PromLegion Ltd.

DX5100 Library for LabView

Quick Start

Moscow, 2016

Version 1.01

Contents

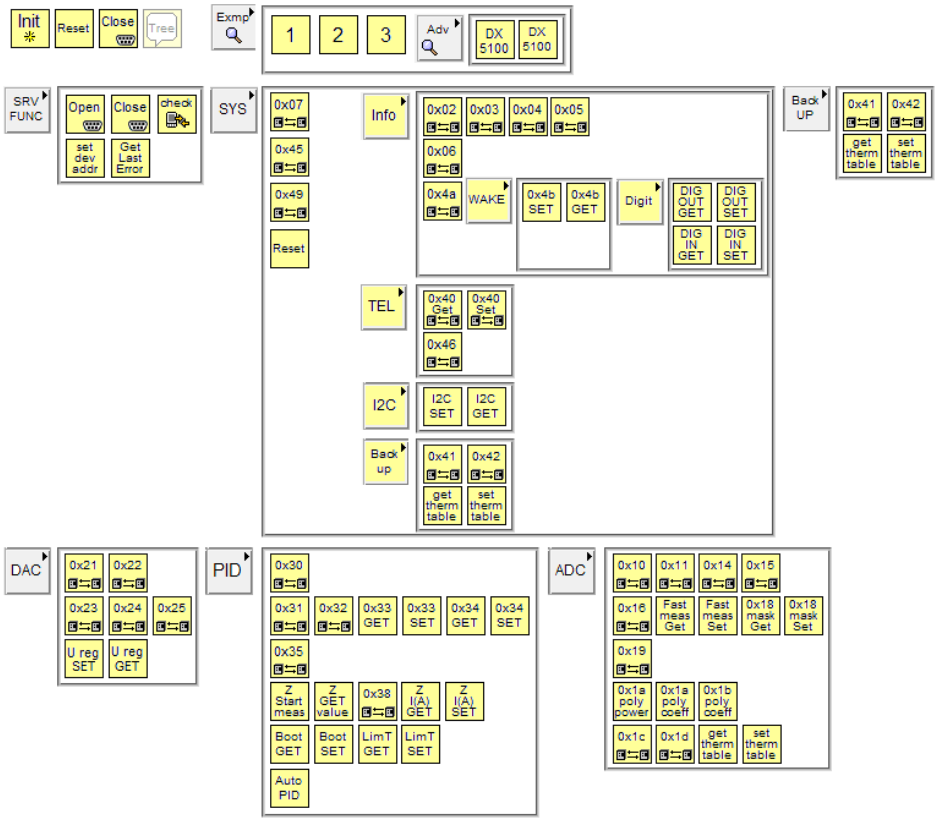
INTRODUCTION

The library is intended for using in your projects on LabView of the thermoelectric coolers controller PX5100.

The full documentation on the controller can be found here:

http://www.promln.com/products/devices/controllers/

After installing the library, **PX5100** will appear in the toolbar **Instrument Drivers.**



All the commands of the controller are formed as separate VI, their names corresponding to the following document:

http://www.promln.com/downloads/manuals/PX5100 System of Commands V3.10.pdf

EXAMPLE 1. CONNECTION WITH CONTROLLER.

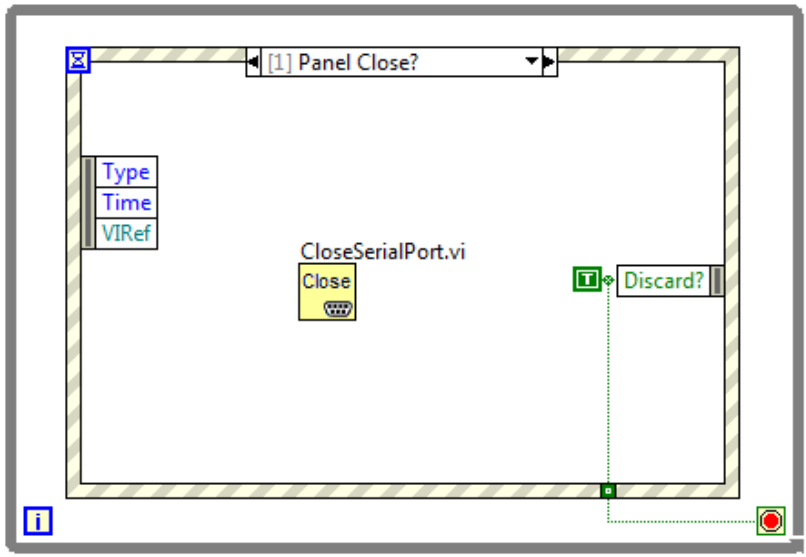
Create a new blank VI;

