

TECHNICAL MANUAL

BOP-BIT LABVIEW, VXI *plug&play* INSTRUMENT DRIVER

KEPCO INC.
An ISO 9001 Company.

MODEL

BOP-BIT
LABVIEW, VXI *plug&play*
INSTRUMENT DRIVER

ORDER NO.

REV. NO.

IMPORTANT NOTES:

- 1) This manual is valid for the following Model and associated serial numbers:

| MODEL | SERIAL NO. | REV. NO. |
|-------|------------|----------|
|-------|------------|----------|

- 2) A Change Page may be included at the end of the manual. All applicable changes and revision number changes are documented with reference to the equipment serial numbers. Before using this Instruction Manual, check your equipment serial number to identify your model. If in doubt, contact your nearest Kepco Representative, or the Kepco Documentation Office in New York, (718) 461-7000, requesting the correct revision for your particular model and serial number.
- 3) The contents of this manual are protected by copyright. Reproduction of any part can be made only with the specific written permission of Kepco, Inc.

Data subject to change without notice.

©2016, KEPCO, INC
P/N 243-1365-r1



KEPCO, INC. • 131-38 SANFORD AVENUE • FLUSHING, NY. 11355 U.S.A. • TEL (718) 461-7000 • FAX (718) 767-1102
email: hq@kepcopower.com • World Wide Web: <http://www.kepcopower.com>

TABLE OF CONTENTS

| SECTION | PAGE |
|---|--|
| SECTION 1 - INTRODUCTION | |
| 1.1 | Scope 1-1 |
| 1.2 | General Description..... 1-1 |
| 1.2.1 | Drivers 1-2 |
| 1.2.2 | Programming 1-2 |
| 1.3 | Driver Overview 1-3 |
| 1.3.1 | Initialization 1-3 |
| 1.3.2 | Output Control 1-3 |
| 1.3.3 | Measurement..... 1-3 |
| 1.3.4 | LIST (User-Programmed Sequences) 1-3 |
| 1.3.5 | Trigger 1-3 |
| SECTION 2 - LABVIEW G DRIVER | |
| 2.1 | Introduction..... 3-1 |
| 2.2 | LabView G Instrument Driver 3-1 |
| 2.3 | LabView G Instrument Driver Functions..... 3-1 |
| 2.4 | Using the Driver Functions 3-3 |
| 2.4.1 | KepecoDCPwr Configure Voltage Level.vi 3-3 |
| 2.4.1.1 | Connector Panel 3-3 |
| 2.4.1.2 | Controls and Indicators 3-3 |
| 2.4.1.3 | Block Diagram Description..... 3-4 |
| 2.4.2 | KepecoDCPwr Measure [MSR].vi..... 3-5 |
| 2.5 | Example of Setting the Output 3-6 |
| 2.5.1 | Initialization Function 3-7 |
| 2.5.2 | Set the OUtput ON or OFF 3-10 |
| 2.5.3 | Get Measurement and Status..... 3-10 |
| 2.5.4 | Close Connection 3-10 |
| 2.6 | Voltage Application example. 3-11 |
| 2.7 | Interactive demonstration program..... 3-12 |
| 2.7.1 | KepecoDCPwr Interactive Example.vi..... 3-12 |
| 2.7.2 | Software timed ramp example 3-14 |
| 2.7.3 | Software Timed Ramp Execution 3-15 |
| 2.7.4 | Function Generator..... 3-16 |
| SECTION 3 - VXI plug&play Driver | |
| 3.2 | Demonstration Program Using the VXI plug&play Driver 2-1 |
| 3.2.1 | Main Panel Controls and Indicators..... 2-2 |
| 3.3 | Software Engineer reference..... 2-4 |
| 3.3.1 | VISA Instrument Driver Functions 2-4 |
| 3.3.2 | Program Practices 2-8 |
| 3.4 | Examples..... 2-8 |

LIST OF FIGURES

| FIGURE | TITLE | PAGE |
|--------|--|------|
| 2-1 | KepecoDCPwr Configure Voltage Level.vi Connector Panel..... | 3-3 |
| 2-2 | KepecoDCPwr Configure Voltage Level.vi Block Diagram | 3-5 |
| 2-3 | KepecoDCPwr Configure Voltage Level.vi Error Block Diagram | 3-5 |
| 2-4 | KepecoDCPwr Measure [MSR].vi Control Panel | 3-5 |
| 2-5 | KepecoDCPwr Measure [MSR].vi Block Diagram..... | 3-6 |
| 2-6 | Setting the Output, Overall Block Diagram | 3-7 |
| 2-7 | Serial Initialization Block Diagram..... | 3-9 |
| 2-8 | LAN Initialization Block Diagram | 3-9 |
| 2-9 | Voltage Application Example | 3-11 |
| 2-10 | Front Panel Window..... | 3-13 |
| 2-11 | Software Ramp Panel, Current Ramp Example..... | 3-14 |
| 2-12 | Ramp Function Panel, Current Ramp | 3-15 |
| 2-13 | Function Generator Panel..... | 3-17 |
| 2-14 | Configure User Sequence.vi Block Diagram..... | 3-18 |
| 2-15 | Ramp Function, Block Diagram | 3-19 |
| 2-16 | Function Generator, List Functionality, Block Diagram..... | 3-21 |
| 3-1 | GPIB VISA Driver Start-up..... | 2-1 |
| 3-2 | Main Panel | 2-2 |
| 3-3 | Transient Pulse Window | 2-3 |
| 3-4 | Trigger Window | 2-3 |
| 3-5 | Calibration Window | 2-4 |

LIST OF TABLES

| TABLE | TITLE | PAGE |
|-------|--|------|
| 1-1 | VISA Resource String Corresponding to Interface | 1-2 |
| 2-1 | BOP-BIT LabView G Driver Functions | 3-1 |
| 2-2 | KepecoDCPwr Configure Voltage Level.vi Input/Output Descriptions | 3-4 |
| 2-3 | Error (or Explain Warning) Codes | 3-4 |
| 2-4 | KepecoDCPwr Measure [MSR].vi Input/Output Descriptions | 3-6 |
| 3-1 | BOP VISA Driver Functions | 2-4 |

SECTION 1 - INTRODUCTION

1.1 SCOPE

This manual provides information for a test system engineer to use Kepco's Labview and VXI *plug&play* drivers with a BIT 4882, BIT 4886 or BIT 802E Interface Card installed in a Kepco BOP 100W, 200W or 400W power supply. The suffix -4882, -4886 or -802E after a BOP model number indicates the BIT Card was installed at the factory. The Labview and VXI plug&play drivers may also be used with a BOP 1KW (MG or GL) power supply. For convenience, all applicable power supplies mentioned above are hereafter referred to as BOP-BIT power supplies in this manual.

The SCPI commands used to control the BOP-BIT power supply are provided in the Operator's manuals for BIT 4886 and BIT 802E, respectively. These manuals can be downloaded from Kepco's web site at

<http://www.kepcopower.com/support/opmanls.htm#bit>

1.2 GENERAL DESCRIPTION

BOP-BIT power supplies can support asynchronous RS-232, GPIB, and LAN communication interfaces. These interfaces, when being controlling through a C program such as CVI, Visual C or Labview use different methods to send the SCPI strings to the instruments. The Virtual Instrument Software Architecture (VISA) is a standardized method of communication with instruments over the various buses using identical calls. The LabView G and VXI plug&play drivers use these VISA calls. Included in each driver are:

The BOP-BIT is a power supply class of instrument. The BOP-BIT supports all power supply class functionality including measurement and trigger. The BOP-BIT has additional functionality beyond the class requirements, including a 1002 step list and state storage. The BOP-BIT power supply can be digitally programmed using SCPI commands and queries sent from a computer via any of the interfaces.

BOP power supplies with BIT 4886 or BIT 4882 installed, as well as BOP 1KW (MG or GL) power supplies include an IEEE 488.2 GPIB and RS-232 interfaces. BOP power supplies with BIT 802E installed include a LAN interface use to control BOP operation via web pages.

Each driver includes the following:

- Source code C for all the functions detailed in table 1-1 along with a file of definitions.
- This manual in pdf form.
- A sample application that can operate and calibrate the BOP-BIT power supply
- The CVI version 14 or LabView 2010 runtime environment.
- A Generic VISA version 14 from National Instruments.

1.2.1 DRIVERS

Two instrument drivers, Labview G and VXI *plug&play*, are available which allow remote operation via virtual front panels. These drivers can be downloaded from the Kepco web site at <http://www.kepcopower.com/drivers/drivers-dl3.htm>

- LabView G driver - written in native LabView code using the VISA write and read functions. (see Section 2).
- VXI *plug&play* driver - written in C using VISA write and read functions. It is compliant with the VXI *plug&play* specification and can be used in an ANSI compatible C program such as LabWindows /CVI. It is also portable to Linux and Apple operating systems, however verification of these environments has not been completed. (see Section 3).

The drivers do not communicate directly to the BOP-BIT. They use the operating system or vendor specific interface drivers to handle the actual communication via the serial, Ethernet or GPIB ports. Vendors such as National instruments, Agilent technologies and Keithley have created a VISA (Virtual Instrument System Architecture) library. The VISA library uses consistent commands to open a connection, write and read data, read the interrupt status and perform standard functions such as lock, unlock and device clear.

The VISA libraries “open” command creates the handle for all other functions. The open command requires a resource string that provides the physical address of the unit and also specifies the type of interface that will be used. Table 1-1 provides the resource strings for the four possible interfaces to the BOP-BIT.

These drivers require a helper application (visa32.dll) to be installed on the computer being used. VISA uses resource strings (see Table 1-1) to address the unit. The VXI *plug&play* driver requires that all calls to an instrument be made through the VISA library. The LabView G driver also uses VISA calls

TABLE 1-1. VISA RESOURCE STRING CORRESPONDING TO INTERFACE

| BIT CARD | INTERFACE | VISA RESOURCE STRING | COMMENT |
|----------|--------------|-----------------------------------|---|
| BIT 4886 | GPIB | GPIB::xx::INSTR | The GPIB address replaces xx. |
| BIT 4886 | SERIAL | ASRLy::INSTR | The com port number replaces y. |
| BIT 802E | LAN-SCPI-RAW | TCIP::192.168.0.100::5025::SOCKET | This is the fastest LAN interface, similar to the serial port with automatic XON XOFF protocol support. |

1.2.2 PROGRAMMING

SCPI and IEEE 488 common commands/queries are the building blocks used to control the BOP-BIT power supply. These sections are provided to allow a user to write their own program to control the BOP-BIT power supply or to use the various interactive tools provided by Measurement Computing, National Instruments, Agilent Technologies and Microsoft to send strings to a device over the RS 232, GPIB or LAN interfaces. SCPI commands and queries are supported by all three interfaces. The applicable Operator's Manual (see PAR. 1.1) provides the following:

- Description of SCPI Syntax
- IEEE 488 Common commands supported
- Listing of SCPI commands supported

1.3 DRIVER OVERVIEW

The two drivers for BOP-BIT have common functional groups. Each group contains similar functions which work together. The two different environments, LabView G and VXI *plug&play*, have common functionality.

The BOP-BIT power supply, like most instrument power supplies, has four subsystems: output, trigger, status and measurement. In addition to these subsystems, the BOP-BIT has a storage system and a list system.

1.3.1 INITIALIZATION

When an instrument is first accessed, it is desirable to have it start at a standard state and to verify that the instrument is the correct one. These functions utilize standard commands which are common to all power supply class instruments such as *IDN? *CLS and *RST. In the various drivers, this functional group also includes an open and close functionality.

1.3.2 OUTPUT CONTROL

The power supply class instrument has specific requirements to support voltage and current settings, output enable and protection. These functions are common to all power supplies in the power supply class.

1.3.3 MEASUREMENT

The Power supply has the ability to measure the voltage and current at the output and display this information on the front panel. The measurement system provides the ability to read this back from the unit.

1.3.4 LIST (USER-PROGRAMMED SEQUENCES)

Using the LIST functions, up to 1002 locations are available for programming the BOP-BIT output. These locations enable the user to program the output using multiple command sequences which may be initiated by a single command. The repeatable user-determined sequences are stored in volatile memory and are retained until reprogrammed, the power supply is turned off, a calibration is performed, a `LIST:CLear` command is received, or limit model settings are changed.

