NB Information Limited

XTM PC/XT Emulator
For Nokia Communicator

User Guide

Version 3.1

NOTICE

The information in this guide is subject to change without notice.

NB Information Limited shall not be liable for technical or editorial omissions contained herein; nor for incidental or consequential damages resulting from the furnishing, performance, or use of this material.

The software described in this guide is furnished under a license agreement or nondisclosure agreement. The software may be used or copied only in accordance with the terms of the agreement. Parts of the software are based on software developed at Carnegie Mellon University and are Copyright © 1991 Carnegie Mellon University. CMU allows us free use of their software on an "as is" basis, but disclaims any liability of any kind for any damages whatsoever resulting from the use of their software.

NB Information would appreciate your views on the contents and organisation of this publication. Please send your comments to:

Technical Publications Department
NB Information Limited
570 Lanark Road West
Balerno
Scotland
EH14 7BN

CONTENTS

1. Introduction

XTM is a software emulation of a classic PC/XT computer for the Nokia 92xx Communicator. Within this standard application for your Communicator you can run any software that would run on a low-end PC clone from the 1980's. XTM provides the following capabilities:

- Full 80186 instruction set with 8087 math coprocessor extensions.
- CGA display emulation with all text modes, and graphics mode at 320 x 200 and 640 x 200 resolutions.
- Support for MCGA display extensions (used in PS/2 Model 30) providing 256 colour palette based display.
- Native code BIOS for improved performance.
- Native code support for HP200LX graphics extensions.
- Full access to the EPOC file system.
- Access to the EPOC machine serial ports as COM1 and COM2 (including Infrared).
- PC keyboard mapping, including 12 function keys implemented as screen buttons.

This means that you can load up your favourite PC operating system and run legacy applications on your communicator. We've tested XTM with a number of different operating systems and versions, including MS-DOS (version 3.11 or later) and DR-DOS (version 7.0 or later). In the documentation, we have assumed you are running MS-DOS.

To set up your XTM system will take a little time and patience - please read the next section carefully. When you've done it, you'll have access to literally thousands of applications for the PC platform - all in the palm of your hand.

2. Start Here

2.1 What you need

To set up XTM and run your favourite PC applications, you will need the following:

1. The XTM installation diskette, or the corresponding downloaded ZIP archive unzipped into a directory on your PC.
2. A Nokia 9200 Communicator with at least 3 megabytes of free space, of which 1.6 megabytes can be on the memory card.
3. A PC running Windows and the Nokia Communicator PC suite, connected to your communicator.
4. A set of diskettes for the operating system of your choice (such as MS-DOS).
5. An hour or more of spare time.

Ideally, you should also have another PC that is already running the operating system noted in item 4 above (such as MS-DOS). This isn't required, but it makes the shuffling of diskettes a lot easier.

In the following discussion we refer to the XTM installation diskette, but you may be installing XTM from a downloaded archive file. The steps are exactly the same, but you need to access the directory on your PC where you unzipped the file instead of the diskette.

2.2 Why is it so difficult?

XTM pretends to be a PC, so to do anything useful you first have to load a PC operating system, like MS-DOS, onto your pretend PC. Loading an operating system onto a real PC is easy - you put a boot diskette in drive A and switch on. Unfortunately your communicator doesn't have any diskette drives - not real ones anyway. Instead it has pretend diskette drives which instead of reading and writing to a real diskette, read and write to special files in the communicator file system. Before we can boot up the emulated PC we have to copy the contents of a real diskette onto one of these special files.

If you use an old enough operating system, it probably fits entirely on one diskette, and so the process is pretty easy. If you're using a later system, it may take several diskettes just for the operating system - MS-DOS 6 for

instance occupies 3 disks. This is a pain in the neck when "inserting a diskette" involves copying the whole thing onto a file on your desktop PC, then downloading the file to your communicator. That's why it helps a whole lot if you have a spare PC handy. You can then install the operating system of your choice onto the hard disk of that machine, and then it's easy to create a boot diskette with just the things you want on it.

If you are careful, there is another solution using the desktop PC you already have. The reason you must be careful is because you're going to have to boot your Windows machine with the installation diskette for MS-DOS 6 or DR-DOS or whatever. DON'T LET IT GO AHEAD AND INSTALL ITSELF ONTO YOUR PC! Follow the instructions under "Making a Diskette Image" very carefully.

2.3 Installing the Emulator

First, install the emulator itself onto the communicator. This process assumes you have already installed the PC suite software on your PC, and that the PC link is working OK.

Put the XTM distribution diskette into the Windows PC, and connect your communicator to the PC. Open the diskette or installation directory in the Windows file explorer, and double click the file called NXTM or NXTM.SIS. This will automatically install the XTM program to your communicator. The installer will ask if you want to put XTM on the **Communicator** memory drive or on the **Memory Card** - you should use the Memory Card, since this frees up the most communicator memory for running programs.

