1. 69 系列 PWM 设置配置
2. UART 中断接收
3. 定时器产生小于 2ms 的中断频率

69 系列 PWM 设置配置

时间：20161019

Timer 的寄存器说明--请参考 AC46 用户手册。

以下是用 timer2 输出 PWM2 的示例：

```
void PWM2_init()
{
    printf("\n/////////clk = %u\n///////", clock_get_lsb_freq());//打印 LSB 时钟==60M

    JL_IOMAP->CON1 |= BIT(12);//BIT(11)||BIT(13)//使用 output channel 1 输出
    JL_PORTB->DIR &= ~BIT(8);//输出 PWM 的 IO
    JL_PORTB->PU   |=   BIT(8);
    JL_PORTB->PD   |=   BIT(8);
    JL_PORTB->DIE |=   BIT(8);

    JL_TIMER2->CNT = 0;//不需用到的寄存器,初始化为 0
    JL_TIMER2->PWM = 1200;//1200/6000*100%==20%占空比为 20%
    JL_TIMER2->PRD = 6000;//周期频率 6000/60M == 100us
    JL_TIMER2->CON = BIT(8) | BIT(0);
    //选择系统时钟(LSB),1 分频,使能 PWM,配置为 PWM 模式
}
```

注意：69 系列 timer 不支持选择 晶振时钟源。

PWM4—SDK 已做好：

```
void pwm4_cfg(u8 toggle,u8 pre_div,u8 duty)
{
    u8 pwm4_scaler;
    u8 pwm4_duty;
    if(toggle) {
        pwm4_scaler = pre_div & 0x7;
        pwm4_duty = duty & 0xF;
        JL_PWM4->CON = (pwm4_scaler << 4) | pwm4_duty;
        JL_IOMAP->CON1 |= BIT(11)|BIT(12)|BIT(13);
        JL_PORTA->DIR &= ~BIT(2);
        JL_PORTA->PU  |= BIT(2);
        JL_PORTA->PD  |= BIT(2);
        JL_PORTA->DIE |= BIT(2);
    } else {
        JL_PWM4->CON = 0;
        JL_IOMAP->CON1 &= ~(BIT(11)|BIT(12)|BIT(13));
    }
}
```

UART 中断接收

时间：20161024

UART 的寄存器说明--请参考 AC46 用户手册。

以下是用 UART2 的参考例子：(建议使用 UART2，避免程序上和 EQ 串口、串口升级、FCC 串口 产生冲突)

```
#define UART_INTERRUPT_RECEIVE_BAUD_RAE      9600
__uart_handle *uart_interrupt_receive_handle;//默认使用 uart2

void uart_interrupt_receive_isr_callback(u8 uto_buf,void *p, u8 isr_flag)
{
    static u8 cnt = 0;//ABCDEFG-HIJKLMN-OPQ-RST-UVW-XYZ == 31
    cnt++;
    if( UART_ISR_TYPE_DATA_COME == isr_flag)
    {
        putchar(uto_buf);
        if(uto_buf == 'Z'){
            printf("...cnt:%d\n",cnt);//请注意测试数据完整性，多测试多种不同波特率，越高越不稳定
            cnt = 0;
        }
    }
}
void uart_interrupt_receive_write(char a)
{
    if(uart_interrupt_receive_handle)
    {
        uart_interrupt_receive_handle->putbyte(uart_interrupt_receive_handle,a);
    }
    else
    {
        puts("uart_interrupt_receive_write err\n");
    }
}
s32 uart_interrupt_receive_set_baud(u32 baud)
{
    uart_module_ctrl(uart_interrupt_receive_handle,
UART_CMD_SET_BAUN_RATE,baud,NULL);
    return 0;
}
s32 uart_interrupt_receive_init(u32 baud)
{
```

```
    s32 status = 0;
    __uart_handle *uart_handle = &uart2_handle;//默认使用 uart2
//  char *uart_name;
    uart_interrupt_receive_handle=NULL;
//    uart_module_on();
    status=uart_module_open(uart_handle,UART2_HARDWARE_NAME);//默认使用 uart2
    if(!status)
    {
        __uart_param cur_param;
        memset(&cur_param,0,sizeof(__uart_param));
        cur_param.baud_rate = baud;
        cur_param.io = UART_TXPA9_RXPA10;
        cur_param.workmode = UART_WORKMODE_NORMAL;
        cur_param.custom |= (BIT(14)|BIT(3));//= 0;
        status=uart_module_init(uart_handle,&cur_param);
        if(status)
        {
            puts("uart_interrupt_receive_module_init err\n");
        }
        uart_reg_isr_callback_fun(&uart2_handle,1,uart_interrupt_receive_isr_callback);
        if(UART_OUTPUT_CHAL == cur_param.io)///设置要作为串口的 GPIO
        {
            JL_PORTA->DIR &= ~BIT(9);
            JL_PORTA->DIE &= ~BIT(9);
            JL_PORTA->PD   |=   BIT(9);
            JL_PORTA->PU   |=   BIT(9);
            JL_PORTA->DIR |=   BIT(10);
            JL_PORTA->DIE &= ~BIT(10);
            JL_PORTA->PD   |=   BIT(10);
            JL_PORTA->PU   |=   BIT(10);
        }
        if(!status)
        {
            status=uart_module_start(uart_handle);
            if(!status)
                uart_interrupt_receive_handle=uart_handle;
        }
    }
    return status;
}
```

