嵌入式系统与开发

实验报告

|  |  |
| --- | --- |
| 实验名称 | 实验 9 串口控制实验 |
| 实验次数 | 实验十 |
| 学生姓名 | 戴高一 |
| 所在专业 | 计算机科学与技术 |
| 所在班级 | 201班 |
| 指导教师 | 李剑 |
| 地点 | 学10-509 |
| 时间 | 2023年5月 |

实验 10 串口按键实验

**一、 实验目的**

1. 掌握中断服务程序及串口通信程序的编写步骤

2. 掌握串口通信程序的编写、编译、运行

3. 熟悉嵌入式系统启动代码、头文件、Makefile文件的编写

**二、 实验内容**

实验10的示例代码：在开发板上按下按键SW4，LED灯从左往右跑马灯闪烁10次，并在串口工具中显示“SW4 PRESSED”；按下按键SW5，LED灯从右往左跑马灯闪烁10次，并在串口工具中显示“SW5 PRESSED”。

实验10的提升奖励1：保留示例代码要求，再增加以下功能：当开发板收到串口工具发

送的ASCII代码“SW4+5T”时，LED灯从左往右跑马灯闪烁5次，开发板收到串口工具发送的代码“SW4+10T”时，LED灯从左往右跑马灯闪烁10次。（串口工具在“D:新509\04-常用工具\串口工具\UartAssist.exe”）

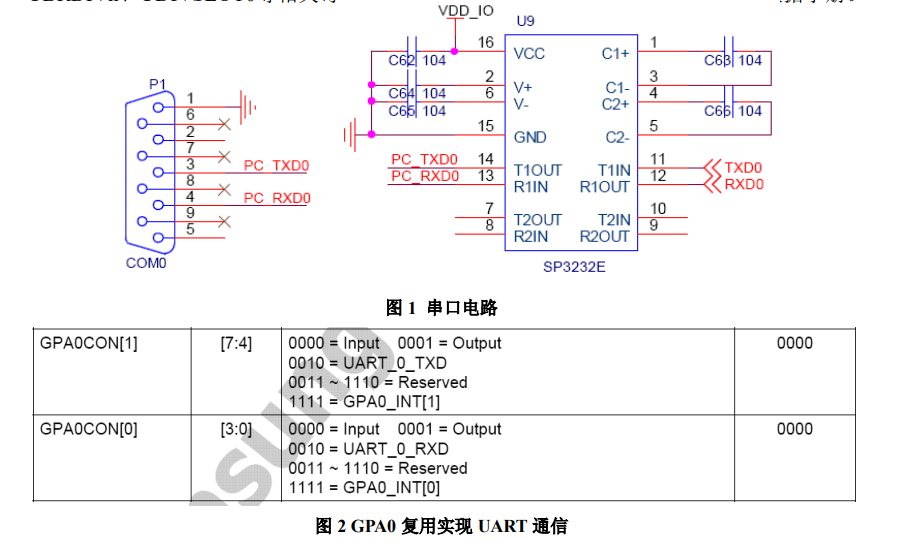
实验10的提升奖励2：利用串口1实现上述功能（奖励项1有没有做出来，没关系）

编写系统的启动代码、串口通信程序、头文件以及Makefile文件，编译得到可执行文

件，下载至开发板，实现在开发板上启动系统。

**三、 实验原理**

实验采用UART0实现串口通信，如图1所示。UART0的数据收发通过复用GPA0实现，

其中，GPA0\_0复用为UART\_0\_RXD，GPA0\_1复用为UART\_0\_TXD，如图2所示寄存器功能表。UART数据通信要求对ULCONn、UCONn、UFCONn、UTRSTATn、UTXHn、URXHn、 UBRDIVn、UDIVSLOT0等相关寄存器进行正确配置，寄存器功能详情参考芯片数据手册。

