WinPLC Software Development Kit

This Software Development Kit provides the tools necessary for developing and installing applications in the WinPLC. Please be aware that this kit is a work in progress. We will be posting additions to the kit on our web page, so check back often!

This document takes you through the entire process of interacting with the WinPLC. This includes:

- Installing the WinPLC SDK.
- Using WinPLC Workbench to configure the WinPLC's networking parameters and startup settings. Then use WinPLC workbench to test the I/O modules and programming.
- Using the Microsoft embedded Visual Tools (C/C++ or Visual Basic) environment to create and download a user program to the WinPLC.

System Requirements

- Windows 95 (OSR2), Windows 98 (SE), Windows NT4.0 (Sp5 or later) or Windows 2000 Professional (SP1 or later).
  Windows NT is required to run the Windows CE emulator, but it is not needed for program development.
- Microsoft eMbedded Visual Tools 3.0 as of now, these tools are available as a free download from the Microsoft web site at http://www.microsoft.com/mobile/downloads/emvt30.asp.
- The WinPLC SDK from the CD.
- PC with a functional Ethernet connection.

You'll need to have both TCP/IP and IPX protocols installed on your development PC to perform all of the network setup functions.

Software Installation

Before you can begin creating and downloading programs for the WinPLC, you must install and configure the software development environment. Presently, this means either the C/C++ compiler or the Visual Basic compiler from the eMbedded Visual Tools 3.0.

Visual C/C++:

- Install Microsoft eMbedded Visual Tools 3.0
- Copy the contents of the CD to your hard drive, maintaining the directory structure. If you change this directory, you'll also need to change the references in the sample programs before you can get them to build.
- Run Microsoft eMbedded Visual C/C++.
- Select the Tools->Options menu. Select the "Directories" tab. Add "X:\WplcSDK\Include" to the include files list, replacing "X" with your installation drive. Add "X:\WplcSDK\Lib" to the library list in the same manner.

Visual Basic:

- Install Microsoft eMbedded Visual Tools 3.0
- Copy the contents of the CD to your hard drive, maintaining the directory structure. If you change this directory, you'll also need to change the references in the sample programs before you can get them to build.
- Run the batch file \WplcSDK\Reglib.Bat to configure the necessary registry entries.

Configuring and Testing the WinPLC

The Windows CE operating system only supports the TCP/IP protocol for Ethernet networking. Before any serious work with the WinPLC can begin, you need to assign it an IP Address. We do not assign a static IP address when the units are manufactured, because any address we choose would be wrong for someone. So, out of the box, the WinPLC looks for a DHCP (Dynamic Host Configuration Protocol) Server to get a valid IP address.

For most development environments a DHCP assigned address is a reasonable choice. But in a runtime environment, we strongly urge you to assign a fixed IP address to the WinPLC. You wouldn't want your DHCP lease to expire while the WinPLC is controlling any machinery.

As mentioned before, your development PC will need both TCP/IP protocol and the IPX protocol loaded. The reason for having the IPX protocol loaded is that you'll need it to get the WinPLC into its boot loader to configure the WinPLC's networking parameters. While running the boot loader, the WinPLC only supports the IPX protocol.

Hardware Connections to the WinPLC

If you haven't already done so, install the WinPLC and I/O modules in base and connect it to the correct power source. The POWER led on the WinPLC will glow to confirm that you've got it right so far.

Now let's get connected to the WinPLC. The WinPLC has two interface ports, a serial port and the Ethernet port. The Ethernet connection to the WinPLC is a standard RJ-45, 8-position connector. We recommend that you use only category 5 UTP cables for your Ethernet connection. The serial port is a standard RJ-12, 6-position connector. We'll only discuss using the Ethernet port now. See Appendix A for more information on the serial port.

If you're connecting point to point (directly from your development PC to the WinPLC), you will need a crossover cable. If you're connecting through a hub, you will need a straight through cable – sometimes called a patch cable. While we recommend using commercially made cables, Appendix A does show diagrams of these cables if you'll be making your own.

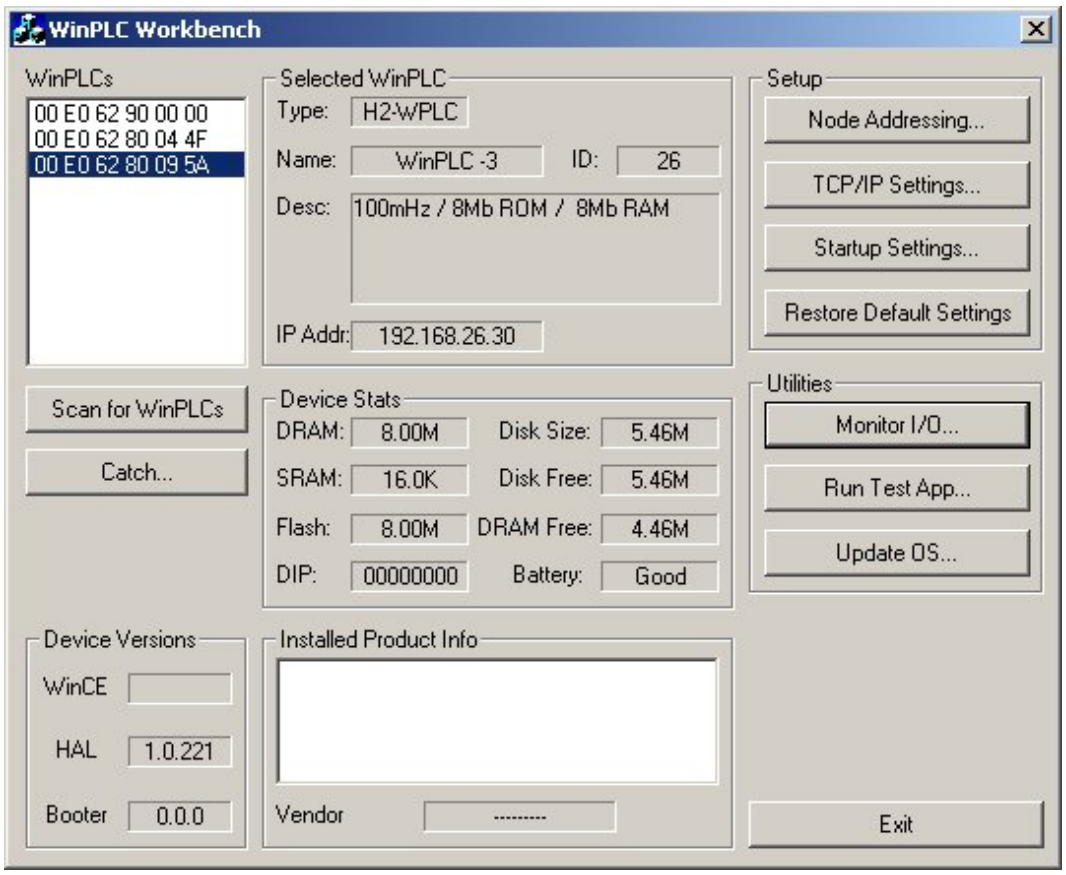
The LINK indicator will glow if the Ethernet connection is correct. If the LINK indicator fails to come on, it's most likely a cabling problem, and this must definitely be resolved before you can continue.

