Programmable Digital Pulse Generator

Firmware Version 1-00

User Manual

Document version 1, created on Dec-31-2019

CGC Instruments
Hübschmannstr. 18 | D–09112 Chemnitz
Tel.: +49 (371) 355 098–55
Fax: +49 (371) 355 098–60
internet: www.cgc-instruments.com
e–mail: info@cgc-instruments.com

Contents

Figure List

Table List

Technical Data

Characteristics

- 24 digital pulse generators, 1 digital oscillator
  resolution: 10 ns, length 32 bit (delay up to 42.9 s)
- arbitrary configuration by selectable
  32 trigger and 32 output sources
- control of 6 signal switches and 5 digital I/O modules
- 6 monitoring outputs
  connectors: LEMO
  signal level: TTL, log. 0: 0..0.4 V, log. 1: 2.4..5.0 V
  output impedance: 50 Ω
- non-volatile memory (NVM) data space: 256 KB
- maximum number of stored configurations: 500

Digital Interface

- USB interface according to USB 2.0 standard
  connector: USB plug type B
  data transfer rate: up to 12 MBit/s (*Full Speed*)
  effective data transfer rate: >100 kBit/s

Description

The pulse controller produces 32 digital signals that can be used to control up to 6 signal switches, 5 digital I/O modules, or 6 monitor outputs at the controller's front panel. The first channel of each digital I/O module can be used as a trigger source.

The device integrates one digital oscillator and 24 digital pulse generators (see Fig. 1). These modules are clocked by 100 MHz, thus provide a time resolution of 10 ns. The maximum pulse delay, pulse width or oscillation period are 2^{32} clock pulses, i.e. about 42.9 s.

The digital oscillator (module Oscillator in Fig. 1) is a free running multivibrator with a period defined by a 32-bit long integer number. The oscillator can be stopped or started at any time by the control signal OscillatorEnable. If enabled, it provides at its output a 1-clock (10 ns) wide positive pulse at the end of the programmed period. The period in clock pulses is given by the 32-bit number Period plus 2. The minimum value of Period is 1, resulting in a period of 3 clock pulses (30 ns).

The pulse generators (modules Pulser00-23 in Fig. 1) are digital monoflops. They are triggered by a rising slope at the trigger input and produce a positive pulse with a specified width (32-bit integer numbers Width00-23) after a specified delay (32-bit integer numbers Delay00-23). The polarity of the trigger signal can be inverted by the control signal Inv00-23. The trigger source is selected by a 32-channel multiplexor (control by a 5-bit integer number Select00-23, see Tab. 1). As trigger input, any output of the pulse generators (signals Puls00-23), output of the oscillator (signal Osc), or the external trigger sources, i.e. the first channels of the I/O modules (signals Trig0-4) can be selected. The trigger input can also be set to 0 or to software trigger (signal SwTrig). If the level 0 is selected, the particular pulse generator is stopped by the signal Clr, by inverting the trigger level, i.e. by setting the corresponding signal InvNN to 1, the pulse generator is triggered. Note that setting the signal InvNN to 0 again, the pulse generator is stopped immediately.

The pulse generator can also be stopped if the signal PulserEnable is 0 or if the control values DelayNN or WidthNN are set to 0. Note that a combination DelayNN = 0 and WidthNN ≠ 0 leads to an activated output, i.e. PulsNN = 1. In all other stopped states, the output is reset, i.e. PulsNN = 0.