Set a type of the window Dialog (File->VI Properties->Category->Window Apperance);

Add While Loop and Event Structure;

Add an exit event **Event Source**: <This VI>, Event: Panel Close?

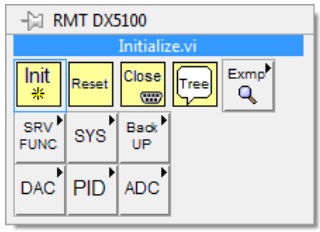
Add CloseSerialPort.vi to make a corresponding port free when you exit.



Add a button “Open” on the front panel, set its property **Operation Switch until released**.

Add **event** for the button Event Source: Open, Event: Value Change

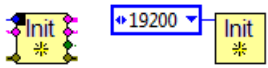
Put **Init** on this event from the library:



Input Parameters Initialize.vi

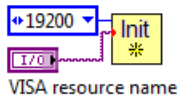
Baud

Enter the context menu and create a constant by the mouse right click on the input terminal **baud**. Select a required baud rate (19200 by default):

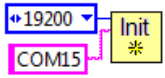


PortName

If the components **visa** are installed, you can use a component that will help you to choose the correct runtime port as the input PortName parameter.

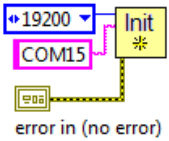


If for some reason you do not want to use it, you can just send a string with the name of the port:



Error in (no error)

The third input parameter is the cluster **Error In**. Create a constant for it:

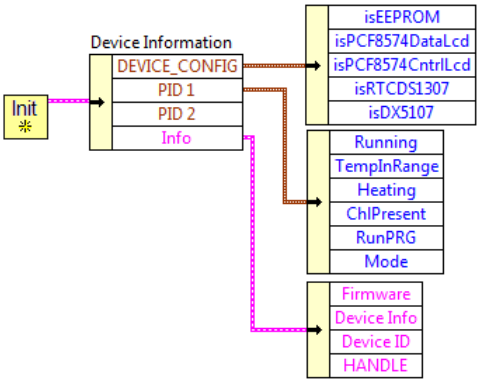


Output Parameters Initialize.vi

Device Information

For simplicity, we do not use the terminal “Device Information”!

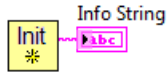
“Device Information” returns a cluster of 4 sub-clusters containing information about the configuration of the connected controller:



This is the result of the commands 0x03, 0x04, 0x05, 0x4a.

The detailed description of the controller commands system is here: http://www.promln.com/downloads/manuals/PX5100_Commands_V33.pdf

Info String

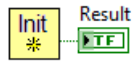


It is a concatenation of 3 strings separated by a space: *Firmware + Device Info + DeviceID.*

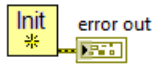
DX5100.336	#271 15.01.2013	0102

In case of a failure the string will contain a description of the error.

