How to Install and Configure MS-DOS 6.22

Last Modified: Sun, 08/11/2013 - 16:29

Skip to: Preparation | Installation | System Configuration | Network Configuration | Device Configuration | System Optimization | Additional Applications | Addendum

Introduction

This walkthrough covers installing MS-DOS 6.22 from the original installation diskettes. Why write this in 2013? That's a very valid question, to which there are a few answers:

> Setting up a fully working DOS system will give you great appreciation for how far computing has come. For old-timers, it will be a walk down memory lane; for youngsters who've never used nor even seen DOS before, it should be quite an eye-opening experience to experience first hand both how primitive DOS was and yet how capable it could be.
>
> A working physical DOS system is the most authentic way to (re-)experience classic PC games. DOSBox does an amazing job of supporting DOS games on modern platforms, but for perfect accuracy, including the full memory management experience (which can be a game unto itself), a real DOS system can't be beat.
>
> There is a dearth of detailed information about MS-DOS on the internet. This makes sense as MS-DOS predates the web as we know it today, but I don't want knowledge of this system to be lost to time. I did a significant amount of research for this project, and I want to document and share what I've discovered and re-learned for future reference.
>
> Perhaps most importantly, why not? This project was inspired by a previous project to resurrect my old Packard Bell, my first computer that, not coincidentally, ran MS-DOS 6.2 and Windows for Workgroups 3.11. Rebuilding and enhancing it from a hardware perspective was a fun experience, and now I'm doing the same from a software perspective.

Honestly, if you have no appreciation for old hardware or software, then this is definitely not for you. If, however, you share my passion for technology, not only for the new hotness[1] of today but also the old and busted (and tried and true) of yesterday that got us to where we are today, then I think you'll find this interesting. If you have some old hardware lying around then I hope you'll follow along, but even if not I think you may still find some of this interesting enough to read.

As an alternative, if you want an easy-to-install version of DOS that includes some nice modern conveniences, check out FreeDOS. It's a great project that I highly recommend. For this project, though, I want a (mostly) authentic, original MS-DOS installation.

Note: I provide download links for all discussed software in the relevant section where it's discussed. Links point to the original download location for each file wherever possible, but for the files that no longer have an official source (or a reliable one, in the case of the files hosted on Microsoft's amazingly unreliable FTP server), I've linked to a local copy you can download instead.

Prerequisites

> Old hardware - if it has ISA slots you're probably good to go; anything newer may require some extra work, but it should still be possible to get at least a basic working system installed.
>
> > Alternatively, you should be able to get this up and running in a virtual machine with VirtualBox or VMware Player, but as with the note about FreeDOS above I'm primarily interested in an authentic experience for this project, which is what's documented here.
>
> A 3.5" floppy disk drive and at least one floppy diskette (two or more recommended) - It may be possible to hack together a solution that will work from a bootable CD-ROM (see this Tech Support Guy thread for details if you prefer to try that route), but MS-DOS is really only intended to be installed from floppy diskettes.

MS-DOS 6.x installation media. If possible, I suggest using or tracking down any original installation media you may have had (in my case, I was able to pull the original MS-DOS 6.2 diskette images off of my Packard Bell recovery CD) or picking up a set on eBay - unless you want a full boxed set, the media itself is quite cheap. If you don't have access to any legit copies and don't want to go the eBay route, you can find a copy online easily enough (I recommend the WinWorld Software Library). I don't generally condone piracy, but given this is twenty year old software that's no longer commercially available, I see no harm at all here.

Patience, basic CLI experience, and a willingness to tinker - this process will take some time, and you'll likely run into issues here and there that'll require some extra time/effort/thought to work out. Part of the experience here is the journey itself, so if you get immediately frustrated at any given setback you will not enjoy this project. Basic CLI experience is also expected; I hope to provide enough guidance to get you through this project without the need for too much prior experience , but I have to assume you have at least a basic familiarity with the command line.

Tip: I also recommend grabbing a copy of either Universal Extractor or 7-Zip if you're running Windows on your main computer, or p7zip and LHa for UNIX if you're running Linux (both should be available in your package management system). You'll probably need/want to unpack some of the software and drivers listed below on your main computer before copying it over to your new DOS system, and some of these are packed in fairly obscure (for today) formats. These applications should cover all the software I tried to unpack, so having these tools available in advance will save some time and hassle.

Return to top

Preparation

Relevant Software:

RawWrite for Windows - Unless you have physical installation media, you'll need to write the floppy disk images to real diskettes. RawWrite is a simple way to do this in Windows. Linux can do this natively with `dd` (discussed below)

Floppy Diskette Creation

If you need to create the installation media:

Under Linux, use this command: `dd if=disk1.img of=/dev/fd0 bs=1024 conv=sync; sync`

Under Windows, use RawWrite and follow the directions to write the first disk

Repeat the process for all three installation disks (you may have a fourth disk as well; this contains the optional Supplemental Utilities, and is not needed for initial installation; see below for additional details)

You can get by with only one floppy diskette by writing the next disk image after the previous one completes installation, but I highly recommend using at least two diskettes so that you can have a copy of Disk 1 on hand at all times. You'll likely need to boot from or otherwise use that disk a number of times

Hard Disk Preparation

If you need/want to partition and/or format your hard drive:

1. Boot your computer using Disk 1
2. Press `F3` at the MS-DOS Setup welcome screen to exit the installer
3. To repartition, run `fdisk`
    1. Use option 3 to delete existing partitions, then option 1 to create new partitions

        Note: MS-DOS 6.x can only recognize FAT16 partitions, which are limited to 2 GB in size.
    2. If necessary, use option 2 on your C: drive to flag the "boot" partition
    3. You will be forced to reboot after making changes
4. To create (or reformat) a file system on your hard drive:
    1. Run `format /v:labelname /u /s c:`

        `/v` sets the disk volume label; this can be omitted

`/u` performs an unconditional format, which omits preserving certain recovery information

`/s` copies DOS system files to the partition, allowing it to self-boot

   2. Repeat the above command for any additional partitions, but omit `/s`

5. Run `setup.exe` to resume installation

Return to top

Installation

Relevant Software:

MS-DOS 6.22 Step-Up - free upgrade for all MS-DOS 6.x installations to 6.22; if you only have the MS-DOS 6.0, 6.2, or 6.21 installation media (my copy is version 6.2), download this (local copy)

MS-DOS 6.22 Supplemental Utilities - optional additional utilities, drivers, and programs only included with previous versions of MS-DOS; not required, but may be of some interest for the curious (grab SUP622.EXE) (local copy)

Aside from possibly needing to FDISK and FORMAT your hard drive, the base MS-DOS installation is actually quite simple:

1. If you haven't already done so, boot your computer using Disk 1
2. To begin setup, press Enter
3. If you already pre-formatted your installation disks as described above, the installer will warn you about already having an existing version of DOS installed. Choose to "Continue Setup and replace your current version of DOS".
4. Set the options for date, time, country and keyboard layout as appropriate, then choose "The settings are correct" to continue.
5. For the installation directory, I recommend choosing the default of `c:\dos`. Press Enter to continue.
6. Switch disks (twice) when prompted, and reboot to complete installation

After rebooting, you'll be in a fresh, and very basic, DOS environment.

Step-Up

If you installed either MS-DOS 6.0, 6.2, or 6.21, you may (optionally) upgrade to 6.22 using MS-DOS 6.22 Step-Up. Since it's free, though, there really isn't a good reason to *not* upgrade.

The most annoying part of this process is just getting the Step-Up files to your DOS system. It's larger than a single floppy, and meant to be downloaded and run directly on the target computer, which is difficult for a freshly installed version of MS-DOS. To workaround, I suggest the following:

1. Download and unpack stepup.exe on your main computer
2. Manually create a floppy disk set as follows:

   Disk 1: setup.bat readme.now 1msdos62.exe

   Disk 2: 2msdos62.exe

   Disk 3: 3msdos62.exe
3. Create a backup directory on your DOS system to hold the Step-Up files (this is useful if you want to reinstall DOS - you can the reinstall Step-Up without copying everything via diskettes again):
   1. `mkdir c:\backup`
   2. `mkdir c:\backup\stepup`

   Tip: Even better, if you have the hard drive space, consider creating a D: drive as well; you can use your D: as your backup and data partition, so that if you want to reinstall you can simply blow away C: at any time but still have all your files, drivers, etc. already saved on D:
4. Copy the upgrade files from each diskette to your backup directory: `copy a:\*.* c:\backup\stepup`
5. Finally, copy your backup stepup directory to a temp directory that can be removed after the upgrade is completed:

1. `mkdir c:\stepup`
2. `copy c:\backup\stepup c:\stepup`

To install/upgrade:

1. `cd c:\stepup`
2. Run `setup.bat`
3. Enter `Y` to agree to the EULA
4. Enter `Y` to verify that C: is your OS drive
5. At the Welcome screen, press `F3` to exit
6. Run `setup.exe /g`

   The setup process will ordinarily require that you create a set of uninstall floppy disks. This extra step bypasses that requirement
7. At the Welcome screen, press Enter
8. Press Enter to confirm system settings
9. Press `Y` to begin upgrade
10. Reboot when prompted, and confirm you're now running MS-DOS 6.22: `ver`
11. You can safely remove the stepup files and old version of MS-DOS:

    `deltree /y c:\stepup`

    `delolddos`

MS-DOS 6.22 Supplemental Utilities

My copy of MS-DOS includes a fourth disk titled "MS-DOS 6.2 Supplemental Utilities," which can also be obtained from the link above (grab the version for MS-DOS 6.22, named SUP622.EXE). This disk is not mandatory, and mostly just includes some extra features from previous versions of DOS that were dropped for version 6.0. If you don't want to bother with it you won't be missing out on much, but just to be thorough let's take a look at how to install the more interesting stuff on here.

1. Insert the diskette and run `a:\setup.bat c:\dos`
2. Enter `S` to install selected components only
3. Choose `Y` or `N` as each component is listed. The only options I find interesting are the Additional MS-DOS Utilities and MS-DOS Shell, you enter `Y` for these if you feel so inclined.
4. When prompted for display type, enter `F5` for VGA
5. Follow the rest of the prompts to complete the installation

Now, why bother with the Additional MS-DOS utilities? One reason, and one reason alone: QBasic Nibbles! I spent many hours playing this relatively simple but addictive Snake-like game as a kid, and was delighted to discover it on the supplemental disk. To try it yourself, just run `qbasic.exe /run c:\dos\nibbles.bas`.

DOSSHELL, however, is a bit more substantial. It's essentially a primitive Windows-like (or perhaps more accurately, 'Windows Light') environment that runs on top of DOS and provides a GUI file manager, file associations, a menu-driven task list, and even primitive pseudo-multitasking (called Task Swapper here). It's not something I'll care to use, but it's a pretty interesting idea that I was completely unfamiliar with before this project, and it seems like it could be a reasonably powerful interface if a little time is spent on configuring it just right (though, honestly, running Windows 3.x instead will provide far more power and flexibility). I'm not going to dive into this here as it's beyond the scope of "(mostly) pure MS-DOS", but here are a couple tips if you want to play around with this:

> You can run it in either text mode (default, `/t`) or GUI mode (`/g`). Both interfaces are quite similar, but GUI mode can be configured to run at a much higher resolution. Try `dosshell.exe /g:h2`, for example.
>
> The program list in the bottom pane seems like the most interesting part to me. In addition to customizing it through the GUI, you can also edit `dosshell.ini` to tweak it to your heart's content, using the default menu items as an example. Like I said, if you're willing to invest a little time in this, it could probably be tricked out pretty nicely.

Return to top

System Configuration

Relevant Software:

> Enhanced DOSKEY.com - enhanced replacement of the stock DOSKEY utility that adds tab completion to the DOS command line interpreter, in addition to command history support and other useful features.
>
> Pedit - enhanced replacement of the stock EDIT editor, which will be useful for editing the DOS config files
>
> Info-Zip UnZip - open source Zip file extraction utility you can use to unpack archives directly on your DOS system (actually finding the correct file to download is far more difficult than it should be, so here's a direct link); there are many other archiving utilities available for DOS, some with support for a great number of formats, but this is easy to obtain and install and supports by far the most common archive format used for DOS programs

Before installing anything else, we'll configure a basic sane working environment:

1. `cd c:\`
2. `mkdir apps` - contains personal installed applications
3. `mkdir temp` - temporary directory used by some applications
4. `mkdir backup` (if necessary) - contains copy of original setup files or drivers
5. `path c:\apps;c:\dos` - set our installed apps to take precedence over built-in apps

Next, I recommend installing a couple utilities that will make the rest of the setup process much easier. As with the step-up files above, getting the files to the system is still annoying at this point. Until we get networking setup later on, I recommend unpacking the programs on your main computer and copying them over with floppies as the easiest option. I also recommend saving a backup copy of all of your installation media/files and drivers under `c:\backup` (or, preferably, `d:\backup` for easy access to it at any later point.

Software Installation

Enhanced DOSKEY.com

1. Copy `doskey.com` to `c:\apps`
2. Run `c:\apps\doskey.com -i` to activate it. Running `doskey -?` will show some additional information about how to use it.

Pedit

1. Copy `peditlgt.exe` to `c:\apps`

   > *Note:* This is the lightweight version of Pedit. If you wish to have access to a spell checker and thesaurus, copy the larger `pedit.exe` instead.

2. For convenience, `move c:\apps\peditlgt.exe c:\apps\edit.exe`. This will let you launch Pedit by simply typing `edit`, making it effectively replace the built-in DOS editor.
3. Configure Pedit by running `edit`, hitting Esc if prompted to open a file, and entering `Alt-F1`. I like the following non-default settings:

   > Editor Style - 2.0
   >
   > Scroll Bar - dark on light (first pattern after 'none')
   >
   > Text Color - 07
   >
   > Background Color - 01
   >
   > Show Char Under Cursor - enabled
   >
   > Show Insert Mode - enabled
   >
   > Tab Size - 4
   >
   > Tab Expand Size - 4
   >
   > F2 is File-Save - disabled
   >
   > Alt Highlights Menu Choices - enabled
   >
   > Save Changed Settings - enabled
   >
   > Also Save Changed Margins - enabled

4. Hit Esc then `Alt-X` to exit and save the changes

UnZip

1. Copy `unzip.exe` to `c:\apps`
2. That's all that's necessary. `unzip32.exe` is more flexible, but requires a separate DPMI (DOS Protected Mode Interface) memory manager, which won't be covered here. Other included binaries simply provide extra functionality not required for Zip extraction.

Configuration

Now that we have a good editor and input processor installed, it's time to configure DOS itself.

Tip: You will be repeatedly hand editing your system configuration files throughout this walkthrough, and it's very possible at some point that you may be stuck with a system that won't boot properly. To recover, hit `F5` when DOS starts booting (when it says "Starting MS-DOS...", usually immediately after your system beeps to indicate it has completed it's POST process). This will perform a clean boot of the system, ignoring your `autoexec.bat` and `config.sys` files. You can then edit them as appropriate to fix the problem, and reboot to load everything again.

To get started, edit `c:\autoexec.bat` and add/change the following (other settings should be left alone for now):

```
loadhigh c:\dos\smartdrv.exe
loadhigh c:\apps\doskey.com -i
path c:\apps;c:\dos;c:\dos\net
set DIRCMD=/o:gne
set TEMP=c:\temp
```

Note: Most DOS commands and options are not case-sensitive. While most of the text in the stock autoexec.bat file is capitalized, I tend to convert it to lowercase just because I find it easier on the eyes. My personal naming convention, used throughout this document, is pretty simple: variables are fully capitalized, and almost everything else is lowercase. Feel free to leave everything uppercase if that works for you, though.

SmartDrive is "a disk caching program ... that improves data transfer rates by storing frequently accessed data in RAM" (per Wikipedia), and is enabled by default in DOS. Some versions of DOS include the `/x` argument by default, which disabled write-behind caching; if you see this, remove it. Unless you're running on a known bad hard disk or expect to have frequent power failures (in which case, you should really fix those problems instead), `/x` won't be needed and only slows down disk writes.

LOADHIGH will load the specified program (when possible) into upper memory, freeing up the all important conventional memory for other applications. This requires that EMM386 also be enabled, as shown below. The PATH setting specifies `c:\apps` first so that our installed applications always take precedence over the OS programs (allowing Pedit to be easily used instead of EDIT, for example). The `c:\dos\net` directory doesn't exist yet, but will shortly.

Finally, the DIRCMD variable instructs the DIR command to display files sorted first by name, then by extension, with directories always listed first (by default, it lists files sorted by date). I also modified TEMP, pointing it to `c:\temp` to make it more general purpose; ie., anything can write to this directory without worrying about possibly overwriting DOS files.

Next, edit `c:\config.sys` and add/change the following (again, other settings should be left alone for now):

```
SWITCHES=/f
DEVICE=c:\dos\himem.sys /testmem:off
DEVICEHIGH=c:\dos\emm386.exe ram i=b000-b7ff
DOS=high,umb
BREAK=on
rem DEVICEHIGH=c:\dos\setver.exe
```

The `/f` option to SWITCHES instructs DOS to skip a two second waiting period when booting so that your system will boot a little faster. Note that this shortens the amount of time you have to press `F5` if you need to clean boot your system, but it's still possible if you're fast, or you hold down `F5` right before DOS starts loading (which is what I do).

HIMEM.SYS is MS-DOS' extended memory manager, allowing the OS and applications to use more than the first 640 KB of available memory (RAM). The `/testmem:off` option instructs HIMEM to not test your RAM on every boot, saving some time in the boot process.

EMM386 is DOS' expanded memory manager and, which (also) allows access to memory over 1 MB. Although this does roughly the same thing as HIMEM.SYS, EMM386 is required in order to load programs and drivers into upper/high memory (the area between 640 KB and 1 MB), so it's generally a good idea to load both. `ram` instructs EMM386 to allocate RAM as EMS (expanded memory - an older version of upper memory management that was superseded by XMS (extended memory)). `noems` can alternatively be used instead to disable expanded memory emulation in order to free up a little extra memory, but many older applications and games (including some discussed here) will only use EMS, so unless you have a specific need I recommend using the `ram` option to enable EMS. I discuss this a bit more in the Addendum below. The `i=b000-b7ff` option frees up a small amount of additional memory that is ordinarily reserved for monochrome monitors.

The `umb` portion of the DOS option instructions MS-DOS to manage the upper memory blocks created by EMM386 (so LOADHIGH and DEVICEHIGH will work), and `high` causes it to attempt to load part of itself into high memory when possible. DEVICEHIGH, like LOADHIGH, instructs the specific program or driver to load itself into high memory. Finally, BREAK enables the use of Ctrl-C for breaking out of misbehaving programs.

I commented out SETVER as it generally shouldn't be needed. Unless you plan on running particularly old (even by MS-DOS 6.x standards) software or drives, you can leave this disabled to free up some space. If you get any errors about a program not supporting this DOS version, uncomment this line and see Microsoft's Knowledge Base article on Using the SETVER command. You can also refer to Microsoft's list of applications that require SETVER

I recommend rebooting at this point to let the new configuration take effect. Run `mem /c /p` after rebooting to verify that high memory is being utilized; at least some programs should be listed in the Upper Memory column.

Return to top

Network Configuration

Relevant Software:

- Microsoft Network Client 3.0 for MS-DOS - this is pretty self-descriptive; will be used if you want to map shared drives on a Samba or Windows server (local copy: disk 1, 2)
- NDIS Packet Driver Shim 1.11 - this is necessary if you want to use both Microsoft and non-Microsoft networking utilities on the same system; specifically, you'll need dis_pkt.zip
- WATTCP - old TCP/IP stack for DOS; this is actually embedded in the applications using it, so the package here is mostly useful for diagnostic applications and documentation. Get wat2002b.zip.
- mTCP - modern TCP/IP stack for DOS; this works similarly to WATTCP, but supports different/additional applications that are bundled with it (including FTP and NTP clients)
- SSH2DOS - DOS versions of SSHv2 client applications, including ssh, scp, and sftp
- wget - DOS version wget, "the non-interactive network downloaded" (if you're not familiar with it, it makes download files from remote http and ftp servers *very* easy in CLI environments)
- XFS Network File System Client 1.91 - NFS client for DOS; this was a commercial program, but has long since been abandoned so I don't have any qualms about telling you to download it (local copy)

NIC Drivers (you'll need to grab drivers for your specific NIC; these drivers cover my system, and will be used for the walkthrough):

- 3Com EtherLink 10/100 PCI 3C905C - 3Com NIC drivers; 3c90x2 (EtherCD Disk 2) from the link contains the DOS drivers (local copy)
- 3Com EtherLink 10 ISA 3C509B - 3Com NIC driver; this can no longer be obtained from any official site, so the FTP mirror linked here is the best current option (I've verified against multiple sources that it's a copy of the original EtherDisk 6.1 release). 3c509x2.exe (EtherDisk Disk 2) contains the DOS drivers (local copy). Crynwr provides an alternative packet driver that's half the size (3 KB vs. 6 KB) and seems to perform just as well as the official driver (grab 3c509116.zip). If

you're only interested in packet driver applications, this is a nice option.

Kingston EtheRx KNE20T - Kingston NIC driver; this can no longer be obtained from any official site, so the website linked here is the best current option (I've verified against multiple sources that it's a copy of the original QStart 1.2 release) (local copy).

Network Options

Now it's time to (finally) configure networking. Getting this up and running is extremely helpful because it largely eliminates the need for floppy disks. Unfortunately, networking under DOS is... primitive. It can be done, but it's somewhat limited, slow, and painful to install due to multiple driver and network protocols. So, get ready for some fun!

The first decision to make is to decide what kind of networking support you want installed. In brief, there are two types of drivers we're going to install:

NDIS - the newer device driver interface co-developed by Microsoft, with somewhat more features and capabilities at the expense of more bugs and significantly more memory usage. This is required for Microsoft networking utilities, including support for drive mapping.

