瑞萨 RL78 系列自升级程序设计

本文档描述瑞萨 **RL78 Bootload** 程序设计方案。

适用产品系列：

Renesas RL78（G12,G13,L13 .....）

程序设计环境：

编译器 IAR 版本：V1.4

驱动芯片型号： RL78 \R7F0C004

上位机 软件： WIN7 超级终端

RL78 仿真器 ： E1

章节：

本文档分两章节：

一、boot 设计

二、APP 设计

一、boot 设计

boot 区程序设计，设计难点在中断向量跳转部分，要实现与 APP 区域的中断函数跳转。

1、R7F0C004 资源简介

R7F0C004 属于瑞萨 RL78 L13 系列。
ROM: 128K 存储地址(00000H--1FFFFH)
RAM: 8K 存储地址(FDF00H--FFEDFH)

2、ROM 擦写库函数 FSL 简介

瑞萨官方提供 RL78 系列程序擦写库函数 FSL。

2.1 FSL 函数

以库函数的形式提供，不开源。

2.2 FLS 占用资源

FSL 在使用时会占用 RAM 空间，依据手册上的描述，当使用自编程功能时，不能将堆栈、数据缓冲器、向量中断处理的转移目标以及 DMA 的转移目标/传送源使用的目的地址分配到 FFE20H~FFEDF 区，另外 FDF00H~FE309H 区用于程序库，所以禁止使用。

3、IAR 版本

本设计使用 IAR 开发环境进行设计，可在 IAR 官网进行下载。

3.1 IAR for RL78

使用最新的 IAR for RL78 IDE,版本 1.4，破解后存在问题，每次再打开 IAR 环境后需要手动修改堆栈的大小，因每次关闭 IAR 后堆栈会复位。

4、程序段划分

R7F0C004 一共 128K ROM，本设计 bootload 使用 8K,应用程序 APP 120K.

4.1 boot 段分配

R7F0C004 boot 程序 ROM 空间分配
Bootload: 0000H~1FFFH 共 8K
APP : 2000H~1FFFFH 共 120K
具体的配置在 IAR 的连接文件 XCL 文件中完成。

5、XCL 连接文件

IAR 具体的 ROM 段划分在配置文件 XCL 中，将 IAR 安装目录下的 R7F0C004 的连接文件 XCL R5F10WMG 拷贝到工程目录下，在调试环境 IAR 中连接到此文件。

5.1 XCL ROM 段分配

Boot 设计在 0000H~1FFFH 之间，启动函数 cstartup，FSL 库函数都要划分到此区间以内。R7F0C004 的中断向量也在此区间内（0000H~0007FH）.

将 XCL 段的定义 CODE,CONST 端定义到 1FFF 以内：如下

```
//-------------------------------------------------------
//      Startup, Runtime-library, Near, Interrupt
//      and CALLT functions code segment and near switch.
//-------------------------------------------------------
-Z(CODE)RCODE,CODE=000D8-01FFF

-Z(CONST)SWITCH=000D8-01FFF

//-------------------------------------------------------
//      Setting for Code Flash Self-programming Library.
//-------------------------------------------------------
-Z(CODE)FSL_FCD=000D8-01FFF
-Z(CODE)FSL_FECD=000D8-01FFF
-Z(CODE)FSL_BCD=000D8-01FFF
-Z(CODE)FSL_BECD=000D8-01FFF
-Z(CODE)FSL_RCD=000D8-01FFF

//-------------------------------------------------------
//      Near data initializer segments.
//-------------------------------------------------------
-Z(CONST)NEAR_ID=000D8-01FFF
-Z(CONST)SADDR_ID=000D8-01FFF
-Z(CONST)DIFUNCT=000D8-01FFF
```

5.2 XCL RAM 段分配

因 FSL 库在使用时需要占用 004 一部分 RAM 资源，为了防止用户应用程序的 RAM 区的数据在使用 FSL 时被 FSL 覆盖掉，所以必须在 XCL 文件中对 FSL 用到的 RAM 区域进行规避。

