

SCPI Command Reference and Examples (with TestController Scripts)

Revision History

Version	Date	Description
v1.0	2026-05-02	Initial release
v1.1	2026-09-02	Added dangerous command summary, error code table, command index, and command tree

Communication Configuration

- **Baud rate:** 115200
- **Data bits:** 8
- **Stop bits:** 1
- **Parity:** None
- **Line ending:** Device-dependent (commonly `\r\n` or `\n`)
- **Hardware:** CH340C (USB-to-serial)

TestController configuration example:

```
#port comFixedBaud
#baudrate 115200
#driver SCPIx
```

Dangerous Commands Summary

The following commands **write Flash, reset/reboot, erase calibration parameters, or modify configuration programs**. Before executing, always confirm device state and complete backups.

Command	Risk	Consequence
<code>*RST</code>	High	Standard reset; jumps to Bootloader
<code>*FWUPD</code> / <code>*FWUPDRST</code>	Very High	Clears product key, writes Flash, jumps to Bootloader
<code>PACKet:RATIo</code>	High	Parses 14 calibration values, writes to calibration Flash
<code>:SYSTem:BACKup:DATA</code>	High	Backs up data region to backup region, writes Flash
<code>:SYSTem:RESEt:PROGram</code>	Very High	Resets programs 0–31 to factory defaults, then jumps to Bootloader
<code>:SYSTem:RESEt:DATA</code>	Very High	Restores data from backup to data region, writes Flash, jumps to Bootloader

Command	Risk	Consequence
:SYSTem:RESEt:CALibration	Very High	Restores calibration parameters from backup, writes Flash, jumps to Bootloader
CONTRol:PROGram:SAVE	Medium-High	Writes program parameters to program storage area

Note: The following GSV configuration commands also write Flash — use with caution:

CONFig:FANS:ENABle, CONFig:BEEPer:KEY:ENABle, CONFig:BEEPer:PROcEsscompletion:ENABle, CONFig:SCREenflash:ENABle, CONFig:LISPfn:ENABle, CONFig:LDSPfn:ENABle, CONFig:AUTOstart:ENABle, CONFig:CALibration:ENABle, CONFig:TESTir:ENABle, CONFig:ACTivatebattery:ENABle, CONFig:DEFAUltprogram

Table of Contents

- 1. [General Conventions](#)
- 2. [Common Commands](#)
- 3. [Measurement and Status Queries](#)
- 4. [Program Settings \(PC Control\)](#)
- 5. [Program Control Flow](#)
- 6. [Super Control](#)
- 7. [GSV Global Configuration](#)
- 8. [Calibration, Backup, and System Reset](#)
- 9. [Typical Complete Workflow](#)
- 10. [Error Code Table](#)

Command Tree Overview

```
*IDN? *RST *FWUPD *FWUPDRST *UPT?
|
├── MEASure
|   ├── ALL?
|   ├── VOLTage?
|   ├── CURRent?
|   ├── CAPAcity?
|   ├── ENERgy?
|   └── IR?
|
├── STATus
|   ├── ?
|   ├── CHANnel?
|   └── LED?
|
├── CHECK
|   ├── STATus:LED?
|   ├── TIME:STAGe?
|   ├── TIME:TOTAL?
|   └── CYCLE:COUNT?
```

```

|
|— PROGram
|   |— SERIalNo
|   |— TYPE
|   |— TASK
|   |— CYCLe:MODE
|   |— CYCLe:COUNTs
|   |— CAPAcity
|   |— CURRent:PRECharge / CHARge / TERMination / DISCharge / DISCharge:REDUce
|   |— VOLTage:TERMination / TARGet / PRECharge to fastcharge / REStArt / CUT
|   |— TIME:PRECharge / CHARge / CUT / RESting:CHARge / RESting:DISCharge
|   |— DELTaPeak
|   |— LONGname
|   |— SHORtname
|   |— SETUp
|
|— CONTroL
|   |— PROGram:PREProcess / SAVE / COMPLeted / RUNNing
|   |— TRANSferChannelControLToPC
|   |— PROCess:StARt
|   |— RELEase
|   |— RESting
|   |— ACTivatebattery:StARt / EXIT
|
|— SUPER
|   |— CONTroL:RUN / STOP
|   |— PROGram:SETUp / TYPE / TASK
|   |— PROGram:CURRent:PRECharge / CHARge / TERMination / DISCharge / DISCharge:REDUce
|   |— PROGram:VOLTage:TERMination / TARGet / PRECharge to fastcharge / CUT (no REStArt)
|   |— PROGram:DELTApeak
|   |— CONTroL:ACTivatebattery:StARt / EXIT
|
|— CONFIg
|   |— FANS:ENABle
|   |— BEEPer
|   |   |— KEY:ENABle
|   |   |— PROCesscompletion:ENABle
|   |— SCREenflash:ENABle
|   |— LISPfn:ENABle / LDSPfn:ENABle
|   |— AUTostart:ENABle
|   |— CALibration:ENABle
|   |— TESTir:ENABle
|   |— ACTivatebattery:ENABle
|   |— DEFaultprogram
|
|— PACKet
|   |— RATIo
|   |— RATIo:RUN?
|   |— RATIo:BACKup?
|
|— SYSTem
|   |— BACKup:DATA
|   |— RESEt:PROGram
|   |— RESEt:DATA
|   |— RESEt:CALibration

```

Command Index Table

Command	Type	Section
*IDN?	Query	2.1
*RST	Action	2.2
*FWUPD / *FWUPDRST	Action	2.3
*UPT?	Query	2.4
STATus? / STATus:CHANne1?	Query	3.2.1
STATus:LED? / CHECK:STATus:LED?	Query	3.2.2
MEASure:ALL?	Query	3.3.1
MEASure:VOLTage?	Query	3.3.2
MEASure:CURREnt?	Query	3.3.3
MEASure:CAPAcity?	Query	3.3.4
MEASure:ENERgy?	Query	3.3.5
MEASure:IR?	Query	3.3.6
CHECK:TIME:STAGe?	Query	3.3.7
CHECK:TIME:TOTA1?	Query	3.3.8
CHECK:CYCLE:COUNT?	Query	3.3.9
PROGram:SERIA1no	Set / Query	4.1.1
PROGram:TYPE	Set / Query	4.1.2
PROGram:TASK	Set / Query	4.1.3
PROGram:CYCLE:MODE	Set / Query	4.1.4
PROGram:CYCLE:COUNTs	Set / Query	4.1.5
PROGram:CAPAcity	Set / Query	4.1.6
PROGram:CURREnt:PRECharge	Set / Query	4.2.1
PROGram:CURREnt:CHARGE	Set / Query	4.2.2
PROGram:CURREnt:TERMination	Set / Query	4.2.3
PROGram:CURREnt:DISCharge	Set / Query	4.2.4
PROGram:CURREnt:DISCharge:REDUce	Set / Query	4.2.5
PROGram:VOLTage:TERMination	Set / Query	4.3.1
PROGram:VOLTage:TARGET	Set / Query	4.3.2

Command	Type	Section
PROGrama:VOLTage:PRECharge:toFastCharge	Set / Query	4.3.3
PROGrama:VOLTage:REStart	Set / Query	4.3.4
PROGrama:VOLTage:CUT	Set / Query	4.3.5
PROGrama:TIME:PRECharge	Set / Query	4.4.1
PROGrama:TIME:CHARGE	Set / Query	4.4.2
PROGrama:TIME:CUT	Set / Query	4.4.3
PROGrama:TIME:RESting:CHARGE	Set / Query	4.4.4
PROGrama:TIME:RESting:DISCharge	Set / Query	4.4.5
PROGrama:DELTApeak	Set / Query	4.5.1
PROGrama:LONGname	Set / Query	4.5.2
PROGrama:SHORTname	Set / Query	4.5.3
CONTRol:PROGrama:PREProcess	Action / Query	5.1
CONTRol:PROGrama:SAVE	Action	5.2
CONTRol:PROGrama:COMPLeted	Action	5.3
PROGrama:SETUp	Action	5.4
CONTRol:PROGrama:RUNning	Action	5.5
CONTRol:TRANSferChannelControlToPC	Action / Query	5.6
CONTRol:PROCEss:START	Action	5.7
STATus:CONTRol:PROCEss:COMPLeted?	Query	5.8
CONTRol:RELEase	Action	5.9
CONTRol:RESting	Set	5.10
CONTRol:ACTivatebattery:START / EXIT	Action	5.11
SUPER:CONTRol:RUN / STOP	Action	6.1
SUPER:PROGrama:SETUp	Set	6.2
SUPER:PROGrama:TYPE	Set	6.3
SUPER:PROGrama:TASK	Set	6.4
SUPER:PROGrama:CURRENT:*	Set	6.5
SUPER:PROGrama:VOLTage:*	Set	6.6
SUPER:PROGrama:DELTApeak	Set	6.7
SUPER:CONTRol:ACTivatebattery:START / EXIT	Action	6.8

Command	Type	Section
CONFig:FANS:ENABle	Set / Query	7.1
CONFig:BEEPer:KEY:ENABle	Set / Query	7.2
CONFig:BEEPer:PROcEsscompletion:ENABle	Set / Query	7.3
CONFig:SCREenflash:ENABle	Set / Query	7.4
CONFig:LISPfn:ENABle	Set / Query	7.5
CONFig:LDSPfn:ENABle	Set / Query	7.6
CONFig:AUTOstart:ENABle	Set / Query	7.7
CONFig:CALibration:ENABle	Set / Query	7.8
CONFig:TESTir:ENABle	Set / Query	7.9
CONFig:ACTIVatebattery:ENABle	Set / Query	7.10
CONFig:DEFAUltprogram	Set / Query	7.11
PACKet:RATIo:RUN?	Query	8.1
PACKet:RATIo:BACKup?	Query	8.2
PACKet:RATIo	Set	8.3
:SYSTem:BACKup:DATA	Action	8.4
:SYSTem:RESEt:PROGram	Action	8.5
:SYSTem:RESEt:DATA	Action	8.6
:SYSTem:RESEt:CALibration	Action	8.7

1. General Conventions

1.1 Channel Parameter

Most commands require `<ch>`, which supports:

```
1, 2, 3, 4
c1, ch1, c2, ch2, c3, ch3, c4, ch4
None, Ignore
```

In TestController, the channel is selected through the `selectedChannel1` variable, which is substituted with `ch1`, `ch2`, etc., before being sent.