Packet Driver (PD) - the older, open device driver interface used by pretty much everything before NDIS. Most non-Microsoft network utilities require a packet driver interface and will not work with only NDIS installed.

Fortunately, this isn't a one-or-the-other choice; it's possible to install support for both with some extra work. If you only need one or the other, though, save yourself the hassle and go with that one. I'm going to walk through installing both below, and point out what you should do differently if you only want to install one of the drivers.

Note: Installation order differs depending on whether you want only PD support, only NDIS support, or both. Also, while the PD option may sound more limited than NDIS, it's really not. You can get it setup much faster, it's more efficient, generally more reliable, and much more widely supported, and among the various utilities that use it are DOS versions of SSH, SCP, SFTP, and even an NFS client. The SSH utilities will let you easily and securely copy files from another system without messing with NDIS or enabling DOS support in Samba (which is not enabled by default), and NFS will let you mount an NFS share as a DOS drive letter, very similar to mapped drives. For the full experience it's worth setting up NDIS (it can be cleanly disabled if you decide it's not worth it), but if you just want to get up and running as quickly is possible, the PD-only option should suffice, and is all I use on my system after experimenting with all of this.

NIC Drivers

Both NDIS and (native) PD require hardware drivers for your NIC. In my case, I'm using a 3Com 3C905C, which is a Plug and Play PCI NIC with a relatively easy configuration. If you're using an ISA NIC, there may be more steps involved for specifying the number, I/O address, etc. Download the appropriate driver package for your NIC and consult the documentation for details.

Edit: Due to various circumstances I've also setup a Kingston KNE20T and a 3Com 3C509B in this system, so instructions for all three NICs are included below.

The following files are required for DOS networking for each of the listed NICs:

- 3Com 3C905C
  - ndis2\dos\el90x.dos - NDIS2 driver
  - ndis2\dos\oemsetup.inf - NDIS2 configuration file for Microsoft Network Client
  - pktdvr\3c90xpd.com - Packet Driver
- 3Com 3C509B
  - ndis2\dos\elnk3.dos - NDIS2 driver
  - ndis2\dos\oemsetup.inf - NDIS2 configuration file for Microsoft Network Client
  - pktdvr\3c5x9pd.com - Packet Driver
  - 3c509.com - Alternative Crynwr Packet Driver; recommended
- Kingston KNE20T
  - ndis2\ktc20.dos - NDIS2 driver
  - ndis2\lanservr.dos\oemsetup.inf - NDIS2 configuration file for Microsoft Network Client

pktdvr\ktc20pkt.com - Packet Driver

Note: For the Kingston KNE20T, the listed files are included in `pnpdata1.exe`. This file can be manually unpacked, but it's probably easiest to use QStart to unpack them for you:

1. Run `qstart.exe`
2. Set the appropriate Plug and Play mode (this will depend on what your computer's BIOS can support) and click choose Continue
3. Select Custom
4. Use one of the Config. options if you'd like to verify your configuration, then click Continue
5. Select Basic Test if, again, you want to verify your configuration, then choose Driver Installation when ready
6. Select "IBM LAN Server ver. 4.0 DOS LAN Services / NDIS 2" or "Packet Driver", choose a reasonable Destination Directory, and choose Copy
7. Choose Exit to complete driver extraction. If you made configuration changes (especially for the PnP mode), you may want to restart to ensure those changes take effect.

Packet Driver Only

If you **only** want to install the PD interface, copy the packet driver to `c:\dos`. Edit `c:\autoexec.bat` and add the following (note: the `/I` for the 3C905C is case sensitive):

3Com 3C905C
```
loadhigh c:\dos\3c90xpd.com /I=0x60
```
3Com 3C509B
Official:
```
loadhigh c:\dos\3c5x9pd.com 0x60
```
Alternative:
```
loadhigh c:\dos\3c509.com -p 0x60
```
Kingston KNE20T
```
loadhigh c:\dos\ktc20pkt.com 0x60
```

The `0x60` option for each specifies the software interrupt; 0x60 is the default, and there shouldn't be a reason to change it unless you're using multiple NICs. This is optional on some cards (such as the 3C905C), but it doesn't hurt to be explicit, and it's required if you want to unload the driver later (with `/u`, where supported).

Run the above command now to manually load the driver; assuming you have an ethernet cable plugged in, it should automatically detect the connection. If all you need is a PD interface, skip to the Packet Driver Configuration section.

NDIS Driver

NDIS driver installation is quite a bit more involved. It can be installed manually like the PD, but instead we're going to let installation and configuration be handled by Microsoft Network Client 3.0 for MS-DOS (MS-Client). MS-Client, as the name implies, is a multi-protocol stack for MS-DOS, requiring the use of an NDIS driver to talk to network hardware.

To prepare for setup, copy both MS-Client disks to your hard drive (as noted above, I recommend keeping a copy in `c:\backup` as well - last time I'll mention this). Running each EXE file will unpack them to the current directory if you didn't unpack them before hand. If prompted, it's safe to overwrite any files as a few exist on both disks (assuming you copied the MS-Client disks to their own directory).; Copy your NIC's NDIS driver and configuration file (noted above) to your hard drive as well; place both in the same directory (if necessary) and note the location.

Note: If you are using a 3c90x card, you must edit `oemsetup.inf` before proceeding with MS-Client installation. Comment out (or delete) the following lines and save the file:

```
ndis3=1:el90x.386
mlid=1:3c90x.com
```

To begin MS-Client installation, run `setup.exe` from Disk 1.

1. Press Enter to skip the Welcome screen
2. Specify `c:\dos\net` for the installation directory (since these are DOS-specific drivers)
3. For network adapter selection, you can choose the driver that's appropriate for your hardware if listed. However, you can almost certainly find a newer driver than what's included with MS-Client, so I recommend choosing "Network adapter not shown on the list below" to use use your downloaded driver instead.
4. Enter the directory path containing your NDIS2 driver configuration file
5. Confirm the chosen driver when prompted
6. Unless your system is extremely short on RAM, I recommend letting MS-Client optimize itself for performance, so press Enter to continue.
7. Enter the username you will use to login to this computer (this is primarily for authenticating against server shares to map drives)
8. Choose Change Names
9. Edit your hostname, workgroup, and domain name as appropriate.
10. Select "The listed names are correct" to return
11. Choose Change Setup Options
12. Select Change Redir Options
13. I recommend using the Basic Redirector if possible to save memory. The only reason you should need to use the Full Redirector is if you plan on authenticating against an NT domain controller.
14. I recommend leaving the Start Options set to "Run Network Client". The Load Pop-up option is only useful if you plan on communicating with other LAN systems using the old netbios messenger service (aka NET SEND, WinPopup, etc.). You can also choose to not run the network client automatically, but we can disable MS-Client from autostarting at any time after installation, so it's easier to let MS-Client configure the startup options itself.
15. Logon Validation and Net Pop Hot Key should be left alone. They can be changed later if desired.
16. Select "The listed options are correct" to return
17. Choose Change Network Configuration
18. This option window is a bit difficult to work with. Press tab to switch to the top pane to choose the adapter/protocol you want to modify, then tab back down to the bottom pane and choose the appropriate action. Select your NIC and choose Change Settings first to review and update any settings as necessary. You generally should not need to modify this unless you have more than one NIC.
19. Select NWLink IPX Compatible Transport and choose Remove - IPX is (generally) no longer used today.
20. When prompted for a new protocol, choose Microsoft TCP/IP - this is what's in common use today
21. Select Microsoft TCP/IP and choose Change Settings
22. If necessary, IP address and network settings can be specified here. By default it's configured for DHCP, so if you're on a DHCP network you can skip this step and return to to Network Configuration.
23. If you plan to communicate with other ancient DOS or Windows systems (pre-Windows 95), you might want to also install and configure Microsoft NetBEUI. This generally shouldn't be needed, though. All reasonably modern server operating systems, including Samba, support NetBIOS over TCP/IP (NBT). NetBEUI itself is not required for NetBIOS communication.
24. When finished, select "Network configuration is correct" to return
25. Return to main configuration menu, and choose "The listed options are correct" to continue installation
26. When prompted for the OEM Driver Disk, specify the path to MS-Client Disk 2, or just press Enter if all files were copied to a common directory
27. Press F3 to exit setup without rebooting

A bug in the MS-Client installer prevents a file necessary for Windows 3.1x support from being installed. This isn't necessary if running solely under DOS, but it doesn't hurt in anyway, so let's install it just to be safe. To do so, from the msclient setup directory run: `expand disk2\wsahdapp.ex_ c:\dos\net\wsahdapp.exe`

If you run into any other trouble, or have questions about some of the options, official MS-Client setup documentation is still available from Microsoft.

MS-Client updated your system configuration files during setup, so let's review those changes and make a couple changes. First, edit `config.sys` and change the following:

```
DEVICEHIGH=c:\dos\net\ifshlp.sys
```

IFSHLP is the installable file system helper, and provides access to file and network APIs used by the MS-Client utilities.

Next, edit autoexec.bat and change:

```
loadhigh c:\dos\net\tcptsr.exe
loadhigh c:\dos\net\tinyrfc.exe
loadhigh c:\dos\net\emsbfr.exe
```

These are some of the various network services installed by MS-Client. We're requesting that they load themselves into high memory. Note that the lines containing the net command, *.com, and nmtsr.exe are NOT set to load in high memory; this is necessary, as these services must be run from conventional memory.

Note: If you get the error "Insufficient memory to load Tiny RFC 1.0" on boot (or a similar error from one of the other services), remove `loadhigh` from the front of that line as well. This indicates you've run out of upper memory, and MS-Client isn't smart enough to load itself into conventional memory instead.

Reboot load the network drivers.

After rebooting, MS-Client should initialize your network card, grab an IP address from DHCP (if enabled), then prompt you to enter a username. This is only required if you plan on mapping shared drives, and can be disabled if you do not wish to auto-mount drives at login. If you **do** plan to mount shared drives, go ahead and enter your username now (or just press Enter to accept what you specified during MS-Client setup), then enter the password for your account. If you **do not** plan to mount shared drives, press Enter, then press Enter again to setup a NULL password. In either case, choose Y to create a password list when prompted.

At this point, Microsoft's TCP/IP stack should be (mostly) up and running, using the NDIS driver. We can perform a few tests to confirm, but note that the utilities described below, though named the same as modern utilities, work differently than what you're probably used to (again, think "primitive").

First, verify that you have an IP address:

```
ipconfig c:\dos\net
```

This should have been provided by DHCP, or should match the manual one you specified during MS-Client setup. Next, make sure you can ping your own IP, eg.:

```
ping 192.168.0.15
```

The bottom line should say "echo received". Finally, make sure you can ping your gateway's IP address (this can be found in the ipconfig output):

```
ping 192.168.0.1
```

Again, the bottom line should say "echo received". If all that worked, congrats, you're up and running!

However, there's one more major issue we need to take care of. Due to another bug in MS-Client, DNS does not work by default, nor is it possible to configure or enable it through the setup utility; this must be done manually.

To enable DNS, first edit `c:\dos\net\tcputils.ini` and append the following:

```
[dnr]
drivername=DNR$
bindings=TCPIP_XIF
```

If you are **not** using DHCP, also add the following below the bindings line:

```
nameserver0=aaa bbb ccc ddd
nameserver1=aaa bbb ccc eee
domain=domain.com
```

Substitute the appropriate values for your network, and be sure to separate the octets with spaces (as shown above) rather than periods as would be done on modern systems.

Next, edit `autoexec.bat` and add the following line immediately above "net start":

```
loadhigh c:\dos\net\dnr.exe
```

Finally, run the above command now to start the Microsoft domain name resolver. The implements DNS functionality in MS-Client. You can verify DNS is working properly with ping again:

```
ping www.google.com
```

As before, you should get an "echo received", along with the IP address for www.google.com.

All of this work will now let us (finally) do something very important: map shared network drives. This is especially useful as we can use this to easily copy files and install new applications and drivers without floppy disks.

To map a drive, assuming you already have a shared drive properly configured in Samba or a Windows server (which is outside the scope of this HOWTO), run:

```
net use e: \\servername\sharename
```

Again, assuming everything has been properly setup on the server, and that you logged in with proper credentials, this should just work; you'll get a "command completed successfully" message if it did. To test, try running `dir e:` and verify that you get output. If it doesn't as expected, and you followed all instructions above, the error is most likely on the server side. As noted, modern versions of both Samba and Windows no longer support DOS or Windows 3.x clients by default, and have to be run in a more insecure mode to make this work. I've had mixed results myself - on my initial testing with MS-Client (about three years ago), I was able to get this working perfectly with Samba; this time, however, I can't. If you have trouble with this, don't worry - SCP, SFTP, and NFS all work perfectly well for copying files using the PD interface, and are described in detail below.

NDIS Packet Driver Shim

The final bit of network setup we're going to do for the NDIS driver is installing a "shim" packet driver. The shim driver basically adds a PD interface to the NDIS driver. This is necessary if you want to use both Microsoft (including mapped drives) and non-Microsoft network utilities on the system (which is highly recommended).

Note that this shim driver is meant specifically for this purpose; it cannot drive your NIC directly without the NDIS driver, nor can your NIC's packet driver be used as a shim. Also, while the driver itself works great, the whole concept is a hack and as a result requires a manual and rather complicated setup process.

To begin, copy `dis_pkt.dos` to `c:\dos\net`. Next, edit `c:\dos\net\protocol.ini` and append the following:

```
[pktdrv]
DriverName=PKTDRV$
BINDINGS=TCM$EL90X
INTVEC=0x60
CHAINVEC=0x68
```

The BINDINGS line needs to be modified for your hardware; everything else (should) be able to stay the same. The BINDINGS line informs the packet driver of which NDIS driver it must bind to. To find this, search for the [TCPIP] section in `protocol.ini`; it should also contain a BINDINGS line. Use the same driver name for the BINDINGS line under [pktdrv].

Save your changes to `protocol.ini` and edit `config.sys` next. Add the following after "ifshlp.sys":

```
DEVICEHIGH=c:\dos\net\protman.dos /i:c:\dos\net
DEVICEHIGH=c:\dos\net\nemm.dos
DEVICEHIGH=c:\dos\net\tcpdrv.dos
DEVICEHIGH=c:\dos\net\el90x.dos
DEVICEHIGH=c:\dos\net\dis_pkt.dos
```

Although it looks like you're adding a lot here, you're only going to be loading one new driver; the `dis_pkt.dos` driver you just copied over. Everything else was already being loaded automatically by MS-Client, but due to some dependency issues you need to load them before dis_pkt, which in turn needs to be loaded before MS-Client.

Two additional notes:

1. The el90x.dos line refers to the NDIS driver for my hardware, a 3com 3C905C. You must change this to point to the NDIS driver for your hardware.
2. Order is important: be sure to leave the lines in the order I have listed above.

We also need to edit autoexec.bat and change the following:

```
rem c:\dos\net\net initialize
```

`net initialize` loads all of the required drivers for MS-Client, which you're now doing manually through `config.sys`. As a result, this command is no longer needed, and will in fact cause errors if you try to run it with the other drivers already loaded.

This is also a good time to decide whether or not you want to automatically map shared drives on boot. If you do not plan on using shared drives, or you prefer to map them manually each time they're needed (which can be done by simply running `net start`), comment out the following line to save a fair amount of conventional memory:

```
rem c:\dos\net\net start
```

Finally, reboot to load the new driver. You should see the following message during the boot process to confirm the shim driver was loaded:

```
MAC/DIS to Packet Driver convert loaded.
```

Packet Driver Configuration

Whether you installed the native PD or the NDIS PD shim, you won't be able to test network functionality using currently available utilities. The PD only provides an interface to your network card; each application using this interface must provide it's own network stack to talk TCP/IP, provide an IP address, etc. Most older applications do this by using the Waterloo TCP/IP stack (WATTCP), which can be embedded into the application and provides all necessary network functionality. A much newer TCP/IP stack called mTCP works similarly, and includes some additional utilities not (well) supported by WATTCP. Fortunately, both stacks can installed and enabled simultaneously, so we'll setup both.

WATTCP

To configure WATTCP, create `c:\apps\wattcp.cfg` and add the following:

```
print = "Configuring WATTCP..."
hostname = <hostname>
my_ip = dhcp
```

If you use a static IP, remove the dhcp line above and add the following:

```
domain.suffix = localdomain
my_ip = 0.0.0.0
netmask = 255.255.255.0
nameserver = 0.0.0.0
nameserver = 0.0.0.0
gateway = 0.0.0.0
```

Edit the values as appropriate, then save and close the file. Finally, add the following line to `autoexec.bat`:

```
set WATTCP.CFG=c:\apps
```

Run the above command now to manually set the configuration variable. Next, let's install a couple diagnostic utilities. Copy `tcpinfo.exe` and `ping.exe` from the WATTCP archive to `c:\apps` (I suggest renaming ping to something like `pingw.exe` to avoid conflicting with the Microsoft or mTCP version of ping).

To test our WATTCP configuration run `tcpinfo.exe`. The top line should state, "Reading Waterloo

TCP configuration file," which indicates that it was able to find your config file. The network parameters output by tcpinfo should either match your static configuration, or should be appropriate for your DHCP network. To test, try the following:

```
pingw www.google.com
```

If you see "Replies lost: 0", everything's working.

mTCP

mTCP works, and is configured, very similarly to WATTCP. You'll need to create another config file, `c:\apps\mtcp.cfg`, and add:

```
packetint 0x60
hostname <hostname>
```

If you use a static IP, also add:

```
ipaddr 0.0.0.0
netmask 255.255.255.0
gateway 0.0.0.0
nameserver 0.0.0.0
mtu 1500
```

Edit the values as appropriate, then save and close the file. Finally, add the following to `autoexec.bat`:

```
set MTCPCFG=c:\apps\mtcp.cfg
c:\apps\dhcp.exe
```

The only practical difference between mTCP and WATTCP (from and end-user's perspective) is that mTCP provides a separate DHCP client, whereas WATTCP applications do this themselves. If you are using a static address you can skip the dhcp line in `autoexec.bat`. Run those `autoexec.bat` line(s) now to initialize mTCP.

To test, copy `ping.exe` to `c:\apps` (again I recommend renaming it to something like `pingm.exe` to avoid conflicts) and run:

```
pingm www.google.com
```

If you see a "Success: 100 %" message, everything's working.

Network Applications

It's time install a few essential network applications. Quite a few exist, but these are the ones I find most useful.

SSH

I primarily run Linux systems, so SSH is very important to me. As a bonus, SCP and SFTP provide great options for transferring files across the network, and is especially useful when NDIS/MS-Client is not setup or mapped drives do not work properly with modern Samba or Windows servers. Fortunately, there's an excellent SSHv2 implementation for DOS called, simply enough, SSH2DOS. It can be downloaded from the link above.

Installation is pretty simple; just copy and rename the following files as suggested:

```
copy scp2d386.exe c:\apps\scp.exe
copy sftpd386.exe c:\apps\sftp.exe
copy ssh2d386.exe c:\apps\ssh.exe
```

SSH2DOS is a WATTCP-based application, so it requires a PD interface and a working `wattcp.cfg` file. If you setup both of these as instructed earlier, you're already good to go. Run `ssh username servername` to verify SSH works as well.

SSH2DOS also includes a thoroughly commented example WATTCP configuration file. If you're having any trouble with your WATTCP apps, or you want to poke around to see what other options are available, take a look at the included `wattcp.cfg`. If you decide to edit and use this version of the file, you can either replace the existing `wattcp.cfg` in `c:\dos`, or point the `WATTTCP.CFG` variable in `autoexec.bat` to the new location (but don't forget to update that variable as well, or reboot, or your apps will continue to use the old configuration file).

wget

wget is another useful utility. This is a command line download manager that can download files from pretty much any web or FTP server and, happily, there's a DOS version available from the link above (though the DOS version is no longer maintained at this point).

Installation is the same as SSH2DOS; `copy bin\wget.exe c:\apps` and you're done. This is another WATTCP-based application, so it should just work at this point.

NTP, FTP, TELNET

Now let's install some mTCP applications. These are all bundled with the mTCP distribution, so just copy the following files to `c:\apps`: `ftp.exe`, `sntp.exe`, `telnet.exe`

Each of these will utilize your mTCP configuration file so you can go ahead and use them directly. FTP and TELNET should be fairly self-explanatory and work how you'd expect, with FTP being especially useful if you run Windows on your main machine and don't have SSH available. The utility I'm most interested in here, though, is SNTP, a simple Network Time Protocol client. You can use this to automatically set your computer's clock to the correct current time, which is *very* handy on ancient computer hardware with bad CMOS batteries. :-) To test this, run the following:

```
set TZ=CST6CDT
sntp.exe pool.ntp.org
```

Note: TZ stands for time zone, and is required so that SNTP knows the correct 'local' time. CST6CDT is the code for the Central Standard Time zone; see `sntp.txt` for additional information about how to set this.

After running the above, you should see your current system time as well as the current NTP server time (listed as "Time should be set to"). To actually set the time, use the `-set` parameter. To always set your system clock to the appropriate time on boot (strongly recommended), add the above, with `-set` to `autoexec.bat`.

Bonus tip: once you get the mTCP utilities installed, try running `telnet towel.blinkenlights.nl` for an unexpected treat. :-)

Note: There are several additional applications/utilities included with both WATTCP and mTCP, but the above are the ones I personally find most useful. Feel free to play around with the others, though.

XFS (NFS client)

If you run an NFS server instead of (or in addition to) an SMB/CIFS server, you'll be happy to know that it is possible to mount an NFS share from DOS. The bad news is that you lose the ability to concurrently use many (but apparently not all) other PD-based applications. The reason is that XFS installs it's own driver literally on top of your packet driver, which conflicts with most other PD applications. wget (for whatever reason) still works with the XFS driver loaded, but none of the other WATTCP- nor mTCP-based utilities can use it.