Each location defines values for the active channel (either output voltage or output current), and a dwell time duration (between 0.00025 and 2 seconds, inclusive) for the programmed settings. By programming the output to change in small increments, complex outputs can be generated.

1.3.5 TRIGGER

The BOP-BIT has the ability to utilize a software trigger. Units with BIT 802E installed have an additional software trigger capability accessed via Port 5044. When a trigger event is properly enabled and received, the output voltage and current are set to the preset trigger levels.

If software trigger is selected, trigger events are activated by *TRG or a GPIB <GET> command.

SECTION 2 - LABVIEW G DRIVER

2.1 INTRODUCTION

The LabView G driver can be used for remote programming over all installed ports of the BOP-BIT (Serial, GPIB and LAN ports - see Table 1-1). It is designed to operate in LabView 2010 environments. It can be opened in all Labview environments up to Labview 2014.

The Version 2010 LabView driver is available in two formats: a standalone version that includes the LabView runtime program, and a version that requires that LabView be installed on the computer. The LabView G driver uses VISA calls to communicate with the power supply.

NOTE: The LabView G driver is modifiable by the user. Once any utility or function is modified, it is the user's responsibility to verify there are no unintended errors introduced.

During installation of the software modules, three subfolders are created under C:\program files\National instruments\LABVIEW 2010

- **instr.driv\KpDcpwr** has the Kepco.lib that contains all driver functions
- **examples\KpDcpwr** has simple examples to set voltage and read back voltage and current.
- **user.lib\KpDcpwr** has the programs that are available via buttons of the interactive example application which was also installed at a folder the customer can specify.

NOTE: This driver is not for use with a LabView Version below LabView 2010. Instead, download the older driver available at <http://www.kepcopower.com/drivers/drivers-dl3a.htm>. Users with a higher version than 2010, such as LabView 2014, must copy the three folders noted above (instr.driv\KpDcpwr, examples\KpDcpwr and user.lib\KpDcpwr) into the existing LABVIEW 2014 folder.

2.2 LABVIEW G INSTRUMENT DRIVER

The LabView G driver is a multipurpose driver. It is capable of operating all Kepco Power supplies using LabView Version 2010 including multiple output units such as MST. The LabView G driver does not report model-specific errors but, using case structures, passes the function and an appropriate model-specific response to the user's program.

2.3 LABVIEW G INSTRUMENT DRIVER FUNCTIONS

Kepco's BOP-BIT LabView G instrument driver provides programming support for Kepco's BOP Power Supply with a BIT 4886 or BIT 802E Card installed. It contains functions for opening, configuring, taking measurements from, testing, calibrating and closing the instrument. To successfully use this module, the instrument must be connected to either the LAN, GPIB or RS 232 port.

The driver supports all functions of the BOP-BIT except the instrument state commands of Store, Recall and Status processing. The driver is also supplied with some simple examples and an interactive demo program. Users that are modifying the driver are encouraged to copy parts of this program to maximize usage of tested functional sections. Table 2-1 lists the functions that are available.

TABLE 2-1. BOP-BIT LABVIEW G DRIVER FUNCTIONS

| Function Name | Purpose |
|---------------------------------------|---|
| KepecoDCPwr initialize with options | Makes Visa Connection. Creates Open handle Provides ability to issue Reset and parse IDN |
| KepecoDCPwr initialize | Makes Visa Connections, No Reset, IDN is parsed and Kepeco model is properly identified. |
| Kepeco Serial initialize with options | Makes Visa Connection. Creates Open handle Provides ability to issue Reset and parse IDN |
| KepecoDCPwr close | Closes the driver, freeing resources |
| Recognize Model | Sends *IDN? And verifies KEPCO string returned |
| KepecoDCPwr IDN parser | Parse the IDN string – determining model |
| KepecoDCPwr Revision Query | Uses IDN parser to determine firmware version |
| KepecoDCPwr Reset | Sends *RST to Power supply, set output off, volt=0, curr min, over voltage and over current to max values |
| KepecoDCPwr Reset with Defaults | Supplied for compatibility purposes uses above |
| KepecoDCPwr Self-test | Sends *TST? To unit and returns result- o= all okay. |
| Output Control functions | |
| KepecoDCPwr Configure Voltage Level | Set voltage output setting to a specific value |
| KepecoDCPwr Configure Current Level | Sets current output setting to a specific level |
| KepecoDCPwr Query Voltage Level | Queries unit and returns voltage level setting |
| KepecoDCPwr Query Current Level | Queries unit and returns Current level setting |
| KepecoDCPwr Configure Voltage Limit | Set virtual voltage setting to a specific value (BOP 1KW only) |
| KepecoDCPwr Configure Current Limit | Sets virtual current setting to a specific level (BOP 1KW only) |
| KepecoDCPwr Query Voltage Limit | Queries unit and returns limit model voltage level setting |
| KepecoDCPwr Query Current Limit | Queries unit and returns limit model Current setting |
| KepecoDCPwr Configure OVP | Set over voltage setting to a specific value |
| KepecoDCPwr Configure OCP | Sets Over current setting to a specific level |
| KepecoDCPwr Configure Output Enabled | Sets output on or off |
| KepecoDCPwr Query Output Enabled | Returns Output on(1) or off(0). |
| Measurement | |
| KepecoDCPwr Measure [MSR] | Returns the voltage read back 9 default or Current read back value from BOP-BIT. |
| List Functions | |
| KepecoDCPwr Configure Current Mode | Sets unit to List or Fixed mode of operation. Fixed operation is default |
| KepecoDCPwr Configure Voltage Mode | Sets unit to List or Fixed mode of operation. Fixed operation is default |

TABLE 2-1. BOP-BIT LABVIEW G DRIVER FUNCTIONS (CONTINUED)

| Function Name | Purpose |
|---|--|
| KepecoDCPwr Configure User Sequence | Initializes list from an array of either voltage and dwell or Current and dwell to execute the list a number of times. |
| KepecoDCPwr Clear All Lists | Clears the list if not executing. |
| KepecoDCPwr Configure List | Loads a specific list from an array of numbers. |
| KepecoDCPwr Configure List Count | Initializes the list count variable |
| Trigger Functions | |
| KepecoDCPwr Configure Triggered Current Level | Establishes the current level upon receipt of a valid trigger |
| KepecoDCPwr Configure Triggered Voltage Level | Establishes the voltage level upon receipt of a valid trigger |
| KepecoDCPwr Configure Triggered Source | Establishes the source for a trigger and arms the trigger system. |
| KepecoDCPwr initiate[TRG] | Issue a software trigger to the BOP-BIT. |
| KepecoDCPwr Abort [TRG] | Clears the trigger system. |
| Error and status reporting | |
| KepecoDCPwr Error-Query | Provides entry from the error queue of the power supply. |

2.4 USING THE DRIVER FUNCTIONS

The interactive example uses various driver functions of the LabView G driver. The following two paragraphs details the inputs and outputs of two of typical VI's used in the interactive example: KepecoDCPwr Configure Voltage Level.vi (PAR. 2.4.1) and KepecoDCPwr Measure [MSR].vi (PAR. 2.4.2). An overall block is provided for each function which shows inputs and outputs, followed by detailed descriptions of inputs and outputs. A detailed block diagram with associated text is also provided to show how the function is implemented.

2.4.1 KepecoDCPwr Configure Voltage Level.vi

This VI configures the DC voltage level the power supply is expected to generate.

2.4.1.1 CONNECTOR PANEL

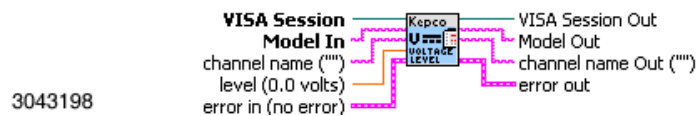


FIGURE 2-1. KepecoDCPwr Configure Voltage Level.vi CONNECTOR PANEL

2.4.1.2 CONTROLS AND INDICATORS

See Table 2-2.

TABLE 2-2. KepcoDCPwr Configure Voltage Level.vi INPUT/OUTPUT DESCRIPTIONS













| DESCRIPTION | SYMBOL |
|--|---|
| channel name - (""") Passes the name of the channel on which to configure the voltage level. It is not required except for Kepco's MST Series power supplies and is provided for compatibility. Valid Channel Names: 1 - 27 Default Value: "" |  |
| level (0.0 volts) - Passes the DC voltage the power supply is to generate. The driver uses this value to set the Voltage Level. The level is not checked by the driver function but is checked by the BOP-BIT or power supply. Default Value: 0.0 volts |  |
| error in (no error) - The error in cluster can accept error information wired from VI's previously called. Use this information to decide if any functionality should be bypassed in the event of errors from other VI's. The pop-up option Error (or Explain Warning) gives more information about the error displayed (see Table 2-3). |  |
| VISA Session - The ViSession handle is obtained from the <i>KepcoDCPwr Init</i> or <i>KepcoDCPwr InitWithOptions</i> function. The handle identifies a particular instrument session. Default Value: None |  |
| Model In - Model In is the type of Kepco power supply. This defines the capabilities and requirements in other Kepco functions to insure proper operation of the command. This function works on All Kepco power Supplies: BOP-HP, BOP-LP, BOP-BIT, STANDARD, MULTI and MULTI_PLUS. The MULTI type power supplies require a channel number to operate correctly. The constant is created by the Recognize Kepco model utility. |  |
| error out - The error out cluster passes error or warning information out of a VI to be used by other VI's. The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed (see Table 2-3). |  |
| VISA Session Out - The ViSession handle is obtained from the <i>KepcoDCPwr Init</i> or <i>KepcoDCPwr InitWithOptions</i> function. The handle identifies a particular instrument session. Default Value: None |  |
| Model Out - The Model Out handle is obtained from the <i>KepcoDCPwr Init</i> or <i>KepcoDCPwr InitWithOptions</i> function. The handle identifies a particular instrument. Default Value: None |  |
| channel name out - (""") Passes the name of the channel on which to configure the OVP limit. Valid Channel Names: 1-27; Default Value: "" |  |

TABLE 2-3. ERROR (OR EXPLAIN WARNING) CODES

| | |
|---|---|
|  | status - The boolean is either TRUE (X) for an error, or FALSE (check mark) for no error or a warning. The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed. |
|  | code - The code input identifies the error or warning. The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed. |
|  | source - The string describes the origin of the error or warning. The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed. |

2.4.1.3 BLOCK DIAGRAM DESCRIPTION

Figure 2-2 shows that the level input is converted to a string that is compatible with the power supply, floating with 5 places to the right of the decimal (%.5f). Then, using the VISA write function, the string is sent to the power supply. In the string conversion, the format string includes "% . ;" which is a Labview 2010 feature that ensures a period is used in the numbers, making the Labview driver functional in all countries.

The case statement in the middle of Figure 2-2 shows that all Kepco power supplies support this same functionality.

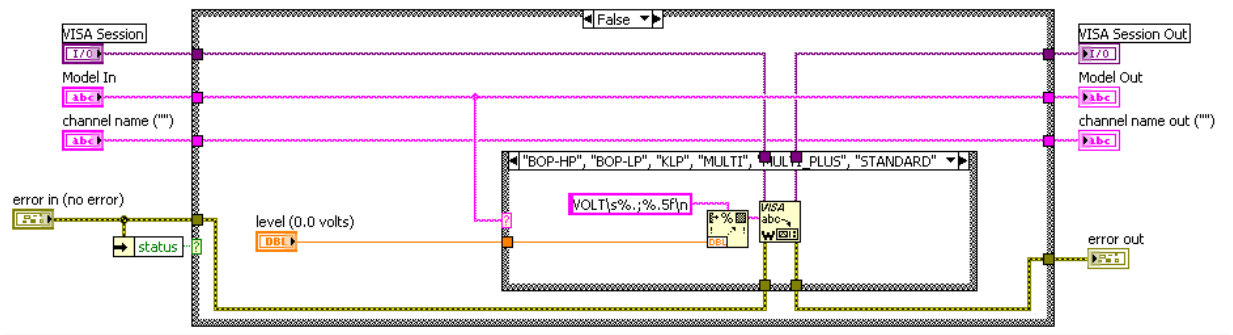


FIGURE 2-2. KeptoDCPwr Configure Voltage Level.vi BLOCK DIAGRAM

Figure 2-3 shows that when the error is true (not equal to zero), the function is a null function, passing the inputs (VISA session, Model, Channel and error) through the function without modification.