**四、实验步骤**

1．编写键控制代码，将代码编译为二进制文件。

1）在 Linux 系统的共享文件夹 forlinux 中新建 uart 目录。

2）切换到 uart 目录，新建启动文件 start.S 并添加代码（参考附录 1）。

3）新建时钟设置函数文件 clock.c 并添加代码（参考附录 2）。

4）新建 UART 初始化文件 uart.c，自行添加代码。

5）新建主函数文件 main.c，自行添加代码。

6）新建 Makefile 文件并添加代码（参考附录 3）。

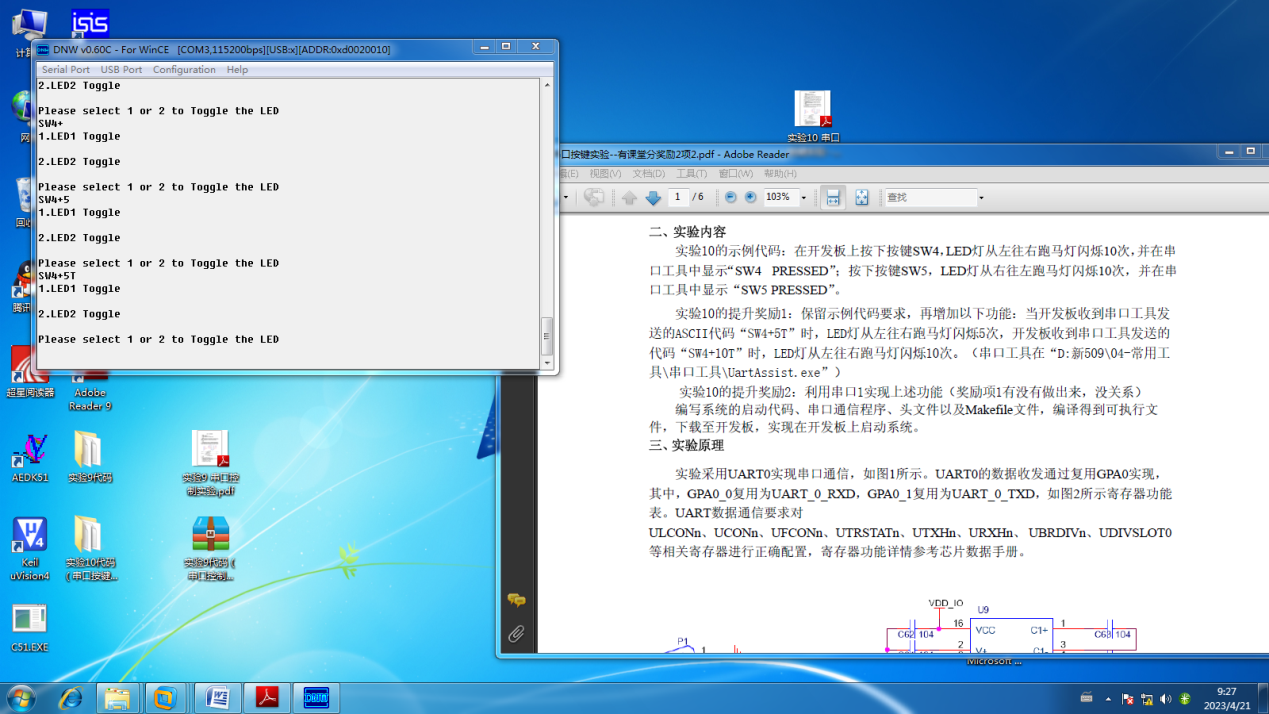
7）执行 make 命令编译生成二进制文件 uart.bin。

2．安装 USB 驱动

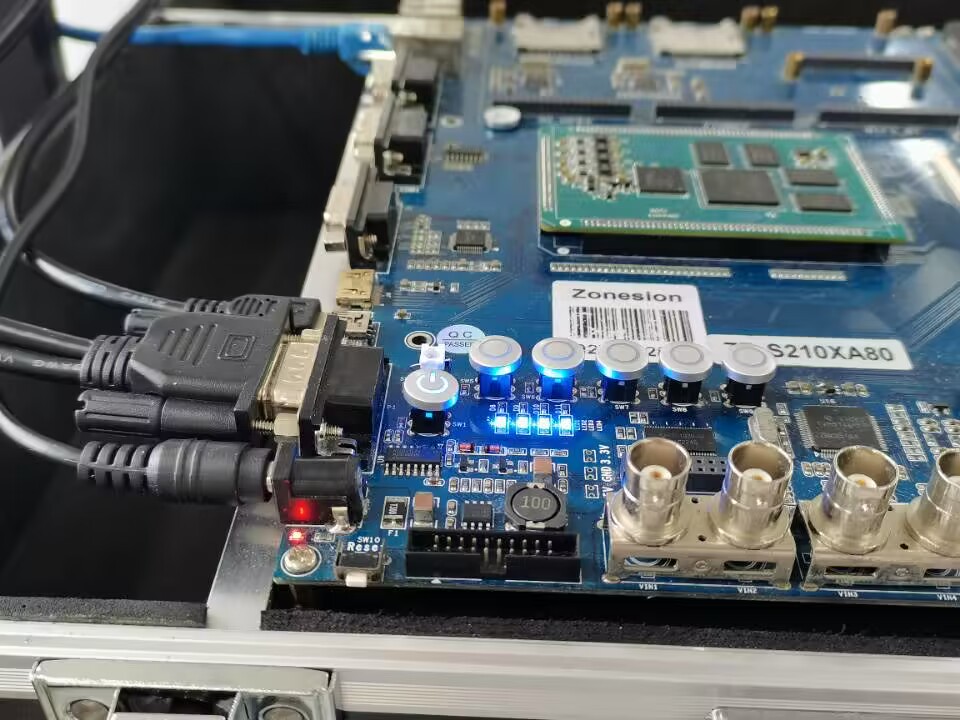
3. 使用 DNW 软件下载裸机程序 uart.bin 至 SRAM 中运行

**五、实验报告**

1. 附主要步骤中的命令及其结果截图。







2. 归纳总结。

本次实验主要帮助我们学习中断服务程序及串口通信程序的编写与运行，在掌握中断的相关操作的基础上，结合实验九流水灯的内容即可完成。提升内容，则是进行字符串对比，注意每次只读取一个字符、字符串长度和读取长度等限制即可。

3.实验代码

**uart.c**

#define GPA0CON \*((volatile unsigned int \*)0xE0200000)

#define ULCON0 \*((volatile unsigned int \*)0xE2900000)

#define UCON0 \*((volatile unsigned int \*)0xE2900004)

#define UFCON0 \*((volatile unsigned int \*)0xE2900008)

#define UTRSTAT0 \*((volatile unsigned int \*)0xE2900010)

