Clean Jump Implementation

Implementation Guide: Clean Jump on PPCL600 and PPCL700

The Pure Photonics products provide a very low frequency noise, making them very suitable for sensing applications. In that low-noise mode, the Pure Photonics firmware adds frequency flexibility, so that the laser can be quickly moved from one frequency to another. Dependent on the distance of the temperature setpoint for the beginning and ending frequency point, the switching time can be as fast as 0.3 seconds or as slow as 1.0 seconds.

Frequency jumps in low multiples of 15GHz and multiples of 275GHz are most suitable for fast switching (typically <0.3 seconds).

This implementation guide describes how to operate the Clean Jump on the PPCL6xx and PPCL7xx product families (micro-ITLA). This will also work on PPCL5xx enclosures with micro ITLA (serial number starting with 'CRT**M**' or 'PP7').

1. Graphical User Interface

The GUI is implemented with separate versions for the ITLA formfactor tunable laser and micro-ITLA tunable lasers. As the micro-ITLA implementation is more self-contained (no external calibration factors needed) the ITLA version will not work for micro-ITLA.

The micro-ITLA version has 3 steps:

- Connect; to connect to the laser
- Settings; to calibrate the laser and to set-up the clean-jump sequence
- Cleanjump; to execute the clean jump.

Calibration is only needed if no prior calibration has been done.

In step 2 the user can initate the calibration of the setpoints. This is done by setting the first channel, the channel spacing and the number of channels (max 500). Upon pressing the calibrate button the calibration will be started and the button will display which set-point is being tested. Pressing the button again will stop the calibration. Each setpoint will take 30-60 seconds.

After calibration, the jump sequence can be set in terms of setpoint ID (i.e. make sure that each setpoint has been calibrated and make sure you know which setpoint is associated with which frequency).

In step 3 the loaded sequence is executed.

2. RS-232 interface

The RS-232 interface follows the conventions, as outlined in the OIF-MSA (http://www.oiforum.com/wp-content/uploads/OIF-ITLA-MSA-01.3.pdf). Several registers have been added to enable the Clean Jump functionality.

Pre-conditions

To start the Clean Jump, the device will need to have been enabled and have been switched to the whispermode (write 2 to register 0x90).

Operating the Clean Jump

The clean jump is operated by loading the desired setpoint and then activating the jump. The setpoint is loaded by writing the desired setpoint index (see under calibration) to register 0xD0. In case you wish to access channel 1 or 2, you will need to provide a value >1 which has the logic ((value & 0x1f = desired setpoint) &&!(value&0x8000)). The bit 15 (0x8000 is used to access the setpoint information (see later) and should be set to 0).

After selecting the setpoint, the Clean Jump is operated by writing 1 to register 0xD0. The Clean Jump is terminated by either writing 0 to register 0xD0 or by waiting for the Clean Jump to complete. Reading register 0xD0 will give you the status (1 for active, 0 for not active).

The offset can be read through register 0xD1 in units of MHz. Note that this is not an actual offset, but rather a calculated frequency offset based on the offset of the different tuning elements. As such, a smaller value means that the frequency is closer to target.

Calibrating for the Clean Jump

In order to operate the Clean Jump the target points will need to be loaded to the permanent memory of the device.

NOTE: saving the setpoints involves writing to the FLASH memory of the device. In the automated method several (16) setpoints are written at the same time. In the manual method one setpoint is written per time. The number of allowed writes to each memory location is limited but allowed for at least 10,000 times.

In the manual method, writing setpoint 5 and writing setpoint 6 would be two writes to the memory location of setpoint 5 and 6. For the automatic method, it would only involve one write.

Method 1 – Grid based, automated

These setpoints need to be on a grid defined by the FCF (First Channel Frequency) and the GRID (grid spacing). These setpoints are all at the same power level (defined by the PWR register).

The calibration is started by writing the number of channels to register 0xD2. After issuing that command the laser will turn itself on at the different channels and write the collected device operating conditions to the device permanent memory. You can monitor this through the output power (and frequency) from the fiber.

For example, if grid is set to 500 (50GHz grid) and FCF1 is set to 192 and FCF2 is set to 5000 (192.5THz for FCF) and PWR is set to 1350 (13.5dBm), then writing 5 to register 0xD2 will collect the setpoints at 192.5, 192.55, 192.6, 192.65 and 192.70 THz.

Register 0xD2 can be read to know if the calibration is ongoing (bit 15 is 1 for active; 0 for not active; The setpoint that is being calibrated is in the first couple of bits). After the calibration the device will need to be reset.

This calibration will only need to be done once. The setpoints are stored in permanent memory.

Method 2- Manual

The calibration is started by writing the channel ID to register 0xD2 with the bit 15 set (i.e. 0x8005 to write to setpoint 5. The device will turn on at the FCF and the power level set in the respective registers and after completion the information is saved in permanent memory.

External memory for storing setpoints

The internal memory of the device is limited to about 500 setpoints. It is possible to port some setpoints from the device for storage by user. The user can then feed those setpoints back when needed.

Each setpoint consists out of 32 bytes. The individual bytes can be read through register 0xD0 with the bit 15 set to 1 and the byte ID in the datafield. E.g. reading from register 0xD0 with data value 0x8003 would give you the 4th byte (0 is for the first byte).

The setpoint can be written to the register through first selecting a (random) setpoint and then overwriting those data (the permanent memory is not altered). The new values are loaded by setting bit 15 to 1 and (data&0x7f00)>>8 being the byte ID and data&0xff being the data. After loading the 32 bytes, the device is primed to jump to this setpoint and a write of the value 1 to register 0xD0 will trigger that jump. Note that if the overload of values is incomplete, the jump will still be executed, but likely with an unfavorable result.