CHAPTER

6

SYSTEM SETUP AND DIAGNOSTICS

Preliminary Concepts:

Host Computer	A Windows 2000/XP PC running *Motion* Perfect 2.
Motor	A tuned servo drive / motor configuration for a servo axis or a stepper motor and drive combination
Prompt	When the PCI 208 is ready to receive a new command, the prompt >> will appear on the left hand side of the current line in the “terminal” under the “tools” menu of Motion Perfect
Axis Parameters	Can be written to or read from. For example the proportional gain of a servo axis has the name **`P_GAIN`**. It can be written to: **`P_GAIN=0.5`** or read from: **`PRINT P_GAIN`**. For further information see chapter 9.

System Setup

A control system should be treated with respect as careless or negligent operation may result in damage to machinery or injury to the operator. For this reason the setting up of the system should not be rushed. This section describes a methodical approach to system configuration and is designed to gradually test each aspect of the system in turn, finally resulting in the connection of the motor. If followed cautiously no unexpected situations should arise.

In cases where the setup procedure for servo and stepper systems differ a separate description is provided for each. In multiple axis systems it is advantageous to set up one axis at a time. The following procedure applies both to all *Motion Coordinator* modules.

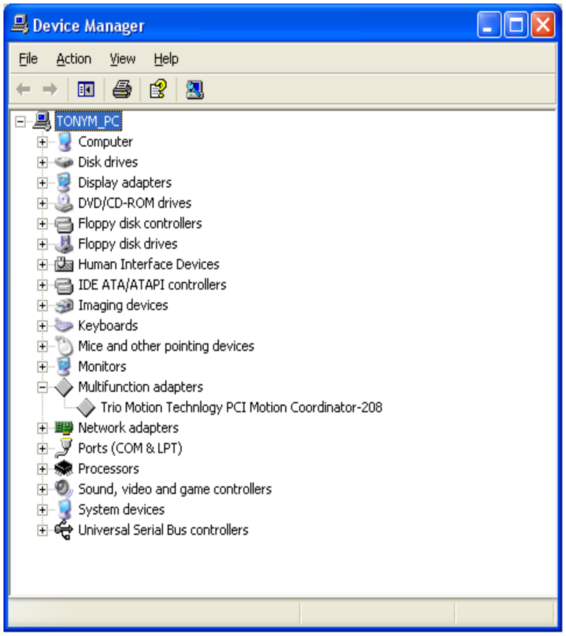
Note:

It is recommended that this section is read in full before attempting to operate the system for the first time.

Preliminary checks

All wiring should be checked for possible misconnection and integrity before any power is applied.

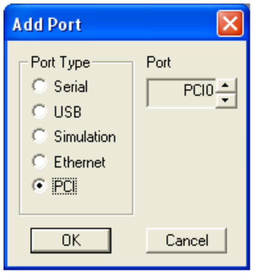
- Install *Motion* Perfect version 2.2.1.10 or higher.
- Install TrioPC Active X version 1.1.0.0 or higher.
- Install the PCI 208 in a free PCI slot with the PC power off.
- Switch on the PC and check PCI 208 is recognised by the PC as “Trio Motion Technology PCI Motion Coordinator 208”. The PCI 208 will be found under “Multifunction Adapters” under System Properties->Hardware->Device Manager.



TONYM_PC
Computer
Disk drives
Display adapters
DVD/CD-ROM drives
Floppy disk controllers
Floppy disk drives
Human Interface Devices
IDE ATA/ATAPI controllers
Imaging devices
Keyboards
Mice and other pointing devices
Monitors
Multifunction adapters
Trio Motion Technlogy PCI Motion Coordinator-208
Network adapters
Ports (COM & LPT)
Processors
Sound, video and game controllers
System devices
Universal Serial Bus controllers

- Run *Motion* Perfect and under the “Options” menu click “Communications”. Press “Add” and add a PCI port type.

- Delete the default COM1 port type unless you intend to use *Motion* Perfect with serially connected *Motion Coordinator*’s.

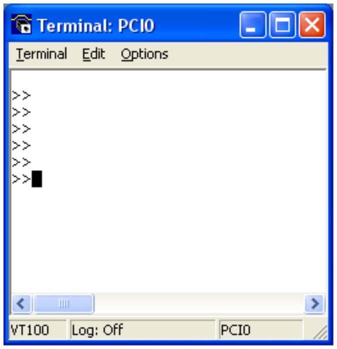


Port: PCI0

Port Type: [✓ PCI]

Checking Communications and System Configuration

- Open a terminal window in *Motion* Perfect and PCI 208 should respond with a >> prompt when the “Return” key is pressed.



