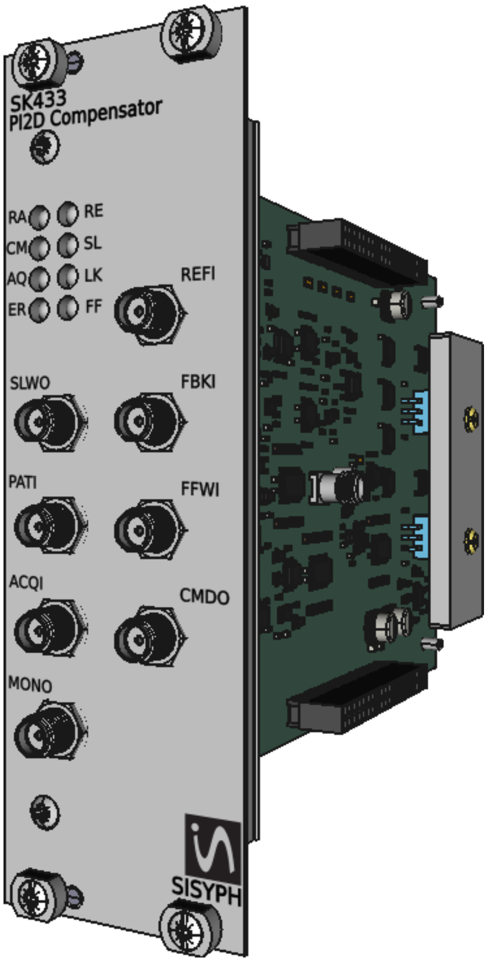
Programming Guide

SK433*Velay*
PI2D Compensator

SK-Series Modules



Signals & Systems for Physics

www.sisyph.com

General Information

Important Notice

Information in this document is subject to change without notice.
Copyright © SISYPH, 2024. All rights reserved.

Signals and Systems for Physics
BP90406
16 place saint-Georges
F31000 Toulouse France
Phone (+33) 781 547 391
www.sisyph.com

Scope

This document describes operating the SK433 PI2D Compensator module over the serial interface.

Contents

1 Introduction

Remote operation of the SK433 is through a simple command language documented in this chapter. Both set and query forms of most commands are supported, allowing the user complete control of the module from a remote computer.

1.1 Power-on Configuration

The settings for serial interface are 9600 baud with no parity and no hardware flow control, and local echo disabled (`CONS 0`).
Most of the instrument settings are stored in non-volatile memory and can be retrieved using the appropriate commands. At power-on the instrument returns to the state noted in the command descriptions. Reset values (`*RST` command) of parameters are shown in **boldface**.

1.2 Buffers

The instrument stores incoming bytes from the host interface in a 128-byte input buffer. Characters accumulate in the input buffer until a command terminator (either `<CR>` or `<LF>`) is received, at which point the message is parsed and executed. Query responses from the instrument are sent when they are ready without any flow control nor output buffering. The input buffer is automatically flushed upon detecting an overflow, and an error is recorded in the EVTS status register.

1.3 Command syntax

The four letter mnemonic (shown in CAPS) in each command sequence specifies the command. The rest of the sequence consists of parameters. The command parser accepts only uppercase mnemonics.
Commands may take either set or query form, depending on whether the `?` character follows the mnemonic. *Set only* commands are listed without the `?`, *query only* commands show the `?` after the mnemonic, and *optionally query* commands are marked with a `(?)`. Parameters shown in `{ }` and `[ ]` are not always required. Parameters in `{ }` are only required to set a value, and should be omitted for queries. Parameters in `[ ]` are optional in both set and query commands. Parameters listed without any surrounding characters are always required. Do not send `()` or `{ }` or `[]` as part of the command. Multiple parameters are separated by commas. Multiple commands may be sent on one command line by separating them with semicolons `;` so long as the input buffer does not overflow. Commands are terminated by either `<CR>` or `<LF>` characters. Null commands and whitespace are ignored. Execution of the command does not begin until the command terminator is received.
The following table summarizes the notation used in the command descriptions:

Symbol	Definition
`b`	boolean
`i, m, n`	unsigned integers
`u, v`	signed integers
`(?)`	required for queries; illegal for set commands.
`p`	parameter always required.
`{p}`	required parameter for set commands; illegal for queries.
`[p]`	optional parameter for both set and query forms.

1.4 Examples

Each command is provided with a simple example illustrating its usage. In these examples, all data sent by the host computer to the instrument are set as `straight teletype font`, while responses received the host computer from the instrument are set as *`slanted teletype font`*. The usage examples vary with respect to set/query, optional parameters, and token formats. These examples are not exhaustive, but are intended to provide a convenient starting point for user programming.

2 List of Commands

This section provides syntax and operational descriptions for remote commands.

2.1 Instrument Settings commands

The Instrument Settings commands provide control of the instrument's physical parameters.

STPS (Reference Setpoint)

Group	Instrument Settings commands
Action	Set/Query
Syntax	`STPS(?){u}`
Description	Set (query) the reference setpoint voltage {to u}, in mV with a resolution of 12-bit.
Allowed range	$-2500\,\text{mV} \leq u \leq +2500\,\text{mV}$.
Power-on value	Restored from non-volatile memory (*cf* *SAV).
Reset value	**0**
Example	`STPS -1000; STPS?`
	-1000
Related commands	REFS.

ERRC (Error Offset Compensation)

Group	Instrument Settings commands
Action	Set/Query
Syntax	`ERRC(?){`u`}`
Description	Set (query) the error offset compensation voltage {to u}, in µV, with a resolution of 12-bit. Whereas this voltage was primary provided to compensate DC-offset introduced by the front stage circuitry, it can be also used to slightly change the locking point.
Allowed range	$-25\,000\,\mu V \leq u \leq +25\,000\,\mu V$.
Power-on value	Restored from non-volatile memory (*cf* *SAV).
Reset value	**0**
Example	`ERRC +100; ERRC?`
	100
Related commands	ERRG.

ERRG (Error Gain)

Group	Instrument Settings commands
Action	Set/Query
Syntax	`ERRG(?){n}`
Description	Set (query) the gain applied to the error signal {to n}. The error gain (or proportional gain) ranges from −22 dB to +23 dB in 3-dB steps.
Allowed range	$n \in \{1, \ldots, 16\}$ where n is used as an index : $1 \longleftrightarrow -22$ dB; $2 \longleftrightarrow -19$ dB; $3 \longleftrightarrow -16$ dB; $4 \longleftrightarrow -13$ dB; $5 \longleftrightarrow -10$ dB; $6 \longleftrightarrow -7$ dB; $7 \longleftrightarrow -4$ dB; **8** $\longleftrightarrow -1$ dB; $9 \longleftrightarrow +2$ dB; $10 \longleftrightarrow +5$ dB; $11 \longleftrightarrow +8$ dB; $12 \longleftrightarrow +11$ dB; $13 \longleftrightarrow +14$ dB; $14 \longleftrightarrow +17$ dB; $15 \longleftrightarrow +20$ dB; $16 \longleftrightarrow +23$ dB.
Power-on value	Restored from non-volatile memory (*cf* *SAV).
Reset value	**8**
Example	`ERRG 9; ERRG?` *9*
Related commands	ERRC.

