RFS-2G42G5050(X)+ Programming Manual

AN-50-010

www.minicircuits.com

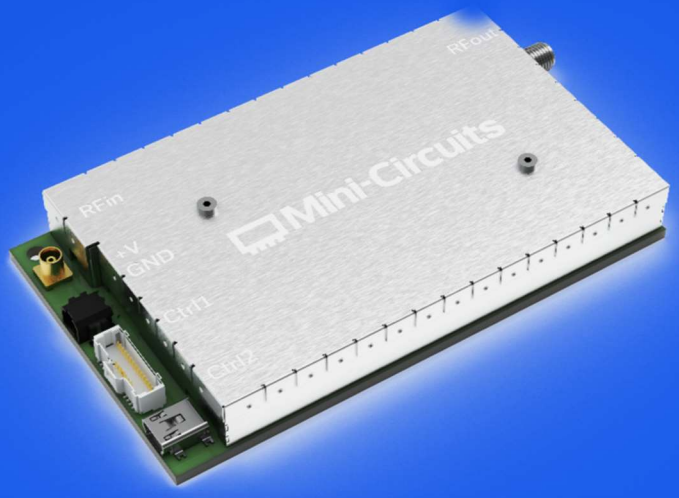


Table of Contents

1 Introduction

This document provides detailed information about the use of application and system-level commands supported by the firmware of the Mini-Circuits RFS-2G42G5050(X)+ high power signal source board. The RFS-2G42G5050(X)+ is a solid state connectorized 50W signal source and amplifier module which can be used in a wide range of industrial, scientific, and medical applications in the 2400-2500 MHz ISM band.

The ISC, RFS, and RFX series can be operated by sending text commands over their serial interface(s). These commands are part of a non-proprietary command protocol made to be both legible by humans and suitable for process automation through software. The serial command set enables users to get started quickly and communicate directly with RFS-2G42G5050(X)+ using nothing more than a standard USB cable and PC. Though this document is specific to the RFS-2G42G5050(X)+, most of the commands and functions described in this document are shared between products in the ISC, RFS, and RFX series.

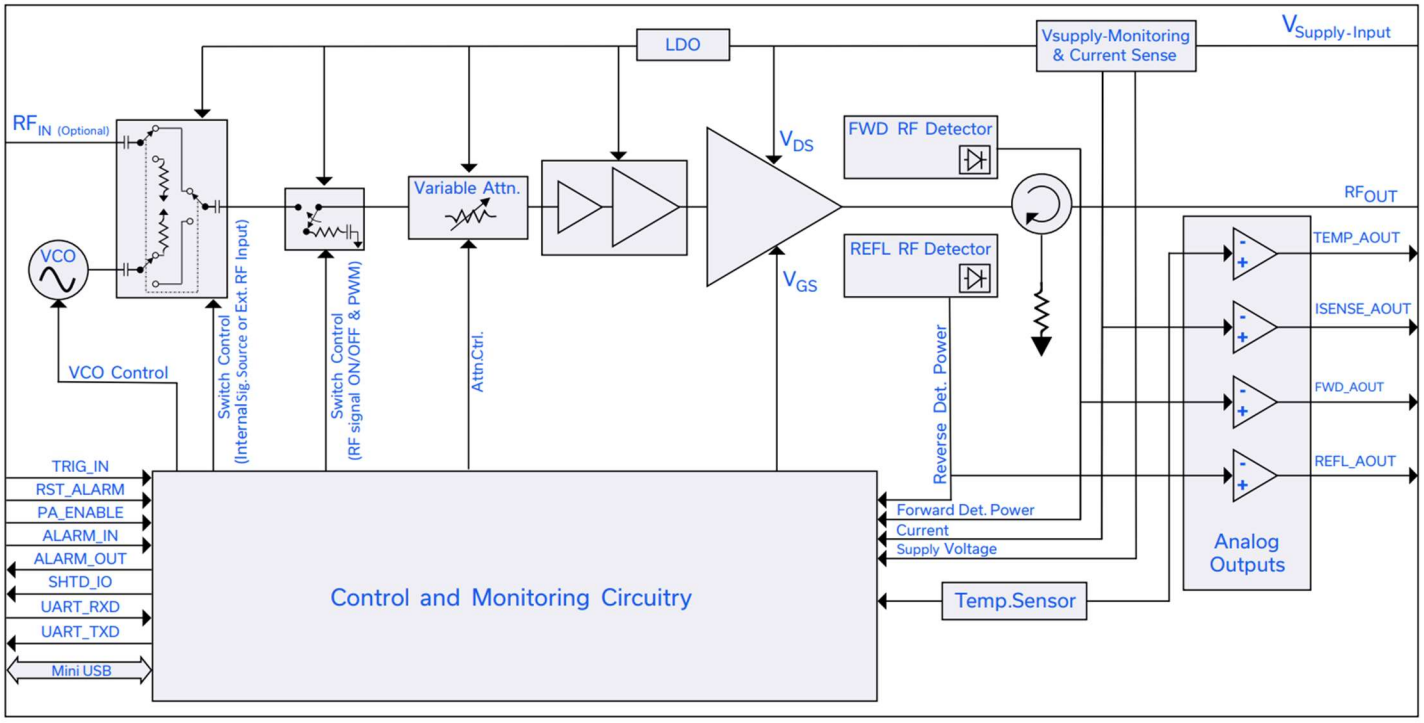
1.1 Style Conventions

This document uses the following style conventions:

`Monospace`	Denotes text that is entered at a keyboard, such as commands, file names, and source code
`[bracketed]`	Denotes a named argument in a command
`(bracketed)`	Denotes text that is optional or an optional argument in a command.

1.2 System Overview

Block Diagram



Functional Description

The RFS-2G42G5050(X)+ is a full RF Energy (RFE) Generator Subsystem incorporating the VCO, Input Switch, Variable Attenuators, Power Amplifier, Circulator, and Control/Monitoring/Protection Circuitry in a single compact module. All functions needed to monitor and control the RFS-2G42G5050(X)+ are accessible through the serial command set. A serial connection can be made over either the 3.3V UART pins on the "CTRL1" 30-pin connector, or the Mini-USB "CTRL2" connector. The same text-based serial commands are used to communicate with the module for both USB and UART mode and only one communication mode can be active at a time with the default being USB. Refer to the $COMS command in this document for switching between USB and UART communication modes. In addition to the serial interface, the RFS-2G42G5050(X)+ offers several functions through dedicated pins on the "CTRL1" 30-pin connector. A full description of these pins is outside of the scope of this document.

1.3 Setting up Communication

Setting up the communication between user and RFS-2G42G5050(X)+ board over USB is a straightforward process. Below is an example using PuTTY on a Windows or Linux PC.

USB Serial Connection with PuTTY

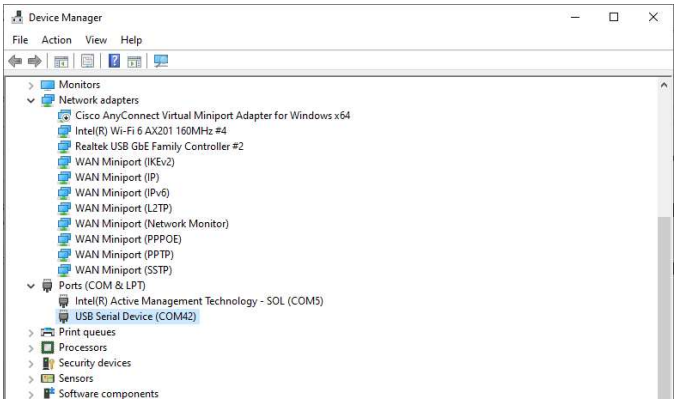
- Plug the RFS-2G42G5050(X)+ into the PC using a USB A to USB Mini-B cable.
- The RFS-2G42G5050(X)+ will appear as a virtual COM port. Find the port name of the device.

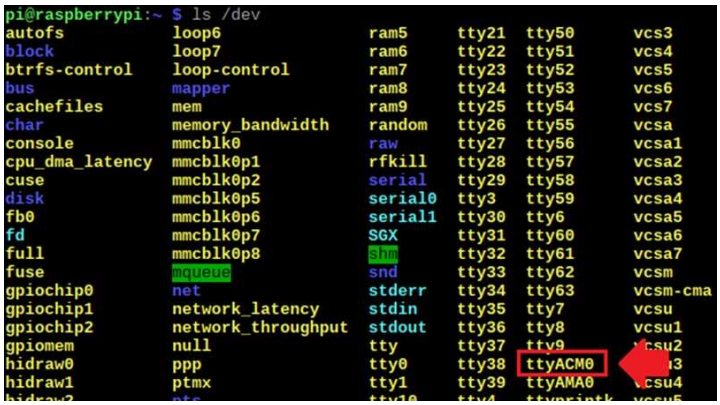
Windows:

Open the 'Device Manager' in Windows and find the port name of your device. It will show up as "USB Serial Device", followed by the port name.

Linux:

Open a terminal (CTRL + ALT + T) and use the `"ls /dev"` command to view available devices. The RFS-2G42G5050(X)+ should appear as a `"ttyACM"` device followed by a number.

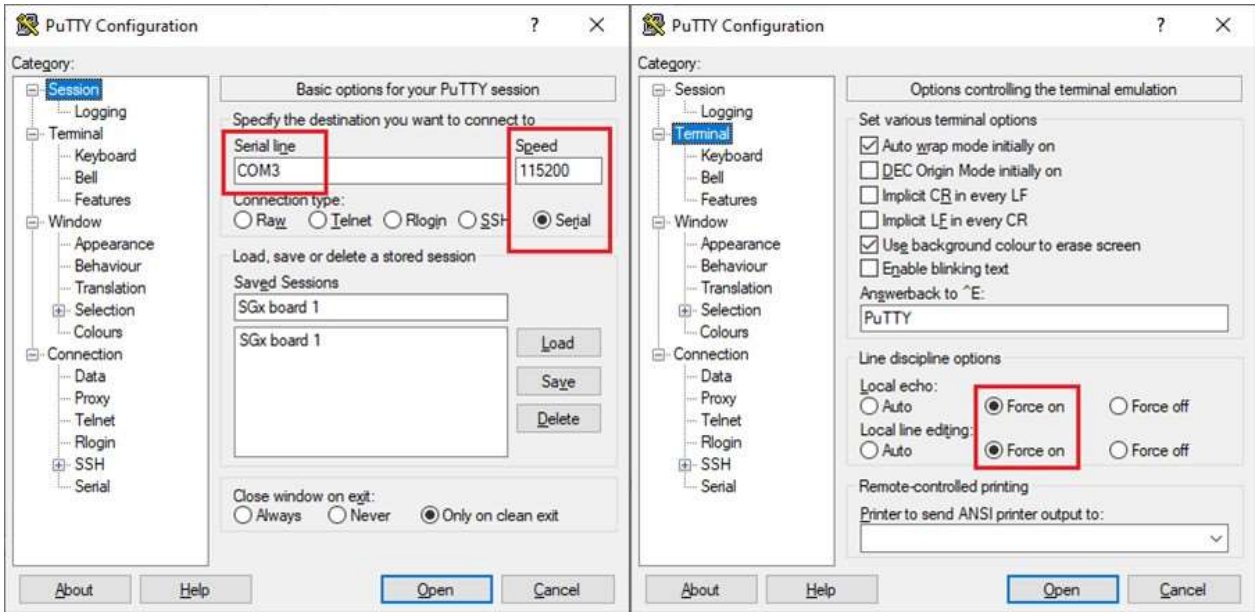




- Open PuTTY on your PC and provide the necessary information for the connection with the device.

