

[PATCH] Remove boot98

Source: <http://linux.derkeiler.com/Mailing-Lists/Kernel/2004-07/5349.html>

From: Brian Gerst (bgerst_at_quark.didntduck.org)

Date: 07/28/04

Date: Tue, 27 Jul 2004 22:33:00 -0400

To: Andrew Morton <akpm@osdl.org>

Remove the orphaned PC9800 boot code.

--

Brian Gerst

```
diff -urN linux-2.6.8-rc2-bk/arch/i386/boot98/compressed/head.S
linux/arch/i386/boot98/compressed/head.S
--- linux-2.6.8-rc2-bk/arch/i386/boot98/compressed/head.S 2003-12-17 21:59:19.000000000 -0500
+++ linux/arch/i386/boot98/compressed/head.S 1969-12-31 19:00:00.000000000 -0500
@@ -1,128 +0,0 @@
-/*
- * linux/boot/head.S
- *
- * Copyright (C) 1991, 1992, 1993 Linus Torvalds
- */
-
-/*
- * head.S contains the 32-bit startup code.
- *
- * NOTE!!! Startup happens at absolute address 0x00001000, which is also where
- * the page directory will exist. The startup code will be overwritten by
- * the page directory. [According to comments etc elsewhere on a compressed
- * kernel it will end up at 0x1000 + 1Mb I hope so as I assume this. - AC]
- *
- * Page 0 is deliberately kept safe, since System Management Mode code in
- * laptops may need to access the BIOS data stored there. This is also
- * useful for future device drivers that either access the BIOS via VM86
- * mode.
- */
-
-/*
- * High loaded stuff by Hans Lermen & Werner Almesberger, Feb. 1996
- */
-.text
-
-#include <linux/linkage.h>
```

Linux-Kernel: [PATCH] Remove boot98

```
–#include <asm/segment.h>
–
– .globl startup_32
–
–startup_32:
– cld
– cli
– movl $(__BOOT_DS),%eax
– movl %eax,%ds
– movl %eax,%es
– movl %eax,%fs
– movl %eax,%gs
–
– lss stack_start,%esp
– xorl %eax,%eax
–1: incl %eax # check that A20 really IS enabled
– movl %eax,0x000000 # loop forever if it isn't
– cmpl %eax,0x100000
– je 1b
–
–/*
– * Initialize eflags. Some BIOS's leave bits like NT set. This would
– * confuse the debugger if this code is traced.
– * XXX – best to initialize before switching to protected mode.
– */
– pushl $0
– popfl
–/*
– * Clear BSS
– */
– xorl %eax,%eax
– movl $_edata,%edi
– movl $_end,%ecx
– subl %edi,%ecx
– cld
– rep
– stosb
–/*
– * Do the decompression, and jump to the new kernel..
– */
– subl $16,%esp # place for structure on the stack
– movl %esp,%eax
– pushl %esi # real mode pointer as second arg
– pushl %eax # address of structure as first arg
– call decompress_kernel
– orl %eax,%eax
– jnz 3f
– popl %esi # discard address
– popl %esi # real mode pointer
– xorl %ebx,%ebx
– ljmp $(__BOOT_CS), $0x100000
```

