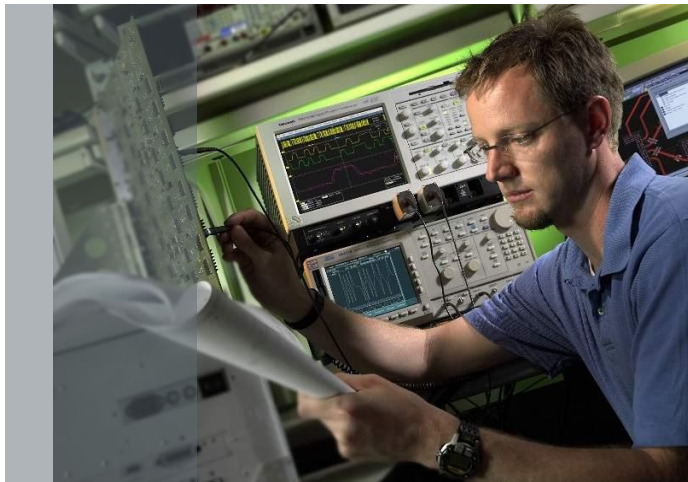
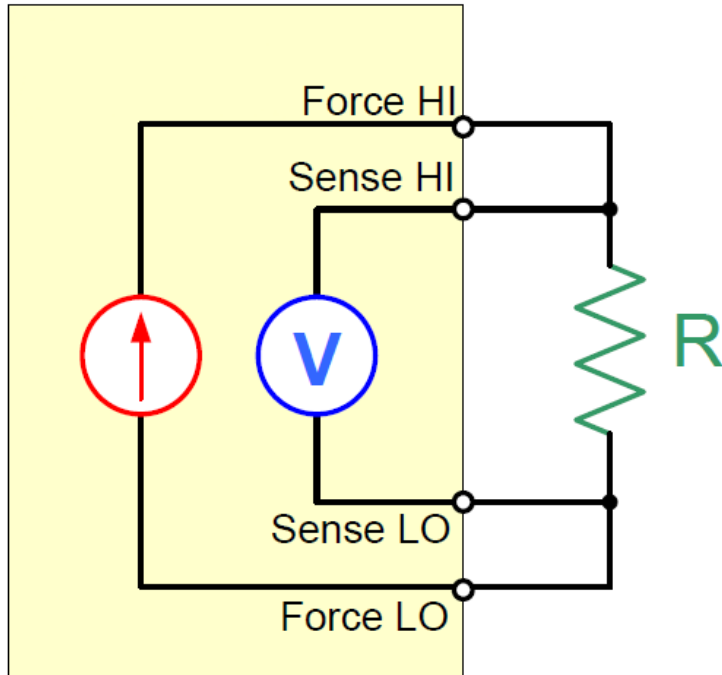


Using the Model 2461 SourceMeter

Pulse Sweep Trigger Out Example



Circuit Diagram for Pulse Sweep Trigger Out



Model 2461 SourceMeter set up for sourcing current and digitizing voltage in a four-wire configuration

This test generates 101, 10ms current pulses that sweep in value from 1V to 5V. A measurement of the current is made at the end of each pulse.

The true off time between pulses is comprised of the delay (the time spent at bias before each pulse) and the off time (the time spent at bias after each pulse). The delay is set to 45ms and the off time is set to 45ms for a total time of 90ms between pulses.

Using the 2461_PulseSweep_TriggerOut.tsp file

Based on your test requirements, you can change the test parameters that are programmed in the code. The pulse sweep command accepts many arguments so the code has variables assigned to each one. Here is a list of some of the default test parameters.

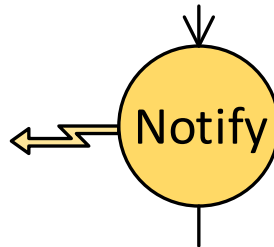
Test Parameter	Command or Variable	Default
Start of sweep	start	1
Stop of sweep	stop	5
Points in sweep	points	101
Pulse width	pulse_width	10e-3
Delay before each pulse	delay	45e-3
Off time after each pulse	offTime	45e-3
Current Limit During Pulse	xPulseLimit	100e-3

Digital I/O Triggering

The code is set up to assert a negative logic pulse on the digital I/O connector's pin 1 when the pulse sweep is completed. This trigger can be used to alert other instruments in a test system that the pulse sweep is completed.