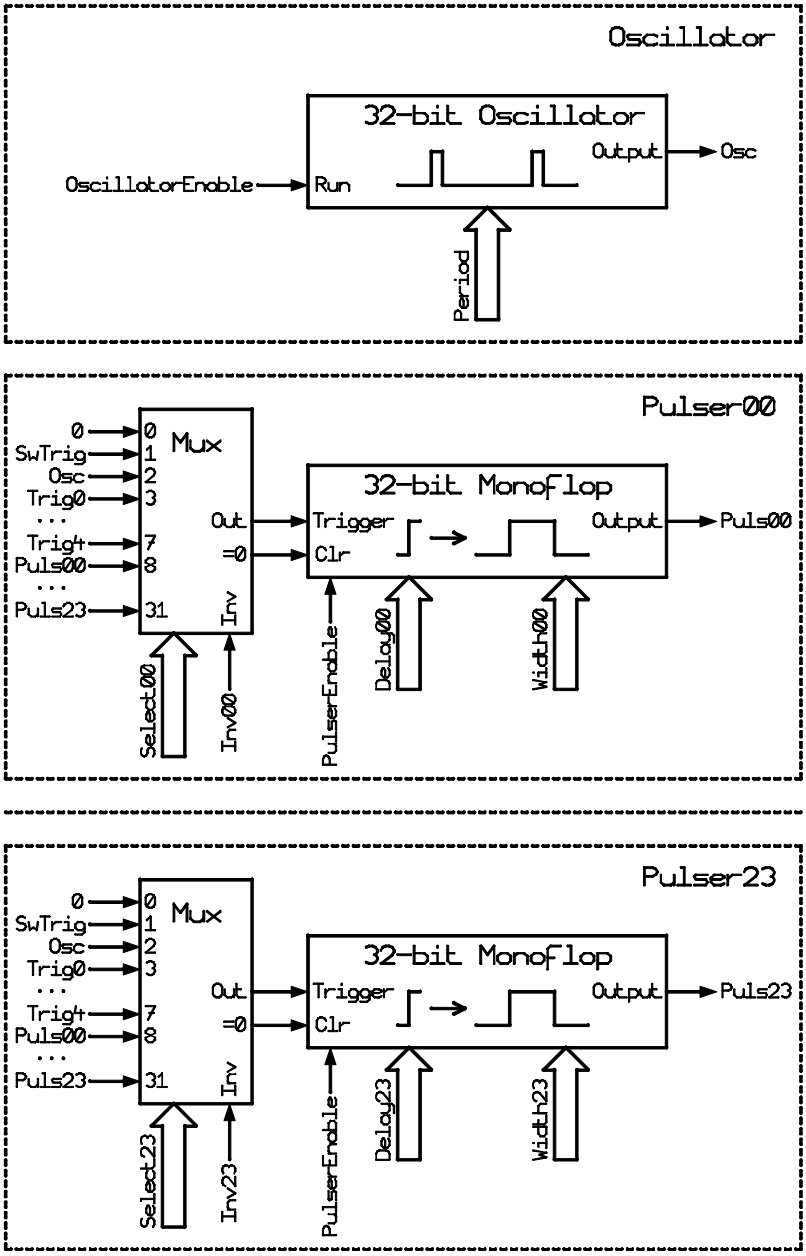

Fig. 1. Pulser Block Diagram.

Tab. 1. Assignment of trigger inputs and signal outputs.

SelectNN	Value	Explanation
0	0	Logic 0
1	SwTrig	Software trigger
2	Osc	Oscillator output
3	Trig0	Input 1 of the digital I/O module #1
...	...	...
7	Trig4	Input 1 of the digital I/O module #5
8	Puls00	Output of the pulse generator #00
...	...	...
31	Puls23	Output of the pulse generator #23

An alternative way of software triggering is to select the input SwTrig. The signal SwTrig is generated in the software-trigger engine (see Fig. 2). The both input signals SoftwareTrigger and SoftwarePulse can be controlled by the software, the latter one triggers a monoflop on its 0-to-1 transition. The monoflop produces a 1-clock (10 ns) wide positive pulse that is XORed with the signal SoftwareTrigger, thus inverting the output signal SwTrig for one clock period.

The symmetrical architecture of the pulse generators offers a large variability of configurations. The pulse generators can be chained to produce complex pulse sequences. They can be triggered periodically by the internal oscillator, by an external event, or by the software. Unused channels can be disabled.

Each pulse generator is a combination of two coupled non-retriggerables monoflops. The first one with a pulse width defined by the numbers Delay00-23 is triggered by the input signal Trigger. When its de-

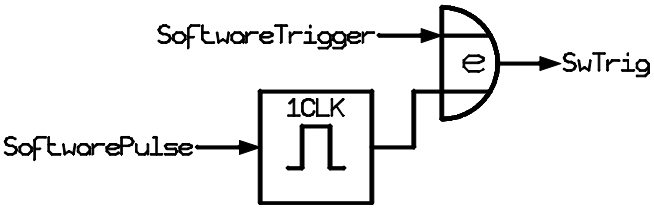

Fig. 2. Software-trigger engine.

lay finishes, the second monoflop with a pulse width defined by the numbers Width00-23 is launched and the output Puls00-23 of the particular module is activated. The pulse delay in clock pulses is given by the 32-bit number DelayNN plus 3. The minimum value of DelayNN is 1, resulting in a delay of 4 clock pulses (40 ns). Similarly, the pulse width in clock pulses is given by the 32-bit number WidthNN plus 2. The minimum value of WidthNN is 1, resulting in a delay of 3 clock pulses (30 ns). Note that setting DelayNN or WidthNN to 0, the pulse generator is stopped.

The pulse controller controls 32 outputs (see Fig. 3). These output signals drive 6 signal switches, 5 I/O modules with 4 outputs each, and 6 monitor outputs at the controller front panel (see Tab. 2). Each output can be configured in a similar way like the trigger inputs of the pulse generators (see Tab. 1). The output may be permanently set to 0 or 1, connected to the software-trigger signal SwTrig, to the output of the oscillator (signal Osc), to the external trigger source (signals Trig0-4) or to the pulse generator (signals Puls00-23). The polarity of the output signal can be inverted by the control signal Inv00-31.

The configuration of the pulse controller is controlled by software. The current configuration is stored in a non-volatile memory and is automatically restored when the device is started. The user can define and save up to 500 additional configurations in the non-volatile memory, they can be easily applied by a software command. Beside its number, each configuration can be labeled by a unique name or description text. Using the stored configurations, the pulse controller can be rapidly reconfigured for a new application or a different measurement procedure.

The controller also provides control signals for auxiliary power supplies of the signal switches. The function of these signals cannot be configured. The user can just decide whether the control signals are modulated using a dithering technique to reduce the spectral noise amplitude at the switch outputs.

