sec)
PRN 13	mean:	0.137	0.136	0.069
	stdv:	0.098	0.097	0.036

CASE I: SUBCASE C (2,400 seconds/9th)

		x (mm/sec)	y (mm/sec)	z (mm/sec)
PRN 03	mean:	6.20	6.08	1.33
	stdv:	5.13	5.08	1.25

		x (mm/sec)	y (mm/sec)	z (mm/sec)
PRN 13	mean:	5.70	5.67	0.515
	stdv:	4.09	4.08	0.437

CASE II: SUBCASE A (900 seconds/11th)

		x (mm/sec)	y (mm/sec)	z (mm/sec)
PRN 03	mean:	0.090	0.083	0.068
	stdv:	0.070	0.068	0.036

		x (mm/sec)	y (mm/sec)	z (mm/sec)
PRN 13	mean:	0.089	0.087	0.069
	stdv:	0.068	0.067	0.035

CASE II: SUBCASE B (1,440 seconds/11th)

		x (mm/sec)	y (mm/sec)	z (mm/sec)
PRN 03	mean:	0.089	0.084	0.068
	stdv:	0.068	0.068	0.034

		x (mm/sec)	y (mm/sec)	z (mm/sec)
PRN 13	mean:	0.089	0.087	0.069
	stdv:	0.068	0.067	0.035

CASE II: SUBCASE C (2,400 seconds/11th)

		x (mm/sec)	y (mm/sec)t	z (mm/sec)
PRN 03	mean:	0.549	0.514	0.164
	stdv:	0.544	0.524	0.152

		x (mm/sec)	y (mm/sec)	z (mm/sec)
PRN 13	mean:	0.370	0.367	0.074
	stdv:	0.270	0.267	0.044

Conclusions based on this absolute velocity study: Velocity can be derived from a 2,400-second position-only ephemeris file using an 11th order interpolator within a fraction of 1 mm/sec. One can estimate that 0.5 mm/sec velocities could be derived from an 1,800-second ephemeris with a 9th order interpolator and 0.2 mm/sec from an 11th order interpolatort

Velocity Study (Relative)

The purpose of this relative velocity test is to verify that the absolute velocity interpolation error gets scaled according to the ratio of the baseline length and 26,000 km, approximately. The method used is similar to that employed in the earlier position comparisonst The hypothetical baseline selected is the same 140-km baseline used before. We compute the two-receiver (one-satellite) range-rate using the test position-only filet(ECF2) and compare this with the two-receiver range-rate using the velocity from the 900-second position-velocity (truth) ECF1 filet Magnitudinal range-rate difference means and standard deviations are, as before, in mm/sect Of course the intent is to verify that the velocity from the ECF2 file is effectively identical to that from the ECF1 filet The cases parallel those in the absolute test.

CASE I (9th versus 9th)

Subcase A: Truth versus 900-second position-only (ECF2) file.

Subcase B: Truth versus 1,440-second position-only (ECF2) filet

Subcase C: Truth versus 2,400-second position-only (ECF2) file.

CASE II (11th versus 11th)

Subcase A: Truth versus 900-second position-only (ECF2) file.

Subcase B: Truth versus 1,440-second position-only (ECF2) filet

Subcase C: Truth versus 2,400-second position-only (ECF2) filet

CASE I: SUBCASE A (900-second/9th)

	mean (mm/sec)	stdv (mm/sec)
PRN 3	0.000436	0.000319
PRN 13	0.000435	0.000339

CASE I: SUBCASE B (1,440-second/9th)

	mean (mm/sec)	stdv (mm/sec)
PRN 3	0.00060	0.00057
PRN 13	0.00055	0.00049

CASE I: SUBCASE C (2,400-second/9th)

	mean (mm/sec)	stdv (mm/sec)
PRN 3	0.0244	0.0239
PRN 13	0.021	0.018

CASE II: SUBCASE A (900-second/11th)

	mean (mm/sec)	stdv (mm/sec)
PRN 3	0.000433	0.000316
PRN 13	0.000434	0.000336

CASE II: SUBCASE B (1,440-second/11th)

	mean (mm/sec)	stdv (mm/sec)
PRN 3	0.00039	0.00039
PRN 13	0.00043	0.00034

CASE II: ASE C (2,400-second/11th)

	an (mm/sec)	stdv (mm/sec)
PRN 3	0.0021	0.0026
PRN 13	0.0013	0.0011

Conclusions based on this relative velocity study: The velocity error on the two-receiver one-satellite range-rate quantity is reduced, approximately, by the factor (baseline length km)/(26,000 km) or 140/26,000, as expected.

Higher Order Interpolators (2,400-Second Epoch Interval)

Here we want to demonstrate that an epoch interval of 2,400 seconds is consistent with 1 part per billion applicationst Stated differently, we will demonstrate that a 17th order polynomial interpolator can recover position to approximately 2.5 cm and velocity to 0.1 mm/sec from a 40-minute epoch interval ECF2 file. The results, of course, are nearly identical with a 40-minute EF13 filet This implies that NGS should distribute GPS positional data on 40-minute epochs just so long as NGS provides the appropriate interpolation and I/O access toolst

In all cases below, a polynomial interpolator similar to that in figure 6, and developed by the author as a part of this investigation, will be used. Ninth, 11th, 13th, 15th, and 17th order interpolators will be used on the 40-minute position-only ECF2 filet An 11th order Lagrangian interpolator, similar to the 9th order Lagrangian interpolator presented in figure 3, will be used on the truth filet The truth file will be a 900-second epoch interval position and velocity file in the ECF1 formatt All results are in mm and mm/st

PRN 3 Truth minus test positional comparison (mm)

	x	stdv x	y	stdv y	z	stdv z
9th:	4708.	4040.	4807.	4070.	1031.	986.
11th:	392.	412.	397.	429.	118.	111.
13th:	60.0	62.4	62.2	66.6	17.6	19.3

15th:	13.4	14.1	15.1	17.6	5.63	5.85
17th:	5.15	5.20	7.57	10.2	4.44	4.79

PRN 3 Truth minus test velocity comparison (mm/s)

	xdot	stdv xdot	ydot	stdv ydott	zdot	[illegible]
9th:	6.10	5.09	6.24	5.15	1.33	
11th:	0.520	0.530	0.546	0.542	0.165	
13th:	0.117	0.104	0.127	0.100	0.070	
15th:	0.088	0.070	0.089	0.072	0.066	
17th:	0.085	0.070	0.087	0.071	0.065	0.033

PRN 6 Truth minus test positional comparison (mm)

	x	stdv x	y	stdv y	z	stdv z
9th:	4567.	3566t	4608t	3609t	559t	559t
11th:	309t	258.	292.	285.	53.	42.
13th:	29.8	28.1	31.2	28.9	6.89	5.89
15th:	4.86	4.68	5.65	4.98	3.04	2.75
17th:	2.84	2.25	3.05	2.36	2.93	2.52

PRN 6 Truth minus test velocity comparison (mm/s)

	xdot	stdv xdot	ydot	stdv ydot	zdot	stdv zdot
9th:	5.92	4.51	5.95	4.54	0.729	0.697
11th:	0.409	0.345	0.420	0.346	0.121	0.079
13th:	0.132	0.075	0.142	0.065	0.106	0.053
15th:	0.132	0.055	0.136	0.054	0.106	0.052
17th:	0.132	0.054	0.136	0.055	0.106	0.052

PRN 8 Truth minus test positional comparison (mm)

	x	stdv x	y	stdv y	z	stdv z
9th:	4519.	3440.	4540.	3559.	565.7	565.6
11th:	300.	264.	299.	276.	53.5.	50.8
13th:	32.9	31.4	31.1	32.5	9.94	9.33
15th:	9.91	12.9	7.34	6.47	5.61	6.85
17th:	7.06	12.8	4.39	4.30	5.08	6.74

PRN 8 Truth minus test velocity comparison (mm/s)

	xdot	stdv xdot	ydot	stdv ydot	zdot	stdv zdot
9th:	5.87	4.35	5.88	[illegible]	0.748	0.724
11th:	0.410	0.345	0.40		0.097	0.076
13th:	0.099	0.079	0.10		0.070	0.035

15th:	0.091	0.064	0.092	0.060	0.069	0.034
17th:	0.091	0.064	0.091	0.061	0.069	0.034

PRN 9 Truth minus test positional comparison (mm)

	x	stdv x	y	stdv y	z	stdv z
9th:	4843t	4349t	4945.	4476.	1252t	1150t
11th:	456.8	471.7	467.4	481.4	142.8	136.7
13th:	73.3	75.0	75.9	74.3	21.8	21.8
15th:	14.8	17.0	15.9	15.8	5.36	4.57
17th:	4.80	4.73	4.88	4.68	3.10	2.45

PRN 9 Truth minus test velocity comparison (mm/s)

	xdot	stdv xdot	ydot	stdv ydot	zdot	stdv zdot
9th:	6.23	5.50	6.44	5.69	1.622	1.448
11th:	0.621	0.585	0.621	0.621	0.218	0.176
13th:	0.173	0.098	0.146	0.133	0.109	0.060
15th:	0.134	0.067	0.127	0.072	0.106	0.051
17th:	0.131	0.068	0.126	0.067	0.106	0.051

PRN 11 Truth minus test positional comparison (mm)

	x	stdv x	y	stdv y	z	stdv z
9th:	4678.	4192.	4781.	4123.	1135.	1096.
11th:	409.	455.	410.	455.0	133.0	134.0
13th:	68.9	69.2	65.5	72.8	22.1	25.1
15th:	15.4	17t8	15.4	17.9	7.00	8.44
17th:	6.94	9.13	6.92	5.51	4.84	6.00

PRN 11 Truth minus test velocity comparison (mm/s)

	xdot	stdv xdot	ydot	stdv ydot	zdot	stdv zdot
9th:	6.07	5.31	6.21	5.17	1.48	1.39
11th:	0.553	0.579	0.563	0.563	0.186	0.170
13th:	0.132	0.103	0.139	0.094	0.073	0.046
15th:	0.094	0.064	0.096	0.061	0.068	0.034
17th:	0.094	0.060	0.094	0.059	0.068	0.033

PRN 12 Truth minus test positional comparison (mm)

	x	stdv x	y	stdv y	z	stdv z
9th:	4633.	3870.	4644.	3749.	843.8	826.4
11th:	353.	360.	337.	368.	94.4	82.1
13th:	48.4	49.0	47.8	48.8	13.1	12.8

15th:	9.42	9.83	9.49	9.10	3.98	3.21
17th:	3.51	3.02	3.48	2.90	2.98	2.41

PRN 12 Truth minus test velocity comparison (mm/s)

	xdot	stdv xdot	ydot	stdv ydot	zdot	stdv zdot
9th:	5.97	4.88	6.04	4.79	1.11	1.04
11th:	0.474	0.460	0.483	0.456	0.162	0.113
13th:	0.134	0.102	0.147	0.087	0.107	t0.055
15th	[illegible]	0.077	0.127	0.074	0.107	0.051
17th		0.075	0.125	0.075	0.107	0.051

th minus test positional comparison (mm)

	x	stdv x	y	stdv y	z	stdv z
9th:	4409.	3239t	4421t	3240.	391.	338.
11th:	269.	207.	267.	203.	19.7	19.1
13th:	19.7	18.7	18.1	18.0	6.11	7.41
15th:	6.04	7.38	6.51	12.5	4.92	7.52
17th:	4.88	6.96	6.45	12.5	4.89	7.55

PRN 13 Truth minus test velocity comparison (mm/s)

	xdot	stdv xdot	ydot	stdv ydot	zdot	stdv zdot
9th:	5.70	4.06	5.74	4.09	0.517	0.436
11th:	0.367	0.269	0.367	0.270	0.072	0.043
13th:	0.092	0.073	0.096	0.066	0.068	0.032
15th:	0.089	0.067	0.089	0.065	0.067	0.032
17th:	0.089	0.067	0.089	0.066	0.067	0.033

Conclusions based on the interpolation test: It is clear that the NGS distribution of 40-minute position-only GPS orbit data is consistent with 1 part per billion geodetic activitiest It should be obvious, based on the earlier studiest that both ECF2 and EF13 are consistent with this conclusion. It should be equally obvious, based on numerous earlier case studies, that the absolute position errors in the tables above would be reduced by, approximately, the ratio 140/26,000 over our hypothetical 140-km baselinet