The LINK indicator does double duty as the network activity indicator. It blinks OFF as data is sent and received. Faster blinking means more data flow. An odd side effect of this is that the LED will actually dim as the data rate goes up.

Using WinPLC Workbench

After a successful installation of the WinPLC SDK, you'll have a directory of utility programs named \WplcSDK\Utils. The first of these utility programs we'll use is Winplcwb.Exe, the WinPLC Workbench. This utility is used to do most all of the initial setup and troubleshooting of a WinPLC system. It provides everything from configuring the networking and startup settings to testing the I/O modules and application interface.

When you run the Winplcwb.Exe program, you'll see the following dialog:



When started, WinPLC Workbench issues a TCPI/IP broadcast query looking for any WinPLCs. The Listbox on the far left will display a list of the WinPLCs that responded to the query. If your WinPLC appears in the list, this means it has an IP address assigned to it.

If there are no WinPLCs listed, don't worry, is probably only means that there's no DHCP server available to supply IP addresses. It means that you must manually assign unique, static IP addresses.

We highly recommend that you assign a static IP address to any WinPLC before it is used in a runtime configuration.

Before you can assign the WinPLC it's static IP address, you'll have to reboot it into a special mode called the boot loader. The boot loader is the power on mode for the WinPLC. While in the boot loader, the WinPLC performs some basic power up checks, and then pauses for two seconds before loading the Windows CE operating system. It's during this two-second window that you can instruct the WinPLC not to load CE. The WinPLC will remain in the boot loader program until you cycle the power. Now you can setup the parameters that CE will use when it loads the networking stack and device drivers.

Here’s the sequence of steps needed to get the WinPLC into it’s boot loader:

1. Record the Ethernet (MAC) address from the label on the back of the WinPLC.
2. Click the "Catch..." button to display the dialog
3. Enter the WinPLC's Ethernet (MAC) address

4. Press "Next >>"
5. Cycle the power to the WinPLC

   If this works, you’ll be greeted with a Success dialog. You'll also know the boot loader is running because Run LED will be flashing On and OFF. Historically, been only two reasons for the countdown timer to expire without locating the WinPLC. The first is incorrectly entering the MAC address. The second, and most prevalent, is not having IPX protocol loaded. The ‘Catch’ function has to use the IPX protocol to communicate with the WinPLC because the CE operating system isn’t loaded.

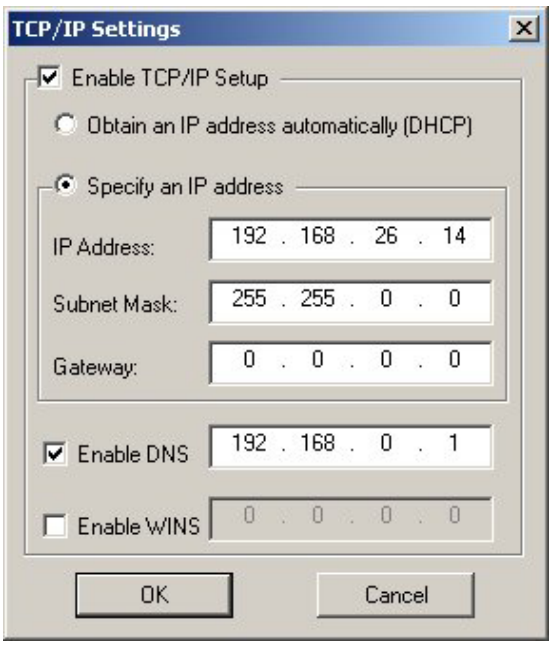
6.
7. Click the ‘Scan for WinPLCs’ button
8. Your WinPLC should now show up in the Listbox

Set the Networking parameters

Now that we can communicate with the WinPLC, we can set the networking parameters.

Highlight your WinPLC in the Listbox and the Device Information section in the middle of the dialog will be filled with data read from the WinPLC.

1. Click TCP/IP Settings, you'll see the following dialog:



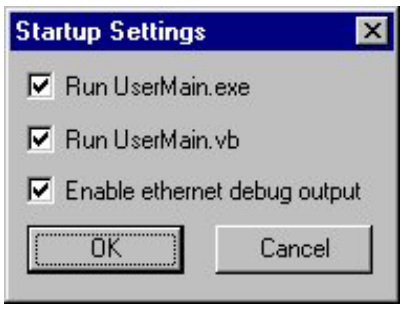
2. Click Enable TCP/IP Setup
3. Choose whether the IP address will come from a DHCP server or manual entry. As previously mentioned, the WinPLC comes to you configured to look for a DHCP server. If using a DHCP Server assigned IP address is what you want, click OK.