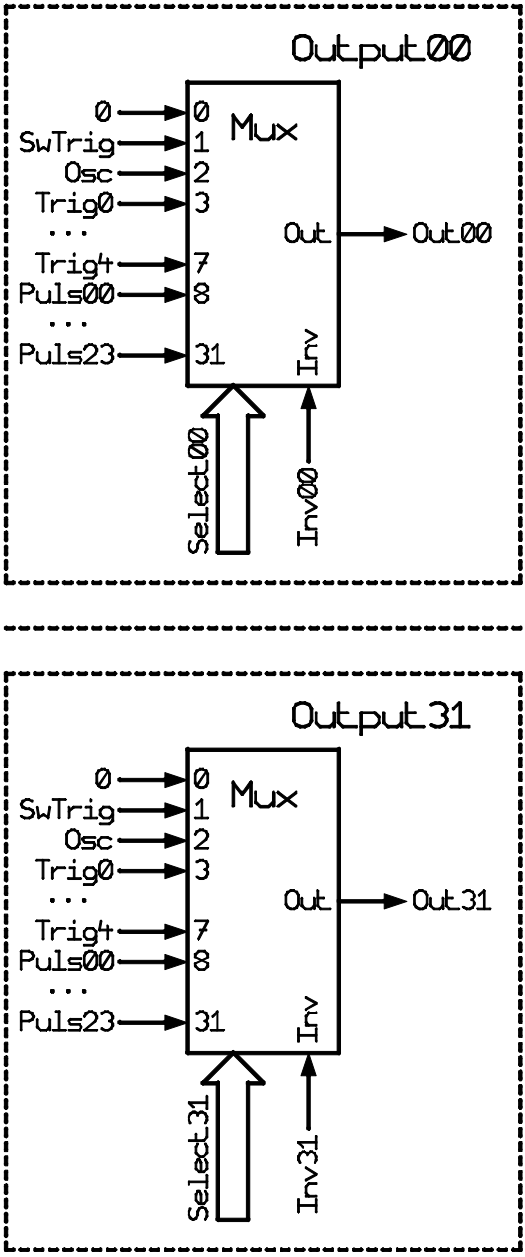

Fig. 3. Output Block Diagram.

Tab. 2. Assignment of pulse controller outputs.

Signal	Output	Explanation
Out00	DIO1-Out1	Output 1 of the digital I/O module #1
...	...	...
Out03	DIO1-Out4	Output 4 of the digital I/O module #1
Out04-07	DIO2-Out1-4	Outputs 1-4 of the digital I/O module #2
Out08-11	DIO3-Out1-4	Outputs 1-4 of the digital I/O module #3
Out12-15	DIO4-Out1-4	Outputs 1-4 of the digital I/O module #4
Out16-19	DIO5-Out1-4	Outputs 1-4 of the digital I/O module #5
Out20	AMX1-Trig	Control input of the digital switch #1
...	...	...
Out25	AMX6-Trig	Control input of the digital switch #6
Out26	Mon-Out1	Monitor output 1
...	...	...
Out31	Mon-Out6	Monitor output 6

The pulse controller is equipped with a USB data interface that allows to transfer configuration data to or from the device and remotely control it. The USB data interface is galvanically connected to the device case. When making a connection to a host computer, a large ground loop is created that can influence the performance of the experimental setup.

The software package for controlling the device contains utilities for uploading or downloading data (see sections "Software Utilities" and "Backing Up and Restoring the Data") and upgrading the firmware (see the section "Utility FlashLoader").

Tab. 3. Assignment of trigger inputs.

Signal	Input	Explanation
Trig0	DIO1-In1	Input 1 of the digital I/O module #1
...	...	...
Trig4	DIO5-In1	Input 1 of the digital I/O module #5

Quick Setup Guide

The device is delivered with cleared non-volatile memory. To setup the device, follow the next steps:

- Connect the mains cable and power the device on, the power LED lights green or yellow.
- Connect a PC to the USB connector and install the port driver (see the section "Driver Installation").
- Copy the software to suitable directory on your computer, start the utility `PulseController.exe` (see the section "Utility PulseController").
- Examine the device state by pressing 's', check if the device is enabled (entry "device enabled", see the section "Device Control") and if the power LED lights green. If the device is disabled, the power LED lights yellow. Check the state of the input "Enable" at device's PSU in such case.
- Plan the device configuration, edit the configuration file accordingly, upload it to the device by pressing 'l' (see the section "Configuration Management"), and check the output signals by an oscilloscope.
- If desired, create further device configurations and upload them (for details see the section "Configuration Management").

If you encounter problems, read carefully the corresponding section in this manual. If you cannot resolve the issue, contact the manufacturer.

Software Utilities

The software utilities can be found in the directory "Program" of the enclosed software package. Before using them, the virtual USB port driver must be installed (see sections "Driver Installation"). The utilities do not require any additional installation, you need only to copy them to a suitable directory on your computer. Before starting the utilities, you need to obtain the virtual port number to that the device is attached, as described in the section "Driver Installation".

Utility PulseController

The `PulseController` is a simple Windows™ program that runs in text mode. It enables you to control and monitor the pulse controller as well as backup and restore its data. Launching the utility `PulseController.exe` in a Windows™ command shell † without any parameters or with the parameter `-?` displays a simple help:

```
PulseController -?
```

To start the program without any error message, at least the number of the COM port must be given:

```
PulseController 6
```

This command starts the utility `PulseController` and assumes that the device is connected to the virtual port COM6. On success, the utility reports the following message:

```
Press '?' for help
```

and waits for command input.

