Board Support Package
ModARM9 for Windows CE V5.00

Board Support Package ModARM9 for Windows CE 5.00

Installation and user's guide

© Copyright 2005:

FS Forth-Systeme GmbH
Postfach 1103, D-79200 Breisach a. Rh., Germany

Release of Document: February 28, 2005
Filename: A9M9xx0CE5
Author: Mike Engel
Program Version   1.1

Table of Contents

1 History

Date	Version	Responsible	Description
29.06.2004	1.0	Mike Engel	Initial version
28.02.2005	1.1	Mike Engel	Updated to Windows CE V5.0

2 Preface

The FS Forth-Systeme Board Support Packages for the ModARM9 family for Windows CE 5.00 contains all the necessary software components to allow a simple and fast start-up of application development with Windows CE 5.00 hardware platform.

The Board Support Packages offers the possibility to reduce the time to market phase for software basing on Windows CE 5.00 on the A9M9750 and A9M9360module. With the integrated Ethernet controller NS9750 on the A9M9750 and NS9360 on the A9M9360 *Downloader and Debugger* from FS Forth-Systeme GmbH, you can now download and debug your platform without the need of additional network cards.

A PC with Windows 2000 Professional is required.

3   License Backround

The BSP includes the full source code of all drivers and the boot loader. The source code is intended to be used internally only e.g. for modifications and debugging. Distribution of the source code and modifications based on it is prohibited.

There are no royalties for Windows CE images created with the BSP, running on hardware purchased from FS Forth-Systeme GmbH. There are royalties for images running on other hardware, not purchased from FS Forth-Systeme GmbH. Royalties accrue e.g. for boot loaders and drivers.

For detailed Information about Licensing and Royalties please contact FS Forth-Systeme GmbH.

4   Contents of the CD

The supplied CD contains all software and documentation you need to start your evaluation of Windows CE 5.00 on one of the ModARM9 platforms.

🗀 Bootloader	Boot loader binaries
🗀 DOC	Documentation
🗀 GPIO	Windows CE example that shows how GPIO accesses are made from the operating system.
🗀 Images	Windows CE kernel binaries and script files
🗀 QFEPacks	Microsoft *Quick Fix Engineering* software updates for Windows CE 5.00.
🗀 SDK	Contains the Software Development Kit for a sample platform.
🗀 TFTP	TFTP Server evaluation application for Windows.
🗎 A9M9xx0.msi	Install file for the A9M9xx0 Board Support Package.
🗎 install9xx0.bat	Batch process to install the platform BSP.

5 Installation of the Board Support Package

First you have to install Microsoft Platform Builder 5.00 from the original Microsoft Windows CE 5.00 DVD.

Before installing the BSP, we have to make sure that Platform Builder is not running, and the environment variable _WINCEROOT is set to the correct path[1]. Open a DOS box and type:

```
set _WINCEROOT=C:\WINCE500
```

(substitute C:\WINCE500 with the path to your Windows CE 5.00 installation directory)

Then change to the CD drive, which holds the provided CD-ROM, and call the batch file *install9xx0.bat* .

```
C:\>set _WINCEROOT=C:\WINCE500
C:\>E:
E:\>install9xx0.bat
```

This will install first the QFE Packs (follow the screen instructions to install them).

Then it will install the corresponding BSP (a wizard will guide you through the install process).
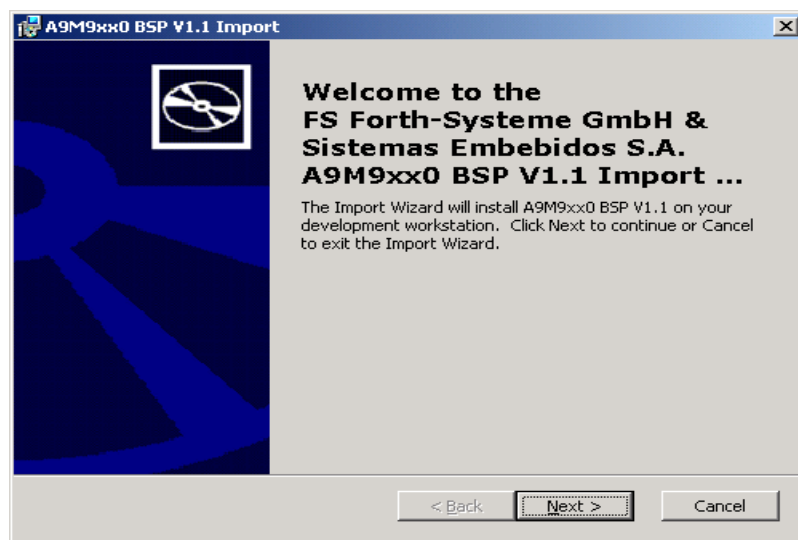
The installation of the BSP will copy the following files and folders to your hard drive:

FILE/FOLDER	PATH	DESCRIPTION
A9M9xx0.CEC	*C:\Wince500\public\common\oak\catalog\cec*	WinCE 5.00 components file
A9M9xx0	*%_WINCEROOT%\Platform*	Platform files and drivers

where %_WINCEROOT% is an environment variable of your system that stands for your Windows CE 5.00 root directory (Usually *C:\WINCE500*).

---

[1] If you have installed different versions of Windows CE this value can be incorrect.

6   Creating a new platform

From Platform Builder *File* menu, select *New Platform*.

A dialog will appear to enter the name of the project you want to do. Enter the name an select *Next*.

On the next screen all the available Board Support Packages (BSP) will appear. Select the *A9M9xx0* platform and click *Next*.
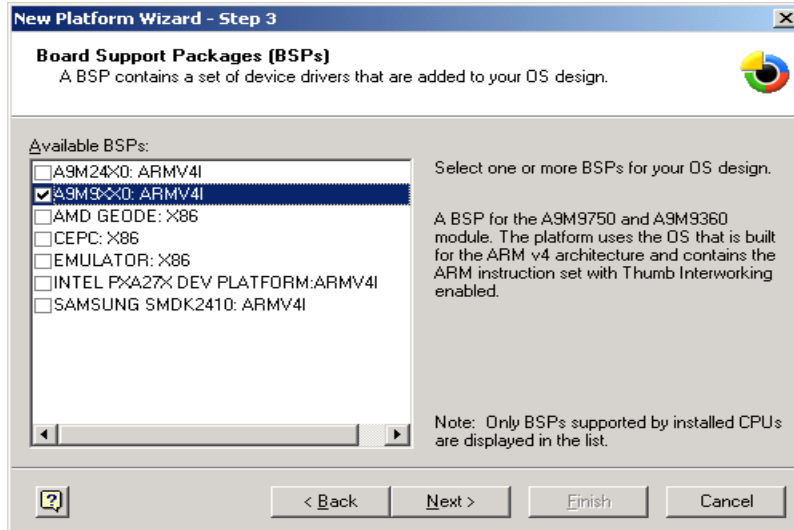
The next screen gives you the option to select an already made configuration for your platform or a personalized configuration. Select *Custom Configuration* so that you can choose only the components that you want in your final Windows CE kernel.

Then, type the name of your platform and click *Next*.
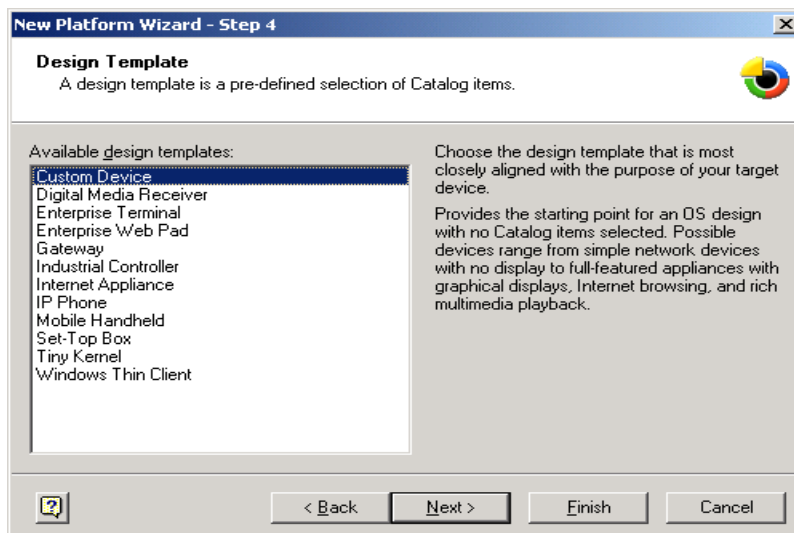
In the following steps (5 ~ 20) you can configure the components and programs that your final Windows CE image will contain.

No matter what you include, Platform Builder will add some components automatically in case they are needed.

As an example we will create a kernel for the A9M9750 development board with support for our hardware and we will include some server applications like a Telnet server for remote control of the device, an FTP server for uploading/downloading of files. The same steps have to be made for creating a A9M9360 platform project.

In step 5 you can include several Applications & Services Development components. Select under *.NET Compact Framework* the *OS Dependencies for .NET Compact Framework 1.0*.
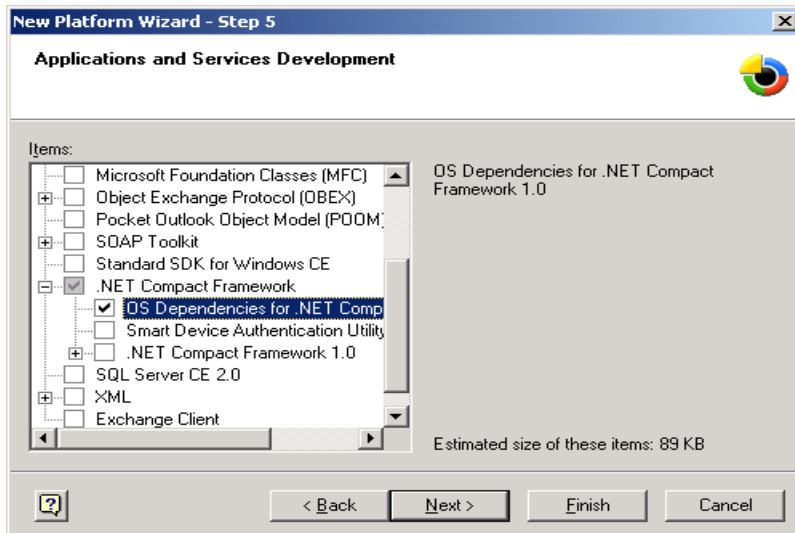
With .NET Framework support application can be written with Visual Studio .NET 2003 for smart devices in Basic or C#.

In step 6 you can include various Applications – End User components. For our example we won't need any of these, so just click on *Next*.

In step 7 you can have support for the Core OS Services. Select *USB Host support* and *Display Support* Click on *Next*. If you want to make an image without display support, do not select the display component. You can also later remove it form the project if you which.
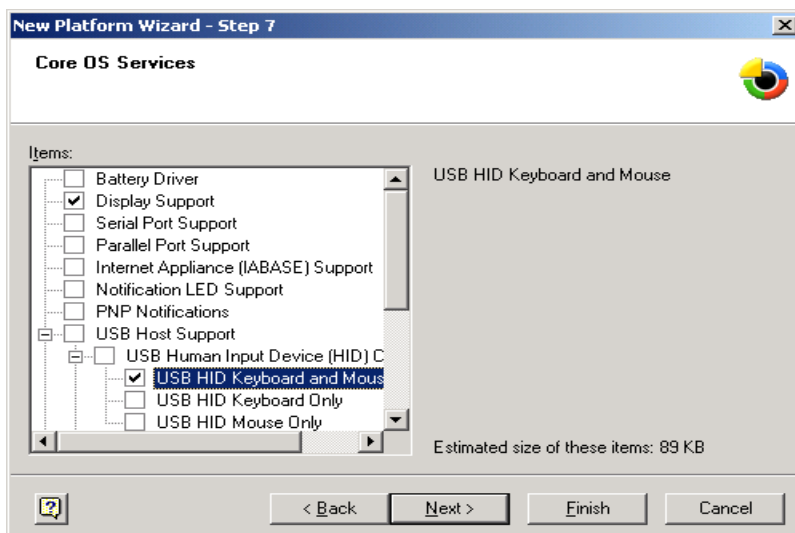
The USB Host and Display Windows CE driver will be included. The registry keys to configure them are stored in the file:

