747

YOHKOH

DATA FOR ALL INSTRUMENTS (EXPERIMENTS)

91-062A-01A, -02B, -03A, -04A

YOHKOH

DATA FOR ALL INSTRUMENTS

91-062A-01A, 02B, 03A, 04A

These data sets consist of 132 magnetic tapes. The 8mm tapes are written in binary, low density. Each tape has 2 header files (the first is written in ASCII and the second is binary) followed by a variable number of CPIO (data) files and 1 software file. The software file is required to read the tapes. During ingest, some of the input tapes did not have a double end-of-file written on them. This DEOF was added to the C copy during the processing but was not added to the D tape.

The D and C numbers are listed below along with the tape name, number of files per tape and the timespans of each.

DD#	DC#	TAPE NAME	FILES	TIMESPAN
DD 104025	DC 031401	91_36A	64	09/01/91-09/07/91
DD 104026	DC 031402	91_37A	67	09/08/91-09/14/91
DD 104027	DC 031403	91_38A	66	09/15/91-09/21/91
DD 104028	DC 031404	91_39A	76	09/22/91-09/28/91
DD 104029	DC 031405	91_40A	57	09/29/91-10/05/91
DD 104030	DC 031406	91_41A	71	10/06/91-10/12/91
DD 104031	DC 031407	91_42A	64	10/13/91-10/19/91
DD 104032	DC 031408	91_43A	71	10/20/91-10/26/91
DD 104033	DC 031409	91_44A	85	10/27/91-11/02/91
DD 104034	DC 031410	91_45A	104	11/03/91-11/09/91
DD 104035	DC 031411	91_46A	92	11/10/91-11/16/91
DD 104036	DC 031412	91_47A	102	11/17/91-11/23/91
DD 104037	DC 031413	91_48A	94	11/24/91-11/30/91
DD 104038	DC 031414	91_49A	91	12/01/91-12/07/91
DD 104039	DC 031415	91_50A	99	12/08/91-12/14/91
DD 104040	DC 031416	91_51A	88	12/15/91-12/21/91
DD 104041	DC 031417	91_52A	88	12/22/91-12/28/91
DD 104042	DC 031418	91_53A	26	12/29/91-12/31/91
DD 104043	DC 031419	92_01A	46	01/01/92-01/04/92
DD 104044	DC 031420	92_02A	88	01/05/92-01/11/92
DD 104045	DC 031421	92_03A	98	01/12/92-01/18/92
DD 104046	DC 031422	92_04A	103	01/19/92-01/25/92
DD 104047	DC 031423	92_05A	103	01/26/92-02/01/92
DD 104048	DC 031424	92_06A	103	02/02/92-02/08/92
DD 104049	DC 031425	92_07A	106	02/09/92-02/15/92
DD 104050	DC 031426	92_08A	102	02/16/92-02/22/92
DD 104051	DC 031427	92_09A	107	02/23/92-02/29/92

	DD 104052	DC 031428	92_10A	106	03/01/92-03/07/92
	DD 104053	DC 031429	92_11A	106	03/08/92-03/14/92
	DD 104054	DC 031430	92_12A	102	03/15/92-03/21/92
	DD 104055	DC 031431	92_13A	106	03/22/92-03/28/92
	DD 104056	DC 031432	92_14A	104	03/29/92-04/04/92
	DD 104057	DC 031433	92_15A	106	04/05/92-04/11/92
	DD 104058	DC 031434	92_16A	103	04/12/92-04/18/92
	DD 104059	DC 031435	92_17A	104	04/19/92-04/25/92
	DD 104060	DC 031436	92_18A	99	04/26/92-05/02/92
	DD 104061	DC 031437	92_19A	102	05/03/92-05/09/92
	DD 104062	DC 031438	92_20A	103	05/10/92-05/16/92
	DD 104063	DC 031439	92_21A	107	05/17/92-05/23/92
	DD 104064	DC 031440	92_22A	108	05/24/92-05/30/92
	DD 104065	DC 031441	92_23A	103	06/01/92-06/06/92
	DD 104066	DC 031442	92_24A	102	06/07/92-06/13/92
	DD 104067	DC 031443	92_25A	107	06/14/92-06/20/92
	DD 104068	DC 031444	92_26A	99	06/21/92-06/27/92
	DD 104069	DC 031445	92_27A	97	06/28/92-07/04/92
	DD 104070	DC 031446	92_28A	96	07/05/92-07/11/92
	DD 104071	DC 031447	92_29A	106	07/12/92-07/18/92
	DD 104072	DC 034448	92_30A	100	07/19/92-07/25/92
	DD 104073	DC 031449	92_31A	93	07/26/92-08/01/92
	DD 104074	DC 031450	92_32A	103	08/02/92-08/08/92
	DD 104075	DC 031451	92_33A	100	08/09/92-08/15/92
	DD 104076	DC 031452	92_34A	103	08/16/92-08/22/92
	DD 104077	DC 031453	92_35A	104	08/23/92-08/29/92
	DD 104078	DC 031454	92_36A	100	08/30/92-09/05/92
	DD 107585	DC 031634	92_37A	107	09/06/92-09/12/92
	DD 107586	DC 031635	92_38A	98	09/13/92-09/19/92
	DD 107587	DC 031636	92_39A	103	09/20/92-09/26/92
	DD 107588	DC 031637	92_40A	102	09/27/92-10/03/92
	DD 107589	DC 031638	92_41A	100	10/04/91-10/10/92
	DD 107590	DC 031639	92_42A	103	10/11/92-10/17/92
	DD 107591	DC 031640	92_43A	103	10/18/92-10/24/92
	DD 107592	DC 031641	92_44A	105	10/25/92-10/31/92
	DD 107593	DC 031642	92_45A	106	11/01/91-11/07/92
	DD 107594	DC 031643	92_46A	106	11/08/92-11/14/92
	DD 107595	DC 031644	92_47A	105	11/15/92-11/21/92
	DD 107596	DC 031645	92_48A	103	11/22/92-11/28/92
	DD 107597	DC 031646	92_49A	104	11/29/92-12/05/92
	DD 107598	DC 031647	92_50A	100	12/06/92-12/12/92
	DD 107599	DC 031648	92_51A	105	12/13/92-12/19/92
	DD 107600	DC 031649	92_52A	107	12/20/92-12/26/92
	DD 107601	DC 031650	92_53A	61	12/27/92-12/31/92
	DD 107602	DC 031651	93_01A	27	01/01/93-01/02/93
	DD 107603	DC 031652	93_02A	104	01/03/93-01/09/93
	DD 107604	DC 031653	93_03A	102	01/10/93-01/16/93
	DD 107605	DC 031654	93_04A	105	01/17/93-01/23/93
	DD 107606	DC 031655	93_05A	106	01/24/93-01/30/93
	DD 107607	DC 031656	93_06A	108	01/31/93-02/06/93
	DD 107608	DC 031657	93_07A	105	02/07/93-02/13/93
	DD 107609	DC 031658	93_08A	105	02/14/93-02/20/93
	DD 107610	DC 031659	93_09A	105	02/21/93-02/27/93
	DD 107611	DC 031660	93_10A	103	02/28/93-03/06/93
	DD 107679	DC 031552	93_11A	99	03/07/93-03/13/93
***	DD 107680	DC 031553	93_12A	103	03/14/93-03/20/93
***	DD 107681	DC 031554	93_13A	100	03/21/93-03/27/93
***	DD 107682	DC 031555	93_14A	104	03/28/93-04/03/93
***	DD 107683	DC 031556	93_15A	100	04/04/93-04/10/93

*** DD 107684	DC 031557	93_16A	102	04/11/93-04/17/93
*** DD 107685	DC 031558	93_17A	101	04/18/93/04/24/93
*** DD 107686	DC 031559	93_18A	102	04/25/93-05/01/93
*** DD 107687	DC 031560	93_19A	103	05/02/93-05/08/93
*** DD 107688	DC 031561	93_20A	105	05/09/93-05/15/93
*** DD 107689	DC 031562	93_21A	101	05/16/93-05/22/93
*** DD 107690	DC 031563	93_22A	102	05/23/93-05/29/94
*** DD 107691	DC 031564	93_23A	99	05/30/93-06/05/93
*** DD 107692	DC 031565	93_24A	105	06/06/93-06/12/93
*** DD 107693	DC 031566	93_25A	102	06/13/93-06/19/93
*** DD 107694	DC 031567	93_26A	102	06/20/93-06/26/93
*** DD 107695	DC 031568	93_27A	101	06/27/93-07/03/93
DD 107753	DC 031661	93_28A	107	07/04/93-07/10/93
DD 107754	DC 031662	93_29A	106	07/11/93-07/17/93
DD 107755	DC 031663	93_30A	102	07/18/93-07/24/93
DD 107756	DC 031664	93_31A	105	07/25/93-07/31/93
DD 107757	DC 031665	93_32A	104	08/01/93-08/07/93
DD 107758	DC 031666	93_33A	107	08/08/93-08/14/93
DD 107759	DC 031667	93_34A	108	08/15/93-08/21/93
DD 107760	DC 031668	93_35A	105	08/22/93-08/28/93
DD 107761	DC 031669	93_36A	92	08/29/93-09/04/93
DD 107762	DC 031670	93_37A	83	09/06/93-09/11/93
DD 107763	DC 031671	93_38A	107	09/12/93-09/18/93
DD 107764	DC 031672	93_39A	105	09/19/93-09/25/93
DD 107765	DC 031673	93_40A	105	09/26/93-10/02/93
DD 107827	DC 031679	93_41A	105	10/03/93-10/09/93
DD 107828	DC 031680	93_42A	102	10/10/93-10/16/93
DD 107829	DC 031681	93_43A	105	10/17/93-10/23/93
DD 107830	DC 031682	93_44A	107	10/24/93-10/30/93
DD 107831	DC 031683	93_45A	107	10/31/93-11/06/93
DD 107866	DC 031684	93_46A	105	11/07/93-11/13/93
DD 107867	DC 031685	93_47A	106	11/14/93-11/20/93
DD 107868	DC 031686	93_48A	86	11/21/93-11/27/93
DD 107869	DC 031687	93_49A	82	11/28/93-12/03/93
DD 107870	DC 031688	93_50A	94	12/05/93-12/11/93
DD 107871	DC 031689	93_51A	105	12/12/93-12/18/93
DD 107872	DC 031690	93_52A	102	12/19/93-12/25/93
DD 107873	DC 031691	93_53A	66	12/26/93-12/31/93
DD 108003	DC 031742	94_07A	100	02/06/94-02/12/94
DD 108004	DC 031743	94_08A	105	02/13/94-02/19/94
DD 108005	DC 031744	94_09A	104	02/20/94-02/26/94
DD 108006	DC 031745	94_10A	103	02/27/94-03/05/94
DD 108028	DC 031787	94_11A	107	03/06/94-03/12/94
DD 108029	DC 031788	94_12A	108	03/13/94-03/19/94
DD 108030	DC 031789	94_13A	103	03/20/94-03/26/94
DD 108031	DC 031790	94_14A	106	03/27/94-04/02/94
DD 108520	DC 032456	94_46a	104	11/06/94 - 11/12/94
DD 108521	DC 032457	94_47a	101	11/13/94 - 11/19/94
DD 108522	DC 032458	94_48a	101	11/20/94 - 11/26/94
DD 108523	DC 032459	94_49a	107	11/27/94 - 12/03/94
DD 108524	DC 032460	94_50a	95	12/04/94 - 12/10/94
DD 108525	DC 032461	94_51a	103	12/11/94 - 12/17/94
DD 108526	DC 032462	94_52a	106	12/18/94 - 12/24/94
DD 108527	DC 032463	94_53a	90	12/25/94 - 12/31/94
DD 108528	DC 032464	95_01a	102	01/01/95 - 01/07/95
DD 108529	DC 032465	95_02a	108	01/08/95 - 01/14/95
DD 108530	DC 032466	95_03a	107	01/15/95 - 01/21/95
DD 108531	DC 032467	95_04a	106	01/22/95 - 01/28/95
DD 108532	DC 032468	95_05a	105	01/29/95 - 02/04/95

DD 108538	DC 032530	94_41a	108	10/02/94 - 10/08/94
DD 108539	DC 032531	94_42a	102	10/09/94 - 10/15/94
DD 108540	DC 032532	94_43a	105	10/16/94 - 10/22/94
DD 108541	DC 032533	94_44a	104	10/23/94 - 10/29/94
DD 108542	DC 032534	94_45a	99	10/30/94 - 11/05/94
DD 108595	DC 032696	95_06a	101	02/05/95 - 02/11/95
DD 108596	DC 032697	95_07a	105	02/12/95 - 02/18/95
DD 108597	DC 032698	95_08a	106	02/19/95 - 02/25/95
DD 108598	DC 032699	95_09a	103	02/26/95 - 03/04/95
DD 108599	DC 032700	95_10a	107	03/05/95 - 03/11/95
DD 108600	DC 032701	95_11a	104	03/12/95 - 03/18/95
DD 108601	DC 032702	95_12a	105	03/19/95 - 03/25/95
DD 108602	DC 032703	95_13a	97	03/26/95 - 04/01/95
DD 108603	DC 032704	95_14a	101	04/02/95 - 04/08/95
DD 108604	DC 032705	95_15a	103	04/09/95 - 04/15/95
DD 108605	DC 032706	95_16a	106	04/16/95 - 04/22/95
DD 108606	DC 032707	95_17a	102	04/23/95 - 04/29/95
DD 108607	DC 032708	95_18a	107	04/30/95 - 05/06/95

THE "C" TAPES ARE 4MM CARTRIDGES

A Installation and Maintenance of the Yohkoh Software

A.1 Network Installation

This section describes the network installation of compressed tar files on Unix systems[6]. Current disk allocation requirements: Approximately 100 MBytes which includes the Software and several supporting databases (35 Mbytes for only IDL and FORTRAN software or 25 Mbytes for only IDL software).

1. Determine which user will own the Yohkoh software and database area. Some sites prefer to create a special "Yohkoh" account for management but this is not an installation requirement. Log into the appropriate account.
2. Determine the path where you want to locate the Yohkoh Software. Let's call this path *yparent*. This pathname should be an nfs name per /etc/fstab if the software is to be shared among machines. Of course, the owner of the Yohkoh Software (determined in preceding step) requires write access to *yparent*.
3. Transfer the boot-up script via anonymous FTP.

```
% mkdir -p yparent/ysw/swmaint/tar      # local directory for tar files
% cd yparent/ysw/swmaint/tar            # move there
% ftp 133.74.8.100                      # user=ftp, passwd=name@node
FTP> cd pub/swmaint/tar                 # remote location (isass0)
FTP> get ys_install                     # boot up script
FTP> quit
```

4. Install the software and standard database using the script *ys_install*. The defaults are usually ok, but see script header for available options. Use the */HELP* switch to get updated instructions and the latest Yohkoh Software and Database descriptions without installing anything. (current directory is still = *yparent*/ysw/swmaint/tar).

```
% csh ys_install                        # INSTALL, take defaults
% csh ys_install [OPTIONS]              # INSTALL, use 1 or more options
% csh ys_install /HELP                  # No installation, Help Information
```

Section B.1 describes the procedure for setting up the Yohkoh environment for new accounts.

A.2 Network Upgrades

This section is applicable if you have previously installed the Yohkoh Software on your Unix workstation. It will upgrade the latest version of the software and database files using the existing locations.

[6] If you have already installed Yohkoh Software in the past and do not need to relocate it, skip to section A.2. (Upgrades) instructions.

Software (p.2)

NOTE: beginning with Yohkoh Software Version# 931107.02, auto-upgrade scripts are included in the distribution. Once you have upgraded to this level, all FTP commands are automatically issued. For older Versions (before 930513.01) you must first retrieve the installation "boot" script – other files will be auto-FTPed as required.

```
% tail -1 $ys/gen/setup/ys.version          # determine current version number
```

A.2.1 New Versions (starting with 931107.02)

Log in as owner of the Yohkoh Software tree; verify that $ys is defined. For possible options see header of $ys/gen/script/ys_install. Next, [7]

```
% csh $ys/gen/script/ys_install
% csh $ys/gen/script/ys_install [OPTIONS]
```

A.2.2 Old Versions (Before 930513.01)

Log in as owner of the Yohkoh Software tree; verify that $ys is defined

```
% cd $ys/swmaint/tar                        # local tar site
% ftp 133.74.8.100                          # user=ftp, passwd=name@node
FTP> cd pub/swmaint/tar                     # remote location (isass0)
FTP> get ys_install                         # boot up script
FTP> quit
```

see header of $ys/swmaint/tar/ys_install for options, then:

```
% csh $ys/swmaint/tar/ys_install
% csh $ys/swmaint/tar/ys_install [OPTIONS]
```

A.3 Automatic Upgrades of Yohkoh Software

The command to automatically upgrade the Yohkoh software on a periodic basis may be inserted into a "cron job" at remote sites[8]. If you are not familiar with function and operation of cron jobs on your particular machine, you should consult with your system manager. Although there may be some system dependencies not covered here we present some sample implementations here. Notes: *ys* is the path assigned to $ys ; *ysowner* is the username of the Yohkoh Software owner.

[7] this command may be placed in a daily "cronjob" to provide automatic upgrades – see discussion on page 125.
[8] Requires Yohkoh Software Version: 931108.02 or higher

Software (p.3)

Some suggested reference books, material, and methods of finding more information:

- The "Red" Book: "The Yohkoh (Solar-A) Mission," Solar Physics, Volume 136, No.1 1991
- The 'Blue' Book: Solar-A (M-3SII-6) Engineering Reference Book (written in Japanese)
- The Yohkoh File Control Document (FCD)
- The Interactive Data Language (IDL) User's Guide
- See the .INC files which are in the /ys/gen/soft/rel/ref_access directory for details on the definitions of the IDL data structure tags and contents.
- The official Yohkoh IDL software should have a detailed description of the routine in the header of the source file. It is possible to access this header from within IDL by using DOC_LIBRARY. A sample call is:

  IDL> **doc_library,'sxt_prep'**

The Yohkoh Analysis Guide is intended as an introduction and quick reference to the Yohkoh software, databases and instruments. This document describes the organization of the software, format of the data, instrument characteristics, and a list of the most commonly used programs.

Version 1 of this guide was compiled from information provided by a small number of people, starting from earlier guides by Keith Strong and Alan McAllister.

Version 2 of this guide was compiled from contributions provided from each Yohkoh instrument team and from Co-I's in Japan, the U.K., and the U.S. The number of individuals is too large to list here, but we are thankful for each contribution and suggestion. The LPARL team would also like to thank Prof. Yoshiaki Ogawara, the Yohkoh Project Manager, for his kind encouragement during the compilation of this guide. We are thankful to Drs. Loren Acton, George Doschek, James Klimchuk, and John Mariska who reviewed the final version and made many helpful comments.

As noted above, this document was compiled from a large number of sources. In the case of software, it was not always possible to verify the correctness of the information contained here. We would welcome any comments or corrections which can be sent to the E-mail address given below. Specific questions or comments will be re-directed to the appropriate individuals as necessary.

The Yohkoh team plans to make future software and documentation updates. If you would like to register to receive notification of significant software or instrument calibration updates, please send your name, postal address, and E-mail address to the E-mail address shown below.

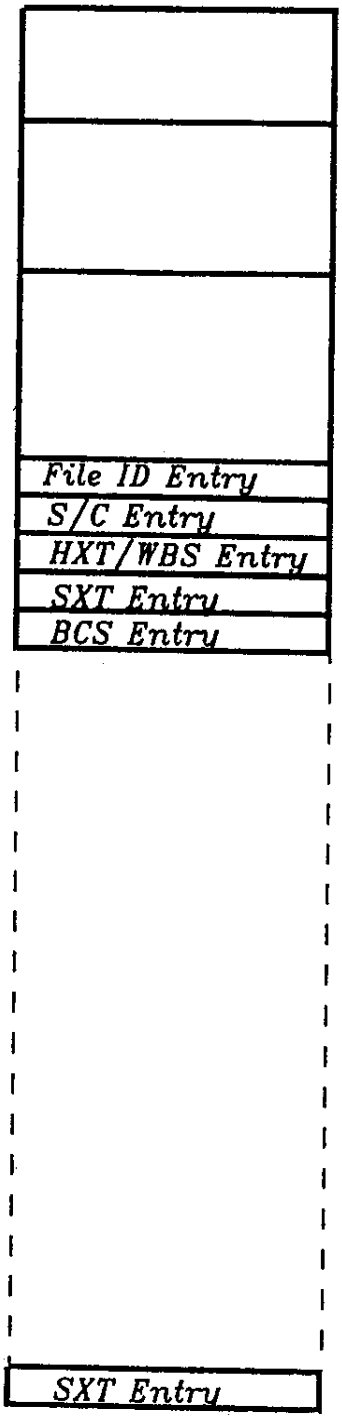
If you have general comments, suggestions, or questions regarding this document or software, please send them to:

`software@isass0.solar.isas.ac.jp`

MDM 18-Apr-90

Observation Log

91-062A-01A
02B
03A
04A



File Info / Pointer Section

File Header

Pointer to Start of New Orbit

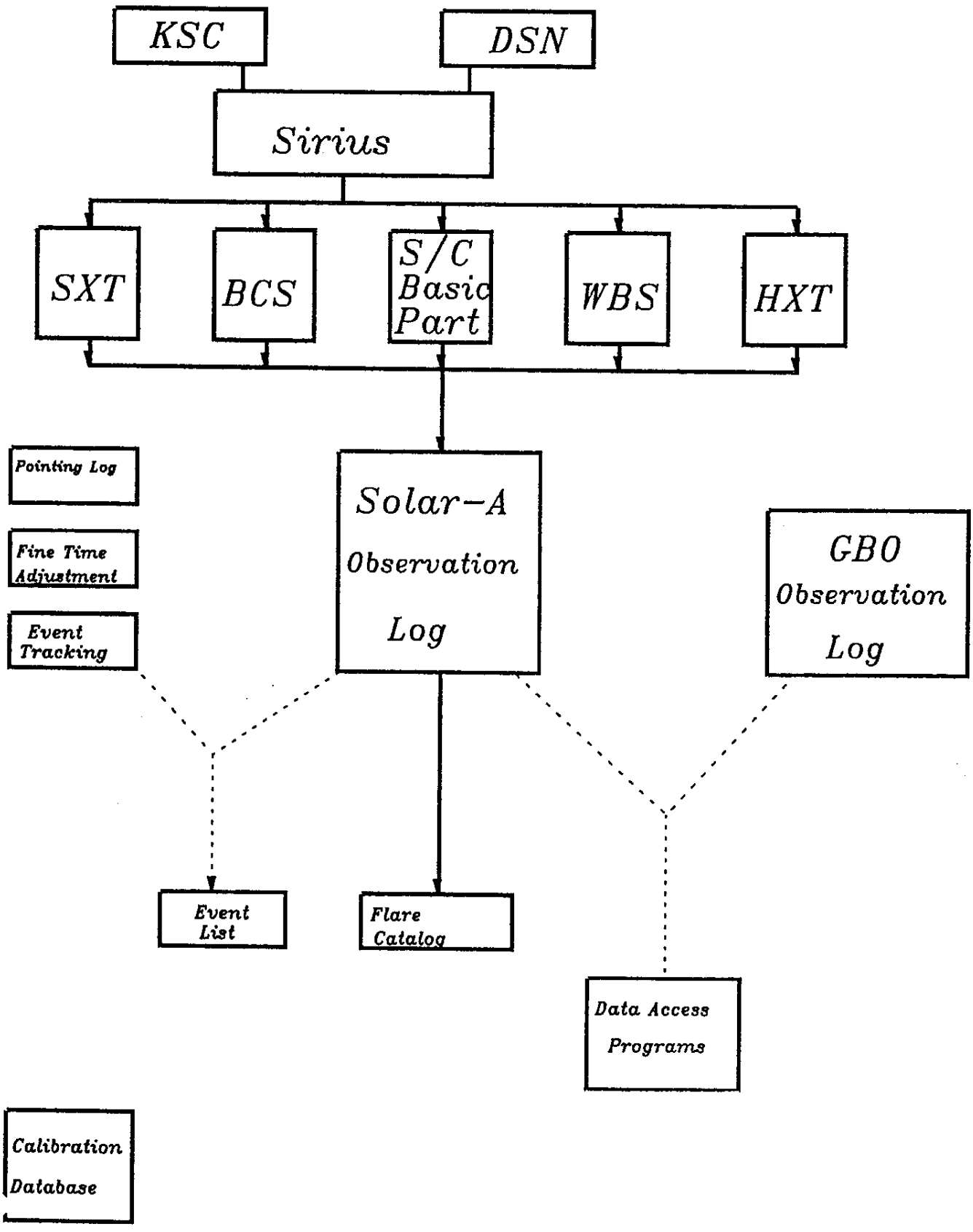
Instrument Entries

* Up to 48000 entries per day
((1280 MF * 1/2 * 3 Entries + 1280 MF * 1 Entry) * 15 Orbits)