If you **only** otherwise use NDIS-based applications, this is not an issue. If you **only** use PD-based applications, theoretically you should be able to get them to work with the XFS driver loaded by configuring them to use interrupt vector 0x62 instead of 0x60 (search for "redirected PKTDRVR" in `xfs.txt`), but I could not get that to work. See the end of this section for an alternative approach that you can use instead. If you need to use both NDIS- and PD-based applications, you're pretty much out of luck; you need to choose one or the other at this point (or, of course, not use NFS).

So here's the deal with NFS under DOS: there's an old version of NFS for DOS called XFS Network File System Client. It was a commercial program, but as noted above it's long since been abandoned, so grab a copy if you're so inclined and let's get to work.

Note: Before installing, take a look at `kernels.txt`, which describes the different kernel drivers available. I use the minimal kernel, XFSKRNLM, because that meets my needs and takes up the least amount of memory, but your needs may vary.

Installation is a bit tricky:

1. Copy the following files to to `c:\apps`:

```
xfskrnlm.exe
hosts
ls.exe
xfstool.exe
```

The rest of the executables can be useful for troubleshooting NFS connections, but shouldn't be required for routine use. I also recommend renaming `ls.exe` to `lsx.exe` to avoid conflicting with the 4DOS alias for `ls` (defined below).

2. Edit hosts, comment out everything currently in there, then define your NFS server as:

```
aaa.bbb.ccc.ddd nfsserver.domain.com    nfsserver
```

3. If you're using a static IP address, add the following to `hosts`, where `nfsclient` is the name of your DOS system and the IP information applies to your network configuration:

```
aaa.bbb.ccc.eee gateway
aaa.bbb.ccc.ggg netmask
aaa.bbb.ccc.fff broadcast
aaa.bbb.ccc.ggg nfsclient.domain.com    nfsclient
```

4. If you're using NDIS, copy `dis_pkt9.tcp` to `c:\dos\net` and edit `config.sys`. Add/change the following:

```
rem DEVICEHIGH=c:\dos\net\dis_pkt.dos
DEVICEHIGH=c:\dos\net\dis_pkt9.tcp
```

Note that this disables the old packet driver and loads the new one provided by XFS. Reboot here if using NDIS before continuing.

5. Run the following command to initialize the kernel:

```
loadhigh c:\apps\xfskrnlm.exe 0x60
```

6. Run the following to initialize the driver:

If using BOOTP (a predecessor to DHCP; many DHCP servers can also support BOOTP, but may require special configuration):

```
xfstool init bootp
```

or, if using a static address (`nfsclient` must be defined as described above):

```
xfstool init nfsclient
```

7. Finally, try to mount a share:

```
xfstool mount f: nfsserver:/home/data/files
```

Note that XFS **requires** a local hostname to be set. If you're using a static IP address, this should be taken care of. If you're using BOOTP, your DHCP server has to be configured to supply a hostname in addition to an IP address to your system. If XFSTOOL complains about a hostname not being set, this is probably what's happening. The easiest workaround would is to configure a static IP address for XFS, as described above.

If everything worked, you should be able to do run `dir f:` and see a list of files on your NFS server. Switch to F: and run `ls` (or `lsx`) to see the long file names. If using NDIS, I also recommend testing an NDIS application to ensure the shim driver is working properly. `ping hostname` should do the trick, but a more thorough test would be to map a shared drive, eg., with `net start`.

If you want to automount any NFS shares on boot, add the three XFSKRNL, INIT, and MOUNT commands shown above to `autoexec.bat`. Be sure it's added *after* any other PD applications included in `autoexec.bat`, such as mTCP's DHCP client, or the other applications will fail for the reasons listed above.

As mentioned previously, it's not possible to (reliably) use XFS with other PD applications. However, XFS provides the ability to temporarily unload and then reload it's PD shim as needed, so you can leverage this to run other PD applications without completely tearing down NFS support or rebooting. As an example of how to do this, say you want to SSH to another system while you have an NFS share mounted; you can do this by calling the following additional commands before and after SSH:

```
xfstool pktdrv stop
ssh username hostname
xfstool pktdrv restart
```

While in the "stopped" state, your mounted shares will still exist but will not be accessible. Issuing the restart will make them available again. It's not ideal, but it's a fairly reasonable compromise, and can be made less annoying by using a 4DOS alias to simplify that much more (see the 4DOS section below).

Return to top

Device Configuration

Relevant Software and Drivers:

> Toshiba ATAPI CD/DVD-ROM Driver - generic ATAPI E-IDE CD-ROM driver for DOS; unfortunately, this is packed in a self-extracting Windows binary using the LHA format, so you'll have to extract the files on your main system before copying it over to DOS (local copy)
>
> CuteMouse - modern mouse driver for DOS
>
> Creative Labs Sound Blaster AWE64 - Sound Blaster AWE64 sound card drivers; you'll need (at least) "Sound Blaster 16/SB32/AWE32 Basic Disk for DOS/Windows 3.1 Installation" and "Creative PnP Configuration Manager (Rev 4)"

Whew, now that we're finally finished with networking, everything else should be a breeze in comparison. :-) Since we now have an easy way to transfer files to our DOS system, it's time to install our remaining drivers and applications. Let's start with the CD-ROM drive.

CD-ROM Driver

Microsoft does not include a CD-ROM driver by default with DOS, and it's difficult (if not impossible) to get a DOS CD-ROM driver from manufacturers today. Fortunately, though, a few generic CD-ROM drivers exist that should cover most standard IDE/ATAPI CD/DVD-ROM drives.

The most widely compatible and widely used MS-DOS CD-ROM drivers are probably from Oak Technologies and Gold Star, both available from the Computer Hope hardware downloads page. Unfortunately, both are memory hogs; the Gold Star driver consumes 25 KB, while the Oak driver consumes a whopping 35 KB. As an alternative I recommend the Toshiba driver linked above. This driver should offer roughly the same compatibility and capabilities, but only uses a svelte 7 KB of memory.

Acer's ATAPI CD-ROM driver (search for VIDE-CDD.SYS v2.15 on the linked page) is another option that uses only 5 KB, as is the modern UDVD2 driver for the FreeDOS project which uses, I think, less than 1 KB. Unfortunately, the UDVD2 driver doesn't behave well on a physical MS-DOS system, and I had some reliability issues with VIDE-CDD.SYS while installing and testing some classic games, so I'd recommend sticking with Toshiba's CDROMDRV.SYS driver.

To install the driver, copy `cdromdrv.sys` to `c:\dos`. Then, edit `config.sys` and add:

```
DEVICEHIGH=c:\dos\cdromdrv.sys /d:mscd001
```

The `/d` option sets a device name to be used by MSCDEX; you can set anything you want here, but mscd001 is the standard name and there's no real benefit to changing it. Next, edit `autoexec.bat` and add:

```
loadhigh c:\dos\mscdex.exe /d:mscd001
```

MSCDEX acts as an interface for DOS to communicate with the CD-ROM driver and, in turn, the device itself. Reboot to load the new driver. If both the driver and MSCDEX were loaded properly, you'll see the following message in the output while the system is booting:

```
Drive D: = Driver MSCD001 unit 0
```

Try inserting a CD-ROM and entering `dir d:` to verify MSCDEX was able to see it as well.

Mouse Driver

Next we'll install a mouse driver. This is optional, as DOS itself doesn't utilize mice, only the applications programmed to support them (such as Pedit). So, unless you plan on running an application or game that uses a mouse, you can skip this.

As with CD-ROM support, MS-DOS doesn't include a mouse driver by default. And again like CD-ROM support, we have a couple options to choose from. The first option is to use the original mouse driver provided by Microsoft with Windows 3.x. This is called MOUSE.COM, and can again be downloaded from the Computer Hope hardware downloads page. The second option is a much newer, open source mouse driver called CuteMouse. It's still actively developed for the FreeDOS project and available from the site linked above.

CuteMouse provides support for modern mice and mouse features such as wheels (though few applications support the wheel), and is significantly smaller than MOUSE.COM, using only about 3.5 KB of memory vs. almost 18 KB for MOUSE.COM. I strongly recommend CuteMouse here, and will use it for the example, but you're welcome to use MOUSE.COM if preferred.

To install CuteMouse, copy `bin\ctmouse.exe` to `c:\dos`. Then, edit `autoexec.bat` and add:

```
c:\dos\ctmouse.exe /3
```

CuteMouse supports quite a few options, so run `ctmouse.exe /?` to get a complete list and set the options appropriate for your hardware. Auto-detection should generally work fine, though. In my case, I use `/3` to force 3-button mode, which isn't enabled by default. Also, note that we're not specifying LOADHIGH here; CuteMouse is smart enough to load itself into upper memory when available, so LOADHIGH isn't needed.

Run the above command to manually load the driver. Pedit supports mice, so you can fire that up to verify that your mouse was properly detected and enabled, or use the included `mousetst.com`.

Sound Card Driver

Next up is sound. You'll need to install drivers appropriate for your sound card. I have a Creative Labs Sound Blaster AWE64, so that's what I'll setup in the example. Your setup should be at least somewhat similar, but certainly some of the details will be different.

To begin, unpack and copy over both the driver disk and configuration manager (linked above), then change to the driver disk directory.

1. Run `install.exe`
2. Press Enter to continue
3. If prompted to install the Creative Configuration Manager (CTCM), press enter to continue, then enter the path to the CTCM files. This is needed for Plug and Play (PnP) cards under DOS. The Intel PnP ISA Configuration Manager (ICM) can be used for this instead, but my experience has been that ICM is much harder to find and really not worth the trouble. CTCM is pretty easy and worked on the first try.
4. If installing CTCM, I recommend setting the path to `c:\apps\ctcm`. Verify the suggested settings are sensible and hit Enter to continue.
5. Hit Enter if prompted to run CTCM, then again when asked about modifying your system config files, then a third time to continue installation
6. For the drivers, I recommend installing to `c:\apps\awe64`. Again, verify the suggested settings are sensible and hit Enter to continue.
7. Make note of the Audio Device settings (you can review this later, if necessary) and hit Enter
8. Hit Enter to make the config file changes, then Enter again to complete installation

We'll need to reboot to activate the sound card, so do that now. After rebooting, change to `c:\apps\awe64` and run `diagnose.exe`. This will let you verify that your sound card is installed and working properly. While here, you can also run `mixerset.exe` to adjust the volume levels as necessary. I find the default config too loud (leading to noticeable distortion on my speakers), so I turn the Master volume down a couple clicks. You may also like to try enabling 3DSE (3D Stereo Enhancement) and see if that improves the sound; I generally don't care for this, but it can make a positive difference with cheap stereo or embedded monitor speakers.

Return to top

System Optimization

At this point it's time to do some configuration cleanup and memory optimization. Remember, you can use `mem /c /p` to view current memory usage and availability. My system, after installing everything listed above and despite tweaking the config files a bit, currently has only 497 KB of conventional memory available. This is actually pretty decent given everything I have loaded, but unfortunately it's not enough for some of the other programs I want to install, and definitely isn't large enough for many games. Generally, you should shoot for >530 KB free, which should cover most (though not all - 587 KB is the highest I've personally encountered) DOS applications and games.

For reference, here's my `config.sys` file at this point:

```
REM configure boot options
SWITCHES=/f

REM enable memory management
DEVICE=c:\dos\himem.sys /testmem:off
DEVICEHIGH=c:\dos\emm386.exe ram 24576 highscan notr i=b000-b7ff
DOS=high,umb

REM load device drivers
DEVICEHIGH=c:\dos\cdromdrv.sys /d:mscd001
DEVICEHIGH=c:\dos\net\ifshlp.sys
DEVICEHIGH=c:\dos\net\protman.dos /i:c:\dos\net
DEVICEHIGH=c:\dos\net\nemm.dos
DEVICEHIGH=c:\dos\net\tcpdrv.dos
DEVICEHIGH=c:\dos\net\elnk3.dos
DEVICEHIGH=c:\dos\net\dis_pkt.dos
DEVICEHIGH=c:\dos\ansi.sys
DEVICEHIGH=c:\dos\power adv:min
DEVICE=c:\apps\ctcm\ctcm.exe
rem DEVICEHIGH=c:\dos\setver.exe

REM configure environment
SHELL=c:\dos\command.com c:\dos /p
FILES=30
BUFFERS=5,0
FCBS=1
STACKS=0,0
LASTDRIVE=H
BREAK=on
```

And my `autoexec.bat` file:

```
@echo off

REM setup display and drivers
c:\dos\mode.com con cols=80 lines=50
loadhigh c:\dos\mscdex.exe /d:mscd001
loadhigh c:\dos\smartdrv.exe
loadhigh c:\apps\doskey.com -i
c:\dos\ctmouse.exe /3

REM setup network
rem loadhigh c:\dos\3c5x9pd.com 0x60
rem c:\dos\net\net.exe initialize
c:\dos\net\netbind.com
c:\dos\net\umb.com
loadhigh c:\dos\net\tcptsr.exe
loadhigh c:\dos\net\tinyrfc.exe
c:\dos\net\nmtsr.exe
loadhigh c:\dos\net\emsbfr.exe
loadhigh c:\dos\net\dnr.exe
c:\dos\net\net start

REM setup sound card
set SOUND=c:\apps\awe64
set BLASTER=A220 I5 D1 H5 P330 E620 T6
set MIDI=SYNTH:1 MAP:E MODE:0
set CTCM=c:\apps\awe64\ctcm
c:\apps\awe64\diagnose.exe /s
c:\apps\awe64\aweutil.com /s
c:\apps\awe64\mixerset.exe /p /q
c:\apps\ctcm\ctcm.exe /s

REM setup environment
path c:\apps;c:\dos;c:\dos\net
prompt $e[1;34m$p$g $e[0;47;0m
set DIRCMD=/o:gne
set TEMP=c:\temp
set WATTCP.CFG=c:\dos\net

echo.
```

Most of this has been covered previously, but there are a few new items:

emm386.exe

`24576` instructs EMM386 to only make 24 MB of extended memory (out of the 64 MB of RAM available on this system) available as EMS. By default, it'll use all available extended memory, up to 32 MB. I added this mostly for troubleshooting: if a game reports that it sees 24 MB of memory, then that means it's using EMS and not XMS. `highscan` enables more aggressive attempts to find available upper memory (and this increase available capacity), though it can cause some systems to hang. `notr` disables searching for token ring adapters, a legacy networking topology no longer in use today.

ansi.sys
: DOS driver that enables the use of graphics and special characters in programs that support them. This isn't needed unless a specific program requires it, but in my case I'm using it to increase the number of rows displayed on screen and enable my custom shell prompt (see below).

power.exe
: POWER enables basic power management support. This isn't especially necessary, but enabling it makes the my BIOS power management settings actually work (put monitor to sleep and spin down the hard drive after 10 minutes). `adv:min` enables power management support, but with a bias for performance. You can run `help power.exe` for information about more advanced options if interested. *Tip*: You definitely want to enable this option if running DOS in VirtualBox or VMware; without it, DOS runs "wide open" and will suck up 100% of the CPU core assigned to that virtual machine.

ctcm.exe
: This is Creative's Plug and Play configuration manager, necessary for proper configuration of my AWE64 sound card.

SHELL
: This default setting specifies the command interpreter. You generally should not change this.

FILES=30, BUFFERS=5,0, FCBS=1, STACKS=0,0, LASTDRIVE=H
: These performance settings are added by default by DOS and MS-Client. See the help page (`help <option>`) for each for more details. Each should generally be set as low as possible to reduce memory, without going too low as to adversely affect your system. I find the settings above provide a good balance for my system.

mode.com
: MODE can be used to configure various system devices. In this case, I'm using it to increase the number of rows displayed on the screen to 50, from the default of 25, which will let me see more information on-screen; this is the closest DOS equivalent that I could find to changing your display resolution. This option requires ANSI.SYS to be loaded.

Sound Blaster stuff
: These are various variables and configuration utilities setup by my sound card. These generally shouldn't need to be modified

prompt $e[1;34m$p$g $e[0;47;0m
: This command sets the shell prompt, aka "c: prompt". By default it's simply `$p$g`, which shows `c:\path>`. My tweaked version shows the same information, but the prompt is now blue instead of white. See the ANSI.SYS help file for additional information.

Startup order is very important under DOS; loading drivers and programs in the "wrong" order can drastically increase memory usage. Unfortunately, there's no way to guarantee an optimal order. You can find a lot of advice about tweaking MS-DOS memory settings on the internet, but the basic advice I've found that works the best boils down to three parts:

- Use a memory manager such as EMM386 to load as many drivers, etc. into upper memory as possible
- Load the largest drivers first, which helps fit as much as possible into upper memory due to how DOS loads programs into memory
- Use smaller drivers whenever practical, such as what we've done with the CD-ROM and mouse drivers

MS-DOS also includes a utility called MEMMAKER that can help optimize settings for memory usage, but it produced slightly worse results than my hand-tweaked files above. I recommend giving it a shot, though; it may work better on your hardware, and even if it doesn't it provides an option to easily undo the changes.

With all that said, I'm still about 35 KB below my target of 530 KB, and this is already optimized. In order to hit 530 KB I'll need to disable some things. Good candidates:

CD-ROM support
: this generally isn't required on every boot, and can free up 30 KB by itself (plus quite a bit more if

using the Gold Star or Oak drivers). Comment out the cdromdrv.sys and MSCDEX lines and reboot to free up that memory.

net start
: this command automatically maps shared drives. If you don't need it available at the start of every boot, you can disable it from autoexec.bat to free up 14 KB. You can run "net start" manually to map all of your shared drives as needed, and then "net stop" to free up that memory again.

smartdrv
: this isn't necessary for your system to run, but it does fairly noticeably improve the performance. I'd recommend leaving it enabled if possible, but it takes up 28 KB all by itself, so it's a big target when you need to free some memory fast.

ansi.sys, ctmouse
: both of these are entirely optional and can be commented out if unneeded, but at only 4 KB and 3 KB each it's not going to get you a whole lot. Still, it may be enough to push you over the edge if needed.

MS-Client
: MS-Client is, by far, the biggest memory hog. If you do not need mapped drive support or NDIS drivers enabled, comment out all of the `c:\dos\net` stuff from both config files (including the shim driver) to free up oodles of space. Note that you'll lose your PD interface as well if you installed the shim driver, so you'll need to (re)setup your NIC's native packet driver if you want to continue to use non-Microsoft network utilities. if you can get by with only using PD applications, this is a great option.

I elected to disable MS-Client and just load the native packet driver, which took me way beyond my target to 576 KB free. Aside from mapped drives, MS-Client provides almost no benefit to a home user over the PD applications, and even mapped drives is very flaky with recent versions of Samba. SCP/SFTP will meet my file transferring needs, even if it's slightly less efficient, and that extra memory will be put to much better use elsewhere.

Return to top

Additional Applications

At this point, I'd consider the base OS install to be complete. We have a tuned and optimized MS-DOS 6.22 system complete with network support and key network utilities. Everything we installed should be very stable and functional, and should, for the most part, be usable under Windows for Workgroups 3.11 as well (we'll probably swap out our network drivers, but that's a later discussion). Using this base, you can install additional applications, games, utilities, drivers, etc. to customize the system to your tastes. I'll cover a few of the more useful or interesting additions I've found here.

4DOS

4DOS is a replacement shell / command interpreter for DOS. It provides a great many significant enhancements over the default DOS shell, command.com, and is amazingly customizable. To read up on it and download a copy, visit it's home page.

4DOS was originally a commercial program, but has since been discontinued and released as open source. The open source version (called "Free 4DOS"?) seems to still be actively maintained (though it hasn't been updated in a while) and is currently at version 8.00; this is the version you'll want to download from the above site.

Basic installation is quite easy. Copy the entire (unzipped) directory to `c:\apps\4dos`, then run `c:\apps\4dos\4dos.com` and follow the prompts. You can choose to have it automatically update `autoexec.bat` and `config.sys` for you, but for some greater control you can enter `C` or `N` and manually make the following changes. Edit `autoexec.bat` and add/change:

```
rem loadhigh c:\apps\doskey.com -i
c:\apps\4dos\kstack.com
```

Edit `config.sys` and add/change:

```
rem SHELL=c:\dos\command.com c:\dos /p
SHELL=c:\apps\4dos\4dos.com c:\apps\4dos /p
```

4DOS includes tab completion and command history by default, so Enhanced DOSKEY.com is no longer necessary (although 4DOS doesn't support command completion; that is, searching your path and completing commands as they're typed. That's a bummer.) Also, note that KSTACK.COM is only needed if you wish to use 4DOS' KEYSTACK command. Run `help keystack`. If you don't need this, comment out the kstack.com line to free up that memory.

