国立天文台殿

データ取得ツールキット

使用手引書

version 5.0

2003年3月

FUJITSU AMERICA, INC.

ツールキットサブシステム部

＜　目次　＞

1. 概要

1.1 適用範囲

本資料は、データ取得システム・ツールキットサブシステム（以下「ツールキット」）の使用手引について記述したものです。

ツールキットの概要、仕様の詳細につきましては、以下の文書を御参照ください。
・ソフトウェア構成設計書・データ取得システム・ツールキットサブシステム

1.2 推奨構成

ツールキットを利用するにあたっての推奨構成は、以下のとおりです。

表1.2　推奨構成

マシン	Sun SPARCstation 20　または　Ultra 1
ＯＳ	Solaris 2.5.1　または　Solaris 8
コンパイラ	SPARCompilerC 3.0.1 , 4.0 または 5.0

2. 提供品目

2.1 提供品目一覧

提供品目は、以下のとおりです。

表2.1　提供品目一覧

No	品目名	種別	提供形態	備考
1	使用手引書	ドキュメント	文書	
2	ソースプログラム	プログラム	8mmテープ	*1
補足*1　ftp, httpなどによる入手も可能とする.				

2.2 ＰＤＳ

ツールキットのモニタ表示ツールを使用する場合，ＰＤＳとして、以下のプログラムを使用しています。

ツールキット利用者側でのインストールをお願いいたします。

表2.2　ＰＤＳ一覧

No	名称	用途	動作確認バージョン
1	SAOtng	データ表示画面	1.5

2.3 ソースプログラムの構成

ツールキットソースプログラムをディスク上に展開すると、表2.3のようなディレクトリ構成（主要なもの）になります。
（展開方法の詳細については、「3.インストール手順」を御参照ください。）

表2.3　ディレクトリ構成 (1/2)

ディレクトリ	内容
...	ツールキットインストール先ディレクトリ
└ daqtk	ツールキットディレクトリ
├ Install	ツールキットインストール用シェル
├ Uninstall	ツールキットアンインストール用シェル
├ bin/	ツールキット実行形式ファイル格納用
├ dat/	データファイル格納用
├ inc/	インクルードファイル格納用
├ lib/	ライブラリファイル格納用
├ log/	ログファイル格納用
├ src/	ツールキットソースファイル格納用
├ cmd/	コマンド通信機能
├ com/	共通関数
├ dat/	取得データ転送機能
├ func/	FITSキーワード値算出機能
├ get/	ステータスデータ取得機能
├ makefits/	FITSファイル作成ツール
├ monitor/	モニタ表示ツール
├ sh/	シェル
├ stars/	STARS登録チェックツール
├ stat/	ステータス転送機能
└ through/	透過モード・コマンド通信機能

表2.3　ディレクトリ構成 (2/2)

ディレクトリ	内容
└ sample/	サンプルプログラム格納用
├ exe/	サンプルプログラム実行形式ファイル格納用
├ data/	サンプルプログラムデータファイル格納用
├ fits/	サンプルプログラム出力fitsファイル格納用
├ log/	サンプルプログラムログファイル格納用
└ src/	サンプルプログラムソースファイル格納用

3. インストール手順

本ツールのインストール手順の概略は以下のとおりです。

・ツールキットのインストール
・ツールキット環境変数の設定
・ツールキットシェル変数の設定
・システムファイルの設定
・SAOtngのインストール（モニタ表示ツールを使用しない場合は必要ありません）

3.1 ツールキットのインストール

1)入手したツールキット圧縮ファイル(daqtk.tar.Z)をインストール先ディレクトリに移します。

```
% mv daqtk.tar.Z ...
```

(...はツールキットのインストール先のディレクトリを指します)

2)圧縮ファイルを解凍し、tarにより展開します。

```
% cd ...
% uncompress daqtk.tar.Z
% tar xvfo daqtk.tar
```

3)以下のシェル（全メイク用シェル）を実行します。

```
% cd daqtk
% Install （OBCP名）                    (*1)
```

実行により、以下のものが生成または変更されます。

・src/配下のRPCLファイル
・src/およびsample/配下の各ディレクトリ内のオブジェクトファイル
・lib/内のアーカイブファイル
・bin/内のツールキット実行形式ファイル
・sample/exe/内のサンプル実行形式ファイル

4)以上でツールキットのインストールは完了です。

補足*1　引数のOBCP名にはツールキットを使用する装置により以下を指定してください。

obcp01　（IRCSの場合）	obcp17　（OTHER17の場合）
obcp02　（AOの場合）	obcp18　（OTHER18の場合）
obcp03　（CIAOの場合）	obcp19　（OTHER19の場合）
obcp04　（OHSの場合）	obcp20　（OTHER20の場合）
obcp05　（FOCASの場合）	obcp21　（OTHER21の場合）
obcp06　（HDSの場合）	obcp22　（OTHER22の場合）
obcp07　（COMICSの場合）	obcp23　（OTHER23の場合）
obcp08　（SPCAMの場合）	obcp24　（OTHER24の場合）
obcp09　（SUKAの場合）	obcp25　（OTHER25の場合）
obcp10　（MIRTOSの場合）	obcp26　（OTHER26の場合）
obcp11　（VTOSの場合）	obcp27　（OTHER27の場合）
obcp12　（CACの場合）	obcp28　（OTHER28の場合）
obcp13　（SKYMONの場合）	obcp29　（OTHER29の場合）
obcp14　（PI1の場合）	obcp30　（OTHER30の場合）
obcp15　（PI2の場合）	obcp31　（OTHER31の場合）
obcp16　（OTHER16の場合）	obcp32　（OTHER32の場合）
obcp99　（WORKの場合）	
vgw　（VGWの場合）	

3.2 ツールキット環境変数の設定

1)表3.2の環境変数を設定してください。

表3.2　環境変数

環境変数	内容
DAQTKHOME	ツールキットディレクトリパス名(.../daqtk)
MAX_BYTE_RPC_LOG	RPC送受信ログ最大バイト数 （非設定時は１メガバイト)
MAX_GNRT_RPC_LOG	RPC送受信ログ世代管理数
MAX_BYTE_ERR_LOG	エラーログ最大バイト数（非設定時は世代管理なし）
DAQTK_IPCKEY_UP	IPCキーベースアップ値（非設定時は0）
FITS_KEYWD_INF	FITSキーワード設定ファイルパス名
WCS_COMMENT_INF	WCSコメントファイルパス名
CMDOBCP	自ホスト名（コマンド通信機能） *1
CMDOBS	通信先ホスト名（コマンド通信機能） *1
THROUGHOBCP	自ホスト名（透過モード・コマンド通信機能） *1
THROUGHOBS	通信先ホスト名(透過モード･コマンド通信機能）*1
STATOBCP	自ホスト名（ステータス転送機能） *1
STATOBS	通信先ホスト名（ステータス転送機能） *1
DATOBCP	自ホスト名（取得データ転送機能） *1
DATOBC	通信先ホスト名（取得データ転送機能） *1
GETOBCP	自ホスト名（ステータスデータ取得機能） *1
GETOBS	通信先ホスト名（ステータスデータ取得機能） *1
OBCPNO	透過モード利用時の利用者OBCP番号 （例：OBCP1,OBCP2, ... ,OBCP32 のいずれか）

補足　*1）ツールキットを利用する（すばる観測制御システムに接続する）ホスト名は
８文字以内である必要があります.
これはRPC電文ヘッダ部フォーマットの制約があるためです.
８文字を超えた設定を行うと，通信を行うことができません.

2)環境設定の例を以下に示します。

```
setenv DAQTKHOME                    /home/subaru/daq/daqtk
setenv FITS_KEYWD_INF \
/home/subaru/daq/daqtk/sample/src/makefits/prime.txt
setenv WCS_COMMENT_INF \
/home/subaru/daq/daqtk/sample/src/makefits/WcsComment.txt
（ツールキットインストール先ディレクトリが "/home/subaru/daq" の場合）
setenv MAX_BYTE_RPC_LOG             1000000
setenv MAX_GNRT_RPC_LOG             5
setenv MAX_BYTE_ERR_LOG             1000000
setenv DAQTK_IPCKEY_UP              0
setenv CMDOBCP                      OBCP
setenv CMDOBS                       OBS
setenv THROUGHOBCP                  OBCP
setenv THROUGHOBS                   OBS
setenv STATOBCP                     OBCP
setenv STATOBS                      OBS
setenv DATOBCP                      OBCP-EF
setenv DATOBC                       OBC-EF
setenv GETOBCP                      OBCP
setenv GETOBS                       OBS
setenv OBCPNO                       OBCP1
                                    （利用者のOBCP番号が1の場合）
```

3.3 ツールキットシェル変数の設定

1)以下のツールキット実行ファイルディレクトリを、シェル変数"path"（サーチパス）に追加してください。

```
.../daqtk/bin
```

3.4 システムファイルの設定

1) rootでログインします。

2) /etc/system でIPC資源の制限を設定します。
ツールキットで使用しているIPC資源を以下に示します。

表3.4.1　メッセージ

資源	値
メッセージキューの数	11
１メッセージの最大長	20 byte
１メッセージキューにおける待機状態メッセージ最大数	64

表3.4.2　共有メモリ

資源	値
共有メモリの数	13
１共有メモリの最大値	103712 byte

表3.4.3　セマフォ

資源	値
セマフォの数	5
セマフォの最大値	1

利用者側で適切な値を設定してください。
当方では、以下の環境で動作を確認しております。

```
set msgsys:msginfo_msgmni=512
set msgsys:msginfo_msgtql=512
set shmsys:shminfo_shmmax=10000000
```

3)システムを再起動（リブート）します。

3.5 SAOtngのインストール

モニタ表示ツールを使用する場合はSAOtngのインストールを行って下さい.
モニタ表示ツールを使用しない場合は必要ありません.

1)利用者側が、SAOtngの配布ファイル(saord-1.5.tar.Z)を入手してください。
（ftp://sao-ftp.harvard.edu/pub/rd/saord_1.5.tar.Zがftp入手先です）

2)展開したファイルをインストール先ディレクトリに移します。

```
% mv saord-1.5.tar.Z ...
```

(...はツールキットのインストール先ディレクトリと同じである必要はありません)

3)圧縮ファイルを解凍し、tarにより展開します。

```
% cd ...
% uncompress saord-1.5.tar.Z
% tar xvfo saord-1.5.tar
```

4)以下のファイルの内容に従い、SAOtngをインストールしてください。

```
.../saord/BUILD
```

ASSIST等のプログラムも同時にインストール可能ですが、モニタ表示ツールが動作するためには、SAOtngさえあれば問題ありません。

5)以下のSAOtngの実行ファイルディレクトリを、シェル変数"path"（サーチパス）に追加してください。

```
.../saord/bin
```

4. 利用手順

4.1 ツールキットライブラリのリンク方法

1)ツールキットの各ライブラリを呼び出すには，以下のインクルードファイルを
インクルードして下さい.

```
#include "DAQtkUser.h"
```

2)呼び出し形式については、本使用手引書の５章、および、表4.1.1のサンプル
プログラムをご参照ください。

3)リンクする際に、以下のライブラリファイルもオブジェクトとして指定してください。

```
.../daqtk/lib/*.a
```

(...はツールキットのインストール先のディレクトリを指します)
リンク方法については、サンプルプログラムのMakefileも御参照ください。

表4.1.1　サンプルプログラム

ディレクトリ	内容
.../daqtk/sample/src	サンプルプログラムソースファイル格納用
├　cmd/	コマンド通信機能サンプル
│　├　init/	初期化処理関数
│　├　exit/	終了処理関数
│　├　rcv/	制御コマンド受信関数
│　├　snd/	処理完了送信関数
│　└　check/	コマンド受信チェック関数
├　through/	透過モード・コマンド通信機能サンプル
│　├　init/	初期化処理関数
│　├　exit/	終了処理関数
│　├　snd/	制御コマンド送信関数
│　└　rcv/	処理完了受信関数
├　stat/	ステータス転送機能サンプル
│　├　init/	初期化処理関数
│　├　exit/	終了処理関数
│　└　snd/	ステータス転送関数
├　dat/	取得データ転送機能サンプル
│　├　init/	初期化処理関数
│　├　exit/	終了処理関数
│　└　snd/	取得データ転送関数
	FITSデータ生成／転送関数
├　get/	ステータスデータ取得機能サンプル
│　├　init/	初期化処理関数
│　├　exit/	終了処理関数
│　├　snd/	ステータスデータ要求送信関数
│　└　rcv/	ステータスデータ完了受信関数
├　makefits/	FITSファイル作成ツールサンプル
├　func/	FITSキーワード値算出機能サンプル
└　monitor/	モニタ表示ツールサンプル

4.2 FITSキーワード設定ファイルの編集

1)FITSファイル作成ツールを使用する場合、環境変数"FITS_KEYWD_INF"で指定した、FITSキーワード設定ファイルを編集してください。
FITSキーワード設定ファイルとは、入力パラメータであるプライマリFITSヘッダ項目をチェックするため、FITSキーワードの属性を定義したファイルです。

2)本ファイルはASCII形式です。１レコードは改行で区切られており、改行コードを含めて80文字固定とします。レコードのフォーマットは規定されていますが、ファイルは不定長とします。
１レコードごとに１つのFITSキーワードに関する以下の属性が設定されます。

- ・格納順
- ・必須
- ・一意
- ・固定
- ・デフォルト値　　固定キーワードの場合のみ有効

3)FITSキーワード設定ファイルのレコードフォーマットを次頁に示します。

表4.2 FITSキーワード設定ファイル レコードフォーマット

No	項目名	オフセット(バイト)	サイズ	値	概要
1	FITSキーワード	0	8	"^\S+"	設定対象FITSキーワード
2	デリミタ	8	2	"= "	デリミタ
3	属性(格納順)	10	3	"-1"/"0"/"1"～	"1"～ : 格納順 "-1" : 終端 "0" : 非設定
4	属性(必須)	13	3	"0"/"1"	"1" : 設定 "0" : 非設定
5	属性(一意)	16	3	"0"/"1"	"1" : 設定 "0" : 非設定
6	属性(固定)	19	3	"0"/"1"	"1" : 設定 "0" : 非設定
7	デフォルト値	22	57	"(.*)\s*$"	固定キーワードのデフォルト値 *1
8	改行	79	1	"\n"	改行コード
計			80		

補足*1 属性(固定)が設定である場合のみ有効です。
また、空白詰めを行ってください。

4)FITSキーワード設定ファイルフォーマット例、およびFITSファイル作成ライブラリ関数のプライマリFITSヘッダ項目に関する入力パラメータと出力結果を次頁以降に示します。

・FITSキーワード設定ファイルフォーマット例

```
←                                  80byte                                  →
01234567890123456789012345678901234567890123456789.....0123456789
SIMPLE  = 1  1  1  0                                                       \n
BITPIX  = 2  1  1  0                                                       \n
OBJECT  = 0  1  0  0                                                       \n
WEATHER = 0  0  1  0                                                       \n
TELESCOP= 0  0  0  1  SUBARU                                               \n
END     = -1 1  1  0                                                       \n
```

以下のファイルも参考にしてください。

.../daqtk/sample/src/makefits/prime.txt

```
...
int    iRslt;
int    iAsciiFlg;
int    iWcsFlg;
int    iSpecFlg;
char*  pFitsName;
struct TFitsHeader  SPrimeHeader;
struct TfitsHeader  SAsciiHeader;
struct TWcsPara     SWcsPara;
struct Tunit        SPrimeDataUnit;
struct Tunit        SAsciiDataUnit;
...
struct TfitsCard    SFitsCard[64];
struct TFitsCard*   qFitsCard;
...
qFitsCard = SFitsCard;
strcpy( (qFitsCard+0)->cKeywd, "SIMPLE" );
strcpy( (qFitsCard+0)->cValue, "T" );
strcpy( (qFitsCard+0)->cComment, "is this fits format" );
strcpy( (qFitsCard+1)->cKeywd, "BITPIX" );
strcpy( (qFitsCard+1)->cValue, "16" );
strcpy( (qFitsCard+1)->cComment, "bit per pixel" );
strcpy( (qFitsCard+2)->cKeywd, "BSCALE" );
strcpy( (qFitsCard+2)->cValue, "1.0E2" );
strcpy( (qFitsCard+2)->cComment, "pixel data scale" );
strcpy( (qFitsCard+3)->cKeywd, "OBJECT" );
strcpy( (qFitsCard+3)->cValue, "'test object'" );
strcpy( (qFitsCard+3)->cComment, "test object" );
strcpy( (qFitsCard+4)->cKeywd, "WEATHER" );
strcpy( (qFitsCard+4)->cValue, "'cloudy'" );
strcpy( (qFitsCard+4)->cComment, "weather at observation" );
...
```

```
...
strcpy( (qFitsCard+5)->cKeywd, "COMMENT" );
strcpy( (qFitsCard+5)->cValue, "test for daqtk" );
strcpy( (qFitsCard+6)->cKeywd, "END" );
...
SPrimeHeader.qFitsCard = qFitsCard;
iRet = DAQtkFitsMakeFile( iAsciiFlg, iWcsFlg, iSpecFlg, pFitsName,
                                &SPrimeHeader, &SAsciiHeader,
                                &SWcsPara, &SPrimeDataUnit, &SAsciiDataUnit,
                                iRslt );
...
```

・FITSファイル出力結果

以下のコマンドで、作成したファイルの内容を確認できます。

```
> fold -80 (FITSファイル名)
```

以下に、出力ファイルのプライマリヘッダ部を示します。

```
←                                  80byte                                  →
01234567890123456789012345678901234567890123456789.....0123456789
SIMPLE  =                    T / is this fits format
BITPIX  =                   16 / bit per pixel
BSCALE  =                1.0E2 / pixel data scale
OBJECT  = 'test object'        / test object
WEATHER = 'cloudy'             / weather at observation
COMMENT test for daqtk
TELESCOP=               SUBARU /
END
```

以下2880byteまで空白文字が格納されています。

4.3 WCSコメントファイルの編集

1)WCSキーワードにコメントを挿入する場合、環境変数"WCS_COMMENT_INF"で指定した、WCSコメントファイルを編集してください。
WCSコメントファイルとは、WCSキーワードのコメント欄に挿入するコメントを定義したファイルです。

2)本ファイルはASCII形式です。１レコードは改行で区切られています。
レコードのフォーマットは規定されていますが、ファイルは不定長とします。

3)FITSキーワード設定ファイルのレコードフォーマットを次頁に示します。

表4.3 WCSコメントファイル レコードフォーマット

No	項目名	オフセット (バイト)	サイズ	値	概要
1	WCSキーワード	0	8	"^\S+"	設定対象FITSキーワード
2	デリミタ	8	2	"/ "	デリミタ
3	コメント文	10	可変長	"(.*)$"	WCSキーワードのコメント
4	改行	不定	1	"\n"	改行コード

4)WCSコメントファイルフォーマット例、およびFITSファイル作成ライブラリ
関数のWCSキーワードに関する出力結果を次頁以降に示します。

・WCSコメントファイルフォーマット例

```
01234567890123456789012345678901234567890123456789.....
EQUINOX / Standard FK5 (years)\n
RADECSYS/ The equatorial coordinate system\n
WCS-ORIG/ Origin of the WCS value\n
CRPIX1  / Reference pixel in X (pixel)\n
CRPIX2  / Reference pixel in Y (pixel)\n
CRVAL1  / Physical value of the reference pixel X\n
CRVAL2  / Physical value of the reference pixel Y\n
CDELT1  / X Scale projected on detector (#/pix)\n
CDELT2  / Y Scale projected on detector (#/pix)\n
CTYPE1  / Pixel coordinate system\n
CTYPE2  / Pixel coordinate system\n
CUNIT1  / Units used in both CRVAL1 and CDELT1\n
CUNIT2  / Units used in both CRVAL2 and CDELT2\n
LONGPOLE/ The North Pole of standard system (deg)\n
PC001001/ Pixel Coordinate translation matrix\n
PC001002/ Pixel Coordinate translation matrix\n
PC002001/ Pixel Coordinate translation matrix\n
PC002002/ Pixel Coordinate translation matrix\n
CD1_1   / Pixel Coordinate translation matrix\n
CD1_2   / Pixel Coordinate translation matrix\n
CD2_1   / Pixel Coordinate translation matrix\n
CD2_2   / Pixel Coordinate translation matrix\n
```

以下のファイルも参考にしてください。

.../daqtk/sample/src/makefits/WcsComment.txt

・FITSファイル出力結果

以下に、出力ファイルのWCSキーワード部を示します。

```
←                                  80byte                                  →
01234567890123456789012345678901234567890123456789.....0123456789
EQUINOX =               2000.0 / Standard FK5 (years)
CRVAL1  =          41.50991500 / Physical value of the reference pixel X
CRVAL2  =          41.50991500 / Physical value of the reference pixel Y
CRPIX1  =                256.0 / Reference pixel in X (pixel)
CRPIX2  =                256.0 / Reference pixel in Y (pixel)
CDELT1  =          -0.00013889 / X Scale projected on detector (#/pix)
CDELT2  =           0.00019444 / Y Scale projected on detector (#/pix)
PC001001=           1.15470054 / Pixel Coordinate translation matrix
PC001002=           0.00000000 / Pixel Coordinate translation matrix
PC002001=          -0.57735027 / Pixel Coordinate translation matrix
PC002002=           1.00000000 / Pixel Coordinate translation matrix
LONGPOLE=            180.00000 / The North Pole of standard system (deg)
CTYPE1  = 'RA---TAN'           / Pixel coordinate system
CTYPE2  = 'DEC--TAN'           / Pixel coordinate system
CUNIT1  = 'degree  '           / Units used in both CRVAL1 and CDELT1
CUNIT2  = 'degree  '           / Units used in both CRVAL2 and CDELT2
WCS-ORIG= 'SUBARU Toolkit'     / Origin of the WCS value
RADECSYS= 'FK5     '           / The equatorial coordinate system
CD1_1   =          -0.00016038 / Pixel Coordinate translation matrix
CD1_2   =          -0.00000000 / Pixel Coordinate translation matrix
CD2_1   =          -0.00011226 / Pixel Coordinate translation matrix
CD2_2   =           0.00019444 / Pixel Coordinate translation matrix
```

4.4 STARS登録チェックツールについて

1)本ツールキットには，装置開発者が作成したFITSファイルがSTARSに登録可能であるか否かのチェックを行う「STARS登録チェックツール」が含まれています.
「STARS登録チェックツール」の利用手順については，「9. STARS登録チェックツール」をご参照ください.