#define UTXH0 \*((volatile unsigned int \*)0xE2900020)

#define URXH0 \*((volatile unsigned int \*)0xE2900024)

#define UBRDIV0 \*((volatile unsigned int \*)0xE2900028)

#define UDIVSLOT0 \*((volatile unsigned int \*)0xE290002C)

/\*

\*\* UART0初始化

\*/

void uart\_init()

{

/\*

\*\* 配置GPA0\_0为UART\_0\_RXD

\*\* 配置GPA0\_1为UART\_0\_TXD

\*/

GPA0CON &= ~0xFF;

GPA0CON |= 0x22; //0b 00100010

/\* 8-bits/One stop bit/No parity/Normal mode operation \*/

/\*每次8位，1个停止位，无奇偶验证，正常发送模式（非红外）\*/

ULCON0 = 0x3 | (0 << 2) | (0 << 3) | (0 << 6); // 0b 0 0xx 0 11

/\* Interrupt request or polling mode/Normal transmit/Normal operation/PCLK/\*/

/\*发送和接受引脚采用中断和轮询查询模式，正常发送，常规操作，时钟选择为PCLK\*/

UCON0 = 1 | (1 << 2) | (0 << 10);

/\* 禁止FIFO \*/

UFCON0 = 0;

/\*

\*\* 波特率计算：115200bps

\*\* PCLK = 66MHz

\*\* DIV\_VAL = (66000000/(115200 x 16))-1 = 35.8 - 1 = 34.8

\*\* UBRDIV0 = 34 (DIV\_VAL的整数部分)

