riscure

Spider

Communication and fault injection
of embedded chips

datasheet
V 1.0

rev 1

Contents

The product

Context

Market

The market for high security embedded chips has seen a huge growth in the past 5 years particularly in content protection and mobile devices. Compared to the fairly standardized smart card world, the large variety of embedded chips causes a big challenge in side channel and fault injection testing. You need a tool to handle this without adding complexity or hard-to-debug setups. Adding flexibility in triggering, control and on-the-fly adjustment of fault injection campaigns greatly increases the tester's surface. Moreover, combining protocols like JTAG, SPI, I2C or CAN with fault injection opens up new avenues of testing.

Approach

We developed Spider, a highly versatile FPGA-based tool to:

- Reduce setup complexity for embedded device testing
- Generate faults for power amplifier, laser or EM-FI equipment
- On-the-fly adjustments of fault injection parameters
- Communicate at low level with embedded chips
- Easy to develop proprietary interface and protocol extensions

Other methods	Limitations
VC Glitcher	- Supports only smart card protocol - Limited protocol flexibility
Custom FPGA design	- Steep learning curve - R&D investment for a good digital and analog combination
Microcontrollers	- Cannot provide rigid triggering - Cannot provide true parallelism

The challenge it solves

- ## Easy interfacing with embedded chips

  Setting up a side channel test environment can be a time-consuming exercise. Embedded targets have a great variety in communication protocols, multiple power domains, and different I/O voltage levels.

  As a tester you want control over the target's interfaces so that you can run exactly the desired tests. With Spider you can sniff and communicate with an embedded chip making use of common chip-to-chip protocols.

- ## Accurate triggering

  Based on what is observed, for example on the chip's data bus, you want to start a power measurement or inject a fault. Spider can generates very accurate triggers to accomplish this. Spyder offers great flexibility and can be used with a wide range of embedded chips because it supports JTAG, I2C and SPI.

- ## Custom fault attack flow

  In order to create effective security tests that will leave no vulnerabilities undetected it must be possible to change the attack flow. Custom attack flows can be easily created using Inspector software or the Spider's SDK library. Multiple options are available for the user who can for example influence the voltage level, change the event ordering and adjust glitch timing. With these possibilities the shape of a glitch can be interactively customized without any user intervention..

Unique features

1. **Unique all-in-one tool to control embedded targets**
    - Wide range of I/O voltage levels (1.0 – 3.3 V)
    - Support for popular protocols
    - Flexible trigger generation
    - Control two lasers for simultaneous multi pulse attacks

2. **Versatile fault generator**
    - Drives faults to glitch amplifiers, lasers, EMFI probes
    - Program any attack flow
    - Arbitrary wave form generation for faults

3. **Flexible and easy to use**
    - SDK in Python, Java and C
    - Plug and play from Inspector