   If you need to manually enter an IP address, keep in mind that the WinPLC's IP address needs to be in the same IP Subnet as the development PC, and that the Subnet Mask cannot be left as 0.0.0.0.
4. The Platform Manager tools you'll use in Visual Studio require some form of IP address-to-name resolution. You have two options for name resolution services:
   a. Enable DNS and enter the address for a valid DSN server for your network. If you are using a DHCP server, this can be left disabled – but make sure the DHCP server has a DNS server configured.
   b. Enable WINS and enter the TCP/IP address of the development PC. The Platform Manager tools will only connect to the PC specified here.

   The Platform Manager tools will not work if you don't have some form of name resolution enabled.
5. You must reboot the WinPLC to get the new network settings to take affect. Simply cycle the power to the WinPLC. After rebooting, the POWER and LINK LEDs should be on and the RUN led should be off.
6. Click Scan for WinPLCs button to make sure the networking configuration is correct.

Set the Startup Parameters

Now that the WinPLC has it's IP address assigned, let's examine the startup options. Click the 'Startup Settings…' button and you'll see the following dialog:



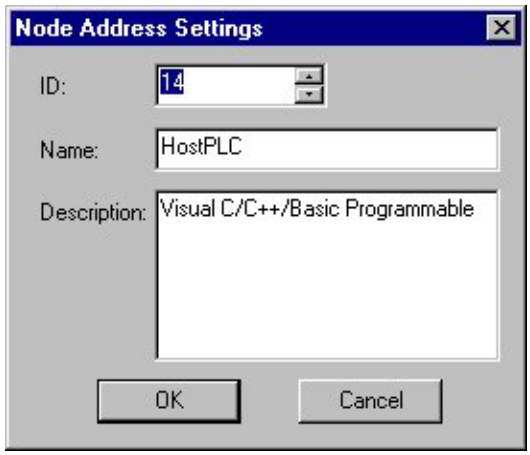
Selecting 'Run UserMain.exe' will instruct the Windows CE operating system to look in the FLASH drive for a program named UserMain.Exe and automatically run it at startup. This will allow you to make the WinPLC startup in run mode. The only caveat is that your program must be named UserMain.Exe.

Selecting 'Run UserMain.vb' will instruct the Windows CE operating system to look in the FLASH drive for a Visual Basic program named UserMain.vb and automatically run it at startup. This will allow you to make the WinPLC startup in run mode. The only caveat is that your program must be named UserMain.vb.

Selecting 'Enable Ethernet debug output' tells the WinPLC to broadcast a string of debug information when it is first powered up.

Set the Node Addressing Information

Click the 'Node Addressing…' button to display the following dialog:



With this dialog, you can assign the WinPLC a Module ID, give the WinPLC a name and / or give the WinPLC a more complete description.

Reset the WinPLC to its Factory Default Configuration

The final button in the Setup group is ‘Restore Default Settings’. It does exactly what it says, it restores the Node Addressing, TCP/IP Settings, and Startup Settings back to their original factory default values. It does NOT however, remove the contents of the FLASH drive or the RAM drive.

WinPLC Workbench Utilities

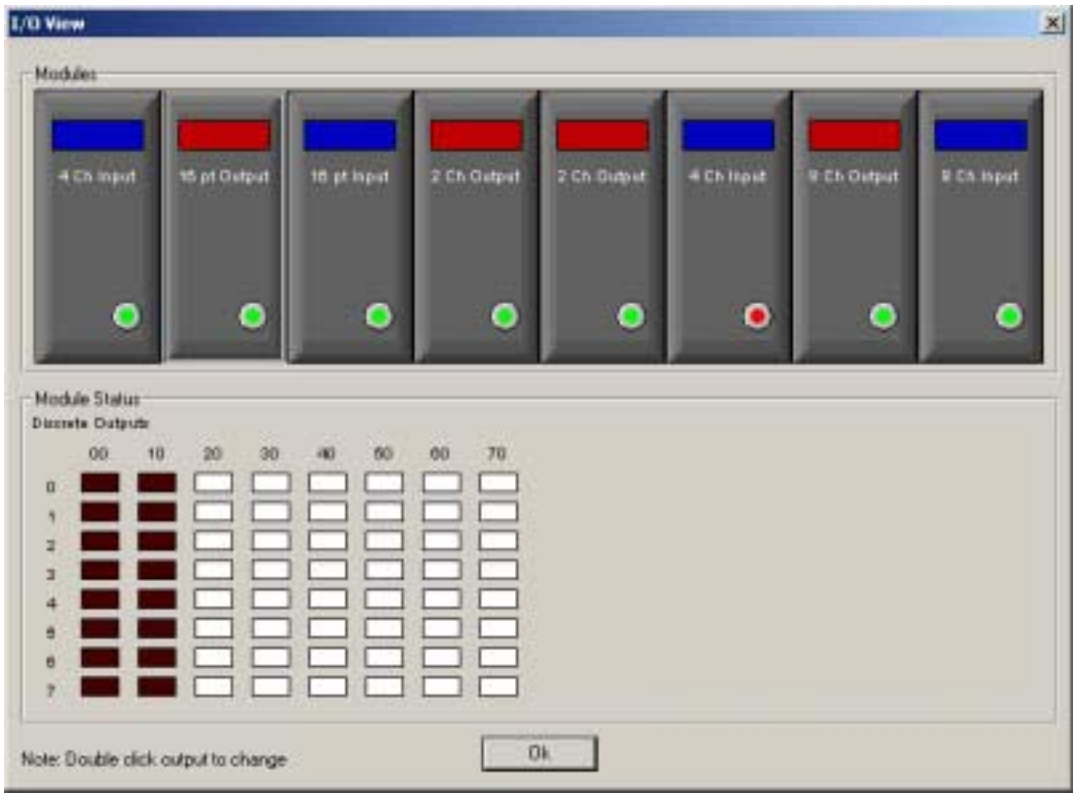
Let's now look at the utility programs provided by the WinPLC Workbench. These utility programs will allow you to monitor and test your I/O modules and your programming connection to the WinPLC without having to write a program to do so. Let's look at each of these in detail.

