ELAN-NC
Quick Start & Data Manual
For Windows CE 3.0

Product Information

Full information about other Arcom products is available by visiting our WebSite at:
www.arcomcontrols.com

Useful Contact Information

Customer Support Europe

Tel: +44 (0)1223 412 428
Fax: +44 (0)1223 403 409
E-mail: support@arcom.co.uk

Customer Support US

Tel: 816 941 7025
Fax: 816 941 7807
E-mail: support@arcomcontrols.com

Sales Offices

United Kingdom:
Arcom Control Systems Ltd
Clifton Road
Cambridge CB1 7EA,
UK
Tel: 01223 411 200
Fax: 01223 410 457

E-mail:
sales@arcom.co.uk

United States:
Arcom Control Systems Inc
13510 South Oak Street
Kansas City MO 64145
USA
Tel: 816 941 7025
Fax: 816 941 0343

E-mail:
icpsales@arcomcontrols.com

Sales Hotlines

Belgium:
Groen Nummer:
Tel: 0800 7 3192
Fax: 0800 7 3191

France:
Numero Vert
Tel: 0800 90 84 06
Fax: 0800 90 84 12

Germany:
Kostenlose Infoline:
Tel: 08001 824 511
Fax: 08001 824 512

Netherlands:
Gratis 0800 Nummer:
Tel: 0800 0221136
Fax: 0800 0221148

Italy:
Numero Verde:
Tel: 0800 790841
Fax: 0800 780841

Whilst Arcom's sales team is always available to assist you in making your decision, the final choice of boards or systems is solely and wholly the responsibility of the buyer. Arcom's entire liability in respect of the boards or systems is as set out in Arcom's standard terms and conditions of sale.

If you intend to write your own low level software, you can start with the source code on the disk, which is supplied. This is example code only to illustrate use on Arcom's products. It has not been commercially tested. No warranty is made in respect of this code and Arcom shall incur no liability whatsoever or howsoever arising from any use made of the code.

Arcom Control Systems Ltd operate a company-wide quality management system which has been certified by the British Standards Institution (BSI) as compliant with ISO9001:1994

Revision History

Manual	Comments	
Issue A	010320	Manual first released in this format.

Preface

Disclaimer

The information in this manual has been carefully checked and is believed to be accurate. Arcom Control Systems assumes no responsibility for any infringement of patents or other rights of third parties that may result from its use.

Arcom Control Systems assumes no responsibility for any inaccuracies that may be contained in this document. Arcom Control Systems makes no commitment to update or keep current the information contained in this manual.

Arcom Control Systems reserves the right to make improvements to this document and /or product at any time and without notice.

Anti-Static Handling

The boards in this development kit contain CMOS devices that may be damaged in by electrostatic discharge. At all times, please observe anti-static precautions when handling the boards. This includes storing the boards in appropriate anti-static packaging and wearing a wrist strap when handling the boards.

Packaging

Please ensure that should a board need to be returned to Arcom Control Systems, it is adequately packed, preferably in the original packing material.

Acknowledgements

ROM-DOS is a trademark of Datalight Inc.

Windows CE is a trademark of the Microsoft Corporation.

All other trademarks acknowledged.

Technical Support

Arcom Control Systems has a team of technical support engineers who will be able to provide assistance if you have any problems with this product.

IMPORTANT NOTICE, PLEASE READ

Each Windows CE Development Kit is supplied by Arcom is licensed. The Windows CE operating system has been configured to run on the Arcom ELAN-NC processor. A copy of the operating system image is included on the Development Kit CD and may be used to restore the image on the ELAN-NC Development Kit. The operating system image on the CD may also be used to produce production copies of the supplied operating system. The Windows product may be used in your product using one of the following license options :

1. If you wish to purchase boards from Arcom with the operating system pre-loaded onto a drive, Arcom can supply the operating system along with the matching Windows CE license number. Please contact Arcom for information about license costs.

2. If you intend to use the Windows CE operating system image supplied by Arcom and purchase the Windows CE licenses directly from your local Microsoft distributor, you must sign an OEM Customer License Agreement (supplied by your Microsoft distributor). The licenses are available in packs of 10, 100 and 1000. You will be responsible for installing the operating system onto the Arcom product and making sure it is licensed .

Contact your local Microsoft distributor for license costs.

In BOTH cases you must include a copy (printed or electronic) of the End User License Agreement (EULA). If you are **not** the end user, you must supply a copy of the EULA to your customers. You should replace ‘Arcom Control Systems’ with your own company name as set out in the first paragraph of the EULA. The EULA must be distributed along the supply chain to the end customer.

Arcom's ELAN-NC Windows CE Development Kit is supplied with a copy of the EULA on the Development Kit CD (see attached copy). The OEM Customer License Agreement and End User License Agreement for the Windows CE product identify the application restrictions (air traffic control, military equipment etc) and limitations of distribution and liability. You must check these before using the product.

Attachment A

COMPANY shall sublicense the Licensed Product to end users by means of an End User License Agreement ("EULA"). COMPANY shall provide clear notice to Embedded System customers before or at the time of purchase that the Embedded System contains software that is subject to a license and that customer must agree to that license before it may use the Embedded System. Company shall distribute the EULA in a manner that forms a contract binding the end user to the EULA terms under applicable law. The EULA shall contain (A) the following terms, and (B) any additional EULA terms set forth by MS in the "additional provisions" on the Licensed Product package. COMPANY may include in the EULA additional notices or other provisions, so long as they are no less protective of MS than the terms set forth below. COMPANY shall substitute its name for the bracketed text [*COMPANY*] in the EULA text. If permitted by COMPANY's other software suppliers for a particular COMPANY Embedded System, COMPANY may substitute for the word “MS” the term "[*COMPANY’s*] software suppliers".

- You have acquired an item ("DEVICE") that includes software licensed by Arcom Control Systems from Microsoft Licensing Inc. or its affiliates ("MS"). Those installed software products of MS origin, as well as associated media, printed materials, and "online" or electronic documentation ("SOFTWARE") are protected by copyright laws and international copyright treaties, as well as other intellectual property laws and treaties. The SOFTWARE is licensed, not sold.

- IF YOU DO NOT AGREE TO THIS END USER LICENSE AGREEMENT ("EULA"), DO NOT USE THE DEVICE OR COPY THE SOFTWARE. INSTEAD, PROMPTLY CONTACT ARCOM CONTROL SYSTEMS FOR INSTRUCTIONS ON RETURN OF THE UNUSED DEVICE(S) FOR A REFUND. **ANY USE OF THE SOFTWARE, INCLUDING BUT NOT LIMITED TO USE ON THE DEVICE, WILL CONSTITUTE YOUR AGREEMENT TO THIS EULA (OR RATIFICATION OF ANY PREVIOUS CONSENT).**
- **GRANT OF LICENSE.** The SOFTWARE is licensed, not sold. This EULA grants you the following rights to the SOFTWARE:

- You may use the SOFTWARE only on the DEVICE.
- **NOT FAULT TOLERANT.** THE SOFTWARE IS NOT FAULT TOLERANT. ARCOM CONTROL SYSTEMS HAS INDEPENDENTLY DETERMINED HOW TO USE THE SOFTWARE IN THE DEVICE, AND MS HAS RELIED UPON ARCOM CONTROL SYSTEMS TO CONDUCT SUFFICIENT TESTING TO DETERMINE THAT THE SOFTWARE IS SUITABLE FOR SUCH USE.
- NO WARRANTIES FOR THE SOFTWARE. THE SOFTWARE is provided "AS IS" and with all faults. THE ENTIRE RISK AS TO SATISFACTORY QUALITY, PERFORMANCE, ACCURACY, AND EFFORT (INCLUDING LACK OF NEGLIGENCE) IS WITH YOU. ALSO, THERE IS NO WARRANTY AGAINST INTERFERENCE WITH YOUR ENJOYMENT OF THE SOFTWARE OR AGAINST INFRINGEMENT. IF YOU HAVE RECEIVED ANY WARRANTIES REGARDING THE DEVICE OR THE SOFTWARE, THOSE WARRANTIES DO NOT ORIGINATE FROM, AND ARE NOT BINDING ON, MS.
- **Note on Java Support.** The SOFTWARE may contain support for programs written in Java. Java technology is not fault tolerant and is not designed, manufactured, or intended for use or resale as online control equipment in hazardous environments requiring fail-safe performance, such as in the operation of nuclear facilities, aircraft navigation or communication systems, air traffic control, direct life support machines, or weapons systems, in which the failure of Java technology could lead directly to death, personal injury, or severe physical or environmental damage. Sun Microsystems, Inc. has contractually obligated MS to make this disclaimer.
- No Liability for Certain Damages. EXCEPT AS PROHIBITED BY LAW, MS SHALL HAVE NO LIABILITY FOR ANY INDIRECT, SPECIAL, CONSEQUENTIAL OR INCIDENTAL DAMAGES ARISING FROM OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THE SOFTWARE. THIS LIMITATION SHALL APPLY EVEN IF ANY REMEDY FAILS OF ITS ESSENTIAL PURPOSE. IN NO EVENT SHALL MS BE LIABLE FOR ANY AMOUNT IN EXCESS OF U.S. TWO HUNDRED FIFTY DOLLARS (U.S.$250.00).
- **Limitations on Reverse Engineering, Decompilation, and Disassembly.** You may not reverse engineer, decompile, or disassemble the SOFTWARE, except and only to the extent that such activity is expressly permitted by applicable law notwithstanding this limitation.
- **SOFTWARE Transfer allowed but with restrictions**. You may permanently transfer rights under this EULA only as part of a permanent sale or transfer of the Device, and only if the recipient agrees to this EULA. If the SOFTWARE is an upgrade, any transfer must also include all prior versions of the SOFTWARE.
- **EXPORT RESTRICTIONS.** If these licensing terms are not labelled "North America Only Version" and the SOFTWARE is not identified as "North America Only Version" on the SOFTWARE packaging or other written materials, then the following terms apply: Export of the SOFTWARE from the United States is regulated by the Export Administration Regulations (EAR, 15 CFR 730-744) of the U.S. Commerce Department, Bureau of Export Administration (BXA). You agree to comply with the EAR in the export or re-export of the SOFTWARE: (i) to any country to which the U.S. has embargoed or restricted the export of goods or services, which as of May 1999 include, but are not necessarily limited to Cuba, Iran, Iraq, Libya, North Korea, Sudan, Syria, and the Federal Republic of Yugoslavia (including Serbia, but not Montenegro), or to any national of any such country, wherever located, who intends to transmit or transport the SOFTWARE back to such country; (ii) to any person or entity who