Baud rate:	115200
Data bits:	8
Parity bits:	0
Stop bits:	1
Flow control:	none

- It is highly recommended to configure the “Terminal” category settings: “Local echo” and “Local line editing” to “Force on”.
- **Remark:** Linux requires the full path to the port (e.g. “/dev/ttyACM0”).



- Save the session, so that it won’t need to be reconfigured again in the future and press ‘Open’ to start a connection with the RFS-2G42G5050(X)+.
- A blank terminal window will pop up. Communication with the RFS board should now be established.

1.4 Sending a command

The command set can be divided into three categories:

- **Set commands** – These are used to write the setpoints and configurations of the RFS board and typically end with the letter 'S'. For example: setting the RF power level (PWRS), enabling DLL mode (DLES), etc.
- **Get commands** – These are used to read the values of the RFS board and, with a few exceptions, typically end with the letter 'G'. For example: printing out setpoints (PWRG), getting the temperature (PTG), etc.
- **Execute commands** – A few remaining commands that don't fall in either of the above categories. They will usually execute actions such resetting the board (RST) or performing a frequency sweep (SWP).

The general syntax for a command is as follows:

```
$[command],[channel],(parameter1),(parameter2),(...)\r\n
```

The protocol uses the '$' symbol as an indicator for the start of a command. A message sent without the '$' symbol will not be recognized as a command. Similarly, '\r\n' (new line) marks the end of a command. Without at least a '\r' or a '\n' the device will keep waiting for more bytes indefinitely.

There is one parameter that is used in every command: the channel identifier.

Each RFS is assigned a numeric channel identifier. The default value is 1, but when using more than one RFS in a multi-channel setup, it may be desirable to assign a unique number to each device beforehand so that they can be identified and differentiated during runtime.

When sending a command, it is necessary to include the correct channel number of the board. Otherwise, the message is ignored under the assumption that it is intended for a different device.

The channel '0', is accepted by every RFS device regardless of the assigned number. In single-channel systems, as well as in multi-channel systems where each RFS device has a dedicated serial connection, it is sufficient to always use channel identifier '0' when sending a command. In multi-channel systems that broadcast serial commands to multiple units over a bus, care should be taken to use the correct channel identifier.

1.5 Receiving a response

The RFS board provides feedback when a command succeeds.A successful set command will always reply with an OK:

```
$[command],[channel],OK\r\n
```

A successful get command will simply return the requested information:

```
$[command],[channel],[parameter1],(parameter2),(...)\r\n
```

Miscellaneous commands do not have a standardized way of relaying successful results but will in most cases be similar to the above examples or some combination of both.

All command response strings will include the name of the command sent and the channel id of the device sending the response. If a command is sent to channel '0', the response will return with the channel the connected device is set to, never channel '0'.

It is also possible for the execution of a command to fail. There can be several reasons for this:

- A mistake could be made when writing out a command.
- The command may not be executable on the device type or configuration.
- A runtime problem occurs.

If something goes wrong during command execution, the RFS board will reply with an error code, indicating that the action has failed. An error response looks as follows:

```
$[command],[channel],ERR##\r\n
```

The possible error codes and their respective descriptions are listed in the following:

Hex code	Error Description
0x01	Reserved
0x02	The serial message exceeded the maximum length.
0x03	The serial message had too few arguments.
0x04	The message had too many arguments.
0x05	The system could not accept this message is the current mode.
0x06	The system was busy and cannot process this message at this time.
0x07	The message was recognized but is not yet implemented in the codebase.
0x10	An argument was in error with the lower nibble indicating the argument number.
0x11	Argument 1 was invalid / out of range.
0x12	Argument 2 was invalid / out of range.
0x13	Argument 3 was invalid / out of range.
0x14	Argument 4 was invalid / out of range.

0x15	Argument 5 was invalid / out of range.
0x16	Argument 6 was invalid / out of range.
0x17	Argument 7 was invalid / out of range.
0x18	Argument 8 was invalid / out of range.
0x19	Argument 9 was invalid / out of range.
0x7E	Command execution failed.
0x7F	An error occurred that is not covered by any of the other error codes.

An example of an error response would be the following

Input:	$VER,1,1
Output:	$VER,1,ERR04

Hex code 0x04 indicates that the input command had too many arguments. This is correct, as the $VER command requires no additional arguments beyond the channel identifier.

1.6 Protocol

The protocol of the command set is straight forward:

1. Send a command.
2. Wait until a full response with a '\r\n' at the end is read.

It is instrumental to always wait for a complete reply from the device before sending another command. Sending commands without waiting for a full reply can result in communication problems. Sending commands in quick succession without waiting for the response may additionally overburden the RFS board with a growing list of tasks to complete, leaving it no time to perform safety operations and resulting in an automatic reset to break the cycle.

2 Basic Functions

These are the essential commands core to microwave generator functionality.

2.1 $ECG – Get RF output enable state

This command returns the enable state of the RFS board's RF output.

Syntax:

Input:	$ECG,[channel]
Output:	$ECG,[channel],[enable]

- **[channel]** – Channel identification number.
- **[enable]** – The RF output enable state.

  0 – OFF
  1 – ON

Example:

Input:	$ECG,1
Output:	$ECG,1,0

This indicates the RFS board's RF power output is disabled (default state)

2.2 $ECS – Set RF output enable state

This command turns RF output of the RFS board ON or OFF.

Syntax:

Input:	$ECS,[channel],[enable]
Output:	$ECS,[channel],OK

- **[channel]** – Channel identification number.
- **[enable]** – The RF output enable state.

  0 – OFF
  1 – ON

Example

Input:	$ECS,1,1
Output:	$ECS,1,OK

This enables the RF power output of the RFS board.

2.3 $FCG – Get frequency

This command returns the frequency of the RFS board's RF output in MHz.

Syntax:

Input:	$FCG,[channel]
Output:	$FCG,[channel],[frequency]

- **[channel]** – Channel identification number.
- **[frequency]** – The current frequency of the RF signal (MHz).

Example:

Input:	$FCG,1
Output:	$FCG,1,2450.000

This indicates the frequency is set to 2450 MHz (default setting)

2.4 $FCS – Set frequency

This command sets the frequency of the RFS board's RF output to the desired value in MHz.

Syntax:

Input:	$FCS,[channel],[frequency]
Output:	$FCS,[channel],OK

- **[channel]** – Channel identification number.
- **[frequency]** – The desired frequency setting for the RF signal (MHz).

Example:

Input:	$FCS,1,2450
Output:	$FCS,1,OK

This sets the frequency to 2450 MHz.

2.5 $PIG – Read PA Current

This command returns the DC current reading of the RFS in Amps.

Syntax:

Input:	$PIG,[channel]
Output:	$PIG,[channel],[current]

- **[channel]** – Channel identification number.
- **[current]** – DC current reading (A).

Example:

Input:	$PIG,1
Output:	$PIG,1,12.45

This indicates the DC Current is 12.45 A

2.6 $PPDG – Get PA forward and reflected power in dBm

This command returns the measured forward and reflected power of the PA in dBm.

Syntax:

Input:	$PPDG,[channel]
Output:	$PPDG,[channel],[forward power],[reflected power]

- **[channel]** – Channel identification number.
- **[forward power]** – The measured RF power output of the PA (dBm).
- **[reflected power]** – The measured RF power reflected back into the PA (dBm).

Example:

Input:	$PPDG,1
Output:	$PPDG,1,47.00000,27.00000

This indicates the PA is outputting 47dBm of forward power, but 27dBm is being reflected back. This corresponds to an S11 of -20dB.

2.7 $PPG – Get PA forward and reflected[1] power in Watt

This command returns the measured forward and reflected power of the PA in Watt.

Syntax:

Input:	$PPG,[channel]
Output:	$PPG,[channel],[forward power],[reflected power]

- **[channel]** – Channel identification number.
- **[forward power]** – The measured RF power output of the PA (W).
- **[reflected power]** – The measured RF power reflected back into the PA (W).

Example:

Input:	$PPDG,1
Output:	$PPDG,1,50.00000,0.50000

[1] Reflected power readings include power going into the amplifier "RF Output" port from reflections of mismatched loads as well as from external sources

This indicates the PA is outputting 50W of forward power, and 0.5W is being reflected back. This corresponds to an S11 of -20dB.

Note: Reflected power readings include power going into the amplifier from other sources

2.8 $PTG – Get PA temperature

This command returns the temperature of the PA in degrees C.

Syntax:

Input:	$PTG,[channel]
Output:	$PTG,[channel],[temperature]

- **[channel]** – Channel identification number.
- **[temperature]** – The current temperature in °C.

Example:

Input:	$PTG,1
Output:	$PTG,1,42.7

This indicates that the current temperature of the PA is 42.7°C.

2.9 $PVG – Read PA Voltage

This command returns the measured DC Voltage of the PA in Volts.

Syntax:

Input:	$PVG,[channel]
Output:	$PVG,[channel],[voltage]

- **[channel]** – Channel identification number.
- **[voltage]** – The current voltage (V).

Example:

Input:	$PVG,1
Output:	$PVG,1,32.00

This indicates that the current voltage of the PA is 32.00V.

2.10 $PWRDG – Get PA output power setpoint in dBm

This command returns the configured output power setpoint in dBm. In autogain mode, output power is adjusted so the forward power reading tracks the output power setpoint. In feedforward mode, the output power setpoint maps to a static internal bias and attenuation setting (See $AGES/$AGEG)

Syntax:

Input:	$PWRDG,[channel]

Output:	$PWRDG,[channel],[power]

- **[channel]** – Channel identification number.
- **[power]** – The current output power setpoint (dBm).

Example:

Input:	$PWRDG,1,
Output:	$PWRDG,1,0.000000

This indicates that the output power setpoint is configured to 0 dBm (default setting).

2.11 $PWRDS – Set PA output power setpoint in dBm

This command configures the output power setpoint in dBm. In autogain mode, output power is adjusted so the forward power reading tracks the output power setpoint. In feedforward mode, the output power setpoint maps to a static internal bias and attenuation setting (See $AGES/$AGEG)

Syntax:

Input:	$PWRDS,[channel],[power]
Output:	$PWRDS,[channel],OK

- **[channel]** – Channel identification number.
- **[power]** – Output power setpoint (dBm).

Example:

Input:	$PWRDG,1,47
Output:	$PWRDG,1,OK

This configures the output power setpoint to 47 dBm.