Before you can use XTM, you will need to load some boot software onto the communicator too.

2.4 Making a Diskette Image

2.4.1 Starting from scratch

Let's assume you've got a copy of MS-DOS, and it's on three diskettes. Shut down your Windows PC, and then put the set-up disk into your A drive. Power up the PC, and it should boot from the set-up disk. Depending on what version of MS-DOS you're using, it will think for a while, and then it will say something like "This is going to install MS-DOS on your hard disk. Type ENTER to continue or F3 to exit." **CHOOSE THE OPTION TO EXIT!**

This will probably chide you that you have not yet installed the software and say “are you sure?”. Go ahead and exit the install script, which should drop you down to the classic “A:>” prompt.

Most installation procedures have an option to create a minimal start-up disk, which is exactly what we want. You will need to look at the documentation to find out how to do it on your version of the system. On MS-DOS version 6.0 for instance, you would type “SETUP /F”.

The installation procedure will ask you for a blank diskette, and with a certain amount of disk swapping will create a single boot disk. This is exactly what we want.

Please note that the minimum boot diskette created by this process may not have all of the files on it that you will need. In particular, if you have a UK, French or German communicator you should make sure that the DOS keyboard driver files are on the boot disk - on MS-DOS these are KEYB.COM and KEYBOARD.SYS. You will need these files to access the local special keys on your keyboard.

2.4.2 Starting with a DOS PC

If you already have a PC running exactly the same version of software that you are installing on XTM, life is much easier. Now, instead of shuffling disks you can format a boot diskette directly, and copy on whatever system files you wanted.

In case you’re still wondering why you can’t just do this from a DOS session on your Windows 95 or Windows 98 PC, remember that most MS-DOS utilities are keyed to a particular version of DOS, and will not run on the wrong version. Windows 95 and 98 will not run on XTM, because they need a 386 or higher CPU, so you can’t make a boot diskette from your Windows system directly. And Windows DOS utilities will not run under older versions of MS-DOS.

2.4.3 Testing, Testing, Testing

What you’ve just made should be a genuine, working, PC boot diskette. Before you spend too much time going much further, it’s probably worth checking to see if it really works. Shut down your PC, and try booting it from the diskette you’ve just made. If it doesn’t work on your real PC, it definitely wouldn’t work on XTM.

Once you're happy that it boots up OK, eject it from the drive, and reboot your PC back into Windows.

2.4.4 Installing the XTM utilities

XTM includes a device driver to let MS-DOS access the whole of your communicator file system. The driver itself is called EPOCFS.SYS, and it has a companion utility called EREDIR.EXE. These files are on the XTM distribution diskette. Copy the two files from the XTM distribution diskette onto the boot diskette you've just made.

Note: The Windows file explorer has an option to hide system files, which includes those with a file extension of SYS. If you've got this option enabled, and you try to copy the files from the distribution diskette under the explorer, EPOCFS.SYS will seem to be missing from your diskette. Either use **View>Options>Show All Files** in explorer, or just pop up a DOS box, and copy the files the old-fashioned way.

In order to use EPOCFS.SYS you must add it to the CONFIG.SYS file on the new boot diskette. You can do that on your communicator later, but you may find it more convenient to do it now. Depending on the version of operating system you are using, the file A:\CONFIG.SYS may not exist yet - if it does not exist, create a new one. CONFIG.SYS is just a text file, so open or create it with Windows Notepad, or your favourite text editor. Add the line:

DEVICE=EPOCFS.SYS C:

to CONFIG.SYS on the boot diskette. This will make the DOS C: drive equivalent to the communicator C: drive when you run XTM. Only add the device driver once - if you need more redirections you can add them with the EREDIR utility later.

By the way, if you skipped the testing stage earlier - you are now too late to try it. For reasons that should not be too surprising, the EPOCFS device driver will NOT run on a real PC. If you want to boot from this diskette outside XTM you must first remove the EPOCFS device driver line from CONFIG.SYS.

2.4.5 Installing additional software

As a final action before you package up the contents of the boot diskette, you might want to copy any other of your favourite utilities onto it that fit. This

means that as soon as you've got it across to your communicator, you might actually be able to do something useful.

A crucial DOS utility which is not part of the minimal boot set is the foreign language keyboard driver KEYB.COM, and its associated configuration file KEYBOARD.SYS. If you have a UK, French or German keyboard on your communicator, you will need these files to correctly map the keyboard.

2.4.6 The MAKEDISK utility

The last thing you need from the XTM distribution diskette is the MAKEDISK utility, which is going to turn the boot diskette we've prepared into a file which XTM can use. Copy the file called MAKEDISK.EXE from the XTM distribution diskette into a directory on your PC hard disk.