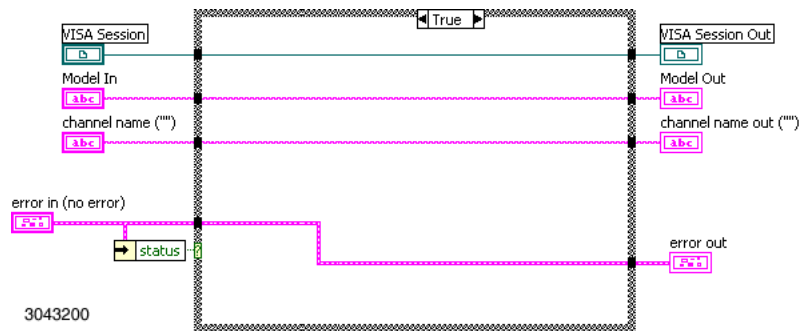


FIGURE 2-3. KeptoDCPwr Configure Voltage Level.vi ERROR BLOCK DIAGRAM

2.4.2 KeptoDCPwr Measure [MSR].vi

This VI takes a measurement of the output signal and returns the measured value (see Figure 2-4).

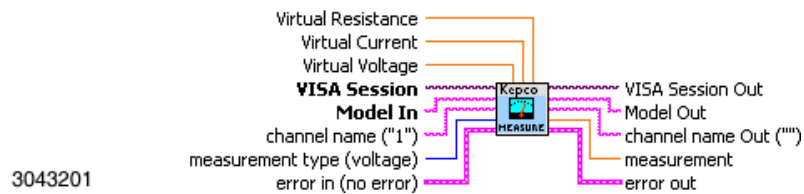


FIGURE 2-4. KeptoDCPwr Measure [MSR].vi CONTROL PANEL

This VI has the standard inputs of all of the functions of the LabView G Driver. It also has three orange inputs at the top of the VI (Virtual Resistance, Virtual Current, Virtual Voltage). These inputs are used in simulation. They do not need to be connected, but provide the ability to verify

the operation without using the power supply. These simulation inputs are only used when error in is true (indicating an error has occurred).

TABLE 2-4. KepcoDCPwr Measure [MSR].vi INPUT/OUTPUT DESCRIPTIONS

| DESCRIPTION | SYMBOL |
|---|--------|
| channel name - ("1") Passes the name of the channel on which to configure the voltage level. It is not required for most supplies and is provided for compatibility. Valid Channel Names: 1 Default Value: "1" | |
| error in (no error) - The error in cluster can accept error information wired from VI's previously called. Use this information to decide if any functionality should be bypassed in the event of errors from other VI's. The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed (see Table 2-3). | |
| VISA Session - The ViSession handle is obtained from the <i>KepcoDCPwr Init</i> or <i>KepcoDCPwr InitWithOptions</i> function. The handle identifies a particular instrument session. Default Value: None | |
| Model Out - The Model Out handle is obtained from the <i>KepcoDCPwr Init</i> or <i>KepcoDCPwr InitWithOptions</i> function. The handle identifies a particular instrument. Default Value: None | |
| channel name out - ("") Passes the name of the channel on which to configure the OVP limit. Valid Channel Names: 1; Default Value: "" | |

Figure 2-5 is a block diagram which shows the method used by the VI to perform the simulated current measurement. The query MEAS:CURR? is transmitted to the power supply which replies with the current measurement.

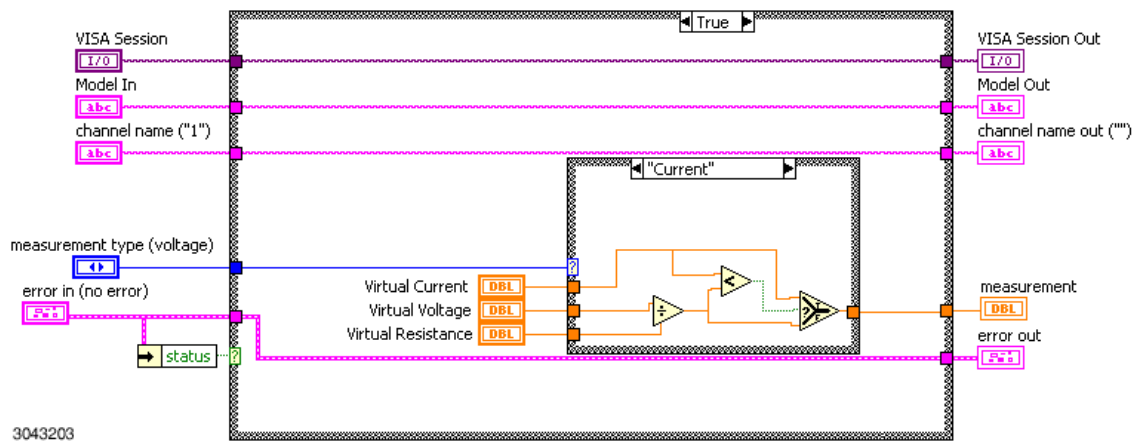


FIGURE 2-5. KepcoDCPwr Measure [MSR].vi BLOCK DIAGRAM

2.5 EXAMPLE OF SETTING THE OUTPUT

Figure 2-6 shows the use of the Kepco subvi's to initialize the driver and the device, set the output on or off, establish a voltage and current setting, get a measurement and unit status, and close the session to release the memory. The file is KepcoDCPwr Voltage Application Example.vi It is located in program files\national Instruments\labview\drivers\kepco. The subvi's used to implement this example are described in the following paragraphs. Although the LabView G driver provides a variety of functions (see Table 2-1), only the ones shown in Figure 2-6, needed for this example.

2.5.1 INITIALIZATION FUNCTION

The first block of Figure 2-6, *KepecoDCPwr initialize* function, performs the open, creating the connection between the LabView program and the power supply using the VISA.dll supplied with LabView. The *KepecoDCPwr initialize* function has two optional settings: reset unit and identify unit. The reset unit option sends *RST to the unit which sets the output to 0 volts, minimum current, turns the output off, and stops list execution. The identify unit option sends an *IDN query to the power supply.

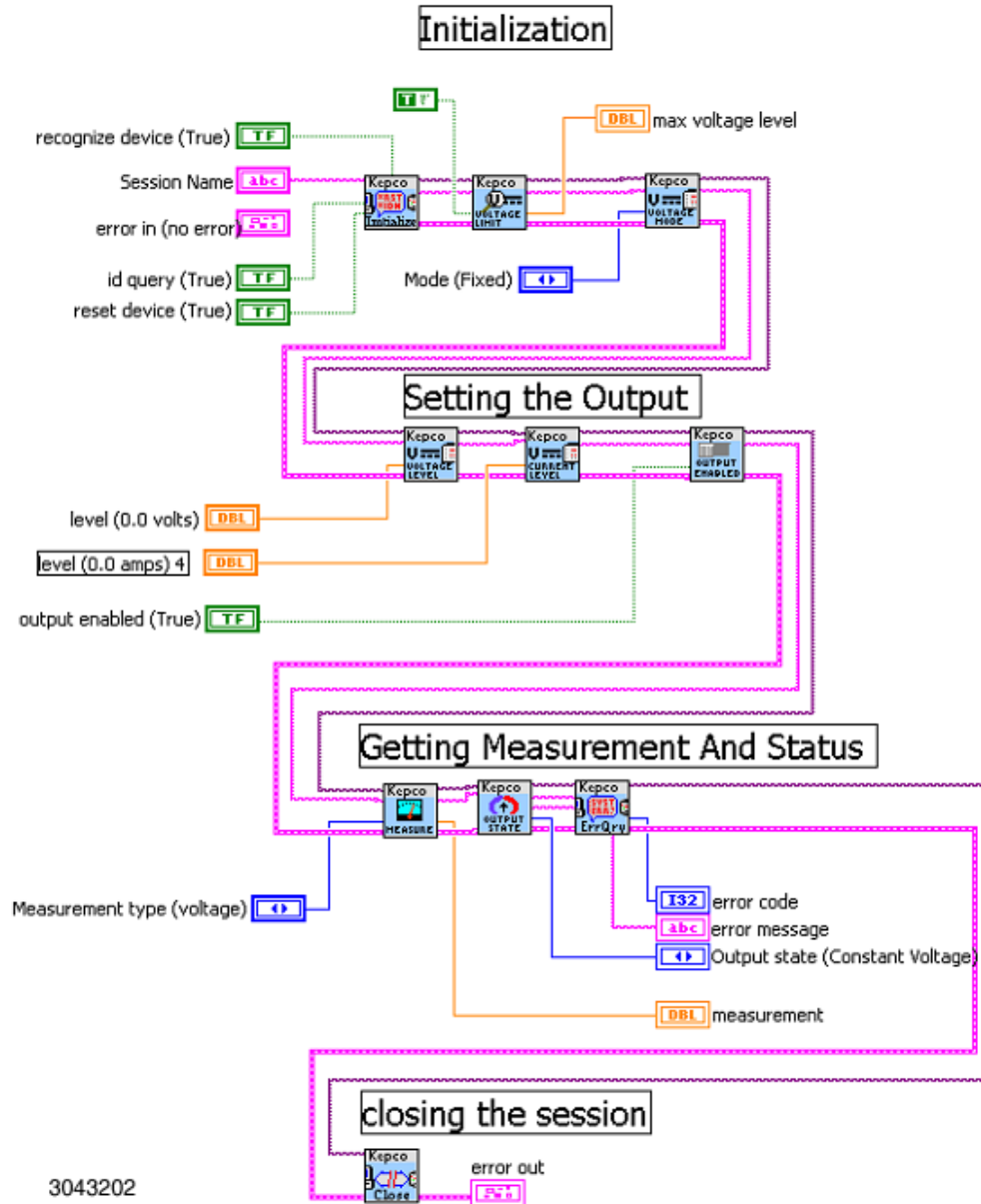


FIGURE 2-6. SETTING THE OUTPUT, OVERALL BLOCK DIAGRAM

The initialization function use three subvi's:

- *KepecoDCPwr initialize* - Initializes the driver.
- *KepecoDCPwr Query Voltage Limit* - Verifies the limit model voltage setting.
- *KepecoDCPwr Configure Voltage Mode* - Sets Mode to fixed, thereby stopping list execution.

The initialization function uses five inputs:

- Recognize device - T-F (True-false) input. If true, IDN string is parsed, looking for proper model (BOP-BIT). If false, model is automatically set to BOP-BIT for simulation mode.
- Resource string - The VISA resource string, e.g., GPIB::6::INSTR
- ID Query - T-F input. If true, allows the unit to be identified by sending *IDN query. If false, program enters simulation mode and no commands are actually sent to the power supply.
- Error in - Described in Table 2-2.
- Reset device - T-F input. If true, sends *RST to unit which turns the output off, sets voltage to zero and current to minimum, and disables execution of a programmed list.

The initialization function provides three outputs:

- Visa Session - a handle defining the connection.
- Model - a string providing the model type for the functions.
- Error out - If the function did not fail, this is a 0.

Initialization of the GPIB is simple and is not needed in Labview 2010. For serial and LAN communication the open is required.

Figure 2-7 shows the serial initialization. This loop is executed to find the baud rate of the BOP-BIT and configure the port correctly. It uses the VISA clear input function and the query <LF><LF>*idn?<LF> to terminate any input in the BOP-BIT buffer and to establish the BOP-BIT baud rate. Only when communication is established will the string that begins with KEPCO be found on the port.

