实用操作系统课程实验报告

实验名称：	实验五 鸿蒙 LiteOS-a 内核移植——串口移植
实验日期：	2023-11-10
实验地点：	文宣楼 B313

学号：	33920212204567
姓名：	任宇
专业年级：	软工 2021 级
学年学期：	2023-2024 学年第一学期

1.实验目的

- 进行鸿蒙 LiteOS-a 内核的串口移植，实现：
  - 串口发送单个字符
  - 注册串口接收中断函数：确定中断号、使能中断、在中断函数中读取数据

2.实验内容和步骤

（1） 修改文件名字：

修改 vendor/democom/demochip/driver，修改对应文件名字，修改文件名字后同时需要修改 Makefile 文件：

原来为：

```
book@ry-virtual-machine:~/openharmony/vendor/democom/demochip/driver$ ls
hello  mtd  stm32mp157-fb  stm32mp157-i2c  stm32mp157-uart  touch
book@ry-virtual-machine:~/openharmony/vendor/democom/demochip/driver$ cd stm32mp157-uart/
book@ry-virtual-machine:~/openharmony/vendor/democom/demochip/driver/stm32mp157-uart$ ls
Makefile  stm32mp157_uart.h  uart_core.c  uart_dev.c  uart_dev.h  uart_stm32mp157.c
```

修改后：

```
book@ry-virtual-machine:~/openharmony/vendor/democom/demochip/driver$ ls
hello  mtd  stm32mp157-fb  stm32mp157-i2c  touch  uart
book@ry-virtual-machine:~/openharmony/vendor/democom/demochip/driver$ cd uart/
book@ry-virtual-machine:~/openharmony/vendor/democom/demochip/driver/uart$ ls
Makefile  uart.c  uart_core.c  uart_dev.c  uart_dev.h  uart.h
book@ry-virtual-machine:~/openharmony/vendor/democom/demochip/driver/uart$
```

修改 Makefile 文件：

```
include $(LITEOSTOPDIR)/config.mk

MODULE_NAME := $(notdir $(shell pwd))

LOCAL_SRCS :=  uart_core.c uart_dev.c uart.c

include $(MODULE)
```

demochip.mk

```
5  LITEOS_BASELIB  += -lboard
6
7  LIB_SUBDIRS     += $(DEMOCHIP_BASE_DIR)/driver/mtd/common
8  LITEOS_BASELIB  += -lmtd_common
9
10 LIB_SUBDIRS     += $(DEMOCHIP_BASE_DIR)/driver/mtd/spi_nor
11 LITEOS_BASELIB  += -lspinor_flash
12
13 ifeq ($(LOSCFG_DRIVERS_VIDEO), y)
14 LIB_SUBDIRS            += $(DEMOCHIP_BASE_DIR)/driver/stm32mp1
15 LITEOS_BASELIB  += -lstm32mp157-fb
16 endif
17
18 LIB_SUBDIRS            += $(DEMOCHIP_BASE_DIR)/driver/uart
19 LITEOS_BASELIB  += -luart
```

（2） 修改串口地址和中断号：

文件路径 openharmony/vendor/democom/demochip/board/include/

asm/platform.h，作以下修改：

```
//#define UART4_REG_PBASE           0x40010000  /* stm32mp157 uart4 */
#define UART1_REG_PBASE           0x40010000  /* imx6ull uart1 */
```

修改中断号为 32+26

```
#define NUM_HAL_INTERRUPT_UART0          58
#define NUM_HAL_INTERRUPT_UART1          (32+26) /*imxu=6ull uart1*/
#define NUM_HAL_INTERRUPT_UART2          41
```

在……/board/bsd_board.c 中为串口添加资源，使用串口 1 及对应中断号：

```
    /*device_t uart_dev;
    UART_ADD_DEVICE(uart_dev, 0);
    callback("uart", SYS_RES_MEMORY, 0, UART4_REG_PBASE,
        UART4_REG_PBASE + UART_IOMEM_COUNT, UART_IOMEM_COUNT);
    callback("uart", SYS_RES_IRQ, 0, NUM_HAL_INTERRUPT_UART4,
        NUM_HAL_INTERRUPT_UART4, 1);*/

    device_t uart_dev;
    UART_ADD_DEVICE(uart_dev, 0);
    callback("uart", SYS_RES_MEMORY, 0, UART1_REG_PBASE,
        UART1_REG_PBASE + UART_IOMEM_COUNT, UART_IOMEM_COUNT);
    callback("uart", SYS_RES_IRQ, 0, NUM_HAL_INTERRUPT_UART1,
        NUM_HAL_INTERRUPT_UART1, 1);
```

