



Thank you for choosing to buy and apply the High Country Tek, Inc DVC710 digital controller, we appreciate your business.

This DVC 710 module has been pre-loaded with the file "025-00070_v1_0" at the factory giving the programmer a simple working example of the type of programming syntax that is used.

More information on the following topics can be found in the '**Programmer's User Guide**' located on the HCT website. This file will give the programmer some examples of the type of programming syntax that is used with the DVC710.

Please refer to the Operational Flow Chart below while using this document.

The program allows the output group PWM to be controlled by the analog input with the assistance of an I/O function curve, i.e. analog 1 will control output group1 coil1HS1 through Input/ Output Function curve valve_shaper_1. Factory default gives a 1:1 ratio of input to output.

Settings for this application can be modified by using the **Intella Program Loader Monitor 700 (PLM700)** which can be downloaded - FREE - from our website.

Start the PLM700; select the DVC710 Master indicator to open the window that displays all the DVC710 I/O. Near the center of the screen contains a button labeled I/O functions. Selecting this button will display the I/O function curves. They are selectable by a pull down menu located in the Lower Right hand corner of the I/O Function curve screen. From these screens, the user can adjust the ratio of input to output demand for each channel, i.e. the user may determine, that 20% input should command 40% of the rated output, these adjustments would be set here. Remember to select 'Send Changes' to make the change active. These changes are automatically stored so they are present on the next power up.

All analog inputs can be 'tuned' online in the same manner as the I/O Functions. Select the Analog Inputs button on the main screen to open a screen that allows setting of the analog inputs. Variables that can be adjusted are Voltage Limits, Voltage Calibration and Deadband volts. These variables are saved to EEmem variable locations. EEmem variables will be discussed later in this document. Always select "Save Changes" to activate new settings.

All PWM outputs can be 'tuned' online as well. Select the Output Groups button on the main screen to open a screen that allows setting of the output groups. Variables that can be adjusted are 'A' coil min / max current, 'B' coil min / max current, Current P & I, 'A' and 'B' coil ramp up and down, and dither amp and Hz. These variables are saved to EEmem variable locations. EEmem variables will be discussed later in this document. Always select "Save Changes" to activate new settings.

The Default program contains a Virtual Display, The Virtual Display allows the user to monitor program variable status while program is executing online. An example of how to program virtual display variables is located in both the 'Always' code section and the virtual display icon located within the project in the Intella programming tool.

To use the supplied default application, the user initially sets values to match the current application requirements for the analog inputs, the Output groups and the I/O Functions through the Program Loader Monitor as mentioned above. When selecting "Send Changes" to activate the new settings, the application automatically updates the EPROM and saves the settings to the unit permanently.



Programming the DVC710

The following sections describe the default application code in more detail. Programmers that wish to modify the default application or write their own application may use this as a template for their own needs.

An example of how to move variables stored in volatile memory (RAM) into the EEmem (EPROM) is demonstrated in the 'dim_vars_here' bubble of the application project. 'dim_code' is where the variables of the program are defined. When writing application code, organize the variable names according to their section or usage definition this allows for a 'cleaner' program that is easier to follow when debugging.

Bubble '2' located in the logic sequence "dim_vars_here" copies the data located in the PLM700 and EEmem variables to a variable located in the program. Bubble '3' sets a timer EEmem_update_timer for 5 seconds. At the end of the 5 seconds, the program pointer transfers into logic bubble '10'. A sample of the application code from bubble '10' is shown below;

```
*****Analog Input 1 logic*****  
'min volts Analog 1  
if (AI1_minVolts_cal_PLM_IV <> Analog_In1.MinVolts) then  
    AI1_minVolts_cal_PLM_IV = Analog_In1.MinVolts  
    AI1_minVolts_cal_eem = Analog_In1.MinVolts  
    state = 100  
end if  
  
if (AI1_minVolts_cal_EEM_IV <> AI1_minVolts_cal_eem) then  
    AI1_minVolts_cal_EEM_IV = AI1_minVolts_cal_eem  
    Analog_In1.MinVolts = AI1_minVolts_cal_eem  
    state = 101  
end if
```

The first two lines are comments as they began with a " ". Comments should be added to programs to aid the programmer in troubleshooting and modification.

