K5/VSSP および K5/VSSP32 用
ソフトウェア
取り扱い説明書

近藤哲朗
情報通信研究機構

2015 年 12 月 22 日版

目 次

1 動作確認環境

K5/VSSP および K5/VSSP32 サンプラー用ソフトウェアの動作確認環境は以下の通りです。

サンプラー	動作確認環境
K5/VSSP	FreeBSD 4.0-RELEASE i386 プラットフォーム Debian GNU/Linux 3.1 (kernel 2.6.8-2-386) i386 プラットフォーム
K5/VSSP32	Debian GNU/Linux 3.1 (kernel 2.6.8-2-386) i386 プラットフォーム

2 ソフトウェアのインストール法

グラフィック表示に PGPLOT または GNUPLOT を使用しますので[1]、予め PGPLOT または GNUPLOT がインストールされている必要があります。

PGPLOT を使用する場合は、2.4 節の PGPLOT のインストール法を参考に PGPLOT をインストール後、以下のプロセスを実行してください。

GNUPLOT は大抵の場合予めインストールされていると思いますが、インストールされていない場合は適宜インストールを行い（GNUPLOT のインストール法はここでは触れません）、gnuplot コマンドにパスを通して下さい。

また、予めサンプラーのドライバーはインストールされている必要があります。ドライバーのインストールは 2.3 節を参照してください。

2.1 観測ソフトウェア群のインストール法

2.1.1 シェルスクリプトによるインストール法

適当なディレクトリに観測ソフトウェア群のアーカイブファイル　ipvlbi_obsXXXXXXXX.tar.gz（ここで XXXXXXXX はアーカイブ年月日情報）を置き、

```
tar xvzf ipvlbi_obsXXXXXXXX.tar.gz
```

で解凍後、

```
cd ipvlbiXXXXXXXX
```

で作業ディレクトリを移動し、従来の K5/VSSP(IP-VLBI) システムでは、

```
install_obs.sh
```

を実行してください。この場合はグラフィック表示には PGPLOT が使用されますが、GNUPLOT を使用する場合は

```
install_obs.sh G=GNUPLOT
```

を実行して下さい。K5/VSSP32 システムでは、

```
install_obs_vssp32.sh
```

を実行してください。PGPLOT の代わりに GNUPLOT を使用したい場合は

```
install_obs_vssp32.sh G=GNUPLOT
```

を実行してください。

FreeBSD システムでは

[1] Ver.2007-11-14 以降ではグラフィック表示に PGPLOT の他に GNUPLOT が使えるようになりました。ただ GNUPLOT は実時間表示に難があるため、pcalcheck のように処理をしながら表示を行うソフトの場合は PGPLOT の使用を推奨します。

```
/usr/lib/libg2c.so: warning: tempnam() possibly used unsafely; consider using mkstemp()
```

という警告メッセージが出ますが、これはインストールされている PGPLOT パッケージ中の関数に関する警告ですので無視してください。

linux システムにてコンパイル時に警告メッセージが多く出る場合は、

```
install_obs.64.sh
```

をお使い下さい（install_obs.64.sh はファイルオープンに fopen64 を使用したコンパイルを行います)。GNUPLOT を使用したい場合は

```
install_obs.64.sh G=GNUPLOT
```

を使って下さい。

2.1.2 makefile によるインストール法

適当なディレクトリに観測ソフトウェア群のアーカイブファイル ipvlbi_obsXXXXXXXX.tar.gz（ここで XXXXXXXX はアーカイブ年月日情報）を置き、

```
tar xvzf ipvlbi_obsXXXXXXXX.tar.gz
```

で解凍後、

```
cd ipvlbiXXXXXXXX/src
```

で作業ディレクトリを移動します。linux の場合、make の仕方は以下の通りです。

```
make [S=VSSP32|VSSP(default)|NONE] [G=GNUPLOT] [X=64] [OPEN=64]

   ここで S=VSSP32   ---  K5/VSSP32 サンプラーユニット用に観測ソフトをコンパイルする場合
          S=VSSP     ---  K5/VSSP サンプラーボード用 にコンパイルする時 (デフォルト)
          S=NONE     ---  サンプラーを直接制御しないプログラムのみをコンパイルする場合
          G=GNUPLOT  ---  グラフィック表示に GNUPLOT を使用する場合
                          （デフォルトは PGPLOT）
          X=64       ---  64bit CPU を使用する場合（/usr/X11R6/lib64 を使用）
          OPEN=64    ---  fopen64 を使用する場合
```

FreeBSD の場合は、以下のように makefile として BSDmakefile を指定して下さい。

```
make -f BSDmakefile [S=VSSP32|VSSP(default)|NONE] [G=GNUPLOT] [X=64] [OPEN=64]

   ここで S=VSSP32   ---  K5/VSSP32 サンプラーユニット用に観測ソフトをコンパイルする場合
          S=VSSP     ---  K5/VSSP サンプラーボード用 にコンパイルする時 (デフォルト)
          S=NONE     ---  サンプラーを直接制御しないプログラムのみをコンパイルする場合
          G=GNUPLOT  ---  グラフィック表示に GNUPLOT を使用する場合
                          （デフォルトは PGPLOT）
          X=64       ---  64bit CPU を使用する場合（/usr/X11R6/lib64 を使用）
          OPEN=64    ---  fopen64 を使用する場合
```

2.2 K5/VSSP と K5/VSSP32 の同時インストール

K5/VSSP32 をインストールしている P C には K5/VSSP サンプラー用の観測ソフトをインストールすることが出来ます。以下の手順でインストールすると、K5/VSSP と K5/VSSP32 を同時に使用することも可能になります。

2.2.1 K5/VSSP を常用とする場合の同時インストール

(1) 両方のドライバ (K5/VSSP 用は linux2.6 用ドライバを使用する）をインストールする
(2) cd ipvlbiXXXXXXXX/src で作業ディレクトリを移動する
(3) make install -f makefile.vssp32 で観測ソフト群を K5/VSSP32 用にコンパイルする
(4) (3) の作業の結果として　ipvlbiXXXXXXXX/bin32 以下に K5/VSSP32 用の観測ソフト（実行プログラム）群が収納される
(5) ipvlbiXXXXXXXX/src ディレクトリで今度は　make clean を実行する
(6) 更に make install を実行する。この作業により観測ソフト群が K5/VSSP（旧 IP-VLBI ボード）用にコンパイルされる
(7) (6) の作業で ipvlbiXXXXXXXX/bin 以下に K5/VSSP 用の観測ソフト（実行プログラム）群と両方 (VSSP および VSSP32）に共通に使用できるプログラム群が収納される。また ipvlbiXXXXXXXX/bin_vssp 以下にも K5/VSSP 用の実行プログラムが収納される

2.2.2 K5/VSSP32 を常用とする場合の同時インストール

前節 (2.2.1) の手順で先に K5/VSSP 用ソフト群をインストールし、後で K5/VSSP32 用ソフト群をインストールする。具体的な手順は以下の通り

(1) 両方のドライバ (K5/VSSP 用は linux2.6 用ドライバを使用する）をインストールする
(2) cd ipvlbiXXXXXXXX/src で作業ディレクトリを移動する
(3) make install で観測ソフト群を K5/VSSP 用にコンパイルする
(4) (3) の作業の結果として　ipvlbiXXXXXXXX/bin_vssp 以下に K5/VSSP 用の観測ソフト（実行プログラム）群が収納される
(5) ipvlbiXXXXXXXX/src ディレクトリで今度は　make clean を実行する
(6) 更に make install -f makefile.vssp32 を実行する。この作業により観測ソフト群が K5/VSSP32 用にコンパイルされる
(7) (6) の作業で ipvlbiXXXXXXXX/bin 以下に K5/VSSP32 用の観測ソフト（実行プログラム）群と両方 (VSSP および VSSP32）に共通に使用できるプログラム群が収納される。また ipvlbiXXXXXXXX/bin32 以下にも K5/VSSP32 用の実行プログラムが収納される。

上記 (2.2.1 および 2.2.2) の手順で ipvlbiXXXXXXXX/bin 以下には最後にコンパイルした観測ソフト（実行プログラム）群と両方 (VSSP および VSSP32）に共通に使用できるプログラム群が収納され、ipvlbiXXXXXXXX/bin_vssp 以下に K5/VSSP 用の観測ソフト（実行プログラム）群、ipvlbiXXXXXXXX/bin32 以下に K5/VSSP32 用の観測ソフト（実行プログラム）群が収納される。従って、例えば signalcheck を K5/VSSP32 ユニットに対して実行するときは

```
ipvlbiXXXXXXXX/bin32/signalcheck
```

K5/VSSP サンプラーに対して実行するときは

```
ipvlbiXXXXXXXX/bin_vssp/signalcheck
```

のようにする。なお、ipvlbiXXXXXXXX/bin 以下には最後にコンパイルしたボードに対する実行プログラムが収納される。

2.3 ドライバーのインストール法

K5/VSSP 用のドライバーは FreeBSD 用と linux2.6 用のドライバーがあります。K5/VSSP32 用ドライバーは今の所、 Debian GNU/Linux 3.1 (kernel 2.6.8-2-386) i386 プラットフォーム用のみです。
ドライバーの最新バージョンは以下で入手して下さい。

```
http://www2.nict.go.jp/w/w114/stsi/K5/VSSP/install_obs.html
```

2.3.1 K5/VSSP 用ドライバー (FreeBSD 用) のインストール法

1. OS(FreeBSD) は、すでに組込まれているものとします。ＰＣの電源を切った状態で 9260 IP-VLBI 用 TimeSyncDataSampler ボードをＰＣＩスロットに挿入します。
2. ドライバソースを展開します。ユーザー vlbi でログインしてください。ログイン後、ユーザーホームディレクトリに src ディレクトリを作成します。

   mkdir src
   mkdir bin
   mkdir schedule

   ドライバ CDROM を挿入して、マウントします。

   su
   <root パスワード入力 >
   mount /cdrom
   exit

   ドライバソースプログラムを CDROM からコピーして展開します。

   cp /cdrom/tds_driverYYYYMMDD.tar.gz ./src/
   cd src
   tar xvzf tds_driverYYYYMMDD.tar.gz

   (YYYYMMDD は西暦年、月、日を表します)
   これで src ディレクトリ内に tds_driver ディレクトリが作成され、ソースコードが展開されます。
3. ドライバ組込みの準備をします。ユーザーホームディレクトリの src ディレクトリ内で作業を進めてください。ドライバの組込みを行いますので、管理者権限を取得して作業を進めます。シェルスクリプトを CDROM からコピーして実行します。
   管理者権限を取得します。

   su
   <root パスワード入力 >
   cp /cdrom/*.sh ./
   ./tds_driver_cp.sh

   その後、いくつかのファイルにドライバ情報を追記します。
   /usr/src/sys/conf/files.i386 ファイルに以下の行を追記します。

```
pci/tds.c             optional      tds
```

/usr/src/sys/conf/majors ファイルに以下の行を追記します。

```
222                tds           Time Sync Data Sampler (Nitsuki)
```

/usr/src/sys/i386/conf/VLBI ファイルに以下の行を追記します。

```
device             tds           # Time Sync Data Sampler (Nitsuki)
```

例） vi エディタを使用します。

vi /usr/src/sys/conf/files.i386
[PageDown] キーで最後の行、[End] キーで最後の文字にカーソルを移動し、
[a] キーで、追記モードにします。
[Enter] キーで改行した後、

```
  pci/tds.c                optional        tds
```

を入力し、[Esc] キーでコマンドモードにします。
:wq [Enter] でファイルをセーブして終了します。
次に、
vi /usr/src/sys/conf/majors
[PageDown] キーで 222 の前の数字で始まる行、[End] キーで最後の文字に
カーソルを移動し、[a] キーで、追記モードにします。
[Enter] キーで改行した後、

```
  222                tds           Time Sync Data Sampler (Nitsuki)
```

を入力し、[Esc] キーでコマンドモードにします。
:wq [Enter] でファイルをセーブして終了します。
同様に、
vi /usr/src/sys/i386/conf/VLBI
[PageDown] キーで最後の行、[End] キーで最後の文字にカーソルを移動し、
[a] キーで、追記モードにします。
[Enter] キーで改行した後、

```
  device             tds           \# Time Sync Data Sampler (Nitsuki)
```

を入力し、[Esc] キーでコマンドモードにします。
:wq [Enter] でファイルをセーブして終了します。

4. Kernel の再構築を行います。以下のシェルスクリプトを実行します。
   ./tds_driver_mk.sh
   再構築している間、しばらくお待ちください。（約６分かかります）
5. システムを再起動します。再構築が終了して、プロンプトが表示されたら、以下のコマンドで再起動します。
   shutdown -r now

これで、ドライバが組込まれました。デバイスファイルは /dev/tds0 となります。

９２６０用ライブラリ（日通機提供 K5/VSSP 用ライブラリ）のインストール

1. OS(FreeBSD)、およびドライバは、すでに組込まれているものとします。ユーザー vlbi でログインしてください。ログイン後、ユーザーホームディレクトリ下の src ディレクトリで作業します。cd src
   管理者権限を取得します。su ¡パスワード入力¿
2. 以下のシェルスクリプトを実行します。./tds_lib_mk.sh

ライブラリの構築は、数秒で終了します。これで、ライブラリがインストールされました。

2.3.2 K5/VSSP 用ドライバー (linux2.6 用) ドライバーのインストール法

K5/VSSP サンプラーボードを Debian GNU/Linux 3.1（kernel 2.6）で動作させるためのデバイスドライバのインストール法は以下の通りです。

1. 動作環境
   Debian GNU/Linux 3.1 (kernel 2.6)
2. インストール
   (a) kernel source 類をインストール apt-get からインストール
   ・Linux kernel headers 2.6 on 386
   # apt-get install kernel-headers-2.6-386
   ・Linux kernel image for version 2.6 on 386
   # apt-get install kernel-image-2.6-386

   (b) ドライバインストール

   デバイスドライバ (tds_linux_26.tgz) を取得する
   ドライバソースの展開
   % tar xvzf tds_linux_26.tgz
   ドライバの組み込み
   % cd ./tds/driver
   # make install <—ドライバのインストール
   # make load <—ドライバのロード
   # cd ../libtds
   # make install <—ライブラリのインストール
   # mknod /dev/tds0 c 222 0 <—デバイスノードの作成

2.3.3 K5/VSSP32 用ドライバーのインストール法

K5/VSSP32 サンプラーユニットを Debian GNU/Linux 3.1（kernel 2.6）で動作させるためのデバイスドライバのインストール法は以下の通りです。

1. ローダブルモジュールの作成

   (a) vlbi-usb-linux.tar.gz を展開して vlbi-usb-linux に移動する。

```
# tar xvfz vlbi-usb-linux.tar.gz
# cd vlbi-usb-linux/
```

   (b) モジュールを構築、インストールする。この時ライブラリも同時にビルド、インストールされます。

```
# make
# make install
```

   (c) デバイスノードを作成する。

```
# cd /dev
# mknod -m 666 utds0 c 180 222
```

デバイスファイルは、/dev/utds0 となります。もし Linux 上で udevd が動作している場合はデバイスノードを作成する必要はありません。

(d) モジュールをロードする。

```
# insmod /lib/modules/2.6.8-2-386/kernel/drivers/usb/misc/utds.ko
```

make install を実行する事によって次のファイルをそれぞれの場所にインストールします。

tds.ko	/lib/modules/2.6.8-2-386/kernel/drivers/usb/misc/utds.ko
tdsio.h	/usr/include/sys/tdsio.h
tdssdh.h	/usr/include/sys/tdssdh.h
libtds.so	/usr/local/lib/libtds.so

2.4 PGPLOT のインストール法

2.4.1 FreeBSD の port を使用する方法

ネットワーク環境（caltech のサーバにアクセスできる) をととのえてから、

```
cd /usr/ports/graphics/pgplot
make install
```

2.4.2 ステップバイステップ法

1. /usr/local/src に pgplot5.2.tar.gz をコピー
   入手先は ftp://ftp.astro.caltech.edu/pub/pgplot/pgplot5.2.tar.gz
2. 解凍する
   cd /usr/local/src
   gunzip -c pgplot5.2.tar.gz | tar xvof -
   または tar xvzf pgplot5.2.tar.gz
   解凍後、/usr/local/src/pgplot および　サブディレクトリが出来る
3. インストール先ディレクトリ作成
   mkdir /usr/local/pgplot
4. デバイスドライバーの選択
   cd /usr/local/pgplot
   cp /usr/local/src/pgplot/drivers.list .
   エディター (vi 等）で選択するドライバーの先頭の”!”をはずす
   vi drivers.list
   ここで、GIF 関係のドライバーや /FILE を選択すると、FreeBSD ではエラーが出ます (linux では未確認)
   おすすめドライバーの選択は
   /LATEX /NULL /PS /VPS /CPS /VCPS /TEK4010 /RETRO /GF /GTERM /XTERM /TK4100 /VT125 /XWINDOW /XSERVE
5. makefile の作成
   cd /usr/local/pgplot
   FreeBSD の場合
   /usr/local/src/pgplot/makemake /usr/local/src/pgplot freebsd
   linux の場合
   /usr/local/src/pgplot/makemake /usr/local/src/pgplot linux g77_gcc

6. makefile の編集
   linux の場合は以下の作業（makefile の編集）は不要です
   FreeBSD の場合 ５．で作った makefile で make を行うと、demo2 プログラムのコンパイル時にエラーが発生して、停止してしまう。そのため、X端末に表示をする際に必要な pgxwin_server が作られません。
   そこで、vi 等で demo2 以降のデモプログラムをコンパイルしないように編集します
   vi makefile
   # Routine lists の中の
   DEMOS= pgdemo1 pgdemo2 .....
   の行の pgdemo2 以降を削除し、
   DEMOS= pgdemo1
   とする
7. make の実行
   make
   更に C で使用するため環境を整えるため
   make cpg を実行
   make clean を実行
   この時点で以下のファイルがディレクトリに存在
   cpgdemo grexec.f libcpgplot.a pgdemo1 pgxwin_server cpgplot.h grfont.dat libpgplot.a pgplot.doc rgb.txt drivers.list grpckg1.inc makefile pgplot.inc
   linux 版では更に libpgplot.so も出来ています
8. ライブラリおよびCインクルードファイルを標準ディレクトリへコピー
   cp libcpgplot.a /usr/lib
   cp libpgplot.a /usr/lib
   cp cpgplot.h /usr/include
   linux では更に
   cp libpgplot.so /usr/lib
9. デモプログラムの実行
   環境変数を以下のように設定
   csh の場合 setenv PGPLOT_DIR /usr/local/pgplot/
   sh の場合 PGPLOT_DIR=”/usr/local/pgplot/”; export PGPLOT_DIR
   または export PGPLOT_DIR=/usr/local/pgplot/
   デバイスに/XWINDOW または/XSERVE を使用する場合は更に以下（表示先）を設定
   csh の場合 setenv DISPLAY IP アドレス又はマシン名:0.0
   sh の場合 export DISPLAY=IP アドレス又はマシン名:0.0
   pgdemo1（FORTRAN デモ）
   または
   cpgdemo（C プログラムデモ）を実行する
   注意事項：Tera Term から実行するときは
   「Setup」− >「Terminal」で「Auto switch」にチェックが入っていることを確認すること
10. C プログラムのコンパイル方法以下のいずれかの方法でコンパイルする。プログラム名は test.c とする