Result



Error out

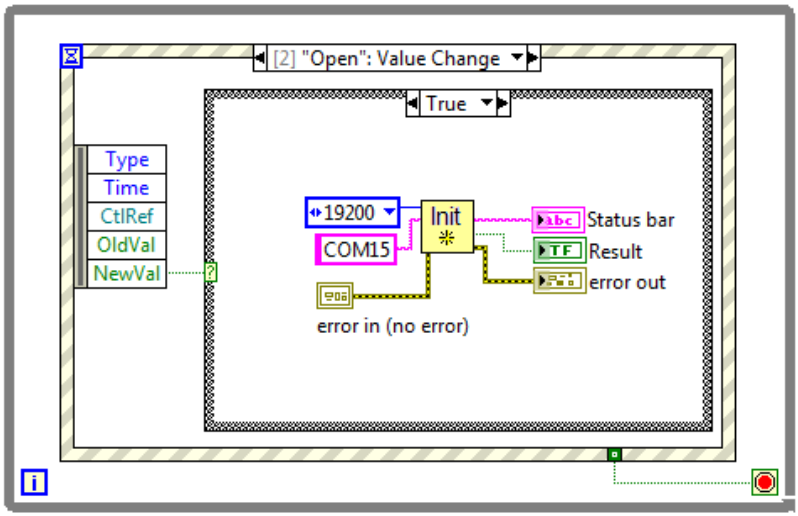


Your Result

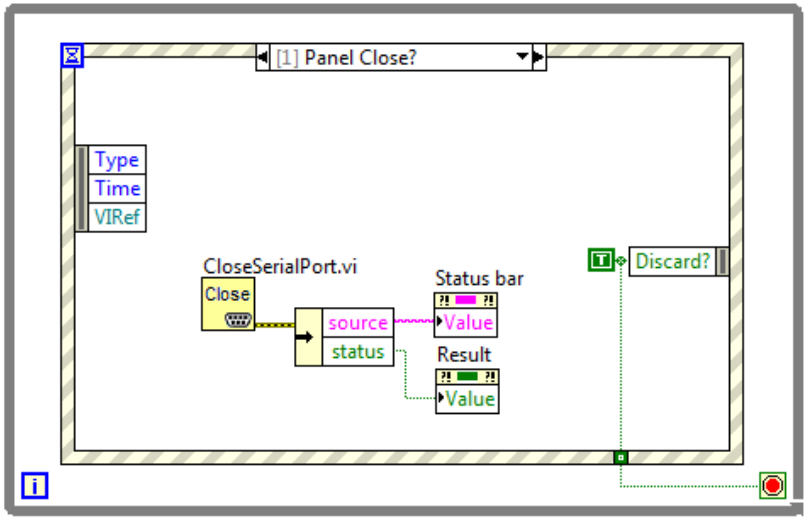
So, our VI has one button “Open”, one Boolean indicator “Result” and one indicator of the type String. The indicator “error out” is to be hidden (context menu -> Hide Indicator).

For convenience, we make the indicator **Info String** in the form of the status line.

“Open”:Value Change

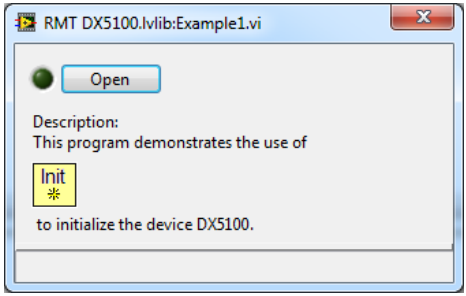


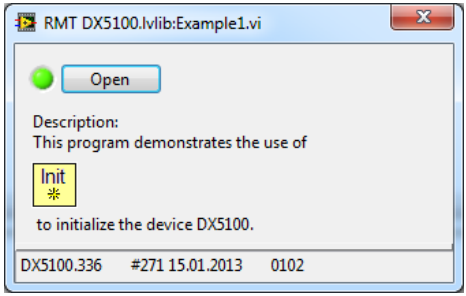
Panel Close?



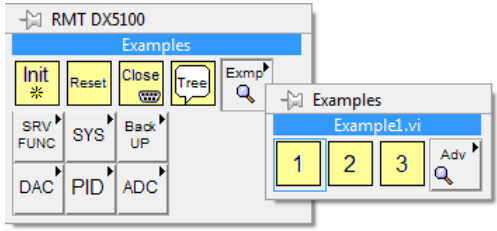
Here we have added a set of values **Status bar** and **Result**, so that they correspond to the real state of COM-port.

Everything should work, let us check:





This example can be found in the library:

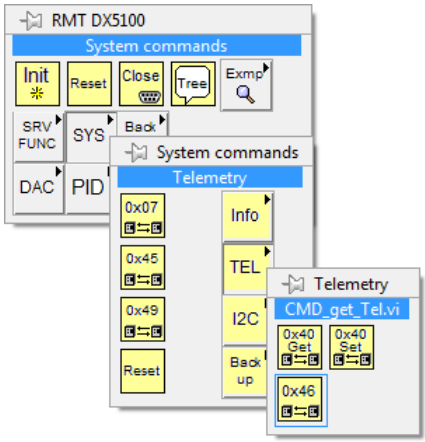


EXAMPLE 2. HOW TO RECEIVE TELEMETRY.

