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

Архітектура комп'ютерів 3. Мікропроцесорні системи

Лабораторна робота 1

«Встановлення ПЗ»

Виконав:
студент групи ІО-23
Корбут М. Я.
Залікова книжка №2313
Перевірив
Каплунов А.В.

Київ - 2025

Лабораторна робота №1

Тема: «Встановлення ПЗ»

Мета: Створити мінімальний програмний проект на мові асемблер, перевірити виконання відлагоджувачем.

Хід роботи:

1. **Підготовка середовища**
   Було встановлено усі необхідні інструменти та тулчейни за інструкцією.

2. **Створення файлу start.S**
   У створеному каталозі проєкту був створений файл start.S, який містить таблицю векторів виключень і мітку __hard_reset__:

```
.syntax unified
.cpu cortex-m4
//.fpu softvfp
.thumb
// Global memory locations.
.global vtable
.global reset_handler
/*
* vector table
*/
.type vtable, %object
vtable:
.word __stack_start
.word __hard_reset__+1
.size vtable, .-vtable
__hard_reset__:
ldr r0, =__stack_start
mov sp, r0
b __hard_reset__
```

3. **Створення скрипта лінкування lscript.ld**
   Скрипт визначає розміщення пам’яті:

```
MEMORY {
     FLASH (rx) : ORIGIN = 0x08000000, LENGTH = 1M
     RAM (rxw)  : ORIGIN = 0x20000000, LENGTH = 128K
}
__stack_start = ORIGIN(RAM) + LENGTH(RAM);
```

4. **Компіляція проєкту:**

```
arm-none-eabi-gcc -x assembler-with-cpp -c -O0 -g3 -
mcpu=cortex-m4 -mthumb -Wall start.S -o start.o
arm-none-eabi-gcc start.o -mcpu=cortex-m4 -mthumb -Wall --
specs=nosys.specs -nostdlib -lgcc -T./lscript.ld -o
firmware.elf
arm-none-eabi-objcopy -O binary -F elf32-littlearm
firmware.elf firmware.bin
```

5. **Запуск в емуляторі QEMU:**

```
qemu-system-gnuarmeclipse --verbose --board STM32F4-
Discovery --mcu STM32F407VG \
-d unimp,guest_errors --image firmware.bin --semihosting-
config enable=on,target=native -s -S
```

6. **Підключення GDB:**

```
arm-none-eabi-gdb firmware.elf
(gdb) target extended-remote :1234
(gdb) step
```