you know or have reason to know will utilise the SOFTWARE or portion thereof in the design, development or production of nuclear, chemical, or biological weapons; or (iii) to any person or entity who has been prohibited from participating in U.S. export transactions by any federal agency of the U.S. government. You warrant and represent that neither the BXA nor any other U.S. federal agency has suspended, revoked or denied your export privileges. For additional information, see http://www.microsoft.com/exporting/.
If the licensing terms are labelled "North America Only Versions" or if the SOFTWARE is labelled "North America Only Version" on the SOFTWARE packaging or other written materials, then the following applies: The SOFTWARE contains strong encryption features. The SOFTWARE may be distributed in the United States, its territories, possessions and dependencies, and Canada without an export license. Export of the SOFTWARE from the United States is regulated under “EI controls” of the Export Administration Regulations (EAR, 15 CFR 730-744) of the U.S. Commerce Department, Bureau of Export Administration (BXA). An export license or applicable license exception is required to export the SOFTWARE outside the United States or Canada. You agree that you will not directly or indirectly export or re-export the SOFTWARE (or portions thereof) to any country, other than Canada, or to any person or entity subject to U.S. export restrictions without first obtaining a Commerce Department export license or determining that there is an applicable license exception. You warrant and represent that neither the BXA nor any other U.S. federal agency has suspended, revoked, or denied your export privileges. For additional information see http://www.microsoft.com/exporting/.

Microsoft OEM Customer License Agreement For
Embedded Systems
Short Form 1299 - Attachment A

ELAN-NC Quick Start Manual for Windows CE 3.0

What's in the Kit?

Arcom's ELAN-104NC kit for Windows CE comprises of the following items:-

- An ELAN-104NC-M16-F8 processor board loaded with a Windows CE image
- A Power supply with US, UK and Euro style plugs, and connector for the ELAN-104NC board (See note 1 below)
- A PS/2 mouse
- A PS/2 to AT keyboard adapter
- A VGA adapter cable
- A DB9 to DB9 null modem cable
- A Floppy drive ribbon cable
- An IDE drive ribbon cable
- A Reset switch
- An ELAN-104NC Windows CE Development Kit CD-ROM
- This set of instructions
- An ELAN-104NC Flat Panel Interface board (FPIF board)
- An ELAN-104NC to FPIF interface cable
- Microsoft eMbedded Visual Tools (See note 2 below)

The following items are also included with the Flat Panel kit:-
- An NEC 6.5" flat panel (Type NL6448AC20-08)
- A backlight inverter board
- An FPIF to flat panel interface cable
- A backlight power cable

Note 1:
The PSU supplied with the kit is sufficient to power the items in the development kit. If additional boards or peripherals are to be used a more powerful supply may be required. The +5v supply should be greater than 4.9 volts.

To use Microsoft eMbedded Visual Tools 3.0, you need to have the following:
- A desktop computer with a Pentium-class processor. A Pentium 150-MHz or higher processor is recommended.
- Microsoft Windows® 2000 Professional; Microsoft Windows NT® Workstation 4.0 with SP5, Internet Explorer 5.01, and MDAC 2.1; or Microsoft Windows® 98 Second Edition. Microsoft Windows® 2000 Professional or Microsoft Windows NT Workstation 4.0 is the recommended debug host for your development environment. The eMbedded Visual Tools can be installed on Windows 98, and you can build your application from there. However, emulation does not work on Windows 98; instead use Windows 2000 or Windows NT as your host machine.
- 24 MB memory for Windows 98 Second Edition (48 MB recommended). 32 MB for Windows NT Workstation 4.0 or Windows 2000 (48 MB recommended).
- CD-ROM drive compatible with multimedia desktop computer specification.
- VGA or higher-resolution monitor required. A Super VGA monitor is recommended.
- Mouse device or compatible pointing device.
- Adequate hard disk space.

Minimum installation (eMbedded Visual C++ and one SDK): 360 MB.
Full installation (eMbedded Visual Basic, eMbedded Visual C++, and three SDKs): 720 MB.

Note2:
An embedded visual tools data sheet is available from Microsoft's embedded products WebSite at
www.microsoft.com/windows/embedded/ce/tools/factsheet.asp

Additional copies of eMbedded Visual Tools can be ordered from Microsoft's embedded products WebSite at
www.microsoft.com/windows/embedded/ce/tools/emvt30order.asp

Links to the above websites are provided on the Arcom Windows CE CD-ROM.

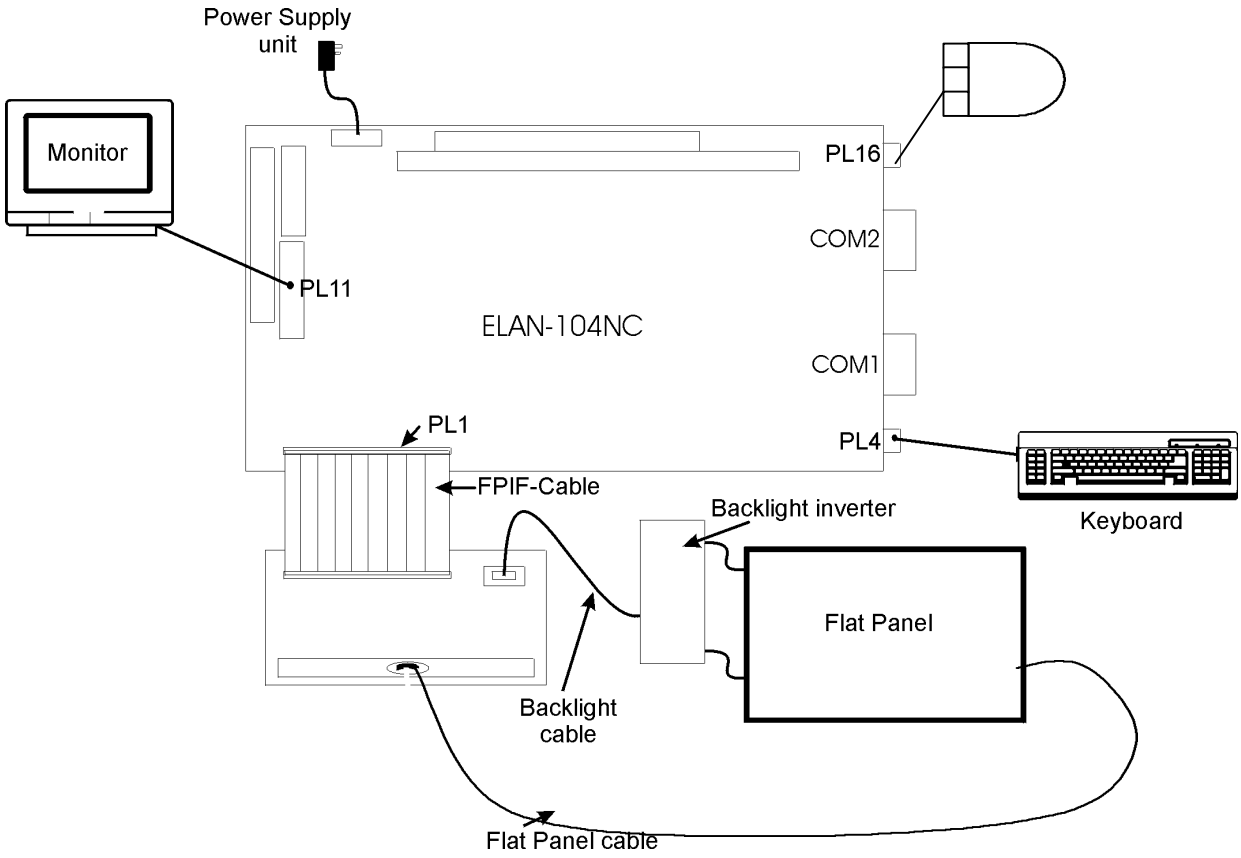
What else do I need?

- A PS/2 or AT Keyboard
- A VGA monitor (If not using the 'Flat Panel' kit)
- Microsoft ActiveSync Version 3.0 or later (See note below)

Note:
The ActiveSync software can be downloaded from Microsoft's Windows CE web site at
www.microsoft.com/pocketpc/downloads/activesync.asp

A link to the above website is provided on the Arcom Windows CE CD-ROM.

Task 1: Powering up the ELAN-104NC board



1. Plug the supplied mouse into the socket labelled 'PL16 MOUSE'

2. Plug a keyboard into the socket labelled 'PL4 KEYBOARD', using the supplied PS/2 adapter if required

3. Connect the reset switch to the connector labelled ‘RESET PL19’

4. If using a CRT monitor:
   - Plug the monitor into the supplied adapter cable, then plug the adapter cable into the socket labelled 'CRT PL11'

   OR

5. If using the 6.5" Flat Panel, supplied in a Flat Panel Development kit:
   - Make Sure a CRT monitor is NOT connected, or the flat panel will not work
   - Make connections to the supplied FPIF (Flat Panel Interface board) as described below:
     PL1: Connect to the ELAN-104NC 'PL1 LCD PANEL' connector using the supplied 40way ribbon cable
     PL2: Connect to the flat panel’s data and control signals header, using the cable supplied
     PL3: Connect to the backlight inverter board, using the cable supplied
   - Set the FPIF and ELAN-104NC links as described below:

FPIF Links		
LK1	B	Shift clock buffered
LK2	B	Shift clock non-inverted

ELAN-104NC Links		
LK1	A	3.3V Flat Panel voltage
LK11*	B	+5V Backlight supply voltage

* Version 2 boards onwards

Further information on Flat Panels
Refer to the ELAN-104NC Technical Manual in the documentation section of the Windows CE CD-ROM.