* Different entry for each instrument (except HXT/WBS combined)
(should not duplicate data unnecessarily)

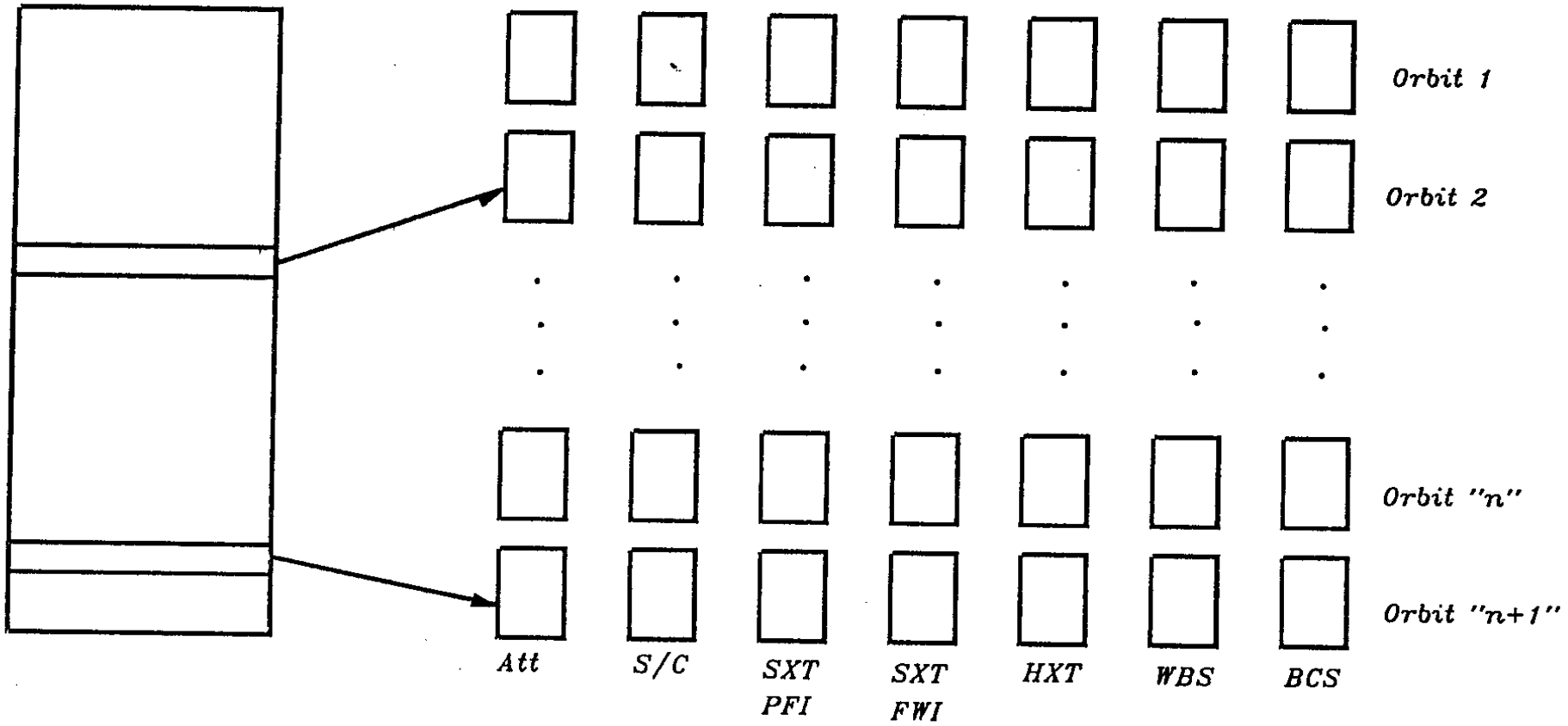
MDM 18-Apr-90

Solar-A Data Flow



Overview of Solar-A Database

MDM 17-Apr-90



Observation Log

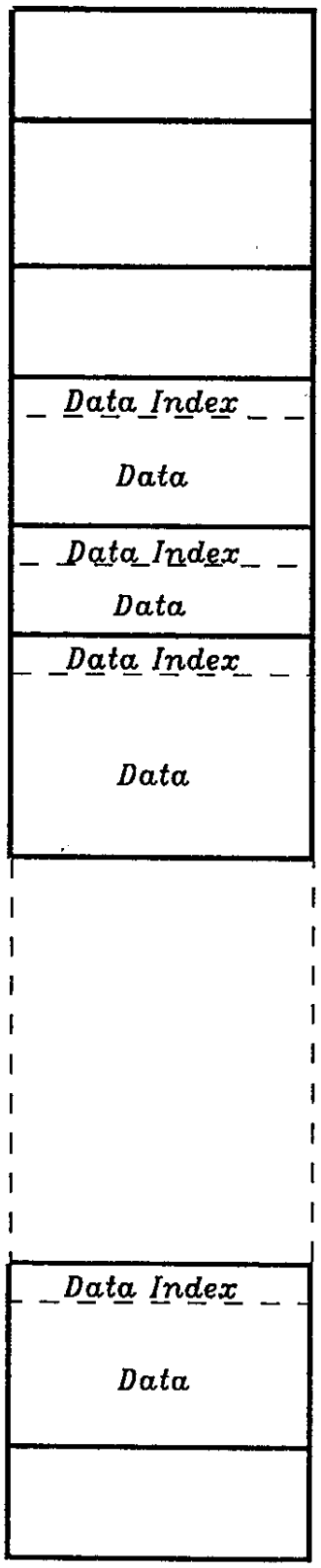
* Whole Solar-A mission on-line
* Separate file each month (35-45 MBytes)
* Different entry types for each instrument

Reformatted Data Files

* One File for each instrument for each orbit
* 3150 files per month (15 orbit * 30 days * 7)
* Each file up to 8 mBytes

MDM 18-Apr-90

Solar-A Reformatted Files



File Info / Pointer Section

File Header

Static Index Information

Data Section

Road Map Section (only SXT and BCS)

* All Solar-A files have this organization
* Allows for the analysis of data from all instruments in a consistent way

OBSERVATION LOG INFORMATION

	Spacecraft High/Avg	SXT High/Avg	BCS High/Avg	HXT/WBS High/Avg	Total High/Avg
Bytes / Entry	32	32	32	32	
# Entry / Orbit	640/ 640	1280/ 400	640/ 640	640/ 640	3200/ 2320
# K Entry / Day	9/ 9	19/ 6	9/ 9	9/ 9	48/ 34
# K Entry / Month	288/ 288	576/ 180	288/ 288	288/ 288	1440/ 1044
# K Entry / Year	3504/ 3504	7008/ 2190	3504/ 3504	3504/ 3504	17520/ 12702
# K Entry / 3 Years	10512/ 10512	21024/ 6569	10512/ 10512	10512/ 10512	52559/ 38106
# K Bytes / Orbit	20.5/ 20.5	41.0/ 12.8	20.5/ 20.5	20.5/ 20.5	102.4/ 74.2
# K Bytes / Day	307.2/ 307.2	614.4/ 192.0	307.2/ 307.2	307.2/ 307.2	1536.0/ 1113.6
# M Bytes / Month	9.2/ 9.2	18.4/ 5.8	9.2/ 9.2	9.2/ 9.2	46.1/ 33.4
# M Bytes / Year	112.1/ 112.1	224.3/ 70.1	112.1/ 112.1	112.1/ 112.1	560.6/ 406.5
# M Bytes / 3 Years	336.4/ 336.4	672.8/ 210.2	336.4/ 336.4	336.4/ 336.4	1681.9/ 1219.4

REFORMATTED DATA INFORMATION

	Spacecraft High/Low	SXT High/Low	BCS High/Low	HXT/WBS High/Low	Total
# M Bytes / Orbit	2.5/ 2.5	6.3/ 5.0	5.0/ 0.6	0.6/ 1.9	10.0
# M Bytes / Day	37.5/ 37.5	93.8/ 75.0	75.0/ 9.4	9.4/ 28.1	150.0
# G Bytes / Month	1.1/ 1.1	2.8/ 2.3	2.3/ 0.3	0.3/ 0.8	4.5
# G Bytes / Year	13.7/ 13.7	34.2/ 27.4	27.4/ 3.4	3.4/ 10.3	54.8
# G Bytes / 3 Years	41.1/ 41.1	102.7/ 82.1	82.1/ 10.3	10.3/ 30.8	164.3

NOTES:
(1) Assumes 15 BDR dumps per day (or equivalent real time downlink)

OBSERVATION LOG INFORMATION

	Spacecraft High/Avg		SXT High/Avg		BCS High/Avg		HXT/WBS High/Avg		Total High/Avg	
Bytes / Entry	32		32		32		32			
# Entry / Orbit	640/	640	1280/	400	640/	640	640/	640	3200/	2320
# K Entry / Day	6/	6	12/	4	6/	6	6/	6	32/	23
# K Entry / Month	192/	192	384/	120	192/	192	192/	192	960/	696
# K Entry / Year	2336/	2336	4672/	1460	2336/	2336	2336/	2336	11680/	8468
# K Entry / 3 Years	7008/	7008	14016/	4380	7008/	7008	7008/	7008	35040/	25404
# K Bytes / Orbit	20.5/	20.5	41.0/	12.8	20.5/	20.5	20.5/	20.5	102.4/	74.2
# K Bytes / Day	204.8/	204.8	409.6/	128.0	204.8/	204.8	204.8/	204.8	1024.0/	742.4
# M Bytes / Month	6.1/	6.1	12.3/	3.8	6.1/	6.1	6.1/	6.1	30.7/	22.3
# M Bytes / Year	74.8/	74.8	149.5/	46.7	74.8/	74.8	74.8/	74.8	373.8/	271.0
# M Bytes / 3 Years	224.3/	224.3	448.5/	140.2	224.3/	224.3	224.3/	224.3	1121.3/	812.9

REFORMATTED DATA INFORMATION

	Spacecraft High/Low		SXT High/Low		BCS High/Low		HXT/WBS High/Low		Total
# M Bytes / Orbit	2.5/	2.5	6.3/	5.0	5.0/	0.6	0.6/	1.9	10.0
# M Bytes / Day	25.0/	25.0	62.5/	50.0	50.0/	6.3	6.3/	18.8	100.0
# G Bytes / Month	0.8/	0.8	1.9/	1.5	1.5/	0.2	0.2/	0.6	3.0
# G Bytes / Year	9.1/	9.1	22.8/	18.3	18.3/	2.3	2.3/	6.8	36.5
# G Bytes / 3 Years	27.4/	27.4	68.4/	54.8	54.8/	6.8	6.8/	20.5	109.5

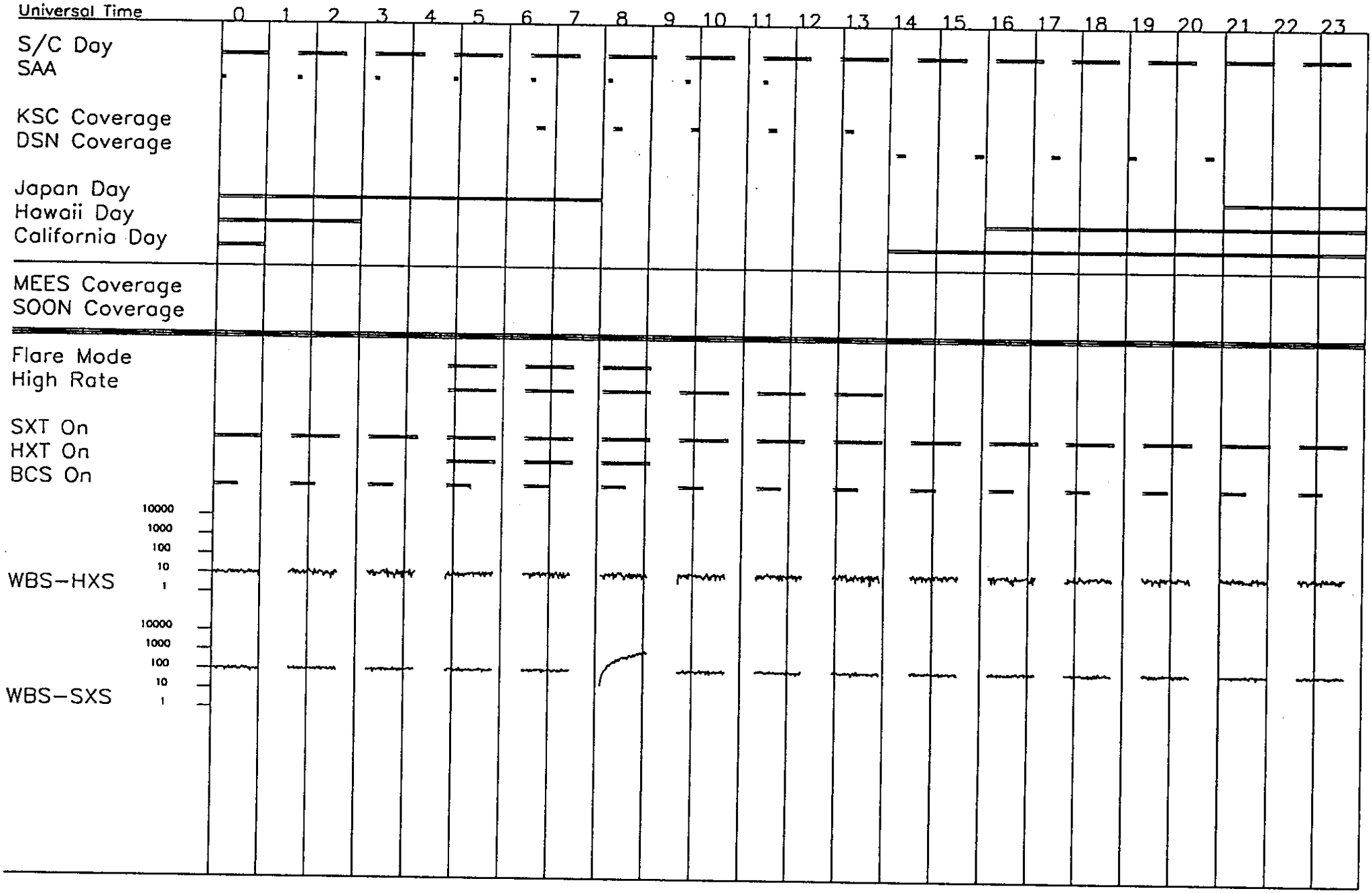
NOTES:

(1) Assumes 10 BDR dumps per day (or equivalent real time downlink)

Input Information: TEST

Created : 18-APR-1990 20:09:32.52

Saturday 1-Dec-91 (305)



MDM 18-Apr-90

Observing Log

1. Searches
   A. MODES: List the cases where the instruments were in a given mode.

      EXAMPLES:
      * Filters used, ...
      * Different instruments in given modes at the same time

   B. DATA: List the occasions where given solar activity was observed

      EXAMPLES:
      * Minimum threshold in a given channel/image

2. Light curves
   A. WBS/BCS time profiles

3. Allows the generation of timelines which show
   A. S/C modes
   B. Active instrument
   C. Ground based collaborative data
   D. Where we received BDR or real time coverage

MDM 18-Apr-90

Proposal for Solar-A Reformatted File Names

```
SPRyymmdd.hhmm
^^^                     S = SXT
 ^^                     H = HXT
 ^^                     B = BCS
 ^^                     W = WBS
 ^^                     A = Attitude control
 ^^                     ? = S/C

 ^^                     For SXT
  ^                             P = Partial Frame Image (PFI)
  ^                             F = Full Width Image (FWI)
                        For BCS/WBS/HXT
                              D = Data (?)

  ^                     R = Raw unprocessed telemetry data
                        C = Ground calibration data
                        B = Background subtracted data
```

Example:

```
SPR911101.1234
SFR911101.1234
BDR911101.1234
HDR911101.1234
WDR911101.1234
ADR911101.1234
?DR911101.1234
```

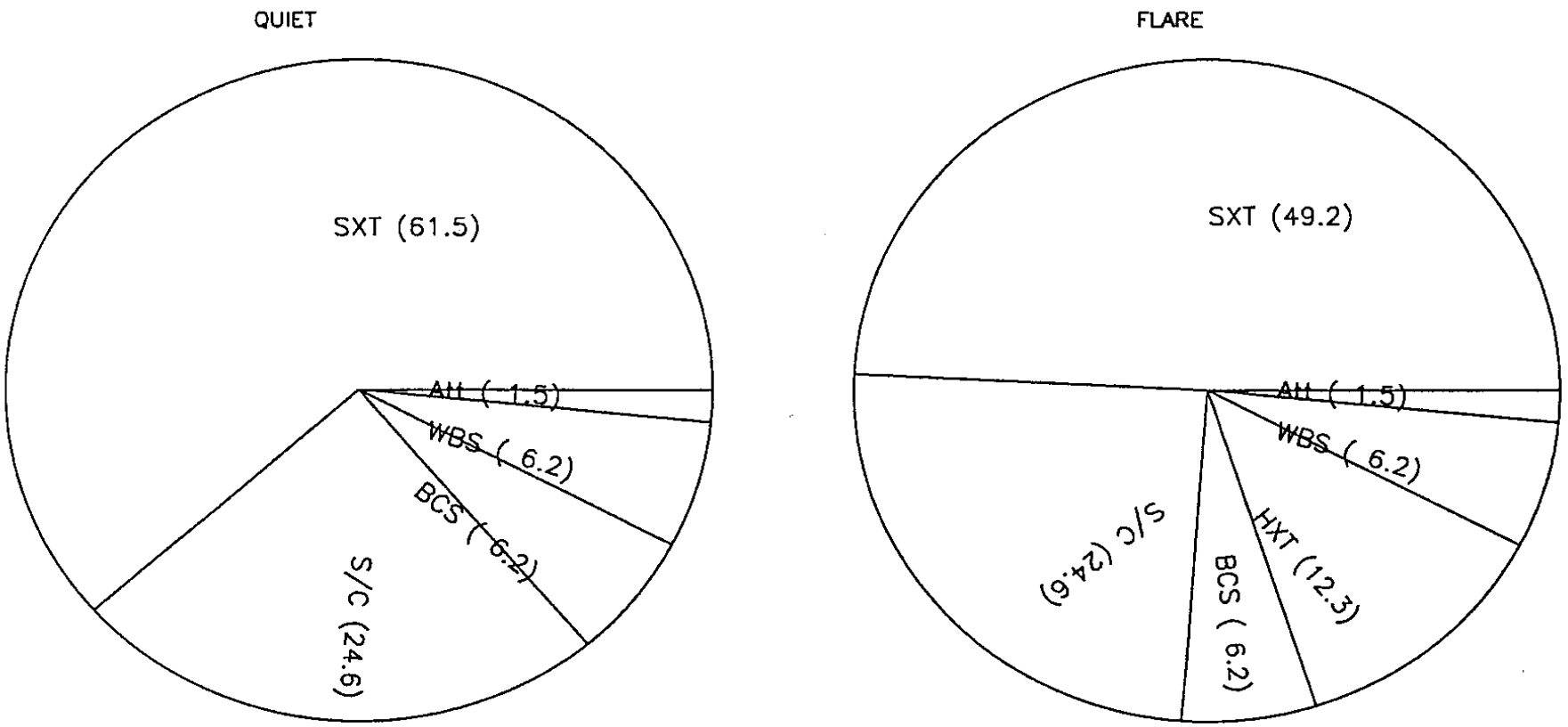
* "yymmdd.hhmm" is the File ID. The time is the ground solution for the "first minute of sun" which is the beginning of the spacecraft day. All instruments have the same File ID.
* One file per orbit covers ~90 minutes of data. The file contains data from start of S/C day to start of the next orbit S/C day.
* Some files will have a single character following the File ID (examples: "T" for temporary file; "L" for listing)

MDM 18-Apr-90

Different Observation Log Entry Types

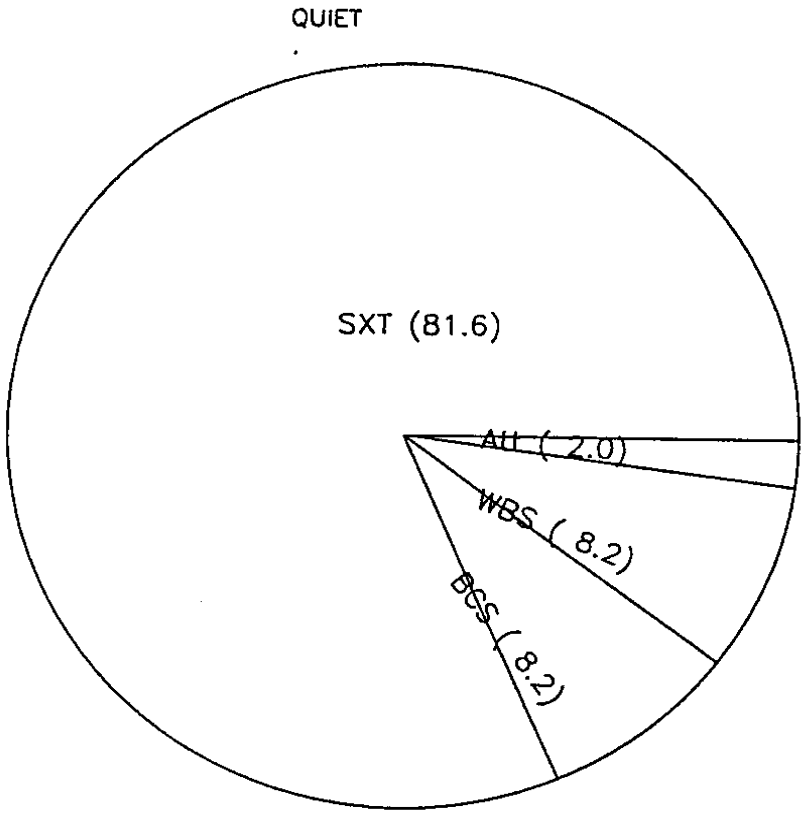
SXT Instrument	- Once per image (not more often than every 2 seconds)
HXT Instrument	- Every 2 MF in HIGH, every MF in MED or LOW
WBS Instrument	- Every 2 MF in HIGH, every MF in MED or LOW
BCS Instrument	- Every 2 MF in HIGH, every MF in MED or LOW
S/C Instrument	- Every 2 MF in HIGH, every MF in MED or LOW
File ID	- Once per orbit
BCS Calibration	- Whenever discriminator, gain, and HV setting change
HXT Calibration	- Whenever discriminator, gain, and HV setting change
WBS Calibration	- Whenever discriminator, gain, and HV setting change
Orbital Solution	- Once per week (?) - S/C orbital element solution

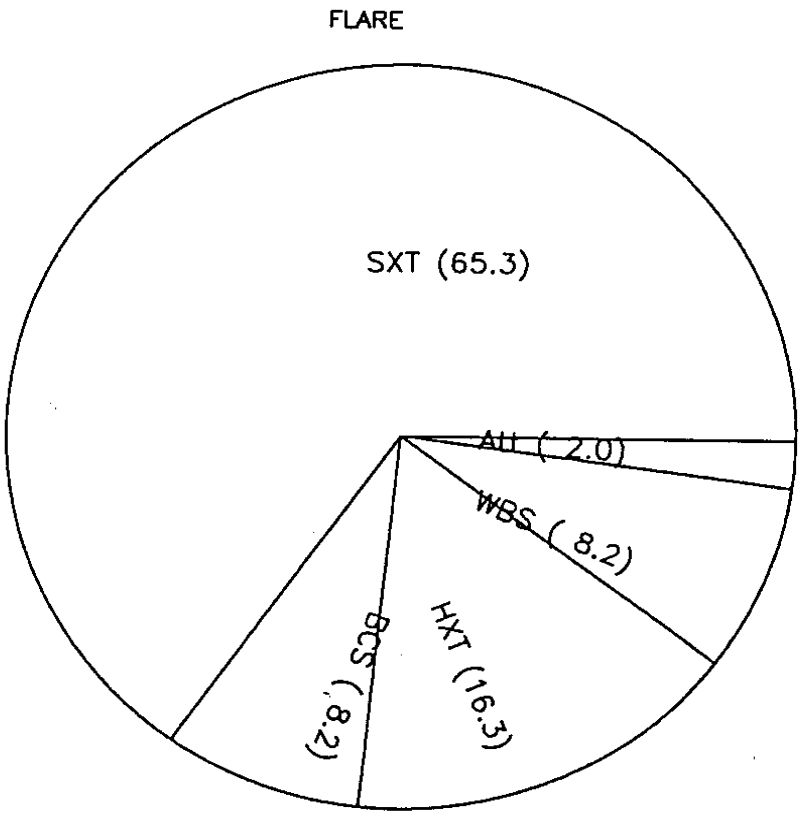
Distributed Data – S/C Data Saved



Plot Made 19-APR-1990 09:49:31.38

Distributed Data – S/C Data Not Saved





Plot Made 19-APR-1990 09:49:36.27

B4367-000A

91-062A-01A
02B
03A
04A

Yohkoh Database and Software User's Guide Volume 1

Ver 1.02 M.Morrison 8-Sep-92
Ver 1.03 M.Morrison 18-Sep-92
Ver 1.04 M.Morrison 28-Oct-92
Ver 1.05 M.Morrison 30-Oct-92
Ver 1.06 M.Morrison 6-Nov-92

This Copy Printed: November 6, 1992

Some suggested reference books are:

- The 'Red' Book: 'The Yohkoh (Solar-A) Mission', Solar Physics, Volume 136, No.1 1991
- The 'Blue' Book: Solar-A (M-3SII-6) Engineering Reference Book (written in Japanese)
- The File Control Document (FCD)
- The Interactive Data Language (IDL) User's Guide

This document is intended to be used as an introduction and quick reference to the Yohkoh Database and Software. The document describes the organization of the data and the software, as well as a list of the most commonly used programs. If there are comments, suggestions, or questions regarding this document, please send them to:

`software@isass0.solar.isas.ac.jp`

There will be weekly or monthly newletters which will describe additions to the User's Guide (new programs) and modifications or corrections made to existing software. If you wish to be on the mailing list, please send your request to the software account listed above.

