参考文献

Application Note

C2000 F29x MCU 的实时固件更新（无器件复位）

Sira Rao, Alex Wasinger

摘要

本文档详细介绍了带 A/B 交换闪存存储体的器件（如 F29H85x）在无器件复位功能时的实时固件更新 (LFU)。

内容

1 简介.....2
2 LFU 的构建块.....2
3 建议设计的详细信息.....2
3.1 闪存组组织.....2
3.2 影响性能的 LFU 概念和因素.....3
3.3 LFU 的硬件支持.....3
4 应用程序 LFU 流程.....4
5 结果和结论.....5
6 实现示例.....5
7 摘要.....5
8 参考资料.....6

1 简介

对服务器电源等应用而言，系统应持续运行，减少停机次数。通常在因错误修复、新增功能和/或改进而进行固件升级期间，系统无法提供服务，导致停机。冗余模块可以解决这个问题，但会使系统总体成本增加。还有一种备选方法被称为实时固件更新 (LFU)，可在系统运行期间更新固件。

在 F29x MCU 上，无论有无器件复位均可切换到新固件。本用户指南适用于在无复位情况下切换。有关在有复位情况下的实现，请参阅 F29H85x 的固件无线升级示例。**请注意，此示例适用于 HS-FS（即非安全）器件。**

2 LFU 的构建块

LFU 设计包含多个构建块：

- 发出 LFU 命令并发送应用映像的桌面主机应用
- 目标器件闪存上的 LFU 辅助引导加载程序 (SBL)，用于与主机通信并编程应用映像
  - 有关实现示例讨论，请参阅 F29H85x 的串行闪存编程应用手册
- 已实现 LFU 支持的 MCU 应用
- 支持 A/B 交换闪存存储体的目标 MCU
- 支持 LFU 的编译器（这将在未来的编译器版本中提供）

3 建议设计的详细信息

3.1 闪存组组织

对于 HS-FS 器件，LFU 目前仅在存储体模式 1 下可用，它具有以下闪存 MAIN 区域地址映射：

FRI	READ PORT	SIZE	START ADDRESS	END ADDRESS	FLASH BANKS (SWAP = 0)	FLASH BANKS (SWAP = 1)
FRI-1 (CPU1 program)	RP0	1MB	0x10000000	0x100FFFFF	FLC1.B0/B1	FLC1.B2/B3
	RP1	1MB	0x10100000	0x101FFFFF	FLC2.B0/B1	FLC2.B2/B3
	RP2	1MB	0x10200000	0x102FFFFF	N/A	N/A
	RP3	1MB	0x10300000	0x103FFFFF	N/A	N/A
FRI-2 (CPU3 program)	RP0	1MB	0x10400000	0x104FFFFF	N/A	N/A
	RP1	1MB	0x10500000	0x105FFFFF	N/A	N/A
FRI-3 (Update region)	RP0	1MB	0x10600000	0x106FFFFF	FLC1.B2/B3	FLC1.B0/B1
	RP1	1MB	0x10700000	0x107FFFFF	FLC2.B2/B3	FLC2.B0/B1

图 3-1. 闪存 MAIN 区域地址映射

当 SWAP = 0 时，活动的闪存存储体为 FLC1.B0/B1 和 FLC2.B0/B1。当 SWAP = 1 时，活动的闪存存储体为 FLC1.B2/B3 和 FLC2.B2/B3。活动和非活动的闪存存储体分别针对 SBL 和应用空间拆分成多个段，如下所示：

