Міністерство освіти і науки України
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра обчислювальної техніки

Лабораторнаробота№1

з дисципліни
Архітектура комп’ютерів

Виконав:

студент групи ІО-14:
Чаплицький Д. А.

Перевірив:

Гайдай
А.Р.

Київ 2023

Лістинг команд з терміналу

sudo apt update	оновити список пакетів
sudo apt install git vim vim-gtk tree curl	встановити наступні утіліти
sudo apt install ccache	Встановлення утіліти ccache щоб можна було використовувати зібрані частини при перезборці ОС
ccache -M 5G	Встановлюємо найбільший розмір кешу 5 Гб
ccache -s	Подивитися конфігурацію ccache
ccache -C	Очистити ccache
tree ~/.ccache	Вивести дерево каталогів ~/.ccache
Скачуємо тулчейни для кроскомпіляції програм	
cd ~/Загрузки	Переходимо в папку ~/Загрузки
sudo tar xJvf gcc-arm-10.3-2021.07-x86_64-arm-none-eabi.tar.xz -C /opt/	Розпаковуємо тулчейн для “голого заліза” у папку /opt/
sudo tar xJvf gcc-arm-10.3-2021.07-x86_64-arm-none-linux-gnueabihf.tar.xz -C /opt	Розпаковуємо тулчейн для BusyBox у папку /opt/
sudo apt install make libncurses5-dev libssl-dev bc bison flex	Встановити утіліти для зборки
mkdir ~/repos	Створюємо директорію для зберігання файлів для зборки
cd ~/repos	Переходимо в цю директорію
git clone https://gitlab.denx.de/u-boot/u-boot.git	Клонуємо репозиторій з вихідним кодом завантажувальника
cd u-boot	Зайти в склонований репозиторій
git tag | grep -v rc | tail -15	Перевіряємо останні реліз-теги
git checkout v2019.07	Переходимо на версію 2019.07
curl https://patchwork.ozlabs.org/series/130450/mbox/ | git am	Застосувати серію патчів (помилка автовизначення імейлу)

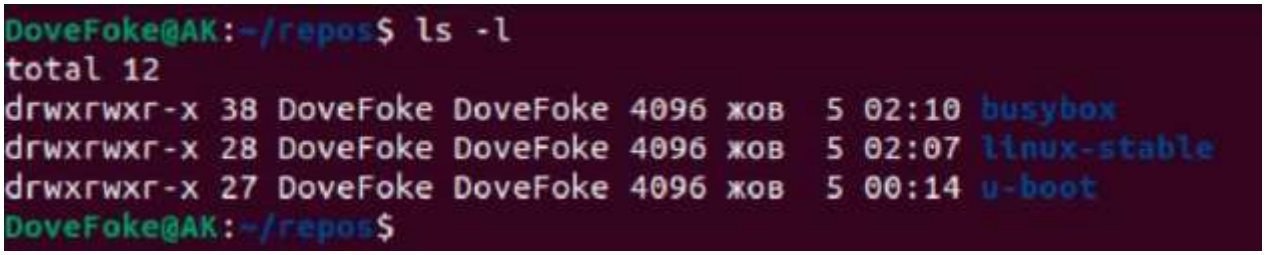
git config --global user.email "dovefoke@gmail.com"	Встановлюємо імейл для гіта
git config --global user.name "Darii"	Встановлюємо ім’я для гіта
curl https://patchwork.ozlabs.org/series/130450/mbox/ | git am	Повторюємо команду (успішно)
export PATH=/opt/gcc-arm-10.3-2021.07-x86_64-a rm-none-eabi/bin:$PATH	Додаємо у перемінну оточення шлях до тулчейну для bare metal
export CROSS_COMPILE='ccache arm-none-eabi-'	Створюємо перемінну оточення PATH для використання ccache
export ARCH=arm	Встановлюємо перемінну оточення архітектури для кросскомпіляції
make am335x_boneblack_defconfig	Створюємо конфіг для нашої плати
make -j4	Збираємо завантажувальник U-Boot
cd ~/repos	Переходимо в папку ~/repos
git clone git://git.kernel.org/pub/scm/linux/kernel/git/stable/linux-stable.git	Клонуємо репозиторій з вихідним кодом ядра
cd linux-stable	Заходимо в склонований репозиторій
git checkout linux-4.19.y	Перейти на вітку linux-4.19.y
export PATH=/opt/gcc-arm-10.3-2021.07-x86_64-a rm-none-eabi/bin:$PATH	Додаємо у перемінну оточення PATH шлях до тулчейну для bare metal
export CROSS_COMPILE='ccache arm-none-eabi-'	Створюємо перемінну оточення для використання ccache
export ARCH=arm	Встановлюємо перемінну оточення архітектури для кросскомпіляції
mkdir fragments	Створюємо директорію fragments
vim fragments/bbb.cfg	Створюємо фрагмент конфігу для плати
./scripts/kconfig/merge_config.sh \ arch/arm/configs/multi_v7_defconfig fragments/bbb.cfg	Змерджимо конфіги