Channel variable definition in TC:

```
#scpiCmd channel none
:setvar: selectedChannel=inputValue

#scpiCmd channel? none?
:readmath: selectedChannel
```

1.2 Boolean Values

True: T, Y, True, Yes, On
False: F, N, False, No, Off

Buttons in TC use `ON` / `OFF`, for example:

```
CONFIg:FANS:ENABle ON
CONFIg:FANS:ENABle OFF
```

1.3 Units and Internal Conversion

Type	External Unit	Internal Unit	Handling in TC
Voltage	V	mV	Send <code>V</code> , query returns mV, <code>readmath: value/1000.0</code>
Current	A	mA	Send <code>A</code> , query returns mA, <code>readmath: value/1000.0</code>
Capacity	Ah / mAh	mAh	Send <code>Ah</code> , query returns mAh, divide by 1000
Time	min / s	min	Send <code>min</code> , query returns minutes
Energy	Wh	Wh	Returned directly
Resistance	mΩ	Raw number	Returned directly

1.4 Query and Set

- With `?`: query.
- Without `?`: set or action.
- Error: `SCPI_Error` outputs a decimal error code, for example `-224`.

1.5 Short Form and Long Form

Uppercase letters in SCPI commands form the short form; lowercase portions may be omitted. For example:

Long Form	Short Form
<code>MEASure:VOLTage?</code>	<code>MEAS:VOLT?</code>
<code>PROGram:CURRent:CHARGe</code>	<code>PROG:CURR:CHAR</code>
<code>CONTro1:TRANSferChannelContro1ToPC</code>	<code>CONT:TRAN</code>

Case-insensitive; `meas:volt?` is equivalent to `MEAS:VOLT?`. Actual line endings and abbreviation acceptance are device-dependent.

2. Common Commands

2.1 *IDN?

- **Type:** Query
- **Syntax:** *IDN?
- **Returns:** Manufacturer,Model,Serial Number,Firmware Version
- **Example:**

```
Send:      *IDN?
Return:    ENOVA TECHNOLOGY,GYRFALCON S8000,0,3.0.38
```

2.2 *RST

- **Type:** Action
- **Syntax:** *RST
- **Description:** Standard reset. ⚠ **Dangerous.**
- **Example:**

```
Send:      *RST
Return:    (none)
```

2.3 *FWUPD / *FWUPDRST

- **Type:** Action
- **Syntax:** *FWUPD or *FWUPDRST
- **Description:** Restart and launch the firmware update program. ⚠ **Dangerous.**
- **Example:**

```
Send:      *FWUPD
Return:    (none)
```

2.4 *UPT?

- **Type:** Query
- **Syntax:** *UPT?
- **Returns:** D|HH:MM:SS
- **TC usage example:**

```
#cmdSetup infoAsk Total_Runtime Global_Settings
:read: *UPT?;
:tip:Click "Get" to get the duration of continuous operation from startup to the current
time.
```

- **Example:**

Send: *UPT?
Return: 3D|01:09:59

3. Measurement and Status Queries

3.1 Measurement Polling Configuration (TC usage example)

```
#askValues MEASure:ALL? 1,2,3,4  
#askValuesReadFormat ffffffffffffffffff  
#readingDelay 5  
#interfaceType Charger Charger:2 Charger:3 Charger:4  
#interface readVoltage 0 4 8 12  
#interface readCurrent 1 5 9 13  
#interface readCapacity 2 6 10 14  
#interface readEnergy 3 7 11 15
```

Notes

- `#askValues MEASure:ALL? 1,2,3,4`: periodically sends `MEASure:ALL? 1,2,3,4`.
- `#askValuesReadFormat ffffffffffffffffff`: all 16 return values are parsed as floats.
- `#interface readVoltage 0 4 8 12`: return values at indexes 0, 4, 8, and 12 correspond to CH1-CH4 voltage.

3.2 Status Queries

3.2.1 STATUS? / STATUS:CHANNEL?

- **Type:** Query
- **Syntax:** `STATUS? <ch>` or `STATUS:CHANNEL? <ch>`
- **Returns:** Status string, such as `Charging`, `Done -5mV`, `ErrTimeout`, `ErrMaxCap`, `ErrVltOver`
- **TC usage example:**

```
#cmdSetup infoAsk Status CH1_Super_Controller  
:read: Status? Ch1;  
:tip:Click "Get" to retrieve the real-time status of the channel 1.
```

- **Example:**

```
Send: Status? Ch1  
Return: CC_Charging
```

3.2.2 STATUS:LED? / CHECK:STATUS:LED?

- **Type:** Query
- **Syntax:** `STATUS:LED? <ch>`
- **Returns:** `OFF`, `WHITE`, `RED`, `BLUE`, `YELLOW`, `PURPLE BLINKING`, `RED BLINKING`, `GREEN`
- **Example:**

```
Send:    STATus:LED? Ch1
Return:  GREEN
```

3.3 Measurement Commands

3.3.1 MEASure:ALL?

- **Type:** Query
- **Syntax:** `MEASure:ALL? <ch1>[,<ch2>...]`
- **Description:** Up to 4 channels, sorted and deduplicated, keeping only 1–4. Each channel returns 4 values.

Return format breakdown:

```
MEASure:ALL? 1,2,3,4
Return: 4.183e+00, 3.510e-01, 2.048e-02, 8.540e-02, 4.202e+00, 3.200e-01, 2.003e-02, 8.411e-02, 4.154e+00, ...

      ↑           ↑           ↑           ↑           ↑           ↑           ↑           ↑
      ↑           ↑           ↑           ↑           ↑           ↑           ↑           ↑
      CH1 volt    CH1 curr    CH1 cap    CH1 energy  CH2 volt    CH2 curr    CH2 cap    CH2 energy
CH2 energy  CH3 volt ...
```

Each channel always returns 4 values, in the following order:

Index	Meaning	Unit	Notes
1	Voltage	V	Real-time voltage
2	Current	A	Positive for charge, negative for discharge, 0 at rest
3	Capacity	Ah	Positive for charge, negative for discharge
4	Energy	Wh	Positive for charge, negative for discharge

- **TC usage example:**

```
#askValues MEASure:ALL? 1,2,3,4
#askValuesReadFormat ffffffffffffffffff
```

- **Full example:**

```
Send:    MEASure:ALL? 1,2,3,4
Return:  4.183e+00,3.510e-01,2.048e-02,8.540e-02,4.202e+00,3.200e-01,2.003e-02,8.411e-02,4.154e+00,3.510e-01,2.046e-02,8.484e-02,3.000e-03,2.000e-03,0.000e+00,0.000e+00
```

3.3.2 MEASure:VOLTage?

- **Type:** Query
- **Syntax:** `MEASure:VOLTage? <ch>`
- **Returns:** `%5.3fv`, e.g., `4.200v`.
- **Example:**

```
Send:    MEASure:VOLTage? Ch1
Return:  4.189V
```

3.3.3 MEASure:CURRent?

- **Type:** Query
- **Syntax:** MEASure:CURRent? <ch>
- **Returns:** %5.3fA. Positive for charge, negative for discharge, 0 at rest.
- **Example:**

```
Send:    MEASure:CURRent? Ch1
Return:  0.351A
```

3.3.4 MEASure:CAPAcity?

- **Type:** Query
- **Syntax:** MEASure:CAPAcity? <ch>
- **Returns:** %5.3fAh. Positive for charge, negative for discharge; at rest, may return the last capacity.
- **Example:**

```
Send:    MEASure:CAPAcity? Ch1
Return:  0.031Ah
```

3.3.5 MEASure:ENERgy?

- **Type:** Query
- **Syntax:** MEASure:ENERgy? <ch>
- **Returns:** %5.3fWh.
- **Example:**

```
Send:    MEASure:ENERgy? Ch1
Return:  0.135Wh
```

3.3.6 MEASure:IR?

- **Type:** Query
- **Syntax:** MEASure:IR? <ch>
- **Returns:** %4u, 4-digit width, right-aligned; short values have leading spaces.
- **Example:**

```
Send:    MEASure:IR? Ch1
Return:   54
```

3.3.7 CHECK:TIME:STAGE?

- **Type:** Query
- **Syntax:** CHECK:TIME:STAGE? <ch>
- **Returns:** HH:MM:SS.
- **Example:**

```
Send:    CHECK:TIME:STAGE? Ch1
Return:  00:06:08
```

3.3.8 CHECK:TIME:TOTAL?

- **Type:** Query
- **Syntax:** CHECK:TIME:TOTAL? <ch>
- **Returns:** D|HH:MM:SS.
- **Example:**

```
Send:    CHECK:TIME:TOTAL? Ch1
Return:  0D|00:06:24
```

3.3.9 CHECK:CYCLE:COUNT?

- **Type:** Query
- **Syntax:** CHECK:CYCLE:COUNT? <ch>
- **Returns:** %2u, 2-digit width, right-aligned; short values have leading spaces.
- **Example:**

```
Send:    CHECK:CYCLE:COUNT? Ch1
Return:   3
```

4. Program Settings (PC Control)

The following commands are used to configure program parameters under PC control. They are heavily used in TC. For detailed usage and scenarios, refer to the complete configuration script in TC.

