昆明学院学报　2016,38(6):48-51,84　ISSN 1674-5639　CN 53-1211/G4
Journal of Kunming University

基于 ARM Cortex M0+内核的 MQX-Lite RTOS 启动流程剖析

文　瑾
(昆明学院 信息技术学院,云南 昆明 650214)

摘要:轻量级 MQX 实时操作系统(MQX-Lite RTOS)是标准 MQX 的轻量级版本,内核精简,针对资源有限的 MCU,可在最小4 KB 的 RAM 空间运行. 因此详细分析了基于 ARM Cortex M0+内核的 KL25 中 MQX-Lite 的启动过程,包括汇编语言的芯片启动,提出启动过程中函数的设置建议,阐述实现机理,给出执行流程,并对关键源码进行了剖析. 为开发人员在精减操作系统环境下开发嵌入式产品,提升系统的启动时间、减少资源消耗、提高执行效率和安全特性等方面提供借鉴和参考.

关键词:RTOS;MQX-Lite;ARM Cortex M0+;启动流程
中图分类号:TP316.2　**文献标识码**:A　**文章编号**:1674-5639(2016)06-0048-04
DOI:10.14091/j.cnki.kmxyxb.2016.06.011

Analysis of the Boot Process of MQX-Lite RTOS Based on the Kernel of ARM Cortex M0+

WEN Jin
(College of Information Technology, Kunming University, Kunming, Yunnan, China　650214)

Abstract: Lightweight MQX Real-Time Operating System (MQX-Lite RTOS) with a Micro kernel of the standard MQX, aiming at the limited resources of MCU, can run in the smallest 4 KB RAM space. Here we analyzed in details the boot process of MQX-Lite based on the kernel of ARM Cortex M0+ in KL25, including chip start. We put forward the suggestion of the parameter of setting the boot function in the starting process, the implementation mechanism, the executive process, and analyze the key source so as to get the reference to help the developers in developing embedded products to use streamline operating system, improve the boot time of the system, reduce the resource consumption, and raise executive efficiency and security features.
Key words: RTOS;MQX-Lite;ARM Cortex M0+;boot process

飞思卡尔(现属于恩智浦半导体公司)2014 年推出的嵌入式轻量级 MQX 实时操作系统(MQX-Lite V1.1)是标准 MQX 的轻量级版本,它只含标准 MQX 内核的必要组件和某些组件的轻量级版本,其内核精简,是标准 MQX 的一个真子集. 主要针对资源(FLASH ROM 和 RAM)有限的 MCU,可在最小 4 KB的 RAM 空间运行[1-2].

相对通用操作系统,在嵌入式操作系统环境下开发工程对系统的稳定性、实时性、启动时间等方面要求更加严格,同时对资源消耗、代码执行效率等方面要求也更高. 由于嵌入式操作系统对时钟、内核数据区、任务调度策略、中断系统等内容均在启动过程中配置,清晰理解启动流程有助于开发人员设计出更加稳定高效的嵌入式应用系统. 段红祥等[3]在 Linux 操作系统下提出加速设备初始化策略;王亚刚[4]在对嵌入式 Bootloader 分析的基础上,提出了移植 Bootloader 的一般方法;廖孝勇等[5]在分析 uC/OS-II 内核启动流程基础上,设计了基于 uC/OS-II 内核的专用引导程序;冯小天等[6]在深入分析 OSEK 操作系统的基础上,给出了基于 OSEK 操作系统标准的车载嵌入式操作系统内核 OSEKernel 的结构. 通过众多嵌入式操作系统方面的文献研究发现,对启动流程的分析改进,

收稿日期:2016-10-31
基金项目:昆明学院物联网应用技术科研创新团队基金资助项目(2015CXTD04).
作者简介:文瑾(1963—),男,云南昆明人,副教授,主要从事嵌入式系统、物联网技术研究.

切实可以提升系统的启动时间、执行效率、可靠性、安全特性等各方面性能[7].

目前业界已有许多针对 Linux 和 uC/OS 等嵌入式操作系统的启动研究,鲜有关于 MQX-Lite 的启动流程的剖析研究. 本文将在 Kinetis Design Studio 3.0.0 IDE 嵌入式软件集成开发环境下,以 ARM Cortex-M0 + 为内核的 SD-FSL-KL25 评估板作为硬件平台中,利用苏州大学飞思卡尔嵌入式系统实验中心发布的 AMQXFW 工程框架[8],对 MQX-Lite 操作系统启动流程进行研究. 剖析从芯片上电,调用入口函数,启动操作系统直至进入启动调度器的全过程,并分析了实现机理,给出执行流程和设置建议,对关键源代码进行剖析. 为在操作系统嵌入低成本、低功耗、资源有限的 MCU 中开发应用产品,提供了很好的借鉴和参考.