You can run `option.exe` to configure the basic 4DOS settings. There are a lot to play with, but I personally like to set the following non-default options:

Configure, Startup
- Resident in UMB - Yes
- Swapping - XMS, EMS, None
- Buffers UMB options - Yes for all
- Local Alias - No
- Local Function - No

Configure, Display
- Tabs - 4

Configure, Command Line
- Default Mode - Insert
- Cursor: Overstrike - 100%
- Cursor: Insert - 10%
- Move to End - Yes

Exit, Save

One handy feature 4DOS provides is the ability to use aliases. If you're familiar with the concept from Linux, it works similarly; if no, it's easiest to see it in action. Enter `alias ls=dir /wbh`, then enter `ls`. You should get a directory listing in the same style as the default `ls` command on Linux. This is a pretty trivial example, but you can use it to provide a number of useful conveniences. For example, here's my `aliases.cfg` file:

```
dir=*dir %dircmd
ls=dir /wbh
rm=del
mv=move
cp=copy
cat=type
date=echo %_DATE %_TIME
eject=ejectmedia
vi=edit
p=xfstool.exe pktdrv stop ^ %1 %2 %3 %4 %5 %6 %7 %8 %9 ^ xfstool.exe pktdrv
restart
moved=mkdir %2\%@NAME[%1] ^ xcopy /e %1 %2\%@NAME[%1] ^ deltree %1
realias=unalias * ^ alias /r d:\etc\aliases.cfg
```

Many of the above are simple shortcuts, but a few deserve special mention:

dir
: this forces 4DOS to respect DIR command flags set in the DIRCMD variable; this way, the same defaults can be used regardless of whether 4DOS is loaded or not

date
: this simply outputs the current date and time without prompting you to set it

eject
: uses a 4DOS built-in command to eject the CD-ROM drive

p
: this is the shortcut I mentioned in the NFS/XFS section above - by prepending `p` to any command, 4DOS will first unload the XFS packet driver, then run the given command, then reload the XFS packet driver; so, you can run `p ssh username server` to SSH to a server even while an NFS share is mounted, and 4DOS will automatically suspend/restart the driver before and after your SSH session

moved
: this is the best option I've been able to come up with to easily move a directory from one location to another; it's not fool-proof, though, so I welcome any suggestions for imporvement

realias

this simply reloads the alias file to activate any new changes

It isn't necessary to use an alias file, but I find it convenient to keep all alieses grouped together. To load, add `alias /r c:\apps\aliases.cfg`, or whatever path you prefer, to `autoexec.bat`.

Another neat, but this time entirely superfluous, trick is to enable support for colored directory listing output. This is similar to what modern Linux distributions do by default (as shown in this simple example if you're not familiar with it), and makes it possible to quickly and easily recognize common file types at a glance. I modeled the following configuration based on Linux's default scheme for dircolors, with a couple differences and pruned down a bit:

```
ColorDir=dirs:bri blue;hidden:bri bla;exe com dos:bri gre;bat btm
cmd:gre;zip tgz:bri red;jpg gif bmp png ico:bri mag;mov mpg:mag;mid mp3
wav:bri cya;txt doc me now 1st diz cfg inf ini:bri yellow;*~* bak:yel
```

The configuration is fairly straightforward, if a bit terse: in order, you specify:

1. List of extensions for a particular file type
2. Colon delimiter
3. Color to apply
4. Semicolon delimiter
5. Repeat

There are a couple additional points to be aware of. The extension list can also be a special type of file, such as `dirs` (directories), `hidden` (hidden files), `rdonly` (read-only files), etc. So, the first item in the line sets my directories blue. The color code is also fairly flexible; as shown above, you can specify both normal colors (`yel`) and bright colors (`bri yel`) (bright colors tend to be easier easier to read against dark backgrounds). You can also change the character background color: `bri blu on yel`, for example, will display blue text on a yellow background.

You can run the above with the `set` command to activate, then run `dir` to test the results. Feel free to customize it a bit, then once you're happy with it you can make it permanent by either adding it to `autoexec.bat` (again, prepended with `set`) or copying it as-is to `c:\4dos\4dos.ini`. I recommend the latter, as it won't clutter up your environmental variables.

For more advanced 4DOS features, refer to the online documentation by running `help`. Also note: if you want to view the original MS-DOS help pages, you'll now need to run `help.com <command>`.

Arachne

Believe it or not, a GUI web browser is available for DOS. Support for modern web standards is quite limited, as should be expected, but basic web browsing should work pretty well. The name of the browser is Arachne, and it can be obtained from it's home page. After a four year hiatus, it very recently had a new release, so it's good to see that it's still under active development.

Installation is a bit involved, but not too complicated. The largest hurdle is that this requires at least 500 KB of free conventional RAM, and if you installed everything listed in this walkthrough so far, you will almost certainly be under that threshold. Please see the above System Optimization for tips on freeing up enough memory i fnecessary. You'll also want to use a mouse for this, so be sure to follow the mouse driver instructions above to get that setup.

Once `mem /c /p` shows >500 KB free of conventional memory, run `a197gpl.exe` to begin the setup process and then follow the prompts. I recommend installing to `c:\apps\arachne` to follow our installation convention, so press `N` when asked and change the directory before continuing.

After everything is unpacked, the GUI setup process will start. If you have just barely more over 500 KB free, the installer will exit with a low memory error, as the memory from the unpacker was still in use at the time, pushing you under the threshold. If this happens, simply run `c:\apps\arachne\arachne.bat` to restart the GUI setup wizard.

Set the video options to your preference. Try to go with at least 1024x768, higher if possible (and you have a reasonably large monitor), but you'll probably be limited by your video card memory. If you have any trouble and need to abort the setup process, you can restart by running `c:\apps\arachne\setup.bat`.

You'll be prompted to set your computer speed profile next; it's probably best to go with Arachne's recommended setting here, as it will do a quick benchmark of your computer first. Select your preferred option and click Next. You'll then be prompted for some system configuration changes. This is entirely personal preference (I prefer to update my files manually after installation, but I do have it create the shortcut batch file for me), so choose what you like and click Next. Set the max video resolution to the same resolution you selected previously and click Next.

Up next is the network configuration. Since we already have a working WATTCP configuration (...right?), choose Manual Setup. Select "Resident packet driver", then select "Use only WATTCP configuration", then set the WATTCP configuration file name to `c:\apps\wattcp.cfg` and click Save.

The next page, Arachne Options, can be configured at any time later on to your preferences, so click "Use new settings" to complete setup. You'll be kicked back into DOS at this point, so now would be a good time to make any system config file changes. Consider changing the following in `config.sys`, if you didn't have Arachne do it for you earlier:

```
FILES=40
```

If you had Arachne create a shortcut, you can move it to `c:\apps` so that it's in your PATH. I also like to rename it back to arachne: `move c:\apps\arachne\a.bat c:\apps\arachne.bat`. If you changed your FILES setting, you'll probably want to reboot now to have that take effect.

Finally, run `arachne.bat` and try loading a website such as www.google.com to verify connectivity. It won't look very pretty, but you should see all of the content in its complete, barely styled glory. :-)

Arachne supports a ton of layout options, performance options, etc., and I strongly suggest you spend some time tuning it to make it a better experience for you. To get started, click on the Desktop link in the right navbar (or press `F10`), then click Options.

Miscellaneous Utilities

This is a fairly random collection of utilities I found while researching this project that I consider useful or interesting. I won't cover them in great detail, but I do recommend checking them out.

Navrátil Software System Information is a comprehensive system information utility, reporting detailed information about the various components in your system. This is by far the best such utility I've found, is one of the only such utilities that is completely free and not crippled, and is even under reasonably active development. As a bonus, it even can even perform a basic CPU benchmark. If you have any questions about what's in your computer or how it's performing, this is a very worthwhile utility.

Snarf is a simple screenshot utility for DOS. It doesn't support much in the way of options, but it works well in my testing. Screen Thief is another screenshot utility. This one has *far* more options, but for some reason saves images as 720 pixels wide rather then 640, which makes everything look stretched horizontally. That seems to be expected behavior and I can't find a way to fix that, and that's a deal breaker for me.

RMENU, in the author's words, is a "'kind of' a telnet server for DOS... that can be used to remotely control a computer running DOS via telnet," providing a menu-driven interface as well as direct command line access. Since this is of somewhat limited practical use I won't spend much time covering the details, but it's a quite nice remote-access solution if that's something you need. A couple other options that might fit the bill are Remoter and Tiny. Unfortunately, Remoter which requires a separate Windows client, so it's not useful for me, and Tiny requires either the Novell or PCTCP TCP/IP stack (yes, there are even *more* network stacks and options that what I covered above...), so can't be used with the network configuration detailed above. So, RMENU is the best option for my situation. The RMENU author has also written a number of additional DOS applications that may be useful, also available from the same link.

ANSIPLUS is a much enhanced re-implementation of the ANSI.SYS driver discussed above, and includes several features that would ordinarily require separate additional utilities, such as a scroll back screen buffer, extended keyboard input buffer, mouse support for console copy/paste, etc. Installation and configuration is a bit wonky, so I recommend reading the included `ansiplus.doc` manual before getting started.

SLOWDOWN is, as the name implies, a utility to slow your computer down. It's useful for very old software (generally games) that run relative to CPU speed. The problem with these games is that they will *always* run faster as CPU speed increases, so once you reach a certain CPU speed (which can be easily hit even with original Pentiums), games become unplayably fast. There are several utilities that can slow your computer down enough to play these games (Mo'Slo probably being the most well known), but my favorite is SLOWDOWN; it's free, has a wealth of features, and is very well documented. In particular, I recommend checking out the included CPUCACHE utility; this can be used to simply disable your CPU's cache, yielding a significant drop in speed. While testing Wing Commander, I found the using CPUCACHE resulted in the most consistent gameplay speed. The SLOWDOWN author also has several additional utilities on his site, including native DOS USB drivers, worth checking out.

Return to top

Addendum

After spending a while playing with the system and, most importantly, playing and installing many of my original DOS games, I have a couple few extra tips I wanted to share.

Memory Management Notes

EMM386

Even through EMS memory is a much older standard than XMS memory, which superceeded it with the release of the Intel 80286 processor and MS-DOS 3.0, a lot of much more recent software still only utilizes EMS for some reason. As a result, I recommend using EMM386 to enable EMS support (with the `ram` option) as part of your standard config. Running EMM386 without EMS support (using `noems` should be avoided as many games will assume that EMS available simply because EMM386 is running and then crash. If you want run a game that uses XMS memory is used, the most reliable option is to simply not load EMM386; this will ensure it uses XMS without any potential conflicts caused by EMM386, but has the unfortunate side-effect of preventing DOS from loading drivers/components into upper memory, which means you'll have less conventional memory available for the game.

Alternative memory managers

If you really want to maximize available memory, you may want to check out alternative memory managers such as Jemm/HimemX and UMBPCI. Both seem to offer more advanced features with a smaller memory footprint, so if you have trouble meeting a particular memory target you may want to give these a shot. I haven't messed with either myself, though; by using a boot menu (discussed below), I was able to free up enough memory for everything I tried, so it just wasn't necessary to add this extra software to the mix.

EMM386/HIMEM bugs

The latest versions of EMM386 and HIMEM that shipped with MS-DOS (including 6.22) contain bugs; generally this shouldn't cause a problem, but if you try to use a slowdown utility or disable your CPU cache (as described in the SLOWDOWN section ablove) it will instantly freeze your computer. You can work around this by replacing the built-in versions with those from Windows 98 SE, the very latest versions released by Microsoft which still work perfectly fine in a native MS-DOS environment. If you run into this problem and don't have a Windows 98 SE system available, you can download my copy from here.

CONFIG.SYS and AUTOEXEC.BAT Notes

I ran into a couple odd problems while testing all of my games that were traced back to the somewhat aggressive settings I had for FILES, STACKS, etc. I'm now using the following more conservative settings:

```
FILES=40
BUFFERS=10,0
FCBS=1
STACKS=9,128
```

This uses an extra few KB of conventional memory, but has resulted in less random application crashes. Now that I'm using boot menus as described below to re-configure my system as necessary for memory hungry games, I can afford to give up a few KB of RAM here, so the aggressive settings aren't really worth it.

Additionally, MSCDEX and SmartDrive can both be tweaked to provide faster performance *and* utilize less conventional memory as shown below:

```
loadhigh c:\dos\mscdex.exe /d:mscd001 /l:f /e /m:30
loadhigh c:\dos\smartdrv.exe 4096 4096 /b:65536 /e:8192 /u
```

For MSCDEX, the `/e` parameter instructs it to utilize expanded memory, and `/m:30` causes it to cache up to 30 buffers, reducing the amount of direct CD-ROM reads your computer needs to make and this increasing CD-ROM performance. Note that EMM386 is required for this; without it, `/e` cannot be used, and `/m:30` will result in a significant increase of conventional memory use; dropping `/m` to 5 or 10 in these circumstances is recommended.

SmartDrive is a bit more complicated. I recommend reading the SmartDrive help page for more details, but the short version is that the above command instructs SmartDrive to always use 4 MB of extended memory for cache, use a read-ahead buffer size of 64 KB, use an element size of 8 KB, and disable support for caching CD-ROM drives. The `/u` option is particularly important - while it seems like caching CD-ROM drives would be a good thing, it has a tendency to corrupt the CD in the process of copying a large number of files. I had issues installing quite a few games because of this, so using `/u` to disable CD-ROM caching while allowing SmartDrive to continue to cache all hard drives is the easiest and most reliable solution. The other options, as mentioned for MSCDEX, all affect memory usage, and without upper memory available from EMM386 this will make SmartDrive consume a large amoutn of conventional memory. Dropping it down to something like `2048 2048 /b:8192 /e:4096` can help a lot in these situations, and you may need to go even lower for particularly memory hungry games.

Boot Menus

MS-DOS 6.0 introduces support for boot menus, which can be used to select and load different system configurations at boot time. This is handy because it let's you easily change system settings by rebooting and choosing the new configuration rather than modifying `config.sys` and/or `autoexec.bat` each time you want to change something. This generally isn't really needed, but it can be useful on a system with multiple games loaded to easily switch between sometimes configurations needed by each different game, eliminating the need for bootdisks and hand-editing each time you want to play a different game.

The DOS 6.2 CONFIG.SYS Menu's for Dummies guide explains how boot menus work pretty well, and includes example `config.sys` and `autoexec.bat` files to show how it's done and to show how you can use the boot menu (handled by `config.sys` to also affect what's loaded by `autoexec.bat`.

For a much more complicated example (probably unnecessarily so), I'm providing my latest `config.sys` and `autoexec.bat` files below. Few notes about this:

1. For a much simpler configuration that provides a perfectly useful and feature-complete system, see the examples above. Boot menus are only necessary if you need to be able to change configurations at boot time, and introduce a good bit of complexity that isn't otherwise needed.
2. My particular configuration is geared toward providing optimal settings for various games. Honestly, I can play most of these games from my default config as noticed above, but I created special configs for many of them any way just to the most optimal setup possible. Treat this as simply an example of what you *could* do rather than what you necessarily *should* do when using boot menus.
3. Again, since this is game-oriented, if any boot option other than the default is chosen, `autoexec.bat` will automatically run a custom game launcher I wrote (`games.bat`) at the end of the process. The `%CONFIG%` variable contains the name of the menu item chosen at boot, so it can be used by both `autoexec.bat` and `games.bat` to determine which drivers/utilities should be loaded and which game(s) should be run.

With that said, here's my fully tricked out `config.sys` with boot menu:

```
REM Configure boot menu
[menu]
menuitem=default,Load standard MS-DOS 6.22 working environment
submenu=games1,Load gaming-optimized environment
menudefault=default,5

REM prompt for game-focused options
[games1]
submenu=games2,Basic optimizations - Disable EMS, disable CD-ROM
menuitem=noemscd,Wing Commander III/IV, Crusader, SkyNET - Disable EMS,
enable CD-ROM
menuitem=noemscdextreme,Privateer 2 - Disable EMS, enable CD-ROM, extreme
optimization
menuitem=emsextreme,Wing Commander I/II - Enable EMS, perform extreme
optimizations
menuitem=emscdnomnosd,Tomb Raider - Enable EMS and CD-ROM, Disable Mouse
and SmartDrive

REM prompt for additional game options
[games2]
```

```
menuitem=basic,All other games - Basic optimizations only
menuitem=noemsnosd,Dark Forces - Disable SmartDrive
menuitem=noemsextreme,Privateer 1 - Perform extreme optimizations

REM common boot/memory settings - always enable XMS
[common]
SWITCHES=/f
DEVICE=c:\dos\himem.sys /testmem:off

REM Enable EMS for default environment or games that require it
[default]
DEVICE=c:\dos\emm386.exe ram 24576 highscan notr i=b000-b7ff
DOS=high,umb
DEVICEHIGH=c:\dos\power.exe adv:min
INSTALLHIGH=c:\apps\ansiplus\ansiplus.exe /e
DEVICEHIGH=c:\dos\cdromdrv.sys /d:mscd001
SHELL=c:\apps\4dos\4dos.com c:\apps\4dos /p

[emsextreme]
DEVICE=c:\dos\emm386.exe ram 24576 highscan notr i=b000-b7ff
DOS=high,umb
SHELL=c:\dos\command.com c:\dos /p

[emscdnomnosd]
DEVICE=c:\dos\emm386.exe ram 24576 highscan notr i=b000-b7ff
DOS=high,umb
DEVICE=c:\dos\cdromdrv.sys /d:mscd001
DEVICEHIGH=c:\dos\ansi.sys
SHELL=c:\apps\4dos\4dos.com c:\apps\4dos /p

REM Disable EMS for all other games
[basic]
DOS=high
DEVICE=c:\dos\ansi.sys
SHELL=c:\apps\4dos\4dos.com c:\apps\4dos /p

[noemscd]
DOS=high
DEVICE=c:\dos\ansi.sys
DEVICE=c:\dos\cdromdrv.sys /d:mscd001
SHELL=c:\apps\4dos\4dos.com c:\apps\4dos /p

[noemscdextreme]
DOS=high
DEVICE=c:\dos\cdromdrv.sys /d:mscd001
SHELL=c:\dos\command.com c:\dos /p

[noemsnosd]
DOS=high
DEVICE=c:\dos\ansi.sys
SHELL=c:\apps\4dos\4dos.com c:\apps\4dos /p

[noemsextreme]
DOS=high
SHELL=c:\dos\command.com c:\dos /p

REM Configure remaining drivers and default environmental settings
[common]
DEVICE=c:\apps\ctcm\ctcm.exe
FILES=40
BUFFERS=10,0
FCBS=1
STACKS=9,128
LASTDRIVE=H
BREAK=on
```

And, the corresponding `autoexec.bat`:

```
@echo off

:: Configure environment based on config.sys menu selection
goto %CONFIG%

:: For default working environment, load all conveniences
:default
c:\apps\ansiplus\setaplus.exe mode 03h height 8 rate 30 delay 1
loadhigh c:\dos\mscdex.exe /d:mscd001 /l:f /e /m:30
loadhigh c:\dos\smartdrv.exe 4096 4096 /b:65536 /e:8192 /u
c:\dos\ctmouse.exe /3 /o
prompt $e[1;34m$p$g $e[0;47;0m
alias /r d:\etc\aliases.cfg
goto network

:: For gaming environment, load memory-optimized configuration
:basic
c:\dos\mode.com con: cols=80 lines=50 rate=32 delay=1
c:\dos\smartdrv.exe 2048 2048 /b:32768 /e:8192 /u
c:\dos\ctmouse.exe /3 /o
prompt $e[1;34m$p$g $e[0;47;0m
alias /r d:\etc\aliases.cfg
goto finish

:: Otherwise, apply more specialized settings
:noemscd
c:\dos\mscdex.exe /d:mscd001 /l:f /m:20
```

```
c:\dos\smartdrv.exe 4096 4096 /b:32768 /e:8192 /u
c:\dos\ctmouse.exe /3 /o
prompt $e[1;34m$p$g $e[0;47;0m
alias /r d:\etc\aliases.cfg
goto finish

:noemscdextreme
c:\dos\mscdex.exe /d:mscd001 /l:f /m:5
c:\dos\smartdrv.exe 2048 2048 /b:2048 /e:1024 /u
c:\dos\ctmouse.exe /3 /o
prompt $p$g
goto finish

:emsextreme
loadhigh c:\dos\smartdrv.exe 4096 4096 /b:32768 /e:8192 /u
c:\dos\ctmouse.exe /3 /o
prompt $p$g
goto finish

:emscdnomnosd
c:\dos\mscdex.exe /d:mscd001 /l:f /m:20
prompt $e[1;34m$p$g $e[0;47;0m
alias /r d:\etc\aliases.cfg
goto finish

:noemsnosd
c:\dos\ctmouse.exe /3 /o
prompt $e[1;34m$p$g $e[0;47;0m
alias /r d:\etc\aliases.cfg
goto finish

:noemsextreme
c:\dos\smartdrv.exe 2048 2048 /b:4096 /e:2048 /u
c:\dos\ctmouse.exe /3 /o
prompt $p$g
goto finish

:: Load network drivers and configure protocol stacks
:network
echo.
echo Initializing 3Com 3C509B-TPO...
loadhigh c:\dos\3c509.com -p 0x60 >nul
echo.
echo Setting mTCP IP address via DHCP...
set MTCPCFG=d:\etc\mtcp.cfg
set TZ=CST6CDT
c:\apps\dhcp.exe >nul
echo Setting system time via NTP...
c:\apps\sntp.exe -set boxdog.legroom.net >nul
set WATTCP.CFG=d:\etc
goto finish

:: Configure remaining common drivers and environmental variables
:finish
echo.
echo Initializing Creative Labs Sound Blaster AWE64...
set SOUND=c:\apps\awe64
set BLASTER=A220 I5 D1 H5 P330 E620 T6
set MIDI=SYNTH:1 MAP:E MODE:0
set CTCM=c:\apps\ctcm
c:\apps\awe64\diagnose.exe /s >nul
c:\apps\awe64\aweutil.com /s >nul
c:\apps\awe64\mixerset.exe /p /q
c:\apps\ctcm\ctcm.exe /s >nul
unset CTCM

path c:\apps;c:\dos
set DIRCMD=/a/o:gne
set TEMP=c:\temp

echo.

:: Run games launcher automatically if appropriate
if not %CONFIG%==default c:\apps\games.bat %CONFIG%
```

Finally, for the curious, you can also obtain my `games.bat` file. It's fairly basic, but does show how you can prompt for basic user input and take action based on that input from a pure MS-DOS system, so it might be handy as a reference if you're not sure how to do that.

With that I think we're finally(!) done. I hope this inspires a few folks to tinker a bit with some of their old hardware, or at least fire up a DOS session under VirtualBox or something, and experience either anew or for the first time the joy of getting MS-DOS configured *just right* for whatever your needs might be. Computing was very different when MS-DOS 6.2 was released back in 1993, twenty years prior to the time this was written, and it's worth remembering that sometimes to better appreciate just how far we've come today.

Hope you enjoyed.

