**Table of Contents**

[1. Purpose 6](#_Toc364431321)

[2. Scope 6](#_Toc364431322)

[3. References 6](#_Toc364431323)

[4. Definitions 7](#_Toc364431324)

[5. Global Defined Values 8](#_Toc364431325)

[5.1 NUM\_ELECTRODES = 26 8](#_Toc364431326)

[5.2 NUM\_PULSES\_PROGRAM = 4 8](#_Toc364431327)

[5.3 NUM\_PROGRAM\_DEFS = 10 8](#_Toc364431328)

[5.4 NUM\_PROGRAM\_FREQUENCIES = 64 8](#_Toc364431329)

[5.5 PROGRAM\_NAME\_CHARS = 16 8](#_Toc364431330)

[5.6 PULSE\_NAME\_CHARS = 16 8](#_Toc364431331)

[5.7 PPC\_TEXT\_CHARS = 30 8](#_Toc364431332)

[5.8 MODEL\_NUMBER\_CHARS = 10 8](#_Toc364431333)

[5.9 SERIAL\_NUMBER\_CHARS = 20 8](#_Toc364431334)

[5.10 FIRMWARE\_VERSION\_CHARS = 10 8](#_Toc364431335)

[5.11 TRUE = 0x01 9](#_Toc364431336)

[5.12 FALSE = 0x00 9](#_Toc364431337)

[5.13 NUM\_STIM\_ASIC\_HV\_SETTINGS = 64 9](#_Toc364431338)

[5.14 NUM\_BYTES\_INST\_DATE\_TIME=8 9](#_Toc364431339)

[6. Limits 10](#_Toc364431340)

[6.1 Default Amplitude Steps 10](#_Toc364431341)

[6.2 Pulse Output Amplitude Step Index Limits 10](#_Toc364431342)

[6.3 Pulse Virtual Amplitude Step Index Limits 10](#_Toc364431343)

[6.4 Displayed Pulse Amplitude Limits 11](#_Toc364431344)

[6.5 Displayed Program Amplitude Limits 11](#_Toc364431345)

[6.6 Pulse Amplitude Limits 11](#_Toc364431346)

[6.7 Pulse Width Limits 11](#_Toc364431347)

[6.8 Pulse Width Step Limits 11](#_Toc364431348)

[6.9 Program Frequency Index Limits 12](#_Toc364431349)

[6.10 PoP and PPC EXID Limits 12](#_Toc364431350)

[6.11 CP EXID Limits 12](#_Toc364431351)

[6.12 Interphase delay Limits 12](#_Toc364431352)

[6.13 Passive recovery Limits 12](#_Toc364431353)

[6.14 Active recovery Limits 12](#_Toc364431354)

[6.15 Holdoff Limits 13](#_Toc364431355)

[6.16 Increment lockout Limits 13](#_Toc364431356)

[6.17 Lead Limit Limits 13](#_Toc364431357)

[6.18 Channel Calibration Limit 13](#_Toc364431358)

[6.19 Test Pulse Stim Phase Amplitude Limits 13](#_Toc364431359)

[6.20 Test Pulse Stim Phase Pulse Width Limits 13](#_Toc364431360)

[6.21 Test Pulse Recovery Ratio Select Limits 14](#_Toc364431361)

[6.22 Test Pulse Passive Recovery Width Limits 14](#_Toc364431362)

[6.23 Test Pulse Charge Balance Correction Width Limits 14](#_Toc364431363)

[6.24 Stim Asic HV Calibration Limits 14](#_Toc364431364)

[6.25 Amplitude Step Size Limits 14](#_Toc364431365)

[6.26 Replacement Interval Limits 14](#_Toc364431366)

[7. Common Command Parameter and Returned Data Constructs 15](#_Toc364431367)

[7.1 Program Number 15](#_Toc364431368)

[7.2 Program Map 15](#_Toc364431369)

[7.3 Pulse Map 15](#_Toc364431370)

[7.4 Pulse Index 15](#_Toc364431371)

[7.5 PPC Options Map 16](#_Toc364431372)

[7.6 Frequency Map 16](#_Toc364431373)

[7.7 Magnet Options Map 16](#_Toc364431374)

[7.8 Recovery ratio enable Bit Map 16](#_Toc364431375)

[7.9 IPG Battery Charging State 16](#_Toc364431376)

[7.10 IPG Temperature State 17](#_Toc364431377)

[7.11 Stimulation State 17](#_Toc364431378)

[8. Command Tokens 18](#_Toc364431379)

[8.1 Command Revision Compatibility 21](#_Toc364431380)

[9. Response Codes 22](#_Toc364431381)

[9.1 Standard Response Codes 22](#_Toc364431382)

[9.2 Command Specific Response Codes 22](#_Toc364431383)

[10. Command Details 27](#_Toc364431384)

[10.1 Unpaired Communications 27](#_Toc364431385)

[10.2 Get xPG Status 28](#_Toc364431386)

[10.3 Stimulation 31](#_Toc364431387)

[10.4 Select Program 33](#_Toc364431388)

[10.5 Increment Program Amplitude 34](#_Toc364431389)

[10.6 Decrement Program Amplitude 35](#_Toc364431390)

[10.7 Get PoP Constants 36](#_Toc364431391)

[10.8 Get PPC Constants 37](#_Toc364431392)

[10.9 Get Program Names 39](#_Toc364431393)

[10.10 Set Pulse Amplitude 40](#_Toc364431394)

[10.11 Increment Pulse Amplitude 41](#_Toc364431395)

[10.12 Decrement Pulse Amplitude 42](#_Toc364431396)

[10.13 Get Pulse Widths 43](#_Toc364431397)

[10.14 Set Pulse Width 44](#_Toc364431398)

[10.15 Increment Pulse Width 45](#_Toc364431399)

[10.16 Decrement Pulse Width 46](#_Toc364431400)

[10.17 Get Program Frequency 47](#_Toc364431401)

[10.18 Set Program Frequency 48](#_Toc364431402)

[10.19 Increment Program Frequency 49](#_Toc364431403)

[10.20 Decrement Program Frequency 50](#_Toc364431404)

[10.21 Charging Control 51](#_Toc364431405)

[10.22 Get xPG Identity 52](#_Toc364431406)

[10.23 Set xPG Identity 53](#_Toc364431407)

[10.24 Restore Program Defaults 54](#_Toc364431408)

[10.25 Get Program Definition / Set Program Definition 55](#_Toc364431409)

[10.26 Get Program Constants / Set Program Constants 59](#_Toc364431410)

[10.27 Get Configurable Device Parameters / Set Configurable Device Parameters 61](#_Toc364431411)

[10.28 Get Pulse Constants / Set Pulse Constants 64](#_Toc364431412)

[10.29 Get Pulse Constants 65](#_Toc364431413)

[10.30 Get Lead Limits / Set Lead Limits 68](#_Toc364431414)

[10.31 Get Channel Calibrations / Set Channel Calibrations 70](#_Toc364431415)

[10.32 Get Stim Asic HV Calibrations / Set Stim Asic HV Calibrations 72](#_Toc364431416)

[10.33 Get CP Data / Set CP Data 74](#_Toc364431417)

[10.34 Clear Log / Read Log / Append Log 76](#_Toc364431418)

[10.35 Impedance Measurement 80](#_Toc364431419)

[10.36 Enter Storage Mode 82](#_Toc364431420)

[10.37 Get General Calibration / Set General Calibration 83](#_Toc364431421)

[10.38 Get Trim List / Set Trim List 87](#_Toc364431422)

[10.39 Get XPG Event Counter / Set XPG Event Counter 90](#_Toc364431423)

[10.40 Echo Short 93](#_Toc364431424)

[10.41 Echo Long 94](#_Toc364431425)

[10.42 Undefined 95](#_Toc364431426)

[10.43 Get Test Stimulation / Set Test Stimulation 96](#_Toc364431427)

[10.44 Set Test Waveform 101](#_Toc364431428)

[10.45 Get Titration Stimulation / Set Titration Stimulation 102](#_Toc364431429)

[10.46 Get Ramp Time / Set Ramp Time 105](#_Toc364431430)

[10.47 Get VBAT 107](#_Toc364431431)

[10.48 Get software description block / Set software description block 108](#_Toc364431432)

[10.49 Read Memory / Write Memory 112](#_Toc364431433)

[10.50 Reset xPG 115](#_Toc364431434)

[10.51 Erase flash block 116](#_Toc364431435)

[10.52 Read Interrupt Vectors / Write Interrupt Vectors 117](#_Toc364431436)

[10.53 Diagnostic Data 120](#_Toc364431437)

[10.54 Get Background Impedance Data / Set Background Impedance Parameters 123](#_Toc364431438)

[10.55 Log Range 125](#_Toc364431439)

[10.56 Calibrate Channel 126](#_Toc364431440)

[10.57 Inject Event 128](#_Toc364431441)

[10.58 MICS Options 129](#_Toc364431442)

[10.59 Pair PPC 130](#_Toc364431443)

[10.60 Pair PoP 131](#_Toc364431444)

[10.61 Stop Bootlaoder 132](#_Toc364431445)

[10.62 Tune MICS 133](#_Toc364431446)

[10.63 MICS Diagnostics 134](#_Toc364431447)

[11. Command Support Specification 136](#_Toc364431448)

[11.1 Action Commands and Data Queries 136](#_Toc364431449)

[11.2 Data Query and Retrieves 139](#_Toc364431450)

[Revision History 142](#_Toc364431451)

# Purpose

This document provides the numerical definitions and data byte-order for all MICS Commands and Responses used to communicate between the SCS External Devices and the SCS Pulse Generators.

These commands are used to configure, control and obtain the operating state of the Pulse Generators. The commands operate in a Master-Slave architecture where a Clinician Programmer, Patient Programmer Charger, or Pocket Programmer are Masters that send commands and expect responses from an Implantable or External Pulse Generator which operates as a Slave.

It details the individual command numbering and byte order/data structure, and the data return (response) byte-order/data structure for each command.

# Scope

The scope of this document is limited to the 24-Channel SCS System.

# References

* SWEX 0080 Zarlink MICS Command Protocol Description
* SWEX 0084 SCS System MICS Command Design Document

# Definitions

* + ASIC – Application Specific Integrated Circuit
  + ADC – Analog to digital converter
  + IPG – Implantable Pulse Generator
  + EPG – External Pulse Generator
  + xPG –Either an IPG or EPG. This is only used when the statement applies to both
  + CP – Clinician Programmer
  + PPC – Patient Programmer/Charger
  + PoP – Pocket Programmer
  + Programmer – Clinician Programmer, Pocket Programmer, or Patient Programmer. This is only used when the statement applies to all three
  + EXID – External Device ID
  + MICS – Medical Implant Communication Service
  + Lsbit – The least significant bit in a byte.
  + Lsbyte – The lest significant byte. Only applies to multi-byte data types.

# Global Defined Values

## NUM\_ELECTRODES = 26

The number of supported electrode channels

## NUM\_PULSES\_PROGRAM = 4

The maximum number of pulses allowed per program

## NUM\_PROGRAM\_DEFS = 10

The number of definable patient programs.

## NUM\_PROGRAM\_FREQUENCIES = 64

The number of program frequencies contained in the program frequency table.

## PROGRAM\_NAME\_CHARS = 16

Number of displayable characters for the program name on external device User Interfaces

Note: Number of characters does not include null for string termination.

## PULSE\_NAME\_CHARS = 16

Number of displayable characters for the pulse name on external device User Interfaces

Note: Number of characters does not include null for string termination.

## PPC\_TEXT\_CHARS = 30

Number of displayable characters for 'PPC TEXT' displayed on PPC User Interface.

Note: Number of characters does not include null for string termination.

## MODEL\_NUMBER\_CHARS = 10

Number of characters allotted to represent the model number of an xPG device.

Note: Number of characters does not include null for string termination.

## SERIAL\_NUMBER\_CHARS = 20

Number of characters allotted to represent the serial number of an xPG device

Note: Number of characters does not include null for string termination.

## FIRMWARE\_VERSION\_CHARS = 10

Number of characters allotted to represent the firmware version of an xPG device

Note: Number of characters does not include null for string termination.

## TRUE = 0x01

A value of 0x01 indicates the logical true Boolean condition.

## FALSE = 0x00

A zero value indicates the logical false Boolean condition.

## NUM\_STIM\_ASIC\_HV\_SETTINGS = 64

The number of settings available from the stim asic’s high voltage boost converter.

## NUM\_BYTES\_INST\_DATE\_TIME=8

Number of bytes in the installation date/time array.

# Limits

## Default Amplitude Steps

### DEFAULT\_AMPLITUDE\_STEPS  = 50

Default number of (hardware) output amplitude steps over which amplitudes can be incremented and decremented.

NOTE: This value is the default value for a SCS application configurable xPG operating constant. It can be overwritten with a new value to use thereafter by external devices.

## Pulse Output Amplitude Step Index Limits

### PULSE\_OUTPUT\_AMPLITUDE\_STEP\_MIN = 1

### PULSE\_OUTPUT\_AMPLITUDE\_STEP\_MAX = DEFAULT\_AMPLITUDE\_STEPS

(Hardware) output pulse amplitude step index limits that bound the step index value for which an amplitude output value can be generated in hardware:

Output = Program[Pn].PulseDef[Px].AmplitudeLowLimit + (stepSize \* stepIndex-1)

NOTE: Values are derived from the DEFAULT\_AMPLITUDE\_STEPS xPG operational constant and need to reflect any change in its value.

## Pulse Virtual Amplitude Step Index Limits

### PULSE\_VIRTUAL\_AMPLITUDE\_STEP\_MIN = 2 - DEFAULT\_AMPLITUDE\_STEPS

### PULSE\_VIRTUAL\_AMPLITUDE\_STEP\_MAX = (DEFAULT\_AMPLITUDE\_STEPS \* 2) - 1

Virtual pulse amplitude step index limits that bound the individual virtual pulse step index values maintained by the xPG used in an algorithm to represent an overall program amplitude level.

NOTE: Values are derived from the DEFAULT\_AMPLITUDE\_STEPS xPG operational constant and need to reflect any change in its value.

## Displayed Pulse Amplitude Limits

### PULSE\_DISPLAY\_AMPLITUDE\_STEP\_MIN = 1

### PULSE\_DISPLAY\_AMPLITUDE\_STEP\_MAX = DEFAULT\_AMPLITUDE\_STEPS

Limits for display of individual pulse amplitude step indexes that bound the range of displayed individual pulse amplitude step indexes.

NOTE: Values are derived from the DEFAULT\_AMPLITUDE\_STEPS xPG operational constant and need to reflect any change in its value.

## Displayed Program Amplitude Limits

### PROGRAM\_DISPLAY\_AMPLITUDE\_STEP\_MIN = 1

### PROGRAM\_DISPLAY\_AMPLITUDE\_STEP\_MAX = (DEFAULT\_AMPLITUDE\_STEPS \* 2) - 1

Limits for display of program amplitude that bound the range of the displayed program amplitude

NOTE: Values are derived from the DEFAULT\_AMPLITUDE\_STEPS xPG operational constant and need to reflect any change in its value.

## Pulse Amplitude Limits

### PULSE\_AMPLITUDE\_MIN = 15

### PULSE\_AMPLITUDE\_MAX = 15000

Limits for pulse amplitude in micro-amps

## Pulse Width Limits

### PULSE\_WIDTH\_ MIN = 10

### PULSE\_WIDTH\_MAX = 1500

Limits for pulse widths in microseconds

## Pulse Width Step Limits

### PULSE\_WIDTH\_STEP\_MIN = 1

### PULSE\_WIDTH\_STEP\_MAX = 200

Limits for pulse step width in microseconds

## Program Frequency Index Limits

### PROGRAM\_FREQUENCY\_INDEX\_MIN = 0

### PROGRAM\_FREQUENCY\_INDEX \_MAX = NUM\_PROGRAM\_FREQUENCIES-1

Limits for index into the zero indexed program frequency table

## PoP and PPC EXID Limits

### POP\_PPC\_EXID\_MIN = 0

### POP\_PPC\_EXID\_MAX = 0xFEFFFF

Limits for Pop and PPC EXIDs that an xPG can be paired to

## CP EXID Limits

### CP\_EXID\_MIN = 0xFF0000

### CP\_EXID\_MAX = 0xFFFFFE

Limits for EXIDs used by Clinician Programmers

## Interphase delay Limits

### INTERPHASE\_DELAY\_MIN = 1

### INTERPHASE\_DELAY\_MAX = 255

Limits for delay between stimulus phase and recovery phase of a pulse, in usecs.

## Passive recovery Limits

### PASSIVE\_RECOVERY\_WIDTH\_MIN = 10

### PASSIVE\_RECOVERY\_WIDTH\_MAX = 15000

Limits for width of the passive recovery pulse and the passive recovery charge balance correction pulse, in usecs.

## Active recovery Limits

### ACTIVE\_RECOVERY\_WIDTH\_MIN = 10

### ACTIVE\_RECOVERY\_WIDTH\_MAX = 15000

Limits for width of the active recovery charge balance correction pulse, in usecs.

## Holdoff Limits

### HOLDOFF\_MIN = 1

### HOLDOFF\_MAX = 15000

Limits for delay between the end of a recovery or charge balance correction pulse and the start of the next pulse or delay, in usecs.

## Increment lockout Limits

### INCREMENT\_LOCKOUT\_MIN = 0

### INCREMENT\_LOCKOUT\_MAX = 2000

Limits for length of time to lockout subsequent increment commands after execution of an increment command, in msecs.

## Lead Limit Limits

### MAX\_CURRENT\_DENSITY LIMIT = 15000

The maximum current allowed on a channel in microamperes.

### MAX\_CHARGE\_DENSITY\_LIMIT = 22500

The maximum charge allowed on a channel in mA X μseconds. Set to maximum pulse: 15 mA X 1500 μsec .

## Channel Calibration Limit

### MAX\_CHANNEL\_CAL = 1024

Maximum scaling factor for channel calibrations. (Results in no scaling of 10-bit amplitude register value.)

## Test Pulse Stim Phase Amplitude Limits

### TP\_MIN\_PULSE\_AMPLITUDE = 15

### TP\_MAX\_PULSE\_AMPLITUDE = 30000

Limits for test pulse amplitude in microamps.