%_WINCEROOT%\Platform\A9M90\Files\Platform.reg

On step 8 you can select the Communication services and Networking. Under *Networking Features* select *Network Utilities (ipconfig, ping, …)* and *Windows Networking API/Redirect*. Under *Networking – Local Area Network (LAN)* you must include the *Wired Local Area Network (802.3, 802.5)* to have support for your Ethernet interface. Under *Servers* select *Telnet Server* and *FTP Server*. With the Telnet Server you will be able to connect to the Windows CE device shell from anywhere in your network.
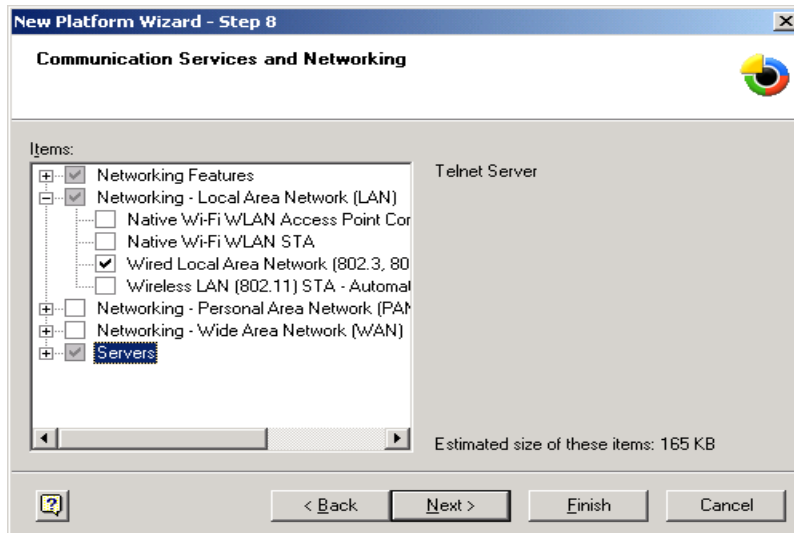
Some registry information have been included for the FTP and the TELNET servers in the file:

%_WINCEROOT%\Platform\A9M9xx0\Files\Platform.reg.

```
; TELNET SERVER
[HKEY_LOCAL_MACHINE\COMM\TELNETD]
      "IsEnabled"=dword:1
      "UseAuthentication"=dword:0      ;Don't use authentication

; FTP SERVER
[HKEY_LOCAL_MACHINE\COMM\FTPD]
      "IsEnabled"=dword:1
      "UseAuthentication"=dword:0      ;Don't use authentication
      "AllowAnonymous"=dword:1         ;Allow anonymous login
      "AllowAnonymousUpload"=dword:1   ;Allow anonymous upload of files
      "DefaultDir"="\\"                ;Root directory
```

WARNING: These registry keys give you complete access to your target via TELNET and FTP, for demonstration purposes. In order to preserve the security of your targets, change these registry keys by enabling the *UseAuthentication* key and adding a list of allowed users[2].

On step 9 you can select parts of the Device Management. Click *Next*.

In order to have support for the any storage device used on you platform, you must include the support for the *FAT File System*, under *Storage Manager*, on step 10.

---

[2] For more information about security, read the chapter *Telnet Server Authentication and Security Considerations* of Windows CE on-line help.
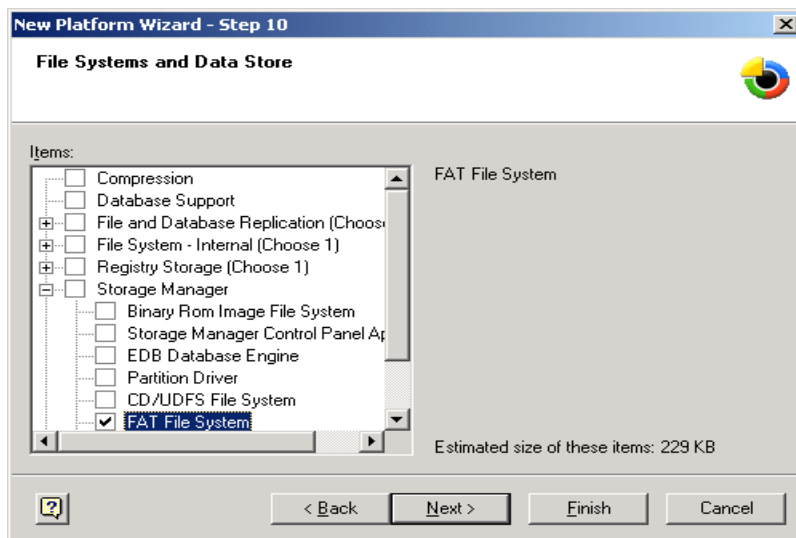
On step 11 you can select support for fonts. Click *Next*.

On step 12 you can select support the different languages. Click *Next*.

On step 13 you can select support for Internet Client Services. Click *Next*.

On step 14 you can select Graphics and Multimedia Technologies. Click *Next*.

On step 15 you can select Security. Click *Next*.

On step 16 select the from *Shell* the *Command shell, Standard Shell* and A*YGShell API Set*. From the *User Interface* the *Network User Interface, Quarter VGA Resources* and from *Software Input Panel* the *Software-Base Input Panel* and the *SIP for Small Screen.*
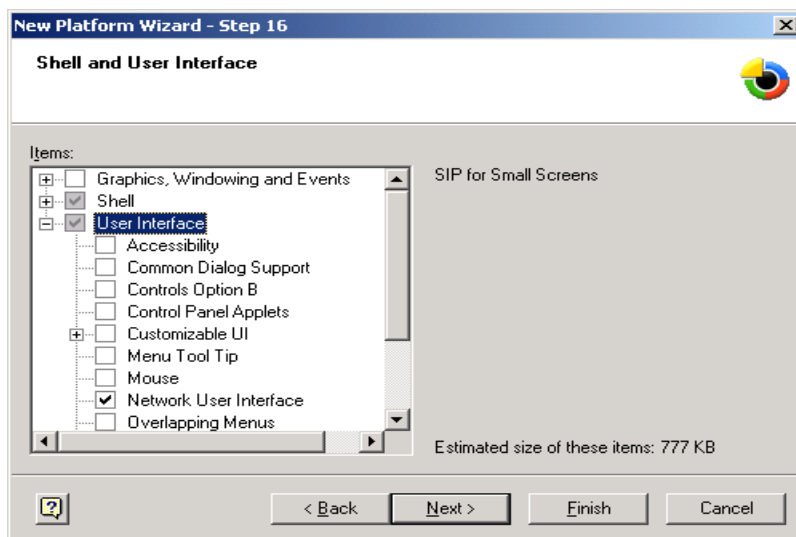
On step 17 you can select Error Reporting. Click *Next*.

On step 18 you can select Voice over IP Phone Services. Click *Next*.

After this step, some help information may appear to warn you about security issues on your platform. Read this information carefully.
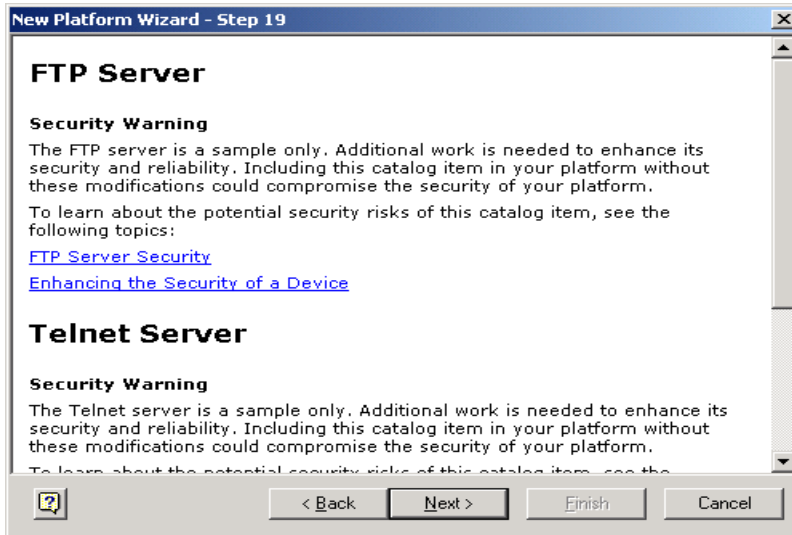
If you enable the checkbox "Notification acknowledged" you won't be warned again about this security issue. If you leave it as it is you will be reminded about this security warning. You can leave it unchecked.

After completion of the final step, click on *Finish*.
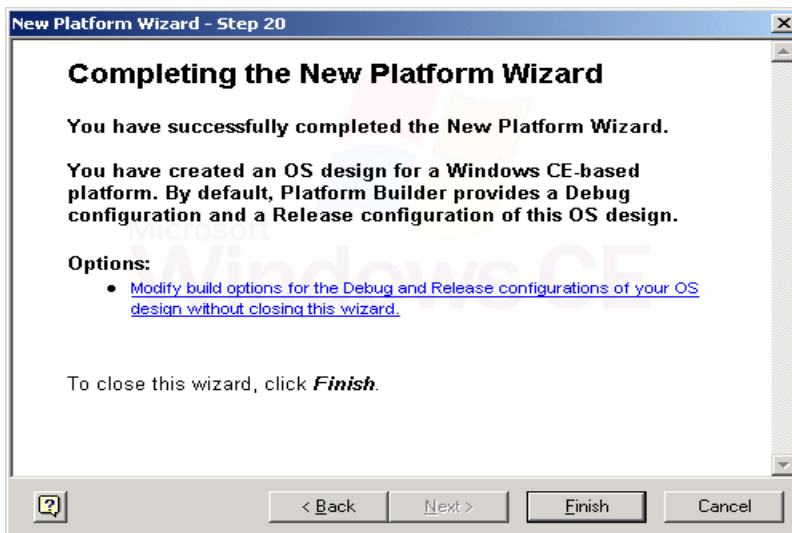
You will see the features of your platform on the Platform window (left side of Platform Builder). Also, in the catalog window (right side) you can see the components of the A9M9xx0 module in the *Third Party* folder under *BSP*.
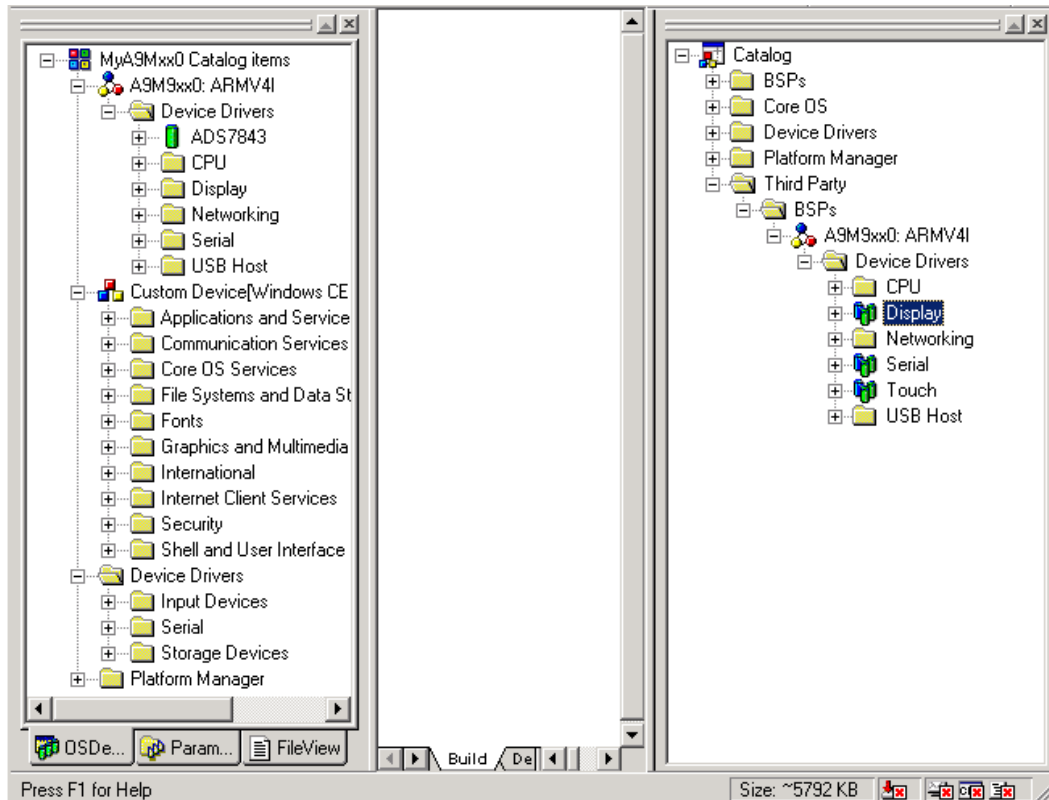
To build a platform for a NetSilicon CPU, it is necessary to select the CPU you want to build the kernel for. If no CPU is selected the Platform Builder will ask for the a CPU type.

If in some cases not all drivers of the platform are shown in the OSDesignView add them manually from the catalog. Than select from the catalog under *Platform Manager* the *Platform Manager* component. This is necessary if you want to use application debugging and the remote tools for a release kernel.

6.1   Platform settings

Before compiling the kernel, it is important to understand the different settings that we can change in the platform. Go to *Platform* menu and click on *Settings…*

6.1.0   Build options

These are the build options of your platform. For a *Release³* platform, only a few things are enabled by default (*Target Control Support, Eboot space in memory* and *KITL*). For a *Debug* version, also the *Full Kernel mode* and the *Kernel Debugger* are enabled.

6.1.1   Language settings

Under the *Locale* tab, you can select the locales that your platform will support and the default language of your image.

The locales include information about currency formats, date and time formats, etc. specific to each country.

