SYSLINUX

From Syslinux Wiki

Contents

What is SYSLINUX?

SYSLINUX is a boot loader for the Linux operating system which runs on an MS-DOS/Windows FAT filesystem. It is intended to simplify first-time installation of Linux, and for creation of rescue and other special purpose boot disks.

When properly configured, SYSLINUX can be used to completely eliminate the need for distribution of raw boot floppy images. A SYSLINUX floppy can be manipulated using standard MS-DOS (or any OS capable of accessing an MS-DOS filesystem) tools once it has been created.

Options

These are the options common to all versions of the SYSLINUX installer:

(Note: The following command line options are valid for older versions of SYSLINUX. Since version 4.01, there are additional options available.)

(For more available options, run **"syslinux --help"**.)

```
       -s      Safe, slow, stupid: uses simpler code that boots better.
                 This version may work on some very buggy BIOSes on which SYSLINUX would otherwise fail.
                 If you find a machine on which the -s option is required to make it boot reliably,
                 please send as much info about your machine as you can, and include the failure mode.
       -f      Force installing.
       -r      Raid mode:
                 If boot fails, tell the BIOS to boot the next device in the boot sequence
                 (usually the next hard disk), instead of stopping with an error message.
                 This is useful for RAID-1 booting.
```

These are only available in the Windows version:

```
       -m      MBR: install a bootable MBR sector to the beginning of the drive.
       -a      Active: marks the partition used active (=bootable)
```

This can only be used in the linux version:

Since version 4.00:

```
       -t      Specifies the byte offset of the filesystem image in the file.
```

```
                It has to be used with a disk image file.
```

For older versions (deprecated):

```
       -o      Specifies the byte offset of the filesystem image in the file.
                 It has to be used with a disk image file.
```

Creating a Bootable Disk

Installing SYSLINUX will alter the boot sector on the disk, and copy a file named **LDLINUX.SYS** into the root directory.

At boot time, by default, the kernel will be loaded from the image named LINUX on the boot disk. This default can be changed, see the section on the SYSLINUX config file.

If the **Shift** or **Alt** keys are held down during boot, or the **Caps** or **Scroll** locks are set, SYSLINUX will display a LILO-style "boot:" prompt. The user can then type a kernel file name followed by any kernel parameters. The SYSLINUX loader does not need to know about the kernel file in advance. All that is required is a file in the root directory of the disk.

In order to create a bootable disk using SYSLINUX, prepare a normal MS-DOS formatted disk. Copy one or more Linux kernel files to it, then execute:

NT/2K/XP

Since version 4.02:

Syntax:

```
syslinux.exe --version
syslinux.exe [--help]
syslinux.exe [options] <drive>: [bootsecfile]
```

syslinux.exe options:

```
   --directory      -d   Directory for installation target

   --install        -i   Install over the current bootsector

   --update         -U   Update a previous installation

   --zip            -z   Force zipdrive geometry (-H 64 -S 32)

   --sectors=#      -S   Force the number of sectors per track
```

```
--heads=#        -H   Force number of heads
--stupid         -s   Slow, safe and stupid mode
--raid           -r   Fall back to the next device on boot failure
--once=...       -o   Execute a command once upon boot
--clear-once     -O   Clear the boot-once command
--reset-adv           Reset auxilliary data
--menu-save=     -M   Set the label to select as default on the next boot
--mbr            -m   Install an MBR
--active         -a   Mark partition as active
--force          -f   Ignore precautions
```

Examples:

Floppy: (`a:` in this example)

```
syslinux.exe --install a:
```

HardDrive/FlashDrive/etc: (`z:` in this example)

```
syslinux.exe --mbr --active --directory /boot/syslinux/ --install z:
```

- In the above example, syslinux.cfg would be expected to be in *`z:\boot\syslinux\syslinux.cfg`*
- Note that the directory path in the command line is using slash "`/`", not backslash "`\`".
- NOTE: Under NT/2K you may get a *dialog box* about not getting exclusive access and with `Abort/Retry/Ignore` buttons; selecting "**`Ignore`**" will make the command execute sucessfully.

