Polarization controller instructions for MCP2210

User guide

Document No.: -

Revision: 1.2

CONFIDENTIAL INFORMATION

1 Legal Notice

PLEASE READ THE FOLLOWING TERMS AND CONDITIONS CAREFULLY.

1.1 Disclaimer and "Terms of Use"

INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH BEYOND SEMICONDUCTOR'S MATERIALS. THE MATERIALS ARE PROVIDED "AS IS" WITHOUT ANY EXPRESS OR IMPLIED WARRANTY OF ANY KIND INCLUDING WARRANTIES OF MERCHANTABILITY, NONINFRINGEMENT OF INTELLECTUAL PROPERTY, OR FITNESS FOR ANY PARTICULAR PURPOSE. IN NO EVENT SHALL BEYOND SEMICONDUCTOR OR ITS SUPPLIERS BE LIABLE FOR ANY DAMAGES WHATSOEVER (INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF PROFITS, BUSINESS INTERRUPTION, LOSS OF INFORMATION) ARISING OUT OF THE USE OF OR INABILITY TO USE THE MATERIALS, EVEN IF BEYOND SEMICONDUCTOR HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. BECAUSE SOME JURISDICTIONS PROHIBIT THE EXCLUSION OR LIMITATION OF LIABILITY FOR CONSEQUENTIAL OR INCIDENTAL DAMAGES, THE ABOVE LIMITATION MAY NOT APPLY TO YOU.

Beyond Semiconductor and its suppliers further do not warrant the accuracy or completeness of the information, text, graphics, links or other items contained within these materials. Beyond Semiconductor may make changes to these materials, or to the products described therein, at any time without notice. Beyond Semiconductor makes no commitment to update the Materials. The copyright for any materials created by the Beyond Semiconductor is reserved. Any duplication or use of any material in other electronic or printed publications is not permitted without the Beyond Semiconductor's agreement. Adherence to all applicable laws and regulations, federal and state and local, governing professional licensing, business practices, advertising and all other aspects of doing business in the US or any other jurisdiction is the sole responsibility of the user

Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined." Beyond Semiconductor reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them.

Any performance information, tests, ratings or specifications provided herein are measured using certain computer systems or components and only indicate the approximate performance of Beyond Semiconductor's products in accordance with those tests. Use of other computer systems or components, software, etc. may affect actual performance. Customers and prospective customers should consult with other, more reliable, sources of information before purchasing Beyond Semiconductor's products.

Proceeding beyond this disclaimer constitutes acceptance of these terms and conditions.

1.2 Trademarks

All brand and product names are trademarks or registered trademarks of their respective owners.

2 Revision History

Rev.	Date	Author(s)	Comment
1.0	20.12.2024	R.B.	Initial version
1.1	08.01.2025	R.P.	Review
1.2	08.01.2025	R.B.	Added a figure of the setup

3 Table of Contents

4 Scope of the Document

Document describes user defined run parameters settings in Python and through command line to control MCP2210 breakout module via USB.

C library is also supported but has not been tested. Refer to MCP2210 documentation for details.