5. 呼び出し形式

ツールキットライブラリの呼び出し形式を示します。

5.1 コマンド通信機能

5.1.1 コマンド通信初期化処理

初期化処理の呼び出し形式を表5.1.1に示します。

表5.1.1 初期化処理呼び出し形式

項目	内容	備考
機能大分類	ツールキット・ライブラリ	
機能分類	通信ツール	
和名	初期化処理	
英名	DAQtkCmdInit	
概要	コマンド通信機能で使用する資源の初期化を行う。	
呼出形式	int DAQtkCmdInit(void)	
インクルード	#include "DAQtkUser.h"	
関数値	・DAQ_TK_OK ：正常終了 ・DAQ_TK_ERR ：異常終了	
入力パラメータ	なし	
出力パラメータ	なし	

5.1.2　コマンド通信終了処理

終了処理の呼び出し形式を表5.１.2に示します。

表5.１.2　終了処理呼び出し形式

項目	内容	備考
機能大分類	ツールキット・ライブラリ	
機能分類	通信ツール	
和名	終了処理	
英名	DAQtkCmdExit	
概要	コマンド通信機能で使用する資源の解放を行う。	
呼出形式	int DAQtkCmdExit(void)	
インクルード	#include　"DAQtkUser.h"	
関数値	・DAQ_TK_OK　：正常終了 ・DAQ_TK_ERR　：異常終了	
入力パラメータ	なし	
出力パラメータ	なし	

5.1.3　制御コマンド受信

制御コマンド受信の呼び出し形式を表5.１.3に示します。

表5.１.3　制御コマンド受信呼び出し形式(1/2)

項目	内容	備考
機能大分類	ツールキット・ライブラリ	
機能分類	通信ツール	
和名	制御コマンド受信	
英名	DAQtkCmdRcv	
概要	制御コマンドを受信し、受付応答を送信する	
呼出形式	int DAQtkCmdRcv(　　struct TRpcCmd* qRpcCmd 　　int* pRslt)	 out　＊1 out
インクルード	#include　　"DAQtkUser.h"	
関数値	・DAQ_TK_OK　　：正常終了 ・DAQ_TK_ERR　　：異常終了	
入力パラメータ	なし	

表5.１.3　制御コマンド受信呼び出し形式(2/2)

項目		内容	備考
出力パラメータ	・qRpcCmd	：制御コマンド	＊２
	・pRslt	：詳細コード（正常終了時） ・正常= DAQ_TK_OK_NONE 詳細コード（異常終了時） ・制御コマンド受信異常= DAQ_TK_ERR_CMND ・受付応答送信異常= DAQ_TK_ERR_ACK ・その他異常= DAQ_TK_ERR_ETC	

補足　＊１　TRpcCmd は構造体タグ名で、メンバは以下のとおり

```
struct TRpcCmd {
        char cCmdNo[DAQ_TK_SIZ_CMDNO];
                        ：受信した制御コマンドのシーケンシャル番号
        char cCmdCode[DAQ_TK_SIZ_CMDCODE];
                        ：受信した制御コマンド
};
#define DAQ_TK_SIZ_CMDNO 9
#define DAQ_TK_SIZ_CMDCODE 2049
```

＊２　正常終了時のみ有効

5.1.4　処理完了送信

処理完了送信の呼び出し形式を表5.１.4に示します。

表5.１.4　処理完了送信呼び出し形式(1/2)

項目	内容	備考
機能大分類	ツールキット・ライブラリ	
機能分類	通信ツール	
和名	処理完了送信	
英名	DAQtkCmdSnd	
概要	処理完了を送信する	
呼出形式	int DAQtkCmdSnd(　　struct TRpcRslt* qRpcRslt, 　　int* pRslt)	 in　＊1 out
インクルード	#include　　"DAQtkUser.h"	
関数値	・DAQ_TK_OK　　：正常終了 ・DAQ_TK_ERR　　：異常終了	
入力パラメータ	・qRpcRslt　：処理完了送信情報	

表5.１.4　処理完了送信呼び出し形式(2/2)

項目		内容	備考
出力パラメータ	・piRslt	：詳細コード（正常終了時） ・正常= DAQ_TK_OK_NONE 詳細コード（異常終了時） ・処理完了送信異常= DAQ_TK_ERR_RPLY ・その他異常= DAQ_TK_ERR_ETC	

補足　＊１　TRpcRslt は構造体タグ名で、メンバは以下のとおり

```
struct TRpcRslt {
        char cCmdNo[DAQ_TK_SIZ_CMDNO];
                        ：受信した制御コマンドのシーケンシャル番号
        char cRslt[DAQ_TK_SIZ_RSLT];
                        ：処理結果　正常の場合，”　0”
                                    異常の場合，その他（ユーザが定義）
                                    を格納する.
        char cRsltPara[DAQ_TK_SIZ_RSLTPARA];
                        ：処理結果パラメータ（ユーザが定義）
};
#define DAQ_TK_SIZ_CMDNO 9
#define DAQ_TK_SIZ_RSLT 5
#define DAQ_TK_SIZ_RSLTPARA 8193
```

文字列はNULL止めすること.

5.1.5　制御コマンド受信チェック

制御コマンド受信チェックの呼び出し形式を表5.1.5に示します。

表5.1.5　制御コマンド受信チェック呼び出し形式

項目	内容	備考
機能大分類	ツールキット・ライブラリ	
機能分類	通信ツール	
和名	制御コマンド受信チェック	
英名	DAQtkCmdRcvCheck	
概要	コマンドを受信している数を返す.	
呼出形式	int DAQtkCmdRcvCheck(void)	
インクルード	#include　"DAQtkUser.h"	
関数値	・コマンド受信数　：正常終了 ・DAQ_TK_ERR　：異常終了	
入力パラメータ	なし	
出力パラメータ	なし	

5.2 透過モード・コマンド通信機能

5.2.1 透過モード・コマンド通信初期化処理

初期化処理の呼び出し形式を表5.2.1に示します。

表5.2.1 初期化処理呼び出し形式

項目	内容	備考
機能大分類	ツールキット・ライブラリ	
機能分類	通信ツール	
和名	初期化処理	
英名	DAQtkThroughInit	
概要	透過モード・コマンド通信機能で使用する資源の初期化を行う。	
呼出形式	int DAQtkThroughInit(void)	
インクルード	#include "DAQtkUser.h"	
関数値	・DAQ_TK_OK ：正常終了 ・DAQ_TK_ERR ：異常終了	
入力パラメータ	なし	
出力パラメータ	なし	

5.2.2 透過モード・コマンド通信終了処理

終了処理の呼び出し形式を表5.2.2に示します。

表5.2.2 終了処理呼び出し形式

項目	内容	備考
機能大分類	ツールキット・ライブラリ	
機能分類	通信ツール	
和名	終了処理	
英名	DAQtkThroughExit	
概要	透過モード・コマンド通信機能で使用する資源の解放を行う。	
呼出形式	int DAQtkThroughExit(void)	
インクルード	#include "DAQtkUser.h"	
関数値	・DAQ_TK_OK ：正常終了 ・DAQ_TK_ERR ：異常終了	
入力パラメータ	なし	
出力パラメータ	なし	

5.2.3 制御コマンド送信

制御コマンド送信の呼び出し形式を表5.2.3に示します。

表5.2.3　制御コマンド送信呼び出し形式(1/2)

項目	内容	備考
機能大分類	ツールキット・ライブラリ	
機能分類	通信ツール	
和名	制御コマンド送信	
英名	DAQtkThroughSnd	
概要	制御コマンドを送信し、受付応答を受信する	
呼出形式	int DAQtkThroughSnd(　　struct TRpcCmd* qRpcCmd　　in 　　int* pRslt　　out)	＊1
インクルード	#include　　"DAQtkUser.h"	
関数値	・DAQ_TK_OK　　：正常終了 ・DAQ_TK_ERR　　：異常終了	
入力パラメータ	・qRpcCmd　　：制御コマンド	

表5.2.3　制御コマンド送信呼び出し形式(2/2)

項目		内容	備考
出力パラメータ	・pRslt	：詳細コード（正常終了時） ・正常= 送信した制御コマンドのシーケンシャル番号（ライブラリ内で発行される） 詳細コード（異常終了時） ・制御コマンド送信異常= DAQ_TK_ERR_CMND ・受付応答受信異常= DAQ_TK_ERR_ACK ・受付応答チェック結果異常= DAQ_TK_ERR_LOGIC ・その他異常= DAQ_TK_ERR_ETC	

補足　＊１	TRpcCmd は構造体タグ名で、メンバは以下のとおり
	struct TRpcCmd {
	char cCmdNo[DAQ_TK_SIZ_CMDNO];
	：未使用
	char cCmdCode[DAQ_TK_SIZ_CMDCODE];
	：送信する制御コマンド
	};
	#define DAQ_TK_SIZ_CMDNO 9
	#define DAQ_TK_SIZ_CMDCODE 2049
	文字列はNULL止めすること.		

5.2.4 処理完了受信

処理完了受信の呼び出し形式を表5.2.4に示します。

表5.2.4 処理完了受信呼び出し形式(1/2)

項目	内容	備考
機能大分類	ツールキット・ライブラリ	
機能分類	通信ツール	
和名	処理完了受信	
英名	DAQtkThroughRcv	
概要	処理完了を受信する	
呼出形式	int DAQtkThroughRcv(struct TRpcRslt* qRpcRslt, int* pRslt)	 out ＊1 out
インクルード	#include "DAQtkUser.h"	
関数値	・DAQ_TK_OK ：正常終了 ・DAQ_TK_ERR ：異常終了 ＊バージョン4.0より，出力パラメータの処理結果がエラーの場合，関数値はDAQ_TK_ERRとして返るようになりました.	
入力パラメータ	なし	

表5.2.4　処理完了受信呼び出し形式(2/2)

項目	内容	備考
出力パラメータ	・qRpcRslt　：処理完了 ・piRslt　：詳細コード（正常終了時） ・正常= DAQ_TK_OK_NONE 詳細コード（異常終了時） ・処理完了受信異常= DAQ_TK_ERR_RPLY ・処理完了 シーケンシャル番号異常= DAQ_TK_ERR_SEQ ・その他異常= DAQ_TK_ERR_ETC	

補足　＊１　TRpcRslt は構造体タグ名で、メンバは以下のとおり

```
struct TRpcRslt {
        char cCmdNo[DAQ_TK_SIZ_CMDNO];
                        ：制御コマンド送信時のシーケンシャル番号
        char cRslt[DAQ_TK_SIZ_RSLT];
                        ：処理結果　正常の場合，“　0”
                                    異常の場合，”　1”が格納される
                                    （関数自体もエラー復帰する）
        char cRsltPara[DAQ_TK_SIZ_RSLTPARA];
                        ：処理結果パラメータが格納される.
                          （詳細は８章を参照のこと．）
};
#define DAQ_TK_SIZ_CMDNO 9
#define DAQ_TK_SIZ_RSLT 5
#define DAQ_TK_SIZ_RSLTPARA 8193
```

5.3 ステータス転送機能

5.3.1 ステータス転送初期化処理

初期化処理の呼び出し形式を表5.3.1に示します。

表5.3.1　初期化処理呼び出し形式

項目	内容	備考
機能大分類	ツールキット・ライブラリ	
機能分類	通信ツール	
和名	初期化処理	
英名	DAQtkStatInit	
概要	ステータス転送機能で使用する資源の初期化を行う。	
呼出形式	int DAQtkStatInit(void)	
インクルード	#include "DAQtkUser.h"	
関数値	・DAQ_TK_OK ：正常終了 ・DAQ_TK_ERR ：異常終了	
入力パラメータ	なし	
出力パラメータ	なし	

5.3.2 ステータス転送終了処理

終了処理の呼び出し形式を表5.3.2に示します。

表5.3.2　終了処理呼び出し形式

項目	内容	備考
機能大分類	ツールキット・ライブラリ	
機能分類	通信ツール	
和名	終了処理	
英名	DAQtkStatExit	
概要	ステータス転送機能で使用する資源の解放を行う。	
呼出形式	int DAQtkStatExit(void)	
インクルード	#include "DAQtkUser.h"	
関数値	・DAQ_TK_OK ：正常終了 ・DAQ_TK_ERR ：異常終了	
入力パラメータ	なし	
出力パラメータ	なし	

5.3.3 ステータス転送

ステータス転送の呼び出し形式を表5.3.3に示します。

表5.3.3　ステータス転送呼び出し形式(1/2)

項目	内容	備考
機能大分類	ツールキット・ライブラリ	
機能分類	通信ツール	
和名	ステータス転送	
英名	DAQtkStatSnd	
概要	ステータスデータを送信する	
呼出形式	int DAQtkStatSnd(　　struct TRpcStat* qRpcStat 　　int* pRslt)	 in　＊1 out
インクルード	#include　"DAQtkUser.h"	
関数値	・DAQ_TK_OK　：正常終了 ・DAQ_TK_ERR　：異常終了	
入力パラメータ	・qRpcStat　：ステータス	

表5.3.3　ステータス転送呼び出し形式(2/2)

項目		内容	備考
出力パラメータ	・pRslt	：詳細コード（正常終了時） ・正常= DAQ_TK_OK_NONE 詳細コード（異常終了時） ・異常= DAQ_TK_ERR_ETC	

補足　＊１　TRpcStat は構造体タグ名で、メンバは以下のとおり

```
struct TRpcStat {
        char cStat[DAQ_TK_SIZ_STAT];
                        ：送信するステータスデータ
        int iSize;      ：cStatに設定したステータスデータのバイト数
};
#define DAQ_TK_SIZ_STAT 8193
```

5.4 取得データ転送機能

5.4.1 取得データ転送初期化処理

初期化処理の呼び出し形式を表5.4.1に示します。

表5.4.1　初期化処理呼び出し形式

項目	内容	備考
機能大分類	ツールキット・ライブラリ	
機能分類	通信ツール	
和名	初期化処理	
英名	DAQtkDatInit	
概要	取得データ転送機能で使用する資源の初期化を行う。	
呼出形式	int DAQtkDatInit(void)	
インクルード	#include "DAQtkUser.h"	
関数値	・DAQ_TK_OK ：正常終了 ・DAQ_TK_ERR ：異常終了	
入力パラメータ	なし	
出力パラメータ	なし	

5.4.2 取得データ転送終了処理

終了処理の呼び出し形式を表5.4.2に示します。

表5.4.2 終了処理呼び出し形式

項目	内容	備考
機能大分類	ツールキット・ライブラリ	
機能分類	通信ツール	
和名	終了処理	
英名	DAQtkDatExit	
概要	取得データ転送機能で使用する資源の解放を行う。	
呼出形式	int DAQtkDatExit(void)	
インクルード	#include "DAQtkUser.h"	
関数値	・DAQ_TK_OK ：正常終了 ・DAQ_TK_ERR ：異常終了	
入力パラメータ	なし	
出力パラメータ	なし	

5.4.3 取得データ転送

取得データ転送の呼び出し形式を表5.4.3に示します。

表5.4.3　取得データ転送呼び出し形式(1/2)

項目	内容	備考
機能大分類	ツールキット・ライブラリ	
機能分類	通信ツール	
和名	取得データ転送	
英名	DAQtkDatSnd	
概要	取得データファイルをOBCに転送する	
呼出形式	int DAQtkDatSnd(　　struct TRpcDat* qRpcDat 　　int* pRslt)	 in/out　＊1 out
インクルード	#include　　"DAQtkUser.h"	
関数値	・DAQ_TK_OK　　：正常終了 ・DAQ_TK_ERR　　：異常終了 ＊バージョン4.0より，転送結果がエラーの場合， 関数値はDAQ_TK_ERRとして返るようになりました.	
入出力パラメータ	・qRpcDat　　：取得データファイル情報構造体配列の先頭へのポインタ	

表5.4.3　取得データ転送呼び出し形式(2/2)

項目		内容	備考
出力パラメータ	・pRslt	：詳細コード（正常終了時） ・正常= DAQ_TK_OK_NONE 詳細コード（異常終了時） ・取得データ転送開始送信異常= DAQ_TK_ERR_START ・受付応答受信異常= DAQ_TK_ERR_ACK ・受付応答チェック結果異常= DAQ_TK_ERR_LOGIC ・取得データ転送完了受信異常= DAQ_TK_ERR_END ・その他異常= DAQ_TK_ERR_ETC	

補足　＊１　TRpcDat は構造体タグ名で、メンバは以下のとおり

```
struct TRpcDat {
        char cPath[DAQ_TK_SIZ_DATPATH];
                        ：(in)取得データファイルパス名
        char cFrame[DAQ_TK_SIZ_DATFRAME];
                        ：(in)フレーム番号
        int iResult;  ：(out)エラーコード
                          0     ：正常
                          1     ：FTPするファイルが存在しない
                          2     ：ファイルサイズが異なる
                          3     ：OBCに取得済みのフレーム
                          4     ：システムエラー
};
#define DAQ_TK_SIZ_DATPATH 65
#define DAQ_TK_SIZ_DATFRAME 33
```

・50ファイルまでの複数指定で同時転送が可能であるため，パラメータには上記構造体配列（50まで）の先頭へのポインタを指定する

・50ファイルに満たない場合，配列の終端はメンバの取得データファイルパス名（cPath）をNULL文字列とすること

・データ転送に失敗した場合，ライブラリはエラー復帰するが，各々のファイルにおけるエラー原因がエラーコード（iResult）に格納される

5.5 ステータスデータ取得機能

5.5.1 ステータスデータ取得初期化処理

初期化処理の呼び出し形式を表5.5.1に示します。

表5.5.1　初期化処理呼び出し形式

項目	内容	備考
機能大分類	ツールキット・ライブラリ	
機能分類	ステータスデータ取得ツール	
和名	初期化処理	
英名	DAQtkGetInit	
概要	ステータスデータ取得機能で使用する資源の初期化を行う。	
呼出形式	int DAQtkGetInit(void)	
インクルード	#include "DAQtkUser.h"	
関数値	・DAQ_TK_OK ：正常終了 ・DAQ_TK_ERR ：異常終了	
入力パラメータ	なし	
出力パラメータ	なし	

5.5.2 ステータスデータ取得終了処理

終了処理の呼び出し形式を表5.5.2に示します。

表5.5.2　終了処理呼び出し形式

項目	内容	備考
機能大分類	ツールキット・ライブラリ	
機能分類	ステータスデータ取得ツール	
和名	終了処理	
英名	DAQtkGetExit	
概要	ステータスデータ取得機能で使用する資源の解放を行う。	
呼出形式	int DAQtkGetExit(void)	
インクルード	#include "DAQtkUser.h"	
関数値	・DAQ_TK_OK ：正常終了 ・DAQ_TK_ERR ：異常終了	
入力パラメータ	なし	
出力パラメータ	なし	

5.5.3 ステータスデータ要求送信

ステータスデータ要求送信の呼び出し形式を表5.5.3に示します。

表5.5.3 ステータスデータ要求送信呼び出し形式(1/2)

項目	内容	備考
機能大分類	ツールキット・ライブラリ	
機能分類	ステータスデータ取得ツール	
和名	ステータスデータ要求送信	
英名	DAQtkGetSnd	
概要	ステータスデータ取得要求を送信し、受付応答を受信する	
呼出形式	int DAQtkGetSnd(struct TRpcCmd* qRpcCmd, int* pRslt)	 in ＊1 out
インクルード	#include "DAQtkUser.h"	
関数値	・DAQ_TK_OK ：正常終了 ・DAQ_TK_ERR ：異常終了	
入力パラメータ	・qRpcCmd ：ステータスデータ取得要求	

表5.5.3　ステータスデータ要求送信呼び出し形式(2/2)

項目		内容	備考
出力パラメータ	・pRslt	：詳細コード（正常終了時） ・正常= 送信したステータスデータ取得要求のシーケンシャル番号（ライブラリ内で発行される） 詳細コード（異常終了時） ・ステータスデータ取得要求送信異常= DAQ_TK_ERR_CMND ・受付応答受信異常= DAQ_TK_ERR_ACK ・受付応答チェック結果異常= DAQ_TK_ERR_LOGIC ・その他異常= DAQ_TK_ERR_ETC	

補足　＊１　TRpcCmd は構造体タグ名で、メンバは以下のとおり

```
struct TRpcCmd {
        char cCmdNo[DAQ_TK_SIZ_CMDNO];
                        ：未使用
        char cCmdCode[DAQ_TK_SIZ_CMDCODE];
                        ：送信するステータスデータ取得要求
}
#define DAQ_TK_SIZ_CMDNO 9
#define DAQ_TK_SIZ_CMDCODE 2049
```

文字列はNULL止めすること

5.5.4 ステータスデータ完了受信

ステータスデータ完了受信の呼び出し形式を表5.5.4に示します。

表5.5.4 ステータスデータ完了受信呼び出し形式(1/2)

項目	内容	備考
機能大分類	ツールキット・ライブラリ	
機能分類	ステータスデータ取得ツール	
和名	ステータスデータ完了受信	
英名	DAQtkGetRcv	
概要	ステータスデータ取得完了を受信する	
呼出形式	int DAQtkGetRcv(　　struct TRpcRslt* qRpcRslt, 　　int* pRslt)	 out　＊1 out
インクルード	#include　　"DAQtkUser.h"	
関数値	・DAQ_TK_OK　　：正常終了 ・DAQ_TK_ERR　　：異常終了 ＊バージョン4.0より，出力パラメータの処理結果がエラーの場合，関数値はDAQ_TK_ERRとして返るようになりました.	
入力パラメータ	なし	