4.1 Basic Parameters

4.1.1 PROGRAM:SERIALno / PROGRAM:SERIALno?

- **Type:** Set / Query
- **Syntax:**
 - Set: PROGRAM:SERIALno <ch> <num|T|Temp|Temporary>
 - Query: PROGRAM:SERIALno? <ch or number>
- **Description:** num = 0–31; T / Temp / Temporary sets to 0xFF.
- **TC usage example:**

```
#scpiCmd SerialNo tx PROGRAM:SERIALno (selectedChannel),(value)
#scpiCmd SerialNo? txrx? PROGRAM:SERIALno? (selectedChannel)
:readmath:
getElement("0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28
,29,30,31,temp",listIndex(value,"0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21
22 23 24 25 26 27 28 29 30 31 255"));
```

- **Example:**

```
Send:    PROGRAM:SERIALno Ch1,0
Send:    PROGRAM:SERIALno? Ch1
Return:  2    (returns the program number bound to the current channel)
```

4.1.2 PROGRAM:TYPE / PROGRAM:TYPE?

- **Type:** Set / Query

- **Syntax:**

- Set: PROGRAM:TYPE <ch> <type>
- Query: PROGRAM:TYPE? <ch>

- **Type mapping:**

```
Ni / NiMh
Li / LiIon / Lithumion
LF / LiFePO4
Li430 / Liio430
Li435 / Liio435
LD / Li150 / Liio150
```

- **TC usage example:**

```
#scpiCmd Type tx PROGRAM:TYPE (selectedChannel),(value)
#scpiCmd Type? txrx? PROGRAM:TYPE? (selectedChannel)
:readmath: getElement("Ni,Li,LF,Li430,Li435,LD",value-1);
```

- **Example:**

```
Send:    PROGRAM:TYPE Ch1,Li
Send:    PROGRAM:TYPE? Ch1
Return:  2    (mapped to Li)
```

4.1.3 PROGRAM:TASK / PROGRAM:TASK?

- **Type:** Set / Query

- **Syntax:**

- Set: PROGRAM:TASK <ch> <mode>
- Query: PROGRAM:TASK? <ch>

- **Modes:**

```
C / Charge  
D / Discharge  
Cy / Cycle  
Br / Breakin  
Re / Refresh  
St / Storage
```

- **TC usage example:**

```
#scpiCmd Task tx PROGRAM:TASK (selectedChannel),(value)  
#scpiCmd Task? txrx? PROGRAM:TASK? (selectedChannel)  
:readmath: getElement("C,D,Cy,Br,Re,St",value-1);
```

- **Example:**

```
Send:    PROGRAM:TASK Ch1,C  
Send:    PROGRAM:TASK? Ch1  
Return:  1 (mapped to C)
```

4.1.4 PROGRAM:CYCLE:MODE / PROGRAM:CYCLE:MODE?

- **Type:** Set / Query
- **Syntax:** PROGRAM:CYCLE:MODE <ch> <mode>
- **Modes:** CDC / C>D>C, DCD / D>C>D, DC / D>C, CD / C>D
- **TC usage example:**

```
#scpiCmd CycleMode tx PROGRAM:CYCLE:MODE (selectedChannel),(value)  
#scpiCmd CycleMode? txrx? PROGRAM:CYCLE:MODE? (selectedChannel)  
:readmath: getElement("CDC,DCD,DC,CD",value-1);
```

- **Example:**

```
Send:    PROGRAM:CYCLE:MODE Ch1,CDC  
Send:    PROGRAM:CYCLE:MODE? Ch1  
Return:  1 (mapped to CDC)
```

4.1.5 PROGRAM:CYCLE:COUNTs / PROGRAM:CYCLE:COUNTs?

- **Type:** Set / Query
- **Syntax:** PROGRAM:CYCLE:COUNTs <ch> <n>, n <= 10
- **TC usage example:**

```
#scpiCmd CycleCounts tx PROGRAM:CYCLE:COUNTs (selectedChannel),(value)  
#scpiCmd CycleCounts? txrx? PROGRAM:CYCLE:COUNTs? (selectedChannel)
```

- **Example:**

```
Send:    PROGRAM:CYCLE:COUNTs Ch1,3  
Send:    PROGRAM:CYCLE:COUNTs? Ch1  
Return:  3
```

4.1.6 PROGram:CAPACity / PROGram:CAPACity?

- **Type:** Set / Query
- **Syntax:** PROGram:CAPACity <ch> <num>[mAh|mWh], < 10000
- **TC usage example:**

```
#scpiCmd MaxCapacity tx PROGram:CAPACity (selectedChannel),(value)Ah
#scpiCmd MaxCapacity? txrx? PROGram:CAPACity? (selectedChannel)
:readmath: (value/1000.0)
```

- **Example:**

```
Send:    PROGram:CAPACity Ch1,2.0Ah
Send:    PROGram:CAPACity? Ch1
Return:  2000    (displays 2.0Ah)
```

4.2 Current Parameters

4.2.1 PROGram:CURRENT:PRECharge / ?

- **Type:** Set / Query
- **Syntax:** PROGram:CURRENT:PRECharge <ch> <num>[A]
- **TC usage example:**

```
#scpiCmd PrechargeCurrent tx PROGram:CURRENT:PRECharge (selectedChannel),(value)A
#scpiCmd PrechargeCurrent? txrx? PROGram:CURRENT:PRECharge? (selectedChannel)
:readmath: (value/1000.0)
```

- **Example:**

```
Send:    PROGram:CURRENT:PRECharge Ch1,0.1A
Send:    PROGram:CURRENT:PRECharge? Ch1
Return:  100    (displays 0.1A)
```

4.2.2 PROGram:CURRENT:CHARGE / ?

- **Type:** Set / Query
- **Syntax:** PROGram:CURRENT:CHARGE <ch> <num>[A]
- **TC usage example:**

```
#scpiCmd ChargeCurrent tx PROGram:CURRENT:CHARGE (selectedChannel),(value)A
#scpiCmd ChargeCurrent? txrx? PROGram:CURRENT:CHARGE? (selectedChannel)
:readmath: (value/1000.0)
```

- **Example:**

```
Send:    PROGram:CURRENT:CHARGE Ch1,1.0A
Send:    PROGram:CURRENT:CHARGE? Ch1
Return:  1000    (displays 1.0A)
```

4.2.3 PROGram:CURRent:TERMination / ?

- **Type:** Set / Query
- **Syntax:** PROGram:CURRent:TERMination <ch> <num>[A]
- **TC usage example:**

```
#scpiCmd EndChargeCurrent tx PROGram:CURRent:TERMination (selectedChannel),(value)A
#scpiCmd EndChargeCurrent? txrx? PROGram:CURRent:TERMination? (selectedChannel)
:readmath: (value/1000.0)
```

- **Example:**

```
Send:    PROGram:CURRent:TERMination Ch1,0.1A
Send:    PROGram:CURRent:TERMination? Ch1
Return:   100
```

4.2.4 PROGram:CURRent:DISCharge / ?

- **Type:** Set / Query
- **Syntax:** PROGram:CURRent:DISCharge <ch> <num>[A]
- **TC usage example:**

```
#scpiCmd DischargeCurrent tx PROGram:CURRent:DISCharge (selectedChannel),(value)A
#scpiCmd DischargeCurrent? txrx? PROGram:CURRent:DISCharge? (selectedChannel)
:readmath: (value/1000.0)
```

- **Example:**

```
Send:    PROGram:CURRent:DISCharge Ch1,0.5A
Send:    PROGram:CURRent:DISCharge? Ch1
Return:   500
```

4.2.5 PROGram:CURRent:DISCharge:REDUce / ?

- **Type:** Set / Query
- **Syntax:** PROGram:CURRent:DISCharge:REDUce <ch> <num>[A]
- **TC usage example:**

```
#scpiCmd DischargeReduceCurrent tx PROGram:CURRent:DISCharge:REDUce (selectedChannel),
(value)A
#scpiCmd DischargeReduceCurrent? txrx? PROGram:CURRent:DISCharge:REDUce?
(selectedChannel)
:readmath: (value/1000.0)
```