6. Fit the appropriate US, UK or Euro plug onto the power supply unit

7. Plug the power supply cable into the socket marked 'PL17 POWER'

8. Plug the power supply unit into a 100-240V AC supply

9. If necessary, enter the BIOS set up by using the F2 key, and make changes as required

Further information on BIOS Set Up Refer to the ELAN-104NC Technical manual in the documentation section of the Windows CE CD-ROM.

The system will boot to a 'Windows CE' backdrop

You have now completed Task 1

Task 2: Saving the registry

This task is to demonstrate the use of the Flushreg utility in order to make sure that any changes to the registry are saved.

1. Power up the board as described in Task 1
2. Click on ‘Start’, then click on ‘Run’
3. In the Open dialog box, type ‘control’, then click on ‘OK’ to open the control panel
4. Double Click on the ‘Display’ icon
5. Select the ‘Tile image on background’ tick box
6. Click the ‘OK’ box
7. Shut down the ‘Control panel’, by clicking on the 'X' in the top right of the panel

Note:
The 'WindowsCE' logo will now be tiled on the screen, but if the board were to be rebooted at this time, the change would not have been saved, and the single ‘WindowsCE’ logo would reappear. The registry needs to be flushed to save these settings, as described below.

8. Click on ‘Start’, then click on ‘Run’
9. In the Open dialog box, type ‘flushreg’, then click on OK to open the Flushreg application
10. Click the ‘Flush registry’ button
11. Wait for the ‘Registry Flushed’ message box. Click on ‘OK’ to close the message box.
12. Click on ‘OK’ to close the ‘Flushreg’ application.
13. Reboot the ELAN-NC board, the backdrop will have changed to a tiled ‘WindowsCE’ logo.

Note:
Windows CE can only be rebooted by power cycling the board, or by pressing the reset switch attached to the connector marked 'RESET PL19'.

Further Information :
Details of the FlushToFile() function to save the registry can be found in the ArcomUtl section of this manual.

You have now completed Task 2

Task 3: Setting up a COM port for the ActiveSync connection

This task is used to set up a COM port that will be used in the next task for serial connection with the Host system.

Configuring a new connection

1. Click on ‘Start’, then click on ‘Run’
2. In the Open dialog box type remnet then click on 'OK'
3. Double click the 'Make New Connection' icon
4. Enter a name for the connection, click the 'Direct Connection' option, then click 'Next'
5. Select 'Serial Cable on COM1'
6. Click on the 'Configure…' button
7. Set the Connection Preferences as follows

    - Baud Rate 19200
    - Data Bits 8
    - Parity None
    - Stop Bits 1
    - Flow Control Hardware

8. Click on the 'OK' button to close the 'Device Properties' window
9. Click on the 'Finish' button to close the 'Make New Direct Connection' window
10. Click on the 'X' button to close the remnet window

Note:
The Baud rate should initially be set as 19200. After an ActiveSync connection has been made for the first time, it can be changed up to 115200

Selecting the new connection

11. Click on ‘Start’, then click on ‘Run’
12. In the Open dialog box type ‘control’ then click on 'OK'
13. Double click the 'Communication' icon
14. Select the 'PC Connection' Tab
15. Select the 'Change…' button
16. Select the name chosen for the connection made above
17. Click the 'OK' button to close the 'Change Connection' window

18. Click the 'OK' button to close the 'Communications Properties' window

19. Click the 'X' button to close the 'Control Panel' window

Saving the changes to the registry

20. Click on 'Start', then click on 'Run'

21. In the Open dialog box type 'flushreg' then click on 'OK'

22. Click the 'Flush registry' button (The on-board Red 'Flash Access' LED will light)

23. Wait for the 'Registry flushed' message box

24. Click the 'OK' button to close the message box

25. Click the 'OK' button to close the 'flushreg' application

The settings just configured will be used the next time the 'repllog.exe' application is used to connect to the host PC that is running ActiveSync, as described in the next task.

You have now completed Task 3

Task 4: Serial Connection to a Host system using ActiveSync

This task is used to show how to connect the Windows CE system to a host PC and establish a partnership. This task must be carried out before an Ethernet connection can be established.

Note:
Microsoft's ActiveSync 3.0 or later is needed for this task. It can be downloaded from the link
www.microsoft.com/pocketpc/downloads/activesync.asp
This link is provided on the WindowsCE CD-ROM.

NULL Modem Cable Connection

1. Connect the included null modem cable from the designated COM port on the host system to COM1 on the CE system.

2. If the ActiveSync software has not been installed on your host system, install it now
   If the ActiveSync software has been installed, click on 'File, Get connected…'

Connecting to the CE System

3. When the 'Get Connected' screen is shown on the host system:-

On the CE system

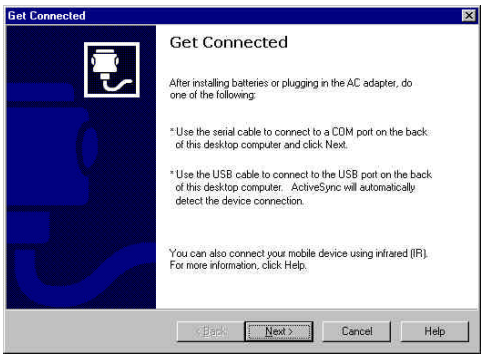
4. Click on 'Start', and click on 'Run'

5. In the Open dialog box, type 'cmd' then click on 'OK'

6. Type 'start repllog' then hit ↵ Enter

On the Host system

7. At the 'Get Connected' screen, click 'Next'



Note:
The order above is important.
The 'Get Connected' 'Next' button should only be clicked after typing 'start repllog' and hitting ↵ Enter on the CE System

If a connection was NOT made

Note:
It may take several attempts to get the first connection established.

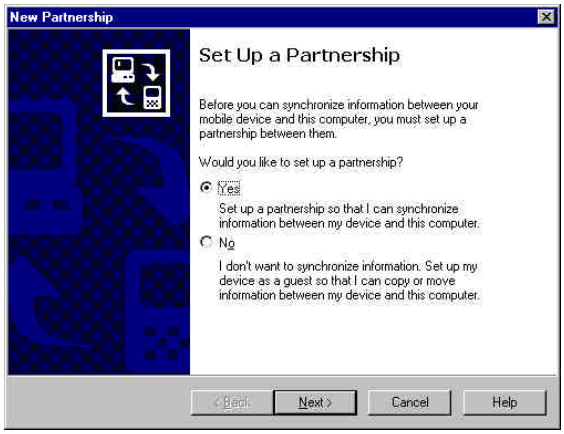
- Check that the serial cable is connected between the Host system and the COM port on the CE system that was set up in Task 3

- Repeat the procedure from step 4

When a connection has been made

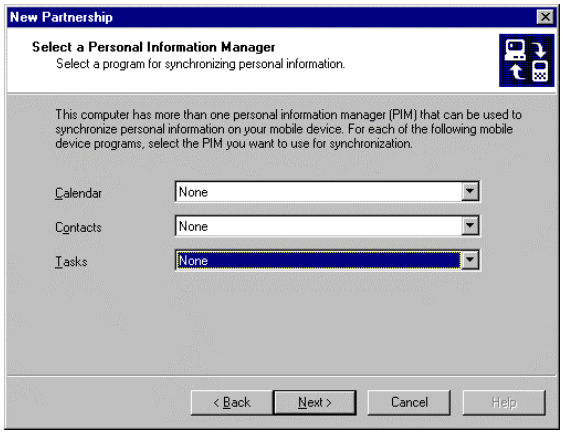
If a connection was made between the host system and the CE system, the 'ActiveSync' icon on the Host system task bar will turn green, and a 'New Partnership' screen will appear.

8. Select the 'Yes' option that states, 'Set up a partnership so that I can synchronise information between my device and this computer'



9. Click ‘Next’

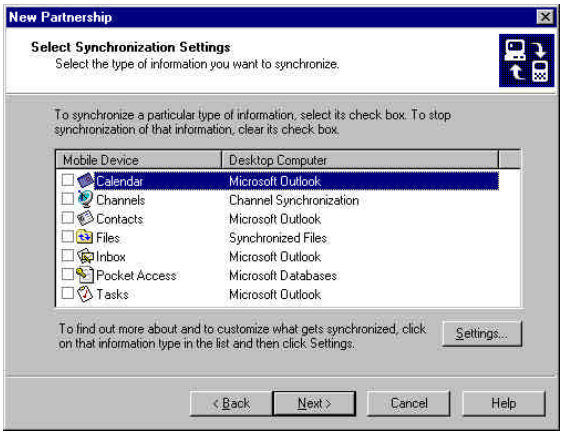
Note: This screen will not appear if using Windows NT 4.0



10. Change the Calendar, Contacts and Tasks settings to ‘None’

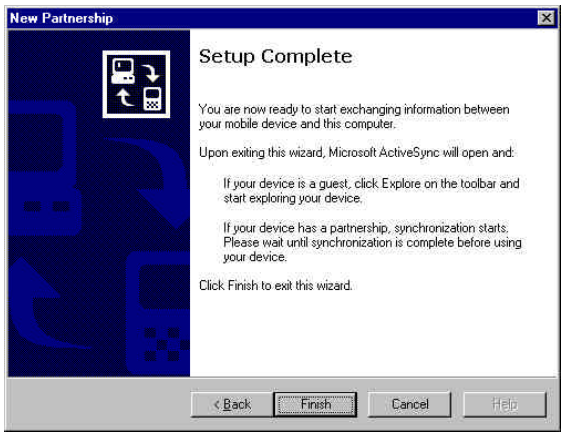
11. Click ‘Next’