make -j4 zImage modules am335x-boneblack.dtb	Компілюємо ядро
cd ~/repos	Переходимо в папку ~/repos
git clone git://git.busybox.net/busybox	Клонуємо репозиторій з вихідним кодом BusyBox
cd busybox	Заходимо в склонований репозиторій
git branch -a | grep stable | sort -V | tail -1	Перевіряємо останню стабільну гілку
git checkout 1_31_stable	Переходимо на гілку 1_31_stable
export ARCH=arm	Встановлюємо перемінну оточення архітектури для кросскомпіляції
export PATH=/opt/gcc-arm-10.3-2021.07-x86_64-a rm-none-linux-gnueabihf/bin:$PATH	Додаємо у перемінну оточення PATH шлях до linux тулчейну
export CROSS_COMPILE="ccache arm-none-linux-gnueabihf-"	Створюємо перемінну оточення для використання ccache
make defconfig	Створюємо конфіг BusyBox
make -j4	Збираємо BusyBox
make install	Копіюємо зібрані виконувані файли у потрібну локацію(інсталюємо)
mkdir -p _install/{boot,dev,etc\/init.d,lib,proc,root,sys\ /kernel\/debug,tmp}	Створюємо папки потрібні для роботи дистрибутиву
vim _install/etc/init.d/rcS	Створюємо init-скрипт
chmod +x _install/etc/init.d/rcS	Даємо йому права на виконання
ln -s bin/busybox _install/init	Створюємо soft link на init у кореневому каталозі
cd _install/boot	Заходимо в директорію _install/boot
cp ~/repos/linux-stable/arch/arm/boot/zImage .	Копіюємо в папку /boot образ ядра
cp~/repos/linux-stable/arch/arm/boot/dts/am33 5x-boneblack.dtb .	Копіюємо в папку /boot файл dtb

cp ~/repos/linux-stable/System.map .	Копіюємо в папку /boot файл System.map
cp ~/repos/linux-stable/.config ./config	Копіюємо в папку /boot файл .config
cd ~/repos/linux-stable	Переходимо в репозиторій з вихідним кодом ядра
export INSTALL_MOD_PATH=~/repos/busybox/_install	Встановлюємо значення перемінної оточення INSTALL_MOD_PATH для утіліти make
export ARCH=arm	Встановлюємо перемінну оточення архітектури для кросскомпіляції
make modules_install	Копіюємо модулі ядра в _install Перевіряємо: скопіювалось.
cd ~/repos/busybox	Перейти у директорію ~/repos/busybox
${CROSS_COMPILE}readelf -d _install/bin/busybox | grep NEEDED	Подивитися залежності busybox
cd _install/lib	Перейти у папку _install/lib
libc_dir=$(${CROSS_COMPILE}gcc -print-sysroot)/lib	Вказуємо папку для встановлення бібліотек
cp -a $libc_dir/*.so* .	Копіюємо туди всі бібліотеки
cd -	Переходимо в попередню директорію
echo '$MODALIAS=.* root:root 660 @modprobe "$MODALIAS"' > _install/etc/mdev.conf	Конфігурація mdev
echo 'root:x:0:' > _install/etc/group	Додаємо групу root
echo 'root:x:0:0:root:/root:/bin/sh' > _install/etc/passwd	Додаємо користувача root
echo 'root::10933:0:99999:7:::' > _install/etc/shadow	Додаємо інформацію про пароль користувача root
echo "nameserver 8.8.8.8" > _install/etc/resolv.conf	Додаємо файл resolv.conf, який використовуватиметься для перетворення імен хостів мережі в IP-адреси
sudo apt install qemu-system-arm	Встановлюємо емулятор для симуляції архітектури
cd ~/repos/busybox/_install/	Перейти в папку ~/repos/busybox/_install/

find . | cpio -o -H newc | gzip > ../rootfs.cpio.gz	Створити архів всіх файлів для емулятора
cd ..	Вийти в директорію на рівень вище
qemu-system-arm -kernel _install/boot/zImage -initrd rootfs.cpio.gz \ -machine virt -nographic -m 512 \ -- append "root=/dev/ram0 rw console=ttyAMA0,115200 mem=512M"	Запускаємо ядро і rootfs у симуляторі
uname -a ls -l dmesg | grep init busybox --help | head - 15 poweroff	Тестуємо систему

