STEREO
IMPACT

IDPU Command Formats

CmdFmt_C.doc
Version C – 2003-Jun-19

David Curtis, UCB IMPACT Project Manager

Document Revision Record

Rev.	Date	Description of Change	Approved By
A	2002-Apr-4	Preliminary Draft	-
B	2002-Jun-20	Change ApID allocations	
C	2003-Jun-19	Add command checksum	

Distribution List

Dave Curtis, UCB
Mike Hashii, UCB
Peter Schroeder, UCB
Lynn Kistler, UNH
Andrew Davis, Caltech
Bob Radocinski, JPL
Kristin Wortman, GSFC

Table of Contents

1. Introduction

This document describes the command formats used by the STEREO IMPACT IDPU. The IMPACT IDPU is the interface between the STEREO spacecraft and the IMPACT and PLASTIC instrument suite. All commands to these instruments are routed through the IDPU via a 1553 spacecraft interface. The SEP suite of instruments has its own processing element, and so commands to SEP are forwarded by the IDPU (based on ApID) to the SEP instrument without interpretation. For this reason, SEP commands are only briefly mentioned in this document.

IMPACT and PLASTIC instrument commands are provided by the Instrument GSE or Payload Operations Center (POC). These commands are routed via the spacecraft emulator to the instrument during bench testing, and via the Mission Operations Center (MOC) during spacecraft Integration and Test and later in post-launch operations. This document also describes the mnemonic formats used by the IMPACT GSE/POC for encoding commands. The PLASTIC GSE/POC may or may not use this same encoding scheme (TBD-UNH-001).

1.1. *Document Conventions*

In this document, TBD (To Be Determined) means that no data currently exists. A value followed by TBR (To Be Resolved) means that this value is preliminary. In either case, the value is typically followed by UCB, indicating who is responsible for providing the data, and a unique reference number.

1.2. *Applicable Documents*

The following documents include drawings and STEREO Project policies, and are part of this specification. In the event of a conflict between this Specification and the following documents, this Specification takes precedence. Most documents and drawings can be found on the Berkeley STEREO/IMPACT FTP site:

http://sprg.ssl.berkeley.edu/impact/dwc/

Or on the APL STEREO web site at:

https://sd-forum.jhuapl.edu/stereo/

1. IMPACT_ICD_RevA - Spacecraft to IMPACT ICD
2. MOC-POC_ICD_RevA - MOC to POC ICD
3. ICD/ImpactSerialInterface_G - IDPU to Instrument Serial Interface Spec.
4. ICD/IMPACT_CTM_H - IMPACT Command & Telemetry database

2. STEREO Command Formats and Routing

All STEREO commands are formatted into CCSDS packets as described in Reference 2. IMPACT and PLASTIC commands are identified and routed by the MOC and spacecraft based on routing information (ApIDs) in the packet header. Command packets are variable length (up to 1088 bytes), and contain a primary header (including APID, sequence counter, and length information), optional secondary header (containing execution time used by the spacecraft stored command system), and data field. The headers are described in reference 2.

The Instrument GSE/POC provides commands to the Spacecraft Emulator or MOC as command packets with an additional header as described in reference 2. The added header includes information on when the MOC should forward the commands to the instrument (not used by the emulator), and some text information about the command. The spacecraft emulator removed the header and routes the command directly to the IDPU over the 1553 interface. The MOC routes the command to the spacecraft during the next command pass subject to the time constraints in the header. The spacecraft routes the commands to the IDPU via the 1553 interface. Commands may be routed to the instrument by the spacecraft either immediately or via a spacecraft stored command buffer based on the time code in the secondary header.

Commands are provided to the instrument in the form of 1553 messages in the FLTC format described in reference 1. The IDPU must first reconstruct command packets from these FLTC. The IDPU then routes the packets to the various tasks based on the packet ApID.

Note that command routing between the ground and the spacecraft is not fully closed-loop. It is not clear how the system will respond to a lost command. It is therefore important that commands be self-contained as much as possible, and rely as little as possible on the sequence of commands. Commands should also be efficiently encoded as command bandwidth is limited (typically 2000bps).