12. Make sure all boxes are un-ticked

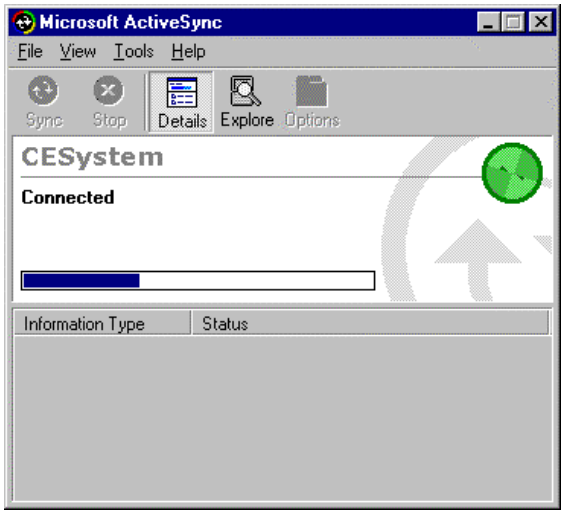


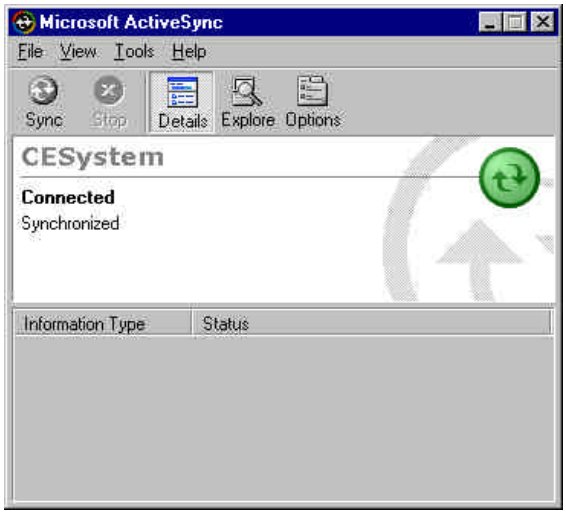
13. Click ‘Next’

14. Click ‘Finish’



The CE System will synchronise with the Host system





15. **Important:**
    After a partnership has been established, the registry on the CE system **MUST** be flushed (saved) as described in Task 2

16. In ActiveSync, Click on 'Explore', then the 'My Computer' icon to explore the files on the CE system (The CE system can also be explored using the Host systems Windows Explorer.)

The connection can be disconnected by clicking on the Connected ICON on the Task Bar of the CE system, and selecting Disconnect.

Note:
After this connection has been made, subsequent serial reconnections should only require typing ‘repllog’ on the CE system. (Nothing has to be done with the host system)

The baud rate of the connection set up for the ActiveSync connection in Task 3 can now be increased from 19200.

You have now completed Task 4

Task 5: Ethernet Connection to a Host system using ActivSync

This task is to show how to communicate with a host PC using an Ethernet connection.

Note:

- Before starting this task, a partnership with the host system must have been established using a serial connection, as described in the previous task, and then disconnected
- If the board is not connected to a network that has a DHCP server running, a static IP address must be set up before continuing with this section. Refer to the Ethernet section for information on setting up a static IP address.
- The files pegobj.dll, aafobj.dll & rra_stm.dll must be loaded onto the \FlashDisk\Arcom directory. If they have been deleted, they can be re-loaded from the Arcom Windows CE CD-ROM

Check the system has a correct IP Configuration

1. Click on 'Start', and click on Run
2. In the Open dialog box, type 'cmd' then click on OK
3. Type 'ipconfig'

   The IP address, Subnet Mask and any other IP parameters set up will be displayed.

Note:
If a 'No Interfaces Present' message appears, check the BIOS setup, as described in the BIOS settings section.

Establishing a connection

4. In the command window, type 'start repllog /remote'
5. In the ActiveSync dialog box, select 'Network Connection' and the name of the host system

Note:

- The host system's name must begin with an alphanumeric character. A name that begins with a number will not work.
- There must not be any '.' (dots) in the host name.

6. Click 'Connect…'

A 'Connecting to Desktop' dialog box will appear for up to 30 seconds

When a connection has been made, a 'Connection Status' dialog box will appear.

Note:

- The Disconnect button will disconnect the ActiveSync connection.
- The Sync Now button is not applicable for the configuration you will be setting up.

You have now completed Task 5

Task 6: Copying files using ActiveSync

This task is used to show how files can be copied from the host PC to the Windows CE system.

1. Establish communications between the Windows CE system and host system, as described in Task 4 or Task 5

2. Insert the Windows CE CD-ROM into the host systems CD ROM drive

3. Use the host systems ‘Windows Explorer’ to open the ‘\Demos’ folder on the CD-ROM

4. Copy ‘spintest.exe’ and ‘spincube.dll’ to the ‘\FlashDisk\User’ folder on the Windows CE system

Note:
To see the ‘Spincube.dII’ file, the host systems ‘Windows Explorer’ has to be set up to ‘show all files’. This can be set up by clicking ‘View’, ‘Option’, then selecting ‘Show all files’.
The Windows CE System ‘\Flashdisk\User’ folder can be found by clicking through the icons under the ‘Mobile Devices’ icon.

5. If a 'No Converter Selected' dialogue box appears, click ‘OK’

6. At the Windows CE system, Click on ‘Start’,

7. Use ‘Run’ and then ‘Browse’ to select ‘ \FlashDisk\User\spintest.exe’

8. Click ‘OK’ to run spintest

You have now completed Task 6

ELAN-NC Data Manual
for Windows CE 3.0

BIOS Settings

Refer to the ELAN-NC Technical Manual in the documentation section of the Windows CE CD-ROM.

The Boot Loader

The Arcom ELAN-NC Windows CE operating system runs on a standard ELAN-NC with the usual BIOS. This boots directly into ROM-DOS, which is supplied with every ELAN-NC board.

The Windows CE boot loader is itself a small ROM-DOS application. It requires the following ROM-DOS files to be present on the ELAN-NC FlashFX filing system:

IBMBIO.COM
IBMDOS.COM
HIMEM.SYS
STARTCE.EXE
BOOTCE.EXE

The boot loader is comprised of 2 executables, called BOOTCE.EXE and STARTCE.EXE. BOOTCE.EXE is called from an autoexec.bat file. This loads the CE image into RAM. STARTCE.EXE is called by BOOTCE.EXE to start the CE operating system running.

BOOTCE has the following switches:

/L:DXxDYxBPP	Set video resolution and colour depth		
	where		
	DX	=	Display X Size
	DY	=	Display Y Size
	BPP	=	Bits per pixel colour depth (8, 16 or 24)
/S	Silent – doesn't display any information while loading Windows CE		
/V	Displays extra status information while loading Windows CE		

Registry

The registry is saved as a file by calling the FlushToFile Utility.

Further Information :
Details of the FlushToFile() function to save the registry can be found in the ArcomUtl section of this manual.

The saved registry is restored using a utility called cereg. This utility takes the name of the registry file as a parameter, and is called from the autoexec.bat file, prior to bootce.

Note:
CE will boot with default registry settings if the cereg isn't called prior to bootce, or if it is given an invalid registry file name.

Video Modes

The video mode can be changed by changing the /L parameters of the boot loader, as detailed in the boot loader section.

The colour resolutions available for various resolutions are detailed below:

Resolution	Bits Per Pixel
640x480	8 / 16 / 24
800x600	8 / 16
1024x768	8

Flat Panels

The default BIOS can be used to drive a 6.5" NEC TFT flat panel
(Part number NL6448AC20-08), as supplied in the flat panel variant of the kit, or a 10.4" NEC TFT flat panel (Part Number NL6448AC33-18/-27/-29).

To use the supplied 6.5" flat panel, connect the flat panel as described in 'Task 1: Powering up the ELAN-NC Board'

The 10.4" panel can be connected as in 'Task 1' but requires an external 12v Supply.

Note:
The default BIOS for the ELAN-NC sets the output to CRT only mode if a CRT monitor is detected when the board is switched on. Therefore, to use a flat panel, make sure that a CRT monitor is NOT connected.

For details on other flat panels supported by the ELAN-NC, contact Arcom Control Systems Technical Support Department.

Touchscreens

For details of supported Touchscreen controllers, and driver applications, use the link provided in the Touchscreen section of the Arcom Windows CE CD-ROM.

Hard Drives

To use a hard drive the following steps must be carried out.
Note – The Hard drive must be Formatted with a FAT-16 partition, and make to boot into DOS.

- Patch the registry using the ATAPI.reg registry patch.
  Details of how to patch the registry can be found in the support software section of this manual.

- Copy the ATAPIPCI.dll & startdev.exe files from the Arcom Windows CE CD-ROM to the \FlashDisk\Arcom directory of the CE system

- Copy the Autoexec.bat & config.sys files from the Arcom Windows CE CD-ROM Hard Drive\Hard Drive files directory to the Hard drive that is to be used.

- Copy the Startup.ini file from the Arcom Windows CE CD-ROM Hard Drive directory to the \flashdisk\startup directory.

Ethernet

The current build of Windows CE 3.0 supports the on-board NE2000 compatible Ethernet Controller.

If the CE system is not connected to a network with a DHCP server running, a static IP address should be set up.

To set an IP address:

- Enter the 'Network' section of the 'Control Panel'
- Under the 'Adapters' section, select 'Realtek RTL8139 based Fast Ethernet Driver'
- Click on 'Properties'
- Click the ‘IP Address’ tab and set the ‘IP Address’, ‘Subnet Mask’ and ‘Default Gateway’ as required
- Click the ‘Name Servers’ tab, and set the ‘DNS’ as required
- Save the registry (see ‘Task 2’)
- Reboot the CE System

Note:
To check the IP configuration, run ‘ipconfig’ from the cmd window

Visual Basic

In order to use visual Basic, the following files must be copied from the Arcom Windows CE CD-ROM VB\Files directory to the Windows CE \Flashdisk\Arcom directory

pvbload.exe
vbscript.dll
pvbform2.dll
pvbhost2.dll

The 3 dll’s need to be registered using regsvrce.exe. See the Software support section for more details.