Linux-Kernel: [PATCH] Remove boot98

```
-
-/*
- * We come here, if we were loaded high.
- * We need to move the move-in-place routine down to 0x1000
- * and then start it with the buffer addresses in registers,
- * which we got from the stack.
- */
-3:
- movl $move_routine_start,%esi
- movl $0x1000,%edi
- movl $move_routine_end,%ecx
- subl %esi,%ecx
- addl $3,%ecx
- shrl $2,%ecx
- cld
- rep
- movsl
-
- popl %esi # discard the address
- popl %ebx # real mode pointer
- popl %esi # low_buffer_start
- popl %ecx # lcount
- popl %edx # high_buffer_start
- popl %eax # hcount
- movl $0x100000,%edi
- cli # make sure we don't get interrupted
- ljmp $(__BOOT_CS), $0x1000 # and jump to the move routine
-
-/*
- * Routine (template) for moving the decompressed kernel in place,
- * if we were high loaded. This _must_ PIC-code !
- */
-move_routine_start:
- movl %ecx,%ebp
- shrl $2,%ecx
- rep
- movsl
- movl %ebp,%ecx
- andl $3,%ecx
- rep
- movsb
- movl %edx,%esi
- movl %eax,%ecx # NOTE: rep movsb won't move if %ecx == 0
- addl $3,%ecx
- shrl $2,%ecx
- rep
- movsl
- movl %ebx,%esi # Restore setup pointer
- xorl %ebx,%ebx
- ljmp $(__BOOT_CS), $0x100000
-move_routine_end:
```

Linux-Kernel: [PATCH] Remove boot98

```
diff -urN linux-2.6.8-rc2-bk/arch/i386/boot98/compressed/Makefile
linux/arch/i386/boot98/compressed/Makefile
--- linux-2.6.8-rc2-bk/arch/i386/boot98/compressed/Makefile 2003-12-17 21:58:45.000000000 -0500
+++ linux/arch/i386/boot98/compressed/Makefile 1969-12-31 19:00:00.000000000 -0500
@@ -1,25 +0,0 @@
-#
-# linux/arch/i386/boot/compressed/Makefile
-#
-# create a compressed vmlinux image from the original vmlinux
-#
-
-# targets := vmlinux vmlinux.bin vmlinux.bin.gz head.o misc.o piggy.o
-# EXTRA_AFLAGS := -traditional
-
-# LDFLAGS_vmlinux := -Ttext $(IMAGE_OFFSET) -e startup_32
-
-$(obj)/vmlinux: $(obj)/head.o $(obj)/misc.o $(obj)/piggy.o FORCE
- $(call if_changed,ld)
- @:
-
-$(obj)/vmlinux.bin: vmlinux FORCE
- $(call if_changed,objcopy)
-
-$(obj)/vmlinux.bin.gz: $(obj)/vmlinux.bin FORCE
- $(call if_changed,gzip)
-
-# LDFLAGS_piggy.o := -r --format binary --oformat elf32-i386 -T
-
-$(obj)/piggy.o: $(obj)/vmlinux.scr $(obj)/vmlinux.bin.gz FORCE
- $(call if_changed,ld)
diff -urN linux-2.6.8-rc2-bk/arch/i386/boot98/compressed/misc.c
linux/arch/i386/boot98/compressed/misc.c
--- linux-2.6.8-rc2-bk/arch/i386/boot98/compressed/misc.c 2003-12-17 21:58:28.000000000 -0500
+++ linux/arch/i386/boot98/compressed/misc.c 1969-12-31 19:00:00.000000000 -0500
@@ -1,379 +0,0 @@
-/*
- * misc.c
- *
- * This is a collection of several routines from gzip-1.0.3
- * adapted for Linux.
- *
- * malloc by Hannu Savolainen 1993 and Matthias Urlichs 1994
- * puts by Nick Holloway 1993, better puts by Martin Mares 1995
- * High loaded stuff by Hans Lermen & Werner Almesberger, Feb. 1996
- */
-
-#include <linux/linkage.h>
-#include <linux/vmalloc.h>
-#include <linux/tty.h>
-#include <asm/io.h>
-#ifdef STANDARD_MEMORY_BIOS_CALL
```

