Novoptel

Application note 3
EPS1000 internal execution tables

Revision history

Version	Date	Remarks	Author
0.9.1	26.06.2015	Draft version	B. Koch
0.9.2	13.11.2017	Adaption to EPS firmware version ≥ 1.1.0.0	B. Koch
0.9.3	19.03.2018	Examples for position and nominal scrambling speed tables added	B. Koch

Contents

Summary

This Application Note describes the configuration of internal tables to be executed by the Novoptel EPS1000 polarization scrambler/transformer. They can be used for various applications such as PDL measurement and recirculating-loop experiments. For EPS with firmware version < 1.1.0.0 please refer to version 0.9.1 of this application note.

Novoptel GmbH
EIM-E
Warburger Str. 100
33098 Paderborn
Germany
www.novoptel.com

EPS1000 internal execution tables

The internal execution table (= Table) stores either

- 7 static waveplate positions,
- 7 waveplate nominal scrambling speeds and rotation directions, or
- 16 static electrode voltages

plus a dwell time for each of up to 1024 settings (=Table Positions).

For device characterization or similar applications, samples of the optical power or SOP can be taken at the end (= after application) of each table step. This can be done externally, preferably timed with the trigger input or output. When an optical receiver input is provided by the EPS1000 model, samples of the power detector are taken and stored directly before the next table step is executed. The averaging time for these measurements is given by $T = 80\ ns \bullet 2^{ATE}$, where ATE is the Averaging Time Exponent (ATE) that can be set in the GUI or by writing to register #129. The internal memory for measurement data stores a maximum of 1024 samples, e.g. for a complete table execution.

Creating tables using text files

Each table text file starts with an identifier string for the table type. This identifier string must be one of the following three:

```
table_mode='speed'
table_mode='position'
table_mode='voltage'
```

The other rows of the text file correspond to one set of nominal scrambling speeds, wave plate positions or electrode voltages. Depending on the table type, the row are interpreted differently:

Waveplate nominal scrambling speed tables:

Each row is a string containing 15 integers, separated by a comma. The first 7 integers define the waveplate rotation direction in sequence QWP0, QWP1, QWP2, HWP, QWP3, QWP4 and QWP5. A value 1 means forward rotation, 3 means backward rotation, and 0 means no rotation.

The next 7 integers define the nominal scrambling speed of each waveplate. The speed of the QWPs are given in rad/s*100, whereas the speed of the QWP is given in krad/s*100.

The last integer in the row defines the dwell time between two table positions (table entries) in nanoseconds. The minimum dwell time is 200 ns. Note that when measuring with an internal power detector, dwell times must be longer than the averaging time of the measurement. While one can generate tables with dwell times of up to 1 second with the GUI (version <= 1.4.4.4), manual creation of table text files allow dwell times of up to 40 seconds.

An example row looks like this:

```
1, 3, 1, 1, 3, 1, 3, 13226, 6137, 17342, 10000, 9451, 11764, 7976, 1000000000
```

QWPs number 0, 2, and 4 rotate in backward direction, whereas QWPs 1, 3, 5 and the HWP rotate in forward direction. The nominal scrambling speed of QWP0 is 13226/100 rad/s = 132,26 rad/s. The nominal scrambling speed of the HWP is 10000/100 krad/s = 100 krad/s. The dwell time is 1.000.000.000 nanoseconds = 1 second.

Waveplate position tables

The waveplate positions are given as a rotation 0..1, where 1 means a full waveplate eigenmode rotation (by 360° electrically or 180° physically), multiplied by 2^{16} and rounded to the nearest integer.

An example row looks like this:

```
1820, 0 0, 63716, 0, 0, 0, 200
```

With this row, QWP0 is positioned at +10° (1820/2^16*360≈10) whereas the HWP is positioned at 350°=-10° (63716/2^16*360≈350). The dwell time is 200 ns.

Electrode voltage tables

The first 16 integers in the row define the electrode voltages. An offset of 8192 corresponds to 0 Volt, the upper and lower limitations of 8192±6000 correspond to about ±45 Volt. The first two elements correspond to electrodes 1 and 2 of the first of eight sections of the $LiNbO_3$ polarization transformer. Elements 3 and 4 correspond to electrodes 1 and 2 of the second section and so on.

An example row looks like this:

```
9192, 7192, 8192, 8192, 8192, 8192, 8192, 8192, 8192, 8192, 8192, 8192, 8192, 8192, 8192, 8192, 1000
```

The first (second) electrode of the first section is set to +1000 (-1000). All other sections are set to 0. the dwell time is 1 µs.

Creating Tables using Matlab

The following Matlab code generates an execution table and stores it in the EPS1000. Two basic communication functions are used for USB transfer to and from a connected EPS100: `writeeps(x, y)` will write a value y into register x, and `readeps(z)` returns the data contained in register z. For details of this functions please refer to the EPC1000 or EPS1000 data sheet. The code can be transferred into any language that allows data transfer using the FTDI USB driver.