HFIF (HF-Integrator Frequency)

Group	Instrument Settings commands
Action	Set/Query
Syntax	`HFIF(?){n}`
Description	Set (query) the unity-gain frequency of the HF-integrator {to n}. Sixteen frequencies are available. Thirteen of them are set between 100 Hz and 1 MHz while the last three are provided for customization purpose. In order to use this feature, the capacitor corresponding to the desired frequency selection must be mounted on the printed-circuit board.
Allowed range	$n \in \{1, \ldots, 16\}$ where n is used as an index : 1 ⟷ 100 Hz; 2 ⟷ 200 Hz; 3 ⟷ 500 Hz; 4 ⟷ 1 kHz; 5 ⟷ 2 kHz; 6 ⟷ 5 kHz; 7 ⟷ 10 kHz; **8** ⟷ 20 kHz; 9 ⟷ 50 kHz; 10 ⟷ 100 kHz; 11 ⟷ 200 kHz; 12 ⟷ 500 kHz; 13 ⟷ 1 MHz; 14 ⟷ user-defined (HIF14); 15 ⟷ user-defined (HIF15); 16 ⟷ user-defined (HIF16).
Power-on value	Restored from non-volatile memory (*cf* *SAV).
Reset value	**8**
Example	`HFIF 10; HFIF?` *10*
Related commands	INTS, LFIF, SLIF.

LFIF (LF-Integrator Frequency)

Group	Instrument Settings commands
Action	Set/Query
Syntax	`LFIF(?){n}`
Description	Set (query) the unity-gain frequency of the LF-integrator {to n}. Sixteen frequencies are available. Thirteen of them are set between 10 Hz and 100 kHz while the last three are provided for customization purpose. In order to use this feature, the capacitor corresponding to the desired frequency selection must be mounted on the printed-circuit board.
Allowed range	$n \in \{1, \ldots, 16\}$ where n is used as an index : 1 ⟷ 10 Hz; 2 ⟷ 20 Hz; 3 ⟷ 50 Hz; 4 ⟷ 100 Hz; 5 ⟷ 200 Hz; 6 ⟷ 500 Hz; 7 ⟷ 1 kHz; **8** ⟷ 2 kHz; 9 ⟷ 5 kHz; 10 ⟷ 10 kHz; 11 ⟷ 20 kHz; 12 ⟷ 50 kHz; 13 ⟷ 100 kHz; 14 ⟷ user-defined (LIF14); 15 ⟷ user-defined (LIF15); 16 ⟷ user-defined (LIF16).
Power-on value	Restored from non-volatile memory (*cf* *SAV).
Reset value	**8**
Example	`LFIF 10; LFIF?` *10*
Related commands	INTS, HFIF, SLIF.

HFDF (HF-Differentiator Frequency)

Group	Instrument Settings commands
Action	Set/Query
Syntax	`HFDF(?){n}`
Description	Set (query) the unity-gain frequency of the HF-differentiator {to n}. Sixteen frequencies are available. Thirteen of them are set between 500 Hz and 5 MHz while the last three are provided for customization purpose. In order to use this feature, the capacitor corresponding to the desired frequency selection must be mounted on the printed-circuit board.
Allowed range	$n \in \{1, \ldots, 16\}$ where n is used as an index : 1 ⟷ 500 Hz; 2 ⟷ 1 kHz; 3 ⟷ 2 kHz; 4 ⟷ 5 kHz; 5 ⟷ 10 kHz; 6 ⟷ 20 kHz; 7 ⟷ 50 kHz; **8** ⟷ 100 kHz; 9 ⟷ 200 kHz; 10 ⟷ 500 kHz; 11 ⟷ 1 MHz; 12 ⟷ 2 MHz; 13 ⟷ 5 MHz; 14 ⟷ user-defined (HDF14); 15 ⟷ user-defined (HDF15); 16 ⟷ user-defined (HDF16).
Power-on value	Restored from non-volatile memory (*cf* *SAV).
Reset value	**8**
Example	`HFDF 11; HFDF?` *11*
Related commands	HFDG, DIFS.

HFDG (HF-Differentiator Gain)

Group	Configuration commands
Action	Set/Query
Syntax	`HFDG(?){`*b*`}`
Description	Set (query) the gain of the HF-differentiator {to b}.
	When $b = 0$ (resp. `1`), the gain is set to +12 dB (resp. +20 dB).
Allowed range	$b \in \{\mathbf{0}, 1\}$.
Power-on value	Restored from non-volatile memory (*cf* *SAV).
Reset value (`*RST`)	**0**
Example	`HFDG 1; HFDG?`
	1
Related commands	HFDF, DIFS.

SLIF (Slow-Integrator Frequency)

Group	Instrument Settings commands
Action	Set/Query
Syntax	`SLIF(?){n}`
Description	Set (query) the unity-gain frequency of the slow-integrator {to n}. Nine frequencies, ranging from 100 mHz to 1 kHz, are available.
Allowed range	$n \in \{1, \ldots, 9\}$ where n is used as an index : 1 ⟷ 100 mHz; 2 ⟷ 330 mHz; 3 ⟷ 1 Hz; **4** ⟷ 3.3 Hz; 5 ⟷ 10 Hz; 6 ⟷ 33 Hz; 7 ⟷ 100 Hz; 8 ⟷ 330 Hz; 9 ⟷ 1 kHz.
Power-on value	Restored from non-volatile memory (*cf* *SAV).
Reset value	**4**
Example	`SLIF 1; SLIF?` *1*
Related commands	INTS, HFIF, LFIF.

OFSS (PI2D Command Offset)

Group	Instrument Settings commands
Action	Set/Query
Syntax	`OFSS(?){`u`}`
Description	Set (query) the offset voltage for the PI2D compensator's output {to u}, in mV, with a resolution of 12-bit.
Allowed range	$-2500\,\mathrm{mV} \leq u \leq +2500\,\mathrm{mV}$.
Power-on value	Restored from non-volatile memory (*cf* *SAV).
Reset value	**0**
Example	`OFSS 1000; OFSS?`
	`1000`
Related commands	SLOS.