This trigger output is accomplished by putting a *notify* block in the instrument's trigger model. The notify block is shown below.

The trigger model's notify block could easily be moved to somewhere else in the trigger model by building the trigger model differently. Included are example code section to show how to add the trigger before each pulse, after each pulse, before the whole pulse sweep starts and between the off time and delay time of the pulses.



Before each pulse

```
trigger.model.load("Empty")
trigger.model.setblock(1, trigger.BLOCK_BUFFER_CLEAR, bufferName)
trigger.model.setblock(2, trigger.BLOCK_CONFIG_RECALL, "MeasmyPulseSweep",
1, "myPulseSweep", 1)
trigger.model.setblock(3, trigger.BLOCK_SOURCE_OUTPUT, smu.ON)
trigger.model.setblock(4, trigger.BLOCK_BRANCH_ALWAYS, 6)
trigger.model.setblock(5, trigger.BLOCK_CONFIG_NEXT, "myPulseSweep")
trigger.model.setblock(6, trigger.BLOCK_DELAY_CONSTANT, delay)
trigger.model.setblock(7, trigger.BLOCK_NOTIFY, trigger.EVENT_NOTIFY1)
trigger.model.setblock(8, trigger.BLOCK_SOURCE_PULSE_OUTPUT, smu.ON)
trigger.model.setblock(9, trigger.BLOCK_DELAY_CONSTANT,
pulse_delay_calc(pulseWidth))
trigger.model.setblock(10, trigger.BLOCK_MEASURE, bufferName)
trigger.model.setblock(11, trigger.BLOCK_SOURCE_PULSE_OUTPUT, smu.OFF)
trigger.model.setblock(12, trigger.BLOCK_DELAY_CONSTANT, offTime)
trigger.model.setblock(13, trigger.BLOCK_BRANCH_COUNTER, points, 5)
trigger.model.setblock(14, trigger.BLOCK_SOURCE_OUTPUT, smu.OFF)
```

After each pulse

```
trigger.model.load("Empty")
trigger.model.setblock(1, trigger.BLOCK_BUFFER_CLEAR, bufferName)
trigger.model.setblock(2, trigger.BLOCK_CONFIG_RECALL, "MeasmyPulseSweep",
1, "myPulseSweep", 1)
trigger.model.setblock(3, trigger.BLOCK_SOURCE_OUTPUT, smu.ON)
trigger.model.setblock(4, trigger.BLOCK_BRANCH_ALWAYS, 6)
trigger.model.setblock(5, trigger.BLOCK_CONFIG_NEXT, "myPulseSweep")
trigger.model.setblock(6, trigger.BLOCK_DELAY_CONSTANT, delay)
trigger.model.setblock(7, trigger.BLOCK_SOURCE_PULSE_OUTPUT, smu.ON)
trigger.model.setblock(8, trigger.BLOCK_DELAY_CONSTANT,
pulse_delay_calc(pulseWidth))
trigger.model.setblock(9, trigger.BLOCK_MEASURE, bufferName)
trigger.model.setblock(10, trigger.BLOCK_SOURCE_PULSE_OUTPUT, smu.OFF)
trigger.model.setblock(11, trigger.BLOCK_NOTIFY, trigger.EVENT_NOTIFY1)
trigger.model.setblock(12, trigger.BLOCK_DELAY_CONSTANT, offTime)
trigger.model.setblock(13, trigger.BLOCK_BRANCH_COUNTER, points, 5)
trigger.model.setblock(14, trigger.BLOCK_SOURCE_OUTPUT, smu.OFF)
```

