Config

From Syslinux Wiki

Introduction

This document describes the configuration for the boot behavior and user experience of **Syslinux** boot loaders, the format of "Display" files and the boot prompt behavior.

Note that the configuration file is not completely decoded. Syntax different from the one described here may still work correctly in some version of **Syslinux**, but may break in another (future) one.

The configuration file 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.

Blank lines are ignored.

Here is a simple example of a Syslinux configuration file, with one entry to boot a Linux kernel:

```
DEFAULT linux
  SAY Now booting the kernel from SYSLINUX...
LABEL linux
  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 all the bootloaders of the Syslinux family, unless otherwise noted.

Contents

Location and name

Note: In the following paragraphs, the "`/`" directory represents the root of the filesystem in which Syslinux (in its several variants) is (going to be) installed.

BIOS

SYSLINUX / EXTLINUX default to searching for the config file in the installed directory (containing `ldlinux.sys` or `extlinux.sys`).

`[3.35+]` SYSLINUX also searches for the config file in "`/boot/syslinux/`", "`/syslinux/`" and "`/`", in this order.

`[-3.xx]` SYSLINUX uses `syslinux.cfg` as config file name. EXTLINUX (merged into SYSLINUX as of 4.00) used `extlinux.conf`.

`[4.00+]` SYSLINUX searches first for `extlinux.conf` and then for `syslinux.cfg` in each searched-for directory, before falling back to the next directory.

`[-4.02]` ISOLINUX uses `isolinux.cfg` as config file name, searching first in "`/boot/isolinux/`" `[2.00+]`, then in "`/isolinux/`" and then in "`/`".

`[4.03+]` ISOLINUX searches for `isolinux.cfg` and then for `syslinux.cfg` in "`/boot/isolinux/`" before searching for the same files in "`/isolinux/`", "`/boot/syslinux/`", "`/syslinux/`", and "`/`", in this order.

Since version 4.03, the resulting behavior is that the same "`/[[boot/]syslinux/]syslinux.cfg`" file can optionally be used for SYSLINUX / EXTLINUX / ISOLINUX, while specific `isolinux.cfg` and/or `extlinux.conf` files would take precedence if present.

Since version 4.03, the resulting behavior is that any of the respective config files (or even all of them) — namely `isolinux.cfg`, and/or `extlinux.conf`, and/or `syslinux.cfg` — can optionally be located together in the same "`/[[boot/]syslinux/]`" directory.

UEFI

SYSLINUX defaults to searching for `syslinux.cfg` in the installed directory, where `syslinux.efi` is located and containing also its corresponding " *`ldlinux.*`* " file.

Note that `syslinux.efi` could be optionally renamed.

Working directory

Note: In the following paragraphs, the "`/`" directory represents the root of the filesystem in which Syslinux (in its several variants) is (going to be) installed.

When booting, the initial working directory for SYSLINUX / ISOLINUX will be the directory containing the initial configuration file.

`[-4.xx]` If no initial configuration file is found, then SYSLINUX / EXTLINUX / ISOLINUX defaults to "`/`".

`[5.00+]` If no initial configuration file is found, then SYSLINUX / ISOLINUX defaults to the install-time working directory (where " *`ldlinux.*`* " is located).

When booting, the initial working directory for PXELINUX will be the parent directory of pxelinux.0 unless overridden with DHCP option 210. If no configuration file is found, then PXELINUX will start a timer to reboot the system in an attempt to restart the boot process and resolve a possible transient issue.

All (paths to) file names inside the configuration file are relative to the Working Directory, unless preceded with a slash. This is also valid for the command line interface.

The CONFIG directive allows to change the Working Directory.

GLOBAL DIRECTIVES - MAIN

Comments

```
# comment
```

A line comment. `[3.10+]` The space between the "`#`" symbol and the comment is no longer required.

MENU

```
MENU any string
```

`[3.00+]` A directive for the Simple Menu system, treated as a comment outside the menu. See also doc/menu.txt.

INCLUDE

```
INCLUDE filename [tagname]
```

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

See also Menu#INCLUDE.

DEFAULT