    方法１
    f77 -o test test.c -lcpgplot -lpgplot -L/usr/X11R6/lib -lX11 -lm
    （注：PGPLOT は元々FORTRAN で開発されたプログラムなので FORTARN コンパイラーおよびリンカーでコンパイルしないと駄目なよう。実際、cc -o test .... でコンパイル、リンクをやると undefined reference to ’xxxxx’ というエラーがたくさん出る）

方法 2

```
cc -O2 -c -I. test.c
f77 -o test test.o -lcpgplot -lpgplot -L/usr/X11R6/lib -lX11 -lm
```

3 観測およびユーティリティソフトウェア

ここでは K5/VSSP および K5/VSSP32 用観測ソフトおよび各種ユーティリティソフトウェアについて説明します。大半のソフトウェアは　プログラム名 [リターン]　で、簡単な使用法の説明が表示されます。また　プログラム名 env[リターン] でプログラムが使用する環境変数に関する情報が得られます。

3.1 ソフトウェア一覧

K5/VSSP および K5/VSSP32 用ソフトウェア（プログラム）はその機能から、サンプラーの時刻セットプログラム、観測準備プログラム、観測実行プログラム、データチェックプログラム、そしてサンプラー試験用シェルスクリプトに大別されます。

3.1.1 時刻セットソフト群

pctimeset – ＰＣの時刻を K5/VSSP（含む K5/VSSP32）ボードの時刻に合わせる（ログ出力あり）。なお観測制御にはボードの時刻を使うのでＰＣの時刻を必ずしも合わせる必要はない
signalcheck – ボードに 10MHz および 1PPS が供給されているかどうかのチェック
timeadjust – ボードの時刻を秒単位で進めたり遅らせたりする（ログ出力あり）
timecheck – K5/VSSP32 時刻表示の誤動作チェックプログラム（ログ出力あり）
timedisp – ボードの時刻表示
timesetpc – ＰＣの時刻を使ってボードの時刻を合わせる。ボードチェック用であり、ＶＬＢＩ観測時には timesetpc は決して使用しないこと！
timesettk – ボードの時刻のセット（ログ出力あり）
timesync – ボードの１ＰＰＳ同期および時刻表示（ログ出力あり）

3.1.2 観測準備ソフト群

monit – K5/VSSP（含む K5/VSSP32）ボードの入力信号レベルのモニター
setdcoffset – K5/VSSP32 サンプラーのＤＣオフセット調整を行う
skdchk – スケジュールファイルのチェック

3.1.3 データチェック（編集）ソフト群

adbitconv – サンプリングデータファイルの AD ビット数を任意の AD ビット数に変換する
aux_recov – K5/VSSP32 データでデータチェック時に AUX MISALIGN (misaligned AUX field) と表示されるデータの修復を行う
datachk – データチェック （ヘッダー部を頼りにヘッダー間のデータビット数をカウント）
datacut – サンプリングデータファイルからある時間範囲のデータを抜き出す
datatime_edit – K5/VSSP および K5/VSSP32 データヘッダー部の時刻情報の書き換えを行う
data_double – サンプリングデータを繰り返すことにより見かけのサンプリング周波数を倍にする
data_half – サンプリングデータを間引きすることにより見かけのサンプリング周波数を半分にする
data_recov – K5/VSSP ヘッダー部の修復を行う
extdata – サンプリングデータファイルから特定のｃｈのデータを抽出しテキストファイルに書き出す
four2one – ４ ch モードで収集したデータから任意のチャンネルのデータを切り出し、１ ch モードのデータに変換する

k5tok5v32 – K5/VSSP フォーマットデータを K5/VSSP32 フォーマットに変換する
k5v32tok5 – K5/VSSP32 フォーマットデータを K5/VSSP フォーマットに変換する
one2four – ４台のＰＣで 1ch モードで収集したデータファイル４つを結合して、4ch モードのデータに変換する
oscillo – K5/VSSP および VSSP32 データファイルのサンプリングデータを表示（ダイナミック表示）(2015-12-22 新規追加)
pcalcheck – K5/VSSP および VSSP32 データファイルから PCAL を検出しグラフ表示する (2007-10-25 新規追加)
speana – オフラインスペクトル表示
speana2 – オフラインスペクトル表示 (高機能版)
speana_n – オフラインスペクトル表示（ダイナミック表示）(2015-12-22 新規追加)
vssplogana – datachk のサマリー出力ファイルおよび sampling（autoobs も可）のログファイルの解析を行う

3.1.4 K5/VSSP32 ユニット（ボード）チェック用シェルスクリプト

vssp32test.sh – K5/VSSP32 ユニットのサンプリング周波数を変えてのテストを自動的に行い、エラー発生の統計結果を表示する。
vssp32test2.sh – K5/VSSP32 ユニットのサンプリングパラメータを固定して、繰り返しの耐久テストを行い、エラー発生の統計結果を表示する。
vssp32test3.sh – K5/VSSP32 ユニットのサンプリング周波数を変えてのテストを自動的に行い、エラー発生の統計結果を表示する。サンプリング周波数の最低と最高を指定できる。

3.2 時刻合わせソフトの使い方

3.2.1 pctimeset

ユーティリティ名

pctimeset

機能

K5/VSSP（含む K5/VSSP32）ボードの時刻を使ってＰＣの時刻をセットする。（注！ 逆の機能の timesetpc も提供するが、ボードチェック用であり、ＶＬＢＩ観測時には timesetpc は決して使用しないこと！）

時刻をセットするにはスーパーユーザーとなってから実行すること。スーパーユーザーでない場合は、時刻あわせはできないが、K5/VSSP（含む K5/VSSP32）ボードとＰＣの時刻ずれのモニターができる。

補足：観測制御には K5/VSSP（含む K5/VSSP32）ボードの時刻が使用されるのでＰＣの時刻が必ずしも合っている必要性はない

実行方法

pctimeset
環境変数のモニター　pctimeset env

環境変数

K5LOGDIR – ログディレクトリの指定
プログラムデフォルトはカレントディレクトリ

ログ出力

ディレクトリ – 環境変数 K5LOGDIR でセットしたディレクトリ。
K5LOGDIR がセットされていないときはカレントディレクトリ
ファイル名 – YYYY+ホスト名+“.log” 例：2003k51d.log
出力形式 – timesettk の項を参照のこと

出力例

```
03169083823:p pctimeset(1.00)
03169083825/p System Time : 2003/06/18 08:38:24
03169083825/p Offset (IP-Board time - PC time)  -0.000103 sec was adjusted!
03169083825/p Current Offset (IP-Board time - PC time) is 0.000032 sec
03169083825/p pctimeset completed
```

3.2.2 signalcheck

ユーティリティ名

signalcheck

機能

K5/VSSP（含む K5/VSSP32）ボードに 10MHz および 1PPS が供給されているかどうかをチェックする

実行方法

signalcheck

実行例

```
K5 k56a>./signalcheck
 signalcheck (Ver. 2007-08-23)

   K5/VSSP32 (IP-VLBI) board Reference and 1PPS signal checking..

    Reference (10MHz) signal is OK!   PLL locked!
    1PPS  signal is OK!

    VSSP32 FPGA Version = 0.00
```

備考

時刻セットに際しては、10MHz および 1PPS が供給されている必要があります。観測中は 10MHz のみ供給されていればＯＫです。FPGA のバージョンは VSSP32 のファームウェアのバージョンが 2007-08-23 (FPGA のバージョン 1.19) 以降でないと正しく表示されません。

3.2.3 timedisp

ユーティリティ名

timedisp

機能

K5/VSSP（含む K5/VSSP32）ボードの時刻を表示する

実行方法

timedisp [period]

ここで　period　–　表示終了までの時間 (sec)
無指定の場合１０秒間ボードの時刻が表示されます。

3.2.4 timesettk

ユーティリティ名

timesettk

機能

K5/VSSP（含む K5/VSSP32）ボードの時刻をセットする（ログ出力あり）

実行方法

timesettk year month day hour min sec

ここで　year　–　年（４桁）
month　–　月
day　–　日
hour　–　時
min　–　分
sec　–　秒

環境変数のモニター　timesettk env

環境変数

K5LOGDIR　–　ログディレクトリの指定
プログラムデフォルトはカレントディレクトリ

実行例

２００１年9月２０日１５時１０分１５秒にセット

```
timesettk 2001 9 20 15 10 15
```

リターンのタイミング：
K5/VSSP の場合　　実時刻が　１５秒となった直後
K5/VSSP32 の場合　　実時刻が　１４秒となった直後

時刻セットが成功すると、数秒間ボードの時刻が表示されますので正しいか確認する。

ログ出力

ディレクトリ　– 環境変数 K5LOGDIR でセットしたディレクトリ。
K5LOGDIR がセットされていないときはカレントディレクトリ
ファイル名　– YYYY+ホスト名+“.log” 例：2003k51d.log
出力形式　– YYDDDHHMMSSXY コメント
ここで　YY　：　年（２桁）
DDD　：　通日
HH　：　時
MM　：　分
SS　：　秒
X　：　セパレータ　（ :, /, ”, ?, #, (space) ）
Y　：　時刻モード
(space) –　サンプラーの時刻
‘p’ – PC の時刻

出力例

```
03168000313:p timesettk(1.10) 2003 6 17 23 58 30
```

3.2.5　timesync

ユーティリティ名

timesync

機能

K5/VSSP（含む K5/VSSP32）ボードの秒を外部１ PPS 信号に同期させる。その後時刻を表示する

実行方法

timesync [period]

ここで　period　– 表示終了までの時間 (sec)
無指定の場合１０秒間ボードの時刻が表示されます。

環境変数のモニター　timesync env

環境変数

K5LOGDIR　– ログディレクトリの指定
プログラムデフォルトはカレントディレクトリ

ログ出力

ディレクトリ　– 環境変数 K5LOGDIR でセットしたディレクトリ。
K5LOGDIR がセットされていないときはカレントディレクトリ
ファイル名　– YYYY+ホスト名+“.log” 例：2003k51d.log
出力形式　– timesettk の項を参照のこと

出力例

```
03168000240:p timesync(1.10)
```

3.2.6 timeadjust

ユーティリティ名

timesadjust

機能

K5/VSSP（含む K5/VSSP32）ボードの時刻を秒単位の進み遅れでセットする（ログ出力あり）

実行方法

timeadjust nsec [SYNC—sync]

ここで　nsec　– 秒単位で進ませる時間（正）または遅らせる時間（負）を指定する)
SYNC | sync – 1PPS 信号と同期を取るときに指定する。デフォルトは時刻のみの変更（秒の時刻ラベルのジャンプのみ時系は連続に保たれる)。

環境変数のモニター　timeadjust env

環境変数

K5LOGDIR – ログディレクトリの指定
プログラムデフォルトはカレントディレクトリ

実行例

時刻を１秒進ませる。時系のリセットは行わない。

```
timeadjust 1
```

時刻を１秒遅らせる。時系のリセットは行わない。

```
timeadjust -1
```

時刻を２秒進ませる。1PPS との同期も行う。

```
timeadjust 2 SYNC
```

リターンのタイミングはいつでも良い

3.2.7 timecheck

ユーティリティ名

timecheck

機能

K5/VSSP32 サンプラーからの時刻読みとりのチェックを行う（ログ出力あり）

実行方法

timecheck mode

ここで　mode – サンプラーから時刻を読み出すモード
1: 二度読みをしない
2: 時刻がおかしいと思われるときは二度読みをする

プログラムを終了させるには ctrl C

環境変数のモニター　timecheck env

環境変数

K5LOGDIR – ログディレクトリの指定
プログラムデフォルトはカレントディレクトリ

実行例

```
> timecheck 1
K5/VSSP32 Sampler Board Time Display and Check
 Mode : single reading   (ctrl C for stop)

timecheck: time icrease abnormaly  oldsec= 10495  newsec= 10751  <== 時刻の異常があ
るとこのように表示される
timecheck: time icrease abnormaly  oldsec= 10751  newsec= 10496
2007/079 02:07:10
```

3.3 観測準備ソフトの使い方

3.3.1 monit

ユーティリティ名

monit

機能

K5/VSSP（含む K5/VSSP32）ボードの入力信号レベルのモニター。８ビット×１MHｚ×４ｃｈ　または　１ｃｈサンプリングデータのレベル（電圧）をヒストグラム形式で実時間表示する。また平均値と標準偏差も表示する。K5/VSSP32 ではＤＣオフセットを自動的に設定することもできる。ＤＣオフセットをセットした場合にはログ出力（ログファイル名：YYYY ＋ホスト名＋.log）が行われる。

実行方法

monit numch [AUTO—ZERO] [NOTIME]

ここで	numch	–	サンプリングチャネル数　　1 または　4
	AUTO｜ZERO	–	AUTO ：K5/VSSP32 においてＤＣオフセットを自動的に設定する
			ZERO ：ＤＣオフセットをリセットする
	NOTIME		ボードの時刻表示を抑制する

注１： K5/VSSP の場合１ｃｈサンプリング時はメインボードからの入力４ｃｈサンプリング時は補助ボードからの入力となる DC オフセットの設定は K5/VSSP32 のみの機能である

注２： K5/VSSP に対して monit を実行した場合、PC との相性によってはハングする場合がある。その時は “NOTIME” オプションを指定して時刻表示を抑制するとハングしなくなる。

プログラムのＳＴＯＰは CTRL-C で行う。なお、入力信号のチェックの前に１０MHｚおよび１ＰＰＳ信号が供給されているかどうかのチェックも行い、何れかの信号が供給されていないときは、警告メッセージを出しプログラムを終了する。

注意！CTRL-C でたまにシステムが応答不能状態に陥ることがあります。その時は仕方がないので、ハードリセットを行ってください。

環境変数

K5LOGDIR　–　ログディレクトリの指定 (プログラムデフォルトはカレントディレクトリ)

実行例

```
./monit 4

**********************************************************************
*                                                                    *
*        MONIT (signal level monitor for K5/VSSP or K5/VSSP32)       *
*                by T.Kondo/NICT  (Ver. 2008-02-13)                  *
*                    Target Board is K5/VSSP32                       *
*                                                                    *
*                        CTRL C  for STOP                            *
*                                                                    *
*******T*************************************************KONDO********
                                          2008/02/13 (044) 05:34:36
  DC_OFST_P        0            0            0           0

             CH 1         CH 2         CH 3         CH 4
   *
   + FULL   |******      |***         |****        |***
            |******      |****        |*****       |*****
            |********    |********    |*******     |*******
            |**********  |*********   |*********   |*********
            |*********   |**********  |**********  |**********
            |********    |********    |********    |*******
            |****        |****        |****        |****
   - FULL   |*****       |***         |****        |***
  DC OFFSET     2.6/256     -1.3/256     -1.1/256      0.1/256
  ONE SIGMA    67.8/256     61.8/256     65.1/256     62.4/256
```

注：DC オフセット (DC OFFSET) の表示は信号レベルを-127.5〜+127.5（８ビットサンプリング）で表した場合の平均値を示している。標準偏差 (ONE SIGMA) も信号レベルを-127.5〜+127.5（８ビットサンプリング）で表した場合の標準偏差を表している。

3.3.2 skdchk

ユーティリティ名

skdchk

機能

スケジュールファイルのチェック

スケジュールファイルを読み、局情報、電波星情報、観測情報を出力する。最後に、全観測時間を秒で示すと共に、必要なディスク容量を局毎に表示する。

実行方法

skdchk sked_file [-NOEARLY]

ここで sked_file – スケジュールファイル名
-NOEARLY – “tape early start”パラメータを無視する

実行例

```
skdchk Ver 2.11    2007-03-16
Schedule file is  ../sked/r1267.skd
../sked/r1267.skd

*** SCHEDULE FILE (../sked/r1267.skd) INFORMATION ***
Experiment code    : R1267
Number of stations : 7
    1  F   FORTLEZA   4985370.048910  -3955020.325840   -428472.303780
    2  E   WESTFORD   1492206.599640  -4458130.507370   4296015.532100
    3  O   TIGOCONC   1492054.156810  -4887961.003330  -3803541.356780
    4  K      KOKEE  -5543837.617470  -2054567.847980   2387851.939000
    5  H   HOBART26  -3950236.742500   2522347.561100  -4311562.549690
    6  T   TSUKUB32  -3957408.751200   3310229.346600   3737494.836000
    7  V   WETTZELL   4075539.899410    931735.270250   4801629.351850
Number of stars : 60
   (only 20 stars are listed here)
    1   0234+285          $  39.468357  28.802497 2000.000000
    2   1226+373          $ 187.197599  37.103360 2000.000000
    3   1040+123      3C245 160.685855  12.058684 2000.000000
    4   0754+100          $ 119.277679   9.943014 2000.000000
    5   2007+777          $ 301.379160  77.878680 2000.000000
    6   1743+173          $ 266.396701  17.333729 2000.000000
    7   1055+018          $ 164.623355   1.566340 2000.000000
    8   1236+077          $ 189.852451   7.504775 2000.000000
    9   0502+049          $  76.346603   4.995201 2000.000000
   10   1451-375          $ 223.614207 -37.792540 2000.000000
   11   0059+581          $  15.690677  58.403094 2000.000000
   12   0104-408          $  16.687950 -40.572211 2000.000000
   13   0106+013          $  17.161546   1.583421 2000.000000
   14   0201+113          $  30.944404  11.579280 2000.000000
   15   0208-512          $  32.692502 -51.017192 2000.000000
   16   0336-019      CTA26  54.878907  -1.776612 2000.000000
   17   0402-362          $  60.973958 -36.083865 2000.000000
   18   0434-188          $  69.256178 -18.746837 2000.000000
   19   0454+844          $  77.176514  84.534596 2000.000000
   20   0537-441          $  84.709840 -44.085816 2000.000000
Number of scans : 1056
   First 5 scans are as follows:
       1      3C446  7/076 17:00:00  138
       2      0J287  7/076 17:00:00   40
       3   1611+343  7/076 17:02:52   40
       4   0234+285  7/076 17:05:22  200
       5   1451-375  7/076 17:06:15   71
   Last 5 scans are as follows:
    1052   1622-253  7/077 16:52:26   40
    1053   1739+522  7/077 16:54:53   40
```

```
1054      CTA26  7/077 16:55:07  200
1055   1044+719  7/077 16:56:26   92
1056   1124-186  7/077 16:59:10   40

 Early Start Parameter (sec) : 10  but omitted by operator!

Maximum Disk requirements
  Total observation time (sec)  = 109874
   Disk requirements
       32Mbps  :    439.496 GBytes
       64Mbps  :    878.992 GBytes
      128Mbps  :   1757.984 GBytes
      256Mbps  :   3515.968 GBytes

                  Disk Requirements by Station (GBytes)
================================================================================
          FORTLEZA  WESTFORD  TIGOCONC     KOKEE  HOBART26  TSUKUB32  WETTZELL
--------------------------------------------------------------------------------
    sec      40749     45734     43656     38295     26183     33668     36458
 #scans        294       439       303       418       314       519       411
--------------------------------------------------------------------------------
 32Mbps      163.0     182.9     174.6     153.2     104.7     134.7     145.8
 64Mbps      326.0     365.9     349.2     306.4     209.5     269.3     291.7
128Mbps      652.0     731.7     698.5     612.7     418.9     538.7     583.3
256Mbps     1304.0    1463.5    1397.0    1225.4     837.9    1077.4    1166.7
================================================================================
```