MAKEDISK is a DOS program, so you'll need to open an MS-DOS prompt window to run it. Put your boot diskette back in drive A, and type the command:

MAKEDISK A

The MAKEDISK utility will then read the entire contents of the diskette into a file on the hard disk called DRIVEA.DSK. This file will be the same size as the diskette capacity - usually 1.44 megabytes.

2.5 Copying the Diskette Image

All you need to do now is copy your diskette image onto your communicator. Connect the machine to the PC in the usual way.

The emulator expects to find the boot image in the file ?:\XTM\DRIVEA.DSK, where the ? represents one of the drives on your communicator.

The Nokia Communicator calls drive C **Communicator** and drive D **Memory Card**. On the Nokia machine, there is generally not enough space on **Communicator** to store XTM and the diskette image, and then run the program. You should always install the program and the diskette image on the Memory Card. You can create a directory to hold the image using the File Manager on the communicator.

Then just drag and drop the file called DRIVEA.DSK from wherever it is on your hard disk onto the XTM directory you've created in your communicator. The PC suite will copy it across. This is a good time to head off for a cup of tea. The DRIVEA.DSK file is usually 1.44 megabytes, and it will take a long

time to copy across the link - usually between 5 and 10 minutes depending on the speed of your PC.

2.6 Starting XTM

At last - you can see it working! On the communicator, bring up the Extras bar, where you should find a new icon for XTM. Select and open the icon, and you will be rewarded with a very PC-like display which boots and runs your operating system.

In the unlikely event that it doesn't seem to work, please refer to the Problem Solving section later in this manual.

2.7 Back it up

Having got this far, it's a good idea to take a backup of your communicator. Again, this will take a while, because it has got some big chunks of data on it, but it is very worthwhile if anything goes wrong.

3. Commands and Controls

3.1 Using the Keyboard

The keyboard of your communicator is somewhat different to the keyboard of a PC. It makes many compromises on layout in order to get larger keys into a smaller space.

It uses the **Chr** key as an extra shift key to access additional characters, and the assignment of punctuation keys around the keyboard is different to a standard PC layout. The XTM emulator therefore cannot directly map keyboard scan codes into the PC equivalent - an extra translation stage is needed. For most applications this is fine, but there are two important restrictions. Firstly it mean that you will not be able to run programs that need to differentiate between multiple keys pressed as "chords". Some games depend on this and can't be used on a communicator. Secondly, XTM uses the **Chr** key to simulate pressing a PC **Alt** key, but the communicator needs the **Chr** key to generate some characters. Some **Alt** key combinations are therefore not accessible as single keystrokes from the communicator keyboard.

XTM translates **Chr + LetterKey** into **Alt + LetterKey** for all alphabetic characters. Some punctuation characters are accessed using **Chr + NumberKey**, so the **Chr** key does not behave like an Alt key for the number keys.

The communicator doesn't have any function keys, so these are implemented using a dialogue box accessed from the command buttons to the right of the screen. Tapping the appropriate command button, then the key corresponding to the function key you wanted, sends the function key-press to XTM.

Finally, remember that PC keyboard layouts are different depending on what country you are in. The emulation generally reflects the keyboard in your communicator, and unless it is a US keyboard you will probably have to run a DOS utility in your AUTOEXEC.BAT that maps the keyboard accordingly. For UK keyboards for example, you would run:

KEYB UK

3.2 Changing Imaginary Diskette Disks

Occasionally you might need to change which diskette is in the drives emulated by XTM. The diskette drive is mapped onto a file in the communicator file system, and if you have the images of several diskettes you can swap from one diskette to another, just like on a real PC.

Select the pop-up menu bar and choose the Change Diskette menu item. It will ask you to choose whether you want to change drive A or drive B. Choose whichever you want, and then select the file corresponding to the diskette from the dialogue box that appears.

3.3 Rebooting the XT Emulator

You can reboot the emulator at any time by clicking on the Reboot button at the right of the screen. This has exactly the same effect as pressing the reset button on the front panel of a real PC - a useful feature that the original PC/XT didn't have!

3.4 Altering the View

3.4.1 Full Screen

You can make the XTM screen fill the whole of the communicator display by selecting Full Screen from the View menu. Because PC applications expect the screen to be a fixed width, switching in and out of Full Screen mode has no effect on the programs running in the emulator. If you have not selected Full Screen mode you will not be able to see all the columns of text on the screen.

3.4.2 Colour Mapping

XTM allows you to alter the way it maps the emulated PC colours onto your LCD screen.. Select View from the pop-up menu bar to choose which of two styles you prefer.

You can choose to display text screens inverted - this may make it easier to read some colour combinations. The View menu does not alter the display in graphics mode. Changing the colour mapping from the View menu has no effect on the programs running in the emulator.