2.12 $PWRG – Get PA output power setpoint in Watts

This command returns the configured output power setpoint in Watts. In autogain mode, output power is adjusted so the forward power reading tracks the output power setpoint. In feedforward mode, the output power setpoint maps to a static internal bias and attenuation setting (See $AGES/$AGEG)

Syntax:

Input:	$PWRG,[channel]
Output:	$PWRG,[channel],[power]

- **[channel]** – Channel identification number.
- **[power]** – The current output power setpoint (W).

Example:

Input:	$PWRG,1,

Output:	$PWRG,1,0.001000

This indicates that the output power setpoint is configured to 0.001W (default setting).

2.13 $PWRS – Set PA output power setpoint in Watts

This command configures the output power setpoint in Watts. In autogain mode, output power is adjusted so the forward power reading tracks the output power setpoint. In feedforward mode, the output power setpoint maps to a static internal bias and attenuation setting (See $AGES/$AGEG)

Syntax:

Input:	$PWRS,[channel],[power]
Output:	$PWRS,[channel],OK

- **[channel]** – Channel identification number.
- **[power]** – Output power setpoint (W).

Example:

Input:	$PWRS,1,50
Output:	$PWRS,1,OK

This configures the output power setpoint to 50W.

2.14 $RFSG – Get RF Source Configuration

This command will return the current signal source configuration. The source can be configured to “Internal VCO” or “External RF input”

Syntax:

Input:	$RFSG,[channel]
Output:	$RFSG,[channel],[source]

- **[channel]** – Channel identification number.
- **[source]** – Current RF source

  0 – Internal VCO
  1 – External RF input

Example:

Input:	$RFSG,1
Output:	$RFSG,1,0

This indicates that the Internal VCO is currently the RF source (default setting)

2.15 $RFSS – Switch RF Source Between Internal VCO and RF Input

This command configures the current signal source configuration. The source can be configured to “Internal VCO” or “External RF input”. The default state is “Internal VCO”, option 0.

When switching the signal source to External RF Input from Internal VCO mode with “$RFSS,1,1”, several actions are taken:

1. RF is disabled ($ECS,1,0)
2. The attenuation is set to 0dB ($GCS,1,0)
3. The magnitude is set to 50% ($MCS,1,50)
4. Feedforward mode is applied ($AGES,1,0)

When in External RF Input mode, the gain of the amplifier can be controlled with $GCS and $MCS commands.

When switching the signal source to External RF Input from Internal VCO mode with “$RFSS,1,0”, several actions are taken:

1. RF is disabled ($ECS,1,0)
2. Autogain mode is applied ($AGES,1,1)

Syntax:

Input:	$RFSS,[channel],[source]
Output:	$RFSS,[channel]

- **[channel]** – Channel identification number.
- **[source]** – Current RF source

  0 – Internal VCO
  1 – External RF input

Example:

Input:	$RFSS,1,0
Output:	$RFSS,1

This configures the RF Source to “Internal VCO” (default setting)

3 Information Request

Request information about the RF Generator Board

3.1 $IDN – Get device identification string

This command returns the identification string signal source.

Syntax:

Input:	$IDN,[channel]
Output:	$IDN,[channel],[manufacturer],[device_name],[serial_number]

- **[channel]** – Channel identification number.
- **[manufacturer]** – Name of the manufacturer:
  Mini-Circuits
- **[device_name]** – Name of the signal generator or system controller. The following types are possible:
  RFS-2G42G5050+
  ISC-2425-25+[2]
- **[serial_number]** – unique serial number of the board.

Device name template:

[AAA]-[LLL][HHH][PPP](X)+

- **[AAA]** – Device type identifier. Possibilities are: ISC, RFS, RFX.
  ISC: Industrial System Controller & Small-Signal Source (<1W)
  RFS: RF Energy Power Amplifier & High Power Source (10W+)
- **[LLL]** – Lower frequency limit. 2G4 indicates 2.4GHz. G90 indicates 900 MHz
- **[HHH]** – Upper frequency limit. 2G5 indicates 2.5GHz. G93 indicates 930 MHz
- **[PPP]** – Maximum RF output power of the signal generator board. For RFS device types, the power is in Watts. For ISC and RFX device types, the power is in dBm.
- **(X)** – An "X" in the device name indicates that the device needs to be mounted to an additional heat sink or cooling plate to operate. "X" versions of amplifiers are "non-heatsink" options.

Example:

Input:	$IDN,1
Output:	$IDN,1,Mini-Circuits,RFS-2G42G5050+,MN0000102101

[2] ISC-2425-25+ device name does not strictly adhere to the established naming convention

This indicates the connected RFS board is an RFS-2G42G5050+ for use in the 2450 MHz ISM band, with serial number MN0000102101.

3.2 $RTG – Get uptime of the RFS board

This command returns the uptime of the RFS board since its initialization. The uptime count restarts when the board is reset.

Syntax:

Input:	$RTG,[channel]
Output:	$RTG,[channel],[uptime]

- **[channel]** – Channel identification number.
- **[uptime]** – The uptime in seconds.

Example:

Input:	$RTG,1
Output:	$RTG,1,51

This indicates that the RFS board has been running for 51 seconds.

3.3 $VER – Get firmware version

This command returns the current version of the firmware.

Syntax:

Input:	$VER,[channel]
Output:	$VER,[channel],[manufacturer identifier],[major revision], [minor revision],[build],(hotfix),[date stamp],[time stamp]

- **[channel]** – Channel identification number.
- **[manufacturer identifier]** – Firmware developer identifier.
- **[major revision]** – The version's major revision number.
- **[minor revision]** – The version's minor revision number.
- **[build]** – The version's build number.
- **(hotfix)** – (optional) The version's hotfix number.
- **[date stamp] –** The date on which the firmware was compiled.
- **[time stamp]** – The time at which the firmware was compiled.

Example:

Input:	$VER,1
Output:	$VER,1,Mini-Circuits,2,7,8,Sep 21 2023,12:44:20

This indicates the firmware version is 2.7.8, compiled on September 21st, 2023.

4 Pulse-Width Modulation

The RFS-2G42G5050X+ has the capability to produce a PWM signal when configured as a signal source. It also has the capability to read the forward power and reflected power during the pulse when configured as a signal source or when configured as a stand-alone PA using an external PWM input signal.

When configured as a signal source, the module can generate a modulated signal without any external RF or trigger inputs. However, the internal RF signal source can be modulated with an external trigger, TRIG_IN, on pin 1 of Crtl1 using the $ETS command to enable this feature. When configured as a stand-alone amplifier with an external PWM RF input signal a trigger, TRIG_IN, is required on pin 1 of Ctrl1 to accurately read the forward and reflect power during the pulse. This feature is enabled using the $ETSS command. More details of these and additional PWM commands can be found in the application note AN-50-010 Programming Manual.

Pulse Width Modulation allows the user to modulate the RF signal, and in turn the average power output of the system, by turning the signal ON and OFF at a set rate.

The following two parameters are used to achieve this:

PWM frequency – Dictates how often the signal switches between ON and OFF.

PWM duty cycle – Dictates the time ratio between ON and OFF each period.

Depending on the duty cycle, the average power output of the system will decrease to a percentage of its set output. For example, 50% duty cycle at 50W results in an average RF power output of 25W. To ensure sufficient measurement time of the RF signal, the pulses generated by any PWM scheme must not be too short. Hence, the permissible PWM frequency and duty cycle are dependent on each other. The PWM frequency can vary between 1000 Hz – 19800Hz. To ensure accurate power readings (and therefore accurate power output), the minimum value of the duty cycle changes along with the PWM frequency according to the following formula:

DCmin = ROUNDUP (fPWM * Tmin pulse width/10,000)

Where:

DCmin is the minimum duty cycle as a percentage.

fPWM is the PWM frequency between 1000 and 19800 Hz.

Tmin pulse width is the minimum pulse length in microseconds. The minimum pulse width to ensure sufficient measurement time of the RF signal is 50µs.

This means that at 1000 Hz, the minimum duty cycle is 5% and at 19800 Hz it is 99%. Going over this frequency value would effectively disable PWM, as the minimum duty cycle becomes 100%. The user needs to keep these limitations in mind – the system will not check for the limits. For a reasonable control range, PWM frequencies between 1 and 2 kHz are recommended.

4.1 $DCFS – Set PWM Frequency

This command sets the frequency of the PWM signal

Syntax:

Input:	$DCFS,[channel],[frequency],[reserved]
Output:	$DCFS,[channel],OK

- **[channel]** – Channel identification number.
- **[frequency]** – PWM frequency in Hz.
- **[reserved]** – Reserved. This parameter should only be written 0.

Example:

Input:	$DCFS,1,1200,0
Output:	$DCFS,1,OK

This sets the PWM frequency to 1200 Hz.

4.2 $DCG – Get PWM Duty Cycle

This command returns all the settings relating to PWM.

Syntax:

Input:	$DCG,[channel]
Output:	$DCG,[channel],[frequency],[reserved],[trigger mode],[reserved],[reserved],[reserved],[reserved],[reserved],[duty cycle]

- **[channel]** – Channel identification number.
- **[frequency]** – The current PWM frequency.
- **[trigger mode]** – The current operational mode of the PWM triggering.

  1 – free running
  2 – Reserved. Parameter should be ignored.
  3 – Reserved. Parameter should be ignored.
- **[duty cycle]** – The current duty cycle percentage value.
- **[reserved]** – Reserved. Parameters should be ignored.

Example:

Input:	$DCG,1
Output:	$DCG,1,1000,0,1,255,255,255,255,0.000000,50

This indicates that the operational mode is configured to ‘free running’.

The PWM signal is configured to a frequency of 1000 Hz with a duty cycle of 50%.There is no correction factor and no delay.

4.3 $DCS – Set PWM Duty Cycle

This command sets the PWM duty cycle between 0% and 100%.

Note: This command doubles as a PWM ON/OFF switch. Setting the duty cycle to 100% is the same as turning PWM off entirely. There is no dedicated PWM Enable command

Syntax:

Input:	$DCS,[channel],[duty cycle]
Output:	$DCS,[channel],OK

- **[channel]** – Channel identification number.
- **[duty cycle]**[3] – PWM duty cycle in %.

Example:

Input:	$DCS,1,50
Output:	$DCS,1,OK

This sets the PWM duty cycle to 50%.

[3] Please note that the duty cycles that would result in a pulse width of less than 50µs will be rejected

5 DLL and Sweep

In RF energy systems, frequency adjustment is used to optimize power transfer into the load. There are two built-in frequency adjustment routines that help the user find, then switch to the optimal frequency.

The sweep function performs a sweep over a user defined frequency range and reports back the forward and reflected powers. The user can request that data be returned on all the frequency points or just request the data on the frequency point with the best forward to reflected power ratio. This document will use the terms Return Loss, S11, and reflection coefficient interchangeably to denote the forward to reflected power ratio. The forward to reflected power ratio can be called Return Loss or S11. The sweep function searches the entire range for the optimal S11, so it will find the global minimum within the granularity of the step size. If a sweep is executed in the middle of a sensitive process, the fluctuating power delivered to the load caused by the sweep itself may not be tolerable, in which case, DLL mode may be preferred. The output of the sweep functions $SWP or $SWPD can be used to generate S11 plots similar to a network analyzer.