Monitor I/O

Monitor I/O gives you a way to read from and write to the I/O modules in the base with your WinPLC. You can see the current state of the discrete and analog inputs, toggle your discrete outputs and write values to your analog outputs.

Needless to say, please be aware that this utility will be manipulating the actual I/O. This has the potential to cause damage to your machinery or possibly injure someone. Take any necessary precautions to assure that this doesn't happen.

Click the Monitor I/O button. WinPLC Workbench will scan the backplane and display a graphical representation of the modules it finds. Slots containing specialty I/O modules like the H2-SERIO modules, empty slots, and nonexistent slots are left blank. When you click on any of the graphically represented modules, the Module Status section will change to reflect an appropriate display mode.



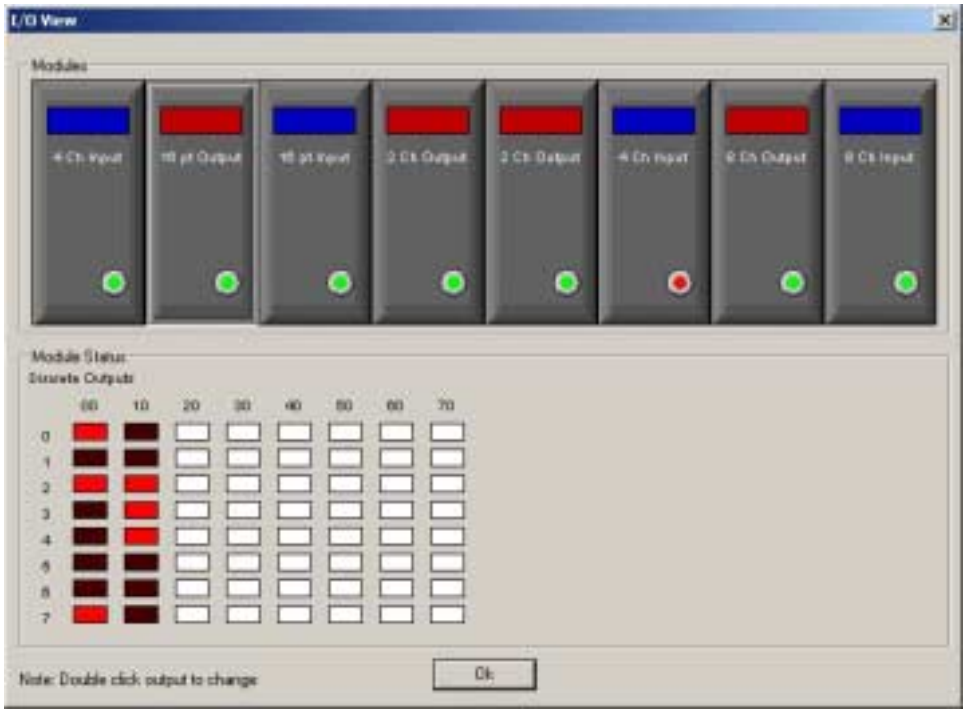
This dialog shows you a typical assortment of discrete and analog I/O modules. The second module from the left has been selected. It's a discrete module that has 16 outputs. For discrete input modules, points that are ON, will be show up in blue, points that are off will SHOW up in black. Discrete output points that are ON, will be shown in red, points that are off will again be shown in black.

The small circle in the lower right corner of each module indicates the module's status. GREEN is OK, RED means the module is reporting an error, and BLUE means the module is reporting a Warning. Click on the circle to display additional information.

Reading & Writing Discrete I/O

To toggle a discrete output point ON or OFF, select the desired module's graphic, double click on the box in the Module Status section that corresponds to the output point you want to change. You'll see the following dialog:

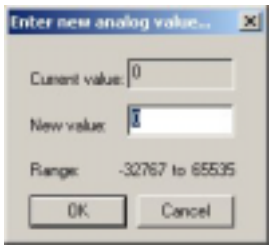
Click 'Yes – write the point' to perform the I/O update. You can click the 'Don't ask again' prompt to avoid seeing this dialog again.



This dialog shows the 8-point discrete output module in slot 5. Points 0 and 3 are ON, the rest are OFF.