Результати роботи



```
DoveFoke@AK:~/repos$ cd busybox
DoveFoke@AK:~/repos/busybox$ qemu-system-arm -kernel _install/boot/zImage -initr
d rootfs.cpio.gz -machine virt -nographic -m 512 --append "root=/dev/ram0 rw con
sole=ttyAMA0,115200 mem=512M"
[    0.000000] Booting Linux on physical CPU 0x0
[    0.000000] Linux version 4.19.295 (ps4k@ps4k-VirtualBox) (gcc version 8.3.0
(GNU Toolchain for the A-profile Architecture 8.3-2019.03 (arm-rel-8.36))) #1 SM
P Thu Oct 5 01:23:01 EEST 2023
[    0.000000] CPU: ARMv7 Processor [412fc0f1] revision 1 (ARMv7), cr=10c5387d
[    0.000000] CPU: div instructions available: patching division code
[    0.000000] CPU: PIPT / VIPT nonaliasing data cache, PIPT instruction cache
[    0.000000] OF: fdt: Machine model: linux,dummy-virt
[    0.000000] Memory policy: Data cache writealloc
[    0.000000] efi: Getting EFI parameters from FDT:
[    0.000000] efi: UEFI not found.
[    0.000000] cma: Reserved 64 MiB at 0x5c000000
[    0.000000] psci: probing for conduit method from DT.
[    0.000000] psci: PSCIv0.2 detected in firmware.
[    0.000000] psci: Using standard PSCI v0.2 function IDs
[    0.000000] psci: Trusted OS migration not required
[    0.000000] percpu: Embedded 16 pages/cpu s36748 r8192 d20596 u65536
[    0.000000] Built 1 zonelists, mobility grouping on.  Total pages: 130048
[    0.000000] Kernel command line: root=/dev/ram0 rw console=ttyAMA0,115200 mem
=512M
[    0.000000] Dentry cache hash table entries: 65536 (order: 6, 262144 bytes)
[    0.000000] Inode-cache hash table entries: 32768 (order: 5, 131072 bytes)
[    0.000000] Memory: 406364K/524288K available (12288K kernel code, 1617K rwda
ta, 4792K rodata, 2048K init, 395K bss, 52388K reserved, 65536K cma-reserved, 0K
 highmem)
[    0.000000] Virtual kernel memory layout:
[    0.000000]     vector  : 0xffff0000 - 0xffff1000   (   4 kB)
[    0.000000]     fixmap  : 0xffc00000 - 0xfff00000   (3072 kB)
[    0.000000]     vmalloc : 0xe0800000 - 0xff800000   ( 496 MB)
[    0.000000]     lowmem  : 0xc0000000 - 0xe0000000   ( 512 MB)
[    0.000000]     pkmap   : 0xbfe00000 - 0xc0000000   (   2 MB)
[    0.000000]     modules : 0xbf000000 - 0xbfe00000   (  14 MB)
[    0.000000]       .text : 0x(ptrval) - 0x(ptrval)   (13280 kB)
[    0.000000]       .init : 0x(ptrval) - 0x(ptrval)   (2048 kB)
```

```
[    1.831457] usbserial: USB Serial support registered for moto_modem
[    1.831573] usbserial: USB Serial support registered for motorola_tetra
[    1.831708] usbserial: USB Serial support registered for nokia
[    1.831848] usbserial: USB Serial support registered for novatel_gps
[    1.831977] usbserial: USB Serial support registered for siemens_mpi
[    1.832113] usbserial: USB Serial support registered for suunto
[    1.832234] usbserial: USB Serial support registered for vivopay
[    1.832379] usbserial: USB Serial support registered for zio
[    1.839786] rtc-pl031 9010000.pl031: rtc core: registered pl031 as rtc0
[    1.841937] i2c /dev entries driver
[    1.853055] sdhci: Secure Digital Host Controller Interface driver
[    1.853208] sdhci: Copyright(c) Pierre Ossman
[    1.854485] Synopsys Designware Multimedia Card Interface Driver
[    1.855978] sdhci-pltfm: SDHCI platform and OF driver helper
[    1.858566] ledtrig-cpu: registered to indicate activity on CPUs
[    1.860668] usbcore: registered new interface driver usbhid
[    1.860801] usbhid: USB HID core driver
[    1.865787] NET: Registered protocol family 10
[    1.870808] Segment Routing with IPv6
[    1.871206] sit: IPv6, IPv4 and MPLS over IPv4 tunneling driver
[    1.873553] NET: Registered protocol family 17
[    1.874043] can: controller area network core (rev 20170425 abi 9)
[    1.874471] NET: Registered protocol family 29
[    1.874610] can: raw protocol (rev 20170425)
[    1.874974] can: broadcast manager protocol (rev 20170425 t)