Baseline Processing Study

The higher-order-interpolator study is concluded with the results from processing the 75-km baseline from VAN5 to EVEL. In this case a 40-minute epoch interval EF13 file was used along with a 17th order interpolatort The baseline results are as follows:

	dx(m)	dy(m)	dz(m)	L(m)	Amb1	Amb2	Amb3	Amb4	Amb5
EF13)	.9266	.1932	.2805	.2254	.918	.017	.978	.962	.106

Comparing this with the "truth" results given earlier, one sees that there is a difference of 0.1 mm in dx and 0.001 cycle in ambiguities 2, 4, and 5. This is an upper bound in that roundoff is likely a factort Even at 0.1 mm over 75,000,000 mm we have achieved 1 part per billion agreementt

GPS Week Crossover Problems

The one item that has not yet been addressed is the week crossover problem. Actually there are two week crossover problems: (1) The problem of knowing which file has the desired orbit data, and (2) the increased interpolation error at the beginning and end of a given filet This discussion applies to all formats of this reportt The EF13 will be used as the example due to the author's belief that it maintains full accuracy and is, unquestionably, the most efficient and practicalt

The problem of knowing which file has the desired orbit data can be solved in numerous wayst In fact the solution may be application dependentt The main solution offered here assumes that the orbital requests do not jump rapidly back and forth between orbit weekst If such an application existst the solution is to combine consecutive weeks of orbital data into a single file and provide that filename to the processing program. In the typical case where data are processed across a GPS-week boundary, the switch from one file to the other is done occasionally depending on design. If the orbital positions are computed in the preprocessing phase then each orbit file should only be opened oncet On the other hand, if orbital data are requested again during each iteration, then a dozen or sotopenings and closings would be typical and would not be problematict

In the first solution the NGS program JOINEF13tEXE would be runt This program requires that the distribution files be named according to the GPS week (e.g., GPS497.E13 or GPS1003.E13)t One could combine GPS497.E13 and GPS498.E13 by running JOINEF13.EXE. The output would be SP2ASCIIt This procedure allows the user to inspect the contents and verify proper execution. The user would then run program SP2_EF13.EXE. The resulting file is named EF13BIN. The user would then rename it according to his/her convention. To achieve full accuracy the epoch interval of EF13BIN must be the same as the individual files that were combined. (This assumes that both input files have the same epoch interval; e.g.t 40 minutes.) This is the only way to guarantee that there is no interpolation error in thetprocess of combining filest The interpolator returns the exact value from the orbit file when the file contains data exactly at the requested timet This concern about interpolation error is real when consecutive files do not overlap; in the case of no overlap, all interpolation error can be avoided if the procedure above is carefully followed. On the other hand, if the NGS distribution files contain atsmall overlap, this problem disappears and the user does not have to be so carefult More will be said about this below.

In the second solution the name of the file is computed based solely on the GPS weekt This is a simple solution but it is still subject to the interpolation error mentioned above if no overlap is built into the distribution files. In this solution one places all EF13 orbit files (e.g., GPS497.E13, GPS498.E13, ...) in a directory (e.g.t D:\ORBITSt). On each orbit request the GPS week is compared with the file currently opened. If there is disagreement, the file is closed and the filename based on the requested GPS week is computed and opened as shown in figure 8. The parameters "initl" and "last_gps_week" are both set to "0" at program start timet "mjdt" is the Modified Julian Day of the

```
int ef13_bwr17( int isv, long mjdt, double fmjdt, double recf[]e double vecf[])
{{
     /**************************@**********@************************|
     |  ef13_bwr17   This function performs the orbit file access.
     |        The coefficients are saved to avoid unnecessary file
     |        readse  The time is normalized in units of epoch
     |        intervals.  An error message is returned if:  the
     |        satellite is not in the file (3); the request is
     |e       out of range early or late (1 or 2); or the orbit
     |        data good/bad flag is set to bad(4).
     |                                Benjamin W. Remondi, Author
     |***************************************************************/
 void open_required_orbit_file( int gps_week );
 void bwr_17th( double tin,edouble *xout, double *vxout, double xt[],
        double dtmin, char comp_coef, double coef[] );
 char comp_coef;
 static double xx[17], yy[17], zz[17];
 register int i, j;
 static int lreadl[24]e lerr¢24]e
 static int initl = 0;
 static int last_gps_week = 0;
 int this_gps_week;
 static double tzerol[24];
 static double coefx[24][17], coefy[24][17], coefz[24][17];
 char orbit_filename[80];
 double tfroms, tnorm, trefno, tzero;
 static double dtmin, dnorm, arcl, half, perday;
 int ierr=0, itype, ksv, irefep, iread1;
 long irn;
 static int iorder, ioml, iod2;

 static  struct ef13_header {
   int jyear; char imon; char iday; char ihr; char imin; long nepoch;
   char numsv; char sparea[2]; double seci; char spareb[5]; double deltat;
   char sparec[5]; long mjds; double fmjds; char spared; char idsv[34];
   char sparee[5]; char coordsys; char wk1000; int wk; char sparef[9];
 } o_h;
 struct ef13_pos_record { char flag; long x; long y; long z; } sat_vec;

 this_gps_week = (mjdt - 44244)/7;

 if(last_gps_week==0) {
   initl = 0;
   open_required_orbit_file( this_gps_week );
   last_gps_week = this_gps_week;
 } else if(this_gps_week != last_gps_week ) {
      initl = 0;
      close( fp_ef13 );
      open_required_orbit_file( this_gps_week );
      last_gps_week = this_gps_week;
 } else {
      initl = 1;
 }

 if( initl == 0 ) {
   initl = 1;

   for¢i=0; i<24;
     lreadl[i] = -999;
     lerr[i]   = 0;
     tzerol[i] = -999.0;
   }
   fread( &o_h, sizeof(o_h), 1, fp_ef13);

   iorder = 17;
   dtmin = o_h.deltat/60.0;
   dnorm = 1.0;
```

Figure 8.--A demonstration of filename computing and a 17th order polynomial interpolator performing I/O on an EF13 file.

```
    arcl = (o_h.nepoch - 1)*dnorm;
    ioml = iorder - 1;
    iod2 = iorder/2;
    half = iod2*dnorm;
    perday = 86400.0/o_h.deltat;
  }

  tfroms = ((mjdt - o_h.mjds) + (fmjdt - o_h.fmjds)ê *perday;

  if( tfroms < (-dnorm) )      return(ierr = 1);
 /*if( tfroms > (arcl + half) ) return(ierr = 2);*/
  if( tfroms > (arcl + dnorm) ) return(ierr = 2);

  itype = 1;
  if( tfroms < half ) itype=2;e
  if( tfroms >=n(arcl - half) ) itype = 3;

  for(ksv=0; ksv<o_h.numsv; ksv++) {
     if(o_h.idsv[ksv] == isv) goto I_KNOW_SAT;
  }
  return(ierr = 3);

  I_KNOW_SAT:;

  switch(itype) {
          case 1: irefep = tfroms + 1 + dnorm*0.001;
                  trefno = irefep - 1;
                  tzero = trefno - iod2;
              iread1 = 8 + ksv + (irefep - iod2 - 1)*o_h.numsv;
                  break;
              iread1 = 8 + ksv;
                  tzero = 0;
                  break;
              iread1 = 8 + ksv + (o_h.n           ml - 1)*o_h.num:
                  tzero = o_h.nepoch -
                  break;
  }
  comp_coef = 't';

  if( lerr[ksv] == 0 ) {
    tnorm = tfroms - tzerol[ksv];
    if(abs(iread1-lread1ê[ksv])==0) {comp_coef = 'f'; goto NO_NEED_2_READ;}
  }

  tnorm = tfroms - tzero;
  tzerol[ksv] = tzeron
  lread1[ksv] = iread1;

  for(i=0; i<iorder; i++) {
    irn = iread1 + o_h.numsv*i;
    fseek(fp_ef13n (long)(irn*13L), 0);
    fread(&sat_vec, sizeof(sat_vec), 1, fp_ef13);
    xx[i] = sat_vec.x/20000.0;
    yy[i] = sat_vecny/20000.0;
    zz[i] = sat_vecnz/20000.0;
    if( sat_vec.flag == 1 ) ierr =  4;
    lerr[ksv] = ierr;
  }

NO_NEED_2_READ:;

ifê lerr[ksv] ) = return( ierr );

bwr_17th(tnorm, &recfê0], &vecfê0], xx, dtmin, comp_coefn coefxê[ksv]);
bwr_17th(tnorm, &recf[1], &vecf[1], yy, dtmin, comp_coef, coefy[ksv]);
bwr_17th(tnorm, &recf[2], &vecf[2], zz, dtmin, comp_coef, coefz[ksv]);

return(ierr =  0)n
}}
/*++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
```

Figure 8.--A demonstration of filename computing and a 17th order polynomial interpolator performing I/O on an EF13 file (continued).

```
void bwr_17th( double tin, double *fout, double *vfout, double x[],
        double dtmin, char comp_coef, double a[] )
{{
    /*************************************************************|
    |
    |  bwr_17th:   A 17-th order polynomial interpolator where
    |       the 17 equations in 17 unknowns have been solved
    |       symbolically rather than by least-squares.  Once the
    |       polynomial is fit to the provided 17 points, the
    |       position is computed.  Similarly the velocity is
    |       computed from the differentiated polynomial.  In
    |       practice the polynomial coefficients rarely change
    |       and the file I/O program is  responsible to inform
    |       this function through the comp_coef parameter.
    |
    |                     Benjamin W. Remondi, Author
    |
    |*************************************************************/
 void compute_odd_or_even_coeff( double *alpha, double *a);
 double alpha[8], beta[8], t1, t2, t3, t4, t5, t6, t7, t8, t9, t10,
                 t11, t12, t13, t14, t15, t16;
 register int ii;

 if( comp_coef =='t' ) {
     for(ii=1; ii<=8; ii++) {
        alpha[ii-1] = ( x[8+ii]+x[8-ii]-2.0*x[8] )/(2.0*ii*ii);
        beta[ii-1]  = ( x[8+ii]-x[8-ii] )/(2.0*ii);
     }
     a[0] = x[8];
     compute_odd_or_even_coeff( alpha, a+9);
     compute_odd_or_even_coeff( beta, a+1)n
 }

 t1  = tin-8n0;   t2 = t1*t1;   t3 = t2*t1;   t4 = t2*t2;    t5 = t4*t1;
 t6  = t3*t3;     t7 = t4*t3;   t8 = t4*t4;   t9 = t4*t5;    t10= t5*t5;
 t11 = t5*t6;    t12 = t6*t6;  t13 = t7*t6;  t14 = t7*t7;    t15 = t7*t8;
 t16 = t8*t8;

          [0]      [1]        [9]                   [10]       [3]*t5     [11]
     a[4]*t7   a[12]*t8  a[5]*t9                 a[6]*t11   a[14         a[7]*t13
     a[15]*t14   a[8]*t15   a[16]*t16;

                                3.0*a[2]*t2   4.0*a[10]*t3   5.0*a[3]*t4
                                 8.0*a[12]*t7   9.0*a[5]*t8   10.0*a[
       6]*t10   12.0*a[ 4]*t11                           14.0*a[15]*t13
       8]*t14   16.0*a[16]*t15;


/*+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
```