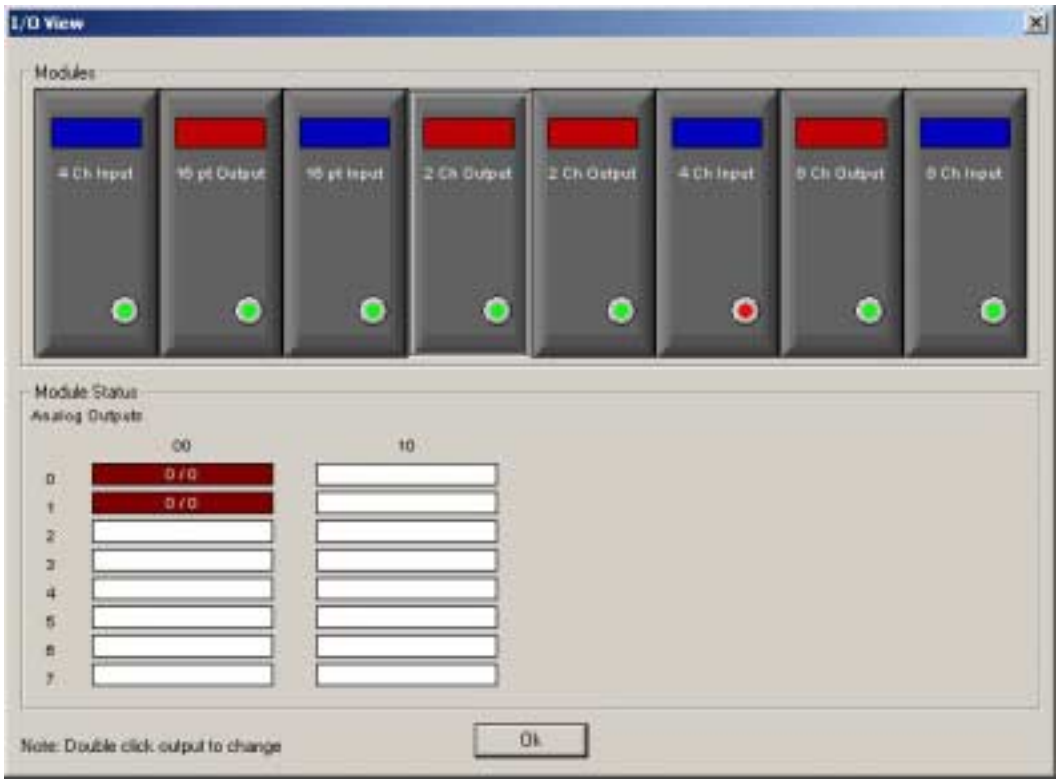
Reading & Writing Analog I/O

To write an analog output value, double-click the box that corresponds to the channel you want to write. You see the following dialog showing you the current channel value, the range of acceptable values and prompts you for the new value.



If this is the first time you're writing to a channel, you'll also see the 'confirm' dialog mentioned above before the value will actually be written.

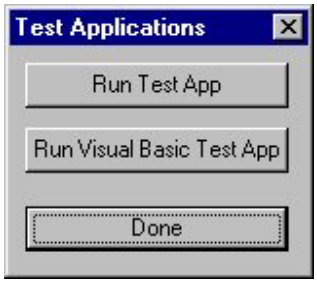
Some of the analog I/O modules can be configured for bipolar operation using either bipolar values (data bits plus a sign bit) or unipolar values (just data bits). Since we can't tell how the module has been configured, Monitor I/O shows the value in two formats. For each analog channel you'll see the unsigned value / signed value in 2's complement form. The following shows a value of 1024 written to channel 1 and –1024 written to channel 2:



Run Test App

This utility will let you test your development PC's ability to download a program to the WinPLC and have the WinPLC run that program. The program does nothing more than some simple sanity checks, then it begins blinking the RUN LED for about 1 second.

The utility will attempt to decide which test applications it can run by looking at the operating system image in the currently selected WinPLC. The test application will be copied to the WinPLC, it will be started, it will run to completion, then it will deleted itself from the WinPLC.

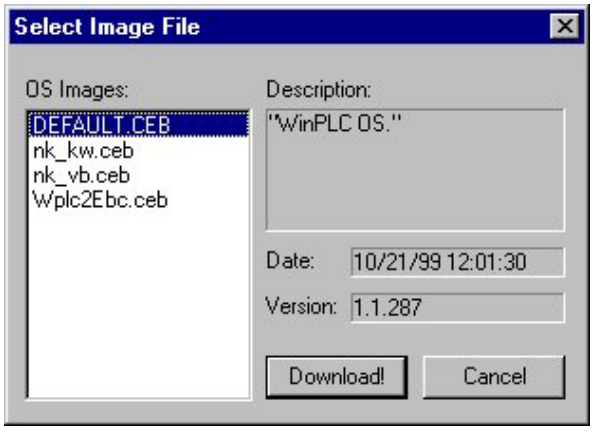


Update OS

From time to time, we will issue new operating system images for the WinPLCs. We'll do this to add or remove APIs, correct errors in the existing images or possibly to create entirely new images. This utility makes the image update as painless and as bullet proof as possible.

Clicking on any entry in the OS Images Listbox will cause the dialog to display the image's description, date of release and any version information that is available. Taken together, the image name and description should give you a pretty good idea of what types of programs the image will accommodate.

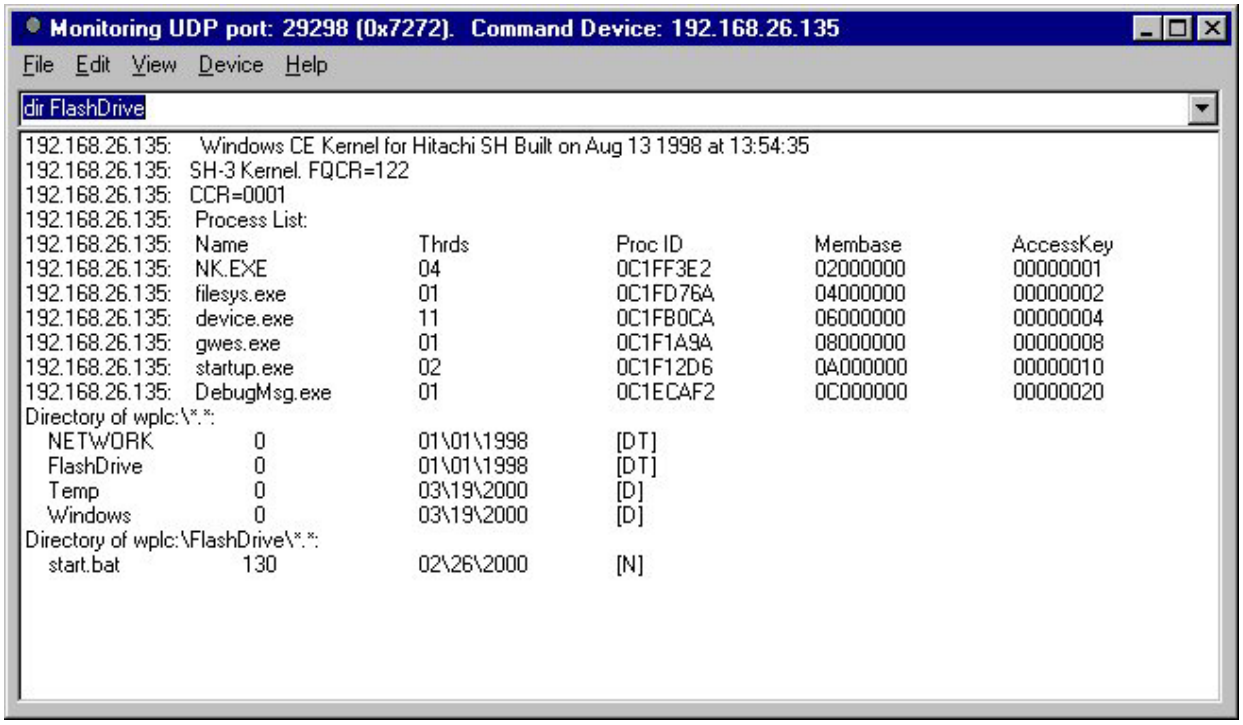
The size of the FLASH drive is determined by the amount of ROM left over after the operating system is loaded. So, updating the operating system image will delete the FLASH drive and rebuild it to accommodate any change in available ROM. **If there's anything in the flash drive you want to save, do so before updating the OS.**