---

³ The *Release* version will create a smaller kernel but won't include information for debugging.

The default language specifies the language of the Windows CE user interface (buttons, menus, windows, etc.). For our example select *English US* because otherwise you will not see the links to some programs in your start menu.

6.1.2 Environment Variables

The platform has several environment variables. Environment variables allow you to configure your platform and drivers. Some settings are made in the A9Mxx0.bat file in the platform root directory. Others are set by each component and can be seen at:

C:\Wince500\public\common\oak\catalog\cec\A9M9xx0.cec

They will exclude the PCI bus and PCMCIA components, since they are not part of the A9M9360. The A9M9750 contains PCI support , when BSP_NOPCIBUS=.

In addition, each driver contains its own environment variables.

The LCD driver has one variables:

BSP_DISPLAY_NS_LCD=1

These variables allows the registry information for the LCD (platform.reg) to be merged in the registry. If you set BSP_DISPLAY_NS_LCD=, the registry keys of the display won't be included into the registry.

The Network driver LANNS9750 has some variables:

BSP_LANNS9XX0=1
BSP_IP2REG=1

These variables will help building the correct registry information for the driver. By setting the variable, BSP_LANNS9XX0= , the Ethernet debugger will be used. The IP2REG is a small program that reads some information passed by U-boot to reconfigure the Ethernet settings of the NDIS driver. If you want to use the registry settings of the Ethernet driver set in platform.reg, than switch of this variable.

The serial driver (COM1) has one variable:

BSP_NOSERIAL=

Each serial driver instance has his own variable to include the corresponding registry settings into the image. By setting the variable, BSP_NOSERIAL=1, the serial driver is excluded.

The USB Host driver has two variables:

BSP_NOUSB=
BSP_USB_NSOHCI=1

To exclude only the USB Host driver, but not the USB support, set BSP_USB_NSOHCI=. When you want to exclude the USB support form your kernel, set BSP_NOUSB=1. This will exclude all USB class drivers, like keyboard, mouse etc.

The Touch driver has one variable:

BSP_NOTOUCH=

To exclude the touch driver from the Kernel, set BSP_NOTOUCH=1. This will remove all registry entries and the BIB file entries for the driver.

6.2   Include virtual Keyboard

If it is necessary to have a software keyboard, Windows CE already has one. The virtual keyboard gives users the possibility, that do not have an real keyboard interface, to enter characters or fill in forms, etc.

To include the keyboard into your platform open the catalog on *Core OS > Shell and User Interface > User Interface > Software Input Panel*. Just right click on *Software-based Input Panel* and then click *Add to Platform.*
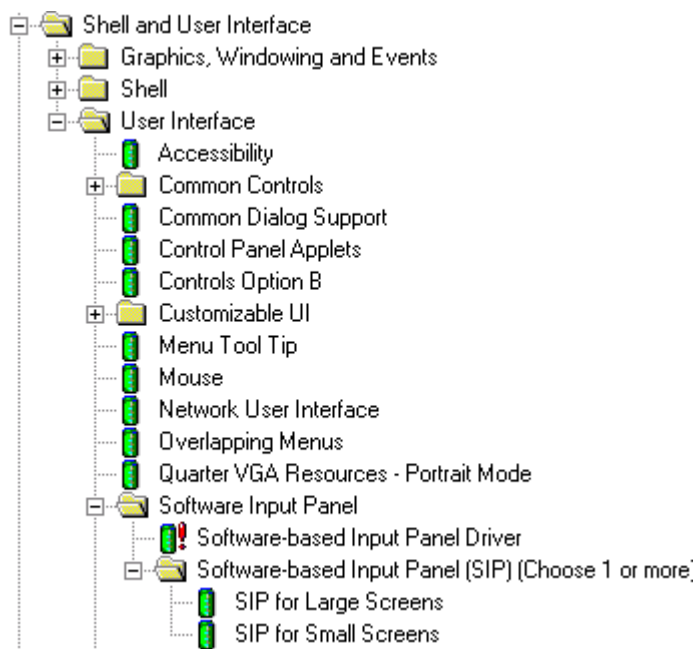
On the catalog in the *Software-based Input Panel (SIP)* select one of the two software keyboards, either small or large and add it to your platform. When you select the *Software-based Input Panel* one of the virtual keyboard is added automatically. Then simply changed it to the one you would like to be in your project.

The Input Method Selection Sample Application (Sipselect) code has been made into an integral part of the shell. This means that the Sipselect component no longer appears as an separate application. The Sipselect code is still available in the Samples folder.

%_WINCEROOT%\public\common\sdk\samples\sipselect

The SIP icon itself is now moved and is in the taskbar, which is reserved for the shell.

6.3   Including USB HID keyboard support

When USB Host support is selected for mouse and keyboard, a keyboard driver needs to be included into the project. This is necessary that Windows CE can generate the virtual key messages. To achieve this we will use a standard Windows CE keyboard driver.

From the catalog view move to *Device Driver\input Devices\Keyboard/Mouse* and select the *NOP (Stub) Keyboard* layout for the project.
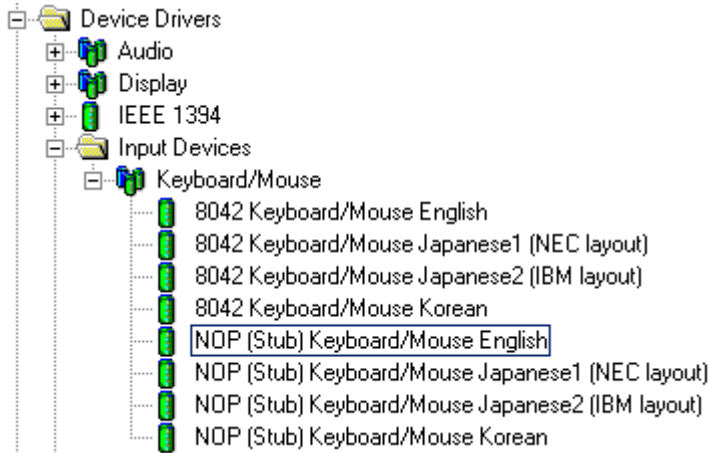
The Stub keyboard driver has the basic functions to receive the scan code of a key and map it a character and virtual key code. By selecting *Add-to-Platform* the component will be included into the project.

6.4   Building the Platform

You can now proceed to compile the *Release* or *Debug* version of your platform. For a first example, select a *Release* version.
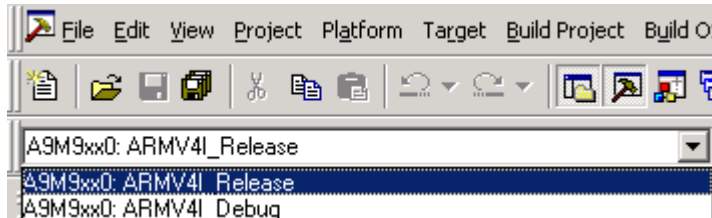
Go to the *Build OS* menu and select *Sysgen*. When ever you add or remove components you should do this step or at the first time you build your project. You can make these modifications and build the kernel without do so, but than your kernel will not include them.
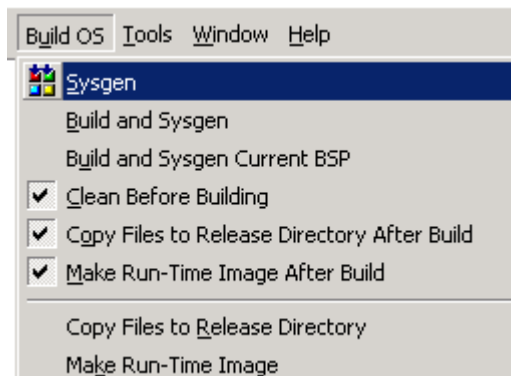
Set in the *Build OS* menu the *Clean Before Building*, this clean all object files and always rebuilt all your drivers.

The build process takes several minutes, depending on the number of components you have included and the speed of your working PC.

```
Start 00833000 Len 00004f7c
Start 00838000 Len 00000010
Start 00839000 Len 00001fe8
Start 0083b000 Len 00019048
Start 00855000 Len 0000a010
Start 00860000 Len 00022ee0
Start 00882ee0 Len 00038380
Start 008bb260 Len 0003d520
Start 008f8780 Len 0002302c
Start 0091b7ac Len 00000054
Start 0091b800 Len 00001148
Creating rom file D:\WINCE500\PBWorkspaces\MyA9Mxx0\RelDir\A9M9xx0_ARMV4I_Release\NK.nb0
Done!
makeimg: Check for D:\WINCE500\PBWorkspaces\MyA9Mxx0\RelDir\A9M9xx0_ARMV4I_Release\PostRomImage.b
makeimg: Check for D:\WINCE500\PBWorkspaces\MyA9Mxx0\RelDir\A9M9xx0_ARMV4I_Release\PostMakeImg.ba
makeimg: Change directory to D:\WINCE500.
makeimg: run command: cmd /C D:\WINCE500\public\common\oak\misc\pbpostmakeimg

MyA9Mxx0 - 0 error(s), 0 warning(s)
```
```
◄ ►\ Build ⟨ Debug ⟩ Log ⟨ Find in Files 1 ⟩ Find in ◄ ►
                                        Ln 1217, Col 1   REC COL OVR READ   Size: ~5792 KB
```

After successful compilation[4], the Windows CE image (a file called NK.BIN/NK.NB0) can be found in:

\%_WINCEROOT%\PBWorkSpaces\YourPlatformName\RelDir

A9M9xx0ARM4VIRelease
A9M9xx0ARM4VIDebug

The size of the kernel image and the RAM size can be changed and configured in the *config.bib* file.

For changes made in the A9Mxx0 folder you only need to use the *Build and Sysgen Current BSP* menu item from the *Build OS* menu. It will recompile all sources in the platform folder and copy them to your project folder.

6.5   Modify CONFIG.BIB

If your kernel image will be bigger than the configuration made in config.bib you have to adjust the size. The default is made for a 16MB RAM module. Both sizes, the Kernel and RAM section, together should never exceed the memory size of the module.

The following example will make it a little bit clearer. We assume 32MB of total module RAM and a Kernel of 14MB. The config.bib settings should look like this.

#define NKSTART 88200000 ; This is always the start point

#define NKLEN 00E00000 ; 14MB

#define RAMSTART 89000000 ; start of RAM section for object/application

#define RAMLEN 01000000 ; 16MB

You can also specify a smaller amount of RAM and a function inside the kernel will look for the rest of RAM available on the target.

---

[4] You may see some warnings that are due to Windows CE 5.00 fixing of some DLLs that use the style from older versions.

7 Download of the image to the target

7.1 U-Boot

A universal boot loader was adapted by FS Forth-Systeme GmbH for the ModARM9 family. The boot loader is capable to download Windows CE via Ethernet to the target or start a Windows CE out of the NAND Flash.

When U-Boot is launched, it configures the serial console and loads the EEPROM settings as environment variables. If no EEPROM is defined as yet on the A9M9750, standard values are compiled in are used. If the environment variable *bootcmd* is defined, it executes these commands (autoboot feature) after *bootdelay* seconds, otherwise the shell is displayed. The automatically booting can be interrupted by pressing a key to display the shell, too.

U-Boot can also retrieve these images from Ethernet (command "tftp"). The MMU is not set up by U-Boot, therefore all memory addresses used by U-Boot are physical addresses.

Ethernet is only configured and enabled while doing network accesses ("tftp"), a ping from the host to the target running U-Boot will fail.

A boot loader version installed in the shipped target can be found on the shipped CD in the *Bootloader* folder.

To copy the boot loader into Flash use the FS Forth-Systeme JTAG Booster. For more information please refer to the JTAG Booster documentation and the chapter 8.7 in this documentation. The target will come with a preinstalled U-Boot.

The boot loader will be located at the first sector of the onboard Flash. After termination of the installation with a reset the boot loader should send some information over the serial port.

Use a terminal program, like HyperTerminal, with the configuration **3840/8/N/1**. The U-Boot has a console interface that will allow to make the necessary configuration.

7.2 Abilities

- Download Kernel images using on board NS9750 Ethernet controller.
- Download Kernel images using on board NS9360 Ethernet controller.
- Enable debugging over Ethernet.
- Configuration of the target system.

7.3 U-Boot Options

The boot loader has various configuration commands. To view the commands type *help* on your terminal program. For more detailed information of each command type *help <command>*.