Command line options for older versions (deprecated):

Syntax:

```
~~syslinux.exe [-sfmar][-d directory] <drive>: [bootsecfile]~~
```

Floppy: (`a:` in this example)

```
syslinux.exe a:
```

HardDrive/FlashDrive/etc: (`z:` in this example)

```
syslinux.exe -m -a -d /boot/syslinux z:
```

* In the above example syslinux.cfg would be expected to be in *`z:\boot\syslinux\syslinux.cfg`*.

* NOTE: Under NT/2K you may get a *dialog box* about not getting exclusive access and with `Abort/Retry/Ignore` buttons; selecting "**`Ignore`**" will make the command execute sucessfully.

DOS

Since version 4.02:

Syntax:

```
syslinux.com --version
syslinux.com [--help]
syslinux.com [options] <drive>: [bootsecfile]
```

syslinux.com options:

```
--directory      -d   Directory for installation target
--install        -i   Install over the current bootsector
--update         -U   Update a previous installation
--zip            -z   Force zipdrive geometry (-H 64 -S 32)
--sectors=#      -S   Force the number of sectors per track
--heads=#        -H   Force number of heads
--stupid         -s   Slow, safe and stupid mode
--raid           -r   Fall back to the next device on boot failure
--once=...       -o   Execute a command once upon boot
--clear-once     -O   Clear the boot-once command
--reset-adv           Reset auxilliary data
--menu-save=     -M   Set the label to select as default on the next boot
```

```
--mbr            -m   Install an MBR

--active         -a   Mark partition as active

--force          -f   Ignore precautions
```

Example:

```
syslinux.com --install a:
```

Command line options for older versions (deprecated):

Syntax:

```
~~syslinux.com [-sfmar][-d directory] <drive>: [bootsecfile]~~
```

Example:

```
syslinux.com a:
```

Linux

Since version 4.00:

Syntax:

```
syslinux --version
syslinux [--help]
syslinux [options] <Device_Or_Image>
```

syslinux options:

```
--offset         -t   Offset of the file system on the device

--directory      -d   Directory for installation target

--install        -i   Install over the current bootsector

--update         -U   Update a previous installation
```

```
--zip            -z   Force zipdrive geometry (-H 64 -S 32)
--sectors=#      -S   Force the number of sectors per track
--heads=#        -H   Force number of heads
--stupid         -s   Slow, safe and stupid mode
--raid           -r   Fall back to the next device on boot failure
--once=...            Execute a command once upon boot
--clear-once     -O   Clear the boot-once command
--reset-adv           Reset auxilliary data
--menu-save=     -M   Set the label to select as default on the next boot
--force          -f   Ignore precautions
```

The "**-t**" option (if specified) is used with a *disk image file* and specifies the *byte offset* of the filesystem image in the file. (Note that the specific letter used for the *byte offset* option was changed in version 4.00+ of syslinux).

Example:

```
syslinux --directory /boot/syslinux/ --install /dev/sdb1
```

Command line options for older versions (deprecated):

Syntax:

```
~~syslinux [-sfr][-d directory][-o offset] <DeviceOrImage>~~
```

Example:

```
syslinux /dev/fd0
```

Command line options for older versions (deprecated):

~~The **-o** option (if specified) is used with a **disk image file** and specifies the **byte offset** of the filesystem image in the file.~~

How do I Configure SYSLINUX?

All the configurable defaults in SYSLINUX can be changed by creating a file called **syslinux.cfg**.