BEWARE: Version 1.06 has not been completely reviewed by all of the different instrument teams and might have errors. In addition, you will notice that there are several sections that have not been written yet.

Contents

Preface

Chapter 1 is an introduction to the Yohkoh spacecraft and the instruments on the spacecraft. It discusses the instrument resolutions and energy ranges as well as the basic modes in which the instruments can be operated.

Chapters 2 and 3 discuss the organization of the Yohkoh database and software. The file names and directory organization is briefly discussed.

Chapter 4 is a simple introduction for how to access the Yohkoh data. A user should be able to read this chapter and follow the step by step instructions and have data displayed on the screen in less than 5 minutes.

Chapters 5 through 21 provide a brief description of the different software that is available. The information is very brief and it is recommended that you use DOC_LIBRARY to get more details on the capabilities of each of these routines. It is organized by function and covers all the fundamental routines needed to analyze the Yohkoh data.

Appendix A is a description of how to log into the Yohkoh computer systems and to set up the X-window displays. Appendix B and C describe how to access Hawaii SPAM and NAOJ H-Alpha data. Appendix D describes how to make laser disk and VCR movies, and appendix E and F describes exabytes and magneto-optical disks. Appedix G gives some guidelines for writing Yohkoh software.

Volume 2 of the Yohkoh Database and Software User's Guide contains a detailed description of making software installations at remote sites; the internal organization of the archive tapes; and how to run the weekly reformatter program.

The conventions used in this manual are:

- Bold face for examples of what the user should type
- Program names are capitalized when mentioned in the text description, but are not capitalized in the user input examples
- Variable names are in italicized when mentioned in the text description
- There are several different types of prompts that appear in the examples. The prompt IDL> is used for when the user has already entered and is running IDL. The prompt % is used to signify Unix commands, and the prompt $ signifies VMS commands. There are also FTP> prompts for 'File Transfer Protocal'.

For example, you would only type .run yodat for the following User's Guide description. You would not type IDL>. Also, for long example strings, the string might wrap around to the next line. The user should use a $ sign to continue on the next line when necessary (the example on the page is not exactly proper)

```
IDL> .run yodat
```

Some of the IDL routines are marked with a notation surrounded by [and]. The notations are:

[IDL] for standard IDL procedures and functions

[IDL-LIB] for standard IDL user library procedures and functions

[IDL-YO] for standard IDL user library procedures and functions that were modified by the Yohkoh team.

[*] for Yohkoh software that is user contributed and might not be completely bug free or checked out by one of the Yohkoh database and software managers. These are also routines which might be renamed at some point in the future. We would be interested in any problems or errors, but might not be able to respond and fix them in a timely manner. However, please still send mail to 'software@isass0.solar.isas.ac.jp'. These routines might also be removed and/or incorporated into other routines. Please be careful when using these routines.

Routines not marked by any of the above are formal Yohkoh procedures or functions which are fully supported by the Yohkoh team. If there are problems or errors with any of these routines, please send mail to the software account on isass0.

1 Introduction to the Yohkoh Instruments

1.1 Yohkoh Observing Modes

There are generally two Data Processor (DP) modes when Yohkoh is taking scientific data, FLARE mode and QUIET mode. Yohkoh uses the WBS HXS and SXS instruments to monitor the solar activity and when a threshold is passed the S/C enters FLARE mode.

There are three different telemetry rates, but only two are used when observing the sun. HIGH rate is 32 Kbits/sec and MEDIUM is 4 Kbits/sec. When first entering FLARE mode, the DP goes into HIGH rate. After 10 minutes, it will go into MEDIUM rate if the intensity of the flare has subsided, but is still flaring.

1.2 Bragg Crystal Spectrometer (BCS)

Instrument:	Bent Crystal Spectrometers	
Spectral lines:	Fe XXVI (1.7636-1.8044 A)	(Chan 1)
	Fe XXV (1.8298-1.8942 A)	(Chan 2)
	Ca XIX (3.1631-3.1912 A)	(Chan 3)
	S XV (5.0160-5.1143 A)	(Chan 4)
Spectral resolution (l/dl):	3000 - 8000	
Angular resolution:	Full disk	
Best time resolution:	0.125 sec	

The BCS has the capability to bin the different channels in a variety of ways. Each different binning group is defined by a separate ModeID. For a given ModeID is it possible to determine how the data original bins were combined before being telemetered to the ground.

1.3 Hard X-Ray Telescope (HXT)

Instrument:	Fourier Synthesis Telescope
Energy bands:	(15 - 100 keV, 4 channels)
	Low 15- 24 keV
	Medium-1 24- 35 keV
	Medium-2 35- 57 keV

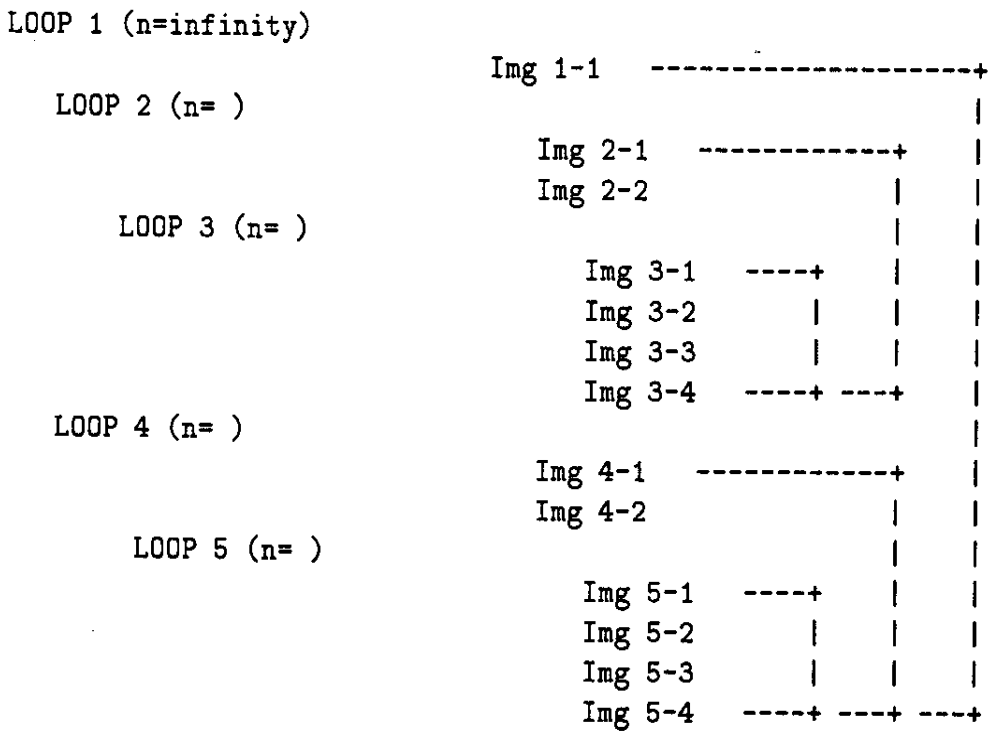
	High	57-100 keV
Angular resolution:	~5 arcsec	
Effective area:	1.5 cm2 avg. x 64 elements	
Best time resolution:	0.5 sec	

1.4 Soft X-Ray Telescope (SXT)

Instrument:	Glancing incidence mirror/CCD sensor	
	Co-aligned optical telescope using same CCD	
Wavelength ranges:	2.5-46 A	no analysis filter
	2.5-36 A	1265 A Al
	2.4-32 A	2930 A Al, 2070 A Mg, 562 A Mn, 190 A C
	2.4-23 A	2.52 micron Mg
	2.4-13 A	11.6 micron Al
	2.3-10 A	119 micron Be
	4600-4800 A	Wide band optical filter
	4290-4320 A	Narrow band optical filter
Spectral discrimination:	Filters	
Angular resolution:	3 arcsec	
Best time resolution:	0.5 sec	
Typical resolution:	2.0 sec in FLARE, 8.0 sec in QUIET	

1.4.1 Sequence Tables

SXT uses sequence tables to define how to take the images will be taken. There are 13 'slots' which are available to set the parameters. For each slot a separate filter, resolution and obserserving region can be specified. The observing region selection is from a table of 9 entries, each entry specifies a location and the size of the output image in pixels (this means that the field of view is different depending on the pixel resolution used).



1.4.2 Automatic Exposure Control (AEC)

1.4.3 Automatic Region Selection (ARS)

1.5 Wide Band Spectrometer (WBS)

Instruments:

Soft X-ray Spectrometer (SXS)
Hard X-ray Spectrometer (HXS)
Gamma-ray Spectrometer (GRS)
Radiation Belt Monitor (RBM)

SXS: Gas Proportional Counter (2-30 KeV, 128 channels)
HXS: NaI scintillation counter (20-400 KeV, 32 channels)
GRS: BGO scintillation counter (0.2-100 MeV, 134 channels)

Angular resolution: Full disk
Best time resolution: 0.125 sec

PC and PH data

SXS has two detectors and two channels for each detector.

```
sxs_pc11 - SXS1 Detector, chan 1 (3-15 keV)
sxs_pc12 - SXS1 Detector, chan 2 (15-40 keV)
sxs_pc21 - SXS2 Detector, chan 1 (3-15 keV)
sxs_pc22 - SXS2 Detector, chan 2 (15-40 keV)

hxs_pc1  - HXS Detector, chan 1 (20-60 keV)
hxs_pc2  - HXS Detector, chan 2 (60-600 keV)

rbm_sc_pc1 - NaI scintillation detector (5-60 keV)
rbm_sc_pc2 - NaI scintillation detector (60-300 keV)
rbm_sd_pc  - Si detector (20 kev)
```

1.6 Spacecraft Attitude Systems

1.6.1 Inertial Reference Unit (IRU)

1.6.2 Two Dimensional Fine Sun Sensor (TFSS)

1.6.3 HXT Aspect Sensor (HXA)

1.6.4 Attitude Determination Software (ADS)

2 Introduction to the Yohkoh Database

For a full description of the Yohkoh database and all of the different data structures, please see the File Control Document.

2.1 Yohkoh Data Format

Each file shall be logically divided into the following six sections. Some files will not use all of the sections described below, but all of them will have a Pointer and File Header Section.

1. File Information and Pointer Section
2. File Header Section
3. Quasi-Static Index Section
4. Index and Data Section
5. Instrument Optional Sections
6. Road Map Section

The program which reads the file will learn from the Pointer Section how to read the rest of the file and where to go to get certain data.

The File Header Section provides information on what data is contained in the file, generally, the extent of the time covered by the contents.

The Quasi-Static Section of the file contains index information that does not vary during the course of an orbit, or varies slowly.

The Index and Data section contain 'data sets'. A data set is a single image for SXT, single spectra for BCS, a single major frame of data for HXT, and two major frames of data for WBS (it takes two major frames for a complete set of WBS data). For each data set there is an index which describes information on the date and time that the data was taken, the mode and position of the instrument's peripherals (filters, HV, ...), and information on temperature and gain information.

The Instrument Optional Section is only used by the BCS data (BDA) files and the spacecraft attitude (ADA) files. The BDA files hold the 'DP_SYNC' information, which is information that is coming down every major frame.

Since the BCS spectra is asynchronous to the major frames, it is stored separately. The ADA file holds the full 2048 point HXA scans.

The Roadmap Section allows a user to access a brief summary of the contents of the file and to perform searches on that summary to select what data should be extracted.

2.2 Yohkoh Data File Names

2.2.1 Yohkoh Directories

There is a set of VMS logicals and UNIX enviroment variables which point to the database directories. By using these logicals in all of the access routines (along with the routine CONCAT_DIR) it is possible for software to be directly portable between VMS and Unix machines. The following is a list of the logicals that exist for the Yohkoh database.

DIR_GEN_ADS	DIR_SXT_CALIBRATE	DIR_BCS_ATODAT
DIR_GEN_EVN	DIR_SXT_DC	DIR_BCS_BALDAT
DIR_GEN_FEM	DIR_SXT_DOC	DIR_BCS_CALDAT
DIR_GEN_GEV	DIR_SXT_ENGIN	DIR_BCS_DOC
DIR_GEN_GXT	DIR_SXT_SENSITIVE	DIR_BCS_EXE
DIR_GEN_MOVIE	DIR_SXT_SFS	DIR_BCS_LOGS
DIR_GEN_NAR	DIR_SXT_SOT	DIR_BCS_MICRO
DIR_GEN_OBS	DIR_SXT_SSL	DIR_BCS_SYNSPEC
DIR_GEN_ORBIT	DIR_SXT_SSX	
DIR_GEN_ORBIT_RAW	DIR_SXT_SXL	DIR_GBO_BBSO
DIR_GEN_ORBIT_SOL	DIR_SXT_TABLES	DIR_GBO_KP
DIR_GEN_ORBIT_SW		DIR_GBO_SELSISI
DIR_GEN_PAN_LASER	DIR_WBS_CAL	DIR_GBO_TEMP
DIR_GEN_PNT		
DIR_GEN_SCRIPT	DIR_HXT_CAL	
DIR_GEN_SETUP		
DIR_GEN_SYNOPTIC		
DIR_GEN_TAPECOPY		
DIR_GEN_XAD		
DIR_GEN_XBD		

2.2.2 Yohkoh Prefixes

Prefix	*Description*	*File Type*
ADA	S/C Attitude Raw Reformatted Data	per Orbit
BDA	BCS Raw Reformatted Data	per Orbit
BSD	*OLD* BCS Instrument Calibrated Spectra	per Orbit
BPC	*OLD* BSDCAL Output (parameters)	per Orbit
BTH	*OLD* BSDFIT Output (Fitted Spectra)	per Orbit
BFT	*OLD* BSDFIT Output (parameters)	per Orbit
BSC	*NEW* BCS Instrument Calibrated Spectra	per Orbit
BSA	*NEW* Answer File for BSC Generation	per Orbit
BSF	*NEW* Output from Spectral Fitting	per Orbit
CBA	S/C Common Basic Raw Reformatted Data	per Orbit
EVN	Yohkoh Event Log	Weekly
FEM	Yohkoh Ephemeris	Weekly
GBC	GBO Big Bear Large Scale H-alpha PFI FITS Image	per Image
GBH	GBO Big Bear H-alpha FITS Image	per Image
GBK	GBO Big Bear Calcium FITS Image	per Image
GBW	GBO Big Bear White Light FITS Image	pre Image
GEV	GOES Event Log	Weekly
GXT	GOES One Minute Light Curve Data	Weekly
GXD	GOES Three Second Light Curve Data	Weekly
GKI	GBO Kitt Peak He 10830 FITS Image	per Image
GKM	GBO Kitt Peak Magnetogram FITS Image	per Image
HDA	HXT Raw Reformatted Data	per Orbit
HXI	HXT Synthesized Image	per Orbit
NAR	NOAA Active Region	Weekly
OBS	Yohkoh Observing Log	Weekly
PNT	S/C Pointing File	Weekly

Prefix	*Description*	*File Type*
SDL	SXT Dark Current Log	Weekly
SDC	SXT Dark Current Images	Weekly (SDA)
SDW	SXT Dark Current Images for Warm CCD	Weekly (SDA)
SFD	SXT FFI Desaturated Composite Images	Weekly (SDA)
SFR	SXT FFI Raw Reformatted Data	per Orbit (SDA)
SFS	SXT FFI Special Images (diffuser, flood...)	Weekly (SDA)
SFW	SXT FFI White Light Images	Weekly (SDA)
SPR	SXT PFI Raw Reformatted Data	per Orbit (SDA)
SOT	SXT Optical Telescope	Weekly
SSC	SXT Synoptic Images (centered)	Weekly (SDA)
SSE	SXT Synoptic Images (east of center)	Weekly (SDA)
SSL	SXT Summary Log	Weekly
SSW	SXT Synoptic Images (west of center)	Weekly (SDA)
SXL	SXT X-Ray Log	Weekly
WDA	WBS Raw Reformatted Data	per Orbit
XAD	Exabyte ASCII Directory for Archive Tape	per Tape
XBD	Exabyte Binary Directory for Archive Tape	per Tape

The PNT file (eg: pnt92_26a.01) has one dataset per major frame for each non-night major frame of data available. Each dataset has the average IRU, TFSS, and HXA raw values for that major frame. The ground derived attitude determination software (ADS) results are also written into the PNT file in the 'sc_pntg' field.

The OBS...

The EVN...

The FEM file has a dataset for each time the S/C has a night to day transition. The times of any SAA passages and station contacts are also saved in those files. The files are created using the orbital solutions which are derived weekly by NASDA.

The SFD files (eg: sfd92_25a.03) are the SXT full-frame desaturated images (combination of long and short exposures to eliminate saturation spikes on the CCD. The two exposures must be taken less than 10 minutes apart, be complete, and not taken during earth eclipse. The background is subtracted using the nearest available suitable dark frame and the exposures and resolutions are normalized.

2.2.3 Yohkoh File IDs

The orbit file ID's are 11 characters long in the format shown below:

```
YYMMDD.HHMM

where YY - Year of data
      MM - Month of data
      DD - Day of data
      HH - Hours
      MM - Minutes
```

2.2.4 Yohkoh Week IDs

The weekly IDs are of the form:

```
YY_WWa.NN

where YY - Year of data
      WW - Week number of the data (1 to 53)
       a - is fixed (reserved for future use)
      NN - is the program version number which created the file
```

The following are the dates covered by each of the weeks for 1991-1993:

Week	*1991*	*1992*	*1993*
01	1-Jan-91 - 5-Jan-91	1-Jan-92 - 4-Jan-92	1-Jan-93 - 2-Jan-93
02	6-Jan-91 - 12-Jan-91	5-Jan-92 - 11-Jan-92	3-Jan-93 - 9-Jan-93
03	13-Jan-91 - 19-Jan-91	12-Jan-92 - 18-Jan-92	10-Jan-93 - 16-Jan-93
04	20-Jan-91 - 26-Jan-91	19-Jan-92 - 25-Jan-92	17-Jan-93 - 23-Jan-93
05	27-Jan-91 - 2-Feb-91	26-Jan-92 - 1-Feb-92	24-Jan-93 - 30-Jan-93
06	3-Feb-91 - 9-Feb-91	2-Feb-92 - 8-Feb-92	31-Jan-93 - 6-Feb-93
07	10-Feb-91 - 16-Feb-91	9-Feb-92 - 15-Feb-92	7-Feb-93 - 13-Feb-93
08	17-Feb-91 - 23-Feb-91	16-Feb-92 - 22-Feb-92	14-Feb-93 - 20-Feb-93
09	24-Feb-91 - 2-Mar-91	23-Feb-92 - 29-Feb-92	21-Feb-93 - 27-Feb-93
10	3-Mar-91 - 9-Mar-91	1-Mar-92 - 7-Mar-92	28-Feb-93 - 6-Mar-93
11	10-Mar-91 - 16-Mar-91	8-Mar-92 - 14-Mar-92	7-Mar-93 - 13-Mar-93
12	17-Mar-91 - 23-Mar-91	15-Mar-92 - 21-Mar-92	14-Mar-93 - 20-Mar-93
13	24-Mar-91 - 30-Mar-91	22-Mar-92 - 28-Mar-92	21-Mar-93 - 27-Mar-93
14	31-Mar-91 - 6-Apr-91	29-Mar-92 - 4-Apr-92	28-Mar-93 - 3-Apr-93
15	7-Apr-91 - 13-Apr-91	5-Apr-92 - 11-Apr-92	4-Apr-93 - 10-Apr-93
16	14-Apr-91 - 20-Apr-91	12-Apr-92 - 18-Apr-92	11-Apr-93 - 17-Apr-93
17	21-Apr-91 - 27-Apr-91	19-Apr-92 - 25-Apr-92	18-Apr-93 - 24-Apr-93
18	28-Apr-91 - 4-May-91	26-Apr-92 - 2-May-92	25-Apr-93 - 1-May-93
19	5-May-91 - 11-May-91	3-May-92 - 9-May-92	2-May-93 - 8-May-93
20	12-May-91 - 18-May-91	10-May-92 - 16-May-92	9-May-93 - 15-May-93
21	19-May-91 - 25-May-91	17-May-92 - 23-May-92	16-May-93 - 22-May-93
22	26-May-91 - 1-Jun-91	24-May-92 - 30-May-92	23-May-93 - 29-May-93
23	2-Jun-91 - 8-Jun-91	31-May-92 - 6-Jun-92	30-May-93 - 5-Jun-93
24	9-Jun-91 - 15-Jun-91	7-Jun-92 - 13-Jun-92	6-Jun-93 - 12-Jun-93
25	16-Jun-91 - 22-Jun-91	14-Jun-92 - 20-Jun-92	13-Jun-93 - 19-Jun-93
26	23-Jun-91 - 29-Jun-91	21-Jun-92 - 27-Jun-92	20-Jun-93 - 26-Jun-93
27	30-Jun-91 - 6-Jul-91	28-Jun-92 - 4-Jul-92	27-Jun-93 - 3-Jul-93
28	7-Jul-91 - 13-Jul-91	5-Jul-92 - 11-Jul-92	4-Jul-93 - 10-Jul-93
29	14-Jul-91 - 20-Jul-91	12-Jul-92 - 18-Jul-92	11-Jul-93 - 17-Jul-93
30	21-Jul-91 - 27-Jul-91	19-Jul-92 - 25-Jul-92	18-Jul-93 - 24-Jul-93
31	28-Jul-91 - 3-Aug-91	26-Jul-92 - 1-Aug-92	25-Jul-93 - 31-Jul-93
32	4-Aug-91 - 10-Aug-91	2-Aug-92 - 8-Aug-92	1-Aug-93 - 7-Aug-93
33	11-Aug-91 - 17-Aug-91	9-Aug-92 - 15-Aug-92	8-Aug-93 - 14-Aug-93
34	18-Aug-91 - 24-Aug-91	16-Aug-92 - 22-Aug-92	15-Aug-93 - 21-Aug-93