3.3.3 setdcoffset

ユーティリティ名

setdcoffset

機能

K5/VSSP32 サンプラーのＤＣオフセットをセットする。任意の値にセットする機能、初期値（０）に戻す機能、ユニット試験に適した値にセットする機能、観測に適した値にセットする機能を有する。ログ出力も行う。K5/VSSP に対してもどのモードも STAT（信号の統計出力）モードと見なされる。

実行方法

setdcoffset dc1 dc2 dc3 dc4 [logfile]

または

setdcoffset TEST|AUTO|OBS|STAT|RESET [logfile]

ここで
- dc1 – CH1 にセットする DC オフセット (-128 から 128)
- dc2 – CH2 にセットする DC オフセット (-128 から 128)
- dc3 – CH3 にセットする DC オフセット (-128 から 128)
- dc4 – CH4 にセットする DC オフセット (-128 から 128)
- TEST – サンプラーボードの信頼性試験を行う際に適したオフセットを自動的にセットする。
- AUTO – 観測に適したオフセットを自動的にセットする。
- OBS – AUTO と同じ
- STAT – 信号の統計結果（平均、標準偏差）のみを出力する
- RESET – DC オフセットを０にセットする
- logfile – ログファイル名
  デフォルト YYYY ＋ホスト名＋.log
  ログはアペンドされる

環境変数

K5LOGDIR – ログディレクトリの指定
プログラムデフォルトはカレントディレクトリ

ログ出力

例１．チャンネル毎に DC オフセット値を指定して走らせたとき

```
07047011017:p setdcoffset: set 5  10  -5  -10
07047011017:p setdcoffset: VSSP32 OFFSET CH1 = 5  CH2 = 10  CH3 = -5  CH4 = -10
07047011017:p setdcoffset:                        Signal Statistics
07047011017:p setdcoffset:               CH1        CH2        CH3        CH4
07047011017:p setdcoffset: average     5.8/256   10.7/256   -4.8/256   -9.4/256
07047011017:p setdcoffset: std dev     0.5/256    0.4/256    0.5/256    0.3/256
```

最後の２行は信号の平均値 (average) と標準偏差 (std dev) を表示している。信号のレベルは-127.5〜+127.5 でさる。この例は、入力信号を入れていないときであり、標準偏差が小さくなっている。

例２．TEST モードで走らせたとき

```
07047011429:p setdcoffset: set TEST mode
07047011429:p setdcoffset: VSSP32 OFFSET CH1 = -6  CH2 = -6  CH3 = -6  CH4 = -6
07047011429:p setdcoffset:                        Signal Statistics
07047011429:p setdcoffset:               CH1        CH2        CH3        CH4
07047011429:p setdcoffset: average    -5.2/256   -5.3/256   -5.9/256   -5.4/256
07047011429:p setdcoffset: std dev     0.5/256    0.4/256    0.5/256    0.3/256
```

自動的に DC オフセットが-6 にセットされているのがわかる。これは、入力がない場合に、１ビットサンプリング後の信号をすべて０として、ヘッダー部の誤動作を起こりにくくするためである。まお、通常の信号が入っている場合は、単に平均値が０となるように、DC オフセットが自動的に調整される。

例３．AUTO モードで走らせたとき

```
07047011806:p setdcoffset: set AUTO mode
07047011806:p setdcoffset: VSSP32 OFFSET CH1 = 0  CH2 = 0  CH3 = 0  CH4 = 0
07047011806:p setdcoffset:                        Signal Statistics
07047011806:p setdcoffset:               CH1        CH2        CH3        CH4
07047011806:p setdcoffset: average     0.8/256    0.7/256    0.0/256    0.6/256
07047011806:p setdcoffset: std dev     0.5/256    0.4/256    0.5/256    0.3/256
```

平均値が０になるように自動的に DC オフセットが調整される。

3.4 観測ソフトの使い方

3.4.1 autoobs

ユーティリティ名

autoobs

機能

自動データ収集。環境変数 K5SATKEY で指定した電波源に対しては強制的に 1ch モードの観測が可能

実行方法

その１

autoobs -runparam_file [-NOSATKEY] [-NOEARLY] [-TEST] [-shift time]

ここで runparam_file – パラメータファイル（注１）名
先頭に '-' を付けること！
-NOSATKEY – サテライトキーの設定の無効化
-NOEARLY – "tape early start"パラメータを無効とする
-TEST – スケジュールをテストモードで実行する
（スケジュールの最初のスキャンを１分後にスタートする）
-shift time – スケジュールの最初のスキャンの開始時刻を time に変更する
ここで time は以下のいずれかの形式
YYYY/DDD-HH:MM:SS
YYYY/MM/DD-HH:MM:SS
YYYYDDDDHHMMSS

または

autoobs sked_file station_id [span [sfreq [adbit [numch [outdir [y_or_n [naming_type [freqg [subnet [f_size_mode]]]]]]]]] [-NOSATKEY] [-NOEARLY] [-TEST] [-shift time]

ここで sked_file – スケジュールファイル名
staion_id – 局 ID (スケジュールファイルで記述されている局ＩＤ)
span – 観測時間 (sec)
（スケジュールファイルで規定している観測時間より短くする目的）
デフォルトは０（スケジュールファイル通りの観測時間の意）
sfreq – サンプリング周波数
40,100,200,500 (for kHz)
1,2,4,8,16 (for MHz)
デフォルトは１６
adbit – A/D 分解能（ビット）
1,2,4,8
デフォルトは１
numch – 入力ｃｈ数
1　または　4
デフォルトは１
outdir – データファイル出力ディレクトリ
デフォルトはカレントディレクトリ
先頭に '-' を付けると、出力情報ファイル（注２）を指す

y_or_n – 観測範囲を制限する場合 Y, すべて観測の場合 N
省略時は対話式で回答する

naming_type – 出力ファイル命名則
1 または-1: タイプ１ 従来タイプ（デフォルト）
XDDDNNNN.[#ch.]dat
ここで X – 局ＩＤ (１文字)
DDD – 一番目のスキャンの通日
NNNN – 観測番号 (４桁)
#ch – チャンネル数 (1|4) (タイプが負の場合のみ)
2 または-2: タイプ２ K5/VSSP 固有タイプ
sidDDDHHMMSSG.[#ch.]dat
ここで sid – 局 ID (１文字か２文字)
DDD – スキャン開始通日（３桁）
HH – スキャン開始時（２桁）
MM – スキャン開始分（２桁）
SS – スキャン開始秒（２桁）
G – 周波数グループ ID (a|b|c|d) または null
#ch – チャンネル数 (1|4)(タイプが負の場合のみ)
3: Type III e-VLBI ファイル命名則準拠
expid_sidG_scanid_YYYYDDDDHHMMSS.k5
ここで expid – 実験コード
sid – 局ＩＤ（小文字２文字）大文字の場合は小文字に変換
G – ターミナル (PC)ID (1|2|3|4) または null。
freqg パラメータで指定
scanid – スキャンＩＤ。VEX ファイル使用時はスキャンＩＤ
そのまま。SKED 使用時は ddd-hhmm。
同じ分の中に複数の観測がある場合は２つ目以降に
時間順に最後に a,b,c,d,... をつけていく
YYYY – 観測開始時刻（年）
DDD – 観測開始時刻（通日）
HH – 観測開始時刻（時）
MM – 観測開始時刻（分）
SS – 観測開始時刻（秒）
.k5 - K5 データ識別子

freqg – 新タイプファイル命名則で使用する周波数グループＩＤ
または PC を示す ID。タイプ３では数字のみ。
1|2|3|4 または a|b|c|d （デフォルトは null）

subnet – サブネット観測モードのＯＮ／ＯＦＦのセット
0: OFF 1: ON（デフォルト）

f_size_mode – ファイルサイズ制限モード
0: 制限無し 1: 2GB 毎のファイルを作成（デフォルト）

-NOSATKEY – サテライトキーの設定の無効化

-NOEARLY – ”tape early start”パラメータを無効とする

-TEST – スケジュールをテストモードで実行する
（スケジュールの最初のスキャンを１分後にスタートする）

-shift time – スケジュールの最初のスキャンの開始時刻を time に変更する

ここで time は以下のいずれかの形式
YYYY/DDD-HH:MM:SS
YYYY/MM/DD-HH:MM:SS
YYYYDDDDHHMMSS

その２

autoobs sked_file [option]

または

autoobs [option]

ここで option（順不同可）は以下のとおり

-run runparam_file – ランパラメータファイル名の指定
この指定を行ったときは以下のパラメータは
“-NOSATKEY”以外は無効となる。
-sked sked_file – スケジュールファイル (SKED 形式または VEX 形式）の指定
-sid staion_id – 局 ID のセット
-span span – 観測時間 (sec) のセット
（スケジュールファイルで規定している観測時間より短くする目的）
デフォルトは０（スケジュールファイル通りの観測時間の意）
-sfreq sfreq – サンプリング周波数のセット (sfreq は 40|100|200|500|1|2|4|8|16)
40,100,200,500 (kHz)
1,2,4,8,16 (MHz)
デフォルトは 16
-ad 1|2|4|8 – ＡＤ分解能（ビット）のセット
1,2,4,8
デフォルトは 1
-nch 1|4 – チャネル数のセット
1 または 4
デフォルトは 1
-odir outdir – データファイル出力ディレクトリのセット
-dinfo dirinfo_file – 出力ディレクトリ切り替え情報ファイル
-nolimit – 観測範囲を制限しない（すべて観測を行う）
省略時は対話式で回答する
-naming 1|-1|2|-2|3 – 出力ファイル命名則のセット
1 または-1: タイプ１ 従来タイプ（デフォルト）
XDDDNNNN.[#ch.]dat
ここで X – 局ＩＤ (１文字)
DDD – 一番目のスキャンの通日
NNNN – 観測番号 (４桁)
#ch – チャンネル数 (1|4) (タイプが負の場合のみ）
2 または-2: タイプ２ K5/VSSP 固有タイプ
sidDDDHHMMSSG.[#ch.]dat

ここで sid – 局 ID (１文字か２文字)
DDD – スキャン開始通日（３桁）
HH – スキャン開始時（２桁）
MM – スキャン開始分（２桁）
SS – スキャン開始秒（２桁）
G – 周波数グループ ID (a|b|c|d) または null
#ch – チャンネル数 (1|4)(タイプが負の場合のみ)

3: Type III e-VLBI ファイル命名則準拠
expid_sidG_scanid_YYYYDDDDHHMMSS.k5
ここで expid – 実験コード
sid – 局ＩＤ（小文字２文字）大文字の場合は小文字に変換
G – ターミナル (PC)ID (1|2|3|4) または null。
freqg パラメータで指定
scanid – スキャンＩＤ。VEX ファイル使用時はスキャンＩＤ
そのまま。SKED 使用時は ddd-hhmm。
同じ分の中に複数の観測がある場合は２つ目以降に
時間順に最後に a,b,c,d,... をつけていく
YYYY – 観測開始時刻（年）
DDD – 観測開始時刻（通日）
HH – 観測開始時刻（時）
MM – 観測開始時刻（分）
SS – 観測開始時刻（秒）
.k5 - K5 データ識別子

4: 未使用

5: タイプ５
expid_scan#.stcode.k5a(-d)
ここで expid – 実験コード
scan# – スキャン番号
stcode – 局コード
k5 – 固定値
a-d – 周波数グループ

-freqg freqg – ファイル命名則タイプ２、３または５で使用する周波数グループＩＤ
または PC を示す ID。タイプ３では数字のみ。
1|2|3|4 または a|b|c|d （デフォルトは null）

-subnet 0|1 – サブネットモードの設定
0: サブネット OFF 1:サブネット ON(デフォルト)

-fsize 0|1 – 出力ファイルサイズ分割モードの設定
0: 分割無し 1: 2GB 毎に分割 (デフォルト)

-NOSATKEY – サテライトキー設定の無効化

-NOEARLY – “tape early start”パラメータを無効とする

-TEST – スケジュールをテストモードで実行
（スケジュールの最初のスキャンを１分後にスタートする）

-shift time – スケジュールの最初のスキャンの開始時刻を time に変更する
ここで time は以下のいずれかの形式
YYYY/DDD-HH:MM:SS
YYYY/MM/DD-HH:MM:SS

YYYYDDDDHHMMSS

環境変数のモニター autoobs env

注１：パラメータファイルの中身例（*以降はコメント）

```
**
** Sample K5 run control file Ver 4.1 (2009-11-22)
**

$SKED
sample.skd        * schedule file  (VEX file is allowed)

$STATION_ID
R                  * station ID.     1 chars for SKED and 2 chars for VEX

$LOGDIR
/usr/home/vlbi/ipvlbi/log     * log directory

$OUTDIR           * up to 10
/k51d/ad5         * 1st out directory candidate
/k51d/ad6         * 2nd out directory candidate

$SAMPLE
span=0    * obs span (sec), 0 means as schduled
sfreq=1   * sampling frequency
          *   40,100,200,500 (for kHz)
          *   0.04,0.1,0.2,0.5,1,2,4,8,16,32,64 (for MHz)
          *  Note1: 32 and 64 are valid only for K5/VSSP32
          *  Note2: -VE means "no use of digital filter" at K5/VSSP32
          *  In case of +VE, relation between sampling frequency (Fs)
          *  and filter frequency(Ff) at K5/VSSP32 is as follows
          *    Fs(MHz)   0.04 0.1 0.2 0.5 1 2 4 8 16 32 64
          *    Ff(MHz)     2   2   2   2  2 2 2 4  8 16  0
          *                                (0 means through)
adbit=1   * A/D bits  1,2,4,8
numch=4   * # of channels   1,4

$NAMING_TYPE * 1 or -1, 2 or -2, or 3
*   out file naming type selection
*1 ** Type I
*
*       XDDDNNNN.[#ch.]dat
*            where X    -- satation id (1 char)
*                  DDD  -- total day at 1st scan (3 digits)
*                  NNNN -- obs number  (4digits)
*                  #ch  -- # of channel included (for Type '-1')
*
*-1 ** Type -I
*
*       XDDDNNNN.#ch.dat
*            where #ch  -- number of channels in data
*
2  ** Type II
*
*       sidDDDHHMMSSG.[#ch.]dat
*            where sid   -- station id  (1 char or 2 char)
*                  DDD   -- total day at current scan (3digits)
*                  HH    -- hour at the start of scan (2digits)
*                  MM    -- minute at the start of scan (2digits)
*                  SS    -- second at the start of scan (2 digits)
*                  G     -- frequency group id (a|b|c|d) or null
*                  #ch  -- # of channel included (for Type '-2')
*
*-2  ** Type -II
*
*       sidDDDHHMMSSG.#ch.dat
*            where #ch  -- number of channels in data
*
*3  ** Type III
*       expid_sidG_scanid_YYYYDDDDHHMMSS.k5
```

```
*            where expid  -- experiment code
*                  sid    -- station ID (2 lower-case characters)
*                  G      -- PC id  1|2|3|4  or null
*                  scanid -- scan id
*                  YYYY   -- year (4digits)
*                  DDD    -- total day at current scan (3digits)
*                  HH     -- hour at the start of scan (2digits)
*                  MM     -- minute at the start of scan (2digits)
*                  SS     -- second at the start of scan (2 digits)

$FREQ_G
*   set frequency group (terminal (PC) ID) used in type II or III
*   naming rule
*   if omitted  null character is used, i.e., file name
*   will be sidDDDHHMMSS.dat
*      1,2,3,4  or a,b,c,d is possible
1   **  means 'a'
*a   **  also OK for 'a'

$SUBNET
*   subnet mode selection  on | off  (default on)
on

$FILE_SIZE_LIMIT
*   file size limitation   on | off  (default on)
*   if set to "on", big file is divided into 2GB each.
*   if set to "off", no limitation on 1 file size.
on

$SATKEYS
*   define satellite code (usually 3 chars, max 8 chars)
*   for 1ch mode observation
*   maximum 10 key words separated by space are allowed
*
* NONE   ***  to turn off the satellite mode explicitly
*              (function is the same as the option '-SATKEY')
*
* HYB     ***  HAYABUSA
* NOZ     ***  NOZOMI
* GEO     ***  GEOTAIL
* HYB NOZ  ***  both Hayabusa and Nozomi
HYB NOZ GEO  ***  Hayabusa, Nozomi, Geotail will be observed by 1ch mode
```

上の例の場合、スケジュールファイル sample.skd を使用して観測が行われる。観測局ＩＤはＲで、ログ出力ファイルは/usr/home/vlbi/ipvlbi/log ディレクトリに作成される。観測毎のデータ出力ファイルは、まず/k51d/ad5 ディレクトリに作成され、そのハードディスクが満杯（残り容量が次の観測データ容量の１．５倍より少ない場合に満杯と判断）の場合、データ出力ディレクトリが/k51d/ad6 に切り替わる。（注：各ディレクトリは異なるＨＤＤ上になければならない！）。切り替えは１０ディレクトリ（ＨＤＤ）まで可能。$SAMPLE はサンプリング条件を記述している。$NAMING_TYPE で命名則（上の例では TypeII）を規定している。$FREQ_G でこのＰＣが受け持つ周波数グループ（上の例ではａ）（※）を規定する。またサブネット観測モードはＯＮにセット。作成されるデータファイルは２ＧＢで分割される。$SATKEYS で “HYB”,“NOZ”, “GEO”で始まる電波源の場合は強制的に１ ch モードの観測を行うように指定。なお、このキーワードは最大８文字まで可能であり、準星の指定等にも使用できる。