\*\* (num of 1's in UDIVSLOTn)/16 = 0.8 (DIV\_VAL的小数部分)

\*\* (num of 1's in UDIVSLOTn) = 12

\*\* UDIVSLOT0 = 0xDDDD (在数据手册上880页查表)

\*/

UBRDIV0 = 34;//波特率分度值

UDIVSLOT0 = 0xDDDD;

}

void uart\_send\_byte(unsigned char byte)

{

while (!(UTRSTAT0 & (1 << 2))); /\* 等待发送缓冲区为空 \*/

UTXH0 = byte; /\* 发送一字节数据 \*/

}

unsigned char uart\_recv\_byte()

{

while (!(UTRSTAT0 & 1)); /\* 等待接收缓冲区有数据可读 \*/

return URXH0; /\* 接收一字节数据 \*/

}

void uart\_send\_string(char \*str)

{

char \*p = str;

while (\*p)

uart\_send\_byte(\*p++);

}

**main.c**

#define GPJ2CON (\*(volatile unsigned long \*) 0xE0200280)

#define GPJ2DAT (\*(volatile unsigned long \*) 0xE0200284)

#define GPH0CON (\*(volatile unsigned int \*)0xE0200C00)

#define GPH0DAT (\*(volatile unsigned int \*)0xE0200C04)

#define GPD0CON (\*(volatile unsigned long \*)0xE02000A0)

#define GPD0DAT (\*(volatile unsigned long \*)0xE02000A4)

#define GPC1CON (\*((volatile unsigned long \*)0xE0200060))

#define GPC1DAT (\*((volatile unsigned long \*)0xE0200064))

#define GPC1PUD (\*((volatile unsigned long \*)0xE0200068))

#define VIC0INTSELECT \*((volatile unsigned int \*)0xF200000C)

#define VIC0INTENABLE \*((volatile unsigned int \*)0xF2000010)

#define VIC0ADDRESS \*((volatile unsigned int \*)0xF2000F00)

#define VIC0VECTADDR2 \*((volatile unsigned int \*)0xF2000108)

#define VIC0VECTADDR3 \*((volatile unsigned int \*)0xF200010C)

#define EXT\_INT\_0\_CON \*((volatile unsigned int \*)0xE0200E00)

#define EXT\_INT\_0\_MASK \*((volatile unsigned int \*)0xE0200F00)

#define EXT\_INT\_0\_PEND \*((volatile unsigned int \*)0xE0200F40)

#define LEDS 1

extern void key\_isr(void);

extern void uart\_send\_byte(unsigned char byte);

extern unsigned char uart\_recv\_byte();

extern void uart\_send\_string(char \*str);

void delay(volatile unsigned int t)

{

volatile unsigned int t2 = 0xFFFF;

while (t--)

for (; t2; t2--);

}

void buzzer\_init(void)

{

GPD0CON |= 1<<8;

}

void buzzer\_on(void)

{

GPD0DAT |= 1<<2;

}

void buzzer\_off(void)

{

GPD0DAT &= ~(1<<2);

}

void key\_handle()

{

volatile unsigned char key\_code = EXT\_INT\_0\_PEND & 0xC;

VIC0ADDRESS = 0;

EXT\_INT\_0\_PEND |= 0xC;

if (key\_code == 0x04) /\* SW4 \*/

{

int num = 10;

uart\_send\_string("\r\nSW4 PRESSED\r\n");

while (num--)

{

GPJ2DAT ^= 1 << 0;

delay(100);

GPJ2DAT ^= 1 << 1;

delay(100);

GPJ2DAT ^= 1 << 2;

delay(100);

GPJ2DAT ^= 1 << 3;

delay(100);

}

}

/\* toggle LED1 \*/

else if (key\_code == 0x08) /\* SW5 \*/

{

int num = 10;

uart\_send\_string("\r\nSW5 PRESSED\r\n");

while (num--)

