PHYTIUM 飞腾

飞腾系统

ACPI 描述规范

（V1.0）

2019 年 10 月

天津飞腾信息技术有限公司

www.phytium.com.cn

当前版本

文件标识	**P-U-SW-ACPI**
当前版本	**1.0**
作　　者	SDG
完成日期	2019.10.09

版本历史

版本	修订时间	修订人	修订章节	修订内容
V1.0	2019.10.09			第一个正式发布的版本

目录

表目录

1 范畴

此文档适用于基于飞腾处理器的系统平台，为支持 ACPI 的固件设计提供参考。

2 定义与缩写

2.1 定义

FT-2000/4	飞腾出品的一款面向桌面应用的高性能通用 4 核处理器。

2.2 缩写

ACPI	Advanced Configuration and Power Interface

3 参考文献

[1] Advanced Configuration and Power Interface Specification Version 6.2.

[2] ARM Functional Fixed Hardware Specification-ARM DEN 0048A

[3] Phytium Base Firmware 接口规范

[4] PCI Firmware Specification

[5] IO Remapping Table Platform Design Document

4 飞腾系统 ACPI 表

飞腾处理器系统平台采用精简硬件 ACPI 模式（Hardware-reduced ACPI mode），在该模式中，没有使用硬件实现 ACPI 固定硬件接口，而是使用对应的软件替代方案。使用精简硬件 ACPI 的系统必须实现版本 5 或以上的 FACP 描述表（即 FADT 表），并且设置 HW_REDUCED_ACPI 标志位。这种方案称为功能型固定硬件（FFH），飞腾处理器遵循《ARM Functional Fixed Hardware Specification》规范要求。

根据 Linux 文档说明，对于 ARM64 上的 ACPI，表格分为如下类型：

- 必需：DSDT、FADT、GTDT、MADT、MCFG、RSDP、SPCR、XSDT
- 推荐：BERT、EINJ、ERST、HEST、PCCT、SSDT
- 可选：BGRT、CPEP、CSRT、DBG2、DRTM、ECDT、FACS、FPDT、IORT、MCHI、MPST、MSCT、NFIT、PMTT、RASF、SBST、SLIT、SPMI、SRAT、STAO、TCPA、TPM2、UEFI、XENV
- 不支持：BOOT、DBGP、DMAR、 ETDT、 HPET、 IBFT、 IVRS、 LPIT、MSDM、OEMx、PSDT、RSDT、SLIC、WAET、 WDAT、WDRT、WPBT

飞腾平台要求实现上述必需表格内容，其它表格可由固件和 OS 根据需要实现。为支持 PCI 的 MSI 中断方式，还必须提供 IORT 表。

4.1 DSDT 表

DSDT 表是系统描述的一部分，DSDT 由一个系统描述表头和定义块（Definition Block）数据组成，这些定义块与其他定义块类似，不同在于这些定义块不能被卸载。定义块是一组 AML 对象，用于描述硬件实现细节信息。操作系统通过解析 DSDT 以及 SSDT（如果存在）中的定义块信息，获取设备信息，生成一个树结构的 ACPI 名字空间，该名字空间描述了系统中的硬件设备。OS 根据 ACPI 名字空间信息（如设备的_HID、_CID），加载相关设备驱动。

在飞腾系统中，处理器特有的 SOC 设备需要在 DSDT 表中描述。关于 SOC 设备的 ACPI 描述方法，参见第 5 章。

4.2 FADT 表

FADT 表是固定 ACPI 描述表，该表定义了对 ACPI 兼容操作系统来说，至关重要的固

定硬件 ACPI 信息，如一些 ACPI 硬件寄存器块的基址。FADT 还包含指向 DSDT 的指针。

对于 ARM64 来说，必需设置该表中的 HW_REDUCED_ACPI 标志。当该标志设置时，将忽略 ACPI 硬件寄存器接口相关的域，这些域应设置为 0。具体包括该表格偏移 46 到 108、偏移 148 到 232 的域，以及 FADT 标志位中的第 1、2、3、7、8、13、14、16 和 17 位

如果 FADT 中相关域包含 32 位和 64 位值，则优先使用 64 位值（如果 64 位值不为 0）。

4.3 GTDT 表

通用定时器描述表 GTDT 描述系统通用定时器配置信息。通用定时器（GT）是基于 ARM 处理器的系统实现的标准定时器接口。GTDT 提供了系统通用定时器的中断配置，包括 per-processor 定时器、平台（内存映射）定时器。

GT 规范定义了下列 per-processor 定时器：

- 安全级别 1（EL1）定时器
- 非安全 EL1 定时器
- 非安全级别 2 EL2 定时器
- 虚拟定时器

以及下列平台（内存映射）定时器：

- GT 块
- SBSA 通用 Watchdog

飞腾处理器遵循上述规范。具体定时器配置信息（实现的定时器类型、中断、Always-on 能力、内存地址等）参见相关型号处理器的说明文档。如何描述这些配置，参见 ACPI 规范[1]5.2.24 节。