To get text to show on forms, need to add font size information, as shown below.

```
Private Sub Form_Load()
        Command1.Font.Size = 10
End Sub
```

Running Files directly from startup

In order to run files automatically when the system boots, a directory called 'startup' should be created in the '\FlashDisk' directory.

Either

- Copy the applications to be started automatically on power up to this 'startup' directory. (Command line parameters cannot be used)

Or

- Create a file called ‘startup.ini’ in the startup directory, based on the sample below (Command line parameters can be used)

Example ‘startup.ini’ file to be placed in the startup directory

```
.delay 1500
app1.exe
app2.exe /d /v:2
pvbload project1.vb
```

Note:
The .delay is used to delay the start of an application, giving the desktop time to load first.

If using a debug build of an application, any dll's associated with the application will also need to be copied into either the \flashdisk, \flashdisk\arcom, \flashdisk\user or \flashdisk\startup folder.

Using ROM-DOS

When the board boots up press either the F5 or F8 key. Pressing the F5 key will allow the user to bypass the autoexec.bat and config.sys. Pressing the F8 key will allow the user to step through the autoexec.bat and config.sys.

The user can then use the platform in a DOS environment.

Further information on ROM-DOS
Refer to the ELAN-NC Technical Manual in the documentation section of the supplied Windows CE CD-ROM

Support Libraries
for Windows CE 3.0

Introduction

Applications for Windows CE can be developed using eMbedded Visual C++ or eMbedded Visual Basic. SDK's are provided for Visual C++ and Visual Basic, to enable applications to be built for Windows CE running on the ELAN-NC board. In order to provide support for the on-board watchdog and the AIM104 range of PC/104 expansion boards a set of DLL's are provided.

General Purpose IO

The general purpose IO DLL (gpio.dll) provides functions that allow direct manipulation of IO registers. Application programs can use these functions to drive expansion boards or some of the special features in the on-board chipsets. NOTE: care should be taken when manipulating IO registers as incorrect settings may result in system failure. The gpio.dll does not provide resource locking, in multi-threaded applications care should be taken around operations that involve reading, modifying, and writing back values or when accessing chipsets that use indexed register addressing. The gpio.dll provides the following functions.

long IOReadByte(long address)

Parameters

long address The address of the location to be read

Return value

The data read from the requested location. NOTE: the data is not sign extended.

Description

This function reads a single byte (8 bit value) from the given address.

long IOReadWord(long address)

Parameters

long address The address of the location to be read

Return value

The data read from the requested location. NOTE: the data is not sign extended.

Description

This function reads a single word (16 bit) value from the given address.

long IOReadLong(long address)

Parameters

long address The address of the location to be read

Return value

The data read from the requested location.

Description

This function reads a single long (32 bit) value from the given address.

void IOWriteByte(long address, long data)

Parameters

long address	The address of the location to be read
long data	The data to be written

Return value

None

Description

This function writes a single byte (8 bit) value to the given address. Only the least significant 8 bits of the given data are used.

void IOWriteWord(long address, long data)

Parameters

long address	The address of the location to be read
long data	The data to be written

Return value

None

Description

This function writes a single word (16 bit) value to the given address. Only the least significant 16 bits of the given data are used.

void IOWriteLong(long address, long data)

Parameters

long address	The address of the location to be read
long data	The data to be written

Return value

None

Description

This function writes a single long (32 bit) value to the given address.

Watchdog Support

The ELAN-NC includes a hardware watchdog with a timeout of either 2 or 8 seconds depending on the position of LK8. The Arcom watchdog DLL (watchdog.dll) provides a simple set of functions that allows the watchdog to be controlled by application programs. Application programs should set the required timeout value with a call to SetWatchdogTimeout, the enable the watchdog with a call to EnableWatchdog. The program should then make regular calls to ToggleWatchdog within the timeout period.

long EnableWatchdog(void)

Parameters

None

Return Values

WD_NOERROR	Success call has completed OK
WD_HARDWAREREAD	Low level problem accessing the watchdog hardware
WD_HAWRWAREWRITE	Low level problem accessing the watchdog hardware

Description

This function enables the watchdog. The watchdog will not trigger until a call to this function has been made. The timeout value will be the one set by a previous call to SetWatchdogTimeout. If there have been no previous calls to SetWatchdogTimeout the timeout will be the longest value available on the hardware being used. A call to EnableWatchdog while it is already enabled will have no effect.

long ToggleWatchdog(void)

Parameters

None

Return Values

WD_NOERROR	Success call has completed OK
WD_HARDWAREREAD	Low level problem accessing the watchdog hardware
WD_HAWRWAREWRITE	Low level problem writing to watchdog hardware

Description

This function is used to reset the watchdog. A call to this function resets the timeout to the value previously given in a call to SetWatchdogTimeout. This function should be called periodically from the application. In multi-threaded applications this function should only be called by one of the threads, this thread in turn should ensure that all other threads are running correctly.

AIM104 Board Support.

The Arcom AIM104 DLL (aim104.dll) provides a simple set of functions that allows Arcom's AIM104 range of PC/104 expansion cards to be driven from application programs.

AIM104 Relay8/In8

The AIM104 Relay 8 offers 8 opto isolated inputs and 8 single pole double throw relay outputs. The AIM104 DLL considers each relay 8 to have one group of 8 relays and one group of 8 inputs. In a multi-board system the first board will contain relay & input group 0 the second board relay & input group 1 etc. Similarly the first board will contain relay or input 0 through 7, the second will contain 8 through 15 etc.

long Relay8Enable(long nGroup, long nState)

Parameters

nGroup	The relay group to be enabled or disabled
nState	Zero to disable or non zero to enable the group

Return Values

R8_NOERROR	Success call has completed OK
R8_BADGROUP	nGroup value greater that the number of boards installed
R8_HAWRWAREWRITE	Low level problem accessing the hardware

Description

This function is used to enable or disable a group of relays. After a system reset all relays are disabled. Calling this function with a non-zero value for nState will enable all the relays in the given group, the relays will immediately take up the value in the relay output register. Calling this function with a value of zero for nState will disable all the relays in the given group, the relays will be switched off immediately but value in the relay output register will not be changed. Enabling a group that is already enabled or disabling a group that is already disabled will have no effect.

long Relay8RelayWrite(long nRelay, long nState)

Parameters

nRelay	The relay group to be written to
nState	Zero to turn off or non zero to turn on the relay

Return Values

R8_NOERROR	Success call has completed OK
R8_BADRELAY	nRelay value greater that the number of available relays
R8_HAWRWAREWRITE	Low level problem accessing the hardware

Description

This function is used to set the state of an individual relay without effecting any of the other relays in the group. A non-zero value for nState will turn on the given relay, a zero value will turn it off. If the corresponding group is disabled at the time it remains disabled but the new value is written into the appropriate output register. Turning on a relay that is already on or turning off one that is already off has no effect.

long Relay8GroupWrite(long nGroup, long nData)

Parameters

nGroup	The relay group to be written to
nData	New value for the outputs (lower 8 bits only)

Return Values

R8_NOERROR	Success call has completed OK
R8_BADGROUP	nGroup value greater that the number of boards installed
R8_HAWRWAREWRITE	Low level problem accessing the hardware

Description

This function allows all eight relays in a group to be written to simultaneously. The lower 8 bits of nData will be written to the relay output register, when a bit is set the corresponding relay will be turned on. If the group is disabled at the time the new value is written to the output register but the relays remain disabled. Only the lower 8 bits of nData are significant.

long Relay8RelayStatus(long nRelay, long *pStatus)

Parameters

nRelay	The number of the relay to be read
pStatus	Pointer to a variable to receive the state of the channel

Return Values

R8_NOERROR	Success call has completed OK
R8_BADRELAY	nRelay value greater that the number of available relays
R8_HARDWAREREAD	Low level problem accessing the hardware

Description

This function is used to test the state of a particular channel in the output register. If the value placed in the variable pointed to by pStatus is zero the corresponding channel is off, a non-zero value indicates that the channel is on. This function returns the value in the output register even if the corresponding group is disabled at the time.

long Relay8GroupStatus(long nGroup, long *pStatus)

Parameters

nGroup	The relay group to be written to
pStatus	Pointer to a variable to receive the state of the channel

Return Values

R8_NOERROR	Success call has completed OK
R8_BADGROUP	nGroup value greater that the number of boards installed
R8_HAWRWAREWRITE	Low level problem accessing the hardware

Description

This function returns the current state of the output register. The value in the lower 8 bits placed into the variable pointed to by pStatus reflect the current state of the output register, a zero bit indicates that the corresponding channel is off. The higher order bits will always be zero. The value of the output register is returned even if the corresponding group is disabled.

long Relay8ReadInput(long nInput, long *pState)

Parameters

nInput	The input to be read
pState	Pointer to a variable to receive the state of the channel

Return Values

R8_NOERROR	Success call has completed OK
R8_BADINPUT	nInput value greater that the number of available inputs
R8_HARDWAREREAD	Low level problem accessing the hardware

Description

This function returns the state of an individual opto-isolated input. If the value placed in the variable pointed to by pState is non-zero the corresponding input is on, if the value is zero the input is off.

long Relay8GroupInput(long nGroup, long *pState)

Parameters

nGroup	The relay group to be written to
pState	Pointer to a variable to receive the state of the channel

Return Values

R8_NOERROR	Success call has completed OK
R8_BADGROUP	nGroup value greater that the number of boards installed
R8_HARDWAREREAD	Low level problem accessing the hardware

Description

This function allows all eight opto-isolated inputs in a group to be read simultaneously. The lower 8 bits of the value place in the variable pointed to by pState indicate the state of each of the eight channels. If a bit is zero the corresponding input is off. NOTE: this is inverse of the actual data read back from the hardware but having a consistent 'zero means off' improves compatibility with other routines.