Digital Locked Loop (DLL) mode is another way to optimize frequency that is intended to be used during a process application. If the measured S11 is less than the threshold (poor match), then DLL will continuously sweep the frequency from 'lower frequency' to 'upper frequency' in steps of 'frequency step' until S11 exceeds the threshold (good match). While S11 exceeds the threshold, the DLL will use a local search strategy to fine-tune the frequency. Every step of the DLL, the current frequency is updated to the frequency with the best S11 of the three points (current frequency +/- frequency step). This way, the DLL can track optimal frequencies that move with changes in cavity temperature, pressure, or contents over the duration of a process application.

When DLL mode enabled, the only thing that changes operation-wise are the constant automatic adjustments to frequency setting, so it is possible to mix and match with other modes such as feed-forward mode, or There are no dropouts in power or fluctuations in frequency when the frequency is changed in the RFS-2G52G5050X+, so it is appropriate for sensitive plasma applications.

5.1 $DLCG – Get DLL configuration

This command returns the configured parameters of the DLL mode.

Syntax:

Input:	$DLCG,1
Output:	$DLCG,[channel],[lower frequency],[upper frequency],[start frequency],[frequency step],[threshold],[main delay]

- **[channel]** – Channel identification number.
- **[lower frequency]** – The lower boundary of the bandwidth for DLL in MHz.
- **[upper frequency]** – The upper boundary of the bandwidth for DLL in MHz.
- **[start frequency]** – The frequency at which the DLL starts its activities in MHz.
- **[frequency step]** – The step size of the DLL in MHz.
- **[threshold]** – The match/efficiency threshold in dB to be met before DLL latches onto a frequency.
- **[main delay]** – The delay between complete runs of the DLL in ms.

Example:

Input:	$DLCG,1
Output:	$DLCG,1,2400.000000,2500.000000,2450.000000,1.000000,0.000000,1

This indicates that DLL is configured to function within the range of 2400-2500 MHz, starting off initially at a frequency of 2450MHz. It makes 1MHz steps with a threshold of 0dB. There is 1ms of delay between DLL runs (default).

5.2 $DLCS – Set DLL configuration

This command configures the parameters of DLL mode.

Syntax:

Input:	$DLCS,[channel],[lower frequency],[upper frequency], [start frequency],[frequency step],[threshold],[main delay]
Output:	$DLCS,[channel],OK

- **[channel]** – Channel identification number.
- **[lower frequency]** – The lower boundary of the bandwidth for DLL in MHz.
- **[upper frequency]** – The upper boundary of the bandwidth for DLL in MHz.
- **[start frequency]** – The frequency at which the DLL starts its activities in MHz.
- **[frequency step]** – The step size of the DLL in MHz.
- **[threshold]** – The match/efficiency threshold in dB to be met before DLL latches onto a frequency.
- **[main delay]** – The delay between complete runs of the DLL in ms.

Example:

Input:	$DLCS,1,2400,2500,2410,5,0.5,25
Output:	$DLCS,1,OK

This sets lower frequency to 2400 MHz, upper frequency to 2500 MHz, start frequency to 2410 MHz, frequency step to 5MHz, threshold to 0.5 dB and the main delay to 25 ms.

5.3 $DLEG – Get DLL enable state

This command indicates whether DLL mode is currently turned ON or OFF.

Syntax:

Input:	$DLEG,[channel]
Output:	$DLEG,[channel],[enable]

- **[channel]** – Channel identification number.
- **[enable]** – The current enable state of the DLL mode.

  0 – OFF (default)
  1 – ON

Example:

Input:	$DLEG,1
Output:	$DLEG,1,0

This indicates that the DLL is currently not enabled (default).

5.4 $DLES – Set DLL enabled / disabled

This command sets the enable state of the DLL. If DLL is disabled, the amplifier is said to be in CW mode.

Syntax:

Input:	$DLES,[channel],[enable]
Output:	$DLES,[channel],OK

- **[channel]** – Channel identification number.
- **[enable]** – The enable state of the DLL.

  0 – OFF (default)
  1 – ON

Example:

Input:	$DLES,1,1
Output:	$DLES,1,OK

This turns on DLL mode.

5.5 $SWP – Perform frequency sweep with powers in Watts

This command executes a user defined frequency sweep at an output power defined in Watts. Forward and reflected power is measured at every point in the sweep and are also reported in Watts.

The completion time of the command will increase as the number of frequency steps increases. This can make it seem as if the RFS board has become unresponsive for some time.

This command offers two output modes, which have different output syntaxes.

Syntax:

Input:	$SWP,[channel],[start frequency],[stop frequency],[step frequency],[power watt],[output mode]
Output (mode 0):	$SWP,[channel],[measurement frequency 0],[forward power 0], [reflected power 0] $SWP,[channel],[measurement frequency 1],[forward power 1], [reflected power 1] … $SWP,[channel],OK
Output (mode 1):	$SWP,[channel],[measurement frequency],[forward power], [reflected power]

- **[channel]** – Channel identification number.
- **[start frequency]** – The beginning of the sweep bandwidth in MHz.
- **[stop frequency]** – The end of the sweep bandwidth in MHz.
- **[step frequency]** – The size of the steps taken between each measurement in MHz.
- **[power watt]** – The output power at which the sweep is performed in Watt.
- **[output mode]** – Dictates what the sweep will output.

  0 – Return all sweep results. It returns one line of text for every point in the sweep, as well as an OK message at the end. All responses arrive simultaneously when the command finishes.

1 – Returns only the best match sweep result. The current frequency and the DLL start frequency is set to this value after the completion of the sweep.

- **[measurement frequency]** – The frequency at which is measured in MHz. The output frequency value is rounded to the 2nd decimal.
- **[forward power]** – The forward power measured at [measurement frequency] in Watt.
- **[reflected power]** – The reflected power measured at [measurement frequency] in Watt.

Example 1:

Input:	$SWP,1,2400,2500,10,100,0
Output:	$SWP,1,2400,10.01,2.01 $SWP,1,2410,9.84,2.00 $SWP,1,2420,9.88,1.95 $SWP,1,2430,9.97,1.97 $SWP,1,2440,10.10,1.98 $SWP,1,2450,9.87,1.87 $SWP,1,2460,9.99,0.75 $SWP,1,2470,9.91,0.21 $SWP,1,2480,10.00,0.69 $SWP,1,2490,9.84,1.44 $SWP,1,2500,9.83,1.89 $SWP,1,OK

This executes an S11 sweep in the 2400 - 2500 MHz range with a step size of 10 MHz, at 100 W output power.

At 2400 MHz the forward power is 100.01 W and the reflected power is 20.12 W. At 2410 MHz, the forward power is 99.84 W and the reflected power is 20.08 W. etc...

When the sweep has run through the entire frequency band “$SWP,1,OK” is returned to indicate it is finished.

Example 2:

Input:	$SWP,1,2400,2500,10,100,1
Output:	$SWP,1,2470,99.91,2.15

This executes an S11 sweep in the 2400 - 2500 MHz range with a step size of 10 MHz, at 100 W output power. However, this time output mode 1 is used, which returns only the best match result.

At 2470 MHz, the forward power is 99.91 W and the reflected power is 2.15 W. This frequency offers the best match or ratio between forward and reflected powers. The current frequency setting and the DLL start frequency are now set to 2470MHz.

5.6 $SWPD – Perform frequency sweep with powers in dBm

This command executes a user defined frequency sweep at an output power defined in dBm. Forward and reflected power is measured at every point in the sweep and are also reported in dBm.

The completion time of the command will increase as the number of frequency steps increases. This can make it seem as if the RFS board has become unresponsive for some time.

This command offers two output modes, which have different output syntaxes.

Syntax:

Input:	$SWPD,[channel],[start frequency],[stop frequency],[frequency step],[power watt],[output mode]
Output (mode 0):	$SWPD,[channel],[measurement frequency 0],[forward power 0], [reflected power 0] $SWPD,[channel],[measurement frequency 1],[forward power 1], [reflected power 1] … $SWPD,[channel],OK
Output (mode 1):	$SWPD,[channel],[measurement frequency],[forward power], [reflected power]

- **[channel]** – Channel identification number.
- **[start frequency]** – The beginning of the sweep bandwidth in MHz.
- **[stop frequency]** – The end of the sweep bandwidth in MHz.
- **[frequency step]** – The size of the steps taken between each measurement in MHz.
- **[power dbm]** – The output power at which the sweep is performed in dBm.
- **[output mode]** – Dictates what the sweep will output.

  0 – Return all sweep results. It returns one line of text for every point in the sweep, as well as an OK message at the end. All responses arrive simultaneously when the command finishes.
  1 – Returns only the best match sweep result. The current frequency and the DLL start frequency is set to this value after the completion of the sweep.
- **[measurement frequency]** – The frequency at which is measured in MHz. The output frequency value is rounded to the 2nd decimal.
- **[forward power]** – The forward power measured at [measurement frequency] in dBm.
- **[reflected power]** – The reflected power measured at [measurement frequency] in dBm.

Example 1:

Input:	$SWPD,1,2400,2500,10,40,0
Output:	$SWPD,1,2400,40.02,33.03 $SWPD,1,2410, 40.10,33.01 $SWPD,1,2420, 40.04,32.90 $SWPD,1,2430, 39.98,32.94 $SWPD,1,2440,40.07,32.97 $SWPD,1,2450,39.89,32.72 $SWPD,1,2460,39.97,28.75 $SWPD,1,2470,40.01,23.22 $SWPD,1,2480,40.12,28.39 $SWPD,1,2490,40.05,31.58 $SWPD,1,2500,39.99,32.76 $SWPD,1,OK

This executes a measurement sweep in the 2400 - 2500 MHz range with a step size of 10 MHz, at 40 dBm output power.

At 2400 MHz the forward power is 40.02 dBm and the reflected power is 33.03 dBm. At 2410 MHz the forward power is 40.10 dBm and the reflected power is 33.01 dBm. etc...

When the sweep has run through the entire frequency band “$SWPD,1,OK” is returned to indicate it is finished.

Example 2:

Input:	$SWPD,1,2400,2500,10,40,1
Output:	$SWPD,1,2470,40.01,23.22

This executes an S11 sweep in the 2400 - 2500 MHz range with a step size of 10 MHz, at 40 dBm output power. However, this time output mode 1 is used, which returns only the best match result.

At 2470 MHz the forward power is 40.01 dBm and the reflected power is 23.22 dBm. The current frequency setting and the DLL start frequency are now set to 2470MHz.