表5.5.4　ステータスデータ完了受信呼び出し形式(2/2)

項目	内容	備考
出力パラメータ	・qRpcRslt　：ステータスデータ取得完了 ・pRslt　：詳細コード（正常終了時） ・正常= DAQ_TK_OK_NONE 詳細コード（異常終了時） ・ステータスデータ取得完了受信異常= DAQ_TK_ERR_RPLY ・ステータスデータ取得完了 シーケンシャル番号異常= DAQ_TK_ERR_SEQ ・その他異常= DAQ_TK_ERR_ETC	

補足　＊１　TRpcRslt は構造体タグ名で、メンバは以下のとおり

```
struct TRpcRslt {
        char cCmdNo[DAQ_TK_SIZ_CMDNO];
                        ：ステータスデータ取得要求送信時の
                          シーケンシャル番号
        char cRslt[DAQ_TK_SIZ_RSLT];
                        ：処理結果　正常の場合，“　0”
                                    異常の場合，”　1”が格納される
                                    （関数自体もエラー復帰する）
        char cRsltPara[DAQ_TK_SIZ_RSLTPARA];
                        ：処理結果パラメータ（ステータス値）
                          が格納される.
                          （詳細は８章を参照のこと．）
};
#define DAQ_TK_SIZ_CMDNO 9
#define DAQ_TK_SIZ_RSLT 5
#define DAQ_TK_SIZ_RSLTPARA 8193
```

5.6 FITSファイル作成機能

5.6.1 FITSファイル作成

FITSファイル作成の呼び出し形式を表5.6.1に示します。

表5.6.1 FITSファイル作成呼び出し形式(1/5)

項目	内容	備考
機能大分類	ツールキット・ライブラリ	
機能分類	FITSファイル作成ツール	
和名	FITSファイル作成	
英名	DAQtkFitsMakeFile	
呼出形式	int DAQtkFitsMakeFile(int iAsciiFlg, int iWcsFlg, int iSpecFlg, char* pFitsName, struct TFitsHeader* qPrimeHeader, struct TFitsHeader* qAsciiHeader, struct TWcsPara* qWcsPara, struct TUnit* qPrimeDataUnit, struct TUnit* qAsciiDataUnit, int* pRsltCode)	 in in in in in in in in in out
インクルード	#include "DAQtkUser.h"	

表5.6.1　FITSファイル作成呼び出し形式(2/5)

項目	内容	備考
関数値	・DAQ_TK_OK　：正常終了 ・DAQ_TK_ERR　：異常終了	
入力パラメータ	・iAsciiFlg　：ASCIIextensionフラグ ・ASCIIextensionを付加したFITSファイルを作成する場合 = 1 ・通常のFITSファイルを作成する場合 = 0	
	・iWcsFlg　：WCSフラグ ・WCSキーワードを付加する場合 = 1 ・WCSキーワードを付加しない場合 = 0	
	・iSpecFlg　：分光装置フラグ（WCSフラグが1の場合のみ） ・分光装置用のWCSキーワードを付加する場合 = 1 ・通常のWCSのキーワードを付加する場合 = 0	
	・pFitsName　：出力ファイルパス名	
	・qPrimeHeader　：プライマリヘッダ項目	*1
	・qAsciiHeader　：ASCIIextensionヘッダ項目	*1
	・qWcsPara　：WCSパラメータ	*2
	・qPrimeDataUnit　：プライマリデータ情報	*3
	・qAsciiDataUnit　：extensionデータ情報	*3

表5.6.1　FITSファイル作成呼び出し形式(3/5)

項目	内容	備考
出力パラメータ	・pRsltCode　：詳細コード（正常終了時） ・正常= DAQ_TK_OK_NONE 詳細コード（異常終了時） ・格納順= DAQ_TK_ERR_ORDER ・必須= DAQ_TK_ERR_MUSTBE ・一意= DAQ_TK_ERR_UNIQUE ・フォーマット= DAQ_TK_ERR_FORMAT ・その他異常= DAQ_TK_ERR_ETC	*4
処理概要	・FITSヘッダ項目，およびFITSデータから、FITSファイルを作成する	

補足　＊1	TfitsHeader は構造体タグ名で、メンバは以下のとおり
	struct TFitsHeader {
	　char cPath[SIZ_FITS_FILE_NAME];　未使用
	　int iCardNum;　カード項目数
	　struct TFitsCard* qFitsCard;　カード項目構造体配列へのポインタ
	};
	TfitsCard も構造体タグ名で、メンバは以下のとおり
	struct TFitsCard {
	　char cKeywd[SIZ_FITS_KEYWD];　キーワード
	　char cValue[SIZ_FITS_VALUE];　値
	　char cComment[SIZ_FITS_COMMENT];　コメント
	};
	#define SIZ_FITS_KEYWD 9
	#define SIZ_FITS_VALUE 63
	#define SIZ_FITS_COMMENT 80	

表5.6.1　FITSファイル作成呼び出し形式(4/5)

項目	内容	備考
	・ユーザプロセス側でカード項目構造体配列を宣言して、カード項目を設定し、その先頭アドレスをqFitsCardに格納するものとする ・またそのカード項目数をiCardNumに格納するものとする ・文字列はNULL止めすること ・qPrimeHeaderは必ず設定すること ・iAsciiFlgにONを設定しているときは、qAsciiHeaderを必ず設定すること iAsciiFlgにOFFを設定しているときは、設定の必要はない （qAsciiHeaderにNULLポインタを格納する）	

＊２　TWcsPara は構造体タグ名で、メンバは以下のとおり

```
struct TWcsPara {
        double dRa;         望遠鏡指向方向α（deg）
        double dDec;        望遠鏡指向方向δ（deg）
        double dEquinox;    望遠鏡指向α，δの分点
        double dCrpix_x;    望遠鏡指向方向のCCD上での位置(pixel)
        double dCrpic_y;    望遠鏡指向方向のCCD上での位置(pixel)
        double dCdelt_x;    CCDのX軸スケール(arcsec/pix)
        double dCdelt_y;    CCDのY軸スケール(arcsec/pix)
        double dDeg_x;      X軸から左回りに測った赤経軸の角度(deg)
        double dDeg_y;      Y軸から左回りに測った赤緯軸の角度(deg)
};
```

・iWcsFlgにONを設定しているときは、qWcsParaを必ず設定すること
iWcsFlgにOFFを設定しているときは、設定の必要はない
（qWcsParaにNULLポインタを格納する）

表5.6.1　FITSファイル作成呼び出し形式(4/5)

項目	内容	備考

＊３	Tunit は構造体タグ名で、メンバは以下のとおり
	struct TUnit {
	　　void* pData;　　データへのポインタ
	　　int iSize;　　pDataで指定したデータサイズ（バイト数）
	};
	・上記構造体にデータの先頭ポインタとデータサイズ（バイト数）を
	指定することで、FITSデータをライブラリに通知する
	・qPrimeDataUnitは必ず設定すること
	（プライマリデータが必要ないときは, iSizeに 0を設定すればよい）
	・iAsciiFlgにONを設定しているときは、qAsciiDataUnitを
	必ず設定すること
	iAsciiFlgにOFFを設定しているときは、設定の必要はない
	（qAsciiDataUnitにNULLポインタを格納する）	

＊４	・チェックの際、最初に得たエラーコードを格納する	

5.7 モニタ表示機能

5.7.1 データ表示画面制御

データ表示画面制御の呼び出し形式を表5.7.1に示します。

表5.7.1　データ表示画面制御呼び出し形式(1/2)

項目	内容	備考
機能大分類	ツールキット・ライブラリ	
機能分類	モニタ表示ツール	
和名	データ表示画面制御	
英名	DAQtkMonitorCtl	
呼出形式	int DAQtkMonitorCtl(　　int iKind, 　　int* pRsltCode)	 in out
インクルード	#include　　"DAQtkUser.h"	
関数値	・DAQ_TK_OK　　：正常終了 ・DAQ_TK_ERR　　：異常終了	

表5.7.1　データ表示画面制御呼び出し形式(2/2)

項目	内容	備考
入力パラメータ	・iKind　：処理種別 ・起動= DAQ_TK_MONITOR_CTL_START（=1） ・終了= DAQ_TK_MONITOR_CTL_END（=0）	
出力パラメータ	・pRsltCode　：詳細コード（正常終了時） ・正常= DAQ_TK_OK_NONE 詳細コード（異常終了時） ・画面操作異常= DAQ_TK_ERR_CTL ＊１ ・その他異常= DAQ_TK_ERR_ETC	
処理概要	・データ表示画面を起動、もしくは、終了する	
補足　＊１	データ表示画面が既に起動されている場合に起動、もしくは、終了している場合に終了が指定された場合	

5.7.2 データ表示

データ表示の呼び出し形式を表5.7.2に示します。

表5.7.2　データ表示呼び出し形式(1/2)

項目	内容	備考
機能大分類	ツールキット・ライブラリ	
機能分類	モニタ表示ツール	
和名	データ表示	
英名	DAQtkMonitorDsp	
呼出形式	int DAQtkMonitorDsp(int* iDataSize, int iDataType, void* pDataArea, int* pRsltCode)	 in in in out
インクルード	#include "DAQtkUser.h"	
関数値	・DAQ_TK_OK ：正常終了 ・DAQ_TK_ERR ：異常終了	

表5.7.2　データ表示呼び出し形式(2/2)

項目	内容	備考
入力パラメータ	・iDataSize　：バイナリ配列データ領域配列要素数 ・[0] X軸 ・[1] Y軸	＊1
	・iDataType　：バイナリ配列データデータタイプ ・char= DAQ_TK_MONITOR_DATA_CHAR（=8） ・short int= DAQ_TK_MONITOR_DATA_SHORT（=16） ・int= DAQ_TK_MONITOR_DATA_INT（=32） ・float= DAQ_TK_MONITOR_DATA_FLOAT（=-32） ・double= DAQ_TK_MONITOR_DATA_DOUBLE（=-64）	
	・pDataArea　：バイナリ配列データ領域ポインタ	
出力パラメータ	・pRsltCode　：詳細コード（正常終了時） ・正常= DAQ_TK_OK_NONE 詳細コード（異常終了時） ・画面操作異常= DAQ_TK_ERR_CTL ・その他異常= DAQ_TK_ERR_ETC	
処理概要	・データ表示画面にバイナリ配列データを表示する	
補足　＊1　ライブラリでは配列要素[0],[1]のみ参照するため、ターミネイトは不要		

5.8 FITSキーワード値算出機能

5.8.1 UTC取得

UTC取得の呼び出し形式を表5.8.1に示します.

表5.8.1 UTC取得呼び出し形式(1/2)

項目	内容	備考
機能大分類	ツールキット・ライブラリ	
機能分類	FITSキーワード値算出ツール	
和名	UTC取得	
英名	DAQtkFuncGetUTC	
概要	UTC及びUTC-Dateを獲得する.	
呼出形式	int DAQtkFuncGetUTC (struct TFuncUTC* qFuncUTC out ＊1)	
インクルード	#include "DAQtkUser.h"	
関数値	・DAQ_TK_OK ：正常終了 ・DAQ_TK_ERR ：異常終了	

表5.8.1　UTC取得呼び出し形式(2/2)

項目	内容	備考
入力パラメータ	なし	
出力パラメータ	・qFuncUTC　：UTC,UTC-Date文字列	＊２

補足　＊１	TFuncUTC は構造体タグ名で，メンバは以下のとおり
	struct TFuncUTC {
	　char cUTC[DAQ_TK_FUNC_SIZ_UTC];　UTC文字列
	　char cUTCD[DAQ_TK_FUNC_SIZ_UTCD];　UTC-Date文字列
	};
	#define DAQ_TK_FUNC_SIZ_UTC 13
	#define DAQ_TK_FUNC_SIZ_UTCD 11
	UTCのフォーマットは，"HH:MM:SS.SSS"
	UTC-Dateのフォーマットは，"yyyy-mm-dd"	

＊２	正常終了時のみ有効	

5.8.2　LST/MJD取得

LST/MJD取得の呼び出し形式を表5.8.2に示します.

表5.8.2　LST/MJD取得呼び出し形式(1/2)

項目	内容	備考
機能大分類	ツールキット・ライブラリ	
機能分類	FITSキーワード値算出ツール	
和名	LST/MJD取得	
英名	DAQtkFuncGetLST	
概要	LST,及び,MJDを獲得する.	
呼出形式	int DAQtkFuncGetLST (　　struct TFuncUTC* qFuncUTC, 　　char* pUT1, 　　struct TFuncOBS* qFuncOBS, 　　struct TFuncLST* qFuncLST)	 in　＊1 in in　＊2 out　＊3
インクルード	#include　"DAQtkUser.h"	
関数値	・DAQ_TK_OK　：正常終了 ・DAQ_TK_ERR　：異常終了	

表5.8.2　LST/MJD取得呼び出し形式(2/2)

項目	内容	備考
入力パラメータ	・qFuncUTC　：UTC文字列 ・pUT1　：補正項UT1-UTC ・qFuncOBS　：観測地点の経度，緯度	
出力パラメータ	・qFuncLST　：LST文字列	＊４
補足　＊１	TFuncUTC は 表5.8.1 補足 ＊１を参照	

＊２	TFuncOBS は構造体タグ名で，メンバは以下のとおり
	struct TFuncOBS {
	　　char cLongitude[DAQ_TK_FUNC_SIZ_LONGI];　　経度の文字列
	　　char cLatitude[DAQ_TK_FUNC_SIZ_LATI];　　緯度の文字列
	};
	#define DAQ_TK_FUNC_SIZ_LONGI 15
	#define DAQ_TK_FUNC_SIZ_LATI 14
	経度のフォーマットは，"+DDD:MM:SS.SSS"，East+，West-
	緯度のフォーマットは，"+DD:MM:SS.SSS"，North+，South-	

＊３	TFuncLST は構造体タグ名で，メンバは以下のとおり
	struct TFuncLST {
	　　char cLST[DAQ_TK_FUNC_SIZ_LST];　　LST文字列
	　　char cMJD[DAQ_TK_FUNC_SIZ_MJD];　　MJD文字列
	};
	#define DAQ_TK_FUNC_SIZ_LST 13
	#define DAQ_TK_FUNC_SIZ_MJD 15
	LSTのフォーマットは，"HH:MM:SS.SSS"
	MJDのフォーマットは，"DDDDD.DDDDDDDD"	

＊４	正常終了時のみ有効	

5.8.3 分点変換

分点変換の呼び出し形式を表5.8.3に示します.

表5.8.3　分点変換呼び出し形式(1/2)

項目	内容	備考
機能大分類	ツールキット・ライブラリ	
機能分類	FITSキーワード値算出ツール	
和名	分点変換	
英名	DAQtkFuncChgEqx	
概要	任意の分点の天球座標から任意の分点の天球座標へ変換する.	
呼出形式	int DAQtkFuncChgEqx (struct TFuncCE* qFuncCDO, struct TFuncCE* qFuncCDN)	 in ＊1 in/out ＊1
インクルード	#include "DAQtkUser.h"	
関数値	・DAQ_TK_OK ：正常終了 ・DAQ_TK_ERR ：異常終了	

表5.8.3　分点変換呼び出し形式(2/2)

項目	内容		備考
入力パラメータ	・qFuncCDO	：入力天球座標	
	・qFuncCDN->cEquinox	：出力天球座標（分点）	
出力パラメータ	・qFuncCDN->cRA	：出力天球座標（赤経）	＊２
	・qFuncCDN->cDEC	：出力天球座標（赤緯）	＊２

補足　＊１　TFuncCE は構造体タグ名で，メンバは以下のとおり

```
struct TFuncCE {
        char cEquinox[DAQ_TK_FUNC_SIZ_EQUINOX];      分点
        char cRA[DAQ_TK_FUNC_SIZ_RA];                赤経の文字列
        char cDEC[DAQ_TK_FUNC_SIZ_DEC];              赤緯の文字列
};
#define DAQ_TK_FUNC_SIZ_RA 13
#define DAQ_TK_FUNC_SIZ_DEC 13
```

赤経のフォーマットは，"HH:MM:SS.SSS"

赤緯のフォーマットは，"+DD:MM:SS.SS"

＊２　正常終了時のみ有効

5.8.4　Pixel座標－天球座標変換

Pixel座標－天球座標変換の呼び出し形式を表5.8.4に示します.

表5.8.4　Pixel座標－天球座標変換呼び出し形式(1/3)

項目	内容	備考
機能大分類	ツールキット・ライブラリ	
機能分類	FITSキーワード値算出ツール	
和名	Pixel座標－天球座標変換	
英名	DAQtkFuncGetRADEC	
概要	WCS，及び，CCD上でのpixel位置から，天体の天球座標を求める.	
呼出形式	int DAQtkFuncGetRADEC (struct TFuncWCS* qFuncWCS, in ＊1 struct TFuncCCD* qFuncCCD, in ＊2 struct TFuncCE* qFuncRADEC out ＊3)	
インクルード	#include "DAQtkUser.h"	
関数値	・DAQ_TK_OK ：正常終了 ・DAQ_TK_ERR ：異常終了	
入力パラメータ	・qFuncWCS ：WCSパラメータ ・qFuncCCD ：CCD上でのpixel位置	
出力パラメータ	・qFuncRADEC ：出力天球座標	＊4

表5.8.4　Pixel座標－天球座標変換呼び出し形式(2/3)

項目	内容	備考

補足　＊１	TFuncWCS は構造体タグ名で，メンバは以下のとおり
	struct TFuncWCS {
	char cEquinox[DAQ_TK_FUNC_SIZ_FITS];
	望遠鏡指向α，δの分点
	char cCrval1[DAQ_TK_FUNC_SIZ_FITS];
	望遠鏡指向方向α（deg）
	char cCrval2[DAQ_TK_FUNC_SIZ_FITS];
	望遠鏡指向方向δ（deg）
	char cCrpix1[DAQ_TK_FUNC_SIZ_FITS];
	望遠鏡指向方向のCCD上での位置(pixel)
	char cCrpix2[DAQ_TK_FUNC_SIZ_FITS];
	望遠鏡指向方向のCCD上での位置(pixel)
	char cCdelt1[DAQ_TK_FUNC_SIZ_FITS];
	CCDのX軸スケール(arcsec/pix)
	char cCdelt2[DAQ_TK_FUNC_SIZ_FITS];
	CCDのY軸スケール(arcsec/pix)
	char cPC001001[DAQ_TK_FUNC_SIZ_FITS];
	座標変換行列要素[1,1](-)
	char cPC001002[DAQ_TK_FUNC_SIZ_FITS];
	座標変換行列要素[1,2](-)
	char cPC002001[DAQ_TK_FUNC_SIZ_FITS];
	座標変換行列要素[2,1](-)
	char cPC002002[DAQ_TK_FUNC_SIZ_FITS];
	座標変換行列要素[2,2](-)
	char cLongpole[DAQ_TK_FUNC_SIZ_FITS];
	座標変換係数
	};	

表5.8.4　Pixel座標－天球座標変換呼び出し形式(3/3)

項目	内容	備考
	#define DAQ_TK_FUNC_SIZ_FITS 21 各メンバに格納する文字列のフォーマットはすばるFITSヘッダ統一案に則る.	

＊２	TFuncCCD は構造体タグ名で，メンバは以下のとおり
	struct TFuncCCD {
	　　double dX;　　pixel座標 X (pixel)
	　　double dY;　　pixel座標 Y (pixel)
	};	

＊３	TFuncCE は 表5.8.3 補足 ＊１を参照	
＊４	正常終了時のみ有効	

5.8.5 時刻変換（文字列－数値）

時刻変換（文字列－数値）の呼び出し形式を表5.8.5に示します.

表5.8.5 時刻変換（文字列－数値）呼び出し形式

項目	内容	備考
機能大分類	ツールキット・ライブラリ	
機能分類	FITSキーワード値算出ツール	
和名	時刻変換（文字列－数値）	
英名	DAQtkFuncCTime2DHour	
概要	時刻の文字列を数値へ変換する.	
呼出形式	double DAQtkFuncCTime2DHour (char* pTime)	in
インクルード	#include "DAQtkUser.h"	
関数値	・時刻（hour）	
入力パラメータ	・pTime ：時刻文字列, フォーマットは, "HH:MM:SS.SSS"	

5.8.6 時刻変換（数値－文字列）

時刻変換（数値－文字列）の呼び出し形式を表5.8.6に示します.

表5.8.6 時刻変換（数値－文字列）呼び出し形式

項目	内容	備考
機能大分類	ツールキット・ライブラリ	
機能分類	FITSキーワード値算出ツール	
和名	時刻変換（数値－文字列）	
英名	DAQtkFuncDHour2CTime	
概要	時刻の数値を文字列へ変換する.	
呼出形式	char* DAQtkFuncDHour2CTime (double dHour in)	
インクルード	#include "DAQtkUser.h"	
関数値	・時刻文字列. フォーマットは, "HH:MM:SS.SSS"	＊1
入力パラメータ	・dHour ：時刻（hour）	
補足 ＊1	文字列領域は関数の中で静的に確保されるので, 本関数を２度続けて呼び出すと領域が破壊される.	

5.8.7　角度変換（赤経文字列－数値）

角度変換（赤経文字列－数値）の呼び出し形式を表5.8.7に示します.

表5.8.7　角度変換（赤経文字列－数値）呼び出し形式

項目	内容	備考
機能大分類	ツールキット・ライブラリ	
機能分類	FITSキーワード値算出ツール	
和名	角度変換（赤経文字列－数値）	
英名	DAQtkFuncCRA2DDeg	
概要	赤経文字列を数値へ変換する.	
呼出形式	double DAQtkFuncCRA2DDeg (char* pRA)	in
インクルード	#include "DAQtkUser.h"	
関数値	・赤経（degree）	
入力パラメータ	・pRA ：赤経文字列，フォーマットは，"HH:MM:SS.SSS"	

5.8.8 角度変換（数値－赤経文字列）

角度変換（数値－赤経文字列）の呼び出し形式を表5.8.8に示します.