In case of any problem, check whether the port number matches the system settings (see sections "Driver Installation") and the connected device is powered on and working properly. If an error occurs, please consult the section "Error Codes". The tables 12 and 13 explain the

† Select "Run" in the start menu of Windows™ and type "cmd". Then change the directory to that with the program files using the command "cd". Finally, execute the given command by copying & pasting and pressing "Enter". A better and more comfortable alternative to the Windows™ command shell are utilities like "File Commander/W" or "File and archive manager (FAR)". Please use the usual search utilities to find out how to obtain these applications.

possible error messages; they should help you to localize the reason for the software failure.

To check the communication, press the key 'p' to obtain the product identification text. The device should respond as follows:

```
Product identification: Pulse Controller, 32 bits,
  24 channels, Rev.1-00
```

If the device responds properly, you may try other program commands. Press '?' to obtain the help listing of all available commands[‡].

In practice, you may prefer to use the command line mode instead of the interactive mode. The former mode allows you, for instance, to save the complete commands in batch files for repeated usage.

Tables 4-11 summarize all allowable command line parameters of the program `PulseController`. The parameters are processed from left to right. If an error in the command line is encountered, the program stops with an error text showing the allowed values of the parameters.

If the parameter `-t` is found, the program stops without processing any following parameter. If you do not specify the parameter `-t` at all, the program does not stop and enters the interactive mode after having processed the complete command line.

If you wish to specify a name parameter containing spaces or special characters, use the conventions valid for your operating system. In Windows™ systems, for instance, enclose the name in quotation marks.

The quiet program mode reduces the program text output, contrary to that, the debug mode provides a detailed output for error analysis.

Device Monitoring

The command line parameters for monitoring of the device are listed in Tab. 4. They are useful in case of a malfunction when you are searching for the reason unwanted issues.

Using the parameter `-u` or `-U`, you can monitor the uptime of the device:

[‡] Note that keyboard layouts different to the US one may cause issues when evaluating several characters. We recommend to switch to the US keyboard layout when using the utility PulseLoader in the interactive mode.

Tab. 4. Command line parameters of the program PulseController - Device monitoring.

Parameter	Explanation
`-V`	get the device firmware version
`-v`	get the device hardware version
`-d, -D`	get the device firmware date
`-p, -P`	get the product identification text
`-n, -N`	get the product number
`-u`	get the device uptime
`-U`	get the device uptime periodically
`-c`	get CPU data
`-C`	get CPU data periodically
`-h`	get device housekeeping data
`-H`	get device housekeeping data periodically
`-b, -B`	get the device buffer status
`-z, -Z`	purge the communication
`-t, -T`	terminate the program
`-q, -Q`	quiet mode
`-g`	debug mode
`-G`	debug mode with output into `Debug.txt`
`-?`	show the online help

```
PulseController 6 -u -t
```

The device should respond as follows:

```
Device uptime: 5:10:06.01 sec.,   total: 69:17:24
  sec., Operation time: 4:34:12 sec.,   total:
  37:24:11 sec.
```

The value `Device uptime` shows the time elapsed from last (re)start of the device, the value `total` is a cumulated value since the device

manufacturing. Similarly, the values `Operation time` show the times during that the device was active, i.e. was not disabled.

To obtain housekeeping data of the device, use the parameters `-h` or `-H`:

```
PulseController 6 -h -t
```

The program output contains information about supply voltages of the device and about the CPU temperature:

```
Device housekeeping:    12V: 12.066V,     5V0:
  4.988V,   3V3: 3.164V,      CPU: 27.86degC
```

The voltage values should match the corresponding label, i.e. 12 V, 5 V, and 3.3 V, respectively. Under normal conditions, the CPU temperature should not exceed 40°C.

When contacting the manufacturer in case of any issue, please execute the following command:

```
PulseController 6 -V -v -d -p -n -u -h -t >
  Monitor.txt
```

The command output is redirected to the file Monitor.txt, please send it together with the description of the observed issues. It should contain the following output:

```
Firmware version: 1.00
Hardware version: 1.00
Firmware date: Dec 29 2019
Product identification: Pulse Controller, 32 bits,
  24 channels, Rev.1-00
Product number: 102703
Device uptime: 5:24:43.18 sec.,    total: 69:32:01
  sec., Operation time: 0 sec.,   total: 0 sec.
Device housekeeping:    12V: 12.066V,     5V0:
  4.988V,   3V3: 3.164V,      CPU: 27.86degC
```

Timing Control

The command line parameters for controlling the device timing are listed in Tab. 5. Using them, you can adjust or monitor the operation frequency of the built-in oscillator, the delay and trigger values of the pulse generators, and assign signals to the outputs (see the section "Description" for more details).

Tab. 5. Command line parameters of the program PulseController - Timing control.