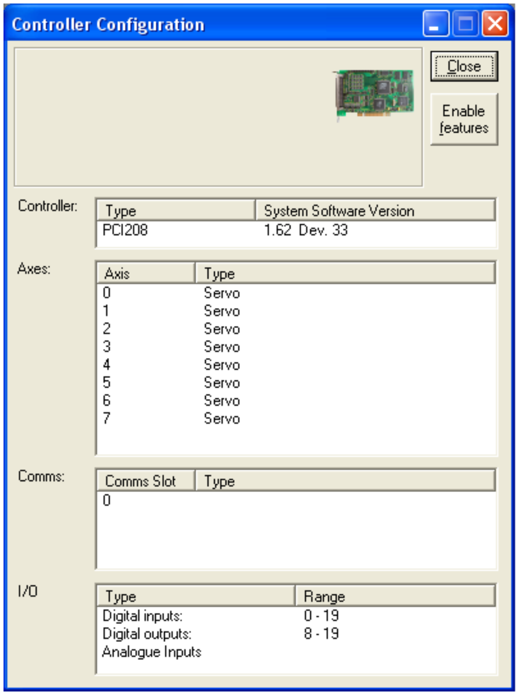
```
>>
>>
>>
>>
>>
>>
```

- Click “Connect” under the “Controller” menu.
- If this is the first time you have connected you will need to select the “New Project” option when *Motion* Perfect tries to ensure that your “Project” on the controller matches its copy on disk.
- When the “Project Consistent” message is received in the “Check Project” window you know:

  1. *Motion* Perfect has made a serial connection between your PC and the PCI 208.
  2. *Motion* Perfect has an exact copy of the programs on the PCI 208.

- The controller hardware configuration can now be checked using the “Controller configuration” option under the “Controller” menu. *Motion* Perfect draws a graphical representation of your system, as shown below.

Example:



Controller:

Type	System Software Version
PCI208	1.62 Dev. 33

Axes:

Axis	Type
0	Servo
1	Servo
2	Servo
3	Servo
4	Servo
5	Servo
6	Servo
7	Servo

Comms:

Comms Slot	Type
0	

I/O

Type	Range
Digital inputs:	0 - 19
Digital outputs:	8 - 19
Analogue Inputs	

This message would be produced by a *Motion Coordinator* PCI 208 with the following configuration:

- System Software version 1.62
- Axes 0..7 set to servo operation using feature enable codes.
- No CAN I/O modules are connected.

Check that the system description corresponds with the modules that are actually present. If this is not so, check the CANBus and feature enable codes and the settings of the address switches on any CAN or axis expander modules.

Input/Output Connections:

- Check each of the 24v input connections with a meter then connect them to the controller.
- Test each of the input channels being used for correct operation in turn. These may be easily viewed in the I/O window. Use “IO Status” under the “Tools” menu.
- Switch each output being used in turn for correct operation. These may be easily set with the IO status window.

Connecting a Servo Motor to a P181 Breakout Board

Note: ***This description assumes the motor / drive combination has been already tested and is functioning optimally.***

Each servo axis should be connected in turn to the X1..X8 encoder pins.

- With the servo drive off or inhibited <u>connect the motor encoder only</u> (or the encoder emulation output from the servo drive).
- Check the encoder counts both up and down by looking at the measured axis position `MPOS` in the Axis parameter window of *Motion* Perfect (“Axis parameters” under the “Tools” menu) whilst turning the axis by hand.
- Ensure the `SERVO` axis parameter is set OFF (0) in the Axis parameter and that the `DAC` axis parameter is set to 0. It may be necessary to use the scroll buttons to view these parameters. This will force 0 volts out of the +/-10v output for the axis. Now connect the servo drive to the V+/V- connections.
- Enable the servo drive by clicking the “Drives disabled” button on the control panel. If the axis runs away the motor/drive combination must be re-checked. (Note: clicking “Drives disabled/ Drives enabled” is equivalent to issuing a `WDOG=ON` or `WDOG=OFF` command).
  The servo motor should now be powered and is likely to be creeping in one direction as the position servo has been switched OFF.
- Set a small positive output voltage by setting `DAC=25`. The motor should then move slowly forward - Check the encoder is counting up by looking at the `MPOS` axis parameter. If this is correct check that the motor reverses and the encoder counts down when `DAC=-25`.
- If the encoder counts down when a positive `DAC` voltage is applied. The motor or position feedback needs to be reversed.