SYSLINUX searches for the **SYSLINUX.CFG** file in the following order:

```
/boot/syslinux/syslinux.cfg
/syslinux/syslinux.cfg
/syslinux.cfg
```

All filenames inside the config file are assumed to be relative to the directory SYSLINUX.CFG is in, unless preceded with a slash or backslash.

syslinux.cfg is a text file in either UNIX or DOS format, containing one or more of the keywords listed below. Keywords are case insensitive. Upper case is used here to indicate a word should be typed verbatim.

Here is a simple example **syslinux.cfg** file, with one entry to boot a Linux kernel:

```
DEFAULT linux
LABEL linux
  SAY Now booting the kernel from SYSLINUX...
  KERNEL vmlinuz.img
  APPEND ro root=/dev/sda1 initrd=initrd.img
```

Note that LILO uses the syntax:

```
image = mykernel
  label = mylabel
  append = "myoptions"
```

... whereas SYSLINUX uses the syntax:

```
LABEL mylabel
  KERNEL mykernel
  APPEND myoptions
```

All options here apply to PXELINUX, ISOLINUX and EXTLINUX as well as SYSLINUX unless otherwise noted.

INCLUDE filename

Inserts the contents of another file at this point in the configuration file. Currently, files can be nested up to 16 levels deep, but it is not guaranteed that more than 8 levels will be supported in future versions.

LABEL command

A human-readable string that describes a kernel and options. The default LABEL is "linux", but you can change this with the "DEFAULT" keyword.

Labels are mangled as if they were filenames, and must be unique after mangling. For example, the labels "v2.1.30" and "v2.1.31" will not be distinguishable under SYSLINUX, since both mangle to the same DOS filename.

KERNEL file

Selects the file SYSLINUX will boot. The "kernel" doesn't have to be a Linux kernel, it can be a boot sector or a COMBOOT file.

Chain loading requires the boot sector of the foreign operating system to be stored in a file in the root directory of the filesystem. Because neither Linux kernel boot sector images, nor COMBOOT files have reliable magic numbers, Syslinux will look at the file extension. The following extensions are recognized (case insensitive):

```
none or other  Linux kernel image
.0             PXE bootstrap program (NBP) [PXELINUX only]
.bin           "CD boot sector" [ISOLINUX only]
.bs            Boot sector [SYSLINUX only]
.bss           Boot sector, DOS superblock will be patched in [SYSLINUX only]
.c32           COM32 image (32-bit COMBOOT)
.cbt           COMBOOT image (not runnable from DOS)
.com           COMBOOT image (runnable from DOS)
.img           Disk image [ISOLINUX only]
```

Using one of these keywords instead of KERNEL forces the filetype, regardless of the filename:

LINUX image

You can use this, instead of using **KERNEL file** to boot a linux kernel image.

BOOT image

Bootstrap program (.bs, .bin)

BSS image

BSS image (.bss)

PXE image

PXE Network Bootstrap Program (.0)

FDIMAGE image

Floppy disk image (.img)

COMBOOT image

COMBOOT program (.com, .cbt)

COM32 image

COM32 program (.c32)

CONFIG file

CONFIG will restart the boot loader using a different configuration file.

Load new config file:

```
LABEL new_config
CONFIG </path/to/cfg/file/><configfile.cfg>
```

Set Syslinux' new home directory to </path/to/new/base/dir> and load new config file:

```
LABEL new_config2
CONFIG </path/to/cfg/file/><configfile.cfg> </path/to/new/base/dir>
```

or:

```
LABEL new_config2
CONFIG </path/to/cfg/file/><configfile.cfg>
APPEND </path/to/new/base/dir>
```

APPEND options...

Adds one or more options to the kernel command line. These are added to both automatic and manual boots. The options are added at the very beginning of the kernel command line, usually permitting explicitly entered kernel options to override them. This is the equivalent of the LILO "append" option.

The entire APPEND statement must be on a single line. A feature to break up a long line into multiple lines will be added eventually.

If you enter multiple APPEND statements in a single menu entry, only the last one will be used.