4. **Extensive protocol support**
    - SPI
    - JTAG
    - I2C
    - UART

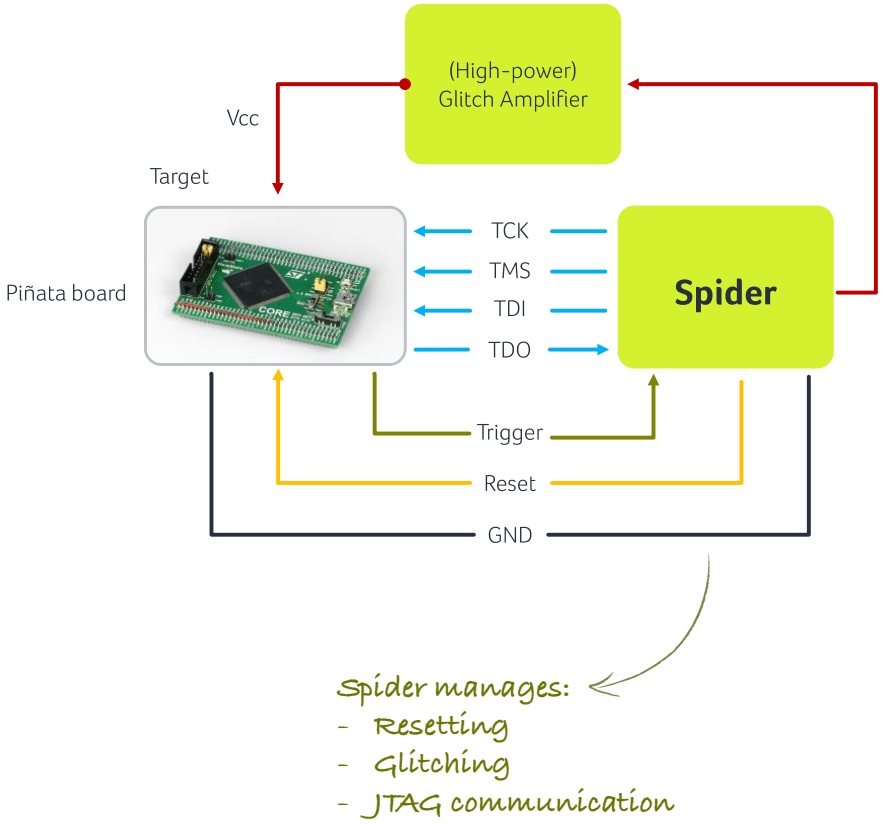
Example use case

Use case – JTAG unlocking

The test scenario

1. Due to security considerations, it is common practice to lock the JTAG interface.
2. Spider can challenge the strength of JTAG locking by controlling the reset line of the target.
3. Glitch during target booting:
    - Apply normal VCC to target for booting
    - Lower VCC to minimum level just before attack
    - Generate glitch via Glitch Amplifier
4. Perform a standard device id read out via JTAG communication.

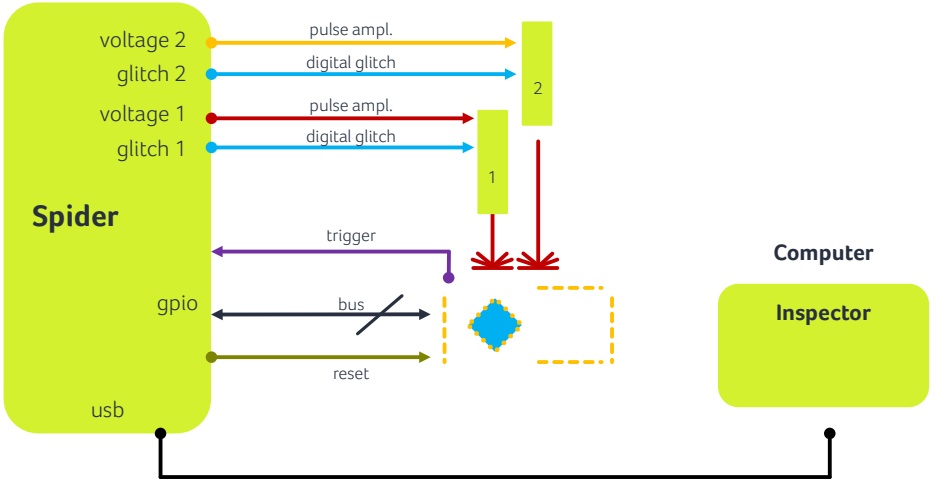
A locked JTAG interface has been proven to become unlocked by injecting faults to the chip