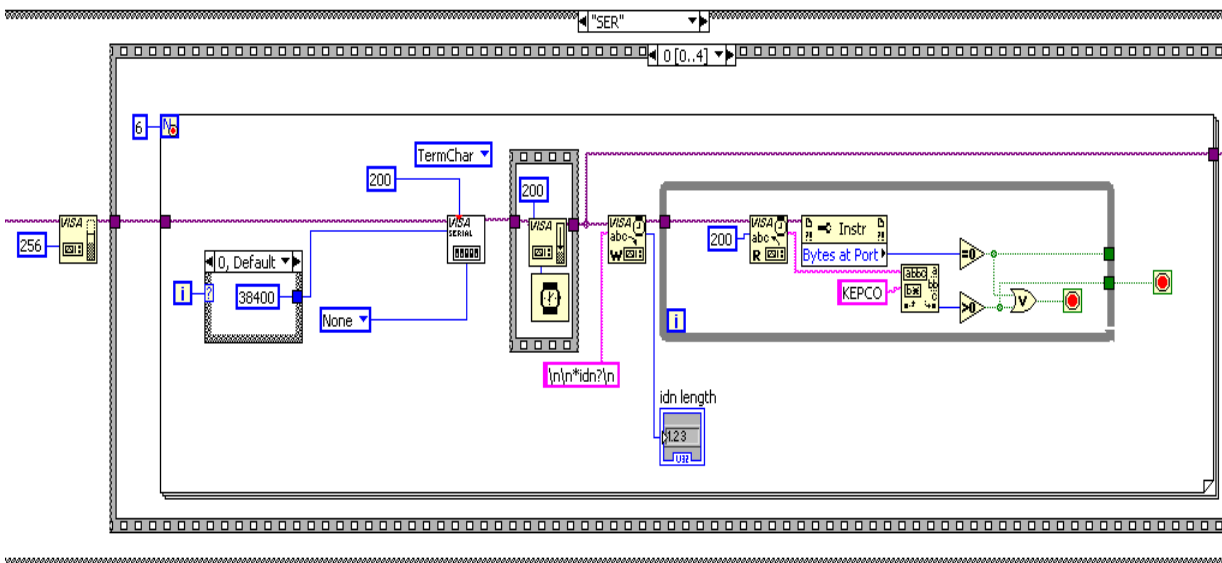


FIGURE 2-7. SERIAL INITIALIZATION BLOCK DIAGRAM

Figure 2-8 shows the LAN initialization. Like the serial port, the <LF><LF>*IDN?<LF> string is used to synchronize the port emptying any leftover data. The VISA resource establishes that the termination of data input is defined as a LF and enabled.

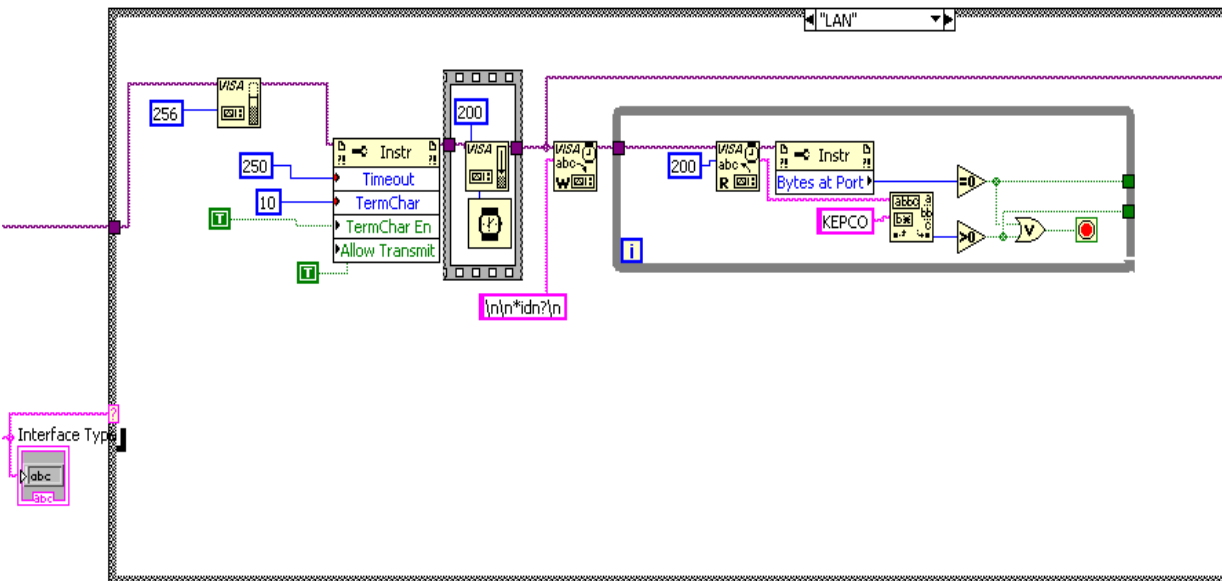


FIGURE 2-8. LAN INITIALIZATION BLOCK DIAGRAM

The recognize unit utility function parses the response to determine the proper model number output. If the identify unit flag is not set, the MODEL OUT of the initialize function reports Standard as the power supply type. The Standard model is a unit that only supports setting voltage, current and output on/off, and performing measurements; all other functions are bypassed. The function *KepecoDCPwr initialize with options* (not used in the example) uses *KepecoDCPwr initialize* except that Kepco-recommended options are preset: the reset and identify flags set.

All subsequent functions have at least two inputs: VISA session and Model In. VISA session is the output of *viOpen*. Model In is a string; for the BOP-BIT the string is *BOP-BIT*.

The second block, *KepecoDCPwr Query Voltage Limit*, is needed to verify the limit model settings. It is recommended that the response from this block be checked to insure that the power supply is configured to supply the voltage and current needed. For example, if the limit model for a BOP-BIT 75-32 was set to 36 Volts, 32 Amperes, the response will be 36 to this query, indicating that the program will not respond to a command to set the output to 50 Volts.

The third subVi is *KepecoDCPwr Configure Voltage Mode*. This block sets the mode to Fixed. It is recommended that the mode be set to Fixed in order to stop a list if it is running, however if the Reset Device input into the first block is true, this block is not needed since *RST also stops a list.

2.5.2 SET THE OUTPUT ON OR OFF

The following three functions are all that are required to set a voltage, current and change the output state:

- *KepecoDCPwr Configure Voltage Level* - Sets the voltage output of the power supply
- *KepecoDCPwr Configure Current Level* - Sets the current output of the power supply
- *KepecoDCPwr Configure Output State* - Turns the output on

2.5.3 GET MEASUREMENT AND STATUS

Three functions are needed to retrieve a measurement (either voltage or current), determine the output state and determine if there have been any errors.

- *KepecoDCPwr Measure [MSR]* - Measures the output (voltage in voltage mode, current in current mode). This function is explained in detail in paragraph 2.4.2.
- *KepecoDCPwr Query Output State* - Queries the output state and returns 1 for output on, 0 for output off.
- *KepecoDCPwr Error-Query* - Retrieves the first error from the error queue. The internal logic of this subvi changes the "0- no error" response to a null string, so no error returns nothing, but if an error is detected returns an error string is returned.

2.5.4 CLOSE CONNECTION

The last section is the close function, *KepecoDCPwr close*. This function is required to remove the connection to the instrument and clean up RAM usage in LabView. It is very important to release the connections in E-Series models as there are only four connections possible over VXI-11 (Port 1024) and only two connections possible over SCPI Raw (Port 5025). The *KepecoDCPwr close* function releases the connection for other programs to utilize the device.

When the LabView program completes, it must close the driver, however the output is not automatically turned off. Therefore it is recommended that the *KepecoDCPwr Configure Output State* function precede the *KepecoDCPwr close* function. This is not shown in the simplified example (Figure 2-6), however it is illustrated in the full block diagram showing implementation of the ramp function, Figure 2-15.

2.6 VOLTAGE APPLICATION EXAMPLE.

The voltage application example (see Figure 2-9) is similar to the simple example with some minor exceptions. In the setup of the power supply, the mode of operation, (voltage mode is established by the Mode function). The BOP-BIT has two modes: either voltage with current protection or current with voltage protection. The Main mode forces the BOP-BIT to Fixed operating state and is used to stop either a transient or a list from being executed. The current limit query determines the max current and it's output is provided to the configure current level which in the BOP-BIT also sets the current protection level.

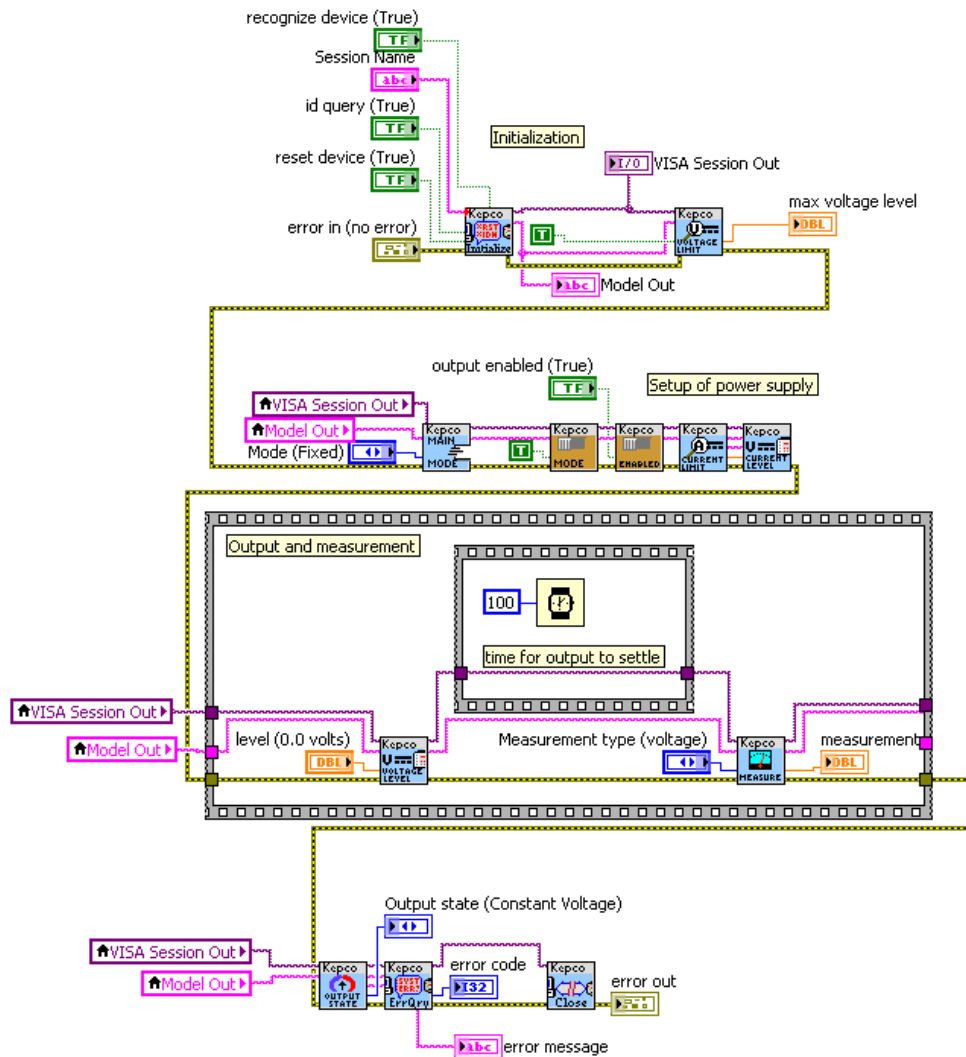


FIGURE 2-9. VOLTAGE APPLICATION EXAMPLE

The sequence section sets the main channel output, waits some amount of time and then sends the measurement. The time is to allow for the measurement system of the BOP-BIT to settle and provide a valid measurement to the user. The VISA session and model are applied globally to the vi's but the error is run through all vi's. The error connection insures the vi's are run sequentially even though it is not important in the setup of the power supplies.

The query error vi at the end will either return no error or will return any error from the start of the example until the end. The only error than might occur is "voltage out of range" as the input control is not limited by the max voltage level.

2.7 INTERACTIVE DEMONSTRATION PROGRAM

The interactive demo program allows the user to operate the power supply and provides samples of how to implement the functions.