Week	*1991*	*1992*	*1993*
35	25-Aug-91 - 31-Aug-91	23-Aug-92 - 29-Aug-92	22-Aug-93 - 28-Aug-93
36	1-Sep-91 - 7-Sep-91	30-Aug-92 - 5-Sep-92	29-Aug-93 - 4-Sep-93
37	8-Sep-91 - 14-Sep-91	6-Sep-92 - 12-Sep-92	5-Sep-93 - 11-Sep-93
38	15-Sep-91 - 21-Sep-91	13-Sep-92 - 19-Sep-92	12-Sep-93 - 18-Sep-93
39	22-Sep-91 - 28-Sep-91	20-Sep-92 - 26-Sep-92	19-Sep-93 - 25-Sep-93
40	29-Sep-91 - 5-Oct-91	27-Sep-92 - 3-Oct-92	26-Sep-93 - 2-Oct-93
41	6-Oct-91 - 12-Oct-91	4-Oct-92 - 10-Oct-92	3-Oct-93 - 9-Oct-93
42	13-Oct-91 - 19-Oct-91	11-Oct-92 - 17-Oct-92	10-Oct-93 - 16-Oct-93
43	20-Oct-91 - 26-Oct-91	18-Oct-92 - 24-Oct-92	17-Oct-93 - 23-Oct-93
44	27-Oct-91 - 2-Nov-91	25-Oct-92 - 31-Oct-92	24-Oct-93 - 30-Oct-93
45	3-Nov-91 - 9-Nov-91	1-Nov-92 - 7-Nov-92	31-Oct-93 - 6-Nov-93
46	10-Nov-91 - 16-Nov-91	8-Nov-92 - 14-Nov-92	7-Nov-93 - 13-Nov-93
47	17-Nov-91 - 23-Nov-91	15-Nov-92 - 21-Nov-92	14-Nov-93 - 20-Nov-93
48	24-Nov-91 - 30-Nov-91	22-Nov-92 - 28-Nov-92	21-Nov-93 - 27-Nov-93
49	1-Dec-91 - 7-Dec-91	29-Nov-92 - 5-Dec-92	28-Nov-93 - 4-Dec-93
50	8-Dec-91 - 14-Dec-91	6-Dec-92 - 12-Dec-92	5-Dec-93 - 11-Dec-93
51	15-Dec-91 - 21-Dec-91	13-Dec-92 - 19-Dec-92	12-Dec-93 - 18-Dec-93
52	22-Dec-91 - 28-Dec-91	20-Dec-92 - 26-Dec-92	19-Dec-93 - 25-Dec-93
53	29-Dec-91 - 31-Dec-91	27-Dec-92 - 31-Dec-92	26-Dec-93 - 31-Dec-93

2.2.5 Yohkoh Carrington IDs

The Carrington IDs are of the form:

```
_crRRRRa.NN

where cr   - signifies a Carrington Rotation ID
      RRRR - is the rotation number
       a   - is fixed (reserved for future use)
      NN   - is the program version number which created the file
```

The following are the dates covered by the Carrington Rotations:

Carrington Rotation	*Starting Date*
1846	21-AUG-91
1847	18-SEP-91
1848	15-OCT-91
1849	11-NOV-91
1850	8-DEC-91
1851	5-JAN-92
1852	1-FEB-92
1853	28-FEB-92
1854	26-MAR-92
1855	23-APR-92
1856	20-MAY-92
1857	16-JUN-92
1858	14-JUL-92
1859	10-AUG-92
1860	6-SEP-92
1861	3-OCT-92
1862	31-OCT-92
1863	27-NOV-92
1864	24-DEC-92
1865	20-JAN-93
1866	17-FEB-93
1867	16-MAR-93
1868	12-APR-93
1869	10-MAY-93
1870	6-JUN-93
1871	3-JUL-93
1872	30-JUL-93
1873	27-AUG-93
1874	23-SEP-93
1875	20-OCT-93
1876	17-NOV-93

3 Introduction to the Yohkoh Software

It is assumed that you have sucessfully installed the Yohkoh software on your system or you are using a machine that has the Yohkoh software. In addition to having made the software installation, it is necessary to have executed the Yohkoh initialization routine ('/ys/gen/script/idl_setup' or '.yslogin' for the Unix machines). If you have not done this yet, please see Volume 2 of the Yohkoh Database and Software User's Guide.

3.1 Interactive Data Language (IDL)

Most of the Yohkoh software is written in the programming language IDL. If certain rules are followed, then the same IDL software can be run on a wide variety of computers.

3.1.1 Some Comments about IDL

In using IDL it is very easy to create many variables to the point where there might be no memory space available. It is possible to delete old variables by using the DELVAR command. For example, if you wanted to delete the variable DATA, you would type:

IDL> delvar, data

There is a peculiarity about using the IDL routine FINDFILE and using the Unix symbol ~ for the user's home directory. FINDFILE does not return any files when using the command:

IDL> ff = findfile('~/*')

It is recommended to use the full path (for example '/2p/morrison/*') instead of ~.

3.1.2 Programs, Procedures, and Functions

An IDL main program requires a .RUN command to run, and the code within that file starts executing after successful compliation.

An IDL procedure is a kind of subroutine and has something like 'PRO PROCEDURE_NAME, PARAM1, PARAM2' at the top of the file. The variables in the procedure definition can be input or output. It is possible to have keywords with a command like 'PRO PROCEDURE_NAME, KEY1=KEY1, KEY2=KEY2'. It can be executed with a command like:

```
IDL> procedure_name, a, b
IDL> procedure_name, a, b, key1=c
```

An IDL function is another kind of subroutine and has something like 'FUNCTION FUNCTION_NAME, PARAM1, PARAM2' at the top of the file. The primary output is passed to *result* but it is possible to have parameters in the call which can receive output. It is executed with a commadnd like:

```
IDL> result = function_name(a,b)
```

3.1.3 Running IDL programs

When running on a Unix system, it is important to use lower case when using the .RUN command because Unix is case sensitive. This is not necessary when doing an 'implied' compilation when accessing procedures or functions since IDL will convert to lower case for you. You cannot use upper case for a procedure to do an explicit compilation (.RUN PROCEDURE_NAME) to recompile a procedure.

3.2 Directory Organization

The Yohkoh software is organized under one tree. The top directory is $ys for Unix systems, and is the logical YS: for the VMS systems. Under that tree are the following.branches:

Branch	*Description*
SITE	Site specific software
GEN	General software and documents that all instruments can use
BCS	BCS specific software and documentation
HXT	HXT specific software and documentation
SXT	SXT specific software and documentation
WBS	WBS specific software and documentation
ATEST	Newly created or modified software and documentation
UCON	User Contributed software and documentation

Under each of the above instrument directories, there are the following directories (for example, '/ys/sxt/doc')

Branch	*Description*
DOC	Documentation
RESPONSE	Instrument calibration and response data files
SOFT	Software
STATUS	Instrument status information

Software that has been thoroughly tested is put into the instrument release directory 'soft'. (The previous organization had an 'atest', 'rel', and 'usercontrib' branch under each 'soft' directory) Software which has just been written, or software that is modified is put into the 'atest' area for a period of a few weeks. Users only have write priveledge to that directory and they need to use the IDL routine ADD_PRO to put some software on-line. If problems develop with modified software, it is possible to recover the old version by copying it from the 'soft' directory. The software developed by general users is placed online under the 'ucon' (usercontrib) branch. Generally each user who is contributing has a directory of their own. The directories under the 'soft' branch are broken up by function for the 'gen' and instrument branchs, but by person under the 'ucon' branch. If a user cannot remember the name of a function, he/she can do a listing on the different directories and he/she will probably recognize it. A list of the directories that currently exist for the instrument teams is:

GEN	*BCS*	*HXT*	*SXT*	*WBS*
dbase	bda	util	register	util
gbo	bsd		sensitivity	
jhuapl	junk		util	
movie	util		widgets	
orbit				
pointing				
ref_access				
reformat				
tape				
util				
utplot				

4 Getting Started with Yohkoh Data

4.1 YODAT (formally TEST_RD)

YODAT will access any data from BCS, HXT, SXT, or WBS. It will also read the FITS files which have been renamed to use the Yohkoh convention.

1. Shows what data is on line
2. Lets you select specific files to read and disply
3. Reads the ROADMAP for the selected files
4. If you select data sets to be read, it creates the variables DATA and INDEX.

This procedure is run by typing (make sure that yodat is in lower case if you are running on a Unix or Ultrix machine):

IDL> .run yodat

The prompt that you will receive will look something like this:

```
% Compiled module: $MAIN$.
               *******  YODAT V9.0 (26-Oct-92)  *******

It is possible to have YODAT extract every "n"th dataset by setting
the variable QYODAT_NSAMP to 1.  You will be asked one extra question

It is possible to read the Ground Based Observation (GBO) FITS files
by using a command like: MENU g*
         gb_ files are from Big Bear, gk_ are from Kitt Peak
RFITS will be called with /SCALE option if QYODAT_SCALE is set to 1

Enter MENU if you want to use the filenames menu option
Enter SAME if you want to access the same fileID for a different instrument
Enter MANY if you want to use menu option and extract many files
Enter TIME if you want to enter the start/end time to extract
```

```
Enter QUIT to abort out of YODAT

Enter file name (or wild cards)
```

The first step is to select the data files to be read. The name of the file(s) selected is saved in the variable INFIL. When a file is selected, the roadmap for those files are read into the variable ROADMAP. There are several different techniques for selecting files.

- You can type the full file name, including the directory if the data file does not exist in the default directory. Wild cards are acceptable.
- You can use the MENU option to get a listing of all of the files that are on-line (this option uses the output of DATA_PATHS to get a list of the data directories). For example, if you type MENU SPR*, you will get a list of all SPR files that are on-line. Of if you type MENU BDA9207* you get a list of all BDA files for Jul-92 that are on-line. Click on the file you wish to read.
- You can use the MANY option to select several files. This option is also menu driven, and is very similar to the MENU option. After you have selected all of the files you want to read, click on QUIT/EXIT.
- If you have already read a file, and wish to get the same file for a different experiment, you can use the SAME option. For example, if you had just read spr911115.2141, and want to get the BDA file for that orbit, type SAME BDA.
- Once you have selected a file, you can re-run YODAT and simply hit RETURN and the same file is selected again.
- If you type QUIT, then YODAT will be stopped.

The second step is to perform a quick review of the data available, and to select the data sets to be read. The options for selecting the data are listed below.

```
Enter the number of data sets to extract
   * If you enter 0, all datasets will be extracted
```

```
* If you enter -99, then it uses the datasets specified in variable "SS"
* If you enter -888, then the file is not read
* For SXT, enter a negative # (from -1 to -13) to access only that seq#
* For SXT, enter -777 for seqence menu option
           Enter -776 to use "show_obs3" and select
           Enter -775 to use "plot_fov" and select
           Enter -774 to list the sequence summary
           Enter -773 to use "show_obs4" and select
* For HXT, enter -666 to plot and select on SUM_L light curve
           Enter -665 to plot and select on SUM_M1 light curve
           Enter -664 to plot and select on SUM_M2 light curve
           Enter -663 to plot and select on SUM_H light curve
* For WBS, enter -555 to plot and select on SXS1 light curve
           Enter -554 to plot and select on SXS2 light curve
           Enter -553 to plot and select on HXS light curve
* For BCS, enter -444 to plot and select on S XV light curve
           Enter -443 to plot and select on Ca XIX light curve
           Enter -442 to plot and select on Fe XXV light curve
           Enter -441 to plot and select on FE XXVI light curve
* For any, enter -333 to extract only flare mode data
```

If you selected data to be read, then the results are saved in the following variables:

infil	the input file(s)
index	the index structure for datasets extracted
data	the data (array or structure) for datasets extracted
roadmap	the complete roadmap for all datasets in *infil*
dset_arr	the list of dataset numbers selected
info_array	for SXT only, a text array describing each image

Most of the options are self explanatory. For SXT options 'SHOW_OBS3' and 'SHOW_OBS4', please see the description in the later chapeters of this guide.

The light curve options (all of the -400, -500, and -600 series) will show a light curve of the selected channel. It is possible to select the range in time you wish to look at by:

1. Click with the left key at the starting time
2. Click with the right key at the ending time
3. Click with the middle key to exit

An example of how one would use the −99 option is:

1. Run YODAT to select the file and to read the roadmap
2. Exit out of YODAT with the −888 option
3. Select the data to be extract using SSWHERE or some other option, putting the results in the variable 'ss'
4. Run YODAT again selecting the same file by just hitting RETURN
5. Selecting option −99

4.2 Most Common BCS Display Routines

After having read the BCS data with YODAT, it is common to use the following commands. Experiment and enjoy.

```
IDL> plott_bda, index
IDL> plott_bda, roadmap, psym=10
IDL> plots_bda, index, data
```

4.3 Most Common HXT Display Routines

After having read the HXT data with YODAT, it is common to use the following commands. Experiment and enjoy.

```
IDL> plott_hda, index
IDL> plott_hda, roadmap, psym=10
IDL> plots_hda, index, data
```

4.4 Most Common SXT Display Routines

After having read the SXT data, it is common to use the following commands. Experiment and enjoy.

```
IDL> stepper, data
IDL> stepper, data, xsiz=512
IDL> stepper, data, info=info_array
IDL> show_obs3, index
```

4.5 Most Common WBS Display Routines

After having read the WBS data with YODAT, it is common to use the following commands. Experiment and enjoy.

```
IDL> plott_wda, index
IDL> plott_wda, roadmap, psym=10
IDL> plots_wda, index, data
```

4.6 Advanced YODAT Options

It is possible to use the −777 option to select SXT data by the sequence table entry. If you wish to abort from this option, click on the title (the heading of the table). The most common technique is to just select one sequence entry, but it is possible to select many entries. The procedure to do selection option −777, and then:

1. Click on "Enable option to select mutiple sequences (reset)"
2. Click on all of the sequence entries you wish to extract
3. Click on "If enabled selecting multiple options, now extract data"

5 Data Selection Routines

5.1 TIM2DSET

TIM2DSET will take the roadmap and return the dataset number which is closest to that time. Examples,

```
IDL> dset = tim2dset(roadmap, input_time)
IDL> dset = tim2dset(roadmap, '23-jun-92 6:00')
```

5.2 SSWHERE (SXT) [*]

SSWHERE creates an array of subscripts (ss) that fulfill a set of criteria that you define to select SXT images, for example to make an ss array to be used by YODAT where you want to select images from a data set represented by roadmap, you merely type:

```
IDL> ss=sswhere(roadmap)
```

SSWHERE works on both roadmap and index. The available selection criteria are

- completeness of the image
- SXT mode (normal/dark image/calibration)
- pixel resolution
- compression mode
- filters
- exposure DPE

It is a widget driven program.

5.3 SHOW_OBS3 (SXT)

SHOW_OBS3 will plot a time line summary of the SXT images that are in the index or roadmap. In addition to plotting a tick for each image available, it also shows the DP mode, DP rate, and where the S/C days are SAA passages occur. Some sample calls to SHOW_OBS3 are:

```
IDL> show_obs3, roadmap
IDL> show_obs3, index
```

It is possible to select images using SHOW_OBS3 by clicking twice on diagonally opposite corners of a box surrounding the images you wish to select. The calling sequence for that is:

```
IDL> show_obs3, roadmap, sel, ss
```

where *sel* is a returned logical array the same length as roadmap, and is set true for all dsets selected, and *ss* is the returned subscripts of the selected datasets. A few examples on how to use the *sel* variable are:

```
IDL> ss = where(sel)
IDL> ss = where(sel and (gt_dp_mode(roadmap) eq 9))
```

5.4 SHOW_OBS4 (SXT)

The calling sequence of SHOW_OBS4 is the same as SHOW_OBS3, except there are a few additional options in SHOW_OBS4. Run SHOW_OBS4 with one of the following commands:

```
IDL> show_obs4, roadmap
IDL> show_obs4, index
```

It is possible to click on a time range for a closer look by:

1. Click with the left button at the start tiem
2. Click with the right button at the end time

3. Click with the left button on the box in the lower right
4. If you wish to exit, click with the middle button somewhere on the plot

If you want to select regions, it is possible to select several regions using the following steps.

1. Click with the right button on the box in the lower left (Select Option). A square box should appear on the screeen.
2. Click and hold the left button on the lower left corner of the box and drag the box to the lower left corner of the region you want to select. Then release.
3. Click and hold the middle button on the upper right corner of the box and stretch the box to encompass the images that you want to select.
4. Repeat steps 2 and three until you have positioned the box where you want it.
5. Click on the right button to exit the selection option when you have successfully positioned the box.
6. Now you can exit SHOW_OBS4 by clicking on the middle button, or you can select another set of images by running steps 1 through 5 again.

5.5 LIST_BDA (BCS)

A sample call would be:

IDL> list_bda, roadmap, start, nrec, ss=ss

where *roadmap* is the input.

5.6 SEL_BDA (BCS) [*]

Use SELECT_BDA to plot the roadmap light curve of a selected channel and select the time period of interest using the cursors. (Note: there is an option to read the data in with select_bda if required).

A sample call would be:

```
IDL> .run sel_bda
```

where *roadmap* is the input.

5.7 WMENU_SEL

WMENU_SEL is an expansion on the IDL WMENU routine. It allows for many menu pages if the number of items will not fit on one page. It also allows a user to select several items. Some sample calls are:

```
IDL> ss = wmenu_sel(array)
IDL> ss = wmenu_sel(files, /one)
```

6 Time Plotting Routines

6.1 UTPLOT

UTPLOT enables one to plot any quantity on a time plot with hours, minutes, and seconds given a corresponding index array by typing:

```
IDL> utplot, roadmap, gt_total_cnts(roadmap, 1)
IDL> utplot, index, timehist
IDL> utplot, x, y, ref_time, xrange=xrange
IDL> utplot, x, y, ref_time
```

Most typical IDL plotting parameters also apply. You can overplot other quantities (timehist2, etc.) by using OUTPLOT. This routine is a modified version of the SMM UTPLOT routine.

6.2 OUTPLOT

OUTPLOT works just like UTPLOT but does not refresh the screen so one can plot several different quantities on the same plot (see UTPLOT first). An example is:

```
IDL> outplot, index, timehist2
```

6.3 PLOTY

PLOTY enables you to plot the lightcurves from all of the Yohkoh instruments on the same plot and the same time axis. If you have the observing log on-line and wish to use it to make the plots, you can specify the plot times. It is recommended not to plot more than approximately 24 hours of data at a time. An example is:

```
IDL> ploty, '8-may-92 15:00', '8-may-92 18:40'
```

If you have identified a instrument file for which you want to see the light curve for all of the other instruments, you can use the following command (it assumes that the other instrument data files exist on the same directory

as the input file):

```
IDL> ploty, infil='/yd5/flares/spr911115.2141'
```

You can read the observing log data, and the use PLOTY with a command like:

```
IDL> rd_obs, '8-may-92', '8-may-92 12:00', bcs, sxtf, sxtp, w_h
IDL> ploty, bcs, w_h, sxtp, w_h
```

6.4 TPROFILES

TPROFILES will plot the lightcurve for any pixel in a data cube (data(x,y,t)). To use it just type:

```
IDL> tprofiles,data
```

The program is mouse driven and will issue instructions for you.

6.5 BCS_24HR_PLOT

If you interested only whether the BCS has seen anything, use BCS_24HR_PLOT. This works using the only roadmap data and allows intervals to de selected by date (all files covering this data are then used). By default, channel 3 (Ca XIX) is plotted. Note: YODAT does NOT need to have been run before BCS_24HR_PLOT. Samples are:

```
IDL> bcs_24hr_plot, '15-nov-91'
IDL> bcs_24hr_plot, '15-nov-91', chan=4
```

6.6 PLOTT_BDA (BCS)

This routine will make a light curve plot using either the index or roadmap. The second 'T' signifies that it is a time plotting routine. Four plots are made on the page, one for each channel. Some sample calls are:

IDL> plott_bda, index
IDL> plott_bda, roadmap
IDL> plott_bda, roadmap(100:200), psym=10

6.7 LCBDA (BCS) [*]

This routine will make a light curve plot using either the index or roadmap. It is almost identical to PLOTT_BDA except that you can specify a channel and only get that channel plotted. Some sample calls are:

IDL> lcbda, index
IDL> lcbda, roadmap, chan=1

6.8 PLOTT_HDA (HXT)

This routine will make a light curve plot of all four channels using either the index or roadmap. The second 'T' signifies that it is a time plotting routine. Four plots are made on the page, one for each channel. Some sample calls are:

IDL> plott_hda, index
IDL> plott_hda, roadmap
IDL> plott_hda, roadmap(100:200), psym=10

6.9 PLOTT_WDA (WBS)

This routine will make a light curve plot for all channels using either the index or roadmap. The second 'T' signifies that it is a time plotting routine. Seven plots are made, one for each channel. Some sample calls are:

IDL> plott_wda, index
IDL> plott_wda, roadmap
IDL> plott_wda, roadmap(100:200), psym=10

6.10 BOX_LC [*]

Plot time series for an interactively determined box of pixels within a given data cube. Calls STEPPER to present an image to interact with. Does EXP_NORM. Plots the results. A sample call is:

```
IDL> box_lc, data, index, timeseries, boxout
IDL> box_lc, data, index, timeseries, boxout, /ave
```

where *data* and *index* are input (from an SPR file). *timeseries* is the output in DN/sec, and *boxout* are the coordinates selected. If */ave* is used then the time series is in DN/sec/pixel (the box average).

7 Spectral Plotting/Display Routines

7.1 PLOTS_BDA (BCS)

This routine allows for the spectra for all four channels to be plotted to one page. A sample calling sequence is:

IDL> plots_bda, index, data

7.2 PLOTBDA (BCS)

The light-curve and spectra from the BDA file may be plotted using PLOTBDA. Thus is an interactive program and it will read its own data.

IDL> .run plotbda

7.3 BCS_MULTI (BCS)

Many spectra may be plotted on a page with BCS_MULTI.

IDL> bcs_multi, index, data
IDL> bcs_multi, index, data, chan=1

7.4 BCS_CONT (BCS)

The evolution of spectra against time can be displayed using a contour using the routine BCS_CONT. Note: The time axis of BCS_CONT is uniform, but that of DISP_BDA and GS are not.

IDL> bcs_cont, index, data
IDL> bcs_cont, index, data, chan=1

7.5 DISP_BDA (BCS)

The evolution of spectra against time can be displayed as a pseudo-image using the routine DISP_BDA. Note: The time axis of DISP_BDA is not

uniform.

IDL> disp_bda, index, data

7.6 DISP_HDA (HXT)

The evolution of sensor intensity against time can be displayed as a pseudo-image using the routine DISP_HDA. Note: The time axis of DISP_HDA is not uniform.

IDL> disp_hda, index, data

7.7 DISP_WDA (WBS)

The evolution of all of the WBS spectra against time can be displayed as a pseudo-image using the routine DISP_WDA. Note: The time axis of DISP_WDA is not uniform.

IDL> disp_wda, index, data

7.8 BCS_SPMOVIE (BCS)

After selecting a time interval, a movie of the changing spectra for a given channel may be displayed by BCS_SPMOVIE. Note: This program will only run on an X-windows workstation.

IDL> bcs_spmovie, index, data
IDL> bcs_spmovie, index, data, chan=1

7.9 GS (BCS) [*]

If you are interested in more detail of what spectra the BCS has observed, use GS - this routine must be run after the data has been read in with YODAT. There is an upper limit of how much data can be handled at a time (about 900 spectra), and some selection of the required time interval may be needed using the methods described above, but for the per-orbit files, this

limit may not be a problem. The advantage of GS is that you can get a good idea of what has been seen in the spectra - a particularly useful tool if you are trawling for data. Warning: The time axis on the greyscale plot is not uniform!

```
IDL> .run gs
```

All the routines that run from GS require that spectra be selected first using the cursor routine.

```
IDL> .run gs_cur
```

8 Image Display and Enhancement Routines

8.1 STEPPER

STEPPER takes a data cube (*data*) and information array (*info_array*) from the output of YODAT and will animate it. Some sample calls are:

```
IDL> stepper, data
IDL> stepper, data, info=info_array
IDL> stepper, data, info=info_array, xs=512
```

where *info* and *xs* are optional parameters. *xs* indicates the size of the displayed image you want (must be an integer multiple of the the original array). It is a window driven program.

8.2 XSTEPPER

XSTEPPER is like stepper but is a widget based program, to run it type:

```
IDL> xstepper, data, info_array
```

It is a widget driven program with self explanatory instructions. Some options include the ability to zoom in on a particular region and the ability to call XLOADCT from within XSTEPPER.