※周波数グループとは測地観測で１６ｃｈの観測をする場合、ＰＣ１台あたり受け持つ４ｃｈの周波数グループを示し、

a:ch1-4 b:ch5-8 c:ch9-12 d:ch13-16

の a,b,c,d で定義する。この定義は命名則 TypeIII でのＰＣを示すＩＤにも用いられる。この場合は a,b,c,d とセットした場合、内部で 1,2,3,4 に自動的に変換される。

注２：出力情報ファイルの中身例（*で始まる行はコメント行）

```
**
```

```
** K5 out directory order definition file
**  each directory must be in different HDD
**  1st directory is outdir parameter in the autoobs run parameters
**  then switched to directories in this file
/k51d/ad5
/k51d/ad6
```

出力情報ファイルはパラメータファイルとは独立に出力ディレクトリ情報を記述したファイルであり、autoobs のランパラメータの後方互換性を保つために導入したファイル。
この例の場合は、autoobs のランパラメータ outdir がディレクトリを指定した場合はまず outdir ディレクトリにデータは出力され、その後 /k51d/ad5 -> /k51d/ad6 とディスクが満杯になるたびにデータ出力先が切り替わる-outdir で上記ファイルを指定した場合はまず/k51d/ad5 ディレクトリにデータは出力され、その後ディスクが満杯になると /k51d/ad6 に出力先が切り替わる

データ出力

ディレクトリ

(1) パラメータファイル指定時
$OUTDIR で指定された（複数）ディレクトリ。最大１０個指定可能

(2) パラメータファイル無指定時

a. outdir でディレクトリを指定したとき
outdir が一杯になるまで出力。一杯になった場合、環境変数 K5OUTINFO で指定する出力情報ファイルの情報に従って順番に出力先が切り替わる。環境変数 K5OUTINFO をセットしていない場合は、デフォルト出力情報ファイルとして k5outinfo.txt が用いられる。このファイルが存在すれば、その中の出力ディレクトリ情報に従って出力先が切り替わる。

b. outdir パラメータの先頭に –(マイナス）を付けた場合
–(マイナス) を除いた部分が出力情報ファイルとなる。

データファイル名

ファイル命名則に Type1（または-1)、Type2（または-2)、Type3 および Type5 の４種類がある。プログラムデフォルトは Type1（従来方式）であり、環境変数 K5NAMING が new または２とセットされていれば Type2、3 とセットされていれば Type3 方式となる。コマンド引数またはパラメータファイルでも指定可能であり、コマンド引数またはパラメータファイルで指定した場合が、最も強い指定となる。

タイプ 1 または-1: 従来タイプ（デフォルト）
XDDDNNNN.[#ch.]dat
ここで X – 局ＩＤ (１文字)
DDD – 一番目のスキャンの通日
NNNN – 観測番号 (４桁)
#ch – チャンネル数 (1|4) (タイプが負の場合のみ）

タイプ 2 または-2: K5/VSSP 固有タイプ
sidDDDHHMMSSG.[#ch.]dat

ここで sid – 局 ID (１文字か２文字)
DDD – スキャン開始通日（３桁）
HH – スキャン開始時（２桁）
MM – スキャン開始分（２桁）
SS – スキャン開始秒（２桁）
G – 周波数グループ ID (a|b|c|d) または null
#ch – チャンネル数 (1|4)(タイプが負の場合のみ)

タイプ 3: e-VLBI ファイル命名則準拠
expid_sidG_scanid_YYYYDDDDHHMMSS.k5
ここで expid – 実験コード
sid – 局ＩＤ（小文字２文字）大文字の場合は小文字に変換
G – ターミナル (PC)ID (1|2|3|4) または null。
freqg パラメータで指定
scanid – スキャンＩＤ。VEX ファイル使用時はスキャンＩＤ
そのまま。SKED 使用時は ddd-hhmm。
同じ分の中に複数の観測がある場合は２つ目以降に
時間順に最後に a,b,c,d,... をつけていく
YYYY – 観測開始時刻（年）
DDD – 観測開始時刻（通日）
HH – 観測開始時刻（時）
MM – 観測開始時刻（分）
SS – 観測開始時刻（秒）
.k5 - K5 データ識別子

タイプ 4: 未使用

タイプ 5:
expid_scan#.stcode.k5a(-d)
ここで expid – 実験コード
scan# – スキャン番号
stcode – 局コード
k5 – 固定値
a-d – 周波数グループ

Type2 または 3 方式を指定した場合の周波数グループまたはターミナル (PC) ＩＤのセットは環境変数 K5FREQG による方法とコマンド引数またはパラメータファイル指定による方法がある。一番優先するのはコマンド引数またはパラメータファイルによる指定である。
指定は
1|2|3|4 または a|b|c|d （デフォルトは null）

環境変数

K5SKED – スケジュールファイルのデフォルトディレクトリ
プログラムデフォルトはカレントディレクトリ

K5LOGDIR – ログディレクトリの指定
プログラムデフォルトはカレントディレクトリ
K5OUTINFO – デフォルト出力ディレクトリ切り替え情報ファイル名
プログラムデフォルト k5outinfo.txt
パラメータファイルを使用するときはパラメータファイル中の情報が優先する
K5NAMING – データファイル命名則 old | new または 1|2|3
注：環境変数でのセットは Type1 は old または１, Type2 は new または２とする
プログラムデフォルトは Type1
K5FREQG – データファイル命名則が Type2 または 3 の場合のＰＣの受け持つ周波数グループ
(ターミナル (PC) ＩＤを兼ねる)
1|2|3|4 または a|b|c|d プログラムデフォルトは null
K5SUBNET – サブネット観測モードのセット on|off
プログラムのデフォルトは on
パラメータファイルやコマンド引数を使用するときはそれらの情報が優先する。
K5SATKEY – 1ch モードで観測する電波源の指定（最大８文字）を行う
プログラムデフォルトは HYB （はやぶさ）

ログ出力

ディレクトリ – パラメータファイル使用時
$LOGDIR で指定されたディレクトリ
パラメータファイル不使用時
環境変数 K5LOGDIR でセットしたディレクトリ。
K5LOGDIR がセットされていないときはカレントディレクトリ
ファイル名 – 実験コード＋ “_”＋局ＩＤ＋ “.”+ホスト名+“.log”
例：NZ123_R.k51d.log
出力形式 – YYDDDHHMMSSXY コメント
ここで YY : 年（２桁）
DDD : 通日
HH : 時
MM : 分
SS : 秒
X : セパレータ :|/|”|?|#|(space)
Y : 時刻モード
(space) – サンプラーの時刻
’p’ – PC の時刻

出力例

```
03168075510:  OBS# 0002 :  2003/06/17 07:55:30  ( 10sec)  1334-127 [10.0MB]
03168075540/  OBS# 0002 : /k51d/ad5/R1680002.dat 10 2 1 4
```

実行例

例１ パラメータファイル k5runinfo.txt を使用して走らせる
autoobs -k5runinfo.txt

または
autoobs -run k5runinfo.txt

例２ パラメータファイル k5runinfo.txt を使用して走らせるが、サテライトキーを無効化する
autoobs -k5runinfo.txt -NOSATKEY
または
autoobs -run k5runinfo.txt -NOSATKEY

例３ スケジュールファイル　ks01240.skd を使って鹿島局 (R) の観測を４ＭＨｚ２ビット４ｃｈサンプリングで行う。出力は/k51d/ad7
autoobs ks01240.skd R 0 4 2 4 /k51d/ad7 N
または
autoobs ks01240.skd -sid R -sfreq 4 -ad 2 -nch 4 -odir /k51d/ad7 -nolimit
または
autoobs -sked ks01240.skd -sid R -sfreq 4 -ad 2 -nch 4 -odir /k51d/ad7 -nolimit

例４ スケジュールファイル　ks01240.skd を使って鹿島局 (R) の観測を４ＭＨｚ２ビット４ｃｈサンプリングで行う。出力ディレクトリは出力情報ファイル (outdirinfo.txt) を使って切り替える。
autoobs ks01240.skd R 0 4 2 4 -outdirinfo.txt N
または
autoobs ks01240.skd -sid R -sfreq 4 -ad 2 -nch 4 -dinfo outdirinfo.txt -nolimit

例５ スケジュールファイル　ks01240.skd を使って鹿島局 (R) の観測を４ＭＨｚ２ビット４ｃｈサンプリングで行う。出力ディレクトリは出力情報ファイル (outdirinfo.txt) を使って切り替える。命名則は新形式 (type2) を使う。ＰＣの受け持つ周波数グループは a
autoobs ks01240.skd R 0 4 2 4 -outdirinfo.txt N 2 a

例５と同じ設定は環境変数を使うと setenv K5NAMING new
setenv K5FREQG a
autoobs ks01240.skd R 0 4 2 4 -outdirinfo.txt N
のように走らせることもできる

例６ パラメータファイル k5runinfo.txt を使用して走らせるが、スケジュールをチェックするためテストモードで走らせる
autoobs -k5runinfo.txt -TEST

テストモードで走らせると、ログファイルの最初にログファイルの最初に
```
07079005601"  ********** AUTOOBS Ver 4.11 (2007-03-20) START **********
07079005601"   This is schedule TEST mode run!
```
のように　”TEST mode”であることが記述されます。
また画面表示の、　Schedule File の項目の最後に　”(TEST mode)” という表示が追加されます。

例７ パラメータファイル k5runinfo.txt を使用して走らせるが、スケジュールの開始時刻を 2007 年 3 月 20 日（通日 79 日) 12 時 30 分 40 秒に変更する
autoobs -k5runinfo.txt -shift 2007/03/20-12:30:40
または
autoobs -k5runinfo.txt -shift 2007/079-12:30:40

または
autoobs -k5runinfo.txt -shift 2007079123040

シフトモードで走らせると、ログファイルの最初にログファイルの最初に

```
07079005601"  ********** AUTOOBS Ver 4.11 (2007-03-20) START **********
07079005601"   This is schedule SHIFT mode run!
```

のように ”SHIFT mode”であることが記述されます。
また画面表示の、 Schedule File の項目の最後に ”(SHIFT mode)” という表示が追加されます。
テストモードとシフトモードを同時に指定するとテストモードが優先されます。

なお、パラメータファイル不使用モードで、y_or_n を省略して走らせると観測の範囲を限定するかどうか尋ねてくるので、限定することもできる。

「のぞみ」モード

「のぞみ」観測時（電波源名が NOZ で始まる時)、自動的に１ ch モード、それ以外の時は numch パラメータで指定したモード（通常４ ch モード）でデータ収集を行いディスク容量を節約する。1ch 観測を「のぞみ」以外の衛星に変更するには環境変数 K5SATKEY で指定する。K5SATKEY で指定するキーワードは最大８文字で通常３文字。電波源との一致の判定は先頭一致方式。（例 bsh で「はやぶさ:電波源名 HYBxxxx」観測を 1ch モードに指定する K5SATKEY=HYB; export K5SATKEY）
注意：1ch サンプラーボードには 4ch サンプラーボードの ch1 と同じビデオ信号を入力すること！

実行時の画面例

```
*********************************************************************
*             AUTOOBS Ver 4.02 (2007-02-22) by NICT                 *
*    K5/VSSP32 unit : Time Get from SAMPLER  : Naming Type 2         *
*********************************************************************

  Schedule File     :  ./r1267.test.skd (type : SKED)
  Experiment Code   :  r1267          Freq_G (a)
  Log File          :  ./r1267_T.k56a.log
  Stations Included :  FORTLEZA WESTFORD TIGOCONC HOBART26 TSUKUB32 KOKEE
  My Station Name   :  TSUKUB32 (My Station ID : T (Ts))   Subnet Mode : ON
  Current Out Dir   :  /k56a/ad4/    271629.2MB left
  Next    Out Dir   :  /k51d/ad5/    111861.3MB left
  Total Scan Number :  1056       Satellite Mode Keys: HYB NOZ GEO
  1st  Scan         :  2007/03/17 17:00:00  ( 138sec)  3C446
  Last Scan         :  2007/03/18 16:59:10  (  40sec)  1124-186
  Obs Range (start) :  2007/03/17 17:00:00    each scan EARLY start : 10 sec
  Obs Range (end)   :  2007/03/18 16:59:50
  Sampling Mode for Next Scan :  16MHz 1bit 4ch (LPF 8MHz)

  Next Scan(No.0002):  2007/03/17 17:00:00  (40sec)  OJ287 [320.0MB]

  Time Now (UTC)    :  2007/03/17 02:54:48
```

各項目の説明

- Schedule File – スケジュールファイル名
- (type :) – スケジュールファイルのタイプ　SKED — VEX
  - “-shift” オプションで走らせると項目の最後に　”(SHIFT mode)” という表示が追加される
  - “-TEST” オプションで走らせると項目の最後に　”(TEST mode)” という表示が追加される
- Experiment Code – 実験コード
- Log File – ログファイル
- Stations Included – スケジュール中に含まれている局表示
- My Station Name – 自局名 と局ＩＤ. 局 ID は SKED タイプの場合は 1 文字 ID と ( ) 内に 2 文字 ID. VEX タイプのスケジュールの場合は 2 文字 ID
- Current Out Dir – 現在の出力先ディレクトリと残容量
- Next Out Dir – 現出力先が一杯になった場合、次に選択されるディレクトリ
- Total Scan Number – 全スキャン数
- Satellite Mode Keys – 「のぞみ」モードで使用するキー
- 1st Scan – 最初のスキャン情報
- Last Scan – 最後のスキャン情報
- Obs Range (start) – 観測範囲（範囲の最初のスキャン開始時刻）
- Obs Range (end) – 観測範囲（範囲の最後のスキャン終了時刻）
- Sampling Condition for Next Scan – 次スキャンのサンプリング情報
- Next Scan(No.XXXX) – 次の観測情報　( ) 内はスキャン時間、[ ] 内はデータ量
- Time Now (UTC) – 現在の時刻（ＵＴＣ）

なお、”-NOSATKEY”で起動したときは、環境変数やパラメータファイルやプログラムデフォルトのサテライトキーをすべて無効にすることができる。その時観測帯域画面の「Satellite Mode Keys:」フィールドには”NONE”と表示される。

3.4.2 sampling

ユーティリティ名

sampling

機能

手動によるデータ収集

実行方法

sampling span sfreq[:lpf] adbit[:bitshift] numch [filename [logfile]]

ここで
- span – 観測時間 (sec)
- sfreq – サンプリング周波数
  40,100,200,500 (for kHz)
  0.04,0.1,0.2,0.5 (MHz)
  0.04,0.1,0.2,0.5,1,2,4,8,16,32,64 (for MHz)
  注１：32,64 は K5/VSSP32 のみ
  注２：負の値でセットすると、K5/VSSP32 においてフィルターをスルーにセット
  正の値の場合、フィルターは以下のようにセットされる

Fsample(MHz)	0.04	0.1	0.2	0.5	1	2	4	8	16	32	64
Ffilter(MHz)	2	2	2	2	2	2	2	4	8	16	0

  (0 はスルーを意味する）
- lpf – LPF フィルター値 (MHz) 2,4,8,16,0 (0 はスルーを意味する）
  サンプリング周波数と独立に LPF をセットするときに使う
- adbit – ＡＤ分解能（ビット）
  1,2,4,8
- bitshift – ２ビット、４ビットサンプリングモード時のレベル最適化のためのビットシフト量を設定する。0-6 の値がセットできる。デフォルトは 0
- numch – チャネル数
  1, 4
- filename – データ出力ファイル名
  デフォルトは tds.data
- logfile – ログファイル名
  デフォルトは sampling.log
  デフォルトのログファイルは上書き
  ユーザーが logfile パラメータで指定したログファイルにはログはアペンドされる

実行例

例１ サンプリング周波数４ＭＨｚ、２ビットＡＤ、４ｃｈ使用して１０秒間データ収集
sampling 10 4 2 4

例２ サンプリング周波数８ＭＨｚ、４ビットＡＤ、４ｃｈ使用して２０秒間データ収集し、カレントディレクトリに　tds2.data というファイルを作成し、出力する
sampling 20 8 4 4 tds2.data

上記　例１，例２ではカレントディレクトリにログファイル sampling.log が作成されるが、その都度ファ

イルは上書きされる。上書きされないようにするには logfile パラメータでログファイル名を指定してやる。

例３ サンプリング周波数４MHｚ、２ビットＡＤ、４ｃｈ使用して０秒間データ収集。出力ファイルを tds.dat、ログファイルを rei3.log とする。
sampling 10 4 2 4 tds.dat rei3.log
rei3.log というログファイルが作成されますが、２回目以降同じログファイルを指定して sampling を走らせると、ログはアペンドされていく。（この時、出力データファイル名は省略不可）

例４ K5/VSSP32 にてサンプリング周波数 8 MHｚ、２ビットＡＤ、４ｃｈ使用して１０秒間データ収集。LPF フィルターは自動セット（４MHｚ）を使用する。
sampling 10 8 2 4

例５ K5/VSSP32 にてサンプリング周波数 8 MHｚ、２ビットＡＤ、４ｃｈ使用して１０秒間データ収集。LPF フィルターはスルーにセットする。
sampling 10 -8 2 4

例６ K5/VSSP32 にてサンプリング周波数 16 MHｚ、1 ビットＡＤ、４ｃｈ使用して１０秒間データ収集。LPF フィルターは２MHｚにセットする。
sampling 10 16:2 1 4

例７ K5/VSSP32 にてサンプリング周波数 16 MHｚ、２ビットＡＤ、４ｃｈ使用して１０秒間データ収集。LPF フィルターは２MHｚにセットする。ＡＤのビットシフトは４にセット。
sampling 10 16:2 2:4 4

3.4.3 sampling2

ユーティリティ名

sampling2

機能

手動によるデータ収集 (高機能版：１秒ファイル生成モードではシンクパターンのチェックあり)

実行方法

sampling span sfreq[:lpf] adbit[:bitshift] numch [filename [tempdir [mode]]]

ここで
- span – 観測時間 (sec)
- sfreq – サンプリング周波数
  40,100,200,500 (for kHz)
  0.04,0.1,0.2,0.5 (MHz)
  0.04,0.1,0.2,0.5,1,2,4,8,16,32,64 (for MHz)
  注１：32,64 は K5/VSSP32 のみ
  注２：負の値でセットすると、K5/VSSP32 においてフィルターをスルーにセット
  正の値の場合、フィルターは以下のようにセットされる

Fsample(MHz)	0.04	0.1	0.2	0.5	1	2	4	8	16	32	64
Ffilter(MHz)	2	2	2	2	2	2	2	4	8	16	0

  (0 はスルーを意味する)
- lpf – LPF フィルター値 (MHz) 2,4,8,16,0 (0 はスルーを意味する)
  サンプリング周波数と独立に LPF をセットするときに使う
- adbit – ＡＤ分解能（ビット）
  1,2,4,8
- bitshift – ２ビット、４ビットサンプリングモード時のレベル最適化のためのビットシフト量を設定する。0-6 の値がセット