The first 'if' statement compares the variable, "AI1_minVolts_cal_PLM_IV", with the variable "Analog_In1.MinVolts". If these two variables are not equal, then the program goes to the next line where the value of Analog_In1.MinVolts is set into AI1_minVolts_cal_PLM_IV. The program then goes to the next line and sets the value Analog_In1.MinVolts, into the variable, AI1_minVolts_cal_eem. Next the program executes the next line and sets the variable 'state' to the value of 100. The contents of the variable 'state' could be monitored by the virtual display to tell the operator that this statement was executed. If the variables in the 'if' statement were equal, the lines below it would not have been executed.

The next 'if' statement shown above compares variables 'AI1_minVolts_cal_EEM_IV' and 'AI1_minVolts_cal_eem'. If these variables are not equal, this piece of logic sets the value of the variable "AI1_minVolts_cal_EEM_IV" stored in RAM so that it can be written to the EPROM the next time the EPROM is written to.

The last command of bubble '10' is EEcommand = EEwrite. This command instructs the BIOS to compare the value of the variables currently stored in RAM memory with the actual EEmem values stored in the EPROM. If these two values are not equal, the BIOS will reload the EPROM with the correct values.



Bubble '5' is executed only when the EEmem variable 'ee_memory_status' is not equal to the value 10318. When executed, it will reset all values in EEmem factory default values. The Variable 'ee_memory_status' is defined in bubble '2'. It is a number that can be set to the date of modification, in this instance 08208, august 20, 2008. The value of "ee_memory_status" is tested in the transition to bubble '5'. If the variables don't match, a transition to bubble '5' is called. In bubble '5', the value 08158 is set into 'ee_memory_status' and the EEmem variables are reset to their defaults.

The Enable_valve_1 Logic Sequence contains the Output logic. Bubble '2' contains logic that enables the Output groups and drives the outputs. A sample of this code is shown below;

```
Valvecoil1.enable = Module_Enable 'eememory variable, enabled from operator input
Valvecoil2.enable = Module_Enable
Valvecoil3.enable = Module_Enable
```

```
Valvecoil1.dir = Analog_In1.dir
Valvecoil2.dir = Analog_In2.dir
Valvecoil3.dir = Analog_In3.dir
*****
```

'allows analog inputs to control the appropriate valve coils

'set input of graph to Analog input

```
valve_shaper_1.in = Analog_In1
valve_shaper_2.in = Analog_In2
valve_shaper_3.in = Analog_In3
```

' set valve to output of graph

```
Valvecoil1 = valve_shaper_1.out
Valvecoil2 = valve_shaper_2.out
Valvecoil3 = valve_shaper_3.out
```

In this bubble there are three lines of code that send the values of the analog inputs to the individual I/O function curve inputs. Then the output of the I/O function curve is set to the outputs demand variables. Remember, the variable on the right hand side of the '=' is set into the variable on the left hand side of the '='.

The outputs will not be driven until the eemem variable 'Module_Enable' is set to a value '> 0'. This can be accomplished by using the PLM700. Start the PLM700; select the EE-MEMORY indicator to open the window that displays all the DVC710 eememory variables. The third variable is 'Module_Enable'. When the controller is initially loaded at the factory, the enable variable will be set to '0'. Modify the value by setting the variable to any number between 1 and 65535. Select the button 'Save Changes' in the lower left hand corner of the screen, this will write the new value to the EPROM.