8.3 OCONTOUR

OCONTOUR allows you to plot a contour on top of an image in the same way as you use CONTOUR.

```
IDL> ocontour,image
```

8.4 XY_RASTER (SXT)

Makes a mosaic of several images and prints the times in the corner of each by typing:

IDL> xy_raster, data, index, factor

where *factor* is the rebin factor (a value of 1 will perform no rebinning)

8.5 UP2, UP4, UP8

To re-bin and display an image, you can use the UPn routines. UP2 will rebin the image to be twice as big and then call TVSCL. UP4 and UP8 increase the size of the image 4 and 8 times respectively. A sample call would be:

IDL> up8, data(*,*,200)

8.6 UNSHARP_MASK

This is a basic enhancement routine with a calling sequence:

IDL> dataout=unsharp_mask(index, data)

where *data* and *dataout* are 2D arrays, and *index* is the index entry that corresponds to *data*. If no keywords are given the default is used. This routine was originally called ENHANCER.

Optional Keyword Parameters:

`smooth`	Size of smoothing box, integer.
`lowdelt`	Lower δ cutoff, a percentage of minimum, decimal 0.xx.
`updelt`	Upper δ cutoff, a percentage of maximum, decimal 0.xx.
`lowint`	Saturation cutoff, don't add back when signal is BELOW a given percentage of the saturation value, decimal 0.xx.
`upint`	Saturation cutoff, don't add back when signal is ABOVE a given percentage of the saturation value, decimal 0.xx.
`deltcoef`	Multiplier for δ field, float x.x.

The keyword parameters allow you to customize the enhancer, though they are all assigned default values if left out of the invocation. These defaults depend on the images resolution, and may be changed as we gain experience.

8.7 SXT_GRID (SXT)

Draws heliocentric grid over an image. It reads the PNT file to get information on the spacecraft pointing, and also calls GET_RB0P to get the solar radius. A sample call could be:

```
IDL> tvscl, data(*,*,0)
IDL> sxt_grid, index(0)
```

8.8 Standard IDL Routines

8.8.1 TV [IDL]

The command TV takes a 2-dimensional variable and displays it directly to the display (you must have x-windows or a image display device). There is no scaling performed on the data. If the data is not byte type, and there are values over 255, then wrap-around will occur. It is probably best to use TVSCL for these datasets. To display the first image in the variable *data*, type:

```
IDL> tv, data(*,*,0)
```

To display the "i"th dataset, type

```
IDL> tv, data(*,*,i)
```

8.8.2 TVSCL [IDL]

TVSCL will first perform a linear byte scaling, making the smallest value 0 intensity and the largest value 255. To display image number 11 in the variable *data*, type:

```
IDL> tvscl, data(*,*,11)
```

8.8.3 CONTOUR [IDL]

You can contour any given image by typing:

```
IDL> contour,image
IDL> contour, data(*,*,0)
IDL> contour, image, levels=[l1,l2,l3,...,ln]
```

where levels defines the contour levels you want to plot. *levels* is an optional parameter which if omitted IDL will choose them for you. You can overplot a contour on an image by using OCONTOUR in the same way.

8.8.4 XLOADCT [IDL-LIB]

XLOADCT is a widget driven IDL facility that allows you to change colour tables and manipulate them. To run it type

```
IDL> xloadct
```

8.8.5 LOADCT [IDL-LIB]

LOADCT will load one of the 16 standard IDL colour tables. To load the red color table, type:

```
IDL> loadct, 3
```

8.8.6 TVLCT [IDL]

TVLCT allows a user to load a color table which he has defined alread. After defining the red, green, and blue intensity profiles, type:

```
IDL> tvlct, red, green, blue
```

to load the color table. If you wish to read the color table which is currently loaded, type:

```
IDL> tvlct, red, green, blue, /get
```

8.8.7 MOVIE [IDL-YO]

The MOVIE routine is an IDL routine that displays animates a image data cube (x,y,t). The data should be byte type. It is run by typing:

IDL> movie,data

You can vary the speed of the movie. No information is printed with the image, for that see STEPPER and XSTEPPER. The standard IDL routine was modified to make !ORDER default to zero which is the proper orientation for Yohkoh images.

8.8.8 PALETTE [IDL-LIB]

PALETTE is an IDL facility that allows you to create a new colour table. To run it just type:

IDL> palette

To save the colour table use TVLCT by typing

IDL> tvlct,r,g,b,/get

This will save the red, green, and blue vectors in *r*, *g*, and *b*. To re-load this color table, type:

IDL> tvlct,r,g,b

8.8.9 PROFILES [IDL]

PROFILES enables you to obtain a plot of a row or column of a two dimensional image by typing

IDL> profile,image

you should have image in the format (size and normalization) that you wish first. It is a mouse driven program.

8.8.10 ZOOM [IDL-LIB]

This routine takes an image that is already being displayed and displays the zoomed region in a new window. The user uses the cursor and clicks with the left button to chose the center of the region to zoom (the selected region is displayed each time the left button is clicked). The middle button allows the zoom factor to be selected using a menu. The right button exits the zoom option (the zoom window is deleted upon exiting ZOOM). An example would be:

```
IDL> tvscl, data(*,*,5)
IDL> zoom
```

9 Information Extraction Routines

9.1 Data Extraction

9.1.1 MK_MOSAIC (SXT)

Since the SXT Automatic ROI Selection (ARS) software might change the pointing as the active region is tracked, it is necessary to have a routine which will take a set of images and re-register them to the same location. MK_MOSAIC will do that. In addition, if a routine like GET_PNT has been run and the offset due to S/C pointing drift is known as an offset relative to a given time, that correction can also be made. Sample calling sequences are:

```
IDL> data_out = mk_mosaic(data, index)
IDL> data_out = mk_mosaic(data, index, offset=offset)
```

9.1.2 EXT_SUBSET (SXT)

EXT_SUBSET extracts a subset (in x and y) of a data cube and creates a new data and index array (*data_out* and *index_out*) by typing:

```
IDL> ext_data,index_in,data_in,index_out,data_out
```

It is a mouse driven program.

9.1.3 EXT_BCSCHAN (BCS)

The spectra for a single channel may be extracted with EXT_BCSCHAN

```
IDL> data_out = ext_bcschan(index, data, chan)
```

9.1.4 CONROI [*]

Define a contoured region-of-interest of an image using the image display system and the cursor/mouse. This function returns a vector containing the subscripts of pixels inside the CONTOURED region. Note that these are 'linear' subscripts, not 'X, Y' pairs.

```
IDL> result = conroi( data(*,*,n) )
IDL> result = conroi( data(*,*,n), zoom=4)
```

9.1.5 PR_IMAGE

PR_IMAGE is used to get a print out and/or the average count rate in a given area on an image (*data(*,*,n)*) by typing:

```
IDL> pr_image, data(*,*,n)
IDL> pr_image, data(*,*,n), size
```

where *size* is the size of the box to be considered (default is 9x9). It is a mouse driven program.

9.2 The Concept Behind the GT Routines

A series of GT routines was created so that a single piece of information can be extracted from a structure, whether the structure was a 'roadmap' or an 'index' (the data is saved in a different location). The routines also allow for conversions to a string mnemonic or to physical units. It is possible to get a list of what the different values by typing a command like:

```
IDL> print, gt_filta()
```

It is also possible to convert the output value to the string mnemonic by using the */STRING* switch. For example:

```
IDL> print, gt_filta(roadmap, /string)
```

This command only works for the routines which are returning a coded value. A list of all of the GT routines currently available are:

GEN	*BCS*	*HXT*	*SXT*	*WBS*
gt_day	gt_blockid	gt_sum_h	gt_comp	gt_grs1
gt_dp_mode	gt_total_cnts	gt_sum_l	gt_dpe	gt_grs2
gt_hxa		gt_sum_m1	gt_entry	gt_hxs
gt_iru		gt_sum_m2	gt_expdur	gt_rbmsc
gt_tfss			gt_expmode	gt_rbmsd
gt_conv2str			gt_filta	gt_sxs1
gt_day			gt_filtb	gt_sxs2
gt_dp_mode			gt_fov_center	
gt_dp_rate			gt_mbe	
gt_time			gt_obsregion	
			gt_or_expnum	
			gt_pfi_ffi	
			gt_res	
			gt_seq_num	
			gt_seq_tab	
			gt_shape_cmd	
			gt_ssl_explab	
			gt_temp_ccd	
			gt_temp_hk	
			gt_pix_size	

9.3 General

9.3.1 GT_DP_RATE

This routine will extract the information on the DP telemetry rate. The values are shown below.

1	Low (1 Kbits/sec)
2	Med (4 Kbits/sec)
4	High (32 Kbits/sec)

IDL> dprate = gt_dp_rate(index)

IDL> dprate = gt_dp_rate(roadmap, /string)

9.3.2 DPRATE2SEC

This routine will take a structure (or an integer value 1, 2 or 4 for low, medium, or high) and return the number of seconds that passes for a major frame at that telemetry rate.

1	64 sec (Low)
2	16 sec (Med)
4	2 sec (High)

IDL> nsec = dprate2sec(index)

9.3.3 GT_DP_MODE

This routine will extract the information on the DP mode. There are several modes, but the most common are:

9	Flare
11	BCS-OUT
12	Night
13	Quiet

IDL> dpmode = gt_dp_mode(index)
IDL> dpmode = gt_dp_mode(roadmap, /string)

9.4 BCS

9.4.1 LIST_BDA

If you are interested in what modes the BCS was executing, or what the countrate in a particular channel has observed at a particular time, this can be determined by using LIST_BDA. By default the count rate for channel 3

(Ca XIX) is given in the listing. Sample calls are:

```
IDL> list_bda, roadmap, start, nrec
IDL> list_bda, roadmap, start, nrec, chan=4
```

where *roadmap* is the input.

9.4.2 GT_TOTAL_CNTS

This routine extracts the information from the TOTAL_CNTS field which took the actual spectra and totaled the number of counts. The routine also normalizes so that it returns counts/sec. In the first example below, all four channels are returned, so the output is 4xN. In the second example, only channel 1 is extracted. It is possible to get a string defining the channel selected by using the *title* keyword option.

```
IDL> x = gt_total_cnts(roadmap)
IDL> x = gt_total_cnts(roadmap,1)
IDL> x = gt_total_cnts(index,2,title=title)
```

9.4.3 GT_BLOCKID

This routine returns information on how the BCS data was blocked.

```
IDL> x = gt_blockid(roadmap)
```

0	Normal Queue Data Block
1	Fast Queue Data Block
2	Micro Dump Block (fixed extraction)
3	Cal Data Block (fixed extraction)
4	Queue data where the modeID in the header is not recognized
5	Normal or fast queue data which have fill data (garabage)

9.5 HXT

9.5.1 GT_SUM_L, GT_SUM_M1, GT_SUM_M2, GT_SUM_H

These routines will extract the average counts/sec of all 64 sensors. The input can be roadmap, index, or observing log and it will get the proper structure tag and decompress it properly to return counts/sec/sensor. It is possible to get a string defining the channel selected by using the *title* keyword option.

```
IDL> y = gt_sum_l(roadmap)
IDL> y = gt_sum_l(w_h, title=title)
```

9.5.2 still need a GET_INFO type routine

9.6 SXT

9.6.1 GET_INFO2 [*]

GET_INFO2 will take the roadmap or index structure and return a string describing the main observing mode parameters for each dataset.

```
IDL> info_array = get_info2(roadmap)
```

9.6.2 GT_FILTA

GT_FILTA enables you to find from an index or roadmap structure what the setting of the A filter was by simply typing:

```
IDL> a = gt_filta(index)
```

The filter setting will be returned as an integer (1-6) where

1	Open (Op)
2	Optical - narrow band (NaBan)
3	Quartz defocusing lens (Quart)
4	Diffuser (Diffu)
5	Optical - Wide band (WdBan)
6	8% neutral density X-ray filter (NuDen)

It is possible to get a string type description of the filters used by typing:

IDL> b = gt_filtb(roadmap, /string)

You can get a listing of these at anytime by typing the following:

IDL> print,gt_filta()

9.6.3 GT_FILTB

Exactly like GT_FILTA but works for the X-ray filters

1	Open (Op)
2	Thin Aluminum (Al.1)
3	Dagwood Sandwich (AlMg)
4	Berilium (Be119)
5	Thick Aluminum (Al12)
6	(Mg3)

9.6.4 GT_RES

Extracts what resolution each SXT image was taken at by typing:

IDL> res = gt_res(roadmap)

It outputs the resolution as a integer where:

0	full resolution
1	half resolution
2	quarter resolution

It is possible to get a string type description of the resolution by typing:

IDL> res = gt_res(roadmap, /string)

9.6.5 GT_COMP

Extracts what compression was used for each SXT image was taken at by typing:

IDL> comp = gt_comp(roadmap)

It outputs the compression as a integer where:

0	Compressed (see routine SXT_DECOMP)
1	Low 8 bits (of 12 bits)
2	High 8 bits (of 12 bits)

9.6.6 GT_DPE

Extracts what DP exposure (DPE) level was used. The DPE is a function of the shutter expsoure level (MBE) and the neutral density filter use. Please see the 'Red Book' for a full description of the DPE table. It is possible to effective exposure duration in milliseconds by using the */CONV* switch).

IDL> dpe = gt_dpe(index)
IDL> msec = gt_dpe(roadmap, /conv)

9.6.7 GT_MPE

Extracts what mail box exposure (MBE) level was used. This is the actual shutter exposure level commanded. It is possible to convert to commanded shutter duration in milliseconds by using the /CONV switch).

IDL> mbe = gt_mbe(index)
IDL> shutdur = gt_mbe(roadmap, /conv)

9.6.8 GT_EXPDUR

GT_EXPDUR will calculate the effective exposure duration not the commanded DPE) of a series of images given an index structure to work with. You can get an array of exposures (exps) in msec by typing one of the following. In the second example, it returns the shutter duration (which is also the CCD integration period).

IDL> expos = gt_expdur(index)
IDL> expos = gt_expdur(index, /shut_dur)

9.6.9 GT_EXPMODE

This routine returns the exposure mode. The values it can return are:

0	Normal shuttered exposure
1	Dark image (shutter stays closed)
2	Calibration image (shutter open during readout)

IDL> expmode = gt_expmode(index)

9.6.10 GT_FOV_CENTER

The output from this routine is arcminutes from a fixed location on the CCD. The fixed location is column 512, line 638. The results for GT_FOV_CENTER are currently not begin corrected for changes in S/C pointing.

IDL> fov_center = gt_fov_center(index)

where *fov_center(0,*)* is arcminutes east/west with east being negative and *fov_center(1,*)* is acrminutes north/south with north being positive. The

routine will be upgraded shortly to use S/C pointing history to get a true location on the sun.

9.6.11 GT_TEMP_CCD

This routine returns the temperature of the CCD in degrees celsius.

IDL> temp = gt_temp_ccd(index)

9.6.12 GT_TEMP_HK (SXT)

This routine returns the temperature measured by the platinum resistance thermometer (PRT) on the spacecraft and SXT instrument. There are 16 temperatures saved in the SXT index.

IDL> temp = gt_temp_hk(index, 0, title=title)

9.6.13 GET_PIX_COOR

This routine returns the coordinate from the bottom left of a full-frame image given an index structure by typing:

IDL> coor = get_pix_coor(index(0), offset=b)

where b is the offest and coor is the pixel location.

9.7 WBS

9.7.1 GT_SXS1, GT_SXS2, GT_HXS, GT_GRS1, GT_GRS2, GT_RBMSD, GT_RBMSC

These routines will extract the average counts/sec for the selected channel.

SXS1	Only SXS_PC21	3-15 keV
SXS2	Only SXS_PC22	15-40 keV
HXS	HXS_PC1 plus HXS_PC2	20-600 keV
GRS1	GRS_PC11 plus GRS_PC21	0.2-0.7 MeV
GRS2	GRS_PC12 plus GRS_PC22	0.7-4 MeV
RBMSD	PC1 plus PC2	5-300 keV
RBMSC	Only RMSSC	20 keV

The input can be roadmap, index, or observing log and it will get the proper structure tag and decompress it properly to return counts/sec/sensor. It is possible to get a string defining the channel selected by using the *title* keyword option.

IDL> y = gt_sxs1(roadmap)
IDL> y = gt_hxs(w_h, title=title)

9.7.2 still need a GET_INFO type routine

10 Calibration and Analysis Routines

10.1 General

10.1.1 RM_DARKLIMB [*]

In order to subtract the limb darkening from Hα or white light pictures you can call:

IDL> img_out = rm_darklimb(Haimage)

where *Haimage* is the name of the Hα image with limb darkening and *img_out* is the new Hα image with limb darkening removed. This routine does not work if the image is oblate. The image should then be viewed with TVSCL (not TV).

10.2 BCS

10.2.1 BCS_DECOMP

The BCS data is normally compressed from a 12-bit? word to an 8-bit word. BCS_DECOMP takes the compressed data array (*data*) and creat an integer array (*ndat*) of the decompressed numbers by typing:

IDL> ndat = bcs_decomp(data)

10.2.2 BCS_NORM

This routine normalizes data recorded by the BCS for time, and extract the Fast Queue data if present. A sample call is:

IDL> dspec = bcs_norm(index, data)

10.2.3 SUM_BDA

A number of spectra may be summed with SUM_BDA

IDL> data_out = sum_bda(index, data, modeid, nsum)

10.2.4 PLOT_REF [*]

Overplot several spectra to show evolution of blue wing and line width.

IDL> plot_ref, index, data, channel, dset_arr

10.2.5 MKBSD and BSDCAL

There are three programs that have been written in standard Fortran-77 to reduce the BCS data to fitted spectral parameters. These are MKBSD, BSDCAL and BSDFIT.

MKBSD extracts the spectra, applies instrument corrections and writes a new file, the BSD file.

BSDCAL fits selected lines in the spectra of the BSD file with Voigt profiles to produce a calibrated wavelength scale and writes this fit information to a BPC file.

BSDFIT fits a full theory spectra to the calibrated BSD data to give electron temperature, emission measure and plasma velocity. The fit parameters are written to a BFT file. The output theory spectra are written to the BTH file.

These routines are being re-written into IDL so they will not be discussed at this time.

10.3 HXT

10.3.1 HXT_IMG [*]

Jim McTiernan took the FORTRAN program which was running on the mainframes at ISAS and converted the program to IDL. The results have not been completely verified and the user interface might change. It takes very long to create a single image (between 1 and 30 minutes depending on the intensity) so we recommend running in batch mode when creating more than a couple of images.

1. The HDA data must be read into the variables *data* and *index*. A common way to do this is to use YODAT.

2. Now start running HXT_IMG by typing

   IDL> .run hxt_img

3. Answer 'No' to the first question ("Have the image synthesis parameters been loaded and modulation patterns been calculated?")

4. Now comes the hard part. It is necessary to know the coordinates of the flare in HXA units (128 arcsec units). Please see the routine HEL2HXA to see how to convert from heliocentric coordinates to HXA coordinates. For the 15-Nov-91 flare the coordinates were (-2.1, -4.8)

5. Answer 'Yes' to the next two questions (use the default parameter files)

6. Select the channel you wish to have an image synthesized for

7. Answer 'YES' for background subtracted

8. Locate a portion of the light curve where the signal is low (just background) Click with the left button on the plot at the start time, click with the right button at the end time, click the middle button to select the portion you have marked.

9. Locate a portion of the light curve where the signal is high where you want to have the HXT image synthesized. Click with the left button on the plot at the start time, click with the right button at the end time, click the middle button to select the portion you have marked.

10. The generation of that image will begin

11. You will be asked if you want to save the results in an output file. That output HXI file can be read with YODAT or directly with RD_HXI.

12. When it is finished, the variables *index_out* and *data_out* will hold the results.

10.3.2 AUTO_HXI [*]

A driver for HXT_IMG was written which will automatically figure out how to integrate the HXT signal for each of the channels to accumulate 2000 counts. The routine is very sensitive to the location of the flare and will not converge if the location is not correct. The flare is specified by selecting a time range and this means that the data files have to be in the directories that are returned by the DATA_PATHS routine (it figures out which files to use from the input times).

It figures out the location of the flare by (1) reading the GOES event log for the flare and converting the heliocentric coordinates, (2) converting the location of the SXT partial-frame images into HXA coordinates, or (3) to pass the location of the flare in the call to AUTO_HXI. The default is to do all channels, to use the GOES event location, to not subtract background, and to use a variable integration time to achieve 2000 counts.

Some sample calls are:

```
IDL> auto_hxi, sttim, entim
IDL> auto_hxi, sttim, entim, chan=0
IDL> auto_hxi, sttim, entim, chan=0, /sxt_pfi
IDL> auto_hxi, sttim, entim, acc_cnts=[1000,2000,3000,2000]
IDL> auto_hxi, sttim, entim, loc=[-2.1,-4.8]
IDL> auto_hxi, sttim, entim, outdir='/yd8/scratch/morrison'
```

where *sttim* is the start date/time in any of the three standard formats, and *entim* is the end time.

10.4 SXT

10.4.1 SXT_DECOMP

The SXT data is normally compressed from a 12-bit word to an 8-bit word via a pseudo-square-root compression algorithm. SXT_DECOMP takes the compressed data array (*data*) and creat a integer array (*ndat*) of the decompressed numbers by typing:

```
IDL> ndat = sxt_decomp(data)
```

10.4.2 SFD_DECOMP

The SFD (SXT Full-frame Desaturated) images use a logrithmic compression in order to save the data as byte type. The decompression algorithm is 'data_out = $10.^{(data\_in/255.*6)}$'. The routine SFD_DECOMP will do this decompression for you. Beware: the output is floating numbers to it is four times larger.

IDL> data_out = sfd_decomp(data_in)

10.4.3 EXP_NORM

EXP_NORM takes a data array and produces an exposure and summation mode normalized floating point array. Note the floating array takes four times more room so don't use too big an array to start with. You need to give it an index array - make sure that the two correspond exactly or else the results will be meaningless. To run it, type:

IDL> ndat = exp_norm(data,index)

To do this it follows the following steps:

- decompresses the data array
- subtracts background
- divides by the exposure time
- divides by the number of pixels binned

10.4.4 MK_SFD

MK_SFD enables you to create a desaturated set of images from a series of long and short exposures by typing:

IDL> mk_sfd,infil,outfil,filpref

where infil is a string array of one or more SFR filenames (and locations), outfil is the string array containing the name of directory where you want the file to be created, and filpref is a string array which contains the 3-letter file prefix name (usually SFD). The desaturated image is compressed such that:

```
  0 =         1 dn/sec/5 arcsec pixel
255 = 1,000,000 dn/sec/5 arcsec pixel
```

Use the routine SFD_DECOMP to restore the data from compressed bytes to real numbers.

10.4.5 DARK_SUB

It is possible to have the dark current removed from PFI and FFI images by using the following command:

IDL> data_out = dark_sub(index, data)

It will call the routine GET_DC_IMAGE to get the proper image and take the subsection for the PFI cases.

10.4.6 GET_DC_IMAGE and GET_DC_WARM

These routines get an appropriate dark frame (*dcdata*) and its index (*dcindex*) to match the images you are working with. Currently the routines find the full-frame dark image that is closest in time and exposure level. In the future, the routine will interpolate between two reference images to simulate the input exposure duration. It can be called with the index for a single image by typing:

IDL> get_dc_image, index(3), dcindex, dcdata

where *index(3)* is the index of the image you are working with, or you can pass a list array of indicies by typing:

IDL> get_dc_image, index, dcindex, dcdata, imap

For this case, *imap* is the same length as *index* and tells you which image to use in *dcdata* for the corresponding index. For example, if *index* is 10 elements long, and *imap(8)=3*, then you should use *dcdata(*,*,3)* for the image that goes with *index(8)*.

It is possible to extract the dark frames manually using a command similar to the following example:

```
IDL> get_dc_image, xxx, dc_index, dc_data, times='1-sep-92
1:00', res=0, dpe=15
```

GET_DC_WARM works exactly like GET_DC_IMAGE but finds a suitable dark frame when the TEC is off, i.e., the detector is warm by typing:

```
IDL> get_dc_warm, index, dcindex, dcdata
```

10.4.7 INTERP_IMG

To linearly interpolate two images to a specified time type:

```
IDL> interp_img, index1, img1, index2, data2, time, index_out,
img_out
IDL> interp_img, idx(10),img(*,*,10),idx(12),img(*,*,12), time,
nidx, nimg
```

where the two input images to be interpolated are specified by the paired variables *index* and *img*. The input interpolation variable *time* can be in any Yohkoh standard time format. The returned parameters *index_out* and *img_out* refer to the interpolated *index* information and the interpolated image *data* respectively.