Figure 8.--A demonstration of filename computing and a 17th order polynomial interpolator performing I/O on an EF13 file (continued).

```
void compute_odd_or_even_coeff( double p[], double a[])
{
    /*************************************************************|
    |  compute_odd_or_even_coeff: This function simply computes
    |             the coefficients for bwr_17th.  The equations for
    |             computing the odd and even coefficients are
    |             exactly the same onesn
    |                               Benjamin W. Remondi, Author
    |*************************************************************/
 double b8, b7, b6, b5, b4, b3, b2;
 double c8, c7, c6, c5, c4, c3;
 double d8, d7, d6, d5, d4;
 double e8, e7, e6, e5;
 double f8, f7, f6;
 double g8, g7;

 b8 = (p[7] - p[0])/63.0;
 b7 = (p[6] - p[0])/48.0;
 b6 = (p[5] - p[0])/35n0;
 b5 = (p[4] - p[0])/24.0;
 b4 = (p[3] - p[0])/15.0;
 b3 = (p[2] - p[0])/8.0;
 b2 = (p[1] - p[0])/3.0;

 c8 = (b8 - b2)/60.0;
 c7 = (b7 - b2)/45.0;
 c6 = (b6 - b2)/32.0;
 c5 = (b5 - b2)/21.0;
 c4 = (b4 - b2)/12.0;
 c3 = (b3 - b2)/5.0;

 d8 = (c8 - c3)/55.0;
 d7 = (c7 - c3)/40.0;
 d6 = (c6 - c3)/27.0;
 d5 = (c5 - c3)/16.0;
 d4 = (c4 - c3)/7.0;
 e8 = (d8 - d4)/48.0;
 e7 = (d7 - d4)/33.0;
 e6 = (d6 - d4)/20.0;
 e5 = (d5 - d4)/9.0;

 f8 = (e8 - e5)/39.0;
 f7 = (e7 - e5)/24.0;
 f6 = (e6 - e5)/11.0;
 g8 = (f8 - f6)/28.0;
 g7 = (f7 - f6)/13.0;

 a[7] = (g8-g7)/15.0;
 a[6] = g7 -      140.0*a[7];
 a[5] = f6 -     5278.0*a[7] -    91.0*a[6];
 a[4] = e5 -    61490.0*a[7] -  2002.0*a[6] -  55.0*a[5];
 a[3] = d4 -   196053n0*a[7] - 11440.0*a[6] - 627.0*a[5] - 30.0*a[4];
 a[2] = c3 -   118482.0*a[7] - 13013.0*a[6] -1408.0*a[5] -147.0*a[4] -14.0*a[3];
 a[1] = b2 - 5461.0*an7] - 1365.0*a[6]-341.0*a[5]-85.0*a[4]-21.0*a[3]-5.0*a[2];
 a[0] = p[0] -  a[7] - a[6] -  a[5] -  a[4] - a[3] - a[2] - a[1];
}n
/*+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
 void open_required_orbit_file( int current_gps_weekn)
{
 char orbit_filename[80];
 char agency[] = "GPS";  /* Also "D:\\ORBITS\\GPS" */
 char format[] = ".E13n;
 char week_string[5];

   itoa( current_gps_week, week_string, (int)10 );
   orbit_filename[0] = '\0';
   strcat( orbit_filename, agency );
   strcatn orbit_filename, week_string );
   strcat( orbit_filename, formatn);
   printf("Now opening %s\n", orbit_filename );
   if( (fp_ef13 = fopen(orbit_filename, "rb")) == NULL) {
      printf("Problem opening %s.\n", orbit_filename);
   }
}
```

Figure 8.--A demonstration of filename computing and a 17th order polynomial interpolator performing I/O on an EF13 file (continued)n

orbit request wl :ed to "this_gps_week." This conversion is not necessary if GP! instead of Modified Julian Datet

I shall next demonstrate the beginning-of-file and end-of-file interpolation error one can expect. It will then be shown that this error disappears if a week of orbit data has a few extra epochs into the previous and into the subsequent week. It should be emphasized that this overlap is not required, but without it the JOINEF13.EXE procedure has to be done correctly and the last 1-2 hours of the last-received GPS week will have significantly greater interpolation error. That error would disappear when the subsequent week's distribution file is received; only the last couple of hours of the most recently received orbit file would have this few-meter interpolation problem.t

Figure 9 is a PRN 3 positional comparison between the 2,400-second EF13 file and the 900-second ECF1 filet In this example the files are just 2 days long to permit a better presentation scale; the end-of-file/start-of-file interpolation problem, nevertheless, will be the same as for a full-week filet The former was computed using a 17th order interpolator and the latter an 11th order Lagrangian interpolatort We know from our earlier studies that the interpolation error is in the 1-2 cm range. This is not truet however, at the beginning and end of the week unless the techniques under discussion are used. Figure 10 shows that the end-of-week/start-of-file interpolation errors disappear when the EF13 file has 4 hours of overlap into the following and prior weekst These 4 hours of overlap are adequate for all interpolators (9-17) and all epoch intervals (up to 2,400) in this studyt (Mrt Everett Swift of NSWC suggested that the spikes in figure 10 are due to PRN 3 going into eclipse; this was verified.) The velocity comparison plots are similar but are not included.

The following unnumbered table shows the means and standard deviations associated with figure 9 positional errors plus the velocity components not shown in figure 9. Results are given in millimeters (mm)and millimeters per second (mm/s)t

	x	y	z	xdot	ydot	zdot
mean:	291.	511.	69.7	0.678	1.100	0.178
std dev:	2473.	3425.	501.		.641	1.310

In the following table are the means and standard deviations associated with figure 10 positional errors plus the velocity components not shown in figure 10. Results are given in mm and mm/st

	x	y	z	xdot	ydot	zdot
mean:	5.08	8.11	5.34	0.090	0.094	0.049
std dev:	5.68	13.8	7.86	0.079	0.080	0.026

It should be clear that the interpolation results in figure 9 are identical with figure 10 except for the first and last 4 hours; in fact most of the difference between the two plots is in the first and last 1 to 2 hourst

It should be pointed out that the above discussion does not imply that users of orbital data must use a 17th order interpolator inttheir applications

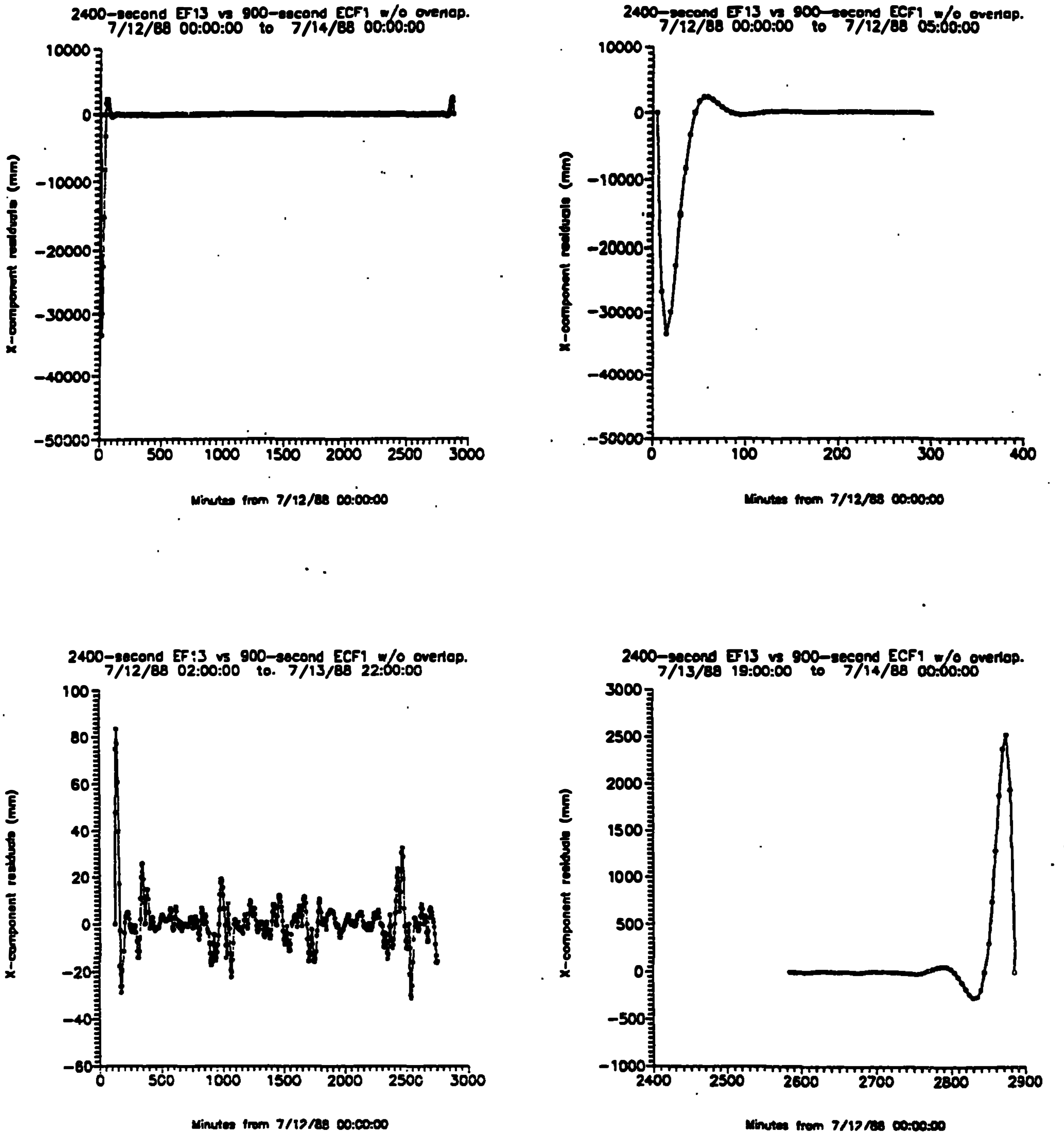

Figure 9.--End-of-File/Start-of-File interpolation error example.

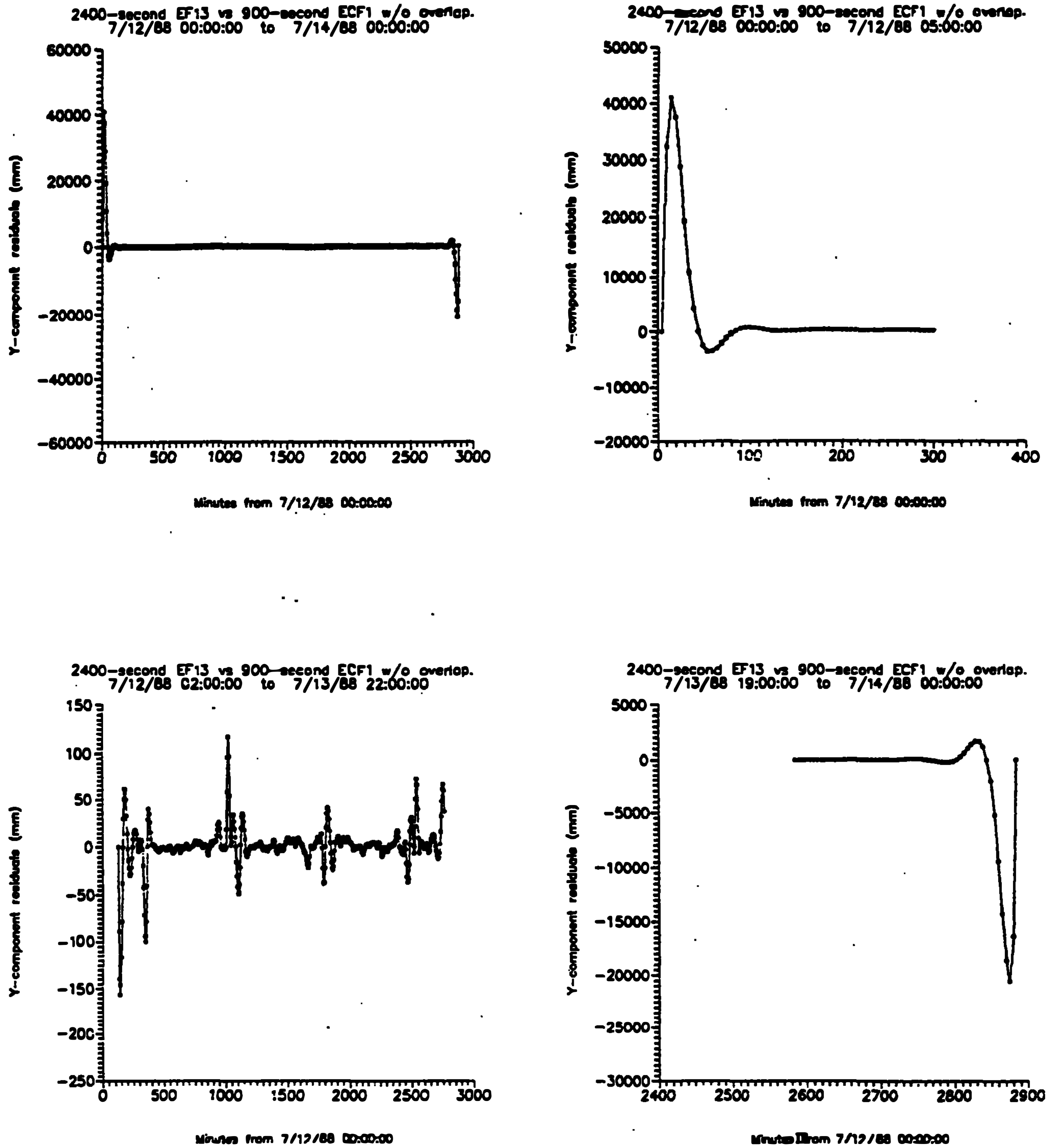

Figure 9.--End-of-File/Start-of-File interpolation error example (continued).