Using Viewer

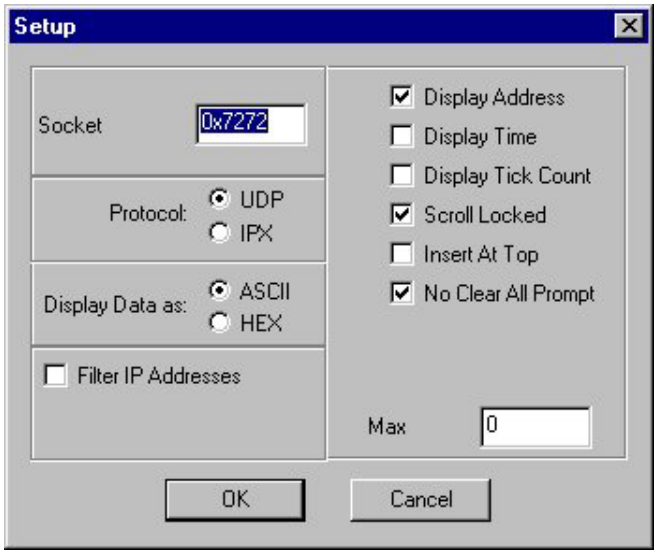
The Viewer utility provides you a way to remotely control the WinPLC. You use this utility to establish the programming connection with Visual Studio, run programs, copy files, delete files, move files, etc. Once you're in the programming environment, this utility will be your eyes into the WinPLC.

The edit field at the top is the remote command line for the WinPLC. Anything entered here gets sent to the WinPLC as a command to be executed. Responses from the WinPLC, like status information, power up information, debug messages, etc. will show up in the window at the bottom.



First, let's look at the different setup options.

View->Setup



Socket: the port number that Viewer listens on. There aren't very many good reasons to change this value from its default of 0x7272.

Protocol: this to should be left as UDP.

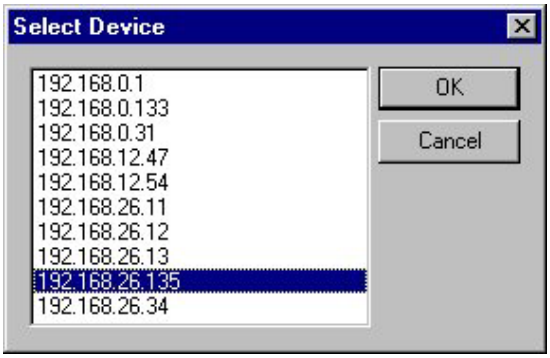
Display Data As: ASCII is correct for most applications

Filter IP Addresses: by default Viewer will display messages from any and all WinPLCs on the network. If you have several WinPLCs on the network this can get confusing. Enabling this will cause Viewer to only display messages from the WinPLC you have selected.

The remaining options affect the message display.

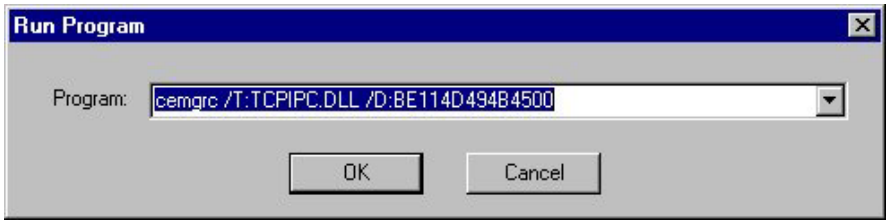
Display Address:	display TCP/IP address of sender
Display Time:	display time messages was received
Display TickCount:	display value of GetTickCount
Scroll Locked:	automatically scroll to last message
Insert At Top:	inserts new messages instead of appending new messages
No Clear All Prompt:	don't allow clear all option
Max:	maximum size of the message buffer (0 = no maximim)

Device->Select Device



It does just what you'd expect; it displays a list of WinPLCs by IP address and lets you pick one to work with. Simply highlight the appropriate entry and click Ok.

Device->Command



This option presents you with a remote command line from the WinPLC. Most people prefer to just type any remote commands into the edit field on the main dialog.

Let's try a few commands and see how they work.

Start Viewer.

Power Cycle your WinPLC- you should see three messages pop up in the window, showing details of the operating system image currently running.

Now use the Device->Select Device dialog to connect with your WinPLC.

Enter 'proclist' (without the quotes) in the edit field and press Enter. You should see a list of the processes currently running in the WinPLC.

Enter 'dir' in the edit field and press Enter. You should see a directory listing of the file system in the WinPLC. The directory FlashDrive is physically in the FLASH ROM of the WinPLC.

Enter 'dir FlashDrive' will display a directory of the files stored in the FLASH ROM.