1　MQX-Lite 的启动流程概要

基于 ARM-Cortex 的芯片为复杂的片上系统集成 SOC,这种复杂的系统里多数硬件都可由软件来设置需要的工作状态,因此在应用程序启动之前,会有一段专门的启动代码来完成芯片硬件的初始化,本文称之为芯片硬件启动,它的目的是将系统的软硬件环境设置到一个合适的状态,是进入 C 语言的 main 函数前执行的一段代码,一般用汇编语言来编写. 完成芯片硬件启动后,系统就将控制权交给操作系统内核的引导程序,开始 MQX-Lite 内核的加载,本文称为操作系统启动[9]. 把 MQX-Lite 的启动采用分阶段的分析方法,是为了增强其通用性,更好地理解启动流程. 如图 1 所示.

图1　MQX-Lite的启动阶段

第 1 阶段(芯片硬件启动)主要完成硬件的启动,与无操作系统的启动是一致的,包括存储映射区域解析、中断向量表解析、引导启动 boot、程序启动 startup,完成初始化 RAM 区、设置处理器的时钟频率、关闭看门狗模块,以及基本的硬件初始化等,如图 2 所示. 因为是对硬件操作,这阶段大多采用了汇编语言编写,所以移植性较差,但代码很少,调试和升级都较为简单.

第 2 阶段(操作系统启动)完成了一些较复杂的功能,包括初始化内核数据区、初始化外设、调度系统的初始化与启动等,如图 2 所示. 这阶段代码量很大,全部用 C 语言编写,因此移植性和扩展性较好.

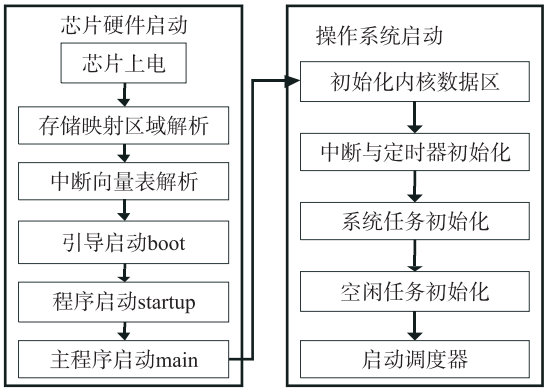

图2　MQX-Lite启动流程图

2　MQX-Lite 芯片硬件启动过程解析

ARM Cortex-M0 + 处理器的芯片上电后自动执行 vectors. c 文件中物理地址 0x00000000 处指令,取出第 1 个表项的内容(&_SP_INIT,即 0x00000000 ~ 0x00000003 地址内容,为 0x20003000,在链接文件 ProcessorExpert. ld 中),赋给内核寄存器 SP(Stack Pointer,堆栈指针),完成堆栈指针初始化. 芯片内部机制将第 2 个表项的内容(&_boot,即 0x00000004 ~ 0x00000007 地址内容,为启动函数 startup 的首地址),赋给内核寄存器 PC(Program Counter,程序计数器).

由于内核寄存器 PC 存放启动函数 startup. c 的首地址,因而转去执行 startup. c 函数,进入 C 语言函数部分. 依次执行禁用看门狗、复制中断向量表至 RAM、清零未初始化 BSS 数据段、将 ROM 中的初始化数据拷贝到 RAM 中、初始化系统时钟、使能端口时钟、进入主函数 main[10]. startup. c 函数具体定义如下:

```
void startup(void)
{ wdog_disable();//禁用看门狗
m_zero_fill_bss ();//清零未初始化 BSS 数据段
m_data_seg_init ();//将 ROM 中的数据拷贝到 RAM 中
sys_init();//初始化系统时钟
sys_pin_enable_port();//使能端口时钟
main();//进入主函数
}
```

除了设置总线时钟频率的初始化系统时钟函数 clock_init( ) 在 sysinit. c 文件中定义外,其余函数都在 startup. c 中定义. 可根据实际需要决定是否启动看门狗;清零未初始化 BSS 数据段涉及链接文件中定义的全局变量和静态变量,包括_START_BSS 和_END_BSS 等,在实际的嵌入式工程开发中,若需要在热启动后保存现场,不需要清零,可将其注释掉;将 ROM 中的数据拷贝到 RAM 中,是为了方便 main( ) 函数调用;初始化系统时钟和使能端口时钟,主要功能是配置系统时钟,芯片主时钟是利用 MCG(Multipurpose Clock Generator,多功能时钟发生器)模块中的锁相环 PLL 模块,由无源晶振信号经锁相环 PLL 编程得到的.