3. IMPACT / PLASTIC Command Routing

IMPACT and PLASTIC are each allocated a set of 128 command ApIDs. The MOC will only accept commands from the IGSE that have ApIDs within the associated instrument's ApID allocation, and the spacecraft will route any command with an IMPACT or PLASTIC ApID to the IMPACT IDPU. The IDPU further subdivides the ApIDs by tasks in the IDPU. Commands with SEP ApIDs are routed to SEP over the IDPU to SEP serial interface as described in reference 3 without further processing (other than removing the CCSDS header and checksum).

Table 3-1 Command ApID Routing

ApID range (Hex)	Task
200-27F	IMPACT
200-20F	System
210-21F	MAG
220-22F	SWEA
230-23F	STE
240-24F	Burst
250-25F	Spare
260-26F	SEP
270-27F	SEP Interface
300-37F	PLASTIC
300-30F	System (same as 200-20F)
310-31F	PLASTIC
320-37F	Spare

4. IMPACT / PLASTIC Command Formats

Each of the tasks has a table of parameters that controls its operations. The table load commands are used to change values in these tables.

The function execute allows executing one of a set of functions such as opening a cover.

The IDPU Memory Load command allows loading data at an arbitrary location in IDPU RAM. The EEPROM load allows writing to the IDPU EEPROM. The EEPROM is divided in to a number of buffers. More details of specific command formats can be found in reference 4. Other system and instrument memory load commands will be added (see reference 4)

Table 4-1 Typical Command Formats

Function	ApID	Parameters
Table Load	2x0, 3x0 (excluding 260,270)	Table Offset (16 bits), followed by data value(s)
Execute Function	2x1, 3x1 (excluding 261,271)	Function number to execute (8 bits)
IDPU Memory Load	202, 302	Starting address (24 bits), followed by data value(s)
IDPU EEPROM Load	203, 303	EEPROM Buffer Select (8 bits) followed by starting address (16 bits), followed by data value(s).

4.1. *Command Checksum*

The first byte of the data field of every command (immediately following the CCSDS header(s)) will be a command checksum byte. This byte shall be set such that the sum of all bytes in the command, including CCSDS header(s) and application data, but not any ground headers such as the SCM header, is zero. The checksum byte shall be checked and removed prior to forwarding the commands to the tasks (such as SEP). Any command received by the IDPU with a bad checksum will be rejected. The GSE should add this checksum byte automatically to each command.

5. IMPACT GSE Command Encoding

The IMPACT GSE/POCC includes a STOL-like text command scripting feature. Commands to the instrument are identified in the script by starting on a new line with a '/' character followed by the command mnemonic and parameter values. Commands may also be typed in directly to the GSE in the same format rather than being read from a script. Each command is encoded into a supplemented command message (SCM - see reference 2) by the GSE and routed to the MOC or spacecraft emulator.

5.1. *SCM Header Encoding*

The SCM header contains a number of fields that will be provided by the GSE. The SCM header format is described in reference 2. Some fields are fixed for the GSE; others are generated automatically, but may be changed by the user by commands preceding the actual instrument command. User-selectable values shall be programmable using commands that may be embedded in the command script. Note that an initialization script, run automatically when the GSE is started, may be used to over-ride the default values.

Table 5.1-1 SCM Header Values

SCM Header Field	Value
Message Type	"SCM"
Message length	Computed by GSE
Message Counter	Computed by GSE; increments by 1 for each message sent. The current value should be viewable on the GSE.
Signature Header, Info	I believe the signature is being replaced by a SSH encryption scheme. TBD-UCB-003
Facility ID	"IMPACT" or "PLASTIC", user selectable using GSE command "SCMFacilityID"; default = "IMPACT"
Spacecraft ID	"EA" for Ahead, "EB" for Behind spacecraft, user selectable using GSE command "SCMSpacecraftID". Default = Ahead.
# Packets	"01"
Enable Time	User selectable using GSE command "SCMEnableTime", default = "0000000000000"
Timeout	User selectable using GSE command "SCMTimeout", default = "-------------"
Delta	User selectable using GSE command "SCMDelta", default = "-----"
Script Line Number	Computed by the GSE based on the line number of the command in the current command script ("0000" if the command does not come from a script)
Script Name	Script file name ("NoScript" if the command does not come from a script). May be over-ridden in a script by the "PROC" command, usually the first line of a script.
Script Version Number	User selectable using GSE command "PROCVersion". Default = "0000"; always "0000" if the command does not come from a script.
Command Description	Copied from the command line, the first 32 characters following the "/" character.