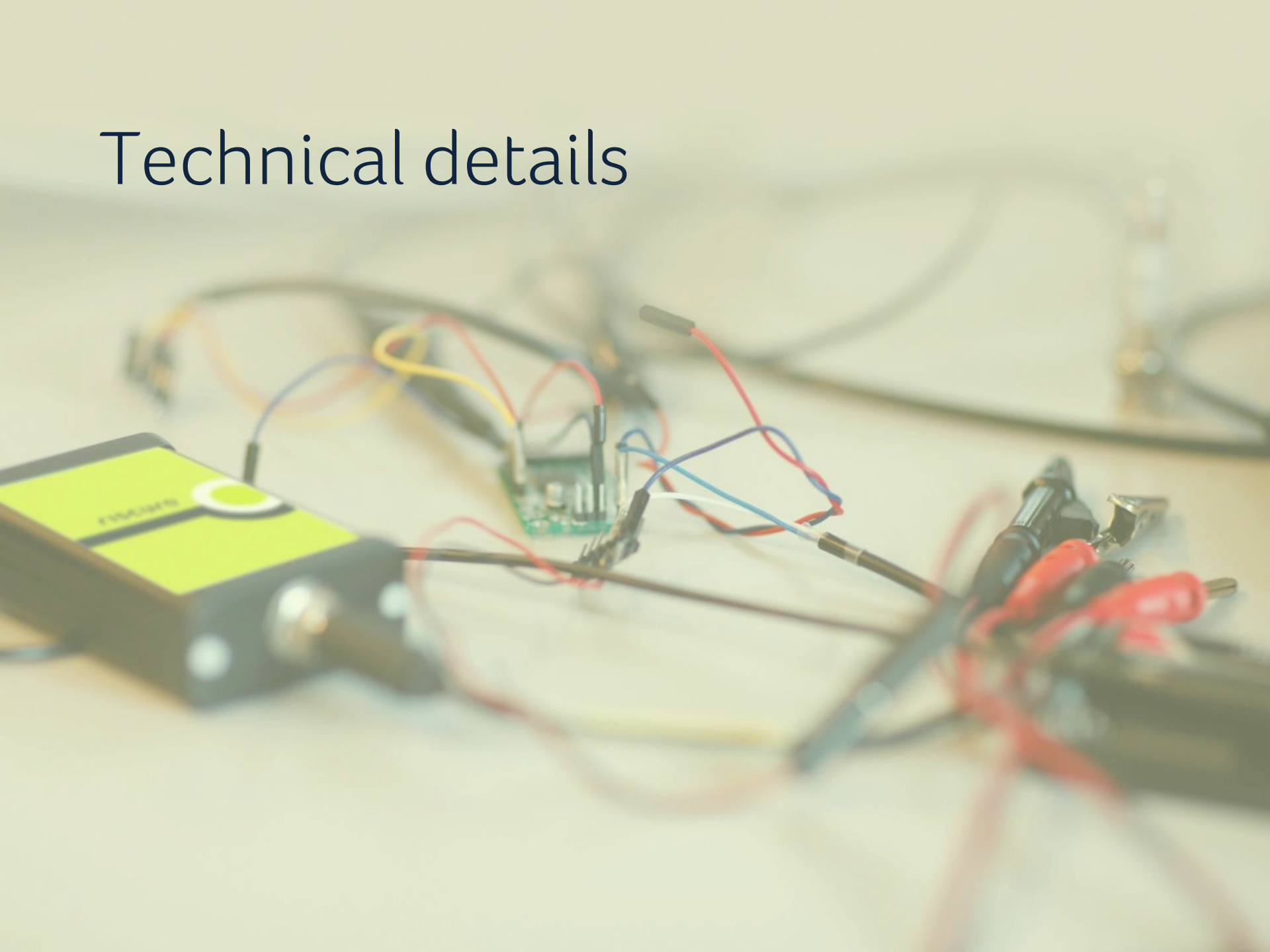
Use case – drive LS2 Twin Scan

The test scenario

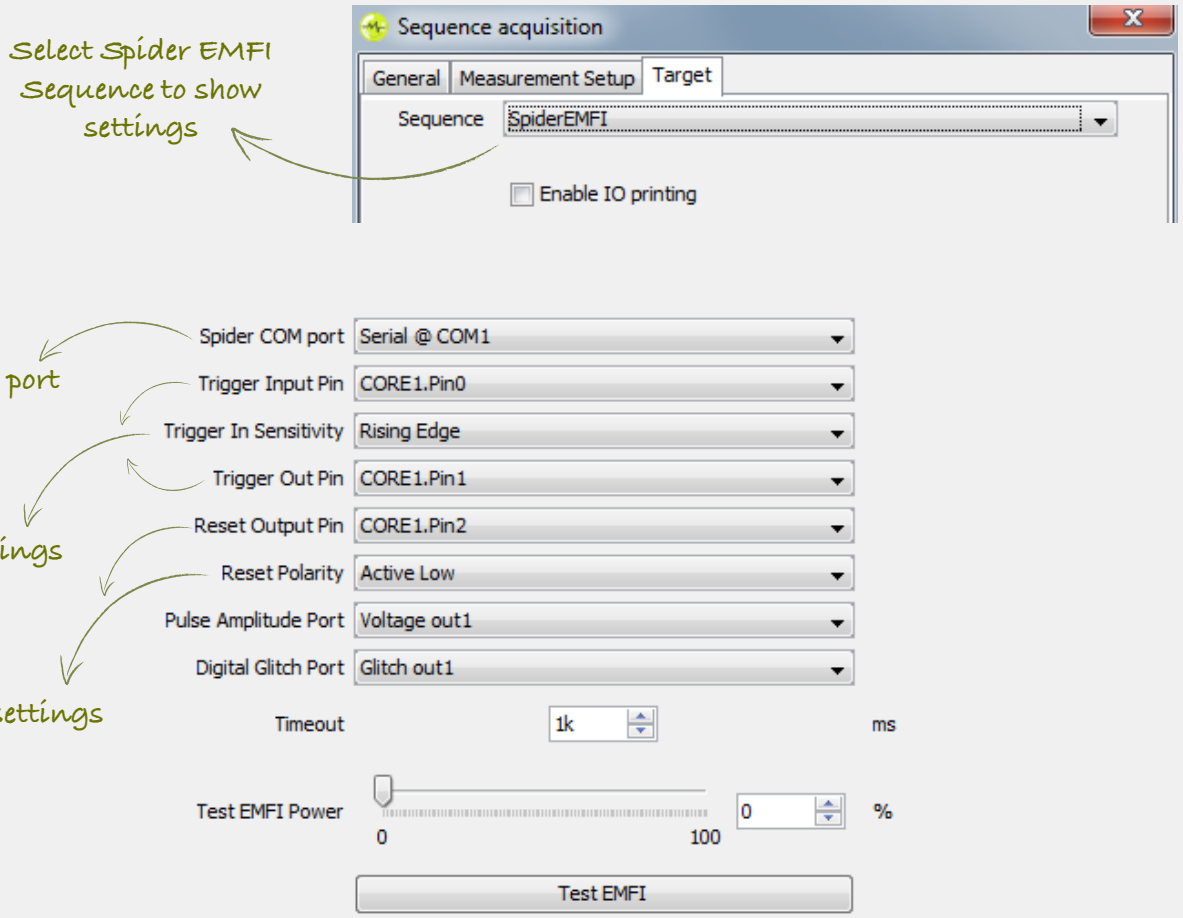
1. Spider can drive the 2 lasers from the Twin Scan independently.
2. This allows for example attacking a crypto-core and a memory storage at the same time to get results that would otherwise be impossible.