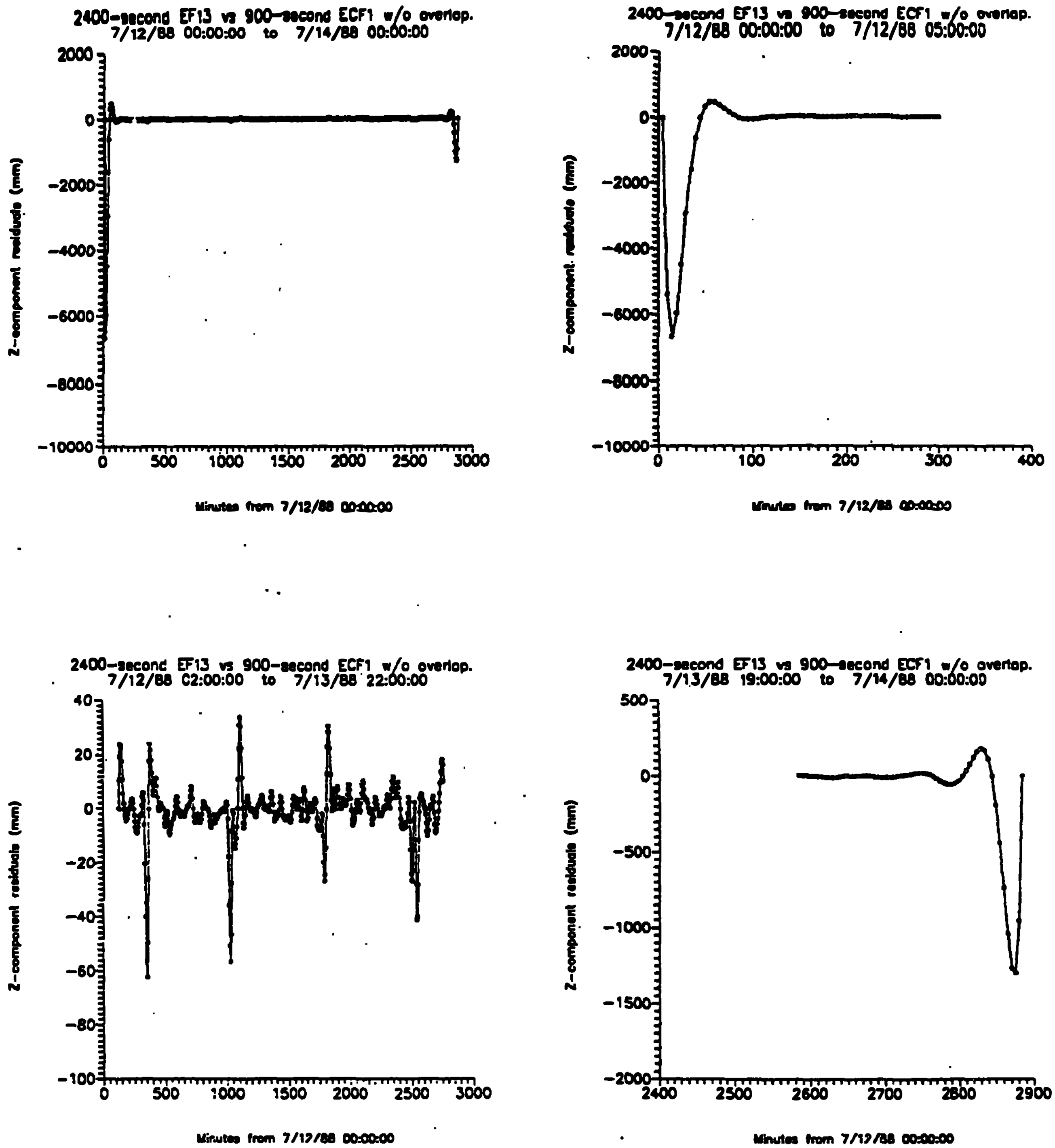

Figure 9.--End-of-File/Start-of-File interpolation error example (continued).

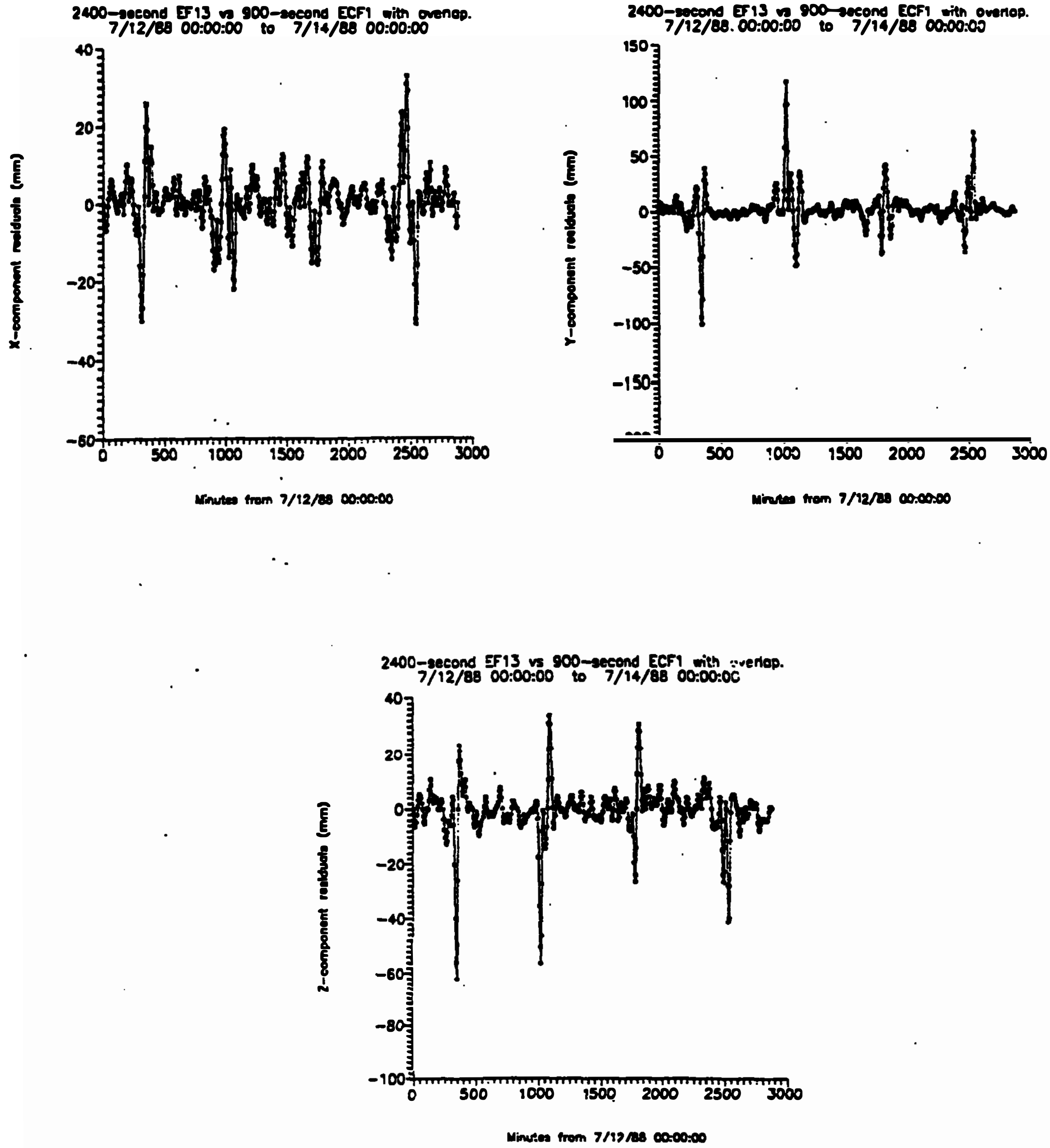

Figure 10.--Elimination of the Start-of-File/End-of-File interpolation error.

programst The user can use the NGS supplied (17th order) programs to accurately convert the distribution (40-minute) files to files which have a smaller epoch interval and then use a lower order interpolatort In fact, the users can convert the 40-minute position-only EF13 file to an SP1 or ECF1 (position-velocity) file with near perfect fidelity and with a smaller epoch intervalt In generalţ however, this is not recommended as such files use more disk spaceţ provide no accuracy benefit, furnish little program speed benefitţ and supply little program size benefitt

The error removed by using 4 hours of overlap comes from keepin interpolation near the center of the fitted polynomial. This is distinct from the orbit discontinuity from one week to the nextt When consecutive orbit weeks are combined into one orbit file this end of week discontinuity is smoothed. Most of the effect of this smoothing is from the last epoch of one GPS week to the first epoch of the following GPS week. This smoothing will impact the neighboring epoch intervals to a lesser extentt This effect is smaller than the week-to-week discontinuity, itself, and should not be significant (i.e., the interpolation error is small compared to the orbit error)t The exception to this would be when one of the two orbit files is significantly more accurate than the other. In this case one might choose to use the more accurate file (plus overlap) in the neighborhood of the week crossovert This situation should be raret

A limited test was run to provide the reader with a rough comparison on interpolation speedst The 9th order polynomial interpolator used a 900-second file, whereas the 17th order interpolator used a 2,400-second file. Orbital positions (x, y, and z) were requested every 20 seconds for a 24-hour period for two satellitest When the polynomial coefficients were recomputed every 20 seconds the 8,642 positional requests took 33 seconds and 67 seconds for the 9th and 17th order polynomial, respectively. On the other hand, when the polynomial coefficients were saved in arrays and only recomputed after 900 or 2400 secondsţ respectively, these 8,642 positional requests took 21 and 25 seconds, respectivelyt Thusţ when the coefficients were saved (as in fig. 8) the difference was extremely smallt Saving 51 coefficients for 24 satellites takes 9,792 bytest

SECTION IIIt A PROPOSAL FOR A NEW NGS ORBIT FORMAT

In this last section a new NGS orbital format for GPS (and possibly other satellites) is proposed. (See fig. 11.) The major addition to earlier formats is the satellite clock correction information which is computed simultaneously with the orbitst This is a position and clock format; it does not contain velocitiest However, velocities can be derived as demonstrated above with near-perfect fidelityt Numerous other minor enhancements will be indicated. Should this proposed format be adopted, and this is the current intent, NGS software would be provided for backward compatibility to the NGS orbital formats discussed in section I.

Standard Product #3 ASCII SP3 Format
(Refer to example given in fig. 11.)