The EEmem variable, 'ee_system_startup_delay_time', is factory set at 30 seconds. This value of 30 will cause a 30 second startup wait time from when the controller is first started, until the output of the valves are enabled. This time can be modified by changing the value of 'ee_system_startup_delay_time' in the EEmem screen. This timer is set in bubble 2 of 'dim_vars_here' logic sequence.

NOTE:

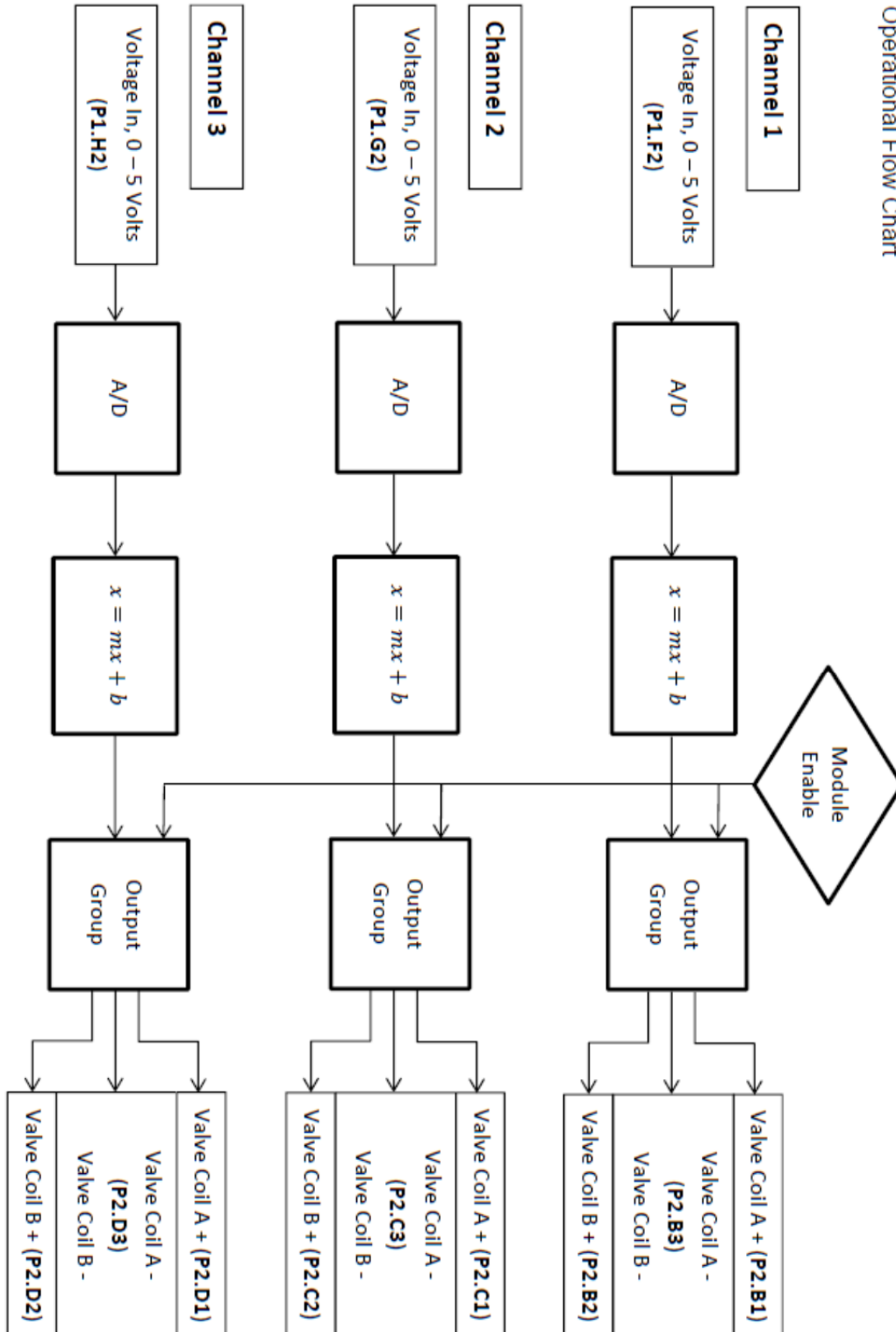
The demonstration application code is pre-loaded into the DVC710 unit and allows the user to apply power and ground, and connect up to three potentiometers or external signal inputs to control up to 3 dual coil outputs.

