Maksymilian Baj
Adam Szokalski

SKPS - Laboratorium 1. - sprawozdanie

Przygotowanie stanowiska

Na początku zgodnie z instrukcją przygotowaliśmy stanowisko do pracy.

Uruchomienie RPi

Za pomocą programu *tio* podłączyliśmy się do terminala UART a następnie zalogowaliśmy się do systemu:

```
user@lab-31:~/skps24l_mbaj_aszokalski/cw1$ ls
user@lab-31:~/skps24l_mbaj_aszokalski/cw1$ tio /dev/ttyUSB0
[tio 12:58:12] tio v1.29
[tio 12:58:12] Press ctrl-t q to quit
[tio 12:58:12] Connected
reboot
Password:

Login incorrect
rescue login: root
# ls
adapter_test.sh
#
```

Przy użyciu polecenia *udhcpc* uruchomiliśmy protokół DHCP:

```
# udhcpc
udhcpc: started, v1.33.1
udhcpc: sending discover
udhcpc: sending select for 10.42.0.203
udhcpc: lease of 10.42.0.203 obtained, lease time 3600
deleting routers
adding dns 10.42.0.1
#
```

Urządzeniu RPi został przyznany adres IP 10.42.0.203:

```
# ifconfig | grep 10.42
          inet addr:10.42.0.203  Bcast:10.42.0.255  Mask:255.255.255.0
#
```

Natomiast host ma adres IP 10.42.0.1:

```
user@lab-31:~/skps24l_mbaj_aszokalski/cw1$ ifconfig | grep 10.42
        inet 10.42.0.1  netmask 255.255.255.0  broadcast 10.42.0.255
```

Przy pomocy polecenia *ping* sprawdziliśmy stan połączenia sieciowego:

ping rpi->host

```
# ping 10.42.0.1
PING 10.42.0.1 (10.42.0.1): 56 data bytes
64 bytes from 10.42.0.1: seq=0 ttl=64 time=0.934 ms
64 bytes from 10.42.0.1: seq=1 ttl=64 time=0.878 ms
64 bytes from 10.42.0.1: seq=2 ttl=64 time=0.867 ms
64 bytes from 10.42.0.1: seq=3 ttl=64 time=0.870 ms
```

ping host->rpi

```
# ping 10.42.0.203
PING 10.42.0.203 (10.42.0.203): 56 data bytes
64 bytes from 10.42.0.203: seq=0 ttl=64 time=0.237 ms
64 bytes from 10.42.0.203: seq=1 ttl=64 time=0.150 ms
```

Wszystkie pingi przechodzą, zatem urządzenia są ze sobą poprawnie połączone.

Kopiowanie plików na RPi

Na komputerze host postawiliśmy serwer HTTP za pomocą polecenia:

python3 -m http.server

```
user@lab-31:~/skps24l_mbaj_aszokalski/cw1$ python3 -m http.server
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
```

Za pomocą polecenia *wget* na terminalu RPi sprawdziliśmy poprawność serwera:

```
# wget http://10.42.0.1:8000/test.txt
--1970-01-01 00:17:21--  http://10.42.0.1:8000/test.txt
Connecting to 10.42.0.1:8000... connected.
HTTP request sent, awaiting response... 200 OK
Length: 5 [text/plain]
Saving to: 'test.txt'

test.txt            100%[===================>]       5  --.-KB/s    in 0s

1970-01-01 00:17:21 (151 KB/s) - 'test.txt' saved [5/5]

# cat test.txt
text
#
```

Z komputera hosta na RPi udało się przesłać plik *test.txt*, co potwierdza poprawne działanie serwera.