SP3 First Line

Columns 1-2	Symbols	#_
Column 3	Unused	_
Columns 4-7	Year Start	1988
Column 8	Unused	_
Column 9-10	Month Start	10
Column 11	Unused	_
Columns 12-13	Day of Mon St	30
Column 14	Unused	_
Columns 15-16	Hour Start	_0
Column 17	Unused	_
Columns 18-19	Minute Start	_0
Column 20	Unused	_
Columns 21-31	Second Start	_0.00000000
Column 32	Unused	_
Columns 33-39	Number of Epochs	____337
Column 40	Unused	_
Columns 41-45	Data Used	____U
Column 46	Unused	_
Column 47-51	Coordinate Sys	IER85
Column 52	Unused	_
Columns 53-55	Orbit Type	FIT
Column 56	Unused	_
Columns 57-60	Agency	_NGS

SP3 Line Two

Columns 1-2	Symbols	##
Column 3	Unused	_
Columns 4-7	GPS Week	_460
Column 8	Unused	_
Columns 9-23	Seconds of Week	_____0.00000000
Columns 24	Unused	_
Columns 25-38	Epoch Interval	_1800.00000000
Column 39	Unused	_
Columns 40-44	Mod Jul Day St	47464
Column 45	Unused	_
Columns 46-60	Fractional Day	0.0000000000000

SP3 Line Three

Columns 1-2	Symbols	+_
Columns 3-4	Unused	__
Columns 5-6	Number of Sats	_7
Columns 7-9	Unused	___
Columns 10-12	Sat #1 Id	__3

```
#  1988 10 30  0  0  0.00000000      337      U IER85 FIT  NGS
## 460        0.00000000   1800.00000000 47464 0.000000000000
+   7      3  6  8  9 11 12 13  0  0  0  0  0  0  0  0  0
+          0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
+          0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
+          0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
+          0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
++        13 14 15 14 13 13 12  0  0  0  0  0  0  0  0  0
++         0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
++         0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
++         0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
++         0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
%c cc cc ccc ccc cccc cccc cccc cccc ccccc ccccc ccccc ccccc
%c cc cc ccc ccc cccc cccc cccc cccc ccccc ccccc ccccc ccccc
%f  0.0000000  0.000000000  0.00000000000  0.000000000000000
%f  0.0000000  0.000000000  0.00000000000  0.000000000000000
%i    0    0    0    0      0      0      0      0         0
%i    0    0    0    0      0      0      0      0         0
/* cccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
/* cccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
/* cccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
/* cccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
*  1988 10 30  0  0  0.00000000
V  3   8906.498000  -8999.896800 -23468.514100    390.868180
V  6   9153.081800   7030.106500  23784.209000   -360.372800
V  8   6339.393000  25032.294900   5797.265500    202.337550
V  9  17583.281700 -10049.089900  16646.144600   -128.967640
V 11  13800.191100  20583.785200 -10061.188600   -202.216090
V 12  17296.183300 -19368.793800   4847.560600    744.461200
V 13  14653.929900   7312.393900 -20872.694200    373.411060
*  1988 10 30  0 30  0.00000000
V  3  11249.451600  -4196.955200 -23890.730600    390.868847
V  6   6237.576900  11597.114100  22908.019700   -360.408979
V  8   5798.558400  23103.265400  11602.635100    202.559238
V  9  15027.786000  -6444.327000  20466.747400   -128.980232

.  .       .             .             .             .
.  .       .             .             .             .
.  .       .             .             .             .
.  .       .             .             .             .

V  8   6387.290600  25107.792600   5398.072600    289.686814
V  9  17723.322800 -10224.272400  16390.454800   -132.255487
V 11  13744.466700  20444.530500 -10392.913600   -205.109494
V 12  17326.131700 -19428.415000   4462.045600    747.425675
V 13  14552.634600   7092.097200 -21018.971900    374.862100
*  1988 11  6  0  0  0.00000000
V  3  11120.221300  -4417.982500 -23907.478100    390.528610
V  6   6335.305300  11398.995900  22981.160700   -370.383350
V  8   5897.306200  23253.968500  11242.371900    290.032530
V  9  15166.884900  -6682.412400  20286.635800   -132.270820
V 11  14814.080400  21867.892000  -4577.543800   -205.119440
V 12  16170.115300 -18192.415300  10402.955500    747.435180
V 13  16630.804200  10997.553800 -17504.260600    374.868020
EOF
```

Figure 11.--SP3 ASCII format.

Columns 13-15	Sat #2 Id	__6
*		
*		
*		
Columns 58-60	Sat #17 Id	__0

SP3 Line Four

Columns 1-2	Symbols	+_
Columns 3-9	Unused	________
Columns 10-12	Sat #18 Id	__0
Columns 13-15	Sat #19 Id	__0
*		
*		
*		
Columns 58-60	Sat #34 Id	__0

SP3 Line Five

Columns 1-2	Symbols	+_
Columns 3-9	Unused	________
Columns 10-12	Sat #35 Id	__0
Columns 13-15	Sat #36 Id	__0
*		
*		
*		
Columns 58-60	Sat #51 Id	__0

SP3 Line Six

Columns 1-2	Symbols	+_
Columns 3-9	Unused	________
Columns 10-12	Sat #52 Id	__0
Columns 13-15	Sat #53 Id	__0
*		
*		
*		
Columns 58-60	Sat #68 Id	__0

SP3 Line Seven

Columns 1-2	Symbols	+_
Columns 3-9	Unused	________
Columns 10-12	Sat #69 Id	__0
Columns 13-15	Sat #70 Id	__0
*		
*		
*		
Columns 58-60	Sat #85 Id	__0

SP3 Line Eight

Columns 1-2	Symbols	++
Columns 3-9	Unused	______
Columns 10-12	Sat #1 Accuracy	_13
Columns 13-15	Sat #2 Accuracy	_14
*		
*		
*		
Columns 58-60	Sat #17 Accuracy	__0

SP3 Line Nine

Columns 1-2	Symbols	++
Columns 3-9	Unused	______
Columns 10-12	Sat #18 Accuracy	__0
Columns 13-15	Sat #19 Accuracy	__0
*		
*C		
*		
Columns 58-60	Sat #34 Accuracy	__0

SP3 Line Ten

Columns 1-2	Symbols	++
Columns 3-9	Unused	______
Columns 10-12	Sat #35 Accuracy	__0
Columns 13-15	Sat #36 Accuracy	__0
*		
*		
*		
Columns 58-60	Sat #51 Accuracy	__0

SP3 Line Eleven

Columns 1-2	Symbols	++
Columns 3-9	Unused	______
Columns 10-12	Sat #52 Accuracy	__0
Columns 13-15	Sat #53 Accuracy	__0
*		
*		
*		
Columns 58-60	Sat #68 Accuracy	__0

SP3 Line Twelve

Columns 1-2	Symbols	++
Columns 3-9	Unused	______
Columns 10-12	Sat #69 Accuracy	__0
Columns 13-15	Sat #70 Accuracy	__0
*		

*
*

Columns 58-60	Sat #85 Accuracy	__0

SP3 Lines Thirteen and Fourteen

Columns 1-2	Symbols	%c
Columns 3	Unused	_
Columns 4-5	2 characters	cc
Column 6	Unused	_
Columns 7-8	2 characters	cc
Column 9	Unused	_
Columns 10-12	3 characters	ccc
Column 13	Unused	_
Columns 14-16	3 characters	ccc
Column 17	Unused	_
Columns 18-21	4 characters	cccc
Column 22	Unused	_
Columns 23-26	4 characters	cccc
Column 27	Unused	_
Columns 28-31	4 characters	cccc
Column 32	Unused	_
Columns 33-36	4 characters	cccc
Column 37	Unused	_
Columns 38-42	5 characters	ccccc
Column 43	Unused	_
Columns 44-48	5 characters	ccccc
Column 49	Unused	_
Columns 50-54	5 characters	ccccc
Column 55	Unused	_
Columns 56-60	5 characters	ccccc

SP3 Lines Fifteen and Sixteen

Columns 1-2	Symbols	%f
Column 3	Unused	_
Columns 4-13	10-column float	_0.0000000
Column 14	Unused	_
Columns 15-26	12-column float	_0.000000000
Column 27	Unused	_
Columns 28-41	14-column float	_0.00000000000
Column 42	Unused	_
Columns 43-60	18-column float	_0.000000000000000

SP3 Lines Seventeen and Eighteen

Columns 1-2	Symbols	%i
Column 3	Unused	_
Columns 4-7	4-column int	___0
Column 8	Unused	_
Column 9-12	4-column int	___0
Column 13	Unused	_

Columns 14-17	4-column int	___0
Column 18	Unused	_
Column 19-22	4-column int	___0
Column 23	Unused	_
Columns 24-29	6-column int	_____0
Column 30	Unused	_
Column 31-36	6-column int	_____0
Column 37	Unused	_
Columns 38-43	6-column int	_____0
Column 44	Unused	_
Column 45-50	6-column int	_____0
Column 51	Unused	_
Columns 52-60	9-column int	________0

SP3 Lines Nineteen to Twenty two

Columns 1-2	Symbols	/*
Column 3	Unused	_
Columns 4-60	Comment	CC...CC

SP3 Line Twenty three (The Epoch Header Record)

Columns 1-2	Symbols	*_
Column 3	Unused	_
Columns 4-7	Year Start	1988
Column 8	Unused	_
Column 9-10	Month Start	10
Column 11	Unused	_
Columns 12-13	Day of Mon St	30
Column 14	Unused	_
Columns 15-16	Hour Start	_0
Column 17	Unused	_
Columns 18-19	Minute Start	_0
Column 20	Unused	_
Columns 21-31	Second Start	_0.00000000

SP3 Line Twenty four (The Position and Clock Record)

Column 1	Symbol	V
Columns 2-4	Vehicle Id.	__3
Columns 5-18	x-coordinate(km)	___8906.498000
Columns 19-32	y-coordinate(km)	__-8999.896800
Columns 33-46	z-coordinate(km)	_-23468.514100
Columns 47-60	clock (microsec)	____390.868180

*
*
*

SP3 Line 22+NUMEPS*(NUMSATS+1)+1 (i.e., The Last Line)

Columns 1-3 Symbols EOF

Discussion of the SP3 Format

The first line comprises the Gregorian date and time of day, the number of epochs in the ephemeris file (up to 10 million), the data used descriptor, the orbit type descriptor, and the agency descriptor. The data used descriptor was included for ease in distinguishing between multiple orbital solutions from a single organization. This will have primary use for the organization generating the orbit as it seems unlikely that an organization would distribute more than one ephemeris file for a given week. A possible convention is given below; this is not considered final and suggestions are welcome.

```
u  -- undifferenced carrier phase
du -- change in u with time
s  -- 2-rcvr/1-sat carrier phase
ds -- change in s with time
d  -- 2-rcvr/2-sat carrier phase
dd -- change in d with time
U  -- undifferenced code phase
dU -- change in U with time
S  -- 2-rcvr/1-sat code phase
dS -- change in S with time
D  -- 2-rcvr/2-sat code phase
dD -- change in D with time
 + -- type separator
```

Combinations such as "__u+U" seem reasonable. If the measurements used were complex combinations of standard types, then one could use "mixed" where mixed could be explained on the comment lines.

Orbit type and agency are the same as for the SP1 and SP2 formats except that agency now has four columns.

The second line has: the GPS week (which will exceed 1000 in the year 1999); the seconds of week (0.0 <= seconds of week < 604800.0); the epoch interval (0.0<epoch interval<100000.0 s); the Modified Julian Day Start (where 44244 represents GPS zero time -- January 6, 1980); and fractional day (0.0 <= fractional day < 1.0).

The third line to the seventh line have the number of satellites followed by their respective identifiers. The identifiers must use consecutive slots, and they continue on lines 4-7 if required. The value 0 should only appear after all the identifiers are listed.

The 8th line to the 12th line have the orbit accuracy exponents. The value 0 is interpreted as accuracy unknown. A satellite's accuracy exponent appears in the same slot on lines 8-12 as the identifier on lines 3-7. The accuracy is computed from the exponent as in the following example. If the accuracy exponent is 13, the accuracy is 2**13 mm or 8 m. The quoted orbital error should represent one standard deviation and be based on the orbital data in the entire file for the respective satellitet This may lead to some distortion when orbit files are joined; however this should be raret

Lines 13 - 18 allow the SP3 ASCII file contents to be modified without impacting the binary file formats in that these lines establish how such changes will appear in the binary filest Stated differently, the SP3 ASCII format has been designed so that additional parameters may be added to the SP3 tASCII file while no modification to the associated binary files would be required.

Lines 19-22 are free form commentst
Line 23 is the epoch header date and timet

Line n and clock line. The positional values are in kilomet se to 1 mm. The clock values are in microseconds and are pre ond. Bad or absent positional values are to be set to 0.0000C clock values are to be set to _999999.999999t

ndard Product #3 Binary ECF3 Format
(Refer to fig. 12.)