3.4.3 Text Zoom

When displaying PC text on a screen like the communicator the characters need to be small so that all 25 rows of the PC screen can be seen at once. If you don't need to see 25 rows you can use the zoom feature to make the characters larger and easier to read. You can zoom in and out using the usual keyboard commands, or by selecting View on the pop-up menu.

Zooming in does not change the emulated display size - your DOS application programs will still think there are 25 rows of text on the screen. XTM is providing a window onto the emulated screen, and it chooses which text lines to display by moving the window to keep the text cursor visible.

XTM only scrolls the window up and down, not left to right. If you use XTM with the status bar and command buttons visible, you will not be able to see all 80 columns of text, even when zoomed out to the smallest text size.

There is no zoom function in graphics modes. Instead, the graphic display is always grown or shrunk to fit the screen you have.

3.5 Using Serial Ports

XTM emulates two PC style COM ports. You can choose how these ports are mapped onto the actual serial interfaces on your communicator by selecting Communication Settings from the Tools menu. Each COM port can then be directed to one of:

- The standard serial port on your communicator.
- The Infrared interface on your communicator.
- Nothing - which is the recommended setting if you are using batteries and are not using the serial interfaces.

The Communications Settings dialogue can also enable CTS (Clear To Send) flow control on the serial port. This makes XTM more responsive to peripheral devices that rely on CTS flow control, such as some printers and plotters. If you're not sure whether you need CTS flow control, leave it turned off.

4. Using the File Server

4.1 The Device Driver

The EPOCFS.SYS device driver that you loaded in the CONFIG.SYS file is a small resident program that collects DOS file redirector calls and sends them to a native EPOC32 file server. As far as DOS is concerned, it thinks it is talking to a network file server.

Just like any other DOS networking program EPOCFS lets you create remote drives, and accesses to files and directories on those drives are redirected to the server. In this case though, the server is your own communicator.

When you launch EPOCFS from your CONFIG.SYS file with a directory name parameter, it allocates the first free drive letter, and points that drive at the directory you specified. Because XTM doesn't have any hard drives, the first free drive letter is C:. If you use the command

DEVICE = EPOCFS.SYS C:

you will make the DOS directory C:\ point to the EPOC32 directory C:\. In other words, you've made your PC emulator map C: directly onto the same drive in your communicator.

You don't have to do it that way - you could try

DEVICE = EPOCFS.SYS C:\DOCUMENTS

instead, which will mean that the top level of C: in your DOS session appears to be the Documents directory of your communicator.

If you want to map more drives, such as directories on a flash disk, you need to use the EPOC redirector control application, EREDIR.

4.2 The Redirector Control Application

The redirector control application lets you set up new drive mappings, delete the ones you've got, or just show the current settings.

4.2.1 Showing Current Redirections

To show all the current drive redirections, simply type:

EREDIR

at the command prompt. You should see an output something like this:

```
Current Drive Redirections:
C:  = EPOC32\C:\
```

4.2.2 Setting up a New Redirection

To add a new redirection - for example to access a directory on a memory card - you would use the following syntax:

EREDIR D: EPOC32\D:\MYFILES

where the first parameter is the drive letter you want to map, and the second parameter is the path name in the host system you want to map it to, prefixed by EPOC32\.

4.2.3 Deleting a Drive Redirection

To unmap one of your drive redirections, use the following syntax:

EREDIR DEL D:

where the first parameter is DEL and the second parameter is the drive you want to unmap.

4.3 Copying Files

Because the EPOCFS redirector provides access to the underlying EPOC32 file system, XTM does not include any dedicated file copying routines to move files to and from your PC. To move a file from your desktop PC to your XTM environment, simply use your existing PC suite software to drag and drop the file onto your communicator. Copy it into a directory which you have made accessible to XTM as described above, and you can immediately use it within XTM.

Any DOS file that you have stored in the EPOC file system can be copied back to your desktop PC by the same method.

One thing you need to watch out for is files which get converted by the PC suite into EPOC32 formats. If you are using a DOS word processor, spreadsheet, or database program with XTM, you may need to alter the conversion settings. Otherwise, when you drag and drop a file from your PC to your communicator, you may find that it has been converted into something the native DOS program doesn't recognise.

4.4 Redirector Limitations

The communicator file system uses long file names, but the DOS environment is still restricted to 8 character names with 3 character extensions. The redirector can't map long filenames, and any file name that is too long for the DOS 8+3 character format is simply ignored. You therefore cannot access files or directories in the communicator file system from within DOS if they have long file names.

5. Registering XTM

5.1 Introduction

Although XTM is available for free evaluation, if you decide you want to keep it and use it, you must purchase a license from NB Information Limited. Details of purchasing arrangements can be found on the NB Information web site at **www.nb-info.co.uk**.

XTM is licensed to an individual user. In order to purchase a license you must tell us what name your copy should be registered to. We will then provide you with a Registration Code which unlocks a copy of XTM installed by that user.