3 MQX-Lite 系统启动流程源码剖析

在 main 函数中,将 MQX-Lite 初始化参数传入_mqx函数,调用操作系统的入口函数_mqx( ) 开始启动 MQX-Lite:初始化内核数据区和 MQX-Lite 运行所需外设,创建系统初始化任务和空闲任务,启动任务调度器.

3.1 初始化内核数据区

因为 MQX-Lite 操作系统的资源使用情况和运行状态存储在内核数据区中,为了保证操作系统工作正常,要为其内核数据区设置合适的初始值.

1)创建内核数据区访问指针 kernel_data. 内核数据区的空间是在链接命令文件中定义的,属于 BSP(Board Support Package,板级支持包)的内容,为了提高相对于 BSP 的独立性,此处创建了一个内核数据区访问指针,并定义了访问接口,以后的程序中,只要将 kernel_data 定义为“volatile KERNEL_DATA_STRUCT_PTR”类型,再使用“_GET_KERNEL_DATA(kernel_data)”宏函数,即可访问内核数据区. 关键代码如下:

```
kernel_data = (KERNEL_DATA_STRUCT_PTR) //获得内核数据区的首地址
_ALIGN_ADDR_TO_HIGHER_MEM(mqx_init->START_OF_KERNEL_MEMORY);//地址对齐
_SET_KERNEL_DATA(kernel_data);//将内核数据区的首地址指向全局变量区
```

2)填充内核数据区. MQX-Lite 启动过程中要将系统设定的初始值填充到内核数据区. 包括赋值操作系统版本号与厂商名称、置内核数据区所有字段为 0、验证内核数据区的读写功能、复制 MQX-Lite 初始化数据到内核数据区、初始化内核数据区数据结构、创建系统默认的内存池等操作.

3)初始化中断栈. 为中断栈分配存储空间并在内核数据区中登记. 当产生嵌套中断时,系统将把嵌套中断的上下文保存在中断栈中,当高优先级中断服务例程执行返回后,从中断栈中恢复低优先级中断上下文继续执行.

4)创建系统空闲任务栈. 系统空闲任务是 MQX-Lite 自动创建,自行维护,无需用户干涉. 当系统空闲任务被中断阻塞时,使用系统栈保存空闲任务的上下文. 系统栈指针链接在 kernel_data->SYSTEM_TD 上,可由全局变量_mqx_system_stack 操作控制.

5)创建就绪任务队列数组. 任务调度系统操纵的主要对象就是就绪任务队列数组,系统初始化中,根据任务模板列表中,用户定义的任务最低优先级创建就绪队列数目,并根据优先级将它们排序. 刚创建的每个就绪任务队列中仅包含一个队列头结构,当有任务进入就绪态后,该任务的描述符就挂载到相应优先级的就绪任务队列上.

6)创建任务操作轻量级信号量. 创建和删除任务时均需对内核数据区进行同步访问,这就需要一个信号量来保证同步访问能够正确进行.

```
_lwsem_create((LWSEM_STRUCT_PTR)&kernel_data->TASK_CREATE_LWSEM, 1).
```

3.2 初始化外设

在系统启动过程中,使用 init_bsp. c 函数初始化硬件平台,几个主要的初始化如下.

1)中断系统的初始化. 为中断系统分配动态中断向量链接表头节点,初始化动态中断向量表. 当中断事件产生后,MQX-Lite 的中断处理系统将在中断向量链表中查询相应的服务函数并调用执行. 动态中断向量表机制使中断服务例程可在系统运行过程中安装更新,根据不同需要可以为同一中断源动态配置中断服务例程,这样使得中断处理更加灵活. 关键代码如下:

```
//初始化 MQX-Lite 的中断管理系统
result = _psp_int_init(FIRST_INTERRUPT_VECTOR_USED, LAST_INTERRUPT_VECTOR_USED);
//注册 systick 中断服务函数到中断管理模块
_int_insall_isr(BSP_TIMER_INTERRUPT_VECTOR);
```

```
//将 systick 设置为系统滴答定时器中断向量
_time_set_timer_vector(BSP_TIMER_INTERRUPT_VECTOR);
//为 systick 中断安装 ISR 服务例程
_time_notify_kernel().
```