6 Manual Power Control

In the default operation of the RFS-2G42G5050X+, a power is set by the user with the $PWRS/$PWRDS command and the autogain feedback control loop will adjust the output power of the generator to minimize the difference between the power measured by the forward power detectors and the set power. The RFS-2G42G5050X+ has two complementary methods for output power adjustment that work together to provide continuous adjustment over the full operating range. The Digital Step Attenuator (DSA) 'Gain' component provides the coarse method to control power in steps of 0.25 dB over a 31.75 dB range. The PA biasing 'Magnitude' component provides near-continuous fine-tuning adjustment over about a +/- 0.5 dB range. The autogain algorithm is constantly adjusting the DSA and bias settings to match the power set by the user. In doing so, autogain compensates for system drifts caused by fluctuations in temperature, DC Voltage, or other factors.

Although it is not recommended for the general use-case, it is possible to disable the autogain feedback control loop and stop automatic updates of the DSA and bias settings. These settings can then be controlled manually. This manual power control mode is called *Feed-Forward* mode, as the feedback has been removed. In feed-forward mode, $PWRS/$PWRDS commands are still used to set the forward power, but their function changes. When a power is set using $PWRS/$PWRDS in feed forward mode, the DSA is set to the attenuation that would result in the requested power. The mapping between the power setting and the DSA setting is defined by the feed-forward calibration performed at the factory. The feed-forward calibration is defined over frequency and over power at a specified default PA bias setting (50%) and at 28V, 25°C, into a matched load. The DSA attenuation and the PA bias setting states can also be changed directly or requested which can be useful for special cases or for system debugging.

6.1 $AGEG – Get auto-gain enabled / disabled

This command returns the enable state of the auto-gain algorithm.

Syntax:

Input:	$AGEG,[channel]
Output:	$AGEG,[channel],[enable]

- **[channel]** – Channel identification number.
- **[enable]** – The current enable state of the auto-gain algorithm.

  0 – OFF
  1 – ON

Example:

Input:	$AGEG,1
Output:	$AGEG,1,1

This indicates the auto-gain algorithm is enabled (default)

6.2 $AGES – Set auto-gain enabled / disabled

This command turns the auto-gain algorithm ON or OFF.

The auto-gain algorithm automatically regulates the power output of the RFS board by configuring the DSA and PA bias according to calibrations that are stored in the device's EEPROM and feedback from the PA.

When auto-gain is enabled, the user can simply request an arbitrary amount of power (in Watt / dBm) from their RF system, and the requested power will be accurately generated (As long as the calibration is good and there are no unexpected interferences).

When auto-gain is disabled, the user can take manual control of the DSA and PA bias. Operating manually is not recommended in most situations but can be useful for troubleshooting and characterizing RF systems.

Disabling auto-gain has consequences for a variety of commands:

- $PWRS and $PWRDS set the DSA state according to the static feed-forward calibration stored in the EEPROM
- Power can be regulated manually using commands like $MCS and $GCS to control the DSA and PA bias directly.
- $SWP and $SWPD ignore the [Sweep Power] argument. Sweeps are performed at whatever power output is configured through the DSA and IQ Modulator at the time.

Syntax:

Input:	$AGES,[channel],[enable]
Output:	$AGES,[channel],OK

- **[channel]** – Channel identification number.
- **[enable]** – The desired enable state of the auto-gain algorithm.

  0 – OFF
  1 – ON

Example:

Input:	$AGES,1,0
Output:	$AGES,1,OK

This disables auto-gain. Putting the generator in “feed-forward” mode

6.3 $GCG – Get DSA attenuation in dB

This command returns the configured attenuation value of the DSA which regulates the RFS board's power output. The higher the value, the lower the power output.

Syntax:

Input:	$GCG,[channel]
Output:	$GCG,[channel],[attenuation]

- **[channel]** – Channel identification number.
- **[attenuation]** – The current attenuation value of the DSA.

  Attenuation Range: 0 – 31.75 dB
  Minimum step size: 0.25 dB

Example:

Input:	$GCG,1
Output:	$GCG,1,10

This indicates the configured attenuation of the DSA is 10dB.

6.4 $GCS – Set DSA attenuation in dB

This command sets the attenuation of the DSA which regulates the RFS board's power output. The higher the value, the lower the power output.

The power output of the RFS board is a result of both the DSA and the IQ modulator working together. The DSA provides a coarse method to control power, whereas the IQ modulator provides a fine method.

Remark: To use this command, auto-gain must be disabled first.

Syntax:

Input:	$GCS,[channel],[attenuation]
Output:	$GCS,[channel],OK

- **[channel]** – Channel identification number.
- **[attenuation]** – The desired attenuation value of the DSA.

  Attenuation Range: 0 – 31.75 dB
  Minimum step size: 0.25 dB

Example:

Input:	$GCS,1,7
Output:	$GCS,1,OK

This will set the attenuation of the DSA to 7dB.

6.5 $MCG – Get magnitude in percent

This command gets the magnitude of the IQ modulator.

Syntax:

Input:	$MCG,[channel]
Output:	$MCG,[channel],[magnitude]

- **[channel]** – Channel identification number.
- **[magnitude]** – The current magnitude configuration of the PA bias in percent.

Example:

Input:	$MCG,1
Output:	$MCG,1,50

This indicates the magnitude of the PA bias is configured to 50%.

6.6 $MCS – Set magnitude in percent

This command sets the PA bias magnitude setting which regulates the RFS board's power output. The higher the value, the higher the power output.

The power output of the RFS board is a result of both the DSA and the PA bias working together. The DSA “gain” setting provides a coarse method to control power, whereas the PA bias “magnitude” setting provides a fine method.

The valid range of magnitudes for the RFS-2G42G5050X+ is 44.6%-56.1%. If the magnitude setting is out of the valid range, then the magnitude will be set to the closest value in the valid range. The default magnitude setting is 50% and the sensitivity of output power to magnitude is roughly 0.2dB/%. The resolution of the magnitude setting is 0.1%, meaning the output power resolution is typically 0.02dB. Sensitivity of the magnitude setting is highly dependent on output power.

To use this command, auto-gain must be disabled first.

Syntax:

Input:	$MCS,[channel],[magnitude]
Output:	$MCS,[channel],OK

- **[channel]** – Channel identification number.
- **[magnitude]** – The desired magnitude of the PA bias in percent.

Example:

Input:	$MCS,1,50.3
Output:	$MCS,1,OK

This will set the magnitude of the PA bias to 53.1%, typically resulting in a roughly +0.06dB shift from nominal.

7 External Triggering

When the RFS-2G42G5050X+ is set to PWM mode, the ADCs are automatically triggered to sample the forward and reflected power during the ON time of the pulse. When in amplifier mode, it is possible to supply an externally modulated signal to RF IN. This requires that a TTL signal be sent to the RFS to trigger the forward and reflected power ADCs to sample when the RF is ON. It is also possible to modulate the RF signal (On/Off) via TTL input. In this case, the ADC sampling trigger and the modulation trigger are the same TTL signal to TRIG_IN. The external triggering commands in this section allow the configuration of the system in these three cases.

7.1 $ETG – External Trigger Get

The external trigger setting status. See $ETS for description of setting.

Syntax:

Input:	$ETG,[channel]
Output:	$ETG,[channel],[enable]

- **[channel]** – Channel identification number.
- **[enable]** – External Trigger enable state.

  0 – OFF (The RF is internally triggered)
  1 – ON (The RF is externally triggered)

Example:

Input:	$ETG,1
Output:	$ETG,1,0

This indicates that external triggering is turned off (default state).

7.2 $ETS – External Trigger Set

The external trigger setting allows the TRIG_IN port to modulate the signal via “RF signal ON/OFF & PWM switch” on the RFS-2G42G5050X+. When the setting is enabled, a TTL high on TRIG_IN will turn the switch on and a TTL low on TRIG_IN will turn the switch off. When the setting is disabled (default), the switch is controlled by the RF Enable or modulated internally as in PWM Mode. If $ETSS is also enabled, then the power measurements will be synchronized with the rising edge of TRIG_IN.

Syntax:

Input:	$ETS,[channel],[enable]
Output:	$ETS,[channel],OK

- **[channel]** – Channel identification number.
- **[enable]** – External Trigger enable state.

  0 – OFF (The RF is internally triggered)
  1 – ON (The RF is externally triggered)

Example:

Input:	$ETS,1,0
Output:	$ETS,1,OK

This turns off external triggering (default state).

7.3 $ETSDG – Get Trigger delay (us)

The external trigger delay setting status. See $ETSDS for description of setting.

Syntax:

Input:	$ETSDG,[channel]
Output:	$ETSDG,[channel],[delay]

- **[channel]** – Channel identification number.
- **[delay]** – trigger delay in us

Example:

Input:	$ETSDG,1
Output:	$ETSDG,30

This indicates that the trigger delay is set to 30 us.

7.4 $ETSDS – Set Trigger delay (us)

Sets the trigger delay. When used in conjunction with the $ETSS command, i.e. when external trigger ADC synchronization is enabled, delays the ADC sampling by a set number of microseconds after the rising edge of TRIG_IN. Due to normal processing and ADC sampling delays, no additional delay is normally required to align the measurement to the pulse when $ETS is enabled.

Syntax:

Input:	$ETSDS,[channel],[delay]
Output:	$ETSDS,[channel]

- **[channel]** – Channel identification number.
- **[delay]** – trigger delay in us

Example:

Input:	$ETSDS,1,0
Output:	$ETSDS,OK

This configures the trigger delay to 0 us (default setting).

7.5 $ETSG – Get External Trigger ADC Synchronization Enable Status (CHECK)

The external trigger ADC synchronization setting status. See $ETSS for description of setting.

Syntax:

Input:	$ETSG,[channel]
Output:	$ETSG,[channel],[enable]

- **[channel]** – Channel identification number.
- **[enable]** – External Trigger ADC Synchronization Enable Status

  0 – OFF
  1 – ON

Example:

Input:	$ETSG,1
Output:	$ETSG,1,0

This indicates that external trigger ADC synchronization is off.

7.6 $ETSS – Set External Trigger ADC Synchronization Enable Status

When external trigger ADC synchronization is enabled on the RFS-2G42G5050X+, the forward and reflected power ADC sampling will be triggered by the rising edge of TRIG_IN. Can be used in conjunction with $ETSDS to synchronize the sampling of an externally modulated RF source.

Syntax:

Input:	$ETSS,[channel],[enable]
Output:	$ETSS,[channel],OK

- **[channel]** – Channel identification number.
- **[enable]** – External Trigger ADC Synchronization Enable Status

  0 – OFF
  1 – ON

Example:

Input:	$ETSS,1,0
Output:	$ETSS,1,OK

This disables external trigger ADC synchronization.

8 Safe Operating Area

The “Safe Operating Area” (SOA) feature defines limits on operating conditions for the RF generator/Amplifier which assure the reliability, longevity, and safe operation of the device. Operating the generator “outside” of the SOA limits could potentially damage the unit or at least reduce its expected lifetime. All Minicircuits generator devices will take self-protection actions that make them robust against accidental misuse.