AIM104 IN16

The AIM104 IN 16 offers 16 opto-isolated inputs in two groups of 8. The AIM104 DLL considers each IN 16 to have two groups of 8 inputs. In a multi-board system the first board will contain input groups 0 and 1 the second board input groups 2 and 3 etc. Similarly the first board will contain individual inputs 0 through 15, the second will contain 16 through 31 etc.

long IN16ReadInput(long nInput, long *pState)

Parameters

nInput	The input to be read
pState	Pointer to a variable to receive the state of the input

Return Values

IN16_NOERROR	Success call has completed OK
IN16_BADINPUT	nInput value greater that the number of available inputs
IN16_HARDWAREREAD	Low level problem accessing the hardware

Description

This function returns the state of an individual opto-isolated input. If the value placed in the variable pointed to by pState is non-zero the corresponding input is on, if the value is zero the input is off.

long IN16GroupInput(long nGroup, long *pState)

Parameters

nGroup	The input group to be read
pState	Pointer to a variable to receive the state of the channel

Return Values

IN16_NOERROR	Success call has completed OK
IN16_BADGROUP	nGroup value greater that the number of boards installed
IN16_HARDWAREREAD	Low level problem accessing the hardware

Description

This function allows a group of eight opto-isolated inputs to be read simultaneously. The lower 8 bits of the value place in the variable pointed to by pState indicate the state of each of the eight channels. If a bit is zero the corresponding input is off. NOTE: this is inverse of the actual data read back from the hardware but having a consistent 'zero means off' improves compatibility with other routines.

AIM104 OUT16

The AIM104 IN 16 offers 16 opto-isolated outputs in two groups of 8. The AIM104 DLL considers each OUT16 to have two groups of 8 outputs. In a multi-board system the first board will contain output groups 0 and 1 the second board input groups 2 and 3 etc. Similarly the first board will contain individual outputs 0 through 15, the second will contain 16 through 31 etc.

long OUT16WriteOutput(long nOutput, long nState)

Parameters

nOutput	The output to be written
nState	New state of the output zero for off non zero for on

Return Values

OUT16_NOERROR	Success call has completed OK
OUT16_BADINPUT	nOutput greater that the number of available outputs
OUT16_HARDWAREWRITE	Low level problem accessing the hardware

Description

This function is used to set the state of an individual output without effecting any of the other output in the group. A non-zero value for nState will turn on the given output, a zero value will turn it off. If the corresponding group is disabled at the time it remains disabled but the new value is written into the appropriate output register. Turning on an output that is already on or turning off one that is already off has no effect.

long OUT16GroupWrite(long nGroup, long nData)

Parameters

nGroup	The input group to be read
nData	New data for the output group

Return Values

OUT16_NOERROR	Success call has completed OK
OUT16_BADGROUP	nGroup value greater that the number of boards installed
OUT16_HARDWAREREAD	Low level problem accessing the hardware

Description

This function allows all eight outputs in a group to be written to simultaneously. The lower 8 bits of nData will be written to the output register, when a bit is set the corresponding relay will be turned on. If the group is disabled at the time the new value is written to the output register but the relays remain disabled. Only the lower 8 bits of nData are significant.

long OUT16ReadStatus(long nOutput, long *pStatus)

Parameters

nOutput	The number of the output to be read
pStatus	Pointer to a variable to receive the state of the channel

Return Values

R8_NOERROR	Success call has completed OK
R8_BADOUTPUT	nOutput value greater that the number of available relays
R8_HARDWAREREAD	Low level problem accessing the hardware

Description

This function is used to test the state of a particular channel in the output register. If the value placed in the variable pointed to by pStatus is zero the corresponding channel is off, a non-zero value indicates that the channel is on. This function returns the value in the output register even if the corresponding group is disabled at the time.

long OUT16GroupStatus(long nGroup, long *pStatus)

Parameters

nGroup	The relay group to be written to
pStatus	Pointer to a variable to receive the state of the channel

Return Values

R8_NOERROR	Success call has completed OK
R8_BADGROUP	nGroup value greater that the number of boards installed
R8_HARDWAREREAD	Low level problem accessing the hardware

Description

This function returns the current state of the output register. The value in the lower 8 bits placed into the variable pointed to by pStatus reflect the current state of the output register, a zero bit indicates that the corresponding channel is off. The higher order bits will always be zero. The value of the output register is returned even if the corresponding group is disabled.

AIM104 IO32

The AIM104 IO offers 32 TTL level open collector outputs. The state of each output can also be monitored, when an output is off it may be driven by an external source allowing the line to be used for input. The AIM104 DLL considers each IO 32 to have four groups of 8 IO lines, each group may be accessed as inputs or outputs. In a multi-board system the first board will contain groups 0 through 3 the second board groups 4 through 7 etc. Similarly the first board will contain individual IO lines 0 through 31, the second will contain 32 through 63 etc.

long IO32Enable(long nGroup, long nState)

Parameters

nGroup	The IO group to be enabled or disabled
nState	Zero to disable or non zero to enable the group

Return Values

IO32_NOERROR	Success call has completed OK
IO32_BADGROUP	nGroup value greater that the available groups
IO32_HAWRWAREWRITE	Low level problem accessing the hardware

Description

This function is used to enable or disable a group of outputs. After a system reset all outputs are disabled. The hardware on the IO32 is only capable of enabling all four groups simultaneously but to improve compatibility with other boards this function still considers each board to have four groups. Enabling any group on a board enables the other three as well. Calling this function with a non-zero value for nState will enable all three groups, the outputs will immediately take up the value in the output registers. Calling this function with a value of zero for nState will disable all three groups, the outputs will be switched off immediately but value in the output registers will not be changed. Enabling a group that is already enabled or disabling a group that is already disabled will have no effect.

long IO32WriteOutput(long nOutput, long nState)

Parameters

nOutput	The output to be written
nState	New state of the output zero for off non zero for on

Return Values

IO32_NOERROR	Success call has completed OK
IO32_BADINPUT	nOutput greater that the number of available outputs
IO32_HARDWAREWRITE	Low level problem accessing the hardware

Description

This function is used to set the state of an individual output without effecting any of the other output in the group. A non-zero value for nState will turn on the given output, a zero value will turn it off. If the corresponding group is disabled at the time it remains disabled but the new value is written into the appropriate output register. Turning on an output that is already on or turning off one that is already off has no effect.

long IO32GroupWrite(long nGroup, long nData)

Parameters

nGroup	The input group to be read
nData	New data for the output group

Return Values

IO32_NOERROR	Success call has completed OK
IO32_BADGROUP	nGroup value greater that the number of boards installed
IO32_HARDWAREREAD	Low level problem accessing the hardware

Description

This function allows all eight outputs in a group to be written to simultaneously. The lower 8 bits of nData will be written to the output register, when a bit is set the corresponding output will be turned on. If the group is disabled at the time the new value is written to the output register but the outputs remain disabled. Only the lower 8 bits of nData are significant.

long IO32ReadStatus(long nOutput, long *pStatus)

Parameters

nOutput	The number of the output to be read
pStatus	Pointer to a variable to receive the state of the channel

Return Values

IO32_NOERROR	Success call has completed OK
IO32_BADOUTPUT	nOutput value greater that the number of outputs
IO32_HARDWAREREAD	Low level problem accessing the hardware

Description

This function is used to test the state of a particular channel in the output register. If the value placed in the variable pointed to by pStatus is zero the corresponding channel is off, a non-zero value indicates that the channel is on. This function returns the value in the output register even if the corresponding group is disabled at the time.

long IO32GroupStatus(long nGroup, long *pStatus)

Parameters

nGroup	The relay group to be written to
pStatus	Pointer to a variable to receive the state of the channel

Return Values

IO32_NOERROR	Success call has completed OK
IO32_BADGROUP	nGroup value greater that the available groups
IO32_HARDWAREREAD	Low level problem accessing the hardware

Description

This function returns the current state of the output register. The value in the lower 8 bits placed into the variable pointed to by pStatus reflect the current state of the output register, a zero bit indicates that the corresponding channel is off. The higher order bits will always be zero. The value of the output register is returned even if the corresponding group is disabled.

long IO32ReadInput(long nInput, long *pState)

Parameters

nInput	The input to be read
pState	Pointer to a variable to receive the state of the input

Return Values

IO32_NOERROR	Success call has completed OK
IO32_BADINPUT	nInput value greater that the number of available inputs
IO32_HARDWAREREAD	Low level problem accessing the hardware

Description

This function returns the state of an individual input. If the value placed in the variable pointed to by pState is non-zero the corresponding input is on, if the value is zero the input is off.

long IO32GroupInput(long nGroup, long *pState)

Parameters

nGroup	The input group to be read
pState	Pointer to a variable to receive the state of the channel

Return Values

IO32_NOERROR	Success call has completed OK
IO32_BADGROUP	nGroup value greater that the number of boards installed
IO32_HARDWAREREAD	Low level problem accessing the hardware

Description

This function allows a group of eight inputs to be read simultaneously. The lower 8 bits of the value place in the variable pointed to by pState indicate the state of each of the eight channels. If a bit is zero the corresponding input is off. The upper bits of the value in pState will always be zero.

AIM104 MULTI IO

The AIM104-MULTI-IO provides 8 opto-isolated digital inputs, 2 analogue outputs (Voltage or Current loop) and 16 single ended or 8 differential analogue inputs.