Let us consider how to get telemetry data from the device.

We continue using the previous example. To keep things simple, we do not consider options with queuing in LabView, but use the event **Timeout**.

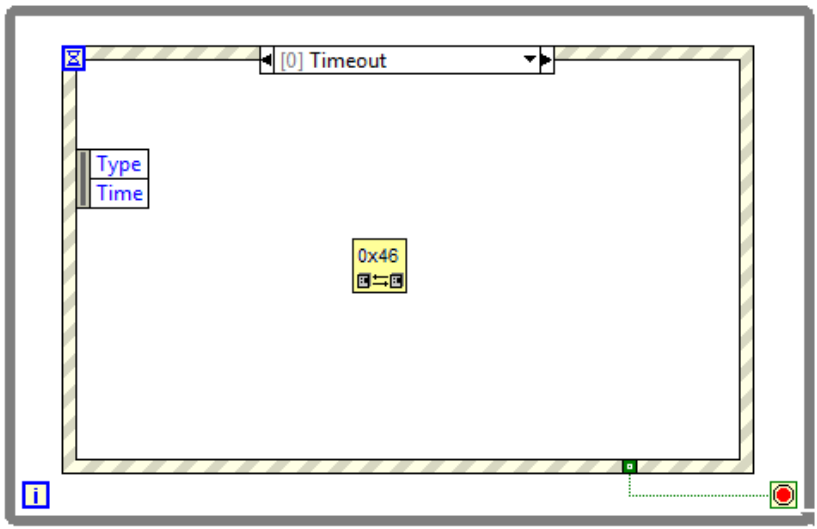
We select an event in Timeout Event Structure and add the command 0x46 from the library.



By default the telemetry mask installed is 0x337F, which corresponds to the full telemetry of two measurement channels. The command 0x40 is used to change the mask.

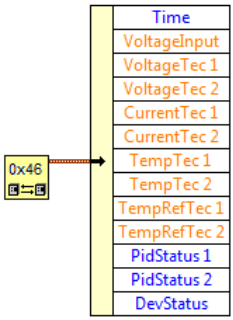
The detailed description of the commands is given here:

http://www.promln.com/downloads/manuals/PX5100_Manual_V331.pdf (Appendix 2)



Output Parameters of Command 0x46 CMD_get_Tel.vi

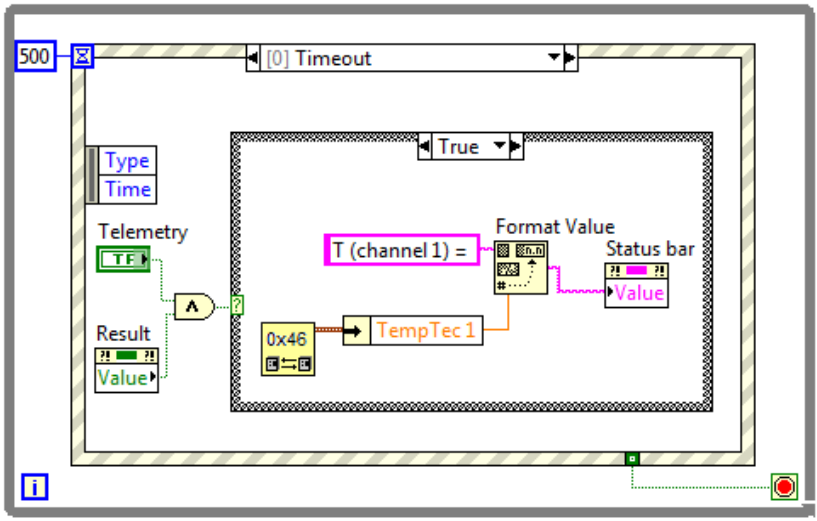
In our example, we use only the temperature of the 1st channel, but all the parameters look like:

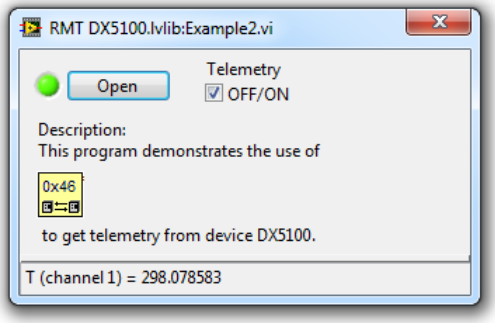


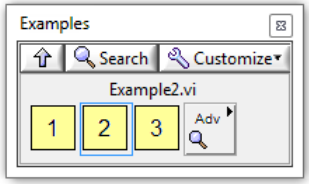
To indicate the necessity for telemetry we add checkbox Telemetry to our VI and set the timeout 500 ms.

So, if PX5100 is active (Result = true) and Telemetry.Checked = true, we request a telemetry.

The event **Timeout** will look so:







EXAMPLE 3. REGULATION

Let us go on. First, a few words about the regulation. The PX5100 controller can make regulation for one channel or two (if any).

The detailed description is here.

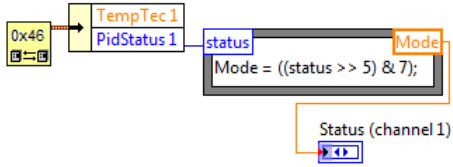
Possible Modes of Regulation

1. Regulation is disabled **“IDLE”**;
2. The control is in the mode **“Program”**;
3. Temperature maintenance: relay control **“T-regulation”**;
4. Temperature maintenance: PID “Temperature maintenance (PID)”;
5. Voltage maintenance **“Constant Voltage”**.

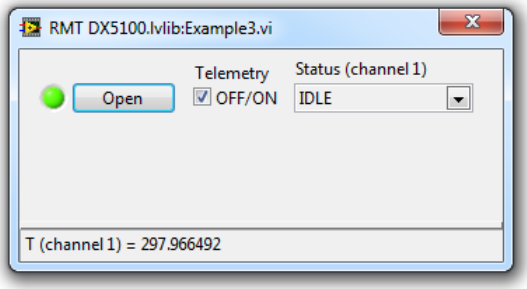
Controller Current State

Here we need to expand the processing of the data obtained in the previous example. To determine the current status of the controller we use another output parameter – the status of the 1st channel “PidStatus1”:

This parameter is of the type U8 and most significant 4 bits specify the current regulation mode To display the current mode we add Enum Indicator with possible modes and call it "**Status (channel 1)".** So, the channel status processing will be as follows:

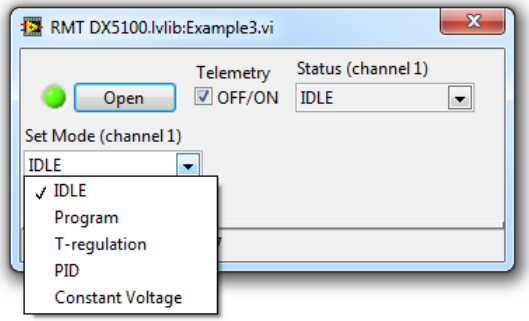


Let us run our example:

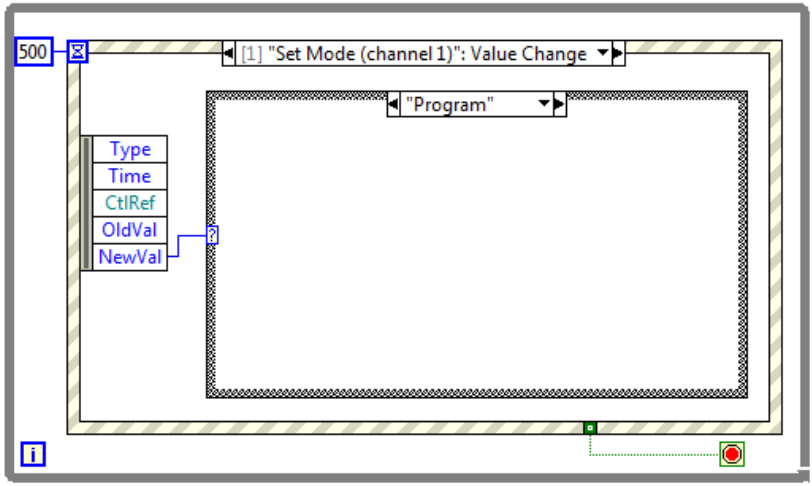


Regulation Mode Choice

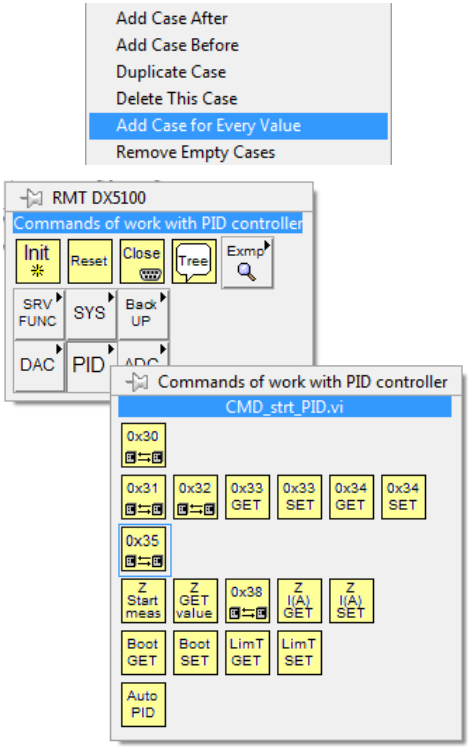
Now it is necessary to provide a choice of a regulation mode. We will create Enum Control on the basis of the indicator **"Status (channel 1)"** (PopupMenu->Create->Control), let us name it **"Set Mode (channel 1)".**



We add the event Event Source**: Set Mode (channel 1)**, Event: Value Change to the Event-structure. We add a Case-structure on this event:

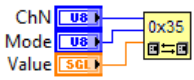


We add all possible variants to our Case-structure:



It is possible to select a proper mode of regulation

by the command 0x35:



The first two parameters should not cause problems, but the third parameter can have different values depending on the selected mode:

1. **IDLE** – parameter **Value** is ignored;
2. **Program** – parameter **Value** is perceived as a program number and must be an integer from 0 to 15;
3. **T-Regulation** – parameter **Value** is perceived as a setting to maintain the temperature [K], and must be within acceptable limits [K] (see the command 0x3d). The factory settings: Tmin = 203K, Tmax = 403 K.

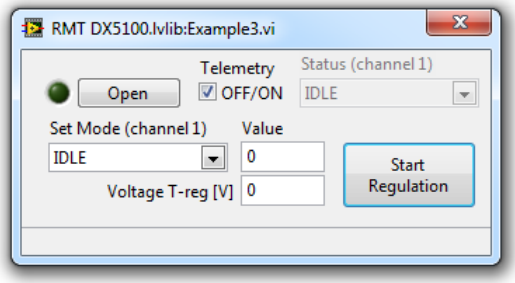
U reg SET

For this mode, we also need to set the regulation voltage [V] (the command 0x26), so we add another Numeric Control and call it **"Voltage T-reg [V]"** ;

LimT GET

4. **PID** – parameter **Value** is a temperature setting [K] for PID-regulation;
5. **Constant Voltage** – parameter **Value** is the voltage setpoint in this case.

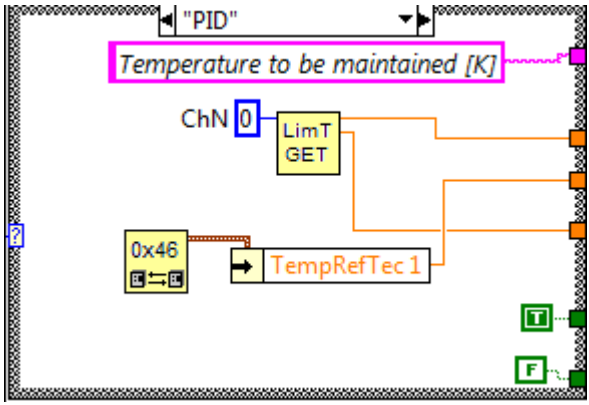
So, we need the following components:



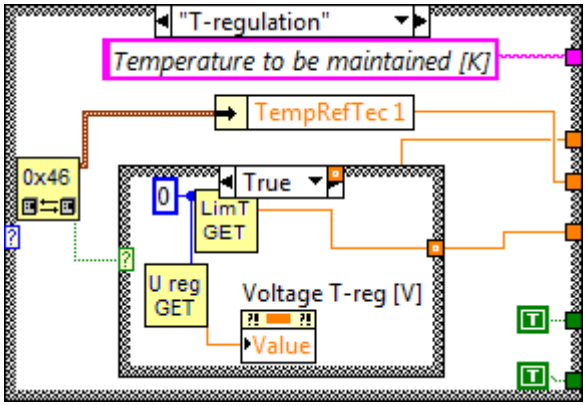
To enter and display data properly, we have to process events and set the properties of the window components of our form. It does not relate to the operation of the controller directly, so for the sake of simplicity, these steps are omitted.

Let us describe the main features of operating the controller:

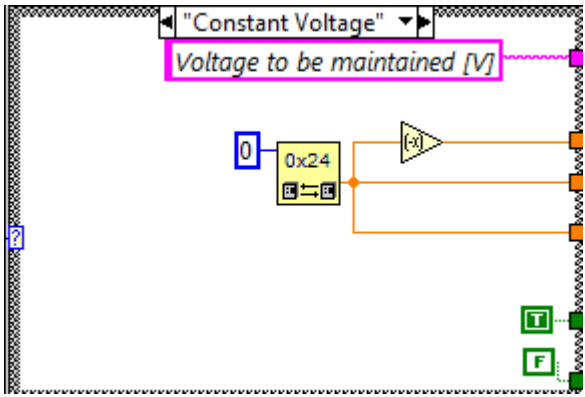
- When choosing a new mode (Enum Control **“Set Mode (channel 1)”**), if telemetry is on, turn it off;
- PID