表5.8.8 角度変換（数値－赤経文字列）呼び出し形式

項目	内容	備考
機能大分類	ツールキット・ライブラリ	
機能分類	FITSキーワード値算出ツール	
和名	角度変換（数値－赤経文字列）	
英名	DAQtkFuncDDeg2CRA	
概要	数値を赤経文字列へ変換する.	
呼出形式	char* DAQtkFuncDDeg2CRA (double dDeg in)	
インクルード	#include "DAQtkUser.h"	
関数値	・赤経文字列. フォーマットは, "HH:MM:SS.SSS"	＊1
入力パラメータ	・dDeg ：赤経（degree）	
補足 ＊1	文字列領域は関数の中で静的に確保されるので, 本関数を２度続けて呼び出すと領域が破壊される.	

5.8.9 角度変換（赤緯文字列－数値）

角度変換（赤緯文字列－数値）の呼び出し形式を表5.8.9に示します.

表5.8.9　角度変換（赤緯文字列－数値）呼び出し形式

項目	内容	備考
機能大分類	ツールキット・ライブラリ	
機能分類	FITSキーワード値算出ツール	
和名	角度変換（赤緯文字列－数値）	
英名	DAQtkFuncCDec2DDeg	
概要	赤緯文字列を数値へ変換する.	
呼出形式	double DAQtkFuncCDec2DDeg (char* pDec)	in
インクルード	#include "DAQtkUser.h"	
関数値	・赤緯（degree）	
入力パラメータ	・pDec ：赤緯文字列. フォーマットは, "+DD:MM:SS.SS"	

5.8.10　角度変換（数値－赤緯文字列）

角度変換（数値－赤緯文字列）の呼び出し形式を表5.8.10に示します.

表5.8.10　角度変換（数値－赤緯文字列）呼び出し形式

項目	内容	備考
機能大分類	ツールキット・ライブラリ	
機能分類	FITSキーワード値算出ツール	
和名	角度変換（数値－赤緯文字列）	
英名	DAQtkFuncDDeg2CDec	
概要	数値を赤緯文字列へ変換する.	
呼出形式	char* DAQtkFuncDDeg2CDec (　　double dDeg　　in)	
インクルード	#include　　"DAQtkUser.h"	
関数値	・赤緯文字列. フォーマットは, "+DD:MM:SS.SS"　　＊1	
入力パラメータ	・pDeg ：赤緯（degree）	
補足　＊1	文字列領域は関数の中で静的に確保されるので, 本関数を２度続けて呼び出すと領域が破壊される.	

5.8.11 角度変換（deg-rad）

角度変換（deg-rad）の呼び出し形式を表5.8.11に示します.

表5.8.11 角度変換（deg-rad）呼び出し形式

項目	内容	備考
機能大分類	ツールキット・ライブラリ	
機能分類	FITSキーワード値算出ツール	
和名	角度変換（deg-rad）	
英名	DAQtkFuncDDeg2DRad	
概要	degreeをradianへ変換する. 本関数はマクロである.	
呼出形式	DAQtkFuncDDeg2DRad(dDeg)	
記述インクルード	#include "DAQtkUser.h"	
マクロ値	・角度（radian）	
入力パラメータ	・dDeg ：角度（degree）	

5.8.12 **角度変換（rad-deg）**

角度変換（rad-deg）の呼び出し形式を表5.8.12に示します.

表5.8.12　角度変換（rad-deg）呼び出し形式

項目	内容	備考
機能大分類	ツールキット・ライブラリ	
機能分類	FITSキーワード値算出ツール	
和名	角度変換（rad-deg）	
英名	DAQtkFuncDRad2DDeg	
概要	radianをdegreeへ変換する. 本関数はマクロである.	
呼出形式	DAQtkFuncDRad2DDeg(dRad)	
記述インクルード	#include　"DAQtkUser.h"	
マクロ値	・角度（degree）	
入力パラメータ	・dRad ：角度（radian）	

5.8.13 角度変換（deg-hour）

角度変換（deg-hour）の呼び出し形式を表5.8.13に示します.

表5.8.13 角度変換（deg-hour）呼び出し形式

項目	内容	備考
機能大分類	ツールキット・ライブラリ	
機能分類	FITSキーワード値算出ツール	
和名	角度変換（deg-hour）	
英名	DAQtkFuncDDeg2DHour	
概要	degreeをhourへ変換する. 本関数はマクロである.	
呼出形式	DAQtkFuncDDeg2DHour(dDeg)	
記述インクルード	#include "DAQtkUser.h"	
マクロ値	・角度（hour）	
入力パラメータ	・dDeg ：角度（degree）	

5.8.14 **角度変換（hour-deg）**

角度変換（hour-deg）の呼び出し形式を表5.8.14に示します.

表5.8.14　角度変換（hour-deg）呼び出し形式

項目	内容	備考
機能大分類	ツールキット・ライブラリ	
機能分類	FITSキーワード値算出ツール	
和名	角度変換（hour-deg）	
英名	DAQtkFuncDHour2DDeg	
概要	hourをdegreeへ変換する. 本関数はマクロである.	
呼出形式	DAQtkFuncDHour2DDeg(dHour)	
記述インクルード	#include　"DAQtkUser.h"	
マクロ値	・角度（degree）	
入力パラメータ	・dHour　：角度（hour）	

5.9 FITS生成／転送高速化機能

5.9.1 FITSヘッダ生成

FITSヘッダ生成の呼び出し形式を表5.9.1に示す.

表5.9.1 FITSヘッダ生成の呼び出し形式 (1/2)

項目	内容	備考
機能大分類	ツールキット・ライブラリ	
機能分類	FITS生成／転送高速化	
和名	FITSヘッダ生成	
英名	DAQtkFitsEditHeader	
概要	FITSヘッダ項目をチェックし，ヘッダユニットを生成する.	
呼出形式	int DAQtkFitsEditHeader (int iAscii, in int iWcsFlg, in int iSpecFlg, in struct TFitsHeader* qFitsHeader, in struct TWcsPara* qWcsPara, in void** pArea, out int* pSize, out int* pRsltCode out)	
インクルード	#include "DAQtkUser.h"	
関数値	・DAQ_TK_OK ：正常終了 ・DAQ_TK_ERR ：異常終了	

表5.9.1 FITSヘッダ生成の呼び出し形式 (2/2)

項目	内容	備考
入力パラメータ	・iAscii	：ASCIIextensionフラグ 1 : ASCIIextensionのヘッダ （通常のヘッダは生成しない） 0 : 通常のヘッダ
	・iWcsFlg	：WCSフラグ（通常のヘッダのみ） 1 : WCSキーワードを付加する 0 : WCSキーワードを付加しない
	・iSpecFlg	：分光装置フラグ（WCSフラグが1の場合のみ） 1 : 分光装置用のWCSキーワードを付加する 0 : 通常のWCSキーワードを付加する
	・qFitsHeader	：FITSヘッダ項目
	・qWcsPara	：WCSパラメータ（WCSフラグが1の場合のみ）
出力パラメータ	・pArea	：ヘッダユニット格納領域　*1
	・pSize	：ヘッダユニットサイズ　*2
	・pRsltCode	：処理結果
補足 *1	ヘッダユニット領域はライブラリ内部でmallocされるため, **ユーザ側でこの領域をfree()する必要がある.** 以下に呼び出し例を示す. …… void* pHeader; → ヘッダユニットへのポインタ定義 …… DAQtkFitsEditHeader(0, 1, 0, …, …, &pHeader, …, …); → ライブラリにはpHeaderへのポインタを指定する pHeaderにmallocしたヘッダ領域が格納される …… free(pHeader); → 呼び出し後, どこかでfreeする必要がある ……	
補足 *2	mallocされたヘッダユニット領域のサイズが格納される	

5.9.2 取得データメモリ転送

取得データメモリ転送の呼び出し形式を表5.9.2に示す.

表5.9.2 取得データメモリ転送の呼び出し形式 (1/2)

項目	内容	備考
機能大分類	ツールキット・ライブラリ	
機能分類	FITS生成／転送高速化	
和名	取得データメモリ転送	
英名	DAQtkDatSndMem	
概要	取得データファイルをメモリからRPCによりOBCに転送する.	
呼出形式	int DAQtkDatSndMem (struct TUnit* qPrimeHeader, in struct TUnit* qPrimeData, in struct TUnit* qAsciiHeader1, in struct TUnit* qAsciiData1, in struct TUnit* qAsciiHeader2, in struct TUnit* qAsciiData2, in struct TUnit* qAsciiHeader3, in struct TUnit* qAsciiData3, in char* pFrameNo, in int* pRsltCode out)	
インクルード #include "DAQtkUser.h"		
関数値	・DAQ_TK_OK ：正常終了 ・DAQ_TK_ERR ：異常終了	

表5.9.2 取得データメモリ転送の呼び出し形式 (2/2)

項目	内容	備考
入力パラメータ	・qPrimeHeader	：プライマリヘッダ
	・qPrimeData	：プライマリデータ
	・qAsciiHeader1～3	：ASCIIextentionヘッダ （ない場合はNULLを設定）
	・qAsciiData1～3	：ASCIIextentionデータ （ない場合はNULLを設定）
	・pFrameNo	：フレーム番号
出力パラメータ	・pRsltCode	：処理結果

5.9.3 複数データメモリ転送

複数データメモリ転送の呼び出し形式を表5.9.3に示す.

表5.9.3 複数データメモリ転送の呼び出し形式 (1/2)

項目	内容	備考
機能大分類	ツールキット・ライブラリ	
機能分類	FITS生成／転送高速化	
和名	複数データメモリ転送	
英名	DAQtkDatSndMem2	
概要	取得データファイルをメモリからRPCによりOBCに転送する.	
呼出形式	int DAQtkDatSndMem2 (struct TFitsMemory* qTFitsMemory in *1 int* pRsltCode out)	
インクルード	#include "DAQtkUser.h"	
関数値	・DAQ_TK_OK ：正常終了 ・DAQ_TK_ERR ：異常終了	

表5.9.3 複数データメモリ転送の呼び出し形式 (2/2)

項目	内容	備考
入力パラメータ	・qTFitsMemory	：FITS情報
出力パラメータ	・pRsltCode	：処理結果

補足　*1　TFitsMemory（FITSメモリデータ）は構造体タグ名で，メンバは以下のとおり

```
struct TFitsMemory {
        struct TUnit*  qPrimeHeader;   ：プライマリヘッダ
        struct TUnit*  qPrimeData;     ：プライマリデータ
        struct TUnit*  qAsciiHeader1;  ：ASCIIextensionヘッダ１
                                         （ない場合はNULLを設定）
        struct TUnit*  qAsciiData1;    ：ASCIIextensionデータ１
                                         （ない場合はNULLを設定）
        struct TUnit*  qAsciiHeader2;  ：ASCIIextensionヘッダ２
                                         （ない場合はNULLを設定）
        struct TUnit*  qAsciiData2;    ：ASCIIextensionデータ２
                                         （ない場合はNULLを設定）
        struct TUnit*  qAsciiHeader3;  ：ASCIIextensionヘッダ３
                                         （ない場合はNULLを設定）
        struct TUnit*  qAsciiData3;    ：ASCIIextensionデータ３
                                         （ない場合はNULLを設定）
        char*     pFrameNo             ：フレーム番号
};
```

・50ファイルまでの複数指定で一度に転送が可能であるため，パラメータには上記構造体配列（50まで）の先頭へのポインタを指定する

・50ファイルに満たない場合，配列の終端はメンバの「フレーム番号」をNULLポインタとすること

5.9.4 FITSファイル出力

FITSファイル出力の呼び出し形式を表5.9.3に示す.

表5.9.3 FITSファイル出力の呼び出し形式

項目	内容	備考
機能大分類	ツールキット・ライブラリ	
機能分類	FITS生成／転送高速化	
和名	FITSファイル出力	
英名	DAQtkFitsFileOutput	
概要	FITSヘッダおよびデータを成形してファイル出力する.	
呼出形式	int DAQtkFitsFileOutput (struct TUnit* qPrimeHeader, in struct TUnit* qPrimeData, in struct TUnit* qAsciiHeader1, in struct TUnit* qAsciiData1, in struct TUnit* qAsciiHeader2, in struct TUnit* qAsciiData2, in struct TUnit* qAsciiHeader3, in struct TUnit* qAsciiData3, in char* pFileName, in int* pRsltCode out)	
インクルード	#include "DAQtkUser.h"	
関数値	・DAQ_TK_OK ：正常終了 ・DAQ_TK_ERR ：異常終了	
入力パラメータ	・qPrimeHeader ：プライマリヘッダ ・qPrimeData ：プライマリデータ ・qAsciiHeader1～3 ：ASCIIextentionヘッダ （ない場合はNULLを設定） ・qAsciiData1～3 ：ASCIIextentionデータ （ない場合はNULLを設定） ・pFileName ：ファイルパス名	
出力パラメータ	・pRsltCode ：処理結果	

5.9.5 FITSデータ生成／転送

FITSデータ生成／転送の呼び出し形式を表5.9.4に示します.

表5.9.4 FITSデータ生成／転送の呼び出し形式 (1/2)

項目	内容	備考
機能大分類	ツールキット・ライブラリ	
機能分類	FITS生成／転送高速化	
和名	FITSデータ生成／転送	
英名	DAQtkFitsEditSnd	
概要	FITSヘッダユニットを生成し，メモリからRPCによりOBCに転送するか，またはファイルに出力する.	
呼出形式	int DAQtkFitsEditSnd (int iWcsFlg, in int iSpecFlg, in struct TFitsHeader* qPrimeHeader, in struct TWcsPara* qWcsPara, in struct TUnit* qPrimeData, in struct TFitsHeader* qAsciiHeader1, in struct TUnit* qAsciiData1, in struct TFitsHeader* qAsciiHeader2, in struct TUnit* qAsciiData2, in struct TFitsHeader* qAsciiHeader3, in struct TUnit* qAsciiData3, in int iSndSaveFlg in char* pFrameNo, in char* pFileName, in int* pRsltCode out)	
インクルード	#include "DAQtkUser.h"	
関数値	・DAQ_TK_OK ：正常終了 ・DAQ_TK_ERR ：異常終了	

表5.9.4 FITSデータ生成／転送の呼び出し形式 (2/2)

項目	内容	備考
入力パラメータ	・iWcsFlg	：WCSフラグ １：WCSキーワードを付加する ０：WCSキーワードを付加しない
	・iSpecFlg	：分光装置フラグ（WCSフラグが1の場合のみ） １：分光装置用のWCSキーワードを付加する ０：通常のWCSキーワードを付加する
	・qPrimeHeader	：プライマリヘッダ項目
	・qWcsPara	：WCSパラメータ（WCSフラグが1の場合のみ）
	・qPrimeData	：プライマリデータ
	・qAsciiHeader1～3	：ASCIIextentionヘッダ項目 （ない場合はNULLを設定）
	・qAsciiData1～3	：ASCIIextentionデータ （ない場合はNULLを設定）
	・iSndSaveFlg	：転送／保存フラグ ０：RPCによりOBCにデータを転送する １：ファイルに保存する
	・pFrameNo	：フレーム番号
	・pFileName	：ファイルパス名
出力パラメータ	・pRsltCode	：処理結果

6. 補足

ツールキットを使用する上での、補足点・注意点を以下に示します。

6.1 各機能共通

1)各ライブラリの復帰値を表6.1.1に示します。

表6.1.1　ライブラリ復帰値

復帰値	値	意味
DAQ_TK_OK	0	正常
DAQ_TK_ERR	-1	異常

2)各ライブラリ出力パラメータの詳細コードを表6.1.2に示します。

表6.1.2　出力パラメータ詳細コード

詳細コード	値	意味
DAQ_TK_OK_NONE	0	正常
DAQ_TK_ERR_ETC	1	その他の異常
DAQ_TK_ERR_CMND	2	制御コマンド異常
DAQ_TK_ERR_ACK	3	受付応答異常
DAQ_TK_ERR_LOGIC	4	受付応答論理的異常
DAQ_TK_ERR_RPLY	5	処理完了異常
DAQ_TK_ERR_SEQ	6	処理完了シーケンシャル番号異常
DAQ_TK_ERR_START	7	転送開始異常
DAQ_TK_ERR_END	8	転送完了異常
DAQ_TK_ERR_ORDER	9	格納順異常
DAQ_TK_ERR_MUSTBE	10	必須異常
DAQ_TK_ERR_UNIQUE	11	一意異常
DAQ_TK_ERR_FORMAT	12	フォーマット異常
DAQ_TK_ERR_CTL	13	画面操作異常
DAQ_TK_ERR_DATARPC	14	データRPC転送異常

3)環境変数の設定は必ず行ってください。一部を除いて、設定されていないものがある場合、ツールキットは正常に動作しません。

4)通信ツール、ステータスデータ取得ツールについては、いちばんはじめに初期化処理関数を呼び出してください。初期化処理関数が呼び出されていない場合、他の関数は処理を行わず異常復帰します。
（詳細コード = DAQ_TK_ERR_ETC）

5)初期化処理関数と終了処理関数は必ず対にして呼び出してください。
初期化処理関数呼び出し後、終了処理関数を呼び出さずにシステムを再起動
（リブート）した場合、再び初期化処理が行えない可能性があります.
^ ^ ^ ^ ^ ^ ^ ^

→万が一、初期化処理が行えない場合，以下のディレクトリにある
ComFcntl.dat以外のファイルをすべて削除してください。ComFcntl.dat は絶対に
削除しないでください。
.../daqtk/dat/

6)初期化処理関数と終了処理関数を早い周期で繰り返し呼び出すことは推奨できません.
プロセスの起動／停止，および資源の初期化／解放が行われるため，多くの時間が
かかり，性能的に不利になります.
プロセスの起動時，停止時に一度ずつだけ呼び出す方法を推奨します.

7)ツールキットを利用する（すばる観測制御システムに接続する）ホスト名は
８文字以内である必要があります.
これはRPC電文ヘッダ部フォーマットの制約があるためです.
８文字を超えた設定を行うと，通信を行うことができません.

6.2 コマンド通信機能

1)制御コマンド受信関数を呼び出すユーザプロセスはOBCP内で１つのみとします。複数プロセスから呼び出した場合の動作は保証されません。処理完了送信関数は、複数プロセスから呼び出すことが可能です。

2)制御コマンド受信関数は、受信した制御コマンドのフォーマットが異常の場合、受付応答のチェック結果にエラーコードを格納してコマンドディスパッチャに送信し、異常復帰します。
（詳細コード = DAQ_TK_ERR_CMD）

3)処理完了を送信する前に、制御コマンドを受信できる数は1024電文までとなっています。これを超えて制御コマンドを受信した場合、制御コマンド受信関数は、受付応答のチェック結果にエラーコードを格納してコマンドディスパッチャに送信し、異常復帰します。
（詳細コード = DAQ_TK_ERR_CMD）

4)本機能は制御コマンド受信関数と処理完了送信関数の間で、シーケンシャル番号のチェックを行っています。これにより、受信していない制御コマンドに対する処理完了を送信しようとすると、送信は行わず、処理完了送信関数は異常復帰します。
（詳細コード = DAQ_TK_ERR_ETC）

6.3 透過モード・コマンド通信機能

1)制御コマンド送信関数、および処理完了受信関数を呼び出すユーザプロセスはOBCP内で１つのみとします。複数プロセスから呼び出した場合の動作は保証されません。

2)制御コマンド送信関数は、受付応答の受信を待ち合わせてから復帰します。制御コマンドを送信してから５秒以内に受付応答を受信しない場合、タイムアウトして異常復帰します。
（詳細コード = DAQ_TK_ERR_ACK）

3)また、受信した受付応答のチェック結果が正常でない場合、つまりエグゼキュータが制御コマンドを受信できない状態の場合も、制御コマンド送信関数は異常復帰します。
（詳細コード = DAQ_TK_ERR_LOGIC）

4)処理完了を受信する前に、制御コマンドを送信できる数は1024電文までとなっています。これを超えて制御コマンド送信関数を呼び出した場合、制御コマンド送信を行わず異常復帰します。
（詳細コード = DAQ_TK_ERR_ETC）

5)処理完了受信関数は、受信した処理完了のフォーマットが異常の場合、および処理完了がエラー（コマンド実行がエラー）の場合，異常復帰します。
（詳細コード = DAQ_TK_ERR_RPLY）

6)本機能は制御コマンド送信関数と処理完了受信関数の間で、シーケンシャル番号のチェックを行っています。これにより、送信していない制御コマンドに対する処理完了を受信した場合、処理完了受信関数は異常復帰します。
（詳細コード = DAQ_TK_ERR_SEQ）

6.4 ステータス転送機能

1)ステータス転送関数は、複数プロセスから呼び出すことが可能です。

6.5 取得データ転送機能

1)取得データ転送関数，および取得データメモリ転送を呼び出すユーザプロセスは
OBCP内で１つのみとします。複数プロセスから呼び出した場合の動作は保証されません。

2)取得データ転送関数は、入力パラメータにOBCP上に存在しないパス名を指定した場合、
取得データ転送開始の送信を行わず、異常復帰します。
（詳細コード = DAQ_TK_ERR_ETC）

3)取得データ転送関数は、受付応答の受信、および取得データ転送完了の受信を待ち合
わせてから復帰します。取得データ転送を送信してから５秒以内に受付応答を受信し
ない場合、タイムアウトして異常復帰します。
（詳細コード = DAQ_TK_ERR_ACK）

