Wyse SX0: Tiny Core

Tiny Core on the Sx0

As it is a neat thin client that I sometimes press into service I occasionally revisit the Sx0 and build an up-to-date Sx0 specific Tiny Core kernel for it. You will find the builds below. I must admit these haven't been extensively tested, but they do work for me. All you need to do is download the new kernel and add it to the **boot** directory on the Tiny Core USB pen drive (or whatever). You can either overwrite the existing **vmlinuz** file or add a new entry to the menu system to let you select the Wyse Sx0 specific kernel - an example of such a **syslinux** menu entry is below.

```
LABEL tcwyse
MENU LABEL TinyCore for the Wyse SX0 (on slow devices, waitusb=5)
TEXT HELP
Boot the Wyse Sx0 build of TinyCore with Embedded X/GUI extensions,
if using a slow device. Boot media is removable. Use TAB to edit
options for specific needs.
ENDTEXT
KERNEL /boot/vmlinuzSx0
INITRD /boot/core.gz
APPEND loglevel=3  kmap=qwerty/uk waitusb=5
```

Issues Addressed

- Edit the file **pata_cs5536.c** to fix the timing issue and to re-enable the IDE interface.
- Include the **pata_cs5535.c** driver to support the early version of the Sx0.
- Remove all IDE drivers other than CS5535 and CS5536 drivers.
- Remove the i8042 driver which crashes on initialisation.

You could further optimise the kernel, but my experience is that the more you remove the more likely it is that something breaks so I ended up leaving the rest alone.

In the March 2020 revisit for Tiny Core release 11.x I couldn't easily determine how to remove the i8042 driver. I dealt with the problem at the time by including the kernel boot parameters: **i8042.noaux** and **i8042.nokbd**. Since then I've built a replacement version without the offending driver removing the need for these parameters.

Downloads

All you really need is the rebuilt kernel for the version of Tiny Core that you're intending to use. If you have any interest in going further and building your own version then you've got the option of using my .config as a starting point along with the patched CS5336 driver. It is relatively straight forward to build a new kernel and the approach I used to build one is documented here.

Download the patched CS5536 file: pata_cs5336.c.

Tiny Core	kernel	.config
12.x	: vmlinuzSx0_12	configSx0_12

11.x	: vmlinuzSx0_11	configSx0_11
10.x	: vmlinuzSx0_10	configSx0_10
9.x	: vmlinuzSx0_9	configSx0_9
8.x	: vmlinuzSx0_8	configSx0_8
7.x	: vmlinuzSx0_7	configSx0_7
5.x	: vmlinuzSx0_5	configSx0_5

History

April 2021, Tiny Core 12.x: Worked out how to remove the i8042 driver removing the need for any specific kernel boot parameters.

March 2020, Tiny Core 11.x: I found that I couldn't easily remove the i8042 driver. The work-around was to include the kernel boot parameters: **i8042.noaux i8042.nokbd**. It has since been rebuilt without the i8042 driver.

May 2019: Tiny Core 10.x & 9.x:I found that the latest versions of Tiny Core using Linux kernels 4.14.10 (v9.x) and 4.19.10 (v10.x) did not run. With v9 the kernel crashed towards the end of the boot process, with v10 it was about half way through. I tracked this down to the i8042 driver which is the legacy PS/2 port driver. The CS5535 and CS5336 do emulate the i8042 but the Sx0 itself does not have any PS/2 ports, so my solution was to remove the driver from the build.

January 2018, Tiny Core 8.x: The timing problem during CS5536 initialisation appears to have returned so that mod is back in. Also, towards the end of the startup sequence, I'm seeing some delay with messages about floppy drives....

```
........
floppy0: no floppy controllers found
work still pending
Floppy drive(s): fd0 is 1.44M
floppy0: no floppy controllers found
work still pending
Floppy drive(s): fd0 is 1.44M
floppy0: no floppy controllers found
.......
```

...repeated a number of times. I don't know why this happens and I haven't spent much time looking into it. It does increase the startup time but otherwise doesn't appear to affect the system.

February 2017: Tiny Core 7.x: It looks like the CS5336 initialisation timing problem has been fixed.

May 2015: Tiny Core 5.x: First release. Added CS5335 driver and patched CS5336 driver to fix the timing problem and to enable the IDE interface.

CS5336 patches

CS5336 driver source file: **drivers/ata/pata_cs5536.c**

To fix the timing issue which can cause a lock-up during the CS5336 initialisation the following is added at the start of the functions following all the #includes.

```
/*
 * The mfgpt timer interrupt is running early, so we must keep the south bridge
 * mmio always enabled. Otherwise we may race with the PCI configuration which
 * may temporarily disable it. When that happens and the timer interrupt fires,
 * we are not able to clear it and the system will hang.
 */
static void cs5536_isa_mmio_always_on(struct pci_dev *dev)
{
        dev->mmio_always_on = 1;
}
DECLARE_PCI_FIXUP_CLASS_EARLY(PCI_VENDOR_ID_AMD, PCI_DEVICE_ID_AMD_CS5536_ISA,
        PCI_CLASS_BRIDGE_ISA, 8, cs5536_isa_mmio_always_on);
```

To override the BIOS disabling the IDE interface two lines are added to the function **cs5536_init_one()**.

```
.......
        if (use_msr)
                printk(KERN_ERR DRV_NAME ": Using MSR regs instead of PCI\n");

        cs5536_read(dev, CFG, &cfg);
        cfg |= IDE_CFG_CHANEN;              <<---ADD
        cs5536_write(dev, CFG, cfg);        <<---ADD

        if ((cfg & IDE_CFG_CHANEN) == 0) {
                printk(KERN_ERR DRV_NAME ": disabled by BIOS\n");
.......
```

Any comments? email me. Added May 2015 Last update April 2021