Download commands to download a release version:

```
setenv ipaddr 192.168.50.55 -> IP address for download
setenv serverip 192.168.50.190 -> IP address of the host PC
mw 8800 0 16                    -> clean boot args area
mw 8858 0x3B32A8C0              -> IP address Windows CE target
mw 885C 0x00FFFFFF              -> Network mask of target
mw 8860 0x0532A8C0             -> Gateway for target
mw 8864 0x02C8A3D4             -> DNS server for target
mw 8868 0x00000000             -> DHCP disabled
tftp 0x200000 nk.nb0           -> RAM storage address
go 201000                      -> Image start address
```

Download commands to download a debug version and debug over Ethernet:

```
setenv ipaddr 192.168.50.55
setenv serverip 192.168.50.190
mw 8800 0 16
mw 8800 0x544F4F42              -> debugger identifier pattern
mw 8804 0x00000058             -> boot args length
mw 8808 0x11070007             -> adapter type Eshell flags
mw 880C 0xA0600000             -> I/O address of Ethernet debugger
mw 8810 0x00010000             -> enable debug zones
mw 8814 0x3732A8C0             -> IP address of Ethernet debugger
mw 8858 0x3B32A8C0             -> IP address of Vmini
mw 885C 0x00FFFFFF
mw 8860 0x0532A8C0
mw 8864 0x02C8A3D4
mw 8868 0x00000000
tftp 0x200000 nk.nb0
go 201000
```

This examples can be found in the *eboot\bin* directory of the platform directory. You can modify the files for your needs and call them by the terminal program to execute the commands. A readme.txt will be available with detailed information of the meaning of that commands.

7.4 TFTP

To download the kernel image into the target RAM a TFTP server is needed on the PC Host side. In the TFTP folder you will find a TFTP server evaluation software from WaluSoft (www.walusoft.co.uk). This unregistered evaluation version has the following restrictions.

- Max 2 simultanious clients
- Max 10 transfers before quiting
- No firewall support
- No create path
- Fixed to port 69
- Server stops after one hour and need to be manually restarted

Start the *TFTPSrvC2001.msi* file and follow the installation instruction. After installation has been terminated successfully go to *Start > Settings > Control Panel*. A *TFTP Server* should be available in the Control Panal.

TFTP Service   TFTP Service Parameters Configuration

Open the TFTP Server and click on the *Outbound Path* tab. Set the path where your nk.nb0 file will be located for download. Typically this path should be the current working directory of your project.

e.g: *c:\Wince420PBWorkspaces\\ProjectName\RelDir\A9M9xx0ARMV4Irelease*

You can also setup a general folder and copy the images to this location.

If there is already installed an TFTP server on the PC Host, it is not necessary to follow this instructions. Any other TFTP server can be used under Windows to download the kernel image.

To manually restart the TFTP server open under *Control Panel* the *Administration Tools*. A Window will be opened. Than open *Services*. A lst of all services will be shown. In the services windows look for the TFTP server service and restart it.

7.5   Terminal configuration

The boot loader starts communication over the external UART on the development board after power-on or reset. Every commercial terminal program can be used to communicate with the target. This example will use the HyperTerminal program to show the basic configuration to setup the serial communication to the target.

The boot loader works with the following configuration **38400/8/N/1**. This configuration can be setup by the properties of the HyperTerminal session.

In some cases it has been seen that launching a script on the HyperTerminal with the *Send text file* entry in the *Send* menu hasn't been terminated successfully. To overcome this problem select the properties and than the *Configuration* tab. Open the *ASCII Configuration* and set the *Line Delay* to 30ms and than apply the changes.

7.6   Download Kernel through Ethernet

Once you have installed the boot loader, the TFTP server and modified, if necessary the download commands, everything is ready to download the kernel image.

Open a terminal program and make the configuration explained in the previous chapter.

Boot the target and you should see the following output on a terminal program.

```
U-Boot 1.1.2 (Jan 12 2005 - 15:06:07)

U-Boot code: 00F80000 -> 00F9D6FC  BSS: -> 00FA1B6C
IRQ Stack: 00f5fd7c
FIQ Stack: 00f5ed7c
RAM Configuration:
Bank #0: 00000000 16 MB
Flash:  0 kB
NAND:32 MB
In:    serial
Out:   serial
Err:   serial
A9M9750 #
```

Check that the TFTP server is setup correctly to download the new image to the target.

Start the command script file from the terminal program. U-Boot will execute the commands and download the kernel image into RAM. After downloading type on the command prompt *go 201000* and than hit the ENTER key. U-Boot will jump to the direction in RAM and start execution. This will start running the Windows CE Kernel.

## 7.7   Debug over Ethernet

First download the Kernel as mentioned in the previous chapter by using the *launch_9xx0_dbg.txt* script file. When the download of a debug kernel is made the first time, the following steps have to be made to debug over Ethernet or when the target module has been changed. Verify that BSP_NOSHAREETH= in A9Mxx0.bat. If the variable is set the NDIS driver will be included into the image and no debugging is possible.

After the kernel has been downloaded and the *go* command has been executed, some information are send over the serial port.

```
NS9XX0 Ethernet Module Initialitation Success
NS9XX0 Ethernet Module Enabled
NS9XX0 Ethernet Address: 0:4:F3:0:23:AC
-NS9XX0Init
Debug Ethernet card initialized, MAC Address:00:04:F3:00:23:AC
VBridge:: built on [Jun 24 2004] time [18:28:30]
VBridgeInit()...TX = [16384] bytes -- Rx = [16384] bytes
Tx buffer [0xA8811300] to [0xA8815300].
Rx buffer [0xA8815320] to [0xA8819320].
VBridge:: NK add MAC: [0-4-F3-0-23-AC]
INFO: Ethdbg IRQ = 0x11  SYSINTR = 0x18
-OEMEthInit
Using device name: A9M9XX0_9132
```

```
Device A9M9XX0_9132, IP 192.168.50.57, Port 981
Calling EdbgInitDHCP
-InitEther
-OEMKitlInit
```

Copy the name under *Using device name*. Open the *Connectivity Options* menu from Platform Builder. If you do not specified a target device yet, first open *Add* Device. Enter a new name for target device. The *Associated OS Design/*SDK you can set to none. Than select Add. This will add you new name to the Target Device name list. Select your new added device from the Target Device list.
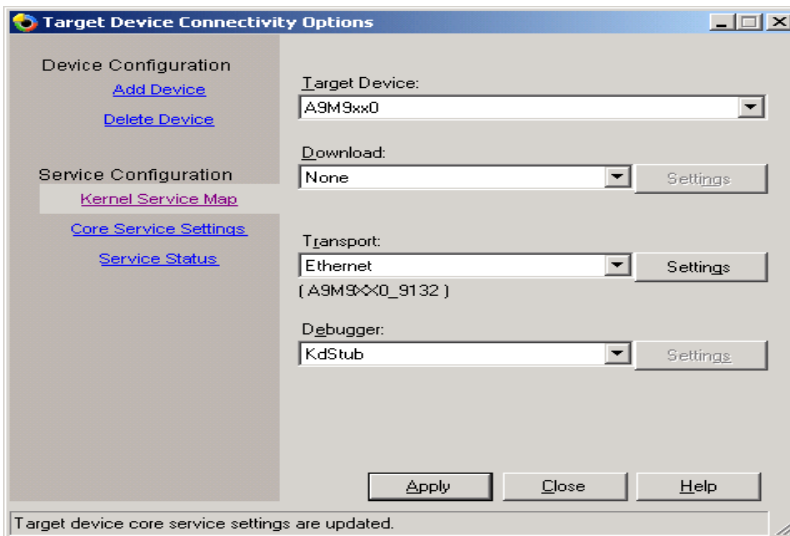
Than select *Kernel Service Map*. From their select for *Download* none and for *Transport* the Ethernet entry. Open the *Settings* of the *Transport* and paste the name copied from the terminal into the dialog.
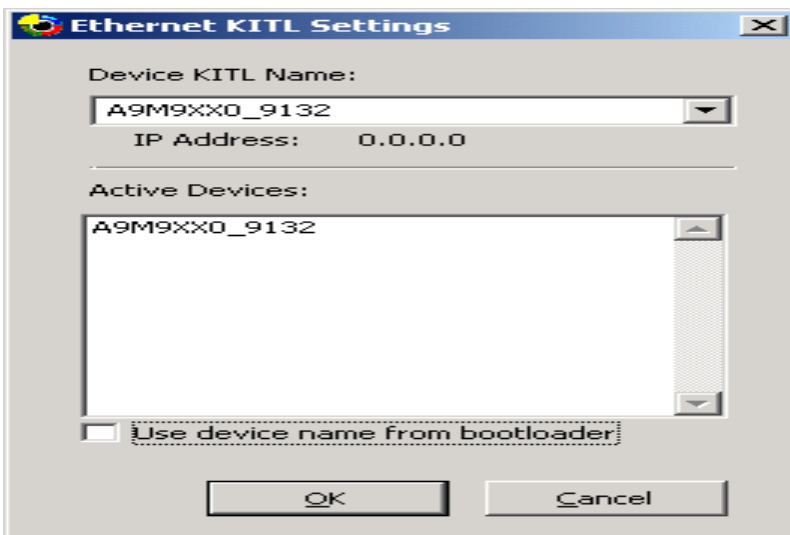
To add the new Device KITL name remove the check box *Use device name from bootloader* and add the name. Than click on OK. The name of the current used device will be shown under the *Transport* entry.

Because U-Boot is not generating BOOTME messages, they are needed that Platform Builder detect a new device. Thus we have to made it manually. This is only an example, the value may be different on your system. The number is

create by the MAC address of the target and therefore, will change with every target.

Than apply the new settings, close the dialog and select the new device from the Platform Builder *Active target* device list. Now you can go to the *Target* menu and click on *Attach Device*.

After starting the debug process, by selecting *Attach Device*, the services CESH (Console Debug Shell tool) and CETerm (Target Messages) are initiated on the target. Windows CE starts without the need of additional tools.

Once the device target name is listed you can make the download with U-Boot, start the kernel image in RAM and select *Attach Device* from the Platform Builder tool bar. If the name changes, because of a different MAC address or you using another module, you may want to repeat the steps mentioned above or simply change the *Device KITL name*.

7.8 <u>Copy Kernel image to Flash</u>

Before a Kernel can be transferred to Flash memory the image need to be download into target RAM. Follow the steps explained in chapter 7.6 and download the Kernel.

When download has been finished check the NAND Flash for bad blocks. If no bad blocks have been found the Kernel should be stored at address 0x40000 or higher. The addresses before are used by U-Boot. To test for bad blocks type in the following command

```
nand bad
```

If there is a bad block found try to store the image on an address higher than the bad block or if the bad block address is higher than the start address plus the image size, you can keep storing the image at 0x40000.

After that finally the image can be stored into Flash. The following script shows the commands necessary by U-boot to transfer the image into the onboard Flash.

```
nand erase 40000 <image_size>
nand write 200000 40000 <image_size>
```

The image size of the downloaded Kernel can be found at the end of the download. To test that the image is stored correctly in Flash you can make a reset of the target device. Than write the following commands on U-Boot and you stored image should come up.

```
nand read 200000 40000 <image_size>
go 201000
```

The next step is to load the image automatically at start up of the target.

7.8.1 Configure start of Kernel image in Flash

Now that image is inside the NAND Flash we need to configure that the image is started automatically at power-on or reset. There are several variables that need to be configured to achieve this. These variables are stored than in Flash or at the EEPROM. Thus, the target can use them even after power-off.

The following commands are necessary to make all this.

```
setenv ipaddr 192.168.50.57
setenv serverip 192.168.50.880
setenv bootargs
setenv setup_win_1  mw  8800  0  16\;mw  8858  0x3B32A8C0\;mw  885C
0x00FFFFFF
setenv setup_win_2  mw  8860  0x0532A8C0\;mw  8864  0x02C8A3D4\;mw
8868 0x00000000
setenv bootcmd run setup_win_1\;run setup_win_2\;nand read 200000
40000 <image_size>\; go 201000
saveenv        -> stores variables to Flash or EEPROM
reset
```

Now each time the target is started this Kernel will be loaded automatically, when no keyboard key is hit during the boot delay time. With a user action durin this wait time U-Boot will come up with the command prompt and waiting for new user commands.

7.9 JTAG Booster

This tool from FS Forth-Systeme GmbH gives the user the possibility to copy the boot loader much faster to the onboard Flash. Before storing a new boot loader to the ModARM9 module, the JTAG Booster hardware have to be connected to the target board. Connect the JTAG Booster to the parallel port of your host PC and the other end to JTAG connector on the target board. Than open command shell on the host PC. Enter the path where the JTAG Booster tools have been stored and type in one of the following commands.

jtag9750 /pnand a9m9750.bin /lpt-base=378

jtag9360 /pnand a9m9360.bin /lpt-base=378

Set the switch S2 to 1-ON and the rest OFF. Than power up the target and start the tool on your PC. Start on the host PC the *JTAG9750*.exe or *JTAG9360.exe* tool to program the boot loader. For more details please refer to the JTAG Booster documentation. When the boot loader is successfully downloaded remove the power from the target and switch back S2 to 1-OFF. Power up the target and over the serial port, as described in the previous chapters, you should see the starting of the boot loader.

Note: The thick strand on the JTAG Booster connector indicates ground and is pin 2. Pin 1 is connected to the square pad on the target board and is marked with a triangle.

8   Kernel Debugging

By setting the environment variable in A9M9xx0.bat like this

BSP_NOSHAREETH=

Ethernet debugging is implemented. This must be made during compile time and cannot be changed during run time. At the same time this will enable the shared Ethernet functionality of Windows CE.