Linux-Kernel: [PATCH] Remove boot98

```

-#undef STANDARD_MEMORY_BIOS_CALL
-#endif
-
-/*
- * gzip declarations
- */
-
-#define OF(args) args
-#define STATIC static
-
-#undef memset
-#undef memcpy
-
-/*
- * Why do we do this? Don't ask me..
- *
- * Incomprehensible are the ways of bootloaders.
- */
-static void* memset(void *, int, size_t);
-static void* memcpy(void *, __const void *, size_t);
-#define memzero(s, n) memset ((s), 0, (n))
-
-typedef unsigned char uch;
-typedef unsigned short ush;
-typedef unsigned long ulg;
-
-#define WSIZE 0x8000 /* Window size must be at least 32k, */
- /* and a power of two */
-
-static uch *inbuf; /* input buffer */
-static uch window[WSIZE]; /* Sliding window buffer */
-
-static unsigned insize = 0; /* valid bytes in inbuf */
-static unsigned inptr = 0; /* index of next byte to be processed in inbuf */
-static unsigned outcnt = 0; /* bytes in output buffer */
-
-/* gzip flag byte */
-#define ASCII_FLAG 0x01 /* bit 0 set: file probably ASCII text */
-#define CONTINUATION 0x02 /* bit 1 set: continuation of multi-part gzip file */
-#define EXTRA_FIELD 0x04 /* bit 2 set: extra field present */
-#define ORIG_NAME 0x08 /* bit 3 set: original file name present */
-#define COMMENT 0x10 /* bit 4 set: file comment present */
-#define ENCRYPTED 0x20 /* bit 5 set: file is encrypted */
-#define RESERVED 0xC0 /* bit 6,7: reserved */
-
-#define get_byte() (inptr < insize ? inbuf[inptr++] : fill_inbuf())
-
-/* Diagnostic functions */
-#ifdef DEBUG
-# define Assert(cond,msg) {if(!(cond)) error(msg);}
-# define Trace(x) fprintf x

```

Linux-Kernel: [PATCH] Remove boot98

```

-# define Tracev(x) {if (verbose) fprintf x ;}
-# define Tracevv(x) {if (verbose>1) fprintf x ;}
-# define Tracec(c,x) {if (verbose && (c)) fprintf x ;}
-# define Tracecv(c,x) {if (verbose>1 && (c)) fprintf x ;}
-#else
-# define Assert(cond,msg)
-# define Trace(x)
-# define Tracev(x)
-# define Tracevv(x)
-# define Tracec(c,x)
-# define Tracecv(c,x)
-#endif
-
-static int fill_inbuf(void);
-static void flush_window(void);
-static void error(char *m);
-static void gzip_mark(void **);
-static void gzip_release(void **);
-
-/*
- * This is set up by the setup-routine at boot-time
- */
-static unsigned char *real_mode; /* Pointer to real-mode data */
-
-#define EXT_MEM_K (*(unsigned short *)(real_mode + 0x2))
-#ifndef STANDARD_MEMORY_BIOS_CALL
-#define ALT_MEM_K (*(unsigned long *)(real_mode + 0x1e0))
-#endif
-#define SCREEN_INFO (*(struct screen_info *)(real_mode+0))
-
-extern char input_data[];
-extern int input_len;
-
-static long bytes_out = 0;
-static uch *output_data;
-static unsigned long output_ptr = 0;
-
-static void *malloc(int size);
-static void free(void *where);
-
-static void puts(const char *);
-
-extern int end;
-static long free_mem_ptr = (long)&end;
-static long free_mem_end_ptr;
-
-#define INPLACE_MOVE_ROUTINE 0x1000
-#define LOW_BUFFER_START 0x2000
-#define LOW_BUFFER_MAX 0x90000
-#define HEAP_SIZE 0x3000
-static unsigned int low_buffer_end, low_buffer_size;
```