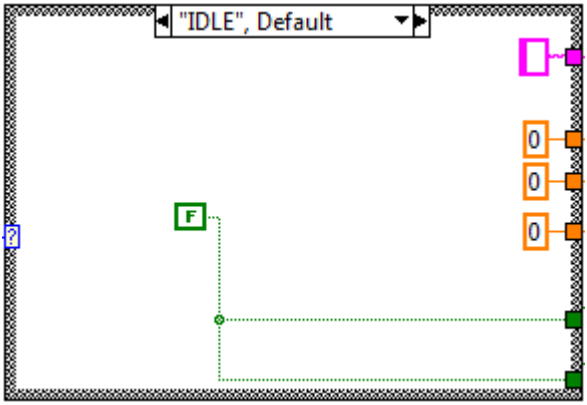
- T-Regulation



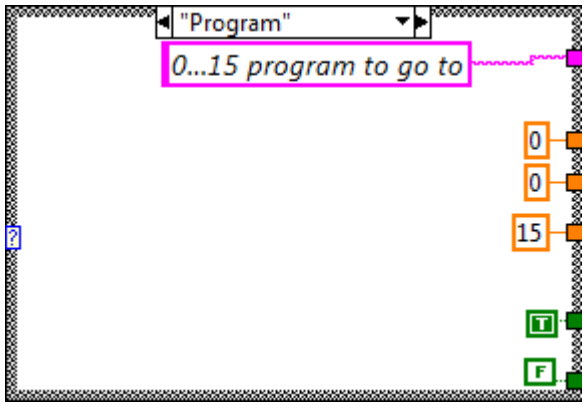
- Constant Voltage



- IDLE



- Program



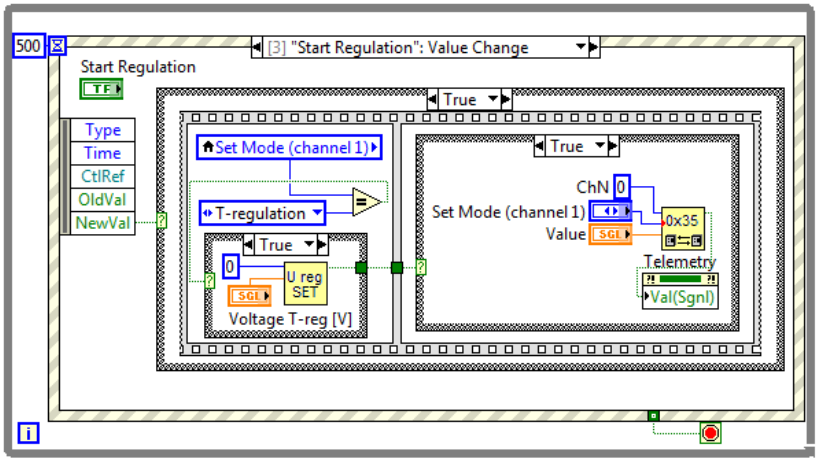
Start Regulation

- if the mode "**T-Regulation"** is chosen , then, before operating it, set the regulation voltage:

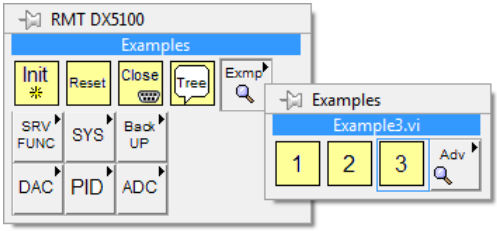
otherwise, you can execute the command *0x35* straight away:

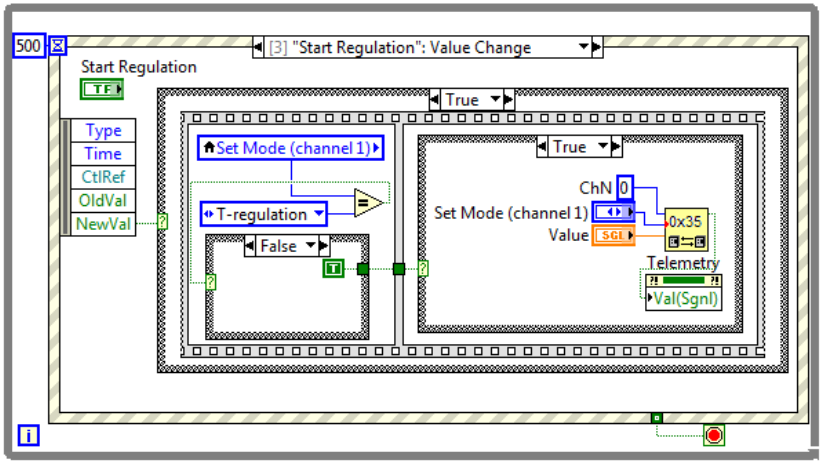
After running the command,

Turn telemetry on:

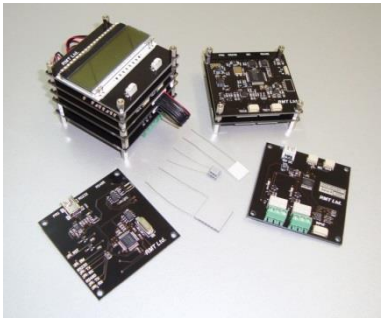

500
[3] "Start Regulation": Value Change
Start Regulation
TF
Type
Time
CtlRef
OldVal
NewVal
True
Set Mode (channel 1)
T-regulation
True
0
U reg SET
Voltage T-reg [V]
True
ChN 0
Set Mode (channel 1)
Value
0x35
Telemetry
Val(Sgnl)
i


Telemetry
Val(Sgnl)


RMT DX5100
Examples
Init
Reset
Close
Tree
Exmp
SRV FUNC
SYS
Back UP
DAC
PID
ADC
Examples
Example3.vi
1
2
3
Adv



This example is here:



PromLegion Ltd.

46 Warshawskoe shosee
Moscow 115230 Russia
e-mail: info@promln.com
phone: +7-499-678-3231
fax: +7-499-678-3258
website: www.promln.ru

Overseas Sales representative

TEC Microsystems GmbH
Schwarzschildstrasse 8
Berlin 12489, Germany
phone: +49-(0)30-6789-3314
fax: +49-(0)30-6789-3315
e-mail: info@tec-microsystems.com
website:www.tec-microsystems.de