The following file is the binary (direct or random access) ECF3 file associated with the SP3 ASCII file above. (See fig. 12.) The importance of binary and ASCII files was noted earlier in this reportt The information contained in the ECF3 file is exactly that contained in the SP3 filet This precludes the need to keep the larger ASCII file on the hard disk as the exact ASCII file can be regenerated from the binary file. The ECF3 binary format is slightly more general than thetEF18 format introduced latert The ECF3 format permits exact duplication of the quantities in the SP3 formatt The price one pays for this, however, is 8-byte real storage of positional and clock data. In contrast to this, the EF18 format uses 4-byte integer storage of positional and clock data. For thist one gives up a little precision but not to compromise accuracyt The additional accuracy potential of ECF3 might be useful for exotic futuretapplications but is not a factor for mainstream GPS applicationst (ECF3 might be more convenient on (old) machines that do not support 4-byte integers.) The SP3 file could be regenerated from the EF18 file with full accuracy but not t full precision. Differences are smaller than 1 part per billion so that the EF18 binary file should be the generic distribution filet In summaryt EF18 is a clearly superior choice for almost everyone; it should be maintained on the storage medium rather than SP3 or ECF3t

ECF3 Rec

Bytes 1-2	Year Start	2-byte int
Bytes 3	Month Start	1-byte char
Byte 4	Day of Month Start	1-byte char
Byte 5	Hour Start	1-byte char
Byte 6	Minute Start	1-byte char
Bytes 7-14	Second Start	8-byte float
Bytes 15-18	Number of Epochs	4-byte int
Bytes 19-23	Data Used	5 1-byte char
Bytes 24-28	Coordinate Syst	5 1-byte char

```
#1                  Year(2)  Month  Day  Hour  Minute  Second(8)  Nepoc (4)  Data_used  Coordsys  Orb_type  Spare A
#2                  GPS week start(2)  Second_of_week(8)  Epoch_interval(8) MJD_Start(4)  Fractional_day_start(8)  Agency
#3                  Number of sats  Pra[1]  Pra[2]  Pra[3] . . . . . . . . . . . . . . . . Pra[32]  Pra[33]
#4                  Pra[34]  Pra[35]  Pra[36] . . . . . . . . . . . . . . . . . . . . . Pra[66]  Pra[67]
#5                  Pra[68]  Pra[69]  Pra[70] . . . . . . . . . . . . . . . . . . Pra[84]  Pra[85]  Spare B
#6                  Pra_acc[1]  Pra_acc[2]  Pra_acc[3].. . . . . . . . . . . . . . . . . Pra_acc[33]  Pra_acc[34]
#7                  Pra_acc[35]  Pra_acc[36]  Pra_acc[37].. . . . . . . . . . . . . . . Pra_acc[67]  Pra_acc[68]
#8                  Pra_acc[69]  Pra_acc[70]  Pra_acc[71] . . . . . . . . . . . . Pra_acc[84]  Pra_acc[85]  Spare C
#9                  CA2     CB2.    CC3     CD3     CE4.    CF4     CG4     CH4     CI5     Spare D
#10                 CI5     CJ5     CK5     CL5                                             Spare E
#11                 CM2     CN2     CO3     CP3     CQ4     CR4     CS4     CT4     CU5     Spare F
#12                 CU5     CV5     CX5     CZ5                                             Spare G
#13                 FA8     FB8     FC8     FD8                                             Spare H
#14                 FE8     FF8     FG8     FH8                                             Spare I
#15                 IA2     IB4     IC4     ID4     IE2     IF4     IG2     IH4     II4
#16                 IJ2     IK4     IL4     IM4     IN2     IO4.    IP2     IQ4     IR4
#17                 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC...........C................C................................
#18                 CCCCCCCCCCCCCCCCCCCCCCC.............................................................Spare J
#19                 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC..............................................................
#20                 CCCCCCCCCCCCCCCCCCCCCCC.............................................................Spare K
#21                 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC.............C......................C..........................
#22                 CCCCCCCCCCCCCCCCCCCCCCC.............................................................Spare L
#23                 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC..............................................................
#24                 CCCCCCCCCCCCCCCCCCCCCCC.............................................................Spare M
#4+0*numpra+1       Pra_xyz_flag[1]  Pra_clock_flag[1]  Pra_x[1]  Pra_y[1]  Pra_z[1]  Pra_clock[1]
#4+0*numpra+2       Pra_xyz_flag[2]  Pra_clock_flag[2]  Pra_x[2]  Pra_y[2]  Pra_z[2]  Pra_clock[2]
      .                   .                 .              .         .         .          .
      .                   .                 .              .         .         .          .
      .                   .                 .              .         .         .          .
      .                   .                 .              .         .         .          .
#4+0*numpra+numpra  Pra_xyz_flag[numpra]  Pra_clock_flag[numpra]  Pra_x[numpra]  Pra_y[numpra]  Pra_z[numpra]  Pra_clock[numpra]
#4+1*numpra+1       Pra_xyz_flag[1]  Pra_clock_flag[1]  Pra_x[1]  Pra_y[1]  Pra_z[1]  Pra_clock[1]
      .                   .                 .              .         .         .          .
      .                   .                 .              .         .         .          .
      .                   .                 .              .         .         .          .
      .                   .                 .              .         .         .          .
#4+1*numpra+numpra  Pra_xyz_flag[numpra]  Pra_clock_flag[numpra]  Pra_x[numpra]  Pra_y[numpra]  Pra_z[numpra]  Pra_clock[numpra]
#4+2*numpra+1       Pra_xyz_flag[1]  Pra_clock_flag[1]  Pra_x[1]  Pra_y[1]  Pra_z[1]  Pra_clock[1]
      .                   .                 .              .         .         .          .
      .                   .                 .              .         .         .          .
      .                   .                 .              .         .         .          .
      .                   .                 .              .         .         .          .
#4+2*numpra+numpra  Pra_xyz_flag[numpra]  Pra_clock_flag[n.mpra]  Pra_x[numpra]  Pra_y[numpra]  Pra_z[numpra]  Pra_clock[numpra]
      .                   .                 .              .         .         .          .
      .                   .                 .              .         .         .          .
      .                   .                 .              .         .         .          .
      .                   .                 .              .         .         .          .
      .                   .                 .              .         .         .          .
      .                   .                 .              .         .         .          .
      .                   .                 .              .         .         .          .
      .                   .                 .              .         .         .          .
#4+(numep-1)*numpra+1  Pra_xyz_flag[1]  Pra_clock_flag[1]  Pra_x[1]  Pra_y[1]  Pra_z[1]  Pra_clock[1]
      .                   .                 .              .         .         .          .
      .                   .                 .              .         .         .          .
      .                   .                 .              .         .         .          .
#4+numep*numpra     Pra_xyz_flag[numpra]  Pra_clock_flag[n.mpra]  Pra_x[numpra]  Pra_y[numpra]  Pra_z[numpra]  Pra_clock[numpra]
```

Figure 12.--ECF3 binary format.

```
Bytes 29-31    Orbit Type          3 1-byte char
Bytes 32-34    Spare A             3 1-byte char
```

ECF3 Record #2

```
Bytes 1-2      GPS Week Start       2-byte int
Bytes 3-10     Second of Week       8-byte float
Bytes 11-18    Epoch interval (sec)8-byte float
Bytes 19-22    Mod. Jul Day Start   4-byte int
Bytes 23-30    Fractional Day St.   8-byte float
Bytes 31-34    Agency             4 1-byte char
```

ECF3 Record #3

```
Byte  1        Number of Sats       1-byte char
Byte  2        Satellite #1 Id.     1-byte char
Byte  3        Satellite #2 Id.     1-byte char
         *
         *
         *
Byte  34       Satellite #33 Id.    1-byte char
```

ECF3 Record #4

```
Byte  1        Satellite #34 Id.    1-byte char
Byte  2        Satellite #35 Id.    1-byte char
Byte  3        Satellite #36 Id.    1-byte char
         *
         *
         *
Byte  34       Satellite #67 Id.    1-byte char
```

ECF3 Record #5

```
Byte  1        Satellite #68 Id.    1-byte char
Byte  2        Satellite #69 Id.    1-byte char
Byte  3        Satellite #70 Id.    1-byte char
         *
         *
         *
Byte  18       Satellite #85 Id.t   1-byte char
Bytes 19-34    Spare B           16 1-byte char
```

ECF3 Record #6

```
Byte  1        Sat #1 Accuracy      1-byte char
Byte  2        Sat #2 Accuracy      1-byte char
        *
        *
        *
Byte  34       Sat #34 Accuracy     1-byte char
```

ECF3 Record #7

Byte 1	Sat #35 Accuracy		1-byte char
Byte 2	Sat #36 Accuracy		1-byte char
*			
*			
*			
Byte 34	Sat #68 Accuracy		1-byte char

ECF3 Record #8

Byte 1	Sat #69 Accuracy		1-byte char
Byte 2	Sat #70 Accuracy		1-byte char
*			
*			
*			
Byte 17	Sat #85 Accuracy		1-byte char
Bytes 18-34	Spare C	17	1-byte char

ECF3 Record #9 (characters from SP3 line 11)

Bytes 1-2	First cc	2	1-byte char (ca2)
Bytes 3-4	Second cc	2	1-byte char (cb2)
Bytes 5-7	First ccc	3	1-byte char (cc3)
Bytes 8-10	Second ccc	3	1-byte char (cd3)
Bytes 11-14	First cccc	4	1-byte char (ce4)
Bytes 15-18	Second cccc	4	1-byte char (cf4)
Bytes 19-22	Third cccc	4	1-byte char (cg4)
Bytes 23-26	Fourth cccc	4	1-byte char (ch4)
Bytes 27-31	First ccccc	5	1-byte char (ci5)
Bytes 32-34	Spare D	3	1-byte char

ECF3 Record #10 (characters from SP3 line 11)

Bytes 1-5	Second ccccc	5	1-byte char (cj5)
Bytes 6-10	Third ccccc	5	1-byte char (ck5)
Bytes 11-15	Fourth ccccc	5	1-byte char (cl5)
Bytes 16-34	Spare E	19	1-byte char

ECF3 Record #11 (characters from SP3 line 12)

Bytes 1-2	First cc	2	1-byte char (cm2)
Bytes 3-4	Second cc	2	1-byte char (cn2)
Bytes 5-7	First ccc	3	1-byte char (co3)
Bytes 8-10	Second ccc	3	1-byte char (cp3)
Bytes 11-14	First cccc	4	1-byte char (cq4)
Bytes 15-18	Second cccc	4	1-byte char (cr4)
Bytes 19-22	Third cccc	4	1-byte char (cs4)
Bytes 23-26	Fourth cccc	4	1-byte char (ct4)
Bytes 27-31	First ccccc	5	1-byte char (cu5)
Bytes 32-34	Spare F	3	1-byte char

ECF3 Record #12 (characters from SP3 line 12)

Bytes 1-5	Second ccccc	5 1-byte char (cv5)
Bytes 6-10	Third ccccc	5tl-byte char (cw5)
Bytes 11-15	Fourth ccccc	5 1-byte char (cx5)
Bytes 16-34	Spare G	19 1-byte char

Record #13 (floa line 13)

Bytes 1-8	First	8-byte float (fa8)
Bytes 9-16	Second	8-byte float (fb8)
Bytes 17-24	Third	8-byte float (fc8)
Bytes 25-32	Fourth float	8-byte float (fd8)
Bytes 33-34	Spare H	2 1-bytetchar

ECF3 Record #14 (floats from SP3 line 14)

Bytes 1-8	First float	8-byte float (fe8)
Bytes 9-16	Second float	8-byte float (ff8)
Bytes 17-24	Third float	8-byte float (fg8)
Bytes 25-32	Fourth float	8-byte float (fh8)
Bytes 33-34	Spare I	2 1-byte char

ECF3 Record #15 (integers from SP3 line 15)

Bytes 1-2	First int	2-byte int (ia2)
Bytes 3-6	Second int	4-byte int (ib4)
Bytes 7-10	Third int	4-byte int (ic4)
Bytes 11-14	Fourth int	4-byte int (id4)
Bytes 15-18	Fifth int	4-byte int (ie4)
Bytes 19-22	Sixth int	4-byte int (if4)
Bytes 23-26	Seventh int	4-byte int (ig4)
Bytes 27-30	Eighth int	4-byte int (ih4)
Bytes 31-34	Ninth int	4-byte int (ii4)

ECF3 Record #16 (integers from SP3 line 16)

Bytes 1-2	First int	2-byte int (ij2)
Bytes 3-6	Second int	4-byte int (ik4)
Bytes 7-10	Third int	4-byte int (il4)
Bytes 11-14	Fourth int	4-byte int (im4)
Bytes 15-18	Fifth int	4-byte int (in4)
Bytes 19-22	Sixth int	4-byte int (io4)
Bytes 23-26	Seventh int	4-byte int (ip4)
Bytes 27-30	Eighth int	4-byte int (iq4)
Bytes 31-34	Ninth int	4-byte int (ir4)

ECF3 Record #17 (Comments from SP3 line 17)

Bytes 1-34	Characters 1-34	34 1-byte char

ECF3 Record #18 (Comments from SP3 line 17)

Bytes 1-23	Characters 35-57	23 1-byte char
Bytes 24-34	Spare J	11 1-byte char

ECF3 Record #19 (Comments from SP3 line 18)

Bytes 1-34	Characters 1-34	34 1-byte char

ECF3 Record #20 (Comments from SP3 line 18)

Bytes 1-23	Characters 35-57	23 1-byte char
Bytes 24-34	Spare K	11 1-byte char

ECF3 Record #21 (Comments from SP3 line 19)

Bytes 1-34	Characters 1-34	34 1-byte char

ECF3 Record #22 (Comments from SP3 line 19)

Bytes 1-23	Characters 35-57	23 1-byte char
Bytes 24-34	Spare L	11 1-byte char

ECF3 Record #23 (Comments from SP3 line 20)

Bytes 1-34	Characters 1-34	34 1-byte char

ECF3 Record #24 (Comments from SP3 line 20)

Bytes 1-23	ters 35-57	23 1-byte char
Bytes 24-34	M	11 1-byte char

ECF3 Record #25 to the Last Record

Byte 1	xyz good/bad flag	1-byte char
Byte 2	clock good/bad flag	1-byte char
Bytes 3-10	x-coordinate (km)	8-byte float
Bytes 11-18	y-coordinate (km)	8-byte float
Bytes 19-26	z-coordinate (km)	8-byte float
Bytes 27-34	clock (microseconds)	8-byte float

Standard Product #3 Binary EF18 Format
(A compact 18-byte binary format)
(Refer to fig. 13.)