Parameter	Explanation
`-o, -O`	get the oscillator period
`-oPeriod, -OPeriod`	set the oscillator period to the value `Period`
`-lPulsNum`	get the pulse delay of the pulse generator with the number `PulsNum`
`-LPulsNum Delay`	set the pulse delay of the pulse generator with the number `PulsNum` to the value `Delay`
`-wPulsNum`	get the pulse width of the pulse generator with the number `PulsNum`
`-WPulsNum Width`	set the pulse width of the pulse generator with the number `PulsNum` to the value `Width`
`-ePulsNum`	get configuration of the pulse generator with the number `PulsNum`
`-EPulsNum Config`	set the configuration of the pulse generator with the number `PulsNum` to the value `Config`
`-e, -E`	get pulse delay, pulse width, state, and configuration of all pulse generators
`-fOutNum`	get the configuration of the output with the number `OutNum`
`-FOutNum Config`	set the configuration of the output with the number `OutNum` to the value `Config`
`-f, -F`	get configuration of all outputs

The oscillator period (parameters `-o` and `-O`) corresponds to the value Period (see module Oscillator in Fig. 1), i.e. the period in clock pulses is given by this number plus 2.

Example: To set the oscillator period to 1 ms, execute the following command:

Tab. 6. Configuration values for the pulse generators.

Bit	7	6	5	4	3	2	1	0
Value	-	-	InvNN	SelectNN				

```
PulseController 6 -O99998 -t
```

The pulse delay and pulse width of the pulse generators (parameters `-l`, `-w`, `-L`, and `-W`) correspond to the values Delay00-23 and Width00-23 (see modules Pulser00-23 in Fig. 1), i.e. the pulse delay in clock pulses is given by the values Delay00-23 plus 3 and pulse width by the values Width00-23 plus 2.

Example: To set the pulse generator 5 to a pulse delay of 10 µs and a pulse width of 50 µs, execute the following command:

```
PulseController 6 -L5 997 -W5 4998 -t
```

The trigger selection of the pulse generators (parameters `-e` and `-E`) corresponds to the values Select00-23 and Inv00-23 (see modules Pulser00-23 in Fig. 1). The configuration value is a bit combination of the values SelectNN (see Tab. 1) and InvNN (see Tab. 6). Note that the two most significant bits are not used for the configuration values.

Example: To set the trigger of the pulse generator 3 to the inverted (Inv03 = 1) oscillator output (Select03 = 2, see Tab. 1), execute the following command:

```
PulseController 6 -E3 34 -t
```

In a similar way, the outputs can be assigned (parameters `-f` and `-F`). The configuration value is again a bit combination of the values SelectNN (see Tab. 1) and InvNN (see Tab. 6).

Example: To assign the monitor output 1 (Mon-Out1 = Out26, see Tab. 2) to the non-inverted (Inv07 = 0) output of the pulse generator 12 (Select07 = 20, see Tab. 1), execute the following command:

```
PulseController 6 -F26 20 -t
```

Note that by using the parameters `-e`, `-f`, `-E`, and `-F` without further arguments, the parameters of all pulse generators or outputs can be listed.

Example: To list the parameters of all pulse generators and all outputs, execute the following command:

Tab. 7. Command line parameters of the program PulseController - Device control.

Parameter	Explanation
`-#`	restart the device
`-r`	issue a positive software trigger pulse
`-R`	issue a negative software trigger pulse
`-rTtPp, -RTtPp`	issue a software trigger with defined parameters, TtPp are 4 Boolean arguments [0,1]
`-s, -S`	get the device status
`-sdopti, -Sdopti`	set device configuration, d, o, p, t, i are 5 Boolean arguments [n,y]

```
PulseController 6 -e -f -t
```

Device Control

Several commands are available for device control: The device can be restarted, triggered by software or its configuration can be adjusted (see Tab. 7).

To restart the device, execute the following command:

```
PulseController 6 -# -t
```

When the device restarts, the complete configuration is stored in the non-volatile memory (NVM). Thus, the restart can be used to store data and prevent an accidental data loss.

To control the software-trigger engine (see Fig. 2), use the parameters `-r` or `-R`. The signals SoftwareTrigger and SoftwarePulse can be controlled by 4 Boolean arguments ('0' or '1') "TtPp". The value "T" is the value SoftwareTrigger in first cycle and the value "t" in the second one, respectively. Similarly to this, the value "P" is the value SoftwarePulse in first cycle, the value "p" in the second one, respectively.

Example: The execution of the following command:

```
PulseController 6 -r1000 -t
```

produces a positive pulse at the signal SoftwareTrigger, while SoftwarePulse is kept at 0. This results to a positive pulse at the engine

Tab. 8. Bit values of the device status.

Bit	15	14	13	12
Value	Trig4	Trig3	Trig2	Trig1
Bit	11	10	9	8
Value	Trig0	Disable	ClrN	SwTrigOut
Bit	7	6	5	4
Value	DisIlock	DisDither	-	SwPulse
Bit	3	2	1	0
Value	SwTrig	PlusEnb	OscEnb	DevEnb

output SwTrig. Note that the pulse length is defined by the software and may vary.

To produce a precise pulses, the signal SoftwarePulse should be used: To produce a positive 1-clock (10 ns) wide pulse at the engine output SwTrig, execute the following command:

```
PulseController 6 -r0000 -r0010 -t
```

The first command sets SoftwareTrigger and SoftwarePulse to 0, the second one pulses SoftwarePulse to 1 and back to 0, this generates the abovementioned pulse. Note that the command

```
PulseController 6 -r0010 -t
```

can be abbreviated as follows:

```
PulseController 6 -r -t
```

Similarly, the command

```
PulseController 6 -r1110 -t
```

can be shortened as follows:

```
PulseController 6 -R -t
```

It sets SoftwareTrigger to 1 and pulses SoftwarePulse to 1 and back to 0, i.e. generates a negative 1-clock (10 ns) wide pulse at the engine output SwTrig.

