

RISC-V MCU启动文件分析

原创 借过风景 于 2022-06-14 09:18:21 发布 255 收藏 4

版权

文章标签: stm32 arm 嵌入式硬件

启动文件由汇编语言编写,是MCU上电复位后第一个执行的程序。主要执行以下内容:

- 初始化gp(global pointer)全局指针寄存器、sp(stack pointer)栈指针寄存器
- 将data数据从flash中加载至RAM中
- 清空bss段数据
- 初始化中断向量表
- 配置系统时钟
- 从Machine模式切换到User模式,进入main函数运行

CH32V103启动文件如下:

```
1  /***** (C) COPYRIGHT *****/
2  * File Name      : startup_ch32v10x.s
3  * Author        : WCH
4  * Version       : V1.0.0
5  * Date          : 2020/04/30
6  * Description   : CH32V10x vector table for eclipse toolchain.
7  *****/
8
9  .section .init,"ax",@progbits /* 声明section 为 .init */
10 .global _start /* 指明标签_start的属性为全局性的 */
11 .align 1
12 _start: /* 标签_start处 */
13 j handle_reset /* 跳转至 handle_reset处 */
14 .word 0x00000013 /* 内核设计需要,不用关注 */
15 .word 0x00000013
16 .word 0x00000013
17 .word 0x00000013
18 .word 0x00000013
19 .word 0x00000013
20 .word 0x00000013
21 .word 0x00000013
22 .word 0x00000013
23 .word 0x00000013
24 .word 0x00000013
25 .word 0x00000013
26 .word 0x00100073
27 .section .vector,"ax",@progbits
28 .align 1
29 _vector_base: /* 中断向量表 */
30 .option norvc;
31 j _start
32 .word 0
33 j NMI_Handler /* NMI Handler */
34 j HardFault_Handler /* Hard Fault Handler */
35 .word 0
36 .word 0
37 .word 0
38 .word 0
39 .word 0
40 .word 0
41 .word 0
42 .word 0
43 j SysTick_Handler /* SysTick Handler */
44 .word 0
45 j SW_handler /* SW Handler */
46 .word 0
47 /* External Interrupts */
48 j WWDG_IRQHandler /* Window Watchdog */
49 j PVD_IRQHandler /* PVD through EXTI Line detect */
50 j TAMPER_IRQHandler /* TAMPER */
51 j RTC_IRQHandler /* RTC */
52 j FLASH_IRQHandler /* Flash */
53 j RCC_IRQHandler /* RCC */
54 j EXTI0_IRQHandler /* EXTI Line 0 */
55 j EXTI1_IRQHandler /* EXTI Line 1 */
56 j EXTI2_IRQHandler /* EXTI Line 2 */
57 j EXTI3_IRQHandler /* EXTI Line 3 */
58 j EXTI4_IRQHandler /* EXTI Line 4 */
59 j DMA1_Channel1_IRQHandler /* DMA1 Channel 1 */
60 j DMA1_Channel2_IRQHandler /* DMA1 Channel 2 */
61 j DMA1_Channel3_IRQHandler /* DMA1 Channel 3 */
62 j DMA1_Channel4_IRQHandler /* DMA1 Channel 4 */
63 j DMA1_Channel5_IRQHandler /* DMA1 Channel 5 */
64 j DMA1_Channel6_IRQHandler /* DMA1 Channel 6 */
65 j DMA1_Channel7_IRQHandler /* DMA1 Channel 7 */
66 j ADC1_2_IRQHandler /* ADC1_2 */
67 .word 0
68 .word 0
69 .word 0
70 .word 0
71 j EXTI9_5_IRQHandler /* EXTI Line 9..5 */
72 j TIM1_BRK_IRQHandler /* TIM1 Break */
73 j TIM1_UP_IRQHandler /* TIM1 Update */
74 j TIM1_TRG_COM_IRQHandler /* TIM1 Trigger and Commutation */
75 j TIM1_CC_IRQHandler /* TIM1 Capture Compare */
76 j TIM2_IRQHandler /* TIM2 */
77 j TIM3_IRQHandler /* TIM3 */
78 j TIM4_IRQHandler /* TIM4 */
79 j I2C1_EV_IRQHandler /* I2C1 Event */
80 j I2C1_ER_IRQHandler /* I2C1 Error */
81 j I2C2_EV_IRQHandler /* I2C2 Event */
82 j I2C2_ER_IRQHandler /* I2C2 Error */
83 j SPI1_IRQHandler /* SPI1 */
84 j SPI2_IRQHandler /* SPI2 */
85 j USART1_IRQHandler /* USART1 */
86 j USART2_IRQHandler /* USART2 */