## Test Pulse Stim Phase Pulse Width Limits

### TP\_MIN\_STIM\_PHASE\_PULSE\_WIDTH = 10

### TP\_MAX\_STIM\_PHASE\_PULSE\_WIDTH = 1500

Limits for test pulse stimulus phase pulse widths, in microseconds.

## Test Pulse Recovery Ratio Select Limits

### TP\_MIN\_RECOVERY\_RATIO\_SELECT = 0

### TP\_MAX\_RECOVERY\_RATIO\_SELECT = 5

Limits for test pulse recovery ratio. Special case 0 indicates use passive recovery.

## Test Pulse Passive Recovery Width Limits

### TP\_MIN\_PASSIVE\_RECOVERY\_PULSE\_WIDTH = 2

### TP\_MAX\_PASSIVE\_RECOVERY\_PULSE\_WIDTH = 10000

Limits for width of the passive recovery pulse for test pulses, in usecs.

## Test Pulse Charge Balance Correction Width Limits

### TP\_MIN\_CBC\_PULSE\_WIDTH = 2

### TP\_MAX\_CBC\_PULSE\_WIDTH = 10000

Limits for width of the Charge Balance Correction phase pulse for test pulses, in usecs.

## Stim Asic HV Calibration Limits

### MIN\_STIM\_ASIC\_HV\_CAL = 0

### MAX\_STIM\_ASIC\_HV\_CAL = 26000

Limits for the stim asic HV calibration, in millivolts.

## Amplitude Step Size Limits

### MIN\_AMPLITUDE\_STEP = 15

### MAX\_AMPLITUDE\_STEP = 1000

Limits for the pulse definition amplitude step size, in microamperes.

## Replacement Interval Limits

### MIN\_REPLACEMENT\_INTERVAL\_MONTHS = 1

### MAX\_REPLACEMENT\_INTERVAL\_MONTHS = 255

# Common Command Parameter and Returned Data Constructs

## Program Number

Program numbers are not zero indexed.

* P1 = 1
* P2 = 2
* …
* P10 = 10

## Program Map

Bitmap of programs - Bit0 is lsbit of lsbyte.

* bit 0: P1
* bit 1: P2
* …
* bit 9: P10
* bit 10: unused = 0
* bit 11: unused = 0
* bit 12: unused = 0
* …
* bit 15: unused = 0

## Pulse Map

Bitmap of pulse PA thru PD. Bit0 is lsbit.

* bit 0: PA
* bit 1: PB
* bit 2: PC
* bit 3: PD

The only permitted values for a pulse map are 0x01, 0x03, 0x07, and 0x0F.

## Pulse Index

Indexed values of pulse PA thru PD

* 0: PA
* 1: PB
* 2: PC
* 3: PD

## PPC Options Map

Bitmap of PPC configurable options: Bit0 is lsbit.

* bit 0: Allow Individual Pulse Amplitude Modification
* bit 1: Allow Individual Pulse Width Modification
* bit 2: Allow Program Frequency Modification

## Frequency Map

Bitmap of frequencies in program frequency table - Bit0 is lsbit of lsbyte. The following table shows the mapping of bits. Note that PR1 is the first entry in the frequency table.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Bit 0** | **Bit 1** | **Bit 2** | **Bit 3** | **Bit 4** | **Bit 5** | **Bit 6** | **Bit 7** |
| **Byte 1** | PR1 | PR2 | PR3 | PR4 | PR5 | PR6 | PR7 | PR8 |
| **Byte 2** | PR9 | PR10 | PR11 | PR12 | PR13 | PR14 | PR15 | PR16 |
| **Byte 3** | PR17 | PR18 | PR19 | PR20 | PR21 | PR22 | PR23 | PR24 |
| **Byte 4** | PR25 | PR26 | PR27 | PR28 | PR29 | PR30 | PR31 | PR32 |
| **Byte 5** | PR33 | PR34 | PR35 | PR36 | PR37 | PR38 | PR39 | PR40 |
| **Byte 6** | PR41 | PR42 | PR43 | PR44 | PR45 | PR46 | PR47 | PR48 |
| **Byte 7** | PR49 | PR50 | PR51 | PR52 | PR53 | PR54 | PR55 | PR56 |
| **Byte 8** | PR57 | PR58 | PR59 | PR60 | PR61 | PR62 | PR63 | PR64 |

## Magnet Options Map

Bitmap of magnet configurable options: Bit0 is lsbit of lsbyte.

* bit 0: Allows magnet to turn stim on/off and put xPG into storage mode

## Recovery ratio enable Bit Map

Bitmap of recovery ratios that are enabled: Bit0 is lsbit.

* bit 0: KREC 1 enabled
* bit 1: KREC 2 enabled
* bit 2: KREC 3 enabled
* bit 3: KREC 4 enabled
* bit 4: KREC 5 enabled

## IPG Battery Charging State

The IPG Battery Charging State bitmap is the upper nibble of the charging state/IPG temperature status byte. This indicates which state of the charging cycle the IPG battery is in.

* Bit 7: Reserved
* Bit 6: IPG Battery Charging State msbit
* Bit 5: IPG Battery Charging State
* Bit 4: IPG Battery Charging State lsbit

The IPG Battery Charging States are:

* CHARGE\_OFF = 000
* CHARGE\_PRE\_CONDITION = 001
* CHARGE\_CONSTANT\_CURRENT\_FAST =010
* CHARGE\_CONSTANT\_CURRENT\_SLOW = 011
* CHARGE\_COMPLETE = 100
* CHARGE\_ERROR = 101
* CHARGE\_CONSTANT\_VOLTAGE\_FAST = 110
* CHARGE\_CONSTANT\_VOLTAGE\_SLOW = 111

The Warning bit is set when the IPG enclosure temperature is approaching or at the upper safe limit.

## IPG Temperature State

The IPG Temperature State is the lower nibble of the charging state/IPG temperature status byte. Range is 0 to 4, 0 being the lowest temperature.

## Stimulation State

The modes of stimulation are identified by this set of constants.

* STIM\_INACTIVE = 0x00 (stimulation off)
* STIM\_ACTIVE = 0x01 (patient stimulation)
* STIM\_TITRATION = 0x02 (titration stimulation)
* STIM\_TEST = 0x03 (test stimulation)
* STIM\_IMPEDANCE = 0x04 (impedance measurement)

# Command Tokens

The following table lists all the currently defined MICS commands. The Token columns show the command number in decimal and hexadecimal. The Command column provides the command name, and the Tag column lists the name used for that command in the rest of this document.

| **Token** | | **Command** | **Tag** |
| --- | --- | --- | --- |
| 1 | 0x0001 | Get xPG Status | TKN\_GET\_XPG\_STATUS |
| 2 | 0x0002 | xPG Stimulation | TKN\_XPG\_STIMULATION |
| 3 | 0x0003 | Select Program | TKN\_SELECT\_PROGRAM |
| 4 | 0x0004 | Increment Program Amplitude | TKN\_INC\_PRGM\_AMPL |
| 5 | 0x0005 | Decrement Program Amplitude | TKN\_DEC\_PRGM\_AMPL |
| 6 | 0x0006 | Get PoP Constants | TKN\_GET\_POP\_CONST |
| 7 | 0x0007 | Get PPC Constants | TKN\_GET\_PPC CONST |
| 8 | 0x0008 | Get Program Names | TKN\_GET\_PRGM\_NAMES |
| 9 | 0x0009 | Set Pulse Amplitude | TKN\_SET\_PULSE\_AMPL |
| 10 | 0x000A | Increment Pulse Amplitude | TKN\_INC\_PULSE\_AMPL |
| 11 | 0x000B | Decrement Pulse Amplitude | TKN\_DEC\_PULSE\_AMPL |
| 12 | 0x000C | Get Pulse Widths | TKN\_GET\_PULSE\_WIDTHS |
| 13 | 0x000D | Set Pulse Widths | TKN\_SET\_PULSE\_WIDTH |
| 14 | 0x000E | Increment Pulse Width | TKN\_INC\_PULSE\_WIDTH |
| 15 | 0x000F | Decrement Pulse Width | TKN\_DEC\_PULSE\_WIDTH |
| 16 | 0x0010 | Get Program Frequency | TKN\_GET\_PRGM\_FREQ |
| 17 | 0x0011 | Set Program Frequency | TKN\_SET\_PRGM\_FREQ |
| 18 | 0x0012 | Increment Program Frequency | TKN\_INC\_PRGM\_FREQ |
| 19 | 0x0013 | Decrement Program Frequency | TKN\_DEC\_PRGM\_FREQ |
| 20 | 0x0014 | Inject Event | TKN\_INJECT\_EVENT |
| 21 | 0x0015 | Charging Control | TKN\_CHARGING\_CONTROL |
| 22 | 0x0016 | Get xPG Identity | TKN\_GET\_XPG\_IDENTITY |
| 23 | 0x0017 | Set xPG Identity | TKN\_SET\_XPG\_IDENTITY |
| 24 | 0x0018 | Restore Program Defaults | TKN\_RESTORE\_DEFAULTS |
| 25 | 0x0019 | Get Program Definition | TKN\_GET\_PRGM\_DEF |
| 26 | 0x001A | Set Program Definition | TKN\_SET\_PRGM\_DEF |
| 27 | 0x001B | Get Program Constants | TKN\_GET\_PRGM\_CONST |
| 28 | 0x001C | Set Program Constants | TKN\_SET\_PRGM\_CONST |
| 29 | 0x001D | Get Configurable Device Parameters | TKN\_GET\_CONFIG\_PARAMS |
| 30 | 0x001E | Set Configurable Device Parameters | TKN\_SET\_CONFIG\_PARAMS |
| 31 | 0x001F | Get Pulse Constants | TKN\_GET\_PULSE\_CONSTANTS |
| 32 | 0x0020 | Set Pulse Constants | TKN\_SET\_PULSE\_CONSTANTS |
| 33 | 0x0021 | Get Lead Limits | TKN\_GET\_LEAD\_LIMITS |
| 34 | 0x0022 | Set Lead Limits | TKN\_SET\_LEAD\_LIMITS |
| 35 | 0x0023 | Get Channel Calibrations | TKN\_GET\_CHANNEL\_CALS |
| 36 | 0x0024 | Set Channel Calibrations | TKN\_SET\_CHANNEL\_CALS |
| 37 | 0x0025 | (reserved) |  |
| 38 | 0x0026 | (reserved) |  |
| 39 | 0x0027 | (reserved) |  |
| 40 | 0x0028 | (reserved) |  |
| 41 | 0x0029 | (reserved) |  |
| 42 | 0x002A | (reserved) |  |
| 43 | 0x002B | Get Stim Asic HV Calibrations | TKN\_GET\_HV\_CALS |
| 44 | 0x002C | Set Stim Asic HV Calibrations | TKN\_SET\_HV\_CALS |
| 45 | 0x002D | (reserved) |  |
| 46 | 0x002E | (reserved) |  |
| 47 | 0x002F | Get CP Data | TKN\_GET\_CP\_DATA |
| 48 | 0x0030 | Set CP Data | TKN\_SET\_CP\_DATA |
| 49 | 0x0031 | Clear Log | TKN\_CLEAR\_LOG |
| 50 | 0x0032 | Read Log | TKN\_READ\_LOG |
| 51 | 0x0033 | Append Log | TKN\_APPEND\_LOG |
| 52 | 0x0034 | (reserved) |  |
| 53 | 0x0035 | (reserved) |  |
| 54 | 0x0036 | Impedance Measurement | TKN\_IMP\_MEAS |
| 55 | 0x0037 | Enter Storage Mode | TKN\_STORAGE |
| 56 | 0x0038 | Echo Short | TKN\_ECHO\_SHORT |
| 57 | 0x0039 | Echo Long | TKN\_ECHO\_LONG |
| 58 | 0x003A | Get General Calibration | TKN\_GET\_GEN\_CAL |
| 59 | 0x003B | Set General Calibration | TKN\_SET\_GEN\_CAL |
| 60 | 0x003C | Get Trim List | TKN\_GET\_TRIM\_LIST |
| 61 | 0x003D | Set Trim List | TKN\_SET\_TRIM\_LIST |
| 62 | 0x003E | Get Counter | TKN\_GET\_COUNTER |
| 63 | 0x003F | Set Counter | TKN\_SET\_COUNTER |
| 64 | 0x0040 | Get Test Stimulation | TKN\_GET\_TEST\_STIM |
| 65 | 0x0041 | Set Test Stimulation | TKN\_SET\_TEST\_STIM |
| 66 | 0x0042 | Get Titration Stimulation | TKN\_GET\_TITR\_STIM |
| 67 | 0x0043 | Set Titration Stimulation | TKN\_SET\_TITR\_STIM |
| 68 | 0x0044 | Get Ramp Time | TKN\_GET\_RAMP\_TIME |
| 69 | 0x0045 | Set Ramp Time | TKN\_SET\_RAMP\_TIME |
| 70 | 0x0046 | Get VBAT | TKN\_GET\_VBAT |
| 71 | 0x0047 | Set Software Description Block | TKN\_SET\_SDB |
| 72 | 0x0048 | Get Software Description Block | TKN\_GET\_SDB |
| 73 | 0x0049 | Write Memory | TKN\_WRITE\_MEMORY |
| 74 | 0x004A | Read Memory | TKN\_READ\_MEMORY |
| 75 | 0x004B | Reset xPG | TKN\_RESET |
| 76 | 0x004C | Erase Flash Block | TKN\_ERASE\_FLASH\_BLOCK |
| 77 | 0x004D | Write Vectors | TKN\_WRITE\_VECTORS |
| 78 | 0x004E | Read Vectors | TKN\_READ\_VECTORS |
| 79 | 0x004F | Diagnostic Data | TKN\_DIAG\_DATA |
| 80 | 0x0050 | Set Background Impedance | TKN\_SET\_BG\_IMPEDANCE |
| 81 | 0x0051 | Get Background Impedance | TKN\_GET\_BG\_IMPEDANCE |
| 82 | 0x0052 | Set Test Waveform | TKN\_SET\_TEST\_WAVEFORM |
| 83 | 0x0053 | Calibrate Channel | TKN\_CALIBRATE\_CHANNEL |
| 84 | 0x0054 | Get Log Range | TKN\_GET\_LOG\_RANGE |
| 85 | 0x0055 | (reserved) |  |
| 86 | 0x0056 | (reserved) |  |
| 87 | 0x0057 | MICS Options | TKN\_MICS\_OPTIONS |
| 88 | 0x0058 | Pair PPC | TKN\_PAIR\_PPC |
| 89 | 0x0059 | Pair PoP | TKN\_PAIR\_POP |
| 90 | 0x005A | Stop Bootloader | TKN\_STOP\_BOOTLOADER |
| 91 | 0x005B | Tune MICS | TKN\_MICS\_TUNE |
| 92 | 0x005C | MICS Diagnostics | TKN\_MICS\_DIAGNOSTICS |
| 32682 | 0x7FAA | Undefined | TKN\_UNDEFINED |

## Command Revision Compatibility

As generations of the xPGs enter the field, care must be taken to ensure that new commands don’t cause problems with previous versions of the xPG firmware and external device software. The xPG information command described below provides the ability to identify xPG firmware versions so compatibility can be determined.

# Response Codes

## Standard Response Codes

The following responses are returnable by all commands:

### COMMAND\_ACCEPTED = 0xFF

The received command (identified in the response) was processed and performed without error. Any data to be returned as a result of the command completion is present in the response.

### INVALID\_EXID = 0x00

The External ID contained in the command is for an external device the xPG has not been configured to process commands from and either unpaired communication is not allowed for that command or the xPG is not in an acceptable state for unpaired communications.

### INVALID\_COMMAND = 0x01

The Token in the received command is invalid/unknown to the xPG.

### INVALID\_PARAM = 0x02

A parameter provided with the received command has been determined to be invalid and prohibits further processing of the command.

### BUSY = 0x03

The xPG cannot execute the received command while in its present operational state.

## Command Specific Response Codes

The following responses are returnable by one or more commands but not all commands.

### COMMAND NOT VALID WHILE IDLE = 0x10

The command is not valid when stimulation is not active. (This code generally applies to commands that are used to modify stimulation.)

### COMMAND NOT VALID WHILE RAMPING = 0x11

The command is not valid when the xPG is ramping the stimulus amplitude immediately after turning stimulation on. (This code generally applies to commands that are used to modify stimulation parameters)

### COMMAND LOCKED OUT = 0x12

The command is temporarily locked out. (This code generally applies to commands that are used to modify stimulation, resulting in an increase in stimulation energy)

### (reserved) 0x13

### PROGRAM NOT VALID = 0x20

The command cannot be executed because the program definition is not valid.

### PROGRAM DISABLED = 0x21

The command cannot be executed because the program definition has been disabled, due to a hardware issue.

### BATTERY TOO LOW = 0x22

A command to turn stimulation on cannot be executed because the battery level is too low for safe stimulation.

### (reserved) 0x23

### NO PROGRAM SELECTED = 0x24

The requested command cannot be executed because no program is presently selected.

### BOOST CONVERTER STARTUP FAILED = 0x25

Stimulation cannot be started be started because the high voltage boost converter did not initialize correctly.

### PULSE GUARD CHECK FAILED = 0x26

Stimulation cannot be started because the pulse guard circuit check failed.

### STIMULATION POWER ON FAILED = 0x27

Stimulation cannot be started because the stimulation chip failed to startup correctly.

### ACTIVE RECOVERY DISABLED = 0x28

Stimulation cannot be started because automatic waveform adjustment failed.

### (reserved) 0x2C

### (reserved) 0x2D

### STIMULATION SETUP ERROR BAD STIM ASIC READBACK = 0x2E

While attempting to setup the stim asic for output, the readback verification of the register data failed.

### POWER SETUP ERROR BAD POWER ASIC READBACK = 0x2F

While attempting to setup the power asic, the readback verification of the register data failed.

### ALL AMPLITUDES AT MINIMUM = 0x31

The Decrement Program Amplitude command cannot be executed because all pulse amplitudes are the minimum allowed setting.

### ALL AMPLITUDES AT MAXIMUM = 0x32

The Increment Program Amplitude command cannot be executed because all pulse amplitudes are the maximum allowed setting.

### PULSE NOT VALID = 0x33

The command to modify the specified pulse amplitude or pulse width cannot be executed because the pulse in not defined in the program that is running.

### PULSE WIDTH EXCEEDS LIMITS = 0x34

The requested pulse width modification command cannot be executed because it would exceed the programmed pulse width limits for the specified pulse.