  This can be achieved by:

  - Swap A and /A connections on the encoder input, or

- Swap BOTH motor terminals and tacho terminals (DC motors only!) On many digital brushless motors the direction can be reversed by a drive setting, or
- If the drive has differential inputs, reverse the voltage as it enters the drive. (This can cause problems with some servo-drives. The VIO GND pins of the Breakout board/PCI 208 are internally connected inside the *Motion Coordinator* so the axis voltage outputs cannot float relative to each other), or
- set a negative **PP_STEP** axis parameter. (This is not possible using SSI encoders)
- Set a negative **DAC_SCALE** axis parameter. (Trio recommend wiring motors consistently and not using a negative DAC_SCALE to correct wiring errors)

We are now ready to apply the position servo as described below.

Setting Servo Gains

The servo system controls the motor by constantly adjusting the voltage output which gives a speed demand to the servo drive. The speed demand is worked out by looking at the measured position of the axis from the encoder comparing it with the demand position generated by the *Motion Coordinator*.

The demand position is constantly being changed by the *Motion Coordinator* during a move. The difference between the demand position (Where you want the motor to be) and the measured position (Where it actually is) is called the following error.

The controller checks the following error typically 1000..4000 times per second and updates the voltage output according to the “servo function”. The *Motion Coordinator* has 5 gain values which control how the servo function generates the voltage output from the following error.

Default Settings:

Gain	Parameter Name	Value
Proportional Gain	`P_GAIN`	1.0
Integral Gain	`I_GAIN`	0.0
Derivative Gain	`D_GAIN`	0.0
Output Velocity Gain	`OV_GAIN`	0.0
Velocity Feedforward Gain	`VFF_GAIN`	0.0

A simple test program can be used to generate movement to and fro for examination of the motion profile generated on an oscilloscope. The oscilloscope should be connected to the tacho or velocity output from the servo drive.

Example:

```
PRINT "Enter Axis Number ":INPUT VR(0)
BASE(VR(0))
SPEED=20000
ACCEL=200000
DECEL=200000
loop:
  MOVE(1000)
  WAIT IDLE
  WA(100)
  MOVE(-1000)
  WAIT IDLE
  WA(100)
GOTO loop
```

The editor built into *Motion* Perfect may be used to enter the test program. Click on Program, New from the pull down menus and enter a program name, replacing the name **`UNTITLEDx`**. Now click on the EDIT button and an edit window will be opened where the program shown above may be typed in. See the *Motion* Perfect section for more details on how to use the editor. Once the program is entered, it can be run by clicking on the red button next to its name or the RUN button in the Controller Status panel.

The servo gain parameters may be set to achieve the desired response from the servo system. The desired response can vary depending on the type of machine.

Different gain settings can be used to obtain:

Smoothest motor running

This can be achieved by using low proportional gain values, adding output velocity gain adds smoothing damping at the expense of higher following errors.

Low following errors during complete motion cycle

This can be achieved by using velocity feed forward to compensate for following errors together with higher proportional gains.

Exact achievement of end points of moves

This can be achieved by using integral gain in the system together with proportional gain. However overshoot will occur at the end of rapid deceleration.

Typically a combination of the above is required.

Note: ***The system should be set with proportional gain alone firstly starting with the default value of 1.0   The other gains should then be introduced if necessary according to the descriptions which follow.***

Proportional Gain

Description The proportional gain creates an output voltage, Op that is proportional to the following error E.

$$Op = Kp \times E$$

Axis parameter is called `P_GAIN`

Syntax: `P_GAIN=0.8`

Note: ***All practical systems use proportional gain, many use this gain parameter alone.***

Integral Gain

Description The Integral gain creates an output Oi that is proportional to the sum of the errors that have occurred during the system operation.

$$Oi=Ki \times SE$$

Integral gain can cause overshoot and so is usually used only on systems working at constant speed or with a slow acceleration.

Axis parameter is called `I_GAIN`

Syntax: `I_GAIN=0.0125`

Derivative Gain

Description This produces an output Od that is proportional to the change in the following error and speeds up the response to changes in error whilst maintaining the same relative stability.

$$Od = Kd \times DE$$

This gain may create a smoother response. High values may lead to oscillation.

Axis parameter is called `D_GAIN`

Syntax: `D_GAIN=5`

Output Velocity Gain

Description This increases the system damping, creating an output that is proportional to the change in measured position.

$$Oov = Kov \times DPm.$$

This parameter can be useful for smoothing motions but will generate high following errors. Note that a NEGATIVE OV_GAIN is required for damping.

Axis parameter is called **OV_GAIN**

Syntax: **OV_GAIN=-5**

Velocity Feed Forward Gain

Description As movement is created by following errors at high speed the following error can be quite appreciable. To overcome this the Velocity Feed Forward creates an output proportional to the change in demand position so creating movement without the need for a following error.