2.7.1 KepcoDCPwr Interactive Example.vi

KepcoDCPwr Interactive Example.vi is the main demonstration program. The file is located in program files\national Instruments\labview\drivers\kepco. Either double-click on the file or open the file using File > Open from within LabView. When the program is opened, the Front Panel (Figure 2-10) is displayed.

Below the logo is the **VISA Resource Name Running** field. With the application stopped, click on this field to display a drop-down menu. Clicking Refresh causes the LabView program to initiate a new search and provide an updated list of all devices found on the GPIB, Serial ports and network connections. You may enter a valid resource name directly or, for demo mode, enter a blank resource name.

The **VOLTAGE** and **CURRENT** knobs are used to set the voltage and current of the power supply. Once the run button is clicked, the **VOLTAGE** and **CURRENT** knob markings will show the limit model minimum and maximum values. The corresponding **DC VOLTS** and **DC AMPERES** displays report the measured voltage and current of the power supply and are updated once every 125 milliseconds. As the knob rotates, the number in the counter below the knob changes; if a number is entered in the counter, the knob rotates to the proper position.

If the instrument is found, the demonstration program runs in normal mode. If the instrument is not found (the **Instrument Error** box at the lower right shows **instrument not found**), the demonstration program runs in simulation mode. In simulation mode the resistance value at the bottom of the front panel is a simulated load used to measure the DC output. The simulated load can be set by the user to control the **DC VOLTS** and **DC AMPERES** displays.

The **Power Supply Mode** switch is used to establish the current or voltage mode of the BOP-BIT. It is used to start either a) the voltage or current software-timed ramps or b) voltage or current list examples when either the **Software Ramp** or **Programmed Lists** button, respectively, is clicked. The indicator to the right of the switch shows whether the unit is primed to produce voltage ramps (green) or current ramps (yellow).

The **DC OUTPUT** switch can be clicked to enable/disable the output. The green indicator within the switch lights when the output is enabled.

The four buttons at the bottom of the front panel invoke standalone functions which can be used to 1) create a computer-timed software ramp (**Software Ramp**), 2) create a ramp using the list functionality (**Programmed List**), 3) create functions and files for executing complex waveforms (**Function Generator**) or 4) create an arbitrary series of points (**Arbitrary Waveform**).

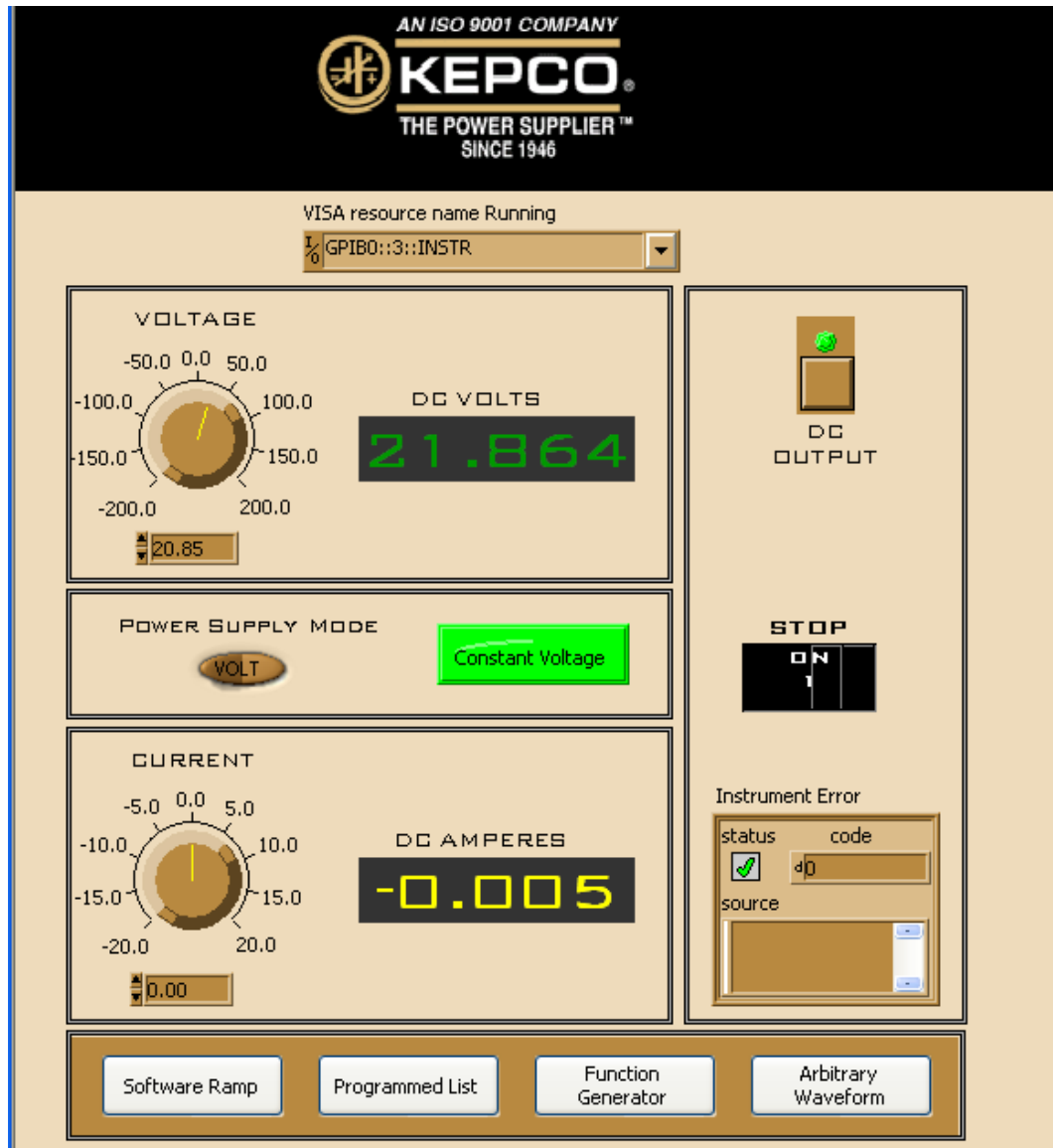


FIGURE 2-10. FRONT PANEL WINDOW

2.7.2 SOFTWARE TIMED RAMP EXAMPLE

When the Software Ramp button on the front panel is depressed, the panel shown in Figure 2-11 is displayed. The figure illustrates a current ramp; the voltage ramp panel is similar. The {Power Supply Mode switch on the front panel (Figure 2-10) determines whether the current ramp or voltage ramp is displayed. It is also possible to run this program by selecting either:

KepecoDCPwr Software Timed Ramp Example.vi for a voltage ramp

KepecoDCPwr Current Software Timed Ramp Example.vi for a current ramp.

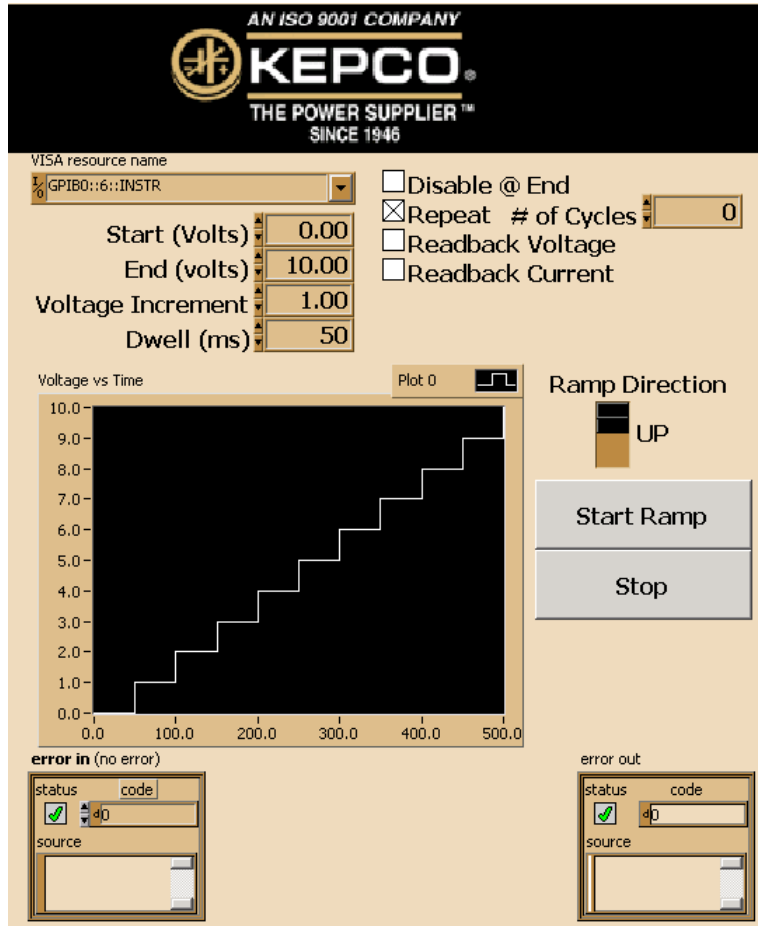


FIGURE 2-11. SOFTWARE RAMP PANEL, CURRENT RAMP EXAMPLE

These files are located in the *Labview/user.libs/kpDcpwr* folder created when the Kepeco LabView G driver is installed and can be executed by selecting File > Run from within LabView. When invoking either of these files directly, either a network address or a GPIB address must be set before clicking Start Ramp (this is automatic when executing Software Ramp via the Front Panel).

The four boxes labeled **Start** (beginning of ramp), **End** (end of ramp), **Voltage** Increment [or **Current** Increment] (determines the size of steps, affecting the smoothness of the ramp) and **Dwell** (length of time from start to end of ramp) are used to build the ramp. As soon as these values are entered, the **Current** [or **Voltage**] vs. **Time** Graph shows what the settings will produce when applied to the output.

The check boxes on the right are effective once the **Start Ramp** button is clicked.

- **Disable @ End.** Checked causes the output to turn off when the Ramp is complete.
- **# of Cycles.** Checked to enter how many ramps will be created; setting of 0 causes ramps to continue indefinitely. Unchecked to create one ramp.
- **Readback Voltage, Readback Current.** Once **Start Ramp** is clicked, checked causes the **DC VOLTS** and **DC AMPERES** indicators on the Ramp Function Panel (see Figure 2-12) to show measured voltage and current; unchecked causes the indicators to show the commanded values of voltage and current, respectively.

When **Start Ramp** is clicked, information from the check boxes and # of Cycles is passed to the example program and used to start ramp execution (see PAR. 2.7.3).

The Error in and Error out boxes are standard error callouts, showing 0 for no error, and an error code and associated string if there is an error.

2.7.3 SOFTWARE TIMED RAMP EXECUTION

Clicking the **Start Ramp** button on the Software Ramp panel (Figure 2-11) causes the Ramp Function Panel (Figure 2-12) to be displayed as the ramp is being generated. Checking the **Readback Voltage** or **Readback Current** checkboxes above the **DC VOLTS** or **DC AMPERES** indicators cause the Voltage and Current indicators to show measured voltage and current vs. commanded values of voltage and current if unchecked.