Now you can build your debug Kernel and follow the download instruction and explanations of the previous chapters.

After downloading the image with U-Boot, explained the previous chapter, simply connected to the target over the Platform Builder. Click on the *Attach Device* icon in the tool bar or over the menu item *Target > Attach Device*.

```
(CoreCon)  12:17:05 02/24/2005 Hora estándar romance:  Download service map set t
The Kernel Debugger is waiting to connect with target.
OEM Extra DRAM Detected @ base = xA8F00000, size=4095 MB
++ Enter in DS1337_Init.
-- Leave from DS1337_Init. Control = 0x18, Status = 0x3.
Booting Windows CE version 5.00 for (ARM)
&pTOC = 8820a62c, pTOC = 8849c274, pTOC->ulRamFree = 88833000, MemForPT = 0000000

Old or invalid version stamp in kernel structures - starting clean!
Configuring: Primary pages: 1483, Secondary pages: 0, Filesystem pages = 741

Booting kernel with clean memory configuration:
Memory Sections:
[0] : start: 88834000, extension: 00001000, length: 005cb000
Sp=ffffc7cc
Windows CE KernelInit
Updated eptr->e32_vsize to = 000c6000
Scheduling the first thread.
0x88dff024: KernelInit2: pCurThread=88dff024 hCurThread=08dff266 hCurProc=08dff0C
0x88dff024: Detecting VFP...
0x88dff024:   VFP Not Found!
0x88dff024: Updated eptr->e32_vsize to = 00006000
0x88dff024: Updated eptr->e32_vsize to = 00029000
0x88dff024: Updated eptr->e32_vsize to = 00009000
0x88dff024: Updated eptr->e32_vsize to = 00022000
0x88dff024: Starting kernel debugger software probe (KdStub) - KD API version 18
0x88dff024: >>> Loading module NK.EXE at address 0x88200000-0x882C6000 (RW data a
Kernel debugger connected.
The Kernel Debugger connection has been established (Target CPU is ARM).
Target name: A9M9xx0
Probe name: KdStub
```

Now we can halt the target and debug the kernel. The debugging windows and menu items in the Platform Builder IDE allow to acquire process, threads, and other target debugging information like watch variables, dump the memory etc.

The target can be halt with the menu item *Debug > Break*.

When the target is halt a source file is opened where the system was stopped.

With F10 you can now step through the code and seethe behavior of a driver or application. If you wish to run through the rest of the code press F5 (Go).
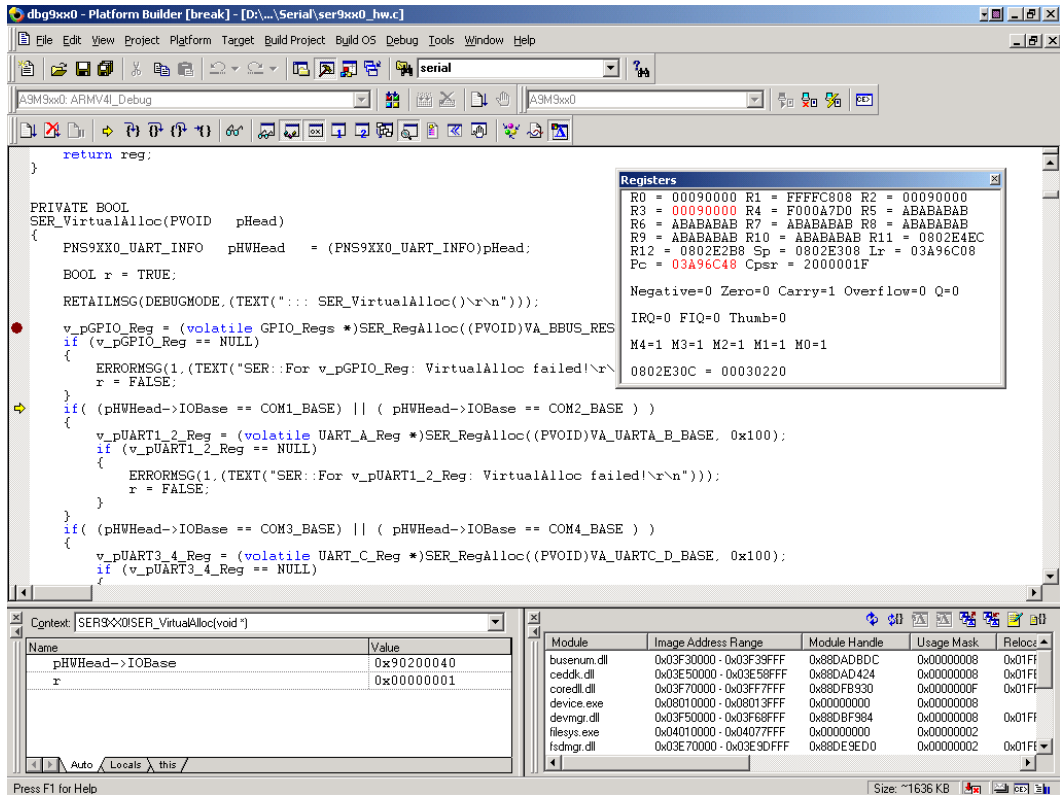
8.1 Sharing Ethernet Debugging Services

There is the possibility to share the Ethernet interface between debugging network and application network traffic. The main registry entries can be found in the *COMMON.REG*. In the *PLATFORM.REG* you can specify the IP address for the application network interface, the debugger will use the IP address set to download the Kernel image. The IP of the debug and the application network interface must be different, to avoid conflicts.

```
IF BSP_NOSHAREETH !
[HKEY_LOCAL_MACHINE\Comm\VMINI1\Parms\TcpIp]
    "EnableDHCP"=dword:0
    "DefaultGateway"="192.168.50.5"
    "UseZeroBroadcast"=dword:0
    "IpAddress"="192.168.50.56"
    "Subnetmask"="255.255.255.0"
    "DNS"="200.193.200.2"
ENDIF BSP_NOSHAREETH !
```

To include the shared Ethernet support you have to set the environment variable BSP_NOSHAREETH= and by doing so you disable at the same time the NDIS driver. By default the variable is set to disable this functionality.

8.2   Run Time Debugging

In certain circumstances you may not want to debug step by step, because of some time critical part in the system. Most of the drivers in Windows CE have a variety of Debug messages. The messages are divided into different zones. These zones can be activated or deactivated. The maximum are 16 zones. One method to set up the debug zones is to alter the registry of your host PC where you are designing the Kernel.

First open the *RegEdit.exe* on the PC the Platform Builder is running. To start the program open *Run* from the *Start* menu.
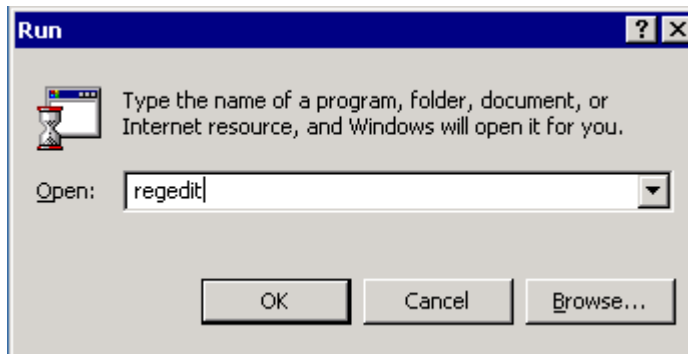
Enter *RegEdit* and select the *OK* button. The registry editor will be opened. Than go to folder *HKEY_CURRENT_USER\Pegasus\Zones*. Here you can create a new entry for the driver or component you which to debug in more deeply.
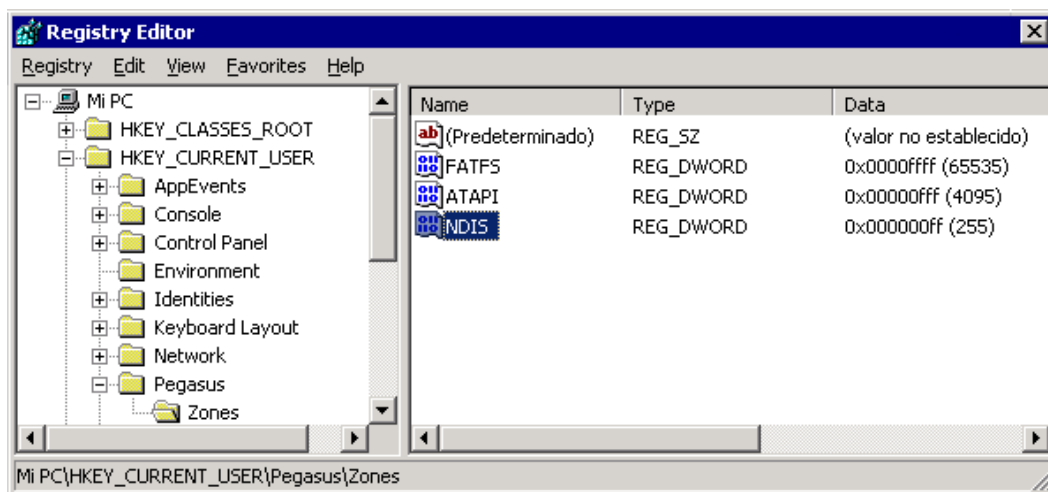

The names that you should enter here can be found normally in the *DBGPARAMS* statements of the corresponding driver or component. For more information, please refer to the Online Help. Most of the sources of Windows CE are located at
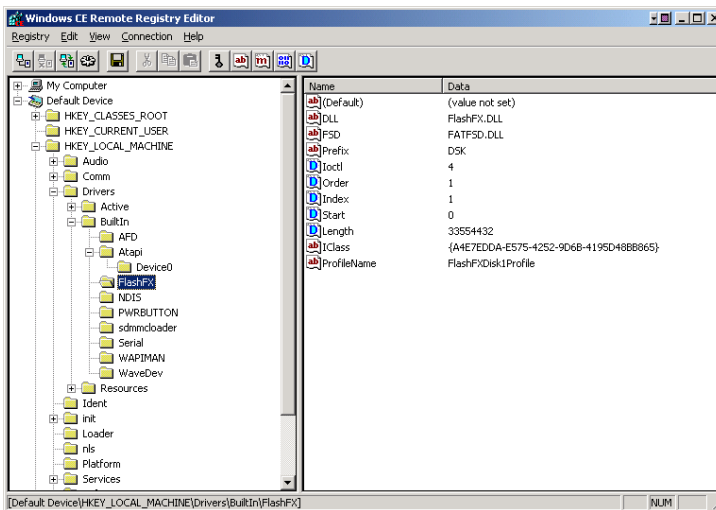
%_WINCEROOT%\Public\Common\Oak\Drivers\

Here you should find the propriety names for most of the drivers supported by Windows CE. The values that you set for each entry depends on the amount of zones that you would like to activate. E.g. the entry *FATFS,* in the picture above, activates all 16 zones. Each bit represents a debug zone that is activated with a one and deactivated with a zero.

9 Remote Tools

If you want to use the Remote Tools in a Release version of the Kernel you have to add the *Platform Manager* to your project. The component can be found in the catalog view of the Platform Builder. Simply move in the catalog view to the component and select *Add to Platform*.
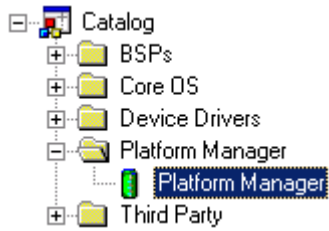
This will include the three possible medias to communicate with the target. These are ActiveSync, KITL or manual server. To use the Remote Tools with the manual server, follow the instruction in the "Application debugging" chapter. See the screen shot of the registry editor, where you can check if your device was loaded and if the values are correct.

10 Windows CE directory tree

Here is an overview of the most important directories and files in the Windows CE tree

10.1 %_WINCEROOT%\PLATFORM

In this folder are the board specific modules. For example:

CEPC   Microsoft reference PC platform
GEODE   Microsoft reference GEODE platform
A9M9xx0   NS9750/NS9360 platform

10.2 %_WINCEROOT%\PBWorkspaces

All project specific parts are stored here, for example:

My9750   One A9M9750 based project.
My9360   One A9M9360 based project.
test_CEPC   One CEPC based project

10.3 %_WINCEROOT%\PUBLIC\COMMON

In this directory all common adjustments and drivers are stored. Be careful with changes in this directory and subfolders. A change in one of the sources in the common directory will be a change for all platforms.