SLOS (Slow Command Offset)

Group	Instrument Settings commands
Action	Set/Query
Syntax	`SLOS(?){u}`
Description	Set (query) the offset voltage for the slow compensator's output {to u}, in mV, with a resolution of 12-bit.
Allowed range	$-5000\,\text{mV} \leq u \leq +5000\,\text{mV}$.
Power-on value	Restored from non-volatile memory (*cf* *SAV).
Reset value	**0**
Example	`SLOS 1000; SLOS?`
	1000
Related commands	OFSS.

FFWG (Feed-Forward Gain)

Group	Instrument Settings commands
Action	Set/Query
Syntax	`FFWG(?){`u`}`
Description	Set (query) the transmission coefficient of the feed-forward input {to u}, in %.
Allowed range	$-100\,\% \leq u \leq +100\,\%$. This command is used to adjust both weight and polarity of the open-loop command signal FFWI. Indeed, the command `FFWG 100` will set the adjustable transmission gain to 100 %, while `FFWG 0` will provide maximal attenuation of the FFWI input signal. In order to reverse the transmission polarity, the command is invoked with negative argument, for instance, `FFWG -100` will multiply the FFWI signal by the scalar -1. Notice that the output stage presents a fixed weight of 0.5 V/V on the FFWI signal path, which is not accounted in the feed-forward transmission gain.
Power-on value	Restored from non-volatile memory (*cf* *SAV).
Reset value	**0**
Example	`FFWG 100; FFWG?`
	`100`
Related commands	FFWE, ACQM.

PATA (Search-Pattern Amplitude)

Group	Instrument Settings commands
Action	Set/Query
Syntax	`PATA(?){n}`
Description	Set (query) the amplitude (resp. gain) of the internal (resp. external) search-pattern signal {to n}. Since the search-pattern signal can be either provided by an internal ramp generator or by an external source *via* the PATI input, the PATA command provides either eight amplitude or equivalent gain values, ranging from 1 V_{pp} to 12 V_{pp} (0.45 V/V to 5 V/V) in 3-dB steps. The gain values should be used to set the amplitude of the search-pattern signal when the external PATI input is selected. Notice that the values listed below apply to the slow output only. Indeed, to adapt the amplitude of the search pattern signal to a lower voltage range, an additional attenuator (0.5 V/V) is installed on the PI^2D's command output. This attenuation must be accounted when the search pattern is transmitted to the PI^2D's output.
Allowed range	$n \in \{1, \ldots, 8\}$ where n is used as an index : $8 \longleftrightarrow 12\,V_{pp}$ (5 V/V); $7 \longleftrightarrow 8.5\,V_{pp}$ (3.5 V/V); $6 \longleftrightarrow 6\,V_{pp}$ (2.5 V/V); $5 \longleftrightarrow 4.5\,V_{pp}$ (1.75 V/V); **4** $\longleftrightarrow 3\,V_{pp}$ (1.25 V/V); $3 \longleftrightarrow 2\,V_{pp}$ (0.85 V/V); $2 \longleftrightarrow 1.5\,V_{pp}$ (0.6 V/V); $1 \longleftrightarrow 1\,V_{pp}$ (0.45 V/V).
Power-on value	Restored from non-volatile memory (*cf* *SAV).
Reset value	**4**
Example	`PATA 1; PATA?` *1*
Related commands	PATP, PATS, PATD.

PATP (Search-Pattern Period)

Group	Instrument Settings commands
Action	Set/Query
Syntax	`PATP(?){n}`
Description	Set (query) the period of the internal search-pattern signal {to n}. Eight values, ranging from 3 ms to 10 s, are provided to set the period of the ramp signal.
Allowed range	$n \in \{1, \ldots, 8\}$ where n is used as an index : 1 ⟷ 3 ms; 2 ⟷ 10 ms; 3 ⟷ 30 ms; **4** ⟷ 100 ms; 5 ⟷ 300 ms; 6 ⟷ 1 s; 7 ⟷ 3 s; 8 ⟷ 10 s.
Power-on value	Restored from non-volatile memory (*cf* *SAV).
Reset value	**4**
Example	`PATP 1; PATP?` *1*
Related commands	PATA, PATS, PATD.

2.2 Instrument Configuration commands

The Instrument Configuration commands provide control of the instrument's physical functionalities.

REFS (Reference Source)

Group	Instrument Configuration commands
Action	Set/Query
Syntax	`REFS(?){`n`}`
Description	Set (query) the reference signal source {to n}.
Allowed range	$n \in \{0, \mathbf{1}, 2\}$ where n is used as an index :
	0 $\longleftrightarrow$ grounded (0 V);
	1 $\longleftrightarrow$ internally generated (from DAC);
	2 $\longleftrightarrow$ externally provided (from REFI input).
Power-on value	Restored from non-volatile memory (*cf* *SAV).
Reset value	**1**
Example	`REFS 0; REFS?`
	0
Related commands	STPS.

LOCK (Locking)

Group	Instrument Configuration commands
Action	Set/Query
Syntax	`LOCK(?){n}`
Description	Set (query) the lock state {to n}.
	Both output command signals of PI2D and slow compensator are controlled *via* the `LOCK` command. Each value of the index n is related to a state of the internal finite-state machine in charge of delivering the command signals. ULK refers to a state where the integrators' outputs are reset while the differentiator's output is switched off. Only offset voltages can be present. The SPA state also corresponds to an unlock state as described previously, but here one integrator is used to drive its related command output with the search-pattern signal. LCK refers to the lock state where both PI2D and slow integrator operate for delivering the closed-loop command signals. The ACQI trigger input can be used to switch between lock and unlock states (ACQ-LCK). It is also possible to acquire lock automatically (ACQ-AUT) by switching between SPA and LCK states. Here, the applied search-pattern signal brings the plant's output inside the sensor range which is detected and fed-back to the ACQI input signal in order to stop scanning and engage the servo loops. Before setting lock operation, the configuration (error signal, integrator, differentiator, search-pattern signal...) must be completed to prevent unexpected behavior. Notice that the proportional error signal is transmitted to the PI²D command output in SPA mode. This feature is used to experimentally find the right loop polarity. Indeed, by sweeping the manipulated variable over the desired locking point, the proportional feedback *via* the PI²D output will counteract the sweeping action if the loop operates with the good polarity. In the opposite, the proportional feedback will add to the sweeping action by driving the sensor's output out-of-range. The right loop polarity can be then restored using the `ERRN` remote command.
Allowed range	$n \in \{\mathbf{0}, \ldots, 4\}$ where n is used as an index : **0** ⟷ manual unlock (ULK); 1 ⟷ manual unlock with search-pattern engaged (SPA); 2 ⟷ manual lock (LCK); 3 ⟷ event-controlled lock/unlock transitions (ACQ-LCK); 4 ⟷ event-controlled lock/search transitions (ACQ-AUT).
Power-on value	Restored from non-volatile memory (*cf* *SAV).
Reset value	**0**
Example	`LOCK 1`
Related commands	INTS, DIFS, ACQM, PATD.