[1] Yes, that's a Men in Black II reference.

How to Install and Configure MS-DOS 6.22

Last Modified: Sun, 08/11/2013 - 16:29

Skip to: Preparation | Installation | System Configuration | Network Configuration | Device Configuration | System Optimization | Additional Applications | Addendum

Introduction

This walkthrough covers installing MS-DOS 6.22 from the original installation diskettes. Why write this in 2013? That's a very valid question, to which there are a few answers:

- Setting up a fully working DOS system will give you great appreciation for how far computing has come. For old-timers, it will be a walk down memory lane; for youngsters who've never used nor even seen DOS before, it should be quite an eye-opening experience to experience first hand both how primitive DOS was and yet how capable it could be.
- A working physical DOS system is the most authentic way to (re-)experience classic PC games. DOSBox does an amazing job of supporting DOS games on modern platforms, but for perfect accuracy, including the full memory management experience (which can be a game unto itself), a real DOS system can't be beat.
- There is a dearth of detailed information about MS-DOS on the internet. This makes sense as MS-DOS predates the web as we know it today, but I don't want knowledge of this system to be lost to time. I did a significant amount of research for this project, and I want to document and share what I've discovered and re-learned for future reference.
- Perhaps most importantly, why not? This project was inspired by a previous project to resurrect my old Packard Bell, my first computer that, not coincidentally, ran MS-DOS 6.2 and Windows for Workgroups 3.11. Rebuilding and enhancing it from a hardware perspective was a fun experience, and now I'm doing the same from a software perspective.

Honestly, if you have no appreciation for old hardware or software, then this is definitely not for you. If, however, you share my passion for technology, not only for the new hotness[1] of today but also the old and busted (and tried and true) of yesterday that got us to where we are today, then I think you'll find this interesting. If you have some old hardware lying around then I hope you'll follow along, but even if not I think you may still find some of this interesting enough to read.

As an alternative, if you want an easy-to-install version of DOS that includes some nice modern conveniences, check out FreeDOS. It's a great project that I highly recommend. For this project, though, I want a (mostly) authentic, original MS-DOS installation.

Note: I provide download links for all discussed software in the relevant section where it's discussed. Links point to the original download location for each file wherever possible, but for the files that no longer have an official source (or a reliable one, in the case of the files hosted on Microsoft's amazingly unreliable FTP server), I've linked to a local copy you can download instead.

Prerequisites

- Old hardware - if it has ISA slots you're probably good to go; anything newer may require some extra work, but it should still be possible to get at least a basic working system installed.
  - Alternatively, you should be able to get this up and running in a virtual machine with VirtualBox or VMware Player, but as with the note about FreeDOS above I'm primarily interested in an authentic experience for this project, which is what's documented here.
- A 3.5" floppy disk drive and at least one floppy diskette (two or more recommended) - It may be possible to hack together a solution that will work from a bootable CD-ROM (see this Tech Support Guy thread for details if you prefer to try that route), but MS-DOS is really only intended to be installed from floppy diskettes.
-

MS-DOS 6.x installation media. If possible, I suggest using or tracking down any original installation media you may have had (in my case, I was able to pull the original MS-DOS 6.2 diskette images off of my Packard Bell recovery CD) or picking up a set on eBay - unless you want a full boxed set, the media itself is quite cheap. If you don't have access to any legit copies and don't want to go the eBay route, you can find a copy online easily enough (I recommend the WinWorld Software Library). I don't generally condone piracy, but given this is twenty year old software that's no longer commercially available, I see no harm at all here.

- Patience, basic CLI experience, and a willingness to tinker - this process will take some time, and you'll likely run into issues here and there that'll require some extra time/effort/thought to work out. Part of the experience here is the journey itself, so if you get immediately frustrated at any given setback you will not enjoy this project. Basic CLI experience is also expected; I hope to provide enough guidance to get you through this project without the need for too much prior experience , but I have to assume you have at least a basic familiarity with the command line.

Tip: I also recommend grabbing a copy of either Universal Extractor or 7-Zip if you're running Windows on your main computer, or p7zip and LHa for UNIX if you're running Linux (both should be available in your package management system). You'll probably need/want to unpack some of the software and drivers listed below on your main computer before copying it over to your new DOS system, and some of these are packed in fairly obscure (for today) formats. These applications should cover all the software I tried to unpack, so having these tools available in advance will save some time and hassle.

Return to top

Preparation

Relevant Software:

- RawWrite for Windows - Unless you have physical installation media, you'll need to write the floppy disk images to real diskettes. RawWrite is a simple way to do this in Windows. Linux can do this natively with `dd` (discussed below)

Floppy Diskette Creation

If you need to create the installation media:

- Under Linux, use this command: `dd if=disk1.img of=/dev/fd0 bs=1024 conv=sync; sync`
- Under Windows, use RawWrite and follow the directions to write the first disk
- Repeat the process for all three installation disks (you may have a fourth disk as well; this contains the optional Supplemental Utilities, and is not needed for initial installation; see below for additional details)

You can get by with only one floppy diskette by writing the next disk image after the previous one completes installation, but I highly recommend using at least two diskettes so that you can have a copy of Disk 1 on hand at all times. You'll likely need to boot from or otherwise use that disk a number of times

Hard Disk Preparation

If you need/want to partition and/or format your hard drive:

1. Boot your computer using Disk 1
2. Press `F3` at the MS-DOS Setup welcome screen to exit the installer
3. To repartition, run `fdisk`
    1. Use option 3 to delete existing partitions, then option 1 to create new partitions
        - *Note:* MS-DOS 6.x can only recognize FAT16 partitions, which are limited to 2 GB in size.
    2. If necessary, use option 2 on your C: drive to flag the "boot" partition
    3. You will be forced to reboot after making changes
4. To create (or reformat) a file system on your hard drive:
    1. Run `format /v:labelname /u /s c:`
        - `/v` sets the disk volume label; this can be omitted

        - `/u` performs an unconditional format, which omits preserving certain recovery information
        - `/s` copies DOS system files to the partition, allowing it to self-boot
    2. Repeat the above command for any additional partitions, but omit `/s`
5. Run `setup.exe` to resume installation

Return to top

Installation

Relevant Software:

- MS-DOS 6.22 Step-Up - free upgrade for all MS-DOS 6.x installations to 6.22; if you only have the MS-DOS 6.0, 6.2, or 6.21 installation media (my copy is version 6.2), download this (local copy)
- MS-DOS 6.22 Supplemental Utilities - optional additional utilities, drivers, and programs only included with previous versions of MS-DOS; not required, but may be of some interest for the curious (grab SUP622.EXE) (local copy)

Aside from possibly needing to FDISK and FORMAT your hard drive, the base MS-DOS installation is actually quite simple:

1. If you haven't already done so, boot your computer using Disk 1
2. To begin setup, press Enter
3. If you already pre-formatted your installation disks as described above, the installer will warn you about already having an existing version of DOS installed. Choose to "Continue Setup and replace your current version of DOS".
4. Set the options for date, time, country and keyboard layout as appropriate, then choose "The settings are correct" to continue.
5. For the installation directory, I recommend choosing the default of `c:\dos`. Press Enter to continue.
6. Switch disks (twice) when prompted, and reboot to complete installation

After rebooting, you'll be in a fresh, and very basic, DOS environment.

Step-Up

If you installed either MS-DOS 6.0, 6.2, or 6.21, you may (optionally) upgrade to 6.22 using MS-DOS 6.22 Step-Up. Since it's free, though, there really isn't a good reason to *not* upgrade.

The most annoying part of this process is just getting the Step-Up files to your DOS system. It's larger than a single floppy, and meant to be downloaded and run directly on the target computer, which is difficult for a freshly installed version of MS-DOS. To workaround, I suggest the following:

1. Download and unpack stepup.exe on your main computer
2. Manually create a floppy disk set as follows:
    - Disk 1: setup.bat readme.now 1msdos62.exe
    - Disk 2: 2msdos62.exe
    - Disk 3: 3msdos62.exe
3. Create a backup directory on your DOS system to hold the Step-Up files (this is useful if you want to reinstall DOS - you can the reinstall Step-Up without copying everything via diskettes again):
    1. `mkdir c:\backup`
    2. `mkdir c:\backup\stepup`

    Tip: Even better, if you have the hard drive space, consider creating a D: drive as well; you can use your D: as your backup and data partition, so that if you want to reinstall you can simply blow away C: at any time but still have all your files, drivers, etc. already saved on D:
4. Copy the upgrade files from each diskette to your backup directory: `copy a:\*.* c:\backup\stepup`
5. Finally, copy your backup stepup directory to a temp directory that can be removed after the upgrade is completed:

1. `mkdir c:\stepup`
2. `copy c:\backup\stepup c:\stepup`

To install/upgrade:

1. `cd c:\stepup`
2. Run `setup.bat`
3. Enter `Y` to agree to the EULA
4. Enter `Y` to verify that C: is your OS drive
5. At the Welcome screen, press `F3` to exit
6. Run `setup.exe /g`
   - The setup process will ordinarily require that you create a set of uninstall floppy disks. This extra step bypasses that requirement
7. At the Welcome screen, press Enter
8. Press Enter to confirm system settings
9. Press `Y` to begin upgrade
10. Reboot when prompted, and confirm you're now running MS-DOS 6.22: `ver`
11. You can safely remove the stepup files and old version of MS-DOS:
    - `deltree /y c:\stepup`
    - `delolddos`

MS-DOS 6.22 Supplemental Utilities

My copy of MS-DOS includes a fourth disk titled "MS-DOS 6.2 Supplemental Utilities," which can also be obtained from the link above (grab the version for MS-DOS 6.22, named SUP622.EXE). This disk is not mandatory, and mostly just includes some extra features from previous versions of DOS that were dropped for version 6.0. If you don't want to bother with it you won't be missing out on much, but just to be thorough let's take a look at how to install the more interesting stuff on here.

1. Insert the diskette and run `a:\setup.bat c:\dos`
2. Enter `S` to install selected components only
3. Choose `Y` or `N` as each component is listed. The only options I find interesting are the Additional MS-DOS Utilities and MS-DOS Shell, you enter `Y` for these if you feel so inclined.
4. When prompted for display type, enter `F5` for VGA
5. Follow the rest of the prompts to complete the installation

Now, why bother with the Additional MS-DOS utilities? One reason, and one reason alone: QBasic Nibbles! I spent many hours playing this relatively simple but addictive Snake-like game as a kid, and was delighted to discover it on the supplemental disk. To try it yourself, just run `qbasic.exe /run c:\dos\nibbles.bas`.

DOSSHELL, however, is a bit more substantial. It's essentially a primitive Windows-like (or perhaps more accurately, 'Windows Light') environment that runs on top of DOS and provides a GUI file manager, file associations, a menu-driven task list, and even primitive pseudo-multitasking (called Task Swapper here). It's not something I'll care to use, but it's a pretty interesting idea that I was completely unfamiliar with before this project, and it seems like it could be a reasonably powerful interface if a little time is spent on configuring it just right (though, honestly, running Windows 3.x instead will provide far more power and flexibility). I'm not going to dive into this here as it's beyond the scope of "(mostly) pure MS-DOS", but here are a couple tips if you want to play around with this:

- You can run it in either text mode (default, `/t`) or GUI mode (`/g`). Both interfaces are quite similar, but GUI mode can be configured to run at a much higher resolution. Try `dosshell.exe /g:h2`, for example.
- The program list in the bottom pane seems like the most interesting part to me. In addition to customizing it through the GUI, you can also edit `dosshell.ini` to tweak it to your heart's content, using the default menu items as an example. Like I said, if you're willing to invest a little time in this, it could probably be tricked out pretty nicely.

Return to top

System Configuration

Relevant Software:

- Enhanced DOSKEY.com - enhanced replacement of the stock DOSKEY utility that adds tab completion to the DOS command line interpreter, in addition to command history support and other useful features.
- Pedit - enhanced replacement of the stock EDIT editor, which will be useful for editing the DOS config files
- Info-Zip UnZip - open source Zip file extraction utility you can use to unpack archives directly on your DOS system (actually finding the correct file to download is far more difficult than it should be, so here's a direct link); there are many other archiving utilities available for DOS, some with support for a great number of formats, but this is easy to obtain and install and supports by far the most common archive format used for DOS programs

Before installing anything else, we'll configure a basic sane working environment:

1. `cd c:\`
2. `mkdir apps` - contains personal installed applications
3. `mkdir temp` - temporary directory used by some applications
4. `mkdir backup` (if necessary) - contains copy of original setup files or drivers
5. `path c:\apps;c:\dos` - set our installed apps to take precedence over built-in apps

Next, I recommend installing a couple utilities that will make the rest of the setup process much easier. As with the step-up files above, getting the files to the system is still annoying at this point. Until we get networking setup later on, I recommend unpacking the programs on your main computer and copying them over with floppies as the easiest option. I also recommend saving a backup copy of all of your installation media/files and drivers under `c:\backup` (or, preferably, `d:\backup` for easy access to it at any later point.

Software Installation

Enhanced DOSKEY.com

1. Copy `doskey.com` to `c:\apps`
2. Run `c:\apps\doskey.com -i` to activate it. Running `doskey -?` will show some additional information about how to use it.

Pedit

1. Copy `peditlgt.exe` to `c:\apps`
   - *Note:* This is the lightweight version of Pedit. If you wish to have access to a spell checker and thesaurus, copy the larger `pedit.exe` instead.
2. For convenience, `move c:\apps\peditlgt.exe c:\apps\edit.exe`. This will let you launch Pedit by simply typing `edit`, making it effectively replace the built-in DOS editor.
3. Configure Pedit by running `edit`, hitting Esc if prompted to open a file, and entering `Alt-F1`. I like the following non-default settings:
   - Editor Style - 2.0
   - Scroll Bar - dark on light (first pattern after 'none')
   - Text Color - 07
   - Background Color - 01
   - Show Char Under Cursor - enabled
   - Show Insert Mode - enabled
   - Tab Size - 4
   - Tab Expand Size - 4
   - F2 is File-Save - disabled
   - Alt Highlights Menu Choices - enabled
   - Save Changed Settings - enabled
   - Also Save Changed Margins - enabled

4. Hit Esc then `Alt-X` to exit and save the changes

UnZip

1. Copy `unzip.exe` to `c:\apps`
2. That's all that's necessary. `unzip32.exe` is more flexible, but requires a separate DPMI (DOS Protected Mode Interface) memory manager, which won't be covered here. Other included binaries simply provide extra functionality not required for Zip extraction.

Configuration

Now that we have a good editor and input processor installed, it's time to configure DOS itself.

Tip: You will be repeatedly hand editing your system configuration files throughout this walkthrough, and it's very possible at some point that you may be stuck with a system that won't boot properly. To recover, hit `F5` when DOS starts booting (when it says "Starting MS-DOS...", usually immediately after your system beeps to indicate it has completed it's POST process). This will perform a clean boot of the system, ignoring your `autoexec.bat` and `config.sys` files. You can then edit them as appropriate to fix the problem, and reboot to load everything again.

To get started, edit `c:\autoexec.bat` and add/change the following (other settings should be left alone for now):

```
loadhigh c:\dos\smartdrv.exe
loadhigh c:\apps\doskey.com -i
path c:\apps;c:\dos;c:\dos\net
set DIRCMD=/o:gne
set TEMP=c:\temp
```

Note: Most DOS commands and options are not case-sensitive. While most of the text in the stock autoexec.bat file is capitalized, I tend to convert it to lowercase just because I find it easier on the eyes. My personal naming convention, used throughout this document, is pretty simple: variables are fully capitalized, and almost everything else is lowercase. Feel free to leave everything uppercase if that works for you, though.

SmartDrive is "a disk caching program ... that improves data transfer rates by storing frequently accessed data in RAM" (per Wikipedia), and is enabled by default in DOS. Some versions of DOS include the `/x` argument by default, which disabled write-behind caching; if you see this, remove it. Unless you're running on a known bad hard disk or expect to have frequent power failures (in which case, you should really fix those problems instead), `/x` won't be needed and only slows down disk writes.

LOADHIGH will load the specified program (when possible) into upper memory, freeing up the all important conventional memory for other applications. This requires that EMM386 also be enabled, as shown below. The PATH setting specifies `c:\apps` first so that our installed applications always take precedence over the OS programs (allowing Pedit to be easily used instead of EDIT, for example). The `c:\dos\net` directory doesn't exist yet, but will shortly.

Finally, the DIRCMD variable instructs the DIR command to display files sorted first by name, then by extension, with directories always listed first (by default, it lists files sorted by date). I also modified TEMP, pointing it to `c:\temp` to make it more general purpose; ie., anything can write to this directory without worrying about possibly overwriting DOS files.

Next, edit `c:\config.sys` and add/change the following (again, other settings should be left alone for now):

```
SWITCHES=/f
DEVICE=c:\dos\himem.sys /testmem:off
DEVICEHIGH=c:\dos\emm386.exe ram i=b000-b7ff
DOS=high,umb
BREAK=on
rem DEVICEHIGH=c:\dos\setver.exe
```

The `/f` option to SWITCHES instructs DOS to skip a two second waiting period when booting so that your system will boot a little faster. Note that this shortens the amount of time you have to press `F5` if you need to clean boot your system, but it's still possible if you're fast, or you hold down `F5` right before DOS starts loading (which is what I do).

HIMEM.SYS is MS-DOS' extended memory manager, allowing the OS and applications to use more than the first 640 KB of available memory (RAM). The `/testmem:off` option instructs HIMEM to not test your RAM on every boot, saving some time in the boot process.

EMM386 is DOS' expanded memory manager and, which (also) allows access to memory over 1 MB. Although this does roughly the same thing as HIMEM.SYS, EMM386 is required in order to load programs and drivers into upper/high memory (the area between 640 KB and 1 MB), so it's generally a good idea to load both. `ram` instructs EMM386 to allocate RAM as EMS (expanded memory - an older version of upper memory management that was superseded by XMS (extended memory)). `noems` can alternatively be used instead to disable expanded memory emulation in order to free up a little extra memory, but many older applications and games (including some discussed here) will only use EMS, so unless you have a specific need I recommend using the `ram` option to enable EMS. I discuss this a bit more in the Addendum below. The `i=b000-b7ff` option frees up a small amount of additional memory that is ordinarily reserved for monochrome monitors.

The `umb` portion of the DOS option instructions MS-DOS to manage the upper memory blocks created by EMM386 (so LOADHIGH and DEVICEHIGH will work), and `high` causes it to attempt to load part of itself into high memory when possible. DEVICEHIGH, like LOADHIGH, instructs the specific program or driver to load itself into high memory. Finally, BREAK enables the use of Ctrl-C for breaking out of misbehaving programs.

I commented out SETVER as it generally shouldn't be needed. Unless you plan on running particularly old (even by MS-DOS 6.x standards) software or drives, you can leave this disabled to free up some space. If you get any errors about a program not supporting this DOS version, uncomment this line and see Microsoft's Knowledge Base article on Using the SETVER command. You can also refer to Microsoft's list of applications that require SETVER

I recommend rebooting at this point to let the new configuration take effect. Run `mem /c /p` after rebooting to verify that high memory is being utilized; at least some programs should be listed in the Upper Memory column.

Return to top

Network Configuration

Relevant Software:

- Microsoft Network Client 3.0 for MS-DOS - this is pretty self-descriptive; will be used if you want to map shared drives on a Samba or Windows server (local copy: disk 1, 2)
- NDIS Packet Driver Shim 1.11 - this is necessary if you want to use both Microsoft and non-Microsoft networking utilities on the same system; specifically, you'll need dis_pkt.zip
- WATTCP - old TCP/IP stack for DOS; this is actually embedded in the applications using it, so the package here is mostly useful for diagnostic applications and documentation. Get wat2002b.zip.
- mTCP - modern TCP/IP stack for DOS; this works similarly to WATTCP, but supports different/additional applications that are bundled with it (including FTP and NTP clients)
- SSH2DOS - DOS versions of SSHv2 client applications, including ssh, scp, and sftp
- wget - DOS version wget, "the non-interactive network downloaded" (if you're not familiar with it, it makes download files from remote http and ftp servers *very* easy in CLI environments)
- XFS Network File System Client 1.91 - NFS client for DOS; this was a commercial program, but has long since been abandoned so I don't have any qualms about telling you to download it (local copy)

NIC Drivers (you'll need to grab drivers for your specific NIC; these drivers cover my system, and will be used for the walkthrough):

- 3Com EtherLink 10/100 PCI 3C905C - 3Com NIC drivers; 3c90x2 (EtherCD Disk 2) from the link contains the DOS drivers (local copy)
- 3Com EtherLink 10 ISA 3C509B - 3Com NIC driver; this can no longer be obtained from any official site, so the FTP mirror linked here is the best current option (I've verified against multiple sources that it's a copy of the original EtherDisk 6.1 release). 3c509x2.exe (EtherDisk Disk 2) contains the DOS drivers (local copy). Crynwr provides an alternative packet driver that's half the size (3 KB vs. 6 KB) and seems to perform just as well as the official driver (grab 3c509116.zip). If

you're only interested in packet driver applications, this is a nice option.

- Kingston EtheRx KNE20T - Kingston NIC driver; this can no longer be obtained from any official site, so the website linked here is the best current option (I've verified against multiple sources that it's a copy of the original QStart 1.2 release) (local copy).

Network Options

Now it's time to (finally) configure networking. Getting this up and running is extremely helpful because it largely eliminates the need for floppy disks. Unfortunately, networking under DOS is... primitive. It can be done, but it's somewhat limited, slow, and painful to install due to multiple driver and network protocols. So, get ready for some fun!

The first decision to make is to decide what kind of networking support you want installed. In brief, there are two types of drivers we're going to install:

NDIS - the newer device driver interface co-developed by Microsoft, with somewhat more features and capabilities at the expense of more bugs and significantly more memory usage. This is required for Microsoft networking utilities, including support for drive mapping.

