DevicePlugin

Programmer's reference

IO interfaces

Revision 2017.11.24

1 Description

This document describes standard command-set of IO plugins:
- Advantech IO (AdvantechIo)
- Papouch Quido/RS (QuidoIo)
- FPC IO (FpcIoV2)
- FPC Relay Boards (FpcRelayBoads)
- Measurement Computing (MeasComp)

2 Advantech IO (AdvantechIo)

Control of Advantech PCI(E)-1730 digital isolated IO cards.

This plug-in requires the **DAQNavi** drivers, which could be downloaded from Advantech website:
- for PCI-1730 (32-bit PCI slot)
- for PCIE-1730 (PCI-Express x1 slot)

3 Papouch Quido/RS (QuidoIo)

Control of Papouch Quido/RS IO interfaces.

Connection over RS232, no software requirements.

4 FPC IO (FpcIoV2)

Control of FPC low-cost USB interfaces 8I8O (8 inputs and outputs) and 16I16O (16 inputs and outputs).

Requirements:

- FTDI D2XX drivers to be installed

5 FPC Relay Boards (FpcRelayBoard)

Control of FPC relay boards:
- CFPC-108, 40 relays, **2A** per relay
- CFPC-138, 40 relays, **6A** per relay

Connection is over RS-232. Because of that, these boards have no software requirements.

Note:

These boards have outputs only! Input commands will not work.

6   Measurement Computing (MeasComp)

Control of Measurement Computing DIO cards.

Supported cards:

- PCIe-DIO96H (PCI-Express x1, 96 TTL IO channels)

6.1   Installation

Card must be properly installed and configured in the system. Drivers and the InstaCal utility is required.

Drivers and utility can be downloaded here:
ftp://ftp.mccdaq.com/downloads/InstaCal/

After drivers are successfully installed, cards in the system must be initiated via the InstaCal utility:

InstaCal utility
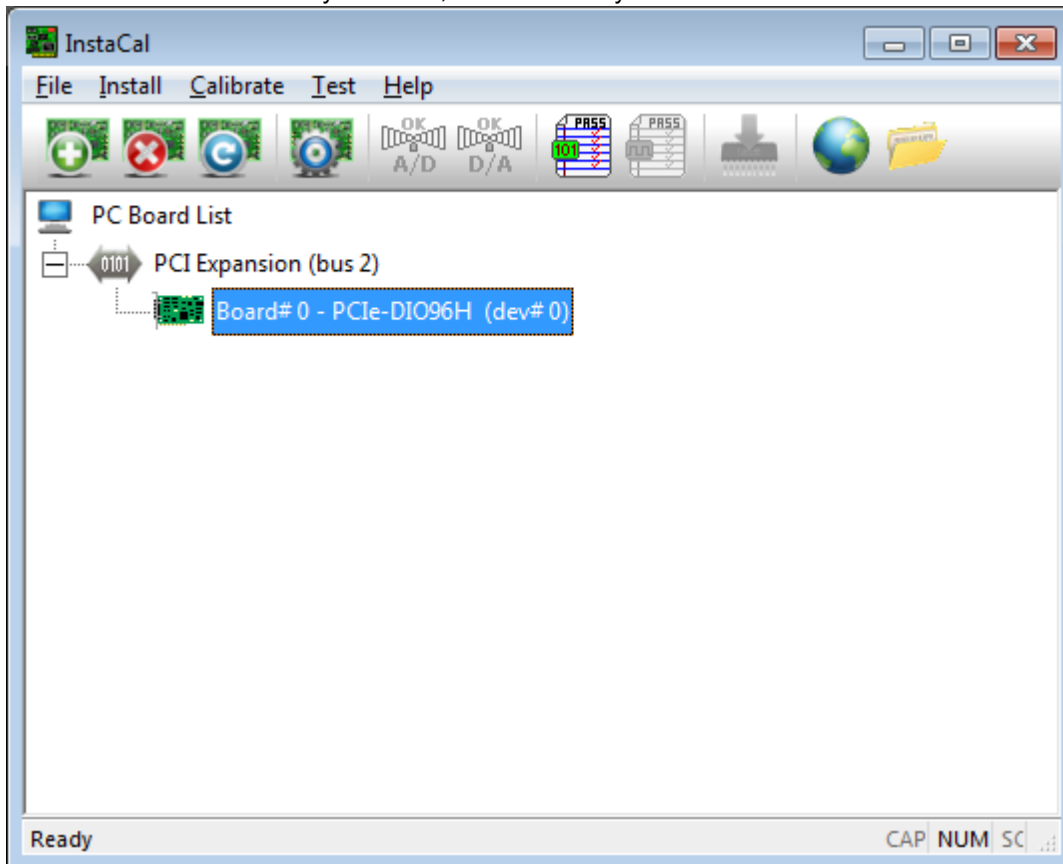
Click "Refresh" button/menu item to refresh installed cards. This will enable cards to be detected by the MeasComp plugin.