As of the latest firmware[4], there are 10 SOA types as shown in the below table. Most SOA types have “high” and “shutdown” limits defined. If a parameter exceeds the shutdown limit, RF will be disabled and the appropriate status register bit will be set to 1 (see $ST). The status bits will remain high until the next call to clear the errors (See $ERRC). If a parameter exceeds the warning limit, the appropriate status bit will be set to 1, but RF will remain enabled. For most SOAs, the purpose of the high limit is informational only. A warning can be displayed to notify the user that they are operating close to the limit. In the case of temperature or reflection SOA, exceeding the high limit may result in throttling of the forward power (see $SPG and $STG for detail).

SOA Type	Description	Enabled in RFS-2G42G5050X+	Status Bit High Limit Exceeded	Status Bit Shutdown Limit Exceeded
0	Temperature SOA	Enabled	0x0000000002	0x0000000004
1	Internal Watchdog	Enabled	-	-
2	Reflection SOA	Enabled	0x0000000008	0x0000000010
3	External Watchdog	Disabled	-	0x0000010000
4	Dissipation SOA	Disabled	0x0000080000	0x0000100000
5	PA status	Disabled	-	0x0004000000
6	IQ modulator IQ lock	N/A	0x0008000000	-
7	Current SOA	Enabled	0x0010000000	0x0020000000
8	Voltage SOA (Min)	Enabled	0x0200000000	0x0100000000
	Voltage SOA (Max)	Enabled	0x0400000000	0x0800000000
9	Forward Power SOA	Enabled	0x0040000000	0x0080000000

SOA settings in the RFS-2G42G5050X+ are initialized at boot and should not be modified during operation. If the PA type is changed, the SOA settings will be set to the appropriate default state for the configured PA type.

8.1 $SCG – Get current SOA configuration

This command returns the currents at which SOA takes action. One of the features of the SOA is protection against improper application of DC Current. Current SOA protects against overcurrent conditions.

The SOA has two reactions to excessive current, depending on the severity

- If the current is higher than the normal operating range, but still tolerable: Raise a ‘SOA High Current’ error.
- If the current is dangerously high: Raise a ‘SOA Shutdown Maximum Current’ error and shutdown RF power.

Syntax:

Input:	$SCG,[channel]
Output:	$SVG,[channel],[high current],[shutdown current]

[4] Firmware versions 2.7.0 and later

- **[channel]** – Channel identification number.
- **[high current]** – The current at which the 'SOA High Current' condition is signaled by the SOA. Units in Amps.
- **[shutdown current]** – The current at which the 'SOA Shutdown Current' condition is signaled by the SOA. Units in Amps.

Example:

Input:	$SCG,1
Output:	$SCG,1,5.50,6.00

This indicates that the 'High Current', and 'Shutdown Current' protection limits are configured to 5.5A, and 6A respectively (default).

8.2 $SDG – Get dissipation SOA configuration

One of the features of the SOA is protection against excessive power dissipation inside a generator. Excessive power dissipation occurs when an RF system draws a disproportionate amount of current from its power supply (PSU) relative to the amount RF energy that is transmitted into a load. High dissipations can be reached when the system is poorly matched or when the system is well matched but still operating with poor efficiency. At the system level, dissipation is the rate that heat needs to be removed from the generator by means of heat sink or cooling plate to maintain a stable temperature. The dissipation SOA could be used in systems with limited cooling capacity to issue a warning to the user or shut the generator down before it has a chance to heat up to the temperature shutdown limit.

Dissipation is measured in Watts. The formula used to calculate it is the following:

Dissipation(W) = PSU Power(W) - FWD Power(W) + RFL Power(W)

The SOA has two reactions to excessive dissipation, depending on the severity

- If the dissipation is high, but still tolerable:
  Raise a 'High Dissipation' error.
- If the dissipation is dangerously high:
  Raise a 'Shutdown Dissipation' error and shutdown RF power.

Note: Dissipation SOA is not enabled in the default configuration of the RFS-2G42G5050X+ (see $SOG).

Syntax:

Input:	$SDG,[channel]
Output:	$SDG,[channel],[high dissipation],[shutdown dissipation]

- **[channel]** – Channel identification number.
- **[high dissipation]** – The dissipation value in W at which the 'High Dissipation' reaction is performed by the SOA.
- **[shutdown dissipation]** – The dissipation value in W at which the 'Shutdown Dissipation' reaction is performed by the SOA.

Example:

Input:	$SDG,1
Output:	$SDG,1,47.25000,47.400000

This indicates the 'High Dissipation' and 'Shutdown Dissipation' protection values are both configured to 0W (default). Since this SOA is not enabled by default, these values have no effect on the system operation.

8.3 $SFG – Get Forward Power Limits (W)

One of the features of the SOA is protection against excessive forward power. This command returns the forward power values at which SOA takes action.

The SOA has two reactions to excessive forward power, depending on the severity

- If the forward power is high, but still tolerable:

  Raise a 'High Forward Power' error.

- If the forward power is dangerously high:

  Raise a 'Shutdown Forward Power' error and shutdown RF power.

Syntax:

Input:	$SFG,[channel]
Output:	$SFG,[channel],[high forward power],[shutdown forward power]

- **[channel]** – Channel identification number.
- **[high forward power]** – The forward power value in dBm at which the 'High Forward Power' reaction is performed by the SOA.
- **[shutdown forward power]** – The forward power value in dBm at which the 'Shutdown Forward Power' reaction is performed by the SOA.

Example:

Input:	$SFG,1
Output:	$SFG,1,47.40,48.15

This indicates the 'High Forward power' and 'Shutdown Forward Power' protection values are configured to 47.40 dBm (55W) and 48.15 dBm (65W) respectively (default).

8.4 $SOG – Get SOA configuration

This command returns the enable state of the SOA's protection systems. SOA limits will only trigger a protection response if that SOA is enabled. Otherwise it will do nothing.

Recommended Syntax:

Input:	$SOG,[channel],[soa type]
Output:	$SOG,[channel],[soa type],[enable]

- **[channel]** – Channel identification number.
- **[soa type]** – SOA Type. Used in the recommended syntax. The alternative syntax only supports SOA types 0-7

  0 – Temperature SOA (See $STG)
  1 – Watchdog (this input is ignored, the MCU watchdog is always enabled)
  2 – Reflection SOA (See $SPG)
  3 – External Watchdog (not used in RFS-2G42G5050X+)
  4 – Dissipation SOA (See $SDG)
  5 – PA status (not used in RFS-2G42G5050X+)
  6 – IQ modulator IQ lock (not used in RFS-2G42G5050X+)
  7 – Current SOA (See $SCG)
  8 – Voltage SOA (See $SVG)
  9 – Forward Power SOA (See $SFG)

- **[enable]** – SOA Enable Status

  0 – OFF
  1 – ON

Example:

Input:	$SOG,[channel],[soa type]
Output:	$SOG,[channel],[soa type],[enable]

This indicates the Dissipation SOA is disabled

Alternative Syntax:[5]

Input:	$SOG,[channel]
Output:	$SOG,[channel],[temperature enable],[software watchdog enable], [reflected enable],[external watchdog enable],[dissipation enable],[PA status enable],[IQ lock detect enable],[current enable]

- **[temperature enable]** – The current enable state of the temperature protection system.

  0 – OFF
  1 – ON
- **[software watchdog enable]** – The current enable state of the software timeout protection system.

  This parameter always reports 0 and can be ignored.
- **[reflected enable]** – The current enable state of the reflected power protection system.

  0 – OFF
  1 – ON
- **[external watchdog enable]** – The current enable state of the board status polling protection system.

  0 – OFF
  1 – ON
- **[dissipation enable]** – The current enable state of the dissipation protection system.

  0 – OFF
  1 – ON
- **[PA status enable]** – The current enable state of the PA status monitoring protection system.

  0 – OFF
  1 – ON
- **[IQ lock detect enable]** – The current enable state of the IQ lock detection protection system.

  0 – OFF
  1 – ON
- **[current enable]** – The current enable state of the current protection system.

  0 – OFF
  1 – ON

Example:

Input:	$SOG,1
Output:	$SOG,1,1,0,1,0,0,0,0,1

This indicates the reflected power, temperature, and current protection systems are enabled, but the other protections systems are

[5] Alternative syntax only supports SOA types 0-7. It does not support Voltage SOA or Forward Power SOA.

disabled (default). Voltage and forward power SOA enable statuses are not shown.

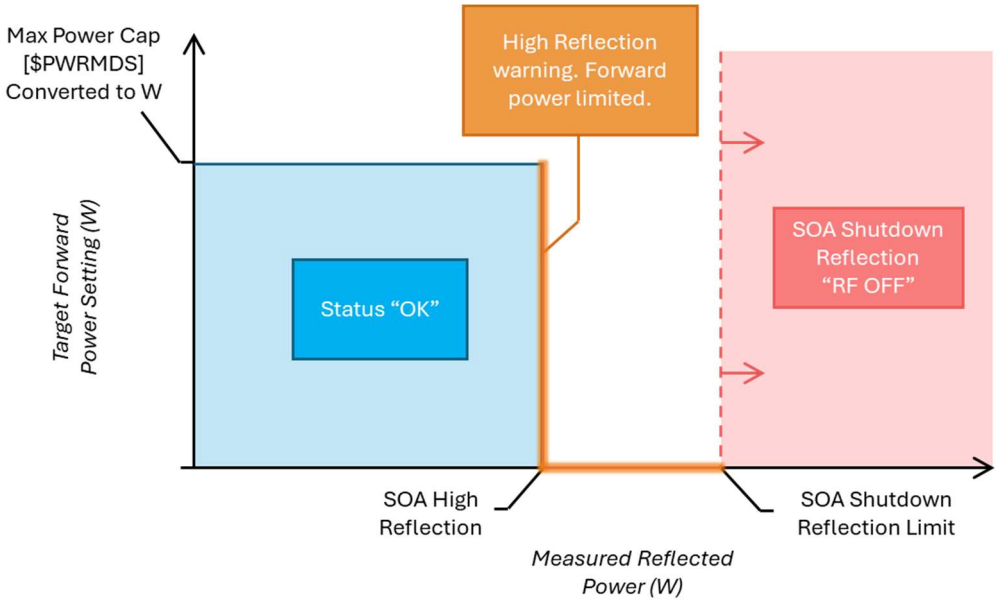
8.5 $SPG – Get reflected power SOA configuration

This command returns the reflected power values at which SOA takes action. One of the features of the SOA is protection against excessive reflected power. Excessive reflection occurs when there is a bad match at the output and RF returns to the generator.

The SOA has two reactions to excessive reflection, depending on the severity

- If the reflection is high, but still tolerable:
  Raise a 'High Reflection' error and throttle the forward power when in autogain mode.
- If the reflection is dangerously high:
  Raise a 'Shutdown Reflection' error and shutdown RF power.

The high reflection condition affects the auto-gain control loop by adding forward power throttling. Normally, the auto-gain algorithm continuously adjusts the output power so that measured forward power is equal to the power setting. When reflected power is greater than the reflected power SOA high limit, autogain will instead reduce/adjust the output power so that the reflected power is equal to the reflected power SOA high limit. If the reflected power reading is coming from a poor VSWR and not an external source, power throttling provides a graceful degradation of performance that will allow continuous operation without damage or interruption into poor VSWR loads. The figure below shows the three operating regions defined by the reflected power SOA.