在 uart.h 做对应修改：

```
#define TTY_DEVICE          "/dev/uartdev-0"
//#define UART_REG_BASE       UART4_REG_BASE
//#define NUM_HAL_INTERRUPT_UART    NUM_HAL_INTERRUPT_UART4  /* TODO,100ask */
#define UART_REG_BASE       UART1_REG_BASE
#define NUM_HAL_INTERRUPT_UART    NUM_HAL_INTERRUPT_UART1
```

在 platform.h 中做如下修改：

```
#define UART_BASE UART1_REG_BASE
#define UART0_INT_NUM NUM_HAL_INTERRUPT_UART1
```

将 uart_imx6ull.h 的内容覆盖到 uart_hardware.h 文件中，接着将 uart_imx6ull.c 的内容覆盖到 uart_hardware.c 文件中，并包含以下头文件：

```
#include "los_magickey.h"
#include "uart_hardware.h"
```

在覆盖后需要将文件中出现的 imx6ull 修改为 demochip。

（3） 添加打印信息调试：

在 \kernel\liteos_a\arch\arm\arm\src\startup\reset_vector_up.S 文件中加入调试信息：

```
#if defined(LOSCFG_PLATFORM_STM32MP157)
  ldr sp, =0xc0000000 + 0x1000000
  mov r0, #'S'
  bl uart_putc_phy
#endif

#if defined(LOSCFG_PLATFORM_DEMOCHIP)
  ldr sp, =0x80000000 + 0x1000000
  mov r0, #'S'
  bl uart_putc_phy
#endif
```

```
323 #if defined(LOSCFG_PLATFORM_DEMOCHIP)
324   mov r0, 'm'
325   bl uart_putc_virt
326 #endif
327
328 #if defined(LOSCFG_PLATFORM_STM32MP157)
329   mov r0, 'm'
330   bl uart_putc_virt
331 #endif
```

（4） 编写 uart_putc_phy、uart_putc_virt 函数：

```
void uart_putc_phy(char c)
{
    UART_Type * uartRegs = (UART_Type *)0x02020000;
    while(!((uartRegs->USR2)&(1<<3)));
    uartRegs->UTXD=(unsigned char)c;
}

void uart_putc_virt(char c)
{
    UART_Type * uartRegs = (UART_Type *)UART_REG_BASE;
    while(!((uartRegs->USR2)&(1<<3)));
    uartRegs->UTXD=(unsigned char)c;
}
```

（5） 修改 GIC_BASE_ADDR：

```
//#define GIC_BASE_ADDR             (GIC_VIRT_BASE + 0x20000)
#define GIC_BASE_ADDR             (GIC_VIRT_BASE)
```

同时修改 GIC_VIRT_BASE：

```
#define GIC_VIRT_BASE    PERIPH_DEVICE_BASE
```

（6） 编译并运行：

```
book@ry-virtual-machine:~/openharmony/kernel/liteos_a$ make -j 8
make[1]: 进入目录“/home/book/openharmony/kernel/liteos_a”
/home/book/openharmony/kernel/liteos_a/tools/menuconfig/conf --silentoldconfig /home/book/openharmony/kernel/liteos_a/Kconfig
*
* Restart config...
*
*
* Compiler
*
LiteOS_Compiler_Type
  1. arm-linux-ohoseabi (COMPILER_HIMIX_32) (NEW)
> 2. clang-llvm (COMPILER_CLANG_LLVM)
choice[1-2?]:
#
# configuration written to .config
#
mv -f /home/book/openharmony/kernel/liteos_a/include/generated/autoconf.h /home/book/openharmony/kernel/liteos_a/platform/include/menuconf
make[1]: 离开目录“/home/book/openharmony/kernel/liteos_a”
make[1]: 进入目录“/home/book/openharmony/kernel/liteos_a/arch/arm/arm”
src/startup/reset_vector_up.S:145:2: warning: deprecated since v7, use 'dsb'
 mcr p15, 0, r0, c7, c10, 4 @ DSB
 ^
src/startup/reset_vector_up.S:146:2: warning: deprecated since v7, use 'isb'
 mcr p15, 0, r0, c7, c5, 4 @ ISB
```

```
## Starting application at 0x81000000 ...
Sm
******************Main******************


******************Welcome******************

Processor   : Cortex-A7
Run Mode    : UP
GIC Rev     : GICv2
build time  : Nov 17 2023 22:00:53
Kernel      : Huawei LiteOS 2.0.0.35/debug

*******************************************

main core booting up...
cpu 0 entering scheduler
proc fs init ...
Mount procfs finished.
mem dev init ...
```

3.实验总结

在本次实验中，我成功实现了串口发送单个字符和注册串口接收中断函数，通过本次实验，我更加理解串口的使用与驱动分层和串口的分离与硬件操作，同时也对虚拟地址和实际地址有了更好地掌握，为后续移植奠定基础。

4.遇到的困难及解决方法

无