The code that implements the API must accommodate multiple boards but can assume that they will be contiguous in memory, with boards containing only single ended Analogue Inputs first, then boards containing Differential Analogue inputs.
In a multi-board system the first board will contain analogue output groups 0 and 1, the second board groups 2 and 3 etc. Similarly the first board will contain digital inputs 0 through 7, the second will contain 8 through 15 etc. If the first board is set for single ended analogue inputs, it will contain channels 0 through 15, a second board configured for single ended analogue inputs will contain channels 16 through 31 etc. A board configured for differential analogue inputs will contain channels 0 through 7 (even if there are boards configured for single ended analogue inputs in the system). A second board configured for differential analogue inputs will contain channels 8 through 15.
The implementation routines must be thread and process safe. Where appropriate implementation routines should check that the group or input number given is within the number of boards installed in the system.

long MULTIIODAC(long nChan, long nData)

Parameters

nChan	The Channel to be written to
nData	The value to be written 0 through 4095

Return Values

MULTIIO_NOERROR	Success call has completed OK
MULTIIO_BADCHAN	nChan greater that the number of available channels

Description

This function is used to set the value of an analogue output. An nData value of 0 will result in a –5V output, an nData value of 2048 will result in a 0V output and an nData value of 4095 will result in a +5V output.

A value of 0 or 1 on nChan will use channels 0 or 1 on the board at the base address.
A value of 2 or 3 will use the channels 0 or 1 on the second board, which must be 4 addresses from the board at the base address.
A value of 4 or 5 will use the channels 0 or 1 on the third board, which must be 8 addresses from the board at the base address.

long MULTIIOADC(long nChan, bool SingleEnded, long *pValue)

Parameters

nChan	The Channel to be read
SingleEnded	True if using Single ended inputs. False if using Differential inputs
pValue	Pointer to a variable to receive the input

Return Values

MULTIIO_NOERROR	Success call has completed OK
MULTIIO_BADCHAN	nChan value greater that the number of boards installed
MULTIIO_BADDIFCHAN	nChan value greater than the number of differential boards installed

Description

Single Ended mode – SingleEnded set as TRUE

A value of 0 through 15 on nChan will use channels 0 through 15 on the board at the base address.
A value of 16 through on nChan 31 will use the channels 0 through 15 on the second board, which must be 4 addresses from the board at the base address.
A value of 32 through 47 on nChan will use the channels 0 through 15 on the third board, which must be 8 addresses from the board at the base address.

An input value of –5V will result in 0 being placed in the variable pointed to by pValue.
An input value of 0V will result in 2048 being placed in the variable pointed to by pValue.
An input value of +5V will result in 4095 being placed in the variable pointed to by pValue.

Differential mode – SingleEnded set as FALSE

A value of 0 through 7 on nChan will use channels 0 through 7 on the board at the base address.
A value of 8 through 15 on nChan will use the channels 0 through 7 on the second board, which must be 4 addresses from the board at the base address.

A value of 16 through 23 on nChan will use the channels 0 through 7 on the third board, which must be 8 addresses from the board at the base address.

A differential input value of –5V will result in 0 being placed in the variable pointed to by pValue.
A differential input value of 0V will result in 2048 being placed in the variable pointed to by pValue.
A differential input value of +5V will result in 4095 being placed in the variable pointed to by pValue.

long MULTIIOReadInput(long nInput, long *pStatus)

Parameters

nInput	The number of the input to be read
pStatus	Pointer to a variable to receive the state of the input

Return Values

MULTIIO_NOERROR	Success call has completed OK
MULTIIO_BADINPUT	nInput value greater that the number of inputs

Description

This function returns the state of a single opto-isolated digital input.

A value of 0 through 8 on nInput will use input 0 through 8 on the board at the base address.
A value of 9 through 16 on nInput will use the input 0 through 8 on the second board, which must be 4 addresses from the board at the base address.
A value of 17 through 24 on nInput will use the input 0 through 8 on the third board, which must be 8 addresses from the board at the base address.

If the value placed in the variable pointed to by pStatus is non-zero the corresponding input is on. If the value is zero, the corresponding input is off.

long MULTIIOReadGroup(long nGroup, long *pStatus)

Parameters

nGroup	The input group to be read
pStatus	Pointer to a variable to receive the state of the group

Return Values

IO32_NOERROR	Success call has completed OK
IO32_BADGROUP	nGroup value greater that the number of groups

Description

This function allows a group of eight opto-isolated inputs to be read simultaneously

A value of 0 on nGroup will read the inputs on the board at the base address.
A value of 1 on nGroup will read the inputs on the second board, which must be 4 addresses from the board at the base address.
A value of 2 on nGroup will read the inputs on the third board, which must be 8 addresses from the board at the base address.

The value read is placed in the variable pointed to by pStatus. If a bit is set the corresponding input is on. If a bit is zero, the corresponding input is off. NOTE: This is the inverse of the actual data read back from the hardware but having a consistent ‘zero means off’ improves compatibility with other routines.

ArcomUtl

ArcomUtl contains 3 utilities, one to Flush the registry to a specified file name, one to return the Board name and one to return a build number.

Registry Saving

int FlushToFile(TCHAR * pFilename)

Parameters

pFilename	The path and the name of the file the registry is to be stored in

Return Values

AU_NOERROR	The file write didn't return an error
AU_FLUSHKEYERR	WindowsCE RegFlushKey() function returned an error
AU_INVALID_FILE_HANDLE	File Creation returned an INVALID_FILE_HANDLE
AU_WRITEFILEERR	Write to file failed

Usage

```
int RetVal;

RetVal = FlushToFile(TEXT("\\Flashdisk\\regfile.bin"));
```

Board Identification

int GetBoardType(TCHAR * pBoardName)

Parameters

pBoardName	The place the board name will be stored

Return Values

AU_NOERROR	The call didn't return an error
AU_NO_BOARD_ID	No Board ID was available
AU_GET_BOARD_TYPE_ERR	Call to IOCTL_HAL_GET_BOARD_TYPE failed

Usage

```
TCHAR Name[20];

GetBoardType(Name);
if(wcscmp(Name, (_T("ElanNC")))  ==0)
{
// Do Something;
}
```

Build Identification

Int GetBuildNum(DWORD * pBuildNum)

Parameters

pBuildNum	The place the build number will be stored

Return Values

AU_NOERROR	The call didn't return an error
AU_GET_BOARD_TYPE_ERR	Call to IOCTL_HAL_GET_BOARD_TYPE failed

Usage

```
DWORD Num;

GetBuildNum(&Num);
printf("\n\nBuild Number is %04X\n", Num);
```

Use with eMbedded Visual C++

Install Microsoft eMbedded Visual C++ according to Microsoft's instructions

Note:
The SDK's supplied with eMbedded Visual Tools are not required, but installing the H/PC Pro SDK will allow application emulation on the host system

Emulation of Arcom Hardware is not possible

Arcom Platform SDK
The Arcom Platform SDK for Visual C++ must be installed in order to build applications for the Arcom ELAN-NC board.

Select the Platform SDK section of the Arcom Windows CE CD ROM, then install the Visual C++ SDK

Arcom dll support
In order to use the support libraries DLL's

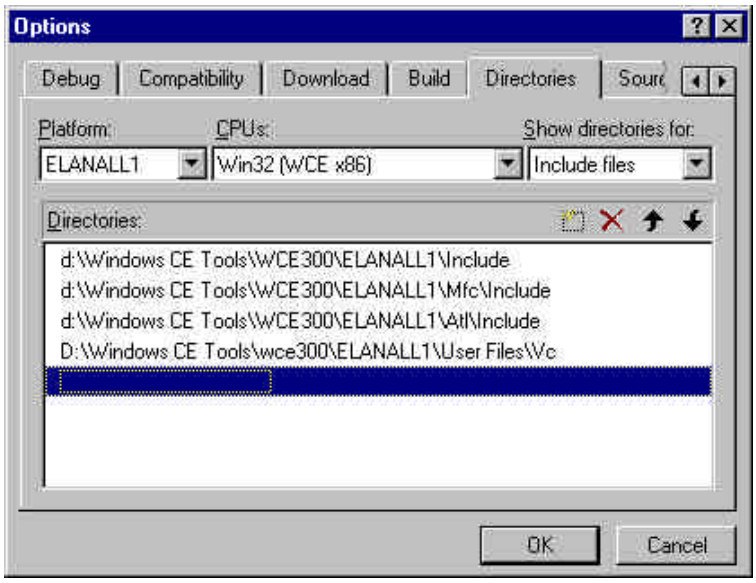
- Make sure the spinlock.dll, gpio.dll watchdog.dll and aim104.dll files are in the \FlashDisk\Arcom directory on the CE system
  They can be copied from the Windows CE CD ROM Support Library\DLL Files directory

- Copy the relevant registry patch from the Windows CE CD ROM Registry Tools\Registry Patch Files directory to a Temp directory on your host PC

- Modify the registry patch to reflect :
  The base address of the first board of that type used
  The number of boards of that type used

- Patch the modified registry patch into the CE systems registry using the regpatch application.
  Details of how to do this can be found in the support software section.

- Make sure the ELANALL1 SDK has been installed on your host system
- Select Options from the Tools menu, then select the ‘directories’ tab.

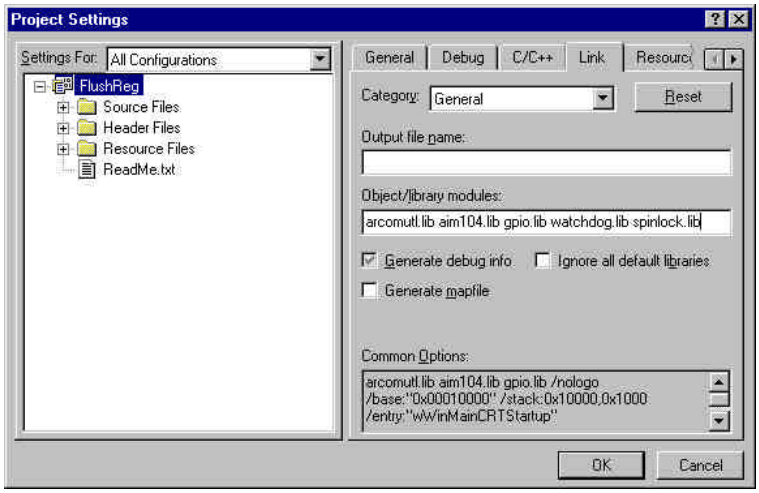


In the ‘Platform’ list box, select the installed ELANALL1 SDK

In the 'Show directories for' list box Select 'Include files' then add the path to the ‘User Files\VC’ direcory of the place where the SDK was installed. It is likely to be a similar path to that shown here, but it depends on where the user chose to install the SDK, and therefore must be checked.