注意调用的地方：

```
AT(.common)
void set_sys_freq(u32 out_freq)
{
    if(out_freq == SYS_CLK)
        return;

    user_prote_bt_process(1);
    clock_set_sys_freq(out_freq);
    user_prote_bt_process(0);

    set_spi_speed_auto();

    uart_set_baud(UART_BAUD_RAE);
    uart_interrupt_receive_set_baud(UART_INTERRUPT_RECEIVE_BAUD_RAE);
```

```
void board_main(u32 cfg_addr, u32 addr, u32 res,u32 update_flag)
{
    u32 tmp;
    FLASH_SYS_CFG sys_cfg;
    /* close_wdt();
     */
    clr_PINR_ctl();
    update_check(update_flag);
    /* JL_SYSTEM->LVD_CON &= ~BIT(2); */


    /* JL_CLOCK->SYS_DIV = 0; */
    clock_init(SYS_CLOCK_IN, OSC_Hz, SYS_Hz);
    set_spi_speed_auto();

    /* SFR(CLK_CON2,0,2,0); */
    /* clock_out_PA4(PA4_CLOCK_OUT_LSB); */
    uart_init(UART_BAUD_RAE); //<串口波特率
    uart_interrupt_receive_init(UART_INTERRUPT_RECEIVE_BAUD_RAE);
```

```
        case MSG_HALF_SECOND:
            //puts(" BT_H ");
            while(1){
                uart_interrupt_receive_write(char_cnt);
                if(char_cnt == 'Z'){
                    uart_interrupt_receive_write('\n');
                    char_cnt = 'A';
                    break;
                }else{
                    char_cnt++;
                }
            }
            if((BT_STATUS_CONNECTING == get_bt_connect_status())   ||
               (BT_STATUS_TAKEING_PHONE == get_bt_connect_status())||
               (BT_STATUS_PLAYING_MUSIC == get_bt_connect_status()))/*连接状态*/
            {
                if(BT_MUSIC_STATUS_STARTING == a2dp_get_status())       /*播歌状态*/
                {
```

程序上，多个地方是测试数据完整性的，实际开发，注意删减。

定时器产生小于 2ms 的中断频率

时间：20161024

UART 的寄存器说明--请参考 AC46 用户手册

由于现在的程序风格，大部分实用功能底层已做好，但万事总有不满意的地方，不要期待我们去帮你们改底层，因为应用场景千奇百怪，要学会自给自足，自己可以参考 46 的程序，在顶层自己打补丁。