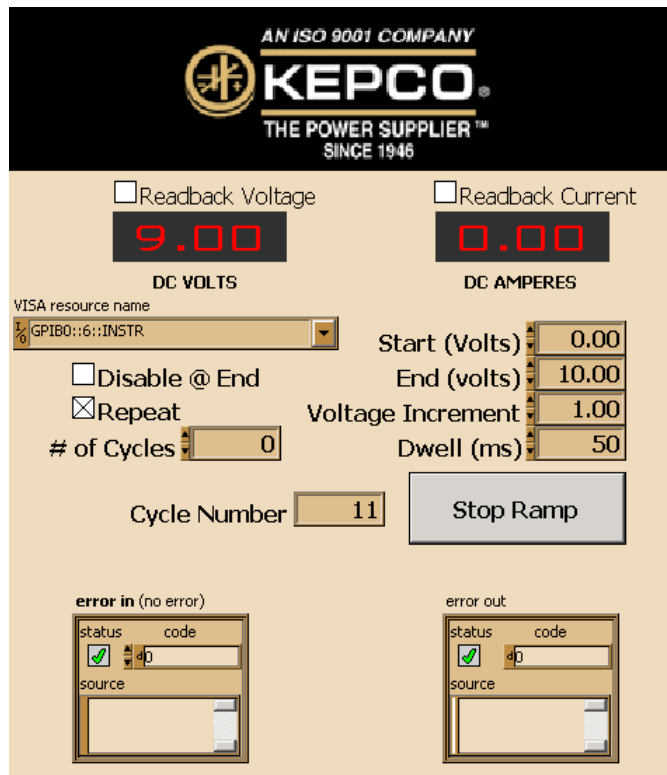


FIGURE 2-12. RAMP FUNCTION PANEL, CURRENT RAMP

The information from the Software Ramp Panel is passed to either *KepecoDCPwr Software Timed Ramp Example.vi* for a voltage ramp or *KepecoDCPwr Current Software Timed Ramp Example.vi* for a current ramp. The computer sends the proper commands to the BOP-BIT that generate the ramp. The checkboxes for Readback Voltage, Readback Current can be altered while the ramp is being generated to allow the indicators to display either measured or commanded voltage or current. The Disable @ End and Repeat checkboxes can also be altered while the ramp is being generated.

The actual function that creates the ramp, a series of points, is shown in Figure 2-15. The logic for each point is very simple: send the voltage to the unit with the voltage level command, measure the voltage using the Kepeco measure vi and do it again for current. A delay is initiated at the start of the function so the time it takes for the commands to execute does not affect the overall ramp operation (unless the dwell time is less than 0.01 second on an older PC or about 0.008 second on a higher performance computer).

2.7.4 FUNCTION GENERATOR

When the Function Generator button on the Front Panel is clicked, the function generator panel (Figure 2-13) is displayed. The frequency slider determines the frequency of the generated waveform. The number of points determines the limits for frequency adjustment. For example, if 10 points are selected, the maximum frequency is 100 Hz, the lowest frequency is 0.0015 Hz. The number of points determines the smoothness of the waveform because it is constructed incrementally, but it also limits the maximum frequency. For example, if the 1002 (maximum) points are selected, the frequency is 0.00001 Hz.

The Amplitude and Offset sliders are to the right of the Frequency slider. When the requested output exceeds the capability of the BOP-BIT, the output is clipped by the function generator. Below these sliders is the Waveform Type dropdown selector, either Sine, Sawtooth, Triangle or Square. Below the Waveform Type dropdown is a visual display of the points that have been specified. To the left of the displayed waveform is the Duty Cycle slider which affects square wave-shapes only. The Phase slider shifts the start point of the waveform. As these sliders are changed, the digital display beneath the slider shows the setting. Changes to the slider settings immediately change the Commanded Waveform display to reflect the new parameters.

Above the Start button, two additional parameters required for the execution of the waveform are displayed. The value of the non-changing parameter, current (in Amperes) for a voltage waveform (shown in Figure 2-16) or voltage (in Volts) for a current waveform must be specified. The # of Cycles specifies how many waveforms to produce; if set to 0 the waveform repeats continuously.

Once the Start button is clicked, the commands are sequentially sent to the BOP-BIT to build the waveform by incrementally changing the output. While the waveform is being generated, periodic measurements of the BOP-BIT output are made and shown in the Measured Waveform display. To stop the waveform click on the Stop button, which is red while the waveform is running.

Figure 2-16 is a block diagram that shows how the function generator was implemented using list commands in this example. The *KepecoDCPwr Clear All Lists.vi* was executed first. This VI clears the previous list, allowing the new list to be sent to the BOP-BIT. Lists are not deleted from the BOP-BIT except when a limit model setting is changed or when the Clear All function is executed. The *KepecoDCPwr Configure User Sequence.vi* does the actual downloading of the points to the BOP-BIT unit. The *KepecoDCPwr Configure Main Mode.vi* causes the list to be executed.

The *KepecoDCPwr Configure User Sequence vi* block diagram is shown in figure 2-14. This vi requires five inputs: the mode of operation, the amplitude points, the dwell points, the protection setting and the number of cycles to be executed. It does not start executing the list nor establish the on/off state or mode of the power supply. The vi uses the individual vi's to format the data and send it to the BOP-BIT. The first function invoked is *KepecoDCPwr Clear All lists*. This function empties the BOP-BIT of any existing list. The case statement establishes the protection level of the BOP-BIT. *KepecoDCPwr Configure List Count* sets the number of list cycles to execute. Setting the list count to 0 causes the list to repeat indefinitely. The Dwell array is tested and if the min and max dwells are the same, then the array is changed to an array of one entry to speed up the download and setup time. The function generator only uses a single dwell time, so this path is always executed in this example. The last *KepecoDCPwr Configure List* downloads the voltage or current set points to the BOP-BIT.

The *KepecoDCPwr Configure User Sequence.vi* is shown in Figure 2-14. This logic is specifically configured to generate a voltage list where two arrays of voltage and dwell times are supplied, along with the operating current and number of cycles for the list to execute. In the left-most case statement there is a command to send one entry for the control and current lists of the internal arbitrary waveform generator. The last two functions in the diagram send the dwell and then the voltage list to the BOP-BIT. These functions, along with the individual write of the LIST:COUN command, set up the arbitrary waveform generator for the selected number of repetitions.

