VLBP Memo No. 16

To: VLBP Project group.

From: W. D. Cotton and J. D. Benson

Subject: Simulation of VLBI data.

In order to test various potential correlator designs by means of computer simulation we need simulated VLBI data with known properties. In order to insure realistic data we imposed the following constraints:

1) Each station 'signal' should contain independent station and source noise. The station noise should be independent for the two stations but the source noise should be identical.

2) The fractional bit delay error and phase difference of the source noise for the two stations should be variable.

3) All noise should be broadband and incoherent.

4) The ratio of source noise to station noise should be known.

Of the above criteria, #3 is the most difficult to satisfy. In order to satisfy this criterion we used a model consisting of 10000 monochromatic, coherent oscillators covering the 2 MHz bandpass used which were incoherent with respect to each other. The time-domain data were obtained by a direct fourier transform of the oscillator values at a given time. Independence of the oscillators was obtained by using a random number generator for the real and imaginary parts of the initial state of the oscillators. The amplitude of the noise was set by the random number generator.

This model should be accurate for times short compared to the inverse of the frequency separation between oscillators. Data for longer times can be obtained by occasionally rerandomizing the oscillators although this may cause minor problems near the change. In our implimentation we use up to 4000 0.25 microsecond time steps and the bandpass is assumed to be rectangular.

In order to satisfy criterion #1 we used 3 independent noise generators; one for each station and one for the source. The 'source' data for the second antenna is rotated in the frequency domain by a phase corresponding to the RF frequency (typically 10 GHz) times the geometric delay assumed. In addition, the source data at the second station were evaluated at a time which differed from the first station time by the fractional bit delay error. This procedure simplifies the problem of aligning the data in the correlator simulator but should not otherwise affect the results. After the data are generated they are clipped to one bit.

Since the above process requires a great deal of computing, it was implemented using the FPS-120B array processor on the VAX and the data were written on a disk file to be read later by the correlator simulator. This file contained, in addition to the time and clipped data, the unclipped data, phase difference and fractional bit delay error for each time point.