できる。デフォルトは 0

numch – チャネル数
1, 4

filename – データ出力ファイル名
デフォルトは tds.data

tempdir – １秒データファイルの出力ディレクトリ
デフォルトは カレントディレクトリ

mode – ファイル出力モード
1: 一括ファイルのみ （デフォルト）
2: １秒データファイルのみ
3: 一括ファイル＋１秒ごとファイル同時
4: １秒データファイル出力、観測後一括ファイル作成
6: １秒データファイルのみ (時間制限無し）
7: ６と同じ（ただしファイル出力無し）システムチェック用

実行例

例１ サンプリング周波数４MHｚ、２ビットＡＤ、４ｃｈ使用して１０秒間データ収集を行い、１秒ファイルのみを作成。１秒ファイル出力ディレクトリは/data
sampling2 10 4 2 4 /data/tds.data /data 2

例２ サンプリング周波数１MHｚ、４ビットＡＤ、４ｃｈ使用して１秒データファイルを/data ディレクトリに作成し続ける。
sampling2 10 1 4 4 /data/tds.data /data 6

3.5 データチェック（編集）ソフトの使い方

3.5.1 datachk

ユーティリティ名

datachk

機能

ヘッダー部を頼りにヘッダー間のデータビット数をカウントすることにより、ビットスリップまたはビットメイクがあったかどうかでデータをチェックする。またアナログ信号が＋のサインであった割合を%で表示することも可能である。更にエラーが起こったデータファイルの記録機能も有する。この機能は、サンプラーの連続試験時にエラーが起こったデータファイルだけ保存したい場合に便利な機能である。

なお、このチェックで AUX MISALIGN (misaligned AUX field) と診断された K5/VSSP32 データは aux_recov を使って修復が可能です。

データの中身については　例えば　ＯＳ付属の　od 等で確認する

実行方法

datachk file_name [mode [logfile [errlog [keepmode]]]]

ここで　file_name　–　データファイル名　（デフォルトは tds.data）

mode　–　サンプリング統計（ゼロバランス）表示モード

- 0　：　サンプリング統計は表示しない（デフォルト）
  最初と最後のフレーム情報およびエラーの生じたフレーム情報のみを表示
- 1　：　サンプリング統計を表示する
- 2　：　モード０と同じ。ただし全フレーム表示する

logfile　–　モード０の場合にチェック結果のサマリーを出力するファイル名。
このファイル名がの先頭が “-” の場合サマリー出力は既存のファイルに追加されていく。
デフォルトはサマリー出力なし。

errlog　–　モード０の場合にエラーが発生したデータファイルの情報を出力するファイル名。
このファイル名がの先頭が “-” の場合、出力は既存のファイルに追加されていく。
チェックしたデータファイルにエラーが無い場合は、このファイルは作成（出力）されない。デフォルトはエラーログ出力なし。

keepmode　–　モード０の場合にエラーの生じたデータファイルの保存モード

- 0　：　何もしない（デフォルト）
- 1　：　データファイルの名前を元の名前＋ “.NNNN.err” に変更する。
- 2　：　データファイルを　　元の名前＋ “.NNNN.err” にコピーする。
  ここで NNNN は 0001 から 9999 で繰り返す
  この通し番号は datachk を実行するディレクトリ下の
  “counter_file_datachk.tmp” という名のテキストファイルで管理する。

ゼロバランス表示はアナログ信号が＋のサインであった割合を%で表示

サマリーファイル

サマリーファイルの例を以下に示す

```
# File Name:
D:\IPVLBI\data\test02.dat
# FMT  A/D CH  f(kHz)  LPF(MHz):
  VS32  1   1   32000       16
# Start and Last Time:
  2006/318 23:20:28  84028
  2006/318 23:25:27  84327
```

```
# Duration:
300
# Byte offset of 1st header:
0
# STATISTICS total bad discon discon_with_bitslip aux_sep EFLG:
300 1 0 0 147 0
# BIT SLIP:
1 26432
```

エラーログファイル

エラーログファイルの例を以下に示す

keepmode=0 の場合

```
# Errored Data File Name:
test02.dat
# FMT  A/D CH  f(kHz)  LPF(MHz):
  VS32  1   1   32000       16
```

keepmode=1 の場合（リネームモード）

```
# Errored Data File Name:
test02.dat
# FMT  A/D CH  f(kHz)  LPF(MHz):
  VS32  1   1   32000       16
# Renamed to:
test02.dat.0006.err
```

keepmode=2 の場合（コピーモード）

```
# Errored Data File Name:
test02.dat
# FMT  A/D CH  f(kHz)  LPF(MHz):
  VS32  1   1   32000       16
# Copied to:
test02.dat.0007.err
```

3.5.2 extdata

ユーティリティ名

extdata

機能

サンプリングデータファイルから特定のｃｈのデータを抽出しテキストファイルに書き出す

実行方法

extdata filename [ch [soffset [period [omode [outfile]]]]]

ここで
- filename – データファイル名
  0 とするとファイルは tds.data を使う
- ch – 抽出するｃｈ （１から４）
  デフォルト値は１
- soffset – スタート時刻オフセット（秒）
  デフォルト値は 0.0
- period – 抽出するデータの時間長（秒）
  デフォルト値は 1.0
- omode – データ出力モード
  0:ヘッダー情報付き
  1:ヘッダー無し
  デフォルト値は 0
- outfile – データ出力ファイル名
  デフォルトは extdata.txt

出力データ

出力テキストファイルの例（サンプリングデータは１行に２０点）（ヘッダー情報は ** で始まる最初の７行）

```
**  Data File : /ad4/R2320002.dat
**  Sampling Freq : 4MHz
**  A/D bits      : 4
**  Number of CHs : 4
**  Extracted CH# : 1
**  Start Time of Data : 12:15:38.00000000
**  Period (sec)       : 0.010000
 10 6 8 8 9 7 8 6 7 8 8 7 9 8 8 8 8 10 7 8
 7 8 7 9 8 9 11 8 8 8 5 7 7 7 7 7 6 7 8 9
 8 9 7 6 10 8 7 7 10 7 7 6 8 7 8 9 6 11 7 6
 6 7 6 6 7 9 8 7 7 7 8 8 8 9 9 8 7 8 8 8
 8 6 6 9 9 6 7 8 6 8 9 7 8 5 7 8 5 7 9 9
```

3.5.3 datacut

ユーティリティ名

datacut

機能

サンプリングデータファイルからある時間範囲のデータを抜き出す

実行方法

datacut filename [soffset [period [outfile]]]

ここで
- filename – データファイル名
  0 とするとファイルは tds.data を使う
- soffset – スタート時刻オフセット（整数秒）
  デフォルト値は 0
- period – 抽出するデータの時間長（整数秒）
  デフォルト値は 10
- outfile – データ出力ファイル名
  デフォルトは datacut.dat

3.5.4 datatime_edit

ユーティリティ名

datatime_edit

機能

サンプリングデータファイルのヘッダー部の時刻情報を書き換える

実行方法

datatime_edit filename [HH:MM:SS|offset [outfile]] [options]
または
datatime_edit [options]

ここで filename – データファイル名
0 とするとファイルは tds.data を使う
HH:MM:SS – データの開始時刻を新たに与える
例えば 01:23:45 で始まるデータの最初の
時刻を 03:10:20 にしたいときは 03:10:20 をセット
offset – データの開始時刻の修正量をオフセット（整数秒）で与える
例えば 01:23:45 を 01:23:44 としたいときは -1 をセット
outfile – データ出力ファイル名
デフォルトは datatime_edit.dat
[options]
-f filename – データファイル名を与える
-o outfile – 時刻情報を編集後の出力ファイル名を与える
-s HH:MM:SS|offset – データの開始時刻または修正する秒を与える
-n – 非会話型で走らす（デフォルトはオペレータに確認を求めてくる）

3.5.5 four2one

ユーティリティ名

four2one

機能

K5/VSSP（含む K5/VSSP32）ボードの４ｃｈモードで取得したデータファイルから特定のｃｈのデータを抽出した１ｃｈモードのデータファイルを作成する

実行方法

four2one filename [ch [outfile [soffset [span]]]]

ここで filename – データファイル名
ch – 抽出するｃｈ （１から４）
デフォルト値は１
outfile – データ出力ファイル名
デフォルトは filename+“. 抽出 ch”
soffset – スタート時刻オフセット（整数秒）
デフォルト値は 0
span – データ変換の区間（整数秒）
デフォルト値はすべて

実行例

データファイル R2320002.dat から 3ch データを抽出して R2320002.dat.3 に出力
four2one R2320002.dat 3

3.5.6 oscillo

ユーティリティ名

oscillo

機能

サンプリングデータの時系列ダイナミック表示。

sampling や autoobs で収集したデータの時系列表示を行う
XTERM もしくは WINDOWS の telnet で実行。ディスプレイに時系列データをダイナミックに表示する。
注意！PGPLOT がインストールされている必要があります（GNUPLOT はサポートしません）。

oscillo[リターン］で使用法が表示される。

実行方法

oscillo filename [options]

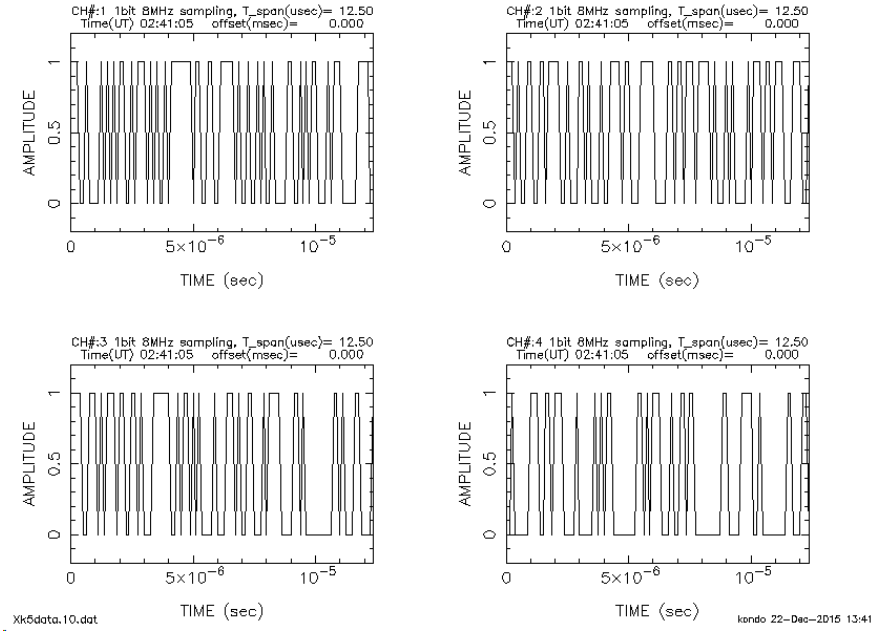
ここで	filename	–	データファイル名 0 とすると tds.data が使用される
	[options]		
	-t[span] tspan	–	表示時間幅の設定 (sec 単位) デフォルトは　100 サンプル点に相当する時間
	-h[alt]	–	シングル表示モードに設定（１表示ごとに停止）
	-s[msec] sleep_msec	–	連続表示モード時（デフォルト）の表示間隔を msec 単位で指定する デフォルトは 0.0

環境変数

PGDISP – PGPLOT デバイス (/XSERVE,/XTERM 等)

実行例

例１. 1 ビットサンプリング、4ch データの場合

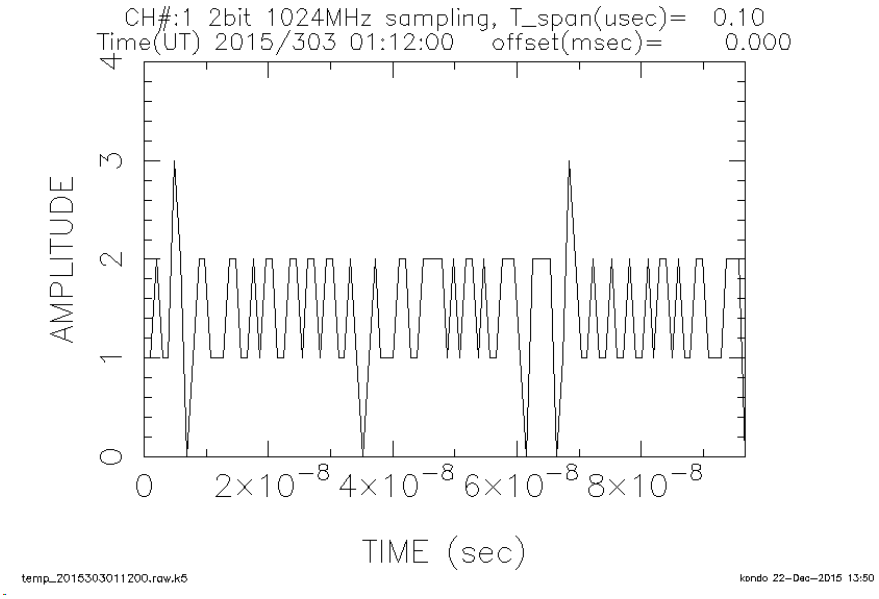


“-h” オプションを指定して走らせると表示後に

```
Hit return key for next span (or "N"on-stop or "Q"uit)
```

と聞いてくるのでリターンキーを押せば次のスパンデータが表示される。“N” を入力すると連続表示モードとなる。

例２. 2 ビットサンプリング、1ch データの場合



3.5.7 speana

ユーティリティ名

speana

機能

オフラインスペクトル表示。speana2 の機能も実装。

sampling や autoobs で収集したデータのスペクトル表示を行う
XTERM もしくは WINDOWS の telnet で実行。ディスプレイにスペクトルを表示するとともに、PostScript ファイル pgplot.ps (GNUPLOT 使用時は gnuplot.ps）を作成する。

注意！PGPLOT または GNUPLOT がインストールされている必要があります。
Tera Term から実行するときは「Setup」->「Terminal」で「Auto switch」にチェックが入っていることを確認すること

speana[リターン］で以下のようにどちらのグラフィック表示でコンパイルされているかの情報が表示される。

```
speana Ver. 2007-11-04 compliled for GNUPLOT
```

引き続き使用法が表示される。

実行方法

speana filename [mode [sekibun [pmode [comment [soffset [f1khz [f2khz [mindbm [maxdbm]]]]]]]]]

または

speana filename [options]

ここで filename – データファイル名
0 とすると tds.data が使用される
mode – 軸表示モードまたは自己相関モードの指定

0 : 強度ログ、周波数ログスケール
1 : 強度ログ、周波数軸リニアスケール （デフォルト）
10 : 強度リニア、周波数軸ログスケール
11 : 強度リニア、周波数軸リニアスケール
-1 : 自己相関関数プロット
-N : N点の自己相関関数プロット

sekibun – 積分時間 (秒単位)
デフォルトは 1.0

pmode – プロット表示デバイスモード
0 : 環境変数 PGDISP でセットされたデバイス (PGPLOT の場合）
(GNUPLOT の場合はディスプレー）および
PostScript ファイル (pgplot.ps または gnuplot.ps) 出力
（デフォルト）
1 : PostScript 出力 (pgplot.ps または gnuplot.ps) のみ
2 : 環境変数 PGDISP でセットされたデバイス出力のみ
(GNUPLOT の場合はディスプレーのみ）

comment – コメント（グラフ上部に表示)。スペースを含まないこと。
省略した場合は会話モード入力になる
注：スペースを含むコメントは会話モードで入力すること

soffset – データの処理開始時刻のオフセット（秒単位）
デフォルトは 0.0

f1khz – 表示周波数範囲の低周波側の設定（kHz 単位）
デフォルトは自動設定

f2khz – 表示周波数範囲の高周波側の設定（kHz 単位）
デフォルトは自動設定

mindbm – 表示パワー強度範囲の最小値の設定（dBm 単位）

maxdbm – 表示パワー強度範囲の最大値の設定（dBm 単位）

[options]

-m[ode] mode – 軸表示モードまたは自己相関モードの指定
-p[mode] pmode – プロット表示デバイスモード
-se|-ti|-i sekibun – 積分時間 (整数秒単位) の設定
-c[omment] comment – コメント（グラフ上部に表示）
-so|-to|-0 soffset – データの処理開始時刻のオフセット（正秒単位）
-f1 f1khz – 表示周波数範囲の低周波側の設定（kHz 単位）
-f2 f2khz – 表示周波数範囲の高周波側の設定（kHz 単位）
-min mindbm – 表示パワー強度範囲の最小値の設定（dBm 単位）
-max maxdbm – 表示パワー強度範囲の最大値の設定（dBm 単位）

環境変数

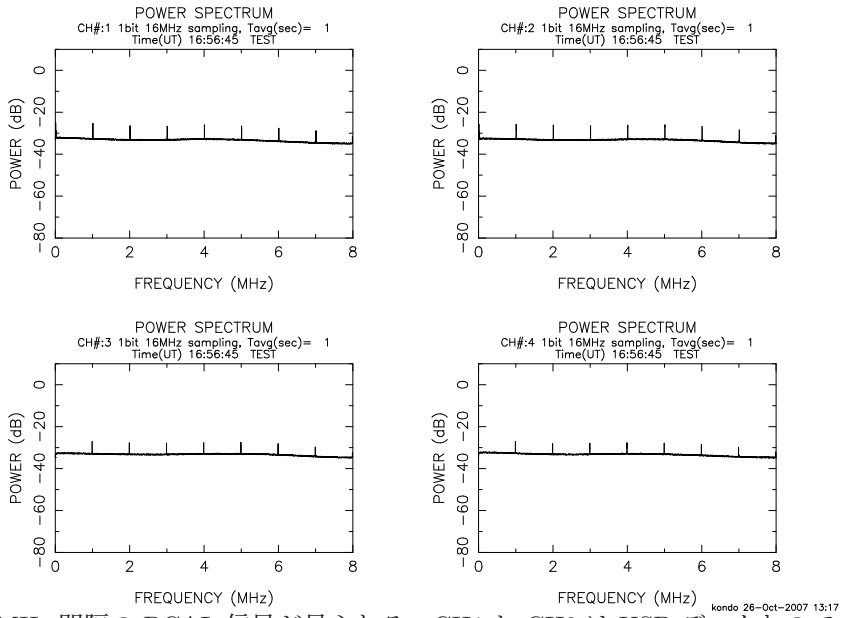
PGDISP – PGPLOT デバイス (/XSERVE,/XTERM 等)