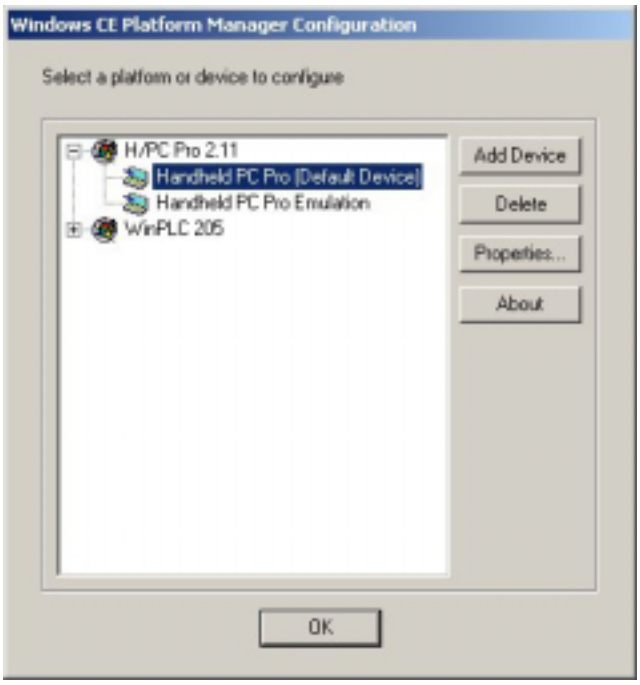
Enter 'copy c:\autoexec.bat wplc:\temp\autoexec.bat' and press Enter. This will put a copy of your PCs autoexec.bat file into the \temp directory on the WinPLC.

You can enter 'help' in the edit field to get a complete list of the supported commands.

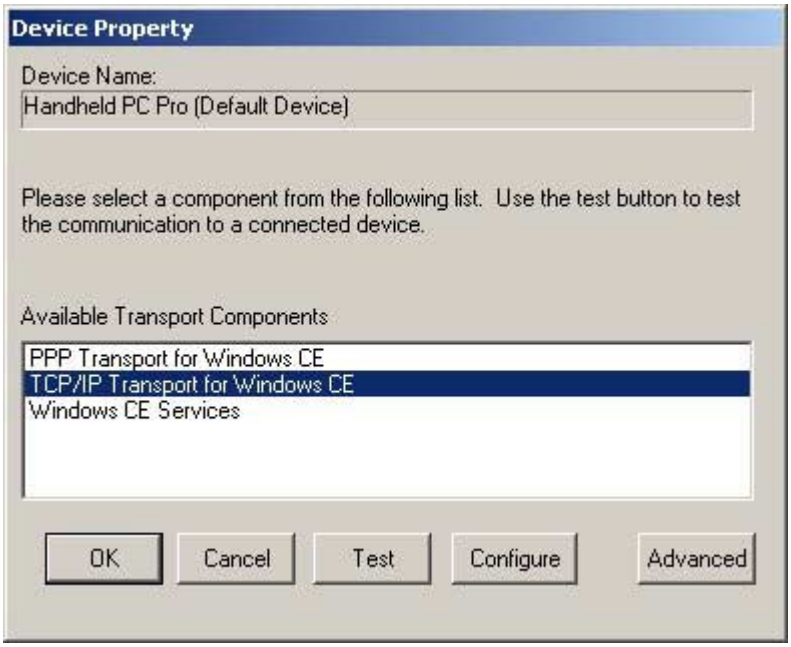
Visual C/C++ for Windows CE

Configure Platform Manager Connection – PC side

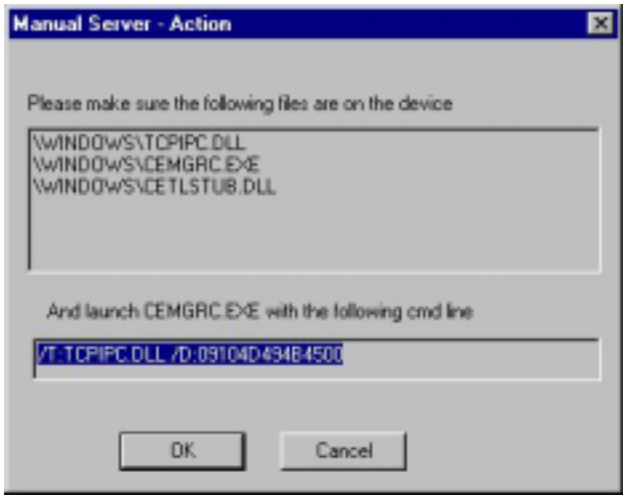
1. Start Visual C/C++.
2. Under the Tools Menu, Select “Configure Platform Manager…”
3. Expand “H/PC PRo”, select “Handheld PC Pro(Default Device)” , then click Properties.



4. On the resulting dialog, select TCP/IP Transport for Windows CE, and click “Test”.



5. Platform manager will pop up a dialog requesting that you run the server on the remote device. Highlight the text from the command line field at the bottom of the dialog and copy it to the clipboard using Ctrl-C, then click the Ok button.



6. Now go back to Viewer and type ‘cemgrc ‘ and paste in the text you copied from the previous step using Ctrl-V, then press the Enter key.

   The command line should look something like: ‘cemgrc /T:TCPIPC.DLL /D:9104D494B4500’.

   If you haven’t selected a WinPLC from the Device menu, you’ll be prompted to do so at this time. This step causes Platform Manager to start a server on the PC that is waiting for a client to log in. The command you just sent caused the WinPLC to start the client app with the necessary information to find the server.

When the connection is established, you'll see this dialog:

7. Click all of the Ok buttons to close all of the open dialogs.

If for any reason you loose the Ethernet connection to the WinPLC, you'll be presented with the dialog asking you to start the remote server, just repeat the copy / paste of the command line contents into Viewer.

Run the Development Tutorial

Open the UserMain project in the C:\WplcSDK\DemoCode directory.

Select 205 WinPLC as target

Under Project->Settings->Link, add WPLCRT.Lib to the Object/Library Modules

Press F7 to build it.