4)また、受信した受付応答のチェック結果が正常でない場合、つまりOBCが取得データ
転送を受信できない状態の場合も、取得データ転送関数は異常復帰します。
（詳細コード = DAQ_TK_ERR_LOGIC）

5)取得データ転送完了に対するタイムアウト処理は行わず、受信するまで取得データ転送
関数は復帰しません。また、受信した取得データ転送完了のフォーマットが異常の場合、
および転送結果がエラーの場合，異常復帰します。
（詳細コード = DAQ_TK_ERR_END）

6.6 ステータスデータ取得機能

1)ステータスデータ要求送信関数、およびステータスデータ完了受信関数を呼び出すユーザプロセスはOBCP内で１つのみとします。複数プロセスから呼び出した場合の動作は保証されません。

2)ステータスデータ要求送信関数は、受付応答の受信を待ち合わせてから復帰します。ステータスデータ要求を送信してから５秒以内に受付応答を受信しない場合、タイムアウトして異常復帰します。
（詳細コード = DAQ_TK_ERR_ACK）

3)また、受信した受付応答のチェック結果が正常でない場合、つまりステータスディストリビューションサービスがステータスデータ要求を受信できない状態の場合も、ステータスデータ要求送信関数は異常復帰します。
（詳細コード = DAQ_TK_ERR_LOGIC）

4)ステータスデータ完了を受信する前に、ステータスデータ要求を送信できる数は1024電文までとなっています。これを超えてステータスデータ要求送信関数を呼び出した場合、ステータスデータ要求送信を行わず異常復帰します。
（詳細コード = DAQ_TK_ERR_ETC）

5)ステータスデータ完了受信関数は、受信したステータスデータ完了のフォーマットが異常の場合､およびステータスデータ完了がエラー(ステータス獲得がエラー)の場合,異常復帰します。
（詳細コード = DAQ_TK_ERR_RPLY）

6)本機能はステータスデータ要求送信関数とステータスデータ完了受信関数の間で、シーケンシャル番号のチェックを行っています。これにより、送信していないステータスデータ要求に対するステータスデータ完了を受信した場合、ステータスデータ完了受信関数は異常復帰します。
（詳細コード = DAQ_TK_ERR_SEQ）

6.7 FITSファイル作成ツール

1)FITSキーワード設定ファイルは、フォーマットに従い正しく設定してください。ファイルフォーマットが異なった場合の動作は保証されません。

2)FITSキーワード設定ファイルの固定キーワードが、プライマリヘッダ項目の入力パラメータで設定された場合、入力パラメータの値が優先されます。

3)FITSヘッダユニット生成において、入力パラメータで指定されたカード項目のサイズの和が80バイトを超える場合、カード項目を80バイトの箇所で切り落とします。

4)ASCIIextensionのヘッダを作成するときに使用する設定ファイルである
･･･daqtk/dat/ascii.txt は、絶対に削除、更新しないでください。
削除、更新を行うと正常に動作しません。

5)WCSキーワード付加を指定した場合は、必ずWCSパラメータを設定してください。

6)ASCIIextensionの指定をした場合は、必ずASCIIextensionのヘッダ項目、及びASCIIextensionのデータを設定してください。

7)WCSのキーワードがプライマリヘッダ項目の入力パラメータで設定された場合、WCSのキーワードが優先されます。

8)ASCIIextensionの固定キーワードがASCIIextensionヘッダ項目の入力パラメータで設定された場合、ASCIIextensionの固定キーワードが優先されます。

9)WCSコメントファイルは、フォーマットに従い正しく設定してください。ファイルフォーマットが異なった場合の動作は保証されません。

10)WCSコメントファイルがなかった場合、および環境変数によるWCSコメントファイルパス名の指定がなかった場合は、WCSキーワードのコメントは空白となります。

6.8 モニタ表示ツール

1)SAOtngの実行ファイルディレクトリを"path"に必ず追加してください。
追加していない場合、本ツールはSAOtngやSAOtng関連の各種コマンドを起動することができません。

2)本ツールはデータ表示画面の２重起動を防ぎます。
２回連続で起動、もしくは、終了を指示すると、データ表示画面制御ライブラリは処理を行わず復帰します。
（詳細コード = DAQ_TK_ERR_CTL）

3)データ表示画面の標準出力、ならびに、標準エラー出力は画面上に出力されません。
ただし、利用者側が発行した xpaget，xpasetコマンドによるデータ表示画面へのアクセス結果は画面上に出力されます。

4)バイナリ配列データ領域のサイズを正しくパラメタに設定して、データ表示ライブラリに通知してください。
正しく設定されていない場合、ユーザプロセス側が異常終了(core dump)する恐れがあります。

5)データ表示ライブラリは、システムにより制限される共有メモリの最大サイズ以上の画像データの表示を指示された場合、処理を行わず復帰します。
（詳細コード = DAQ_TK_ERR_ETC）
共有メモリの最大値設定については、3.4システムファイルの設定 を参照してください。

6.9 FITSキーワード値算出機能

1)以下のライブラリでは復帰値の領域が関数内で静的に確保されるので，２度続けて呼び出すとその領域が破壊されます.

- DAQtkFuncDHour2CTime
- DAQtkFuncDDeg2CRA
- DAQtkFuncDDeg2Cdec

7. サンプルメインプログラム

以下の機能についてサンプルメインプログラムのプロセスフローを次頁以降に示します。

・コマンド通信機能
・透過モード・コマンド通信機能
・ステータス転送機能
・取得データ転送機能
・ステータスデータ取得機能
・FITSファイル作成ツール
・モニタ表示ツール
・FITSキーワード値算出機能

7.1 コマンド通信機能

コマンド通信機能におけるサンプルプログラム一覧を以下に示します。

表7.1.1 コマンド通信機能サンプルプログラム一覧

No	プログラム名	パス名
1	コマンド通信初期化処理呼び出し	.../daqtk/sample/src/cmd/init/cmdinit.c
2	コマンド通信終了処理呼び出し	.../daqtk/sample/src/cmd/exit/cmdexit.c
3	処理完了送信呼び出し	.../daqtk/sample/src/cmd/snd/cmdsnd.c
4	制御コマンド受信呼び出し	.../daqtk/sample/src/cmd/rcv/cmdrcv.c
5	制御コマンド受信チェック呼び出し	.../daqtk/sample/src/cmd/check/cmdcheck.c

（注 ...はインストール先ディレクトリを示します）

1) コマンド通信初期化処理呼び出し

コマンドイメージで起動されるバッチプログラムです。コマンド通信機能を使用する場合には、いちばん初めに必ず起動してください。

表7.1.2 コマンド通信初期化処理呼び出しプロセスフロー

No	処理	処理内容	次No
1	初期化処理	以下の関数を呼び出し、初期化処理を行う。 DAQtkCmdInit()	
2	結果出力	初期化処理関数の復帰値を出力する。	

2) コマンド通信終了処理呼び出し

コマンドイメージで起動されるバッチプログラムです。コマンド通信機能を使用しない場合には、最後に必ず起動してください。

表7.1.3 コマンド通信終了処理呼び出しプロセスフロー

No	処理	処理内容	次No
1	終了処理	以下の関数を呼び出し、終了処理を行う。 DAQtkCmdExit()	
2	結果出力	終了処理関数の復帰値を出力する。	

3) 処理完了送信呼び出し

制御コマンド受信呼び出しと、FIFOを用いてインタフェースを取っています。

処理完了送信呼び出しがFIFOを生成して入力を待ち合わせ、読み込んだ場合に処理完了を送信します。制御コマンド受信呼び出しは制御コマンドを受信した場合に、FIFOに制御コマンドを書き込みます。

処理完了送信呼び出しがFIFOを生成するため、こちらを先に起動します。

表7.1.4 処理完了送信呼び出しプロセスフロー

No	処理	処理内容	次No
1	FIFO生成	制御コマンド受信呼び出しとのインタフェース用FIFOを生成する。	
2	FIFOオープン	FIFOをオープンする。	
3	FIFO入力待ち	制御コマンド受信呼び出しからのFIFOへの入力を待ち合わせる。	
4	処理完了編集	FIFOから読み込んだ制御コマンドより、処理完了電文を編集する。	
5	処理完了送信	以下の関数を呼び出し、処理完了を送信する。 DAQtkCmdSnd()	
6	結果出力	処理完了送信関数の復帰値を出力する。	3

4) 制御コマンド受信呼び出し

処理完了送信呼び出しがFIFOを生成するため、処理完了送信呼び出しよりも後に起動します。

表7.1.5 制御コマンド受信呼び出しプロセスフロー

No	処理	処理内容	次No
1	制御コマンド受信	以下の関数を呼び出し、制御コマンドを受信する。 DAQtkCmdRcv()	
2	結果出力	制御コマンド受信関数の復帰値を出力する。	
3	FIFOオープン	FIFOをオープンする。	
4	FIFO書き込み	受信した制御コマンドをFIFOに書き込む。	
5	FIFOクローズ	FIFOをクローズする。	1

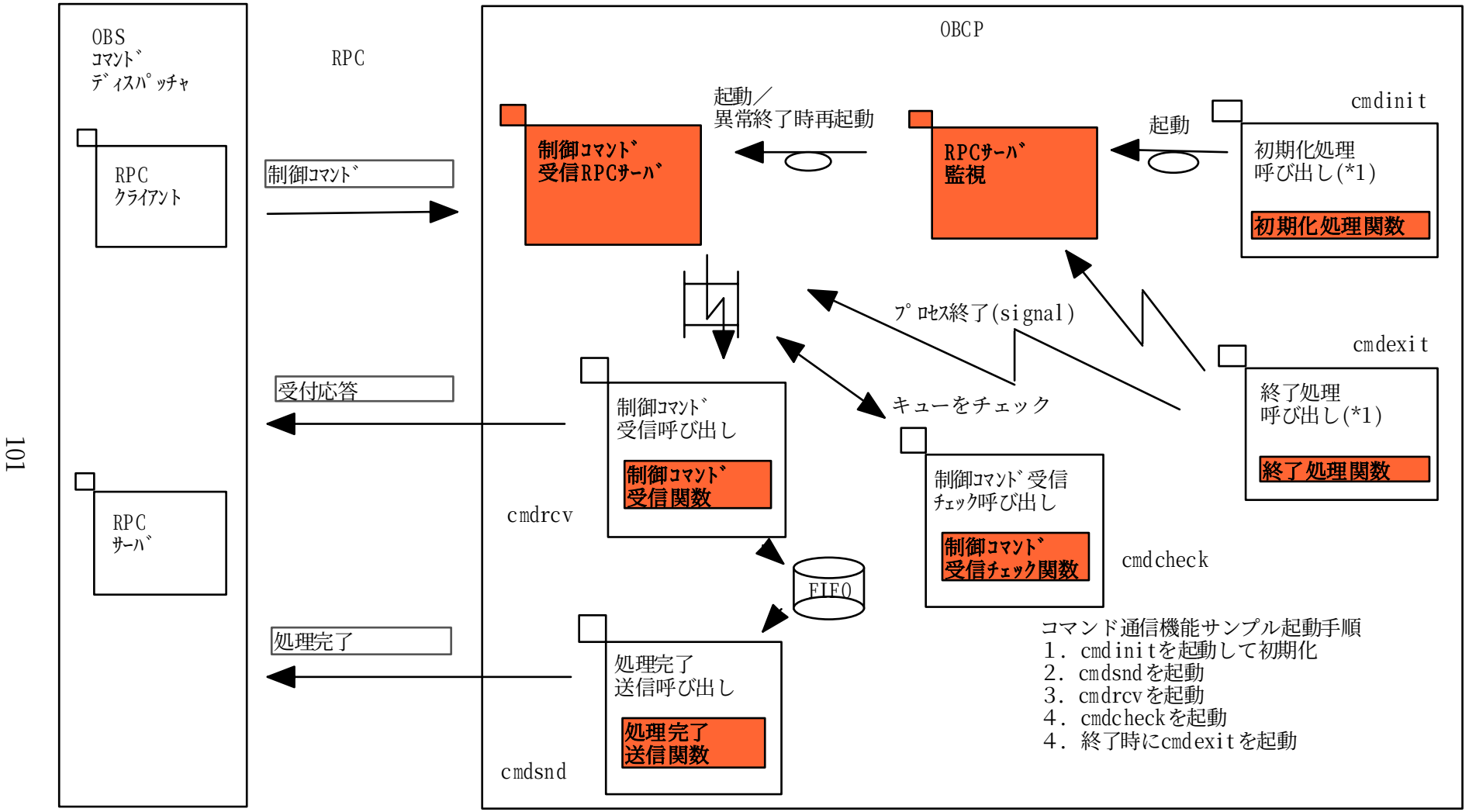
5) 制御コマンド受信チェック呼び出し

コマンド通信機能を使用している際に、使用してください。

表7.1.6 制御コマンド受信チェック呼び出しプロセスフロー

No	処理	処理内容	次No
1	制御コマンド受信チェック	以下の関数を呼び出し、制御コマンドの受信状態をチェックする。 DAQtkCmdRcvCheck()	
2	結果出力	制御コマンド受信チェック関数の復帰値を出力する。	

コマンド通信機能サンプルプログラム構成図を図7.1.1に示します。



*1　コマンドイメージで起動されるバッチプログラム

図7.1.1 コマンド通信機能サンプルプログラム構成図

7.2 透過モード・コマンド通信機能

透過モード・コマンド通信機能におけるサンプルプログラム一覧を以下に示します。

表7.2.1 透過モード・コマンド通信機能サンプルプログラム一覧

No	プログラム名	パス名
1	透過モード・コマンド通信初期化処理呼び出し	.../daqtk/sample/src/through/init/throughinit.c
2	透過モード・コマンド通信終了処理呼び出し	.../daqtk/sample/src/through/exit/throughexit.c
3	処理完了受信呼び出し	.../daqtk/sample/src/through/rcv/throughrcv.c
4	制御コマンド送信呼び出し	.../daqtk/sample/src/through/snd/throughsnd.c
5	FIFO書き込み	.../daqtk/sample/src/through/snd/throughwrite.c

1) 透過モード・コマンド通信初期化処理呼び出し

コマンドイメージで起動されるバッチプログラムです。透過モード・コマンド通信機能を使用する場合には、いちばん初めに必ず起動してください。

表7.2.2 透過モード・コマンド通信初期化処理呼び出しプロセスフロー

No	処理	処理内容	次No
1	初期化処理	以下の関数を呼び出し、初期化処理を行う。 DAQtkThroughInit()	
2	結果出力	初期化処理関数の復帰値を出力する。	

2) 透過モード・コマンド通信終了処理呼び出し

コマンドイメージで起動されるバッチプログラムです。透過モード・コマンド通信機能を使用しない場合には、最後に必ず起動してください。

表7.2.3 透過モード・コマンド通信終了処理呼び出しプロセスフロー

No	処理	処理内容	次No
1	終了処理	以下の関数を呼び出し、終了処理を行う。 DAQtkThroughExit()	
2	結果出力	終了処理関数の復帰値を出力する。	

3) 処理完了受信呼び出し

制御コマンドを送信する前に起動してください。

表7.2.4 処理完了受信呼び出しプロセスフロー

No	処理	処理内容	次No
1	処理完了受信	以下の関数を呼び出し、処理完了を受信する。 DAQtkThroughRcv()	
2	結果出力	処理完了受信関数の復帰値を出力する。	
3	ロギング文字列編集	受信した処理完了をファイルに格納するための文字列を編集する。	
4	ファイル格納	編集した文字列をファイルに格納する。	

1

4) 制御コマンド送信呼び出し

FIFO書き込みと、FIFOを用いてインタフェースを取っています。

制御コマンド送信呼び出しがFIFOを生成して入力を待ち合わせ、読み込んだ場合に制御コマンドを送信します。FIFO書き込みは、任意のタイミングでFIFOに制御コマンドを書き込みます。

制御コマンド送信呼び出しがFIFOを生成するため、こちらを先に起動します。

表7.2.5 制御コマンド送信呼び出しプロセスフロー

No	処理	処理内容	次No
1	FIFO生成	FIFO書き込みとのインタフェース用FIFOを生成する。	
2	FIFOオープン	FIFOをオープンする。	
3	FIFO入力待ち	FIFO書き込みからのFIFOへの入力を待ち合わせる。	
4	制御コマンド送信	FIFOから読み込んだ制御コマンドを入力パラメータに指定し、以下の関数を呼び出して制御コマンドを送信する。 DAQtkThroughSnd()	
5	結果出力	制御コマンド送信関数の復帰値を出力する。	3

5) FIFO書き込み

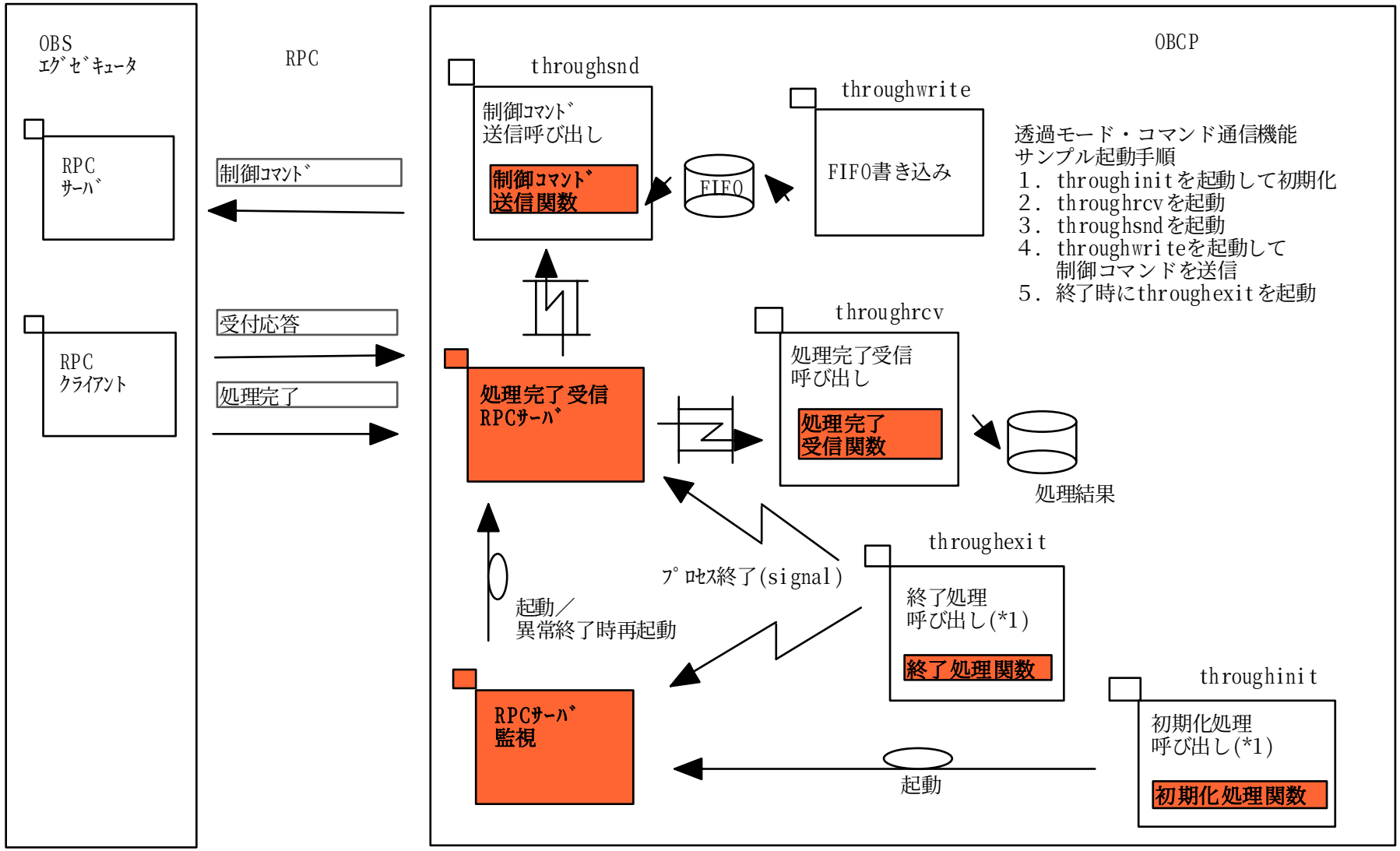
制御コマンド送信呼び出しがFIFOを生成するため、制御コマンド送信呼び出しを起動した後、制御コマンドを送信するタイミングで起動します。

FIFOに書き込む制御コマンド、書き込み回数、および書き込み周期などをプログラム内の変数で指定していますので、これらを自由に設定して、makeしてから起動してください。

表7.2.6 FIFO書き込みプロセスフロー

No	処理	処理内容	次No
1	制御コマンド編集	FIFOに書き込む制御コマンドを編集する。	
2	FIFO書き込みループ	指定回数だけ以下の処理を行う。	
		・指定回数を超えた	終
3	FIFOオープン	FIFOをオープンする。	
4	FIFO書き込み	編集した制御コマンドをFIFOに書き込む。	
5	FIFOクローズ	FIFOをクローズする。	
6	待ち合わせ	指定した秒数だけ待ち合わせる。	

透過モード・コマンド通信機能サンプルプログラム構成図を図7.2.1に示します。



*1 コマンドイメージで起動されるバッチプログラム

図7.2.1 透過モード・コマンド通信機能サンプルプログラム構成図

7.3 ステータス転送機能

ステータス転送機能におけるサンプルプログラム一覧を以下に示します。

表7.3.1 ステータス転送機能サンプルプログラム一覧

No	プログラム名	パス名
1	ステータス転送初期化処理呼び出し	.../daqtk/sample/src/stat/init/statinit.c
2	ステータス転送終了処理呼び出し	.../daqtk/sample/src/stat/exit/statexit.c
3	ステータス転送呼び出し	.../daqtk/sample/src/stat/snd/statsnd.c
4	FIFO書き込み	.../daqtk/sample/src/stat/snd/statwrite.c