The parameters `-s` or `-S` can be used to control and monitor the device status (see Tab. 8). Note that only the bits 0-7 can be modified, the bits 8-15 are read-only. The values Trig0-4 are the logic levels at

the trigger inputs, i.e. the inputs 1 of the digital I/O modules DIO1-5 (see Tab. 3). The utility shows these bits as "DIOx-In1 input", where x = 1-5.

The bit ClrN is the module reset signal active at 0. If the bit is 0, the module is reset and not operable. This state is indicated by the power LED, it lights yellow and changes its color to green if the reset signal is deactivated, i.e. set to 1. The utility shows this bit as the value "device enabled". The module can be reset by putting the bit DevEnb (shown as "device enable") to 0 or by opening the interlock (BNC connector "Enable" at the module PSU) while the bit DisIlock (displayed as "interlock disable") is 0. If the bit DisIlock is set to 1, the interlock state does not play any role and the device can be disabled by the bit DevEnb only.

The interlock state is replicated by the bit Disable, shown as the value "interlock". It the bit is set, the interlock is open, i.e. deactivates the device. Note that the real physical interlock state depends on the polarity switch at the module PSU.

The bit SwTrigOut (displayed as "soft trigger out") is the logic level at the output of the software-trigger engine (see Fig. 2). The input signals SoftwareTrigger and SoftwarePulse correspond to the bits SwTrig and SwPulse shown as "software trigger" and "software pulse". Note that these signals can be controlled by the parameters `-r` or `-R` only.

The bit DisDither (displayed as "dithering disable") indicates whether dithering is used to reduce the spectral noise amplitude at the switch outputs or not (see the section "Description" for more details). If the bit is reset (0), the dithering is enabled. If set (1), the auxiliary power supplies of the signal switches run at a fixed frequency stabilized by a quartz. This may produce noise spectrum with sharp peaks in the region above 25 kHz. Switching between these two modes may be helpful when identifying unwanted interference caused by the device.

The bits OscEnb and PlusEnb (shown as "oscillator enable" and "pulser enable") enable the oscillator and the pulse generators, respectively. They correspond to the signals OscillatorEnable and PulserEnable (see Fig. 1).

To set the device configuration, 5 Boolean arguments have to be specified behind the parameters `-s` or `-S`. They are shown in Tab. 7 as `dopti`, these letters have to be replaced by arguments 'n'/'N' or

'y'/'Y'. The argument 'd' stays for the bit DevEnb ("device enable"), 'o' is OscEnb ("oscillator enable"), 'p' PlusEnb ("pulser enable"), 't' means DisDither ("dithering disable"), and 'i' DisIlock ("interlock disable").

Example: To enable the device, the oscillator, the pulse generators, the dithering, and the interlock, execute the following command:

```
PulseController 6 -Syyynn -t
```

Configuration Management

The device manages several configurations: the current one describing the present setting of the device and up to 500 user configurations that can be loaded or saved and labeled by a name. All configurations are stored in the non-volatile memory (NVM) of the device. Thus, if the device restarts, the device configuration is reloaded from the NVM so that the device state is restored exactly to the state immediately before the shutdown.

Tab. 9. Items of configuration files.

Entry	Value
Oscillator	Period (see Fig. 1)
PulserNN (NN = 00-23)	DelayNN, WidthNN (see Fig. 1), SignalNN (see Value in Tab. 1), InvNN ("Pos" or "Neg")
DIOx-Outy (x = 1-5, y = 1-4)	SignalNN (see Value in Tab. 1), InvNN ("Pos" or "Neg")
AMXx-Trig (x = 1-6)	
Mon-Outx (x = 1-6) (see Output in Tab. 2)	
DeviceEnable	DevEnb ("Y"/"N" see Tab. 8)
OscillatorEnable	OscEnb ("Y"/"N" see Tab. 8)
PulserEnable	PlusEnb ("Y"/"N" see Tab. 8)
SoftwareTrigger	SwTrig ("Y"/"N" see Tab. 8)
SoftwarePulse	SwPulse ("Y"/"N" see Tab. 8)
DisableDithering	DisDither ("Y"/"N" see Tab. 8)
DisableInterlock	DisIlock ("Y"/"N" see Tab. 8)

To manage the configurations, several commands are available (see Tab. 10). It is possible to save and load a specified configuration to or from a data file. The configurations can be copied, deleted, and undeleted, finally, a list of available configurations can be obtained.

Using the parameters `-i` or `-I`, the current configuration can be saved to or loaded from a text file. The file uses a format of Windows™ INI files. It begins with a section label `[CurrentConfiguration]` followed by rows with a format `Entry=Content`, where `Entry` is the name of the partial item and `Content` its value. For the available items, see Tab. 9.

Example: To save the current configuration in the file `Config.ini`, execute the following command:

```
PulseController 6 -i Config.ini -t
```

You may edit it and reload by the following command:

```
PulseController 6 -I Config.ini -t
```

Note that you must not delete any row if you edit the configuration file. It must contain all items, otherwise an error will be reported when you try to load the data from the file.