87 j USART3_IRQHandler /* USART3 */
88 j EXTI15_10_IRQHandler /* EXTI Line 15..10 */
89 j RTCArm_IRQHandler /* RTC Alarm through EXTI Line */
90 j USBWakeUp_IRQHandler /* USB Wakeup from suspend */
91 j USBHD_IRQHandler /* USBHD */
92
93 .option rvc;
94
95 .section .text, "ax", @progbits /* 中断服务程序弱定义 */
96 .weak NMI_Handler
97 .weak HardFault_Handler
98 .weak SysTick_Handler
99 .weak SW_handler
100 .weak WWDG_IRQHandler
101 .weak PVD_IRQHandler
102 .weak TAMPER_IRQHandler
103 .weak RTC_IRQHandler
104 .weak FLASH_IRQHandler
105 .weak RCC_IRQHandler
106 .weak EXTI0_IRQHandler
107 .weak EXTI1_IRQHandler
108 .weak EXTI2_IRQHandler
109 .weak EXTI3_IRQHandler
110 .weak EXTI4_IRQHandler
111 .weak DMA1_Channel1_IRQHandler
112 .weak DMA1_Channel2_IRQHandler
113 .weak DMA1_Channel3_IRQHandler
114 .weak DMA1_Channel4_IRQHandler
115 .weak DMA1_Channel5_IRQHandler
116 .weak DMA1_Channel6_IRQHandler
117 .weak DMA1_Channel7_IRQHandler
118 .weak ADC1_2_IRQHandler
119 .weak EXTI9_5_IRQHandler
120 .weak TIM1_BRK_IRQHandler
121 .weak TIM1_UP_IRQHandler
122 .weak TIM1_TRG_COM_IRQHandler
123 .weak TIM1_CC_IRQHandler
124 .weak TIM2_IRQHandler
125 .weak TIM3_IRQHandler
126 .weak TIM4_IRQHandler
127 .weak I2C1_EV_IRQHandler
128 .weak I2C1_ER_IRQHandler
129 .weak I2C2_EV_IRQHandler
130 .weak I2C2_ER_IRQHandler
131 .weak SPI1_IRQHandler
132 .weak SPI2_IRQHandler
133 .weak USART1_IRQHandler
134 .weak USART2_IRQHandler
135 .weak USART3_IRQHandler
136 .weak EXTI15_10_IRQHandler
137 .weak RTCArm_IRQHandler
138 .weak USBWakeUp_IRQHandler
139 .weak USBHD_IRQHandler
140
141 .section .text.handle_reset,"ax",@progbits
142 .weak handle_reset
143 .align 1
144 handle_reset: /* handle_reset起始位置 */
145 .option push
146 .option norelax
147 la gp, __global_pointer$ /* 将ld文件中的标签__global_pointer所处的地址值赋给gp寄存器 */
148 .option pop
149 l:
150 la sp, _eusrstack /* 将ld文件中的标签_eusrstack所处的地址值赋给sp寄存器 */
151 2:
152 /* Load data section from flash to RAM */
153 la a0, _data_lma
154 la a1, _data_vma
155 la a2, _edata
156 bgeu a1, a2, 2f
157 l:
158 lw t0, (a0)
159 sw t0, (a1)
160 addi a0, a0, 4
161 addi a1, a1, 4
162 bltu a1, a2, 1b
163 2:
164 /* clear bss section */
165 la a0, _sbss
166 la a1, _ebss
167 bgeu a0, a1, 2f
168 l:
169 sw zero, (a0)
170 addi a0, a0, 4
171 bltu a0, a1, 1b
172 2:
173 /* enable all interrupt */ /* csrs, 根据寄存器中每个为1的位,把CSR寄存器对应位置位 */
174 li t0, 0x88
175 csrw mstatus, t0 /* 状态寄存器mstatus赋值为0x88, 打开所有中断,设置MPP值为00 */
176 la t0, _vector_base
177 ori t0, t0, 1
178 csrw mtvec, t0 /* 将中断向量表的首地址赋给mtvec寄存器(中断发生时PC的地址) */
179
180 jal SystemInit /* 设置MCU系统时钟 */
181 la t0, main
182 csrw mepc, t0
183 mret
184
185 /*
186 mret返回指令(M模式特有的指令),调用该指令会进行如下操作:
187 - 将PC指针设置为mepc的值
188 - 将mstatus的MPIE域复制到MIE来恢复之前的中断使能
189 - 将权限模式设置为mstatus的MPP域中的值。
190
191 芯片上电默认进入的是机器模式,通过将mstatus中的MPP值设置为00(00: User, 01: Supervisor, 11: Machine),
192 并将main函数的地址赋给mepc,调用mret,使得用户在进入main函数运行时,芯片由机器模式切换为用户模式。
193
194 */
```