APPEND -

Append nothing. APPEND with a single hyphen as argument in a LABEL section can be used to override a global APPEND.

IPAPPEND flag_val [PXELINUX only]

The IPAPPEND option is available only on PXELINUX. The flag_val is an OR of the following options:

1: indicates that an option of the following format should be generated and added to the kernel command line:

```
ip=<client-ip>:<boot-server-ip>:<gw-ip>:<netmask>
```

... based on the input from the DHCP/BOOTP or PXE boot server.

The use of this option is not recommended. If you have to use it, it is probably an indication that your network configuration is broken. Using just ip=dhcp on the kernel command line is a preferrable option, or, better yet, run dhcpcd/dhclient, from an initrd if necessary.

2: indicates that an option of the following format should be generated and added to the kernel command line:

```
BOOTIF=<hardware-address-of-boot-interface>
```

... in dash-separated hexadecimal with leading hardware type (same as for the configuration file; see PXELINUX documentation).

This allows an initrd program to determine which interface the system booted from.

```
LABEL label
KERNEL image
  APPEND options...
  IPAPPEND flag_val [PXELINUX only]
```

... indicates that if "label" is entered as the kernel to boot, SYSLINUX should instead boot "image", and the specified APPEND and IPAPPEND options should be used instead of the ones specified in the global section of the file (before the first LABEL command.) The default for "image" is the same as "label", and if no APPEND is given the default is to use the global entry (if any). Up to 128 LABEL entries are permitted. (for ISOLINUX, 64 LABEL entries.)

LOCALBOOT type [ISOLINUX, PXELINUX]

With PXELINUX, specifying "**LOCALBOOT 0**" instead of a "KERNEL" option means invoking this particular label will cause a local disk boot instead of booting a kernel.

The argument 0 means perform a normal boot. The argument 4 will perform a local boot with the Universal Network Driver Interface (UNDI) driver still resident in memory. Finally, the argument 5 will perform a local boot with the entire PXE stack, including the UNDI driver, still resident in memory. All other values are undefined. If you don't know what the UNDI or PXE stacks are, don't worry, you don't want them, just specify 0.

With ISOLINUX, the "type" specifies the local drive number to boot from; 0x00 is the primary floppy drive and 0x80 is the primary hard drive. The special value -1 causes ISOLINUX to report failure to the BIOS, which, on recent BIOSes, should mean that the next device in the boot sequence should be activated.

INITRD initrd_file

Starting with version 3.71, an initrd can be specified in a separate statement (INITRD) instead of as part of the APPEND statement. This functionally appends "initrd=initrd_file" to the kernel command line.

It supports **multiple filenames separated by commas**. This is mostly useful for initramfs, which can be composed of multiple separate cpio or cpio.gz archives. **Note:** all files except the last one are zero-padded to a 4K page boundary. This should not affect initramfs.

DEFAULT command

Sets the default command line. If SYSLINUX boots automatically, it will act as if the entries after DEFAULT had been typed in at the "boot:" prompt, except that the option "auto" is automatically added, indicating an automatic boot.

In versions prior to v.3.85, if no configuration file is present, or no DEFAULT entry is present in the config file, the default kernel name is "linux", with no options.

Since v.3.85, there is no longer a "linux auto" built-in default. If a configuration file is not found, an error will be displayed. If the DEFAULT statement is not found in the configuration file (and `UI` is not present either - see below), an error will be displayed.

UI module options...

Selects a specific user interface module (typically **menu.c32** or **vesamenu.c32**). The command-line interface treats this as a directive that overrides the DEFAULT and PROMPT directives.

PROMPT flag_val

If flag_val is 0, display the boot: prompt only if the Shift or Alt key is pressed, or Caps Lock or Scroll lock is set (this is the default). If flag_val is 1, always display the **boot:** prompt.

NOESCAPE flag_val

If flag_val is set to 1, ignore the Shift/Alt/Caps Lock/Scroll Lock escapes. Use this (together with PROMPT 0) to force the default boot selection.