```
DEFAULT command
```

Set the default command line (which often references a LABEL). If Syslinux boots automatically, it will act just as if the commands after DEFAULT had been typed in at the "`boot:`" prompt. Multiple uses will result in an override.

`[3.85+]` If no configuration file is found, or neither DEFAULT nor UI entries are present in the config file, then an error message is displayed and the "`boot:`" prompt is shown.

Note: `[-3.84]`, if no configuration file was found, or no DEFAULT entry was present in the configuration file, the default kernel name was "linux", with no options.

Note: Earlier versions of SYSLINUX used to automatically append the string "auto" to whatever the user specified using the DEFAULT command. As of version 1.54, this is no longer true, as it caused problems when using a shell as a substitute for "init". You might want to include this option manually.

See also Menu#DEFAULT.

UI

`UI` *`module options...`*

Select a specific user interface *module* (typically `menu.c32` or `vesamenu.c32`).

Multiple uses will result in an override.

UI overrides the PROMPT directive and takes precedence over the DEFAULT directive. Therefore, if UI is used, the PROMPT directive is ignored and the UI command -- not the DEFAULT command -- is automatically launched:

```
UI menu.c32
DEFAULT mylabel
# Even if present, the PROMPT directive is ignored when UI is used.

LABEL mylabel
KERNEL mykernel
```

With the UI directive specifying the menu system, the DEFAULT directive can be used to select the default entry inside the menus.

The PROMPT and DEFAULT directives might still have effects on other configuration parsers.

## LABEL

`LABEL` *`mylabel`*

Begin a new LABEL clause. If *mylabel* is entered as the kernel to boot, Syslinux should instead boot "image" (specified by a directive from #KERNEL-LIKE DIRECTIVES) with any specified #DUAL-PURPOSE DIRECTIVES being used instead of the global instance.

mylabel must be unique. If not, the first instance is used, but might result in an error or undesired behavior.

mylabel ends at the first character that is not a non-white-space printable character and should be restricted to non-white-space typeable characters.

Prior to version 3.32, this would be transformed to a DOS-compatible format of "8.3" with a restricted character set.

A LABEL clause must contain exactly one of the #KERNEL-LIKE DIRECTIVES and may contain one of each of the #LABEL-ONLY DIRECTIVES or #DUAL-PURPOSE DIRECTIVES.

Within a LABEL, using multiple #KERNEL-LIKE DIRECTIVES or reuse of #LABEL-ONLY DIRECTIVES or #DUAL-PURPOSE DIRECTIVES will result in an override. In other words, multiple instances of the same directive will result in only the last one being effective.

# DUAL-PURPOSE DIRECTIVES

`[-4.xx]` Use of any of the DUAL-PURPOSE DIRECTIVES as GLOBAL DIRECTIVES is discouraged if there will be any non-Linux images loaded, as all images will get these, including those manually entered at the "`boot:`" prompt.

APPEND

`APPEND` *`options...`*

Add 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.

Each APPEND statement shall not span multiple lines; it must be solely on a single line in the configuration file.

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

See also Directives/append.

For information about the "`initrd=`" parameter, see #INITRD.

APPEND -

APPEND -

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

SYSAPPEND

(**IPAPPEND** *bitmask*)
SYSAPPEND *bitmask*

`[IPAPPEND: PXELINUX only; SYSAPPEND: 5.10+]`
The SYSAPPEND option, introduced in Syslinux 5.10, is an enhancement of a previous IPAPPEND option which was only available on PXELINUX.

bitmask is interpreted as decimal format unless prefixed with "0x" for hexadecimal or "0" (zero) for octal. The *bitmask* is an OR (sum) of the following integer options:

1: An option of the following format should be generated, based on the input from the DHCP/BOOTP or PXE boot server, and added to the kernel command line (see note below; empty for non-PXELINUX variants):

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

> Note: The use of option 1 is no substitute for running a DHCP client in the booted system and should instead only be used to seed the client for a request. Without regular renewals, the lease acquired by the PXE BIOS will expire, making the IP address available for reuse by the DHCP server.
>
> Note: 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: An option of the following format should be generated, in dash-separated hexadecimal with leading hardware type (same as for the configuration file; see "doc/pxelinux.txt"), and added to the kernel command line, allowing an initrd program to determine from which interface the system booted (empty for non-PXELINUX variants):

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

4: An option of the following format should be generated, in lower case hexadecimal in the format normally used for UUIDs (same as for the configuration file; see "doc/pxelinux.txt") and added to the kernel command line:

```
SYSUUID=<system uuid>
```

8: `[5.10+]` indicate the CPU family and certain particularly significant CPU feature bits:

```
CPU=<family><features>
```

The *<family>* is a single digit from 3 (i386) to 6 (i686 or higher). The following CPU features are currently reported; additional flags may be added in the future:

```
P   Physical Address Extension (PAE)
V   Intel Virtualization Technology (VT/VMX)
T   Intel Trusted Exection Technology (TXT/SMX)
X   Execution Disable (XD/NX)
L   Long Mode (x86-64)
S   AMD SMX virtualization
```

DMI: `[5.10+]` The following strings are derived from DMI/SMBIOS information if available:

```
    Bit     String          Significance
    ------------------------------------------------------------
    0x00010 SYSVENDOR=      System vendor name
    0x00020 SYSPRODUCT=     System product name
    0x00040 SYSVERSION=     System version
    0x00080 SYSSERIAL=      System serial number
    0x00100 SYSSKU=         System SKU
    0x00200 SYSFAMILY=      System family
    0x00400 MBVENDOR=       Motherboard vendor name
```

```
        0x00800 MBPRODUCT=      Motherboard product name
        0x01000 MBVERSION=      Motherboard version
        0x02000 MBSERIAL=       Motherboard serial number
        0x04000 MBASSET=        Motherboard asset tag
        0x08000 BIOSVENDOR=     BIOS vendor name
        0x10000 BIOSVERSION=    BIOS version
        0x20000 SYSFF=          System form factor
```

If these strings contain white-space characters, they are replaced with underscores (_).

The system form factor value is a number defined in the SMBIOS specification ("System Enclosure or Chassis Types"), available at http://www.dmtf.org/ . As of version 2.7.1 of the specification, the following values are defined:

```
          1     Other
          2     Unknown
          3     Desktop
          4     Low profile desktop
          5     Pizza box
          6     Mini tower
          7     Tower
          8     Portable
          9     Laptop
         10     Notebook
         11     Handheld
         12     Docking station
         13     All-in-one
         14     Subnotebook
         15     Space-saving
         16     Lunch box
         17     Main server chassis
         18     Expansion chassis
         19     Subchassis
         20     Bus expansion chassis
         21     Peripheral chassis
         22     RAID chassis
         23     Rack mount chasss
         24     Sealed-case PC
         25     Multi-system chassis
         26     Compact PCI
         27     Advanced TCA
         28     Blade
         29     Blade enclosure
```

0x40000: `[5.10+]` An option of the following format should be generated, appending a filesystem UUID string to the kernel command line; for EXT2/3/4, the resulting string will be the typical filesystem UUID; for FAT12/16/32, the resulting string will be the 32-bit filesystem serial number (e.g. DA1A-0B2E):

```
    FSUUID=<filesystem uuid>
```

KERNEL-LIKE DIRECTIVES

KERNEL

`KERNEL` *`image`*

Load a kernel-like file *image* with automatic filetype detection based on file extension (case insensitive), listed under the non-auto-detecting directives, defaulting to LINUX.

LINUX

`LINUX` *`image`*

Load *image* as a Linux-like kernel. MEMDISK is an example of a non-Linux kernel loaded in a Linux-like fashion.

BOOT

`BOOT` *`image`*

`[ISOLINUX only: .bin; SYSLINUX only: .bs]` Load a boot sector. `.bin` is a "CD boot sector" and `.bs` is a regular disk boot sector.

BSS

`BSS` *`image`*

`[SYSLINUX only: .bss]` Load a BSS image, a `.bs` image with the DOS superblock patched in.

COMBOOT

`COMBOOT` *`image`*

`[-4.xx; .com, .cbt]` Load a Syslinux COMBOOT image. `.com` images may also be runnable from DOS while `.cbt` images are not. See also doc/comboot.txt.

COM32

`COM32` *`image`*

`[.c32]` Load a Syslinux COM32 (32-bit COMBOOT and/or ELF) image. See also doc/comboot.txt, comboot_API , Category:Modules.

FDIMAGE

`FDIMAGE` *`image`*

`[1.65-4.05; ISOLINUX only: .img]` Load a disk image.

PXE

`PXE` *`image`*

`[PXELINUX only: .0]` Load a PXE NBP (Network Boot Program) image. The PXE protocol does not provide any means for specifiying or using a command line or initrd. See also Pxechn.c32.

CONFIG

`CONFIG` *`config_file`* `[`*`new_WD`*`]`

Load a new configuration file. The new configuration file is read, the Working Directory is optionally changed (if specified via an optional second parameter), and then the new configuration file is parsed.

If *new_WD* is not specified, then the Current Working Directory is maintained, unchanged.

The Working Directory may be different from the path to the config file.

Load new configuration file :

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

Set Syslinux's new home directory to "`/path/to/new/base/dir`" and load new configuration 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>
```

See also #Working directory.