Konfiguracja Buildroot na komputerze hosta

Konfiguracja dla płytki RPi4:

Na początku wgraliśmy domyślną konfigurację dla urządzenia RPi4:

```
user@lab-31:~/skps24l_mbaj_aszokalski/cw1/buildroot-2021.08$ make raspberrypi4_64_defconfig
mkdir -p /home/user/skps24l_mbaj_aszokalski/cw1/buildroot-2021.08/output/build/buildroot-config/lxdialog
PKG_CONFIG_PATH="" make CC="/usr/bin/gcc" HOSTCC="/usr/bin/gcc" \
    obj=/home/user/skps24l_mbaj_aszokalski/cw1/buildroot-2021.08/output/build/buildroot-config -C support
/kconfig -f Makefile.br conf
/usr/bin/gcc -D_GNU_SOURCE -D_DEFAULT_SOURCE -DCURSES_LOC="<ncurses.h>" -DLOCALE  -I/home/user/skps24l_mb
aj_aszokalski/cw1/buildroot-2021.08/output/build/buildroot-config -DCONFIG_=\"\"  -MM *.c > /home/user/sk
ps24l_mbaj_aszokalski/cw1/buildroot-2021.08/output/build/buildroot-config/.depend 2>/dev/null || :
/usr/bin/gcc -D_GNU_SOURCE -D_DEFAULT_SOURCE -DCURSES_LOC="<ncurses.h>" -DLOCALE  -I/home/user/skps24l_mb
aj_aszokalski/cw1/buildroot-2021.08/output/build/buildroot-config -DCONFIG_=\"\"   -c conf.c -o /home/use
r/skps24l_mbaj_aszokalski/cw1/buildroot-2021.08/output/build/buildroot-config/conf.o
/usr/bin/gcc -D_GNU_SOURCE -D_DEFAULT_SOURCE -DCURSES_LOC="<ncurses.h>" -DLOCALE  -I/home/user/skps24l_mb
aj_aszokalski/cw1/buildroot-2021.08/output/build/buildroot-config -DCONFIG_=\"\"  -I. -c /home/user/skps2
4l_mbaj_aszokalski/cw1/buildroot-2021.08/output/build/buildroot-config/zconf.tab.c -o /home/user/skps24l_
mbaj_aszokalski/cw1/buildroot-2021.08/output/build/buildroot-config/zconf.tab.o
/usr/bin/gcc -D_GNU_SOURCE -D_DEFAULT_SOURCE -DCURSES_LOC="<ncurses.h>" -DLOCALE  -I/home/user/skps24l_mb
aj_aszokalski/cw1/buildroot-2021.08/output/build/buildroot-config -DCONFIG_=\"\"   /home/user/skps24l_mba
j_aszokalski/cw1/buildroot-2021.08/output/build/buildroot-config/conf.o /home/user/skps24l_mbaj_aszokalsk
i/cw1/buildroot-2021.08/output/build/buildroot-config/zconf.tab.o  -o /home/user/skps24l_mbaj_aszokalski/
cw1/buildroot-2021.08/output/build/buildroot-config/conf
rm /home/user/skps24l_mbaj_aszokalski/cw1/buildroot-2021.08/output/build/buildroot-config/zconf.tab.c
#
# configuration written to /home/user/skps24l_mbaj_aszokalski/cw1/buildroot-2021.08/.config
#
```

Włączenie ramdysku:

Następnie włączyliśmy opcję initramfs, kompresję obrazu i wyłączyliśmy ext2/3/4:

```
[ ] axfs root filesystem
[ ] btrfs root filesystem
[ ] cloop root filesystem for the target device
-*- cpio the root filesystem (for use as an initial RAM filesystem)
      Compression method (gzip)  --->
[ ]   Create U-Boot image of the root filesystem (NEW)
[ ] cramfs root filesystem
[ ] erofs root filesystem
[ ] ext2/3/4 root filesystem
[ ] f2fs root filesystem
[*] initial RAM filesystem linked into linux kernel
[ ] jffs2 root filesystem
[ ] romfs root filesystem
[ ] squashfs root filesystem
[ ] tar the root filesystem
[ ] ubi image containing an ubifs root filesystem
[ ] ubifs root filesystem
[ ] yaffs2 root filesystem
```

Rozszerzenie rozmiaru partycji:

W pliku */board/raspberrypi4-64/genimage-raspberrypi4-64.cfg* dokonaliśmy modyfikacji pola size. Ustawiliśmy wielkość partycji na 128 MB, żeby system na pewno się na niej zmieścił.