5.2 Registering the Program

When you buy a license from us, you will receive an 8 or 9 digit Registration Code. To register your copy of XTM you need to run it, and then pop up the Menu bar. Choose Tools and then Registration, and type your Registered Name **exactly as you provided it when you purchased the licence** and the Registration Code into the dialogue box. When you click OK, the program validates your Registration Code - if the code is incorrect it will tell you.

If the code was correct then XTM continues running, but in future when you launch the program it will identify itself in the little information box as a Registered Copy.

6. Tips and Hints

6.1 Turn Off COM Ports

Whenever a COM port is mapped to a real device, such as the infrared port, the hardware is activated. This can consume significant battery power, and also slightly slows down the emulator. If you are not using a serial port, map it to "nothing" in the Communications Settings dialogue.

6.2 Switch Off, Save Power

All the time it is loaded, even when it's just sitting at the A:\> prompt, XTM is running the CPU. This will quickly run down your batteries if you're not careful - battery consumption with XTM running in the foreground is nearly twice the normal current. When you are not using XTM, you should save your files, and then exit by clicking on the bottom right button. This is like switching off your PC. Next time you run XTM it will boot up the operating system again, and you can carry on from there.

6.3 Snooze, Save Power

What if you're three hours into a major project, and you really don't want to have to shut down XTM? You need to use the communicator for something else though, but you don't want it to eat up the batteries. When you switch to another task while on battery power, XTM automatically goes into snooze mode. It still runs the emulated CPU, but only very slowly. In this low power mode, the emulator draws very little extra current from the batteries.

Occasionally a DOS program will get confused by this process, because it will think it is in a weird time warp. It is therefore always a good idea to save your work before switching to another task.

6.4 Arranging Your Files

You can't boot XTM from the internal drive of your communicator - you must have a diskette image file to boot from. These files are a fixed size, and depending on how you created it, the file is probably 1.44 megabytes long. It will be the same size even if the diskette it emulates isn't actually full. This is wasteful of space, especially if you've got other DOS files stored elsewhere on

your Memory Card. Try to cram as many of your files onto the diskette image as you can, because this will save you space overall.

6.5 More about MAKEDISK images

The MAKEDISK utility that you use to make the initial boot image will try to make an image from any normal sized PC disk. It can read from disks of 360 kilobytes, 720 kilobytes, 1.2 megabytes, 1.44 megabytes and 2.88 megabytes. The emulator will correctly recognise each of these sizes when it boots from DRIVEA.DSK. This means that you are not restricted to having a 1.44 megabyte boot image on your communicator.

In order to take advantage of these different sizes, you must first have the ability to make a valid boot disk of that size. To make a 1.2 megabyte or 360 kilobyte disk image you will need a 5.25 inch disk drive. Most modern PC's can use only 1.44 megabyte or 720 kilobyte diskettes in their 3.5 inch drives. Still, if you are short of space on your communicator, making a smaller boot diskette image may be very worthwhile.

7. Problem Solving

7.1 Problems running MAKEDISK

7.1.1 Sector could not be read

MAKEDISK prints an analysis of the diskette drive and media before it starts copying. If this doesn't match what is in the drive the diskette is probably incorrectly formatted.

If it does match, there is probably a bad sector on the diskette - try running SCANDISK on it.

7.2 Problems starting XTM

7.2.1 Fatal Error: Couldn't start Emulator

The most likely is that you have not provided a DOS boot disk image in "D:\XTM\DRIVEA.DSK". See section 2 for instructions.

If you do have the boot disk image, the next most likely problem is that your communicator does not have enough free memory to run the program. On the Control Panel, accessed from the Extras button, run the Memory tool. If there's less than a megabyte before you run XTM, it will not be able to allocate the memory it needs for the PC address space - which is a megabyte. If you are close to having enough space, try closing other open applications, which will free up some memory. Otherwise, you will need to delete some other files or applications to make room.

7.2.2 Fatal Error: Couldn't construct Display

Again, memory problems are the most likely culprit. See comments above.

7.3 Problems running applications

7.3.1 I have a UK/French/German Communicator keyboard; some of the keys come out wrong.

XTM translates UK, French and German communicator keyboards into the corresponding UK, French or German PC keyboard. Just as on a real PC, you must load a local language keyboard driver to access all the keys correctly. Under MS-DOS this would usually involve running KEYB UK/FR/GR from your AUTOEXEC.BAT file.

There are still a couple of keys on the communicator which are Nokia specific, and there is no equivalent key on the PC keyboard. These keys are ignored by XTM.