FSL 需占用 FDF00H~FE309H,FDF00H 为 RAM 区域起始地址。

所以 XCL 文件中 RAM 起始地址工 FE30AH 开始，如下图：

```
//-------------------------------------------------------
//      Near data segments.
//-------------------------------------------------------
-Z(DATA)NEAR_I,NEAR_Z,NEAR_N=FE30A-FFE1F
//-------------------------------------------------------
//      Huge data segments.
//-------------------------------------------------------
-Z(DATA)HUGE_Z=FE30A-FFE1F
-P(DATA)HUGE_N=FE30A-FFE1F
//-------------------------------------------------------
//      Far data segments.
//-------------------------------------------------------
-Z(DATA)FAR_Z=[FE30A-FFE1F]/10000
-P(DATA)FAR_N=[FE30A-FFE1F]/10000
//-------------------------------------------------------
//      Heap segments.
//-------------------------------------------------------
-Z(DATA)NEAR_HEAP+_NEAR_HEAP_SIZE=FE30A-FFE1F
-Z(DATA)HUGE_HEAP+_HUGE_HEAP_SIZE=FE30A-FFE1F
-Z(DATA)FAR_HEAP+_FAR_HEAP_SIZE=[FE30A-FFE1F]/10000
//-------------------------------------------------------
//      Stack segment.
//-------------------------------------------------------
-Z(DATA)CSTACK+_CSTACK_SIZE=FE30A-FFE1F
```

FSL 需占用 FFE20H~FFEDF 区,FFEDF 为 RAM 区域结束地址。

在 XCL 文件中将此区域屏蔽掉，保证应用程序不会访问到此区域，在 XCL 文件中对此区域内的内存段做如下处理：如下图

```
//-------------------------------------------------------------
//       Short address data and workseg segments.
//-------------------------------------------------------------
-Z(DATA)WRKSEG=FFE20-FFEDF
//-Z(DATA)SADDR_I,SADDR_Z,SADDR_N=FFE20-FFEDF
```

6、boot 中断向量处理

RL78 系列只有一个中断向量表处在 0000H~0007FH 区域，

共 128 字节，此中断向量表不仅要响应 boot 区域的中断函数，还要响应应用程序 APP 过来的中断函数。

6.1 中断函数跳转

因中断向量表 0000H~0007FH 在 1FFFH 以内，处于 boot 区域，为了能响应应用程序的中断，则需要在 boot 区实现中断跳转，响应 APP 硬件中断向量后需要跳转到 APP 区去执行 APP 中断服务函数。

本设计中所有的中断向量函数都设计在 boot 区，在响应了中断后再判断是 boot 区触发的还是 APP 区触发的，来执行不同的中断服务函数。

如果 boot 区没有有到中断，则中断向量函数不需要判断是否 boot 区还是 APP 区传入的中断。

6.2 跳转函数

当 CPU 上电启动后，首先执行的是 boot 程序，判断是否进入 APP,如果上次升级成功，且没有升级请求，则直接调转到 APP。

7、代码配置

Boot 选型字节配置 R7F0C004 时钟为 24M,InitCPU 初始化中将时钟源分频为 12M，boot 接收 APP 升级 HEX 文件的通道为 uart0，uart1，uart2，代码会依次扫描这三个串口是否收到空格键按下来告知 boot 此串口作为升级通道，串口默认波特率为 9600。通讯波特率可调整。

7.1 时钟配置

因代码中波特率时钟源为 12M，所以 R7F0C004 选型字节设定时钟时必须配成 24M 或者 12M 然后通过分频寄存器得到 12M 作为 boot 的系统时钟源。

此选型字节的时钟源一旦设定后，APP 中是不能修改的，所以 APP 中的系统时钟源必须以 boot 选项字节设定的时钟源为基础。

如果 APP 中需要高精度 RTC 秒输出信号，则 boot 选项字节设定时钟源时只能设定为 24M。

7.2 看门狗配置

看门狗在选项字节中进行配置，通过代码烧录到 ROM，程序中不能修改，所以一旦设定开启或关闭看门狗后就不能在关闭会这开启。boot 中开启了看门狗定时器，所以 APP 中必须得喂狗，否则会发生看门狗复位。