1) ステータス転送初期化処理呼び出し

コマンドイメージで起動されるバッチプログラムです。ステータス転送機能を使用する場合には、いちばん初めに必ず起動してください。

表7.3.2 ステータス転送初期化処理呼び出しプロセスフロー

No	処理	処理内容	次No
1	初期化処理	以下の関数を呼び出し、初期化処理を行う。 DAQtkStatInit()	
2	結果出力	初期化処理関数の復帰値を出力する。	

2) ステータス転送終了処理呼び出し

コマンドイメージで起動されるバッチプログラムです。ステータス転送機能を使用しない場合には、最後に必ず起動してください。

表7.3.3 ステータス転送終了処理呼び出しプロセスフロー

No	処理	処理内容	次No
1	終了処理	以下の関数を呼び出し、終了処理を行う。 DAQtkStatExit()	
2	結果出力	終了処理関数の復帰値を出力する。	

3) ステータス転送呼び出し

FIFO書き込みと、FIFOを用いてインタフェースを取っています。

ステータス転送呼び出しがFIFOを生成して入力を待ち合わせ、読み込んだ場合にステータスデータを送信します。FIFO書き込みは、任意のタイミングでFIFOにステータスデータを書き込みます。

ステータス転送呼び出しがFIFOを生成するため、こちらを先に起動します。

表7.3.4 ステータス転送呼び出しプロセスフロー

No	処理	処理内容	次No
1	FIFO生成	FIFO書き込みとのインタフェース用FIFOを生成する。	
2	FIFOオープン	FIFOをオープンする。	
3	FIFO入力待ち	FIFO書き込みからのFIFOへの入力を待ち合わせる。	
4	ステータス転送	FIFOから読み込んだステータスデータを入力パラメータに指定し、以下の関数を呼び出してステータスデータを転送する。 DAQtkStatSnd()	
5	結果出力	ステータス転送関数の復帰値を出力する。	3

4) FIFO書き込み

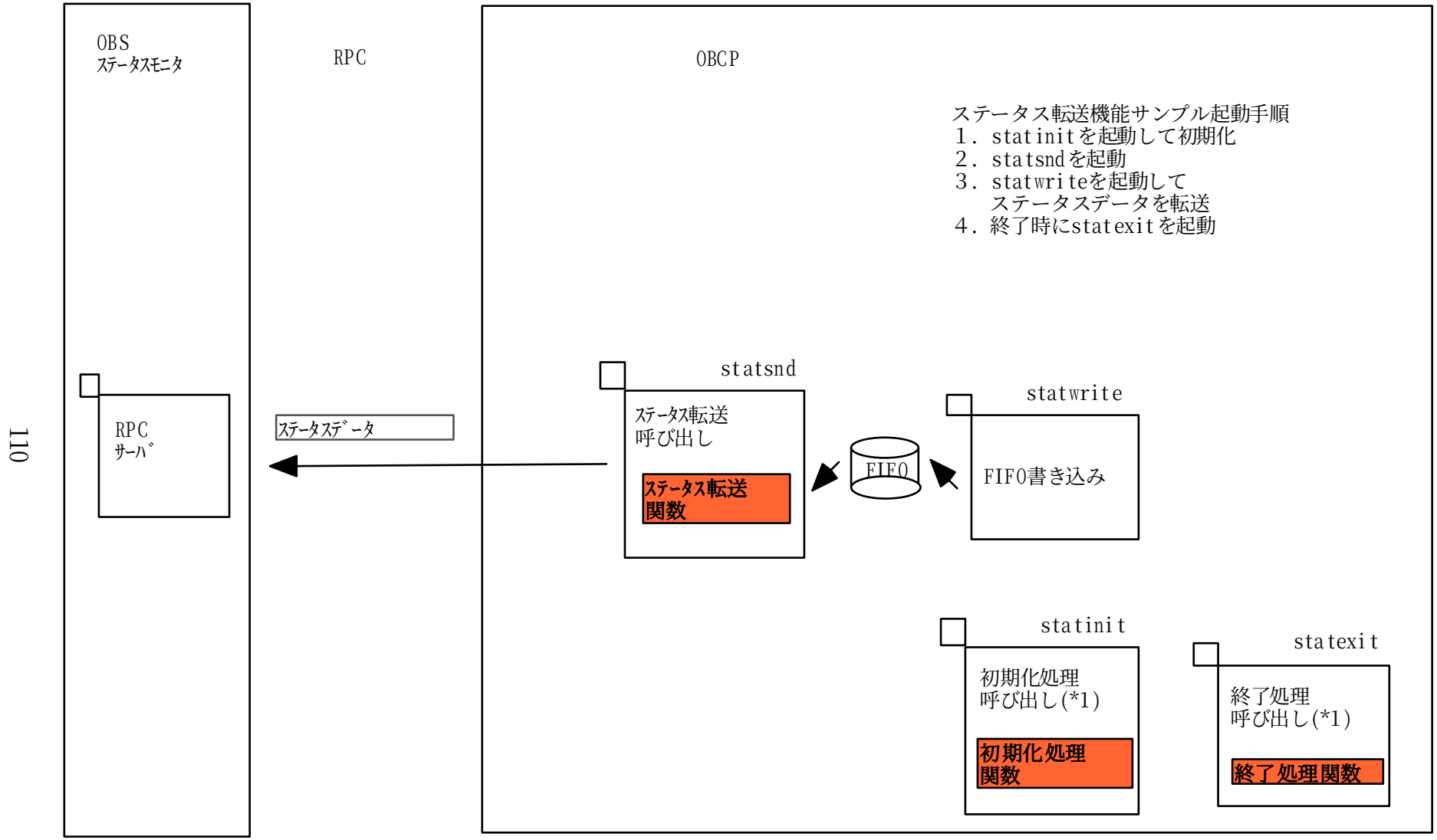
ステータス転送呼び出しがFIFOを生成するため、ステータス転送呼び出しを起動した後、ステータスデータを転送するタイミングで起動します。

FIFOに書き込むステータスデータ、書き込み回数、および書き込み周期などをプログラム内の変数で指定していますので、これらを自由に設定して、makeしてから起動してください。

表7.3.5 FIFO書き込みプロセスフロー

No	処理	処理内容	次No
1	ステータスデータ編集	FIFOに書き込むステータスデータを編集する。	
2	FIFO書き込みループ	指定回数だけ以下の処理を行う。	
		・指定回数を超えた	終
3	FIFOオープン	FIFOをオープンする。	
4	FIFO書き込み	編集したステータスデータをFIFOに書き込む。	
5	FIFOクローズ	FIFOをクローズする。	
6	待ち合わせ	指定した秒数だけ待ち合わせる。	

ステータス転送機能サンプルプログラム構成図を図7.3.1に示します。



*1 コマンドイメージで起動されるバッチプログラム

図7.3.1 ステータス転送機能サンプルプログラム構成図

7.4 取得データ転送機能

取得データ転送機能におけるサンプルプログラム一覧を以下に示します。

表7.4.1 取得データ転送機能サンプルプログラム一覧

No	プログラム名	パス名
1	取得データ転送初期化処理呼び出し	.../daqtk/sample/src/dat/init/datinit.c
2	取得データ転送終了処理呼び出し	.../daqtk/sample/src/dat/exit/datexit.c
3	取得データ転送呼び出し	.../daqtk/sample/src/dat/snd/datsnd.c
4	FIFO書き込み	.../daqtk/sample/src/dat/snd/datwrite.c
5	FITSデータ生成／転送呼び出し	.../daqtk/sample/src/dat/snd/datsndmem.c
6	複数メモリ転送	.../daqtk/sample/src/dat/snd/datsndmem2.c

1) 取得データ転送初期化処理呼び出し

コマンドイメージで起動されるバッチプログラムです。取得データ転送機能を使用する場合には、いちばん初めに必ず起動してください。

表7.4.2 取得データ転送初期化処理呼び出しプロセスフロー

No	処理	処理内容	次No
1	初期化処理	以下の関数を呼び出し、初期化処理を行う。 DAQtkDatInit()	
2	結果出力	初期化処理関数の復帰値を出力する。	

2) 取得データ転送終了処理呼び出し

コマンドイメージで起動されるバッチプログラムです。取得データ転送機能を使用しない場合には、最後に必ず起動してください。

表7.4.3 取得データ転送終了処理呼び出しプロセスフロー

No	処理	処理内容	次No
1	終了処理	以下の関数を呼び出し、終了処理を行う。 DAQtkDatExit()	
2	結果出力	終了処理関数の復帰値を出力する。	

3) 取得データ転送呼び出し

FIFO書き込みと、FIFOを用いてインタフェースを取っています。

取得データ転送呼び出しがFIFOを生成して入力を待ち合わせ、読み込んだ場合に取得データ転送を呼び出します。FIFO書き込みは、任意のタイミングでFIFOに取得データ転送入力パラメータを書き込みます。

取得データ転送呼び出しがFIFOを生成するため、こちらを先に起動します。

表7.4.4 取得データ転送呼び出しプロセスフロー

No	処理	処理内容	次No
1	FIFO生成	FIFO書き込みとのインタフェース用FIFOを生成する。	
2	FIFOオープン	FIFOをオープンする。	
3	FIFO入力待ち	FIFO書き込みからのFIFOへの入力を待ち合わせる。	
4	取得データ転送	FIFOから読み込んだ取得データパス名を入力パラメータに指定し、以下の関数を呼び出して取得データを転送する。 DAQtkDatSnd()	
5	結果出力	取得データ転送関数の復帰値を出力する。	

3

4) FIFO書き込み

取得データ転送呼び出しがFIFOを生成するため、取得データ転送呼び出しを起動した後、取得データを転送するタイミングで起動します。

FIFOに書き込む取得データ転送入力パラメータ、書き込み回数、および書き込み周期などをプログラム内の変数で指定していますので、これらを自由に設定して、makeしてから起動してください。

表7.4.5 FIFO書き込みプロセスフロー

No	処理	処理内容	次No
1	取得データ転送 入力パラメータ編集	FIFOに書き込む取得データ転送の入力パラメータを編集する。	
2	FIFO書き込みループ	指定回数だけ以下の処理を行う。 ・指定回数を超えた	終
3	FIFOオープン	FIFOをオープンする。	
4	FIFO書き込み	編集した取得データ転送の入力パラメータをFIFOに書き込む。	
5	FIFOクローズ	FIFOをクローズする。	
6	待ち合わせ	指定した秒数だけ待ち合わせる。	

5) FITSデータ生成／転送呼び出し

FITSデータ生成／転送関数を起動引数により呼び出します.

FITSデータ生成／転送関数は，FITSヘッダ生成関数，取得データメモリ転送関数，FITSファイル出力関数を内部で呼び出すため，これらのライブラリの呼び出し確認も同時に行えます.

表7.4.6 FITSデータ生成／転送呼び出しプロセスフロー

No	処理	処理内容	次No
1	プライマリヘッダ項目設定	プライマリヘッダ項目をプログラム内部で設定する.	
2	ASCII extension ヘッダ項目設定	ASCII extension ヘッダ項目をプログラム内部で設定する.	
3	プライマリデータ設定	プライマリデータとして適当な領域を設定する.	
4	ASCII extension データ設定	ASCII extension データとして適当な領域を設定する.	
5	FITSデータ生成／転送呼び出し	FITSデータ生成／転送ライブラリを起動引数により呼び出す.	

6) 複数メモリ転送

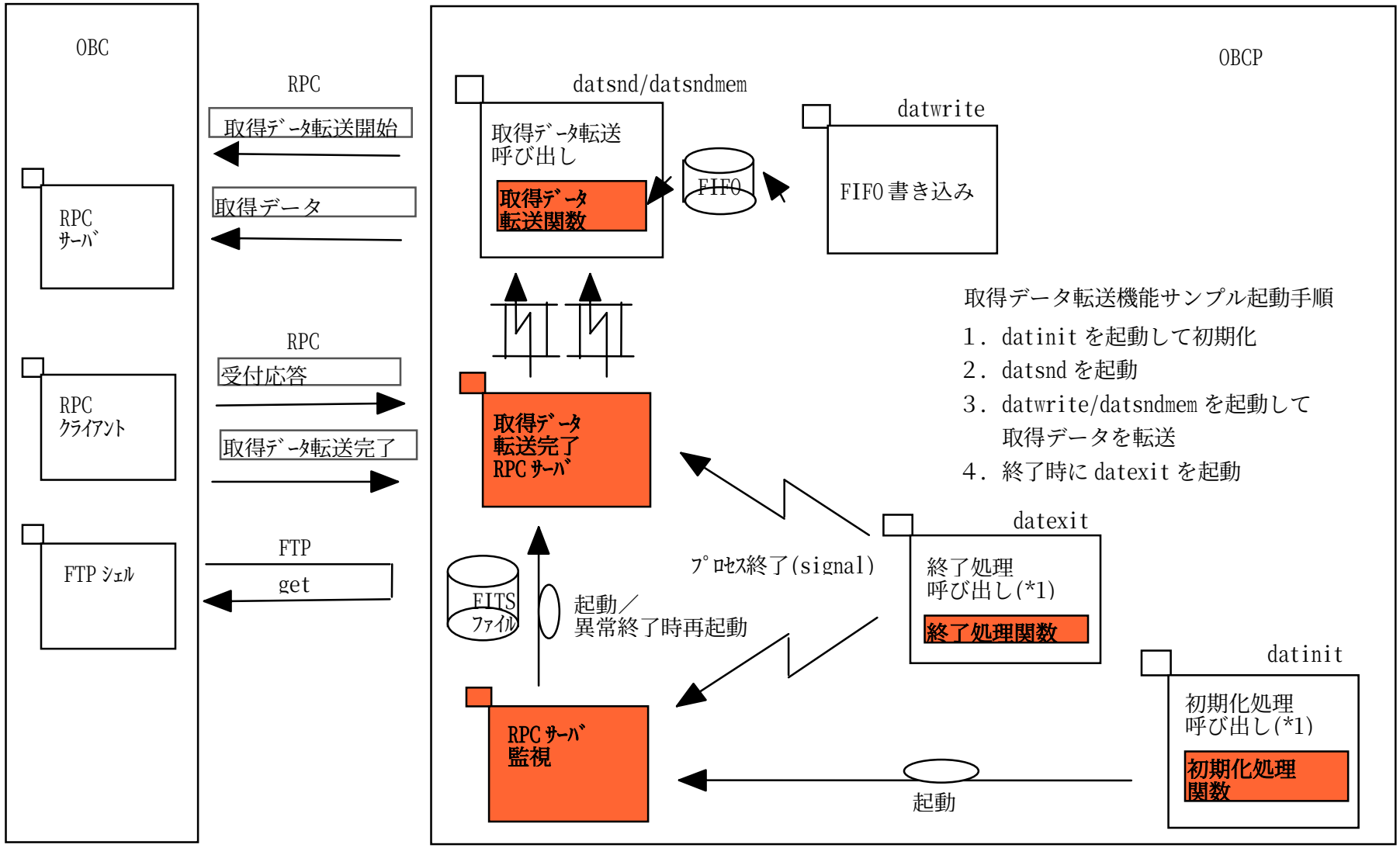
複数メモリ転送関数を起動引数により呼び出します.

複数メモリ転送関数は，FITSヘッダ生成関数，複数データメモリ転送関数，FITSファイル出力関数を内部で呼び出すため，これらのライブラリの呼び出し確認も同時に行えます.

表7.4.6 複数メモリ転送呼び出しプロセスフロー

No	処理	処理内容	次No
1	プライマリヘッダ項目設定	プライマリヘッダ項目をプログラム内部で設定する.	
2	ASCII extension ヘッダ項目設定	ASCII extension ヘッダ項目をプログラム内部で設定する.	
3	プライマリデータ設定	プライマリデータとして適当な領域を設定する.	
4	ASCII extension データ設定	ASCII extension データとして適当な領域を設定する.	
5	複数メモリ転送呼び出し	複数メモリ転送ライブラリを起動引数により呼び出す.	

取得データ転送機能サンプルプログラム構成図を図7.4.1に示します。



*1 コマンドイメージで起動されるバッチプログラム

図 7.4.1 取得データ転送機能サンプルプログラム構成図

7.5 ステータスデータ取得機能

ステータスデータ取得機能におけるサンプルプログラム一覧を以下に示します。

表7.5.1 ステータスデータ取得機能サンプルプログラム一覧

No	プログラム名	パス名
1	ステータスデータ取得初期化処理呼び出し	.../daqtk/sample/src/get/init/getinit.c
2	ステータスデータ取得終了処理呼び出し	.../daqtk/sample/src/get/exit/getexit.c
3	ステータスデータ完了受信呼び出し	.../daqtk/sample/src/get/rcv/getrcv.c
4	ステータスデータ要求送信呼び出し	.../daqtk/sample/src/get/snd/getsnd.c
5	FIFO書き込み	.../daqtk/sample/src/get/snd/getwrite.c

1) ステータスデータ取得初期化処理呼び出し

コマンドイメージで起動されるバッチプログラムです。ステータスデータ取得機能を使用する場合には、いちばん初めに必ず起動してください。

表7.5.2 ステータスデータ取得初期化処理呼び出しプロセスフロー

No	処理	処理内容	次No
1	初期化処理	以下の関数を呼び出し、初期化処理を行う。 DAQtkGetInit()	
2	結果出力	初期化処理関数の復帰値を出力する。	

2) ステータスデータ取得終了処理呼び出し

コマンドイメージで起動されるバッチプログラムです。ステータスデータ取得機能を使用しない場合には、最後に必ず起動してください。

表7.5.3 ステータスデータ取得終了処理呼び出しプロセスフロー

No	処理	処理内容	次No
1	終了処理	以下の関数を呼び出し、終了処理を行う。 DAQtkGetExit()	
2	結果出力	終了処理関数の復帰値を出力する。	

3) ステータスデータ完了受信呼び出し

ステータスデータ取得要求を送信する前に起動してください。

表7.5.4 ステータスデータ完了受信呼び出しプロセスフロー

No	処理	処理内容	次No
1	ステータスデータ取得完了受信	以下の関数を呼び出し、ステータスデータ取得完了を受信する。 DAQtkGetRcv()	
2	結果出力	ステータスデータ完了受信関数の復帰値を出力する。	
3	ロギング文字列編集	受信したステータスデータ取得完了をファイルに格納するための文字列を編集する。	
4	ファイル格納	編集した文字列をファイルに格納する。	1

4) ステータスデータ要求送信呼び出し

FIFO書き込みと、FIFOを用いてインタフェースを取っています。

ステータスデータ要求送信呼び出しがFIFOを生成して入力を待ち合わせ、読み込んだ場合にステータスデータ取得要求を送信します。FIFO書き込みは、任意のタイミングでFIFOにステータスデータ取得要求を書き込みます。

ステータスデータ要求送信呼び出しがFIFOを生成するため、こちらを先に起動します。

表7.5.5 ステータスデータ要求送信呼び出しプロセスフロー

No	処理	処理内容	次No
1	FIFO生成	FIFO書き込みとのインタフェース用FIFOを生成する。	
2	FIFOオープン	FIFOをオープンする。	
3	FIFO入力待ち	FIFO書き込みからのFIFOへの入力を待ち合わせる。	
4	ステータスデータ取得要求送信	FIFOから読み込んだステータスデータ取得要求を入力パラメータに指定し、以下の関数を呼び出してステータスデータ取得要求を送信する。 DAQtkGetSnd()	
5	結果出力	ステータスデータ要求送信関数の復帰値を出力する。	3

5) FIFO書き込み

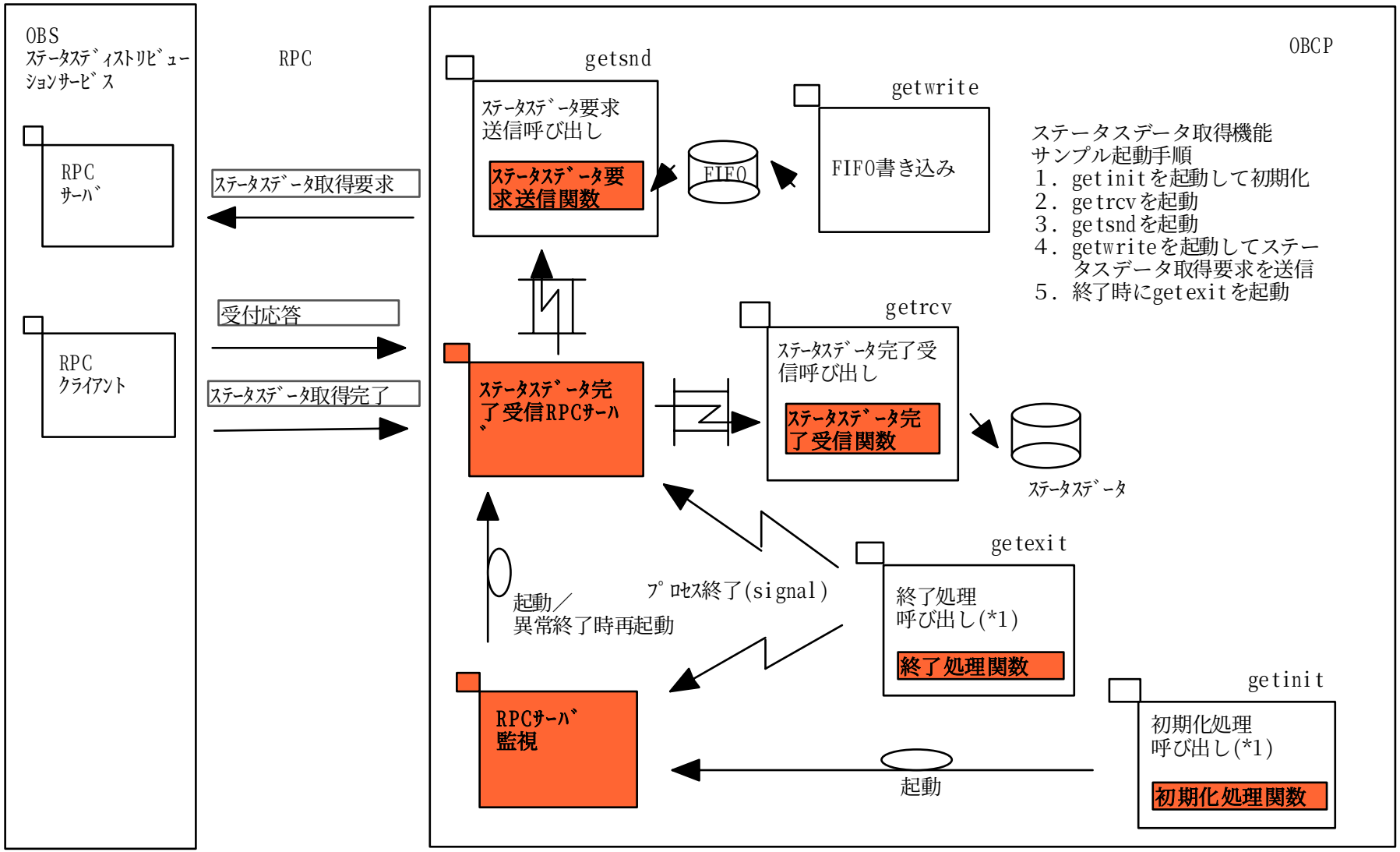
ステータスデータ要求送信呼び出しがFIFOを生成するため、ステータスデータ要求送信呼び出しを起動した後、ステータスデータ取得要求を送信するタイミングで起動します。