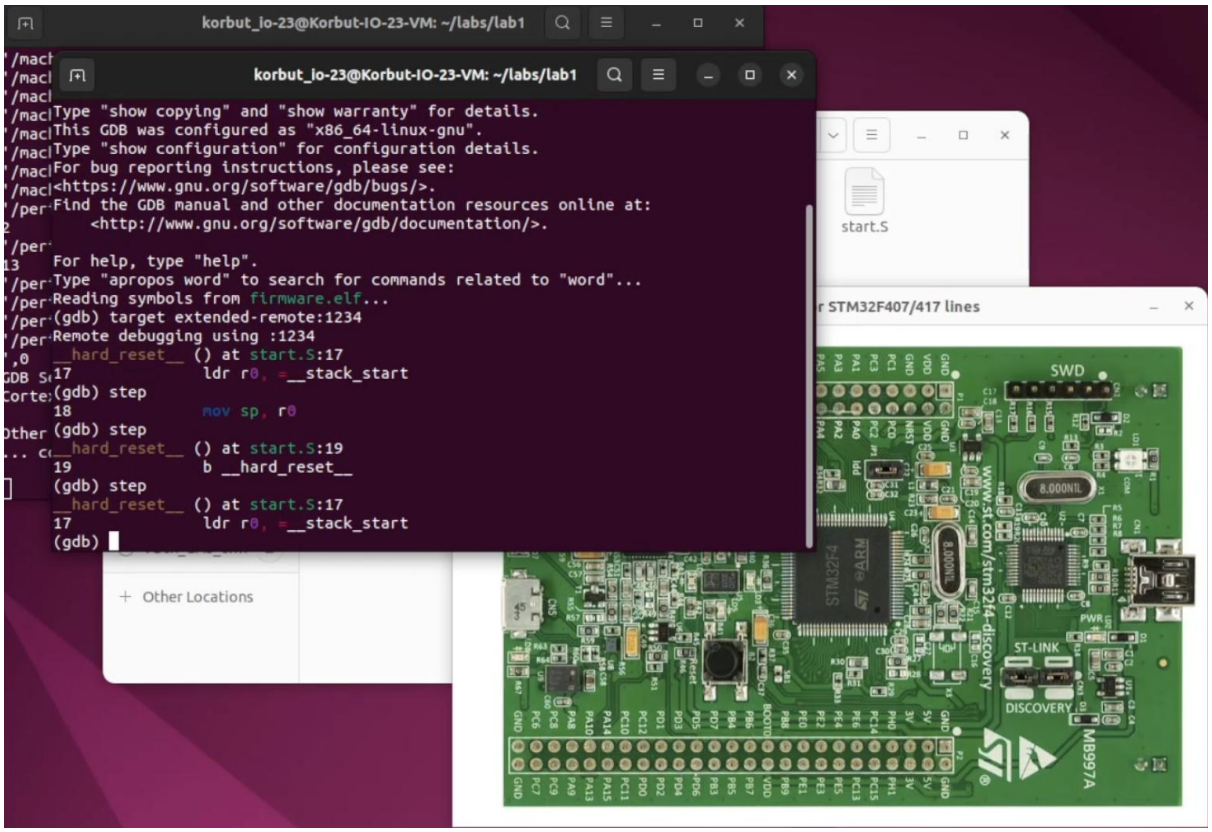
Таким чином виконується покрокове тестування прошивки в емуляторі.

7. **Makefile**

Було створено Makefile для автоматизації збирання та запуску:

```
SDK_PREFIX?=arm-none-eabi-
CC = $(SDK_PREFIX)gcc
LD = $(SDK_PREFIX)ld
SIZE = $(SDK_PREFIX)size
OBJCOPY = $(SDK_PREFIX)objcopy
QEMU = ~/opt/xPacks/qemu-arm/xpack-qemu-arm-7.2.0-1/bin/qemu-system-
gnuarmeclipse
BOARD ?= STM32F4-Discovery
MCU=STM32F407VG
TARGET=firmware
CPU_CC=cortex-m4
TCP_ADDR=1234
deps = \
start.S \
lscript.ld
all: target
target:
    $(CC) -x assembler-with-cpp -c -O0 -g3 -mcpu=$(CPU_CC) -Wall start.S -
o start.o
    $(CC) start.o -mcpu=$(CPU_CC) -Wall --specs=nosys.specs -nostdlib -
lgcc -T./lscript.ld -o $(TARGET).elf
    $(OBJCOPY) -O binary -F elf32-littlearm $(TARGET).elf $(TARGET).bin
qemu:
    echo $$PATH
    $(QEMU) --verbose --verbose --board $(BOARD) --mcu $(MCU) -d
unimp,guest_errors --image $(TARGET).bin --semihosting-config
enable=on,target=native -gdb tcp::$(TCP_ADDR) -S
clean:
    -rm *.o
    -rm *.elf
    -rm *.bin
```

Скріншот роботи програми:



Розгортання сторінки

Код було завантажено до репозиторію GitHub. Переглянути його можна за посиланням.

Висновки:

У результаті виконання лабораторної роботи:

- Створено асемблерний проєкт з базовим стартовим кодом та таблицею векторів.
- Опановано базові принципи організації прошивки для Cortex-M4.
- Навчився збирати прошивку вручну та за допомогою Makefile.
- Запущено проєкт у емуляторі QEMU та протестовано за допомогою gdb.

Робота дозволила закріпити практичні навички роботи з асемблером, інструментами GNU ARM Toolchain та засобами відлагодження.