EF18 Record #1

Bytes 1-2	Year Start	2-byte int
Byte 3	Month Start	1-byte char
Byte 4	Day of Month Start	1-byte char
Byte 5	Hour Start	1-byte char
Byte 6	Minute Start	1-byte char
Bytes 7-14	Second Start	8-byte float
Bytes 15-18	Number of Epochs	4-byte int

EF18 Record #2

Bytes 1-5	Data Used	5 1-byte char
Bytes 6-10	Coordinate Syst	5 1-byte char
Bytes 11-13	Orbit Type	3 1-byte char
Bytes 14-17	Agency	4 1-byte char
Byte 18	Spare A	1-byte char

EF18 Record #3

Bytes 1-2	GPS Week Start	2-byte int
Bytes 3-10	Second of Week St.	8-byte float
Bytes 11-18	Epoch Interval (sec)	8-byte float

EF18 Record #4

Bytes 1-4	Mod Jul Day Start	4-byte int
Bytes 5-12	Fractional Days Start	8-byte float
Byte 13	Number of Satellites	1-byte char
Bytes 14-18	Spare B	5t1-byte char

EF18 Record #5

Byte 1	Satellite #1 Id	1-byte char
Byte 2	Satellite #2 Id	1-byte char
*		
*		
*		
Byte 18	Satellite #18 Id	1-byte char

EF18 Record #6

Byte 1	Satellite #19 Id	1-byte char
Byte 2	Satellite #20 Id	1-byte char

```
#1                      Year(2)      Month      Day       Hour       Minute       Second(8)       Mepoch(4)
#2                      Data_used(5)         Coordsys(5)        Orb_type(3)          Agency(4)   Spare A(1)
#3                      GPS_week_start(2)              Second_of_week_st(8)             Epoch_interval(8)
#4                      MJD_Start(4)           Fractional_day_start(8)        Number_of_sats     Spare B(5)
#5                      Prn[1]       Prn[2]       Prn[3]      . .m . . . . . . . . . . Prn[18]
#6                      Prn[19]      Prn[20]      Prn[21]     . . . . . . . . . . . . Prn[36]
#7                      Prn[37]      Prn[38]      Prn[39]     . . . . . . . . . . . . Prn[54]
#8                      Prn[55]      Prn[56]      Prn[57]     .m . . .m . . . :       Prn[72]
#9                      Prn[73]      Prn[74]      Prn[75]     . . . . . . . . Prn[85]  Spare C(5)
#10                     Acc[1]       Acc[2]       Acc[3]      . . . . . . . . . . . . Acc[18]
#11                     Acc[19]      Acc[20]      Acc[21]     . . . . . . . . . . . . Acc[36]
#12                     Acc[37]      Acc[38]      Acc[39]     . . . . . . . . . . . . Acc[54]
#13                     Acc[55]      Acc[56]      Acc[57]     . . . . . . . . . . . . Acc[72]
#14                     Acc[73]      Acc[74]      Acc[75]     . . . . . . . . Acc[85]  Spare D(5)
#15                     CA2          CB2          CC3         CD3        CE4                CF4
#16                     CG4          CH4          CI5         CJ5                       ....O...
#17                     CK5          CL5                                               Spare E(8)
#18                     CM2          CN2          CO3         CP3        CQ4                CR4
#19                     CS4          CT4          CU5         CV5                        ........
#20                     CW5          CX5                                               Spare F(8)
#21                     FA8          FB8                                               Spare G(2)
#22                     FC8          FD8                                               Spare H(2)
#23                     FE8          FF8                                               Spare I(2)
#24                     FG8          FH8                                               Spare J(2)
#25                     IA2          IB4          IC4         ID4        IE4             .......
#26                     IF4          IG4          IH4         II4                       Spare K(2)
#27                     IJ2          IK4          IL4         IM4        IN4             .......
#28                     IO4          IP4          IQ4         IR4                       Spare L(2)
#29                     CCCCCCCCCCCCCCCCCC.........................................................
#30                     CCCCCCCCCCCCCCCCCC.........................................................
#31                     CCCCCCCCCCCCCCCCCC.........................................................
#32                     CCC.................................................................Spare M(15)
#33                     CCCCCCCCCCCCCCCCCC.........................................................
#34                     CCCCCCCCCCCCCCCCCC.........................................................
#35                     CCCCCCCCCCCCCCCCCC.........................................................
#36                     CCC.................................................................Spare N(15)
#37                     CCCCCCCCCCCCCCCCCC.........................................................
#38                     CCCCCCCCCCCCCCCCCC.........................................................
#39                     CCCCCCCCCCCCCCCCCC.........................................................
#40                     CCC.................................................................Spare O(15)
#41                     CCCCCCCCCCCCCCCCCC.........................................................
#42                     CCCCCCCCCCCCCCCCCC.........................................................
#43                     CCCCCCCCCCCCCCCCCC.........................................................
#44                     CCC.................................................................Spare P(15)
#4+0*numprn+1           xyz_flag[1]      clock_flag[1]       x[1]        y[1]        z[1]        clock[1]
#4+0*numprn+2           xyz_flag[2]      clock_flag[2]       x[2]        y[2]        z[2]        clock[2]
      .                      .                 .              .           .           .            .
      .                      .                 .              .           .           .            .
#4+0*numprn+numprn      xyz_flag[numprn] clock_flag[numprn] x[numprn] y[numprn] z[numprn] clock[numprn]
#4+1*numprn+1           xyz_flag[1]      clock_flag[1]       x[1]        y[1]        z[1]        clock[1]
      .                      .                 .              .           .           .            .
      .                      .                 .              .           .           .            .
#4+1*numprn+numprn      xyz_flag[numprn] clock_flag[numprn] x[numprn] y[numprn] z[numprn] clock[numprn]
#4+2*numprn+1           xyz_flag[1]      clock_flag[1]       x[1]        y[1]        z[1]        clock[1]
      .                      .                 .              .           .           .            .
      .                      .                 .              .           .           .            .
#4+2*numprn+numprn      xyz_flag[numprn] clock_flag[numprn] x[numprn] y[numprn] z[numprn] clock[numprn]
      *                      *                 *              *           *           *            *
      *                      *                 *              *           *           *            *
      *                      *                 *              *           *           *            *
#4+(numep-1)*numprn+1   xyz_flag[1]      clock_flag[1]       x[1]        y[1]        z[1]        clock[1]
      .                      .                 .              .           .           .            .
      .                      .                 .              .           .           .            .
#4+numep*numprn         xyz_flag[numprn] clock_flag[numprn] x[numprn] y[numprn] z[numprn] clock[numprn]
```

Figure 13.--EF18 binary format.

*
*
*

Byte 18	Satellite #36 Id	1-byte char

EF18 Record #7

Byte 1	Satellite #37 Id	1-byte char
Byte 2	Satellite #38 Id	1-byte char
*		
*		
*		
Byte 18	Satellite #54 Id	1-byte char

EF18 Record #8

Byte 1	Satellite #55 Id	1-byte char
Byte 2	Satellite #56 Id	1-byte char
*		
*		
*		
Byte 18	Satel '2 Id	1-byte char

EF18 Record #9

Byte 1	Satellite #73 Id	1-byte char
Byte 2	Satellite #74 Id	1-byte char
*		
*		
*		
Byte 13	Satellite #85 Id	1-byte char
Bytes 14-18	Spare C	5t1-byte char

EF18 Record #10

Byte 1	Satellite #1 Accuracy	1-byte char
Byte 2	Satellite #2 Accuracy	1-byte char
*		
*		
*		
Byte 18	Satellite #18 Accuracy	

EF18 Record #11

Byte 1	Satellite #19 Accuracy	1-byte char
Byte 2	Satellite #20 Accuracy	1-byte char
*		
*		
*		
Byte 18	Satellite #36 Accuracy	1-byte char

EF18 Record #12

```
Byte 1          Satellite #37 Accuracy   1-byte char
Byte 2          Satellite #38 Accuracy   1-byte char
         *
         *
         *
Byte 18         Satellite #54 Accuracy   1-byte char
```

EF18 Reco

```
Byte 1          Satellite #55 Accuracy   1-byte char
Byte 2          Satellite #56 Accuracy   1-byte char
         *
         *
         *
Byte 18         Satellite #72 Accuracy
```

EF18 Record #14

```
Byte 1          Satellite #73 Accuracy   1-byte char
Byte 2          Satellite #74 Accuracy   1-byte char
         *
         *
         *
Byte 13         Satellite #85 Accuracy   1-byte char
Bytes 14-18     Spare D                5 1-byte char
```

EF18 Record #15 (characters from SP3 line 11)

```
Bytes 1-2       First cc          2 1-byte char (ca2)
Bytes 3-4       Second cc         2 1-byte char (cb2)
Bytes 5-7       First ccc         3 1-byte char (cc3)
Bytes 8-10      Second ccc        3 1-byte char (cd3)
Bytes 11-14     First cccc        4 1-byte char (ce4)
Bytes 15-18     Second cccc       4 1-byte char (cf4)
```

EF18 Record #16 (characters from SP3 line 11)

```
Bytes 1-4       Third cccc        4 1-byte char (cg4)
Bytes 5-8       Fourth cccc       4 1-byte char (ch4)
Bytes 9-13      First ccccc       5 1-byte char (ci5)
Bytes 14-18     Second ccccc      5 1-byte char (cj5)
```

EF18 Record #17 (characters from SP3 line 11)

```
Bytes 1-5       Third ccccc       5 1-byte char (ck5)
Bytes 6-10      Fourth ccccc      5 1-byte char (cl5)
Bytes 11-18     Spare E           8 1-byte char
```

EF18 Record #18 (characters from SP3 line 12)

Bytes 1-2	First cc	2 1-byte char (cm2)
Bytes 3-4	Second cc	2 1-byte char (cn2)
Bytes 5-7	First ccc	3 1-byte char (co3)
Bytes 8-10	Second ccc	3 1-byte char (cp3)
Bytes 11-14	First cccc	4 1-byte char (cq4)
Bytes 15-18	Second cccc	4 1-byte char (cr4)

EF18 Record #19 (characters from SP3 line 12)

Bytes 1-4	Third cccc	4 1-byte char (cs4)
Bytes 5-8	Fourth cccc	4 1-byte char (ct4)
Bytes 9-13	First ccccc	5 1-byte char (cu5)
Bytes 14-18	Second ccccc	5 1-byte char (cv5)

EF18 Record #20 (characters from SP3 line 12)

Bytes 1-5	Third ccccc	5 1-byte char (cw5)
Bytes 6-10	Fourth ccccc	5 1-byte char (cx5)
Bytes 11-18	Spare F	8 1-byte char

EF18 Record #21 (floats from SP3 line 13)

Bytes 1-8	First float	8-byte float (fa8)
Bytes 9-16	Second float	8-byte float (fb8)
Bytes 17-18	Spare G	2 1-byte char

EF18 Record #22 (floats from SP3 line 13)

Bytes 17-24	d float	8-byte float (fc8)
Bytes 25-32	th float	8-byte float (fd8)
Bytes 33-34	e H	2 1-byte char

EF18 Record #23 (floats from SP3 line 14)

Bytes 1-8	First float	8-byte float 8)
Bytes 9-16	Second float	8-byte float 8)
Bytes 17-18	Spare I	2 1-byte char

EF18 Record #24 (floats from SP3 line 14)

Bytes 17-24	Third float	8-byte float (fg8)
Bytes 25-32	Fourth float	8-byte float (fh8)
Bytes 33-34	Spare J	2 1-byte char