Linux-Kernel: [PATCH] Remove boot98

```
-static int high_loaded =0;
-static uch *high_buffer_start /* = (uch *)(((ulg)&end) + HEAP_SIZE)*/;
-
-static char *vidmem = (char *)0xa0000;
-static int lines, cols;
-
-#ifdef CONFIG_X86_NUMAQ
-static void * xquad_portio = NULL;
-#endif
-
-#include "../lib/inflate.c"
-
-static void *malloc(int size)
-{
- void *p;
-
- if (size <0) error("Malloc error");
- if (free_mem_ptr <= 0) error("Memory error");
-
- free_mem_ptr = (free_mem_ptr + 3) & ~3; /* Align */
-
- p = (void *)free_mem_ptr;
- free_mem_ptr += size;
-
- if (free_mem_ptr >= free_mem_end_ptr)
- error("Out of memory");
-
- return p;
-}
-
-static void free(void *where)
-{ /* Don't care */
-}
-
-static void gzip_mark(void **ptr)
-{
- *ptr = (void *) free_mem_ptr;
-}
-
-static void gzip_release(void **ptr)
-{
- free_mem_ptr = (long) *ptr;
-}
-
-static void scroll(void)
-{
- int i;
-
- memcpy ( vidmem, vidmem + cols * 2, ( lines - 1 ) * cols * 2 );
- for ( i = ( lines - 1 ) * cols * 2; i < lines * cols * 2; i += 2 )
- vidmem[i] = ' ';
```

```

-}
-
-static void puts(const char *s)
-{
- int x,y,pos;
- char c;
-
- x = SCREEN_INFO.orig_x;
- y = SCREEN_INFO.orig_y;
-
- while ( ( c = *s++ ) != '\0' ) {
- if ( c == '\n' ) {
- x = 0;
- if ( ++y >= lines ) {
- scroll();
- y--;
- }
- } else {
- vidmem [ ( x + cols * y ) * 2 ] = c;
- if ( ++x >= cols ) {
- x = 0;
- if ( ++y >= lines ) {
- scroll();
- y--;
- }
- }
- }
- }
-
- SCREEN_INFO.orig_x = x;
- SCREEN_INFO.orig_y = y;
-
- pos = x + cols * y; /* Update cursor position */
- while (!(inb_p(0x60) & 4));
- outb_p(0x49, 0x62);
- outb_p(pos & 0xff, 0x60);
- outb_p((pos >> 8) & 0xff, 0x60);
-}
-
-static void* memset(void* s, int c, size_t n)
-{
- int i;
- char *ss = (char*)s;
-
- for (i=0;i<n;i++) ss[i] = c;
- return s;
-}
-
-static void* memcpy(void* __dest, __const void* __src,
- size_t __n)
-{

```

Linux-Kernel: [PATCH] Remove boot98

```

- int i;
- char *d = (char *)__dest, *s = (char *)__src;
-
- for (i=0;i<__n;i++) d[i] = s[i];
- return __dest;
-}
-
-/* =====
- * Fill the input buffer. This is called only when the buffer is empty
- * and at least one byte is really needed.
- */
-static int fill_inbuf(void)
-{
- if (insize != 0) {
- error("ran out of input data");
- }
-
- inbuf = input_data;
- insize = input_len;
- inptr = 1;
- return inbuf[0];
-}
-
-/* =====
- * Write the output window window[0..outcnt-1] and update crc and bytes_out.
- * (Used for the decompressed data only.)
- */
-static void flush_window_low(void)
-{
- ulg c = crc; /* temporary variable */
- unsigned n;
- uch *in, *out, ch;
-
- in = window;
- out = &output_data[output_ptr];
- for (n = 0; n < outcnt; n++) {
- ch = *out++ = *in++;
- c = crc_32_tab[((int)c ^ ch) & 0xff] ^ (c >> 8);
- }
- crc = c;
- bytes_out += (ulg)outcnt;
- output_ptr += (ulg)outcnt;
- outcnt = 0;
-}
-
-static void flush_window_high(void)
-{
- ulg c = crc; /* temporary variable */
- unsigned n;
- uch *in, ch;
- in = window;