FIFOに書き込むステータスデータ取得要求、書き込み回数、および書き込み周期などをプログラム内の変数で指定していますので、これらを自由に設定して、makeしてから起動してください。

表7.5.6 FIFO書き込みプロセスフロー

No	処理	処理内容	次No
1	ステータスデータ取得要求編集	FIFOに書き込むステータスデータ取得要求を編集する。	
2	FIFO書き込みループ	指定回数だけ以下の処理を行う。 ・指定回数を超えた	 終
3	FIFOオープン	FIFOをオープンする。	
4	FIFO書き込み	編集したステータスデータ取得要求をFIFOに書き込む。	
5	FIFOクローズ	FIFOをクローズする。	
6	待ち合わせ	指定した秒数だけ待ち合わせる。	

ステータスデータ取得機能サンプルプログラム構成図を図7.5.1に示します。



*1 コマンドイメージで起動されるバッチプログラム

図7.5.1 ステータスデータ取得機能サンプルプログラム構成図

7.6 FITSファイル作成機能

FITSファイル作成機能におけるサンプルプログラム一覧を以下に示します。

表7.6.1 FITSファイル作成機能サンプルプログラム一覧

No	プログラム名	パス名
1	FITSヘッダ生成	.../daqtk/sample/src/makefits/testfits.c

1) FITSファイル作成

表7.6.2 FITSファイル作成プロセスフロー

No	処理	処理内容	次No
1	パラメタチェック	入力パラメタをチェックする。	
2	Primary ヘッダ項目設定	FITSヘッダ項目構造体に、Primaryヘッダ項目を設定する。	
3	WCSパラメータ設定	WCSパラメータ構造体に、WCSパラメータを設定する。	
4	ASCIIextension ヘッダ項目設定	FITSヘッダ項目構造体に、ASCIIextensionヘッダ項目を設定する。	
5	Primaryデータ設定	FITSデータ情報構造体に、Primaryデータ情報を設定する。	
6	ASCIIextension データ設定	FITSデータ情報構造体に、ASCIIextensionデータ情報を設定する。	
7	FITSヘッダ生成	以下の関数を呼び出し、FITSファイルを作成する。 DAQtkFitsMakeFile() 復帰値に応じて、以下のように処理する。 ・正常終了 ： そのまま終了する ・異常終了 ： 詳細コードを表示して終了する	

7.7 モニタ表示機能

モニタ表示機能におけるサンプルプログラム一覧を以下に示します。

表7.7.1 モニタ表示機能サンプルプログラム一覧

No	プログラム名	パス名
1	モニタ表示	.../daqtk/sample/src/monitor/testmonior.c

1) モニタ表示

表7.7.2 モニタ表示プロセスフロー

No	処理	処理内容	次No
1	パラメタチェック	入力パラメタをチェックする。	
2	FITSヘッダ項目設定	入力パラメタに応じてデータ表示画面を制御する	
		・起動	3
		・データ表示	4
		・終了	3
3	データ表示画面制御	以下の関数を起動/終了のパラメタで呼び出し、 データ表示画面を起動/終了する。 DAQtkMonitorCtl()	終
4	データファイル読込み	以下のバイナリデータファイルを読み込む。 $(DAQTKHOME)/sample/data/barray.dat	
5	データ表示	読み込んだデータをパラメタに設定して 以下の関数を呼び出し、 データ表示画面にデータを表示する。 DAQtkMonitorDsp()	終

7.8 FITSキーワード値算出機能

FITSキーワード値算出機能におけるサンプルプログラム一覧を以下に示します。

表7.8.1 FITSキーワード値算出機能サンプルプログラム一覧

No	プログラム名	パス名
1	UTC取得	.../daqtk/sample/src/func/utc/test-utc.c
2	LST/MJD取得	.../daqtk/sample/src/func/lst/test-lst.c
3	分点変換	.../daqtk/sample/src/func/equinox/test-eq.c
4	Pixel座標－天球座標変換	.../daqtk/sample/src/func/pix/test-pix.c
5	時刻変換	.../daqtk/sample/src/func/time/test-time.c
6	角度変換（赤経）	.../daqtk/sample/src/func/ra/test-ra.c
7	角度変換（赤緯）	.../daqtk/sample/src/func/dec/test-dec.c
8	マクロ呼び出し	.../daqtk/sample/src/func/macro/test-macro.c

1) UTC取得

表7.8.2 UTC取得プロセスフロー

No	処理	処理内容	次No
1	UTC取得	以下のUTC取得関数を呼び出す. DAQtkFuncGetUTC()	
2	結果表示	関数の出力を標準出力する.	

2) LST/MJD取得

表7.8.3 LST/MJD取得プロセスフロー

No	処理	処理内容	次No
1	LST/MJD取得	様々なパラメータについて以下のLST/MJD取得関数を呼び出す. DAQtkFuncGetLST()	
2	結果表示	関数の出力を標準出力する.	

3) 分点変換

表7.8.4 分点変換プロセスフロー

No	処理	処理内容	次No
1	分点変換	様々なパラメータについて以下の分点変換関数を呼び出す. DAQtkFuncChgEqx()	
2	結果表示	関数の出力を標準出力する.	

4) Pixel座標－天球座標変換

表7.8.5 Pixel座標－天球座標変換プロセスフロー

No	処理	処理内容	次No
1	Pixel-天球座標変換	様々なパラメータについて以下のPixel－天球座標変換関数を呼び出す. DAQtkFuncGetRADEC()	
2	結果表示	関数の出力を標準出力する.	

5) 時刻変換

表7.8.6 時刻変換プロセスフロー

No	処理	処理内容	次No
1	時刻変換	様々なパラメータについて以下の時刻変換関数を呼び出す. DAQtkFuncCTime2DHour() DAQtkFuncDHour2CTime()	
2	結果表示	関数の出力を標準出力する.	

6) 角度変換（赤経）

表7.8.7 角度変換（赤経）変換プロセスフロー

No	処理	処理内容	次No
1	角度変換（赤経）	様々なパラメータについて以下の角度変換関数（赤経）を呼び出す. DAQtkFuncCRA2DDeg() DAQtkFuncDDeg2CRA()	
2	結果表示	関数の出力を標準出力する.	

7) 角度変換（赤緯）

表7.8.8 角度変換（赤緯）変換プロセスフロー

No	処理	処理内容	次No
1	角度変換（赤緯）	様々なパラメータについて以下の角度変換関数（赤緯）を呼び出す. DAQtkFuncCDec2DDeg() DAQtkFuncDDeg2CDec()	
2	結果表示	関数の出力を標準出力する.	

8) マクロ呼び出し

表7.8.9 マクロ呼び出しプロセスフロー

No	処理	処理内容	次No
1	マクロ呼び出し	様々なパラメータについて以下のマクロを呼び出す. DAQtkFuncDDeg2DRad() DAQtkFuncDRad2DDeg() DAQtkFuncDDeg2DHour() DAQtkFuncDHour2DDeg()	
2	結果表示	関数の出力を標準出力する.	

8. インタフェース規約

8.1 コマンド通信機能

スケジューラモード時のスケジューラとの装置依存コマンドインタフェースについて補足する.
関連するライブラリは以下の通りである.
・DAQtkCmdRcv()
・DAQtkCmdSnd()

制御コマンド受信ライブラリ(DAQtkCmdRcv)における出力パラメータ
・qRpcCmd->cCmdCode
また，処理完了送信ライブラリ(DAQtkCmdSnd)における入力パラメータ
・qRpcRslt->cRslt
・qRpcRslt->cRsltPara
について補足する.

qRpcCmd->cCmdCodeにはスケジューラから受信したコマンドのデータ部（装置依存コマンド）が格納される.
装置依存コマンドの例を8.1.1に示す.

qRpcRslt->cRsltにはスケジューラに送信する完了応答電文の処理結果を格納する.
処理結果の項目説明を表8.1に示す.

表8.1　ＲＰＣコマンド型インタフェースの完了応答電文の処理結果の項目説明

小項目名	内容
処理結果	処理結果を以下のコードで指定する. ”　　０”　：　正常（固定） それ以外　　：　異常（ユーザが定義する）

qRpcRslt->cRsltParaにはスケジューラに送信する完了応答電文のデータ部を格納する.
データ部の内容はユーザが定義する.

8.1.1 装置依存コマンド例

装置依存コマンドとは，望遠鏡や観測装置などの制御を行うためのコマンドである．
装置依存コマンドの例を以下に示すので参考とされたい．
装置依存コマンドは観測所またはそれぞれの観測装置の開発グループが提供するものなので，
コマンドの仕様や機能については本書では扱わない．

「例」（装置依存コマンド）

```
--------------------------------------------------------------------------------
EXEC  SUKA  CAMERA  P1=ON P2=[1 2 3]
EXEC              動詞（EXEC：エクゼキュート，SET：セット）（必須）
                  （READ：リードは将来機能として予約されている）
SUKA              装置名（コマンドの発行対象の装置）（必須）
CAMERA            オブジェクト（コマンドの作用対象の名前）（必須）
P1,P2,P3.....  パラメタ（装置依存コマンド仕様に依存）
各項目のデリミタは空白またはＴＡＢである．（複数可）
--------------------------------------------------------------------------------
```

8.2 透過モード・コマンド通信機能

透過モード時のエグゼキュータとの装置依存コマンドインタフェースについて補足する.
関連するライブラリは以下の通りである.
・DAQtkThroughSnd()
・DAQtkThroughRcv()

制御コマンド送信ライブラリ(DAQtkThroughSnd)における出力パラメータ
・qRpcCmd->cCmdCode
また, 処理完了受信ライブラリ(DAQtkThroughRcv)における入力パラメータ
・qRpcRslt->cRslt
・qRpcRslt->cRsltPara
について補足する.

qRpcCmd->cCmdCodeにはエグゼキュータに送信するコマンドのデータ部（装置依存コマンド）を格納する.
装置依存コマンドの例は8.1.1を参照のこと.

qRpcRslt->cRsltにはエグゼキュータから受信した完了応答電文の処理結果が格納される.
処理結果の項目説明を表8.2.1に示す.

表8.2.1　ＲＰＣコマンド型インタフェースの完了応答電文の処理結果の項目説明

小項目名	内容
処理結果	処理結果が以下のコードで指定される ”　　０”　：　正常 ”　　１”　：　異常

処理結果が異常（コマンド実行がエラー）の場合, DAQtkThroughRcv()ライブラリは異常復帰する.

qRpcRslt->cRsltParaにはスケジューラから受信した完了応答電文のデータ部が格納される.
完了応答電文のデータ部の仕様を8.2.1に示す.

8.2.1　完了応答仕様

本項では，ＲＰＣ完了応答電文のデータ部の完了応答本文についての仕様を記述する．

表8.2.1-1にコマンド一覧を示す．

表8.2.1-1　完了応答一覧

完了応答	概要
正常	コマンドに対応する処理を正常に完了した．
コマンドコードエラー	コマンドコードが不当である．
パラメタエラー	コマンドのフォーマットが不当であるか，またはパラメタの値が不当である．
スケジューラ動作モードエラー	現在，スケジューラ動作モードで指定したモードでの動作が不可能である．
使用装置重複エラー	装置コードで指定した装置が既に使用中である．
開始宣言未発行	開始宣言が発行されていない．または，開始宣言で利用宣言していない装置にコマンドを発行しようとした．
ビジー	コマンドを受け付けられる状態でない．終了宣言の場合は，未完了の装置依存コマンドがあるの意味となる．
抽象化コマンド実行エラー	抽象化コマンドの実行で異常が発生した．
装置依存コマンド実行エラー	装置依存コマンドの実行で異常が発生した．
デコードエラー	デコードで異常が発生した．
キャンセル	抽象化コマンド，または装置依存コマンドがキャンセルされた．
システムエラー	スケジューラサブシステム内のエラー

（1）正常

（a）概要
コマンドに対応する処理を正常に完了した.

（b）完了応答フォーマット

| COMPLETE | , | パラメタ1 | , | パラメタ2 | , | パラメタ3 |

| , | パラメタ4 |

パラメタ1：利用者のホスト名　　（8バイト左詰め，余り空白）
パラメタ2：利用者のプロセスID（8バイト左詰め，余り空白）
パラメタ3：コマンドID　　（40バイト左詰め，余り空白）
パラメタ4：ディスパッチャ番号　（1バイト，0～6）
注）パラメタ3は，抽象化コマンド実行，および装置依存コマンド実行の場合にのみ設定する．その他の場合は，空白を設定する．
注）パラメタ4は，要求を受け付けたディスパッチャの系の番号を返す．
0が割込ディスパッチャで1～6がディスパッチャ1～6に対応する．

（2）コマンドコードエラー

（a）概要
コマンドコードが不当である.

（b）完了応答フォーマット

| ERROR | , | COMMAND |

（3）パラメタエラー

（a）概要
コマンドのフォーマットが不当であるか，または指定されたコードが不当である.

（b）完了応答フォーマット

| ERROR | , | PARAMETER |

（4）スケジューラ動作モードエラー

（a）概要
現在，スケジューラ動作モードで指定したモードでの動作が不可能である．

（b）完了応答フォーマット

ERROR	,	OP_MODE

（5）使用装置重複エラー

（a）概要
装置コードで指定した装置が既に使用中である．

（b）完了応答フォーマット

ERROR	,	UNIT	,	装置コード

装置コード：開始宣言コマンドで指定された装置コードのうち，使用中の装置コードを設定する．（8バイト左詰め，余り空白）

（6）開始宣言未発行

（a）概要
開始宣言が発行されていない．または，開始宣言で利用宣言していない装置にコマンドを発行しようとした．

（b）完了応答フォーマット

ERROR	,	ALLOC

（7）ビジー

（a）概要
コマンドを受け付けられる状態でない．
終了宣言の場合は，未完了の装置依存コマンドがある．

（b）完了応答フォーマット

BUSY

（8）抽象化コマンド実行エラー

（a）概要
抽象化コマンドの実行で異常が発生した.

（b）完了応答フォーマット

ERROR	,	EXEC__A	,	コマンドＩＤ	,	装置コード

コマンドＩＤ：エラーが発生した抽象化コマンド，または装置依存コマンドのコマンドＩＤを設定する．（４０バイト左詰め，余り空白）
装置コード　：開始宣言コマンドで指定された装置コードのうち，使用中の装置コードを設定する．（８バイト左詰め，余り空白）

（9）装置依存コマンド実行エラー

（a）概要
装置依存コマンドの実行で異常が発生した.

（b）完了応答フォーマット

ERROR	,	EXEC__D	,	コマンドＩＤ

コマンドＩＤ：エラーが発生した抽象化コマンド，または装置依存コマンドのコマンドＩＤを設定する．（４０バイト左詰め，余り空白）

ERROR	,	EXEC__I	,	コマンドＩＤ

コマンドＩＤ：エラーが発生した抽象化コマンド，または装置依存コマンドのコマンドＩＤを設定する．（４０バイト左詰め，余り空白）

（１０）デコードエラー

（a）概要
デコードで異常が発生した.

（b）完了応答フォーマット

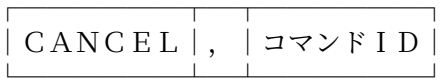
ERROR	,	DECODE	,	コマンドＩＤ

コマンドＩＤ：エラーが発生した抽象化コマンド，または装置依存コマンドのコマンドＩＤを設定する．（４０バイト左詰め，余り空白）

（１１）キャンセル

（ａ）概要
抽象化コマンド，または装置依存コマンドがキャンセルされた．

（ｂ）完了応答フォーマット

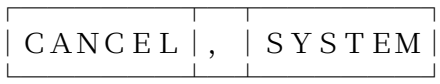


コマンドＩＤ：エラーが発生した抽象化コマンド，または装置依存コマンドのコマンドＩＤを設定する．（４０バイト左詰め，余り空白）

（１２）システムエラー

（ａ）概要
スケジューラサブシステム内でのエラー．
コンソールメッセージを参照する。

（ｂ）完了応答フォーマット



コマンドＩＤ：エラーが発生した抽象化コマンド，または装置依存コマンドのコマンドＩＤを設定する．（４０バイト左詰め，余り空白）

8.3 ステータス転送機能

ステータスロガーへのステータスデータ転送インタフェースについて補足する.
関連するライブラリは以下の通りである.
・DAQtkStatSnd()

ステータス転送ライブラリ(DAQtkStatSnd)における入力パラメータ
・qRpcStat->cStat
について補足する.

qRpcStat->cStatにはステータスロガーに転送するステータスのデータ部を格納する.
ステータスデータ部の仕様を8.3.1に示す.

8.3.1 ステータスデータ部仕様

本項では，ＲＰＣステータス電文のデータ部の仕様を記述する.

ＲＰＣステータス型インタフェースで扱う電文は，全て文字型のデータからなる.

データ部の基本フォーマットを図8.3.1-1に示す.

ステータス名，ステータスデータから構成される.
ステータス名は，8byteとする.
ステータスデータは可変長とするが，データ部は最大8192byteである.

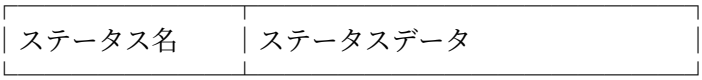


図8.3.1-1　データ部基本フォーマット

データ部の項目説明を表8.3.1-1に示す.

表8.3.1-1　ＲＰＣステータス型インタフェースのデータ部基本フォーマット項目説明

小項目名	内容
ステータス名	ステータスを一意に識別する名前 左詰めで余りは空白
ステータス本文	全て文字とする 各ステータスの仕様に依存する

8.4 取得データ転送機能

OBCへの取得データ転送インタフェースについて補足する.
関連するライブラリは以下の通りである.
・DAQtkDatSnd()

取得データ転送ライブラリ(DAQtkDatSnd)における入力パラメータ
・qRpcDat->cFrame
について補足する.

qRpcDat->cFrameにはOBCに転送するフレーム番号を格納する.
フレーム番号の命名規約を8.4.1, フレーム番号獲得方法を8.4.2に示す.

8.4.1 フレーム番号の命名規約

フレーム番号は, すばる望遠鏡で撮像したフレーム（画像）, および簡易処理した結果のフレームに対して付けられるユニークな名称である.

フレーム番号は, 観測制御システムにより観測装置毎にユニークな名称が発行される. フレーム番号の命名規約を以下に示す.

フレーム番号の命名規約

フォーマット ：	ＡＡＡＢ９９９９９９９９	
	ＡＡＡ	観測装置につけられたコード（３文字）
	Ｂ	ＡまたはＱ（１文字）
		Ａ：取得データ
		Ｑ：簡易処理済みデータ
	９９９９９９９９	通番（８桁）

現在，観測装置に対して付けられているコードの一覧を以下に示す.

装置名	コード
IRCS	IRC
AO	AOS
CIAO	CIA
OHS	OHS
FOCAS	FCS
HDS	HDS
COMICS	COM
SPCAM	SUP
SUKA	SUK
MIRTOS	MIR
VTOS	VTO
CAC	CAC
OTHER13	SKY
PI1	PI1
PI2	PI2
OTHER16	016
OTHER17	017
OTHER18	018
OTHER19	019
OTHER20	020
OTHER21	021
OTHER22	022
OTHER23	023
OTHER24	024
OTHER25	025
OTHER26	026
OTHER27	027
OTHER28	028
OTHER29	029
OTHER30	030
OTHER31	031
OTHER32	032
VGW	VGW

＊）装置名は８文字以内．コードは３文字.