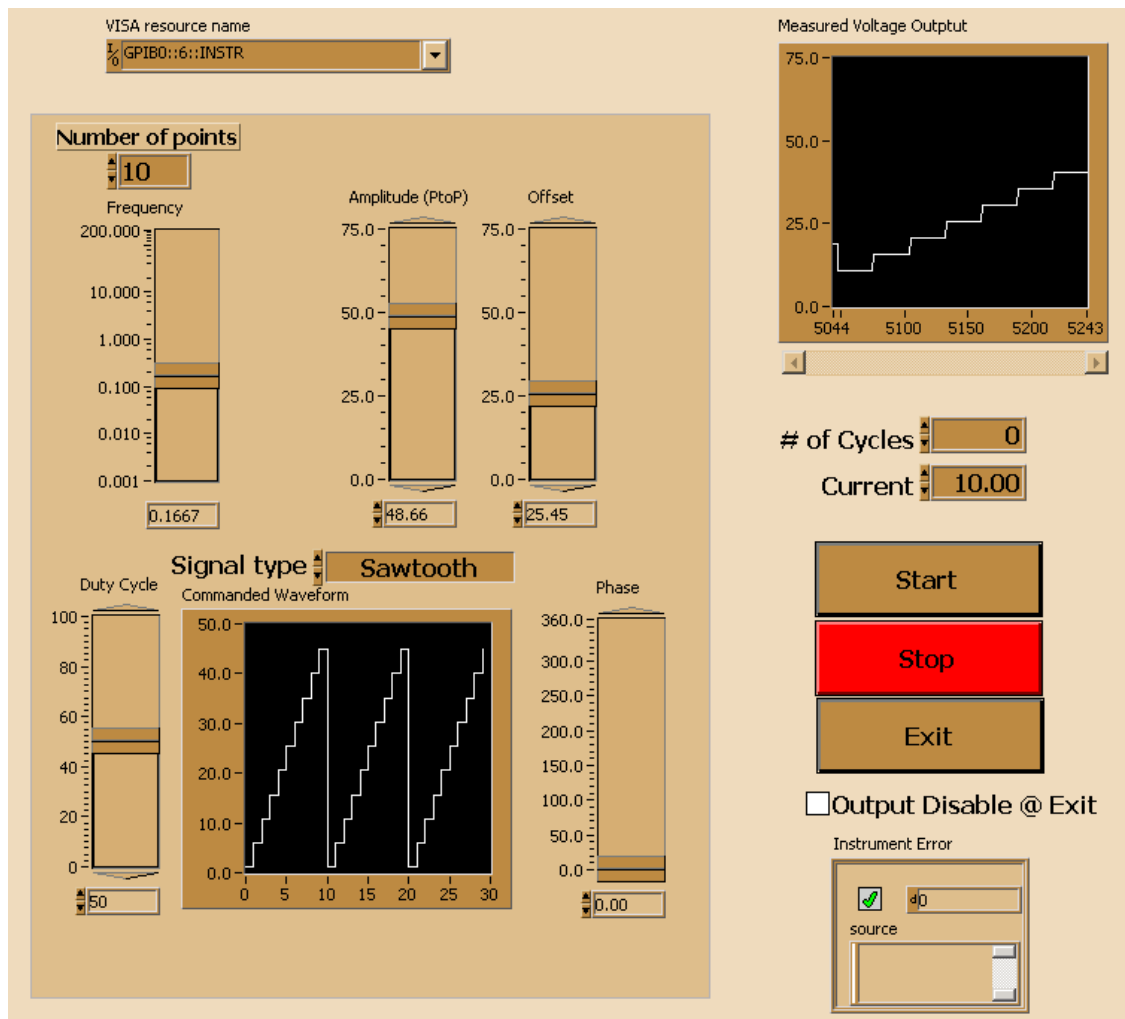


FIGURE 2-13. FUNCTION GENERATOR PANEL

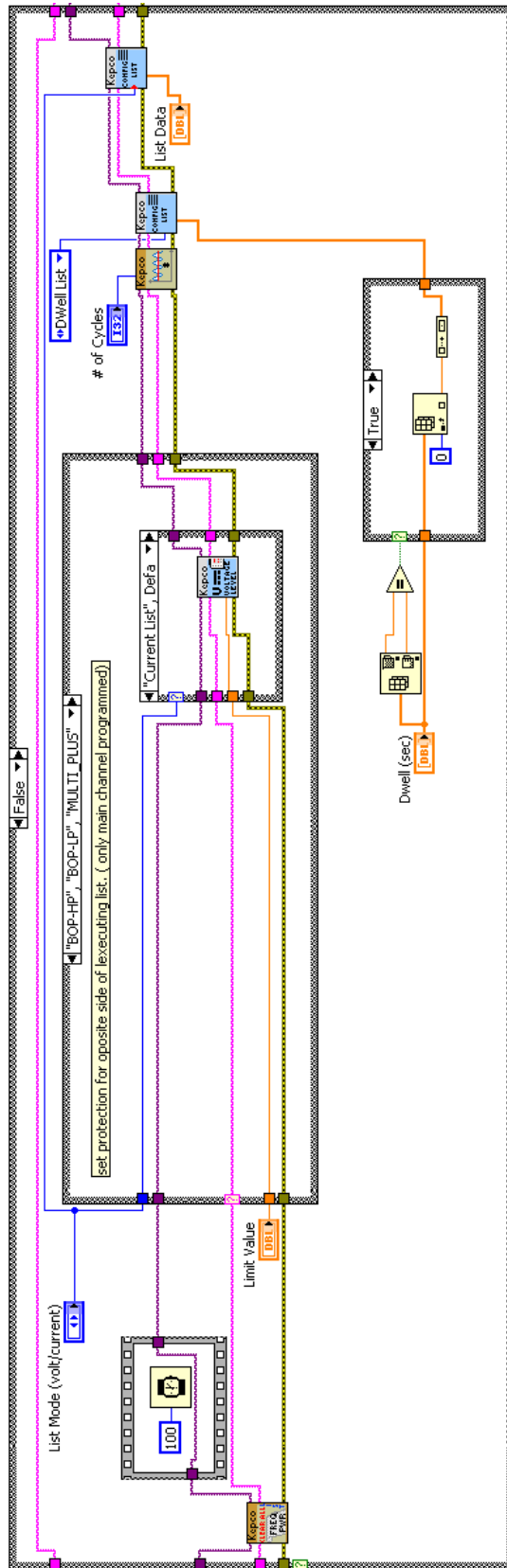


FIGURE 2-14. CONFIGURE USER Sequence.vi BLOCK DIAGRAM

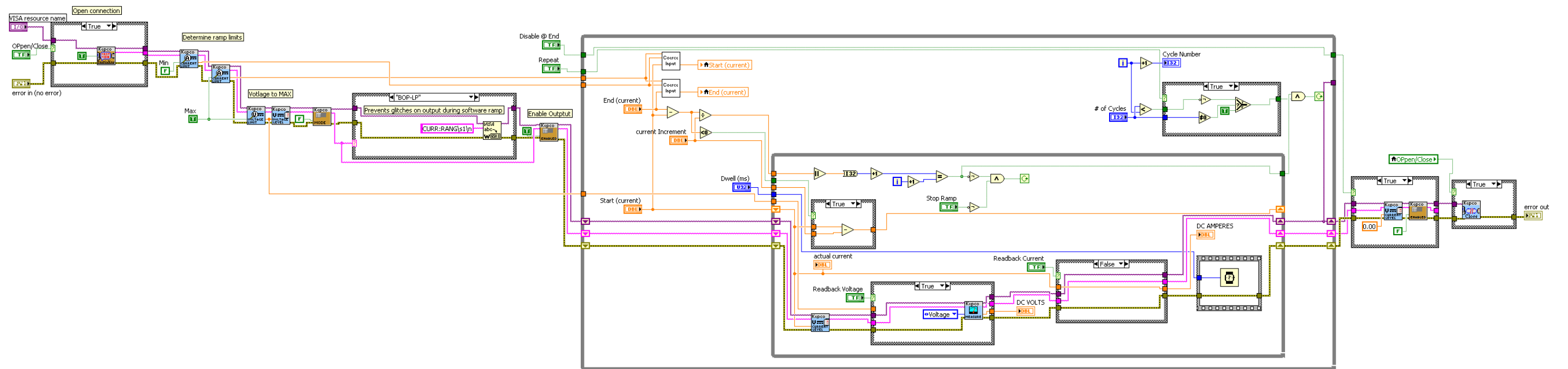


FIGURE 2-15. RAMP FUNCTION, BLOCK DIAGRAM

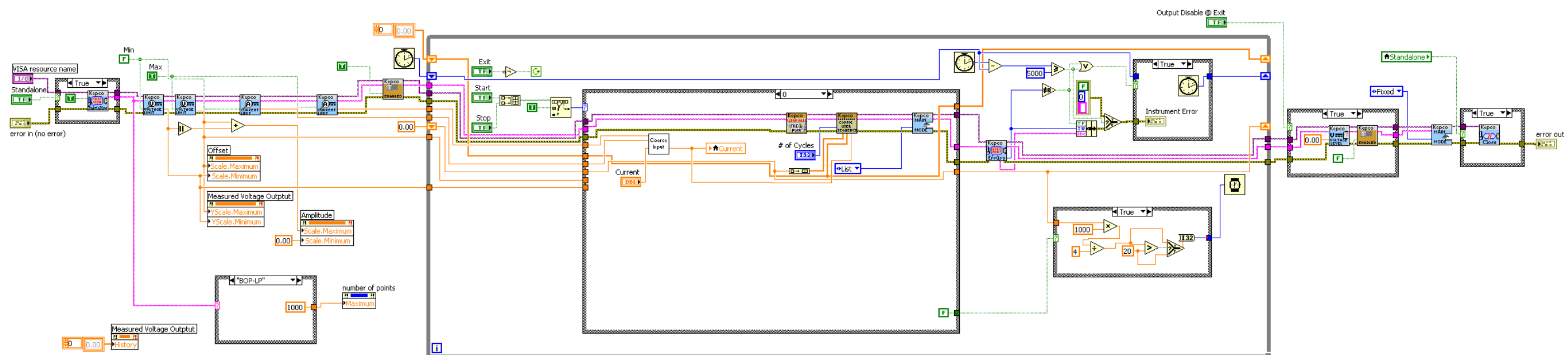


FIGURE 2-16. FUNCTION GENERATOR, LIST FUNCTIONALITY, BLOCK DIAGRAM (

SECTION 3 - VXI *plug&play* Driver

3.1 INTRODUCTION

Kepeco's power supplies support asynchronous RS-232, GPIB, and LAN communication interfaces. These interfaces, when being controlled through a C program such as CVI or Visual C, have different methods of sending the SCPI strings to the instruments. The Virtual Instrument Software Architecture (VISA) is a standardized method for communication with instruments over the various buses using identical calls. The VXI *plug&play* driver uses these visa calls.

3.2 DEMONSTRATION PROGRAM USING THE VXI *PLUG&PLAY* DRIVER

The demonstration program is intended to illustrate the use of the VISA functions included with the VXI *plug&play* driver. The demonstration program is installed under Windows by running SETUP.EXE. The program can be used to program and view the virtual front panels of up to 10 BOP Power Supplies. After the program is installed, double click on BOP.exe to run the program.

1. Unzip the files and once exploded, click on setup to install the driver, the CVI runtime and the VISA driver as needed. The screen provided allows the user to change the name and location of the two default directories: Kepeco_BOP_LAN_driver and National instrument components. The VISA driver and the CVI runtime are provided by National Instruments, Inc. and when the Next button is clicked, you must click Accept to install the National components. The installation may require a restart of the computer to complete the installation of the program. The BOP_LAN DRIVER folder will be added to the Start menu.
2. Install the VISA driver per the text file included with the driver.
3. At the Start-up screen (see Figure 3-1) select the type of interface, LAN or GPIB, and enter the correct GPIB address or LAN address in the **Address** field and click **CONNECT**.

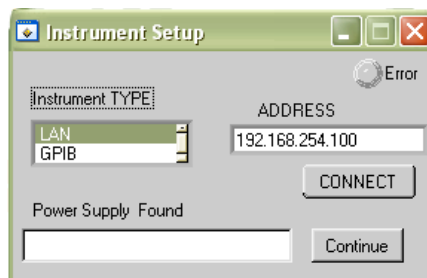


FIGURE 3-1. GPIB VISA DRIVER START-UP

4. When the **Power Supply Found** window shows BIT 4882, BIT 4886, BIT 802E or BOP1K, click **Continue**.
5. Once you are connected to the VISA interface, the virtual front panel appears (see Main Panel, Figure 3-2).

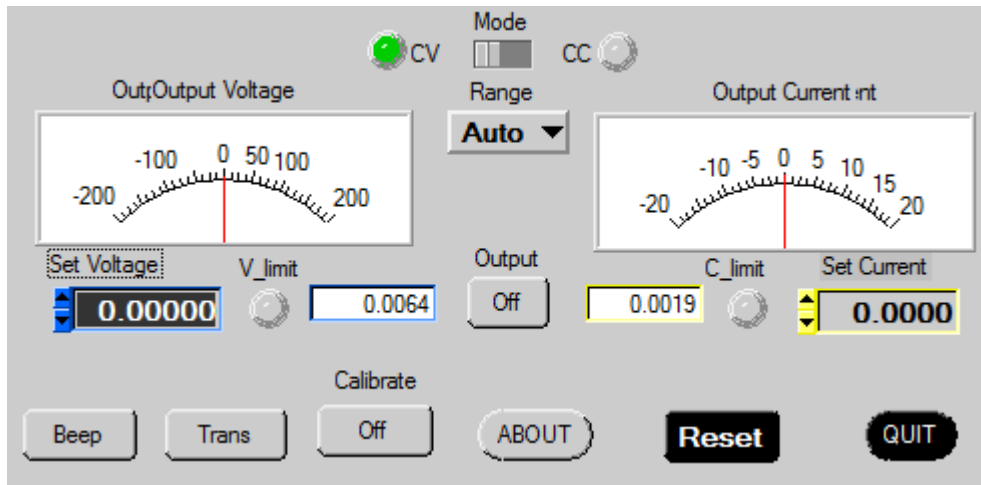


FIGURE 3-2. MAIN PANEL

3.2.1 MAIN PANEL CONTROLS AND INDICATORS

The main panel shows all power supply parameters simultaneously, without having to execute local controls and read the corresponding display. The main panel is a real-time display of output values and programmed parameters. All functions are available from the main panel. The title bar at the top of the window (not shown) indicates the BOP model connected to the GPIB interface.

NOTE: Some buttons may be disabled for special BOP models or those using special BIT cards (special models include a 5-digit suffix).

The **Mode** switch changes the operating mode (CV or CC); Mode indicators light to indicate whether the power supply is operating in either **CV** (constant voltage) or **CC** (constant current) mode.

The analog meters read actual output voltage and current; Two display windows on either side of the **Output** button provide a more precise digital readout of actual output voltage and current.

The **Range** button opens a dropdown box; to change the operating range: either High, Low or Auto range must be selected.

The **Output** button applies the programmed settings to the output terminals (ON) or keeps the output voltage and current at zero (OFF).

The **Gain** button (not shown - available only with BOP's of 1000W and above or with MDAC BIT card) opens a dropdown box which allows the selection of the reference voltage: INT for Internal or EXT1 or EXT2 for one of two external analog reference voltages used to calibrate the gain.

Indicators on either side of the digital display window, **V limit** and **C limit**, light (yellow) if the programmed voltage or current limit is exceeded and light (red) if a protection error is detected

Set Voltage and **Set Current** windows are used to program the output voltage and current for the unit; settings can be changed either by clicking on the arrows to the left of the display window, or by using the mouse to highlight the setting, then typing in the new value.

The **Beep** button causes the power supply to beep.

The **ABOUT** button displays the model, serial number and firmware version number.

The **Trans** button (not available with BIT 4882 card) generates a transient pulse with the amplitude and duration specified by the user in the Transient Pulse window (see Figure 3-3). The output level of the transient will either be V or A, depending on which mode is active. After the transient pulse is generated, the output will return to the programmed values in effect before the transient was generated.



FIGURE 3-3. TRANSIENT PULSE WINDOW

The **Trigger** button opens the Trigger window (Figure 3-4) which allows 99 different trigger voltage and current values, as well as mode to be stored and recalled. The STORE button stores the Trigger Mode, Trigger Voltage and Trigger Current values at the selected Memory location (1-99). The Recall button displays the Trigger Mode, Trigger Voltage and Trigger Current values stored in the selected Memory location. Clicking the Trigger button (or double-clicking RECALL) within the Trigger window causes the power supply output to be programmed to the settings stored in the selected Memory location.

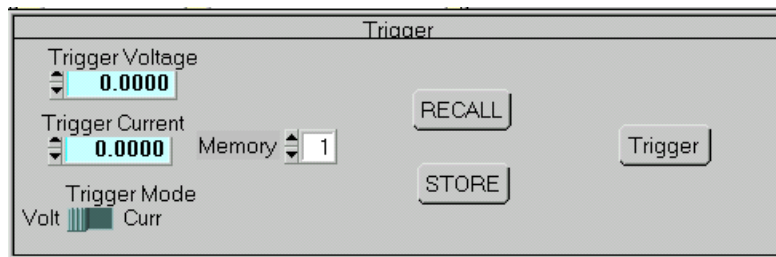


FIGURE 3-4. TRIGGER WINDOW

The Calibrate button opens the Calibration Window (Figure 3-4), and is used to recalibrate the unit; refer to the applicable Operator's Manual (see PAR. 1.1) for further details.

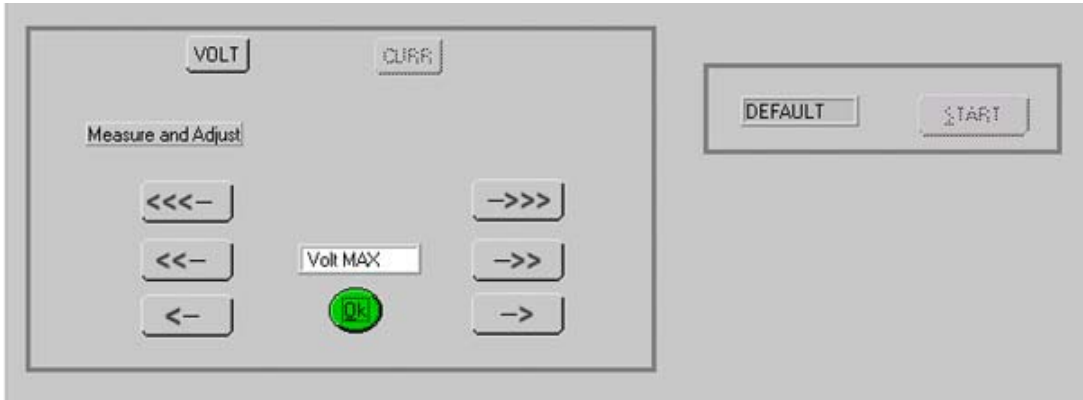


FIGURE 3-5. CALIBRATION WINDOW

The **Reset** button on the Main Panel (Figure 2) resets the unit to the power up defaults: output voltage and current set to zero, and output off.