### (reserved) 0x35

### MOD VIOLATES ELECTRODE CHARGE DENSITY LIMIT = 0x36

The requested pulse output modification cannot be executed because it would violate the charge density limit for one or more of the selected electrode channels.

### MOD VIOLATES ELECTRODE CURRENT DENSITY LIMIT = 0x37

The requested pulse output modification cannot be executed because it would violate the current density limit for one or more of the selected electrode channels.

### PULSE WIDTH FREQUENCY CONFLICT= 0x38

The requested pulse width or frequency increase cannot be executed because the program cannot be implemented at the specified program frequency.

### (reserved) 0x39

### FREQUENCY MOD NOT ALLOWED = 0x3A

The requested program frequency change is not allowed per the program definition. (This could be because the specified frequency is not allowed or there is no frequency index lower or higher than the present index allowed.)

### PULSE AMPLITUDE EXCEEDS LIMITS = 0x3B

The requested pulse amplitude change cannot be executed because it would exceed the allowed limits for the specified pulse.

### (reserved) 0x3C

### TIMED OUT = 0x40

The operation timed out waiting for the hardware to return a response.

### (reserved) 0x50

### (reserved) 0x51

### (reserved) 0x53

### CHARGING CONTROL ERROR = 0x55

An error occurred while trying to read the temperature or battery data.

### DATA CORRUPTED = 0x60

The requested Set Program Definition command failed, or the requested data was detected as corrupt.

### PROGRAM DEFINITION CORRUPTED = 0x61

The data integrity check for the program definition data for the requested program failed.

### PROGRAM ACTIVE SETTINGS CORRUPTED = 0x62

The data integrity check for the active settings for the requested program failed.

### PROGRAM CONSTANTS CORRUPTED = 0x63

The data integrity check for the program constants data block failed.

### PULSE CONSTANTS CORRUPTED = 0x64

The data integrity check for the pulse constants data block failed.

### CHANNEL CALIBRATIONS CORRUPTED = 0x65

The data integrity check for the channel calibrations data block failed.

### LEAD LIMITS CORRUPTED = 0x66

The data integrity check for the lead limits data block failed.

### STIM ASIC HV CALIBRATIONS CORRUPTED = 0x67

The data integrity check for the stimulation ASIC high-voltage calibrations data block failed.

### RAMP TIME CORRUPTED = 0x68

The data integrity check for the ramp time data block failed.

### (reserved) 0x6A

### FEATURE NOT ENABLED = 0x70

The requested feature or function is not presently enabled.

### (reserved) 0x71

### (reserved) 0x72

### (reserved) 0x73

### (reserved) 0x74

### (reserved) 0x75

### (reserved) 0x76

### WRITE FAILED = 0x77

The command failed while attempting to write data to memory or registers. Writes to flash shall use more specific error codes from the range 0xB0-0xCF described below.

### READ FAILED = 0x78

The command failed while attempting to read data from memory or registers. Reads from flash shall use more specific error codes from the range 0xB0-0xCF described below.

### (reserved) 0x80

### BACKGROUND IMPEDANCE CHECK FAILED = 0x81

The pre-stimulation background impedance check failed.

### BASIC OUTPUT CHECK FAILED = 0x82

The pre-stimulation basic output check failed.

### STIMULATION FAILED TO START UNKNOWN SW ERROR = 0x83

Stimulation failed to start because of an unknown software error.

### STIMULATION FAILED TO START SEVERE ERROR = 0x84

Stimulation failed to start because a severe error exists that has not been cleared.

### Flash Memory Programming Errors = 0xB0-0xCF

The requested programming command failed while attempting to save data to the XPG’s flash memory.

# Command Details

The following sections provide complete details for all the currently defined commands in the SCS System. Elements within a command’s paramaters that are larger than 8 bits are written to the Zarlink MICS chip with the least significant byte first. The parameters are listed in the prescribed (transmit/receive) order.

In order to maintain word addressability on the MSP430 processor, there are reserved bytes in some data structures. Reserved bytes must be set to 0x00 before transmission of a command, and should be ignored in the returned data.

## Unpaired Communications

Some commands are processed under certain circumstances even when the EXID in the command is not recognized by the xPG. This is called Unpaired Communications. The descriptions of the commands below that allow this mode of operation provide details on the restrictions of the use of this mode.

Get xPG Status

Get operational status of an xPG. (This includes data required for the PoP UI display.) This command is allowed on an IPG with unpaired EXIDs when VRECT is present. Unpaired processing is not allowed on an EPG.

Sends

**Token:**

* TKN\_GET\_XPG\_STATUS

**Parameters:**

* NONE

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* No Command Specific Response Codes

**Data:**

**UINT8 ActiveError**

Indicates which error (if any) is preventing normal xPG operation.

* 0 - no error in operation.
* non-0 - number corresponding to error.

**UINT8 xPgGeneralStatusBitMap**

xPG general bit-mapped status byte. Bit Mapped enumeration of general xPG status.

* bit 0: stimulation state 1 = active, 0 = inactive
* bit 1: ramping status 1 = ramping, 0 = not ramping
* bit 2: increment lockout active 1 = lockout active, 0 = lockout not active
* bit 3: unused = 0
* bit 4: unused = 0
* bit 5: unused = 0
* bit 6: unused = 0
* bit 7: bootloader mode 1 = bootloader running, 0 = application running

**UINT16 SelectableProgramMap**

Bitmap indicating if program P1 thru P10 is selectable  
1 = the program is selectable.  
See for details

**UINT8 SelectedProgram**

Number of the currently selected program  
See for details

**UINT8 DefinedPulseMap**

Map indicating if pulse PA thru PD is defined for presently selected program.  
1 = pulse is defined.  
See for details

**INT8 PulseVirtualAmplitudeStepIndex[NUM\_PULSES\_PROGRAM]**

Virtual amplitude step indexes of individual pulses in presently selected program.  
See 6.3 Pulse Virtual Amplitude Step Index Limits for details

**UINT8 xPgBatteryState**

The xPGs present battery state.

* XPG\_BATT\_EMPTY = 0
* XPG\_BATT\_LOW = 1
* XPG\_BATT\_MEDIUM = 2
* XPG\_BATT\_HIGH = 3
* XPG\_BATT\_FULL = 4

**UINT8 ChargingError**

Indicates which error (if any) is preventing normal IPG charging operation.

* 0 - no error in charging.
* non-0 - number corresponding to error condition.

**UINT8 StimState**

The current Stimulation state of the system. See section 7.11, “Stimulation State”.

**UINT8 reserved**

Unused byte, retained to keep the structure length the same as in earlier versions of the protocol. This byte may be used for data in future protocol versions.

xPGs transmit 0 in the reserved byte. External devices ignore the contents of this byte.

Stimulation

Sets the stimulation mode:

* STIM\_INACTIVE turns off stimulation.
* STIM\_ACTIVE runs the currently selected patient program.
* STIM\_TITR enables titration stimulation.
* STIM\_TEST enables test stimulation.
* STIM\_IMPEDANCE enables impedance on demand measurements.

Transitions are permitted both directions between STIM\_INACTIVE and the three states in which stimulation is on. Attempts to transition directly between active-stimulation states are an error. For example, when in STIM\_TITR, requesting a transition directly to STIM\_ACTIVE is an error. Instead, it is necessary to first request a transition to STIM\_INACTIVE, whereupon it is permissible to go to STIM\_ACTIVE.

STIM\_ACTIVE attempts to turn on stimulation immediately. Entering STIM\_TITR or STIM\_TEST enables those modes of stimulation and their commands, but do not begin stimulation.

Sends

**Token:**

* TKN\_XPG\_STIMULATION

**Parameters:**

**UINT8** **StimState**

The requested new stimulation state for the system. See section 7.11, “Stimulation State”.

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* 0x20 PROGRAM NOT VALID
* 0x21 PROGRAM DISABLED
* 0x22 BATTERY TOO LOW
* 0x24 NO PROGRAM SELECTED
* 0x25 BOOST CONVERTER STARTUP FAILURE
* 0x26 PULSE GUARD CHECK FAILED
* 0x27 STIMULATION POWER ON FAILED
* 0x28 ACTIVER RECOVERY DISABLED
* 0x2E STIM SETUP ERROR BAD STIM ASIC READBACK
* 0x40 TIMED OUT
* 0x60 DATA CORRUPTED
* 0x61 PROGRAM DEFINITION CORRUPTED
* 0x62 PROGRAM ACTIVE SETTINGS CORRUPTED
* 0x65 CHANNEL CALIBRATION CORRUPTED
* 0x67 STIM HV CALIBRATION CORRUPTED
* 0x77 WRITE FAILED
* 0x78 READ FAILED
* 0x81 BACKGROUND IMPEDANCE CHECK FAILED
* 0x82 BASIC OUTPUT CHECK FAILED
* 0x83 STIMULATION FAILED UNKNOWN SW ERROR
* 0x84 STIMULATION FAILED SEVERE ERROR

**Data:**

* NONE

Select Program

Select Program to run when stimulation is turned active.

Sends

**Token:**

* TKN\_SELECT\_PROGRAM

**Parameters:**

**UINT8 Program**

Indicates which program number to run when stimulation is active. See for details.

As a special case, a program number of 0 is accepted by this command to indicate “no program selected”.

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* 0x20 PROGRAM NOT VALID
* 0x2E STIM SETUP ERROR BAD STIM ASIC READBACK
* 0x61 PROGRAM DEFINITION CORRUPTED
* 0x62 PROGRAM ACTIVE SETTINGS CORRUPTED

**Data:**

* NONE

Increment Program Amplitude

Increment Program Amplitude and return the virtual pulse step indexes for the defined pulses in the presently selected program.

Sends

**Token:**

* TKN\_INC\_PRGM\_AMPL

**Parameters:**

* NONE

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* 0x10 COMMAND NOT VALID WHILE IDLE
* 0x11 COMMAND NOT VALID WHILE RAMPING
* 0x12 COMMAND LOCKED OUT
* 0x2E STIM SETUP ERROR BAD STIM ASIC READBACK
* 0x32 ALL AMPLITUDES AT MAXIMUM
* 0x36 MOD VIOLATES ELECTRODE CHARGE DENSITY LIMIT
* 0x37 MOD VIOLATES ELECTRODE CURRENT DENSITY LIMIT
* 0x3B PULSE AMPLITUDE EXCEEDS LIMITS
* 0x38 PW FREQUENCY CONFLICT
* 0x77 WRITE FAILED
* 0x78 READ FAILED

**Data:**

**UINT8 DefinedPulseMap**

Bitmap indicating if pulse PA thru PD is defined for presently selected program  
1 = Pulse is defined.  
See for details

**INT8 PulseVirtualAmplitudeStepIndex[NUM\_PULSES\_PROGRAM]**

Virtual amplitude step indexes of individual pulses in presently selected program.  
See 6.3 Pulse Virtual Amplitude Step Index Limits for details

Decrement Program Amplitude

Decrement Program Amplitude and return the virtual pulse step indexes for the defined pulses in the presently selected program.

Sends

**Token:**

* TKN\_DEC\_PRGM\_AMPL

**Parameters:**

* NONE

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* 0x10 COMMAND NOT VALID WHILE IDLE
* 0x11 COMMAND NOT VALID WHILE RAMPING
* 0x2E STIM SETUP ERROR BAD STIM ASIC READBACK
* 0x31 ALL AMPLITUDES AT MINIMUM
* 0x77 WRITE FAILED
* 0x78 READ FAILED

**Data:**

**UINT8 DefinedPulseMap**

Bitmap indicating if pulse PA thru PD is defined for presently selected program  
1 = Pulse is defined.  
See for details

**INT8 PulseVirtualAmplitudeStepIndex[NUM\_PULSES\_PROGRAM]**

Virtual amplitude step indexes of individual pulses in presently selected program.  
See 6.3 Pulse Virtual Amplitude Step Index Limits for details

Get PoP Constants

Get the xPG constants needed by the PoP.

Sends

**Token:**

* TKN\_GET\_POP\_CONST

**Parameters:**

* NONE

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* 0x60 PROGRAM DATA CORRUPTED

**Data:**

**UINT8 AmplitudeSteps**

Number of steps over which the (hardware) output amplitude can be adjusted  
See for details.

**UINT8 Reserved**

* 0x00

**UINT16** **RampTime**

Number of msecs for ramping stimulation output when a stimulation program is started. See 10.46 Get Ramp Time / Set Ramp Timefor further information regarding the ramp time setting.

**UINT16** **IncrementLockoutMsecs**

Number of msecs to lockout subsequent increment commands after executing an increment command. See 10.28 Get Pulse Constants / Set Pulse Constants for further information regarding the increment lockout setting.

Get PPC Constants

Get the xPG constants needed by the PPC.

This command is allowed on an IPG with unpaired EXIDs when VRECT is present. Unpaired processing is not allowed on an EPG.

Sends

**Token:**

* TKN\_GET\_PPC\_CONST

**Parameters:**

* NONE

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* 0x60 PROGRAM DATA CORRUPTED

**Data:**

**UINT8 PPCText[PPC\_TEXT\_CHARS]**

Displayable characters for 'PPC TEXT' displayed on PPC User Interface.  
8 - bit ASCII non-control, non-graphic characters.

**UINT16 PulseWidthStep**

(Global) Step size used when incrementing or decrementing pulse width.

See 6.8 Pulse Width Step Limits for details.

**UINT8 AmplitudeSteps**

Number of steps over which the (hardware) output amplitude can be adjusted  
See 6.1 Default Amplitude Steps for details

**UINT8 PPCOptions**

Configured settings of PPC operation.  
1 = Enabled  
See 7.5 PPC Options Map for details

**UINT16** **RampTime**

Number of msecs for ramping stimulation output when a stimulation program is started. See 10.26 Get Program Constants / Set Program Constants for further information regarding the ramp time setting.

**UINT16** **IncrementLockoutMsecs**

Number of msecs to lockout subsequent increment commands after executing an increment command. See 10.28 Get Pulse Constants / Set Pulse Constants for further information regarding the increment lockout setting.

**UINT8 ImplantDateTime[NUM\_BYTES\_INST\_DATE\_TIME]**

Implantation date / time as Microsoft DateTime data type, 64 bit integer.

**UINT8 ReplacementIntervalMonths**

The replacement interval for the implant, in months.

**UINT8 Reserved**

0x00.

Get Program Names

Get the mapping of which programs are selectable and the names of all selectable programs. (For programs that are not presently selectable, the program name field is filled with space characters.)

Sends

**Token:**

* TKN\_GET\_PRGM\_NAMES

**Parameters:**

* NONE

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* 0x61 PROGRAM DEFINITION CORRUPTED

**Data:**

**UINT16 SelectableProgramMap**

Bitmap indicating if program P1 thru P10 is selectable  
1 = Program is selectable  
See 7.2 Program Map for details

**UINT8 ProgramName [NUM\_PROGRAM\_DEFS][PROGRAM\_NAME\_CHARS]**

Array of program name strings

The program names are concatenated together starting with program 1. Unused characters are set to the space character.

Set Pulse Amplitude

Set the pulse amplitude for the specified pulse of the presently selected program to a specified value and return its virtual pulse step index.

Sends

**Token:**

* TKN\_SET\_PULSE\_AMPL

**Parameters:**

**UINT8** [**PulseIndex**](#AAAAAAAARM)

Indicates which pulse in presently selected program to set amplitude for  
See 7.4 Pulse Index for details.

**UINT8 DisplayedAmplitudeStepIndex**

Displayed Amplitude Step Index for the specified individual pulse  
See 6.4 Displayed Pulse Amplitude Limits for details

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* 0x10 COMMAND NOT VALID WHILE IDLE
* 0x11 COMMAND NOT VALID WHILE RAMPING
* 0x12 COMMAND LOCKED OUT
* 0x2E STIM SETUP ERROR BAD STIM ASIC READBACK
* 0x3B PULSE AMPLITUDE EXCEEDS LIMITS
* 0x33 PULSE NOT VALID
* 0x36 MOD VIOLATES ELECTRODE CHARGE DENSITY LIMIT
* 0x37 MOD VIOLATES ELECTRODE CURRENT DENSITY LIMIT
* 0x38 PW FREQUENCY CONFLICT
* 0x77 WRITE FAILED
* 0x78 READ FAILED

**Data:**

**INT8 PulseAmplitudeStepIndex**

Amplitude step index of the selected pulse in presently selected program. If the command is successful, echoes back the value set in the command. If not, returns the current setting.

Increment Pulse Amplitude

Increment the amplitude for the pulse at the specified index of the presently selected program and return its virtual pulse step index.

Sends

**Token:**

* TKN\_INC\_PULSE\_AMPL

**Parameters:**

**UINT8** [**PulseIndex**](#AAAAAAAARM)

Indicates which pulse in the presently selected program for which to increment the amplitude.  
See 7.4 Pulse Index for details.

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* 0x10 COMMAND NOT VALID WHILE IDLE
* 0x11 COMMAND NOT VALID WHILE RAMPING
* 0x12 COMMAND LOCKED OUT
* 0x2E STIM SETUP ERROR BAD STIM ASIC READBACK
* 0x3B PULSE AMPLITUDE EXCEEDS LIMITS
* 0x33 PULSE NOT VALID
* 0x36 MOD VIOLATES ELECTRODE CHARGE DENSITY LIMIT
* 0x37 MOD VIOLATES ELECTRODE CURRENT DENSITY LIMIT
* 0x38 PW FREQUENCY CONFLICT
* 0x77 WRITE FAILED
* 0x78 READ FAILED

**Data:**

**INT8 PulseVirtualAmplitudeStepIndex**

Virtual amplitude step index of the individual pulse in the presently selected program.  
See 6.3 Pulse Virtual Amplitude Step Index Limits for details

Decrement Pulse Amplitude

Decrement the amplitude for the pulse at the specified index of the presently selected program and return its virtual pulse step index.

Sends

**Token:**

* TKN\_DEC\_PULSE\_AMPL

**Parameters:**

**UINT8** [**PulseIndex**](#AAAAAAAARM)

Indicates which pulse in the presently selected program for which to decrement the amplitude.  
See 7.4 Pulse Index for details.