Right click on specified board, then "Configure" will open the dialog to set up pull-up or pull-down internal resistor on DIO ports. This configuration is persistent even after restart the PC.
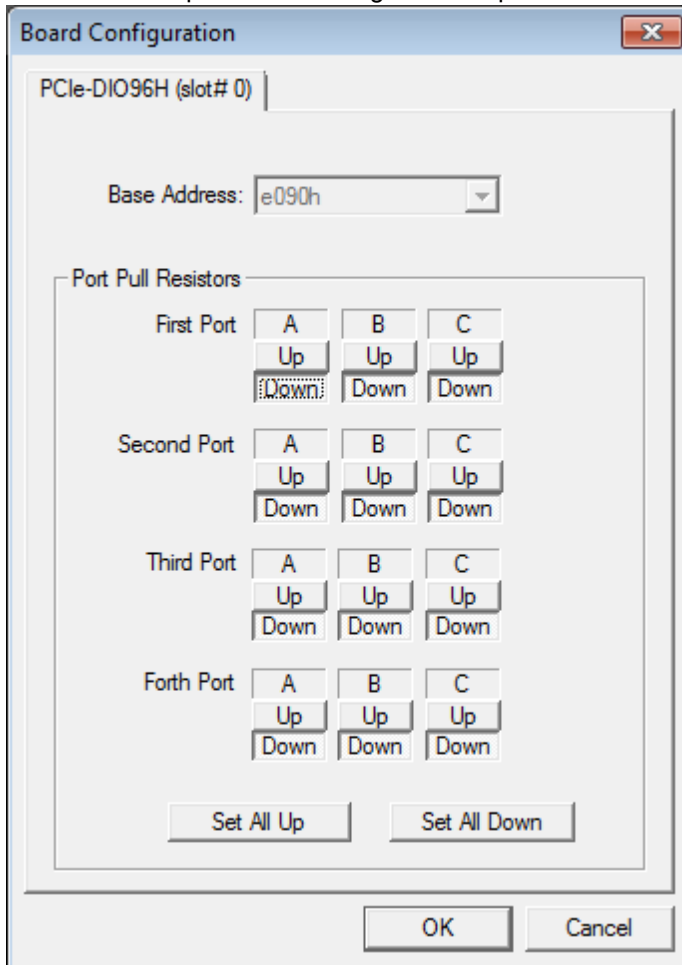
Board configuration - pull-up/down resistors

7 Standard commands

Commands to control IO interface. **Chaining of commands is supported.** This allows to pass more commands to IO interface at once.

Chaining example:

```
s: 0: d: 100: r: 1: d: 10: c: 0
```

Will do:
- Set output 0
- Delay for 100 ms
- Read state of input 1
- Delay for 10 ms
- Clear output 0

7.1 *idn? (Identification)

```
*idn?
```

Gets the plug-in identification string.

Parameters

No parameters.

Return value

The identification string in standard format "`<company>`, `<product/name>`, `<serial-no>`, `<version>`".