In the NVM, up to 500 user configurations can be saved, numbered from 1 to 500. Each user configuration can be labeled by a name of up to 255 characters.

Using the parameters `-j` or `-J`, the current configuration can be loaded from or saved to a specified user configuration in the NVM. To save the current configuration to the user configuration with the number 1, execute the following command:

```
PulseController 6 -J1 -t
```

By executing the following command:

```
PulseController 6 -A1 "Name of the config." -t
```

the user configuration can be labeled by the name "`Name of the config.`".

You may list the available user configuration by executing the following command:

```
PulseController 6 -m -t
```

Tab. 10. Command line parameters of the program PulseController - Configuration management.

Parameter	Explanation
`-i FileName`	save current configuration to a text file with the name `FileName`
`-I FileName`	load current configuration from a text file with the name `FileName`
`-jConfNum`	load current configuration from the user configuration with the number `ConfNum`
`-JConfNum`	save current configuration to the user configuration with the number `ConfNum`
`-m`	list active user configurations
`-M`	list deleted user configurations
`-xConfNum`	delete user configuration `ConfNum`
`-XConfNum`	undelete user configuration `ConfNum`
`-aConfNum`	get the name of the user configuration with the number `ConfNum`
`-AConfNum Name`	set the name of the user configuration with the number `ConfNum` to the text `Name`
`-kConfNum FileName`	save data of the user configuration with the number `ConfNum` to a text file with the name `FileName`
`-KConfNum FileName`	load data of the user configuration with the number `ConfNum` from a text file with the name `FileName`
`-k FileName`	save data of all available user configurations to a text file with the name `FileName`
`-K FileName`	load data of all available user configurations from a text file with the name `FileName`

By the following command, the user configuration with the number 1 can be deleted:

```
PulseController 6 -x1 -t
```

You will be asked for confirmation before the user configuration is deleted. If it occurs in error, you may restore it by the command:

```
PulseController 6 -X1 -t
```

Similarly to the current configuration, also the user configurations can be saved to or loaded from a text file. The file uses the same format like the files of the current configuration (see Tab. 9), the only difference is the section label that reads `[ConfigurationN]` where N = 1-500 is the number of the user configuration.

Example: To save the data of the user configuration with the number 10 in the file `ConfigUser10.ini`, execute the following command:

```
PulseController 6 -k10 ConfigUser10.ini -t
```

To save all current configurations in the file `ConfigAll.ini`, execute the following command:

```
PulseController 6 -k ConfigAll.ini -t
```

Note that in this case, the configuration file contains several section labels corresponding to all available user configurations.

Backing Up and Restoring the Data

The data stored in the device's NVM can be backed up or restored (see Tab. 11). The data includes all configuration settings. This means that a restore procedure rolls back the device exactly to the state it was in at the backup time. Thus, if any tuning of the settings are planned, it is advisable to create a backup before with which the original state can be restored.

To back up the system memory into a data file `Memory.txt`, execute the following command:

```
PulseController 6 -y MemoryData.txt -t
```

This command downloads the memory data from the device into the file `Memory.txt`.

To restore the data, execute the following command:

Tab. 11. Command line parameters of the program PulseController - Backup and restore.

Parameter	Explanation
`-y FileName`	backup memory data into a text file with the name `FileName`
`-Y FileName`	restore memory data from a text file with the name `FileName`

```
PulseController 6 -Y MemoryData.txt -t
```

This command uploads the memory data from the file `Memory.txt` to the device.

Utility FlashLoader

The FlashLoader is a simple Windows™ program running in text mode. It enables you to upgrade the firmware of the pulse controller. You should perform the upgrade if you have received or downloaded a new firmware file from the device manufacturer. Launching the utility `FlashLoader.exe` without any parameters displays a simple help text with the expected syntax of the command line.

Before upgrading the firmware, you should first test the device and the communication by verifying the current firmware version. To do so, start the following command:

```
FlashLoader 6 Firmware.txt -v
```

where `Firmware.txt` is the file containing the current firmware and the number 6 indicates the port COM6 to which the device is connected. The program should produce the following output:

```
Code file Firmware.txt from 12/20/2019, 12:00:00
Flash Loader 1.12
Verifying code file Firmware.txt
Verifying finished at Fri, 12/20/2019, 13:42:23
15725 (3D6Dh) bytes processed, 15104 (3B00h) bytes
verified
Resetting the target
Program finished ok
```

For the verify procedure, a flash loader utility at the device is activated. When the verify finishes without any error, the device is restarted.

! **Attention:** To be sure that the device cannot activate the attached switches or other peripherals when the FlashLoader is active, disconnect the output cables of the switches and digital I/O modules or remove these plug-in modules from the chassis.

If any error occurs, do not proceed with the firmware upgrade. If you cannot resolve the issues, contact the manufacturer. Note that even if the verify fails and the flash loader at the device remains active, it is safe to power the device off to restart it. However, a more safe and comfortable alternative to that is to execute the following command:

```
FlashLoader 6 -i -f
```

This prevents the utility at the host computer from initializing the flash loader utility at the microcontroller again and sends the reset command to the device.