If you require more information, would like to learn more, book a training class or order more product or accessories, please visit our website at: www.hctcontrols.com

Or call our California based customer service line at: 1 530 265 3236



Operational Flow Chart





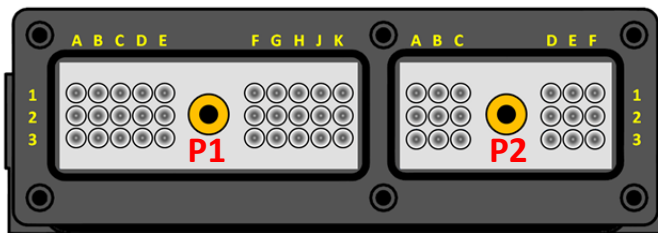
DVC 710 Connector Designation Tables

P1 - 30 Pin Metri-Pak Connector (Male, Plug)

PIN	Name	Function	PIN	Name	Function	PIN	Name	Function
A1	RXD	Receive RS232 Data - Pin 'A'	D2	Uni-2	Universal Input #2	G3	GND	Ground / 0V / Signal Common
A2	TXD	Transmit RS232 Data - Pin 'C'	D3	GND	Ground / 0V / Signal Common	H1	Dig3	Digital Input #3
A3	RTS	Request To Send - RS232 Pin 'D'	E1	Vref	+5V regulated user output	H2	Ana3	Analog input #3
B1	CAN-H	CAN #1 High	E2	Uni-3	Universal Input #3	H3	GND	Ground / 0V / Signal Common
B2	CAN-L	CAN #1 Low	E3	GND	Ground / 0V / Signal Common	J1	Dig4	Digital Input #4
B3	COM	0V/COM RS232 Data - Pin 'B'	F1	DIG-1	Digital Input #1	J2	Dig5	Digital Input #5
C1	CAN-H	CAN #2 High	F2	ANA-1	Analog input #1	J3	Dig6	Digital Input #6
C2	Uni-1	Universal Input #1	F3	GND	Ground / 0V / Signal Common	K1	+Pwr	+V Supply Power Input
C3	GND	Ground / 0V / Signal Common	G1	DIG-2	Digital Input #2	K2	Dig7	Digital Input #7
D1	CAN-L	CAN #2 Low	G2	ANA-2	Analog input #2	K3	Dig8	Digital Input #8

P2 - 18 Pin Metri-Pak Connector (Male, Plug)

PIN	Name	Function	PIN	Name	Function	PIN	Name	Function
A1	+VPwr	+V Supply Power Input	C1	HS-3	ON/Off output	E1	Gnd for DVC	Ground / 0V Common
A2	+VPwr	+V Supply Power Input	C2	HS-4	ON/Off output	E2	Not Used	Not Used -
A3	Not Used	Not Used -	C3	PWM-2	PWM Output #2	E3	Not Used	Not Used -
B1	HS-1	ON/Off output	D1	HS-5	ON/Off output	F1	Gnd for DVC	Ground / 0V Common
B2	HS-2	ON/Off output	D2	HS-6	ON/Off output	F2	Gnd for DVC	Ground / 0V Common
B3	PWM-1	PWM Output #1	D3	PWM-3	PWM Output #3	F3	Gnd for DVC	Ground / 0V Common



- ALWAYS use a suitable fuse in the module supply connection for protection and to maintain warranty eligibility.
- Ensure connectors are pushed fully together and that BOTH lock screws are tightened to ensure correct and continuous connections.
- RS 232 connection is only used when connected to the user PC and not for normal use.