Syntax:

Input:	$SPG,[channel]
Output:	$SPG,[channel],[high reflection],[shutdown reflection]

- **[channel]** – Channel identification number.
- **[high reflection]** – The reflection value in dBm at which the 'High Reflection' reaction is performed by the SOA.
- **[shutdown reflection]** – The reflection value in dBm at which the 'Shutdown Reflection' reaction is performed by the SOA.

Example:

Input:	$SPG,1

Output:	$SPG,1,47.25000,47.400000

This indicates the 'High Reflection' and 'Shutdown Reflection' protection values are configured to 47.25 dBm (53W) and 54 dBm (55W) respectively (default).

8.6 $STG – Get temperature SOA configuration

This command configures the temperature values at which SOA takes action. One of the features of the SOA is protection against excessive temperatures. Excessive temperatures can occur for any number of reasons: side effects of high RF power reflection, faulty cooling, excessive use, etc. The SOA has two reactions to excessive temperatures, depending on the severity:

- If the temperature is high, but still tolerable:

  Raise a 'High Temperature' error.

- If the temperature is dangerously high:

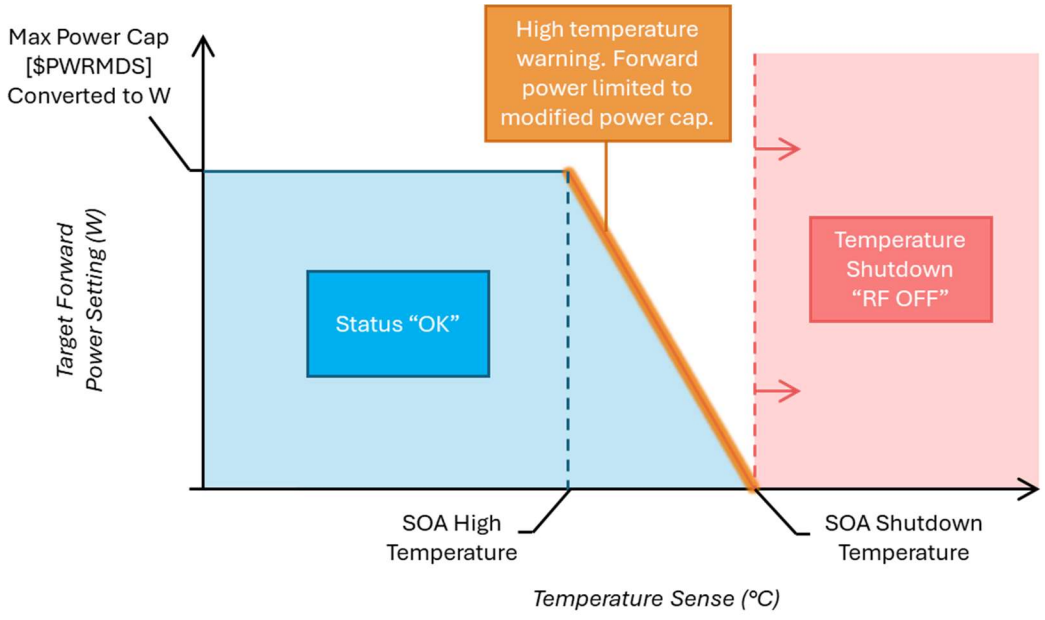
  Raise a 'Shutdown Temperature' error and shutdown RF power.

The high temperature condition affects the auto-gain control loop by adding forward power throttling. Normally, the auto-gain algorithm continuously adjusts the output power so that measured forward power is equal to the power setting. When the PA temperature is greater than the temperature SOA high limit, throttling will limit the forward power if the forward power exceeds the modified max power cap. The modified max power cap is the standard max power cap ($PWRMDG) at temperatures below the high threshold. At temperatures between the high and shutdown thresholds, the modified max power cap is calculated with a linear derating:

$$MPC'(W) = MPC(W) * \frac{T_{shutdown} - T_{final}}{T_{shutdown} - T_{high}}$$

Where:

MPC is the original max power cap in Watts
MPC' is the modified max power cap in Watts
T_{final} is the measured PA temperature in °C
$T_{shutdown}$ T_{high} are the shutdown and high temperature SOA limits in in °C



The above figure provides a visualization of the max power cap derating defined by the Temperature SOA. Power throttling provides a graceful degradation of performance that will allow continuous operation without damage or interruption.

Syntax:

Input:	$STG,[channel]
Output:	$STG,[channel],[high temperature],[shutdown temperature]

- **[channel]** – Channel identification number.
- **[high temperature]** – The temperature value in °C at which the 'SOA High Temperature' condition is signaled by the SOA.
- **[shutdown temperature]** – The temperature value in °C at which the 'SOA Shutdown Temperature' reaction is performed by the SOA.

Example:

Input:	$STG,1
Output:	$STG,1,55.0,65.0

This indicates the 'High Temperature' and 'Shutdown Temperature' protection values are configured to 55°C and 65°C respectively.

8.7 $SVG – Get Voltage Limits (V)

This command returns the voltages at which SOA takes action. One of the features of the SOA is protection against improper application of DC Voltage. Voltage SOA protects against both undervoltage and overvoltage conditions.

The SOA has two reactions to excessive voltage, depending on the severity

- If the voltage is outside of the normal operating range, but still tolerable:

  Raise a 'SOA High Voltage' or a 'SOA Low Voltage' error.

- If the voltage is dangerously low or high:

  Raise a 'SOA Shutdown Minimum Voltage' or 'SOA Shutdown Maximum Voltage' error and shutdown RF power.

Syntax:

Input:	$SVG,[channel]
Output:	$SVG,[channel],[shutdown min voltage],[low voltage],[high voltage],[shutdown max voltage]

- **[channel]** – Channel identification number.
- **[shutdown min voltage]** – The voltage at which the 'Min Voltage Shutdown' condition is signaled by the SOA. Units in Volts.
- **[low voltage]** – The voltage at which the 'Low Voltage' condition is signaled by the SOA. Units in Volts.
- **[high voltage]** – The voltage at which the 'High Voltage' condition is signaled by the SOA. Units in Volts.
- **[shutdown max voltage]** – The voltage at which the 'Max Voltage Shutdown' condition is signaled by the SOA. Units in Volts.

Example:

Input:	$SVG,1
Output:	$SVG,1,24.00,26.00,36.00,38.00

This indicates that the 'Shutdown Min Voltage', ' Low Voltage', 'High Voltage', and 'Shutdown Max Voltage' protection limits are configured to 24V, 26V, 30V, and 32V respectively (default).

9 Error Handling

9.1 $ERRC – Clear error

An error on the RFS board is accompanied by an informative error code ($ST). To ensure the code can be viewed, it stays in memory until manually cleared away. For safety purposes, depending on the exact value, further RF output of the RFS board may be blocked for as long as that non-zero error status remains.

The $ERRC command clears the error state and resets the protective systems that impede the RFS board while an error is present.

The process of status checking and error clearing is the core of error handling. The error status of the RFS board is polled to retrieve current information about any errors that have occurred during operation. This information can be used to decide an appropriate response. After the problem has been resolved, the errors reported by the RFS board should be cleared to signal to the device that the error status is no longer actual.

A situation may occur where an error state appears to persist despite attempts to clear it. In fact, the error is getting cleared properly but is immediately reactivated again because the underlying cause of the error is still present.

Syntax:

Input:	$ERRC,[channel]
Output:	$ERRC,[channel],OK

- **[channel]** – Channel identification number.

Example:

Input:	$ERRC,1
Output:	$ERRC,1,OK

This clears the error state of the RFS board.

If the root cause of the problem has been resolved, the error code will be 0 the next time the error state of the RFS board is polled. If the root cause of the problem remains (e.g. the external shutdown remains triggered),the error state will keep reverting back to a non-zero value until it has been dealt with appropriately.

9.2 $ST – Request status

The $ST command is used to monitor the status of the RFS board.

It returns hexadecimal codes that can be compared against bitmasks to provide an overview of the system status.

Error status:

RFS boards have a safety feature called the ‘Safe Operating Area’ (SOA). If a fault occurs during operation, the SOA raises an error and takes action to protect the system. This is indicated by a red LED on the board. An error on the RFS board is accompanied by an informative error code which can be used to trace the problem.

To ensure the error code can be viewed, it stays in memory until manually cleared away. Error codes that shutdown RF will set the enable state to 0 and will also block the RF power from being turned on again until the error is cleared with $ERRC. Clearing the error will not enable the generator, but it will allow the generator to be re-enabled as long as the error remains cleared. If the user wants to see the current status of the PA, they must clear the errors before requesting status. In some applications, it may be desirable to periodically clear errors so that error statuses displayed always reflect the current state of the system.

The following table provides an overview of the error status bitmask:

Bit #	Hex Status Code	System Status	System Response
N/A	0x000000000	No Errors or Warnings	-
0	0x000000001	Unspecified Error	RF output OFF; Reset controller
1	0x000000002	High PA Temperature	With autogain enabled, unit throttles output power (see SOA section for more detail)
2	0x000000004	Shutdown PA Temperature	RF output OFF
3	0x000000008	High Reflected Power	With autogain enabled, unit throttles output power (see SOA section for more detail)
4	0x000000010	Shutdown Reflected Power	RF output OFF
5	0x000000020	Reset Detected	Warning only; no action
6	0x000000040	Temperature Read-out Error	RF output OFF
7	0x000000080	Power Measurement Failure	RF output OFF
8	0x000000100	RF Enable Failure	Warning only; no action
9	0x000000200	Multiplexer Failure	RF output OFF
10	0x000000400	External Shutdown Triggered	RF output OFF (Non-Blocking)
11	0x000000800	Out of Memory	Warning only; no action
12	0x000001000	I2C Communication Error	RF output OFF in case of critical measurements
13	0x000002000	SPI Communication Error	RF output OFF in case of critical measurements
14	0x000004000	Reserved/Not Applicable	RF output OFF
15	0x000008000	SOA Measurement Error	RF output OFF
16	0x000010000	External Watchdog Timeout	RF output OFF
17	0x000020000	Calibration Missing	RF output OFF
18	0x000040000	External Protection Triggered	Warning only; no action
19	0x000080000	SOA High Dissipation	Warning only; no action
20	0x000100000	SOA Shutdown Dissipation	RF output OFF
21	0x000200000	Calibration EEPROM outdated	RF output OFF
22	0x000400000	Reserved/Not Applicable	RF output OFF
23	0x000800000	Reserved/Not Applicable	RF output OFF
24	0x001000000	Reserved/Not Applicable	RF output OFF
25	0x002000000	Reserved/Not Applicable	RF output OFF
26	0x004000000	Alarm In	RF output OFF
27	0x008000000	Reserved/Not Applicable	Warning only; no action
28	0x010000000	SOA High Current	Warning only; no action
29	0x020000000	SOA Shutdown Current	RF output OFF
30	0x040000000	SOA High Forward Power	Warning only; no action

31	0x080000000	SOA Shutdown Forward Power	RF output OFF
32	0x100000000	SOA Shutdown Minimum Voltage	RF output OFF
33	0x200000000	SOA Low Voltage	Warning only; no action
34	0x400000000	SOA High Voltage	Warning only; no action
35	0x800000000	SOA Shutdown Maximum Voltage	RF output OFF

Note: Unless otherwise stated, all commands that turn RF output OFF are blocking. "Blocking" here means that the RF output is switched off and it remains off until the $ERRC command clears the error. Only then can the RF output be switched on again.