由于 69 旧的 SDK 底层定时器分频失误，此例子为在顶层自己打补丁。

以下是用 Timer2 的参考例子。

```
__timer_handle    *timer2_hl;
void timer2_isr_callback()
{
    static u8 flag = 0;
    if(flag){ //自己使用示波器或逻辑分析仪测量频率
//                  puts("PA11-1...");//\n
        JL_PORTA->DIR &= ~BIT(11);
        JL_PORTA->OUT |=    BIT(11);
        flag = 0;
    }else{
//                  puts("PA11-0...");//\n
        JL_PORTA->DIR &= ~BIT(11);
        JL_PORTA->OUT &= ~BIT(11);
        flag = 1;
    }
}
s32 timer2_init(void)
{
    s32 ret;
    __timer_param     timer_parm;
//  timer_module_on();
    timer2_hl = timer_open(TIMER2,TIMER_MAX_ISR_FUN);
    if(NULL == timer2_hl)
    {
        printf("timer_open err");
        ret = TIMER_DRV_OPEN_ERR;
        return ret;
    }
    timer_parm.work_mode    = TIMER_WORK_MODE_COUNTER;
    timer_parm.tick_time    = 10;//(2ms)
    ret = timer_init_api(timer2_hl,&timer_parm);
//    JL_TIMER2->PRD = 0xBB8;
```

```
//      JL_TIMER2->CON = 0x11;
        if(ret != TIMER_NO_ERR)
        {
            printf("timer_init err = %x\n",ret);
            return ret;
        }

        ret = timer_start(timer2_hl);
        if(ret != TIMER_NO_ERR)
        {
            printf("timer_start err = %x\n",ret);
            return ret;
        }
        ret = timer_reg_isr_callback_fun(timer2_hl,1,timer2_isr_callback);
        if(ret != TIMER_NO_ERR)
        {
            printf("timer_reg_isr_callback_fun err = %x\n",ret);
            return ret;
        }
        //这个要根据你们自己的实际应用，查看用户手册，自己再做修改，以下为参考写法而
已
        JL_TIMER2->PRD = 0x7530/40;/// 30000/15M/40 == 2ms/40 == 50us
        JL_TIMER2->CON = 0x11;//60M/4 == 15M(4 分频)
        printf("JL_TIMER2->PRD:%4x,JL_TIMER2->CON:%4x\n",
                    JL_TIMER2->PRD,JL_TIMER2->CON);
        return ret;
}
s32 timer2_clk_reset(void) //切换模式后，如果切换时钟了，要注意在这个函数自行重新赋值
{
        s32 ret;
        __timer_param      timer_parm;
        puts("timer_clk_reset\n");
        timer_parm.work_mode    = TIMER_WORK_MODE_COUNTER;
        timer_parm.tick_time    = 10;//(2ms)
//      JL_TIMER2->PRD = 0xBB8;
//      JL_TIMER2->CON = 0x11;
        ret = timer_init_api(timer2_hl,&timer_parm);
        if(ret != TIMER_NO_ERR)
        {
            printf("timer_init err = %x\n",ret);
            return ret;
        }
        ret = timer_start(timer2_hl);
        if(ret != TIMER_NO_ERR)
```

```
    {
        printf("timer_start err = %x\n",ret);
        return ret;
    }
    JL_TIMER2->PRD = 0x7530/40;
    JL_TIMER2->CON = 0x11;
    printf("JL_TIMER2->PRD:%4x,JL_TIMER2->CON:%4x\n",
            JL_TIMER2->PRD,JL_TIMER2->CON);
    puts("timer_init_OK\n");
    return ret;
}
```

注意程序调用地方：

```
    vm_open_all();
#if BT_2_1_DEBUG
    bt_app_cfg();
#endif
    irq_handler_register(0, exception_isr, 0); //exception isr a
    timer0_init();
    timer2_init();

    sys_init();
    /* printf("read_reset_power=0x%x\n",read_reset_power() ); */
#if BT_2_1_DEBUG
    /* 2:LDO  3:DCDC*/
    power_init(PWR_MODE_SELECT);
#endif
```

```
#if KEY_AD_EN
    ad_key0_clk_reset();
#endif

    timer0_clk_reset();
    timer2_clk_reset();

/* #if UART_UPDATA_EN */
    /* UT1_BAUD = (UART_CLK / 460800) / 4 -1; */
/* #endif // UART_UPDATA_EN */

    printf("set_sys_freq = %d\n",SYS_CLK);
}
```

先按 69 风格定义一套，再添加 46 的程序代码就可以了。