- **Example:**

```
Send:    PROGram:CURRent:DISCharge:REDUce Ch1,0.1A
Send:    PROGram:CURRent:DISCharge:REDUce? Ch1
Return:   100
```

4.3 Voltage Parameters

4.3.1 PROGram:VOLTage:TERMination / ?

- **Type:** Set / Query
- **Syntax:** PROGram:VOLTage:TERMination <ch> <num>[V]
- **TC usage example:**

```
#scpiCmd EndChargeVoltage tx PROGram:VOLTage:TERMination (selectedChannel),(value)V
#scpiCmd EndChargeVoltage? txrx? PROGram:VOLTage:TERMination? (selectedChannel)
:readmath: (value/1000.0)
```

- **Example:**

```
Send:    PROGram:VOLTage:TERMination Ch1,4.2V
Send:    PROGram:VOLTage:TERMination? Ch1
Return:  4200
```

4.3.2 PROGram:VOLTage:TARGet / ?

- **Type:** Set / Query
- **Syntax:** PROGram:VOLTage:TARGet <ch> <num>[V]
- **TC usage example:**

```
#scpiCmd TargetVoltage tx PROGram:VOLTage:TARGet (selectedChannel),(value)V
#scpiCmd TargetVoltage? txrx? PROGram:VOLTage:TARGet? (selectedChannel)
:readmath: (value/1000.0)
```

- **Example:**

```
Send:    PROGram:VOLTage:TARGet Ch1,1.5V
Send:    PROGram:VOLTage:TARGet? Ch1
Return:  1550
```

4.3.3 PROGram:VOLTage:PREChargeToFastcharge / ?

- **Type:** Set / Query
- **Syntax:** PROGram:VOLTage:PREChargeToFastcharge <ch> <num>[V]
- **TC usage example:**

```
#scpiCmd PrechargeToFastcharge tx PROGram:VOLTage:PREChargeToFastcharge
(selectedChannel),(value)V
#scpiCmd PrechargeToFastcharge? txrx? PROGram:VOLTage:PREChargeToFastcharge?
(selectedChannel)
:readmath: (value/1000.0)
```

- **Example:**

```
Send:    PROGram:VOLTage:PREChargeToFastcharge Ch1,3.0V
Send:    PROGram:VOLTage:PREChargeToFastcharge? Ch1
Return:  3000
```

4.3.4 PROGram:VOLTage:REStArt / ?

- **Type:** Set / Query
- **Syntax:** PROGram:VOLTage:REStArt <ch> <num>[V]
- **TC usage example:**

```
#scpiCmd RestartVoltage tx PROGram:VOLTage:REStArt (selectedChannel),(value)V
#scpiCmd RestartVoltage? txrx? PROGram:VOLTage:REStArt? (selectedChannel)
:readmath: (value/1000.0)
```

- **Example:**

```
Send:    PROGram:VOLTage:REStArt Ch1,3.0V
Send:    PROGram:VOLTage:REStArt? Ch1
Return:  3000
```

4.3.5 PROGram:VOLTage:CUT / ?

- **Type:** Set / Query
- **Syntax:** PROGram:VOLTage:CUT <ch> <num>[V]
- **TC usage example:**

```
#scpiCmd DischargeCutVoltage tx PROGram:VOLTage:CUT (selectedChannel),(value)V
#scpiCmd DischargeCutVoltage? txrx? PROGram:VOLTage:CUT? (selectedChannel)
:readmath: (value/1000.0)
```

- **Example:**

```
Send:    PROGram:VOLTage:CUT Ch1,3.0V
Send:    PROGram:VOLTage:CUT? Ch1
Return:  3000
```

4.4 Time Parameters

4.4.1 PROGram:TIME:PRECharge / ?

- **Type:** Set / Query
- **Syntax:** PROGram:TIME:PRECharge <ch> <num>[s]
- **TC usage example:**

```
#scpiCmd MaxPreChargeTime tx PROGram:TIME:PRECharge (selectedChannel),(value)Min
#scpiCmd MaxPreChargeTime? txrx? PROGram:TIME:PRECharge? (selectedChannel)
```

- **Example:**

```
Send:    PROGram:TIME:PRECharge Ch1,60Min
Send:    PROGram:TIME:PRECharge? Ch1
Return:  60
```

4.4.2 PROGram:TIME:CHARge / ?

- **Type:** Set / Query
- **Syntax:** PROGram:TIME:CHARge <ch> <num>[s]
- **TC usage example:**

```
#scpiCmd MaxChargeTime tx PROGram:TIME:CHARge (selectedChannel),(value)Min  
#scpiCmd MaxChargeTime? txrx? PROGram:TIME:CHARge? (selectedChannel)
```

- **Example:**

```
Send:    PROGram:TIME:CHARge Ch1,120Min  
Send:    PROGram:TIME:CHARge? Ch1  
Return:  120
```

4.4.3 PROGram:TIME:CUT / ?

- **Type:** Set / Query
- **Syntax:** PROGram:TIME:CUT <ch> <num>[s]
- **TC usage example:**

```
#scpiCmd MaxDischargeTime tx PROGram:TIME:CUT (selectedChannel),(value)Min  
#scpiCmd MaxDischargeTime? txrx? PROGram:TIME:CUT? (selectedChannel)
```

- **Example:**

```
Send:    PROGram:TIME:CUT Ch1,120Min  
Send:    PROGram:TIME:CUT? Ch1  
Return:  120
```

4.4.4 PROGram:TIME:RESTing:CHARge / ?

- **Type:** Set / Query
- **Syntax:** PROGram:TIME:RESTing:CHARge <ch> <num>[s]
- **TC usage example:**

```
#scpiCmd ChargeRestingTime tx PROGram:TIME:RESTing:CHARge (selectedChannel),(value)Min  
#scpiCmd ChargeRestingTime? txrx? PROGram:TIME:RESTing:CHARge? (selectedChannel)
```

- **Example:**

```
Send:    PROGram:TIME:RESTing:CHARge Ch1,10Min  
Send:    PROGram:TIME:RESTing:CHARge? Ch1  
Return:  15
```

4.4.5 PROGram:TIME:RESTing:DISCharge / ?

- **Type:** Set / Query
- **Syntax:** PROGram:TIME:RESTing:DISCharge <ch> <num>[s]
- **TC usage example:**

```
#scpiCmd DischargeRestingTime tx PROGram:TIME:RESTing:DISCharge (selectedChannel),  
(value)Min  
#scpiCmd DischargeRestingTime? txrx? PROGram:TIME:RESTing:DISCharge? (selectedChannel)
```

- **Example:**

```
Send:    PROGram:TIME:RESTing:DISCharge Ch1,10Min  
Send:    PROGram:TIME:RESTing:DISCharge? Ch1  
Return:  15
```

4.5 Names and Others

4.5.1 PROGram:DELTApeak / ?

- **Type:** Set / Query
- **Syntax:** PROGram:DELTApeak <ch> <Advice|num[mV|V]>
- **TC usage example:**

```
#scpiCmd DeltaPeak tx PROGram:DELTApeak (selectedChannel),(value)mV  
#scpiCmd DeltaPeak? txrx? PROGram:DELTApeak? (selectedChannel)
```

- **Example:**

```
Send:    PROGram:DELTApeak Ch1,5mV  
Send:    PROGram:DELTApeak? Ch1  
Return:  5
```

4.5.2 PROGram:LONGname / ?

- **Type:** Set / Query
- **Syntax:** PROGram:LONGname <ch> <string>, maximum 16 characters (excluding terminator).
- **TC usage example:**

```
#scpiCmd LongName tx PROGram:LONGname (selectedChannel),"(value)"  
#scpiCmd LongName? txrx? PROGram:LONGname? (selectedChannel)
```

- **Example:**

```
Send:    PROGram:LONGname Ch1,"MyCell"  
Send:    PROGram:LONGname? Ch1  
Return:  CHLI↑1.00A┐4.20V
```

Note: Special characters in the return value (↑ ↓ ┐ ┌, etc.) may appear as  in some terminals. Use `replaceCharsR()` to restore them when writing.