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* 0x10 COMMAND NOT VALID WHILE IDLE
* 0x11 COMMAND NOT VALID WHILE RAMPING
* 0x2E STIM SETUP ERROR BAD STIM ASIC READBACK
* 0x33 PULSE NOT VALID
* 0x3B PULSE AMPLITUDE EXCEEDS LIMITS
* 0x77 WRITE FAILED
* 0x78 READ FAILED

**Data:**

**INT8 PulseVirtualAmplitudeStepIndex**

Virtual amplitude step index of the individual pulse in the presently selected program.  
See 6.3 Pulse Virtual Amplitude Step Index Limits for details

Get Pulse Widths

Get the individual pulse widths for presently selected program.

Sends

**Token:**

* TKN\_GET\_PULSE\_WIDTHS

**Parameters:**

* NONE

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* 0x24 NO PROGRAM SELECTED
* 0x20 PROGRAM NOT VALID

**Data:**

**UINT8 DefinedPulseMap**

Map indicating if pulse PA thru PD is defined for presently selected program  
1 = Pulse is defined.  
See 7.3 Pulse Map for details.

**UINT8 Reserved**

* 0x00

**UINT16 PulseWidth[NUM\_PULSES\_PROGRAM]**

Width values for each individual pulse in presently selected program  
See 6.7 Pulse Width Limits for details.

Set Pulse Width

Set the pulse width of the pulse at the specified index of the presently selected program to the specified pulse width value.

Sends

**Token:**

* TKN\_SET\_PULSE\_WIDTH

**Parameters:**

**UINT8** [**PulseIndex**](#AAAAAAAARM)

Indicates which pulse in presently selected program to set pulse width for  
See 7.4 Pulse Index for details.

**UINT8 Reserved**

* 0x00

**UINT16 PulseWidth**

Width values for the selected pulse in presently selected program.  
See 6.7 Pulse Width Limits for details.

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* 0x10 COMMAND NOT VALID WHILE IDLE
* 0x11 COMMAND NOT VALID WHILE RAMPING
* 0x12 COMMAND LOCKED OUT
* 0x2E STIM SETUP ERROR BAD STIM ASIC READBACK
* 0x33 PULSE NOT VALID
* 0x34 PULSE WIDTH EXCEEDS LIMITS
* 0x36 MOD VIOLATES ELECTRODE CHARGE DENSITY LIMIT
* 0x38 PW FREQUENCY CONFLICT
* 0x77 WRITE FAILED
* 0x78 READ FAILED

**Data:**

* NONE

Increment Pulse Width

Increment the pulse width for the pulse at the specified index of the presently selected program and return its pulse width value.

Sends

**Token:**

* TKN\_INC\_PULSE\_WIDTH

**Parameters:**

**UINT8** [**PulseIndex**](#AAAAAAAARM)

Indicates which pulse in presently selected program to increase the pulse width of.  
See 7.4 Pulse Index for details.

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* 0x10 COMMAND NOT VALID WHILE IDLE
* 0x11 COMMAND NOT VALID WHILE RAMPING
* 0x12 COMMAND LOCKED OUT
* 0x2E STIM SETUP ERROR BAD STIM ASIC READBACK
* 0x33 PULSE NOT VALID
* 0x34 PULSE WIDTH EXCEEDS LIMITS
* 0x36 MOD VIOLATES ELECTRODE CHARGE DENSITY LIMIT
* 0x38 PW FREQUENCY CONFLICT
* 0x77 WRITE FAILED
* 0x78 READ FAILED

**Data:**

**UINT16 PulseWidth**

Width value for the selected pulse in presently selected program.  
See 6.7 Pulse Width Limits for details.

Decrement Pulse Width

Decrement the pulse width for the pulse at the specified index of the presently selected program and return its pulse width value.

Sends

**Token:**

* TKN\_DEC\_PULSE\_WIDTH

**Parameters:**

**UINT8** [**PulseIndex**](#AAAAAAAARM)

Indicates which pulse in presently selected program to decrease the pulse width of.  
See 7.4 Pulse Index for details.

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* 0x10 COMMAND NOT VALID WHILE IDLE
* 0x11 COMMAND NOT VALID WHILE RAMPING
* 0x2E STIM SETUP ERROR BAD STIM ASIC READBACK
* 0x33 PULSE NOT VALID
* 0x34 PULSE WIDTH EXCEEDS LIMITS
* 0x77 WRITE FAILED
* 0x78 READ FAILED

**Data:**

**UINT16 PulseWidth**

Width value for the selected pulse in presently selected program.  
See 6.7 Pulse Width Limits for details.

Get Program Frequency

Get the index to the Program Frequency Table that identifies the program frequency for the presently selected program.

Sends

**Token:**

* TKN\_GET\_PRGM\_FREQ

**Parameters:**

* NONE

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* 0x24 NO PROGRAM SELECTED

**Data:**

**UINT8 ProgramFrequencyIndex**

Index into the program frequency table, which contains the frequency at which the presently selected program runs  
See 6.9 Program Frequency Index Limits for details

Set Program Frequency

Set the program frequency for the presently selected program by specifying its index in the program frequency table.

Sends

**Token:**

* TKN\_SET\_PRGM\_FREQ

**Parameters:**

**UINT8 ProgramFrequencyIndex**

Index into the program frequency table, which contains the frequency at which the presently selected program runs  
See 6.9 Program Frequency Index Limits for details

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* 0x10 COMMAND NOT VALID WHILE IDLE
* 0x11 COMMAND NOT VALID WHILE RAMPING
* 0x12 COMMAND LOCKED OUT
* 0x2E STIM SETUP ERROR BAD STIM ASIC READBACK
* 0x3A FREQUENCY MOD NOT ALLOWED
* 0x38 PW FREQUENCY CONFLICT
* 0x77 WRITE FAILED
* 0x78 READ FAILED

**Data:**

* NONE

Increment Program Frequency

Increment to the next program frequency in the program frequency table allowed for the presently selected program and returns its index. Skip over any frequencies not allowed for the program per its program definition.

Sends

**Token:**

* TKN\_INC\_PRGM\_FREQ

**Parameters:**

* NONE

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* 0x10 COMMAND NOT VALID WHILE IDLE
* 0x11 COMMAND NOT VALID WHILE RAMPING
* 0x12 COMMAND LOCKED OUT
* 0x2E STIM SETUP ERROR BAD STIM ASIC READBACK
* 0x3A FREQUENCY CHANGE NOT ALLOWED
* 0x38 PW FREQUENCY CONFLICT
* 0x77 WRITE FAILED
* 0x78 READ FAILED

**Data:**

**UINT8 ProgramFrequencyIndex**

Index into the program frequency table, which contains the frequency at which the presently selected program runs  
See 6.9 Program Frequency Index Limits for details

Decrement Program Frequency

Decrement to the next program frequency in the program frequency table allowed for the presently selected program and return its index. Skip over any frequencies not allowed for the program per its program definition.

Sends

**Token:**

* TKN\_DEC\_PRGM\_FREQ

**Parameters:**

* NONE

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* 0x10 COMMAND NOT VALID WHILE IDLE
* 0x11 COMMAND NOT VALID WHILE RAMPING
* 0x2E STIM SETUP ERROR BAD STIM ASIC READBACK
* 0x77 WRITE FAILED
* 0x78 READ FAILED

**Data:**

**UINT8 ProgramFrequencyIndex**

Index into the program frequency table, which contains the frequency at which the presently selected program runs  
See 6.9 Program Frequency Index Limits for details

Charging Control

Read the present battery charging conditions of the IPG including the state in the charging cycle, IPG enclosure temperature, and the received input level.

This command is allowed on an IPG with unpaired EXIDs when VRECT is present. Unpaired processing is not allowed on an EPG.Sends

**Token:**

* TKN\_CHARGING\_CONTROL

**Parameters:**

* NONE

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* 0x55 CHARGING CONTROL ERROR

**Data:**

**UINT8 Charging State/Temperature Level**

Identifies if there is a charging warning, identifies which state of the charging cycle the IPG battery is in, and identifies which temperature range the IPG is in.  
See 7.9 IPG Battery Charging State and 7.10 IPG Temperature State

**UINT8 IPG Received Level**

Received charger output level on the IPG side of the inductive link  
See **Error! Reference source not found.** **Error! Reference source not found.** for details

Get xPG Identity

Get the xPG’s Model Number, Serial Number, and Firmware Version

Sends

**Token:**

* TKN\_GET\_XPG\_IDENTITY

**Parameters:**

* NONE

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* 0x60 DATA CORRUPTED

**Data:**

**UINT8 ModelNumber[MODEL\_NUMBER\_CHARS]**

Alphanumeric identification of the responding xPG device model number (i.e., device type) that utilizes 7 - bit ASCII non-control, non-graphic characters only

**UINT8 SerialNumber[SERIAL\_NUMBER\_CHARS]**

Unique alphanumeric identification of the responding xPG device serial number that utilizes 7 - bit ASCII non-control, non-graphic characters only

**UINT8 FirmwareVersion[FIRMWARE\_VERSION\_CHARS]**

Unique alphanumeric identification of the responding xPG device firmware revision that utilizes 7 - bit ASCII non-control, non-graphic characters only

Set xPG Identity

Set the xPG’s Model Number, Serial Number, and MICS ID

Sends

**Token:**

* TKN\_SET\_XPG\_IDENTITY

**Parameters:**

**UINT8 MICSID[4]**

Unique 4 byte MICS identification of the xPG

* MICSID[0] – Zarlink ZL70102 MICS Transceiver reg\_mac\_imdtransid3
* MICSID[1] – Zarlink ZL70102 MICS Transceiver reg\_mac\_imdtransid2
* MICSID[2] – Zarlink ZL70102 MICS Transceiver reg\_mac\_imdtransid1
* MICSID[3] – Zarlink ZL70102 Company ID

**UINT8 ModelNumber[MODEL\_NUMBER\_CHARS]**

Alphanumeric identification of the responding xPG device model number (i.e., device type)   
8 - bit ASCII non-control, non-graphic characters

**UINT8 SerialNumber[SERIAL\_NUMBER\_CHARS]**

Unique alphanumeric identification of the responding xPG device serial number  
8 - bit ASCII non-control, non-graphic characters

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* (0xB0-0xCF ERROR SAVING TO NON-VOLATILE MEMORY)

**Data:**

* NONE

Restore Program Defaults

Restore patient alterable parameters such as pulse amplitude and width, and program frequency for identified programs back to their original values as set by the clinician in the program definitions.

Sends

**Token:**

* TKN\_RESTORE\_DEFAULTS

**Parameters:**

**UINT16 RestoredProgramMap**

Bitmap indicating if program P1 thru P10 should be restored to its original program definition values.  
Set indicates program should be restored.  
See 7.2 Program Map for details

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* 0x61 PROGRAM DEFINITION CORRUPTED

**Data:**

* NONE

Get Program Definition / Set Program Definition

Get/Set the complete definition of the specified program (including its pulses,) as defined by the clinician.

### Get Program Definition

Sends

**Token:**

* TKN\_GET\_PRGM\_DEF

**Parameters:**

**UINT8 Program**

Number of the program definition.  
See 7.1 Program Number for details

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* 0x61 PROGRAM DEFINITION CORRUPTED

**Data:**

**UINT8 Program**

Number of the program definition.  
See 7.1 Program Number for details

**UINT8 Reserved**

* 0x00

**PROGRAM\_DEFINITION ProgramDefinition**

The detailed program definition which includes pulse definitions  
See 10.25.3 Set/Get Program Definition Common Command Parameter / Response Data

### Set Program Definition

Sends

**Token:**

* TKN\_SET\_PRGM\_DEF

**Parameters:**

**UINT8 Program**

Number of the program definition.  
See 7.1 Program Number for details

**UINT8 Reserved**

* 0x00

**PROGRAM\_DEFINITION ProgramDefinition**

The detailed program definition which includes pulse definitions  
See 10.25.3 Set/Get Program Definition Common Command Parameter / Response Data

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* 0x77 WRITE FAILED

**Data:**

* NONE

### Set/Get Program Definition Common Command Parameter / Response Data

PROGRAM\_DEFINITION

**UINT 8 ProgramDefined**

Boolean indicating that the program has been created  
TRUE or FALSE

**UINT8 ProgramDisabled**

Boolean indicating that the program definition is unusable because of a non-definition issue  
TRUE or FALSE

**UINT8 [PROGRAM\_NAME\_CHARS] ProgramName**

See 5.5 PROGRAM\_NAME\_CHARS = 16 for details

**UINT8 [8] AllowedFrequencies**

Bitmap of frequency table indexes indicating the frequencies allowed for this program.  
See 7.6 Frequency Map for details.

**UINT8 ProgramFrequencyIndex**

Index into the program frequency table identifying the default frequency for the program  
See 6.9 Program Frequency Index Limits for details

**UINT8 DefinedPulseMap**

Bitmap indicating if pulse PA thru PD is defined for presently selected program  
1 = Pulse is defined.  
See 7.3 Pulse Map for details.

**PULSE\_DEFINITION PulseDefinition[NUM\_PULSES\_PROGRAM]**

Individual pulse definitions for this program

PULSE\_DEFINITION

**UINT8 [PULSE\_NAME\_CHARS] PulseName**

See 5.6 PULSE\_NAME\_CHARS = 16 for details

**UINT16 AmplitudeLowLimit**

Lower limit of pulse amplitude for the pulse in microamps  
See 6.6 Pulse Amplitude Limits for details

**UINT16 AmplitudeStep**

Amplitude step size for the pulse in microamps

See 6.25 Amplitude Step Size Limits for details.

**INT16 AmplitudeStepIndex**

Default amplitude step index for pulse  
See 6.2 Pulse Output Amplitude Step Index Limits for details

**UINT16 PulseWidthLowLimit**

Lower limit on pulse width for pulse width in microseconds  
See 6.7 Pulse Width Limits for details

**UINT16 PulseWidthHighLimit**

Upper limit on pulse width for pulse in microseconds  
See 6.7 Pulse Width Limits for details

**UINT16 PulseWidth**

Default pulse width for pulse in microseconds  
See 6.7 Pulse Width Limits for details

**INT8 [NUM\_ELECTRODES] ElectrodeAmpPercentage**

Signed percentage of pulse amplitude proportioned to electrode for the pulse.  
-100 to +100

Get Program Constants / Set Program Constants

Get/Set the contents of the program frequency table.

### Get Program Constants

Sends

**Token:**

* TKN\_GET\_PRGM\_CONST

**Parameters:**

* NONE

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* 0x63PROGRAM CONSTANTS CORRUPTED

**Data:**

PROGRAM\_CONSTANTS

* See 10.26.3 Set/Get Program Constants Common Command Parameter / Response Data

### Set Program Constants

Sends

**Token:**

* TKN\_SET\_PRGM\_CONST

**Parameters:**

PROGRAM\_CONSTANTS

* See 10.26.3 Set/Get Program Constants Common Command Parameter / Response Data

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* (0xB0-0xCF ERROR SAVING TO NON-VOLATILE MEMORY)

**Data:**

* NONE

### Set/Get Program Constants Common Command Parameter / Response Data

PROGRAM\_CONSTANTS

**UINT16 Reserved**

0x00

**UINT16 ProgramFrequencies[NUM\_PROGRAM\_FREQUENCIES]**

Table of frequencies at which program can run in cycles per second (Hz)  
2 (Hz) to 2000 (Hz)

Get Configurable Device Parameters / Set Configurable Device Parameters

Get/Set configurable device parameters such as EXIDs, PPC Text, Pulse Width Steps, Amplitude Steps, PPC Options, Magnet Options, Implant Date, and Replacement Interval.

### Get Configurable Device Parameters

Sends

**Token:**

* TKN\_GET\_CONFIG\_PARAMS

**Parameters :**

* NONE

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* 0x60 DATA CORRUPTED

**Data:**

CONFIGURABLE\_DEVICE\_PARAMETERS  
See 10.27.3 Set/Get Configurable Device Parameters Common Command Parameter / Response Data

### Set Configurable Device Parameters

Sends

**Token:**

* TKN\_SET\_CONFIG\_PARAMS

**Parameters:**

CONFIGURABLE\_DEVICE\_PARAMETERS  
See 10.27.3 Set/Get Configurable Device Parameters Common Command Parameter / Response Data

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* 0x77 WRITE FAILED

**Data:**

* NONE

### Set/Get Configurable Device Parameters Common Command Parameter / Response Data

**CONFIGURABLE\_DEVICE\_PARAMETERS**

**UINT8 PPCEnabled**

Indicates whether PPC is enabled for use and its assigned PPC EXID used to validate received commands.  
TRUE or FALSE

**UINT8[3] PPCEXID**

EXID of PPC paired with xPG  
See 6.10 PoP and PPC EXID Limits for details

* PPCEXID[0] – least significant byte
* PPCEXID[1] – middle byte
* PPCEXID[2] – most significant byte

**UINT8 Pop1Enabled**

Indicates whether Pop1 is enabled for use and its assigned Pop1 EXID used to validate received commands.  
TRUE or FALSE

**UINT8[3] Pop1EXID**

EXID of Pop1 paired with xPG  
See 6.10 PoP and PPC EXID Limits for details

* Pop1Exid[0] – least significant byte
* Pop1Exid[1] – middle byte
* Pop1Exid[2] – most significant byte

**UINT8 Pop2Enabled**

Indicates whether Pop2 is enabled for use and its assigned Pop2 EXID used to validate received commands.  
TRUE or FALSE

**UINT8[3] Pop2EXID**

EXID of Pop2 paired with xPG.  
See 6.10 PoP and PPC EXID Limits for details

* Pop2Exid[0] – least significant byte
* Pop2Exid[1] – middle byte
* Pop2Exid[2] – most significant byte

**UINT8 PPCText[PPC\_TEXT\_CHARS]**

Displayable characters for 'PPC TEXT' displayed on PPC User Interface.  
8 - bit ASCII non-control, non-graphic characters

**UINT16 PulseWidthStep**

(Global) Step size used when incrementing or decrementing pulse width.  
See 6.8 Pulse Width Step Limits for details

**UINT8 AmplitudeSteps**

Number of steps over which the (hardware) output amplitude can be adjusted  
See 6.1 Default Amplitude Steps for details

**UINT8 PPCOptions**

Configured settings of PPC operation   
See 7.5 PPC Options Map for details

**UINT8 IPGMagnetOptions**

Bit map of IPG magnet options.  
Bit assignment, see 7.7 Magnet Options Map.

**UINT8 Reserved**

* 0x00

**UINT8 ImplantDateTime[NUM\_BYTES\_INST\_DATE\_TIME]**

Implantation date / time expressed as the number of 100-nanosecond “ticks” since 12:00 AM January 1st 1601 UTC.

