

Testing and Installing Kernel 2.6.22

Panelcard, Portux920T and Portux PanelPC and Molux

Kernel 2.6.22

Kernel 2.6.22 for taskit ARM products has now been released as a testing version. After a successful testing phase 2.6.22 will eventually become the release version for all products. In 2.6.22 some AT91 related drivers have been improved and updated: The serial driver is now completely DMA-driven and much more reliable. The new SD/MMC-card driver and framework supports SD-Cards over 1GB as well as MMCmobile cards. There is a new SPI-framework, support for Power-management and much more.

You will find a binary package, sources for the kernel and the new rootfilesystem available for download on our website.

Contents:

images-2.6.22.tgz	binary images needed for installing and testing
rootfs-source-2.6.22.tgz	sources for new packages in the rootfilesystem
linux-2.6.22-taskit2.tgz	linux kernel sources

Changes and Differences

As always some things have to be changed to accommodate for new features. One major change for users is, that device-filesystem and devfsd are no longer supported. That means, that the path of some device-nodes has changed. Devfsd is now replaced by busybox's internal mdev. The naming convention for the physically mapped flash has changed one more time. This affects the variable name used for the boot parameters.

The rootfilesystem has been reworked and contains now some new commands such as diff, top, mke2fs, mkdosfs and fdisk. Also there is now a libjpeg and the newer libz-1.2.3. Also the standard c++ library and libgcc_s.so have been included.

As the images have grown, we will change the standard flash partition again, but the new borders will yield for all our Linux related products:

Name	Startaddress	Endaddress	Size
Boot	0x10000000	0x1005ffff	384k
Linux	0x10060000	0x101ffffff	1664k
Initrd	0x10200000	0x104ffffff	3072k
Config	0x10500000	-	-

One change also includes that the password for the user root on the targetboards has been changed, it is now taskit instead of portux.

Testing the new kernel

The best way for testing the new kernel is to load the new kernel and rootfilesystem into ram and boot it from there. This has the advantage, that your already installed system will not be affected. As said earlier, the naming convention for the physically mapped flash has changed, so a new variable has to be defined in u-boot:

```
U-Boot>setenv mtdparts-2.6.22 mtdparts=physmap-
flash.0:384k(boot),1664k(linux),3072k(initrd),-(cfg)
U-Boot>saveenv
```

Be careful to use the partition borders from your previous installation, which may vary.

Now the variable for tftpbooting kernel and rootfilesystem can be set:

```
U-Boot>setenv ramboot-2.6.22 tftpboot 20000000 uImage-2.6.22\;tftpboot
\$(inaddr) rootfs-2.6.22\;setenv insize $(filesize)\;setenv bootargs
\$(basicargs) initrd=0x\$(inaddr),0x\$(insize) \$(mtdparts-2.6.22)\;bootm
20000000
U-Boot>saveenv
```

Now, after the images have been copied to the tftpboot directory of your development computer run ramboot-2.6.22 will boot the new kernel and rootfilesystem.

Installing the new kernel

As, at least for Portux920T and the Portux PanelPC, the partitions borders have changed, you will lose your flashfile system (/config). There are two ways of creating a backup: One is to copy the contents of your system to a NFS-mounted share and install the default image contained in the package and copy the contents back after installing or creating an Image by yourself and use this flashimage during the install process. You can create an image of your /config directory with this command:

```
root@Portux920T:# mkfs.jffs2 -d /config -e 128KiB -o /mnt/flash-xxx.img
```

where /mnt is a network-mounted share and flash-xxx.img is the name of your flash image.

During the install process we will erase the entire flash except u-boot and it's variables and then install the images uImage-2.6.22, rootfs-2.6.22 and flash-xxx.img on the filesystem and then adjust mtdparts and other variables. First erasing the flash:

```
U-Boot> erase 10060000 11ffffff
```

Now installing the images:

```
U-Boot> tftpboot 20000000 uImage-2.6.22
TFTP from server 192.168.4.238; our IP address is 192.168.4.241
Filename 'uImage-2.6.22'.
Load address: 0x20000000
Loading: #####
#####
#####
#####
#####
done
Bytes transferred = 1395872 (154ca0 hex)
U-Boot> cp.b 20000000 10060000 $(filesize)
```



```
Protected 2 sectors
```

```
U-Boot>boot
```

Compiling the new Kernel

All taskit products are now integrated in a single kernel source. To compile for your product you have to set some environment variables:

```
export ARCH=arm
export CROSS_COMPILE=arm-linux-3.4.2-
```

If you are tired of typing these environment variables every time you want to use the kernel source, you can add those two lines in your `/home/xyz/.profile` or `/home/xyz/.bashrc`.

Now you can change to the root directory of the kernel sources and configure for your product:

```
make portux920t_defconfig
make portuxpanelpc_defconfig
make panelcard_defconfig
make molux_defconfig
```

Now you can make the kernel like always:

```
make zImage
./install_linux.sh img_name
```

to produce the standard uImage. To configure your kernel issue a

```
make menuconfig
```