{

GPJ2DAT ^= 1 << 3;

delay(100);

GPJ2DAT ^= 1 << 2;

delay(100);

GPJ2DAT ^= 1 << 1;

delay(100);

GPJ2DAT ^= 1 << 0;

delay(100);

}

}

/\* toggle LED2 \*/

}

int main()

{

char byte[10];

char \* str = byte;

int n=0;

int i=0;

char t5[6]="SW4+5T";

char t10[7]="SW4+10T";

GPJ2CON &= ~(0xFF << 0);

GPJ2CON |= ((0x01 << 0) | (0x01 << 4) | (0x01 << 8) | (0x01 << 12));//((0x01 << 0) | (0x01 << 4));

GPJ2DAT |= (0xFF << 0);

GPH0CON |= 0xFF << 8;

EXT\_INT\_0\_CON &= ~(0xFF << 8);

EXT\_INT\_0\_CON |= (2 << 8) | (2 << 12);

EXT\_INT\_0\_MASK &= ~0xC;

uart\_send\_string("\r\nUART Test in S5PV210\r\n");

for(i=0;i<10;i++)byte[i]=0;

n=0;

while (1)

{

loop:

VIC0INTSELECT &= ~0xC;

VIC0INTENABLE |= 0xC;

VIC0VECTADDR2 = (int)key\_isr;

VIC0VECTADDR3 = (int)key\_isr;

uart\_send\_string("\r\n1.LED1 Toggle\r\n");

uart\_send\_string("\r\n2.LED2 Toggle\r\n");

uart\_send\_string("\r\nPlease select 1 or 2 to Toggle the LED\r\n");

byte[n++] = uart\_recv\_byte();

uart\_send\_string(str);

if (n==1 && byte[0]=='1')

{

GPJ2DAT ^= 1 << 3;

delay(1000);

GPJ2DAT ^= 1 << 2;

delay(1000);

GPJ2DAT ^= 1 << 1;

delay(1000);

GPJ2DAT ^= 1 << 0;

delay(2000);

GPJ2DAT = 0xFF;

for(i=0;i<n;i++)byte[i]=0;

n=0;

}

else if(n==1 && byte[0]=='2'){

GPJ2DAT ^= 1 << 0;

delay(1000);

GPJ2DAT ^= 1 << 1;

delay(1000);

GPJ2DAT ^= 1 << 2;

delay(1000);

GPJ2DAT ^= 1 << 3;

delay(2000);

GPJ2DAT = 0xFF;

for(i=0;i<n;i++)byte[i]=0;

n=0;

}

else if(n==7){

for(i=0;i<n;i++){

if(byte[i]!=t10[i])goto loop;

}

for(i=0;i<10;i++){

GPJ2DAT ^= 1 << 3;

delay(1000);

GPJ2DAT ^= 1 << 2;

delay(1000);

GPJ2DAT ^= 1 << 1;

delay(1000);

GPJ2DAT ^= 1 << 0;

delay(2000);

GPJ2DAT = 0xFF;

}

for(i=0;i<n;i++)byte[i]=0;

n=0;

} else if(n==6){

for(i=0;i<n;i++){

if(byte[i]!=t5[i])goto loop;

}

for(i=0;i<5;i++){

GPJ2DAT ^= 1 << 3;

delay(1000);

GPJ2DAT ^= 1 << 2;

delay(1000);

GPJ2DAT ^= 1 << 1;

delay(1000);

GPJ2DAT ^= 1 << 0;

delay(2000);

GPJ2DAT = 0xFF;

}

for( i=0;i<n;i++)byte[i]=0;

n=0;

} else if(n==10){

for( i=0;i<n;i++)byte[i]=0;

n=0;

}

delay(1000);

}

return 0;

}

**start.S**

.global \_start

.global key\_isr

\_start:

ldr sp, =0x40000000

mrs r0, cpsr

bic r0, r0, #0x00000080

msr cpsr, r0

@ bl clock\_init

bl uart\_init

bl main

halt:

b halt

key\_isr:

sub lr, lr, #4

stmfd sp!, {r0-r12, lr}

bl key\_handle

ldmfd sp!, {r0-r12, pc}^

**clock.c**

#define APLLCON0 \*((volatile unsigned int \*)0xE0100100)