**UINT8 ReplacementIntervalMonths**

The replacement interval for the implant expressed in months.  
See 6.26 Replacement Interval Limits for details

**UINT8 Reserved2**

* 0x00

## Get Pulse Constants / Set Pulse Constants

Get/Set pulse-related constants applicable to all programs.

Get Pulse Constants

Sends

**Token:**

* TKN\_GET\_PULSE\_CONSTANTS

**Parameters :**

* NONE

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* 0x64 PULSE CONSTANTS CORRUPTED

**Data:**

PULSE\_CONSTANTS  
See 10.29.2 Set/Get Pulse Constants Common Command Parameter / Response Data

### Set Pulse Constants

Sends

**Token:**

* TKN\_SET\_PULSE\_CONSTANTS

**Parameters:**

PULSE\_CONSTANTS  
See 10.29.2 Set/Get Pulse Constants Common Command Parameter / Response Data

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* (0xB0-0xCF ERROR SAVING TO NON-VOLATILE MEMORY)

**Data:**

* NONE

### Set/Get Pulse Constants Common Command Parameter / Response Data

**PULSE\_CONSTANTS**

**UINT16 Reserved**

**UINT16 PassiveRecoveryWidth**

Width of passive recovery pulse, in usecs.

See 6.13 for limits.

**UINT16 PassiveRecoveryHoldoff**

Duration of holdoff between end of passive recovery pulse and start of next pulse stim phase or next delay, in usecs.

See 6.15 Holdoff Limits for limits.

**UINT16 PassiveCBCWidth**

Width of charge balance correction pulse, when using passive recovery, in usecs.

See 6.13 for limits.

**UINT16 PassiveCBCHoldoff**

Duration of holdoff between end of passive CBC pulse and start of next pulse or delay, in usecs.

See 6.15 Holdoff Limits for limits.

**UINT16 Reserved**

**UINT16 ActiveRecoveryHoldoff**

Duration of holdoff between end of active recovery pulse and start of next pulse stim phase or next delay, in usecs.

See 6.15 Holdoff Limits for limits.

**UINT16 ActiveCBCWidth**

Width of charge balance correction pulse, when using active recovery, in usecs.

See 6.14 Active recovery Limits for limits.

**UINT16 ActiveCBCHoldoff**

Duration of holdoff between end of active CBC pulse and start of next pulse or delay, in usecs.

See Holdoff Limits for limits.

**UINT16 KrecsEnabledBitMap**

Indication of recovery ratios that are allowed for active recovery.

See 7.8 Recovery ratio enable Bit Map for format.

**UINT16 IncrementLockoutMsecs**

Number of msecs to lockout subsequent increment commands after executing an increment command.

See 6.16 for limits.

**UINT8 StimPhasePSDisableActive**

Indicates whether the feature to disable stimulation power supply is active.

TRUE or FALSE

**UINT8 UncontrolledSourceSinkEnabled**

Indicates whether the feature to implement uncontrolled sourcing or sinking is enabled.

TRUE or FALSE

Get Lead Limits / Set Lead Limits

Get/Set current density and charge density limits for the 26 electrode channels, per the installed electrode configuration.

### Get Lead Limits

Sends

**Token:**

* TKN\_GET\_LEAD\_LIMITS

**Parameters :**

* NONE

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* 0x66 LEAD LIMITS CORRUPTED

**Data:**

LEAD\_LIMITS  
See 10.30.3 Set/Get Lead Limits Common Command Parameter / Response Data

### Set Lead Limits

Sends

**Token:**

* TKN\_SET\_LEAD\_LIMITS

**Parameters:**

LEAD\_LIMITS  
See 10.30.3 Set/Get Lead Limits Common Command Parameter / Response Data

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* 0x77 WRITE FAILED

**Data:**

* NONE

### Set/Get Lead Limits Common Command Parameter / Response Data

**LEAD\_LIMITS**

**UINT16 [NUM\_ELECTRODES] currentLimit**

The current limit for the channel in microamperes. (For a typical electrode, the current density limit can be multiplied by the electrode’s surface area, to provide a simple current limit.) (Note that a setting of 0 disables the channel.)

For parameter limits see 6.17 Lead Limit Limits.

**UINT16 [NUM\_ELECTRODES] chargeLimit**

The charge limit for the channel in units of milliampere X microseconds. (mA X μsec). (For a typical electrode, the charge density limit can be multiplied by the electrode’s surface area, to provide a simple charge limit. Note that mA is used to ensure the value will fit into a two-byte integer.)

For parameter limits see 6.17 Lead Limit Limits.

Get Channel Calibrations / Set Channel Calibrations

Get/Set channel calibration values for the 26 electrode channels.

### Get Channel Calibrations

Sends

**Token:**

* TKN\_GET\_CHANNEL\_CALS

**Parameters :**

**UINT16 index**

* There are 4 channel calibration tables that are stored in the IPG. These tables correspond to stimulation amplitude ranges. The index is 0 based, so the possible index values are 0, 1, 2, and 3.

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* 0x65 CHANNEL CALIBRATIONS CORRUPTED

**Data:**

CHANNEL\_CALIBRATIONS  
See 10.31.3 Set/Get Channel Calibrations Common Command Parameter / Response Data.

### Set Channel Calibrations

Sends

**Token:**

* TKN\_SET\_CHANNEL\_CALS

**Parameters:**

CHANNEL\_CALIBRATIONS  
See 10.31.3 Set/Get Channel Calibrations Common Command Parameter / Response Data

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* (0xB0-0xCF ERROR SAVING TO NON-VOLATILE MEMORY)

**Data:**

* NONE

### Set/Get Channel Calibrations Common Command Parameter / Response Data

**CHANNEL\_CALIBRATIONS**

**UINT16 index**

There are 4 channel calibration tables that are stored in the IPG. These tables correspond to stimulation amplitude ranges. The index is 0 based, so the possible index values are 0, 1, 2, and 3.

**UINT16 [NUM\_ELECTRODES] channelCalSourcing**

Scaling factor applied to channel amplitude register to achieve correct output current when configured as a controlled current source.

For parameter limits see 6.18 Channel Calibration Limit.

**UINT16 [NUM\_ELECTRODES] channelCalSinking**

Scaling factor applied to channel amplitude register to achieve correct output current when configured as a controlled current sink.

For parameter limits see 6.18 Channel Calibration Limit.

Get Stim Asic HV Calibrations / Set Stim Asic HV Calibrations

Get/Set stim asic high voltage calibration value for the range of settings of the stim asic’s boost converter.

### Get Stim Asic HV Calibrations

Sends

**Token:**

* TKN\_GET\_ HV\_CALS

**Parameters :**

* NONE

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* 0x67 STIM ASIC HV CALIBRATIONS CORRUPTED

**Data:**

STIM\_ASIC\_HV\_CALIBRATIONS

See 10.32.3 Set/Get Stim Asic HV Calibrations Common Command Parameter / Response Data

### Set Stim Asic HV Calibrations

Sends

**Token:**

* TKN\_SET\_ HV\_CALS

**Parameters:**

STIM\_ASIC\_HV\_CALIBRATIONS  
See 10.32.3 Set/Get Stim Asic HV Calibrations Common Command Parameter / Response Data.

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* (0xB0-0xCF ERROR SAVING TO NON-VOLATILE MEMORY)

**Data:**

* NONE

### Set/Get Stim Asic HV Calibrations Common Command Parameter / Response Data

**STIM\_ASIC\_HV\_CALIBRATIONS**

**UINT16 [NUM\_STIM\_ASIC\_HV\_SETTINGS] stimAsicHVmVOutput**

The calibrated output, in millivolts, for the indexed setting of the stim asic’s high voltage boost converter.

For calibration limits see 6.24 Stim Asic HV Calibration Limits.

Get CP Data / Set CP Data

Get/Set the requested CP data block. The XPG allocates 40 256-byte blocks of storage in non-volatile memory for the CP to store configuration and historical case data. The XPG does not use or evaluate this data, nor does it check for corruption when stored or retrieved.

### Get CP Data Block

Sends

**Token:** TKN\_GET\_CP\_DATA

**Parameters:**

**UINT8 CPDataBlockSelect**

Number of the CP Data Block to get. (0-39)

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* 0x78 READ FAILED

**Data:**

**UINT8 CPDataBlockSelect**

Number of the CP Data Block returned

**UINT8 Reserved**

* 0x00

**UINT8 CPDataBlockBytes[256]**

Opaque binary data

### Set CP Data Block

Sends

**Token:** TKN\_SET\_CP\_DATA

**Parameters:**

**UINT8 CPDataBlockSelect**

* Number of the CP Data Block to set. (0-39)

**UINT8 Reserved**

* 0x00

**UINT8 CPDataBlockBytes[256]**

Opaque binary data

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* 0x77 WRITE FAILED

**Data:** NONE

Clear Log / Read Log / Append Log

Read or add to the xPG event logs. The xPG maintains two event logs, representing two levels of expected frequency of events.

Each log assigns each of its events a serial number. The serial number for an event remains unchanged for as long the event remains in the log. Serial numbers are 32 bits long. All of the events in the log at a given time have sequential serial numbers. Serial numbers are unique within a given log, but may be duplicated in different logs.

The read command is idempotent.

The xPG may respond to a read command with any number of entries from 1 to the maximum. If the read command requests a serial number that is less than or equal to the serial number of the first record in the log, the first record in the log shall be the first record returned. If serial number requested exists in the log, that record shall be the first record returned.  If the serial number requested is greater than the largest serial number in the log, no records shall be returned and Count will equal zero.

The append command allows the CP to add an entry to the log. This can be used, for example, to simulate log clearing by marking the point up to which all log entries have been reviewed. It may also be useful in xPG firmware development.

### Clear Log

This command clears all event entries in the given log by writing zeros over each memory location. This ensures that the log entry data is erased. The serial number for the log is reset to 0 and the log contains 0 log entries after this command is executed.

Sends

**Token:** TKN\_CLEAR\_LOG

**Parameters:**

**UINT8 LogNum**

Number of the Event Log.

0 = Major Event Log

1 = Normal Event Log

Returns

**Response Codes:**

* 9.1 Standard Response Codes

### Read Log

Retrieves log data starting with the specified serial number. At most 42 log entries are returned. If there are further entries in the log, they are not returned. They must be retrieved with a subsequent Read Log command.

Sends

**Token:** TKN\_READ\_LOG

**Parameters:**

**UINT8 LogNum**

Number of the Event Log.

0 = Major Event Log

1 = Normal Event Log

**UINT8 Reserved**

Always zero.

**UINT32 Serial**

Serial number of the first event to return

Returns

**Response Codes:**

* 9.1 Standard Response Codes

**Data:**

**UINT8 LogNum**

Number of the Event Log returned.

**UINT8 Count**

Number of log entries returned (0-42)

**UINT16 reserved**

**UINT32 Serial**

Serial number of the first log entry

**LOG\_ENTRY Log[42]**

See 10.34.4 Log Entry Structure

### Append Log

Sends

**Token:** TKN\_APPEND\_LOG

**Parameters:**

**UINT8 LogNum**

Identifier of the Event Log to set.

**UINT8 Reserved**

Always zero.

**UINT16 Event**

Event identifier.

**UINT8 EventData[4]**

Supplemental data describing the event.

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* 0x77 WRITE FAILED

**Data:**

**UINT32 Serial**

Serial number of the new log entry

### Log Entry Structure

**UINT32 Timestamp**

Time the event occurred in seconds since xPG last left Storage Mode.

**UINT16 Event**

Event identifier.

**UINT8 EventData[4]**

Supplemental data describing the event.

Impedance Measurement

Instructs the XPG to measures the impedance between the two selected electrode channels per the specified parameters.

(If the parameters other than channel\_A or channel\_B are all 0, the XPG uses the standard ‘on-demand’ impedance measurement parameters.)

Returns the requested impedance measurement.

This command is valid only when in the STIM\_IMPEDANCE stimulation state which is enabled and disabled by the stimulation command.

Sends

**Token:** TKN\_IMP\_MEAS

**Parameters:**

**UINT8 channel\_A**

Electrode channel for measurement (1-26)

**UINT8 channel\_B**

Electrode channel for measurement (1-26)

**UINT8 pulseAmplitude**

**0x01 Low amplitude pulse (200 uAmp)**

**0x02 High amplitude pulse (500 uAmp)**

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* 0x22 BATTERY TOO LOW
* 0x36 MOD VIOLATES CHARGE DENSITY LIMIT
* 0x37 MOD VIOLATES CURRENT DENSITY LIMIT
* 0x3B PULSE AMPLITUDE EXCEEDS LIMITS
* 0x70 FEATURE NOT ENABLED

**Data:**

**UINT8 channel\_A**

Electrode channel for measurement (1-26, 0 for no connection, or 27 for internal voltage reference)

**UINT8 channel\_B**

Electrode channel for measurement (1-26, 0 for no connection, or 27 for internal voltage reference)

**UINT16 impedanceReading**

Impedance reading for the selected channel pair, in ohms.

Enter Storage Mode

Instructs the IPG to power itself down and put itself into storage mode.

Sends

**Token:** TKN\_ STORAGE

**Parameters:**

**None.**

Returns

**Response Codes:**

* 9.1 Standard Response Codes

**Data:** NONE

Get General Calibration / Set General Calibration

Get/Set the group of calibration parameters.

### Get General Calibration

Sends

**Token:** TKN\_GET\_GEN\_CAL

**Parameters:** NONE

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* 0x60 DATA CORRUPTED

**Data:**

**CALIBRATION DATA**

**See**  .

### Set General Calibration

Sends

**Token:** TKN\_SET\_GEN\_CAL

**Parameters:**

**CALIBRATION DATA**

**See**  .

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* (0xB0-0xCF ERROR SAVING TO NON-VOLATILE MEMORY)

**Data:** NONE

### Temperature Measurement

The following diagram shows the basic approach for measuring the resistance of a thermistor for the purpose of determining the temperature of the IPG.



The resistance of the thermistor (RTHERM) can be determined using the relative voltages between THERM\_BIAS, THERM\_INPUT and THERM\_OFFSET and given a known resistance RREF with the following equation:

The value of RTHERM in Ohms is too large to fit in a 16-bit integer for the range that we wish to read, so it has been scaled by 156/47000. This is a unit we call Scaled RTHERM

### Get/Set General Calibration Common data

**UINT16 Reserved[15]**

**UINT16 temperatureMinimumThreshold**

Minimal threshold to allow charging. Expressed in scaled RTHERM.

**UINT16 temperatureWarningThreshold**

Temperature at which a warning is indicated. Expressed in scaled RTHERM.

**UINT16 temperatureCriticalThreshold**

Temperature at which a critical temperature level is indicated and charging is slowed. Expressed in scaled RTHERM.

**UINT16 temperatureAbortThreshold**

Temperature above which charging is not allowed. Expressed in scaled RTHERM.

**UINT16 thermOffsetMinimum**

THERM\_OFFSET readings below this value result in a thermistor error. Expressed in ADC units.

**UINT16 thermOffsetMaximum**

THERM\_OFFSET readings above this value result in a thermistor error. Expressed in ADC units.

**UINT16 thermBiasMinimum**

THERM\_BIAS readings below this value result in a thermistor error. Expressed in ADC units.

**UINT16 thermBiasMaximum**

THERM\_BIAS readings above this value result in a thermistor error. Expressed in ADC units.

**UINT16 thermInputMinimum**

THERM\_INTPUT readings below this value result in a thermistor error. Expressed in ADC units.

**UINT16 thermInputMaximum**

THERM\_INPUT readings above this value result in a thermistor error. Expressed in ADC units.

**UINT16 batteryIncreasePreCharge**

Increase in battery voltage required within each 2-minute window while in the pre-charging state. Expressed in ADC counts.

**UINT16 batteryIncreaseConstantCurrent**

Increase in battery voltage required within each 2-minute window while in the constant current state. Expressed in ADC counts.

**UINT16 Reserved[15]**

**UINT16 batteryStateCalStimOff[6]**

Threshold values that determine battery states when stimulation is off. Expressed in ADC units.

**UINT16 batteryStateCalStimOn[6]**

Threshold values that determine battery states when stimulation is on. Expressed in ADC units.

**UINT16 RTarget**

The value of RTarget to be used when performing stimulation calculations.

Get Trim List / Set Trim List

Get/Set a list of trim/tune/calibration values to write to an ASIC’s registers.

### Get Trim List

Sends

**Token:** TKN\_GET\_TRIM\_LIST

**Parameters:**

**UINT8 ListNum**

Select the trim list to read:

TRIM\_PLUTO 0x00 Pluto (power ASIC)

TRIM\_ZL 0x01 ZL7010x (radio)

TRIM\_SATURN 0x02 Saturn (stim ASIC)

TRIM\_ZL\_FACTORY 0x03 ZL7010x (radio) factory trim list

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* 0x60 DATA CORRUPTED

**Data (TRIM\_PLUTO and TRIM\_ZL):**

**UINT 8 ListNum**

The trim list number that was read.

**UINT 8 ListLen**

Number of entries in this trim list (0-32).

**TRIM\_VAL trims[32]**

**Data (TRIM\_SATURN):**

**UINT 8 ListNum**

The trim list number that was read.

**UINT 8 ListLen**

Number of entries in this trim list (0-32).

**UINT 8 STIM\_BG\_TRIM**

The stimulation BG trim value.

**UINT 8 STIM\_PGO\_TRIM**

The stimulation PGO trim value.

**UINT 8 STIM\_CASN\_TRIM**

The stimulation CASN trim value.

**UINT 8 STIM\_CASP\_TRIM**

The stimulation CASP trim value.

**UINT 8 STIM\_HVDDL\_TRIM**

The stimulation HVDDL trim value.

### Set Trim List

Sends

**Token:** TKN\_SET\_TRIM\_LIST

**Parameters:**

**UINT 8 ListNum**

The trim list number that was read.

**UINT 8 ListLen**

Number of entries in this trim list (0-32).

**TRIM\_VAL trims[32]**

**See 10.38.3** Get/Set Trim List Common data**.**

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* (0xB0-0xCF ERROR SAVING TO NON-VOLATILE MEMORY)

**Data:** NONE

### Get/Set Trim List Common data

Trim list data is sent and received byte-aligned, not word aligned. Take care to make each TRIM\_VAL three bytes long and do not insert a padding byte between each TRIM\_VAL.

TRIM\_VAL:

**UINT16 Address**

Address, within the corresponding ASIC’s address space, of the register to which to write.