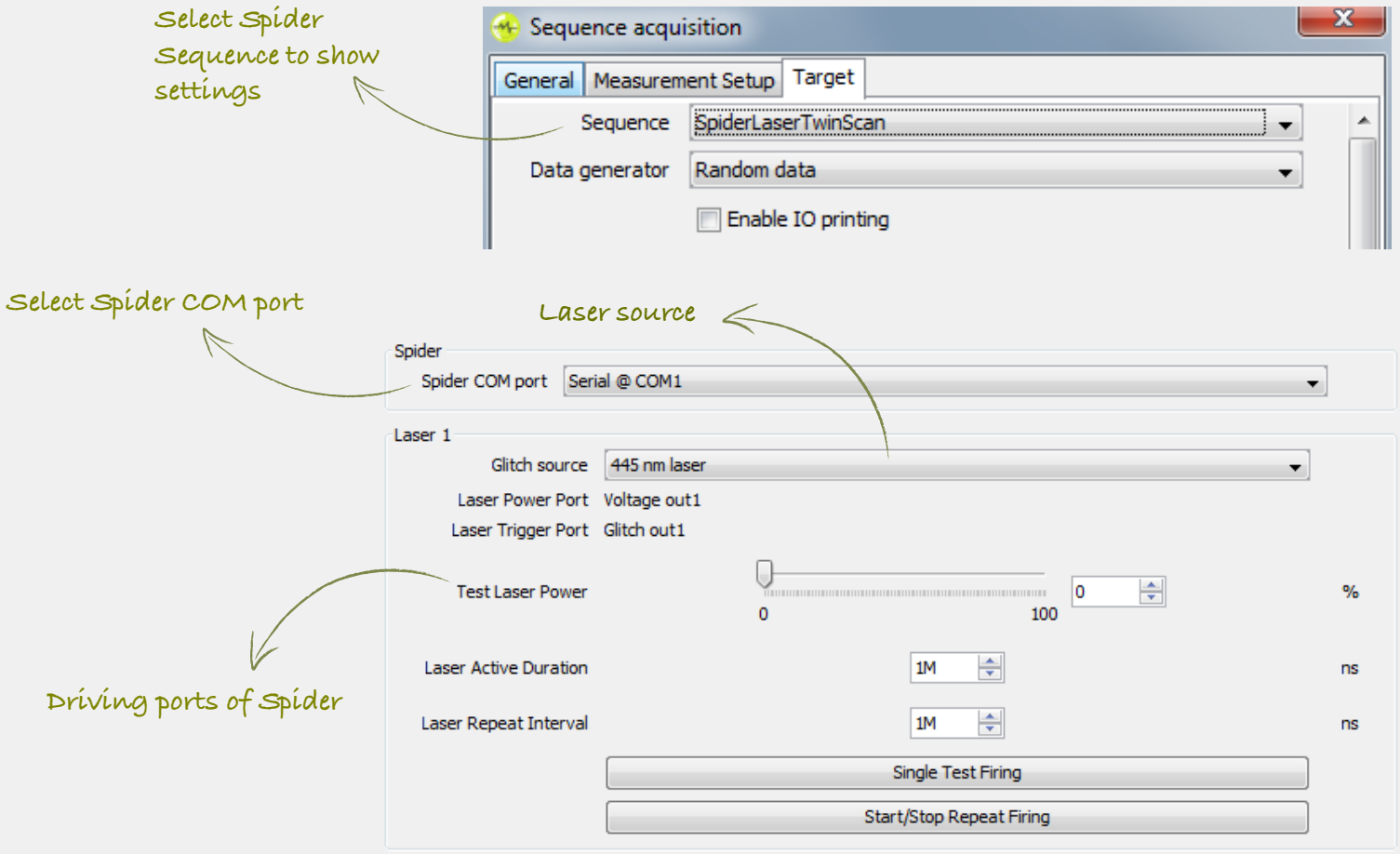
Technical details



User control – EMFI



User control – Twin Scan



Programming – Inspector

```
spiderCore1 = new Spider(Spider.CORE1, connections.get(rawIOSpider));
spiderCore1.resetSettings();
glitcher = new Chronology(spiderCore1);
```

Create a 'glitcher' using Spider Core 1

```
// Add new events
glitcher.setVcc(selectedGlitchPort, normalVcc);
glitcher.setGPIO(triggerOutIndex, 0);
glitcher.waitTrigger(triggerInIndex, selectedTriggerSensitivity, 1);
glitcher.setGPIO(triggerOutIndex, 1);
glitcher.glitch(selectedGlitchPort, glitchVcc, glitchDelay, glitchDuration);
```

Add events and customize their order

Programming – Python

Assign and open Spider COM port

```
com_port = serial.Serial();
com_port.port = "COM1";
com_port.open();
```

Create a 'glitcher' using Spider Core 1

```
spiderCore1 = Spider(Spider.CORE1, com_port);
spiderCore1.resetSettings();
glitcher = Chronology(spiderCore1)
```

Add events and customize their order

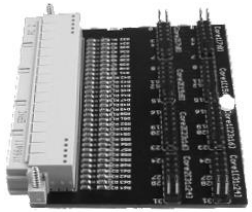
```
glitcher.waitTrigger(0, Spider.RISING_EDGE, 1)
glitcher.setVcc(Spider.GLITCH_OUT1, 3.3)
glitcher.waitTime(100e-6)
glitcher.glitch(Spider.GLITCH_OUT1, -4.0, 0, 100e-9)
glitcher.glitch(Spider.GLITCH_OUT1, -1.0, 100e-6, 500e-9)
glitcher.waitTime(0.5)
glitcher.setVcc(Spider.GLITCH_OUT1, 0)
```

Technical specifications

Parameter	Min.	Typical	Max.	Unit
gpio voltage level (V_{Logic})	1.0	-	3.3	V
gpio V_{OH}	V_{Logic}-0.45	-	-	V
gpio V_{OL}	-	-	0.4	V
gpio V_{IH}	0.65V_{Logic}	-	V_{Logic}+0.3	V
gpio V_{IL}	-0.3	-	0.35V_{Logic}	V
voltage output	0.0	-	5.0	V
voltage output current	-	-	100	mA
glitch outputs Voltage	-4.0	-	4.0	V
glitch outputs current	-	-	72	mA
glitch output timing resolution	-	4	-	nS
uart signal voltage level	-	3.3	-	V
uart baud rate	1907	-	1.5M	baud

Package

	Description	
1	Spider	
1	15V DC Power Supply Unit, input 100 - 240 V, AC 50 - 60 Hz Included: power cable with country specific jack	 
1	Communication cable: USB-A - USB-B, 2 m	
4	Signal cable: SMB – SMB	
10	Jumper wires: male - female	

	Description	
10	Jumper wires: female - female	
4	Output impedance adapter - SMB, 50 Ohm	
1	Breakout Board	
1	Spider SDK USB stick	
1	Quick Start Guide	

Please contact Riscure for more information.

You can reach us by email : inforequest@riscure.com,

by phone : +31 15 251 4090 US: +1 650 646 9979

Or on the web: www.riscure.com

riscure