7.3.2 I tried to run a program under XTM but it crashes. It works fine on my Windows 95 PC in a DOS window.

Unfortunately, a DOS window on a Windows 95 PC is a long way from running real MS-DOS on XTM, and many things behave differently as a result. For example, you can run 32 bit programs in a Windows 95 "DOS box" which real MS-DOS wouldn't even recognise. If your desktop machine is a 80486 DX or Pentium based PC it will include a floating point coprocessor, which is not present in XTM.

Before you go much further, you should try to run your program on a real DOS system. If it works on a PC under DOS, but not on XTM, let us know what the program is, what exactly happens, and we'll see if we can come up with a solution for you.

7.3.3 I have a compiled BASIC program which seems to run but then crashes the whole system

Certain BASIC compilers use 8086 software interrupts to access internal functions. In extreme cases the BASIC run-time system trashes all interrupts from 0x80 to 0xED, and they are not restored when the BASIC program exits. Unfortunately, XTM uses a couple of software interrupts in this range to access the EPOC file system. Not surprisingly, XTM crashes next time anything tries to access drive C: after such a program has run.

Restricting the system to emulated diskettes, and not using EPOCFS.SYS will allow such programs to run, but you will have to reboot the system before you can access the EPOC file system.

7.3.4 When I run XTM it says "COM1 Unavailable". Why?

Only one program can access the communicator serial port at once. This message means another program was already using the serial port when XTM tried to open it. The most likely culprit is the PC link program.

Once XTM has found the interface to be busy, it will not go back and try to open the COM ports later. You must either exit and restart XTM, or select Communication Settings on the XTM Tools menu if you want to change the COM port configuration.

7.3.5 Problems with communications applications

The timing characteristics of the emulated serial I/O ports is very different to a real PC. Many programs which expect particular behaviour will be confused by XTM. Sometimes this can be improved by running the serial port at a lower bit rate. For more details, see section 8.7.

7.4 Registration

7.4.1 I tried to register, but it said “Incorrect Registration Code”

Check that the Registered Name that we listed with your Registration Code matches what it says in the registration dialogue box. It is possible that we transcribed it wrongly from your order, which would generate a mismatched Registration Code. If there is a mistake, contact us for a new Registration Code.

7.4.2 XTM says “Your test and evaluation period has expired”

You need to buy a license for XTM to continue using the program. If your copy has completely expired you will not even be able to run it to type in the registration code.

Change the date on your communicator to a time in the past before XTM expired, type in your Registration Code, and then you can re-set the date correctly.

8. Technical Details

8.1 Basics

XTM should make your PC operating system and applications believe that they are running in a rather slow PC/XT based computer. The machine looks to the software like an 80186 computer with an 8087 math coprocessor, two diskette drives, a MCGA display, an AT style keyboard, and two serial ports.

If you're running off the shelf software, and not trying to do anything obscure, you should never need to know any more than that. If you're interested in the details of the emulator, read on.

8.2 CPU Emulation

The CPU emulator implements the complete Intel 80186 instruction set, including so-called "undocumented" instructions. The 80186 is an enhanced version of the 8088 used in the original IBM PC. It implements all of the instructions of the 8088 and many 80286 extensions, but does not include any of the 80286 memory management functions.

Conventional software should find no differences whatsoever in running under XTM compared with running on a real 8088. There are however a small number of practical differences as follows:

1. After certain instructions, Intel documents certain condition codes as undefined. These may behave differently under XTM.
2. On a real 8088, fetching or storing to a 2 byte memory location where the lower byte is the last byte in a 64k segment will always wrap the upper byte address to the bottom of that segment. XTM will access the two bytes at contiguous locations, therefore overflowing the 64k segment boundary. Since other x86 processors will crash the program in these circumstances we do not believe any commercial software relies on this behaviour.
3. Similar to the previous point, if valid instruction codes run right up to and across the end of a 64k code segment, the emulator will keep executing off the end of the code segment, rather than wrapping the instruction pointer. It is very unlikely that any commercial software would rely on the wrap-around effect of the instruction pointer in a real 8088.
4. There is little correlation between instruction execution times on a real 8088 or 80186 and the XTM CPU.

Notwithstanding the comments about accesses across segment boundaries, the XTM CPU does correctly implement the overall address space wrap-around of a real 8088. That is to say, address FF00:1100 and 0000:0100 are both the same place.

8.3 Math Coprocessor Emulation

XTM includes an emulation of the 8087 math coprocessor. The 8087 was the floating point chip designed to complement the 8086 and 8088 processors used in the original PC. The 8087 can also be used with the 80186 processor, and although an 80187 chip has since been introduced the 8087 was the standard for early PC's.

Emulation features include:

- All 8087 instructions implemented.
- All basic math operations (add, subtract, multiply, divide) implemented exactly as in the real 8087.
- All precision and rounding modes implemented exactly.