NOCOMPLETE flag_val

If flag_val is set to 1, the Tab key does not display labels at the **boot:** prompt.

IMPLICIT flag_val

If flag_val is 0, do not load a kernel image unless it has been explicitly named in a LABEL statement. The default is 1.

ALLOWOPTIONS flag_val

If flag_val is 0, the user is not allowed to specify any arguments on the kernel command line. The only options recognized are those specified in an APPEND statement. The default is 1.

TIMEOUT timeout

Indicates how long to pause at the boot: prompt until booting automatically, in units of 1/10 s. The timeout is cancelled when any key is pressed, the assumption being the user will complete the command line. A timeout of zero will disable the timeout completely. The default is 0.

NOTE: The maximum possible timeout value is 35996. (Just under an hour.)

TOTALTIMEOUT timeout

Indicates how long to wait until booting automatically, in units of 1/10 s. This timeout is *not* cancelled by user input, and can thus be used to deal with serial port glitches or "the user walked away" type situations. A timeout of zero will disable the timeout completely. The default is 0.

Both TIMEOUT and TOTALTIMEOUT can be used together, for example:

```
# Wait 5 seconds unless the user types something, but
# always boot after 15 minutes.
TIMEOUT 50
TOTALTIMEOUT 9000
```

ONTIMEOUT kernel options...

Sets the command line invoked on a timeout. Normally this is the same thing invoked by DEFAULT. If this is specified, then DEFAULT is used only if the user presses <Enter> to boot.

ONERROR kernel options...

If a kernel image is not found (either it doesn't exist, or IMPLICIT is set), run the specified command. The faulty command line is appended to the specified options, so if the ONERROR directive reads:

```
ONERROR xyzzy plugh
```

... and the command line entered by the user is:

```
foo bar baz
```

... SYSLINUX will execute the following as if it were entered by the user:

```
xyzzy plugh foo bar baz
```

SERIAL port [[baudrate] flowcontrol]

Enables a serial port to act as the console. "port" is a number (0 = /dev/ttyS0 = COM1, etc.) or an I/O port address (e.g. 0x3F8). If "baudrate" is omitted, the baud rate defaults to 9600 bps. The serial parameters are hardcoded to 8 bits, no parity and 1 stop bit.

"**flowcontrol**" is a combination of the following bits:

```
0x001 - Assert DTR
0x002 - Assert RTS
0x010 - Wait for CTS assertion
0x020 - Wait for DSR assertion
0x040 - Wait for RI assertion
0x080 - Wait for DCD assertion
0x100 - Ignore input unless CTS asserted
0x200 - Ignore input unless DSR asserted
0x400 - Ignore input unless RI asserted
0x800 - Ignore input unless DCD asserted
```

All other bits are reserved.

Typical values are:

```
    0 - No flow control (default)
0x303 - Null modem cable detect
0x013 - RTS/CTS flow control
0x813 - RTS/CTS flow control, modem input
0x023 - DTR/DSR flow control
0x083 - DTR/DCD flow control
```

For the SERIAL directive to work properly, it must be the **first** directive in the configuration file.

NOTE: "port" values from 0 to 3 mean the first four serial ports detected by the BIOS. They may or may not correspond to the legacy port values 0x3F8, 0x2F8, 0x3E8, 0x2E8.

CONSOLE flag_val

If flag_val is 0, disable output to the normal video console. If flag_val is 1, enable output to the video console (this is the default.) Some BIOSes try to forward this to the serial console which can make a total mess of things, so this option lets you disable the video console on these systems.

FONT filename

Load a font in .psf format before displaying any output (except the copyright line, which is output as ldlinux.sys itself is loaded.) SYSLINUX only loads the font onto the video card. If the .psf file contains a Unicode table, it is ignored. This only works on EGA and VGA cards. Hopefully, it does nothing on others.