8.4.2 フレーム番号の獲得方法

フレーム番号を獲得するには，以下の２つの方法がある．

（１）装置依存コマンドのパラメタとして獲得する

装置依存コマンドのパラメタの値にフレーム番号獲得を記述することにより，装置に通知する．

＜１つ獲得＞
装置依存コマンドのパラメタ値の記述は，「&GET_F_NO[SUKA A]」の形式で行う．
上記の例だと装置名がSUKAで取得データ用のフレーム番号を要求している．このときまでに「SUKA000000026」まで使われていたとすると，「SUKA00000027」が獲得される．（SUKAでAの00000027）もう１回同じ記述でフレーム番号を獲得すると次は「SUKA00000028」が獲得される．このように，通番の部分が１つづつカウントアップしていく．

＜複数の獲得＞
複数のフレーム番号を一度に行いたい場合，装置依存コマンドのパラメタ値の記述は，「&GET_F_NO[SUKA A 3]」の形式でも行える．
上記の例だと装置名がSUKAで取得データ用のフレーム番号を３つ要求している．このときまでに「SUKA000000026」まで使われていたとすると，「SUKA00000027:0003」が獲得される．（SUKAでAの00000027から３つ．コロンの後にその個数が記される．）もう１回同じ記述でフレーム番号を獲得すると次は「SUKA00000030:0003」が獲得される．このように，通番の部分が指定した分だけカウントアップしていく．

（２）透過モードでフレーム番号を獲得する

観測時に観測制御システムの観測操作機能から透過モードとすると，観測装置に対して透過モードへの移行を通知するコマンドが送信される．このコマンドにより観測装置が使用して良いフレーム番号の開始番号を通知する．観測装置はこのフレーム番号以降の番号を使用して観測データを保存する．
観測装置は透過モードでの観測が終了したら，最後に使用したフレーム番号を観測制御システムに返さなくてはならない．この場合は，観測制御システムにＯＢＳの装置依存コマンドとして通知しなくてはならない．この通知を行わない場合，フレーム番号がユニークで無くなり観測データを保存できなくなる．

以下に観測制御システムと観測装置との間でやり取りされる装置依存コマンドの例を示す．例は，観測装置がSUKAの場合として記述している．

「例」（観測装置「SUKA」の場合の装置依存コマンドの例）

○透過モードの開始の時（観測制御システム－－＞観測装置）
（Ａが５０番，Ｑが１１番から使って良い場合）

```
EXEC SUKA SCHEDULERMODE MOTOR=ON \
        MODE=OBCP FRAME=SUKA00000050 QFRAME=SUKQ00000011
```

というコマンドが観測装置に通知される．
MODE=OBCPが透過モードの開始を示す

○透過モードの終わりの時（観測装置－－＞観測制御システム）
（Ａが９９番まで使って，Ｑは使わなかった場合）

```
EXEC OBS SCHEDULERMODE MOTOR=ON \
        MODE=OBS FRAME=SUKA00000099 QFRAME=NOP
```

というコマンドをＯＢＳに通知する．
MODE=OBSが透過モードの終了を示す

8.5 ステータスデータ取得機能

ステータスディストリビューションサービスとのコマンドインタフェースについて補足する．
関連するライブラリは以下の通りである．
・DAQtkGetSnd()
・DAQtkGetRcv()

ステータスデータ要求送信ライブラリ(DAQtkGetSnd)における入力パラメータ
・qRpcCmd->cCmdCode
また，ステータスデータ完了受信ライブラリ(DAQtkGetRcv)における出力パラメータ
・qRpcRslt->cRslt
・qRpcRslt->cRsltPara
について補足する．

qRpcCmd->cCmdCodeにはステータスディストリビューションサービスに送信するコマンドのデータ部を格納する．
コマンドのデータ部の仕様を8.5.1に示す．

qRpcRslt->cRsltにはステータスディストリビューションサービスから受信した完了応答電文の処理結果が格納される．．
処理結果の項目説明を表8.5-1に示す．

表8.5-1　ＲＰＣコマンド型インタフェースの完了応答電文の処理結果の項目説明

小項目名	内容
処理結果	処理結果が以下のコードで指定される ”　　０”　：　正常 ”　　１”　：　異常

処理結果が異常（ステータス獲得がエラー）の場合，DAQtkGetRcv()ライブラリは異常復帰する．

qRpcRslt->cRsltParaにはステータスディストリビューションサービスから受信した完了応答電文のデータ部が格納される．
完了応答電文のデータ部の仕様を8.5.2に示す．

8.5.1　コマンド仕様

本項では，ＲＰＣコマンド電文のデータ部の仕様，および正常時の完了応答の仕様を記述する．

図8.5.1-1にコマンド基本フォーマットを示す．

コマンドコード，パラメタから構成される．
コマンドコードとパラメタの間は必ずカンマで区切る．
コマンドコードは，可変長とする．
パラメタは，可変長とする．パラメタの個数，および内容は，各コマンドコード毎の仕様に依存する．

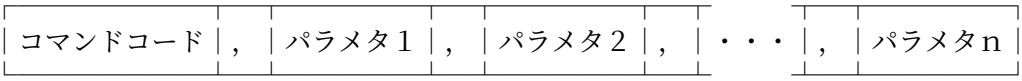


図8.5.1-1　コマンド基本フォーマット

表8.5.1-1にコマンド種別を示す．

表8.5.1-2にコマンド一覧を示す．

表8.5.1-1　コマンド種別

コマンド種別	該当するコマンド	概要
ステータス要求	ステータス要求	ステータス情報を即時に返す
イベント要求	オートガイダー誤差瞬間値通知 オートガイダー誤差平均値通知 シーイング状態通知 風速通知 湿度通知 温度通知	ステータスがある条件になったら通知する
特殊要求	オートガイダー誤差の 平均値と偏差値 環境諸値の平均値	ステータスの統計計算結果を通知する

表8.5.1-2　コマンド一覧

コマンド	概要
ステータス要求 （ＳＴＡＴＵＳ）	ステータス要求
オートガイダー誤差 瞬間値通知 （Ｅ＿ＡＧＥ＿Ｒ）	オートガイダー誤差瞬間値によるイベント要求
オートガイダー誤差 平均値通知 （Ｅ＿ＡＧＥ＿Ａ）	オートガイダー誤差平均値によるイベント要求
シーイング状態通知 （Ｅ＿ＳＥＥＩＮＧ）	シーイング状態によるイベント要求
風速通知 （Ｅ＿ＷＩＮＤＳ）	風速によるイベント要求
湿度通知 （Ｅ＿ＨＵＭＩＤ）	湿度によるイベント要求
温度通知 （Ｅ＿ＴＥＭＰ）	温度によるイベント要求
オートガイダー誤差の 平均値と偏差値 （Ｓ＿ＡＧ＿Ｅ）	オートガイダー誤差による特殊要求
環境諸値の平均値 （Ｓ＿ＥＮＶ）	環境諸値による特殊要求

注）括弧内はコマンドコード.

（1）ステータス要求（ＳＴＡＴＵＳ）

（a）概要

ステータスの内容を即時に返す.
返されるステータスは，文字列に変換されたものとなる.

（b）パラメタ

番号	名称	サイズ	概要
1	ステータス	任意	ステータスエイリアス，またはステータス位置定義 複数指定時は空白で区切って記述

ステータスの記述方法としては，ステータス位置定義とエイリアスを使う方法がある.
エイリアスを使う場合は，エイリアス名の先頭に『＄』文字を付加する.

（c）完了応答

送出タイミング：処理完了時，または異常発生時.

・正常

ＣＯＭＰＬＥＴＥ	,	ステータス情報

ステータス情報：
要求されたステータスをエイリアス，またはステータス位置定義で置換した値を
設定する．複数要求された場合は，空白で区切って設定する.

＊＊＊また，装置開発者用に定義されたステータスエイリアス（エイリアス名：FITS.XXX.～）
は６４byteの固定長で値を設定する.
よって複数要求された場合は，６４byteごとに空白で区切って設定する.

・パラメタエラー
・コマンドコードエラー
・エイリアス定義エラー
・ステータスがない

（2）オートガイダー誤差瞬間値通知（Ｅ＿ＡＧＥ＿Ｒ）

（a）概要

オートガイダー誤差（瞬間値）が条件値以上の場合の通知.

（b）パラメタ

番号	名称	サイズ	概要
1	監視時間	任意	監視時間（秒単位で１～８６４００）
2	条件値	任意	条件値（絶対値で０．０～９．９９９９９９ＤＥＧ）

（c）完了応答

送出タイミング：条件達成時，または異常発生時.

・正常

```
｜COMPLETE｜
```

・タイムアウト
・パラメタエラー
・コマンドコードエラー

（d）タイムアウト

監視時間＋５秒

（3）オートガイダー誤差平均値通知（Ｅ＿ＡＧＥ＿Ａ）

（ａ）概要

オートガイダー誤差（平均値）が条件値以上の場合の通知.

（ｂ）パラメタ

番号	名称	サイズ	概要
１	監視時間	任意	監視時間（秒単位で１～８６４００）
２	条件値	任意	条件値（絶対値で０．０～９．９９９９９９ＤＥＧ）

（ｃ）完了応答

送出タイミング：条件達成時，または異常発生時.

・正常

```
｜COMPLETE｜
```

・タイムアウト
・パラメタエラー
・コマンドコードエラー

（ｄ）タイムアウト

監視時間＋５秒

（4）シーイング状態通知（E＿ＳＥＥＩＮＧ）

（a）概要

シーイング状態が条件値以上の場合の通知.

（b）パラメタ

番号	名称	サイズ	概要
1	監視時間	任意	監視時間（秒単位で１～８６４００）
2	条件値	任意	条件値（－９９９９．９～＋９９９９．９ａｒｃｓｅｃ）

（c）完了応答

送出タイミング：条件達成時，または異常発生時.

・正常

```
｜COMPLETE｜
```

・タイムアウト
・パラメタエラー
・コマンドコードエラー

（d）タイムアウト

監視時間＋５秒

（5）風速通知（E_WINDS）

（a）概要

風速が条件値以上の場合の通知.

（b）パラメタ

番号	名称	サイズ	概要
1	監視時間	任意	監視時間（秒単位で1～86400）
2	条件値	任意	条件値（0.0～99.9m／s）

（c）完了応答

送出タイミング：条件達成時，または異常発生時.

・正常

```
|COMPLETE|
```

・タイムアウト
・パラメタエラー
・コマンドコードエラー

（d）タイムアウト

監視時間＋5秒

（6）湿度通知（E＿HUMＩD）

（a）概要

湿度が条件値以上の場合の通知.

（b）パラメタ

番号	名称	サイズ	概要
1	監視時間	任意	監視時間（秒単位で1～86400）
2	条件値	任意	条件値（0．0～99．9%）

（c）完了応答

送出タイミング：条件達成時，または異常発生時.

・正常

```
|COMPLETE|
```

・タイムアウト
・パラメタエラー
・コマンドコードエラー

（d）タイムアウト

監視時間＋5秒

（7）温度通知（E＿TEMP）

（a）概要

温度が条件範囲値以上または以下の場合の通知.

（b）パラメタ

番号	名称	サイズ	概要
1	監視時間	任意	監視時間（秒単位で１～８６４００）
2	上限値	任意	上限値（－９９９．９～＋９９９．９C）
3	下限値	任意	下限値（－９９９．９～＋９９９．９C）

（c）完了応答

送出タイミング：条件達成時，または異常発生時.

・正常

```
｜COMPLETE｜
```

・タイムアウト
・パラメタエラー
・コードエラー

（d）タイムアウト

監視時間＋５秒

（8）オートガイダー誤差の平均値と偏差値（Ｓ＿ＡＧ＿Ｅ）

（ａ）概要

オートガイダー誤差信号の平均値と偏差値を通知する.
応答は，即時である.

（ｂ）パラメタ

番号	名称	サイズ	概要
1	シーケンシャル番号	8	ＳＴＡＲＴサブコマンド時は全て空白 ＲＥＡＤ，ＳＴＯＰサブコマンド時はＳＴＡＲＴサブコマンドのＲＰＣ電文のシーケンシャル番号
2	監視時間	任意	監視時間（秒単位で１～８６４００） ただし，ＲＥＡＤ，ＳＴＯＰサブコマンド時は無視 ＳＴＡＲＴサブコマンド受付からの統計計算終了するまでの時間である．つまり，ＳＴＯＰサブコマンドを受信しなくても監視時間経過すると要求は削除される.
3	サブコマンド	任意	サブコマンドを以下のコードで指定する ＳＴＡＲＴ：情報の取得開始を要求 ＲＥＡＤ　：情報の途中結果の通知を要求 ＳＴＯＰ　：情報の取得終了を要求

（ｃ）完了応答

送出タイミング：処理完了時，または異常発生時.

・正常（ＳＴＡＲＴサブコマンド）

ＣＯＭＰＬＥＴＥ

・正常（ＲＥＡＤ，ＳＴＯＰサブコマンド）

ＣＯＭＰＬＥＴＥ	,	平均値	,	偏差値

・パラメタエラー
・コマンドコードエラー
・要求シーケンスエラー

（9）環境諸値の平均値（S＿ENV）

（a）概要

温度，湿度，風速，風向，気圧，透明度，シーイングの平均値を通知する．
応答は，即時である．

（b）パラメタ

番号	名称	サイズ	概要
1	シーケンシャル番号	8	ＳＴＡＲＴサブコマンド時は全て空白 ＲＥＡＤ，ＳＴＯＰサブコマンド時はＳＴＡＲＴサブコマンドのＲＰＣ電文のシーケンシャル番号
2	監視時間	任意	監視時間（秒単位で１～８６４００） ただし，ＲＥＡＤ，ＳＴＯＰサブコマンド時は無視 ＳＴＡＲＴサブコマンド受付からの統計計算終了するまでの時間である．つまり，ＳＴＯＰサブコマンドを受信しなくても監視時間経過すると要求は削除される．
3	サブコマンド	任意	サブコマンドを以下のコードで指定する ＳＴＡＲＴ：情報の取得開始を要求 ＲＥＡＤ　：情報の途中結果の通知を要求 ＳＴＯＰ　：情報の取得終了を要求

（c）完了応答

送出タイミング：処理完了時，または異常発生時．

・正常（ＳＴＡＲＴサブコマンド）

ＣＯＭＰＬＥＴＥ

・正常（ＲＥＡＤ，ＳＴＯＰサブコマンド）

ＣＯＭＰＬＥＴＥ	,	温度平均値	,	湿度平均値	,	風速平均値	,

風向平均値	,	気圧平均値	,	透明度平均値	,	シーイング平均値

・パラメタエラー
・コマンドコードエラー
・要求シーケンスエラー

8.5.2　完了応答仕様

本項では，ＲＰＣ完了応答電文のデータ部の完了応答本文についての仕様を記述する．

表8.5.2-1に完了応答一覧を示す．

表8.5.2-1　完了応答一覧

完了応答	概要
正常	コマンドに対応する処理を正常に完了した．
タイムアウト	監視時間に達したが，イベント条件にならなかった．
コマンドコードエラー	コマンドコードが不当である．
パラメタエラー	コマンドのフォーマットが不当であるか，またはパラメタの値が不当である．
エイリアスが定義エラー	エイリアスが定義されていなかった．
ステータスがない	ステータステーブルがなかった．
要求シーケンスエラー	ＳＴＡＲＴを受けていないのに，ＲＥＡＤ，ＳＴＯＰが来た．
リセット	起動時または終了時に特殊要求テーブルに要求が残っていたものに対してリセットを通知する．

（1）正常

（a）概要
コマンドに対応する処理を正常に完了した.

（b）完了応答フォーマット

| COMPLETE | , | パラメタ1 | , | パラメタ2 | ～ | パラメタn |

パラメタ1～n：コマンドに応じて変わる．各コマンドの仕様を参照せよ.

（2）タイムアウト

（a）概要
監視時間に達したが，イベント条件にならなかった.

（b）完了応答フォーマット

| TIMEOUT |

（3）コマンドコードエラー

（a）概要
コマンドコードが不当である.

（b）完了応答フォーマット

| ERROR | , | COMMAND |

（4）パラメタエラー

（a）概要
コマンドのフォーマットが不当であるか，または指定されたコードが不当である.

（b）完了応答フォーマット

| ERROR | , | PARAMETER |

（5）エイリアスが定義エラー

（a）概要
エイリアスが定義されていなかった.

（b）完了応答フォーマット

ERROR	,	ALIAS

（6）ステータスがない

（a）概要
ステータステーブルがなかった.

（b）完了応答フォーマット

ERROR	,	STATUS

（7）要求シーケンスエラー

（a）概要
コマンドのシーケンスが誤っている.
以下のコマンドにおいて，ＳＴＡＲＴを受けていないのに，ＲＥＡＤ，ＳＴＯＰが来た場合のエラーである.
・オートガイダー誤差の平均値と偏差値
・環境諸値の平均値

（b）完了応答フォーマット

ERROR	,	SEQUENCE

（8）リセット

（a）概要
起動時または終了時に特殊要求テーブルに要求が残っていた要求に対してリセットを通知する.

（b）完了応答フォーマット

ERROR	,	RESET

9. STARS登録チェックツール

本ツールキットには，装置開発者が作成したFITSファイルがSTARSに登録可能であるか否かのチェックを行う「STARS登録チェックツール」が含まれています．

「STARS登録チェックツール」の使用方法について，以下にご説明いたします．

9.1 概要

STARS登録チェックツールは，装置開発者が事前にFITSヘッダの誤りを認識するために，STARS登録のチェックを行うツールです．本ツールは，データ取得ツールキット(daqtk)の一機能として提供されています．

9.2 利用手順

「STARS登録チェックツール」のコマンドの利用手順について，以下にご説明いたします.

9.2.1 コマンド仕様

```
% DAQtkStarsCheck (FITS file name) (Conf file name)
```

コマンド名　：　DAQtkStarsCheck
第１引数　　：　FITSファイル名
　　チェックを行うFITSファイル名を指定します.
第２引数　　：　登録項目ファイル名
　　STARS登録項目を設定したファイル名を指定します.
戻り値　　：　チェック結果によらず，チェックが終了した場合は０
　　チェック処理がエラーで中断した場合は１
標準出力　　：　チェック結果が標準出力されます.

9.2.2 標準出力仕様

コマンドの実行により，STARS登録のチェック結果が標準出力に出力されます.
出力される項目は以下のとおりです.

１）STARS登録が可能であるか否かについて

STARSへの登録が可能な場合は以下のメッセージが出力されます.

```
<< REGIST TO STARS DB >>
Result  : OK
```

STARSへの登録がエラーになる場合は以下のメッセージが出力されます.

```
<< REGIST TO STARS DB >>
Result  : NG
```

２）各FITSヘッダ項目の問題点について

STARSへの登録に問題があるＦＩＴＳヘッダ項目について，以下のメッセージが出力されます.

<< FITS KEYWORD CHECK >>

・文字列の入るべき項目に数字が入っている場合

Warning : String is required to the value. String [999] is registered to STARS.

上記メッセージが出力されたFITSヘッダ項目は，文字列の入るべき項目に数値が書かれていますが，数値は文字列として登録されます.

・数字の入るべき項目に文字列が入っている場合

Warning : Numeric is required to the value. Zero value [0.0] is registered to STARS.

上記メッセージが出力されたFITSヘッダ項目は，数値の入るべき項目に文字列が書かれているため，登録時には０の数値が登録されます.

・数字において小数点以下の桁があふれている場合

Warning : The value is registerd as the following format [9.99].

上記メッセージが出力されたFITSヘッダ項目は，登録は可能ですが，検索の際に，小数点以下の桁が登録時よりも少なくなる可能性があります.

・数字において整数部の桁があふれている場合

Fatal Error : The value should be smaller than [1000].

上記メッセージが出力されたFITSヘッダ項目は，STARSへの登録の際に整数部の桁があふれているため，登録処理がエラーとなります.

・プロポーザルIDのフォーマットが規格外の場合

Error : Proposal ID format error.

上記メッセージは，プロポーザルIDのフォーマットが異常であることを示しています．STARSへの登録は可能ですが，検索時に問題があります.

また，各メッセージの後に，以下のように，STARS DB登録項目名，
FITSキーワード名，およびFITSヘッダ値の情報も同時に出力されます.

```
---> STARS keyword      : [（STARS DB登録項目名）]
---> FITS  keyword      : [（FITSキーワード名）]
---> FITS  header value : [（FITSヘッダ値）]
```

３）STARSでの天球座標変換の計算について

STARSでの天球座標変換の計算に問題がある場合に，以下のメッセージが出力されます．
STARSではRA,DEC,EQUINOX,UT,DATE-OBSの必須キーワードから，
RASEC,DECSEC,GALLONG,GALLAT,ECLLONG,ECLLAT,X_2000,Y_2000,Z_2000の
キーワードを計算して登録を行います．

```
<< RA,DEC,EQUINOX,UT,DATE_OBS CHECK >>
Error    : Common keywords cannot be found in the header.
 ---> FITS keyword        : [（FITSキーワード名）]
        ・・・・                ・・・
        ・・・・                ・・・
```

上記メッセージは，STARS登録時に天球座標変換の計算を行う際に必須なFITSヘッダ項目がないことを示しています．このメッセージが出力された場合，登録は可能ですが，変換座標の項目は登録されません．
上記で表示されるFITSキーワード名は，座標変換に必要なFITSキーワードで，
FITSヘッダになかったものを示しています．

４）FITSヘッダに項目がない場合について

STARS登録項目として定義されているが，FITSヘッダに項目がなかった場合に，
以下のメッセージが出力されます．

```
<< NOT REGISTERD KEYWORDS >>
Warning : STARS keyword cannot be found in the header.
 ---> STARS keyword       : [（STARS DB登録項目名）]
        ・・・・                ・・・
        ・・・・                ・・・
```

9.3 登録項目ファイルについて

本ファイルはSTARSへの登録項目を設定するチェックのもととなるファイルで，
コマンド実行の際に第２引数で指定します．
STARSへの登録は観測装置ごとに異なったFITSヘッダ項目，およびフォーマット
で行われるため，この設定ファイルもまた観測装置ごとに内容が異なります．

ツールキットでは，以下の場所に各観測装置について2000年9月11日時点での最新
サンプルを含めます．

$DAQTKHOME/sample/data/CheckXXX.conf　（XXXは装置略称）

STARS登録の仕様に変更があった場合は，ツールキットにおけるこの設定ファイルを
更新する必要があります．
この場合，s03におけるhttpアクセスにより最新版を入手することが可能であるため，
装置開発者での更新をお願いします．

9.4 その他

1）必須キーワードのチェックについては既存のFITSファイル作成ツールを使用してください．なお，すばる必須キーワードをチェックする「FITSキーワード設定ファイル」は以下の場所に格納されています．

```
$DAQTKHOME/dat/SubaruCommonFits.conf
```