10.5 SXT - Filter Responses

In order to read the SXT filter responses you have to do the following:

```
IDL> filen='$DIR_SXT_SENSITIVE/et_910709.genx'
```

IDL> restgen,file=filen,db,text=text,header=header

where *filen* is the file name of the data file containing the SXT filter responses, RESTGEN is a generic file reading program. and the response data is put into the *db* variable array which has the dimension (7,26) where

0	log T(e) array (5.5-8.0)
1	Open/Open or Noback case
2	Al 1400 A
3	Dagwood sandwich
4	Be
5	AL 12m
6	Magnesium

10.6 SXT - Temperature and Emission Measure Determination

10.6.1 SXT_TE

This routine enables you to derive a temperature map (te) from two filters taken of the same region by typing:

IDL> te = sxt_te(index1, img1, index2, img2, em=em, /getem)

where *img1* and *img2* are the two images in the two filters, *index1* and *index2* are their respective index structures, and the */getem* switch (optional) provided an emission measure map (*em*).

10.6.2 LWA_TE [*]

LWA_TE used SXT_TE as a starting point, but has fallen behind in some of the corrections made to SXT_TE. This routine will be removed shortly. The parameters are the same as SXT_TE, but the order is different.

IDL> te = lwa_te(img1, img2, index1, index2, em=em, /getem)

10.6.3 HARAT [*]

HARAT will derive the temperature (*temp*) and emission measure (*em*) arrays for a 64 x 64 filter image pair (*img1, img2*) by typing:

IDL> harat, img1, img2, f1, f2, thold, temp, em

where *f1* and *f2* are the numbers of the filters used, *thold* is the threshold for filter 2. Note that there must be nomalized images.

10.6.4 T6_MAIN [*]

This routine was written by Jim McTiernan at University of Berkeley. The capabilities and options found in this routine are being incorporated into SXT_TE, so this routine will probably not be on-line much longer.

10.7 PLOT_SOT

PLOT_SOT will plot out the aspect sensor degradation as a function of time by typing:

IDL> .run plot_sot

It will produce plots for both the narrow-band and wide-band filters.

10.8 PLOT_TEMPS2 [*]

This routine plots the SXT instrument temperatures. This program can only be run after the variable *index* exists which came from an SXT file. A common practice is to run PLOT_SOT which gets images over the whole mission.

IDL> .run plot_temps2

A mosaic of time histories of the various instrument temperatures will be plotted.

11 Spacecraft Attitude and Solar Ephemeris Routines

11.1 HXA_SUNCENTER [*]

Calculate the suncenter position (in SXT pixel coordinates) from the HXA info in the PNT files. Tries to reconstruct hidden limbs.

```
IDL> sunc = hxa_suncenter(pnt)
IDL> sunc = hxa_suncenter(index=index)
```

If the input *pnt* or *index* have N elements, then the result will be a floating array of 4xN elements. *sunc(0,*)* is the SXT column number in 'IDL coordinates', *sunc(1,*)* is the SXT line number, *sunc(2,*)* is the milliseconds of day for the input time, and *sunc(3,*)* is the days since 1-Jan-79 number for the input time.

11.2 HEL2PIX [*]

This routine takes heliocentric coordinates on the sun and converts to SXT pixel coordinates (with 2.46 arcsec units). This routine does not take changes in S/C pointing into consideration. North is positive and west is negative (NOTE: this will be changed shortly to west being positive) For a case where PR_GEV returns S13W19, it should be entered

```
IDL> xy = hel2pix(-13, -19)
IDL> xy = hel2pix(nn, ee)
```

and the results *xy(0)* is the SXT column number (E/W) in 'IDL coordinates', and *xy(1)* is the SXT line number (N/S).

11.3 HEL2HXA [*]

This routine takes heliocentric coordinates on the sun and converts to HXA coordinates (which have 128(?) arcsec units). This routine does not take changes in S/C pointing into consideration. North is positive and west is

negative (NOTE: this will be changed shortly to west being positive) For a case where PR_GEV returns S13W19, it should be entered

```
IDL> xy = hel2hxa(-13, -19)
IDL> xy = hel2hxa(nn, ee)
```

and the results *xy(0)* is the E/W HXA address, and *xy(1)* is the N/S HXA address.

11.4 PIX2HEL

Under development.

11.5 GET_RB0P

For a given date and time, return the solar radius, b0 angle, and p angle for that time. This routine used to be called PB0R, but GET_RB0P can handle any of the three standard time formats.

```
IDL> rb0p = get_rb0p('15-nov-91')
IDL> rb0p = get_rb0p(index)
```

If N input times are passed in, then the output is 3xN where *rb0p(0)* is (R) solar radius in arcseconds measured outside earth's atmosphere, *rb0p(1)* is (B0) heliographic latitude of the central point of the solar disk, and *rb0p(3)* is (P) position angle of the northern extremity of the axis of the sun's rotation, measured eastward from the geographic north point of the solar disk,

11.6 RD_PNT

This routine allows a user to read the S/C pointing information. The PNT files have the IRU, HXA, TFSS, and ADS information for every major frame. A sample call would be:

```
IDL> rd_pnt, sttim, entim, pnt
```

IDL> rd_pnt, '1-nov-91', '3-nov-91', pnt

11.7 GET_PNT

For a given set of input times, this routine will read the proper PNT records (if 100 input times are passed in, then 100 PNT records are returned). This routine is useful when trying to access PNT data that covers a large range of times. A sample call would be:

IDL> get_pnt, index, pnt

11.8 GT_HXA

This routine will extract the HXA data from a PNT structure or from an ADA data structure. Some sample calls are:

IDL> hxa = gt_hxa(pnt)
IDL> sxtcen = gt_hxa(pnt_data, /sxtpix)
IDL> sxtcen = gt_hxa(pnt_data, /sxtpix, /x)
IDL> hxacen = gt_hxa(pnt_data, /hxacen)
IDL> hxacen = gt_hxa(pnt_data, /hxacen, /y)
IDL> x1 = gt_hxa(pnt_data, 0)
IDL> x2 = gt_hxa(pnt_data, 1)
IDL> y1 = gt_hxa(ada_data, 2)

The *hxa* result would be 4xN where there are four addresses for HXA limbs. The */x* or */y* switches will result in a single address being returned, and the */sxtpix* will return the results in SXT pixel coordinates.

11.9 GT_IRU

It is possible to extract the IRU information from the PNT or ADA structures using this routine. The output is in arcseconds. It is also possible to have the drift in the IRU removed for short time periods by using the /RESID switch (it fits a line to the drift and subtracts that drift). Sample calls are:

```
IDL> iru = gt_iru(pnt)
IDL> iru = gt_iru(pnt, /resid)
IDL> iru = gt_iru(ada)
IDL> iru = gt_iru(ada, index, /resid)
```

12 Alignment Routines

The alignment of SXT X-ray and white-light images can be done directly provided that you take out the effects of spacecraft pointing shifts between images. This is important for:

- desaturation of images (combining images to remove saturation)
- summation or differencing of images
- making temperature or EM diagnostics
- comparing with HXT or ground-based observations

Generally you align the X-ray images to the nearest white-light image then use the white-light to compare to other data. BUT remember there is a systematic offset between them, it is:

- x-axis (EW): the X-ray image is WEST of the optical image by $0.36 \pm .177$ pixels
- y-axis (NS): the X-ray image is SOUTH of the optical image by $1.47 \pm .283$ pixels

where the pixels are full resolution SXT pixels (2.46 arcsec) and the parenthetical numbers represent uncertainties on those quantities.

12.1 Finding the Center of the Image

12.1.1 DSK_LOCB [*]

DISK_LOCB a simple locator created by Keith Strong and Alan McAllister. Currently works on white light, Hα, and magnetogram images. Anything with a clearly defined edge. The images are all treated using the basic histogram technique. (There is an old routine called DSK_LOC that uses a using a derivative technique for white light etc., as they have a high and bouncy background). DSK_LOC will find the center coordinates, disk radius of an optical solar image by typing:

IDL> dsk_loc, image, horzcnt=x0, vertcnt=y0, horzrad=rx, vertrad=ry

where the keywords return the values of the radius and disk center.

A complete list of the optional keywords follows:

`horzcnt`	horizontal center, floating point
`vertcnt`	vertical center, floating point
`horzrad`	horizontal diameter, floating point
`vertrad`	vertical diameter, floating point
`eastlimb`	east limb, in pixels
`westlimb`	west limb, in pixels
`npole`	north pole, in pixels
`spole`	south pole, in pixels

12.1.2 GEN_LOCB [*]

A general limb finder. [Use doc_library on it]

12.1.3 FIND_LIMB

FIND_LIMB also finds the location of the center of the sun and it's radius. A sample calling sequence is:

IDL> find_limb, img, x0, y0, r0

where *img* contains the input image and *x0,y0,r0* receive the output center location and diameter of the disk.

12.1.4 OCENTER

12.2 Co-Alignment Routines

12.2.1 RD_AR [*]

RD_AR takes a file name (or list of file names - infil) and a list of subscripts (ss) like that produced by SSWHERE and produces a data and index array that is corrected for:

- SXT PFI repoints
- S/C Jitter
- FOV changes
- pixel resolution changes

To run it you type:

IDL> rd_ar, infil, ss, index, data, missing=0

where *infil* and *ss* are the input, and *index* and *data* are the output. The value passed by keyword *missing* is the DN value that should be inserted into the array where there is no data (since the output array could be something like 78x83xN when the input array is 64x64xN)

12.2.2 SXT_CENTER [*]

SXT_CENTER calculates the row (x) and column (y) of the center of the solar disk and the radius (r) of the Sun given a data cube (*data*) and the index structure (*index*) by entering:

IDL> sxt_center,data,index,x,y,r

There are many options that are possible to use, please check them with DOC_LIBRARY.

12.2.3 ALIGN_CUBE (SXT) [*]

ALIGN_CUBE is a routine to co-align full-frame SXT images. The calling sequence is:

IDL> outcube=align_cube(incube, index, i)

where *incube* is the input data, *index* is the respective index entries, and i indicates which frame to use for the alignment. N.B. Be careful to ensure that the image size matches the index entry otherwise image co-alignement won't work properly.

12.2.4 GBO_SCALE2 [*]

GBO_SCALE2 can be used for aligning SXT images with each other or with other images. This routine takes two images, resizes them and lays them on top of one another by typing:

```
IDL> outimg=GBO_SCALE2(trgimg, gboimg)
IDL> outimg=GBO_SCALE2(sxtimg, gboimg, index1, sxt=1)
IDL> outimg=GBO_SCALE2(sxtimg1, sxtimg2, index1, index2,
sxt=2, /mktv)
```

which will scale the input image *gboimg* and center it to match a target image *trgimg*. If either of these is an SXT image then it needs to have its index passed as well, and the SXT keyword should be set to 1 or 2 appropriately, (i.e., for one or two SXT images).

This routine corrects for different vertical and horizontal diameters. The MKTV keyword allows you to plot the result directly. This should work with general sized images.

12.2.5 COMPST [*]

There is a routine called COMPST that interleaves two images and scales them to each use half the color table. It does no alignment and requires that you have or develop appropriate color tables. The calling sequence is

```
IDL> imageout= compst(image1, image2)
```

It is currently set up to work with 512x512 images. It is recommended that the images are massaged so that they emphasize the intensity ranges of interest, e.g., for low-end stuff a decompressed image gives poor results. Creative color tables may also be useful.

Check the rescaling and positioning of the images in each half of the color table.

13 Directly Reading Yohkoh Reformatted Data

13.1 RD_BDA, RD_HDA, RD_SDA, RD_WDA, RD_XDA

The RD_xxx routines allow you to read the reformatted data files. Each routine has the same calling sequence, and there is a generic reading routine called RD_XDA which will read any of the instruments. After you have established the input file name(s) (*infil*), and which data sets to extract (*dset_arr*), you can read the data by a command like:

IDL> rd_sda, infil, dset_arr, index, data
IDL> rd_sda, infil, dset_arr, index, data, roadmap

index is a structure which describes the data. There is one index structure for each dataset. The roadmap is for all datasets in the files (not just the selected datasets).

13.2 RD_ROADMAP

RD_ROADMAP will read the roadmap from a file (or list of files) by typing:

IDL> rd_roadmap,infil,roadmap,ndset

where *infil* is a string array of file names. The roadmap variable is the same as you get from YODAT

13.3 RD_QS

13.4 Concatenation of Datasets

13.4.1 STR_CONCAT

It is possible to concatenate two structures of same or similar type by using STR_CONCAT. For example, if you wanted to concatenate the variable *index1* and *index2*, the command would be:

IDL> index_out = str_concat(index1, index2)

13.4.2 Concatenating Data Arrays

A sample command for concatenating two 3-D arrays is:

IDL> out = [[[in1]],[[in2]]]

If *in1* was 100x200x3 and *in2* was 100x200x3, then *out* would be 100x200x7. The first two dimensions of *in1* and *in2* must be identical. It is possible to have *in2* be 100x200, in which case *out* would be 100x200x4.

A sample command for concatenating two 2-D arrays is:

IDL> out = [[in1],[in2]]

If *in1* was 100x10 and *in2* was 100x20, then *out* would be 100x30. In this case, the first dimension of *in1* and *in2* must match.

14 Saving Yohkoh Data

14.1 SAVEGEN

SAVEGEN is a routine available to save variables of any type (including structures). Some sample calls are shown below. If no file name is specified, it will use 'save.genx'.

```
IDL> savegen, var1, var2
IDL> savegen, var1, var2, var3, var4, var5, file='flare27oct92'
```

Use RESTGEN to restore the data. NOTE: It is recommended to use SAV_SDA and SAV_BDA whenever possible for standard SXT and BCS data sets.

14.2 SAV_SDA

SDA_SDA will store an index and data array in an SDA format file. The data array can be byte, integer*2, integer*4 or real*4. The dimensions of the image does not have to be the original dimensions. A sample is:

```
IDL> sav_sda, outfil, index, data
IDL> sav_sda, outfil, index, data, qs
```

where *outfil* is the name of the file you wish to store the data in. If you wish to append to an existing file, then you can use the /APPEND switch. For example:

```
IDL> sav_sda, outfil, index, data, /append
```

Use RD_SDA to restore the data.

14.3 SAV_BDA

SAV_BDA works in the same manner as SAV_SDA. A sample call would be:

IDL> sav_bda, outfil, index, data, qs, dp_sync

14.4 SAV_HXI

SAV_HXI allows you to save HXT synthesized images. A sample call would be:

IDL> sav_hxi, outfil, index, data

14.5 SAVE [IDL]

SAVE is an IDL facility that will save variables, etc. into a file of your choice. To save a variables *a*, *b*, and *c* to the default file 'idlsave.dat', type:

IDL> save, a, b, c

To save to the file 'junk.save', type:

IDL> save, a, b, c, filename='junk.save'

To save all variables, type:

IDL> save, /all, filename='junk.save'

CAUTION: When restoring the data, the variable names are restored exactly as they were saved, so if you saved *a, b and c*, then when you restore that save set, any existing variables *a, b and c* are deleted and when the restore occurs. A sample command to restore data is:

IDL> restore, 'junk.save

14.6 TIFF_WRITE [IDL-LIB]

To write some image data (*image*) to a file (*outfile*) in TIFF format all you have to do is type the following for grey scale:

IDL> tiff_write, outfil, image

For 256 element (pseudo) color, use:

IDL> tiff_write, outfil, image, red=r, green=g, blue=b

where the optional parameters red, green and blue are used to save the colour table characterized by the byte arrays (r, g, b - see TVLCT for details about obtaining them).

To make full (24 bit) color TIFF format files see the documentation for tiff_write (i.e. use the Doc_library utility on tiff_write).

14.7 MK_TIFFB

To make full (24 bit) color TIFF format files to be read on Macintosh computers use:

IDL> mk_tiffb, outfile, image, r, g, b

where the required color table is specified by 256 element byte arrays *r, g, b*. The output TIFF file contains the specified image which has been flipped vertically for direct display on (most) tiff-readers on the Macintosh and has a resolution of 100 pixels per inch. The default output file size is 774 Kbytes which results from rebining into image to 508x508 and fits on the standard double sided Mac floppies (779 Kbytes capacity). For more options (such as changing image size which is 508x508 by default, noflip, etc.) or information on MK_TIFFB use the Doc_library utility.

14.8 WRT_FITS [*]

WRT_FITS creates a FITS file from an image (*data*) by typing:

IDL> wrt_fits, outfil, header, data(*,*,n)

where *outfil* is the string containing the filename. *header* is optional input which is the FITS string array ASCII header. If it is undefined, then

WRT_FITS will build the minimal header. The routine RFITS should be used to read it.

14.9 SXT2FITS

It is possible to take an SXT data file or an index and data that has already been read in, and write a single FITS file for each image. The FITS header has all of the information on the date and time, the filters used, the exposure duration, the resolution, and the DP mode and rate. The following command will create a file for each image in the *data* array:

```
IDL> sxt2fits, index, data
```

The default file names are 'SF_FITSyymmdd.hhmmss' for FFI images and 'SP_FITSyymmdd.hhmmss' for PFI images. If a single output file name was passed to the SXT2FITS routine, but there are several images to save, then it will append an image number to the end of the file name. In the following example, the input array is 512x512x3, so it will create the files 'flare.0001', 'flare.0002' and 'flare.0003'.

```
IDL> sxt2fits, index, data, outfil='flare'
```

It is possible to specify an input file name in which case all images in that file will have FITS files created. It is also possible to specify a list of the images to be saved. Some examples of these calls are:

```
IDL> sxt2fits, index, data, ss=ss
IDL> sxt2fits, infil=infil
IDL> sxt2fits, infil=infil, ss=ss
```

15 Accessing Yohkoh Database

15.1 SXT Tables

15.1.1 GTAB_COMM, GTAB_ENTRY, GTAB_ROI

It is possible to get information on what was used in a SXT table by using one of these routines. GTAB_COMM returns information on the common table, GTAB_ENTRY returns information on the entry tables, and GTAB_ROI returns information on the ROI location and image shape tables. Some sample calls:

```
IDL> print, gtab_comm(index)
IDL> print, gtab_comm('15-nov-91')
```

15.1.2 GTAB_PFI and GTAB_FFI

These routines allow a user to print information on the partial-frame and full-frame observing sequence being used. There are four possible sequences for PFI and four more for FFI. Which sequence is used is determined by the DP mode and telemetry rate. If the SXT index for an image is passed, then it will figure out which of the four was running at that time.

```
IDL> print, gtab_ffi(index)
IDL> print, gtab_pfi('15-nov-91', 2)
```

If a time is passed, it will default to show sequence number 0. It is possible to speicify which sequence table by passing it as a second parameter.

15.2 Printing out Summaries

15.2.1 CONTACTS

This routine will give information on when the ground station contacts are for a given day. The output from the commmand:

```
IDL> contacts,'2-jun-92'
```

would be

```
                Kagoshima Space Center Contacts                  Minutes of
                     Starts                          Ends        Day  Ngt  Tot
          JST                (UT)                    JST
2-JUN-92 02:56:14 ( 1-JUN-92 17:56:14)   03:07:14    3.5  7.5 11.0
2-JUN-92 04:39:59 ( 1-JUN-92 19:39:59)   04:48:59    7.8  1.2  9.0

2-JUN-92 20:27:14 ( 2-JUN-92 11:27:14)   20:37:14    4.0  6.0 10.0
2-JUN-92 22:09:14 ( 2-JUN-92 13:09:14)   22:21:14    0.0 12.0 12.0
2-JUN-92 23:52:29 ( 2-JUN-92 14:52:29)   00:04:14    0.0 11.8 11.8
```

A blank line is inserted to show that the second set of contacts is for a different series of contacts (they come in clusters of five or six per day). It is possible to get the station contacts times for DSN stations by using the */CANBERRA*, */GOLDSTONE* or */MADRID* switches. It is possible to specify an end time and output files in the following example:

```
IDL> contacts, '1-jun-92', '10-jun-92', outfil='contacts.txt'
```

15.2.2 PR_FEM

PR_FEM will print out Yohkoh's day and night events by typing:

```
IDL> pr_fem, '1-jan'
```

15.2.3 PR_EVN

PR_EVN helps you find when there is Yohkoh data available. An event is driven by a change between QUIET and FLARE mode, or when there is a data gap of more than 60 seconds. By typing:

```
IDL> pr_evn, '23-jun-92'
```

a list of the times that Yohkoh data is available and the number of datasets available for each instrument is listed for 24 hours starting at 23-jun-92 00:00.

By typing:

IDL> pr_evn, '15-nov-91 20:00', '17-nov-91 15:00', /flare

all times that Yohkoh was in FLARE mode between those times is listed. By typing:

IDL> pr_evn, '1-jan-92', '1-jan-93', /flare, /counts, outfil='pr_evn.results'

the FLARE modes for 1992, and since the /COUNTS option was used, the maximum counting rate for certain WBS, HXT, and BCS channels is printed instead of the number of datasets available. It also prints the GOES classification when it is available.

15.2.4 PR_GEV

The GOES event log files are available with the Yohkoh database. To get a printout of the x-ray events for a given day, use something like:

IDL> pr_gev,'2-nov-92

The following would be displayed:

```
PR_GEV.PRO Run on:  6-Nov-1992 14:41:55.00

   Date       Time (UT)       X-Ray Opt Loca- NOAA      Peak Radio     Reports
           Begin  Max   End   Class Imp tion  Region

 2-NOV-92  00:40 00:57 01:09  C1.4
 2-NOV-92  02:31 03:08 03:28  X9.0                 5000 17000-22536 II,III,IV
 2-NOV-92  07:48 07:48 08:04       SF N15W19 7324
 2-NOV-92  09:09 09:10 09:31       SF N16W21 7324
 2-NOV-92  09:40 09:57 10:02       SF S16W46 7323
 2-NOV-92  12:26 12:27 13:25       SF S16W49 7323
 2-NOV-92  12:26 13:22 13:39       SF S06E61 7330                    III
 2-NOV-92  17:30 17:39 17:48       SF S08E60 7330
```

15.3 Plotting Summaries

15.3.1 PLOTY

Please see the section described in the chapter on 'Time Plotting Routines'.

15.3.2 PLOT_GOES

This routine will plot the GOES one minute light curve data. The following are a few examples:

```
IDL> plot_goes, '2-nov-92'
IDL> plot_goes, '2-nov-92', '4-nov-92'
```

The default is to plot 24 hours of data.

15.3.3 PLOT_EVN

This routine will plot a very basic time line showing when the station contacts are available, when the SAA passages are, when the S/C day and nights are, and the periods when there is Yohkoh data available. It shows when there is FLARE and QUIET data.

15.4 Reading and Using the Database

15.4.1 RD_FEM, RD_EVN, RD_GEV, RD_GXT and RD_NAR

To read the FEM, EVN ,GEV or NAR structures, use a command similar to:

```
IDL> rd_fem, sttim, entim, fem
IDL> rd_evn, index(0), index(n-1), evn
IDL> rd_nar, '1-nov-91', '2-nov-91', evn
```

Look at the File Control Document for a description of the strutures which are returned.

15.4.2 RD_OBS

A sample standard calling sequence for reading the observing log data files is:

```
IDL> rd_obs, '8-may-92', '8-may-92 12:00', bcs, sxtf, sxtp, w_h,
fid
```

It is possible to only read SXT full-frame data by using the one of the following command:

```
IDL> rd_obs, '8-may-92', '8-may-92 12:00', bcs, sxtf, sxtp, w_h,
fid, /sxtf
IDL> rd_obs, '8-may-92', '8-may-92 12:00', bcs, sxtf, /sxtf
```

16 Accessing Ground Based FITS Images

All Ground Based images that are stored in the Yohkoh database use the FITS file format.

16.1 Directory and File Organization

The ground based data is stored in the following directories:

$DIR_GBO_BBSO	holds the Big Bear Data
$DIR_GBO_KP	holds the Kitt Peak Data
$DIR_GBO_SELSISI	holds data from many observatories. These images are copied daily from the SELSIS VAX.

The Yohkoh file naming convention for ground based data is very similar to that used by Yohkoh data. There is a 3 letter prefix followed by the normal file ID portion (YYMMDD.HHMM) (see the section defining the Yohkoh FileID). The first letter of the three letter prefix is always 'g' to signify ground based data. The second letter is the institute where the data was taken. The letters assigned to the institutes are:

b	Big Bear Observatory
C	Boulder, Colorado
g	GSFC
h	Holloman
k	Kitt Peak
l	Leamonth, Australia

The last letter is the type of data that is in the file. The letters assigned to the types of images are:

h	H-alpha
i	HeI 10830
m	Magnetogram
s	Spectrohelioscope
w	White light
x	X-Ray

A sample file name with the directory path would be $DIR_GBO_KP/gkm920907.1442 for a Unix machine.