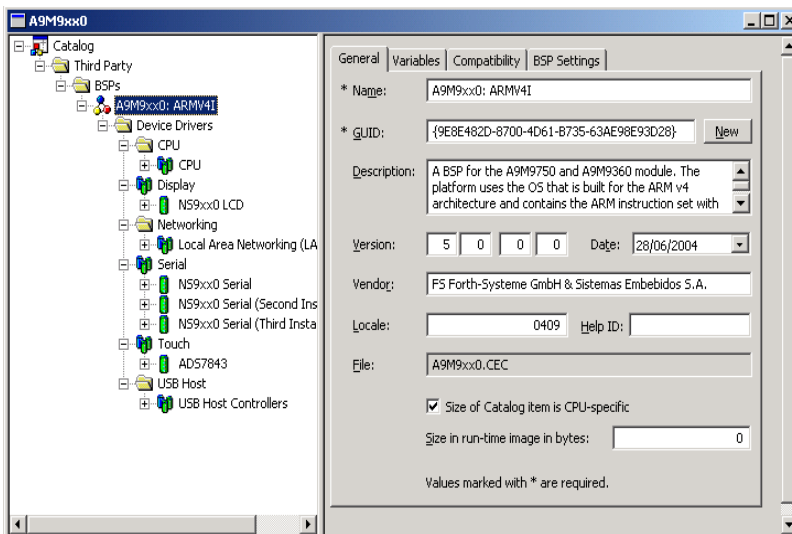### 10.4 Important file types

SOURCES   Contains source files, include/library files and
                          the names of the output files.

DIRS   Subfolders that should be compiled

*.reg   Windows CE registry files

11 CE component file

With the CEC Editor included in Windows CE 5.00, you can explore the components of the BSP just by double-clicking the file:

C:\Wince500\public\common\oak\catalog\cec\A9M9xx0.cec

This package includes the following components:

- LCD display driver
- Ethernet driver
- Serial port
- USB Host
- Touch
- IP2REG
- CPU

12 Drivers

12.1 Display

The LCD display driver has been testes with the following display.

- LQ57Q3DC2

In order to have this component included, you must select *Display support* when you create a new custom platform project or added later to your project.

12.1.1 Modify Display Driver

If you want to use another LCD on the ModARM9 module modify the display driver NS9XX0DISP() function and make the corresponding changes in platform.reg. The datasheet of the corresponding LCD should show all the necessary values that you need to set-up the display driver correctly.

12.2 Ethernet

The TCP/IP configuration expects the system running without DHCP server. The IP address is configured to 192.168.50.56 with the subnet mask 255.255.255.0.

To change the default IP, gateway, subnet mask and DNS of your target you must modify the following key values on the file:

%_WINCEROOT%\Platform\A9M9xx0\Files\Platform.reg

```
;Settings for static IP configuration, if enabled
[HKEY_LOCAL_MACHINE\Comm\LAN91C111_1\Parms\TcpIp]
   "EnableDHCP"=dword:0
   ; This should be MULTI_SZ
   "DefaultGateway"="192.168.50.5"
   ; Use zero for broadcast address? (or 255.255.255.255)
   "UseZeroBroadcast"=dword:0
   ; This should be MULTI_SZ, the IP address list
   "IpAddress"="192.168.50.56"
   ; This should be MULTI_SZ, the subnet masks for the above IP addresses
   "Subnetmask"="255.255.255.0"
   ; This should be MULTI_SZ, the Domain Name Server
   ;"DNS"="212.163.200.2"
```

If you have a DHCP service and prefer that your target gets automatically a free IP address from the DHCP server you must do the following: set in the registry configuration of the Ethernet the *EnableDHCP=dword:1*.

You can now recompile the platform and the target will take a free IP from the DHCP server automatically.

In order to have the component included, you must add to your platform the following components:

- *Wired Local Area Network (802.3, 802.5)*

To delete it from your image, right click over the component group *Network* and select *Delete*. Or you can also click on *Settings…* and then *Exclude from build and Image*.

12.3 USB Host

The USB Host can be included by selecting USB Host support form the catalog and by setting the environment variable

```
BSP_NOUSB=
BSP_USB_NSOHCI=1
BSP_NOUSBSER=1
```

Over the catalog, differend device driver classes, like mouse, keyboard or mass storage can be included into the kernel. The A9M9360 currently doesn't support that functionality of the host.

12.4 Touch

The touch driver has one environment variable.

BSP_NOTOUCH=

To exclude the touch driver registry settings set BSP_NOTOUCH=1. The driver uses the internal SPI channel A to communicate with the ADS7483 Touch controller. When the touch is used this serial port cannot be used for other purposes.

12.5 IP2REG

IP2REG is added with the Ethernet driver to the platform. This driver reads the network configuration set by U-Boot and writes it to the registry. In this way the network settings can be changed of the used target. The driver works as well for the standard NDIS driver as for the VMINI Ethernet debugger.

To exclude the driver from the kernel image set BSP_IP2REG=.

12.6 CPU

The CPU component is necessary to build the project for one of the two supported microprocessors. If no CPU is selected the Kernel will be build always for the A9M9750 module.

13 Common Hints

13.1 Automatically start of applications

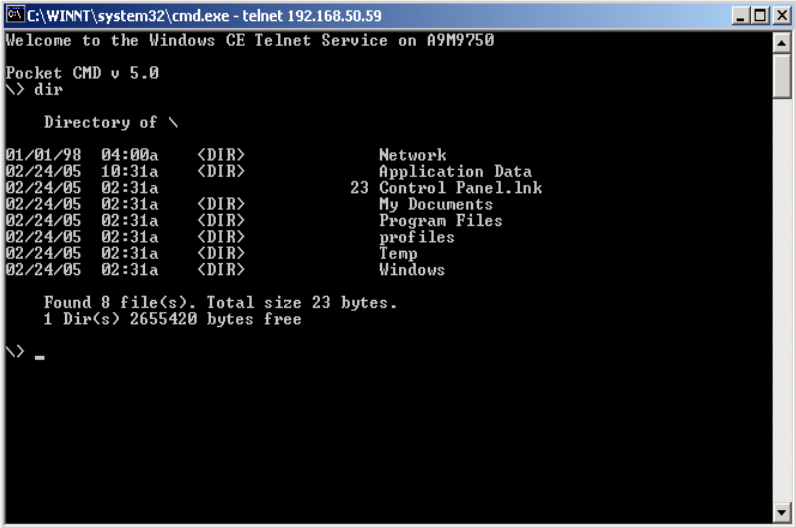Here is an extract of COMMON.REG.

```
[HKEY_LOCAL_MACHINE\init]
; @CESYSGEN IF CE_MODULES_SHELL
        "Launch10"="shell.exe"
; @CESYSGEN ENDIF
IF IMGTINY !
; @CESYSGEN IF CE_MODULES_DEVICE
        "Launch20"="device.exe"
; @CESYSGEN ENDIF
; @CESYSGEN IF CE_MODULES_GWES
```

In order to launch your own applications automatically, you need to add similar entries to the file PROJECT.REG and create variables in case you want to control the launching or not of your applications.

```
IF MY_VARIABLE1
        "Launch40"="my_application1.exe"
ENDIF
IF MY_VARIABLE2
        "Launch50"="my_application2.exe"
ENDIF
...
```

13.2 Telnet server

To connect via TELNET simply open a DOS box and do a TELNET to the IP address of your target. You won't be requested to enter a username and password, because we have disabled the User Authentication. To exit the telnet server simply enter `exit` at the prompt.

```
C:\WINNT\system32\cmd.exe - telnet 192.168.50.59
Welcome to the Windows CE Telnet Service on A9M9750

Pocket CMD v 5.0
\> dir

    Directory of \

01/01/98  04:00a   <DIR>            Network
02/24/05  10:31a   <DIR>            Application Data
02/24/05  02:31a              23 Control Panel.lnk
02/24/05  02:31a   <DIR>            My Documents
02/24/05  02:31a   <DIR>            Program Files
02/24/05  02:31a   <DIR>            profiles
02/24/05  02:31a   <DIR>            Temp
02/24/05  02:31a   <DIR>            Windows

    Found 8 file(s). Total size 23 bytes.
    1 Dir(s) 2655420 bytes free

\> _
```
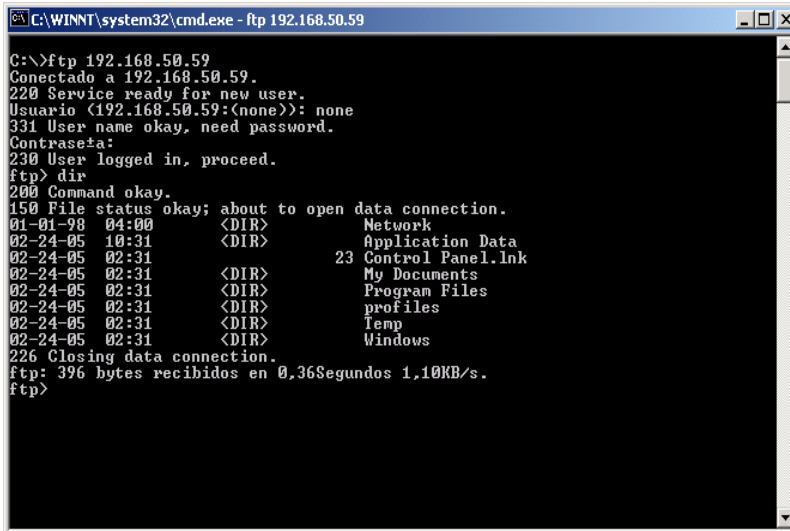
13.3 <u>FTP server</u>

To connect via FTP to the FTP server of the target open a DOS box or the Internet Explorer from Microsoft and simply do an FTP to your target IP address. The target will ask you for a username and a password, in case of the DOS box. Since we have enabled the anonymous login you can enter anything as username and anything as password (you must enter something when using the DOS box).

To finish the FTP connection simply type the word `bye` at the FTP prompt.
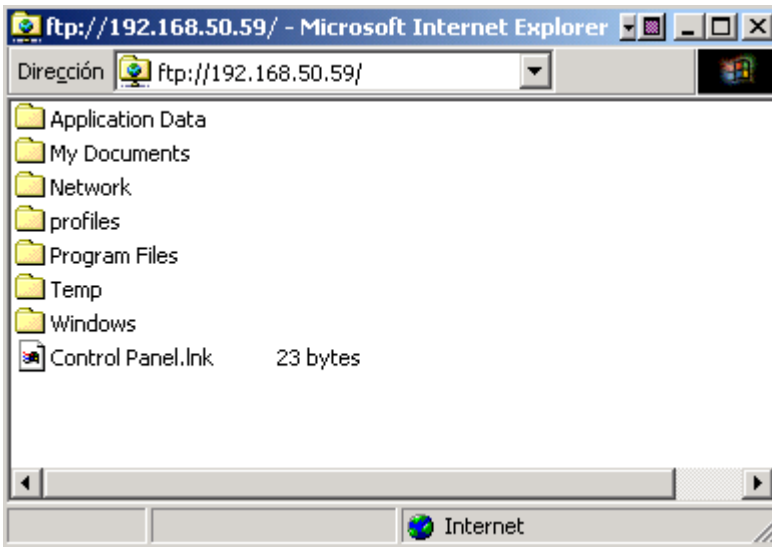
We have discovered some problems when copying files via FTP to target folders that have spaces in their name, e.g.: *My Documents*. Therefore, we recommend the Internet Explorer from Microsoft to do file transfers to the target. The Internet Explorer can handle names with spaces.

14 Creating a Software Development Kit (SDK)

The CD already contains a SDK made with the supported components of the example project. If your application will need more functionality or other components, you should make a new SDK related to all the components inside your Kernel. This section will describe the necessary steps, to create the new SDK.

Under *Platform > SDK > New* SDK the *SDK Wizard* will guide you through the process of configuring and building a software development kit (SDK) for Microsoft Visual Studio .NET, Smart Device Extensions for Visual Studio .NET, and Microsoft eMbedded Visual C++ 4.0. It generates a Microsoft Windows Installer (.msi) file that contains the necessary header files, libraries, Platform Manager components, run-time files, platform extensions, and documentation to generate a new application for you specific kernel.