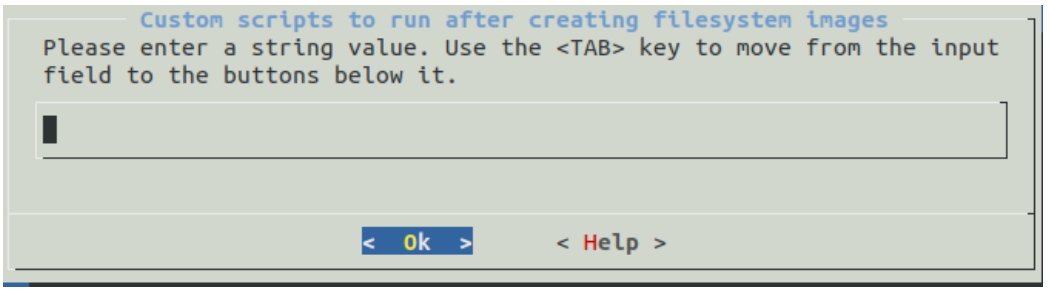
```
image boot.vfat {
  vfat {
    files = {
      "bcm2711-rpi-4-b.dtb",
      "rpi-firmware/cmdline.txt",
      "rpi-firmware/config.txt",
      "rpi-firmware/fixup.dat",
      "rpi-firmware/start.elf",
      "rpi-firmware/overlays",
      "Image"
    }
  }
  size = 128M
}
```

Zbudowanie obrazu Linuxa:

Za pomocą polecenia *make* zbudowaliśmy obraz Linuxa, jednakże pojawił się błąd dotyczący post-image skryptu:

```
>>>   Executing post-image script board/raspberrypi4-64/post-image.sh
Adding 'dtoverlay=miniuart-bt' to config.txt (fixes ttyAMA0 serial console).
board/raspberrypi4-64/genimage-raspberrypi4-64.cfg:31: no sub-section title/index for 'config'
INFO: cmd: "mkdir -p "/home/user/skps24l_mbaj_aszokalski/cw1/buildroot-2021.08/output/build/genimage.tmp"
" (stderr):
INFO: cmd: "rm -rf "/home/user/skps24l_mbaj_aszokalski/cw1/buildroot-2021.08/output/build/genimage.tmp"/*
" (stderr):
INFO: cmd: "mkdir -p "/home/user/skps24l_mbaj_aszokalski/cw1/buildroot-2021.08/output/build/genimage.tmp"
" (stderr):
INFO: cmd: "cp -a "/tmp/tmp.3aLJvIgTYV" "/home/user/skps24l_mbaj_aszokalski/cw1/buildroot-2021.08/output/
build/genimage.tmp/root"" (stderr):
INFO: cmd: "find '/home/user/skps24l_mbaj_aszokalski/cw1/buildroot-2021.08/output/build/genimage.tmp/root
' -depth -type d -printf '%P\0' | xargs -0 -I {} touch -r '/tmp/tmp.3aLJvIgTYV/{}' '/home/user/skps24l_mb
aj_aszokalski/cw1/buildroot-2021.08/output/build/genimage.tmp/root/{}'" (stderr):
ERROR: file(rootfs.ext4): stat(/home/user/skps24l_mbaj_aszokalski/cw1/buildroot-2021.08/output/images/roo
tfs.ext4) failed: No such file or directory
ERROR: hdimage(sdcard.img): could not setup rootfs.ext4
Makefile:821: recipe for target 'target-post-image' failed
make[1]: *** [target-post-image] Error 1
Makefile:84: recipe for target '_all' failed
make: *** [_all] Error 2
```

W celu pozbycia się błędu w menuconfig w system configuration zmodyfikowaliśmy pole dotyczące skryptu uruchamiającego się po stworzeniu obrazu:



Po ponownym uruchomieniu *make* udało się poprawnie zbudować obraz:

```
# Copy the kernel image(s) to its(their) final destination
/usr/bin/install -m 0644 -D /home/user/skps24l_mbaj_aszokalski/cw1/buildroot-2021.08/output/build/linux-c
ustom/arch/arm64/boot/Image /home/user/skps24l_mbaj_aszokalski/cw1/buildroot-2021.08/output/images/Image
# If there is a .ub file copy it to the final destination
test ! -f /home/user/skps24l_mbaj_aszokalski/cw1/buildroot-2021.08/output/build/linux-custom/arch/arm64/b
oot/Image.ub || cp /home/user/skps24l_mbaj_aszokalski/cw1/buildroot-2021.08/output/build/linux-custom/arc
h/arm64/boot/Image.ub /home/user/skps24l_mbaj_aszokalski/cw1/buildroot-2021.08/output/images
```

W folderze /output/images pojawił się obraz wraz z linią poleceń (w folderze rpi-firmware) i drzewem urządzeń:

```
user@lab-31:~/skps24l_mbaj_aszokalski/cw1/buildroot-2021.08/output/images$ ls
bcm2711-rpi-4-b.dtb  Image  rootfs.cpio  rootfs.cpio.gz  rpi-firmware
```

Uruchomienie zbudowanego obrazu

Zamontowanie partycji 1 karty SD w katalogu /mnt:

```
# mount /dev/mmcblk0p1 /mnt
[ 4016.661091] FAT-fs (mmcblk0p1): Volume was not properly unmounted. Some data may be corrupt. Please ru
n fsck.
#
```

Skopiowanie plików Image, cmdline.txt i bcm2711-rpi-4-b.dtb do katalogu /mnt/user:

```
# cd /mnt/user
# wget http://10.42.0.1:8000/buildroot-2021.08/output/images/Image
--1970-01-01 01:16:33--  http://10.42.0.1:8000/buildroot-2021.08/output/images/Image
Connecting to 10.42.0.1:8000... connected.
HTTP request sent, awaiting response... 200 OK
Length: 47213056 (45M) [application/octet-stream]
Saving to: 'Image'

Image               100%[===================>]  45.03M  11.2MB/s    in 4.0s

1970-01-01 01:16:37 (11.2 MB/s) - 'Image' saved [47213056/47213056]

# wget http://10.42.0.1:8000/buildroot-2021.08/output/images/bcm2711-rpi-4-b.dtb

--1970-01-01 01:16:48--  http://10.42.0.1:8000/buildroot-2021.08/output/images/bcm2711-rpi-4-b.dtb
Connecting to 10.42.0.1:8000... connected.
HTTP request sent, awaiting response... 200 OK
Length: 49749 (49K) [application/octet-stream]
Saving to: 'bcm2711-rpi-4-b.dtb'

bcm2711-rpi-4-b.dtb 100%[===================>]  48.58K  --.-KB/s    in 0.004s

1970-01-01 01:16:48 (13.2 MB/s) - 'bcm2711-rpi-4-b.dtb' saved [49749/49749]
```

```
# wget http://10.42.0.1:8000/buildroot-2021.08/output/images/rpi-firmware/cmdlin
e.txt
--1970-01-01 01:18:18--  http://10.42.0.1:8000/buildroot-2021.08/output/images/rpi-firmware/cmdline.txt
Connecting to 10.42.0.1:8000... connected.
HTTP request sent, awaiting response... 200 OK
Length: 65 [text/plain]
Saving to: 'cmdline.txt'

cmdline.txt         100%[===================>]      65  --.-KB/s    in 0s

1970-01-01 01:18:18 (1.88 MB/s) - 'cmdline.txt' saved [65/65]

# ls
Image               bcm2711-rpi-4-b.dtb  cmdline.txt
#
```

Zmiana nazwy pliku Image na kernel8.img:

```
# mv Image kernel8.img
# ls
bcm2711-rpi-4-b.dtb  cmdline.txt          kernel8.img
#
```