FBKE (Feed-Back State)

Group	Configuration commands
Action	Set/Query
Syntax	`FBKE(?){b}`
Description	Set (query) the feed-back signal's state {to b}.
	When $b = 0$ (resp. `1`), the feed-back signal is disabled (resp. enabled).
Allowed range	$b \in \{0, \mathbf{1}\}$.
Power-on value	Restored from non-volatile memory (*cf* *SAV).
Reset value (`*RST`)	**1**
Example	`FBKE 1; FBKE?`
	1
Related commands	REFS.

ERRN (Error Inverter)

Group	Configuration commands
Action	Set/Query
Syntax	`ERRN(?){b}`
Description	Set (query) the inverter's state of the PI2D error signal {to b}. The error inverter allows the user to change the polarity of the feed-back of the PI2D loop. When $b = 0$ (resp. 1), the error inverter is disabled (resp. enabled).
Allowed range	$b \in \{\mathbf{0}, 1\}$.
Power-on value	Restored from non-volatile memory (*cf* *SAV).
Reset value (*RST)	**0**
Example	`ERRN 0; ERRN?`
	0
Related commands	

SLEN (Slow Error Inverter)

Group	Configuration commands
Action	Set/Query
Syntax	`SLEN(?){`*b*`}`
Description	Set (query) the inverter's state of the slow error signal {to *b*}. The error inverter allows the user to change the polarity of the feed-back of the slow loop. When $b = 0$ (resp. 1), the error inverter is disabled (resp. enabled).
Allowed range	$b \in \{\mathbf{0}, 1\}$.
Power-on value	Restored from non-volatile memory (*cf* *SAV).
Reset value (`*RST`)	**0**
Example	`SLEN 0; SLEN?`
	0
Related commands	SLIF.

FFWE (Feed-Forward State)

Group	Configuration commands
Action	Set/Query
Syntax	`FFWE(?){`*b*`}`
Description	Set (query) the feed-forward signal's state {to b}. When $b = 0$ (resp. 1), the feed-forward signal is switched-off (resp. switched-on). The ACQI input can be also used to control the feed-forward signal's state.
Allowed range	$b \in \{\mathbf{0}, 1\}$.
Power-on value	Restored from non-volatile memory (*cf* *SAV).
Reset value (`*RST`)	**0**
Example	`FFWE 1; FFWE?`
	0
Related commands	FFWG, ACQM.

OFSE (PI2D Command Offset)

Group	Configuration commands
Action	Set/Query
Syntax	`OFSE(?){`b`}`
Description	Set (query) the state of the PI2D command offset {to b}. When $b = 0$ (resp. 1), the command offset voltage of the PI2D compensator is switched off (resp. switched on).
Allowed range	$b \in \{\mathbf{0}, 1\}$.
Power-on value	Restored from non-volatile memory (*cf* *SAV).
Reset value (*RST)	**0**
Example	`OFSE 0; OFSE?`
	0
Related commands	OFSS.

SLOE (Slow Command Offset)

Group	Configuration commands
Action	Set/Query
Syntax	`SLOE(?){b}`
Description	Set (query) the state of the slow command offset {to b}. When $b =$ 0 (resp. 1), the command offset voltage of the slow compensator is switched off (resp. switched on).
Allowed range	$b \in \{\mathbf{0}, 1\}$.
Power-on value	Restored from non-volatile memory (*cf* *SAV).
Reset value (`*RST`)	**0**
Example	`SLOE 0; SLOE?`
	0
Related commands	SLOS.

INTS (Integrator Selection)

Group	Configuration commands
Action	Set/Query
Syntax	`INTS(?){m}`
Description	Set (query) which are the integrators engaged when locking {to bit-mask m}. In order to engage the integrator i when locking, the command must be invoked with $m = 2^i$ as argument. For instance, `INTS 3` must be executed to only engage the LF integrator of the PI2D compensator and the integrator of the slow loop. The selected integrators also operate when acquiring lock to transmit the search pattern signal to the related command output.
Allowed range	$m \in \{1, \ldots, \mathbf{7}\}$, where m can be any combination of : $m_0 = 1 = 2^0 \longleftrightarrow$ slow-integrator; $m_1 = 2 = 2^1 \longleftrightarrow$ LF-integrator; $m_2 = 4 = 2^2 \longleftrightarrow$ HF-integrator.
Power-on value	Restored from non-volatile memory (*cf* *SAV).
Reset value	**7**
Example	`INTS 3; INTS?` *3*
Related commands	SLIF, LFIF, HFIF, LOCK, DIFS.

DIFS (Differentiator Selection)

Group	Configuration commands
Action	Set/Query
Syntax	`DIFS(?){`*b*`}`
Description	Set (query) whether the differentiator is engaged when locking {to *b*}. When $b = 0$ (resp. `1`), the action of the differentiator is disabled (resp. enabled).
Allowed range	$b \in \{\mathbf{0}, 1\}$.
Power-on value	Restored from non-volatile memory (*cf* *SAV).
Reset value (`*RST`)	**0**
Example	`DIFS 0; DIFS?`
	0
Related commands	INTS, HFDF, HFDG.

PATS (Search-Pattern Source)

Group	Configuration commands
Action	Set/Query
Syntax	`PATS(?){n}`
Description	Set (query) the source for the search-pattern signal {to n}.
Allowed range	$n \in \{\mathbf{0}, 1, 2\}$, where n is used as an index : **0** ⟷ internal ground voltage (0 V); 1 ⟷ internal source (ramp generator); 2 ⟷ external source (PATI input).
Power-on value	Restored from non-volatile memory (*cf* *SAV).
Reset value	**0**
Example	`PATS 0; PATS?` *0*
Related commands	PATA, PATP, PATD.

PATD (Search-Pattern Destination)

Group	Configuration commands
Action	Set/Query
Syntax	`PATD(?){`*b*`}`
Description	Set (query) the destination for the search-pattern signal when scanning {to *b*}. When $b = 0$ (resp. `1`), the search-pattern signal is routed to the LF-integrator (resp. slow-integrator).
Allowed range	$b \in \{0, \mathbf{1}\}$.
Power-on value	Restored from non-volatile memory (*cf* *SAV).
Reset value (`*RST`)	**1**
Example	`PATD 0; PATD?`
	`0`
Related commands	PATA, PATP, PATS.