With this SDK, you can create applications for your Microsoft Windows CE platform. Enter the properties for your SDK to uniquely identify it.
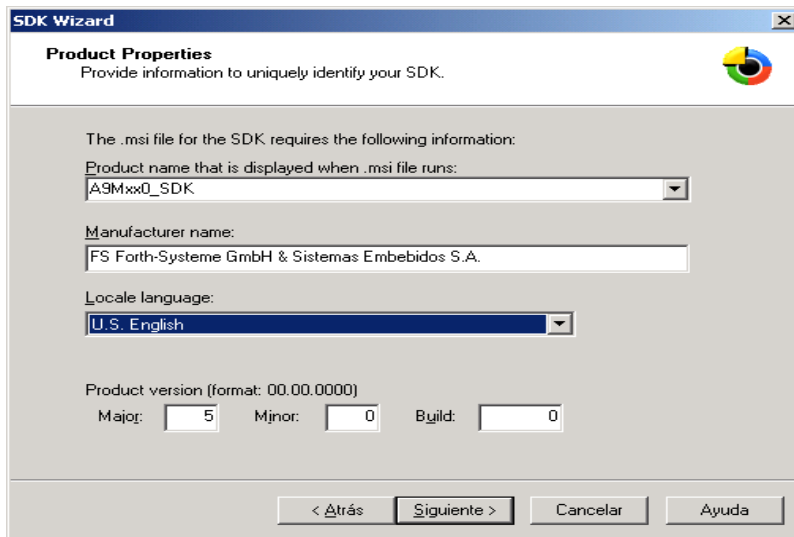
Select the development languages features you want to support.
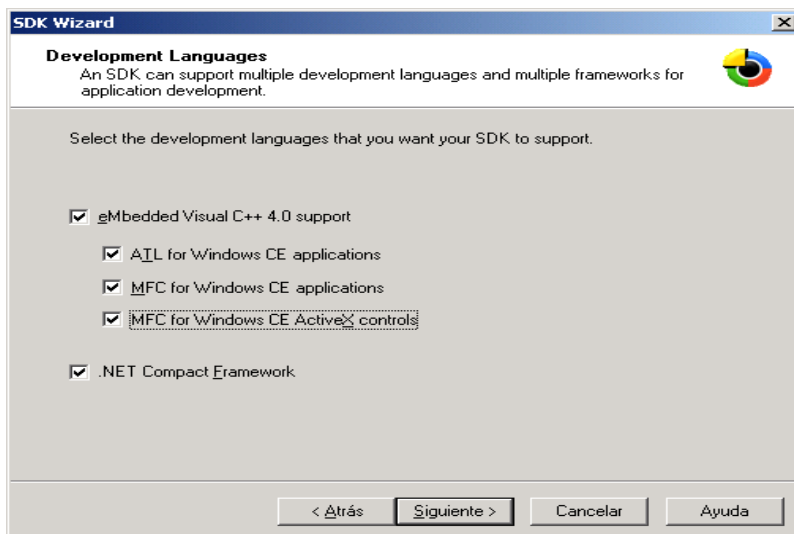
Next click on the Finish button. The configuration of the new SDK is located at

%_WINCEROO%\PBWorkspaces\YouProject\MakeSDK

Now you have made the configuration of the SDK you want to build the SDK. To build the SDK go again to *Platform > SDK* and than select *Build SDK*.

This will generate a self-installable MSI file on:

*%_WINCEROOT%\PBWorkspaces\YourProject\SDK\YourProject_SDK.MSI*

Double click this file to install the SDK into your Host PC. You can also include other or further libraries or components to your SDK.

There is an already built SDK for the previous sample platform on the provided CD, within the folder \SDK. Once you have installed it you can begin to develop your embedded visual application for your platform. If you are changing the Kernel image, by including or removing components you should make a new SDK for a secure application development.

Be sure that your embedded Visual Studio installation contains all Software Packages available by Microsoft. If it is not complete the installation of the SDK may fail. For Visual Studio .NET 2003 you need to install the *Windows CE add-in*.

15 Insert Existing Projects

The sample program to access GPIOs of the processor can be easily recompiled. To do so move to the *Project* menu item of the Platform Builder.
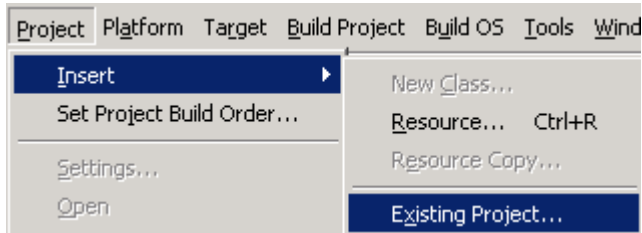
Than open *Insert* and select *Existing Project.* A dialog box will be opened and you have to select the *.pbpxml file from the directory where you want to include your sources. Confirm with Open and the project will be included in your platform. The output (*.exe) will be located in the sub directory where you copied the sources, either Release or Debug.
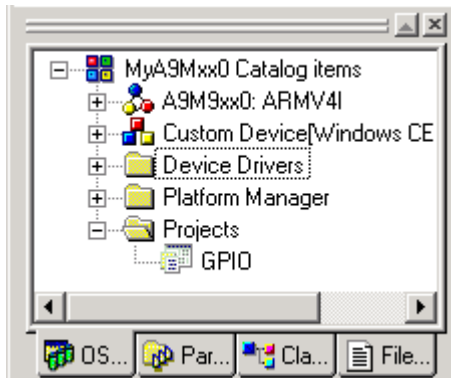
To edit the sources double click on the component of the *OSDesignView* or click on the *FileView* tab.

16 Developing Applications with Embedded Visual C++

16.1 Creating a new project

In this example, we will create a console application for the Kernel that we have made in the beginning and with the SDK for that particular Kernel. Before starting you have to make sure that the SDK is already installed on the Host PC. Than open the embedded Visual C++ and select from the *File* menu *New*.
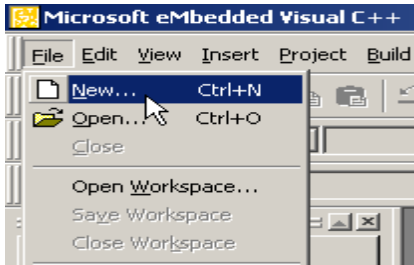

Select *WCE Application* and enter the name. Than select from CPUs the *ARMv4I* and click *OK*.
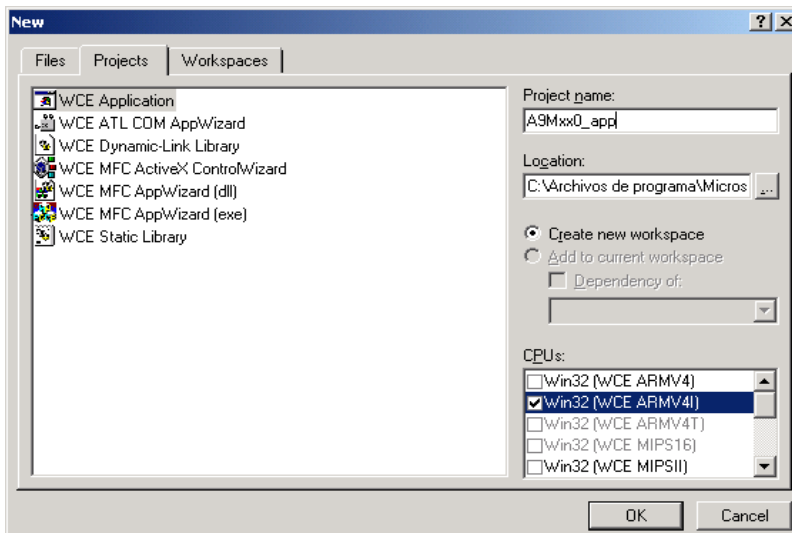
On the next screen select *Typical Hello World Application* and select *Finish*. The wizard will create all files and sources for a small graphical application.
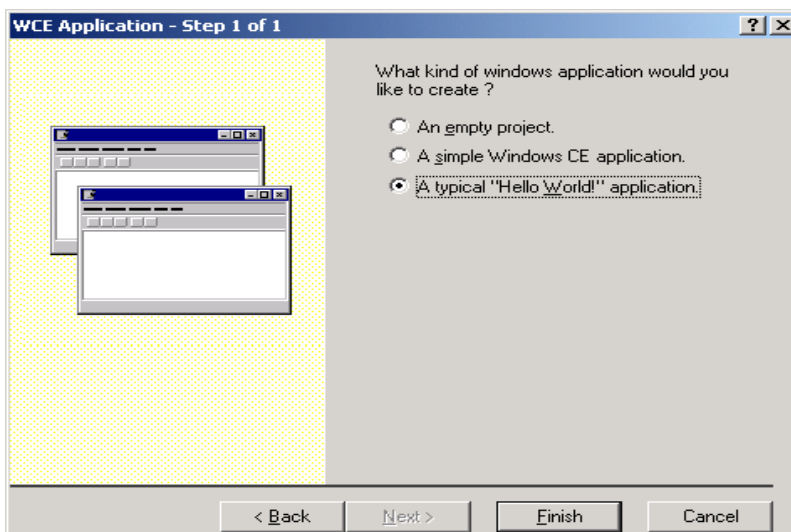
Build the application. You will see three warnings. There is no workaround for that and you should ignore them. After building the application correctly, embedded Visual C++ will try to connect to the target device. It will use the configuration of the device that is selected in the *Default Device* List.

| A9Mxx0_app ▼ | A9Mxx0 ▼ | Win32 (WCE ARMV4I) Debug ▼ | A9Mxx0 Device ▼ |
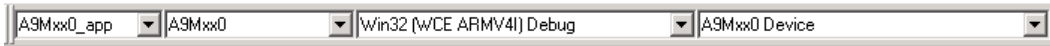
By default the transport is set to manual server. If you want to change this select under the *Tools* menu item *Configure Platform Manager*.

When you have successful downloaded the application you can execute it over the embedded Visual C++ or by launching it directly from the target.
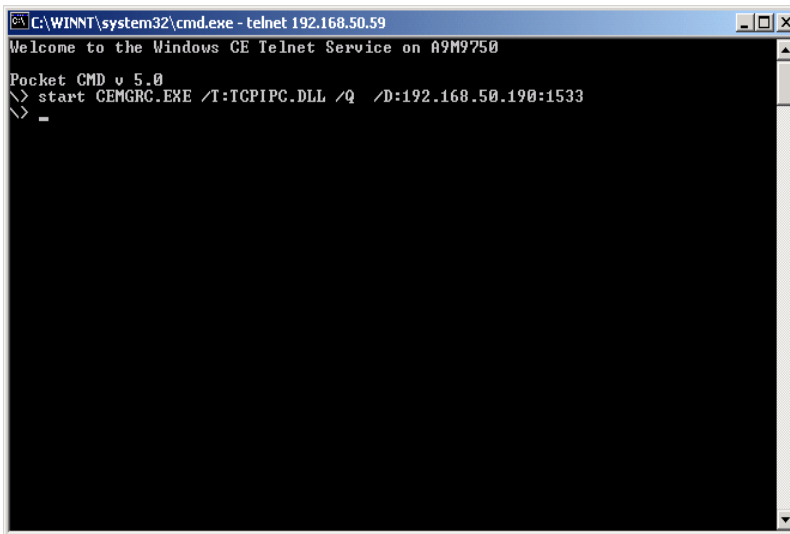

The application will be located in the *Windows* folder of the target device.

16.2 Downloading the Application to the Target

When the build have been terminated successfully embedded Visual C++ will automatically try to download the application to the target device. As we have selected our specific SDK, the embedded Visual C++ has selected the corresponding device configuration. A dialog will come up and ask to verify that certain files are on the target side.
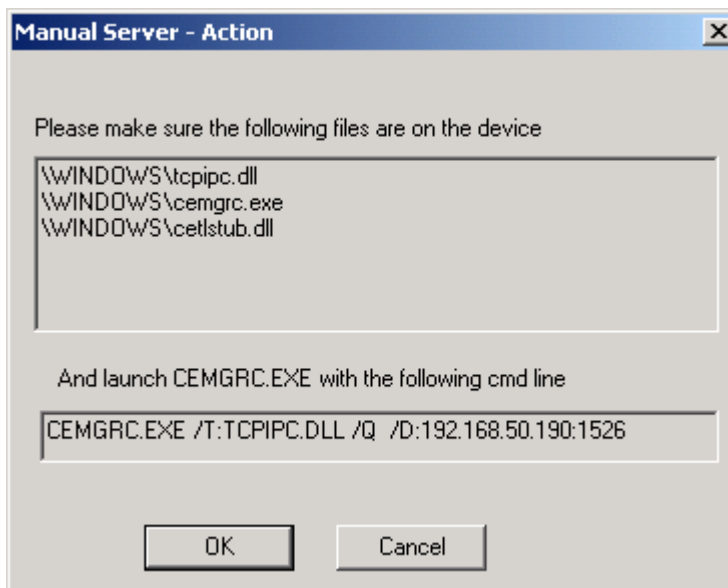
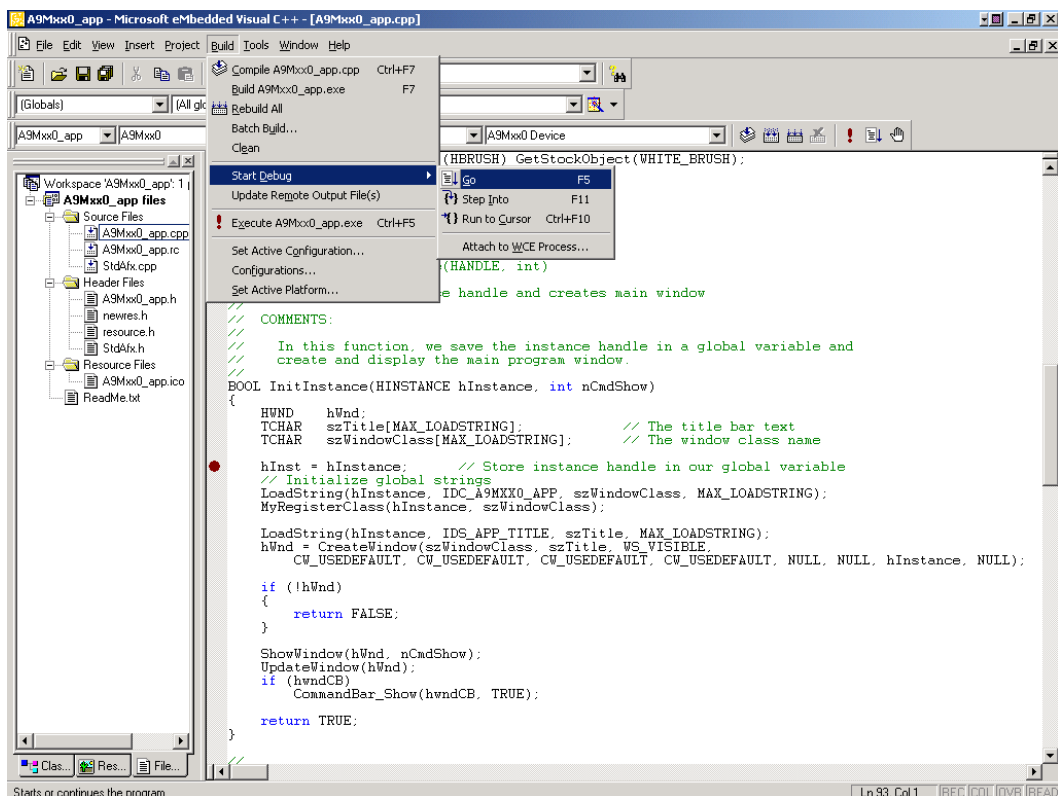Copy the *CEMGRC.EXE* line. Open a telnet to the target and paste the command line to the telnet session.

