南京凌鸥创芯电子有限公司

LKS_ROM
应用笔记

1 目 录

2 ROM 功能概述

使用 lks 的离线烧录器，可以将 07x 支持 ROM 功能芯片打开 rom 功能，打开 ROM 功能之后，ROM 空间内的程序将只能执行，不能被读取。

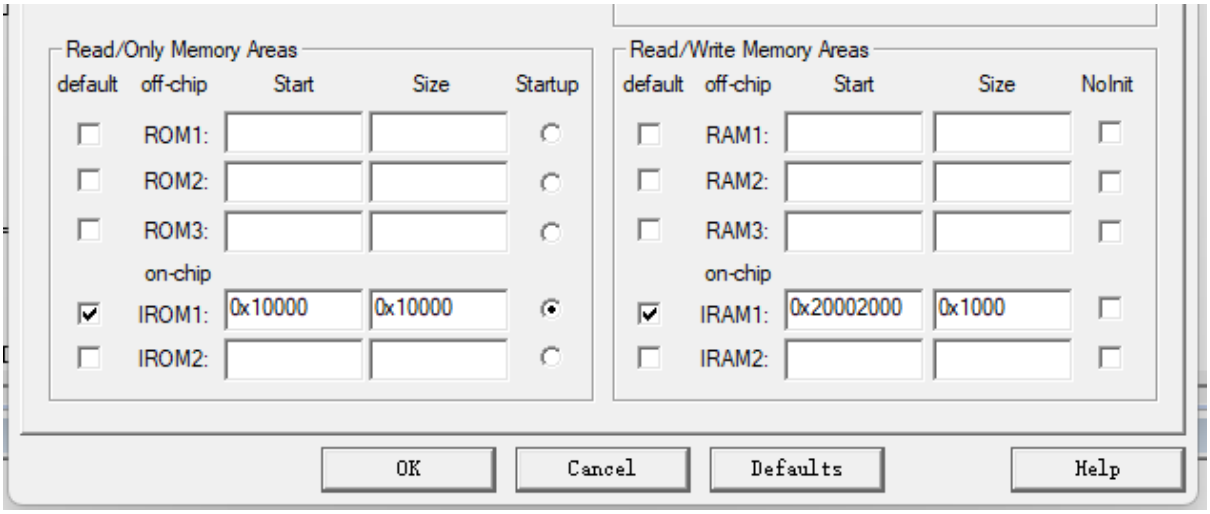
限制：

如果 ROM 中需要使用全局变量，应该手动对 RAM 进行分区，将 RAM 空间中靠后的空间分配给 ROM 的程序进行使用。

3 ROM 程序配置流程

3.1 KEIL 环境配置

ROM 的程序需要将 ROM 地址设置到 0x10000 到 0x20000 的范围内，对应分配给 ROM 使用的 RAM 这里分配大小为 4k，也就是 0x20002000 到 0x20003000。



3.2 程序修改

接下来修改在 main 函数中添加对 ROM 函数的调用，以防止编译器把 ROM 中的程序给优化掉，这里以 lks32_sin 和 lks32_cos 举例。

```
int main()
{
    for(;;)
    {
        j=lks32_sin(i123++);
        k=lks32_cos(i123++);
        if(user_key==1)
        {
            user_key=0;
            ROM_ENABLE(); // 打开03的rom功能
        }
    }
}
void SystemInit()
```

最后是导出 ROM 中的函数，可以在 map 中找到各个函数的地址，随后通过函数指针的方式去调用各个函数。

```
0x00010210  0x00010210  0x00000002  Code  RO    222  .text                c_p.l(use_no_semi.o)
0x00010212  0x00010212  0x00000000  Code  RO    224  .text                c_p.l(indicate_semi.o)
0x00010212  0x00010212  0x00000002  Code  RO      3  i.SystemInit        main.o
0x00010214  0x00010214  0x00000078  Code  RO     81  i.Trig_Functions    program1.o
0x0001028c  0x0001028c  0x00000010  Code  RO     82  i.lks32_cos         program1.o
0x0001029c  0x0001029c  0x00000010  Code  RO     83  i.lks32_sin         program1.o
0x000102ac  0x000102ac  0x00000040  Code  RO      4  i.main              main.o
0x000102ec  0x000102ec  0x00000020  Data  RO    279  Region$$Table       anon$$obj.o

Execution Region RW_IRAM1 (Exec base: 0x20002000, Load base: 0x0001030c, Size: 0x00000870, Max: 0x00001000, ABSOLUTE)
```

3.3 调用 ROM 中的程序

可以通过函数指针的方式调用 ROM 中的程序，函数指针的定义可以参考（https://c.biancheng.net/view/2023.html）：

函数指针的定义形式为：

returnType (*pointerName)(param list);

returnType 为函数返回值类型，pointerName 为指针名称，param list 为函数参数列表。参数列表中可以同时给出参数的类型和名称，也可以只给出参数的类型，省略参数的名称，这一点和函数原型非常类似。

注意()的优先级高于*，第一个括号不能省略，如果写作 returnType *pointerName(param list); 就成了函数原型，它表明函数的返回值类型为 returnType *。

```
21  volatile int32_t initflg;
22  int main()
23  {
24      j=lks32_sin(i123++);
25      k=lks32_cos(i123++);
26      initflg = 0x12345678;
27      NVIC_SystemReset();
28      for(;;);
29  }
```

3.4 初始化 ROM 使用的内存

初始化 rom 的全局变量可以通过调用 ROM 的 RESET()程序来实现，也就是说直接跳转到 ROM 的起始地址。通过在 rom 程序的 main 中写一个复位实现跳转回用户区，可以在 ROM 中设置一个全局变量，通过判断这个全局变量的值判断 ROM 程序是否完成初始化的动作，由于 RAM 的值在上电时是随机的，为了保证程序的稳定，也可以通过某一个寄存器实现，注意，用户区的程序必须要在 SystemInit 函数中进行调用，以免影响用户程序的堆栈初始化。

```
26      }
27  }
28  void SystemInit()
29  {
30      SYS_WR_PROTECT = 0x7a83;    /* 解除系统寄存器写保护 */
31      SYS_CLK_CFG = 0x000001ff; /* BIT8:0: CLK_HS,1:PLL  | BIT[7:0]CLK_DIV  | 1ff对应96M时钟 */
32      IWDG_CFG = 0x3c00;
33      SYS_WR_PROTECT = 0;    /* 解除系统寄存器写保护 */
34      if(REG32(0x20002008) != 0x12345678)
35      {
36          rom_RESET();
37      }
38  }
39
```