4.5.3 PROGram:SHORTname / ?

- **Type:** Set / Query
- **Syntax:** PROGram:SHORTname <ch> <string>, maximum 10 characters (excluding terminator).
- **TC usage example:**

```
#scpiCmd ShortName tx PROGram:SHORTname (selectedChannel),"(value)"  
#scpiCmd ShortName? txrx? PROGram:SHORTname? (selectedChannel)
```

- **Example:**

```
Send:    PROGram:SHORTname Ch1,"MC"  
Send:    PROGram:SHORTname? Ch1  
Return:  CHLI↑1.00A
```

5. Program Control Flow

5.1 CONTro1:PROGram:PREProcess / ?

- **Type:** Action / Query
- **Syntax:**
 - CONTro1:PROGram:PREProcess <ch>
 - CONTro1:PROGram:PREProcess? <ch>
- **Returns:** Ready or Not ready
- **TC usage example:**

```
#scpiCmd Preprocess tx CONTro1:PROGram:PREProcess (selectedChannel)  
#scpiCmd Preprocess? txrx? CONTro1:PROGram:PREProcess? (selectedChannel)
```

- **Example:**

```
Send:    CONTro1:PROGram:PREProcess Ch1  
Send:    CONTro1:PROGram:PREProcess? Ch1  
Return:  Not ready
```

5.2 CONTro1:PROGram:SAVE

- **Type:** Action
- **Syntax:** CONTro1:PROGram:SAVE <ch>
- **Requirement:** Preprocessing Ready. Whether it writes to Flash or to temporary parameters is determined by SETUP_SERIAL_NO, set through PROGram:SERIa1no.
- **TC usage example:**

```
#scpiCmd SaveCurrentProgram tx CONTro1:PROGram:SAVE (selectedChannel)
```

- **Example:**

```
Send:    CONTro1:PROGrama:SAVE  ch1
Return:  (none)
```

5.3 CONTro1:PROGrama:COMPlated

- **Type:** Action
- **Syntax:** CONTro1:PROGrama:COMPlated <ch>
- **TC usage example:**

```
#scpiCmd ExitProgramConfiguration tx CONTro1:PROGrama:COMPlated (selectedChannel)
```

- **Example:**

```
Send:    CONTro1:PROGrama:COMPlated ch1
Return:  (none)
```

5.4 PROGrama:SETUp

- **Type:** Action
- **Syntax:** PROGrama:SETUp <ch> <program 0~31>
- **TC usage example:**

```
#scpiCmd SelectAProgramToConfigure tx PROGrama:SETUp (selectedChannel),(value)
```

- **Example:**

```
Send:    PROGrama:SETUp ch1,0
Return:  (none)
```

5.5 CONTro1:PROGrama:RUNning

- **Type:** Action
- **Syntax:** CONTro1:PROGrama:RUNning <ch> <program 0~31>
- **Example:**

```
Send:    CONTro1:PROGrama:RUNning ch1,0
Return:  (none)
```

5.6 CONTro1:TRANSferChannelControlToPC / ?

- **Type:** Action / Query
- **Syntax:**
 - CONTro1:TRANSferChannelControlToPC <ch> <bool>
 - CONTro1:TRANSferChannelControlToPC? <ch>
- **Returns:** Transferred or Not transferred
- **TC usage example:**

```
CONTROL:TRANSFERChannelControlToPC (channel),(runNow);[200];  
# if (deviceRead(device,command_TRANSFERChannelControlToPCQ) == "Transferred")
```

- **Example:**

```
Send:    CONTROL:TRANSFERChannelControlToPC Ch1,Yes  
Send:    CONTROL:TRANSFERChannelControlToPC? Ch1  
Return:  Transferred
```

5.7 CONTROL:PROCESS:START

- **Type:** Action
- **Syntax:** CONTROL:PROCESS:START <ch>
- **TC usage example:**

```
CONTROL:PROCESS:START (channel);[50];
```

- **Example:**

```
Send:    CONTROL:PROCESS:START Ch1  
Return:  (none)
```

5.8 STATUS:CONTROL:PROCESS:COMPLETED?

- **Type:** Query
- **Syntax:** STATUS:CONTROL:PROCESS:COMPLETED? <ch>
- **Returns:** Completed or CH_x status: n
- **TC usage example:**

```
# while (deviceRead(device,command_ProcessCompletedQ) != "Completed")
```

- **Example:**

```
Send:    STATUS:CONTROL:PROCESS:COMPLETED? Ch1  
Return:  Completed
```

5.9 CONTROL:RELEASE

- **Type:** Action
- **Syntax:** CONTROL:RELEASE <ch>
- **TC usage example:**

```
CONTROL:RELEASE (channel);[50];
```

- **Example:**

```
Send:    CONTROL:RELEASE Ch1  
Return:  (none)
```

5.10 CONTROL:RESTing

- **Type:** Set
- **Syntax:** CONTROL:RESTing <ch> <time>[s], ≤1000 minutes.
- **TC usage example:**

```
CONTROL:RESTing (channel),(restingTime)min;[50];
```

- **Example:**

```
Send:    CONTROL:RESTing Ch1,10min
Return:  (none)
```

5.11 CONTROL:ACTivatebattery:START / EXIT

- **Type:** Action
- **Syntax:** CONTROL:ACTivatebattery:START <ch> / CONTROL:ACTivatebattery:EXIT <ch>
- **Example:**

```
Send:    CONTROL:ACTivatebattery:START Ch1
Send:    CONTROL:ACTivatebattery:EXIT Ch1
Return:  (none)
```

6. Super Control

6.1 SUPER:CONTROL:RUN / STOP

- **Type:** Action
- **Syntax:** SUPER:CONTROL:RUN <ch> / SUPER:CONTROL:STOP <ch>
- **TC usage example (CH1):**

```
#cmdSetup button Start_Process CH1_Super_Controller
:write:SUPER:CONTROL:RUN Ch1;[200];
:update:CH1_Super_Controller.Choose_A_Program_To_Run CH1_Super_Controller.Battery_Type
CH1_Super_Controller.Task CH1_Super_Controller.Nothing
:tip:The button start the program.

#cmdSetup button Stop_Process CH1_Super_Controller
:write:SUPER:CONTROL:STOP Ch1;[200];
:tip:The button stop the program.
```

- **Example:**

```
Send:    SUPER:CONTROL:RUN Ch1
Send:    SUPER:CONTROL:STOP Ch1
Return:  (none)
```

6.2 SUPER:PROGrama:SETUp

- **Type:** Set
- **Syntax:** `SUPER:PROGrama:SETUp <ch> <program 0~31>`
- **TC usage example:**

```
#cmdSetup combobox Choose_A_Program_To_Run CH1_Super_Controller
:write: SUPER:PROGrama:SETUp Ch1,#;[200]
:read: PROGrama:SERIALno? Ch1 ;
:readmath:
getElement("0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28
,29,30,31,temp",listIndex(value,"0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21
22 23 24 25 26 27 28 29 30 31 255"));
:update: CH1_Super_Controller.Battery_Type CH1_Super_Controller.Task
CH1_Super_Controller.Nothing CH1_Super_Controller.End_Charge_Current
CH1_Super_Controller.End_Charge_Voltage CH1_Super_Controller.Target_Voltage
CH1_Super_Controller.Delta_Peak_Voltage
:tip:Select a program to run. Subsequent options allow real-time adjustments to key
configurations, with limited safety checks by the charger.
:string:
Program_0 0
Program_1 1
...
Temporary temp
```

- **Example:**

```
Send:    SUPER:PROGrama:SETUp Ch1,0
Return:  (none)
```

6.3 SUPER:PROGrama:TYPE

- **Type:** Set
- **Syntax:** `SUPER:PROGrama:TYPE <ch> <type>`
- **TC usage example:**

```
#cmdSetup combobox Battery_Type CH1_Super_Controller
:write:SUPE:PROGrama:TYPE ch1, #;[200];
:read: PROGrama:TYPE? ch1;
:readmath: getElement("Ni,Li,LF,Li430,Li435,LD",value-1);
:update: CH1_Super_Controller.Task CH1_Super_Controller.Choose_A_Program_To_Run
:tip:Modifying this configuration is not recommended.
:string:
NiMh/Cd Ni
LiionStd Li
LiFePO4 LF
Liion430 Li430
Liion435 Li435
Liion150 LD
```

Note: The `SUPE:` in the TC example is verbatim from the configuration file; it is equivalent to `SUPER:` (the SCPI parser matches short forms).

- **Example:**

```
Send:    SUPe:PROGrama:TYPE Ch1,Li
Return:  (none)
```

6.4 SUPe:PROGrama:TASK

- **Type:** Set
- **Syntax:** SUPe:PROGrama:TASK <ch> <mode>
- **TC usage example:**

```
#cmdSetup combobox Task CH1_Super_Controller
:write:SUPe:PROGrama:TASK ch1,#; [200];
:read: PROGrama:TASK? ch1;
:readmath: getElement("C,D,Cy,Br,Re,St",value-1);
:tip:Modifying this configuration is not recommended.
:string:
Charge C
Discharge D
Cycle Cy
Refresh Re
Storage St
:update:CH1_Super_Controller.Nothing CH1_Super_Controller.Choose_A_Program_To_Run
:visible: inList(CH1_Super_Controller.Battery_Type,"Li LF Li430 Li435")
```

- **Example:**

```
Send:    SUPe:PROGrama:TASK Ch1,C
Return:  (none)
```

6.5 SUPe:PROGrama:CURRENT:*

Includes:

- SUPe:PROGrama:CURRENT:PRECharge
- SUPe:PROGrama:CURRENT:CHARGE
- SUPe:PROGrama:CURRENT:TERMination
- SUPe:PROGrama:CURRENT:DISCharge
- SUPe:PROGrama:CURRENT:DISCharge:REDUce
- **Syntax:** SUPe:PROGrama:CURRENT:XXX <ch> <num>[A]
- **TC usage example (partial):**

```
#cmdSetup number Precharge_Current CH1_Super_Controller
:write: SUPe:PROGrama:CURRENT:PRECharge ch1,#A; [200];
:read: PROGrama:CURRENT:PRECharge? ch1;
:readmath: (value/1000.0)
:update:CH1_Super_Controller.Choose_A_Program_To_Run
A 0.030 2.000
:visible:inList(CH1_Super_Controller.Battery_Type, "Ni Li LF Li430 Li435")

#cmdSetup number Charge_Current CH1_Super_Controller
```

```
:write: SUPE:PROGrama:CURRENT:CHARGE ch1,#A; [200];
:read: PROGrama:CURRENT:CHARGE? ch1;
:readmath: (value/1000.0)
:update:CH1_Super_Controller.Choose_A_Program_To_Run
A 0.030 2.000
:visible:inList(CH1_Super_Controller.Battery_Type, "Ni Li LF Li430 Li435")
```

- **Example:**

```
Send:    SUPER:PROGrama:CURRENT:CHARGE ch1,1.0A
Send:    SUPER:PROGrama:CURRENT:DISCHARGE ch1,0.5A
Return:  (none)
```

6.6 SUPER:PROGrama:VOLTage:*

Includes:

- SUPER:PROGrama:VOLTage:TERMination
- SUPER:PROGrama:VOLTage:TARGet
- SUPER:PROGrama:VOLTage:PREChargeToFastCharge
- SUPER:PROGrama:VOLTage:CUT

Note: There is **no** REStArT under SUPER:PROGrama:VOLTage. To set restart voltage, use PROGrama:VOLTage:REStArT.