Once it builds, Visual Studio will automatically copy it to the root drive of the WinPLC. This Demo program was written for a system that has an input simulator in the first slot and an 8-point output module in the second slot. Input point 0 is used like a Run / Program switch. We recommend that you have a similar method of shutting down your user program.

By default, Visual Studio will download UserMain to the RAM. As such, the program will be lost with a power cycle. To make it persistent, you simply copy UserMain.Exe to the directory called FlashDrive. This directory is a ROM based file system and is non-volatile. To copy UserMain from the root to FlashDrive type: "copy wplc:\UserMain.Exe wplc:\FlashDrive\".

The WinPLC startup code will look on the FlashDrive for UserMain and execute it if present. The WinPLC Workbench has an option for disabling the UserMain startup. If you run into problems with your UserMain and can't get it stopped, you can change the setup with WinPLC Workbench to prevent it from starting. You can also use the kill command from Viewer to shut down an application. Simply type "kill appname" from the Viewer command line to kill a process.

Visual Basic for Windows CE

- Start Visual Basic
- Open the UserMain.Vb project in directory 'C:\WplcSDK\DemoCode\Usermain.vb'
- Select Project->Project Properties from the menu bar

  The Project Type should be 'H/PC Pro 2.11'
  Remote Path should be UserMain.vb
  Run on Target should be 'Handheld PC Pro (Default Device)

- Click Configure.
- Expand "H/PC Pro 2.11"
- Select "Handheld PC Pro (Default Device)"
- Select "Properties".
- Select "Advanced".
- Select "Manual Server" and press "Ok".
- Select "TCP/IP Transport for Windows CE".
- Press "Ok".
- Press "Test".
- Platform manager should pop up a dialog requesting that you run the server on the remote device. Highlight all of the text from the command line field at the bottom of the dialog and copy it to the clipboard using Ctrl-C.
- Press Ok

Configure Platform Manager Connection – WinPLC side

- From Viewer, type 'cemgrc ' and paste in the text you copied from the previous step using Ctrl-V. The command line should look something like: 'cemgrc /T:TCPIPC.DLL /D:9F0C424F423200'.
- Press Enter.
- This step causes Platform Manager to start a server on the PC that is waiting for a client to log in. The command you just sent caused the WinPLC to start the client app with the necessary information to find the server. You'll see a few lines of text show up in Viewer telling you about the process that was just created.
- Close all of the dialogs. Upon completion of this step you have all of the connections needed to do program development!

Development Tutorial

- Press F5 to run the program. Once it builds, Visual Studio will copy it to the root of the RAM drive of the WinPLC.
- UserMain.vb assumes that you have an input simulator in the first slot and uses the first switch to terminate. It is recommended that you have a similar method of shutting down – it will speed development.
- Switch the first switch on and UserMain.vb will terminate, displaying accumulated scan time statistics.
- UserMain.vb is currently in RAM and will be lost with a power cycle. To make it persistent, copy it to the directory called FlashDrive. This directory is a flash ROM based file system and is non-volatile. You can store any files here that need to be persistent.

  You can copy files to and from the WinPLC using Viewer's copy command. To copy UserMain from the root to FlashDrive type: "copy wplc:\UserMain.vb wplc:\FlashDrive\". To copy to or from your PC, simply use the full local pathname of your files.

  The WinPLC startup code will look on the FlashDrive for UserMain.vb and execute it if present. The WinPLC Workbench has an option for disabling the UserMain startup.

  If you run into problems with your UserMain.vb and can't get it stopped, you can change the setup with WinPLC Workbench to prevent it from starting. You can also use the kill command from Viewer to shut down an application. Simply type "kill appname" from the Viewer command line to kill a process.

Appendix A – Onboard Communications Ports and Cables

Ethernet connecting cable should be Category 5, UTP cables.

Crossover cable (PC-to-WinPLC)

```
1 ---------- 3
2 ---------- 6
3 ---------- 1
6 ---------- 2
```

Straight cable (Hub to WinPLC)

```
1 ---------- 1
2 ---------- 2
3 ---------- 3
6 ---------- 6
```

Onboard Serial Port

Looking at the front of the WinPLC, Pin 1 is the bottom pin and Pin 6 is the top pin of the RJ-12 connector

Pin Assignments:

Pin 1	0V	Chassis Ground
Pin 2	5V	+5V Power
Pin 3	RxD	Receive Data
Pin 4	TxD	Transmit Data
Pin 5	RTS	Request To Send
Pin 6	0V	Signal Ground

Appendix B – Retentive Memory

The WinPLCs provide the programmer with 16K of battery-backed RAM, also known as retentive memory. This memory is typically used to hold variable data that needs to be restored on power up. It's up to the program to keep the data in the retentive memory block up to date.

Applications using a PLC-like scan typically update the values on a scan-interval basis, once per scan, once every ten scans, etc. Event based applications typically update them at specified time intervals.

Here's an example of setting up and using the retentive RAM.

```
// Function prototypes
int WPLCMapBattRam( DWORD Offset, DWORD NumBytes, BYTE **pBBR_Ptr );
int WPLCUnMapBattRam( BYTE *BBR_Ptr, DWORD NumBytes );

// variables that will exist in the battery backed RAM block
typedef struct
{
        DWORD       dwItem;
        WORD        wItem;
        BYTE        bItem;
} BatteryBackedRAM;

main()
{
        BatteryBackedRAM      *BBR_Ptr= NULL;

        DWORD Offset= 0;
        DWORD Val= 1;

        if( int Error= WPLCMapBattRam(  Offset, sizeof( BatteryBackedRAM), (BYTE **)&BBR_Ptr )  )
        {
                OutputDebugString( ( TEXT(  "Error on call to WPLCMapBattRam" )  )  );
        }

        BBR_Ptr->dwItem= Val;

        Val++;
        BBR_Ptr->wItem= (WORD)Val;

        Val++;
        BBR_Ptr->bItem= (BYTE)Val;
}
```