ACQT (ACQI Trigger)

Group	Configuration commands
Action	Set/Query
Syntax	`ACQT(?){n}`
Description	Set (query) the threshold voltage for the ACQI input {to n}. The related event is triggered when the ACQI is asserted, when the ACQI's voltage is above the threshold level v_t.
Allowed range	$n \in \{1, \ldots, 7\}$, where n is used as an index : $1 \longleftrightarrow v_t = 1\,\text{V};$ $2 \longleftrightarrow v_t = 1.5\,\text{V};$ $3 \longleftrightarrow v_t = 2\,\text{V};$ $\mathbf{4} \longleftrightarrow v_t = 2.5\,\text{V};$ $5 \longleftrightarrow v_t = 3\,\text{V};$ $6 \longleftrightarrow v_t = 3.5\,\text{V};$ $7 \longleftrightarrow v_t = 4\,\text{V}.$
Power-on value	Restored from non-volatile memory (*cf* *SAV).
Reset value	**4**
Example	`ACQT 7; ACQT?` *7*
Related commands	ACQM, LOCK.

ACQM (ACQI Operating Mode)

Group	Configuration commands
Action	Set/Query
Syntax	`ACQM(?){n}`
Description	Set (query) the operating mode of the ACQI input {to n}. The ACQI input can be configured as an event driving the finite-state machine used for locking (LCK). This input can also control the feed-forward signal (FFW). When configured for report-only operation (RPT), an ACQ-event only switches the ACQ led indicator on and raises the related flag in the INSS status register.
Allowed range	$n \in \{0, \ldots, 3\}$, where n is used as an index : **0** $\longleftrightarrow$ no operation (NOP); 1 $\longleftrightarrow$ report-only (RPT); 2 $\longleftrightarrow$ event-driven locking (LCK); 3 $\longleftrightarrow$ event-driven feed-forwarding (FFW);
Power-on value	Restored from non-volatile memory (*cf* *SAV).
Reset value	**0**
Example	`ACQM 2; ACQM?` *2*
Related commands	ACQT, LOCK.

MONS (MONO Output Selection)

Group	Configuration commands
Action	Set/Query
Syntax	`MONS(?){n}`
Description	Set (query) the source of the monitoring output signal {to n}. This command is used to select which signal is routed to the MONO coaxial connector.
Allowed range	$n \in \{\mathbf{0}, \ldots, 7\}$ where : **0** ⟷ analog ground voltage (0 V); 1 ⟷ PI2D error signal; 2 ⟷ PI2D command signal; 3 ⟷ slow-loop command signal; 4 ⟷ search pattern signal (inverted); 5 ⟷ slow-loop error signal; 6 ⟷ PI2D error signal, bandwidth limited ($f_c = 200\,\text{kHz}$); 7 ⟷ PI2D error signal, AC-coupled ($f_c = 10\,\text{Hz}$).
Power-on value	Restored from non-volatile memory (*cf* *SAV).
Reset value	**0**
Example	`MONS 1; MONS?` *1*
Related commands	

2.3 Instrument Monitoring commands

The Instrument Monitoring commands provide the host computer with the last measurements of the instrument's physical parameters.

RMON (Remote Monitoring)

Group	Monitoring commands
Action	Query only
Syntax	`RMON?`*n*
Description	Return to the host computer the last measurement of the parameter specified by n. These parameters are periodically sampled (100 ms) by an internal task.
Allowed range	$n \in \{0, \ldots, 4\}$, where the index n : 0 ⟷ PI2D error, in µV, range ±20 000 µV; 1 ⟷ PI2D command, positive peak-voltage, in mV; 2 ⟷ PI2D command, negative peak-voltage, in mV; 3 ⟷ slow command, positive peak-voltage, in mV; 4 ⟷ slow command, negative peak-voltage, in mV.
Example	`RMON? 1` *-17*
Related commands	

TDIE (Die Temperature)

Group	Monitoring commands
Action	Query only
Syntax	`TDIE?`
Description	Return the die temperature.
	`TDIE?` returns the last measurement of the temperature (in K) of the die provided by the MCU on-chip sensor. The precision is about ±1 K. This reading can be used to get an approached value of the main printed circuit board's temperature where the MCU is mounted. This measurement is automatically updated every 100 ms.
Example	`TDIE?`
	298
Related commands	

STMS (Streamed Channels Selection)

Group	Monitoring commands
Action	Set/Query
Syntax	`STMS(?){m}`
Description	Set (query) the channels selection register {to bit-mask m}. In order to stream the channel i to output, the command must be invoked with $m = 2^i$ as argument. For instance, `STMS 6` will be executed to stream the positive and negative peak-voltages of the PI2D command to output. Data are output on a single line where a comma delimiter is used to separate channel data. The rightmost position in the row is occupied by the channel with the lowest weight. Data streaming operation is enabled using the STME command while the number of measurements that will streamed out is set by the STMN command.
Allowed range	$m \in \{\mathbf{1}, \ldots, 31\}$, where m can be any combination of : $m_0 = \mathbf{1} = 2^0 \longleftrightarrow$ PI2D error, in µV, range ±20 000 µV; $m_1 = 2 = 2^1 \longleftrightarrow$ PI2D command, positive peak-voltage [mV]; $m_2 = 4 = 2^2 \longleftrightarrow$ PI2D command, negative peak-voltage [mV]; $m_3 = 8 = 2^3 \longleftrightarrow$ slow command, positive peak-voltage [mV]; $m_4 = 16 = 2^4 \longleftrightarrow$ slow command, negative peak-voltage [mV].
Power-on value	Restored from non-volatile memory (*cf* *SAV).
Reset value	**1**
Example	`STMN 5; STMS 3 ; STME 1` *-1628,2413* *-1637,2402* *-1670,2450* *-1641,2437* *-1633,2445*
Related commands	STME, STMN.

STME (Data Streaming Enable)

Group	Monitoring commands
Action	Set/Query
Syntax	`STME(?){`b`}`
Description	Set (query) the state of the data streaming functionality {to b}. If $b = 0$ (resp. 1), data streaming is disabled (resp. enabled). When data streaming is enabled, the number of measurements that will be streamed out is specified using the STMN command.
Allowed range	$b \in \{\mathbf{0}, 1\}$.
Power-on value	0
Reset value (`*RST`)	**0**
Example	`STME 1; STME?`
	1
Related commands	STMS, STMN.