If the verify has succeeded, you may start the firmware upgrade by entering the command:

```
FlashLoader 6 Firmware.txt
```

where `Firmware.txt` is the file with the new firmware. The program should produce the following output:

```
Code file Firmware.txt from 12/20/2019, 12:00:00
Flash Loader 1.12
Programming code file Firmware.txt
Programming finished at Fri, 12/20/2019, 13:42:23
15725 (3D6Dh) bytes processed, 15104 (3B00h) bytes
programmed
Resetting the target
Program finished ok
```

Also for the programming procedure, a flash loader utility at the device is activated. When the programming finishes, the device is restarted with the new firmware.

If an error occurs, the flash loader utility at the microcontroller may remain active. This is the case if the device did not restart. In this case, you may retry the action with the command line parameter '-i':

```
FlashLoader 6 Firmware.txt -i
```

This will prevent the utility at the host computer from initializing the flash loader utility at the microcontroller again and it will just try to re-program the file `Firmware.txt`. If the error persists, contact the manufacturer.

! **Attention:** You must not power down the device if the firmware upgrade did not succeed. Otherwise, the device will not operate properly or might even not restart at all. Did this happen, it would be necessary to reprogram the device in the factory.

Error Codes

Tab. 12. Return values of the interface functions

Return value	Error message	Description
0	No error	The data transfer finished successfully.
-2	Error opening the port	The port could not be opened. For the possible reasons, see Tab. 13.
-3	Error closing the port	The port could not be closed. For the possible reasons, see Tab. 13.
-4	Error purging the port	The port buffers could not be cleared.
-5	Error setting the port control lines	The port control lines could not be set.
-6	Error reading the port status lines	The port status lines could not be read.
-7	Error sending command	The data transfer to the device failed. For the possible reasons, see Tab. 13.
-8	Error sending data	
-9	Error sending termination character	
-10	Error receiving command	The data transfer from the device failed. For the possible reasons, see Tab. 13.
-11	Error receiving data	
-12	Error receiving termination character	
-13	Wrong command received	The device sent an unexpected response.
-14	Wrong argument received	
-15	Wrong argument passed to the function	One of the arguments passed to the function was out of the allowable range.

Return value	Error message	Description
-100	Device not connected	The port status lines indicate that the device is not connected.
-101	Device not ready	The port status lines indicate that the device is not ready. The communication with the device is possible only if it does not execute any process.
-102	Device state could not be set to not ready	The device did not react properly. Try to reset the communication or restart the device by powering it off and on.
-400	Error opening the file for debugging output	The file for debugging output cannot be opened for writing. Check if you have permissions to perform this action or if the file exists and is opened by another application.
-401	Error closing the file for debugging output	The file for debugging output cannot be closed. Check if the access to the file is still possible.

Tab. 13. I/O errors

Return value	Error message	Description
0	No error	The data transfer finished successfully.
1	Port has not been opened yet	You attempted to use the communication channel before having opened it.
2	Cannot open the port	The specified port could not be opened. Either the port does not exist or it is being currently used by another program.
3	Cannot get the state of the port	The system could not get the state of the port.
4	Cannot set the state of the port	The system could not set the state of the port.
5	Cannot set the timeouts for the port	The system could not set the timeouts for the port.
6	Cannot clear the port	The system could not clear the port buffers.
7	Error reading data from the port	The system could not read data from the port. Most probably, no data is available because the device is either disconnected or does not respond.
8	Error writing data to the port	The system could not write data to the port.
9	Wrong data amount written to the port	The system could not write the proper data amount to the port.
10	Error setting the control lines of the port	The system could not set the state of the port control lines.
11	Error reading the status lines of the port	The system could not get the state of the port status lines.
12	Device is busy	The system could not access the device since menus or dialog boxes are active.

Driver Installation

Installation of the Virtual Port for the USB Interface

The virtual port driver is required for the operation of the device with a USB interface. If you use the operating system Windows™, please note the following:

- Please use the update function of the operating system at the host computer or download the most recent driver from the homepage of the manufacturer of the USB adapter. The drivers are located at the following address: http://www.ftdichip.com/Drivers/VCP.htm. Please choose the correct driver version according to your operating system.
- To install the driver, administrative rights are required.
- The installation is described in detail in the "Installation Guides" available at the abovementioned address. Please read this description carefully before starting the installation.
- After the installation, the number of the virtual port can be set. You can change the settings in the device manager by opening the settings of the device *USB Serial Port (COMx)*. To modify the settings, administrative rights are required. The settings are applied immediately, you do not need to reboot the PC to activate them.

The software can also be used at computers running the Linux operating system. You can run them using the Windows™ emulator wine (see http://www.winehq.org/).

Starting with Linux Kernel 3.0.0-19, all FTDI devices are already supported without the necessity of compiling additional kernel modules. For more details, consult the homepage of the manufacturer of the USB adapter: http://www.ftdichip.com/Drivers/VCP.htm.

The system has to be configured an the following way:

- Use, for instance, the program 'dmesg' to find out to which USB port the device is attached: Look for a line similar to "FTDI USB Serial Device converter now attached to ttyUSB0"
- Link the Linux device to the virtual COM port of wine:
  `ln -s /dev/ttyUSB0 ~ /.wine/dosdevices/com6`
  This assumes that the device is attached to ttyUSB0 and will be linked with COM6.