#define MPLLCON \*((volatile unsigned int \*)0xE0100108)

#define EPLLCON0 \*((volatile unsigned int \*)0xE0100110)

#define VPLLCON \*((volatile unsigned int \*)0xE0100120)

#define CLK\_SRC0 \*((volatile unsigned int \*)0xE0100200)

#define CLK\_DIV0 \*((volatile unsigned int \*)0xE0100300)

#define CLK\_DIV1 \*((volatile unsigned int \*)0xE0100304)

#define CLK\_DIV2 \*((volatile unsigned int \*)0xE0100308)

#define CLK\_DIV3 \*((volatile unsigned int \*)0xE010030C)

void clock\_init()

{

/\* 1、设置PLL\_LOCK寄存器（这里使用默认值） \*/

/\* 2、设置PLL\_CON寄存器（使用芯片手册推荐的值） \*/

APLLCON0 = (1 << 0) | (3 << 8) | (125 << 16) | (1 << 31); /\* FOUTAPLL = 1000MHz \*/

MPLLCON = (1 << 0) | (12 << 8) | (667 << 16) | (1 << 31); /\* FOUTMPLL = 667MHz \*/

EPLLCON0 = (2 << 0) | (3 << 8) | (48 << 16) | (1 << 31); /\* FOUTEPLL = 96MHz \*/

VPLLCON = (3 << 0) | (6 << 8) | (108 << 16) | (1 << 31); /\* FOUTVPLL = 54MHz \*/

/\* 3、选择PLL为时钟输出 \*/

/\* MOUT\_MSYS = SCLKAPLL = 1000MHz

\*\* MOUT\_DSYS = SCLKMPLL = 667MHz

\*\* MOUT\_PSYS = SCLKMPLL = 667MHz

\*/

CLK\_SRC0 = (1 << 0) | (1 << 4) | (1 << 8) | (1 << 12);

/\* 4、设置系统时钟分频值 \*/

/\* freq(ARMCLK) = MOUT\_MSYS / (APLL\_RATIO + 1) = 1000MHz / (0 + 1) = 1000MHz

\*\* freq(HCLK\_MSYS) = ARMCLK / (HCLK\_MSYS\_RATIO + 1) = 1000MHz / (4 + 1) = 200MHz

\*\* freq(PCLK\_MSYS) = HCLK\_MSYS / (PCLK\_MSYS\_RATIO + 1) = 200MHz / (1 + 1) = 100MHz

\*\* freq(HCLK\_DSYS) = MOUT\_DSYS / (HCLK\_DSYS\_RATIO + 1) = 667 / (3 + 1) = 166MHz

\*\* freq(PCLK\_DSYS) = HCLK\_DSYS / (PCLK\_DSYS\_RATIO + 1) = 166 / (1 + 1) = 83MHz

\*\* freq(HCLK\_PSYS) = MOUT\_PSYS / (HCLK\_PSYS\_RATIO + 1) = 667 / (4 + 1) = 133MHz

\*\* freq(PCLK\_PSYS) = HCLK\_PSYS / (PCLK\_PSYS\_RATIO + 1) = 133 / (1 + 1) = 66MHz

\*/

CLK\_DIV0 = (0 << 0) | (4 << 8) | (1 << 12) | (3 << 16) | (1 << 20) | (4 << 24) | (1 << 28);

}

**Makefile**

uart.bin: start.o clock.o uart.o main.o

arm-linux-ld -Ttext 0xD0020010 -o uart.elf $^

arm-linux-objcopy -O binary uart.elf $@

arm-linux-objdump -D uart.elf > uart.dis

%.o : %.c

arm-linux-gcc -c $< -o $@ -nostdlib

%.o:%.S

arm-linux-gcc -c $< -o $@ -nostdlib

clean:

rm \*.o \*.elf \*.bin \*.dis