実行例

例 1 . 周波数軸リニア、強度軸ログ表示で 1 秒積分結果のスペクトルをディスプレーに表示しポストスクリプトファイル出力 (pgplot.ps) も行う。

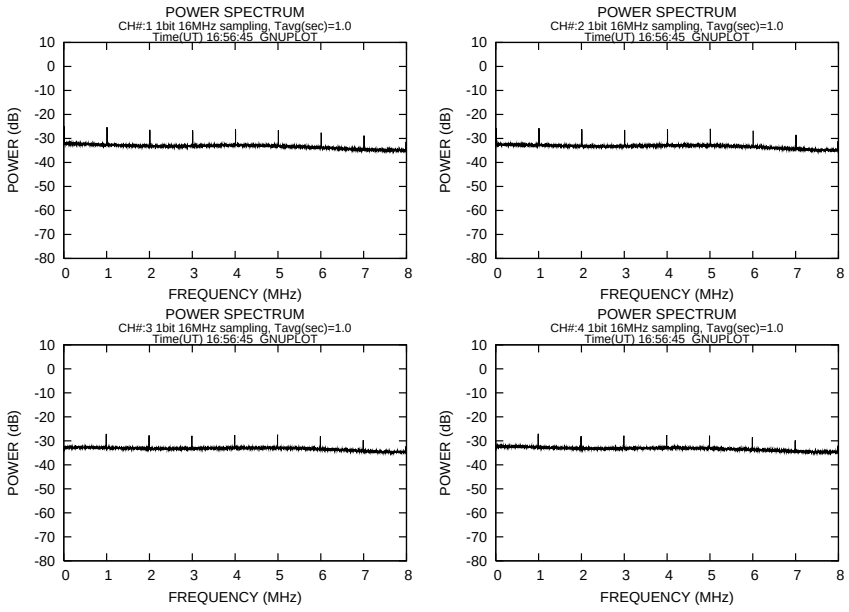
```
speana VSSP32sample.dat 01 1 0 TEST
```

以下に PGPLOT 出力例を示す。



この例では 1MHz 間隔の PCAL 信号が見られる。CH1 と CH2 は USB データなので PCAL 信号は 10kHz, 1010kHz, … に見られる。CH3 と CH4 は LSB データなので PCAL 信号は 990kHz, 1990kHz, … に見られる。

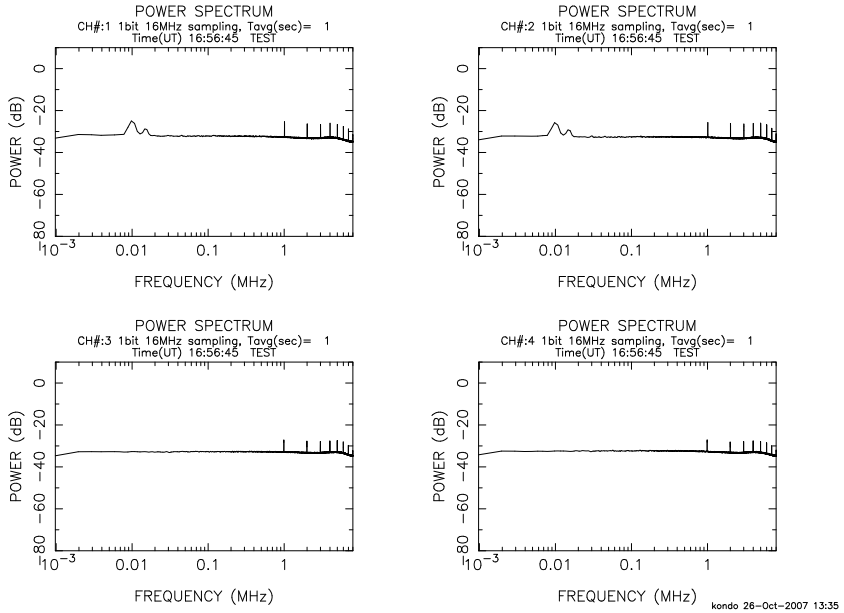
GNUPLOT 出力例を以下に示す。



例２. 例１と同じ。ただし周波数軸をログに設定。

speana VSSP32sample.dat 00 1 0 TEST

以下に出力例を示す。



3.5.8 speana_n

ユーティリティ名

speana_n

機能

オフラインスペクトルのダイナミック表示。

sampling や autoobs で収集したデータのスペクトルのダイナミック表示を行う

注意！PGPLOT のみサポート

実行方法

speana_n filename [options]

ここで filename – データファイル名
0 とすると tds.data が使用される

[options]

-m[ode] mode – 軸表示モードまたは自己相関モードの指定
0 : 強度ログ、周波数ログスケール
1 : 強度ログ、周波数軸リニアスケール （デフォルト）
10 : 強度リニア、周波数軸ログスケール
11 : 強度リニア、周波数軸リニアスケール
-1 : 自己相関関数プロット
-N : N点の自己相関関数プロット

-a|-i|-t sekibun – 単位積分時間の設定 (秒単位)
デフォルトは 0.01

-o soffset – データの処理開始時刻のオフセット（秒単位）
デフォルトは 0.0

-f1 f1khz – 表示周波数範囲の低周波側の設定（kHz 単位）

デフォルトは自動設定

-f2 f2khz – 表示周波数範囲の高周波側の設定（kHz 単位）
デフォルトは自動設定

-min mindbm – 表示パワー強度範囲の最小値の設定（dBm 単位）
デフォルトは自動設定

-max maxdbm – 表示パワー強度範囲の最大値の設定（dBm 単位）
デフォルトは自動設定

-h[alt] – シングル表示モードに設定（１表示ごとに停止）

-s[msec] sleep_msec – 連続表示モード時（デフォルト）の表示間隔を msec 単位で指定する
デフォルトは 0.0

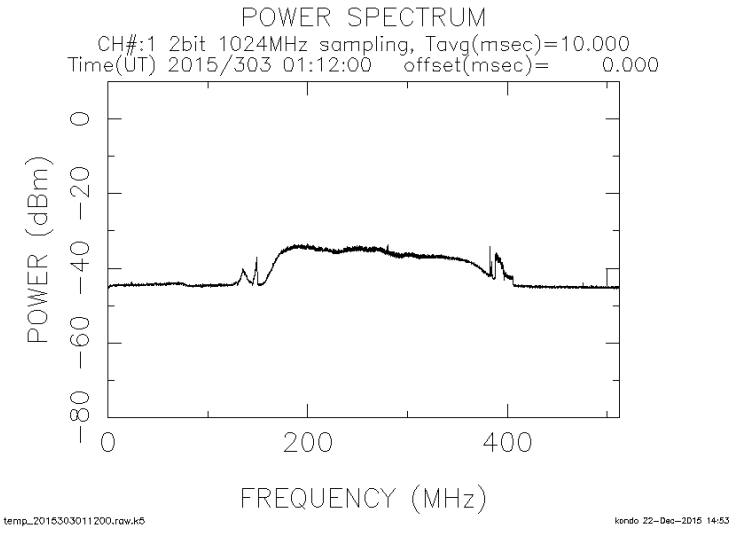
環境変数

PGDISP – PGPLOT デバイス (/XSERVE,/XTERM 等)

実行例

例１. ディスプレーにシングルモードで表示。データは 1ch のみのデータ

speana_n VSSP32sample.dat -h

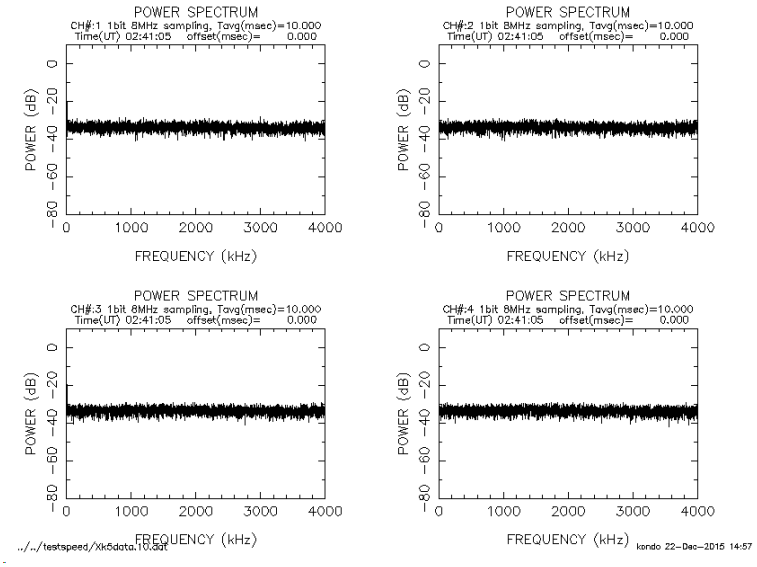


“-h” オプションを指定して走らせると表示後に

```
Hit return key for next span (or "N"on-stop or "Q"uit)
```

と聞いてくるのでリターンキーを押せば次の積分区間データが表示される。“N” を入力すると連続表示モードとなる。

例２. 4ch データの場合



3.5.9 speana2

ユーティリティ名

speana2

機能

オフラインスペクトル表示の高機能版。後処理のためスペクトル値（周波数、値）を speana2.txt に出力すると共に、パワー最大値のサーチを行い表示する。ディスプレイにスペクトルを表示するとともに、PostScript ファイル pgplot.ps (GNUPLOT 使用時は gnuplot.ps）を作成する。

speana[リターン］で以下のようにどちらのグラフィック表示でコンパイルされているかの情報が表示される。

```
speana2 Ver. 2007-11-04
  compliled for GNUPLOT
```

引き続き使用法が表示される。

実行方法

speana2 filename [smode [sekibun1 [skibun [soffset [comment [f1khz [f2khz]]]]]]]

ここで filename – データファイル名
0 とすると tds.data が使用される
smode – 周波数分解能モード
0: 4096 点 （約１ｋＨｚ@４MHｚサンプリング）
1: 4194304 点（約１Ｈｚ@４MHｚサンプリング）
sekibun1 – 単位積分時間 (整数秒単位)
デフォルトは １（秒）
sekibun – 全積分時間 (整数秒単位)
デフォルトは ３００（秒）
負の値とすると絶対値が繰り返し間隔（秒）となる
soffset – データの処理開始時刻のオフセット（整数秒単位）
デフォルトは０
comment – コメント（グラフ上部に表示)。スペースを含まないこと。
省略した場合は会話モード入力になる
注：スペースを含むコメントは会話モードで入力すること
f1khz – パワー最大をサーチする周波数下限（デフォルトは 0）
f2khz – パワー最大をサーチする周波数上限（デフォルトはビデオ上限周波数）

環境変数

PGDISP – PGPLOT デバイス (/XSERVE,/XTERM 等)

実行例

例１. 観測全体のスペクトルを３０秒毎に積分して speana2.txt ファイルに出力する（画面には３０秒ごとのスペクトルと最大値（４番目まで）を表示)。パワー最大を見つける範囲は 920kHz〜960kHz。周波数分解能は 4kHz@4MHz サンプリング

speana2 /k56a/ad4/nz0079/U0790013.dat 0 30 1800 0 USUDA 920.0 960.0

例２. 周波数分解能 1Hz(@4MHz サンプリング) で観測開始から３００秒後から３０秒間だけの積分を行い結果を表示。パワー最大を見つける範囲は 920kHz〜960kHz。（いつの場合も speana2.txt にスペクトルデータが出力されます）

speana2 /k56a/ad4/nz0079/U0790013.dat 1 30 30 300 USUDA 920.0 960.0

例３. 周波数分解能 1Hz(@4MHz サンプリング) で観測開始から０秒後のデータから２秒間積分を６００秒毎に繰り返し結果を表示。パワー最大を見つける範囲は 920kHz〜960kHz。（いつの場合も speana2.txt にスペクトルデータが出力されます）

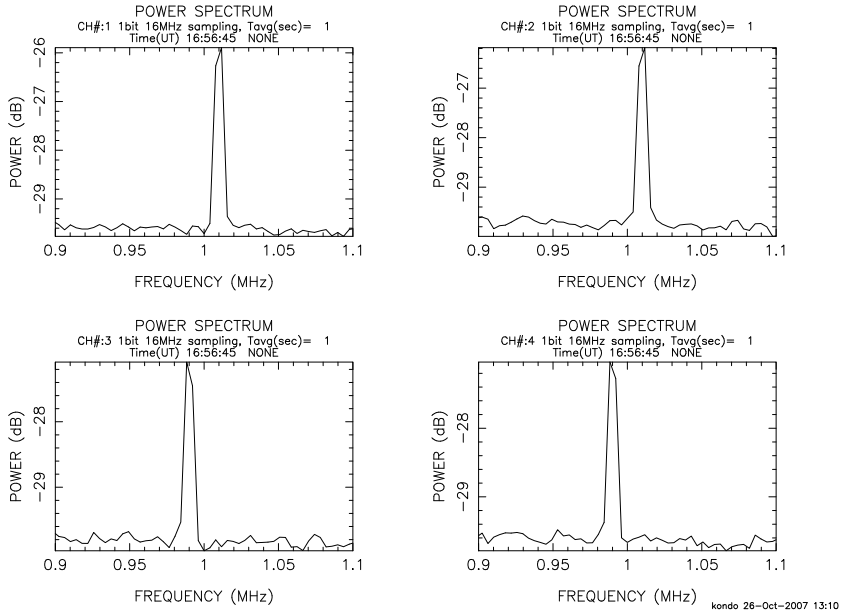
speana2 /k56a/ad4/nz0079/U0790013.dat 1 2 -600 0 USUDA 920.0 960.0

注意：1Hz 分解能@4MHz サンプリングモードで処理を行うと speana2.txt ファイルサイズが大きくなりますので注意すること

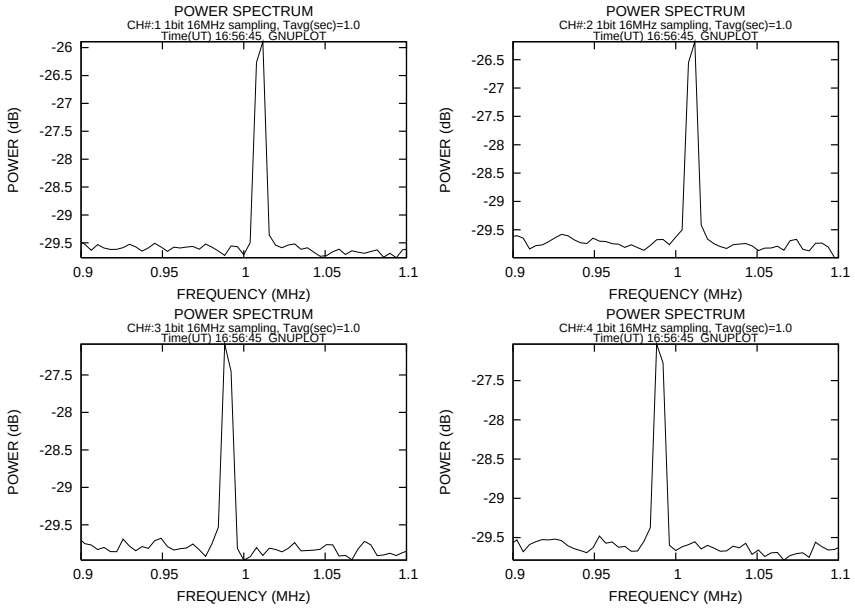
例４. データの最初から１秒間積分したスペクトル表示。パワー最大を見つける範囲は 900kHz〜1100。このデータは CH3 と CH4 が LSB データであるためそれらのチャンネルでは PCAL 信号が 990kHz に見られる。

speana2 VSSP32sample.dat 0 1 1 0 NONE 900 1100

以下に PGPLOT でのグラフ出力例を示す。



以下に示すのは GNUPLOT でのグラフ出力例である。



3.5.10 adbitconv

ユーティリティ名

adbitconv

機能

サンプリングデータファイルの AD ビット数を任意の AD ビット数に変換する

実行方法

adbitconv filename [adbit [outfile [soffset [span]]]]

ここで filename – データファイル名
adbit – 新たなＡＤ分解能 (1,2,4,8) （デフォルトは１）
outfile – データ出力ファイル名
デフォルトは filename+‘.adN’
ここで N は adbit
soffset – スタート時刻オフセット（整数秒）
デフォルト値は 0
span – データ変換の区間（整数秒）
デフォルトはすべて

3.5.11 one2four

ユーティリティ名

one2four

機能

４台の K5/VSSP（含む K5/VSSP32）ボードから１ｃｈモードで取得したデータファイル４つを結合し、１ｃｈモードのデータファイルを作成する

実行方法

one2four file1 file2 file3 file4 [outfile [soffset [span]]]

ここで file1 – 結合するデータファイル名１
file2 – 結合するデータファイル名２※
file3 – 結合するデータファイル名３※
file4 – 結合するデータファイル名４※
outfile – データ出力ファイル名
デフォルトは fil1+‘.k5m’
ここで N は adbit
soffset – スタート時刻オフセット（整数秒）
デフォルト値は 0
span – データ変換の区間（整数秒）
デフォルトはすべて

※ file2〜file4,outfile 名においてディレクトリ（フォルダ）を省略すると file1 と同じディレクトリにあるものと想定した処理を行う

実行例

/vlbi ディレクトリにある R2320002.k5a R2320002.k5b R2320002.k5c R2320002.k5d ファイルを合成して、/vlbi/dat.k5 に出力

one2four /vlbi/R2320002.k5a R2320002.k5b R2320002.k5c R2320002.k5d dat.k5

3.5.12 data_half

ユーティリティ名

data_halfr

機能

サンプリングデータを間引きすることにより見かけのサンプリング周波数を半分にする

実行方法

data_half filename [soffset [period [outfile]]]

ここで filename – データファイル名
0 とするとファイルは tds.data を使う
soffset – スタート時刻オフセット（整数秒）
デフォルト値は 0
period – データ変換の区間（整数秒）
デフォルトはすべて
outfile – データ出力ファイル名
デフォルトは filename+‘.half’

3.5.13 data_double

ユーティリティ名

data_double

機能

サンプリングデータを繰り返すことにより見かけのサンプリング周波数を倍にする

実行方法

data_double filename [soffset [period [outfile]]]

ここで filename – データファイル名
0 とするとファイルは tds.data を使う
soffset – スタート時刻オフセット（整数秒）
デフォルト値は 0
period – データ変換の区間（整数秒）
デフォルトはすべて
outfile – データ出力ファイル名
デフォルトは filename+‘.dbl’

3.5.14 k5v32tok5

ユーティリティ名

k5v32tok5

機能

K5/VSSP32 フォーマットデータを K5/VSSP フォーマットに変換する

実行方法

k5v32tok5 datafile [ofile]

ここで datafile – K5/VSSP32 データファイル名
ofile – 作成する K5/VSSP ファイル名
デフォルト値は datafile+‘.k5’

3.5.15 k5tok5v32

ユーティリティ名

k5tok5v32

機能

K5/VSSP フォーマットデータを K5/VSSP32 フォーマットに変換する

実行方法

k5tok5v32 datafile [yyyymmdd|yyyyddd [ofile]]