STMN (Number of Streamed Measurements)

Group	Monitoring commands
Action	Set/Query
Syntax	`STMN(?){n}`
Description	Set (query) the number of measurements to be streamed out {to n}. When $n = 0$, measurements will be output indefinitely, until the STME command is invoked. If $n > 0$, data streaming will stop once n measurements will have been output. Measurements are streamed out at a constant rate of approximately one measurement per second.
Allowed range	$\mathbf{0} \leq n \leq 10000$.
Reset value	**0**
Example	`STMN 1000; STMN?` *1000*
Related commands	STME, STMS.

2.4 Status Reporting commands

The Status commands query and configure registers associated with status reporting of the instrument.

*CLS (Clear Status Registers)

Group	Status reporting commands
Action	Query only
Syntax	`*CLS`
Description	Clear immediately all status registers, which are : CTSS, STAS, LEXE, LCMD, LINS, LURQ, INSS, OVLS, COMS and EVTS.
Example	`*CLS`
Related commands	

MSTS (Master Summary Status)

Group	Status reporting commands
Action	Query only
Syntax	`MSTS? [`*n*`]`
Description	Return the Master Summary Status register [bit-mask n]. The execution of the `MSTS?` query – without the optional bit-mask n – always causes the /STATUS signal to be de-asserted. Note that `MSTS?` *n* will not clear /STATUS, even if bit $i \mid n = 2^i$ is the only bit presently causing the /STATUS signal.
Power-on value	0
Example	`MSTS?; MSTS? 128;` *129* *128*
Related commands	MSTE

MSTE (Master Summary Enable)

Group	Status reporting commands
Action	Set/Query
Syntax	`MSTE(?)[`*n*`]{`*m*`}`
Description	Set (query) the Master Summary Enable register [bit-mask n] {to bit-mask m}. The set-form command will clear the bits outside the bit-mask.
Power-on value	0
Example	`MSTE 128; MSTE?`
	128
Related commands	MSTS

EVTS (Event Status)

Group	Status reporting commands
Action	Query only
Syntax	`EVTS? [`*n*`]`
Description	Read the Event Summary Status register [bit-mask *n*].
Power-on value	1
Example	`EVTS?`
	4
Related commands	EVTE

EVTE (Event Enable)

Group	Status reporting commands
Action	Set/Query
Syntax	`EVTE(?)[n]{m}`
Description	Set (query) the Event Summary Enable register [bit-mask n] {to bit-mask m}. The set-form command will clear the bits outside the bit-mask.
Power-on value	0
Example	`EVTE 4; EVTE?`
	4
Related commands	EVTS

COMS (Communications Status)

Group	Status reporting commands
Action	Query only
Syntax	`COMS? [n]`
Description	Read the Communications Status register [bit-mask n].
Power-on value	0
Example	`COMS?`
	0
Related commands	COME

COME (Communications Enable)

Group	Status reporting commands
Action	Set/Query
Syntax	`COME(?)[n]{m}`
Description	Set (query) the Communications Enable register [bit-mask n] {to bit-mask m}. The set-form command will clear the bits outside the bit-mask.
Power-on value	0
Example	`COME 1`
Related commands	COMS

OVLS (Overload Status)

Group	Status reporting commands
Action	Query only
Syntax	`OVLS? [n]`
Description	Read the Overload Status register [bit-mask n].
Power-on value	0
Example	`OVLS?`
	2
Related commands	OVLE, OVLC.

OVLE (Overload Enable)

Group	Status reporting commands
Action	Set/Query
Syntax	`OVLE(?)[`*n*`]{`*m*`}`
Description	Set (query) the Overload Enable register [bit-mask n] {to bit-mask m}. The set-form command will clear the bits outside the bit-mask.
Power-on value	0
Example	`OVLE 2`
Related commands	OVLS, OVLC.

OVLC (Overload Condition)

Group	Status reporting commands
Action	Query only
Syntax	`OVLC? [n]`
Description	Read the Overload Condition register [bit-mask n]. The values of the bits in the OVLC condition register are determined by the current (real-time) condition of the events defined in the OVLS status register. Reading the condition register does not affect the register.
Power-on value	0
Example	`OVLC?` *2*
Related commands	OVLS, OVLE.

INSS (Instrument Status)

Group	Status reporting commands
Action	Query only
Syntax	`INSS? [`*n*`]`
Description	Read the Instrument Status register [bit-mask *n*].
Power-on value	0
Example	`INSS?`
	1
Related commands	LINS, INSE, INSC.

INSE (Instrument Enable)

Group	Status reporting commands
Action	Set/Query
Syntax	`INSE(?)[`n`]{`m`}`
Description	Set (query) the Instrument Enable register [bit-mask n] {to bit-mask m}. The set-form command will clear the bits outside the bit-mask.
Power-on value	0
Example	`INSE 2`
Related commands	LINS, INSS, INSC.

INSC (Instrument Condition)

Group	Status reporting commands
Action	Query only
Syntax	`INSC? [`*n*`]`
Description	Read the Instrument Condition register [bit-mask *n*]. The values of the bits in the INSC condition register are determined by the current (real-time) condition of the events defined in the INSS status register. Reading the condition register does not affect the register.
Power-on value	0
Example	`INSC?` *2*
Related commands	LINS, INSE, INSS.

2.5 Interface commands

The Interface commands provide control over the interface between the instrument and the host computer.

*RST (Reset)

Group	Interface commands
Action	Set only
Syntax	`*RST`
Description	Reset the instrument to its default configuration. When a parameter is affected by the *RST command, its value is reset according to the information given by the Reset value field within the related command section. Whereas status registers are unaffected by *RST, the content of some conditions registers may have been modified upon resetting the intrument.
Example	`*RST`
Related commands	*RCL, *SAV.

*OPC (Operation Complete)

Group	Interface commands
Action	Set/Query
Syntax	`*OPC(?)`
Description	Set the OPC flag in the EVTS register.
	The query form `*OPC?` returns `1` when complete, but does not affect the EVTS register.
Example	`*OPC?`
	1
Related commands	

CONS (Console Mode)

Group	Interface commands
Action	Set/Query
Syntax	`CONS(?){`*m*`}`
Description	Set (query) the Console mode {to m}. `CONS 1` causes each character received to be returned to the host computer.
Allowed range	$m \in \{\mathbf{0}\,(\text{disabled}), 1\,(\text{enabled})\}$
Reset (`*RST`) value	**0**
Power-on value	0
Example	`CONS 1` *1*
Related commands	

*IDN (Identify)

Group	Interface commands
Action	Query only
Syntax	`*IDN?`
Description	Read the device identification string. This string is formatted as: Signals and Systems for Physics, model SK433, hw Rppx, fw Rqqy, s/n dddddd. In this string, `SK433` is the model number, `Rnnx` and `Rppy` are revision numbers identifying the hardware or the firmware versions and `dddddd` is the 6-digit serial number.
Example	`*IDN?` *`Signals and Systems for Physics, model SK433, hw R24B, fw R24A, s/n 123456.`*
Related commands	

LINS (Last Instrument Error Status)

Group	Status reporting commands
Action	Query only
Syntax	`LINS?`
Description	Query the last execution instrument error. `LINS?` returns the unique code number associated with this event.
Valid codes are	0 ⟷ no execution error since last `LINS?`; 1 ⟷ on-chip ADC error; 10 ⟷ detected hardware is in invalid condition; 20 ⟷ some parameters have been be adapted or clamped; 21 ⟷ some functionalities have been be disabled.
Power-on value	0
Example	`LINS?` *0*
Related commands	LCMD, LEXE, LURQ.