Than click on the *OK* button. The target and embedded Visual C++ will start to communicate and download the application and execute it. The download is made once to the target. Now you can control the application directly by your target or even execute it by embedded Visual C++. If you have made a change in the application you need to repeat the steps mentioned above to download the application to the target.

16.3 Debugging the Application

First you have to download the application and set a breakpoint in the source code. To download follow the steps mentioned in the previous chapter.

From the *Build* menu item select *Start Debug > Go*. Again a dialog may appear, depends if you have made already a download or if you have modified to application and want to debug it now. If the download completes embedded Visual C++ will open the debugger interface and will stop at the breakpoint that you have set.
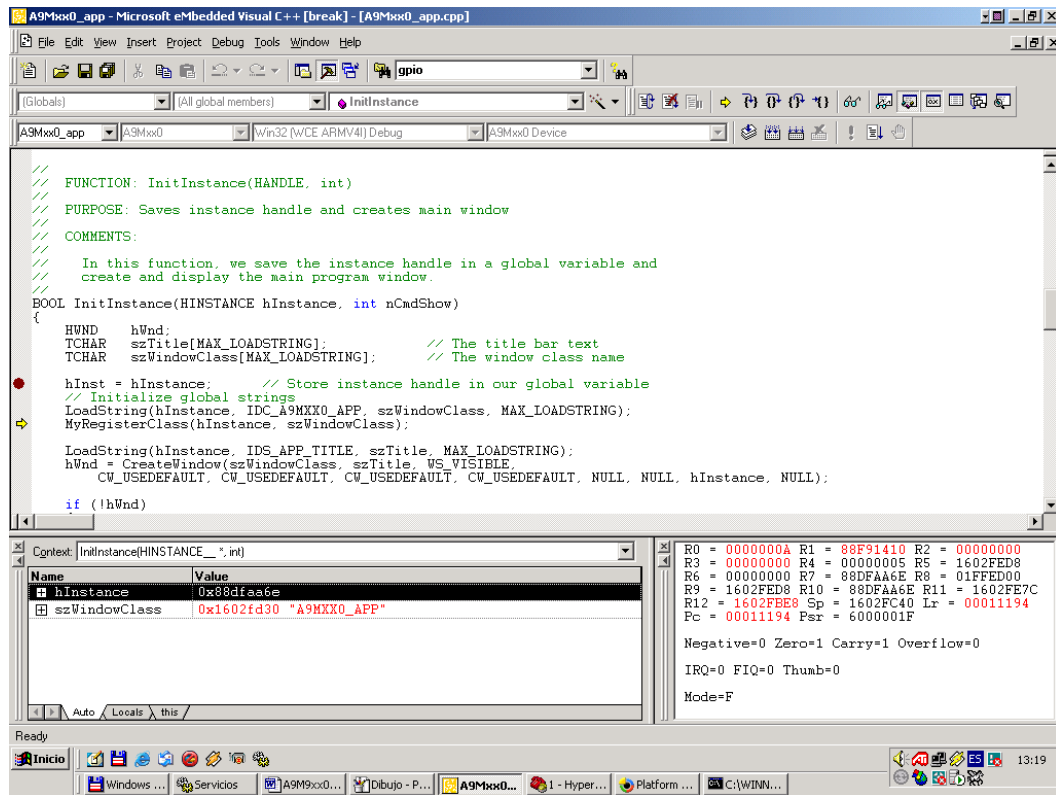
Now you can do single stepping or watch variables or the processor register. When you terminate the application the debugging interface is closed automatically.

16.4 Modifying the Platform Manager Configuration

When you want to use a different transport for your device you need to change the *Platform Manager* configuration. Under the *Tools* menu item select *Configure Platform Manager*.

This will open a dialog with all the devices that can be configured and used. Select the device where you want to change the configuration.
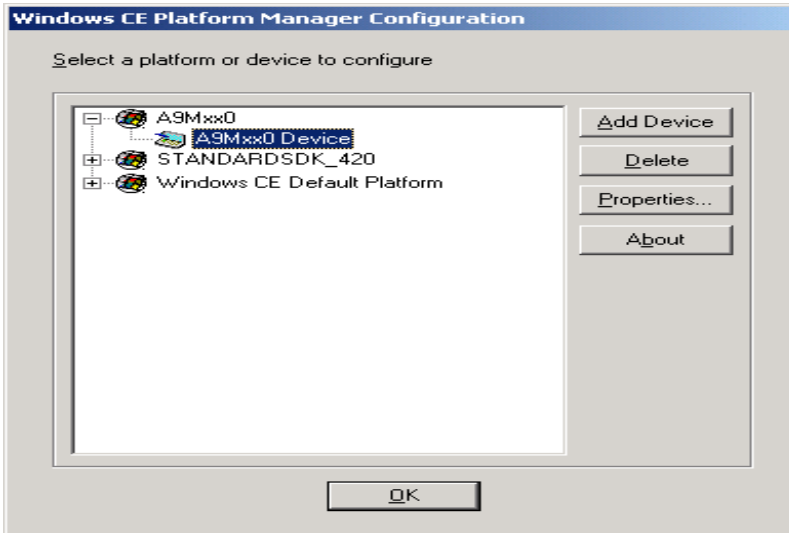
The Platform Manager supports three different transport and startup servers. Select *Properties* and than the corresponding transport and startup server you want to use for your target device.
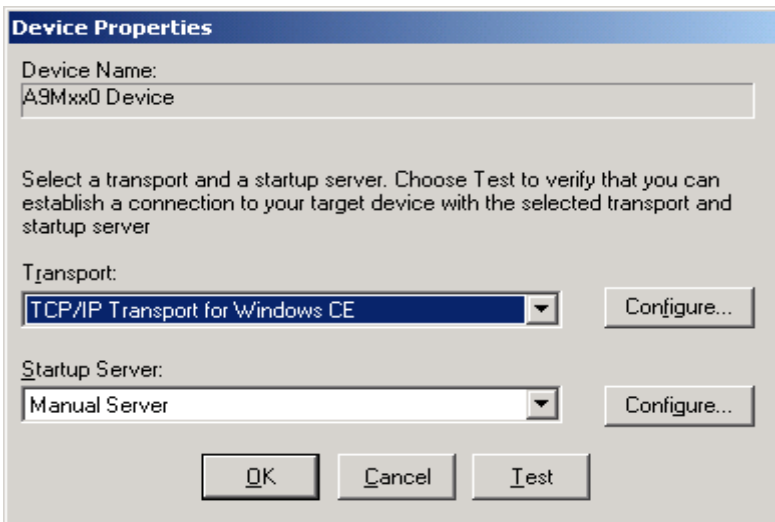
With *OK* you will confirm the changes and the next time you start the Platform Manager it will use the new settings.

17 Troubleshooting

17.1 Language settings

When using different languages it might be that during the build process an error occurs. That is, because the default language for the NS9750 package is English. This is important if you would like to add you own application to the Kernel. Some folder have different names, depending on the selected language. Thus, it could happen when you have included your application and change the language, that it seems, your application has disappeared. This occurs, because the path is now incorrect, caused only by the different language. E.g. in English you have a folder called *Program Files* , the same folder in Spanish is called *Archivos de Programa*.

To use your application in different languages use the *.dat files inside your project. Here you can specify the name of the folder depending of the used language. For more information please refer to the online Help of Platform Builder.

17.2 Quick Fix Engineering (QFE)

Microsoft releases from time to time updates of the Platform Builder and the source codes. This package already includes some QFEs. In case of some misbehaviour of the kernel, please check the Microsoft web side for new QFE releases.

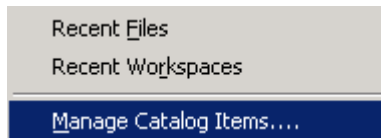http://msdn.microsoft.com/embedded/downloads/ce.net/wince/default.aspx

Each QFE contains a readme file that list all the updates that have been made with the new installation. We recommend to install always all new QFEs.
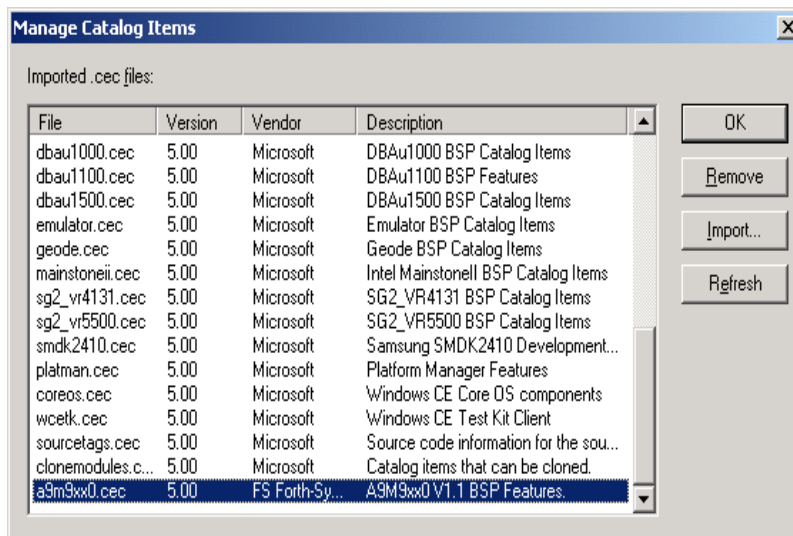
18 Removing the BSP

In order to remove completely the BSP follow these instructions:

- Delete the CE Components file
  C:\Wince500\public\common\oak\catalog\cec\A9M9xx0.cec

- Go to the *File* menu of Platform Builder and click on *Manage Catalog Items*

Recent Files
Recent Workspaces

Manage Catalog Items....

- Select the platform to remove and then click *Remove*.

Manage Catalog Items					
Imported .cec files:					
File	Version	Vendor	Description		OK
dbau1000.cec	5.00	Microsoft	DBAu1000 BSP Catalog Items		Remove
dbau1100.cec	5.00	Microsoft	DBAu1100 BSP Features		
dbau1500.cec	5.00	Microsoft	DBAu1500 BSP Catalog Items		Import...
emulator.cec	5.00	Microsoft	Emulator BSP Catalog Items		
geode.cec	5.00	Microsoft	Geode BSP Catalog Items		Refresh
mainstoneii.cec	5.00	Microsoft	Intel Mainstonell BSP Catalog Items		
sg2_vr4131.cec	5.00	Microsoft	SG2_VR4131 BSP Catalog Items		
sg2_vr5500.cec	5.00	Microsoft	SG2_VR5500 BSP Catalog Items		
smdk2410.cec	5.00	Microsoft	Samsung SMDK2410 Development...		
platman.cec	5.00	Microsoft	Platform Manager Features		
coreos.cec	5.00	Microsoft	Windows CE Core OS components		
wcetk.cec	5.00	Microsoft	Windows CE Test Kit Client		
sourcetags.cec	5.00	Microsoft	Source code information for the sou...		
clonemodules.c...	5.00	Microsoft	Catalog items that can be cloned.		
a9m9xx0.cec	5.00	FS Forth-Sy...	A9M9xx0 V1.1 BSP Features.		

Delete the directory *%_WINCEROOT%\PLATFORM\A9M9xx0* and subfolders.

19 Links

These links represent the current status as of the printing of this documentation and may have changed when you read this.

http://www.fsforth.de	Manufacturer of ModARM9 module
http://www.embebidos.com	Windows CE integrator
http://www.netsilicon.com	Manufacturer of NS9xx0 processor
http://www.modarm9.com/	Information and newsgroup