Following is the source version of the program.

```
      PROGRAM GENRAT
C-----------------------------------------------------------------------
C   GENRAT simulates 1-bit sampled 2 station interferometer data.
C   Data is written on unit 8 with unformatted FORTRAN writes in the
C   following format:
C     Record 1:  LABEL,NPTS    (C*80,I*2)
C         LABEL is a description of how the data was generated.
C
C     Record 2: ITIME,IDELAY,IPHS,ISA,ISB,SA,SB
C         ITIME time count, zero relative.  (I*2)
C         IDELAY is the fractional bit delay in the low order bits
C            in the range +/- 1/2 bit.   I*2
C         IPHS is the interferometer phase in the low order bits
C             in the range (0,2*pi)   (I*2)
C         ISA is the 1-bit sampled data point station A (L*1)
C         ISB is the 1-bit sampled data point station B (L*1)
C         SA is the real*4 unclipped value corresponding to ISA
C         SB is the Real*4 unclipped value corresponding to ISB
C----------------------------------------------------------------------
      INTEGER*4 ITEMP
      INTEGER*2 ITIME,IDELAY,IPHS,NPTS,IAPWRK,IAPSTR,IAPTIM,N1,N2,
     *   NDBITS,NPBITS,LABEL(40),IAPHS
      REAL*8 DELAY,TDELAY,DMOD
      LOGICAL*1 ISA,ISB
      REAL*4 SIGNLA(2000),SIGNLB(2000),TIMEA(2000),TIMEB(2000),
     *   PHASE(2000),FDELAY(2000),FREQ,DELFRE,SEED,FACSIG,FACNS,
     *   RATIO,TWOPI,TEMP
      DATA N1,N2/1,2/,NDBITS,NPBITS/4,4/,NPTS/100/,
     *   IAPWRK,IAPTIM,IAPSTR,IAPHS/0,10,15000,10000/,
     *   RATIO/1.0/,SEED/.251637948/,FREQ/1.27E10/,DELFRE/200./
C---------------------------------------------------------------------
C                             Statment function for delay
      DELAY(XTIME) = XTIME*2.5D-3
C                             Set relative noise levels.
      FACNS = 1.0/600.0
      FACSIG = FACNS*RATIO
      TWOPI = 2.0*3.1415926
      DELFRE = DELFRE*TWOPI
C                             Generate time, delay and phase arrays.
      DO 50 I = 1,NPTS
         TIMEA(I) = (I-1)*0.25E-6
         TDELAY = DELAY(TIMEA(I))
         PHASE(I) = TWOPI*DMOD (TDELAY*FREQ,1.0D0)
         TIMEB(I) = TIMEA(I)+TDELAY
         ITEMP = NINT (TIMEB(I)*4.0E6)
         FDELAY(I) = (TIMEB(I)-ITEMP*0.25E-6)
         TIMEB(I) = TIMEA(I)+FDELAY(I)
         SIGNLA(I) = TIMEA(I)
         SIGNLB(I) = TIMEB(I)
 50      CONTINUE
```

```
C                                  Compute "signal"
        CALL NOISE (IAPWRK,IAPTIM,IAPHS,IAPSTR,FACSIG,NPTS,SEED,
       *   DELFRE,N2,SIGNLA,SIGNLB,PHASE)
C                                  Compute "noise" for each antenna.
        CALL NOISE (IAPWRK,IAPTIM,IAPHS,IAPSTR,FACNS,NPTS,SEED,
       *   DELFRE,N1,TIMEA,TIMEA,PHASE)
        CALL NOISE (IAPWRK,IAPTIM,IAPHS,IAPSTR,FACNS,NPTS,SEED,
       *   DELFRE,N1,TIMEB,TIMEB,PHASE)
C                                  Write results.
        ENCODE (80,1100,LABEL) RATIO,FREQ,NDBITS,NPBITS
  1100  FORMAT(' SIGNAL/NOISE=',F10.5,' FREQUENCY=',E15.5,
       *    ' DELAY',I2,' BITS, PHASE',I2,' BITS ')
C   PRINTER OUTPUT
        WRITE(1,2000) LABEL,NPTS
  2000  FORMAT(1H1/,1X,40A2,' NPTS =',I6/,
       *    ' ITIME,PHASE,FRAC DELAY,IDELAY,IPHS,ISA,ISB,SA,SB')
        WRITE(8) LABEL,NPTS
        DFAC = 2.0**(NDBITS-1)
        PFAC = 2.0**NPBITS
        DO 100 I = 1,NPTS
           SA = SIGNLA(I)+TIMEA(I)
           SB = SIGNLB(I)+TIMEB(I)
           ISA = SA.GT.0.0
           ISB = SB.GT.0.0
           ITIME = I-1
           IDELAY = NINT (DFAC*(FDELAY(I)*8.0E6))
           IPHS = NINT (PFAC*PHASE(I)/TWOPI)
C   PRINTER OUTPUT
        FDELAY(I)=FDELAY(I)*4.0E6
        WRITE(1,2001) ITIME,PHASE(I),FDELAY(I),IDELAY,IPHS,ISA,ISB,
       *   SA,SB,SIGNLA(I),SIGNLB(I)
  2001  FORMAT(I6,2F10.5,4I5,4F10.6)
           WRITE(8) ITIME,IDELAY,IPHS,ISA,ISB,SA,SB
  100      CONTINUE
        WRITE(1,2002) SIGX,SIGY,SIGXY
  2002  FORMAT(' SIGX,SIGY,SIGXY=',3E15.5)
        ENDFILE 8
        STOP
        END
```

```
      SUBROUTINE NOISE (IAPWRK,IAPTIM,IAPHS,IAPSTR,FACTOR,NPTS,SEED,
     *   DELFRE,NTIMES,ARRAY1,ARRAY2,PHASE)
C---------------------------------------------------------------------
C   NOISE generates gaussian noise in the frequency domain and
C   transforms it to the time domain. Does 1Ø,ØØØ cells in the
C   frequency domain, single sideband (upper).
C   INPUTS:
C   IAPWRK  I*2  Base address for 5 element work array in AP
C   IAPTIM  I*2  Base address for time/result array in AP
C   IAPHS   I*2  Base address for additional phase array.
C   IAPSTR  I*2  Base address of frequency array in AP
C                Ø = time Ø real part
C                1 = time Ø imaginary part
C                2 = video frequency (Hz)
C                3 = work
C                4 = work
C   FACTOR  R*4  Amplitude scaling factor
C   NPTS    I*2  Number of input times.
C   SEED    R*4  Seed for random number generator.
C   DELFRE  R*4  Video frequency increment.
C   NTIMES  I*2  Number of input time arrays (1 or 2)
C   ARRAY1  R*4  First array containing times
C   ARRAY2  R*4  Second array containing times.
C   PHASE   R*4  Array of additional phases for second array.
C
C   OUTPUT:
C   SEED    R*4  New seed for random number generator.
C   ARRAY1  R*4  First result array
C   ARRAY2  R*4  Second result array
C-----------------------------------------------------------------------
      INTEGER*2 IAPWRK,IAPTIM,IAPSTR,N1,N2,N3,N4,N5,N6,N1ØK
      REAL*4 WRK(5),FACTOR,SEED,DELFRE,ARRAY1(NPTS),ARRAY2(NPTS),
     *   PHASE(NPTS)
      DATA N1,N2,N3,N4,N5,N6,N1ØK/1,2,3,4,5,6,1ØØØØ/
C-----------------------------------------------------------------------
C                             Prepare work array for AP
      WRK(1) = SEED
      WRK(2) = FACTOR
      WRK(3) = -6.Ø*FACTOR
      WRK(4) = DELFRE
      WRK(5) = Ø.Ø
C                             Aquire AP
      CALL APINIT (Ø,Ø,I)
      CALL APPUT (WRK,IAPWRK,N5,N2)
      CALL APPUT (ARRAY1,IAPTIM,NPTS,N2)
      CALL APWD
C                             Set AP addresses
      IST1 = IAPSTR+1
      IST2 = IAPSTR+2
      IST3 = IAPSTR+3
C                             Clear AP accumulators.
      CALL VCLR (IAPSTR,N5,N1ØK)
      CALL VCLR (IST1,N5,N1ØK)
```