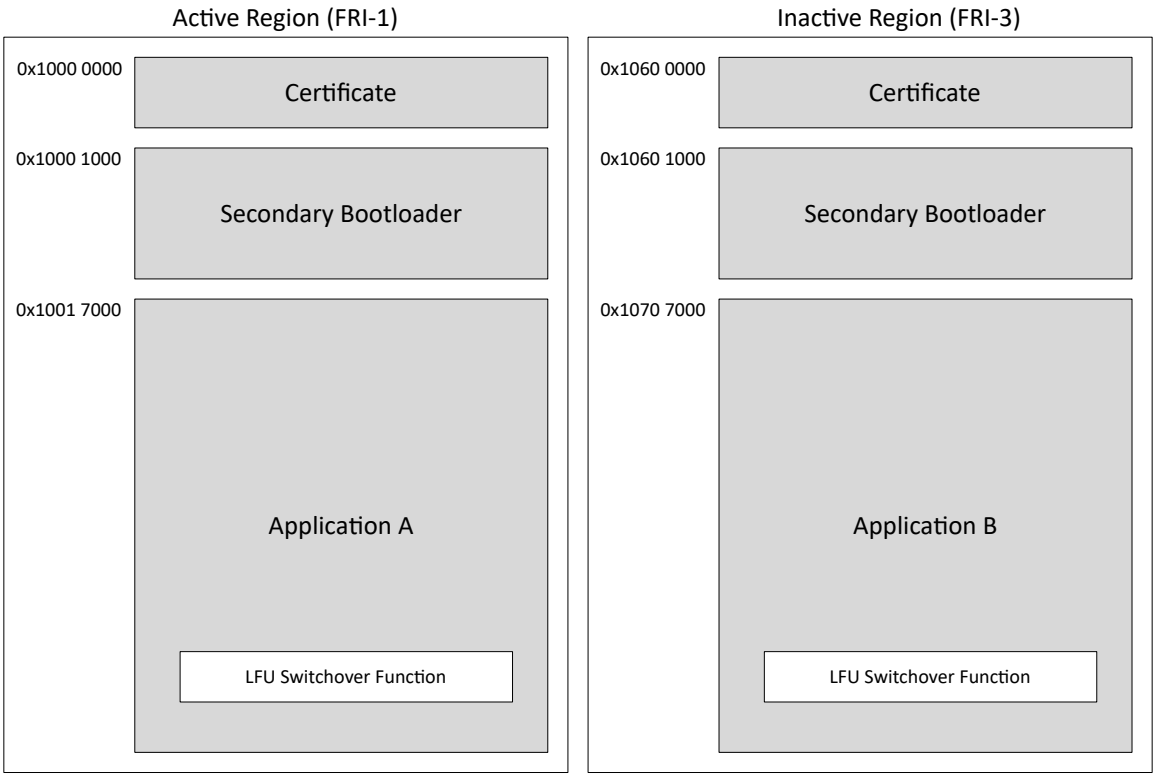

图 3-2. 闪存组

执行固件更新时，非活动的闪存存储体将被擦除，并使用新的证书、SBL 和应用程序映像对其进行编程。

3.2 影响性能的 LFU 概念和因素

创建 LFU 就绪固件时的关键注意事项是 LFU 和 LFU 切换时间期间的操作连续性。两者密切相关。运行连续性是通过状态的持续性来实现的，在固件升级期间将 RAM 中当前的静态变量和全局变量保存在相同的地址，并避免在新固件激活时重新初始化这些变量。LFU 切换时间是器件在更改固件时禁用中断情况下所花费的时间，并通过 A/B 交换闪存存储体和矢量表大幅减少时间。

3.3 LFU 的硬件支持

3.3.1 A/B 可交换闪存存储体

如 *图 3-1* 中所示，有对活动和非活动的闪存存储体的硬件支持。可以通过对 SSU_GEN_REGS.BANKMAP 寄存器进行“异或”运算来随时交换这些存储体。

3.3.2 中断向量表交换

更新中断矢量表是 LFU 例程中其中一个最耗时的部分。为了减少切换时间，会在应用执行过程中更新影子矢量表。在切换期间，当禁用中断时，使用 PIPE_REGS.INT_VECT_MAPPING 寄存器交换影子矢量表和活动的矢量表。