ここで	datafile	–	K5/VSSP データファイル名
	yyyymmdd|yyyyddd	–	yyyymmdd：4 桁年, 2 桁月, 2 桁日　（デフォルトは０，０，０） または yyyyddd：4 桁年, 3 桁通日　（デフォルトは０，０）
	ofile	–	作成する K5/VSSP32 ファイル名 デフォルト値は datafile+‘.v32’

3.5.16 data_recov

ユーティリティ名

data_recov

機能

機能１ ヘッダーデータが不具合等で datachk においてヘッダーが検出できない K5/VSSP および K5/VSSP32 データファイルのヘッダー部分の修復を行います。

機能２ 途中のヘッダーが不具合になっている K5/VSSP および K5/VSSP32 データファイルのヘッダー部分の修復を行います。

原理（機能１）：ヘッダーデータの破損具合を調査したところ、破損したヘッダーデータには以下に示される共通パターンが見られることが判明しました（ただし特定のサンプラーボード a で取得したデータのみの調査結果）。

正常ヘッダー：　FFFFFFFF 8Bxxxxxx (x の部分はモード、時刻で変化)
異常ヘッダー：　FFFFxxxx 8BxxxxFF (x の部分は変化)
（注：ヘッダー６４ビットを３２ビット整数×２の１６進数で表示）

そこで、“FFFF???? 8B????FF”（?は任意) のパターンをサーチすることにより、ヘッダー位置を検出します。（データによってはファイルの最初がヘッダーでない場合もあり、この場合に上記パターンのサーチにより最初のヘッダー位置を認識します。）なお、破損したヘッダーから時刻情報の復元は不可能であったため、最初のヘッダーにおける時刻はユーザーが与えなくてはなりません。
※現在は VSSP に対するサーチパターンは “FFFF???? 8B????FF” 、VSSP32 に対しては “FFFF???? 8C??????”としている。

使用上の注意：

1. サンプリングパラメータ、データ開始時刻はユーザーが与える必要があります。（最初のデータが正常の場合は必要ありません）
2. ヘッダー部分が修復されたファイルはデータ部分にも若干の不具合が残っているようです。実際の相関処理の例では遅延方向に山がスプリットしましたが、一番高いピークは正常のようです。ただしＳＮＲは劣化しています。

原理（機能２）：１秒間のデータは正常（データの欠損も超過もない）でヘッダー部分だけが異常であると仮定して、ヘッダー部分を正常なデータに置きかえる

実行方法

data_recov [stype] K5file [sfreq adbit numch start_time [HDpattern1 HDpattern2]] [option]

ここで
- stype – サンプラータイプ ‘32 ’:VSSP32。省略時は VSSP
- K5file – ヘッダーを修復する K5 ファイル名

以下のパラメータは機能１による修復の場合に指定する
- sfreq – サンプリング周波数 (MHz)
  0.04,0.1,0.2,0.5,1,2,4,8,16,32,64
- adbit – ＡＤビット数 (1,2,4,8)
- numch – チャンネル数 (1,4)
- start_time – データ開始時刻（以下のどちらかのフォーマットで）
  “HH:MM:SS” または “seconds in day”

以下のパラメータは異常ヘッダーパターンをデフォルト以外に設定するときに指定する
- HDpattern1 – 異常ヘッダーパターン１ ８桁１６進アスキー
  ‘?’ はその部分は何でもいいことを示す
  （例：FFFF????) (デフォルトは ‘FFFF????’)
- HDpattern2 異常ヘッダーパターン２ ８桁１６進アスキー
  ‘?’ はその部分は何でもいいことを示す
  （例：8B????FF)
  デフォルトは ‘8B????FF’ ただし VSSP32 を指定したときは ‘8C??????’

option（順不同可）は以下のとおり
- -recover – 非会話型で修復まで行う
- -check – 非会話型でチェックのみ行う
- -pat1 HDpattern1 – 異常ヘッダーサーチパターン１のセット
- -pat2 HDpattern2 – 異常ヘッダーサーチパターン２のセット
- -vssp32 – サンプラータイプを VSSP32 にセットする（デフォルトは VSSP）
- -out ofile – 出力ファイル名をセットする
- -in k5file – 修復する K5 ファイル名を指定する

備考：HDpattern1,HDpattern2 は特定のサンプラー（k51a）で見られたタパーン以外に対応するためのパラメータ

ヘッダーが修復されたデータ出力ファイル名はオプションで指定しない限り K5 ファイル名＋ “.edit” となります。

実行例

例１ K5/VSSP データファイル k5sample.dat のデータ修復を会話型で行う
```
data_recov k5sample.dat
```

例２ K5/VSSP データファイル k5sample.dat のデータ修復を会話型で行う。サンプリングパラメータはサンプリング周波数８ＭＨｚ、１ビットＡＤ、４ｃｈ、データ開始時刻は 10:45:00 であることが分かっている
```
data_recov k5sample.dat 8 1 4 10:45:00
```

例３ K5/VSSP32 データファイル sample.k5 のデータ修復を会話型で行う
```
data_recov 32 k5sample.dat
```
または
```
data_recov k5sample.dat -vssp32
```

例４ K5/VSSP32 データファイル sample.k5 のデータ修復を非会話型で行う

data_recov 32 k5sample.dat -recover

または

data_recov k5sample.dat -vssp32 -recover

3.5.17 vssplogana

ユーティリティ名

vssplogana

機能

sampling（autoobs も可）のログファイルおよび datachk のサマリー出力ファイルを解析し、発生エラー状況の統計結果を表示する。ログファイルの形式は自動的に判定され、datachk の出力に対しては、スキャン中のエラーのあったフレーム数の統計結果を表示し、sampling（または autoobs）のログファイルに対してはサンプリング中のエラーの発生件数に関する統計結果を表示する。

実行方法

vssplogana logfile1 [logfile2 […]]

ここで　logfile1, logfile2 ･･･　–　datachk のサマリー出力ファイルまたは sampling(autoobs) のログファイル名

統計結果の例

sampling ログファイルの場合

```
******** SUMMARY OF K5/VSSP32 SAMPLING LOG FILE ANALYSIS **************
  FILE NAME         : v32smplg070207231200.k56b.log
  SCAN LENGTH (sec) : MIN =   300       MAX =   300
  TOTAL # of SCANS  :      305
  START TIME        : 2007/02/07 (038) 23:12:50
  END   TIME        : 2007/02/09 (040) 00:59:06
  PERIOD (hour)     :       25.7711
=========================================================================
    SAMPLER MODE   # of SCANs               # of ERRORS
 fs(MHz) #AD #CH   ALL   Bad       ALL   Err16   Err19   Err20  Others
-------------------------------------------------------------------------
    2     1   1    26     0          0       0       0       0       0
    2     1   4    26     0          0       0       0       0       0
    4     1   1    26     1          1       0       0       1       0
    4     1   4    26     0          0       0       0       0       0
    8     1   1    26     0          0       0       0       0       0
    8     1   4    25     0          0       0       0       0       0
   16     1   1    25     0          0       0       0       0       0
   16     1   4    25     0          0       0       0       0       0
   32     1   1    25     0          0       0       0       0       0
   32     1   4    25     0          0       0       0       0       0
   64     1   1    25     1          1       0       0       1       0
   64     1   4    25     0          0       0       0       0       0
=========================================================================
  Note: Err16  -- DMA process error
        Err19  -- Header is not found at expected position in data
        Err20  -- Header-like data is found in first garbage data
        Others -- Other error code
```

[解釈]

4MHz × 1bit × 1ch モードおよび 64MHz × 1bit × 1ch モードでエラーコード２０が１回発生している。それ以外のエラーはない。

注：2007.3.2 以降のドライバーではエラー１９，２０の発生を抑制しているのでこのエラーがでることはない。

datachk サマリー出力の場合

```
*************** SUMMARY OF DATACHK LOG FILE ANALYSIS ********************
  FILE NAME          : v32dtchk070207231200.k56b.log
  SCAN LENGTH (sec) : MIN =   300       MAX =   301
  TOTAL # of SCANS  :       304
  START TIME         : 2007/02/07 (038) 23:12:52
  END   TIME         : 2007/02/09 (040) 00:59:02
  PERIOD (hour)      :        25.7694
================================================================================
   SAMPLER MODE   # of SCANs   TOTAL              # of BAD FRAMES
 fs(MHz) #AD #CH   ALL  Herr   FRAME    SLIP  T_DIS T_W_BS AUXMIS   EFLG  B_REV
--------------------------------------------------------------------------------
    2     1   1     0     0       0       0      0      0      0      0      0
    2     1   1    26     0    7800       0      0      0      0      0      0
    2     1   4    26     0    7800       0      0      0      0      0      0
    4     1   1    26     0    7801       1      0      0      0      0      0
    4     1   4    26     0    7800       0      0      0      0      0      0
    8     1   1    25     0    7500       0      0      0      0      0      0
    8     1   4    25     0    7500       0      0      0      0      0      0
   16     1   1    25     0    7500       0      0      0      0      0      0
   16     1   4    25     0    7500       0      0      0      0      0      0
   32     1   1    25     0    7500       1      0      0    300      0      0
   32     1   4    25     0    7500       0      0      0      0      0      0
   64     1   1    25     0    7501       1      0      0      0      0      0
   64     1   4    25     0    7500       0      0      0      0      0      0
================================================================================
  Note: Herr   -- # of scans where 1st 64bit is not a header
        SLIP   -- # of frames where bit lack or make occurred
        T_DIS  -- # of frames where time discontinuity occurred
        T_W_BS -- # of frames where both time discontinuity and
                  bit slip or make occurred
        AUXMIS -- # of frames where AUX field misalignment occurred (data is OK)
        EFLG   -- # of frames where EFLG (burst error) detected
        B_REV  -- # of frames where one bit reversal in a sync field detected
```

[解釈]

4MHz × 1bit × 1ch モードおよび 64MHz × 1bit × 1ch モードでビットスリップまたはビットメークのあったフレームが１つある。それは sampling のエラー解析からこれらのモードでエラーコード２０が１回発生しているので、エラーコード２０発生に伴う再サンプリング開始により、フレーム内のデータが不連続となったためと思われる。

32MHz × 1bit × 1ch モードで AUXMIS(AUX フィールドの並び異常) のあったフレームが３００ある。そのモードでビットスリップまたはビットメークのあったフレームが１つであり、また１スキャンは３００秒（３００フレーム）なので、恐らくスキャンの最初のフレームでビットスリップまたはビットメークが発生し、その後スキャンの終わりまで AUXMIS が発生していると思われる。

3.5.18 aux_recov

ユーティリティ名

aux_recov

機能

K5/VSSP32 データを datachk でチェックした際に AUX MISALIGN (misaligned AUX field) と表示されるデータの修復を行います。

修復の原理：こうした不具合の生じたデータを詳細に調査したところ、今の所以下のような規則にしたがった不具合であることが判明しています。

１. 最初のフレームのヘッダーは正常である

２. 最初のフレームのサンプリングデータはいつも１６ビット多い。

３．２フレーム以降のヘッダー（３２バイト）の９バイト以降に常に以下の規則に従った配列乱れが発生している

正常バイト並び　0,1,2,3,4, ⋯ ,8, 9,10,11, ⋯ ,28,29,30,31

異常バイト並び　0,1,2,3,4, ⋯ ,8,11,12,13, ⋯ ,30,31, 9,10

４．２フレーム以降のフレームあたりのサンプリングデータ数は正常である

この規則にしたがって、最初のフレームデータでは最後の１６ビットデータを削り、２フレーム以降のヘッダーデータは正常のバイト並びに修復しています。

実行方法

aux_recov datafile [ofile]

ここで　datafile　–　修復を行う K5/VSSP32 データファイル名

ofile　–　修復後の K5/VSSP32 ファイル名

デフォルト値は datafile+@.recov’

修復されたデータ出力ファイル名は　K5 ファイル名＋”.recov” となります。

実行例

```
> ./aux_recov tds.data
************ aux_recov  Ver. 2007-03-15 ************
FILE : tds.data  (64028672 bytes)

 This is K5/VSSP32 format data.

  1st Extended Header Info is as follows.
    Year = 2007    Total Day = 74
    Version  Major = 1  Minor = 0    # of AUX Bytes = 20
    AUX Field Data :  55551002 55555555 55555555 69626568 20202020
        Format # = 2
        Filter (MHz) : 16
        PC host name :  hebi

  Recoverrable AUX field error was detected!
  Recover condition : (slip byte = -2)

AUX FIELD recover start!!

FILE : tds.data  (64028672 bytes)

 This is K5/VSSP32 format data.

  1st Extended Header Info is as follows.
    Year = 2007    Total Day = 74
    Version  Major = 1  Minor = 0    # of AUX Bytes = 20
    AUX Field Data :  55551002 55555555 55555555 69626568 20202020

 FMT A/D CH  f(kHz)  TIME   seconds      AUX FIELD
VS32  1   4  32000 00:41:39   2499   10140E4A 55551002  ....
VS32  1   4  32000 00:41:40   2500   10140E4A 55551002  ....
VS32  1   4  32000 00:41:41   2501   10140E4A 55551002  ....
VS32  1   4  32000 00:41:42   2502   10140E4A 55551002  ....
VS32  1   4  32000 00:41:43   2503   10140E4A 55551002  ....

Original File : tds.data
Created  File : tds.data.recov
Time elapsed for processing is 5.698000 sec
```

3.5.19 pcalcheck

ユーティリティ名

pcalcheck

機能

K5/VSSP および VSSP32 データファイルからを PCAL を検出しグラフ表示を行います。なおグラフィッ

ク表示は PGPLOT を使う方がスムースに表示されます。

speana[リターン］で以下のようにどちらのグラフィック表示でコンパイルされているかの情報が表示される。

```
pcalcheck (Ver. 2007-11-04)
   compiled for GNUPLOT
```

引き続き使用法が表示される。

実行方法

pcalcheck k5file [option]

または

pcalcheck ofile [-pgplot device | -ps] — 再描画の場合

ここで
- k5file – K5 ファイル名
- ofile – pcalcheck 出力ファイル名

option（順不同可）は以下のとおり
- -fa[ll] pcalf – すべてのチャンネルの PCAL 周波数をセット (kHz)
  デフォルトは 10kHz
- -f1 pcalf1 – CH1 の PCAL 周波数をセット (kHz)
- -f2 pcalf2 – CH2 の PCAL 周波数をセット (kHz)
- -f3 pcalf3 – CH3 の PCAL 周波数をセット (kHz)
- -f4 pcalf4 – CH4 の PCAL 周波数をセット (kHz)
- -integ integ – 積分時間（処理スパン）をセット (sec)
  デフォルトは全スパン
- -pgplot device – PGPLOT デバイスをセット (/NULL：表示抑制)
  /CPS ポストスクリプト出力
  デフォルトはディスプレー
- -ps – POSTSCRIPT 出力にセット（再描画モードの時有効）
- -after – 全部の処理が終わってからグラフ表示を行うモードにセット
  （GNUPLOT の場合のデフォルト）
- -noafter – 処理を行いながらグラフ表示を更新するモードにセット
  （PGPLOT の場合のデフォルト）
- -out ofile – 結果の出力ファイルをセット
  （デフォルトは pcalcheck_out.txt）
- -nocheck – ヘッダーのチェックを行わないモードをセット

‘-nocheck’ を指定した時は以下のパラメータをセットすること
- -vssp32 – サンプラーは VSSP32 モード（デフォルトは VSSP）
- -sfreq sfreq – サンプリング周波数をセット (MHz)
- -adbit adbit – AD ビットをセット
- -numch numch – CH 数をセット

