



— The bright side of industry automation

# **PWM LabView Driver Manual**

© 2000–2007 OLTOM Engineering AB

Email: [support@oltom.com](mailto:support@oltom.com)

Revision 1.2

# Introduction to the environment

The PWM3 stepper motor card supports a number of different operating systems and programming languages. The main design goal for the drivers delivered by OLTOM is ease of use.

A problem with all communication, for example RS-232, is the risk of corrupted messages. A garbled command message could be fatal to a system. Take for example a system where a motor steps 1000 instead of 100 steps and causes a production line to stop.

The PWM3 communication protocol is sophisticated with several security features. Emphasis is placed on high reliability. All packets have data security features, and the sender receives an explicit acknowledge for all messages. This handling also enables the ability to reliably address cascaded PWM3 cards through the I2C bus.

The LabView development environment from National Instruments is a graphical, user friendly, programming environment for development of automation and measurement systems on ordinary PCs. LabView programming uses a concept called Virtual Instruments (VI). The PWM3 LabView Driver is delivered with a number of VIs that are easy to use and understand.

## Installation of the Driver

Installation of the PWM3 LabView driver from OLTOM is easy. Insert the provided CD and double-click the "Install" icon.

The driver will now install itself on your system automatically, and after this the driver will be available in LabView.

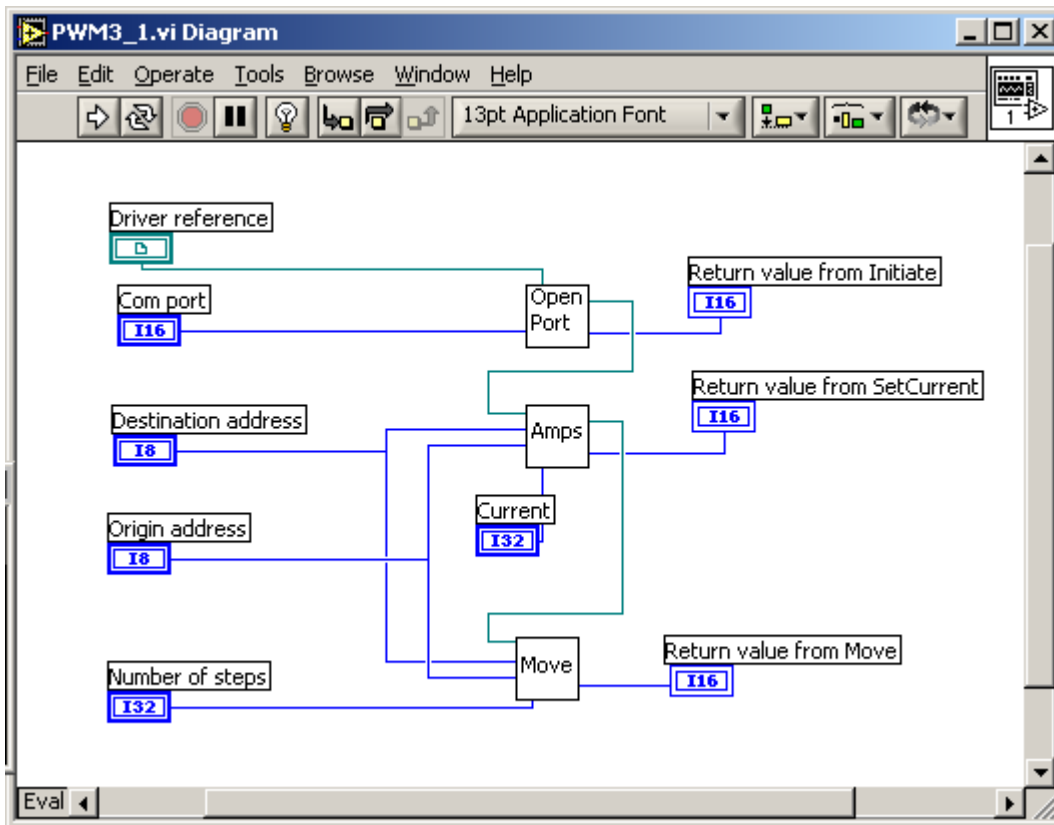
Now, take the provided VI's and copy them from the CD to a convenient place in your computer. Include the VI's in your projects, or open them and learn how to use the driver functions directly. It's all up to you!

The PWM3 windows driver can be installed on a Windows 95, 98, 2000, ME, XP or NT (Service Pack 3 or later) PC. You should be logged on as administrator. The driver takes less than 5 Mbytes of storage space on your computer. The driver is tested on LabView versions from 5.1 to 6i but should work on earlier versions that support ActiveX components.

## The LabView PWM3 VI's

On the CD there are a number of LabView VI's that take care of the more complex handling of the driver functions for you. These VI's make the control of the PWM3 cards very easy. Every VI has a built in Context Help as supported by LabView.

Before the commands can be sent to the hardware the PWM3 driver software must be activated. To be able to run the card a communication port must also be opened. Since these two functions are vital they are incorporated into the same VI. This VI is called "PWM3 Open Port.VI". Always start with this VI. From this VI all "driver reference" wires must originate. The picture below shows an example of a minimal application.