16.2 RFITS

RFITS reads a standard format FITS file by typing:

```
IDL> img = rfits(infil, header=header)
```

where *img* is the image, *infil* is the input file name, and header is the string array with the data header in it. It is possible to use the /SCALE option to convert the data into some physical units. For example, the magnetogram data would be converted to gauss.

```
IDL> img = rfits('$DIR_GBO_KP/gkm920907.1442', head=head,
/scale)
```

17 Time Routines

17.1 Time Conventions

There are three time conventions used in the Yohkoh database.

17.1.1 Internal Representation

The internal representation uses two variables to define the date and time of an event. One variable (integer*2) is the days since 1-Jan-79 (DS79) and the other (integer*4) is the milliseconds of the event within that day (MSOD). Almost all data is saved using this internal representation which uses a total of 6 bytes.

17.1.2 External Representation

The external representation uses a 7 element integer array to define the time of an event. This representation uses 14 bytes.

```
(0) = hour
(1) = minute
(2) = second
(3) = millisec
(4) = date
(5) = month
(6) = year
```

17.1.3 String Representation

The time of an event can be represented with a string. The general rules are:

- The month uses the first three letters of the month (Jan, Feb, Mar...)
- The date and year are separated from the month using '-'
- The year can come before or after the month (7-Jan-92 or 92-Jan-7)
- The hour, minutes and seconds is separated by ':'

- The milliseconds is entered as a fraction of seconds. For example, '16-Dec-91 02:29:40.819'
- There is a space between the date and the time

17.2 FMT_TIM

The routine FMT_TIM will take a structure and return a string with the date and time. For example, the command:

```
IDL> a = fmt_tim(roadmap(0))
```

will return put the string '16-DEC-91 02:29:40'. It is possible to use the /MSEC keyword and get '16-DEC-91 02:29:40.819' instead.

17.3 ANYTIM2INTS

This routine takes any of the three representations and returns a structure with the internal representation. The structure has a .TIME and a .DAY field. An example is:

```
IDL> a = anytim2ints('23-jun-92 3:00')
```

This routine has an optional keyword input to allow the user to offset the input time by some number of seconds. If the input is an array of times, then the offset can be a scalar value or it can be an array of offsets which is the same length as the input. For example,

```
IDL> a = anytim2ints(roadmap, off=60)
IDL> a = anytim2ints(roadmap, off=off)
```

17.4 ANYTIM2EX

This routine takes any of the three representations and returns an array with the external representation. If the input is only one time, then the output is a 7 element array. If it is an array of 'N' elements, then the output is a 7xN array.

IDL> b = anytim2ex(roadmap)

17.5 GT_DAY

GT_DAY is a function routine that returns the date from the index or roadmap structure in a string variable other options available) by typing:

IDL> date = gt_day(index,/string)

17.6 GT_TIME

GT_TIME is a function routine that returns the UT time from the index or roadmap structure in a string variable other options available) by typing:

IDL> time = gt_time(index,/string)

17.7 INT2SECARR

INT2SECARR takes a structure (e.g., roadmap or index) containing time and will create an array of times (t) in seconds from some given reference. If the second parameter is not passed, then the first time in the input array is used

IDL> t = int2secarr(roadmap)

This will produce an array of times with respect to the first time.

IDL> t = int2secarr(roadmap, ref_time)
IDL> t = int2secarr(roadmap, '1-sep-91')

17.8 ANYTIM2DOY

This routine takes any of the three representations and returns the day of year number.

IDL> b = anytim2doy(roadmap)

17.9 EX2DOW

This routine will take the external representation and return the day of the week number, 0 for sunday, 1 for monday, ...

IDL> dow = ex2dow(tarr)

17.10 EX2FID

EX2FID will take the external representation data and return a string with the fileID.

IDL> fid = ex2fid(input)

17.11 TIM2ORBIT

This routine will take a set of input times and will determine which orbit and which week the data falls in. It can also return the number of minutes into the orbit for an input time. A sample interactive use of the routine would be:

IDL> tim2orbit,'23-jun-92 6:00',/print

The item returned for this would be:

```
     Input                 S/C Day      Night in   FileID     WeekID  D/N    SAA?
   Date      Time          Time  Min Ago  (min)
23-JUN-92  06:00:00    05:13:14 46.77    14.98  920623.0513  92_26  Day
```

The S/C day started 46.77 minutes ago, and night will occur in 14.98 minutes (it is S/C day mode). The FileID for this time is 920623.0513 and the WeekID is 92_26.

Optional Keyword Output Parameters (the number of output elements is the same as the input):

FEM	the FEM structure
FID	the FileID for each input time
WID	the WeekID for each input time
SAA	a boolean array, TRUE if in SAA
SCDAY	a boolean array, TRUE if in S/C Day
TIM2FMS	the number of minutes since sunrise
TIM2NIGHT	the number of minutes before sunset

17.12 SEL_TIMRANGE

Given an array of times, this routine will return the subscripts of the elements which fall between a set of input times.

IDL> ss = sel_timrange(roadmap, st_tim, en_tim)
IDL> ss = sel_timrange(orbit_arr, '1-jul-92', '22-jul-92')

18 Windows and Output Devices

18.1 WDEF

WDEF creates a window of a set size. The first example will make a 512x512 window in the top right hand corner of your screen. The second example makes a 128x128 window, and the last example makes a 900x512.

```
IDL> wdef,0
IDL> wdef,0,128
IDL> wdef,0,900,512
```

Window 1 is in the lower right, 2 is in the upper left, and 3 is in the lower left (even windows on top, window modulo 4 equaling 0 or 1 is on the right)

18.2 WSHOW [IDL]

WSHOW pushes a covered IDL image window to the forground while keeping the text window active. Just type

```
IDL> wshow
```

18.3 HARDCOPY

Dumps the contents of a 'X' screen to a laser printer with either b/w or colour options by simply typing:

```
IDL> hardcopy
IDL> hardcopy, /black
```

It is recommended to only use this routine to get hardcopies for images. If you want to get a hardcopy of a line plot, then first change the output device to a postscript file, re-run the plotting program or plotting commands, and then issue the print command. An example would be:

```
IDL> set_plot, 'ps
```

```
IDL> .run plot_sot
IDL> pprint
IDL> set_plot, 'x
```

18.4 LPRINT/PPRINT

LPRINT outputs a hardcopy of an idl.ps file. An example of how to use it would be

```
IDL> set_plot,'ps
IDL> device, /land ;optionally use landscape orientation
IDL> .... (your plot commands)
IDL> lprint
IDL> set_plot,'x
```

This will output the plot on the laser printer.

19 Low Level Routines

19.1 Plotting Routines

19.1.1 PLOT_LCUR

PLOT_LCUR is a low level generic plotting routine used by several other routines. The routine returns the list of subscripts which are between the times selected by the user. The user clicks on the plot with the left button to define a new start time, the right button to define an end time, and the middle button to exit. The user can expand a time range after marking the start/stop times by clicking with the left button on the box in the lower right corner. Some of the more simple calls are:

```
IDL> ss = plot_lcur(roadmap, gt_sxs1(roadmap))
IDL> ss = plot_lcur(index, gt_sum_l(index), plotr=a, title=title)
IDL> ss = plot_lcur(index, gt_sum_l(index), /nohard)
```

19.1.2 CLEARPLOT

This routine will reset the IDL plotting variables.

19.1.3 CLEAR_UTPLOT

This routine will reset the UTPLOT plotting variables.

19.2 File and Directory Routines

19.2.1 CONCAT_DIR

This routine is fundamental in allowing for software to be transported between Unix and VMS systems. Given a directory logical name (or environment variable) it will figure out whether to put a '/' between the directory and the file name (for Unix systems), or to use a ':' (for VMS systems). A restriction is that the directory must be a logical or environment variable. Some sample calls are:

```
IDL> outfil = concat_dir(dir, filnam)
```

IDL> outfil = concat_dir('$DIR_GEN_OBS', 'obs92_23a.01')

19.2.2 FILE_LIST

FILE_LIST enables you to get a string array *infil* with all the files that fulfill a set of specificatios by typing:

IDL> infil = file_list(dir,filen)

where *dir* is a string array of directory name(s) and *filen* is a string with the file disgnation (wild cards are ok - eg sfd* or bda920101.*)

19.2.3 FILE_MENU

FILE_MENU works very much like FILE_LIST but is menu driven. It only gets you a single file (infil) by typing:

IDL> infil = file_menu(dir,filen)

19.2.4 DATA_PATHS

This routine is fundamental to several routines including YODAT. The routine should return a string array with all of the directories where there is Yohkoh reformatted data. The routine is site specific since each institute organizes the data slightly differently. A common practice is to use DATA_PATHS2 to get the standard data directories, and the append the specialized data directories. See the sample program in the 'site' branch of the Yohkoh software.

19.2.5 DATA_PATHS2

This routine searches all '/yd' directories for directories with YY_WW names (for example, '/yd1/92_36a' or '/yd10/91_35a'). Since the generation of the data directory list is done automatically, it is not necessary for a programmer to edit DATA_PATHS every time a new week of data is placed on-line.

19.2.6 GBO_PATHS

GBO_PATHS is similar to DATA_PATHS, but it looks for the environment variables beginning with 'DIR_GBO' and creates a list. This routine is used by YODAT to find where the ground based FITS files are located.

19.2.7 GET_SUBDIRS

This routine will return all of the subdirectories under a given directory. An example:

```
IDL> dirs = get_subdirs('/ys')
```

19.2.8 PR_PATH

This routine will display all of the directories that are in the IDL variable !path.

19.2.9 PATH_LIB

PATH_LIB will search directories in the IDL !path variable in the order of the list to find a given IDL program. The default is to only give the first instance that the file is found, but it is possible to get all locations by using the */multi* switch. A wild card ('*') effectively sets the */multi* switch). Same examples are:

```
IDL> path = path_lib()
IDL> path = path_lib('add*')
IDL> path = path_lib('yodat', /multi)
```

19.2.10 FILES_SEARCH

This routine will search all files in sub-directories from a given directory for a given string. In the following example, all .PRO files under the /ys tree are searched to find the string 'HEL2PIX'.

```
IDL> files_search, '/ys', 'hel2pix'
```

19.3 Miscellaneous

19.3.1 ADD_PRO

This routine will allow a user to copy an IDL routine onto the YS software tree. The routine will check that the routine has a minimal documentation header. ADD_PRO is not available at this time but will be shortly.

19.3.2 STR2ARR

Given a string, this routine will separate the items into an array using a ',' as the delimiter by default. Some examples are:

```
IDL> arr = str2arr('a,b,c,d,e,f,ggg,hh')
IDL> arr = str2arr(!path, delim=':')
```

19.3.3 ARR2STR

Given an array, convert it to a string. The items will be separated by a comma by default.

```
IDL> str = arr2str(indgen(10)
```

19.3.4 REBIN [IDL]

REBIN is an IDL routine that enables you to make a 2 dimension array larger or smaller (it also works on multidimensional arrays). To use it to create a new 512X512 image (ndat) from a 64x64 image (data) type:

```
IDL> ndat=rebin(data,512,512,/sample)
```

Note that the new x and y dimensions must be an integral multiple (or divisor) of the original array dimensions.

19.3.5 PLOT_TRAV

It is possible to plot a time line showing when the SXT and BCS team members will be on travel to Japan by using PLOT_TRAV. The plot separates

the people by institution. Some examples are:

```
IDL> plot_trav
```

```
IDL> plot_trav, /lparl
```

```
IDL> plot_trav, /bcs
```

The default is to plot SXT personnel. To show BCS personnel use the */bcs* switch. The */lparl* switch will only plot the key Lockheed SXT personnel.

20 Getting Help on Software and Data

20.1 DOC_LIBRARY [IDL-LIB]

IDL has a facility called DOC_LIBRARY which you can use to get the details of any procedure that you know the name of by typing:

```
IDL> doc_library,'yodat'
```

This will obtain a listing of the header information of the program YODAT (any comments put in at the beginning of the code between ';+' and ';-') provided it is in your path. Wild cards are allowable, e.g.,

```
IDL> doc_library,'test*'
```

DOC_LIBRARY is a script driven program (i.e., it will work on ordinary terminals as well as x-terminals or workstations. Do not include the '.PRO' in the call to DOC_LIBRARY.

20.2 XDL [IDL-LIB]

This is a widget driven version of DOC_LIBRARY and will present you with a complete list of all the IDL software in your path. You can search using the mouse and slider on the screen. It is run by typing:

```
IDL> xdl
```

20.3 PRO_LIST

To return a list of the IDL programs, functions, and procedures (.PRO files) that are in the IDL path (!path).

```
IDL> pro_list
IDL> pro_list, outfil='pro_list.txt'
```

20.4 CATALOG_LIST

This routine is under development and will be released shortly.

20.5 HELP [IDL]

The IDL command HELP is available to get information on all of the variables, procedure and functions currently in IDL's memory. By typing:

```
IDL> help
IDL> help, data
```

you get a list of all of the variables that are defined within the current IDL session. If you want information the definition of a structure, you can use the /STRUCTURE option. Some examples are:

```
IDL> help, /st, roadmap
IDL> help, /st, index.gen
```

It is possible to get information on the calling parameters of procedures and functions that have already been compiled by typing:

```
IDL> help, /routine
```

The procedure and function name is listed first. The positional parameters are listed in lower case and the keyword parameters are in upper case.

21 Yohkoh Distribution and Archive Tapes

Please see Volume 2 of the Yohkoh User Guide for more information on the archive tapes.

21.1 GO_RDTAP

There is one archive tape for each week of data. The first step to accessing archive data is to determine which tape it is on. Use the WeekID table shown earlier in this document to figure out what tape you need. For example, if you wish to look at data on 27-Mar-92 you first determine that it is week 13 in 1992, so the weekID is 92_13. Put that tape into the exabyte tape drive.

You need to move to the directory where you want the data to be written.

GO_RDTAP is a menu driven program to retrieve data from the exabyte archive tapes via a menu driven program. The ISAS and LPARL machines have a copy of the directory listing for each tape saved on the disk. This means that you can simply type:

```
IDL> go_rdtap
```

and it will list all of the tapes that have been made. You need to select the tape that you are reading. If your system does not have the XBD files on-line, then type:

```
IDL> go_rdtap, /tape
```

after you have loaded the tape, and the program will read the directory listing from the tape.

A menu will appear a series of weekly log files. If you do not need any of the weekly files, then just select QUIT/EXIT. The next menu will be the file prefixes that are available. Click on each file prefix that you wish to read down to the disk. When you have selected all of the prefixes you need, click on QUIT/EXIT. The final menu is the FileIDs that are on the tape. Select the fileIDs you need, and then click on QUIT/EXIT. You will be asked a couple of more questions about whether you are ready to read the tape. GO_RDTAP will also display the number of kilobytes that you have selected to read, so check the available space on the disk selected before continuing.

A Using the Yohkoh Package from a Remote Node

A.1 Logging in from a remote node

There are several ways to log into a remote node. Here are a few of them:
From Unix or DEC-Ultrix:

% telnet node_name
% telnet isass0
% telnet node_nuber
% telnet 192.68.162.109
% rlogin node
% rlogin isass2
% rlogin node -l username
% rlogin isass2 -l morrison

From DEC-Ultrix:

% dlogin node
% dlogin 24707::
% dlogin 24.131

From VMS:

$ set host node
$ set host sag
$ set host 24707

When using TELNET, DLOGIN, and SET HOST you will be prompted for the username and password of the account you want to log into. For RLOGIN, it will check the file .rhosts to see if the account and machine that you are logging in from should automatically log you into the remote account. A sample .rhost line is:

```
sxt1.space.lockheed.com        morrison        #  DECstation 5000
```

A.2 Redirecting X-Window Outputs

If you have logged into a remote computer from a workstation, it is possible to redefine the machine that will display X-window output. The default for IDL is to make X-windows for plots and image displays. Two things have to be done before you can redirect the X-windows output: first the output device needs to be defined to be the local node, and second the security for the local node needs to be modified to give permission for the remote node to create the window.

A.2.1 Defining X-Window Output Node

For Unix and DEC-Ultrix machines, a sample command would be:

```
% setenv DISPLAY sxt2:0
% setenv DISPLAY 192.68.162.109:0
```

For DEC-Ultrix machines, it is possible to use DECNET protocal to create the local X-window. This is not recommeded since it is slow, but if it is necessary, a sample command is:

```
% setenv DISPLAY sag::0
% setenv DISPLAY 24707::0
```

For VMS machines, a sample command would be:

```
$ set/display/create/super/node=sxt2
```

A.2.2 Modifying Security to Accept Remote X-Windows

For the DEC-Ultrix machines using DEC Windows, the following steps are required:

- Find the 'Session Manager' window and make that the current window
- Click and hold on the 'Customize' item
- Drag and select the 'Security' item (by releasing the button)

- Enter the remote host name. An example would be 'sxt2' (without the quotes) if sxt2 was defined in the /etc/hosts file. If not, you would enter the full name 'sxt2.spac.lockheed.com'
- Click on 'Add' and then 'OK' to exit

For Sun machines... (necessary to have node name in the /etc/hosts file?)
For VMS machines...

A.3 Basics for Using FTP

binary use binary file type for transfer

prompt toggle between interactive and non-interactive prompting (it is important to be non-interactive when using 'mget')

cd change default directory on the remote machine

lcd change default directory on the local machine

get copy one file from remote to local

mget copy several files from remote to local

put copy one file from local to remote

A sample session might look like:

```
% ftp 133.74.8.100
username: guest
passwd: fuchinobe
FTP> binary
FTP> cd /ys/sxt/doc
FTP> get travel_sxt.txt
FTP> cd /ys/gen/doc
FTP> prompt
FTP> mget papers.*
FTP> quit
%
```

A.4 List of Institutes with the Yohkoh Package

- Institute of Space and Astronautical Sciences (ISAS) [isass0 thru isass4, flare1 thru flare12, isasxa thru isasxc]
- Lockheed Palo Alto Research Laboratory (LPARL) [sxt0 thru sxt4, sag]
- University of Hawaii [koa?, mamane?]
- University of California, Berkeley [sunspot]
- Stanford University [flare]
- Solar Physics Research Corporation (SPRC) [NOAO?]
- Goddard Space Flight Center/NSSDC (GSFC) [umbra]
- National Astronautical Observatory of Japan (NAOJ) [??]
- Mullard Space Science Laborartory (MSSL) [??]
- Rutherford Appleton Laboratory (RAL) [??]
- Navel Research Laboratory (NRL) [aspen,ssd0?]
- University of Tokyo [??]
- Kyoto University [??]
- CRL - Morabashi? [??]
- CRL - Akioka? [??]
- Nagoya ? [??]
- Nobayama ? [??]

A.5 List of Node Names and Numbers

	DECNET	Internet	Machine
SAG	24.131	192.68.162.134 (sag.space.lockheed.com)	DEC-VMS
SXT	24.134	192.68.162.107 (sxt.space.lockheed.com)	SGI-Unix
SXT1	24.362	192.68.162.100 (sxt1.space.lockheed.com)	DEC-Ultrix
SXT2	24.365	192.68.162.109 (sxt2.space.lockheed.com)	DEC-Ultrix
SXT3	24.366	192.68.162.110 (sxt3.space.lockheed.com)	DEC-Ultrix
SXT4	24.369	192.68.162.137 (sxt4.space.lockheed.com)	DEC-Ultrix
spot		133.40.2.120	SUN?
ISASS0	40.932	133.74.8.100 (isass0.solar.isas.ac.jp)	DEC-Ultrix
ISASS1	40.933	133.74.8.101 (isass1.solar.isas.ac.jp)	DEC-Ultrix
ISASS2	40.934	133.74.8.102 (isass2.solar.isas.ac.jp)	DEC-Ultrix
ISASS3	40.935	133.74.8.103 (isass3.solar.isas.ac.jp)	DEC-Ultrix
ISASS4	40.936	133.74.8.104 (isass4.solar.isas.ac.jp)	DEC-Ultrix
ISASXA	40.927	133.74.8.110 (isasxa.solar.isas.ac.jp)	DEC-VMS
ISASXB	40.928		DEC-VMS
ISASXC	40.929		DEC-VMS
FLARE1		133.74.8.1 (flare1.solar.isas.ac.jp)	SUN
FLARE2		133.74.8.7 (flare2.solar.isas.ac.jp)	SUN
FLARE3		133.74.8.3 (flare3.solar.isas.ac.jp)	SUN
FLARE4		133.74.8.4 (flare4.solar.isas.ac.jp)	SUN
FLARE5		133.74.8.5 (flare5.solar.isas.ac.jp)	SUN
FLARE6		133.74.8.6 (flare6.solar.isas.ac.jp)	SUN
FLARE7		133.74.8.87 (flare7.solar.isas.ac.jp)	MIPS
FLARE8		133.74.8.88 (flare8.solar.isas.ac.jp)	MIPS
FLARE9		133.74.8.89 (flare9.solar.isas.ac.jp)	MIPS
FLARE10		133.74.8.90 (flare10.solar.isas.ac.jp)	MIPS
FLARE11		133.74.8.91 (flare11.solar.isas.ac.jp)	MIPS
FLARE12		133.74.8.92 (flare12.solar.isas.ac.jp)	MIPS

MSSL	19.253		DEC-VMS
ssd0	128.60.8.1	(ssd0.nrl.navy.mil)	SUN?
aspen	128.60.8.72	(aspen.nrl.navy.mil)	SUN
koa	128.171.167.1	(koa.ifa.hawaii.edu)	SUN?
mamane		(mamane.ifa.hawaii.edu)	SUN?
sunspot		(sunspot.ssl.berkeley.edu)	SUN?
star	36.10.0.5	(star.stanford.edu)	???
flare		(flare.stanford.edu)	DEC-Ultrix?
panache		(panache.stanford.edu)	DEC-Ultrix?
umbra	128.183.57.24	(UMBRA.gsfc.nasa.gov)	DEC-VMS?

NOTE: To go from DECNET xx.yyy convention to the single node number notation, take the xx portion, multiply by 1024, and add 'yyy'.
So SAG becomes 24*1024 + 131 = 24707::

B Accessing the University of Hawaii SPAM Database

SPAM: A Canned Internet-Accessible Database
of Interest to Solar Flare Researchers
R. C. Canfield, H. S. Hudson, E. Kiernan, T. R. Metcalf, J.-P. Wuelser
University of Hawaii, Institute for Astronomy

The University of Hawaii has established a searchable database, called SPAM (Spectroscopy and Polarimetry at Mees), which contains logs of observations made at Mees Solar Observatory (Haleakala, Maui). Of more general interest, the database also includes the Events List and Region Report from the Space Environment Laboratory (Boulder). Logs from YOHKOH are currently being added. Hence, SPAM can be used to determine, for example, whether Mees has vector magnetograms of a certain NOAA AR or whether YOHKOH has certain types of observations in specified time ranges. As well, it can be used to search the SEL database for flares with selected attributes.

Included logs (and searchable attributes, in addition to date, day of year, and time) are: Mees Solar Observatory Log (instrument, NOAA AR, data type, observing setup), SEL Event List (NOAA AR, X-ray Class), SEL Region Report (NOAA AR), YOHKOH Orbit Summary, YOHKOH SXT Quiet Mode PFI Observations (latitude, longitude, X-ray and optical image size), YOHKOH Flare Observations (latitude, longitude, specific channel counts or ratios).

SPAM runs on a Sun workstation at Mees Solar Observatory, and is available over Internet. Simply access (e.g., telnet) koa.ifa.hawaii.edu 128.171.167.1) from any vt100, Sun, or xterm emulator. Log on as spam lower case); there is no password. New users are asked to read release notes and hints.

B.1 Procedure for SPAM Access

1. telnet to koa.ifa.hawaii.edu (128.171.167.1)
2. login name spam
3. no password

4. enter terminal type (vt100, xterm, etc - it prompts)

5. read 'help for new users' as offered if it's the first time you've used it.

B.2 MGRAMS

The files in this directory contain data from the Mees Solar Observatory Stokes Polarimeter (Mickey, 1985, Solar Physics, 97, 223). Please feel free to use these data; if you do so, please send a one or two sentence mail message describing how you use them to the Mees Data Use Coordinator, Dr. Richard C. Canfield, canfield@mamane.ifa.hawaii.edu. In publications please acknowledge that they were obtained at Mees Solar Observatory, and were supported by NASA grant NAGW 1542 to the University of Hawaii. Please send a copy of your paper to the Data Use Coordinator at Institute for Astronomy, 2680 Woodlawn Drive, Honolulu, HI 96822, USA.