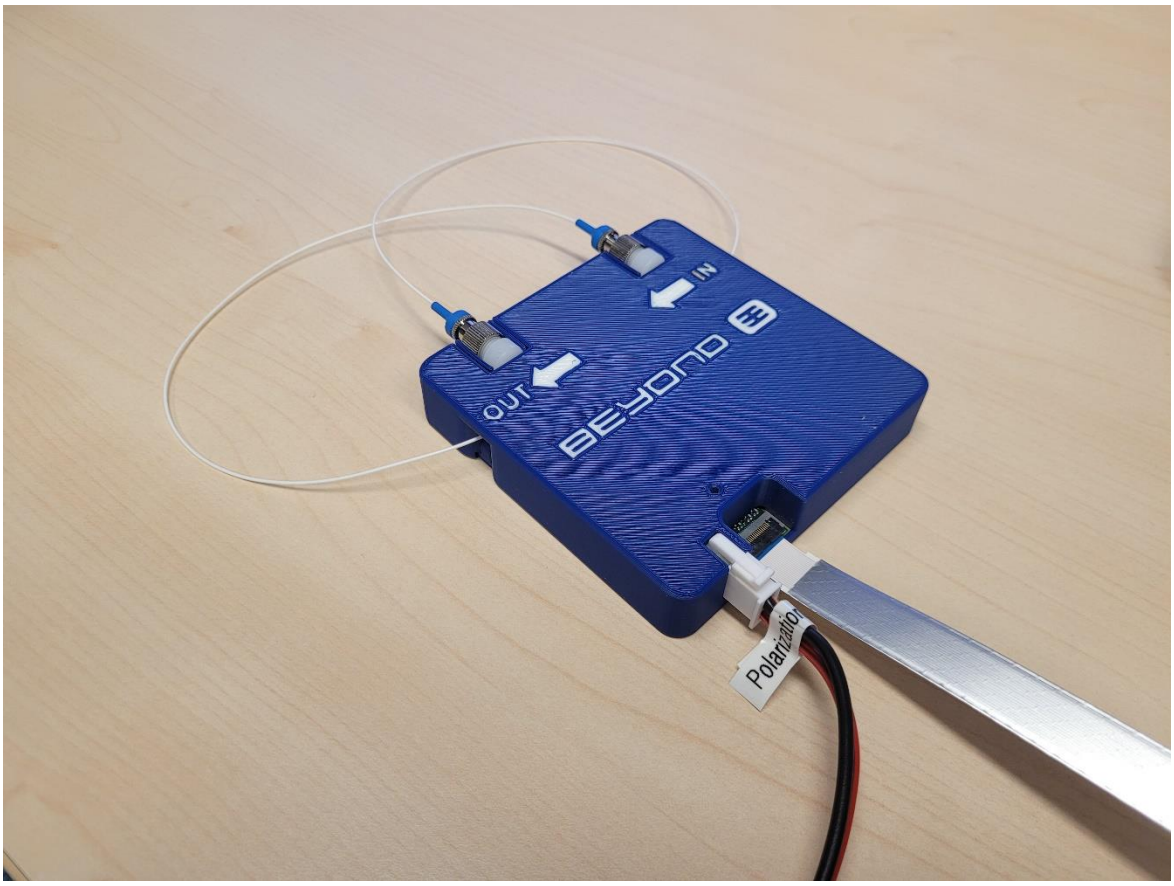
Run PC_control_application (PC_control_application.py) with following files in the same folder:

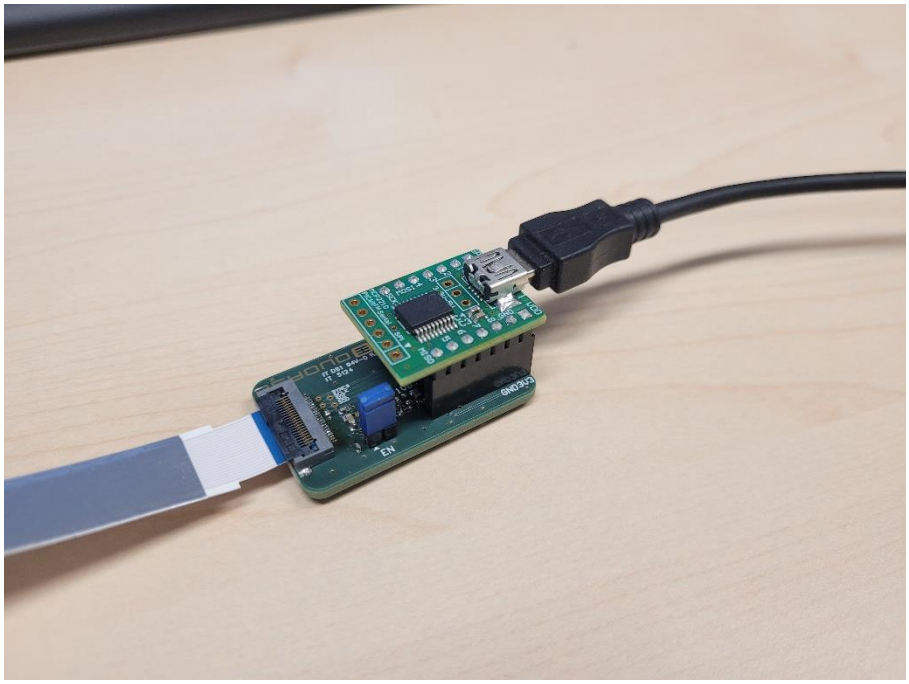
- functions.py
- MCP2210CLI.exe
- MCP2210DLL-UM.dll