**UINT8 Data**

Value to be written.

Get XPG Event Counter / Set XPG Event Counter

The XPG maintains a number of event counters. Examples include:

* Number of times charging has gone from constant current to constant voltage state
* Number of times charge has reached the end of constant voltage state or charge complete state
* Number of times the system has come out of storage mode

These commands allow and external controller to determine how many times a specific event has occurred, and to set (normally reset to 0) the counters.

A counter will not roll over if it achieves the maximum count. Rather, the value must be reset to 0 explicitly by an external.

### Get Counter

Gets the current count for the selected event counter

Sends

**Token:** TKN\_GET\_COUNTER

**Parameters:**

**UINT8 eventCounterID**

|  |  |  |
| --- | --- | --- |
| **Table of Counter IDs** | | |
| **ID** | **Enum** | **Description** |
| 0 | COUNTER\_X\_TEST | Used for testing only. |
| 1 | COUNTER\_PRECHARGE\_COMPLETE | Number of times charging has gone from constant current to constant voltage state |
| 2 | COUNTER\_CHARGE\_COMPLETE | Number of times charging has reached the end of constant voltage state or charge complete stage |
| 3 | COUNTER\_STORAGE\_MODE | Number of times the system has come out of storage mode |
| 4 | COUNTER\_RUN\_TIME\_SECS\_PRG\_1 | The Cumulative Run-time of Program 1 |
| 5 | COUNTER\_RUN\_TIME\_SECS\_PRG\_2 | The Cumulative Run-time of Program 2 |
| 6 | COUNTER\_RUN\_TIME\_SECS\_PRG\_3 | The Cumulative Run-time of Program 3 |
| 7 | COUNTER\_RUN\_TIME\_SECS\_PRG\_4 | The Cumulative Run-time of Program 4 |
| 8 | COUNTER\_RUN\_TIME\_SECS\_PRG\_5 | The Cumulative Run-time of Program 5 |
| 9 | COUNTER\_RUN\_TIME\_SECS\_PRG\_6 | The Cumulative Run-time of Program 6 |
| 10 | COUNTER\_RUN\_TIME\_SECS\_PRG\_7 | The Cumulative Run-time of Program 7 |
| 11 | COUNTER\_RUN\_TIME\_SECS\_PRG\_8 | The Cumulative Run-time of Program 8 |
| 12 | COUNTER\_RUN\_TIME\_SECS\_PRG\_9 | The Cumulative Run-time of Program 9 |
| 13 | COUNTER\_RUN\_TIME\_SECS\_PRG\_10 | The Cumulative Run-time of Program 10 |
| 14 | COUNTER\_BATTERY\_DEPLETION | The Counter for number of times battery is depleted |
| 15 | COUNTER\_EVENT\_QUEUE\_MAX\_COUNT | Used for testing only.  The Counter for diagnostic purposes to see how full the event queue is getting |

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* 0x2F POWER ASIC READBACK FAILED

**Data:**

**UINT8 eventCounterID**

Echo of event counter ID

**UINT8 reserved (0x00)**

**UINT32 eventCount**

Current count of selected event counter.

### Set Counter

Sets the current count for the selected event counter.

Sends

**Token:** TKN\_SET\_COUNTER

**Parameters:**

**UINT8 eventCounterID**

**UINT8 reserved (0x00)**

**UINT32 eventCount**

New count for selected event counter.

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* 0x2F POWER ASIC READBACK FAILED

**Data:** NONE

Echo Short

An ech**o** command that uses a single MICS data block. The data sent in the command is returned verbatim in the response.

Sends

**Token:**

* TKN\_ECHO\_SHORT

**Parameters:**

**UINT8 data[8]**

Returns

**Response Codes:**

* 9.1 Standard Response Codes

**Data:**

**UINT8 data[8]**

Echo Long

An echo command that uses multiple MICS data blocks. The data payload in the command is returned verbatim in the response

Sends

**Token:**

* TKN\_ECHO\_LONG

**Parameters:**

**UINT8 data[256]**

Returns

**Response Codes:**

* 9.1 Standard Response Codes

**Data:**

**UINT8 data[256]**

Undefined

This command is reserved as an explicitly undefined command token. Having a token reserved for an unimplemented command is useful in XPG software testing and development.

The XPG’s handling of TKN\_UNDEFINED is identical to any other command token left undefined by this specification.

Sends

**Token:**

* TKN\_UNDEFINED

**Parameters:**

**none**

Returns

**Response Codes:**

* 9.1 Standard Response Codes

**Data:**

**none**

Get Test Stimulation / Set Test Stimulation

Get/Set the settings for test stimulation.

Test stimulation, once enabled with the Stimulation command, does not start until a Set Test Stimulation command containing a definition of the stimulation to be generated. When a valid Set Test Stimulation command is received by the xPG, the requested stimulation begins immediately.

If test stimulation is already being generated when a Set Test Stimulation command arrives, the xPG switches from the old parameters to the new cleanly at the end of a program cycle.

Rectangular waveforms are generated by default. Flag values in each pulse can select alternate waveforms from the two waveform RAMs for the stimulation and/or recovery phase. Use the Set Test Waveform command to load the waveform RAMs prior to beginning test stimulation.

This command is valid only when in the STIM\_TEST stimulation state.

### Get Test Stimulation

Sends

**Token:**

* TKN\_GET\_TEST\_STIM

**Parameters:**

none

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* 0x70 FEATURE NOT ENABLED

**Data:**

**TEST\_STIM\_DEFINITION**

The test stimulation definition data block.

See 10.43.3, “Set/Get Test ”.

### Set Test Stimulation

Starts or changes the test stimulation parameters. When test stimulation is already active, the change takes place at the end of a program cycle.

This command is valid only when in the STIM\_TEST stimulation state.

**Token:**

* TKN\_SET\_TEST\_STIM

**Parameters:**

**TEST\_STIM\_DEFINITION**

The test stimulation definition data block.

See 10.43.3, “Set/Get Test ”.

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* 0x36 MOD VIOLATES CHARGE DENSITY LIMIT
* 0x37 MOD VIOLATES CURRENT DENSITY LIMIT
* 0x3B PULSE AMPLITUDE EXCEEDS LIMITS
* 0x70 FEATURE NOT ENABLED

**Data:**

* NONE

### Set/Get Test Stimulation Common Data

TEST\_STIM\_DEFINITION

**UINT8 FrequencyIndex**

Index into the program frequency table identifying the desired frequency for the test pulse. Limits are the same as for normal program definition, see 6.9 Program Frequency Index Limits.

**UINT8 ComplianceVoltage**

The stim ASIC boost converter voltage setting, in the DAC units used by the stim ASIC. See EESP 0085 *Stim ASIC Specifications.*

**UINT8 NumPulses**

The number of active pulse definitions. This number of pulses are generated, starting from the first definition.

NumPulses = 0 turns off test stimulation without leaving the STIM\_TEST Stimulation State.

**UINT8 reserved**

Reserved for future use. Must be set to 0 on transmit and ignored on receive.

**TEST\_PULSE Pulse[NUM\_PULSES\_PROGRAM]**

Pulse definitions for the test stimulation. See below. Pulse definitions beyond the number in NumPulses are ignored.

TEST\_PULSE

**UINT16 Amplitude**

Test pulse stimulus phase amplitude in microamps. See 6.6 Pulse Amplitude Limits for limits.

**INT8 [NUM\_ELECTRODES] ChannelPercentage**

Signed percentage of pulse amplitude apportioned to electrode channel for the pulse.

-100 to +100.

Special setting to designate that a channel should be in uncontrolled-current (voltage) mode:

UNCONTROLLED\_SRC\_SNK\_CHAN = 127

**UINT16 StimWidth**

Test pulse stimulus phase pulse width in microseconds. (Setting to 0 disables definition.) If not set to 0, see 6.20 Test Pulse Stim Phase Pulse Width Limits for limits.

**UINT8 InterphaseDelay**

The interphase delay, in microseconds, if non-standard recovery phase is requested. The interphase delay limits are the same as for normal pulse definitions, see 6.12 Interphase delay Limits.

**UINT8 RecoveryRatioSelect**

The recovery ratio (and type) to use if non-standard recovery phase is requested. A setting of 0 indicates passive recovery. See 6.21 Test Pulse Recovery Ratio Select Limits.

**UINT16 PassiveRecoveryWidth**

The passive recovery pulse width, in microseconds. See 6.22 Test Pulse Passive Recovery Width Limits.

**UINT16 CBCWidth**

The Charge Balance Correction (CBC) pulse width, in microseconds. A value of 0 indicates that this pulse has no CBC phase. See 6.23 Test Pulse Charge Balance Correction Width Limits.

**UINT8 Flags**

Bitfield controlling stimulation options. The following flags are defined:

|  |  |  |
| --- | --- | --- |
| Mnemonic | Value | Interpretation |
| USRC\_STIM | 0x01 | If 1, uncontrolled channels during the stimulation phase are sources. Otherwise, they are sinks. |
| USRC\_REC | 0x02 | If 1, uncontrolled channels during the recovery phase are sources. Otherwise, they are sinks. This applies to both active and passive recovery. |
| USRC\_CBC | 0x04 | If 1, the CBC phase uses uncontrolled sources. Otherwise, it uses uncontrolled sinks. If the pulse has no CBC phase, this flag is ignored. |
| WAVE\_STIM\_RECT | 0x00 | If neither WAVE\_STIM\_RAM0 nor WAVE\_STIM\_RAM1 are set, a rectangular waveform is used. |
| WAVE\_STIM\_RAM0 | 0x10 | Use the waveform in RAM0 in the stimulation phase. |
| WAVE\_STIM\_RAM1 | 0x18 | Use the waveform in RAM1 in the stimulation phase. |
| WAVE\_RECOVERY\_RECT | 0x00 | If neither WAVE\_RECOVERY\_RAM0 nor WAVE\_RECOVERY\_RAM1 are set, a rectangular waveform is used. |
| WAVE\_RECOVERY\_RAM0 | 0x40 | Use the waveform in RAM0 in the recovery phase. (Applies to active recovery only.) |
| WAVE\_RECOVERY\_RAM1 | 0x60 | Use the waveform in RAM1 in the recovery phase. (Applies to active recovery only.) |

**UINT8 reserved**

Reserved for future use. Must be set to 0 on transmit and ignored on receive.

Set Test Waveform

Loads a stimulation waveform into the stimulation hardware for test purposes.

This command is valid only when in the STIM\_TEST stimulation state and stimulation is not being generated.

Sends

**Token:**

* TKN\_SET\_TEST\_WAVEFORM

**Parameters:**

**UINT8 ram**

Select which waveform RAM to use. Two RAMs are available, numbered 0 and 1.

**UINT8 reserved**

Pad byte for word alignment. Must be set to 0 on transmit.

**UINT8 waveform[256]**

256-sample waveform template. Each sample must be between 0 (no current) and 255 (full scale output). The stimulation output current will be scaled proportionally. The template will be automatically under- or over-sampled for pulse widths shorter or longer than 256 µs.

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* 0x70 FEATURE NOT ENABLED

Get Titration Stimulation / Set Titration Stimulation

Get/Set the current titration program. Valid only when in the STIM\_TITRATE stimulation state.

### Get Titration Stimulation

Sends

**Token:**

* TKN\_GET\_TITR\_STIM

**Parameters:**

None

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* 0x70 FEATURE NOT ENABLED

**Data:**

**TITR\_STIM**

The current titration stimulation definition.

See 10.45.3, “Set/Get Titration Program Common Command Parameter / Response Data”.

### Set Titration Stimulation

When in the STIM\_TITRATE stimulation state, this command controls the stimulation output. If titration stimulation is not already active, the xPG will start stimulation with the parameters in this command. Otherwise, it will switch to the program settings contained in this command at the end of the current stimulation program cycle.

The command is not accepted when in any other stimulation state.

**Token:**

* TKN\_SET\_TITR\_STIM

**Parameters:**

**TITR\_STIM**

The titration program definition data block.

See 10.45.3, “Set/Get Titration Program Common Command Parameter / Response Data”.

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* 0x2E STIM SETUP ERROR BAD STIM ASIC READBACK
* 0x36 MOD VIOLATES CHARGE DENSITY LIMIT
* 0x37 MOD VIOLATES CURRENT DENSITY LIMIT
* 0x3B PULSE AMPLITUDE EXCEEDS LIMITS
* 0x70 FEATURE NOT ENABLED
* 0x38 PW FREQUENCY CONFLICT

**Data:**

* NONE

### Set/Get Titration Program Common Command Parameter / Response Data

TITR\_STIM

**UINT8 frequencyIndex**

Index into the program frequency table identifying the desired frequency for the program. Limits are the same as for normal program definition, see 6.9 Program Frequency Index Limits.

**UINT8 numPulses**

The number of active pulse definitions. This number of pulses are generated, starting from the first definition.

NumPulses = 0 turns off titration stimulation without leaving the STIM\_TITRATE Stimulation State.

**TITR\_PULSE pulse[NUM\_PULSES\_PROGRAM]**

Pulse definitions for the titration stimulation. See below. Pulse definitions beyond the number in NumPulses are ignored.

TITR\_PULSE

**UINT16 Amplitude**

Stimulus phase amplitude in microamps. See 6.6 Pulse Amplitude Limits for limits.

**UINT16 pulseWidth**

The pulse width of the stimulation phase of the pulse, in microseconds.

**INT8 [NUM\_ELECTRODES] channelPercentage**

Signed percentage of pulse amplitude apportioned to electrode channel for the pulse.

-100 to +100.

Get Ramp Time / Set Ramp Time

Get/Set the time over which program ramping should occur.

### Get Ramp Time

Sends

**Token:**

* TKN\_GET\_RAMP\_TIME

**Parameters:**

* NONE

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* 0x68 RAMP TIME CORRUPTED
* 0x78 READ FAILED

**Data:**

**UINT16 RampTime**

Time in milliseconds over which to ramp the program to its full amplitude values  
0 (No Ramping) to 8 seconds. (Limited to multiples of 1000, to achieve integer seconds.)

### Set Ramp Time

Sends

**Token:**

* TKN\_SET\_RAMP\_TIME

**Parameters:**

**UINT16 RampTime**

Time in milliseconds over which to ramp the program to its full amplitude values  
0 (No Ramping) to 8 seconds. Limited to multiples of 1000, to achieve integer seconds.

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* 0x77 WRITE FAILED

**Data:**

* NONE

Get VBAT

Returns the VBAT voltage in internal measurement units. Used for calibrations and device testing. May be useful for troubleshooting in the field.

Sends

**Token:**

* TKN\_GET\_VBAT

**Parameters:**

* NONE

Returns

**Response Codes:**

* 9.1 Standard Response Codes

**Data:**

**UINT16 vbat**

### The voltage of the xPG’s internal battery, also known as VBAT. The value is returned in internal units. These internal units are the same as those used for the battery state calibration tables. See 10.37.4 Get/Set General Calibration Common data.

Get software description block / Set software description block

These messages read or write the Software Description Block, used by the Bootloader to set and read the parameters for booting the xPG software.

### Get Software Description Block

Sends

**Token:**

* TKN\_GET\_SDB

**Parameters:**

* NONE

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* 0x60 DATA CORRUPTED

**Data:**

**UINT32 HighMemStartAddress**

This is the starting address of the high-memory (greater than 0xffff) area used for code and static data. The bootloader computes the CRC of the code between HighMemStartAddress and the HighMemEndAddress.

**UINT32 HighMemEndAddress**

This is the ending address of the high-memory (greater than 0xffff) area that is used for code and static data. The bootloader computes the CRC of the code between the HighMemStartAddress and the HighMemEndAddress.

**UINT16 LowMemStartAddress**

This is the starting address of the low-memory (less than 0x10000) area used for code and static data. The bootloader computes the CRC of the code between LowMemStartAddress and the LowMemEndAddress.

**UINT16 LowMemEndAddress**

This is the ending address of the low-memory (less than 0x10000) area that is used for code and static data. The bootloader computes the CRC of the code between the LowMemStartAddress and the LowMemEndAddress.

**UINT16 VectorStartAddress**

This is the starting address of the interrupt vector table. The bootloader computes the CRC of the data in the interrupt vector table.

**UINT16 VectorEndAddress**

This is the ending address of the interrupt vector table. The bootloader computes the CRC of the data in the interrupt vector table.

**UINT16 ApplicationEntryPoint**

This is the address of the application entry point.

**UINT16 ApplicationCRC**

This is the CRC calculated on the three areas of flash described above. The first area is the low memory, then the interrupt vector table, and finally the high-memory area.

**UINT8 BootFlag**

The BootFlag is a combination of the following possible values:

| Flag Name | Value | Description |
| --- | --- | --- |
| BOOTFLAG\_NO\_APPLICATION | 0x01 | When set, this bit indicates that there is no application. |
| BOOTFLAG\_CALCULATE\_CRC | 0x02 | When set, this bit tells the bootloader to compute the CRC of the application areas. |
| BOOTFLAG\_STORAGE\_IF\_NO\_APP | 0x04 | This bit, when set, tells the bootloader to enter storage mode if the CRC fails or if there is no application. |

**UINT8 MICSListeningWindowDuration**

This is the time in seconds that the bootloader will wait before trying to boot.

**UINT16 SoftwareDescriptionBlockCRC**

This is the CRC of the software description block.

### Set Software Description Block

Sets the values in the Software Description Block.

**Token:**

* TKN\_ SET\_SDB

**Parameters:**

**UINT32 HighMemStartAddress**

This is the starting address of the high-memory (greater than 0xffff) area used for code and static data. The bootloader computes the CRC of the code between HighMemStartAddress and the HighMemEndAddress.

**UINT32 HighMemEndAddress**

This is the ending address of the high-memory (greater than 0xffff) area that is used for code and static data. The bootloader computes the CRC of the code between the HighMemStartAddress and the HighMemEndAddress.

**UINT16 LowMemStartAddress**

This is the starting address of the low-memory (less than 0x10000) area used for code and static data. The bootloader computes the CRC of the code between LowMemStartAddress and the LowMemEndAddress.

**UINT16 LowMemEndAddress**

This is the ending address of the low-memory (less than 0x10000) area that is used for code and static data. The bootloader computes the CRC of the code between the LowMemStartAddress and the LowMemEndAddress.

**UINT16 VectorStartAddress**

This is the starting address of the interrupt vector table. The bootloader computes the CRC of the data in the interrupt vector table.