[    1.875182] can: netlink gateway (rev 20170425) max_hops=1
[    1.876153] Key type dns_resolver registered
[    1.876598] ThumbEE CPU extension supported.
[    1.876798] Registering SWP/SWPB emulation handler
[    1.878762] Loading compiled-in X.509 certificates
[    1.889883] input: gpio-keys as /devices/platform/gpio-keys/input/input0
[    1.892506] rtc-pl031 9010000.pl031: setting system clock to 2024-01-07 10:16
:35 UTC (1704622595)
[    1.897925] uart-pl011 9000000.pl011: no DMA platform data
[    1.982934] Freeing unused kernel memory: 2048K
[    1.992190] Run /init as init process

Please press Enter to activate this console.
/ #
```

```
/ # uname -a
Linux (none) 4.19.295 #1 SMP Thu Oct 5 01:23:01 EEST 2023 armv7l GNU/Linux
/ # ls -l
total 100
drwxrwxr-x    2 1000     1000             0 Oct  4 23:02 bin
drwxrwxr-x    2 1000     1000             0 Oct  4 23:06 boot
drwxrwxr-x    3 1000     1000             0 Jan  7 10:47 dev
drwxrwxr-x    3 1000     1000             0 Oct  4 23:09 etc
-rw-r--r--    1 1000     1000          5336 Dec 19 14:41 hello.ko
-rw-r--r--    1 1000     1000          5348 Dec 19 14:46 hello1.ko
-rw-r--r--    1 1000     1000          3508 Dec 19 14:46 hello2.ko
-rw-r--r--    1 1000     1000          5392 Dec 19 14:48 hello5.ko
-rw-r--r--    1 1000     1000         73104 Dec 19 14:48 hello5.ko.unstripped
drwxrwxr-x    2 1000     1000             0 Dec 11 19:34 inc
lrwxrwxrwx    1 1000     1000            11 Oct  4 23:05 init -> bin/busybox
drwxrwxr-x    3 1000     1000             0 Oct  4 23:08 lib
lrwxrwxrwx    1 1000     1000            11 Oct  4 23:02 linuxrc -> bin/busybox
dr-xr-xr-x   91 root     root             0 Jan  1  1970 proc
drwxrwxr-x    2 1000     1000             0 Oct  4 23:03 root
drwxrwxr-x    2 1000     1000             0 Oct  4 23:02 sbin
dr-xr-xr-x   12 root     root             0 Jan  7 10:47 sys
drwxrwxr-x    2 1000     1000             0 Oct  4 23:03 tmp
drwxrwxr-x    4 1000     1000             0 Oct  4 23:02 usr
```

```
/ # dmesg | grep init
[    0.000000] Memory: 406364K/524288K available (12288K kernel code, 1617K rwdata, 4792K rodata, 2048K init, 395K bss, 52388K reser
ved, 65536K cma-reserved, 0K highmem)
[    0.000000]       .init : 0x(ptrval) - 0x(ptrval)   (2048 kB)
[    0.873751] devtmpfs: initialized
[    0.102450] pinctrl core: initialized pinctrl subsystem
[    0.236288] SCSI subsystem initialized
[    0.381341] Trying to unpack rootfs image as initramfs...
[    1.635832] Freeing initrd memory: 25156K
[    1.755866] SuperH (H)SCI(F) driver initialized
[    1.757005] msm_serial: driver initialized
[    1.757330] STMicroelectronics ASC driver initialized
[    1.758566] STM32 USART driver initialized
[    2.016144] Run /init as init process
/ # busybox --help | head -15
BusyBox v1.31.1 (2023-10-05 01:59:40 EEST) multi-call binary.
BusyBox is copyrighted by many authors between 1998-2015.
Licensed under GPLv2. See source distribution for detailed
copyright notices.
```

```
/ # busybox --help | head -15
BusyBox v1.31.1 (2023-10-05 01:59:40 EEST) multi-call binary.
BusyBox is copyrighted by many authors between 1998-2015.
Licensed under GPLv2. See source distribution for detailed
copyright notices.

Usage: busybox [function [arguments]...]
   or: busybox --list[-full]
   or: busybox --show SCRIPT
   or: busybox --install [-s] [DIR]
   or: function [arguments]...

        BusyBox is a multi-call binary that combines many common Unix
        utilities into a single executable.  Most people will create a
        link to busybox for each function they wish to use and BusyBox
        will act like whatever it was invoked as.
```

Висновки

Створив виконувані файли для операційної системи Linux, призначені для використання на процесорах архітектури ARM. Встановив ядро Linux у віртуальному середовищі емулятора Qemu для ARM-процесорів.

При виконанні завдання проблем не виникало