2)系统时间滴答的初始化. 操作系统运行时,需要一个滴答(tick)时钟信号,驱动时间管理系统工作. ARM Cortex-M0 + 处理器内部提供了一个可编程的 Systick 定时器模块,它被捆绑在 NVIC(Nested Vectored Interrupt Controller,嵌套向量中断控制器)模块上,Systick 的有效位是 24 位,采用减 1 计数方式工作,当到 0 时,产生中断,中断向量号为 15,它就是 Systick 中断,其中断服务例程负责执行需要周期性运行的任务,并更新系统时间. 初始化过程如下:

```
SYST_CSR = 0; //关闭 SYSTICK
SYST_CVR = 0;//清除计数器
//设定倒计时计数值
SYST_RVR = BSP_CORE_CLOCK / BSP_ALARM_FREQUENCY - 1;
//设定 SysTick 优先级
SCB_SHPR3 | = SCB_SHPR3_PRI_15(CORTEX_PRIOR(BSP_TIMER_INTERRUPT_PRIORITY));
//使用内核时钟,倒计时到 0 时产生 SYSTICK 中断,使能 SYSTICK
SYST_CSR = 7.
```

3.3　初始化任务

1)设定缺省时间片. 若使用时间片轮转调度机制,且未在任务模板中明确指定时间片长度,则使用系统缺省的设定的时间片长度. 在 MQX 启动的过程中对系统缺省时间片长度进行设定.

```
//设定内核缺省时间片长度
#define MQX_DEFAULT_TIME_SLICE ((_mqx_uint)10)
```

缺省时间片长度的配置值 MQX_DEFAULT_TIME_SLICE 在“…\MQX\include”文件夹中的“mqxlite_prv.h”文件中定义,代码中默认将时间片设置为 10 ms.

2)空闲任务初始化. 当其他应用任务不再执行,系统仍需运行空闲任务,以保证 MQX-Lite 的调度系统处于工作状态. 在 MQX-Lite 操作系统创建空闲任务后,会将该任务设定为“就绪”态. 空闲任务位于“…\kernel\idletask.c”文件中,它是一个永久循环,执行一个 3 层嵌套循环的加法过程.

3)初始化任务. 完成创建内部同步所需的轻量级信号量并开启系统时钟,创建空闲任务和自启动任务并加入就绪队列,通过“…\kernel\mqxlite.c”函数完成. 关键代码如下:

```
//创建轻量级信号量
_lwsem_create((LWSEM_STRUCT_PTR) & kernel_data->TASK_CREATE_LWSEM, 1);
//创建空闲任务
#if MQX_USE_IDLE_TASK
td_ptr = _task_init_internal((TASK_TEMPLATE_STRUCT_PTR)&kernel_data->IDLE_TASK_TEMPLATE,
kernel_data->ACTIVE_PTR->TASK_ID,
_task_ready_internal(td_ptr);//加入就绪队列
#endif
//创建自启动任务
td_ptr = _task_init_internal(template_ptr,
kernel_data->ACTIVE_PTR->TASK_ID,
template_ptr->CREATION_PARAMETER,
_task_ready_internal(td_ptr);//加入就绪队列
```

3.4　启动任务调度系统

完成初始化内核数据区、初始化外设、初始化任务后,系统启动调度程序_sched_start_internal(),将控制权转移给调度系统,此后就在调度系统和中断处理系统的管理下,根据调度策略决定哪一个任务运行、何时运行、运行多久. 在 MQX-Lite 中调度策略有 3 种. 1)SCHED_FIFO:优先级抢占调度策略;2)SCHED_RR:时间片轮转调度策略;3)SCHED_OTHER:可由工程开发人员自行定义扩展指定任务队列. MQX-Lite 的调度策略是在系统上电初始化时在 mqxlite.h 中设置,默认为 SCHED_FIFO 调度.