Syntax 1:

Input:	$ST,[channel]
Output:	$ST,[channel],[reserved],[error status code]

- **[channel]** – Channel identification number.
- **[reserved]** – Reserved. Will always return 0.
- **[error status code]** – A hexadecimal value representing various errors which have occurred on the RFS board.

Example 1:

Input:	$ST,1
Output:	$ST,1,0,460

The codes returned by the command represent combinations of several statuses. To decipher a code, it must be converted from hexadecimal to binary and compared against the bitmask to see which bits are high.

The error status code is 0x460. The code is comprised of three separate errors summing up to 0x460:

0x20 = 0b0000001**1**00000 – Reset detected.
0x40 = 0b0000010000**0**0 – Temperature read-out error.
0x400 = 0b0**1**0000000000 – External shutdown triggered.
The error status code will remain in memory until an error clear command has been sent.

Syntax 2:

This command supports an alternative syntax which displays errors in a more user-friendly format.

Input:	$ST,[channel],(output mode)
Output:	$ST,[channel],[error status message]... $ST,[channel],OK

- **[channel]** – Channel identification number.
- **(output mode)** – (optional)

  0 – bitmasked error codes (same output as syntax 1)
  1 – list of legible errors messages (multi-line output)
- **[error status message]** – A hexadecimal value representing various errors which have occurred on theRFS board.

Example 2:

Input:	$ST,1,1
Output:	$ST,1,RESET_DETECTED $ST,1,TEMPERATURE_MEASUREMENT_FAILURE $ST,1,EXTERNAL_SHUTDOWN_DETECTED $ST,1,OK

This time, the optional (output mode) argument is used. The RFS board has encountered the following problems:

Reset detected.
Temperature read-out error.
External shutdown triggered.

The error status code will remain in memory until an error clear command has been sent.

10 System Configuration

System configuration commands define the system-level behavior of the RFS Board.

10.1 $CHANG – Get channel identifier

This command returns the channel number assigned to the RFS board.

Remark: This command does not adhere to established syntax. It does not accept a channel argument at the input.

Syntax:

Input:	$CHANG
Output:	$CHANG,[channel]

- **[channel]** – Channel identifier.

Example:

Input:	$CHANG
Output:	$CHANG,1

This indicates that the channel identifier is 1 (default state).

10.2 $CHANS – Set channel identification number

Every RFS board is assigned a numeric value as a channel identifier for communication. The default value of the identifier is '1', which serves its purpose in single-channel systems. In setups that deploy more than one RFS board, it is often necessary to assign a unique number to each board beforehand, so that they can all be commanded as separate entities. An RFS board will not respond to commands intended for a different channel.

The $CHANS command assigns a channel identification number to an RFS board.

Syntax:

Input:	$CHANS,[channel],[new channel]
Output:	$CHANS,[channel],OK

- **[channel]** – Channel identifier (default = 1)
- **[new channel]** – New desired channel identification number.

Example:

Input:	$CHANS,1,2
Output:	$CHANS,2,OK

This assigns the channel identification number '2' to the RFS board.

From now on, this board can only be addressed by providing '2' as the [channel] argument for commands.

E.g. “$VER,2”

10.3 $COMS – Set Communication Interface

This command sets the communication interface to UART (3.3V TTL) or USB. Only one communication interface can be active at a time. The default communication interface is USB. If the user switches to UART by sending a $COMS,1,1 command, the USB serial port will no longer be active. Communication may only resume over UART during that session. There is no difference in serial command syntax between UART and USB communication interfaces. Rebooting will return the unit back to its default communication interface.

Syntax:

Input:	$COMS,[channel],[interface]
Output:	$COMS,[channel],OK

- **[channel]** – Channel identifier.
- **[interface]** – Serial communication interface.

  1 – UART (TTL 3.3V)
  2 – USB

Example:

Input:	$COMS,1,2
Output:	$COMS,2,OK

This configures the RFS board to communication mode 2, USB serial communication. This is the default mode.

10.4 $PATG – Get PA Type

Get the PA type. Power amplifier (PA) type is a parameter that allows an ISC or RFS board to operate with different system configurations, interfaces, and capabilities. 28 is the only valid PA type for the RFS-2G42G5050X+.

Syntax:

Input:	$PATG,[channel]
Output:	$PATG,[channel],[pa type]

- **[channel]** – Channel identifier.
- **[pa type]** – The PA Type defining the operating mode or system configuration.

  28 – 1 x RFS-2G42G5050X+ (signal source mode)

Example:

Input:	$PATG,1
Output:	$PATG,1,28

This indicates that the PA Type is 28 for the RFS-2G42G5050X+

10.5 $PODG– Get Power Offset in dB

This command sets the power offset of the system. Check the setter ($PODS), for more detail on power offset.

Syntax:

Input:	$PODG,[channel],[offset]
Output:	$PODG,[channel],OK

- **[channel]** – Channel identifier.
- **[offset]** – Power offset in dB.

Example:

Input:	$PODG,1
Output:	$PODG,1,0

This indicates that the offset is 0 dB (default).

10.6 $PODS– Get Power Offset in dB

This command sets the power offset of the system. Power offset is used when there is a fixed attenuation at the output of the generator and the user would like to see power referenced to the plane after that attenuation. For example, an offset setting of 3 would mean that there is 3dB of loss between the generator output and the new reference plane. This affects the behavior of several functions:

- PPG and PPDG normally return the forward and reflected powers. Now forward powers are reduced by [offset] dB and reflected powers are increased by [offset] dB. Note that this means that any calculation of Return loss will be 2 * [offset] dB lower than normal.
- In both auto-gain and feed-forward modes, PWRS and PWRDS are now referencing the power at the new reference plane. The minimum and maximum power settings are adjusted accordingly (reduced by the offset).

Syntax:

Input:	$PODS,[channel],[offset]
Output:	$PODS,[channel],OK

- **[channel]** – Channel identifier.
- **[offset]** – Power offset in dB.

Example:

Input:	$PODS,1,10
Output:	$PODS,1,OK

This sets the offset to 10dB. Now all powers are referenced to the location after the 10dB attenuation. Forward power readings are reduced by 10dB and reflected power readings are increased by 10dB. The new range of power settings is. Setting the power to 5W will result in 50W coming out of the generator.

10.7 $PWRMDG – Get maximum output power cap in dBm

This command gets the maximum output power cap. This is the setting that limits the forward power setpoint ($PWRS / $PWRDS) to be no larger than the configured maximum value.

Syntax:

Input:	$PWRMDG,[channel]
Output:	$PWRMDG,[channel],[power cap]

- **[channel]** – Channel identifier.
- **[power cap]** – The maximum permitted forward power setting in dBm

Example:

Input:	$PWRMDG,1
Output:	$ PWRMDG,1,47.1

This indicates that the maximum output power cap is set to 47.1 dBm (default setting)

10.8 $PWRMDS – Set maximum output power cap in dBm

This command configures a maximum output power cap. This prevents inputting a forward power setpoint ($PWRS / $PWRDS) beyond the configured maximum value. Useful for configuring or ignoring limits in special situations.

Syntax:

Input:	$PWRMDS,[channel],[power cap]
Output:	$PWRMDS,[channel],OK

- **[channel]** – Channel identifier.
- **[power cap]** – The maximum permitted forward power setting in dBm

Example:

Input:	$PWRMDS,1,47.1
Output:	$ PWRMDS,1,OK

This configures the maximum output power cap to 47.1dBm (default setting)

10.9 $PWRMINDG – Get minimum output power setting limit in dBm

This command gets the minimum output power cap. This is the setting that limits the forward power setpoint ($PWRS / $PWRDS) to be no lower than the configured minimum value. This minimum power limit ensures that power setting inputs stay within the valid calibration range of the instruments.

Syntax:

Input:	$PWRMINDG,[channel]
Output:	$PWRMINDG,[channel],[power cap]

- **[channel]** – Channel identifier.
- **[power cap]** – The minimum permitted forward power setting in dBm

Example:

Input:	$PWRMINDG,1
Output:	$ PWRMINDG,1,27.000000

This indicates that the minimum output power limit is set to 27 dBm (default setting)

10.10 $PWRMINDS – Set minimum output power setting limit in dBm

This command configures the minimum output power cap. This is the setting that limits the forward power setpoint ($PWRS / $PWRDS) to be no lower than the configured minimum value. This minimum power limit ensures that power setting inputs stay within the valid calibration range of the instruments. This is especially important when operating in feedforward mode where the internal attenuation settings are only well-defined for powers within the operating range

Syntax:

Input:	$PWRMINDS,[channel],[power cap]
Output:	$PWRMINDS,[channel],OK

- **[channel]** – Channel identifier.
- **[power cap]** – The minimum permitted forward power setting in dBm

Example:

Input:	$PWRMINDG,1,27
Output:	$PWRMINDG,1,OK

This configures the minimum output power limit to 27 dBm (default setting)

10.11 $RST – Execute system reset

This command executes a reset of the RFS board. All board settings will return to their default states.

Following a reset, whether intentional or as the result of a fault, the ‘reset detected’ error flag (0x20) will beraised.

Syntax:

Input:	$RST,[channel]
Output:	$RST,[channel],OK

- **[channel]** – Channel identifier.

Example:

Input:	$RST,1
Output:	$RST,1,OK

After sending the $RST command, the board returns an OK and then promptly resets.The ‘reset detected’ error flag (0x20) is raised.

10.12 $UARTS – Set UART baud rate

This command sets the baud rate used for communication through UART. Any value can be entered, but unsurprisingly, ongoing communication will break the moment this value is changed.

Changing the baud rate affects communication speed. Lowering it can cause noticeable communication delay, while increasing it can speed up communication and leave a larger CPU time-slice for other tasks. However, setting the baud rate too high may cause communication issues to arise, as the UART transceivers have limitations.

Remark: This setting does not affect communication through USB, only through UART.

Syntax:

Input:	$UARTS,[channel],[baud rate]
Output:	<Command Does Not Reply>

- **[channel]** – Channel identifier.
- **[baud rate]** – Baud rate in symbols per second. For UART to work, the baud rate on the Tx and the Rx side must be configured to the same value.

Example:

Input:	$UARTS,1,115200
Output:	

This sets the baud rate of the UART to 115200 baud. After changing the baud rate, the communication line needs to be reinitialized on the user side with the updated baud value as well. There is no response to this command that indicates successful configuration of the setting.

11 Revision History

Revision	Date	FW Version	Description
OR	7/12/24	2.8.12	Release to Web
X3	7/11/24	2.8.12	Preliminary Revision