The data files are kept in this directory for about a month. If you need earlier data, please contact poldata@koa.ifa.hawaii.edu.

The data are reduced using the method of Ronan, Mickey and Orrall (1987), Solar Physics 113, 353. This method is known to introduce certain systematic errors, and these data should not be used for quantitative analyses. If you wish data for such analyses, please contact the Mees Data Use Coordinator.

These data are in the FITS format. Be sure to use binary ftp format when transferring files. The file name convention is as follows:

```
IYYMMDD.hhmmss

where

I = Data code:   B = Vector Magnetic Field
                 P = Raw polarimeter data

YY = Year of observation (2 digit)
MM = Month of observation (2 digit)
DD = Day of observation (2 digit)
hh = hour of start of observation
mm = minute of start of observation
```

```
 ss = second of start of observation (optional)

To read and display the FITS file in IDL, the following routines in this
directory should be used:

 bfits.pro     Reads the FITS file
 Bvector.pro   Displays a vector magnetogram

The following are called by bfits.pro or Bvector.pro

 rfits.pro
 rkey.pro
 trmvect.pro
 specialct.pro
 tag_index.pro
```

An example is:

IDL> mag = bfits('B910419.1234') ; Read the FITS file
IDL> Bvector,mag ; Display magnetogram on screen
IDL> Bvector,mag,/incolor ; Display magnetogram on color screen
IDL> Bvector,mag,out='print' ; Generate post-script file idl.ps

C Accessing the NAOJ H-Alpha Data

The following is the procedure to access NAOJ H-Alpha images.

% telnet spot
username: guest
passwd: [clue: where is NAOJ?]
% show

After this it is straightforward as pop-up menus appear to guide you.

To copy across the data rather than just simply viewing the images or dumping the images to the color printer you should do the following:
FTP to spot (133.40.2.120):

% ftp spot
username: guest
passwd: [clue: where is NAOJ?]
FTP> cd /solar/monocro
FTP> binary

List the files in the diretory by typing the usual Unix 'ls' command, e.g.,

FTP> ls ha920503*

Then copy the file(s) you want using `mget`, e.g.,

FTP> mget ha920503*

D Making Laser Disk and VCR Movies

D.1 NVS-Sony Laser System at ISAS

D.1.1 Hardware: Positioning the Sony Laser Disk

1. Check that the green light on the NVS is on. Power cycle if necessary.
2. Check that the RS-232 cable is connected from the NVS to the Sony LVR-5000.
3. Make sure the correct optical disk is inserted (check side A or B).
4. Put the 'LOCAL/REMOTE' switch in the local position.
5. Depress the 'REC STANDBY' button to initiate blank frame check. This will find positions of all blank areas on the disk. When completed, the disk will be positioned to the last recorded frame before the nearest (not necessary the first) blank frame.
6. Use the record 'SKIP' buttons to position the laser disk to the last recorded frame before the desired starting frame number (consult the Sony Laser disk log book). On the display, Cont. Rec = is the last recorded frame before the blank frame; DUR = number of blank frames.
7. Toggle the format of the numbers displayed to show the time-format. Record numbers in the log book. To start over, press the 'CLEAR' button.
8. Set the left console switch to 'remote'.

D.1.2 Software

1. Login onto a non-isass0 machine (isass1-4).
2. Enter IDL. Make sure no other IDL window jobs are running on the same workstation. If there are two jobs, the color table could be corrupted.
3. Then enter one of the following:

IDL> go_nvs5,/sfd ; Standard weekly movie making
IDL> go_nvs5,/help ; For on-line documentation
IDL> go_nvs5,increm=5,/sfd ; Less of an impact on workstation

The temporary NVS files will be written to your home directory. Use the nvsdirfn='/1p/user/nvs0' switch to redirect to another location.

The default increment is 10. This is good if no one else is using the workstation at the same time.

If the /sfd switch is given, go_nvs5 will read the files from /yd1/sfd. It will run the program with the usual default answers. These are:

```
* Use default sfd files for input data?      [Default: Y ] ; Y for /yd1/sfd
* Display a 2nd image set?                   [Default: Y ] ; use default
* Decompress the 2nd image?                  [Default: Y ] ; use default
* Hist_scale type scaling on 2nd image?      [Default: Y ] ; use default
* Enter first comment line (10 char max)     [Default: SXT Movie ]   ;use default
* Enter second comment line (15 char max)    [Default: sfd92_34a.03 ];use default
* Enter a color table value:                 [Default: 3 ] ; use default
* Enter a color gama  value:                 [Default: 1.00000 ] ; use default
* Enter color table for 2nd image display:   [Default: 0 ] ; use default
* Enter the first image to process           [Default: 0 ] ; use default
* Enter the last  image to process           [Default: xxx ] ; use default
   Next you will get a message about the current frame number.  Verify that
   it is the desired frame number of time.  Then, ...
* Enter C to continue or anything to quit    [Default: C ] ; use default
```

D.1.3 Adding Another Movie

The above procedure is for starting from scratch. If you have just completed a run and the Sony frame number has not been changed, then just type:

IDL> go_nvs5 [,/sfd]

to make the next movie, answering the questions as before.

D.1.4 Problems

1. Invalid Num – If the message 'Invalid Num' appears on the display monitor, the nvs box is not in sync with the Sony laser disk. To correct this situation begin with a blank frame check (see above hardware procedure) and follow that with software procedure section II).

2. One common problem is to forget to attach the RS-232 cable. This is sometimes switched to ISASS3 to control the recorder during playback to record with the VCR).

3. How to toggle the format frame/time format: depress the frame/time button located on the lower-left side of the console number keypad– where frame # is a simple number and time has the format of mm:ss:ff.

D.2 Peritek-Panasonic Laser System at LPARL

D.3 Making VCR Movies from the ISAS Sony System

MK_VCR is an IDL procedure which can be used to control the Sony Laserdisk recorder. In this way sequences can be displayed easily in order to make VCR video recordings.

The MK_VCR procedure reads an ascii file (described) below and then calls sonyloop to actually control the laser disk player.

D.3.1 Initial setup

- First, make sure the RS-232 cable from isass3 is connected to the back of the Sony recorder (it requires an adapter connector).
- Switch the local/remote switch to remote.
- Log on to isass3 and enter IDL.

D.3.2 Short test of hardware communications.

You can preform a simple test to see if things are working correctly by typing:

IDL> sonyloop,6000 ; Position laser disk to frame number 6000
IDL> sonyloop,6000,6200 ; Play from 6000 to 6200 at normal speed
IDL> sonyloop,6000,6200,sec=2 ; Play a factor of 2 slower

D.3.3 Running MK_VCR

If things look like they are working correctly, you can then try mk_vcr. To get a template of the file the mk_vcr will read, execute the following unix command to copy the example file to your own directory:

% cp /ys/gen/soft/atest/movie/mk_vcr_example.txt .

After you copy the example file, do the following. Insert Side B of the laser optical disk. Then,

IDL> mk_vcr,'mk_vcr_example.txt',1,3

which execute lines 1 to 3 of the file. (Line 0 puts up a blank frame). If you type

IDL> mk_vcr,'mk_vcr_example.txt

the routine will inform you of the number of lines in the file and ask you to enter the start and stop line numbers.

If you want to make a movie, you could try the following IDL program:

```
sonyloop,8000                          ;Get it to a blank frame
for i=0,2 do mk_vcr,'mk_vcr_example.txt',0,23 ;There are 24 lines in the file
sonyloop,8000                          ;End on a blank frame
end
```

As you can see, the sequence in the 'for loop' will execute 3 times. Each time through takes about 7 minutes, so the whole movie will be just over 20 minutes in length.

D.3.4 How to create/modify the file that MK_VCR uses

The file has some descriptions about how to change it. Basically, the lines in the file look like the following:

```
 8000 ; Position to a blank frame (provide leader)
 6077, 6077,    4,    0, [ -1  0  0  0] "Movie Title Page"
 6078, 6078,    2,    1, [ -1  0  0  0] "Nov 1991 Movie - Title"
 5048, 5225,    2,    1, [  1  2  1 -1] "Nov 1991 Movie"
; c0    c1      c2    c3       [c4]          "  c5   "
```

The above sequence will display the Nov 91 press release sequence if Side B is inserted. Some notes about the format:

- Text following a ':' is considered to be a comment.
- Only the first column is required. All subsequent columns are optional.
- c0 is starting frame number.
- c1 is the ending frame number.
- c2 is the wait (in sec) after displaying the first frame
- c3 is the wait (in sec) after displaying the last frame
- c4: The number in the boxes are the speeds that are used by sonyloop. Up to four different speeds can be specified.

  ```
  0 is no operation
  1 is normal speed
  2 is twice slower
  3 is 3x slower, etc
  -2 is 3x faster.
  ```

If a fifth parameter is specified in the brackets, it is a repeat number. If this number is negative, sonyloop will play the sequence forward and backward. For example,

```
5048, 5225, 2,  1, [  1  0  0  0  1] "Nov 1991 Movie" ;1 time forward
5048, 5225, 2,  1, [  1  0  0  0 -1] "Nov 1991 Movie" ;1 time forward/backward
5048, 5225, 2,  1, [  1  0  0  0 -3] "Nov 1991 Movie" ;3 times forward/backward
```

SPECIAL NOTES:

- Sonyloop doesn't send the correct commands to the Sony Laser Recorder to go backwards. Thus, no matter what speed you specify, it always goes at normal speed in reverse. (Maybe a Japanese reader can study the manual and fix this problem).
- Sonyloop tries to compute the time it will take the laser disk recorder to play the specified sequence. Because the timings in IDL can be off by a certain amount (less than or equal to about 1sec), it is wise to make one of the wait parameters (c2 or c3) equal to at least 1. If they are set to 0, IDL will sometimes send the next sequence before the previous sequence has had a chance to complete.

E Using the Exabyte Drives

E.1 Drives At ISAS

There are exabyte drives on isass0, flare3, and flare4.

F Using Magneto-Optical (MO) Disks at ISAS

For data on MO you must first put the disk with the data into one of the MO readers on FLARE3 or FLARE4. There are two drives on FLARE3 and one drive on FLARE4. The directory names are:

```
/flare3.mo0
/flare3.mo1
/flare4.mo0
```

If there is another MO in the reader already you must first type one of the following (depending on the drive) before removing one MO and inserting the new one:

```
% mo_umount30
% mo_umount31
% mo_umount40
```

You may now physically swap the MO disks. Then type one of the following:

```
% mo_mount30
% mo_mount31
% mo_mount40
```

followed by

```
% exportfs -a
```

Then type one of the following:

```
% umount /flare3.m0
% umount /flare3.m1
% umount /flare4.m0
```

and then finally, type one of the following: (possibly not necessary, but if mounting alone doesn't work then try it) then type:

```
% umount /flare3.m0
% umount /flare3.m1
% umount /flare4.m0
```

Also note that it is necessary to unprotect the MO disk drives before they will mount.

G Guidelines for Writing Distributed Software

G.1 Names of Routines

The first concern is not to use a name that is already being used. To check if the name exists on the /ys tree on a Unix machine, you can use the symbol 'chk_name'. Here is an example (note that the .PRO portion is required):

```
% chk_name yodat.pro
```

On a VMS machine, the following command should work:

```
$ dir ys:[...]yodat.pro
```

It is possible to do the search from within IDL on any of the machines using a command like:

```
IDL> print, path_lib('yodat')
IDL> print, path_lib('yo*') ;for all routines beginning with 'yo'
```

The second concern is use care when chosing the name. It should not be general enough to cause conflicts or confusion between all of the other intstrument teams. For example, the original name of SXT_DECOMP was DECOMP. Every instrument has a decompression routine, so the name of the instrument needs to be incorporated in the name of the routine for general routines. The name of the routine should not be shorter than four letters, and underscores are acceptable.

G.2 Standard Headers

The following standard IDL should be included at the top of each routine.

```
;+
; NAME:
; PURPOSE:
; CATEGORY:
```

```
; CALLING SEQUENCE:
; INPUTS:
; OPTIONAL INPUT PARAMETERS:
; OUTPUTS:
; OPTIONAL OUTPUT PARAMETERS:
; COMMON BLOCKS:
; SIDE EFFECTS:
; RESTRICTIONS:
; PROCEDURE:
; MODIFICATION HISTORY:
;-
```

G.3 Categories

In the near future we will supply a list of categories. Each routine will require at least one category, but can have several categories listed.

G.4 Modularize

Instead of making large main programs, try to break the program up into smaller general purpose procedures or functions. This will allow users to take advantage of the smaller building blocks to make different programs and to avoid duplication of code. It will also allow for a single point of maintenance.

G.5 Portability

It is important to have all software be portable between different computers and different operating systems. One of the biggest mistakes is to put full file names with directories. Since the software needs to be able to be used on VMS and UNIX machines, we make use of the routine CONCAT_DIR. This routine takes a logical directory name and the file name, and puts them together correctly depending on the system. For example, the command

IDL> outfil = concat_dir('$DIR_GEN_OBS', 'obs92_40a.01')

will return '$DIR_GEN_OBS/obs92_40a.01' when being run on a UNIX machine, and will return '$DIR_GEN_OBS:obs92_40a.01' when being run on a

VMS machine. Always use logicals pointing to files in the YS directory area - do not write software that points to personal files since those will not be available at other locations.

G.6 Don't Hardwire to Fixed Units

Use /GET_LUN on the open statements, and /FREE on the window commands

GOOD EXAMPLE:

```
IDL> openr, lun, filename, /get_lun
```

BAD EXAMPLE:

```
IDL> openr, 1, filename
```

H Listing of All Routines by Category

Index

ASCII LIST OF CMWC02/C107679

FILE 1 RECORD 1 8192 BYTES

```
MkTap ver # 0.005
Binary Tape directory file size:  6934
date written: Sat Mar 27 22:13:39 1993
Number of CPIO files:  99, CPIO Swit
ches:   -oBc
IDL !verison info:      mipsel,ultrix,3.0.0, Reformatter Ver #  1.17
Tape Rel.#   1, Tape Name: 93_11a.01-1.17
Prefixes
 for Weekly logs:                        fem,obs,pnt,sdc,sfd
Extensions for Weekly logs: 0201010107
1st fileID: 930307.0127, Last file
 fileID: 930313.1455
Following the last cpio file there are   1 Software Archive files.
tar command: cd /ys/swmaint/tar,tar -cvtf /d
ev/nrmt0h -C /ys/swmaint/tar *.tar
Tar  File  name                                                  Size(bytes)
/ys/swmaint/tar/ydb_evn
.tar                                  1280000
/ys/swmaint/tar/ydb_fem.tar                                      1003520
/ys/swmaint
/tar/ydb_gev.tar                                    573440
/ys/swmaint/tar/ydb_gxt.tar                                     24965120

/ys/swmaint/tar/ydb_nar.tar                                       317440
/ys/swmaint/tar/ydb_sot.tar
     1740800
/ys/swmaint/tar/ydb_ssl.tar                                       491520
/ys/swmaint/tar/ydb_xad.tar
               1341440
/ys/swmaint/tar/ydb_xbd.tar                                       870400
/ys/swmaint/tar/ys_atest.tar
                      757760
/ys/swmaint/tar/ys_bcs.tar                                      11192320
/ys/swmaint/tar/ys_gen.
tar                                   6277120
/ys/swmaint/tar/ys_hxt.tar                                        61440
/ys/swmaint
/tar/ys_idlfix.tar                                  10240
/ys/swmaint/tar/ys_site.tar                                     2314240

/ys/swmaint/tar/ys_sxt.tar                                      2170880
/ys/swmaint/tar/ys_ucon.tar
     3450880
/ys/swmaint/tar/ys_wbs.tar                                        471040
  FileID    cpio     fem     obs      pnt      sd
c     sfd
     93_11a   89    7088 3019680 4595056  590864 *******       0       0       0       0       0
  FileID    cpio     ADA
    BDA      CBA      HDA      SFR      SPR      WDA      SB1
930307.0127   91   29648   17584  106272    5792  132048   22128   20464
   0       0        0
930307.0304   91  170560  164352  579312   72064  986928  242448  236720       0       0       0
930307.0442
91  170560  157968  579312   72368  986928  242448  236720       0       0       0
930307.0619   91  170560  149936  579312   71456
```

ASCII LIST OF CMW403/C107680

FILE 1 RECORD 1 8192 BYTES

```
MkTap ver # 0.005
Binary Tape directory file size:  7142
date written: Tue Mar 30 08:17:51 1993
Number of CPIO files: 103, CPIO Swit
ches:   -oBc
IDL !verison info:      mipsel,ultrix,3.0.0, Reformatter Ver #  1.17
Tape Rel.#   1, Tape Name: 93_12a.01-1.17
Prefixes
 for Weekly logs:                        fem,obs,pnt,sdc,sfd
Extensions for Weekly logs: 0201010107
1st fileID: 930314.0217, Last file
 fileID: 930320.2349
Following the last cpio file there are   1 Software Archive files.
tar command: cd /ys/swmaint/tar,tar -cvtf /d
ev/nrmt0h -C /ys/swmaint/tar *.tar
Tar  File  name                                                  Size(bytes)
/ys/swmaint/tar/ydb_evn
.tar                                  1280000
/ys/swmaint/tar/ydb_fem.tar                                      1003520
/ys/swmaint
/tar/ydb_gev.tar                                    573440
/ys/swmaint/tar/ydb_gxt.tar                                     24965120

/ys/swmaint/tar/ydb_nar.tar                                       317440
/ys/swmaint/tar/ydb_sot.tar
     1740800
/ys/swmaint/tar/ydb_ssl.tar                                       491520
/ys/swmaint/tar/ydb_xad.tar
                1341440
/ys/swmaint/tar/ydb_xbd.tar                                       870400
/ys/swmaint/tar/ys_atest.tar
                       757760
/ys/swmaint/tar/ys_bcs.tar                                      11192320
/ys/swmaint/tar/ys_gen.
tar                                   6277120
/ys/swmaint/tar/ys_hxt.tar                                        61440
/ys/swmaint
/tar/ys_idlfix.tar                                   10240
/ys/swmaint/tar/ys_site.tar                                      2314240

/ys/swmaint/tar/ys_sxt.tar                                       2170880
/ys/swmaint/tar/ys_ucon.tar
     3450880
/ys/swmaint/tar/ys_wbs.tar                                        471040
  FileID    cpio     fem     obs      pnt      sd
c     sfd
     93_12a   89    7088 1110624 1798432  590864 *******       0        0        0        0        0
  FileID    cpio     ADA
    BDA      CBA      HDA      SFR      SPR      WDA      SB1
930314.0217   91    87552   81376  298512   32240  526608   117168  107760
   0        0        0
930314.0354   91   168512  170784  570672   70240  921168   242448  232752        0        0        0
930314.0532
91   169024  171776  572832   70848  986928  238128  232752        0        0        0
930314.0709   91   186976  203104  613872   86048
```

ASCII LIST OF CMW404/C107681

```
FILE 1      RECORD 1      8192 BYTES

MkTap ver # 0.005
Binary Tape directory file size:  6986
date written: Wed Apr  7 15:42:50 1993
Number of CPIO files: 100, CPIO Swit
ches:   -oBc
IDL !verison info:       mipsel,ultrix,3.0.0, Reformatter Ver #  1.17
Tape Rel.#   1, Tape Name: 93_13a.01-1.17
Prefixes
 for Weekly logs:                          fem,obs,pnt,sdc,sfd
Extensions for Weekly logs: 0201010107
1st fileID: 930321.0126, Last file
 fileID: 930327.2256
Following the last cpio file there are   1 Software Archive files.
tar command: cd /ys/swmaint/tar,tar -cvtf /d
ev/nrmt0h -C /ys/swmaint/tar *.tar
Tar  File  name                                                    Size(bytes)
/ys/swmaint/tar/ydb_evn
.tar                                    1280000
/ys/swmaint/tar/ydb_fem.tar                                         1003520
/ys/swmaint
/tar/ydb_gev.tar                                        573440
/ys/swmaint/tar/ydb_gxt.tar                                        24965120

/ys/swmaint/tar/ydb_nar.tar                                          317440
/ys/swmaint/tar/ydb_sot.tar
      1740800
/ys/swmaint/tar/ydb_ssl.tar                                          491520
/ys/swmaint/tar/ydb_xad.tar
                1341440
/ys/swmaint/tar/ydb_xbd.tar                                          870400
/ys/swmaint/tar/ys_atest.tar
                        819200
/ys/swmaint/tar/ys_bcs.tar                                         11192320
/ys/swmaint/tar/ys_gen.
tar                                     6277120
/ys/swmaint/tar/ys_hxt.tar                                           61440
/ys/swmaint
/tar/ys_idlfix.tar                                      10240
/ys/swmaint/tar/ys_site.tar                                         2314240

/ys/swmaint/tar/ys_sxt.tar                                          2170880
/ys/swmaint/tar/ys_ucon.tar
      3543040
/ys/swmaint/tar/ys_wbs.tar                                           471040
   FileID     cpio      fem      obs      pnt      sd
c      sfd
     93_13a    89     7088 2373728 4183456  590864 *******        0        0        0        0        0
   FileID     cpio      ADA
     BDA       CBA       HDA       SFR       SPR       WDA       SB1
930321.0126    91   161328   166384   557712    68112   921168   229488   224816
    0        0         0
930321.0303    91   515392   557632 1756512   930784   263568 2972688   748592        0        0        0
930321.0441
91   801328   872608 2702592   363600 5772624 1313424 1218800        0        0        0
930321.0618    91   172624   189088   570672    80272
```

ASCII LIST OF CMW405/C107682

```
FILE 1     RECORD 1     8192 BYTES

MkTap ver # 0.005
Binary Tape directory file size:  7194
date written: Sun Apr 18 15:19:31 1993
Number of CPIO files: 104, CPIO Swit
ches:   -oBc
IDL !verison info:      mipsel,ultrix,3.0.0, Reformatter Ver #  1.18
Tape Rel.#   1, Tape Name: 93_14a.01-1.18
Prefixes
 for Weekly logs:                        fem,obs,pnt,sdc,sfd
Extensions for Weekly logs: 0201010107
1st fileID: 930328.0033, Last file
 fileID: 930403.2343
Following the last cpio file there are   1 Software Archive files.
tar command: cd /ys/swmaint/tar,tar -cvtf /d
ev/nrmt0h -C /ys/swmaint/tar *.tar
Tar  File  name                                                  Size(bytes)
/ys/swmaint/tar/ydb_evn
.tar                                  1280000
/ys/swmaint/tar/ydb_fem.tar                                      1003520
/ys/swmaint
/tar/ydb_gev.tar                                    573440
/ys/swmaint/tar/ydb_gxt.tar                                     24965120

/ys/swmaint/tar/ydb_nar.tar                                       317440
/ys/swmaint/tar/ydb_sot.tar
     1740800
/ys/swmaint/tar/ydb_ssl.tar                                       491520
/ys/swmaint/tar/ydb_xad.tar
                1341440
/ys/swmaint/tar/ydb_xbd.tar                                       870400
/ys/swmaint/tar/ys_atest.tar
                      1863680
/ys/swmaint/tar/ys_bcs.tar                                      11192320
/ys/swmaint/tar/ys_gen.
tar                                   6277120
/ys/swmaint/tar/ys_hxt.tar                                        61440
/ys/swmaint
/tar/ys_idlfix.tar                                  10240
/ys/swmaint/tar/ys_site.tar                                     2314240

/ys/swmaint/tar/ys_sxt.tar                                      2170880
/ys/swmaint/tar/ys_ucon.tar
     3584000
/ys/swmaint/tar/ys_wbs.tar                                        471040
  FileID    cpio     fem     obs     pnt     sd
c     sfd
     93_14a   89    7088 2014048 3714256 1181136 *******       0       0       0       0       0
  FileID    cpio     ADA
    BDA     CBA      HDA     SFR     SPR     WDA     SB1
930328.0033   91  638880  673488 2173392  279696 4068240  974928  937072
   0       0       0
930328.0211   91  811568  895152 2745792  374544 5182800 1229264 1238640       0       0       0
930328.0348
91  164416  185536  553392   78448  787632  185680  256560       0       0       0
930328.0525   91  818768  917136 2741472  386400
```