4　结语

本文在对 MQX-Lite 操作系统的启动流程的研究基础上,借鉴相关文献以及研究人员对其他操作系统研究分析的思想和方法,将 MQX-Lite 启动流程也分为芯片硬件启动和操作系统启动两个过程,其中芯片硬件启动与无操作系统相同. 在解析芯片硬件的启动

(下转第 84 页)

2002,33(6):449 - 455.
[5] YEMM E W, WILLIS A J. The estimation of carbohydrates in plant extracts by anthrone[J]. Biochemical Journal, 1954,57(3):508 - 514.
[6] BRADFORD M M. A rapid and sensitive method for the quantitation of microgram quantities of protein utilizing the principle of protein-dye binding[J]. Anal Biochem, 1976, 72:248 - 254.
[7] 赵宏伟,田秀珠,王波. 差异蛋白质组学研究与应用进展[J]. 医学与哲学,2006,27(8):45 - 47.
[8] 姜红,田丽萍. 作物的耐盐生理生化特性[J]. 山西农业科学,2007,35(4):42 - 44.
[9] 王军,梁长东,张灿宏,等. 作物耐盐碱性鉴定评价方法探讨[J]. 大麦与谷类科学,2010(3):35 - 36.
[10] JAMES R A, BLAKE C, BYRT C S, et al. Major genes for Na^+ exclusion, *Nax*1 and *Nax*2 (wheat *HKT*1;4 and *HKT*1;5), decrease Na^+ accumulation in bread wheat leaves under saline and waterlogged conditions[J]. Journal of Experimental Botany, 2011, 62(8):2939 - 2947.
[11] ROZEMA J, FLOWERS T. Crops for a salinized world[J]. Science, 2008, 322(5907):1478 - 1480.
[12] WANG W, VINOCUR B, ALTMAN A. Plant responses to drought, salinity and extreme temperatures: towards genetic engineering for stress tolerance[J]. Planta, 2003, 218(1):1 - 14.
[13] 周建刚,丁少丽,袁金友,等. NaCl 胁迫对苎麻叶片中 SOD、POD 及 CAT 活性的影响[J]. 武汉科技学院学报,2007,20(4):35 - 37.
[14] 陈洁. 水稻幼苗耐盐性的定量鉴定及耐盐生理生化研究[D]. 海口:华南热带农业大学,2003.
[15] HOQUE M A, BANU M N A, NAKAMURA Y, et al. Proline and glycinebetaine enhance antioxidant defense and methylglyoxal detoxification systems and reduce NaCl-induced damage in cultured tobacco cells[J]. Journal of Plant Physiology, 2008, 165(8):813 - 824.
[16] ISLAM M T, ARA M I, HOSSAIN M A, et al. Identification of tomato genotypes for salt tolerance[J]. International Journal of Sustainable Crop Production, 2011, 6(1):17 - 21.
[17] 陈洁,林栖凤. 植物耐盐生理及耐盐机理研究进展[J]. 海南大学学报(自然科学版),2003,21(2),177 - 182.

(上接第 51 页)

过程中,对各初始化函数功能进行了详细说明,并从实际工程的需求对其设置提出了相应的建议. 此外,分析了进入操作系统后的初始化内核数据区、初始化外设、初始化任务,以及启动任务调度器的过程及关键源代码. 为工程开发人员对启动流程的分析改进及提升系统启动时间、执行效率、可靠性、安全特性等方面性能提供参考和借鉴.

[参考文献]

[1] Freescale. Freescale MQX Lite RTOS reference manual [EB/OL]. [2016 - 04 - 28]. http://www.freescale.com/mqx.
[2] Freescale. KL25 Sub-family reference manual [EB/OL]. [2016 - 06 - 29]. http://www.freescale.com/mqx.
[3] 段红祥,孙棣华,刘卫宁,等. 基于内核启动优化的嵌入式 Linux 快速启动方案[J]. 计算机工程与设计,2009, 30(1):16 - 18.
[4] 王亚刚. 嵌入式 Bootloader 机制的分析与移植[J]. 计算机工程,2010,36(6):267 - 269.
[5] 廖孝勇,孙棣华,赵君杰. 基于 uC/OS - II 的 ARM9 引导程序设计与实现[J]. 控制工程,2011,18(6):997 - 1000.
[6] 冯小天,陈香兰,李曦. OSEK/VDX 标准的车载嵌入式操作系统内核的结构与设计方法[J]. 计算机应用与软件,2009,26(9):49 - 51.
[7] 罗蕾. 嵌入式实时操作系统及应用开发[M]. 北京:北京航空航天大学出版社,2011.
[8] 王宜怀,朱仕浪,姚望舒. 嵌入式实时操作系统 MQX 应用开发技术:ARM Cortex-M 微处理器[M]. 北京:电子工业出版社,2014.
[9] STALLINGS W. 操作系统:精髓与设计原理[M]. 陈向群,陈渝,译. 北京:机械工业出版社,2011.
[10] 王宜怀,朱仕浪,郭云. 嵌入式技术基础与实践:ARM Cortex-M0 + Kinetis L 系列微控制器[M]. 3 版. 北京:清华大学出版社,2013.