There is an important difference between emulating a standard CPU and emulating a math coprocessor, and that is in the accuracy of emulation. As well as the obvious add, subtract, multiply and divide functions, the math coprocessor also provides implementations of Logarithm functions, Exponential functions and Trigonometry functions. The 8087 implements these so-called transcendental functions using algorithms that return approximations to the correct answers. That is, although the functions provide a "good" answer, the answer may not be correct to the least significant bits. Because our emulation also uses approximating algorithms, it is likely that our corresponding functions may also not be correct to the least significant bits. More importantly from the point of view of an emulator, it is very likely that our "approximation error" is NOT the same error as that of a real 8087, and therefore our answer is likely to be slightly different to theirs.

The following features are NOT guaranteed to match a real 8087, and programs that rely on them may not work correctly:

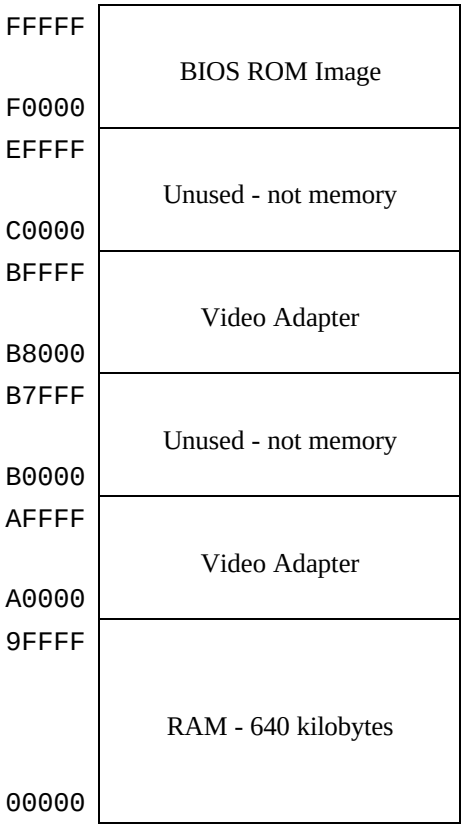
- Transcendental function results may differ by several ULP's.
- Operations that cause overflow or underflow may behave differently.
- Floating point exceptions may not be signalled with the same priority, or in the same order, as on a real 8087.

8.4 Memory Map

The XTM memory map is a conventional PC/XT layout, as shown below.

Note that the BIOS ROM image is implemented in RAM, and therefore an errant program can corrupt what would in a real PC be read-only memory. The ROM contents are recreated every time the emulator is rebooted.

XTM Memory Map



8.5 XTM BIOS Implementation

The BIOS implementation is a reasonable subset of the full BIOS in a real PC. Most of the work of the BIOS is performed by native EPOC32 code for performance reasons. The following BIOS calls are implemented:

Interrupt	Function	Subfunction
10h	Video	00h - Set Video Mode
		01h - Set Cursor Type
		02h - Set Cursor Position
		03h - Read Cursor Position and Type
		05h - Set Display Page
		06h - Scroll Page Up
		07h - Scroll Page Down
		08h - Read Character and Attribute
		09h - Write Character and Attribute
		0Ah - Write Character Only
		0Bh - Select CGA Palette
		0Ch - Write Dot
		0Dh - Read Dot
		0Eh - Write Teletype Mode
		0Fh - Read Video State
		10h - Set MCGA Palette
		13h - Write String
		DCh - Graphics Cursor Control
11h	Equipment Flags	N/A
12h	Memory	N/A
13h	Diskette	00h - Controller Reset
		01h - Read Diskette Status
		02h - Read Diskette Sectors
		03h - Write Diskette Sectors

		04h - Verify Diskette Sectors 08h - Read Drive Parameters 15h - Read Drive Type 16h - Get Change Line Status
14h	Serial	00h - Initialise Port 01h - Send Byte 02h - Receive Byte 03h - Read Status 04h - Extended Initialise 05h - Extended Control Read/Write
15h	System	41h - Wait for External Event 42h - Power Off Request 60h - Read Battery Information C0h - Return BIOS Configuration
16h	Keyboard	00h - Read Keyboard 01h - Check Status 02h - Get Shift Flags 05h - Store Key Code 10h - Read Extended Keyboard 11h - Check Extended Status 12h - Get Extended Shift Flags
17h	Printer	00h - Print Character 01h - Initialise Printer 02h - Get Printer Status
19h	Reboot Machine	N/A
1Ah	Time	00h - Get Ticks 01h - Set Ticks 02h - Get System Time 03h - Set System Time 04h - Get System Date 05h - Set System Date

8.6 Video Modes

XTM implements all video modes of an MCGA adapter. These are:

Mode	Type	Resolution
0	Text	40 x 25 monochrome
1	Text	40 x 25 colour
2	Text	80 x 25 monochrome
3	Text	80 x 25 colour
4	Graphics	320 x 200, 4 colours
5	Graphics	320 x 200, 4 grey scales
6	Graphics	640 x 200, monochrome
17	Graphics	640 x 480, monochrome
19	Graphics	320 x 200, 256 colours

The MCGA interface was introduced in the IBM PS/2 Model 30 as a link between the older CGA interface and the VGA interface. The MCGA is completely backward compatible with the CGA, and all software designed for the CGA will work. The main advantage of the MCGA is the availability of mode 19, which offers 256 colours from a palette of 262,000 and is used by many games and other graphics applications.