Restart RPi za pomocą polecenia reboot.

```
Welcome to Buildroot
buildroot login:
```

```
# uname -a
Linux buildroot 5.10.46-v8 #3 SMP PREEMPT Wed Mar 6 13:57:16 CET 2024 aarch64 GNU/Linux
#
```

Obraz dla Raspberry Pi 4B bez initramfs

Po usunięciu poprzedniego obrazu poleceniem *make linux-dirclean* dokonaliśmy w menuconfig następujące zmiany:

- wyłączyliśmy initial RAM filesystem
- włączyliśmy wsparcie dla ext2/3/4
- zwiększyliśmy rozmiar systemu do 128 MB

Opcja Toolchain --> Toolchain type pozostała ustawiona na *External toolchain*.

```
[ ] axfs root filesystem
[ ] btrfs root filesystem
[ ] cloop root filesystem for the target device
[ ] cpio the root filesystem (for use as an initial RAM filesystem)
[ ] cramfs root filesystem
[ ] erofs root filesystem
[*] ext2/3/4 root filesystem
      ext2/3/4 variant (ext2 (rev1))  --->
(rootfs) filesystem label
(128M) exact size
(0)   exact number of inodes (leave at 0 for auto calculation)
(5)   reserved blocks percentage
(-O ^64bit) additional mke2fs options
      Compression method (no compression)  --->
[ ] f2fs root filesystem
[ ] initial RAM filesystem linked into linux kernel
[ ] jffs2 root filesystem
[ ] romfs root filesystem
[ ] squashfs root filesystem
[ ] tar the root filesystem
[ ] ubi image containing an ubifs root filesystem
[ ] ubifs root filesystem
[ ] yaffs2 root filesystem
```

Efekt budowania komendą make

```
mke2fs 1.46.2 (28-Feb-2021)
Creating regular file /home/user/skps24l_mbaj_aszokalski/cw1/buildroot-2021.08/output/images/rootfs.ext2
Creating filesystem with 131072 1k blocks and 32768 inodes
Filesystem UUID: e4d7ed55-4ce9-4f97-b188-4b34ae14ca80
Superblock backups stored on blocks:
        8193, 24577, 40961, 57345, 73729

Allocating group tables: done
Writing inode tables: done
Copying files into the device: done
Writing superblocks and filesystem accounting information: done

user@lab-31:~/skps24l_mbaj_aszokalski/cw1/buildroot-2021.08$
```

Plik Image jest mniejszy, ponieważ nie zawiera on systemu plików

Kopiujemy te same pliki co wcześniej i rootfs.ext2

```
bcm2711-rpi-4-b.dtb  cmdline.txt          kernel8.img
# wget http://10.42.0.1:8000/buildroot-2021.08/output/images/rootfs.ext2
--1970-01-01 00:06:40--  http://10.42.0.1:8000/buildroot-2021.08/output/images/rootfs.ext2
Connecting to 10.42.0.1:8000... connected.
HTTP request sent, awaiting response... 200 OK
Length: 134217728 (128M) [application/octet-stream]
Saving to: 'rootfs.ext2'

rootfs.ext2         100%[===================>] 128.00M  11.2MB/s    in 11s

1970-01-01 00:06:52 (11.2 MB/s) - 'rootfs.ext2' saved [134217728/134217728]
```

Kopiowanie systemu plików na 2 partycję

```
# dd if=rootfs.ext2 of=/dev/mmcblk0p2 bs=4096
32768+0 records in
32768+0 records out
#
```

Uruchomienie systemu i utworzenie pliku w katalogu głównym w celu sprawdzenia, czy załadowany system rzeczywiście korzysta z systemu plików na karcie SD:

```
Welcome to Buildroot
buildroot login: root
# ls
# touch test
# ls
test
#
```

Po wyłączeniu i ponownym uruchomieniu urządzenia plik dalej istnieje:

```
Welcome to Buildroot
buildroot login: root
# ls
test
#
```