```

Linux-Kernel: [PATCH] Remove boot98

```
- for (n = 0; n < outcnt; n++) {
- ch = *output_data++ = *in++;
- if ((ulg)output_data == low_buffer_end) output_data=high_buffer_start;
- c = crc_32_tab[((int)c ^ ch) & 0xff] ^ (c >> 8);
- }
- crc = c;
- bytes_out += (ulg)outcnt;
- outcnt = 0;
-}
-
-static void flush_window(void)
-{
- if (high_loaded) flush_window_high();
- else flush_window_low();
-}
-
-static void error(char *x)
-{
- puts("\n\n");
- puts(x);
- puts("\n\n -- System halted");
-
- while(1); /* Halt */
-}
-
-#define STACK_SIZE (4096)
-
-long user_stack [STACK_SIZE];
-
-struct {
- long * a;
- short b;
- } stack_start = { & user_stack [STACK_SIZE] , __BOOT_DS };
-
-static void setup_normal_output_buffer(void)
-{
-#ifdef STANDARD_MEMORY_BIOS_CALL
- if (EXT_MEM_K < 1024) error("Less than 2MB of memory");
-#else
- if ((ALT_MEM_K > EXT_MEM_K ? ALT_MEM_K : EXT_MEM_K) < 1024) error("Less than 2MB of
memory");
-#endif
- output_data = (char *)0x100000; /* Points to 1M */
- free_mem_end_ptr = (long)real_mode;
-}
-
-struct moveparams {
- uch *low_buffer_start; int lcount;
- uch *high_buffer_start; int hcount;
-};
-
```

Linux-Kernel: [PATCH] Remove boot98

```
-static void setup_output_buffer_if_we_run_high(struct moveparams *mv)
-{
- high_buffer_start = (uch *)(((ulg)&end) + HEAP_SIZE);
-#ifdef STANDARD_MEMORY_BIOS_CALL
- if (EXT_MEM_K < (3*1024)) error("Less than 4MB of memory");
-#else
- if ((ALT_MEM_K > EXT_MEM_K ? ALT_MEM_K : EXT_MEM_K) < (3*1024)) error("Less than 4MB
of memory");
-#endif
- mv->low_buffer_start = output_data = (char *)LOW_BUFFER_START;
- low_buffer_end = ((unsigned int)real_mode > LOW_BUFFER_MAX
- ? LOW_BUFFER_MAX : (unsigned int)real_mode) & ~0xfff;
- low_buffer_size = low_buffer_end - LOW_BUFFER_START;
- high_loaded = 1;
- free_mem_end_ptr = (long)high_buffer_start;
- if ( (0x100000 + low_buffer_size) > ((ulg)high_buffer_start) ) {
- high_buffer_start = (uch *) (0x100000 + low_buffer_size);
- mv->hcount = 0; /* say: we need not to move high_buffer */
- }
- else mv->hcount = -1;
- mv->high_buffer_start = high_buffer_start;
-}
-
-static void close_output_buffer_if_we_run_high(struct moveparams *mv)
-{
- if (bytes_out > low_buffer_size) {
- mv->lcount = low_buffer_size;
- if (mv->hcount)
- mv->hcount = bytes_out - low_buffer_size;
- } else {
- mv->lcount = bytes_out;
- mv->hcount = 0;
- }
-}
-
-asm linkage int decompress_kernel(struct moveparams *mv, void *rmode)
-{
- real_mode = rmode;
-
- vidmem = (char *)(((unsigned int)SCREEN_INFO.orig_video_page) << 4);
-
- lines = SCREEN_INFO.orig_video_lines;
- cols = SCREEN_INFO.orig_video_cols;
-
- if (free_mem_ptr < 0x100000) setup_normal_output_buffer();
- else setup_output_buffer_if_we_run_high(mv);
-
- makecrc();
- puts("Uncompressing Linux... ");
- gunzip();
}
```