Packet Driver (PD) - the older, open device driver interface used by pretty much everything before NDIS. Most non-Microsoft network utilities require a packet driver interface and will not work with only NDIS installed.

Fortunately, this isn't a one-or-the-other choice; it's possible to install support for both with some extra work. If you only need one or the other, though, save yourself the hassle and go with that one. I'm going to walk through installing both below, and point out what you should do differently if you only want to install one of the drivers.

Note: Installation order differs depending on whether you want only PD support, only NDIS support, or both. Also, while the PD option may sound more limited than NDIS, it's really not. You can get it setup much faster, it's more efficient, generally more reliable, and much more widely supported, and among the various utilities that use it are DOS versions of SSH, SCP, SFTP, and even an NFS client. The SSH utilities will let you easily and securely copy files from another system without messing with NDIS or enabling DOS support in Samba (which is not enabled by default), and NFS will let you mount an NFS share as a DOS drive letter, very similar to mapped drives. For the full experience it's worth setting up NDIS (it can be cleanly disabled if you decide it's not worth it), but if you just want to get up and running as quickly is possible, the PD-only option should suffice, and is all I use on my system after experimenting with all of this.

NIC Drivers

Both NDIS and (native) PD require hardware drivers for your NIC. In my case, I'm using a 3Com 3C905C, which is a Plug and Play PCI NIC with a relatively easy configuration. If you're using an ISA NIC, there may be more steps involved for specifying the number, I/O address, etc. Download the appropriate driver package for your NIC and consult the documentation for details.

Edit: Due to various circumstances I've also setup a Kingston KNE20T and a 3Com 3C509B in this system, so instructions for all three NICs are included below.

The following files are required for DOS networking for each of the listed NICs:

3Com 3C905C
- ndis2\dos\el90x.dos - NDIS2 driver
- ndis2\dos\oemsetup.inf - NDIS2 configuration file for Microsoft Network Client
- pktdvr\3c90xpd.com - Packet Driver

3Com 3C509B
- ndis2\dos\elnk3.dos - NDIS2 driver
- ndis2\dos\oemsetup.inf - NDIS2 configuration file for Microsoft Network Client
- pktdvr\3c5x9pd.com - Packet Driver
- 3c509.com - Alternative Crynwr Packet Driver; recommended

Kingston KNE20T
- ndis2\ktc20.dos - NDIS2 driver
- ndis2\lanservr.dos\oemsetup.inf - NDIS2 configuration file for Microsoft Network Client

pktdvr\ktc20pkt.com - Packet Driver

Note: For the Kingston KNE20T, the listed files are included in `pnpdata1.exe`. This file can be manually unpacked, but it's probably easiest to use QStart to unpack them for you:

1. Run `qstart.exe`
2. Set the appropriate Plug and Play mode (this will depend on what your computer's BIOS can support) and click choose Continue
3. Select Custom
4. Use one of the Config. options if you'd like to verify your configuration, then click Continue
5. Select Basic Test if, again, you want to verify your configuration, then choose Driver Installation when ready
6. Select "IBM LAN Server ver. 4.0 DOS LAN Services / NDIS 2" or "Packet Driver", choose a reasonable Destination Directory, and choose Copy
7. Choose Exit to complete driver extraction. If you made configuration changes (especially for the PnP mode), you may want to restart to ensure those changes take effect.

Packet Driver Only

If you **only** want to install the PD interface, copy the packet driver to `c:\dos`. Edit `c:\autoexec.bat` and add the following (note: the `/I` for the 3C905C is case sensitive):

3Com 3C905C

```
loadhigh c:\dos\3c90xpd.com /I=0x60
```

3Com 3C509B

Official:

```
loadhigh c:\dos\3c5x9pd.com 0x60
```

Alternative:

```
loadhigh c:\dos\3c509.com -p 0x60
```

Kingston KNE20T

```
loadhigh c:\dos\ktc20pkt.com 0x60
```

The `0x60` option for each specifies the software interrupt; 0x60 is the default, and there shouldn't be a reason to change it unless you're using multiple NICs. This is optional on some cards (such as the 3C905C), but it doesn't hurt to be explicit, and it's required if you want to unload the driver later (with `/u`, where supported).

Run the above command now to manually load the driver; assuming you have an ethernet cable plugged in, it should automatically detect the connection. If all you need is a PD interface, skip to the Packet Driver Configuration section.

NDIS Driver

NDIS driver installation is quite a bit more involved. It can be installed manually like the PD, but instead we're going to let installation and configuration be handled by Microsoft Network Client 3.0 for MS-DOS (MS-Client). MS-Client, as the name implies, is a multi-protocol stack for MS-DOS, requiring the use of an NDIS driver to talk to network hardware.

To prepare for setup, copy both MS-Client disks to your hard drive (as noted above, I recommend keeping a copy in `c:\backup` as well - last time I'll mention this). Running each EXE file will unpack them to the current directory if you didn't unpack them before hand. If prompted, it's safe to overwrite any files as a few exist on both disks (assuming you copied the MS-Client disks to their own directory).; Copy your NIC's NDIS driver and configuration file (noted above) to your hard drive as well; place both in the same directory (if necessary) and note the location.

Note: If you are using a 3c90x card, you must edit `oemsetup.inf` before proceeding with MS-Client installation. Comment out (or delete) the following lines and save the file:

```
ndis3=1:el90x.386
mlid=1:3c90x.com
```

To begin MS-Client installation, run `setup.exe` from Disk 1.

1. Press Enter to skip the Welcome screen
2. Specify `c:\dos\net` for the installation directory (since these are DOS-specific drivers)
3. For network adapter selection, you can choose the driver that's appropriate for your hardware if listed. However, you can almost certainly find a newer driver than what's included with MS-Client, so I recommend choosing "Network adapter not shown on the list below" to use use your downloaded driver instead.
4. Enter the directory path containing your NDIS2 driver configuration file
5. Confirm the chosen driver when prompted
6. Unless your system is extremely short on RAM, I recommend letting MS-Client optimize itself for performance, so press Enter to continue.
7. Enter the username you will use to login to this computer (this is primarily for authenticating against server shares to map drives)
8. Choose Change Names
9. Edit your hostname, workgroup, and domain name as appropriate.
10. Select "The listed names are correct" to return
11. Choose Change Setup Options
12. Select Change Redir Options
13. I recommend using the Basic Redirector if possible to save memory. The only reason you should need to use the Full Redirector is if you plan on authenticating against an NT domain controller.
14. I recommend leaving the Start Options set to "Run Network Client". The Load Pop-up option is only useful if you plan on communicating with other LAN systems using the old netbios messenger service (aka NET SEND, WinPopup, etc.). You can also choose to not run the network client automatically, but we can disable MS-Client from autostarting at any time after installation, so it's easier to let MS-Client configure the startup options itself.
15. Logon Validation and Net Pop Hot Key should be left alone. They can be changed later if desired.
16. Select "The listed options are correct" to return
17. Choose Change Network Configuration
18. This option window is a bit difficult to work with. Press tab to switch to the top pane to choose the adapter/protocol you want to modify, then tab back down to the bottom pane and choose the appropriate action. Select your NIC and choose Change Settings first to review and update any settings as necessary. You generally should not need to modify this unless you have more than one NIC.
19. Select NWLink IPX Compatible Transport and choose Remove - IPX is (generally) no longer used today.
20. When prompted for a new protocol, choose Microsoft TCP/IP - this is what's in common use today
21. Select Microsoft TCP/IP and choose Change Settings
22. If necessary, IP address and network settings can be specified here. By default it's configured for DHCP, so if you're on a DHCP network you can skip this step and return to to Network Configuration.
23. If you plan to communicate with other ancient DOS or Windows systems (pre-Windows 95), you might want to also install and configure Microsoft NetBEUI. This generally shouldn't be needed, though. All reasonably modern server operating systems, including Samba, support NetBIOS over TCP/IP (NBT). NetBEUI itself is not required for NetBIOS communication.
24. When finished, select "Network configuration is correct" to return
25. Return to main configuration menu, and choose "The listed options are correct" to continue installation
26. When prompted for the OEM Driver Disk, specify the path to MS-Client Disk 2, or just press Enter if all files were copied to a common directory
27. Press F3 to exit setup without rebooting

A bug in the MS-Client installer prevents a file necessary for Windows 3.1x support from being installed. This isn't necessary if running solely under DOS, but it doesn't hurt in anyway, so let's install it just to be safe. To do so, from the msclient setup directory run: `expand disk2\wsahdapp.ex_ c:\dos\net\wsahdapp.exe`

If you run into any other trouble, or have questions about some of the options, official MS-Client setup documentation is still available from Microsoft.

MS-Client updated your system configuration files during setup, so let's review those changes and make a couple changes. First, edit `config.sys` and change the following:

```
DEVICEHIGH=c:\dos\net\ifshlp.sys
```

IFSHLP is the installable file system helper, and provides access to file and network APIs used by the MS-Client utilities.

Next, edit autoexec.bat and change:

```
loadhigh c:\dos\net\tcptsr.exe
loadhigh c:\dos\net\tinyrfc.exe
loadhigh c:\dos\net\emsbfr.exe
```

These are some of the various network services installed by MS-Client. We're requesting that they load themselves into high memory. Note that the lines containing the net command, *.com, and nmtsr.exe are NOT set to load in high memory; this is necessary, as these services must be run from conventional memory.

Note: If you get the error "Insufficient memory to load Tiny RFC 1.0" on boot (or a similar error from one of the other services), remove `loadhigh` from the front of that line as well. This indicates you've run out of upper memory, and MS-Client isn't smart enough to load itself into conventional memory instead.

Reboot load the network drivers.

After rebooting, MS-Client should initialize your network card, grab an IP address from DHCP (if enabled), then prompt you to enter a username. This is only required if you plan on mapping shared drives, and can be disabled if you do not wish to auto-mount drives at login. If you **do** plan to mount shared drives, go ahead and enter your username now (or just press Enter to accept what you specified during MS-Client setup), then enter the password for your account. If you **do not** plan to mount shared drives, press Enter, then press Enter again to setup a NULL password. In either case, choose Y to create a password list when prompted.

At this point, Microsoft's TCP/IP stack should be (mostly) up and running, using the NDIS driver. We can perform a few tests to confirm, but note that the utilities described below, though named the same as modern utilities, work differently than what you're probably used to (again, think "primitive").

First, verify that you have an IP address:

```
ipconfig c:\dos\net
```

This should have been provided by DHCP, or should match the manual one you specified during MS-Client setup. Next, make sure you can ping your own IP, eg.:

```
ping 192.168.0.15
```

The bottom line should say "echo received". Finally, make sure you can ping your gateway's IP address (this can be found in the ipconfig output):

```
ping 192.168.0.1
```

Again, the bottom line should say "echo received". If all that worked, congrats, you're up and running!

However, there's one more major issue we need to take care of. Due to another bug in MS-Client, DNS does not work by default, nor is it possible to configure or enable it through the setup utility; this must be done manually.

To enable DNS, first edit `c:\dos\net\tcputils.ini` and append the following:

```
[dnr]
drivername=DNR$
bindings=TCPIP_XIF
```

If you are **not** using DHCP, also add the following below the bindings line:

```
nameserver0=aaa bbb ccc ddd
nameserver1=aaa bbb ccc eee
domain=domain.com
```

Substitute the appropriate values for your network, and be sure to separate the octets with spaces (as shown above) rather than periods as would be done on modern systems.

Next, edit `autoexec.bat` and add the following line immediately above "net start":

```
loadhigh c:\dos\net\dnr.exe
```

Finally, run the above command now to start the Microsoft domain name resolver. The implements DNS functionality in MS-Client. You can verify DNS is working properly with ping again:

```
ping www.google.com
```

As before, you should get an "echo received", along with the IP address for www.google.com.

All of this work will now let us (finally) do something very important: map shared network drives. This is especially useful as we can use this to easily copy files and install new applications and drivers without floppy disks.

To map a drive, assuming you already have a shared drive properly configured in Samba or a Windows server (which is outside the scope of this HOWTO), run:

```
net use e: \\servername\sharename
```

Again, assuming everything has been properly setup on the server, and that you logged in with proper credentials, this should just work; you'll get a "command completed successfully" message if it did. To test, try running `dir e:` and verify that you get output. If it doesn't as expected, and you followed all instructions above, the error is most likely on the server side. As noted, modern versions of both Samba and Windows no longer support DOS or Windows 3.x clients by default, and have to be run in a more insecure mode to make this work. I've had mixed results myself - on my initial testing with MS-Client (about three years ago), I was able to get this working perfectly with Samba; this time, however, I can't. If you have trouble with this, don't worry - SCP, SFTP, and NFS all work perfectly well for copying files using the PD interface, and are described in detail below.

NDIS Packet Driver Shim

The final bit of network setup we're going to do for the NDIS driver is installing a "shim" packet driver. The shim driver basically adds a PD interface to the NDIS driver. This is necessary if you want to use both Microsoft (including mapped drives) and non-Microsoft network utilities on the system (which is highly recommended).

Note that this shim driver is meant specifically for this purpose; it cannot drive your NIC directly without the NDIS driver, nor can your NIC's packet driver be used as a shim. Also, while the driver itself works great, the whole concept is a hack and as a result requires a manual and rather complicated setup process.

To begin, copy `dis_pkt.dos` to `c:\dos\net`. Next, edit `c:\dos\net\protocol.ini` and append the following:

```
[pktdrv]
DriverName=PKTDRV$
BINDINGS=TCM$EL90X
INTVEC=0x60
CHAINVEC=0x68
```

The BINDINGS line needs to be modified for your hardware; everything else (should) be able to stay the same. The BINDINGS line informs the packet driver of which NDIS driver it must bind to. To find this, search for the [TCPIP] section in `protocol.ini`; it should also contain a BINDINGS line. Use the same driver name for the BINDINGS line under [pktdrv].

Save your changes to `protocol.ini` and edit `config.sys` next. Add the following after "ifshlp.sys":

```
DEVICEHIGH=c:\dos\net\protman.dos /i:c:\dos\net
DEVICEHIGH=c:\dos\net\nemm.dos
DEVICEHIGH=c:\dos\net\tcpdrv.dos
DEVICEHIGH=c:\dos\net\el90x.dos
DEVICEHIGH=c:\dos\net\dis_pkt.dos
```

Although it looks like you're adding a lot here, you're only going to be loading one new driver; the `dis_pkt.dos` driver you just copied over. Everything else was already being loaded automatically by MS-Client, but due to some dependency issues you need to load them before dis_pkt, which in turn needs to be loaded before MS-Client.

Two additional notes:

1. The el90x.dos line refers to the NDIS driver for my hardware, a 3com 3C905C. You must change this to point to the NDIS driver for your hardware.
2. Order is important: be sure to leave the lines in the order I have listed above.

We also need to edit autoexec.bat and change the following:

```
rem c:\dos\net\net initialize
```

`net initialize` loads all of the required drivers for MS-Client, which you're now doing manually through `config.sys`. As a result, this command is no longer needed, and will in fact cause errors if you try to run it with the other drivers already loaded.

This is also a good time to decide whether or not you want to automatically map shared drives on boot. If you do not plan on using shared drives, or you prefer to map them manually each time they're needed (which can be done by simply running `net start`), comment out the following line to save a fair amount of conventional memory:

```
rem c:\dos\net\net start
```

Finally, reboot to load the new driver. You should see the following message during the boot process to confirm the shim driver was loaded:

```
MAC/DIS to Packet Driver convert loaded.
```

Packet Driver Configuration

Whether you installed the native PD or the NDIS PD shim, you won't be able to test network functionality using currently available utilities. The PD only provides an interface to your network card; each application using this interface must provide it's own network stack to talk TCP/IP, provide an IP address, etc. Most older applications do this by using the Waterloo TCP/IP stack (WATTCP), which can be embedded into the application and provides all necessary network functionality. A much newer TCP/IP stack called mTCP works similarly, and includes some additional utilities not (well) supported by WATTCP. Fortunately, both stacks can installed and enabled simultaneously, so we'll setup both.

WATTCP

To configure WATTCP, create `c:\apps\wattcp.cfg` and add the following:

```
print = "Configuring WATTCP..."
hostname = <hostname>
my_ip = dhcp
```

If you use a static IP, remove the dhcp line above and add the following:

```
domain.suffix = localdomain
my_ip = 0.0.0.0
netmask = 255.255.255.0
nameserver = 0.0.0.0
nameserver = 0.0.0.0
gateway = 0.0.0.0
```

Edit the values as appropriate, then save and close the file. Finally, add the following line to `autoexec.bat`:

```
set WATTCP.CFG=c:\apps
```

Run the above command now to manually set the configuration variable. Next, let's install a couple diagnostic utilities. Copy `tcpinfo.exe` and `ping.exe` from the WATTCP archive to `c:\apps` (I suggest renaming ping to something like `pingw.exe` to avoid conflicting with the Microsoft or mTCP version of ping).

To test our WATTCP configuration run `tcpinfo.exe`. The top line should state, "Reading Waterloo

TCP configuration file," which indicates that it was able to find your config file. The network parameters output by tcpinfo should either match your static configuration, or should be appropriate for your DHCP network. To test, try the following:

```
pingw www.google.com
```

If you see "Replies lost: 0", everything's working.

mTCP

mTCP works, and is configured, very similarly to WATTCP. You'll need to create another config file, `c:\apps\mtcp.cfg`, and add:

```
packetint 0x60
hostname <hostname>
```

If you use a static IP, also add:

```
ipaddr 0.0.0.0
netmask 255.255.255.0
gateway 0.0.0.0
nameserver 0.0.0.0
mtu 1500
```

Edit the values as appropriate, then save and close the file. Finally, add the following to `autoexec.bat`:

```
set MTCPCFG=c:\apps\mtcp.cfg
c:\apps\dhcp.exe
```

The only practical difference between mTCP and WATTCP (from and end-user's perspective) is that mTCP provides a separate DHCP client, whereas WATTCP applications do this themselves. If you are using a static address you can skip the dhcp line in `autoexec.bat`. Run those `autoexec.bat` line(s) now to initialize mTCP.

To test, copy `ping.exe` to `c:\apps` (again I recommend renaming it to something like `pingm.exe` to avoid conflicts) and run:

```
pingm www.google.com
```

If you see a "Success: 100 %" message, everything's working.

Network Applications

It's time install a few essential network applications. Quite a few exist, but these are the ones I find most useful.

SSH

I primarily run Linux systems, so SSH is very important to me. As a bonus, SCP and SFTP provide great options for transferring files across the network, and is especially useful when NDIS/MS-Client is not setup or mapped drives do not work properly with modern Samba or Windows servers. Fortunately, there's an excellent SSHv2 implementation for DOS called, simply enough, SSH2DOS. It can be downloaded from the link above.

Installation is pretty simple; just copy and rename the following files as suggested:

```
copy scp2d386.exe c:\apps\scp.exe
copy sftpd386.exe c:\apps\sftp.exe
copy ssh2d386.exe c:\apps\ssh.exe
```

SSH2DOS is a WATTCP-based application, so it requires a PD interface and a working `wattcp.cfg` file. If you setup both of these as instructed earlier, you're already good to go. Run `ssh username servername` to verify SSH works as well.

SSH2DOS also includes a thoroughly commented example WATTCP configuration file. If you're having any trouble with your WATTCP apps, or you want to poke around to see what other options are available, take a look at the included `wattcp.cfg`. If you decide to edit and use this version of the file, you can either replace the existing `wattcp.cfg` in `c:\dos`, or point the `WATTTCP.CFG` variable in `autoexec.bat` to the new location (but don't forget to update that variable as well, or reboot, or your apps will continue to use the old configuration file).

wget

wget is another useful utility. This is a command line download manager that can download files from pretty much any web or FTP server and, happily, there's a DOS version available from the link above (though the DOS version is no longer maintained at this point).

Installation is the same as SSH2DOS; `copy bin\wget.exe c:\apps` and you're done. This is another WATTCP-based application, so it should just work at this point.

NTP, FTP, TELNET

Now let's install some mTCP applications. These are all bundled with the mTCP distribution, so just copy the following files to `c:\apps`: `ftp.exe`, `sntp.exe`, `telnet.exe`

Each of these will utilize your mTCP configuration file so you can go ahead and use them directly. FTP and TELNET should be fairly self-explanatory and work how you'd expect, with FTP being especially useful if you run Windows on your main machine and don't have SSH available. The utility I'm most interested in here, though, is SNTP, a simple Network Time Protocol client. You can use this to automatically set your computer's clock to the correct current time, which is *very* handy on ancient computer hardware with bad CMOS batteries. :-) To test this, run the following:

```
set TZ=CST6CDT
sntp.exe pool.ntp.org
```

Note: TZ stands for time zone, and is required so that SNTP knows the correct 'local' time. CST6CDT is the code for the Central Standard Time zone; see `sntp.txt` for additional information about how to set this.

After running the above, you should see your current system time as well as the current NTP server time (listed as "Time should be set to"). To actually set the time, use the `-set` parameter. To always set your system clock to the appropriate time on boot (strongly recommended), add the above, with `-set` to `autoexec.bat`.

Bonus tip: once you get the mTCP utilities installed, try running `telnet towel.blinkenlights.nl` for an unexpected treat. :-)

Note: There are several additional applications/utilities included with both WATTCP and mTCP, but the above are the ones I personally find most useful. Feel free to play around with the others, though.

XFS (NFS client)

If you run an NFS server instead of (or in addition to) an SMB/CIFS server, you'll be happy to know that it is possible to mount an NFS share from DOS. The bad news is that you lose the ability to concurrently use many (but apparently not all) other PD-based applications. The reason is that XFS installs it's own driver literally on top of your packet driver, which conflicts with most other PD applications. wget (for whatever reason) still works with the XFS driver loaded, but none of the other WATTCP- nor mTCP-based utilities can use it.