7.2 s (Set outputs)

```
s: <out_0>{: <out_1>: ...: <out_N>}
```

Set (switch ON) specified output(s).

Parameters

out   [int]   Number of output signal in range of 0 to N.

Return value

No return value.

Examples

`s: 0: 5: 12`
Set outputs 0, 5 and 12 to ON.

7.3 c (Clear outputs)

```
c: <out_0>{: <out_1>: ...: <out_N>}
```

Clear (switch OFF) specified output(s).

Parameters

out   [int]   Number of output signal in range of 0 to N.

Return value

No return value.

Examples

`c: 0: 5: 12`
Set outputs 0, 5 and 12 to OFF.

```
c: *
c: all
ca
```

Clear all output(s).

Parameters

No parameters.

Return value

No return value.

7.4   r (Read inputs)

```
r:<in_0>{:<in_1>:...:<in_N>}
```

Read current state of specified input(s).

Parameters

in	[int]	Number of input signal in range of 0 to N.

Return value

"0" when input is OFF
"1" when inputs is ON

If there is more than 1 input to read, the return value contains all input states (0 or 1), separated by a colon (":").

Examples

```
r:7
```
Read state if input number 7, return value will be for example "1" (ON).

```
r:2:8:11
```
Read state if inputs 2, 8 and 11, return value will be for example "$1:0:1$" (2 = ON, 8 = OFF, 11 = ON).

```
r:all
```

Read current state of all inputs.

Parameters

No parameters.

Return value

N states ("0" or "1") in **ordered by input N to 0** (MSB first), separated by a colon (":").

7.5   d (Delay)

```
d:<delay>
```

Put a delay between sub-commands in the sequence.

Parameters

delay	[int]	Delay in milliseconds

Return value

No return value.

Examples

```
s:0:d:250:c:0
```
Set output 0 for 250 ms (make a pulse).

8 MeasComp commands

8.1 conf (IO configration)

Measurement Computing cards have IO channels which can act like IN or OUT. There is no section of separated IN pins and OUT pins. Function (IN/OUT) of pins must be configured before use in the application. This configuration is not persistent, it must be done always at application (project) startup. Default function is IN.

PCIe-DIO96H

There are total of 16 ports with groups of 8 pins and 4 pins, giving total of 96 IO channels. Only whole port can be configured like IN or OUT, it is not possible to configure each pin separately.

Ports:
- `FIRSTPA` (8)
- `FIRSTPB` (8)
- `FIRSTPCL` (4)
- `FIRSTPCH` (4)
- `SECONDPA` (8)
- `SECONDPB` (8)
- `SECONDPCL` (4)
- `SECONDPCH` (4)
- `THIRDPA` (8)
- `THIRDPB` (8)
- `THIRDPCL` (4)
- `THIRDPCH` (4)
- `FOURTHPA` (8)
- `FOURTHPB` (8)
- `FOURTHPCL` (4)
- `FOURTHPCH` (4)

Commands can be combined at one time. It is possible for example set all port like IN and then selected like OUT using one merged command - see example of out below.

```
conf: in: *
conf: in: port_0{: port_1}..{: port_N}
```

Set all (*) or specified ports like IN.

Parameters

`port`	[string]	Port(s) to be set like IN, use port names (case-insensitive) written above ("firstpa", "firstpb"...).

Return value

No return value.

Examples

```
conf: in: secondpa: thirdpb: fourthpcl
```
Set 3 ports like IN.

```
conf: out: *
conf: out: port₀{: port₁}..{: port_N}
```

Set all (*) or specified ports like OUT.

Parameters

`port`	[string]	Port(s) to be set like OUT, use port names (case-insensitive) written above ("firstpa", "firstpb"...).

Return value

No return value.

Examples

`conf: out: secondpa: thirdpb: fourthpcl`
Set 3 ports like IN.

`conf: in: *: out: firstpa: secondpch`
Set all ports like IN, then immediately 2 ports like OUT.