**UINT16 VectorEndAddress**

This is the ending address of the interrupt vector table. The bootloader computes the CRC of the data in the interrupt vector table.

**UINT16 ApplicationEntryPoint**

This is the address of the application entry point.

**UINT16 ApplicationCRC**

This is the CRC calculated on the three areas of flash described above. The first area is the low memory, then the interrupt vector table, and finally the high-memory area.

**UINT8 BootFlag**

The BootFlag is a combination of the following possible values:

| Flag Name | Value | Description |
| --- | --- | --- |
| BOOTFLAG\_NO\_APPLICATION | 0x01 | When set, this bit indicates that there is no application. |
| BOOTFLAG\_CALCULATE\_CRC | 0x02 | When set, this bit tells the bootloader to compute the CRC of the application areas. |
| BOOTFLAG\_STORAGE\_IF\_NO\_APP | 0x04 | This bit, when set, tells the bootloader to enter storage mode if the CRC fails or if there is no application. |

**UINT8 MICSListeningWindowDuration**

This is the time in seconds that the bootloader will wait before trying to boot.

**UINT16 SoftwareDescriptionBlockCRC**

This is the CRC of the software description block. The IPG overwrites this value with its computed CRC.

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* 0xC9 FLASH\_WRITE\_ERROR\_BAD\_WRITE

**Data:**

* NONE

Read Memory / Write Memory

Read or write memory and registers in the IPG.

The xPG has several devices with distinct address spaces. In addition, some addresses within the MSP430’s own address space have to be accessed with word operations and some have to be accessed with byte operations. For the purposes of the read and write memory commands, these distinct address spaces, plus the different access methods for the MSP430’s built-in bus, are mapped into a single 32-bit address space. The upper 8 bits indicate in which physical address space the other 24 bits should be interpreted.

The address space is decoded as follows:

|  |  |  |  |
| --- | --- | --- | --- |
| **Address** | **Address space** | **Access** | **Notes** |
| 0x000nnnnn | MSP430 | Word | Bootloader: All writes use flash write algorithm.  Application: Writes are not permitted. |
| 0x010nnnnn | MSP430 | Byte | Not in bootloader. |
| 0x020nnnnn | MSP430 | Word | Not in bootloader. |
| 0x1000nnnn | NVRAM | Byte | Not in bootloader. |
| 0x200000nn | Power ASIC (Pluto) | Byte | Not in bootloader. |
| 0x30000nnn | Stim ASIC (Saturn) | Byte | Not in bootloader. |
| 0x400000nn | MICS (ZL7010x) | Byte | Not in bootloader. |

In other words, to read the waveform RAM that is at the Stim ASIC’s register address 0x800, send a Read Memory command for address 0x30000800.

When accessing a region with word access, the address must be even and the length of the access must be even. Regions with byte access do not have this restriction.

Bootloader mode does not support any address region other than 0x00, and only supports writing to flash. When not in bootloader mode, writes in region 0x00 are not permitted and will result in an error.

All addresses other than those listed above are reserved for future use.

The first and last address in any read or write must fall within the same region. Additional constraints on addressing and additional protocols for reading and writing may be imposed by the hardware or, in some cases, by the device driver layer, but are beyond the scope of this document.

### Write Memory

Write one or more bytes to registers or memory.

**Token:**

* TKN\_WRITE\_MEMORY

**Parameters:**

**UINT32 address**

The address at which to begin the write. See the address space table above. Addresses in word-access regions must be even.

**UINT16 length**

The number of bytes to write. Lengths in word-access regions must be even.

**UINT8 data[64]**

The data to write, starting at element 0 and proceeding to element (length – 1).

Returns

**Response Codes:**

* 9.1 Standard Response Codes

**Data:**

NONE

### Read Memory

Reads one or more bytes from a region of registers or memory.

**Token:**

* TKN\_READ\_MEMORY

**Parameters:**

**UINT32 address**

The address at which to begin the read. See the address space table above. Addresses in word-access regions must be even.

**UINT16 length**

The number of bytes to read. Lengths in word-access regions must be even.

Returns

**Response Codes:**

* 9.1 Standard Response Codes

**Data:**

**UINT32 address**

The address at which to begin the read. See the address space table above. Addresses in word-access regions must be even.

**UINT16 length**

The number of bytes to read. Lengths in word-access regions must be even.

**UINT8 data[64]**

The data to write, starting at element 0 and proceeding to element (length – 1).

Reset xPG

Causes the xPG to reset itself. This command is used both by the bootloader and by the application.

**Token:**

* TKN\_RESET

**Parameters:**

None

Returns

**Response Codes:**

* 9.1 Standard Response Codes

**Data:**

NONE

Erase flash block

Erase one or more blocks of flash ram.

**Token:**

* TKN\_ERASE\_FLASH\_BLOCK

**Parameters:**

**UINT32 startAddress**

The address at which to begin the erasing. Note that the memory map includes segment boundaries and all addresses within a segment will cause that segment to be erased.

**UINT32 endAddress**

The final address to be erased. Note that the memory map includes segment boundaries and all addresses within a segment will cause that segment to be erased.

Returns

**Response Codes:**

* 9.1 Standard Response Codes

**Data:**

NONE

Read Interrupt Vectors / Write Interrupt Vectors

Reads and writes the interrupt vectors.

### Read Interrupt Vectors

Read the interrupt vectors from memory.

**Token:**

* TKN\_READ\_VECTORS

**Parameters:**

None

Returns

**Response Codes:**

* 9.1 Standard Response Codes

**Data:**

The interrupt vectors are returned in the following order:

**UINT16 DAC12**

**UINT16 DMA**

**UINT16 USCI\_A1\_USCI\_B1\_transmit**

**UINT16 USCI\_A1\_USCI\_B1\_receive**

**UINT16 I\_O\_Port\_P1**

**UINT16 I\_O\_Port\_P2**

**UINT16 unused**

**UINT16 ADC12**

**UINT16 USCI\_A0\_USCI\_B0\_transmit**

**UINT16 USCI\_A0\_USCI\_B0\_receive**

**UINT16 Timer\_A3\_TACCR1\_TACCR2\_TAIFG**

**UINT16 Timer\_A3\_TACCRO**

**UINT16 Watchdog\_Timer**

**UINT16 Comparator\_A**

**UINT16 Timer\_B7\_TBCCR1\_TBCCR6\_TBIFG**

**UINT16 Timer\_B7\_TBCCR0**

**UINT16 NMI**

### Write Interrupt Vectors

Writes the interrupt vector table

**Token:**

* TKN\_WRITE\_VECTORS

**Parameters:**

None

Returns

**Response Codes:**

* 9.1 Standard Response Codes

**Data:**

**UINT16 DAC12**

The address of the DAC12 interrupt service routine.

**UINT16 DMA**

The address of the DMA interrupt service routine.

**UINT16 USCI\_A1\_USCI\_B1\_transmit**

The address of the USCI\_A1\_USCI\_B1\_transmit interrupt service routine.

**UINT16 USCI\_A1\_USCI\_B1\_receive**

The address of the USCI\_A1\_USCI\_B1\_receive interrupt service routine.

**UINT16 I\_O\_Port\_P1**

The address of the I\_O\_Port\_P1interrupt service routine.

**UINT16 I\_O\_Port\_P2**

The address of the I\_O\_Port\_P2 interrupt service routine.

**UINT16 unused**

The address of the unused interrupt service routine. Any 16-bit address may be used.

**UINT16 ADC12**

The address of the ADC12 interrupt service routine.

**UINT16 USCI\_A0\_USCI\_B0\_transmit**

The address of the USCI\_A0\_USCI\_B0\_transmit interrupt service routine.

**UINT16 USCI\_A0\_USCI\_B0\_receive**

The address of the USCI\_A0\_USCI\_B0\_receive interrupt service routine.

**UINT16 Timer\_A3\_TACCR1\_TACCR2\_TAIFG**

The address of the Timer\_A3\_TACCR1\_TACCR2\_TAIFG interrupt service routine.

**UINT16 Timer\_A3\_TACCRO**

The address of the Timer\_A3\_TACCRO interrupt service routine.

**UINT16 Watchdog\_Timer**

The address of the Watchdog\_Timer interrupt service routine.

**UINT16 Comparator\_A**

The address of the Comparator\_A interrupt service routine.

**UINT16 Timer\_B7\_TBCCR1\_TBCCR6\_TBIFG**

The address of the Timer\_B7\_TBCCR1\_TBCCR6\_TBIFG interrupt service routine.

**UINT16 Timer\_B7\_TBCCR0**

The address of the Timer\_B7\_TBCCR0 interrupt service routine.

**UINT16 NMI**

The address of the NMI interrupt service routine.

Diagnostic Data

Retrieve a diagnostic data block from the xPG. These data blocks contain outputs from various automatic diagnostics performed by the xPG. Each block holds the data from the most recent run of the diagnostic.

The format of the data from each diagnostic is documented below.

### Diagnostic Data Command

Retrieves a block of diagnostic data from the xPG.

**Token:**

* TKN\_DIAG\_DATA

**Parameters:**

**UINT8 command**

A code for the operation to be performed. The following operations are defined:

|  |  |  |
| --- | --- | --- |
| **Mnemonic** | **Value** | **Meaning** |
| DIAG\_READ | 0 | Read diagnostic data |
| DIAG\_READ\_CLEAR | 1 | Read, then clear, diagnostic data |

Additional command codes may be defined in the. At present, the data returned is that from the most recent automatic run, if any.

Diagnostic data is cleared to all 0’s.

**UINT8 select**

Selects the data to be read:

|  |  |  |
| --- | --- | --- |
| **Mnemonic** | **Value** | **Data Returned** |
| DIAG\_CAPACITOR | 0 | Output capacitor check data. |
| DIAG\_BG\_IMPEDANCE | 1 | Background impedance check data. |
| DIAG\_VBAT | 2 | Battery monitor voltage data. |
| DIAG\_THERMISTOR | 3 | Charging temperature data. |

Returns

**Response Codes:**

* 9.1 Standard Response Codes

**Data:**

**UINT8 command**

The command code for the operation performed, as described above.

**UINT8 select**

The selection of the data read, as described above.

**UINT8 data[250]**

The diagnostic data. The actual data, documented below, will be padded to fill this fixed-length field.

### Output Capacitor Check Data

Diagnostic data from an Output Capacitor Check.

**UINT16 voltage[26]**

The voltage read on each channel during the Output Capacitor Check, in ADC units.

**UINT8 dataValid**

A flag to indicate whether the data is valid.

If the Output Capacitor Check has been run since the last clear operation, dataValid = 1. Otherwise, dataValid = 0.

### Background Impedance Check Data

Diagnostic data from a Background Impedance Check.

Note that the xPG also supports an Impedance On Demand command. That data is returned in the response to that command and does not affect the data here.

**UINT16 voltage[26]**

The voltage read on each channel during the Background Impedance Check, in ADC units.

**UINT8 dataValid**

A flag to indicate whether the data is valid.

If the Background Impedance Check has been run since the last clear operation, dataValid = 1. Otherwise, dataValid = 0.

### Battery Monitor Voltage Data

Diagnostic data from the Battery Monitor’s last measurement of battery voltage.

Note that the xPG also supports a Get VBAT command that returns the battery voltage on demand. The Get VBAT command does not affect the value returned here.

**UINT16 voltage**

The voltage read from the battery during the Battery Monitor’s last measurement, in ADC units. If the voltage has not been read since the last clear operation, voltage = 0.

### Temperature Data

Diagnostic data from the Charge Manager’s last measurement of temperature.

**UINT16 thermBias**

The voltage read at the THERM\_BIAS input, in ADC units.

**UINT16 thermInput**

The voltage read at the THERM\_INPUT input, in ADC units.

**UINT16 thermOffset**

The voltage read at the THERM\_OFFSET input, in ADC units.

Get Background Impedance Data / Set Background Impedance Parameters

Gives the ability to set and get Background Impedance parameters. Background Impedance is measured when stimulation is stopped. The data is available through the diagnostic commands.

### Get Background Impedance Parameters

Sends

**Token:**

* TKN\_GET\_BG\_IMPEDANCE

**Parameters:**

* None

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* 0x60 DATA CORRUPTED

**Data:**

**BACKGROUND\_IMPEDANCE\_PARAMS BackgroundImpedanceParams**

The detailed Background Impedance Parameters. See Section 10.54.3 Set/Get Background Impedance Parameters Common Command Parameter / Response Data

### Set Background Impedance Parameters

Sends

**Token:**

* TKN\_SET\_BG\_IMPEDANCE

**Parameters:**

**BACKGROUND\_IMPEDANCE\_PARAMS BackgroundImpedanceParams**

The detailed Background Impedance Parameters. See Section 10.54.3 Set/Get Background Impedance Parameters Common Command Parameter / Response Data

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* 0x77 WRITE FAILED

**Data:**

* NONE

### Set/Get Background Impedance Parameters Common Command Parameter / Response Data

**BACKGROUND\_IMPEDANCE\_PARAMS**

**UINT8 bgiAmplitude**

**0x01 Low amplitude pulse (100 uAmp)**

**0x02 High amplitude pulse (200 uAmp)**

**UINT8 enabled**

Enable and disables Background Impedance measurement and Output Capacitor Check.

1 = Enabled, 0 = Disabled.

Log Range

Queries and retrieves maximum and minimum serial number available in the specified log. This is especially useful in retrieving the last several log entries to see what has happened most recently in time.

### Get Log Range

Sends

**Token:**

TKN\_GET\_LOG\_RANGE

**Parameters:**

**UINT8 LogNum**

Number of the Event Log.

0 = Major Event Log

1 = Normal Event Log

Returns

**Response Codes:**

* 9.1 Standard Response Codes

**Data:**

**UINT8 LogNum**

Number of the Event Log returned.

**UINT8 minSerialNum**

The serial number of the oldest entry in the log

**UINT8 maxSerialNum**

The serial number of the newest entry in the log

Calibrate Channel

Uses the xPG’s on-board stimulation amplitude calibrator to measure the output current of one channel.

The output current may be selected to be of either polarity (source or sink). The output current can optionally use the xPG’s calibration table. If the calibration table is used, the command can be used to verify the output after calibration. If the calibration table is not used, the command can be used to generate a new calibration table or to check the stimulation ASIC’s inherent accuracy.

This command is accepted only in the STIM\_TEST state. Stimulation must be off.

When this command is used, the xPG’s outputs are disconnected from the stimulation circuitry. No stmiulation pulse is generated externally.

Sends

**Token:**

* TKN\_CALIBRATE\_CHANNEL

**Parameters:**

**UINT8 chan**

The channel to calibrate

**UINT8 flags**

Flag bits for the calibration.

The following flag bits are defined:

|  |  |  |
| --- | --- | --- |
| Mnemonic | Value | Meaning |
| CAL\_SRC | 0x01 | If set, the channel shall source current.  If clear, the channel shall sink current. |
| CAL\_RAW | 0x02 | If set, scale the amplitude into DAC units with the ratio 15 µA = 1 MDAC LSB.  If clear, scale the amplitude into DAC units using the Channel Calibration table. |

**UINT16 amplitude**

The pulse amplitude (in microamps) to request.

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* 0x2E SETUP ERROR BAD STIM ASIC READBACK
* 0x40 TIMED OUT
* 0x70 FEATURE NOT ENABLED

**Data:**

**UINT8 chan**

The channel that was calibrated

**UINT8 flags**

Flag bits for the calibration.

See flags, above.

**UINT16 amplitude**

The pulse amplitude that was measured, in microamps.

Inject Event

Injects an event into the xPG for testing purposes.

### Inject Event

Sends

**Token:**

TKN\_INJECT\_EVENT

**Parameters:**

**UINT16 eventId**

ID of event.

**UINT16 eventData**

If event ID is an active or charge error, this represents the error ID.

Returns

**Response Codes:**

* 9.1 Standard Response Codes

### Table of Event IDs

|  |  |  |
| --- | --- | --- |
| **EVENT\_ID** | **EVENT** | **Description** |
| 3 | E\_BATT\_MEASURE\_BATTERY | 1 Minute Check When System is Idle |
| 3 | E\_BATT\_MEASURE\_BATTERY | 5 Minute Check When System is Charging |
| 3 | E\_BATT\_MEASURE\_BATTERY | 4 Hour Check When System Battery Level is High |
| 3 | E\_BATT\_MEASURE\_BATTERY | 1 Hour Check When System Battery Level is Low |
| 3 | E\_BATT\_MEASURE\_BATTERY | 15 Minute Check When System Battery Level is Critical |
| 8 | E\_CHARGE\_PRECON\_STATE\_TO | 1 Hour Check if Charging Pre-Condition is Finished. |
| 5 | E\_CHARGE\_CONCUR\_STATE\_TO | 8 Hour Check if Constant Current is Going Too Long. |
| 15 | E\_CLOCK\_VERIFY | 1 Hour Clock Check While Stim is On |
| 15 | E\_CLOCK\_VERIFY | 24 Hour Clock Check No Matter What |
| 18 | E\_ERR\_CHARGER\_ERROR | Inject a charger error - see SWEX 0091 for error codes. |
| 19 | E\_ERR\_ERROR\_DETECTED | Inject an active error - see SWEX 0091 for error codes. |

Reserved

MICS Options

Changes MICS communications options. The options affect the current session only.

At the start of each new session, the MICS options are set to 0x00. They may then be changed with the MICS Options command.

Sends

**Token:**

* TKN\_MICS\_OPTIONS

**Parameters:**

**UINT8 flags**

Bitmapped flags, as described in the following table:

|  |  |  |
| --- | --- | --- |
| **Name** | **Value** | **Definition** |
| *MICS\_LOW\_LATENCY* | 0x01 | If set, optimizes the connection for minimum communications latency  If clear, optimizes the connection for low power but higher latency. |

All undefined flag bits are reserved and must be zero.

Returns

**Response Codes:**

* 9.1 Standard Response Codes

**Data:**

No data.

Pair PPC

Instructs the IPG to recognize the EXID sent with this command as valid for future commands. This command is only enabled when VRECT is present. This overwrites the old PPC pairing data. This command is not valid on an EPG.

This command is allowed on an IPG with unpaired EXIDs when VRECT is present. Unpaired processing is not allowed on an EPG.

**Token:**

* TKN\_PAIR\_PPC

**Parameters:**

No Parameters

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* FEATURE NOT ENABLED

**Data:**

No data.