If you **only** otherwise use NDIS-based applications, this is not an issue. If you **only** use PD-based applications, theoretically you should be able to get them to work with the XFS driver loaded by configuring them to use interrupt vector 0x62 instead of 0x60 (search for "redirected PKTDRVR" in `xfs.txt`), but I could not get that to work. See the end of this section for an alternative approach that you can use instead. If you need to use both NDIS- and PD-based applications, you're pretty much out of luck; you need to choose one or the other at this point (or, of course, not use NFS).

So here's the deal with NFS under DOS: there's an old version of NFS for DOS called XFS Network File System Client. It was a commercial program, but as noted above it's long since been abandoned, so grab a copy if you're so inclined and let's get to work.

Note: Before installing, take a look at `kernels.txt`, which describes the different kernel drivers available. I use the minimal kernel, XFSKRNLM, because that meets my needs and takes up the least amount of memory, but your needs may vary.

Installation is a bit tricky:

1. Copy the following files to to `c:\apps`:

```
xfskrnlm.exe
hosts
ls.exe
xfstool.exe
```

The rest of the executables can be useful for troubleshooting NFS connections, but shouldn't be required for routine use. I also recommend renaming `ls.exe` to `lsx.exe` to avoid conflicting with the 4DOS alias for `ls` (defined below).

2. Edit hosts, comment out everything currently in there, then define your NFS server as:

```
aaa.bbb.ccc.ddd nfsserver.domain.com    nfsserver
```

3. If you're using a static IP address, add the following to `hosts`, where `nfsclient` is the name of your DOS system and the IP information applies to your network configuration:

```
aaa.bbb.ccc.eee gateway
aaa.bbb.ccc.ggg netmask
aaa.bbb.ccc.fff broadcast
aaa.bbb.ccc.ggg nfsclient.domain.com    nfsclient
```

4. If you're using NDIS, copy `dis_pkt9.tcp` to `c:\dos\net` and edit `config.sys`. Add/change the following:

```
rem DEVICEHIGH=c:\dos\net\dis_pkt.dos
DEVICEHIGH=c:\dos\net\dis_pkt9.tcp
```

Note that this disables the old packet driver and loads the new one provided by XFS. Reboot here if using NDIS before continuing.

5. Run the following command to initialize the kernel:

```
loadhigh c:\apps\xfskrnlm.exe 0x60
```

6. Run the following to initialize the driver:
   - If using BOOTP (a predecessor to DHCP; many DHCP servers can also support BOOTP, but may require special configuration):

     ```
     xfstool init bootp
     ```

   - or, if using a static address (`nfsclient` must be defined as described above):

     ```
     xfstool init nfsclient
     ```

7. Finally, try to mount a share:

```
xfstool mount f: nfsserver:/home/data/files
```

Note that XFS **requires** a local hostname to be set. If you're using a static IP address, this should be taken care of. If you're using BOOTP, your DHCP server has to be configured to supply a hostname in addition to an IP address to your system. If XFSTOOL complains about a hostname not being set, this is probably what's happening. The easiest workaround would is to configure a static IP address for XFS, as described above.

If everything worked, you should be able to do run `dir f:` and see a list of files on your NFS server. Switch to F: and run `ls` (or `lsx`) to see the long file names. If using NDIS, I also recommend testing an NDIS application to ensure the shim driver is working properly. `ping hostname` should do the trick, but a more thorough test would be to map a shared drive, eg., with `net start`.

If you want to automount any NFS shares on boot, add the three XFSKRNL, INIT, and MOUNT commands shown above to `autoexec.bat`. Be sure it's added *after* any other PD applications included in `autoexec.bat`, such as mTCP's DHCP client, or the other applications will fail for the reasons listed above.

As mentioned previously, it's not possible to (reliably) use XFS with other PD applications. However, XFS provides the ability to temporarily unload and then reload it's PD shim as needed, so you can leverage this to run other PD applications without completely tearing down NFS support or rebooting. As an example of how to do this, say you want to SSH to another system while you have an NFS share mounted; you can do this by calling the following additional commands before and after SSH:

```
xfstool pktdrv stop
ssh username hostname
xfstool pktdrv restart
```

While in the "stopped" state, your mounted shares will still exist but will not be accessible. Issuing the restart will make them available again. It's not ideal, but it's a fairly reasonable compromise, and can be made less annoying by using a 4DOS alias to simplify that much more (see the 4DOS section below).

Return to top

Device Configuration

Relevant Software and Drivers:

- Toshiba ATAPI CD/DVD-ROM Driver - generic ATAPI E-IDE CD-ROM driver for DOS; unfortunately, this is packed in a self-extracting Windows binary using the LHA format, so you'll have to extract the files on your main system before copying it over to DOS (local copy)
- CuteMouse - modern mouse driver for DOS
- Creative Labs Sound Blaster AWE64 - Sound Blaster AWE64 sound card drivers; you'll need (at least) "Sound Blaster 16/SB32/AWE32 Basic Disk for DOS/Windows 3.1 Installation" and "Creative PnP Configuration Manager (Rev 4)"

Whew, now that we're finally finished with networking, everything else should be a breeze in comparison. :-) Since we now have an easy way to transfer files to our DOS system, it's time to install our remaining drivers and applications. Let's start with the CD-ROM drive.

CD-ROM Driver

Microsoft does not include a CD-ROM driver by default with DOS, and it's difficult (if not impossible) to get a DOS CD-ROM driver from manufacturers today. Fortunately, though, a few generic CD-ROM drivers exist that should cover most standard IDE/ATAPI CD/DVD-ROM drives.

The most widely compatible and widely used MS-DOS CD-ROM drivers are probably from Oak Technologies and Gold Star, both available from the Computer Hope hardware downloads page. Unfortunately, both are memory hogs; the Gold Star driver consumes 25 KB, while the Oak driver consumes a whopping 35 KB. As an alternative I recommend the Toshiba driver linked above. This driver should offer roughly the same compatibility and capabilities, but only uses a svelte 7 KB of memory.

Acer's ATAPI CD-ROM driver (search for VIDE-CDD.SYS v2.15 on the linked page) is another option that uses only 5 KB, as is the modern UDVD2 driver for the FreeDOS project which uses, I think, less than 1 KB. Unfortunately, the UDVD2 driver doesn't behave well on a physical MS-DOS system, and I had some reliability issues with VIDE-CDD.SYS while installing and testing some classic games, so I'd recommend sticking with Toshiba's CDROMDRV.SYS driver.

To install the driver, copy `cdromdrv.sys` to `c:\dos`. Then, edit `config.sys` and add:

```
DEVICEHIGH=c:\dos\cdromdrv.sys /d:mscd001
```

The `/d` option sets a device name to be used by MSCDEX; you can set anything you want here, but mscd001 is the standard name and there's no real benefit to changing it. Next, edit `autoexec.bat` and add:

```
loadhigh c:\dos\mscdex.exe /d:mscd001
```

MSCDEX acts as an interface for DOS to communicate with the CD-ROM driver and, in turn, the device itself. Reboot to load the new driver. If both the driver and MSCDEX were loaded properly, you'll see the following message in the output while the system is booting:

```
Drive D: = Driver MSCD001 unit 0
```

Try inserting a CD-ROM and entering `dir d:` to verify MSCDEX was able to see it as well.

Mouse Driver

Next we'll install a mouse driver. This is optional, as DOS itself doesn't utilize mice, only the applications programmed to support them (such as Pedit). So, unless you plan on running an application or game that uses a mouse, you can skip this.

As with CD-ROM support, MS-DOS doesn't include a mouse driver by default. And again like CD-ROM support, we have a couple options to choose from. The first option is to use the original mouse driver provided by Microsoft with Windows 3.x. This is called MOUSE.COM, and can again be downloaded from the Computer Hope hardware downloads page. The second option is a much newer, open source mouse driver called CuteMouse. It's still actively developed for the FreeDOS project and available from the site linked above.

CuteMouse provides support for modern mice and mouse features such as wheels (though few applications support the wheel), and is significantly smaller than MOUSE.COM, using only about 3.5 KB of memory vs. almost 18 KB for MOUSE.COM. I strongly recommend CuteMouse here, and will use it for the example, but you're welcome to use MOUSE.COM if preferred.

To install CuteMouse, copy `bin\ctmouse.exe` to `c:\dos`. Then, edit `autoexec.bat` and add:

```
c:\dos\ctmouse.exe /3
```

CuteMouse supports quite a few options, so run `ctmouse.exe /?` to get a complete list and set the options appropriate for your hardware. Auto-detection should generally work fine, though. In my case, I use `/3` to force 3-button mode, which isn't enabled by default. Also, note that we're not specifying LOADHIGH here; CuteMouse is smart enough to load itself into upper memory when available, so LOADHIGH isn't needed.

Run the above command to manually load the driver. Pedit supports mice, so you can fire that up to verify that your mouse was properly detected and enabled, or use the included `mousetst.com`.

Sound Card Driver

Next up is sound. You'll need to install drivers appropriate for your sound card. I have a Creative Labs Sound Blaster AWE64, so that's what I'll setup in the example. Your setup should be at least somewhat similar, but certainly some of the details will be different.

To begin, unpack and copy over both the driver disk and configuration manager (linked above), then change to the driver disk directory.

1. Run `install.exe`
2. Press Enter to continue
3. If prompted to install the Creative Configuration Manager (CTCM), press enter to continue, then enter the path to the CTCM files. This is needed for Plug and Play (PnP) cards under DOS. The Intel PnP ISA Configuration Manager (ICM) can be used for this instead, but my experience has been that ICM is much harder to find and really not worth the trouble. CTCM is pretty easy and worked on the first try.
4. If installing CTCM, I recommend setting the path to `c:\apps\ctcm`. Verify the suggested settings are sensible and hit Enter to continue.
5. Hit Enter if prompted to run CTCM, then again when asked about modifying your system config files, then a third time to continue installation
6. For the drivers, I recommend installing to `c:\apps\awe64`. Again, verify the suggested settings are sensible and hit Enter to continue.
7. Make note of the Audio Device settings (you can review this later, if necessary) and hit Enter
8. Hit Enter to make the config file changes, then Enter again to complete installation

We'll need to reboot to activate the sound card, so do that now. After rebooting, change to `c:\apps\awe64` and run `diagnose.exe`. This will let you verify that your sound card is installed and working properly. While here, you can also run `mixerset.exe` to adjust the volume levels as necessary. I find the default config too loud (leading to noticeable distortion on my speakers), so I turn the Master volume down a couple clicks. You may also like to try enabling 3DSE (3D Stereo Enhancement) and see if that improves the sound; I generally don't care for this, but it can make a positive difference with cheap stereo or embedded monitor speakers.

Return to top

System Optimization

At this point it's time to do some configuration cleanup and memory optimization. Remember, you can use `mem /c /p` to view current memory usage and availability. My system, after installing everything listed above and despite tweaking the config files a bit, currently has only 497 KB of conventional memory available. This is actually pretty decent given everything I have loaded, but unfortunately it's not enough for some of the other programs I want to install, and definitely isn't large enough for many games. Generally, you should shoot for >530 KB free, which should cover most (though not all - 587 KB is the highest I've personally encountered) DOS applications and games.

For reference, here's my `config.sys` file at this point:

```
REM configure boot options
SWITCHES=/f

REM enable memory management
DEVICE=c:\dos\himem.sys /testmem:off
DEVICEHIGH=c:\dos\emm386.exe ram 24576 highscan notr i=b000-b7ff
DOS=high,umb

REM load device drivers
DEVICEHIGH=c:\dos\cdromdrv.sys /d:mscd001
DEVICEHIGH=c:\dos\net\ifshlp.sys
DEVICEHIGH=c:\dos\net\protman.dos /i:c:\dos\net
DEVICEHIGH=c:\dos\net\nemm.dos
DEVICEHIGH=c:\dos\net\tcpdrv.dos
DEVICEHIGH=c:\dos\net\elnk3.dos
DEVICEHIGH=c:\dos\net\dis_pkt.dos
DEVICEHIGH=c:\dos\ansi.sys
DEVICEHIGH=c:\dos\power adv:min
DEVICE=c:\apps\ctcm\ctcm.exe
rem DEVICEHIGH=c:\dos\setver.exe

REM configure environment
SHELL=c:\dos\command.com c:\dos /p
FILES=30
BUFFERS=5,0
FCBS=1
STACKS=0,0
LASTDRIVE=H
BREAK=on
```

And my `autoexec.bat` file:

```
@echo off

REM setup display and drivers
c:\dos\mode.com con cols=80 lines=50
loadhigh c:\dos\mscdex.exe /d:mscd001
loadhigh c:\dos\smartdrv.exe
loadhigh c:\apps\doskey.com -i
c:\dos\ctmouse.exe /3

REM setup network
rem loadhigh c:\dos\3c5x9pd.com 0x60
rem c:\dos\net\net.exe initialize
c:\dos\net\netbind.com
c:\dos\net\umb.com
loadhigh c:\dos\net\tcptsr.exe
loadhigh c:\dos\net\tinyrfc.exe
c:\dos\net\nmtsr.exe
loadhigh c:\dos\net\emsbfr.exe
loadhigh c:\dos\net\dnr.exe
c:\dos\net\net start

REM setup sound card
set SOUND=c:\apps\awe64
set BLASTER=A220 I5 D1 H5 P330 E620 T6
set MIDI=SYNTH:1 MAP:E MODE:0
set CTCM=c:\apps\awe64\ctcm
c:\apps\awe64\diagnose.exe /s
c:\apps\awe64\aweutil.com /s
c:\apps\awe64\mixerset.exe /p /q
c:\apps\ctcm\ctcm.exe /s

REM setup environment
path c:\apps;c:\dos;c:\dos\net
prompt $e[1;34m$p$g $e[0;47;0m
set DIRCMD=/o:gne
set TEMP=c:\temp
set WATTCP.CFG=c:\dos\net

echo.
```

Most of this has been covered previously, but there are a few new items:

emm386.exe
: `24576` instructs EMM386 to only make 24 MB of extended memory (out of the 64 MB of RAM available on this system) available as EMS. By default, it'll use all available extended memory, up to 32 MB. I added this mostly for troubleshooting: if a game reports that it sees 24 MB of memory, then that means it's using EMS and not XMS. `highscan` enables more aggressive

attempts to find available upper memory (and this increase available capacity), though it can cause some systems to hang. `notr` disables searching for token ring adapters, a legacy networking topology no longer in use today.

ansi.sys
: DOS driver that enables the use of graphics and special characters in programs that support them. This isn't needed unless a specific program requires it, but in my case I'm using it to increase the number of rows displayed on screen and enable my custom shell prompt (see below).

power.exe
: POWER enables basic power management support. This isn't especially necessary, but enabling it makes the my BIOS power management settings actually work (put monitor to sleep and spin down the hard drive after 10 minutes). `adv:min` enables power management support, but with a bias for performance. You can run `help power.exe` for information about more advanced options if interested. *Tip*: You definitely want to enable this option if running DOS in VirtualBox or VMware; without it, DOS runs "wide open" and will suck up 100% of the CPU core assigned to that virtual machine.

ctcm.exe
: This is Creative's Plug and Play configuration manager, necessary for proper configuration of my AWE64 sound card.

SHELL
: This default setting specifies the command interpreter. You generally should not change this.

FILES=30, BUFFERS=5,0, FCBS=1, STACKS=0,0, LASTDRIVE=H
: These performance settings are added by default by DOS and MS-Client. See the help page (`help <option>`) for each for more details. Each should generally be set as low as possible to reduce memory, without going too low as to adversely affect your system. I find the settings above provide a good balance for my system.

mode.com
: MODE can be used to configure various system devices. In this case, I'm using it to increase the number of rows displayed on the screen to 50, from the default of 25, which will let me see more information on-screen; this is the closest DOS equivalent that I could find to changing your display resolution. This option requires ANSI.SYS to be loaded.

Sound Blaster stuff
: These are various variables and configuration utilities setup by my sound card. These generally shouldn't need to be modified

prompt $e[1;34m$p$g $e[0;47;0m
: This command sets the shell prompt, aka "c: prompt". By default it's simply `$p$g`, which shows `c:\path>`. My tweaked version shows the same information, but the prompt is now blue instead of white. See the ANSI.SYS help file for additional information.

Startup order is very important under DOS; loading drivers and programs in the "wrong" order can drastically increase memory usage. Unfortunately, there's no way to guarantee an optimal order. You can find a lot of advice about tweaking MS-DOS memory settings on the internet, but the basic advice I've found that works the best boils down to three parts:

- Use a memory manager such as EMM386 to load as many drivers, etc. into upper memory as possible
- Load the largest drivers first, which helps fit as much as possible into upper memory due to how DOS loads programs into memory
- Use smaller drivers whenever practical, such as what we've done with the CD-ROM and mouse drivers

MS-DOS also includes a utility called MEMMAKER that can help optimize settings for memory usage, but it produced slightly worse results than my hand-tweaked files above. I recommend giving it a shot, though; it may work better on your hardware, and even if it doesn't it provides an option to easily undo the changes.

With all that said, I'm still about 35 KB below my target of 530 KB, and this is already optimized. In order to hit 530 KB I'll need to disable some things. Good candidates:

CD-ROM support
: this generally isn't required on every boot, and can free up 30 KB by itself (plus quite a bit more if

using the Gold Star or Oak drivers). Comment out the cdromdrv.sys and MSCDEX lines and reboot to free up that memory.

net start
: this command automatically maps shared drives. If you don't need it available at the start of every boot, you can disable it from autoexec.bat to free up 14 KB. You can run "net start" manually to map all of your shared drives as needed, and then "net stop" to free up that memory again.

smartdrv
: this isn't necessary for your system to run, but it does fairly noticeably improve the performance. I'd recommend leaving it enabled if possible, but it takes up 28 KB all by itself, so it's a big target when you need to free some memory fast.

ansi.sys, ctmouse
: both of these are entirely optional and can be commented out if unneeded, but at only 4 KB and 3 KB each it's not going to get you a whole lot. Still, it may be enough to push you over the edge if needed.

MS-Client
: MS-Client is, by far, the biggest memory hog. If you do not need mapped drive support or NDIS drivers enabled, comment out all of the `c:\dos\net` stuff from both config files (including the shim driver) to free up oodles of space. Note that you'll lose your PD interface as well if you installed the shim driver, so you'll need to (re)setup your NIC's native packet driver if you want to continue to use non-Microsoft network utilities. if you can get by with only using PD applications, this is a great option.

I elected to disable MS-Client and just load the native packet driver, which took me way beyond my target to 576 KB free. Aside from mapped drives, MS-Client provides almost no benefit to a home user over the PD applications, and even mapped drives is very flaky with recent versions of Samba. SCP/SFTP will meet my file transferring needs, even if it's slightly less efficient, and that extra memory will be put to much better use elsewhere.

Return to top

Additional Applications

At this point, I'd consider the base OS install to be complete. We have a tuned and optimized MS-DOS 6.22 system complete with network support and key network utilities. Everything we installed should be very stable and functional, and should, for the most part, be usable under Windows for Workgroups 3.11 as well (we'll probably swap out our network drivers, but that's a later discussion). Using this base, you can install additional applications, games, utilities, drivers, etc. to customize the system to your tastes. I'll cover a few of the more useful or interesting additions I've found here.

4DOS

4DOS is a replacement shell / command interpreter for DOS. It provides a great many significant enhancements over the default DOS shell, command.com, and is amazingly customizable. To read up on it and download a copy, visit it's home page.

4DOS was originally a commercial program, but has since been discontinued and released as open source. The open source version (called "Free 4DOS"?) seems to still be actively maintained (though it hasn't been updated in a while) and is currently at version 8.00; this is the version you'll want to download from the above site.

Basic installation is quite easy. Copy the entire (unzipped) directory to `c:\apps\4dos`, then run `c:\apps\4dos\4dos.com` and follow the prompts. You can choose to have it automatically update `autoexec.bat` and `config.sys` for you, but for some greater control you can enter `C` or `N` and manually make the following changes. Edit `autoexec.bat` and add/change:

```
rem loadhigh c:\apps\doskey.com -i
c:\apps\4dos\kstack.com
```

Edit `config.sys` and add/change:

```
rem SHELL=c:\dos\command.com c:\dos /p
SHELL=c:\apps\4dos\4dos.com c:\apps\4dos /p
```

4DOS includes tab completion and command history by default, so Enhanced DOSKEY.com is no longer necessary (although 4DOS doesn't support command completion; that is, searching your path and completing commands as they're typed. That's a bummer.) Also, note that KSTACK.COM is only needed if you wish to use 4DOS' KEYSTACK command. Run `help keystack`. If you don't need this, comment out the kstack.com line to free up that memory.

You can run `option.exe` to configure the basic 4DOS settings. There are a lot to play with, but I personally like to set the following non-default options:

Configure, Startup
- Resident in UMB - Yes
- Swapping - XMS, EMS, None
- Buffers UMB options - Yes for all
- Local Alias - No
- Local Function - No

Configure, Display
- Tabs - 4

Configure, Command Line
- Default Mode - Insert
- Cursor: Overstrike - 100%
- Cursor: Insert - 10%
- Move to End - Yes

Exit, Save

One handy feature 4DOS provides is the ability to use aliases. If you're familiar with the concept from Linux, it works similarly; if no, it's easiest to see it in action. Enter `alias ls=dir /wbh`, then enter `ls`. You should get a directory listing in the same style as the default `ls` command on Linux. This is a pretty trivial example, but you can use it to provide a number of useful conveniences. For example, here's my `aliases.cfg` file:

```
dir=*dir %dircmd
ls=dir /wbh
rm=del
mv=move
cp=copy
cat=type
date=echo %_DATE %_TIME
eject=ejectmedia
vi=edit
p=xfstool.exe pktdrv stop ^ %1 %2 %3 %4 %5 %6 %7 %8 %9 ^ xfstool.exe pktdrv
restart
moved=mkdir %2\%@NAME[%1] ^ xcopy /e %1 %2\%@NAME[%1] ^ deltree %1
realias=unalias * ^ alias /r d:\etc\aliases.cfg
```

Many of the above are simple shortcuts, but a few deserve special mention:

dir
: this forces 4DOS to respect DIR command flags set in the DIRCMD variable; this way, the same defaults can be used regardless of whether 4DOS is loaded or not

date
: this simply outputs the current date and time without prompting you to set it

eject
: uses a 4DOS built-in command to eject the CD-ROM drive

p
: this is the shortcut I mentioned in the NFS/XFS section above - by prepending `p` to any command, 4DOS will first unload the XFS packet driver, then run the given command, then reload the XFS packet driver; so, you can run `p ssh username server` to SSH to a server even while an NFS share is mounted, and 4DOS will automatically suspend/restart the driver before and after your SSH session

moved
: this is the best option I've been able to come up with to easily move a directory from one location to another; it's not fool-proof, though, so I welcome any suggestions for imporvement

realias

this simply reloads the alias file to activate any new changes

It isn't necessary to use an alias file, but I find it convenient to keep all alieses grouped together. To load, add `alias /r c:\apps\aliases.cfg`, or whatever path you prefer, to `autoexec.bat`.