LURQ (Last User Request Status)

Group	Interface commands
Action	Query only
Syntax	`LURQ?`
Description	Query the last User request. `LURQ?` returns the unique code number associated with this event.
Valid codes are	0 ⟷ No User request since last `LURQ?`
Power-on value	0
Example	`LURQ?`
	`0`
Related commands	LCMD, LEXE, LINS.

LCMD (Last Command Error Status)

Group	Interface commands
Action	Query only
Syntax	`LCMD?`
Description	Query the last command error. `LCMD?` returns the unique code number associated with this error.
Valid codes are	0 ⟷ no execution error since last `LCMD?`
	1 ⟷ illegal (unknown) command.
	2 ⟷ illegal query.
	3 ⟷ illegal set (read-only command).
	4 ⟷ extra parameter.
	5 ⟷ missing parameter.
	6 ⟷ null command.
Power-on value	0
Example	`*RST?;LCMD?`
	2
Related commands	LURQ, LEXE, LINS.

LEXE (Last Execution Error Status)

Group	Interface commands
Action	Query only
Syntax	`LEXE?`
Description	Query the last execution error. `LEXE?` returns the unique code number associated with this error.
Valid codes are	0 ⟷ no execution error since last `LEXE?` 1 ⟷ invalid parameter. 2 ⟷ argument value out-of-range. 3 ⟷ some parameters have been adapted or clamped. 4 ⟷ a conflict due to the current operation has been avoided. 5 ⟷ no change upon executing the command. 6 ⟷ the operation was aborted due to a fault condition.
Power-on value	0
Example	`CONS2;LEXE?;LEXE?` *1* *0*
Related commands	LURQ, LCMD, LINS.

TERM (Response Termination)

Group	Interface commands
Action	Set/Query
Syntax	`TERM(?){`*m*`}`
Description	Set (query) the termination sequence {to m}. The termination sequence is appended to all query responses sent by the instrument. It is constructed of ASCII character(s) `<CR>` (carriage return) or `<LF>` (line feed).
Allowed range	$m \in \{1, 2, 3, 4\}$ where : 1 ⟷ `<CR>` character appended, 2 ⟷ `<LF>` character appended, **3** ⟷ both `<CR>` and `<LF>` characters appended, 4 ⟷ no character appended.
Power-on value	3
Reset (`*RST`) value	**3**
Example	`TERM?` *3*
Related commands	

2.6 Memory commands

The Memory commands allow the User to save and recall the instrument's settings in non-volatile memory.

*RCL (Recall Settings)

Group	Memory commands
Action	Set only
Syntax	`*RCL`
Description	Recall the settings stored in the non-volatile memory.
Example	`*RCL`
Related commands	*RST, *SAV.

*SAV (Save Current Settings)

Group	Memory commands
Action	Set only
Syntax	`*SAV`
Description	Save the current settings in the non-volatile memory.
Example	`*SAV`
Related commands	*RCL, *RST.

3 Status Model

The complete block diagram of the status register array is available online at the related product page. There are four categories of registers in this model :

Last Event registers These four read registers (LINS, LCMD, LURQ and LEXE) store the last event that they monitor. A query command i) return the last registered event since the previous query and ii) clears the register's content.

Condition registers These read-only registers correspond to the real-time condition of some underlying physical properties under monitoring. Queries return the latest value of the property, and have no other effect.
Condition register names end with C.

Status registers These read-only registers record the occurrence of defined events. If the event occurs, the corresponding status bit is set to 1. Upon querying a status register, any set bits within it are cleared. These are sometimes known as sticky bits since once set, a bit can only be cleared by reading its value. Status register names end with S.

Enable registers These read/write registers define a bitwise mask for their corresponding status register. If any bit position is set in a status register while the same bit position is also set in the enable register, then the corresponding summary bit is set in either the Event Summary or Master Summary register. Enable register names end with E.

3.1 Master Summary Status (MSTS)

The Master Summary Status (MSTS) is the top-level summary register of the status model. When masked by the Master Summary Status Enable (MSTE) register, a bit set in the Status Byte causes the /STATUS signal to be asserted on the DIN41612 connector. This register is queried with the MSTS?[n] command.

Weight $n = 2^i$	**Bit** i	**Flag**	**Description**
1	0	MSS	Master Summary Status. Indicates whether one or more of the enabled status messages in the Status Byte register is true.
2	1	COM	Communication Summary Bit. Indicates whether one or more of the enabled flags in the Communication Status register has become true.
4	2	EVT	Event Summary Bit. Indicates whether one or more of the enabled flags in the Event Status register is true.
8	3	RFU	Undefined (read 0).
16	4	RFU	Undefined (read 0).
32	5	RFU	Undefined (read 0).
64	6	INS	Instrument Summary Bit. Indicates whether one or more of the enabled flags in the Instrument Status register is true.
128	7	OVL	Overload Summary Bit. Indicates whether one or more of the enabled flags in the Overload Status register is true.

3.2 Master Summary Enable (MSTE)

Each bit in the MSTE register corresponds one-to-one with a bit in the MSTS register, and acts as a bitwise AND of the MSTS flags to generate the MSS flag. Bit 0 of the MSTE is undefined—setting it has no effect, and reading it always returns 0. This register is set and queried with the MSTE(?) command and cleared at power-on.

3.3 Event Status (EVTS)

The Event Status register consists of 8 event flags. These flags are set by the corresponding event, and cleared only by reading or with the *CLS command ("sticky bits"). Querying the single bit i with the command EVTS? n where the bit-mask $n = 2^i$ will only clear the bit i. For instance, issuing the command `EVTS?128` will clear the bit 7 (INS) only.