Pair PoP

Instructs the IPG arm itself for paring of a PoP. The next device to connect to the IPG will be accepted for all future communications. This command is only enabled when VRECT is present. This command is not valid on an EPG.

This command is allowed on an IPG with unpaired EXIDs when VRECT is present. Unpaired processing is not allowed on an EPG.

**Token:**

* TKN\_PAIR\_POP

**Parameters:**

No Parameters

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* FEATURE NOT ENABLED

**Data:**

No data.

Stop Bootlaoder

When the bootloader is running, this command instructs it to stop executing, drop the connection, and begin running the application immediately. The command response is transmitted before the connection is ended.

If the bootloader is not running, the system will return COMMAND\_ACCEPTED without doing any further processing.

**Token:**

* TKN\_STOP\_BOOTLOADER

**Parameters:**

No Parameters

Returns

**Response Codes:**

* 9.1 Standard Response Codes
* DATA CORRUPTED

**Data:**

No data.

Tune MICS

Instructs the IPG to perform an automatic tune-up of the MICS radio.

**Token:**

* TKN\_MICS\_TUNE

**Parameters:**

No Parameters

Returns

**Response Codes:**

* 9.1 Standard Response Codes

**Data:**

No data.

MICS Diagnostics

Causes the MICS radio to go into a diagnostic mode.

**Token:**

* TKN\_MICS\_DIAGNOSTICS

**Parameters:**

UINT8 Channel

Indicates the channel to use while in diagnostic mode. When this parameter is set to 255, the measurements are taken on the current channel and the session is kept open where possible. Otherwise, the session is dropped and the specified channel is used. Channels 0-9 are MICS-band. Channels 10-11 are ISM-band.

UINT16 Delay

The delay in ms before the diagnostic is started.

UINT16 Duration

The amount of time in ms over which the diagnostic mode is active. If set to 0, the diagnostic mode is run until the system is reset, or until the current session ends when the session is kept open.

UINT8 Diagnostic Mode

The diagnostic to run. This field may have any of the following enumerated values:

|  |  |
| --- | --- |
| **Value** | **Diagnostic Operation** |
| 0 | RSSI Measurement |
| 1 | Transmit CW Mode |
| 2 | Transmit Modulated Signal |
| 3 | Receive Mode |

Returns

**Response Codes:**

* 9.1 Standard Response Codes

**Data:**

No data.

**Diagnostic Modes**

**RSSI**

Causes the MICS radio to perform RSSI measurements using the internal ADC. This measurement is either made on the current session, or else the session is closed and the measurements are taken on the channel specified. The Zarlink chip’s internal ADC is used to make these measurements. The RSSI results are placed in the activity log. While active, the system will read RSSI readings as fast as possible.

**Transmit CW Mode**

The MICS radio will transmit a constant wave on the specified channel. The session is always dropped when running this diagnostic mode.

**Transmit Modulated Signal**

The MICS radio will transmit a continuous modulated signal on the specified channel. The session is always dropped when running this diagnostic mode.

**Receive Mode**

The MICS radio is configured such that all of the receive circuitry is enabled, but is not actively in use to perform communication.

# Command Support Specification

## Action Commands and Data Queries

### The following table identifies those commands which either result in an action performed on the xPG device, or returns specified data from the xPG for which there is no corresponding command to “set” that data.

### “” Indicates that a device supports the command.

### “🗴” Indicates that a device does not support the command.

### “(MAY)” Indicates that a device can support the command, but is not required to as the same effective outcome can be achieved through use of a different command on the Programmer, or the command could be used for an optional secondary function use.

### “Mnfctr” Indicates that the command is intended for manufacturing purposes. Such commands might be implemented on a special release of firmware that runs on the Clinician Programmer hardware, or through some other TBD device to be developed specifically for use in manufacturing. For identifying the required commands and their functionality, these special purpose commands are indicated in the CP column.

| **Programmer** | | | **xPG** | | **Name** | **Command Token** |
| --- | --- | --- | --- | --- | --- | --- |
| *CP* | *PPC* | *PoP* | *IPG* | *EPG* |
|  |  |  |  |  | Get xPG Status | 0x0001 |
|  |  |  |  |  | xPG Stimulation | 0x0002 |
|  |  |  |  |  | Select Program | 0x0003 |
|  |  |  |  |  | Increment Program Amplitude | 0x0004 |
|  |  |  |  |  | Decrement Program Amplitude | 0x0005 |
| 🗴 | 🗴 |  |  |  | Get PoP Constants | 0x0006 |
| 🗴 |  | 🗴 |  |  | Get PPC Constants | 0x0007 |
|  |  | 🗴 |  |  | Get Program Names | 0x0008 |
|  |  | 🗴 |  |  | Set Pulse Amplitude | 0x0009 |
|  |  | 🗴 |  |  | Increment Pulse Amplitude | 0x000A |
|  |  | 🗴 |  |  | Decrement Pulse Amplitude | 0x000B |
|  |  | 🗴 |  |  | Set Pulse Width | 0x000D |
|  |  | 🗴 |  |  | Get Pulse Widths | 0x000C |
|  |  | 🗴 |  |  | Increment Pulse Width | 0x000E |
|  |  | 🗴 |  |  | Decrement Pulse Width | 0x000F |
|  |  | 🗴 |  |  | Get Program Frequency Index | 0x0010 |
|  | (MAY) | 🗴 |  |  | Set Program Frequency | 0x0011 |
|  |  | 🗴 |  |  | Increment Program Frequency Index | 0x0012 |
|  |  | 🗴 |  |  | Decrement Program Frequency Index | 0x0013 |
| 🗴 |  | 🗴 |  | 🗴 | Charging Control | 0x0015 |
| Mnfctr | 🗴 | 🗴 |  |  | Set xPG Identity | 0x0017 |
|  |  | (MAY) |  |  | Get xPG Identity | 0x0016 |
| (MAY) |  | 🗴 |  |  | Restore Program Parameter Defaults | 0x0018 |
|  | 🗴 | 🗴 |  |  | Get Program Runtime | 0x0034 |
|  | 🗴 | 🗴 |  |  | Reset Program Runtime | 0x0035 |
| Mnfctr   | 🗴 | 🗴 |  | 🗴 | Enter Storage Mode | 0x0037 |
|  | 🗴 | 🗴 |  |  | Impedance Test | 0x0036 |
|  | 🗴 | 🗴 |  | 🗴 | Get Temperature Diagnostics | 0x004F |
| Mnfctr | 🗴 | 🗴 |  |  | Get Output Capacitor Diagnostics | 0x004F |
| Mnfctr | 🗴 | 🗴 |  |  | Get Background Impedance Diagnostics | 0x004F |
| Mnfctr | 🗴 | 🗴 |  |  | Get Batter Monitor Voltage Diagnostics | 0x004F |
|  | 🗴 | 🗴 |  |  | Get Event Log | 0x0032 |
|  | 🗴 | 🗴 |  |  | Clear Event Log | 0x0031 |
|  | 🗴 | 🗴 |  |  | Get Error Log | 0x0032 |
|  | 🗴 | 🗴 |  |  | Clear Error Log | 0x0031 |
|  | 🗴 | 🗴 |  |  | Append Event Log | 0x0033 |
|  | 🗴 | 🗴 |  |  | Append Error Log | 0x0033 |
|  | 🗴 | 🗴 |  |  | Test Pulse Feature Enable | 0x0002 |
|  | 🗴 | 🗴 |  |  | Test Pulse Output Control | 0x0041 |
| Mnfctr | 🗴 | 🗴 |  |  | Turn Debug Messages On | 0x0025 |
| Mnfctr | 🗴 | 🗴 |  |  | Turn Debug Message Off | 0x0026 |
| Mnfctr | 🗴 | 🗴 |  |  | Debug Message | 0x0027 |
| Mnfctr | 🗴 | 🗴 |  |  | Echo Short | 0x0038 |
| Mnfctr | 🗴 | 🗴 |  |  | Echo Long | 0x0039 |
| Mnfctr | 🗴 | 🗴 |  |  | Get VBAT | 0x0046 |
| Mnfctr | 🗴 | 🗴 |  |  | Reset xPG | 0x004B |
| Mnfctr | 🗴 | 🗴 |  |  | Erase Flash Block | 0x004C |
| Mnfctr | 🗴 | 🗴 |  |  | Get Log Range | 0x0054 |
|  | 🗴 | 🗴 |  |  | MICS Options | 0x0057 |
| 🗴 |  | 🗴 |  | 🗴 | Pair PPC | 0x0058 |
| 🗴 |  | 🗴 |  | 🗴 | Pair PoP | 0x0059 |
|  | 🗴 | 🗴 |  | 🗴 | Stop Bootloader | 0x005A |
|  | 🗴 | 🗴 |  |  | Tune MICS | 0x005B |
| 🗴 | 🗴 | 🗴 |  |  | MICS Diagnostics | 0x005C |

## Data Query and Retrieves

### The following table identifies those commands which operate on specified data in the xPG device by providing a means to “set” the specified values, or “get” its present values.

### “” Indicates that a device supports “Set” and “Get” for the data.

### “G” Indicates that a Programmer device supports only “Get” for the data.

### “🗶” Indicates that a device does not support the command.

### “Mnfctr” Indicates that the command is intended for manufacturing purposes. Such commands might be implemented on a special release of firmware that runs on the Clinician Programmer hardware, or through some other TBD device to be developed specifically for use in manufacturing. For identifying the required commands and their functionality, these special purpose commands are indicated in the CP column.

| **Programmer** | | | **xPG** | | **Name** | **Command Token** |
| --- | --- | --- | --- | --- | --- | --- |
| *CP* | *PPC* | *PoP* | *IPG* | *EPG* |
|  | G | 🗴 |  |  | Set Program Definition  Get Program Definition | 0x001A  0x0019 |
|  | G | 🗴 |  |  | Set Program Constants  Get Program Constants | 0x001C  0x001B |
|  | 🗴 | 🗴 |  |  | Set Pulse Constants  Get Pulse Constants | 0x0020  0x001F |
|  | 🗴 | 🗴 |  |  | Set Impedance Constants  Get Impedance Constants | 0x0050  0x0051 |
|  | 🗴 | 🗴 |  |  | Set Lead Limits  Get Lead Limits | 0x0022  0x0021 |
| Mnfctr | 🗴 | 🗴 |  |  | Set Channel Calibrations  Get Channel Calibrations | 0x0024  0x0023 |
|  | 🗴 | 🗴 |  |  | Set Configurable Device Parameters  Get Configurable Device Parameters | 0x001E  0x001D |
| Mnfctr | 🗴 | 🗴 |  |  | Set Stim Asic HV Calibrations  Get Stim Asic HV Calibrations | 0x002C  0x002B |
|  | 🗴 | 🗴 |  |  | Set Test Pulse Definition  Get Test Pulse Definition | 0x0041  0x0040 |
|  | 🗴 | 🗴 |  |  | Set CP Data  Get CP Data | 0x0030  0x002F |
| Mnfctr | 🗴 | 🗴 |  |  | Set General Calibration  Get General Calibration | 0x003B  0x003A |
| Mnfctr | 🗴 | 🗴 |  |  | Set Trim List  Get Trim List | 0x003D  0x003C |
|  | 🗴 | 🗴 |  |  | Set Counter  Get Counter | 0x003F  0x003E |
|  | 🗴 | 🗴 |  |  | Set Titration Pulse Definition  Get Titration Pulse Definition | 0x0043  0x0042 |
|  | 🗴 | 🗴 |  |  | Set Ramp Time  Get Ramp Time | 0x0045  0x0044 |
| Mnfctr | 🗴 | 🗴 |  |  | Set Software Description Block  Get Software Description Block | 0x0047  0x0048 |
| Mnfctr | 🗴 | 🗴 |  |  | Write Memory  Read Memory | 0x0049  0x004A |
| Mnfctr | 🗴 | 🗴 |  |  | Write Vectors  Read Vectors | 0x004D  0x004E |
| Mnfctr   | 🗴 | 🗴 |  |  | Set Background Impedance Parameters  Get Background Impedance Parameters | 0x0050  0x0051 |

# Revision History

| **Revision Level** | **Revision Description** | **ECN**  **No#** | **Effective Date** |
| --- | --- | --- | --- |
| 1.1 | Initial Release | 1138 | 05/07/10 |
| 1.2 | Modified Sections:   * 6.10 – updated charging level limits * 8 Command Tokens – added commands 31 & 32 * 9.2 Command Specific Response Codes – added 9.2.1 – 9.2.29 * Added all Command Specific Response Codes to sections: 10.2 - 10.20, 10.22, 10.23, 10.25.1, 10.25.2, 10.26.1, 10.26.2, 10.27.1, 10.27.2 * Added Reserved Byte to Sections 10.12, 10.13 * 10.21 Charging Control command   Removed Sections:   * Charging Temperature Level Limits – previously section 6.11   Added Sections:   * 6.12 Interphase Delay Limits * 6.13 Passive recovery Limits * 6.14 Active recovery limits * 6.15 Holdoff limits * 6.16 Increment lockout limits * 7.8 Recovery ratio enable bit map * 7.9 IPG battery charging state * 10.28 Get Pulse Constants/Set Pulse Constants   General Fixes:   * All Pulse Amplitude Step Index entries were updated from UINT8 to INT8 | 1153 | 06/18/10 |
| 1.3 | * Miscellaneous corrections of typos / grammar. * Modified description of 10.21Charging Control command * Clarified return data for Get Program Names command * Noted that Charging command (10.20) is not presently used in the charging protocol. * Added section 7.10 IPG Temperature State * Added ChargingError byte to Get xPG Status response * Added RampTime and IncrementLockoutMsecs to response data block for Get PoP Constants and Get PPC Constants commands | 1199 | 11/30/10 |
| 1.4 | * Added new programmable data / calibration Get/Set pairs:   + Lead Limits   + Channel Calibrations   + Stim Asic HV Calibrations * Added new CP-only commands related to the Test Pulse feature:   + Test Pulse Feature Enable   + Get/Set Test Pulse Definition   + Test Pulse Output Control   + Test Pulse HV Override * Added new response codes to several existing commands related to the Test Pulse feature * Added new data feedback to Get XPG Status data block:   + Test pulse status byte   + Compliance voltage status byte (current for TP only) * Added new response codes, mostly related to the Test Pulse feature * Added implant date / time and replacement months (ERI – elective replacement interval) to Get/Set Configurable Device Parameters and Get PPC Constants * Miscellaneous corrections. | 1254 | 05/13/11 |
| 1.5 | * Modified Test Pulse feature to reflect addition of Test Program functionality * Added Test Program, CP Data , Log, Program Run-time, Impedance Measurement, Storage Mode, General Calibration,Event Counter, Echo, Trim List, Undefined commands * Deleted references to ‘real-time’ patient program (Replaced by new Test Program functionality.). | 1417 | 02/23/12 |
| 1.6 | * Replaced Test Pulse and Test Program functionality with Test Stim and Titration Stim. * Removed Ramp Time from Get / Set Program Constants. * Added Get/Set Ramp Time commands. * Removed CP-only restrictions on commands. * Clarified permissible values for pulse map. | 1503 | 08/01/12 |
| 1.7 | * Add Table of Contents * Add USS polarity flags to Get / Set Test Stim * Add Set/Get Background Impedance Commands * Removed deprecated responses 0x23, 0x28, 0x29, 0x2A, 0x2C, 0x2D, 0x39, 0x71 * Added description for 0x73 * Changed name for 0x38 to PULSE WIDTH FREQUENCY CONFLICT * Added section 11 which includes additional information from SWEX 0084, effectively obsoleting that document   + Removed Charge, Delete Program, Link Quality, Acknowledge Error, Test Pulse HV Override, Get/Set Recovery Mode Constants, Get/Set Header Constants, Get/Set Run Time   + Added Get/Set CP Data, Get/Set General Calibration, Echo Short, Echo Long, Get/Set Trim List, Get/Set Counters, Get/Set Titration, Get/Set Ramp Time, Get VBAT, Get/Set Background Impedance, Get/Set Software Description Block, Memory Read/Write, Flash Memory Write, Read/Write Vectors, Get Diagnostic Data – Output Capacitor, Background Impedance, and Battery Monitor Voltage * Update General Calibration structure with additional threshold value. * Update return amplitude index for TKN\_SET\_PULSE\_AMPL. Should echo back amplitude if successful, not virtual amplitude. * Clarified variable name for amplitude step index returned in TKN\_GET\_XPG\_STATUS command. * Added description to the bootloader commands. * Added out of band debug monitor commands * Removed numerous TBD and clarified ambiguities. * Removed response codes that have been deprecated as of Rev 1.5 of SWEX 0091. Updated response code list * Add comment that counters do not rollover * Updated temperature and charge control states | 1668 | 02/15/13 |
| 1.8 | * Removed unused charging level limits. * Added CP EXID limits. * Changed MIN\_STIM\_ASIC\_HV\_CAL value to 0. * Added a description of unpaired communications, and provided details on all commands that allow it. * Clarified the use and units of the general calibration common data values. * Added Pair PPC and Pair PoP commands. * Added Stop Bootloader command. * Removed Get/Set Patient Program Runtime commands. * Corrected equation in section 6.2.2 for calculating output amplitude. * Removed unused command responses from Set Test Stimulation and Set Titration Stimulation. | 1767 | 05/21/13 |

|  |  |  |  |
| --- | --- | --- | --- |
| 1.9 | * Added index value to data for get/set channel calibration * Updated counter definitions for get/set counter commands * Removed antennae Get Antenna Tuning command. * Removed antennae Set Antenna Tuning command. * Removed warning bit for charging command. * Reduced the number of trim values from 140 to 32. * Added DATA CORRUPTED as a response to the Stop Bootloader command. * Added Tune MICS command. * Added MICS Diagnostics command. | 1882 | 07/25/13 |
| 1.10 | * Update section 10.35 for new impedance measurement parameters * Update section 10.54.3 for new background impedance measurement parameters * Update section 10.34.2 Read Log. The data returned did not include the 16 bit reserved field * COUNTER\_CONSTANT\_CURRENT\_COMPLETE counter changed to COUNTER\_PRECHARGE\_COMPLETE | 1919 | 09/10/13 |
| 1.11 | * Corrected errors in the description of Get/Set Trim List * Updated event IDs for Inject Event * Defined RTarget as a parameter of Get/Set General Calibration | 2074 | 11/22/13 |