8、软件流程

9、软件仿真

在 cstartup V14.S87 里面改写成#if（0） else 就能仿真调试。

IAR 设置成 Debug 模式。

10、操作流程

二、APP 设计

APP 区程序设计，设计难点在中断向量跳转部分，要实现与 boot 区域的中断函数跳转。

1、APP 设计注意事项

软件 flash 地址从 2000H 开始，在生产 hex 文件时，不需要再定义中断向量以及选项字节，中断向量以及选项字节在 boot 里面设定好，且 APP 不可更改。

在 APP 软件仿真调试时不用通过 boot，在 XCL 文件里面讲中断向量定义，选项字节定义打开即可，在启动函数 CSTARTUP 中将中断跳转表与选项字节配置打开。

2、XCL 连接文件

APP 连接文件在软件仿真时将以下选项打开如图 2.1：

```
//-------------------------------------------
//      Interrupt vector segment.
//-------------------------------------------
//-Z(CODE)INTVEC=02000-02300
-Z(CODE)INTVEC=00000-0007F
//-------------------------------------------
//      CALLT vector segment.
//-------------------------------------------
-Z(CODE)CLTVEC=00080-000BF

//-------------------------------------------
//      OPTION BYTES segment.
//-------------------------------------------
-Z(CODE)OPTBYTE=000C0-000C3

//-------------------------------------------
//      SECURITY_ID segment.
//-------------------------------------------
-Z(CODE)SECUID=000C4-000CD
```

图 2.1

```
//-------------------------------------------
//      Interrupt vector segment.
//-------------------------------------------
-Z(CODE)INTVEC=02000-02300
//-Z(CODE)INTVEC=00000-0007F
//-------------------------------------------
//      CALLT vector segment.
//-------------------------------------------
//-Z(CODE)CLTVEC=00080-000BF

//-------------------------------------------
//      OPTION BYTES segment.
//-------------------------------------------
//-Z(CODE)OPTBYTE=000C0-000C3

//-------------------------------------------
//      SECURITY_ID segment.
//-------------------------------------------
//-Z(CODE)SECUID=000C4-000CD
```

图 2.2

APP 软件方针是需要定义以上选项，不需要软件仿真，需要生产 hex 文件通过 boot 烧录目标板时，则需要将以上选项屏蔽掉图 2.2，因为 boot 程序中对芯片的这些配置有定义。

APP 程序 ROM 分配：

代码从 2000H 开始往后，如下图，具体请参考 RL78APP 代码范例。

```
//-------------------------------------------------------------------
//      Startup, Runtime-library, Near, Interrupt
//      and CALLT functions code segment and near switch.
//-------------------------------------------------------------------
//-Z(CODE)RCODE,CODE=000D8-0FFFF

//-Z(CONST)SWITCH=000D8-0FDFF

-Z(CODE)RCODE,CODE=02000-0FFFF

-Z(CONST)SWITCH=02000-0FDFF
```

3、Cstartup 启动函数

CPU 启动函数主要做 MCU 的环境初始化操作，比如 RAM 区域初始化等，我们将选项字节配置，中断跳转地址在启动函数中配置好，方便 APP 仿真与 hex 文件之间的切换。

```
#if (1)
        ASEGN   RCODE:CODE, 0x2000
        DW      __program_start                 ; 0000H,  Vector  0:
#else
        COMMON  INTVEC:CODE:ROOT(1)
        DW      __program_start                   ;RST_vect
        DW      0FFFFH;INTDBG_vect                (0x02)
        DW      02020H;INTWDTI_vect               (0x04)
        DW      02030H;INTLVI_vect                (0x06)
        DW      02040H;INTP0_vect                 (0x08)
        DW      02050H;INTP1_vect                 (0x0A)
        DW      02060H;INTP2_vect                 (0x0C)
        DW      02070H;INTP3_vect                 (0x0E)
        DW      02080H;INTP4_vect                 (0x10)
        DW      02090H;INTP5_vect                 (0x12)
        DW      020A0H;INTST2_vect                (0x14)
        DW      020B0H;INTSR2_vect                (0x16)
        DW      020C0H;INTSRE2_vect               (0x18)

        DW      02340H;INTSCT1                    (0x68)
        DW      02350H;INTSCR1                    (0x6A)
        DW      02360H;INTSCE1                    (0x6C)
        DW      02370H;BRK_I_vect                 (0x7E)
;-------------------------------------------------------------------
;       // 设置选项字节
;-------------------------------------------------------------------
        COMMON  OPTBYTE:CODE:ROOT(1)

        DC8     07CH
        DC8     0FFH
        DC8     0E0H
        DC8     085H


#endif
```