- **Syntax:** SUPER:PROGrama:VOLTage:XXX <ch> <num>[V]
- **TC usage example (partial):**

```
#cmdSetup number End_Charge_Voltage CH1_Super_Controller
:write: SUPE:PROGrama:VOLTage:TERMination ch1,#V; [200];
:read: PROGrama:VOLTage:TERMination? ch1;
:readmath: (value/1000.0)
:update:CH1_Super_Controller.Choose_A_Program_To_Run
V 0.030 4.850
:visible:inList(CH1_Super_Controller.Battery_Type, "Li LF Li430 Li435")
```

- **Example:**

```
Send:    SUPER:PROGrama:VOLTage:TERMination ch1,4.2V
Send:    SUPER:PROGrama:VOLTage:CUT ch1,3.0V
Return:  (none)
```

6.7 SUPER:PROGrama:DELTApeak

- **Type:** Set
- **Syntax:** SUPER:PROGrama:DELTApeak <ch> <Advice|num[mV|V]>
- **TC usage example:**

```
#cmdSetup combobox Delta_Peak_Voltage CH1_Super_Controller
:write: SUPER:PROGrama:DELTApeak ch1,#; [200];
:read: PROGrama:DELTApeak? ch1;
:update:CH1_Super_Controller.Choose_A_Program_To_Run
:string:
```

```

0dv 0
1_mv 1
2_mv 2
3_mv 3
4_mv 4
5_mv 5
6_mv 6
7_mv 7
8_mv 8
9_mv 9
Advice 10
:visible: CH1_Super_Controller.Battery_Type=="Ni"

```

- **Example:**

```

Send:    SUPER:PROGrama:DELTApeak Ch1,5mv
Send:    SUPER:PROGrama:DELTApeak Ch1,Advice
Return:  (none)

```

6.8 SUPER:CONTROL:ACTivatebattery:START / EXIT

- **Type:** Action
- **Syntax:** SUPER:CONTROL:ACTivatebattery:START <ch> / EXIT <ch>
- **TC usage example:**

```

#cmdSetup button Start_Battery_Activation_Process CH1_Super_Controller
:write:SUPER:CONTROL:ACTivatebattery:START CH1;[200];
:tip:Protected cells with Low Voltage Protection triggered are generally safe. But
activation is not recommended for unprotected over-discharged cells. The process lasts 2
minutes.

#cmdSetup button Exit_Battery_Activation_Process CH1_Super_Controller
:write:SUPER:CONTROL:ACTivatebattery:EXIT CH1;[200];
:tip:Terminate the battery activation process early. After 2 minutes, the charger will
automatically end the process.

```

- **Example:**

```

Send:    SUPER:CONTROL:ACTivatebattery:START Ch1
Send:    SUPER:CONTROL:ACTivatebattery:EXIT Ch1
Return:  (none)

```

7. GSV Global Configuration

7.1 CONFIG:FANS:ENABLE / ?

- **Type:** Set / Query
- **Syntax:** CONFIG:FANS:ENABLE <bool> / CONFIG:FANS:ENABLE?
- **TC usage example:**

```
#cmdSetup buttonson Fans Global_Settings
:write: CONFIG:FANS:ENABLE #; [100]
:read: CONFIG:FANS:ENABLE?;
:readmath: getElement("OFF,ON", value );
:tip: Toggle the fans On or Off
:string:
Off OFF
On ON
:update: Global_Settings.Fans_Status
```

- **Example:**

```
Send:    CONFIG:FANS:ENABLE ON
Send:    CONFIG:FANS:ENABLE?
Return:  1  (displays ON)
```

7.2 CONFIG:BEEPer:KEY:ENABLE / ?

- **Type:** Set / Query
- **TC usage example:**

```
#cmdSetup buttonson Button_Beeper Global_Settings
:write: CONFIG:BEEPer:KEY:ENABLE #; [100]
:read:CONFIG:BEEPer:KEY:ENABLE?;
:readmath: getElement("OFF,ON", value );
:tip: Toggle the key beeper On or off
:string:
Off OFF
On ON
```

- **Example:**

```
Send:    CONFIG:BEEPer:KEY:ENABLE ON
Send:    CONFIG:BEEPer:KEY:ENABLE?
Return:  1
```

7.3 CONFIG:BEEPer:PROcesscompletion:ENABLE / ?

- **Type:** Set / Query
- **TC usage example:**

```
#cmdSetup buttonson Process_Completion_Beeper Global_Settings
:write: CONFIG:BEEPer:PROcesscompletion:ENABLE #; [100]
:read:CONFIG:BEEPer:PROcesscompletion:ENABLE?;
:readmath: getElement("OFF,ON", value );
:tip: Toggle the Process completion beeper On or off
:string:
Off OFF
On ON
```

- **Example:**

```
Send:    CONFIg:BEEPer:PROCEsscompletion:ENABle ON
Send:    CONFIg:BEEPer:PROCEsscompletion:ENABle?
Return:  1
```

7.4 CONFIg:SCREenflash:ENABle / ?

- **Type:** Set / Query
- **TC usage example:**

```
#cmdSetup buttonson Process_Completion_Screen_Flash Global_Settings
:write: CONFIg:SCREenflash:ENABle #; [100]
:read:CONFIg:SCREenflash:ENABle?;
:readmath: getElement("NO,YES", value );
:tip: Toggle the Process completion beeper On or off
:string:
No NO
Yes YES
```

- **Example:**

```
Send:    CONFIg:SCREenflash:ENABle YES
Send:    CONFIg:SCREenflash:ENABle?
Return:  1
```

7.5 CONFIg:LISPfn:ENABle / ?

- **Type:** Set / Query
- **Description:** Li-ion strong protection; inverted logic.
- **TC usage example:**

```
#cmdSetup buttonson LiIon&LifeP04_Strong_Protect Global_Settings
:read: CONFIg:LISPfn:ENABle?
:readmath: getElement("OFF ON",value);
:write: CONFIg:LISPfn:ENABle #; [100]
:tip: Charging won't start for liion batteries below 2v when LISP is enabled.
:string:
Off OFF
On ON
:update: Global_Settings.LiIon&LifeP04_Strong_Protect_Status
```

- **Example:**

```
Send:    CONFIg:LISPfn:ENABle ON
Send:    CONFIg:LISPfn:ENABle?
Return:  1
```

7.6 CONFIG:LDSPfn:ENABLE / ?

- **Type:** Set / Query
- **Description:** 1.5V Li-ion strong protection.
- **TC usage example:**

```
#cmdSetup buttonson 1.50V_LiIon_Strong_Protect Global_Settings
:read: CONFIG:LDSPfn:ENABLE?
:readmath: getElement("OFF ON",value);
:write: CONFIG:LDSPfn:ENABLE #; [100]
:tip: Charging won't start for 1.5V liion batteries above 2V when LISP is enabled.
:string:
Off OFF
On ON
:update: Global_Settings.1.50V_LiIon_Strong_Protect_Status
```

- **Example:**

```
Send:    CONFIG:LDSPfn:ENABLE ON
Send:    CONFIG:LDSPfn:ENABLE?
Return:  1
```

7.7 CONFIG:AUTOstart:ENABLE / ?

- **Type:** Set / Query
- **TC usage example:**

```
#cmdSetup buttonson Enable_Auto-start_Program Global_Settings
:read: CONFIG:AUTOstart:ENABLE?
:readmath: getElement("NO YES",value);
:write: CONFIG:AUTOstart:ENABLE #; [100]
:tip: Allows auto-start of the program. The program will not auto-start on the first
power-up.
:string:
No NO
Yes YES
:update: Global_Settings.Enable_Auto-start_Program_Status
```

- **Example:**

```
Send:    CONFIG:AUTOstart:ENABLE YES
Send:    CONFIG:AUTOstart:ENABLE?
Return:  1
```