LOCALBOOT

`LOCALBOOT` *`type`*

`[PXELINUX 1.53+; ISOLINUX 3.10+; SYSLINUX 3.70+]`
Attempt a different local boot method. Specifying LOCALBOOT instead of a KERNEL option means that invoking this particular label will cause a local disk boot instead of booting a kernel. Values other than those documented may produce undesired results.

type `-1` (minus one) : Cause the boot loader to report failure to the BIOS, which, on recent BIOSes, should mean that the next boot device in the boot sequence should be activated.

`[PXELINUX]` *type* `0` (zero) : Perform a normal local boot.

`[PXELINUX]` *type* `4` : Perform a local boot with the Universal Network Driver Interface (UNDI) driver still resident in memory.

`[PXELINUX]` *type* `5` : Perform a local boot with the entire PXE stack, including the UNDI driver, still resident in memory.

If you do not know what the UNDI or PXE stacks are, don't worry -- you don't want them, just specify `0` (zero).

`[ISOLINUX/SYSLINUX]` The *type* specifies the local drive number to boot from; `0x00` is the primary floppy drive and `0x80` is the primary hard drive.

LABEL-ONLY DIRECTIVES

INITRD

`INITRD` *`initrd_file`*

`[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.

The features of the "`initrd=`" parameter are also valid for the INITRD directive.

The "`initrd=`" parameter supports multiple filenames separated by commas (i.e. "`initrd=initrd_file1,initrd_file2`") within a single instance. This is mostly useful for initramfs, which can be composed of multiple separate cpio or cpio.gz archives.

Note: all initrd files except the last one are zero-padded to a 4K page boundary. This should not affect initramfs.

Note: Only the last effective "`initrd=`" parameter is used for loading initrd files.

See also linux.c32.

GLOBAL DIRECTIVES - SECONDARY

These are global directives that are of lesser importance, often affecting the user experience and not the boot process.

SERIAL

`SERIAL` *`port`* `[`*`baudrate`* `[`*`flowcontrol`*`]]`

Enable 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 be 8 bits, no parity, 1 stop bit.

flowcontrol is a combination of the following bits:

```
 1      - 0x001 - Assert DTR
 2      - 0x002 - Assert RTS
 8      - 0x008 - Enable interrupts
 16     - 0x010 - Wait for CTS assertion
 32     - 0x020 - Wait for DSR assertion
 64     - 0x040 - Wait for RI assertion
 128    - 0x080 - Wait for DCD assertion
 256    - 0x100 - Ignore input unless CTS asserted
 512    - 0x200 - Ignore input unless DSR asserted
 1024   - 0x400 - Ignore input unless RI asserted
 2048   - 0x800 - Ignore input unless DCD asserted
```

All other bits are reserved.

Typical values are:

```
0      -     0 - No flow control (default)
771    - 0x303 - Null modem cable detect
19     - 0x013 - RTS/CTS flow control
2067   - 0x813 - RTS/CTS flow control, modem input
35     - 0x023 - DTR/DSR flow control
131    - 0x083 - DTR/DCD flow control
```

For the SERIAL directive to be guaranteed to work properly, it should be the first directive in the configuration file.

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

Enabling interrupts (by setting the `0x008` bit) may give better responsiveness without setting the NOHALT option, but could potentially cause problems with buggy BIOSes.

This option is "sticky" and it is not automatically reset when loading a new configuration file with the CONFIG command; the SERIAL directive would need to be explicitly used so to change (or reset) its prior values.

See also Common_Problems#Serial.

NOHALT

`NOHALT` *`flag_val`*

If *flag_val* is `1`, do not halt the processor while idle. Halting the processor while idle significantly reduces the power consumption, but can cause poor responsiveness to the serial console, especially when using scripts to drive the serial console, as opposed to human interaction.

CONSOLE

`CONSOLE` *`flag_val`*

If *flag_val* is `0`, disable output to the normal video console. If *flag_val* is `1` (default), enable output to the video console.

Some BIOSes try to forward this to the serial console and sometimes make a total mess thereof, so this option lets you disable the video console on these systems.

SENDCOOKIES

`SENDCOOKIES` *`bitmask`*

`[PXELINUX 5.10+]` When downloading files over http, the SYSAPPEND strings are prepended with "`_Syslinux_`" (the word "Syslinux" in between one underscore character on each side) and sent to the server as cookies. The cookies are URL-encoded; whitespace is not replaced with underscore.

This command limits the cookies to be sent; `0` (zero) means no cookies. The default is "`-1`" (minus one), meaning "send all cookies".

This option is "sticky" and it is not automatically reset when loading a new configuration file with the CONFIG command.