Please remember that rendering an image in mode 19 may take longer than it does in mode 4. For a program where the benefits of 256 colour support are limited you may find that configuring your software to use mode 4 instead will give performance advantages.

Software designed specifically for EGA, VGA or other advanced graphics controllers will not work on XTM.

8.7 HP200LX Graphics Support

XTM includes a native code implementation of the graphics functions provided on the Hewlett-Packard palmtop computers through Interrupt 5Fh. Functions implemented are:

Function	Description	Restrictions (if any)
00h	Set Mode	Only mode 6 (640 x 200) supported
01h	Set Fill Pattern	
02h	Get Info	
03h	Set Origin	
04h	Set Clip Region	
05h	Draw Rectangle	
06h	Draw Line	
07h	Set Pixel	
08h	Move Pen	
09h	Set Pen Colour	
0Ah	Set Replace Rule	
0Bh	Set Line Type	
0Ch	Get Pixel	
0Dh	Get Image	
0Eh	Put Image	
0Fh	Write Text	Rotated text not supported
10h	Get Font	3 built-in fonts provided (8x8, 11x10, 12x16)
11h	Set Font	
12h	Scroll	
13h	Set Info	
14h	Set Defaults	
16h	Draw Button	

Because XTM does not implement many other hardware and software features of the Hewlett-Packard palmtop machines, the Interrupt 15h installation check function does NOT identify XTM as an HP machine. Applications written for the HP machines that check this function may not run under XTM, even though the Interrupt 5Fh services are present.

8.8 Serial Ports

XTM supports two serial ports, mapped according to the settings in the Communication Settings dialogue box.

The serial ports can be accessed either by using the BIOS interrupt 14h services, or by directly writing to the emulated hardware. The serial ports emulate Intel 8250A devices.

8.8.1 Serial Port Restrictions

There are a number of differences between serial ports on a real PC and the XTM emulation.

1. The biggest restriction on serial communications is the fact that XTM is slower than most real PCs. Although the XTM serial ports can be set to speeds of 115200 bits per second, not many applications will be able to keep up with receiving data at that speed.
2. On a real PC, software that directly programs the UART chip can set any bit rate it wants. EPOC32 only supports the following bit rates: 50, 75, 110, 134, 150, 300, 600, 1200, 1800, 2000, 2400, 3600, 4800, 7200, 9600, 19200, 38400, 57600 and 115200 bps. Any other figures set by your PC software will be rounded up to the nearest EPOC bit rate.
3. The Communication Settings dialogue allows you to force XTM serial ports to always obey hardware flow control on output, using the Clear To Send (CTS) signal. That is, XTM will not transmit characters when the CTS signal is off, even if your DOS program tries to ignore CTS. This is so that hardware flow control works fast enough for external devices like modems and printers, even though the emulator response to modem signals is rather slow. The side effect is that simple serial cables which ignore the CTS pin will not work. If you are not using CTS for flow control you must turn it off in the Communication Settings dialogue box.
4. The XTM serial port modem signals have significant timing differences compared with real PC ports. The delay between setting a bit in the

UART and the corresponding serial port wire changing state in a real PC is a few microseconds. In XTM the same period could be 2 milliseconds. The delay between a change of state in an external modem pin and the corresponding change being observed in the UART is a few microseconds on a real PC. In XTM, the same change can take 50 milliseconds to be reported. Many direct-connect communications packages which use the modem control lines to handshake between two PC's will therefore not work on XTM.

5. Because the IrDA interface is very different from a PC serial port, the mapping between a COM port and an infrared connection in XTM is slow.

8.9 LPT Interface

XTM implements a BIOS printer interface, but unlike a real PC this drives a serial port, not a parallel port. The effect is similar to that achieved by mapping the printer to the COM port with the MODE command:

MODE LPT1:=COM1:

The BIOS printer driver has an advantage over this technique, because it has better flow control capabilities. The standard BIOS driver for the COM port uses only hardware flow control using the DSR and CTS modem control lines. The BIOS driver for the LPT port uses both hardware and software flow control. The LPT driver will transmit when it receives DSR and either (or both) of DCD or CTS. It will also correctly respond to XON/XOFF flow control from the printer.

To allow maximum flexibility of printer type, the BIOS printer driver does NOT alter the COM port settings such as bit rate and parity. Before using the LPT interface you must set the appropriate parameters for your printer.