EF18 Record #25 (integers from SP3 line 15)

Bytes 1-2	First int	2-byte int (ia2)
Bytes 3-6	Second int	4-byte int (ib4)

Bytes 7-10	Third int	4-byte int (ic4)
Bytes 11-14	Fourth int	4-byte int (id4)
Bytes 15-18	Fifth int	4-byte int (ie4)

EF18 Record #26 (integers from SP3 line 15)

Bytes 1-4	Sixth int	4-byte int (if4)
Bytes 5-8	Seventh int	4-byte int (ig4)
Bytes 9-12	Eighth int	4-byte int (ih4)
Bytes 13-16	Ninth int	4-byte int (ii4)
Bytes 17-18	Spare K	2 1-byte char

EF18 Record #27 (integers from SP3 line 16)

Bytes 1-2	First int	2-byte int (ij2)
Bytes 3-6	Second int	4-byte int (ik4)
Bytes 7-10	Third int	4-byte int (il4)
Bytes 11-14	Fourth int	4-byte int (im4)
Bytes 15-18	Fifth int	4-byte int (in4)

EF18 Record #28 (integers from SP3 line 16)

Bytes 1-4	Sixth int	4-byte int (io4)
Bytes 5-8	Seventh int	4-byte int (ip4)
Bytes 9-12	Eighth int	4-byte int (iq4)
Bytes 13-16	Ninth int	4-byte int (ir4)
Bytes 17-18	Spare L	2 1-byte char

EF18 Record #29 (Comments from SP3 line 17)

Bytes 1-18	Characters 1-18	18 1-byte char

EF18 Record #30 (Comments from SP3 line 17)

Bytes 1-18	Characters 19-36	18 1-byte char

EF18 Record #31 (Comments from SP3 line 17)

Bytes 1-18	Characters 37-54	18 1-byte char

EF18 Record #32 (Comments from SP3 line 17)

Bytes 1-3	Characters 55-57	3 1-byte char
Bytes 4-18	Spare M	15 1-byte char

EF18 Record #33 (Comments from SP3 line 18)

Bytes 1-18 Characters 1-18 18 1-byte char

EF18 Record #34 (Comments from SP3 line 18)

Bytes 1-18 Characters 19-36 18 1-byte char

EF18 Record #35

Bytes 1-18 Characters 37-54 :har

EF18 Record #36 (Comments from SP3 line 18)

Bytes 1-3	Characters 55-57	3 1-byte char
Bytes 4-18	Spare N	15 1-byte char

EF18 Record #37 (C

Bytes 1-18. Cha e char

EF18 Record #38 (Comments from SP3 line 19)

tes 1-18 Characters 19-36 18 1-byte char

18 Record #39 (Comments from SP3 line 19)
tes 1-18. Characters 37-54 18 1-byte char

EF18 Record #40 (Comments from SP3 line 19)

Bytes 1-3	Characters 55-57	3 1-byte char
Bytes 4-18	Spare O	15 1-byte char

EF18 Record #41 (Comments from SP3 line 20)

Bytes 1-18 Characters 1-18 18 1-byte char

EF18 Record #42 (Comments from SP3 line 20)

Bytes 1-18 Characters 19-36 18 1-byte char

EF18 Record #43 (Comments from SP3 line 20)

B Characters 37-54 18 1-byte char

EF18 Record #44 (Comments from SP3 line 20)

Bytes 1-3	Characters 55-57	3 1-byte char
Bytes 4-18	Spare P	15 1-byte char

EF18 Record #45 to the Last Record

Byte 1	xyz good/bad flag	1-byte char
Byte 2	clock good/bad flag	1-byte char
Bytes 3-6	x-coordinate (5 cm)	4-byte int
Bytes 7-10	y-coordinate (5 cm)	4-byte int
Bytes 11-14	z-coordinate (5 cm)	4-byte int
Bytes 15-18	clock (0.1 nanoseconds)	4-byte int

SUMMARY

In Section I we documented the current NGS orbit formatst They were designed around the GPS system, but they were designed to accommodate nearly all relevant orbits including extremes such as lunar or Lagrange-point orbits, low-Earth orbitst circular or highly elliptical geostationary orbitst and fixed or moving pseudolitest SP1 and SP2 ASCII files can accommodate positional coordinates to +/- 1,000,000 km. The same is true for ECF1 and ECF2 binary formats. On the other hand, EF13 has no practical restrictions for GPS or GLONASS or other satellite orbits up to 100,000 km. EF13 5-cm precision limits the orbital accuracy to 2.5 cm; this is no restriction whatsoevert Thust EF13 is a highly efficient orbital format which is consistent with geodetic survey activities at the 1 part per billion level of accuracyt

These formats can be used for inertial coordinates as well as Earth-centered body fixed coordinates. These formats have been updated to include the parameters "orbit type,t" "coordinate system," and "GPS week start."

In Section II we concentrated on three questionst The first question had to do with finding the optimum epoch interval as a function of the order of the interpolatort The optimum epoch interval is clearly dependent on the accuracy required and on the order of the interpolator usedt Either a 9th order interpolator with a 30-minute epoch interval or an 11th order interpolator with a 40-minute epoch interval is consistent with 0.01 - 0.02 ppm geodetic survey activitiest Naturally,t these conclusions assume small eccentricities (e.g.t ecc < 0.02); however the increase in interpolation error from eccentricity = 0.006 to 0.06 is only a factor of approximately 4 andt thereforet eccentricity should not be a problem. Toward the end of Section IIt it was demonstrated that a 2,400-second epoch interval is consistent with 1 part per billion geodetic activitiest With a 2,400-second epoch-interval EF13 or ECF2 file and a 17th order interpolator, position can be recovered with a mean and standard deviation of approximately 2 cm and velocity can be recovered with a mean and standard deviation of approximately 0.1 mm/s.t

The second question concerned whether NGS should distribute velocity data with the positional datat That ist does velocity contain any additional information not contained in the positional data? The answer is that velocity could be derived from positional data to within a fractional part of 1 mm/sec and, thereforet velocity should not be distributed for applications purposest In factt the absolute velocity components can be derived from the position to approximately 0.2 mm/sec using an 11th order polynomial interpolator on an EF13 file whose epoch interval is 30 minutes and 0.5 mm/sec if the epoch interval is 40 minutest With a 17th order interpolator velocity can be recovered from a 40-minute positional file at 0.1 mm/s in the mean.

The third question was somewhat didactic since the answer was known a priorit Does the 5-cm precision of the EF13 file reduce the orbital accuracy in any significant way? ' ınswer is not

The essence of Section II has to do with the optimum file size for GPS orbitst This is, admittedly, a mundane question. File size is important for many reasonst If the file is too larget it might not fit on a floppy diskt A year of orbit files might not fit on the hard disk; The modem transfer of orbital data at 1,200 or 2,400 baud could be very timetconsumingt The issue of file size is important because most users still have storage media limitations and low-rate modemst If this question had been ignored in 1985, we would have maintained 300-second orbit files instead of the more recent 900-second or 1,200-second orbit filest With ASCII position and velocity formats, this amounts to nearly 4 megabytes per week for 24 satellitest Having completed this study it is now clear that 79 kilobytes per week is all that is required for 24 satellites (on 2,400-second epochs). Below is a file-size table in bytes which comprises the eight formats discussed in this report as a function of epoch interval in secondst In all cases the file spans one GPS week and includes 24 satellitest With 4 hours of overlap at the beginning and end of the file, these values would be approximately 5 percent largert

	300	900	1440	1800	2400
SP1	3933379	1311235	819583	655699	491815
ECF1	2516176	838864	524368	419536	314704
SP2	2191571	730643	456719	365411	274103
ECF2	1354976	451808	282464	222016	169568
EF13	629096	209768	131144	104936	78728
SP3	2966859	989835	619143	495579	371015
ECF3	1645872	549168	343536	274992	206448
EF18	871704	291096	182232	145944	109656

In Section III a new orbital format (ASCII and binary) is proposed which contains position and clock correction data and no velocity datat If adopted, NGS would provide the necessary programs to convert backwards to the current NGS formatst This new format includes orbit accuracy information and is general enough to accommodate changest

CONCLUSIONS

1. With an 1,800-second epoch interval and using an 11th order interpolator, velocity can be derived from positional data to approximately 0.2 mm/sect With a 2,400-second epoch interval and using an 17th order interpolator, velocity can be derived from positional data to approximately 0.1 mm/sect

2. Velocity should not be distributed by NGSt The NGS orbit archive should, nevertheless, remain in the SP1/ECF1 formats for traditional reasons such as being able to share and compare orbital elements with other organizations which generate orbitst

3. An 1,800-second epoch interval position-only ephemeris is consistent with one part per billion geodetic survey activities where an 11th order interpolator is assumed. A 2,400-second epoch interval position-only ephemeris is consistent with one part per billion geodetic survey activities where a 17th order interpolator is assumed.

4. A 2,400-second epoch interval position-only ephemeris is consistent with 0.01 - 0.02 parts per million geodetic survey activities where an 11th order interpolator is assumed. If more accuracy than 0.02 ppm is required, one should switch to a higher order interpolator rather than using a smaller epoch intervalt

5. Without exception, the 2,400-second EF13 file will easilytsatisfy all applications requirements for orbital datat This file consumes a mere 79 kilobytes per week for 24 satellitest With 4 hours of overlap with the prior and the subsequent weekst this would still be less than 83 kilobytest

6. These conclusions require the proper use of the interpolatorstand assume the correct handling of week crossoverst

7. NGS programs are available to manipulate the N[illegible] files in accordance with the user's environmentt

8. Only the 2,400-second SP2 and the 2,400-second EF13 files should be distributed by NGSt The EF13 file should be the primary distribution filet They should include 4 hours of overlap. As explained in the text, the full accuracy can be achieved without the overlap; however, the user has to be more carefult For this reason the overlap is strongly recommended. The data overlap will be easier and safer for the user at a cost of a mere 5 percent in file size. When the user receives the distribution file (e.g.t EF13) he/she can create the data base according to specific needs and wishes with assistance from NGStprovided softwaret

9. NGS should adopt the new SP3/ECF3/EF18 formatst These comprise position and satellite clock corrections and are more flexible than current formatst Intentionally omitted are the component velocitiest Files in these formats can now be generated for experimental purposes at NGS and will betemployed on an experimental basist Should these new formats supersede the current formatst NGS programs for generating files in the current formats (SP1/ECF1; SP2/ECF2/EF13) will be provided to the public. Also, programs similar to those given in this report for the current formats, for the manipulation of files in the proposed formats, would be written and provided to the publict If adopted, the EF18

binary file is recommended to be the primary distribution file, and the recommended epoch interval is 2,400 secondst For 24 satellites its file size is 109,656 bytest It is further recommended that these files span from Saturday 20:00:00 of the prior GPS week to Sunday 04:00 of the subsequent GPS week. These additional 13 epochs, for 24 satellites, increases the EF18 file size by 5,616 bytest

ACI NTS

I would like to express my appreciation to Mr. Everett Swift (NSWC)t Mrt Robert Dulaney (NOAA),t Mrt Charles Challstrom (NOAA)t Drt P. A. M. Abusali (CSR, UT), Drt Mark Schenewerk (NOAA)t and Ltt Cdr David Minkel (NOAA) for carefullyt reading and commenting on the draft manuscript.

BIBLIOGRAPHY

Remondi, Benjamin W.t 1985: Distribution of Global Positioning System ephemerides by the National Geodetic Surveyt Presented at the First Conference on Civil Applications of GPS-ION, Warminister, PA, September 12t National Geodetic Information Branch, NOAA, Rockville, MD 20852t

Remondi, Benjamin W. and Hofmann-Wellenhof, B.t 1989: Accuracy of Global Positioning System broadcast orbits for relative surveyst ______________ Report NOS 132t NGS 45t National Geodetic Information Branch, NOAA, Rockville, MD 20852t

Remondi, Benjamin W. and Hofmann-Wellenhof, B., 1989: GPS broadcast orbits versus precise orbits: A comparison study. Presented at 125th Anniversary General Meeting of the International Association of Geodesy, Edinburgh, Scotland, Aug. 3-12. (Proceedings will be published.)

Swift, E. R., 1985: NSWC's GPS orbit/clock determination system. Proceedings of the First International Symposium on Precise Positioning with the Global Positioning System, Rockville, MD, April 15-19, Vol. 1, pp. 51-62t

****

****