5 Connecting MCP2210 to polarization controller

Use 12V supply with 1.5A supply current (max startup current of 1.5A). Operating current at steady state is ~200mA.

Only use provided power supply cable.





6 Use of application

6.1 Python script

This is the code that needs to be run in order to set desired DAC settings on the polarization controller driver PCB (BYSP296), which in turn set polarization controller (PC) channel voltage and consequently polarization angle.

We can achieve voltage range of 0V to 130V on polarization controller channels by setting »PC_channel_1/2/3/4« variables to desired voltage (number in volts [V]).

Temperature can be set by setting »temperature« variable (number in degrees [°C]) from 10°C to 75°C. Note that cooling is not supported (set temperature should be 15°C above max ambient temperature to have stabilization within 1°C accuracy).

```
##USER defined run parameters
PC_channel_1 = 60 #from 0 to 130V
PC_channel_2 = 60
PC_channel_3 = 60
PC_channel_4 = 60
temperature = 50 #from 10 to 75degC (cooling is not supported)

#END USER INPUT

import os
from funs import *      # functions file

print("configure DAC6, DAC3-DAC0")
os.system("mcp2210cli -spitxfer=28,4f -bd=100000, -cs=gp4 -md=1")  # configure
DAC6, DAC3-DAC0

voltage("DAC0", PC_channel_1)
voltage("DAC1", PC_channel_2)
voltage("DAC2", PC_channel_3)
voltage("DAC3", PC_channel_4)

temp_set(temperature)
```

6.2 Run command from terminal

For direct control in terminal execute command as shown below.
In the figure below is an example of running the code above where setting the temperature to 50°C translates to thermistor voltage around 1.34V.

```
Ukazni poziv
        >TxData: 28,4F
        >RxData: 44,3A
mcp2210cli -spitxfer=81,37 -bd=100000, -cs=gp4 -md=1

SPI transfer 2 bytes using as chip select pins: GP4
        >TxData: 81,37
        >RxData: 44,3A
mcp2210cli -spitxfer=91,37 -bd=100000, -cs=gp4 -md=1

SPI transfer 2 bytes using as chip select pins: GP4
        >TxData: 91,37
        >RxData: 44,3A
mcp2210cli -spitxfer=a1,37 -bd=100000, -cs=gp4 -md=1

SPI transfer 2 bytes using as chip select pins: GP4
        >TxData: A1,37
        >RxData: 44,3A
mcp2210cli -spitxfer=b1,37 -bd=100000, -cs=gp4 -md=1

SPI transfer 2 bytes using as chip select pins: GP4
        >TxData: B1,37
        >RxData: 44,3A
Thermistor voltage: ~1.34V
mcp2210cli -spitxfer=E3,d3 -bd=100000, -cs=gp4 -md=1

SPI transfer 2 bytes using as chip select pins: GP4
        >TxData: E3,D3
        >RxData: 44,3A

C:\Program Files (x86)\Microchip\MCP2210CLI>
```