Another neat, but this time entirely superfluous, trick is to enable support for colored directory listing output. This is similar to what modern Linux distributions do by default (as shown in this simple example if you're not familiar with it), and makes it possible to quickly and easily recognize common file types at a glance. I modeled the following configuration based on Linux's default scheme for dircolors, with a couple differences and pruned down a bit:

```
ColorDir=dirs:bri blue;hidden:bri bla;exe com dos:bri gre;bat btm
cmd:gre;zip tgz:bri red;jpg gif bmp png ico:bri mag;mov mpg:mag;mid mp3
wav:bri cya;txt doc me now 1st diz cfg inf ini:bri yellow;*~* bak:yel
```

The configuration is fairly straightforward, if a bit terse: in order, you specify:

1. List of extensions for a particular file type
2. Colon delimiter
3. Color to apply
4. Semicolon delimiter
5. Repeat

There are a couple additional points to be aware of. The extension list can also be a special type of file, such as `dirs` (directories), `hidden` (hidden files), `rdonly` (read-only files), etc. So, the first item in the line sets my directories blue. The color code is also fairly flexible; as shown above, you can specify both normal colors (`yel`) and bright colors (`bri yel`) (bright colors tend to be easier easier to read against dark backgrounds). You can also change the character background color: `bri blu on yel`, for example, will display blue text on a yellow background.

You can run the above with the `set` command to activate, then run `dir` to test the results. Feel free to customize it a bit, then once you're happy with it you can make it permanent by either adding it to `autoexec.bat` (again, prepended with `set`) or copying it as-is to `c:\4dos\4dos.ini`. I recommend the latter, as it won't clutter up your environmental variables.

For more advanced 4DOS features, refer to the online documentation by running `help`. Also note: if you want to view the original MS-DOS help pages, you'll now need to run `help.com <command>`.

Arachne

Believe it or not, a GUI web browser is available for DOS. Support for modern web standards is quite limited, as should be expected, but basic web browsing should work pretty well. The name of the browser is Arachne, and it can be obtained from it's home page. After a four year hiatus, it very recently had a new release, so it's good to see that it's still under active development.

Installation is a bit involved, but not too complicated. The largest hurdle is that this requires at least 500 KB of free conventional RAM, and if you installed everything listed in this walkthrough so far, you will almost certainly be under that threshold. Please see the above System Optimization for tips on freeing up enough memory i fnecessary. You'll also want to use a mouse for this, so be sure to follow the mouse driver instructions above to get that setup.

Once `mem /c /p` shows >500 KB free of conventional memory, run `a197gpl.exe` to begin the setup process and then follow the prompts. I recommend installing to `c:\apps\arachne` to follow our installation convention, so press `N` when asked and change the directory before continuing.

After everything is unpacked, the GUI setup process will start. If you have just barely more over 500 KB free, the installer will exit with a low memory error, as the memory from the unpacker was still in use at the time, pushing you under the threshold. If this happens, simply run `c:\apps\arachne\arachne.bat` to restart the GUI setup wizard.

Set the video options to your preference. Try to go with at least 1024x768, higher if possible (and you have a reasonably large monitor), but you'll probably be limited by your video card memory. If you have any trouble and need to abort the setup process, you can restart by running `c:\apps\arachne\setup.bat`.

You'll be prompted to set your computer speed profile next; it's probably best to go with Arachne's recommended setting here, as it will do a quick benchmark of your computer first. Select your preferred option and click Next. You'll then be prompted for some system configuration changes. This is entirely personal preference (I prefer to update my files manually after installation, but I do have it create the shortcut batch file for me), so choose what you like and click Next. Set the max video resolution to the same resolution you selected previously and click Next.

Up next is the network configuration. Since we already have a working WATTCP configuration (...right?), choose Manual Setup. Select "Resident packet driver", then select "Use only WATTCP configuration", then set the WATTCP configuration file name to `c:\apps\wattcp.cfg` and click Save.

The next page, Arachne Options, can be configured at any time later on to your preferences, so click "Use new settings" to complete setup. You'll be kicked back into DOS at this point, so now would be a good time to make any system config file changes. Consider changing the following in `config.sys`, if you didn't have Arachne do it for you earlier:

```
FILES=40
```

If you had Arachne create a shortcut, you can move it to `c:\apps` so that it's in your PATH. I also like to rename it back to arachne: `move c:\apps\arachne\a.bat c:\apps\arachne.bat`. If you changed your FILES setting, you'll probably want to reboot now to have that take effect.

Finally, run `arachne.bat` and try loading a website such as www.google.com to verify connectivity. It won't look very pretty, but you should see all of the content in its complete, barely styled glory. :-)

Arachne supports a ton of layout options, performance options, etc., and I strongly suggest you spend some time tuning it to make it a better experience for you. To get started, click on the Desktop link in the right navbar (or press `F10`), then click Options.

Miscellaneous Utilities

This is a fairly random collection of utilities I found while researching this project that I consider useful or interesting. I won't cover them in great detail, but I do recommend checking them out.

Navrátil Software System Information is a comprehensive system information utility, reporting detailed information about the various components in your system. This is by far the best such utility I've found, is one of the only such utilities that is completely free and not crippled, and is even under reasonably active development. As a bonus, it even can even perform a basic CPU benchmark. If you have any questions about what's in your computer or how it's performing, this is a very worthwhile utility.

Snarf is a simple screenshot utility for DOS. It doesn't support much in the way of options, but it works well in my testing. Screen Thief is another screenshot utility. This one has *far* more options, but for some reason saves images as 720 pixels wide rather then 640, which makes everything look stretched horizontally. That seems to be expected behavior and I can't find a way to fix that, and that's a deal breaker for me.

RMENU, in the author's words, is a "'kind of' a telnet server for DOS... that can be used to remotely control a computer running DOS via telnet," providing a menu-driven interface as well as direct command line access. Since this is of somewhat limited practical use I won't spend much time covering the details, but it's a quite nice remote-access solution if that's something you need. A couple other options that might fit the bill are Remoter and Tiny. Unfortunately, Remoter which requires a separate Windows client, so it's not useful for me, and Tiny requires either the Novell or PCTCP TCP/IP stack (yes, there are even *more* network stacks and options that what I covered above...), so can't be used with the network configuration detailed above. So, RMENU is the best option for my situation. The RMENU author has also written a number of additional DOS applications that may be useful, also available from the same link.

ANSIPLUS is a much enhanced re-implementation of the ANSI.SYS driver discussed above, and includes several features that would ordinarily require separate additional utilities, such as a scroll back screen buffer, extended keyboard input buffer, mouse support for console copy/paste, etc. Installation and configuration is a bit wonky, so I recommend reading the included `ansiplus.doc` manual before getting started.

SLOWDOWN is, as the name implies, a utility to slow your computer down. It's useful for very old software (generally games) that run relative to CPU speed. The problem with these games is that they will *always* run faster as CPU speed increases, so once you reach a certain CPU speed (which can be easily hit even with original Pentiums), games become unplayably fast. There are several utilities that can slow your computer down enough to play these games (Mo'Slo probably being the most well known), but my favorite is SLOWDOWN; it's free, has a wealth of features, and is very well documented. In particular, I recommend checking out the included CPUCACHE utility; this can be used to simply disable your CPU's cache, yielding a significant drop in speed. While testing Wing Commander, I found the using CPUCACHE resulted in the most consistent gameplay speed. The SLOWDOWN author also has several additional utilities on his site, including native DOS USB drivers, worth checking out.

Return to top

Addendum

After spending a while playing with the system and, most importantly, playing and installing many of my original DOS games, I have a couple few extra tips I wanted to share.

Memory Management Notes

EMM386

Even through EMS memory is a much older standard than XMS memory, which superceeded it with the release of the Intel 80286 processor and MS-DOS 3.0, a lot of much more recent software still only utilizes EMS for some reason. As a result, I recommend using EMM386 to enable EMS support (with the `ram` option) as part of your standard config. Running EMM386 without EMS support (using `noems` should be avoided as many games will assume that EMS available simply because EMM386 is running and then crash. If you want run a game that uses XMS memory is used, the most reliable option is to simply not load EMM386; this will ensure it uses XMS without any potential conflicts caused by EMM386, but has the unfortunate side-effect of preventing DOS from loading drivers/components into upper memory, which means you'll have less conventional memory available for the game.

Alternative memory managers

If you really want to maximize available memory, you may want to check out alternative memory managers such as Jemm/HimemX and UMBPCI. Both seem to offer more advanced features with a smaller memory footprint, so if you have trouble meeting a particular memory target you may want to give these a shot. I haven't messed with either myself, though; by using a boot menu (discussed below), I was able to free up enough memory for everything I tried, so it just wasn't necessary to add this extra software to the mix.

EMM386/HIMEM bugs

The latest versions of EMM386 and HIMEM that shipped with MS-DOS (including 6.22) contain bugs; generally this shouldn't cause a problem, but if you try to use a slowdown utility or disable your CPU cache (as described in the SLOWDOWN section ablove) it will instantly freeze your computer. You can work around this by replacing the built-in versions with those from Windows 98 SE, the very latest versions released by Microsoft which still work perfectly fine in a native MS-DOS environment. If you run into this problem and don't have a Windows 98 SE system available, you can download my copy from here.

CONFIG.SYS and AUTOEXEC.BAT Notes

I ran into a couple odd problems while testing all of my games that were traced back to the somewhat aggressive settings I had for FILES, STACKS, etc. I'm now using the following more conservative settings:

```
FILES=40
BUFFERS=10,0
FCBS=1
STACKS=9,128
```

This uses an extra few KB of conventional memory, but has resulted in less random application crashes. Now that I'm using boot menus as described below to re-configure my system as necessary for memory hungry games, I can afford to give up a few KB of RAM here, so the aggressive settings aren't really worth it.

Additionally, MSCDEX and SmartDrive can both be tweaked to provide faster performance *and* utilize less conventional memory as shown below:

```
loadhigh c:\dos\mscdex.exe /d:mscd001 /l:f /e /m:30
loadhigh c:\dos\smartdrv.exe 4096 4096 /b:65536 /e:8192 /u
```

For MSCDEX, the `/e` parameter instructs it to utilize expanded memory, and `/m:30` causes it to cache up to 30 buffers, reducing the amount of direct CD-ROM reads your computer needs to make and this increasing CD-ROM performance. Note that EMM386 is required for this; without it, `/e` cannot be used, and `/m:30` will result in a significant increase of conventional memory use; dropping `/m` to 5 or 10 in these circumstances is recommended.

SmartDrive is a bit more complicated. I recommend reading the SmartDrive help page for more details, but the short version is that the above command instructs SmartDrive to always use 4 MB of extended memory for cache, use a read-ahead buffer size of 64 KB, use an element size of 8 KB, and disable support for caching CD-ROM drives. The `/u` option is particularly important - while it seems like caching CD-ROM drives would be a good thing, it has a tendency to corrupt the CD in the process of copying a large number of files. I had issues installing quite a few games because of this, so using `/u` to disable CD-ROM caching while allowing SmartDrive to continue to cache all hard drives is the easiest and most reliable solution. The other options, as mentioned for MSCDEX, all affect memory usage, and without upper memory available from EMM386 this will make SmartDrive consume a large amoutn of conventional memory. Dropping it down to something like `2048 2048 /b:8192 /e:4096` can help a lot in these situations, and you may need to go even lower for particularly memory hungry games.

Boot Menus

MS-DOS 6.0 introduces support for boot menus, which can be used to select and load different system configurations at boot time. This is handy because it let's you easily change system settings by rebooting and choosing the new configuration rather than modifying `config.sys` and/or `autoexec.bat` each time you want to change something. This generally isn't really needed, but it can be useful on a system with multiple games loaded to easily switch between sometimes configurations needed by each different game, eliminating the need for bootdisks and hand-editing each time you want to play a different game.

The DOS 6.2 CONFIG.SYS Menu's for Dummies guide explains how boot menus work pretty well, and includes example `config.sys` and `autoexec.bat` files to show how it's done and to show how you can use the boot menu (handled by `config.sys` to also affect what's loaded by `autoexec.bat`.

For a much more complicated example (probably unnecessarily so), I'm providing my latest `config.sys` and `autoexec.bat` files below. Few notes about this:

1. For a much simpler configuration that provides a perfectly useful and feature-complete system, see the examples above. Boot menus are only necessary if you need to be able to change configurations at boot time, and introduce a good bit of complexity that isn't otherwise needed.
2. My particular configuration is geared toward providing optimal settings for various games. Honestly, I can play most of these games from my default config as noticed above, but I created special configs for many of them any way just to the most optimal setup possible. Treat this as simply an example of what you *could* do rather than what you necessarily *should* do when using boot menus.
3. Again, since this is game-oriented, if any boot option other than the default is chosen, `autoexec.bat` will automatically run a custom game launcher I wrote (`games.bat`) at the end of the process. The `%CONFIG%` variable contains the name of the menu item chosen at boot, so it can be used by both `autoexec.bat` and `games.bat` to determine which drivers/utilities should be loaded and which game(s) should be run.

With that said, here's my fully tricked out `config.sys` with boot menu:

```
REM Configure boot menu
[menu]
menuitem=default,Load standard MS-DOS 6.22 working environment
submenu=games1,Load gaming-optimized environment
menudefault=default,5

REM prompt for game-focused options
[games1]
submenu=games2,Basic optimizations - Disable EMS, disable CD-ROM
menuitem=noemscd,Wing Commander III/IV, Crusader, SkyNET - Disable EMS,
enable CD-ROM
menuitem=noemscdextreme,Privateer 2 - Disable EMS, enable CD-ROM, extreme
optimization
menuitem=emsextreme,Wing Commander I/II - Enable EMS, perform extreme
optimizations
menuitem=emscdnomnosd,Tomb Raider - Enable EMS and CD-ROM, Disable Mouse
and SmartDrive

REM prompt for additional game options
[games2]
```

```
menuitem=basic,All other games - Basic optimizations only
menuitem=noemsnosd,Dark Forces - Disable SmartDrive
menuitem=noemsextreme,Privateer 1 - Perform extreme optimizations

REM common boot/memory settings - always enable XMS
[common]
SWITCHES=/f
DEVICE=c:\dos\himem.sys /testmem:off

REM Enable EMS for default environment or games that require it
[default]
DEVICE=c:\dos\emm386.exe ram 24576 highscan notr i=b000-b7ff
DOS=high,umb
DEVICEHIGH=c:\dos\power.exe adv:min
INSTALLHIGH=c:\apps\ansiplus\ansiplus.exe /e
DEVICEHIGH=c:\dos\cdromdrv.sys /d:mscd001
SHELL=c:\apps\4dos\4dos.com c:\apps\4dos /p

[emsextreme]
DEVICE=c:\dos\emm386.exe ram 24576 highscan notr i=b000-b7ff
DOS=high,umb
SHELL=c:\dos\command.com c:\dos /p

[emscdnomnosd]
DEVICE=c:\dos\emm386.exe ram 24576 highscan notr i=b000-b7ff
DOS=high,umb
DEVICE=c:\dos\cdromdrv.sys /d:mscd001
DEVICEHIGH=c:\dos\ansi.sys
SHELL=c:\apps\4dos\4dos.com c:\apps\4dos /p

REM Disable EMS for all other games
[basic]
DOS=high
DEVICE=c:\dos\ansi.sys
SHELL=c:\apps\4dos\4dos.com c:\apps\4dos /p

[noemscd]
DOS=high
DEVICE=c:\dos\ansi.sys
DEVICE=c:\dos\cdromdrv.sys /d:mscd001
SHELL=c:\apps\4dos\4dos.com c:\apps\4dos /p

[noemscdextreme]
DOS=high
DEVICE=c:\dos\cdromdrv.sys /d:mscd001
SHELL=c:\dos\command.com c:\dos /p

[noemsnosd]
DOS=high
DEVICE=c:\dos\ansi.sys
SHELL=c:\apps\4dos\4dos.com c:\apps\4dos /p

[noemsextreme]
DOS=high
SHELL=c:\dos\command.com c:\dos /p

REM Configure remaining drivers and default environmental settings
[common]
DEVICE=c:\apps\ctcm\ctcm.exe
FILES=40
BUFFERS=10,0
FCBS=1
STACKS=9,128
LASTDRIVE=H
BREAK=on
```

And, the corresponding `autoexec.bat`:

```
@echo off

:: Configure environment based on config.sys menu selection
goto %CONFIG%

:: For default working environment, load all conveniences
:default
c:\apps\ansiplus\setaplus.exe mode 03h height 8 rate 30 delay 1
loadhigh c:\dos\mscdex.exe /d:mscd001 /l:f /e /m:30
loadhigh c:\dos\smartdrv.exe 4096 4096 /b:65536 /e:8192 /u
c:\dos\ctmouse.exe /3 /o
prompt $e[1;34m$p$g $e[0;47;0m
alias /r d:\etc\aliases.cfg
goto network

:: For gaming environment, load memory-optimized configuration
:basic
c:\dos\mode.com con: cols=80 lines=50 rate=32 delay=1
c:\dos\smartdrv.exe 2048 2048 /b:32768 /e:8192 /u
c:\dos\ctmouse.exe /3 /o
prompt $e[1;34m$p$g $e[0;47;0m
alias /r d:\etc\aliases.cfg
goto finish

:: Otherwise, apply more specialized settings
:noemscd
c:\dos\mscdex.exe /d:mscd001 /l:f /m:20
```

```
c:\dos\smartdrv.exe 4096 4096 /b:32768 /e:8192 /u
c:\dos\ctmouse.exe /3 /o
prompt $e[1;34m$p$g $e[0;47;0m
alias /r d:\etc\aliases.cfg
goto finish

:noemscdextreme
c:\dos\mscdex.exe /d:mscd001 /l:f /m:5
c:\dos\smartdrv.exe 2048 2048 /b:2048 /e:1024 /u
c:\dos\ctmouse.exe /3 /o
prompt $p$g
goto finish

:emsextreme
loadhigh c:\dos\smartdrv.exe 4096 4096 /b:32768 /e:8192 /u
c:\dos\ctmouse.exe /3 /o
prompt $p$g
goto finish

:emscdnomnosd
c:\dos\mscdex.exe /d:mscd001 /l:f /m:20
prompt $e[1;34m$p$g $e[0;47;0m
alias /r d:\etc\aliases.cfg
goto finish

:noemsnosd
c:\dos\ctmouse.exe /3 /o
prompt $e[1;34m$p$g $e[0;47;0m
alias /r d:\etc\aliases.cfg
goto finish

:noemsextreme
c:\dos\smartdrv.exe 2048 2048 /b:4096 /e:2048 /u
c:\dos\ctmouse.exe /3 /o
prompt $p$g
goto finish

:: Load network drivers and configure protocol stacks
:network
echo.
echo Initializing 3Com 3C509B-TPO...
loadhigh c:\dos\3c509.com -p 0x60 >nul
echo.
echo Setting mTCP IP address via DHCP...
set MTCPCFG=d:\etc\mtcp.cfg
set TZ=CST6CDT
c:\apps\dhcp.exe >nul
echo Setting system time via NTP...
c:\apps\sntp.exe -set boxdog.legroom.net >nul
set WATTCP.CFG=d:\etc
goto finish

:: Configure remaining common drivers and environmental variables
:finish
echo.
echo Initializing Creative Labs Sound Blaster AWE64...
set SOUND=c:\apps\awe64
set BLASTER=A220 I5 D1 H5 P330 E620 T6
set MIDI=SYNTH:1 MAP:E MODE:0
set CTCM=c:\apps\ctcm
c:\apps\awe64\diagnose.exe /s >nul
c:\apps\awe64\aweutil.com /s >nul
c:\apps\awe64\mixerset.exe /p /q
c:\apps\ctcm\ctcm.exe /s >nul
unset CTCM

path c:\apps;c:\dos
set DIRCMD=/a/o:gne
set TEMP=c:\temp

echo.

:: Run games launcher automatically if appropriate
if not %CONFIG%==default c:\apps\games.bat %CONFIG%
```

Finally, for the curious, you can also obtain my `games.bat` file. It's fairly basic, but does show how you can prompt for basic user input and take action based on that input from a pure MS-DOS system, so it might be handy as a reference if you're not sure how to do that.

With that I think we're finally(!) done. I hope this inspires a few folks to tinker a bit with some of their old hardware, or at least fire up a DOS session under VirtualBox or something, and experience either anew or for the first time the joy of getting MS-DOS configured *just right* for whatever your needs might be. Computing was very different when MS-DOS 6.2 was released back in 1993, twenty years prior to the time this was written, and it's worth remembering that sometimes to better appreciate just how far we've come today.

Hope you enjoyed.

[1] Yes, that's a Men in Black II reference.