7.8 CONFIG:CALibration:ENABLE / ?

- **Type:** Set / Query
- **TC usage example:**

```
#cmdSetup buttonson Enable_Calibration Global_Settings
:read: CONFIG:CALibration:ENABLE?
:readmath: getElement("NO YES",value);
:write: CONFIG:CALibration:ENABLE #; [100]
:tip: Allow calibration of voltage, charge current, and discharge current.
:string:
No NO
Yes YES
:update: Global_Settings.Enable_Calibration_Status
```

- **Example:**

```
Send:    CONFIG:CALibration:ENABLE YES
Send:    CONFIG:CALibration:ENABLE?
Return:  1
```

7.9 CONFIG:TESTir:ENABLE / ?

- **Type:** Set / Query
- **TC usage example:**

```
#cmdSetup buttonson Enable_Test_IR Global_Settings
:read: CONFIG:TESTir:ENABLE?
:readmath: getElement("NO YES",value);
:write: CONFIG:TESTir:ENABLE #; [100]
:tip: Enable battery internal resistance test? Measurements are for reference only.
:string:
No NO
Yes YES
:update: Global_Settings.Enable_Test_IR_Status
```

- **Example:**

```
Send:    CONFIG:TESTir:ENABLE YES
Send:    CONFIG:TESTir:ENABLE?
Return:  1
```

7.10 CONFIG:ACTivatebattery:ENABLE / ?

- **Type:** Set / Query
- **TC usage example:**

```
#cmdSetup buttonson Enable_0V_Battery_Activation Global_Settings
:read: CONFIG:ACTivatebattery:ENABLE?
:readmath: getElement("NO YES",value);
:write: CONFIG:ACTivatebattery:ENABLE #; [100]
:tip: Enable 0V battery activation?Activating over-discharged lithium-ion batteries is
not recommended.
:string:
No NO
Yes YES
:update: Global_Settings.Enable_0V_Battery_Activation_Status
```

- **Example:**

```
Send:    CONFIg:ACTIvatebattery:ENABle YES
Send:    CONFIg:ACTIvatebattery:ENABle?
Return:  1
```

7.11 CONFIg:DEFAuLtprogram / ?

- **Type:** Set / Query
- **Syntax:** CONFIg:DEFAuLtprogram <ch> <program 0~31>
- **TC usage example (CH1):**

```
#cmdSetup combobox CH1_Default_Program Global_Settings
:write: CONFIg:DEFAuLtprogram Ch1,#:[100]
:read: CONFIg:DEFAuLtprogram? Ch1;
:tip: The default program is loaded when the charger is powered on or when the "Restore
Default Program" operation is performed.
:string:
Program_0 0
Program_1 1
...
Program_31 31
```

- **Example:**

```
Send:    CONFIg:DEFAuLtprogram Ch1,0
Send:    CONFIg:DEFAuLtprogram? Ch1
Return:  0
```

8. Calibration, Backup, and System Reset

8.1 PACKet:RATIo:RUN?

- **Type:** Query
- **Syntax:** PACKet:RATIo:RUN? <ch>
- **Returns:** PACKet:RATIo #<len><data>,<ch>,<checksum>\r\n
- **Example:**

```
Send:    PACKet:RATIo:RUN? Ch1
Return:  PACKet:RATIo
#31611.5465085506,0.9604977369,0.3318018615,1.0000000000,3.2911419868,0.0024977922,1.000
0000000,3.2911419868,0.0024977922,1.0000000000,3.3124411106,-0.0238245726,1,28
```

8.2 PACKet:RATIo:BACKup?

- **Type:** Query
- **Syntax:** PACKet:RATIo:BACKup? <ch>
- **Returns:** PACKet:RATIo #<len><data>,<ch>,<checksum>\r\n
- **Example:**

```
Send:    PACKet:RATIo:BACKup? Ch1
Return:  PACKet:RATIo
#31611.5465085506,0.9604977369,0.3318018615,1.0000000000,3.2911419868,0.0024977922,1.000
0000000,3.2911419868,0.0024977922,1.0000000000,3.3124411106,-0.0238245726,1,28
```

8.3 PACKet:RATIo

- **Type:** Set
- **Syntax:** `PACKet:RATIo <block>`
- **Description:** Parses 14 calibration parameters and writes them to Flash. ⚠ **Dangerous.**
- **Actual calibration parameter download script example** (from `Gyrfalcon_S4000Pro_calibration_parameter_download_Script_*.txt`):

```
CONTRol:PROGram:PREProcess (channel);[200];
#if (deviceRead(device,command_PreprocessBeforeSettingProgramQ) == "Ready")
PACKet:RATIo
#31611.5465085506,0.9604977369,0.3318018615,1.0000000000,3.2911419868,0.0024977922,1.000
0000000,3.2911419868,0.0024977922,1.0000000000,3.3124411106,-0.0238245726,1,28;[200];
PACKet:RATIo
#31611.5465085506,0.9604977369,0.3318018615,1.0000000000,3.2911419868,0.0024977922,1.000
0000000,3.2911419868,0.0024977922,1.0000000000,3.3124411106,-0.0238245726,2,27;[200];
PACKet:RATIo
#31611.5465085506,0.9604977369,0.3318018615,1.0000000000,3.2911419868,0.0024977922,1.000
0000000,3.2911419868,0.0024977922,1.0000000000,3.3124411106,-0.0238245726,3,26;[200];
PACKet:RATIo
#31611.5465085506,0.9604977369,0.3318018615,1.0000000000,3.2911419868,0.0024977922,1.000
0000000,3.2911419868,0.0024977922,1.0000000000,3.3124411106,-0.0238245726,4,25;[200];
CONTRol:PROGram:COMPleted (channel);[50];
#endif
```

Note: Each channel is written with one set of calibration parameters. The last two fields are the channel number and the checksum. Channels 1–4 use checksums 28, 27, 26, and 25 respectively.

Special case: In this script, `channel = "None"`, and `CONTRol:PROGram:PREProcess None` and `CONTRol:PROGram:COMPleted None` are both legal (they accept `channel_no = 0`).

8.4 :SYSTem:BACKup:DATA

- **Type:** Action
- **Syntax:** `:SYSTem:BACKup:DATA`
- **Description:** Backs up data to the backup region. ⚠ **Dangerous.**
- **Example:**

```
Send:    :SYSTem:BACKup:DATA
Return:  (none)
```

8.5 :SYSTem:RESEt:PROGrama

- **Type:** Action
- **Syntax:** :SYSTem:RESEt:PROGrama
- **Description:** Resets programs 0–31 to factory defaults, jumps to Bootloader. ⚠ **Dangerous.**
- **Example:**

```
Send:      :SYSTem:RESEt:PROGrama
Return:    (none)
```

8.6 :SYSTem:RESEt:DATA

- **Type:** Action
- **Syntax:** :SYSTem:RESEt:DATA <bank1|bank2|1|2>
- **Description:** Restores backup data, jumps to Bootloader. ⚠ **Dangerous.**
- **Example:**

```
Send:      :SYSTem:RESEt:DATA 1
Return:    (none)
```

8.7 :SYSTem:RESEt:CALIbration

- **Type:** Action
- **Syntax:** :SYSTem:RESEt:CALIbration
- **Description:** Restores calibration parameters from backup and reboots. ⚠ **Dangerous.**
- **Example:**

```
Send:      :SYSTem:RESEt:CALIbration
Return:    (none)
```

9. Typical Complete Workflow

9.1 Control Script Template

Use case: Execute multiple Process steps in sequence, such as "discharge → rest → charge → rest → discharge".

Key steps:

1. CONTro1:TRANSferChannelContro1ToPC — transfer control
2. Paste configuration commands before each Process
3. CONTro1:PROcess:START — start
4. STATUS:CONTro1:PROcess:COMPLeted? — poll for completion
5. CONTro1:RESTing — rest
6. CONTro1:RELEase — release control

```
#scriptMenu Gyrfalcon_S4000Pro_Control_Script_Template_20261002124904
```

```

#logcmds 0
=var device=getDevice("Charger")
(device):
=var channel="Ch1";
=var runNow="Yes";
=var restingTime=10;

=var command_TRANSFERChannelControlToPCQ= "CONTROL:TRANSFERChannelControlToPC? "+channel
=var command_ProcessCompletedQ= "STATUS:CONTROL:PROCESS:COMPLETED? "+channel

;;Start

CONTROL:TRANSFERChannelControlToPC (channel),(runNow);[200];

#if (deviceRead(device,command_TRANSFERChannelControlToPCQ) == "Transferred")

;;*****Process 1*****

;;The content generated by the "Configuration File for Control Scripts"
;;can be directly copied. Please refer to the format specified in the reference
;;documentation.

;;- - - - - For example - - - - -
;;PROGRAM:TYPE (channel),Li;[50];
;;PROGRAM:TASK (channel),D;[50];
;;PROGRAM:CURRENT:DISCharge (channel),1.0A;[50];
;;PROGRAM:TIME:CUT (channel),600min;[50];
;;PROGRAM:CURRENT:DISCharge:REDUce (channel),0.4A;[50];
;;PROGRAM:VOLTage:CUT (channel),3.3V;[50];

;;CONTROL:PROCESS:START (channel);[50];
;;- - - - - Example end - - - - -

;;*****
*

#while (deviceRead(device,command_ProcessCompletedQ) != "Completed")
=sleep(5000)
#endwhile

CONTROL:RESTing (channel),(restingTime)min;[50];

;;*****Process 2*****

;;The content generated by the "Configuration File for Control Scripts"
;;can be directly copied. Please refer to the format specified in the reference
;;documentation.

;;*****
*

#while (deviceRead(device,command_ProcessCompletedQ) != "Completed")
=sleep(5000)
#endwhile

CONTROL:RESTing (channel),(restingTime)min;[50];