4 应用程序 LFU 流程

在以下部分，当提及代码和示例项目时，SBL = flash_based_uart_sbl_with_lfu 和 LFU 应用 = lfu_uart_cpu1_application。

与 LFU 相关的详细步骤概述如下：

1. *引导至 SBL 并执行应用*：在器件复位时，在 SBL 中开始执行。通信外设经过初始化，执行转移到应用的 codestart 例程。在代码中，这发生在 SBL 的 main() 函数 (flash_based_uart_sbl.c) 中。
2. *启动 LFU*：用户通过主机发起的 LFU 命令在目标 MCU 上调用 LFU。
3. *在应用中接收 LFU 命令*：应用在其通信外设 ISR 中接收 LFU 命令。在代码中，这发生在 LFU 应用的 SBL_uartNotifyISR() (uart_led_blinky_cpu1.c) 中，该函数设置一个标志，指示应用跳回 SBL，以处理命令。
4. *处理 LFU 命令*：应用转移回 SBL，以解析和处理 LFU 命令。在代码中，这发生在 SBL 的 commandJumpTable() 函数 (flash_based_uart_sbl.c) 中，该函数读取命令并执行相应的操作。
5. *将新固件和程序下载到闪存*：SBL 中的 LFU 流从主机接收应用映像，并将其编程到非活动的闪存存储体中。旧固件中的后台任务函数已停止执行，但控制 ISR 继续可用，以保持应用功能不受影响。在代码中，这发生在 SBL 的 cpu1LFUFlow() 函数 (sbl_command_flow.c) 中。
6. *返回到应用并报告成功*：如果固件映像编程成功，SBL 转移回应用并报告成功的 LFU 命令
7. *将 LFU 切换函数复制到 RAM 中*：应用将新固件的 LFU 切换函数从不可执行的非活动的闪存复制到 RAM 中，然后转移到其中。由于旧固件执行新固件的切换函数的复制，因此它必须位于固定地址。在代码中，LFU 应用的 performLFUSwitchover() 函数 (lfu_switchover.c) 负责此操作。
8. *初始化新 PIPE 矢量表*：影子 PIPE 矢量表中会填充新固件的 ISR。在代码中，这在 LFU 应用的 lfuSwapBanks() 函数 (lfu_switchover.c) 中完成。
9. *等待更优 LFU 切换点*：使用一个带有软件标志的简单状态机来确定控制 ISR 的结束和空闲时间的开始。这是执行切换的更优时间，因为它更大限度地利用控制循环中断之间的空闲时间。在代码中，lfuSwapBanks 函数 (lfu_switchover.c) 通过读取 lfu_switchover_proceed 标志来实现这一点，该标志在 INT_myCPUTIMER0_ISR() 例程 (uart_led_blinky_cpu1.c) 末尾设置。
10. *执行 LFU 切换*：
    a. 禁用中断
    b. 交换中断表
    c. 交换活动和非活动的闪存存储体
    d. 堆栈指针重新初始化
    e. 重新启用中断
    f. 执行转移到新固件的 main() 例程

    在代码中，这是 LFU 应用的 lfuSwapBanks() 函数 (lfu_switchover.c) 的一段，标记为时间关键型阶段。
11. 在新固件中跳过器件初始化，后台控制循环立即开始执行