Linux-Kernel: [PATCH] Remove boot98

```
- puts("Ok, booting the kernel.\n");
- if (high_loaded) close_output_buffer_if_we_run_high(mv);
- return high_loaded;
-}
-
-/* We don't actually check for stack overflows this early. */
-__asm__(".globl mcount ; mcount: ret\n");
-
diff -urN linux-2.6.8-rc2-bk/arch/i386/boot98/compressed/vmlinux.scr
linux/arch/i386/boot98/compressed/vmlinux.scr
--- linux-2.6.8-rc2-bk/arch/i386/boot98/compressed/vmlinux.scr 2003-12-17 21:58:48.000000000 -0500
+++ linux/arch/i386/boot98/compressed/vmlinux.scr 1969-12-31 19:00:00.000000000 -0500
@@ -1,9 +0,0 @@
-SECTIONS
-{
- .data : {
- input_len = .;
- LONG(input_data_end - input_data) input_data = .;
- *(.data)
- input_data_end = .;
- }
-}
diff -urN linux-2.6.8-rc2-bk/arch/i386/boot98/tools/build.c linux/arch/i386/boot98/tools/build.c
--- linux-2.6.8-rc2-bk/arch/i386/boot98/tools/build.c 2003-12-17 21:59:06.000000000 -0500
+++ linux/arch/i386/boot98/tools/build.c 1969-12-31 19:00:00.000000000 -0500
@@ -1,185 +0,0 @@
-/*
- * $Id: build.c,v 1.5 1997/05/19 12:29:58 mj Exp $
- *
- * Copyright (C) 1991, 1992 Linus Torvalds
- * Copyright (C) 1997 Martin Mares
- */
-
-/*
- * This file builds a disk-image from three different files:
- *
- * - bootsect: exactly 512 bytes of 8086 machine code, loads the rest
- * - setup: 8086 machine code, sets up system parm
- * - system: 80386 code for actual system
- *
- * It does some checking that all files are of the correct type, and
- * just writes the result to stdout, removing headers and padding to
- * the right amount. It also writes some system data to stderr.
- */
-
-/*
- * Changes by tytso to allow root device specification
- * High loaded stuff by Hans Lermen & Werner Almesberger, Feb. 1996
- * Cross compiling fixes by Gertjan van Wingerde, July 1996
- * Rewritten by Martin Mares, April 1997
- */
```

```

-
-#include <stdio.h>
-#include <string.h>
-#include <stdlib.h>
-#include <stdarg.h>
-#include <sys/types.h>
-#include <sys/stat.h>
-#include <sys/sysmacros.h>
-#include <unistd.h>
-#include <fcntl.h>
-#include <asm/boot.h>
-
-typedef unsigned char byte;
-typedef unsigned short word;
-typedef unsigned long u32;
-
-#define DEFAULT_MAJOR_ROOT 0
-#define DEFAULT_MINOR_ROOT 0
-
-/* Minimal number of setup sectors (see also bootsect.S) */
-#define SETUP_SECTS 4
-
-byte buf[1024];
-int fd;
-int is_big_kernel;
-
-void die(const char * str, ...)
-{
- va_list args;
- va_start(args, str);
- vfprintf(stderr, str, args);
- fputc('\n', stderr);
- exit(1);
-}
-
-void file_open(const char *name)
-{
- if ((fd = open(name, O_RDONLY, 0)) < 0)
- die("Unable to open `%s': %m", name);
-}
-
-void usage(void)
-{
- die("Usage: build [-b] bootsect setup system [rootdev] [> image]");
-}
-
-int main(int argc, char ** argv)
-{
- unsigned int i, c, sz, setup_sectors;
- u32 sys_size;
- byte major_root, minor_root;