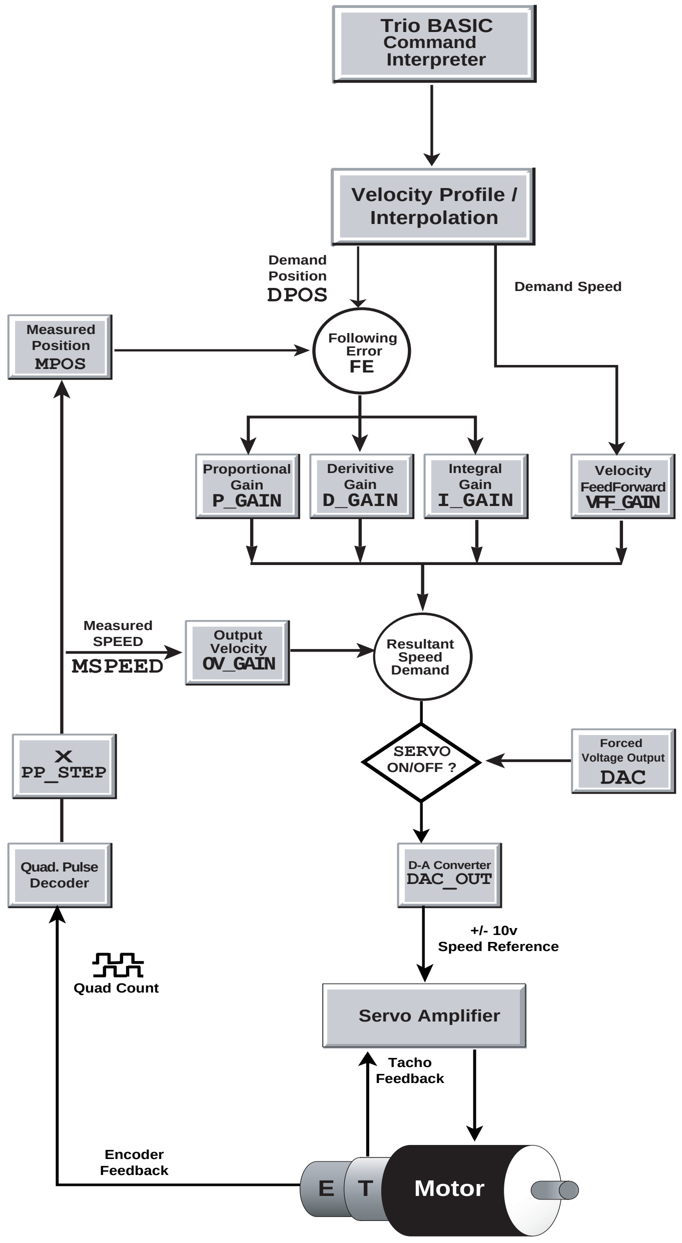
$$Ov = Kvff \times DPd$$

Axis parameter is called **VFF_GAIN**

Syntax: **VFF_GAIN=10**

The VFF_GAIN parameter can be set by minimising the following error at a constant machine speed AFTER the other gains have been set.

Servo Loop Diagram



Diagnostic Checklists

Problem	Potential reasons
Motion Perfect flashes “Enable” button	• Following error on at least one axis. The axis demand position and measured position exceed the programmed limit
Motor runs away without issuing a move command	• tacho/drive polarity • encoder/controller polarity • gains (drive and/or controller)
Motor runs away upon issu-ing a move command	• tacho feedback • encoder feedback • gains (drive and/or controller)
Motor does not move upon issuing a move command	• wiring (enables/inhibits/limits on drive and controller) • check status on all axes • drive power • feedhold applied • speed, acceleration and/or deceleration set to zero • servo set off/watchdog set off • gains (drive and/or controller) • axis is already running a move which has not completed - Check MTYPE and NTYPE
Axis goes out on following error after a time	• speed being requested requires more than 10v - check drive tacho gain and motor/ drive speed characteristics. • drive shutting down on current limit after a time

Problem	**Potential reasons**
Axis losing position	• encoder coupling • encoder signal (wire length, differential/single ended encoder) • mechanics
Motion Perfect cannot "connect" with the control-ler	• Controller running a program which transmits on serial channel 0. If this prevents *Motion* Perfect connecting to the controller, open Terminal screen in *Motion* Perfect unconnected mode and type a "halt" command at the command prompt. • *Motion* Perfect is not set to use the PCI port: Set under "options" menu • Check *Motion* Perfect version. The latest version can be downloaded from `www.triomotion.com`