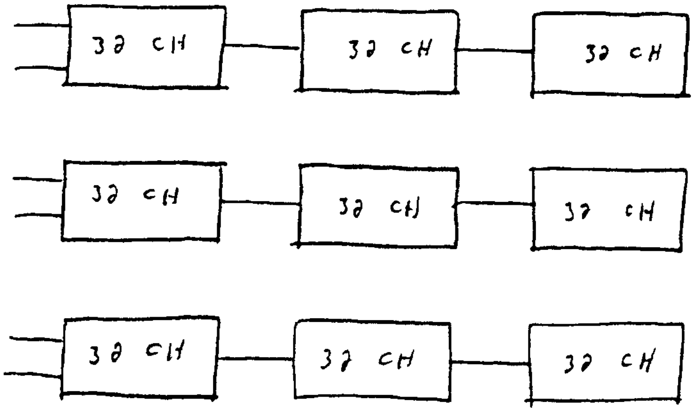
```
C                            Use VRAND to create uniform distributed
C                            values, then add 12 together and subtract
C                            6.0 to simulate gaussian noise.  Do sin
C                            and cosine independently.
      DO 100 I = 1,12
         CALL VRAND (IAPWRK,IST2,N5,N10K)
         CALL VRAND (IAPWRK,IST3,N5,N10K)
         CALL VADD  (IAPSTR,N5,IST2,N5,IAPSTR,N5,N10K)
         CALL VADD  (IST1,N5,IST3,N5,IST1,N5,N10K)
         CALL APWR
 100     CONTINUE
C                            Subtract 6.0 and multiply by FACTOR.
      CALL VSMSA (IAPSTR,N5,IAPWRK+1,IAPWRK+2,IAPSTR,N5,N10K)
      CALL VSMSA (IST1,N5,IAPWRK+1,IAPWRK+2,IST1,N5,N10K)
C                            Put bandpass function here if desired.
C                            Fill video frequencies.
      CALL VRAMP (IAPWRK+4,IAPWRK+3,IST2,N5,N10K)
C                           Zero offset frequencies.
      CALL VCLR (IAPHS,N1,NPTS)
      CALL APWR
C                            Get new seed
      CALL APGET (SEED,IAPWRK,N1,N2)
      CALL APWD
C                            Call VFC routine to compute array.
      CALL NXFORM (IAPSTR,N5,IAPTIM,IAPHS,NPTS,N10K)
      CALL APWR
C                            Get results
      CALL APGET (ARRAY1,IAPTIM,NPTS,N2)
      CALL APWD
C                            See if second array to be done.
      IF (NTIMES.EQ.1) GO TO 999
         CALL APPUT (ARRAY2,IAPTIM,NPTS,N2)
         CALL APPUT (PHASE,IAPHS,NPTS,N2)
         CALL APWD
         CALL NXFORM (IAPSTR,N5,IAPTIM,IAPHS,NPTS,N10K)
         CALL APWR
         CALL APGET (ARRAY2,IAPTIM,NPTS,N2)
         CALL APWD
 999     CALL APRLSE
      RETURN
      END
```

```
"
"
"  The following is a Vector Function Chainer routine for the
"  FPS AP-12ØB array processor.
"
        DEFINE NXFORM (IAPSTR,INC,IAPTIM,IAPHS,OUTCNT,INCNT)
        LOCAL IAP1,IAP2,IAP3,IAP4
"
"   NXFORM GENERATES A RANDOM NOISE SPECTRUM AND DOES A DIRECT
"   FOURIER TRANSFORM TO THE TIME DOMAIN.  IN ORDER TO MAINTAIN
"   AN ACCURATE REPRESENTATION OF AN INCOHERENT SOURCE THE RATIO
"   OF INCNT/OUTCNT SHOULD BE NO LESS THAN ABOUT 2.5
"   INPUTS:
"   IAPSTR  BASE ADDRESS OF ARRAY.
"           Ø = INITIAL REAL PART
"           1 = INITIAL IMAGINARY PART
"           2 = VIDEO ANGULAR FREQUENCY
"           3 = WORK
"           4 = WORK
"   INC  INCREMENT OF IAPSTR
"   IAPTIM    BASE ADDRESS OF TIME/RESULT VECTOR
"   IAPHS     BASE ADDRESS OF ADDITIONAL PHASE VECTOR
"   OUTCNT    NUMBER OF TIMES/RESULTS
"   INCNT     NUMBER OF FREQUENCY RESULTS
"
"          SET ADDRESSES
      SPØ9 = IAPSTR
      IAP1 = SPØ9+1
      IAP2 = SPØ9+2
      IAP3 = SPØ9+3
      IAP4 = SPØ9+4
"          LOOP THROUGH DATA
LOOP: CALL VSMSA (IAP2,INC,IAPTIM,IAPHS,IAP3,INC,INCNT)  "COMPUTE PHASE
      CALL VSIN (IAP3,INC,IAP4,INC,INCNT)             "SIN(PHASE)
      CALL VCOS (IAP3,INC,IAP3,INC,INCNT)
      CALL CVMUL (IAPSTR,INC,IAP3,INC,IAP3,INC,INCNT)"ROTATE
      CALL SVE (IAP3,INC,IAPTIM,INCNT)          "SUM
"          UPDATE POINTER AND COUNTER
      IAPTIM = IAPTIM+1
      IAPHS = IAPHS+1
      OUTCNT = OUTCNT-1
      IF OUTCNT >= Ø GOTO LOOP
      END
```