KBDMAP keymap

Install a simple keyboard map. The keyboard remapper used is very simplistic (it simply remaps the keycodes received from the BIOS, which means that only the key combinations relevant in the default layout -- usually U.S. English -- can be mapped) but should at least help people with QWERTZ or AZERTY keyboard layouts and the locations of = and , (two special characters used heavily on the Linux kernel command line.)

The included program keytab-lilo.pl from the LILO distribution can be used to create such keymaps. The file keytab-lilo.doc contains the documentation for this program.

Syslinux also ships a comboot module named *kbdmap.c32* which allows changing the keyboard mapping on the fly, making it possible to add a keyboard-selection menu and/or keyboard-selection labels from within the syslinux config file.

SAY message

Prints the message on the screen.

DISPLAY filename

Displays the indicated file on the screen at boot time (before the boot: prompt, if displayed). Please see the section below on DISPLAY files.

```
NOTE: If the file is missing, this option is ignored.
```

F[1-12] filename

```
F1 filename
F2 filename
  ...etc...
F9 filename
F10 filename
F11 filename
F12 filename
```

Displays the indicated file on the screen when a function key is pressed at the **boot:** prompt. This can be used to implement pre-boot online help (presumably for the kernel command line options).

Please see the section below on **DISPLAY files**.

When using the **serial console**, press **<Ctrl-F><digit>** to get to the help screens:

```
<Ctrl-F><1>                     to get the F1 screen
<Ctrl-F><2>                     to get the F2 screen
   ...etc...
<Ctrl-F><9>                     to get the F9 screen
<Ctrl-F><A> (or <Ctrl-F><0>)    to get the F10 screen
<Ctrl-F><B>                     to get the F11 screen
<Ctrl-F><B>                     to get the F11 screen
<Ctrl-F><C>                     to get the F12 screen
```

In the configuration file, **blank lines** and **comment lines** beginning with a hash mark (#) are **ignored**.

Note that the configuration file is not completely decoded. Syntax different from the one described above may still work correctly in this version of SYSLINUX, but may break in a future one.

Is There A Way To Define Constants Or Variables?

At present, there is no way to define constants or variables in the configuration file. That feature will be added eventually.

Can SYSLINUX Handle Large Kernels?

SYSLINUX supports large kernels (bzImage format), eliminating the 500K size limit of the zImage kernel format. bzImage format kernels are detected automatically and handled transparently to the user.

SYSLINUX also supports a boot-time-loaded ramdisk (initrd). An initrd is loaded from a DOS file if the option "initrd=filename" (where filename is the filename of the initrd image; the file must be located in the root directory on the boot floppy) is present on the processed command line (after APPEND's have been added, etc.). If several initrd options are present, the last one has precedence; this permits user-entered options to override a config file APPEND. Specifying "initrd=" without a filename inhibits initrd loading. The file specified by the initrd= option will typically be a gzipped filesystem image.

EXAMPLE (extlinux):

```
APPEND ro root=/dev/hda1 initrd=/boot/initrd.img
```

NOTE: One of the main advantages of SYSLINUX is that it makes it very easy to support users with new or unexpected configurations, especially in a distribution setting. If initrd is used to extensively modularize the distribution kernel, it is strongly recommended that a simple way of adding drivers to the boot floppy be provided. The suggested manner is to let the initrd system mount the boot floppy and look for additional drivers in a predetermined location.

To bzImage and recent zImage kernels, SYSLINUX 1.30 and higher will identify using the ID byte 0x31. PXELINUX identifies using the ID byte 0x32, ISOLINUX 0x33, and EXTLINUX 0x34. The ID range 0x35-0x3f is reserved for future versions of derivatives of SYSLINUX.

What is the DISPLAY File Format?

DISPLAY and function-key help files are text files in either DOS or UNIX format (with or without <CR>). In addition, the following special codes are interpreted:

Clear the screen, home the cursor:

```
<FF>  <FF> = <Ctrl-L> = ASCII 12
```

Note that the screen is filled with the current display color.