```

Linux-Kernel: [PATCH] Remove boot98

```
- struct stat sb;
-
- if (argc > 2 && !strcmp(argv[1], "-b"))
- {
- is_big_kernel = 1;
- argc--, argv++;
- }
- if ((argc < 4) || (argc > 5))
- usage();
- if (argc > 4) {
- if (!strcmp(argv[4], "CURRENT")) {
- if (stat("/", &sb)) {
- perror("/");
- die("Couldn't stat /");
- }
- major_root = major(sb.st_dev);
- minor_root = minor(sb.st_dev);
- } else if (strcmp(argv[4], "FLOPPY")) {
- if (stat(argv[4], &sb)) {
- perror(argv[4]);
- die("Couldn't stat root device.");
- }
- major_root = major(sb.st_rdev);
- minor_root = minor(sb.st_rdev);
- } else {
- major_root = 0;
- minor_root = 0;
- }
- } else {
- major_root = DEFAULT_MAJOR_ROOT;
- minor_root = DEFAULT_MINOR_ROOT;
- }
- fprintf(stderr, "Root device is (%d, %d)\n", major_root, minor_root);
-
- file_open(argv[1]);
- i = read(fd, buf, sizeof(buf));
- fprintf(stderr, "Boot sector %d bytes.\n", i);
- if (i != 512)
- die("Boot block must be exactly 512 bytes");
- if (buf[510] != 0x55 || buf[511] != 0xaa)
- die("Boot block hasn't got boot flag (0xAA55)");
- buf[508] = minor_root;
- buf[509] = major_root;
- if (write(1, buf, 512) != 512)
- die("Write call failed");
- close (fd);
-
- file_open(argv[2]); /* Copy the setup code */
- for (i=0 ; (c=read(fd, buf, sizeof(buf)))>0 ; i+=c )
- if (write(1, buf, c) != c)
- die("Write call failed");
```

Linux-Kernel: [PATCH] Remove boot98

```
- if (c != 0)
- die("read-error on `setup");
- close (fd);
-
- setup_sectors = (i + 511) / 512; /* Pad unused space with zeros */
- if (!(setup_sectors & 1))
- setup_sectors++; /* setup_sectors must be odd on NEC PC-9800 */
- fprintf(stderr, "Setup is %d bytes.\n", i);
- memset(buf, 0, sizeof(buf));
- while (i < setup_sectors * 512) {
- c = setup_sectors * 512 - i;
- if (c > sizeof(buf))
- c = sizeof(buf);
- if (write(1, buf, c) != c)
- die("Write call failed");
- i += c;
- }
-
- file_open(argv[3]);
- if (fstat (fd, &sb))
- die("Unable to stat `%s': %m", argv[3]);
- sz = sb.st_size;
- fprintf (stderr, "System is %d kB\n", sz/1024);
- sys_size = (sz + 15) / 16;
- /* 0x40000*16 = 4.0 MB, reasonable estimate for the current maximum */
- if (sys_size > (is_big_kernel ? 0x40000 : DEF_SYSSIZE))
- die("System is too big. Try using %smodes.",
- is_big_kernel ? "" : "bzImage or ");
- while (sz > 0) {
- int l, n;
-
- l = (sz > sizeof(buf)) ? sizeof(buf) : sz;
- if ((n=read(fd, buf, l)) != l) {
- if (n < 0)
- die("Error reading %s: %m", argv[3]);
- else
- die("%s: Unexpected EOF", argv[3]);
- }
- if (write(1, buf, l) != l)
- die("Write failed");
- sz -= l;
- }
- close(fd);
-
- if (lseek(1, 497, SEEK_SET) != 497) /* Write sizes to the bootsector */
- die("Output: seek failed");
- buf[0] = setup_sectors;
- if (write(1, buf, 1) != 1)
- die("Write of setup sector count failed");
- if (lseek(1, 500, SEEK_SET) != 500)
- die("Output: seek failed");
```

Linux-Kernel: [PATCH] Remove boot98

```
- buf[0] = (sys_size & 0xff);  
- buf[1] = ((sys_size >> 8) & 0xff);  
- if (write(1, buf, 2) != 2)  
- die("Write of image length failed");  
-  
- return 0; /* Everything is OK */  
-}
```

-
To unsubscribe from this list: send the line "unsubscribe linux-kernel" in
the body of a message to majordomo@vger.kernel.org

More majordomo info at <http://vger.kernel.org/majordomo-info.html>

Please read the FAQ at <http://www.tux.org/lkml/>