Weight $n = 2^i$	**Bit** i	**Flag**	**Description**
1	0	PON	Power On event. Indicates that an off-to-on transition has occurred.
2	1	OPC	Operation Complete. Set by the *OPC command.
4	2	CMD	Command Error event. Indicates an error detected by the command parser. The error code can be queried with LCMD?
8	3	EXE	Execution Error event. Indicates an error in a command that was successfully parsed. The error code can be queried with LEXE?
16	4	RXQ	Reception Buffer event. Indicates that the RX buffer has been cleared.
32	5	TXQ	Transmission Buffer event. Indicates that the TX buffer has been cleared.
64	6	URQ	User Request event. Indicates that a User request has occured. The request code can be queried with LURQ?
128	7	INS	Instrument event. Indicates whether one or more of the enabled flags in the Instrument Status register is true.

3.4 Event Enable (EVTE)

Each bit in the EVTE register corresponds one-to-one with a bit in the EVTS register, and acts as a bitwise AND of the EVTS flags to generate the EVT flag in the Master Summary Status (MSTS) register. This register is set and queried with the EVTE command and cleared at power-on. For instance, issuing the command `EVTE 128` enable the bit 7 (INS) only.

3.5 Instrument Status (INSS)

The Instrument Status register consists of 8 event flags. These flags are set by the corresponding event, and cleared only by reading or with the *CLS command ("sticky bits"). Querying the single bit i with the command INSS? n where the bit-mask $n = 2^i$ will only clear the bit i. For instance, issuing the command `INSS?1` will clear the bit 0 only.

Weight $n = 2^i$	**Bit** i	**Flag**	**Description**
1	0	PUV	Power Supply Under-Voltage. At least, one power supply is under its low-level threshold (bit set).
2	1	IKS	Internal 10-MHz clock source used. The module is not synchronized to the platform's timebase. *The platform's timebase synchronization feature is not yet implemented. The bit is therefore always set (read 1).*
4	2	ACQ	Event triggered from the ACQI input (bit set).
8	3	SPA	Scanning mode operation using the search-pattern signal (bit set).
16	4	LCK	Lock mode operation (bit set).
32	5	ULK	Unlock mode operation (bit set).
64	6	RFU	Undefined (read 0).
128	7	FFW	The feed-forward signal is switched on (bit set).

3.6 Instrument Enable (INSE)

Each bit in the INSE register corresponds one-to-one with a bit in the INSS register, and acts as a bitwise AND of the INSS flags to generate the INS flag in the Master Summary Status (MSTS) register. This register is set and queried with the INSE command and cleared at power-on.

3.7 Instrument Condition (INSC)

Each bit in the INSC register corresponds one-to-one with a bit in the INSS register. The bits in the INSC register reflect the real-time values of their corresponding signals. Reading the entire register, or individual bits within it, does not affect the value of INSC. This register is queried with the INSC command and cleared at power-on.

3.8 Overload Status (OVLS)

The Overload Status register consists of 8 event flags. These event flags are set by the corresponding event, and cleared only by reading or with the *CLS command ("sticky bits"). Querying the single bit i with the command OVLS? n where the bit-mask $n = 2^i$ will only clear the bit i. For instance, issuing the command `OVLS?2` will clear the bit 1 only.

Weight $n = 2^i$	**Bit** i	**Flag**	**Description**
1	0	CML	The PI2D command output reaches (bit set) its lower limit, -3 V.
2	1	CMH	The PI2D command output reaches (bit set) its upper limit, $+3$ V.
4	2	SLL	The slow command output reaches (bit set) its lower limit, -8 V.
8	3	SLH	The slow command output reaches (bit set) its upper limit, $+8$ V.
16	4	PGA	Saturation of the programmable amplifier detected (bit set).
32	5	ERR	Saturation of the error amplifier detected (bit set).
64	6	SLI	Saturation of the slow-loop integrator.
128	7	LFI	Saturation of the low-frequency integrator.

3.9 Overload Enable (OVLE)

Each bit in the OVLE register corresponds one-to-one with a bit in the OVLS register, and acts as a bitwise AND of the OVLS flags to generate the OVL flag in the Master Summary Status (MSTS) register.

3.10 Overload Condition (OVLC)

Each bit in the OVLC register corresponds one-to-one with a bit in the OVLS register. The bits in the OVLC register reflect the real-time values of their corresponding signals. Reading the entire register, or individual bits within it, does not affect the value of OVLC. This register is queried with the OVLC command and cleared at power-on.

3.11 Communication Status (COMS)

The Communication Status register consists of 8 event flags. These flags are set by the corresponding event, and cleared only by reading or with the *CLS command ("sticky bits"). Querying the single bit i with the command COMS? n where the bit-mask $n = 2^i$ will only clear the bit i.
Because the COMS register is not used in the SK433 , querying this register always returns 0. Therefore, the corresponding summary bit in the MSTS register (bit COM) is never set whatever the value of the COME register.

Weight $n = 2^i$	**Bit** i	**Flag**	**Description**
1	0	PRY	Parity violation.
2	1	COL	Bus collision.
4	2	RFU	Undefined (read 0).
8	3	RFU	Undefined (read 0).
16	4	RFU	Undefined (read 0).
32	5	RFU	Undefined (read 0).
64	6	RFU	Undefined (read 0).
128	7	RFU	Undefined (read 0).

3.12 Communication Enable (COME)

Each bit in the COME register corresponds one-to-one with a bit in the COMS register, and acts as a bitwise AND of the COMS flags to generate the COM flag in the Master Summary Status (MSTS) register. This register is set and queried with the COME command and cleared at power-on.

3.13 Last Command Error (LCMD)

The LCMD register holds the last error detected by the command parser. The related error code can be retrieved by the command LCMD?. When such an error is detected, the corresponding bit in the Event Status register is set (bit CMD in EVTS).

3.14 Last Execution Error (LEXE)

The LEXE register holds the last error detected during the execution of a command. The related error code can be retrieved by the command LEXE?. When such an error is detected, the corresponding bit in the Event Status register is set (bit EXE in EVTS).

3.15 Last Instrument Error (LINS)

The LINS register holds the last error detected during the opertation of the instrument. The related error code can be retrieved by the command LINS?. When such an error is detected, the corresponding bit in the Event Status register is set (bit INS in EVTS).

3.16 Last User Request (LURQ)

The LURQ register holds the last User's request. The related request code can be retrieved by the command LURQ?. When such a request is reported, the corresponding bit in the Event Status register is set (bit URQ in EVTS).
Because the LURQ register is not used in the SK433, querying this register always returns 0 and the corresponding summary bit in the Event Status register is never set (bit URQ in EVTS).

4 Index of commands

5 Document Revision History

5.1 Version Number

This document is identified by SK433-SU01-P24A.

5.2 Revision History

P24A (2024-07-03)

Initial version.