Specify *background* and *foreground* colors:

```
<SI><bg><fg>  <SI> = <Ctrl-O> = ASCII 15
```

where <bg> and <fg> are hex digits, corresponding to the standard PC display attributes:

```
0 = black               8 = dark grey
1 = dark blue           9 = bright blue
2 = dark green          a = bright green
3 = dark cyan           b = bright cyan
4 = dark red            c = bright red
5 = dark purple         d = bright purple
6 = brown               e = yellow
7 = light grey          f = white
```

Picking a bright color (8-f) for the background results in the corresponding dark color (0-7), with the foreground flashing.

Colors are not visible over the serial console.

Example *color.txt* file:

```
☼O9eBlinking Yellow on Blue Background
```

```
# xxd color.txt
0000000: 0f39 6542 6c69 6e6b 696e 6720 5965 6c6c  .9eBlinking Yell
0000010: 6f77 206f 6e20 426c 7565 2042 6163 6b67  ow on Blue Backg
0000020: 726f 756e 640a                           round.
```

Display graphic from filename:

```
<CAN>filename<newline>  <CAN> = <Ctrl-X> = ASCII 24
```

If a VGA display is present, enter graphics mode and display the graphic included in the specified file. The file format is an ad hoc format called LSS16; the included Perl program "ppmtolss16" can be used to produce these images. This Perl program also includes the file format specification.

The image is displayed in 640x480 16-color mode. Once in graphics mode, the display attributes (set by <SI> code sequences) work slightly differently: the background color is ignored, and the foreground colors are the 16 colors specified in the image file. For that reason, ppmtolss16 allows you to specify that certain colors should be assigned to specific color indices.

Color indices 0 and 7, in particular, should be chosen with care: 0 is the background color, and 7 is the color used for the text printed by SYSLINUX itself.

Return to text mode:

```
<EM>  <EM> = <Ctrl-Y> = ASCII 25
```

If the output is currently printing in graphic mode, return to text mode.

Output printing modes:

```
<DLE>..<ETB>  <Ctrl-P>..<Ctrl-W> = ASCII 16-23
```

These codes can be used to select which modes to print a certain part of the message file in. Each of these control characters select a specific set of modes (text screen, graphics screen, serial port) for which the output is actually displayed:

```
Character                       Text    Graph   Serial
-----------------------------------------------------
<DLE> = <Ctrl-P> = ASCII 16     No      No      No
<DC1> = <Ctrl-Q> = ASCII 17     Yes     No      No
<DC2> = <Ctrl-R> = ASCII 18     No      Yes     No
<DC3> = <Ctrl-S> = ASCII 19     Yes     Yes     No
<DC4> = <Ctrl-T> = ASCII 20     No      No      Yes
<NAK> = <Ctrl-U> = ASCII 21     Yes     No      Yes
<SYN> = <Ctrl-V> = ASCII 22     No      Yes     Yes
<ETB> = <Ctrl-W> = ASCII 23     Yes     Yes     Yes
```

For example:

```
<DC1>Text mode<DC2>Graphics mode<DC4>Serial port<ETB>
```

will actually print out which mode the console is in!

End of file:

```
<SUB>  <SUB> = <Ctrl-Z> = ASCII 26
```

End of file (DOS convention).

Beep:

```
<BEL>  <BEL> = <Ctrl-G> = ASCII 7
```

Beep the speaker.

Write DISPLAY file with IsoLinux Mate

IsoLinux Mate (ILM) (http://members.chello.at/bobby100/) is a Windows 32-bit executable that makes it

easier to write a DISPLAY file.

With ILM you can insert color commands, insert ASCII graphics with "not a real on-screen keyboard", preview your work, ...

Categories: Configuration | Menu

- This page was last modified on 24 September 2012, at 03:14.