This is the diagram of one of the example files that is delivered with the Driver and VI's. This is the most basic of the examples, and can be used as a base for more complex systems.

There are eight VI's delivered with the LabView driver. There are also two example files that use some of these VI's, enough for any basic level LabView user to get started.

The different VI's are:

- **PWM3 Open Port.VI** This starts the connection to the PWM3 Driver software and also opens a communication port. Always start with this.
- **PWM3 Current .VI** This VI sets the current to the motor that have been addressed. This is set in deciampère. Example: Setting 100 gives a current of 1.0 Ampere.
- **PWM3 Move.VI** Use this VI to get the addressed motor to start moving. Negative values for one direction and positive values for the other, depending on the motor wiring. This is the only command that starts a movement. Use the Speed and Ramp commands to set up the movement, *before* starting the movement with PWM3 Move.VI.
- **PWM3 Speed.VI** This Vi sets the maximum speed of the addressed motor. The value should be in steps/second. Since this is a digital system with a discrete resolution all values are not exactly resolvable. However, since the resolution of the system is 51.2µs misalignment from the desired value will not be large. The system calculates the closest possible value of the desired value. To be able to ensure an exact speed calculation for the user the VI will return with the used value in 51.2µs ticks. Example: If the function returns with a value of 100 the step period will be 100\* 51.2µs. Note that this command should be used *before* the Move command. The VI returns 0 if the command was received and executed.
- **PWM3 Ramp.VI** With this function the acceleration and retardation ramp will be set. The value is in steps/second<sup>2</sup>. Note that this command should be used *before* the Move command. The function returns 0 if the command was received and executed without error.
- **PWM3 Inquiry.VI** This function sends an inquire command to check if the addressed motor is busy moving. This function returns 0 if the motor is not busy.
- **PWM3 Reset.VI** This VI will reset the addressed card to its power on state. Note that this

means that the power will be set to the default power on value of 0.3 A.

- **PWM3 Close.VI** This VI closes the COM port. Only necessary to use if you want to use the selected port during execution of your LabView program. If you close your program the port will automatically be available for other applications.

# The Available Functions in the driver

If you decide that you want to create your own sub VI's directly connected to the driver, this is possible, and fairly easy. The driver gives the advanced user the possibility to use its methods directly.

The PWM3 LabView driver is an ActiveX component that is especially adapted to suit the LabView development environment. There are a number of different functions that can be used directly; you can see how it works, and where to find them, inside the VI's. Below is an explanation for each of the functions, their parameters and return values.

## **SendMove(I8 DestAdr, I8 OriginAdr, I32 Data)**

This function sends a move command to start a stepping movement on motor **DestAdr**. The number of steps is put in **Data** and can be 24 bits long. The value is negative (two complements) for one direction and positive for the other. The function returns 0 if the command was received and executed.

## **Initate(I8 Port)**

Opens COM port number **Port**. The function returns 0 if the port was opened.

## **ClosePort(I8 Port)**

Closes COM port number **Port**. The function returns 0 if the port was closed.

## **Inquiry(I8 DestAdr, I8 OriginAdr)**

This function sends an inquire command to check if motor **DestAdr** is busy moving. The function returns 0 if the motor is idle.

## **ResetCard(I8 DestAdr, I8 OriginAdr)**

The card that has the address **DestAdr** will reset to power on state.

## **SetCurrent(I8 DestAdr, I8 OriginAdr, I32 Data)**

The current on motor **DestAdr** will be set. The value in data is the current in deciAmpere. Example: 100 in **Data** will give a current of 1.0 Ampere. The function returns 0 if the command was received and executed. The value 1 will be returned if a value larger than allowed is sent.

## **SetSpeed(I8 DestAdr, I8 OriginAdr, I32 Data)**

The Maximum speed of motor **DestAdr** will be set in steps/second in the **Data** parameter.

Since this is a digital system with a certain resolution all values are not exactly resolvable. However, since the resolution of the system is 51.2 $\mu$ s misalignment from the desired value will not be large. This means that the system calculates the closest possible value of the desired value. To be able to ensure an exact speed calculation for the user the **Data** parameter will return with the used value of 51.2 $\mu$ s ticks. Example: If the function returns with a **Data** value of 100 the step period will be 100\* 51.2 $\mu$ s. Note that this command should be used *before* the Move command. The function returns 0 if the command was received and executed.

## **SetRamp(I8 DestAdr, I8 OriginAdr, I32 Data)**

With this function the acceleration and retardation ramp will be set. The value that is used in **Data** is steps/second<sup>2</sup>. Note that this command should be used *before* the Move command. The function returns 0 if the command was received and executed.

# Disclaimer

We believe that the information contained herein was accurate in all respects at the time of printing. OLTOM Engineering AB cannot, however, assume any responsibility for errors or omissions in this text. Also note that the information in this document is subject to change without notice and should not be construed as a commitment by OLTOM Engineering AB.