5 结果和结论

图 5-1 显示实际的 LFU 切换。顶部波形表示 LFU 切换，底部波形显示 ISR CPU 负载。在 300ns 内或大约 60 个 CPU 周期发生切换。

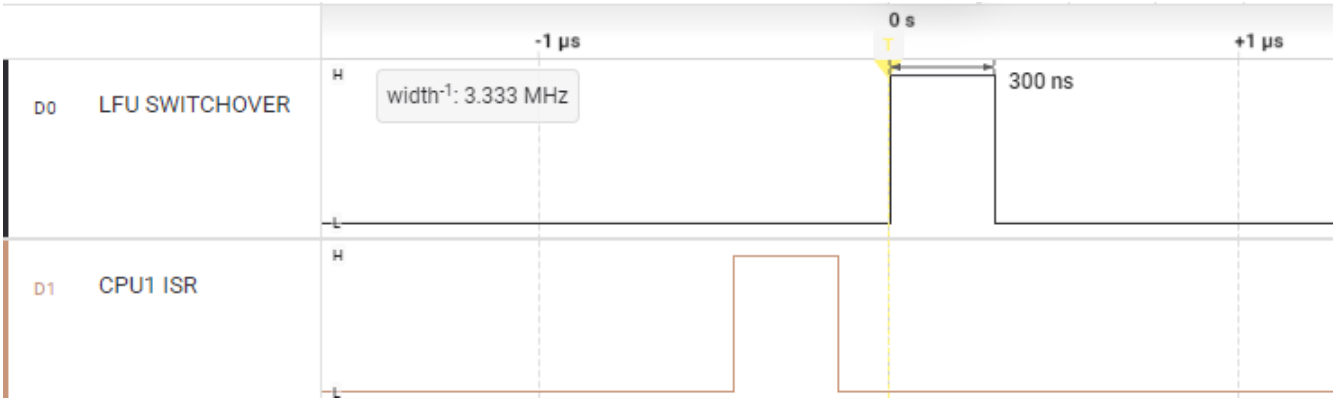

图 5-1. LFU 切换

6 实现示例

F29H85x

F29 SDK 中可用，位于： F29_SDK/examples/driverlib/single_core/flash/flash_based_sbl_with_lfu

7 摘要

本应用手册演示了适合于实时控制应用程序的 LFU 的系统实现，尤其是需要在无停机条件下运行的高可用性系统。借助现有的 LFU 构建模块（包括全新的应用层 LFU 软件流程和硬件层 LFU 支持），切换至新固件可在 60 个 CPU 时钟周期内完成。

8 参考资料

1. 德州仪器 (TI)，F29H85x 和 F29P58x 实时微控制器技术参考手册，技术参考手册。
2. 德州仪器 (TI)，F29H85x、F29P58x 和 F29P32x 实时微控制器数据表，数据表。
3. 德州仪器(TI)，F29H85x 串行闪存编程，应用手册。
4. 德州仪器 (TI)，C2000 MCU 在无器件复位时的实时固件更新，应用说明。

重要通知和免责声明

TI“按原样”提供技术和可靠性数据（包括数据表）、设计资源（包括参考设计）、应用或其他设计建议、网络工具、安全信息和其他资源，不保证没有瑕疵且不做出任何明示或暗示的担保，包括但不限于对适销性、与某特定用途的适用性或不侵犯任何第三方知识产权的暗示担保。

这些资源可供使用 TI 产品进行设计的熟练开发人员使用。您将自行承担以下全部责任：(1) 针对您的应用选择合适的 TI 产品，(2) 设计、验证并测试您的应用，(3) 确保您的应用满足相应标准以及任何其他安全、安保法规或其他要求。

这些资源如有变更，恕不另行通知。TI 授权您仅可将这些资源用于研发本资源所述的 TI 产品的相关应用。严禁以其他方式对这些资源进行复制或展示。您无权使用任何其他 TI 知识产权或任何第三方知识产权。对于因您对这些资源的使用而对 TI 及其代表造成的任何索赔、损害、成本、损失和债务，您将全额赔偿，TI 对此概不负责。

TI 提供的产品受 TI 销售条款)、TI 通用质量指南 或 ti.com 上其他适用条款或 TI 产品随附的其他适用条款的约束。TI 提供这些资源并不会扩展或以其他方式更改 TI 针对 TI 产品发布的适用的担保或担保免责声明。 除非德州仪器 (TI) 明确将某产品指定为定制产品或客户特定产品，否则其产品均为按确定价格收入目录的标准通用器件。

TI 反对并拒绝您可能提出的任何其他或不同的条款。

最后更新日期：2025 年 10 月