The QUIT button on the virtual panel (Figure 2) is used to exit the sample VISA application.

3.3 SOFTWARE ENGINEER REFERENCE

3.3.1 VISA INSTRUMENT DRIVER FUNCTIONS

Kepeco's BOP-BIT VISA instrument driver provides programming support for Kepeco's BOP Power Supply (VISA I/O). It contains functions for opening, configuring, taking measurements from, test, calibration and closing the instrument. As these drivers are written using VISA calls, VISA is a required prerequisite. The installer installed the VISA operating system support component but not the visa.h file which defines the functions and return codes for use by a developer

Table 3-1 lists the functions that are available.

TABLE 3-1. BOP VISA DRIVER FUNCTIONS

| Purpose | Function Name | Description |
|---|---------------------|---|
| INITIALIZE FUNCTION | | |
| Initialize | KpBop_init | Initializes the instrument communication channel and configures the received data termination, transmission data termination, timeout and the various serial protocol configurations. Initialization also issues *CLS (resets the error and status system), *RST (sets power supply output off, setpoints to 0 and mode to voltage), and *IDN? (identifies power supply model and determines revision level). |
| APPLICATION FUNCTIONS — This class of functions contains high-level test and measurement routines. These functions call other instrument driver functions to configure, start, and get readings from the instrument. | | |
| Set and Measure | KpBop_ApplicSetMeas | Used to either set and read back, or just read back the output voltage, current and operating mode of the power supply |
| CONFIGURATION FUNCTIONS — This class of functions configures the instrument by setting system configuration parameters. | | |
| Get Current Range Status | Kpbop_getCurrRange | Unit can operate in either auto, high or low range. The function will return 0 for auto, 1 for high range and 4 for low range |
| Set Current Range | Kpbop_setCurrRange | Sets the current operating range (auto, high or low) |
| Get Voltage Range Status | Kpbop_getVoltRange | Unit can operate in either auto, high or low range. The function will return 0 for auto, 1 for high range and 4 for low range |

TABLE 3-1. BOP VISA DRIVER FUNCTIONS (CONTINUED)

| Purpose | Function Name | Description |
|--|-------------------------|--|
| Set Voltage Range | Kpbop_setVoltRange | Sets the voltage operating range (auto, high or low) |
| Get Minimum or Maximum Value | KpBop_GetMinMaxValue | Gets the minimum or the maximum output voltage, output current of the power supply depending on the min_max switch and slide selection position. |
| Set Trigger Voltage and Current | KpBop_SetTrig_Volt_Curr | Sets the trigger voltage and trigger current at the same time |
| Set Trigger Value | KpBop_SetTrigValue | Sets the trigger voltage or trigger current level, depending on the switch position. |
| Get Trigger Value | KpBop_GetTrigValue | Gets the trigger voltage or the trigger current level, depending on the switch position. |
| ACTION/STATUS FUNCTIONS — This class of functions executes commands and queries. It also provides functions which allow the user to determine the current status of the instrument. | | |
| Set Voltage and Current | KpBop_Set_Volt_Curr | Sets the output voltage and current at the same time. |
| Set Value | KpBop_SetValue | Sets the output voltage, the output current of the power supply, depending on the slide selection position. The values are checked against the maximum acceptable values for the corresponding power supply. |
| Get Value | KpBop_GetValue | Gets the output voltage, the output current of the power supply depending on slide selection position. |
| Transient pulse in voltage mode | Kpbop_TransVolt | Sets the power supply to the required value for the indicated time |
| Transient pulse in current mode | Kpbop_TransCurr | Sets the power supply to the required value for the indicated time |
| Clear List parameter | Kpbop_ListClr | Clears all the parameters of a list, set direction to UP and SEQ to Default (DSEQ) |
| Count list parameter | Kpbop_ListCount | Sets the LIST seq to run for COUNT periods |
| Dwell list parameter | Kpbop_ListDwell | Sets the time a list will run the current step (if entered for only location 0 will apply for all steps) |
| Executes a list | Kpbop_ListExec | Starts the execution of a list, if all the parameters were correctly set. |
| Defines list parameters (V or C) | Kpbop_ListData | Calculates the value (V or C, depending on which mode is active) for each of the STEPS of a list, in accordance with the waveform and the max and min values selected |
| Trigger | KpBop_Trig | Triggers the instrument once. The output will go to the trigger voltage and current values. |
| Measure Current and Voltage Output Values | Kpbop_MeasVoltCurr | Measures the values of output (voltage and current). |
| Measure Current and Voltage Output Values | Kpbop_MsVoltCurr | Measures the value of output voltage and then the value of output current. |
| Measure Output Value | KpBop_MeasValue | Measures the value of output voltage or current depending on the switch position. |
| Get Source Mode | KpBop_GetSourceMode | Gets the operating mode of the power supply. |
| Set Source Mode | Kpbop_SetSourceMode | Set power supply mode: voltage or current depending on the switch position. |
| Output On/Off | KpBop_OutputOnOff | Sets the output on or off. |
| Get Output Status | KpBop_getOutputState | Returns the output status (on or off). |
| Abort Measurement | KpBop_abortMeas | Sends the abort command. |
| Beep | KpBop_beep | The selected BOP power supply will emit a beep. |

TABLE 3-1. BOP VISA DRIVER FUNCTIONS (CONTINUED)

| Purpose | Function Name | Description |
|---|----------------------|---|
| DATA FUNCTIONS — This class of functions transfers data to or from the instrument. | | |
| Save/Recall Settings | KpBop_SaveRecSet | Saves the current settings to the selected memory location or restores previously stored settings from a memory location. The memory location range is from 1 to 40. |
| UTILITY FUNCTIONS — This class of functions provides lower level functions to communicate with the instrument and to change instrument parameters. | | |
| Identify Power Supply | KpBop_identify | Returns the full message returned by the power supply to the *IDN? query. This message contains four fields: manufacturer, power supply type, serial number and firmware version. |
| Revision Query | KpBop_RevisionQuery | Returns the revision numbers of the instrument driver and instrument firmware version from the *idn? query. This instrument driver's Revision Number is "Rev 1.0, 9/99, CVI 5.1" and the BOP firmware version is Firmware Version "1.0". This data is necessary when requesting technical support. |
| Model Query | KpBop_ModelQuery | Return the model number of the BOP power supply. |
| Serial Number Query | KpBop_SerialInQuery | Returns the serial number of the BOP power supply. The serial number is extracted from the answer to the *idn? query. |
| Query SCPI Version | KpBop_GetScpiVersion | Returns the power supply answer to the Query SCPI Version command. Checks the Standard Commands for Programmable Instruments (SCPI) language version. |
| Write To Instrument | KpBop_writeInstrData | This function writes commands and queries to the instrument to modify parameters and query device settings. |
| Read Instrument Data | KpBop_readInstrData | This function reads data from the instrument's output buffer and returns it to the specified variable in memory. Because the instrument may return both numeric and text data in response to queries, this function returns the data in string format. NOTE: If valid data is not available at the instrument's output buffer when this function is called, the instrument will hang up and the function will not return until it times out. If the time-out is disabled, this function will hang indefinitely and it may be necessary to reboot the computer to break out. |
| Reset | KpBop_psReset | Resets the instrument to a known state and sends initialization commands to the instrument. |
| Self-Test | KpBop_selfTest | Runs the instrument's self test routine and returns the test result(s). |
| Error-Query | KpBop_errorQuery | Reads an error code from the instrument's error queue. |
| Error Message | KpBop_errorMessage | Takes the Status Code returned by the instrument driver functions, interprets it and returns it as a user readable string. |
| Calibration Status On/Off | KpBop_CalStatus | Used to enter or exit the calibration state. To enter the calibration state an 11-character password is required. When the instrument is shipped from the factory the calibration password is "DEFAULT". The password protects the instrument against unauthorized calibrations. |
| Calibration Mode | KpBop_CalMode | Allows the user to select the calibration mode (either voltage or current calibration) and also allows selection of 0 (min) or full scale (max) calibration. |
| Calibration Output | KpBop_CalOutput | Allows the user to perform the power supply calibration in both voltage or current mode. By moving the digital to analog converter the number of LSB's specified in the repeat count, the user can approach the 0 or full scale value of the corresponding power supply. This is a fine adjustment for the 0 or full scale value. The user can specify a repeat count between 1 and 9 in order to avoid having to repeat sending the command |

TABLE 3-1. BOP VISA DRIVER FUNCTIONS (CONTINUED)

| Purpose | Function Name | Description |
|---------------------------------|-------------------|---|
| Calibration Zero | KpBop_CalZero | Used to zero the output of the power supply before switching from voltage calibration to current calibration (to allow a calibration resistor to be connected between the power supply output terminals). |
| Zero Voltage Calibration | Kpbop_CalVoltZero | Calibrate output zero in Voltage mode. |
| Zero Current Calibration | Kpbop_CalCurrZero | Calibrate output zero in Current mode. |
| Volt_low Calibration constants | Kpbop_CalVLSD | Read Calibration constants when unit is in low-volt calibration mode. |
| Volt Calibration constants | Kpbop_CalVSND | Read Calibration constants when unit is in voltage calibration mode. |
| Curr_low Calibration constants | Kpbop_CalCLSD | Read Calibration constants when unit is in low-current calibration mode. |
| Curr_high Calibration constants | Kpbop_CalCSND | Read Calibration constants when unit is in current calibration mode. |
| Calibration Save | KpBop_CalSave | Stores the calibration results in the nonvolatile calibration memory of the instrument. |
| Security Code | KpBop_CalCode | Allows the user to change the password to prevent accidental or unauthorized calibrations of the instrument. The password is stored in non-volatile memory, and does not change when power has been off or after a reset. To change the password, the instrument must already be in calibration status, ensuring that the user knows the current password. If the password is lost, call the factory for support. |
| Disable Password | Kpbop_PassDis | Allow user to disable the security code. |
| Dump Calibration | Kpbop_CalDUMP | Save Calibration constants in files. |
| Restore Calibration | Kpbop_CalRestore | Restore Calibration constants from files. |
| Close | KpBop_close | This function takes the instrument off-line. |

3.3.2 PROGRAM PRACTICES

In addition to providing prototype functions listed in Table 3-1, the BOP_MDAC.H file, also provides enumerations for various Kepco-specific variables. The PS_TYPE enumeration defines the various Kepco BOP models supported by the driver and the associated INVALID TYPE. The CAL:MODE and CAL:STATE enumerations provide details for the calibration of various BOP/BIT combinations.

3.4 EXAMPLES

The following examples show the functions needed to set voltage to 15V, current to 2 amp and output ON.

Example 1:

```
#INCLUDE "BOP_MDAC.N"/*add definition for driver prototype*/
ViByte ps_type;
ViSession BOP_Session;
Kpbop_init ("GPIB0::6", 1, &ps_type, &BOP_Session); //init ps
Kpbop_Set_Volt_Curr (BOP_Session, 15, 2);//voltage and current
Kpbop_OutputOnOff (BOP_Session, KEPCO_ON);//output on
```

Example 2:

```
#INCLUDE "BOP_MDAC.N"/*add definition for driver prototype*/
ViByte ps_type;
ViSession BOP_Session;
Kpbop_init ("GPIB0::6", 1, &ps_type, &BOP_Session); //init ps
Kpbop_SetValue (BOP_Session, KEPCO_VOLT, 15); //voltage
Kpbop_SetValue (BOP_Session, KEPCO_CURRENT, 2);//current
Kpbop_OutputOnOff (BOP_Session, KEPCO_ON);//output on
Delay (1);
Kpbop_SetValue BOP_SESSION, KEPCO_VOLT, 25);//voltage now 25 volts
Delay (1);
Kpbop_SetValue BOP_SESSION, KEPCO_VOLT, 10);//voltage now 10 volts
```