```

```
;;*****Process 3*****

;;The content generated by the "Configuration File for Control scripts"
;;can be directly copied. Please refer to the format specified in the reference
;;documentation.

;;*****
*

#while (deviceRead(device,command_ProcessCompletedQ) != "Completed")
=sleep(5000)
#endwhile

CONTROL:RELEase (channel);[50];
#endif

;;Finished
```

9.2 Configuration Copy Script Template

Use case: Batch write configurations for programs 0–31. Program 0 is discharge, programs 1, 2, and 4–31 are charge, and program 3 is NiMH discharge.

```
#scriptMenu Gyrfalcon_S4000Pro_Configuration_Copy_Script_20261002124620

#logcmds 0
=var device=getDevice("Charger")
(device):
=var channel = "None"
=var command_PreprocessBeforeSettingProgramQ = "CONTROL:PROGrama:PREProcess? " + channel

=popupShowProgressAsync("Configuring GYRFALCON S4000Pro programs",32);

;;Start

CONTROL:PROGrama:PREProcess (channel);[200];

#if (deviceRead(device,command_PreprocessBeforeSettingProgramQ) == "Ready")

;;*****
;;Details of program 0

;;Battery Type: LiionStd
;;Task: Discharge
;;Discharge Current: 0.95A
;;Max Discharge Time: 600min
;;Discharge Reduce Current: 0.1A
;;Discharge Cut Voltage: 2.8V
;;Program Long Name: DILI↓0.95A↓2.80V
;;Program Short Name: DILI↓0.95A

;;Configuration Script of Program 0

=popupShowProgressSetPosition(0);

PROGrama:SERIALno (channel),0;[50];
```

```

PROGrama:TYPE (channel),Li;[50];
PROGrama:TASK (channel),D;[50];
PROGrama:CURRENT:DISCharge (channel),0.95A;[50];
PROGrama:TIME:CUT (channel),600min;[50];
PROGrama:CURRENT:DISCharge:REDUce (channel),0.1A;[50];
PROGrama:VOLTage:CUT (channel),2.8V;[50];
program:LongName
(replaceCharsR("DILI↓0.95A↓2.80V", "\x7f\u2191\u2193\u29bb\u23b2\u2195\u22A5\u2126\u250d\u2515"
));[50];
program:ShortName
(replaceCharsR("DILI↓0.95A", "\x7f\u2191\u2193\u29bb\u23b2\u2195\u22A5\u2126\u250d\u2515"));
[50];

CONTroL:PROGrama:SAVE (channel);[50];

;;*****
;;Details of program 1

;;Battery Type: LiionStd
;;Task: Charge
;;Max Capacity: 5500mAh
;;Precharge Current: 0.1A
;;Max Precharge Time: 180min
;;Precharge To Fast Charge Voltage: 3.0V
;;Charge Current: 1.0A
;;Max Charge Time: 600min
;;End Charge Current: 0.1A
;;End Charge Voltage: 4.2V
;;Restart Voltage: 0.0V
;;Program Long Name: CHLI↑1.00A↑4.20V
;;Program Short Name: CHLI↑1.00A

;;Configuration Script of Program 1

=popupShowProgresssetPosition(1);

PROGrama:SERIALno (channel),1;[50];
PROGrama:TYPE (channel),Li;[50];
PROGrama:TASK (channel),C;[50];
PROGrama:CAPAcity (channel),5500mAh;[50];
PROGrama:CURRENT:PRECharge (channel),0.1A;[50];
PROGrama:TIME:PRECharge (channel),180min;[50];
PROGrama:VOLTage:PREChargetoFastcharge (channel),3.0V;[50];
PROGrama:CURRENT:CHARGE (channel),1.0A;[50];
PROGrama:TIME:CHARGE (channel),600min;[50];
PROGrama:CURRENT:TERMination (channel),0.1A;[50];
PROGrama:VOLTage:TERMination (channel),4.2V;[50];
PROGrama:VOLTage:REStArt (channel),0.0V;[50];
program:LongName
(replaceCharsR("CHLI↑1.00A↑4.20V", "\x7f\u2191\u2193\u29bb\u23b2\u2195\u22A5\u2126\u250d\u2515"
));[50];
program:ShortName
(replaceCharsR("CHLI↑1.00A", "\x7f\u2191\u2193\u29bb\u23b2\u2195\u22A5\u2126\u250d\u2515"));
[50];

CONTroL:PROGrama:SAVE (channel);[50];

```

```
;; Additional programs (2, 3, ... up to 31) can be added similarly

;;*****

CONTRol:PROGrama:COMPleted (channel);[50];
=popupShowProgressSetPosition(100);

#endif

;;Finished
```

9.3 Calibration Parameter Download Script

Use case: Write calibration parameters to CH1-CH4.

```
#scriptMenu GyrfaLcon_s4000Pro_calibration_parameter_download_script_20261002124528

#logcmds 0
=var device=getDevice("Charger")
(device):
=var channel = "None"
=var command_PreprocessBeforeSettingProgramQ = "CONTRol:PROGrama:PREProcess? " + channel

=popupShowProgressAsync("Download GYRFALCON S4000Pro calibration parameter",4);

;;Start

CONTRol:PROGrama:PREProcess (channel);[200];

#if (deviceRead(device,command_PreprocessBeforeSettingProgramQ) == "Ready")

;;*****Calibration Parameter Download Script*****

=popupShowProgressSetPosition(1);
PACKet:RATIo
#31611.5465085506,0.9604977369,0.3318018615,1.0000000000,3.2911419868,0.0024977922,1.0000000000
0,3.2911419868,0.0024977922,1.0000000000,3.3124411106,-0.0238245726,1,28;[200];

=popupShowProgressSetPosition(2);
PACKet:RATIo
#31611.5465085506,0.9604977369,0.3318018615,1.0000000000,3.2911419868,0.0024977922,1.0000000000
0,3.2911419868,0.0024977922,1.0000000000,3.3124411106,-0.0238245726,2,27;[200];

=popupShowProgressSetPosition(3);
PACKet:RATIo
#31611.5465085506,0.9604977369,0.3318018615,1.0000000000,3.2911419868,0.0024977922,1.0000000000
0,3.2911419868,0.0024977922,1.0000000000,3.3124411106,-0.0238245726,3,26;[200];

=popupShowProgressSetPosition(4);
PACKet:RATIo
#31611.5465085506,0.9604977369,0.3318018615,1.0000000000,3.2911419868,0.0024977922,1.0000000000
0,3.2911419868,0.0024977922,1.0000000000,3.3124411106,-0.0238245726,4,25;[200];

;;*****END*****

CONTRol:PROGrama:COMPleted (channel);[50];
```

```
=popupShowProgressSetPosition(100);

#endif

;;Finished
```

10. Error Code Table

The decimal error codes output by `SCPI_Error` follow the SCPI standard. Common error codes:

Error Code	Macro	Meaning
-100	<code>SCPI_ERROR_COMMAND_ERROR</code>	Command error
-109	<code>SCPI_ERROR_MISSING_PARAMETER</code>	Missing parameter
-113	<code>SCPI_ERROR_UNDEFINED_HEADER</code>	Undefined header
-120	<code>SCPI_ERROR_NUMERIC_DATA_ERROR</code>	Numeric data error
-140	<code>SCPI_ERROR_CHARACTER_DATA_ERROR</code>	Character data error
-160	<code>SCPI_ERROR_BLOCK_DATA_ERROR</code>	Block data error
-161	<code>SCPI_ERROR_INVALID_BLOCK_DATA</code>	Invalid block data
-200	<code>SCPI_ERROR_EXECUTION_ERROR</code>	Execution error
-221	<code>SCPI_ERROR_SETTINGS_CONFLICT</code>	Settings conflict
-222	<code>SCPI_ERROR_PARAMETER_VALUE_OUT_OF_RANGE</code>	Parameter value out of range
-224	<code>SCPI_ERROR_ILLEGAL_PARAMETER_VALUE</code>	Illegal parameter value
-350	<code>SCPI_ERROR_QUEUE_OVERFLOW</code>	Error queue overflow
-410	<code>SCPI_ERROR_QUERY_INTERRUPTED</code>	Query interrupted
-420	<code>SCPI_ERROR_QUERY_UNTERMINATED</code>	Query unterminated
-430	<code>SCPI_ERROR_QUERY_DEADLOCKED</code>	Query deadlocked
-440	<code>SCPI_ERROR_QUERY_UNTERMINATED_AFTER_INDEFINITE_RESPONSE</code>	Query unterminated after indefinite response

Error output is typically a decimal number followed by a newline, for example:

```
Send:    PROGram:TYPE Ch1,XYZ
Return:  -224
```