In the 'Show directories for' list box Select 'Library files' and add the same path to the SDK’s ‘User Files\VC’ as that described above.

- Programs must also be linked with the appropriate library file.
  To do this, select Settings from the Project menu and click on the 'Link' tab.



Ensure that 'All configurations' is shown in the 'Settings For' list box.
Add ArcomUtl.lib gpio.lib splinlock.lib watchdog.lib and aim104.lib to the 'Object/Library modules' list.

- Include the relevant .h as a `#include <filename.h>` statement in your codenameDlg.cpp file.

Use with eMbedded Visual Basic

Install Microsoft eMbedded Visual Basic according to Microsoft's instructions

Note:
The SDK's supplied with eMbedded Visual Tools are not required, but installing the H/PC Pro SDK will allow application emulation on the host system

Emulation of Arcom Hardware is not possible

Arcom Platform SDK
The Arcom Platform SDK for Visual Basic must be installed in order to build applications for the Arcom ELAN-NC board.

Select the Platform SDK section of the Arcom Windows CE CD ROM, then install the Visual Basic SDK

Arcom dll support

The Arcom support DLLs can be used directly from Microsoft's Visual Basic.

In order to use the support libraries DLL's

- Copy the relevant dll (gpio.dll, watchdog.dll, or aim104.dll) from the Windows CE CD ROM Support Library\DLL Files directory to the \FlashDisk\Arcom directory on the CE system.

- Copy the relevant registry patch from the Windows CE CD ROM Registry Tools\Registry Patch Files directory to a Temp directory on your host PC
  Modify the registry patch to reflect :
  1. The base address of the first board of that type used
  2. The number of boards of that type used

- Patch the modified registry patch into the CE systems registry using the regpatch application.
  Details of how to do this can be found in the support software section.

- Declare the library functions used in the source code
  The code extract below shows how the AIM104 DLL functions are declared in Visual Basic.

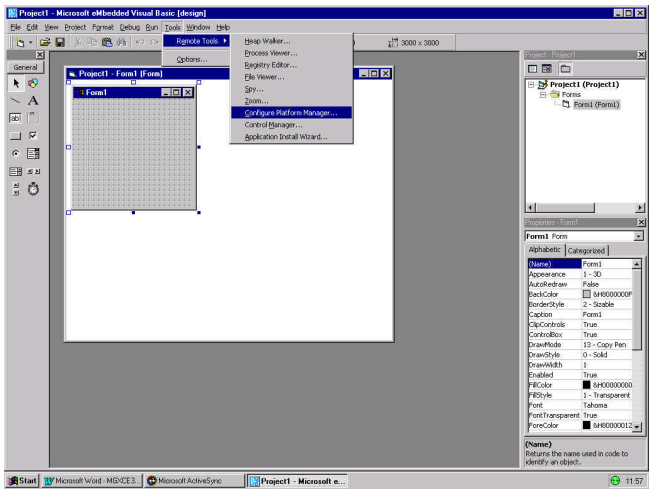
```
Option Explicit

Declare Function IO32GroupStatus Lib "aim104.dll" (ByVal nGroup As Long,
pStatus As Long) As Long

Declare Function IN16ReadInput Lib "aim104.dll" (ByVal nInput As Long,
pState As Long) As Long
```

Note:
- Visual Basic treats 32 bit values as long not integer
- Parameters must be passed by value unless really are pointers.
- To run an eMbedded visual basic application use pvbload projectname.vb

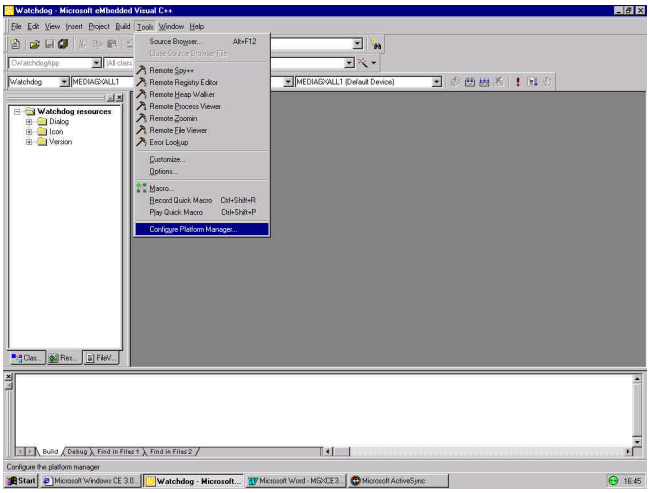
Establishing a remote debugging connection

Connect the Host system and CE system using an ActiveSync connection, as described in the Quick start manual.
When an ActiveSync connection has been made, select Configure Platform Manager from the eMbedded Visual C++ Tools menu, or from the eMbedded Visual Basic Tools, Remote Tools menu

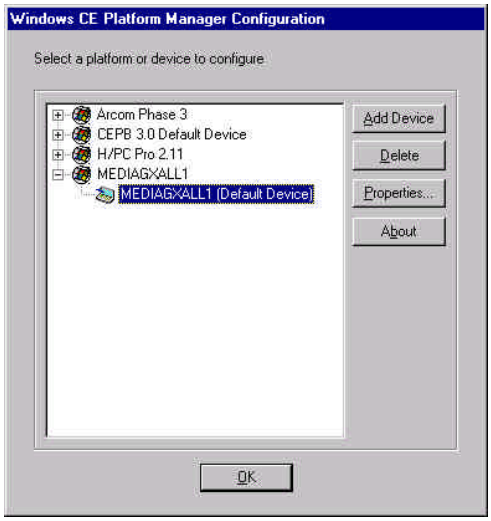
eMbedded Visual Basic



eMbedded Visual C++

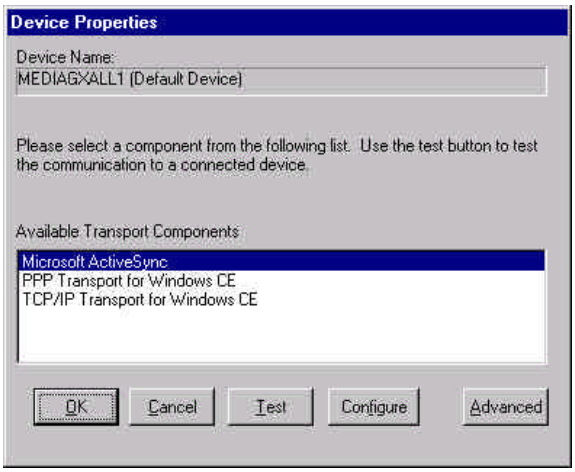


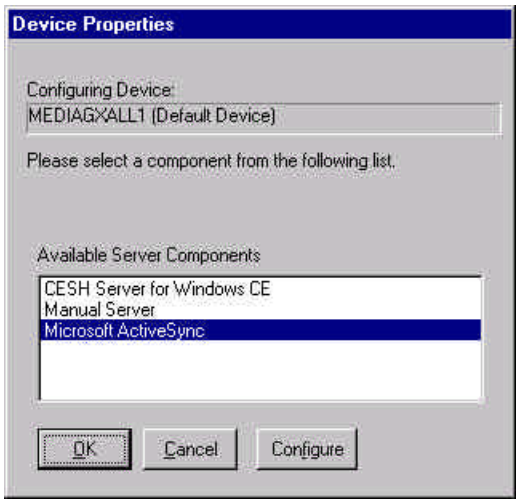
.



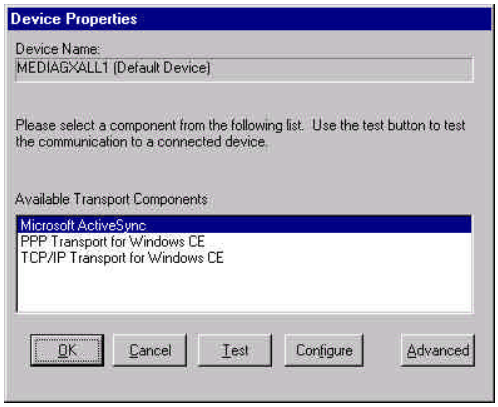
Open the SDK that will be used and select the default device .
Click on properties

Select ActiveSync as the Available Transport component.
Click on Advanced





Select ActiveSync as the Available Server component.
Click on OK



Click on the Test Box

The connection will be established

When a connection has been established, close all windows to get back to the eMbedded Visual Tools development environment.

Support software

The following Windows CE tools are provided to support programmers working with Windows CE on Arcom hardware. They are intended for use from the cmd.exe command line:

regsvrce.exe
This executable registers DLLs in the same way as regsvr32.exe does for Windows 95 / NT. In particular, this is useful for registering DLLs which are required for VB to operate on embedded Windows CE hardware. Type regsvrce followed by the pathname of the DLL which you wish to register.

regpatch.exe
This is provided to aid the task of applying changes to the registry to enable or disable features.

The patches and full details of their functions can be found on the Windows CE CD-ROM under registry patches.

To run a registry patch:

- Copy the regpatch.exe file (and mfcce211.dll & arcom.dll), and required patchfile.reg file to the FlashDisk directory of the Windows CE system
  This can be achieved by using ActiveSync to copy the files serially, or using a floppy drive and booting into ROM-DOS
- Click on the 'Run' button
- In the ‘Run’ dialogue box, Type \FlashDisk\regpatch \FlashDisk\patchfilename.reg
- Flush the registry (as described in ‘Task 2’)

flushreg.exe

This copies the current registry to persistent storage, using the ArcomFlushReg function.
An example of using this application can be found in ‘Task 2’ of the ‘Quick Start Manual’ section.
More information on the Persistent registry can be found in the ‘Data Manual’ section.