Note that care must be taken in nested procs to keep track of the correct script name, version, and line number.

5.2. ***CCSDS Command Header Encoding***

The CCSDS command format is described in reference 2. Much of the information is fixed or computed automatically by the GSE (such as command length). The ApID shall be encoded based on the command database as described below, but the GSE should ensure that it is in the correct range (corresponding to the facility ID). The sequence count should increment for each command sent.

The default shall be real-time commands that have no secondary header. However the GSE can be directed to generate time-tagged commands with the “CommandTime” command. This command shall specify the absolute time of the next command, or the relative time to the current value of CommandTime. The value of CommandTime shall increment by a specified value (default = 1 second, settable by command ‘CommandDeltaTime’) as each command is sent. The system can be returned to real-time commands by a ‘CommandTime” command with no parameter.

5.3. *Command Authorization Return Receipt (ARR)*

The MOC or Spacecraft Emulator will send this message to the GSE when the command is received. No further commands should be sent and the command script processing should stop until this message is received. If the receipt indicates a failure, the user should be given the decoded failure code, plus the option of resending the command, continuing, or canceling the script.

5.4. *Command Log*

The GSE should keep a command log. This should include the transmission date/time, plus all the information in the SCM header (with the full command text rather than the first 32 bytes that are in the header), a hex version of the command packet, plus the verification code from the ARR.

5.5. *Command Encoding*

The basic command code shall be a series of values separated by one or more spaces. The first value shall be the ApID, followed by the command data. Data values shall be decimal unless preceded by “0x”, indicating hexadecimal coding for that value. Values may be signed or unsigned. Values may be signed (coded as 2’s complement). Values coded in 3 characters or less (excluding sign), 2 characters or less for hexadecimal (excluding 0x prefix) represents a 1-byte value. Values coded with 4 or 5 characters (3 or 4 for Hexadecimal) shall be coded as 16-bit values in 2 bytes, entered into the packet least significant byte first (with the exception of the ApID, which is coded Most significant byte first). Values coded as 6, 7,or 8 characters (5 or 6 for hexadecimal) shall be coded as 24-bit values in 3 bytes, while longer values shall be coded as 32-bit, 4-byte values. In addition, ASCII text values can be generated, one byte per character, by enclosing the values in quotes. Any mix of lengths types is allowed in a command. For example:

/0x220 0x1234 00001 “AB” -1

This is encoded as a command packet with ApId=220 (hex)
data=34 12 01 00 40 41 FF (hex)

5.6. *Command Mnemonics*

A command database shall be used to encode commands. This database shall be in a text format consisting of one command mnemonic per line. Following the mnemonic shall be a sequence of values that the mnemonic shall be converted into (in the same format as described in section 5.5). A comment may be embedded into the database preceded by a semicolon. When the GSE decodes a command line, an instance of a mnemonic will be replaced by the sequence of values indicated in the database. A command line shall contain a series of mnemonics and values intermixed. Further, mnemonics may be defined in terms of other mnemonics. For example, a command database might contain:

```
SWEA_LOAD        0x220         ;defined SWEA Table load ApID
MODE_ADDR        0x0019        ;defines the address of the SWEA Mode parameter
;                              define a command to load the SWEA Mode
SWEA_MODE        SWEA_LOAD MODE_ADDR
```

Then the command:

/SWEA_MODE 22

would be expanded into

/0x220 0x0019 22

which would then become a packet with ApID=220 (hex), data = 19 00 16 (hex)

Mnemonics should follow the normal conventions to aid in parsing, such as not starting with a digit, and containing no embedded spaces.

5.7. ***GSE variables***

The command encoding can be further enhanced by the use of variables. GSE commands can be defined to set and operate on variables. Variable names can then be embedded into commands, and will be replaced by the GSE with the current value of that variable. GSE variables might have a fixed naming convention like R0, or at least a fixed prefix like a period to avoid confusion between mnemonics and variable names. GSE variables are a feature that may be added later or eliminated. Some scheme must be used to indicate how many bytes the variable represents. (TBD-UCB-004)