如果需要 APP 仿真，则在启动函数中写上 if(0)else{}即可，如果需要生成 hex 文件，则选择 if(1)即可。

4、APP 中断

因中断向量表在 boot 区间,所以需要仿真时,中断向量里面的函数入口地址需要从 boot 区间跳转到 APP 区，在 APP 启动函数 CSTARTUP 中完成此操作，不不要 boot 程序的参与，如下图，在中断向量区知道函数跳转地址，然后在跳转后的地址再次跳转到 APP 的中断服务函数。

```
COMMON  INTVEC:CODE:ROOT(1)
DW      __program_start                 ;RST_vect
DW      0FFFFH;INTDBG_vect              (0x02)
DW      02020H;INTWDTI_vect             (0x04)
DW      02030H;INTLVI_vect              (0x06)
DW      02040H;INTP0_vect               (0x08)
DW      02050H;INTP1_vect               (0x0A)
DW      02060H;INTP2_vect               (0x0C)
DW      02070H;INTP3_vect               (0x0E)
DW      02080H;INTP4_vect               (0x10)
DW      02090H;INTP5_vect               (0x12)
DW      020A0H;INTST2_vect              (0x14)
DW      020B0H;INTSR2_vect              (0x16)
DW      020C0H;INTSRE2_vect             (0x18)
DW      020D0H;INTDMA0_vect             (0x1A)
DW      020E0H;INTDMA1_vect             (0x1C)
DW      020F0H;INTCSI00_vect            (0x1E)
DW      02100H;INTTM00_vect             (0x20)
```

中断函数的跳转在 RL78_004_ISR.asm 此汇编文件中实现，比如我需要实现 INTTM00 定时器 0 的中断函数响应，在 CSTARTUP 中，我们定义 INTTM00 的调转地址为 02100H,如下图所示：

```
;********中断服务函数声明**********
    #include "cfi.m"
    EXTERN      r_tau0_channel0_interrupt
    EXTERN      r_rtc_interrupt


                                        ; 返回
    ASEGN   RCODE:CODE, 0x2100

    CALL      r_tau0_channel0_interrupt     ; 跳转中断服务函数
    RETI                                    ; 返回
```

r_tau0_channel0_interrupt 为 APP 真正的中断服务函数，其在 APP 中的定义如下，一定要定义成中断函数，才能在执行时实现现场保护的操作。

```
//#pragma vector = INTTM00_vect
__interrupt  void r_tau0_channel0_interrupt(void)
{
    /* Start user code. Do not edit comment generated here */
     LED=~LED;
    /* End user code. Do not edit comment generated here */
}
```

5、软件流程

6、软件仿真

6.1 XCL 配置

将 XCL 文件中的下图所示区域打开。

6.2 cstartup 配置

将启动函数中的 if else 配置为 if(0)。

IAR 设置成 Debug 模式。

7、操作流程

修订：

Rev.	发行日	修订内容	
		页	要点
1.00	2016.7	-	初始发行

深圳市欣瑞利科技有限公司深圳公司

网址 http://www.superfly.com.cn
http://www.superfly.cn

电话（0755）8316 5141/83165142

深圳市欣瑞利科技有限公司长沙办

电话（0731）8992 5143