Before the whole pulse sweep

```
trigger.model.load("Empty")
trigger.model.setblock(1, trigger.BLOCK_BUFFER_CLEAR, bufferName)
trigger.model.setblock(2, trigger.BLOCK_CONFIG_RECALL, "MeasmyPulseSweep",
1, "myPulseSweep", 1)
trigger.model.setblock(3, trigger.BLOCK_SOURCE_OUTPUT, smu.ON)
trigger.model.setblock(4, trigger.BLOCK_NOTIFY, trigger.EVENT_NOTIFY1)
trigger.model.setblock(5, trigger.BLOCK_BRANCH_ALWAYS, 7)
trigger.model.setblock(6, trigger.BLOCK_CONFIG_NEXT, "myPulseSweep")
trigger.model.setblock(7, trigger.BLOCK_DELAY_CONSTANT, delay)
trigger.model.setblock(8, trigger.BLOCK_SOURCE_PULSE_OUTPUT, smu.ON)
trigger.model.setblock(9, trigger.BLOCK_DELAY_CONSTANT,
pulse_delay_calc(pulseWidth))
trigger.model.setblock(10, trigger.BLOCK_MEASURE, bufferName)
trigger.model.setblock(11, trigger.BLOCK_SOURCE_PULSE_OUTPUT, smu.OFF)
trigger.model.setblock(12, trigger.BLOCK_DELAY_CONSTANT, offTime)
trigger.model.setblock(13, trigger.BLOCK_BRANCH_COUNTER, points, 6)
trigger.model.setblock(14, trigger.BLOCK_SOURCE_OUTPUT, smu.OFF)
```

Between the off time and delay time

```
trigger.model.load("Empty")
trigger.model.setblock(1, trigger.BLOCK_BUFFER_CLEAR, bufferName)
trigger.model.setblock(2, trigger.BLOCK_CONFIG_RECALL, "MeasmyPulseSweep",
1, "myPulseSweep", 1)
trigger.model.setblock(3, trigger.BLOCK_SOURCE_OUTPUT, smu.ON)
trigger.model.setblock(4, trigger.BLOCK_BRANCH_ALWAYS, 6)
trigger.model.setblock(5, trigger.BLOCK_CONFIG_NEXT, "myPulseSweep")
trigger.model.setblock(6, trigger.BLOCK_NOTIFY, trigger.EVENT_NOTIFY1)
trigger.model.setblock(7, trigger.BLOCK_DELAY_CONSTANT, delay)
trigger.model.setblock(8, trigger.BLOCK_SOURCE_PULSE_OUTPUT, smu.ON)
trigger.model.setblock(9, trigger.BLOCK_DELAY_CONSTANT,
pulse_delay_calc(pulseWidth))
trigger.model.setblock(10, trigger.BLOCK_MEASURE, bufferName)
trigger.model.setblock(11, trigger.BLOCK_SOURCE_PULSE_OUTPUT, smu.OFF)
trigger.model.setblock(12, trigger.BLOCK_DELAY_CONSTANT, offTime)
trigger.model.setblock(13, trigger.BLOCK_BRANCH_COUNTER, points, 6)
trigger.model.setblock(14, trigger.BLOCK_SOURCE_OUTPUT, smu.OFF)
```

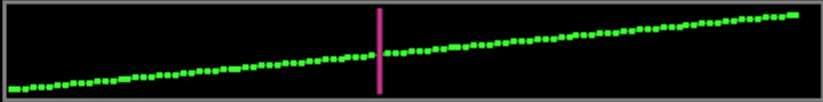

The `pulse_delay_calc()` Function

The `pulse_delay_calc()` function is included in the code to automatically calculate the length of the constant delay block that sets the pulse width.

This function compensates for instrument settings that would impact measurement time such as NPLC, auto zero, source readback and filtering.

The Model 2461 always does this calculation to determine how long it should wait before taking a measurement during a pulse, but the value must be computed manually in this example because the trigger model is being rewritten.

Results

READING TABLE			
Buffer	Active (defbuffer1)		
Buffer Index	Time	Reading	Source
48	11/18 15:02:23.695397	+025.808 mA	+2.879280 V
49	11/18 15:02:23.795785	+026.167 mA	+2.919308 V
50	11/18 15:02:23.896185	+026.525 mA	+2.959335 V
51	11/18 15:02:23.996558	+026.884 mA	+2.999504 V
52	11/18 15:02:24.096936	+027.241 mA	+3.039532 V
53	11/18 15:02:24.197307	+027.600 mA	+3.079697 V
54	11/18 15:02:24.297667	+027.959 mA	+3.119587 V
55	11/18 15:02:24.398060	+028.317 mA	+3.159340 V
56	11/18 15:02:24.498455	+028.676 mA	+3.199368 V
57	11/18 15:02:24.598825	+029.035 mA	+3.239399 V