9 SHALLOWAY MOD IV CORR CARDS:

32 MULT /CARD USED AT 80 MHZ

1	2	3	4	5
A	$A \cdot B_s$	$A \cdot B_s$	$A\ C_c$	$A\ C_c$
B	$A \cdot B_c$	$A \cdot B_c$	$B \cdot C_s$	$B\ C_s$
C	$A\ C_s$	$A \cdot C_s$	$B \cdot C_c$	$B\ C_c$
AUTO	LEAD	LAG	LEAD	LAG



3 ANT

		F_{SAMP}	FREQ CH/BASELINE	POLZ (NO X-PROD)	POLZ
1, B, C		16 MHZ (R=5)	96	—	—
B_s, $A \cdot B_c$	*	8 MHZ	192	96	—
C_s, $A \cdot C_c$	*	4 MHZ	384	192	96
C_s, $B \cdot C_c$					

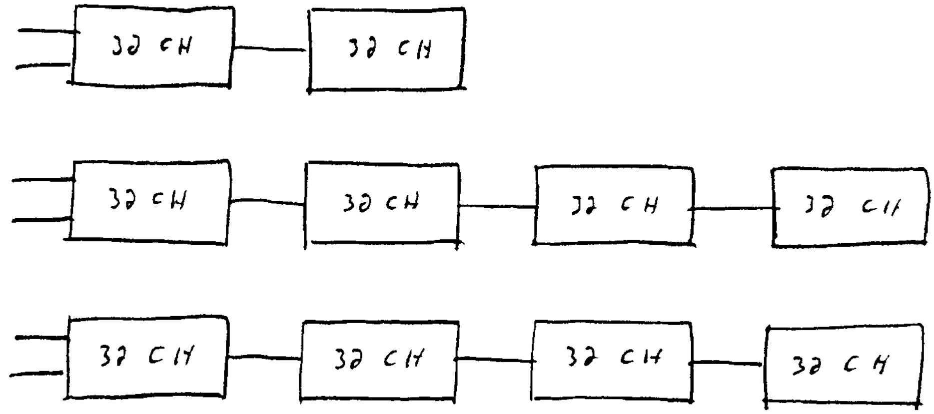
* CAN BE 2 BIT - 3 LEVEL

2 ANT

F_{SAMP}	FREQ CH/B	POLZ (NO X-PROD)	POLZ
16 MHZ	192	—	—
* 8 MHZ	480	240	—
* 4 MHZ	960	480	240

10 SHALLOWAY MOD IV CORR. CARDS

32 MULT/CARD USED AT 96 MHZ



3 ANT

		F_{SAMP}	FREQ CH / BASELINE	POLZ (NO-X PROD)	POLZ
A, B, C		16 MHZ	128	—	—
A·Bs, A·Bc, B·Cs, B·Cc	*	8 MHZ	256	128	—
A·Cs, A·Cc	*	4 MHZ	512	256	128

2 ANT

	F_{SAMP}	FREQ CH./BASELINE	POLZ (NO X PROD)	POLZ
	16 MHZ	288	—	—
*	8 MHZ	576	288	—
*	4 MHZ	1152	576	288

* CAN BE 2 BIT - 3 LEV