PATH

`PATH` *`mylibpath`*

`[5.11+]` Specify a space-separated list of directories to be searched when attempting to load modules. This directive is useful for specifying the directories containing the "lib*.c32" library files, as other modules may depend on these files but might not reside in the same directory. Multiple instances will append additional paths.

`[5.00-5.10]` The list separator was previously a colon ":" character.

See also Directives/path, #Working directory.

TIMEOUT

`TIMEOUT` *`timeout`*

If more than one label entry is available, this directive indicates how long to wait at the "`boot:`" prompt until booting automatically, in units of 1/10 s.

The timeout is cancelled as soon as the user types anything on the keyboard; the assumption being that the user will complete the command line already begun.

The timer is reset to *timeout* upon return from an unsuccessful attempt to boot or from a module. A *timeout* of zero (the default) will disable the timeout completely.

Note: The maximum possible *timeout* value is 35996 (just under an hour).

When only one label entry is available, the initial *timeout* is ignored. To avoid automatically and immediately booting the (only and default) entry, use "escape" keys while booting, or add "`PROMPT 1`" to the configuration file, or add at least one additional label entry.

TOTALTIMEOUT

`TOTALTIMEOUT` *`timeout`*

Indicate 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 of situations. A *timeout* of zero (the default) will disable the timeout completely.

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

`ONTIMEOUT` *`kernel options...`*

Set the command line (which often references a LABEL) to be invoked on *timeout*. If not specified, then UI (if present) or DEFAULT is used.

ALLOWOPTIONS

`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`.

IMPLICIT

`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`.

NOCOMPLETE

`NOCOMPLETE` *`flag_val`*

If *flag_val* is `1`, the [Tab] key does not display labels at the "`boot:`" prompt.

NOESCAPE

`NOESCAPE` *`flag_val`*

If *flag_val* is `1`, ignore the [Shift] / [Alt] / [Caps Lock] / [Scroll Lock] "escape" keys. Use this (together with "`PROMPT 0`") to force the default boot alternative.

ONERROR

`ONERROR` *`kernel options...`*

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

```
ONERROR xyzzy plugh
```

... and the (faulty) command line as entered by the user was:

```
foo bar baz
```

... then Syslinux will execute the following command as if it had been entered by the user:

```
xyzzy plugh foo bar baz
```

PROMPT

PROMPT *flag_val*

If *flag_val* is `0` (default), do not display the "`boot:`" prompt, unless either the [Shift] or the [Alt] key is pressed, or unless either [Caps Lock] or [Scroll Lock] is set. If *flag_val* is `1`, always display the "`boot:`" prompt.

KBDMAP

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 "`doc/keytab-lilo.txt`" contains the documentation for this program.

Syslinux also ships a 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 configuration file.

FONT

FONT *filename*

Load a font in ".psf" format before displaying any output (except the copyright line, which is output as soon as the very first step of the boot loader 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 should do nothing on others.

SAY

SAY *message*

Print the *message* on the screen. Multiple SAY directives are allowed.

DISPLAY

DISPLAY *filename*

Display the indicated file on the screen at boot time (before the "`boot:`" prompt, if this is displayed). Please see the section below on DISPLAY files.

Note: If the file is missing, this option is simply ignored.

F1..F12

```
F1 textfile [background]

...

F12 textfile [background]
```

Display 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 on DISPLAY files.

See also Menu#F1..F12.

When using the serial console, press *[Ctrl-F][digit]* to get to the help screens, e.g. [Ctrl-F][2] to get to the F2 screen. For F10-F12, hit [Ctrl-F][A], [Ctrl-F][B], [Ctrl-F][C]. For compatibility with earlier versions, F10 can also be entered as [Ctrl-F][0].

DISPLAY file format

DISPLAY and function-key help files are text files in either DOS or UNIX format (i.e. with or without <CR>).
See also IsoLinux Mate to help with writing DISPLAY files under Windows (or under WINE (https://www.winehq.org/)).

In addition, the following special codes are interpreted:

Clear the screen and 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
```

Set the display colors to the specified background and foreground colors, where *<bg>* and *<fg>* are the 2 hex digits representing 1 byte, 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.

See also Examples/message file.

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 we are currently in graphics mode, return to text mode.

Output printing modes

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

These codes can be used to select in which modes to print a certain part of the message file. Each of these control characters selects 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, the following will actually print out which mode the console is in:

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

End of file

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

End of file (DOS convention).

Beep

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

Beep the speaker.

Retrieved from "http://www.syslinux.org/wiki/index.php?title=Config&oldid=4470"

Categories: Configuration | Directives | Examples

- This page was last modified on 19 March 2016, at 09:14.
- This page has been accessed 3,050 times.