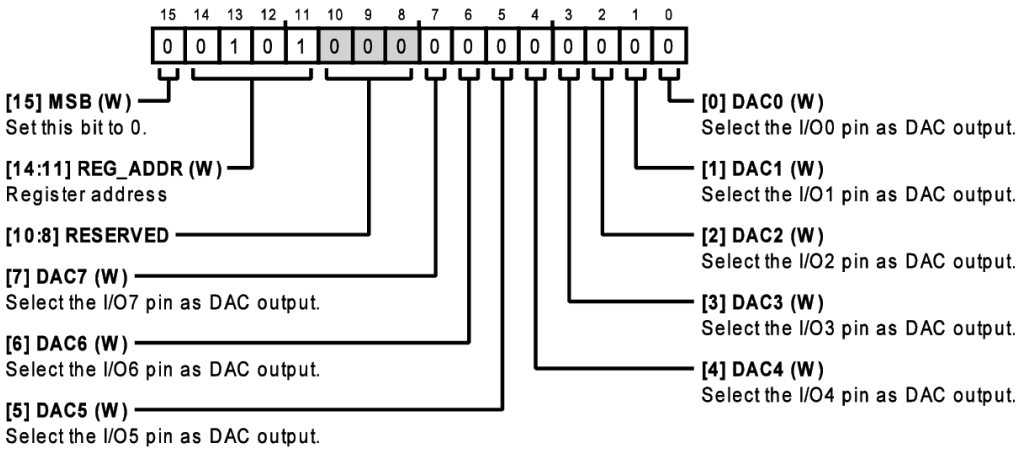
System is initialized with command:

```
mcp2210cli -spitxfer=28,4f -bd=100000, -cs=gp4 -md=1
```

where “28” is a fixed register value address and reserved bits and “4f” represents selected I/Ox output as DAC. “-bd” sets the baud rate, “-cs” sets which GPIO output of MCP2210 will work as a chip select and “-md” set the SPI mode. For more information on command line switches type

```
mcp2210cli -help
```

With bits “4f” DAC0, DAC1, DAC2, DAC3 and DAC6 are selected as DAC outputs as shown in the figure below.



To set desired CHx channel to desired voltage of Polarization controller send command

```
mcp2210cli -spitxfer={dac_num}{dac_code1},{dac_code2} -bd=100000, -cs=gp4 -md=1
```

where 4 bit wide dac_num’s first bit must be set to ‘1’ and the other 3 represent which DAC to send the data to as shown in the figure below.

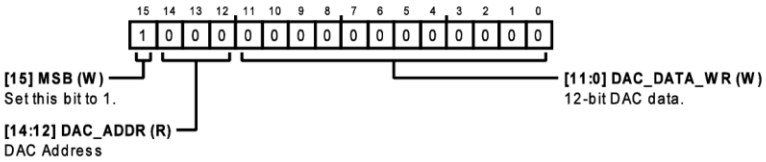


Table 29. Bit Descriptions for DAC_WR

Bits	Bit Name	Description	Default Value
15	MSB	Set this bit to 1.	0x1
[14:12]	DAC_ADDR	DAC Address. 000: DAC0. 001: DAC1. 010: DAC2. 011: DAC3. 100: DAC4. 101: DAC5. 110: DAC6. 111: DAC7.	0x0
[11:0]	DAC_DATA_WR	12-bit DAC data.	0x0

12-bit DAC data are represented with 4-bit Dac_code1 and 8-bit Dac_code2 eg. if we want to set the one CHx high voltage output to 60V (say CH1, which is connected to DAC1) we calculate the DAC code by the following equation

$$dac = V_{CH_1} * \frac{4095}{5.088V * 25.877} = 1866.14$$

Where V_{CH_1} is a desired output voltage, 4095 maximum code for 12-bit DAC, analog supply is 5.088V and OpAmp gain is set to 25.877. Value of $dac = 1866.14$ is rounded to 1866 and converted to hexadecimal value 0x74a.

Values to set CH1 to 60V are displayed below:

- Dac_num = '9'
- Dac_code1 = '7'
- Dac_code2 = "4a"

Example of a command ("`mcp2210cli -spitxfer={dac_num}{dac_code1},{dac_code2} -bd=100000, -cs=gp4 -md=1`") to send to ADC/DAC chip:

- `mcp2210cli -spitxfer=97,4a -bd=100000, -cs=gp4 -md=1`