Electrode voltage tables:

In the following example, 2 (“steps”) times 4 electrode voltages are applied to the section that normally acts as the HWP. The other waveplates are not affected by the table and can still be used for constant nominal scrambling speeds or positions. The electrode voltage values are stored in the matrix “mvolthwp”:

```
steps=2;
mvolthwp=[ 0 0 0 0 0 0 -100 -100 -100 -100 0 0 0 0 0 0 ;
           0 0 0 0 0 0  100  100  100  100 0 0 0 0 0 0 ]+8192;

dwell_time_ns= [200; 300]; % only used if reg 220 is set to 1, or in table mode

writeeps(239, 3); % 3: Electrode Voltage Table
writeeps(229, bin2dec('0001000')); % only hwp: Enable Table Sync

% Write Table to EPS
WriteTable(mvolthwp, dwell_time_ns, 0)
% set new table length
writeeps(228, steps);

writeeps(126, 0); % Use only Photodetector 1
writeeps(132, 0); % Disable Timed Output
writeeps(224, 0); % Disable Ext Trigger
writeeps(218, 1); % Table mode: row
writeeps(220, 0); % Disable continuous table execution

% Internal trigger: only used if reg 225 is set to 1
Execution_Time_Int=250000; % 250000 * 40 ns = 10 ms
writeeps(222, mod(Execution_Time_Int, 2^16)); % lower 16 Bits
writeeps(223, bitshift(Execution_Time_Int, -16)); % upper 16 Bits

writeeps(226, 2); % Disable trigger output but leave bnc as output
```

```
   % select one of the following trigger sources:
   %writeeps(225, 1); % Enable INT Trigger, Reset table & counter
   writeeps(220, 1); % Enable continuous table execution
```

A separate function for writing the table into the EPS registers, used for all table types:

```
% writes voltage table to EPS
function  WriteTable(V, dwell_time_ns, nvolt)

% determine the number of table entries (rows)
npos=size(V,1);

if nvolt==0, % write all voltages of mvolt
   xvolt = 1:size(V,2);
else
   xvolt = nvolt; % write only specified row of mvolt
end;

for ii=1:npos

   writeeps(219, ii-1); % select row

   % dwell time in 40ns-ticks:
   dwell_time_ticks=max(4, round(dwell_time_ns(ii)/40)-1);
   writeeps(250, mod(dwell_time_ticks, 2^16)); % lower 16 Bits
   writeeps(251, bitshift(dwell_time_ticks, -16)); % upper 16 Bits
   trigger = 1;

   % table data:
   for iii=xvolt
      writeeps(252+iii-1, V(ii,iii)); % store current electrode voltages in registers
      trigger = bitor(trigger, 2^(iii-1));
   end;

   % Table write trigger
   writeeps(221, trigger);

end;
```

Waveplate position tables:

In the example above, replace the first few code lines with the following lines. Please note that in contrast to voltage tables, the WP sequence in the registers is HWP-QWP0-QWP1-…-QWP5.

```
   steps=2;
   wppos=[  1820 0 0 0 0 0 0;
          63716 0 0 0 0 0 0];

   dwell_time_ns= [2000; 3000]; % only used if reg 220 is set to 1, or in table mode

   writeeps(239, 1); % 1: WP Position Table
   writeeps(229, bin2dec('0001000')); % only hwp: Enable Table Sync

   % Write Table to EPS
   WriteTable(wppos, dwell_time_ns, 0)
```

Waveplate nominal scrambling speed tables:

The WP nominal scrambling speed table has the same WP sequence as the WP position table, but the nominal scrambling speed needs to be split into two registers:

```
   steps=2;
   wpspeeds=[  1000 200000 0 0 0 0 0;  % HWP=10krad/s, QWP0=2krad/s
               1000 200000 0 0 0 0 0]; % HWP=10krad/s, QWP0=2krad/s
   wprotdir=[  1    3      0 0 0 0 0;  % HWP=forward,  QWP0=backward
               3    1      0 0 0 0 0]; % HWP=backward, QWP0=forward

   clear table;
```

```
for st=1:steps,
   for wp=1:7,
      table(st,(wp-1)*2+1)=mod(wpspeeds(st,wp), 2^16);
      table(st,(wp-1)*2+2)=bitshift(wpspeeds(st,wp), -16) + wprotdir(st,wp)*2^14;
   end;
end;

dwell_time_ns= [1000000; 2000000]; % only used if reg 220 is set to 1, or in table mode

writeeps(239, 2); % 2: WP Nominal Scrambling Speed Table
writeeps(229, bin2dec('1001000')); % only hwp and qwp0: Enable Table Sync

% Write Table to EPS
WriteTable(table, dwell_time_ns, 0)
```

Table execution

First, one has to decide if a trigger event shall just set the table to the next row (row mode) or if the whole table shall be executed upon each trigger (table mode). Select one of the following:

```
writeeps(218, 1); % Table mode: row
```

```
writeeps(218, 0); % Table mode: table
```

A trigger event can be launched by a USB/SPI/LAN command, by an internal counter or by an external trigger signal:

```
writeeps(227, 1) % Single trigger by USB/SPI/LAN command
```

```
writeeps(225, 1); % Enable INT Trigger, thereby reset table & counter
```

```
writeeps(226, 0); writeeps(224, 1); % Config. BNC as input; Enable Ext Trigger
```

The EPS can also execute the table continuously, independent of trigger signals:

```
writeeps(220, 1); % Enable continuous table execution
```