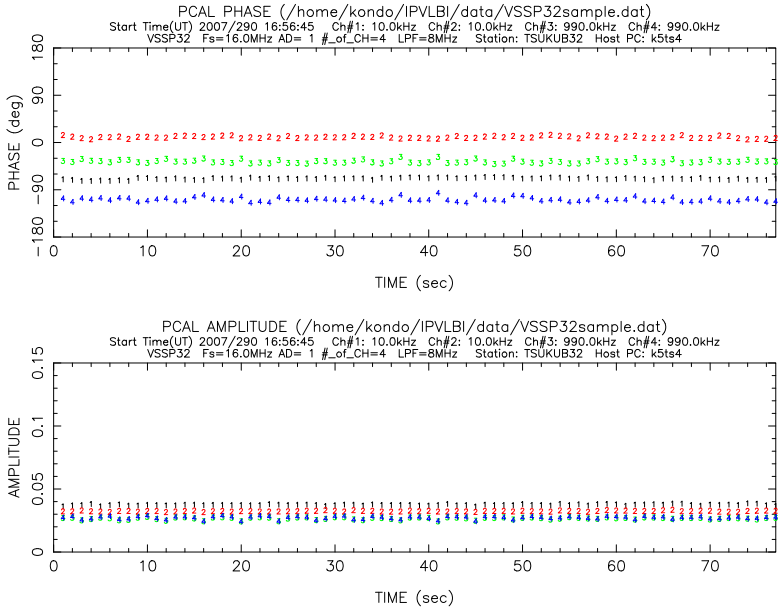
実行例

以下の実行例は CH3 と CH4 が LSB データで PCAL 周波数が 990kHz となっている場合の実行例である。

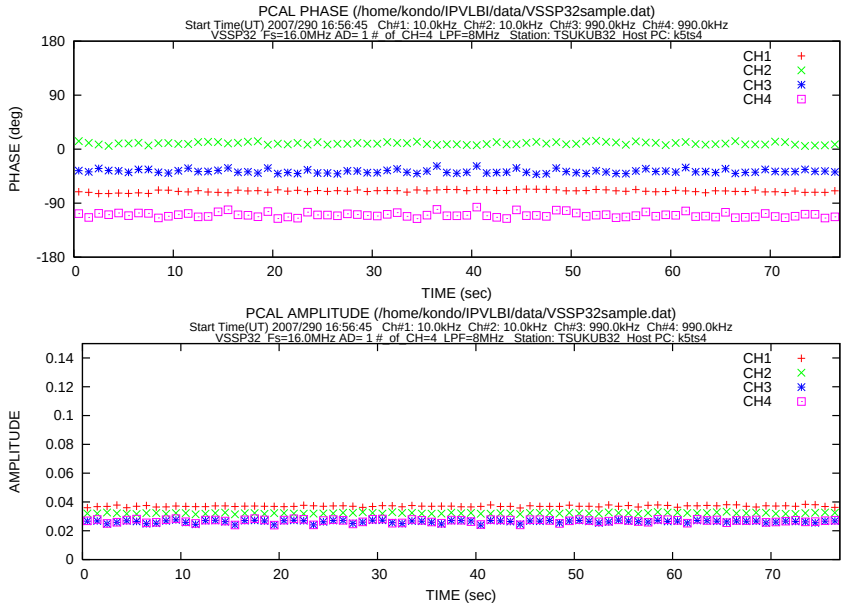
```
> pcalcheck /home/kondo/IPVLBI/data/VSSP32sample.dat -f3 990 -f4 990
```

```
pcal_engine:  Ver. 2007-10-25
pcal_engine: Data File is /home/kondo/IPVLBI/data/VSSP32sample.dat
checkheader: Header (K5/VSSP32) Sync Detected!!
checkheader:   File : /home/kondo/IPVLBI/data/VSSP32sample.dat
checkheader:   A/D(bits) 1  CHs 4  SFreq(kHz) 16000  Time 16:56:45  sec 61005
pcal_engine:  << VSSP32 format
pcal_engine:  << # of samples in a unit (usampl) = 200000
pcal_engine:  << PP period in sec (t1pp)         = 1.0
pcal_engine:  << # of usampl  in a PP (nspp)     = 80
pcal_engine:  << # of usampl  in 1 sec (imax)    = 80
pcal_engine:  << # of bytes   in a usampl (numb) = 100000
pcal_engine:  << File size in bytes              = 616038400
pcal_engine:  << Supposed scan length (sec)      = 77.004492
pcal_engine:   dtimex 61005.000000
pcal_engine: Start X data time : 61005.000000
pcal_engine:  << Detected PCAL >>
pcal_engine:   CH  samples  Freq(kHz)    Amp  Phase(deg)
pcal_engine:    0 16000000     10.0  0.035995   -70.96
pcal_engine:    1 16000000     10.0  0.032027    13.44
pcal_engine:    2 16000000    990.0  0.026666   -35.83
pcal_engine:    3 16000000    990.0  0.026923  -107.31
pcal_engine:   Time elapsed for 1PP processing is 10.000000 sec
pcal_engine:   X data time (BOPP) : 61005.987500
checkheader: Header (K5/VSSP32) Sync Detected!!
checkheader:   File : /home/kondo/IPVLBI/data/VSSP32sample.dat
checkheader:   A/D(bits) 1  CHs 4  SFreq(kHz) 16000  Time 16:56:46  sec 61006
pcal_engine:  << Detected PCAL >>
pcal_engine:   CH  samples  Freq(kHz)    Amp  Phase(deg)
pcal_engine:    0 16000000     10.0  0.036730   -71.94
pcal_engine:    1 16000000     10.0  0.031975    10.46
pcal_engine:    2 16000000    990.0  0.026837   -37.84
pcal_engine:    3 16000000    990.0  0.027834  -113.84
pcal_engine:   Time elapsed for 1PP processing is 10.000000 sec
pcal_engine:   X data time (BOPP) : 61006.987500
checkheader: Header (K5/VSSP32) Sync Detected!!
checkheader:   File : D:\IPVLBI\data\k5vssp32\VSSP32sample.dat
checkheader:   A/D(bits) 1  CHs 4  SFreq(kHz) 16000  Time 16:56:47  sec 61007
.........（中略）

pcal_engine:  << Detected PCAL >>
pcal_engine:   CH  samples  Freq(kHz)    Amp  Phase(deg)
pcal_engine:    0 16000000     10.0  0.036269   -69.42
pcal_engine:    1 16000000     10.0  0.032516     7.94
pcal_engine:    2 16000000    990.0  0.026824   -37.98
pcal_engine:    3 16000000    990.0  0.027339  -112.75
pcal_engine:   Time elapsed for 1PP processing is 9.000000 sec
pcal_engine:   X data time (BOPP) : 61081.987500
checkheader: Header (K5/VSSP32) Sync Detected!!
checkheader:   File : D:\IPVLBI\data\k5vssp32\VSSP32sample.dat
checkheader:   A/D(bits) 1  CHs 4  SFreq(kHz) 16000  Time 16:58:02  sec 61082
DatRead1bitSv: EOF found. Try extended file if existed
DatRead1bitSv: File to be read is D:\IPVLBI\data\k5vssp32\VSSP32sample.dat1
DatRead1bitSv: Extended file not found!


====================  PCALCHECK SUMMARY  ================
 FILE NAME = D:\IPVLBI\data\k5vssp32\VSSP32sample.dat
 START TIME= 2007Y290D16h56m45sec
 SCAN LENGTH  (sec)  =   77.0
 ACCUM. PERIOD(sec)  at pcalcheck  =   77.0
 SAMPLER : K5/VSSP32  Fs=16.0MHz  AD= 1  #CH=4  LPF=8MHz

  CH#  PCAL FREQ.(kHz)      PHASE(deg.)     AMPLITUDE
 ------------------------------------------------------
   1       10.0               -69.86         0.0371
   2       10.0                 9.60         0.0322
   3      990.0               -37.05         0.0263
   4      990.0              -109.76         0.0267
 ------------------------------------------------------
============================================================
```

1秒毎の PCAL 位相が以下の図のように表示され、最後にサマリーがディスプレーに表示される。この図は PGPLOT の場合の例。



以下は GNUPLOT での表示例。



3.6 K5/VSSP32 ユニット（ボード）チェック用シェルスクリプトの使い方

それぞれのシェルスクリプトは引数無しで実行すると簡単な使用法が表示されます。

3.6.1 vssp32test.sh

サンプリング周波数を変えての自動テスト。src ディレクトリにてシェルスクリプト vssp32test.sh を実行する（vssp32test.sh に実行権限を与えておくこと！）

使用法

vssp32test.sh span [nkai [sfmax [folder [logfile1 [logfile2]]]]

ここで
- span – スキャン長 (sec)
- nkai – 一連のテストの繰り返し回数 （デフォルトは１）
- sfmax – 最大のサンプリング周波数 (2,4,8,16,32,64)
  （デフォルトは 64）
- folder – サンプリングデータを書き込むディレクトリ
  （デフォルトは現在のディレクトリ）
- logfile1 – datachk の結果を出力するファイル名
  （デフォルトは v32dtchkYYMMDDHHMMSS.HOST.log）
- logfile2 – sampling プログラムのログ出力ファイル名
  （デフォルトは v32smplgYYMMDDHHMMSS.HOST.log）
  ここで、YYMMDDHHMMSS はチェック開始時の時刻
  HOST はＰＣのホスト名

* 一連のテストとはサンプリング条件を
  サンプリング周波数　２，４，８，１６，３２，６４MHｚ
  （オペレータが sfmax を与えたときはその値が最大値となる）
  ＣH数　１，４
  ＡＤビット数　１ビット　固定
の１２通りに変えながら span で与えられた時間ずつ sampling によるデータ収集と datachk によるデータチェックおよび vssplogana によるログファイルの解析を繰り返す。

span を３００秒とするとデフォルトパラメータ時の一連のテストを行う時間は約３００秒×１２＝１時間となる。

注１：テスト開始時に sampling のログファイル中の時刻（ＰＣ時刻を使用）とサンプリング中のデータの時刻の同期をとるため、サンプラーボードの時刻はＰＣの時間を使ってセットされる (timesetpc を使用) ので、実際の観測時には時刻を新たにセットし直すこと！

実行例

vssp32test.sh 300 24

上記は各スキャン長３００秒で一連のテストを２４回繰り返す。所要時間の概算はデフォルトパラメータの時は

所要時間＝ span × 12 × nkai　（秒）

で求めることができる。上の例の場合、約２４時間となる。
ホストＰＣが k56a だった場合、以下のような２つのログファイルが作成される
v32dtchk070209132512.k56a.log <== datachk のサマリーログ
v32smplg070209132512.k56a.log <== sampling のサマリーログ
ただし、テスト開始の時間が 2007 年 2 月 9 日 13:25:12 だったとする

ログファイルは自動的に１スキャンごとに解析されるが、テスト後再度 vssplogana を起動して独立に行うことも可能である。

3.6.2 vssp32test2.sh

サンプリングパラメータを固定しての耐久テスト。src ディレクトリにてシェルスクリプト vssp32test2.sh を実行する（vssp32test2.sh に実行権限を与えておくこと！）

使用法

vssp32test2.sh span [sfreq [adbit [ch [nkai [folder [logfile1 [logfile2]]]]]]

ここで
- span – スキャン長 (sec)
- sfreq – サンプリング周波数 (MHz) （デフォルトは 32）
- adbit – ＡＤビット数　1,2,4,8 （デフォルトは１）
- ch – ｃｈ数　1,4 （デフォルトは１）
- nkai – 一連のテストの繰り返し回数 （デフォルトは１）
- folder – サンプリングデータを書き込むディレクトリ
  （デフォルトは現在のディレクトリ）
- logfile1 – datachk の結果を出力するファイル名
  （デフォルトは v32dtchkYYMMDDHHMMSS.HOST.log）
- logfile2 – sampling プログラムのログ出力ファイル名
  （デフォルトは v32smplgYYMMDDHHMMSS.HOST.log）
  ここで、YYMMDDHHMMSS はチェック開始時の時刻
  HOST はＰＣのホスト名

オペレータで与えられたサンプリングパラメータで span で与えられた時間ずつ sampling によるデータ収集と datachk によるデータチェックおよび vssplogana によるログファイルの解析を繰り返す。

注１：テスト開始時に sampling のログファイル中の時刻（ＰＣ時刻を使用）とサンプリング中のデータの時刻の同期をとるため、サンプラーボードの時刻はＰＣの時間を使ってセットされる (timesetpc を使用) ので、実際の観測時には時刻を新たにセットし直すこと！

実行例

vssp32test2.sh 300 32 1 4 24

上記は各スキャン長３００秒で 32MHz × 1bit × 4ch モードのサンプリング２４回繰り返す。所要時間の概算は　　所要時間＝ span × nkai　（秒）で求めることができる。上の例の場合、約 120 分となる
ホストＰＣが k56a だった場合、以下のような２つのログファイルが作成される
　v32dtchk070209132512.k56a.log <== datachk のサマリーログ
　v32smplg070209132512.k56a.log <== sampling のサマリーログ
ただし、テスト開始の時間が 2007 年 2 月 9 日 13:25:12 だったとする

ログファイルは自動的に１スキャンごとに解析されるが、テスト後再度 vssplogana を起動して独立に行うことも可能である。

3.6.3 vssp32test3.sh

サンプリング周波数を変えての自動テスト。周波数の範囲を設定。vssp32test.sh とほぼ同じだが、最低のサンプリング周波数を引数に追加。src ディレクトリにてシェルスクリプト vssp32test3.sh を実行する（vssp32test3.sh に実行権限を与えておくこと！）

使用法

vssp32test3.sh span [nkai [sfmin [sfmax [folder [logfile1 [logfile2]]]]]

ここで
- span – スキャン長 (sec)
- nkai – 一連のテストの繰り返し回数 （デフォルトは１）
- sfmin – 最低のサンプリング周波数 (2,4,8,16,32,64)
  （デフォルトは 2）
- sfmax – 最大のサンプリング周波数 (2,4,8,16,32,64)
  （デフォルトは 64）
- folder – サンプリングデータを書き込むディレクトリ
  （デフォルトは現在のディレクトリ）
- logfile1 – datachk の結果を出力するファイル名
  （デフォルトは v32dtchkYYMMDDHHMMSS.HOST.log）
- logfile2 – sampling プログラムのログ出力ファイル名
  （デフォルトは v32smplgYYMMDDHHMMSS.HOST.log）
  ここで、YYMMDDHHMMSS はチェック開始時の時刻
  HOST はＰＣのホスト名

実行例

vssp32test3.sh 10 24 16

上記は各スキャン長１０秒でサンプリング周波数を 16,32,64MHz と変えながら一連のテストを２４回繰り返す。

4 実観測手順

ここでは autoobs を使用しての実観測の手順を説明します。

4.1 信号入力と時刻合わせ

1. 基準信号（１ＰＰＳ　ＴＴＬレベル，１０ＭＨｚ）を K5/VSSP（含む K5/VSSP32）ボード（メイン）に入力。signalcheck でボードに基準信号が供給されていることを確認する。
2. サンプラーを４ｃｈモードで観測する場合は補助ボードのアナログ入力ｃｈ（１－４ｃｈ）にビデオ信号を入力。サンプラーを１ｃｈモードで観測する場合は主ボード（10MHz,1PPS を入力しているボード）のデータ入力にビデオ信号を入力する。ｃｈのアサインについては別途決める。
3. 「のぞみ」モード (※) の観測を行うときは、1ch サンプラーボードには 4ch サンプラーボードの ch1 と同じビデオ信号を入力すること！
4. timesettk yyyy mm dd hh mm ss でボードの時刻をＵＴでセットする。セットした時刻は　timedisp で確認すること（１０秒間表示）　（ＰＣの時刻もセットしたい場合は、スーパーユーザー権限で pctimeset を実行。観測には必ずしもＰＣの時刻を合わせる必要はない)。秒単位でセット時刻を修正するには timeadjust を使用する。

※「のぞみ」モード]：
「のぞみ」観測時（電波源名が NOZ で始まる時)、自動的に１ ch モード、それ以外の時は numch パラメータで指定したモード（通常４ ch モード）でデータ収集を行いディスク容量を節約する。1ch 観測を「のぞみ」以外の衛星に変更するには環境変数 K5SATKEY で指定する。K5SATKEY で指定するキーワードは最大８文字で通常３文字。電波源との一致の判定は先頭一致方式。（例　bsh で「はやぶさ:電波源名 HYBxxxx」観測を 1ch モードに指定する　　K5SATKEY=HYB; export K5SATKEY）
注意：1ch サンプラーボードには 4ch サンプラーボードの ch1 と同じビデオ信号を入力すること！

4.2 ビデオ信号レベルと DC オフセットの設定

1. monit 4　を実行することにより４ｃｈの信号レベルがモニターできるので、１ビット量子化モード (1 ビット AD) で観測する場合は、AD 変換器の不感帯の影響を低減するため、できるだけサチリ気味に、多ビット量子化モード (2,4,8 ビット AD) で観測する場合はフルスケールのガウス分布になるように入力レベルを調整する。サンプラーを１ｃｈモードで使用する場合は monit 1 です。
2. setdcoffset AUTO　を実行する。これで K5/VSSP32 を使用している際には最適なＤＣオフセットに自動的に設定されるとともに、各 ch ごとの設定値、信号の平均値、信号の標準偏差がログに出力される。K5/VSSP では信号の平均値、信号の標準偏差のみがログに出力される。

4.3 自動観測の実行

autoobs で自動観測を実行する。autoobs の詳細は 3.4.1 autoobs の項 (21 ページ) を参照。スケジュール通りのスパン時間で観測する際は span パラメータは０（プログラムデフォルトは０です）とすること。

予め 3.4.1 autoobs の項で説明されているパラメータファイルを準備しておいて、autoobs を実行するのが間違いも少なく便利です。

事前にテストを行う場合は autoobs 実行時にオプション “-TEST” や “-shift” を使うと便利です。

よく使う実行方法

観測の実行：

autoobs -k5runinfo.txt　　（ただし、パラメータファイルを “k5runinfo.txt” とする）

テストの実行その１（“-TEST” オプションによる方法）：
autoobs -k5runinfo.txt -TEST （１分後にテストスタート）

テストの実行その２（“-shift” オプションによる方法）：
autoobs -k5runinfo.txt -shift 2007/03/20-12:10:20 （2007 年 3 月 20 日 12 時 10 分 20 秒からテストスタート）

実行時の画面例

```
*****************************************************************
*              AUTOOBS Ver 4.02 (2007-02-22) by NICT            *
*    K5/VSSP32 unit : Time Get from SAMPLER  : Naming Type 2     *
*****************************************************************

  Schedule File     :  ./r1267.test.skd (type : SKED)
  Experiment Code   :  r1267          Freq_G (a)
  Log File          :  ./r1267_T.k56a.log
  Stations Included :  FORTLEZA WESTFORD TIGOCONC HOBART26 TSUKUB32 KOKEE
  My Station Name   :  TSUKUB32 (My Station ID : T (Ts))   Subnet Mode : ON
  Current Out Dir   :  /k56a/ad4/    271629.2MB left
  Next    Out Dir   :  /k51d/ad5/    111861.3MB left
  Total Scan Number :  1056       Satellite Mode Keys: HYB NOZ GEO
  1st  Scan         :  2007/03/17 17:00:00  ( 138sec)  3C446
  Last Scan         :  2007/03/18 16:59:10  (  40sec)  1124-186
  Obs Range (start) :  2007/03/17 17:00:00    each scan EARLY start : 10 sec
  Obs Range (end)   :  2007/03/18 16:59:50
  Sampling Mode for Next Scan :  16MHz 1bit 4ch (LPF 8MHz)

  Next Scan(No.0002):  2007/03/17 17:00:00  (40sec)  OJ287 [320.0MB]

  Time Now (UTC)    :  2007/03/17 02:54:48
```

各項目の説明

Schedule File	– スケジュールファイル名
(type :)	– スケジュールファイルのタイプ　SKED — VEX
	– “-shift” オプションで走らせると項目の最後に　”(SHIFT mode)” という表示が追加される
	– “-TEST” オプションで走らせると項目の最後に　”(TEST mode)” という表示が追加される
Experiment Code	– 実験コード
Log File	– ログファイル
Stations Included	– スケジュール中に含まれている局表示
My Station Name	– 自局名 と局ＩＤ. 局 ID は SKED タイプの場合は 1 文字 ID と ( ) 内に 2 文字 ID. VEX タイプのスケジュールの場合は 2 文字 ID
Current Out Dir	– 現在の出力先ディレクトリと残容量
Next Out Dir	– 現出力先が一杯になった場合、次に選択されるディレクトリ
Total Scan Number	– 全スキャン数
Satellite Mode Keys	– 「のぞみ」モードで使用するキー
1st Scan	– 最初のスキャン情報
Last Scan	– 最後のスキャン情報
Obs Range (start)	– 観測範囲（範囲の最初のスキャン開始時刻）
Obs Range (end)	– 観測範囲（範囲の最後のスキャン終了時刻）
Sampling Condition for Next Scan	– 次スキャンのサンプリング情報

Next Scan(No.XXXX) – 次の観測情報　( ) 内はスキャン時間、[ ] 内はデータ量
Time Now (UTC) – 現在の時刻（ＵＴＣ）

5 文書更新履歴

2015.12.22 oscillo, speana_n を追加したことによるマニュアルの追加。

2009.11.22 autoobs のパラメータファイル中の$SATKEYS に ‘NONE’ というキーワードを許したことによる改訂。

2009.10.15 データチェックソフトウェア群に観測データの時刻情報を書き換えるプログラム datatime_edit を追加したことによるマニュアルの追加。monit に時刻表示抑制オプションを追加したことによるマニュアルの改訂。

2008.02.13 monit に時刻表示抑制オプションを追加したことによるマニュアルの改訂。

2007.11.14 GNUPLOT でのグラフ表示のサポートを開始したことによるマニュアルの改訂。

2007.10.30 PCAL チェックプログラム pcalcheck を追加した事によるマニュアルの追加。

2007.10.20 ヘッダー修復プログラム data_recov を VSSP32 にも対応させ、さらに非会話型の実行を可能にしたことを反映させるためのマニュアルの改修。