4.4 MADT 表

MADT 表即多 APIC 描述表，用于描述系统的中断控制器信息。对于飞腾系统，采用 GIC 中断控制器，因此在该表中需要使用 GIC 中断控制器结构（类型为 0xA-0xF），分别描述 GIC CPU 接口、GIC Distributor、GIC Redistributor、GIC ITS 信息。具体描述方法参见 ACPI 规范 5.2.12 节。

4.5 MCFG 表

MCFG 表即 PCI Express 内存映射地址空间基地址描述表。对于支持 PCI/PCIe 的平台，MCFG 表是必需的。参见《PCI Firmware Specification》。

OS 通过 ACPI 的 MCFG 表获得 PCI Express 的 ECAM 的基址。MCFG 表描述了非热插拔的 PCI 段组（Segment Group）的内存映射配置空间基址，或系统启动时可用的 PCI 段组的基址范围。该表格描述的地址范围对当前启动是非重定位的、非热插拔的。

_BBN: Boot Bus Numer 引导总线号。在多主桥系统中，_BBN 方法用于为某一特定总线提供 PCI_Config 操作区访问

_PRT：用于描述 PCI 中断路由：即 PCI INTx 中断到中断控制器的路由。_PRT 对象对于所有 PCI 主桥是必须的，用于提供 PCI INTx 路由信息。

_OSC：对于 PCI Express 主桥，该接口是必须的。

还需在 DSDT 或 SSDT 表中，在/_SB 范围下声明一个设备，保留 ECAM 占据的内存空间，描述方式如下，注意设备的 HID 为“PNP0C02”。

```
// reserve ECAM memory range
  Device(RES0)
  {
      Name(_HID, EISAID("PNP0C02"))
      Name(_UID, 0)
      Name(_CRS, ResourceTemplate() {

       QWordMemory (ResourceConsumer, PosDecode, MinFixed, MaxFixed, Cacheable, ReadWrite,
            0x0000000000000000, // Granularity
            0x0000080040000000, // Range Minimum
            0x000008004FFFFFFF, // Range Maximum
            0, // Translation Offset
```

```
            0x10000000, // Length
            ,,)
    })
}
```

4.6 RSDP

根系统描述指针 RSDP 用于定位根系统描述表 RSDT（32 位）或扩展系统描述表 XSDT（64 位）。对于 ARM64，该指针结构是必须的，由系统固件提供给 OS。

在 UEFI 系统中，EFI 系统表内有一个指向 RSDP 结构的指针。OS 加载器执行时，会获得 EFI 系统表指针，然后从中获得 RSDP 结构指针，并传递给 OSPM。

OS 加载器通过检查 EFI 系统表中的 EFI 配置表，定位出 RSDP 结构的指针。EFI 配置表表项包括一个 GUID/表格指针对。UEFI 为 APCI 定义了两个 GUID，一个用于 ACPI 1.0，另一个用于 ACPI 2.0 及后续版本。

ACPI 1.0 规范的 RSDP 结构的 EFI GUID 为：

EB9D2D30-2D88-11D3-9A16-0090273FC14D

ACPI 2.0 及以后规范的 RSDP 结构的 EFI GUID 为：

8868E871-E4F1-11D3-BC22-0080C73C8881

OS 加载器将首先使用当前版本的 GUID 查找 RSDP 结构指针，如果找到，则使用对应指针；如果没找到，将使用 ACPI 1.0 版本的 GUID 查找。

4.7 SPCR

如果支持内核启动时不带 console=<device>参数，则需要提供 SPCR 表。对于 Linux 系统，该表提供 earlycon 控制台的配置信息。

4.8 XSDT

扩展系统描述表 XSDT 功能和根系统描述表 RSDT 相同，但描述头中使用的物理地址为 64 位。RSDP 结构可以指向 RSDT 和 XSDT。如果存在 XSDT，则使用 XSDT。ACPI 系统通过根系统描述表获取其它 ACPI 表格（如 DSDT）的物理地址。Phytium 处理器为 64 位处理器，因此使用该表，而不是 RSDT。

4.9 IORT

为支持 PCI 的 MSI/MSI-X 中断方式，需提供 IORT 表（输入输出重映射表）。参见《IO Remapping Table Platform Design Document》。IORT 表描述了基于 ARM 的系统的 IO 拓扑。具体来说，IORT 提供下列描述：

- 提供 IO 拓扑、SMMU 和 GIC ITS 的 ACPI 描述
- 标识位于 SMMU 后的组件
- 标识位于 ITS 或 ITS 组后的组件
- 描述 PCIe Root Complex 与 MCFG 表和 ACPI 名字空间的联系
- 描述 ACPI 名字空间中设备的 IO 关系

使用 IORT 结构描述 GIC ITS 与 PCIe 控制器之间的关联关系。

5 飞腾系统设备 ACPI 描述

5.1 概述

飞腾系列处理器采用 ARMv8 指令集。对于支持 ACPI 的固件设计者，需要关注的飞腾处理器模块有：

1) CPU 核模块：处理器中 CPU 核的数目、层次结构、电源状态相关描述。
2) PCIE 模块：PCIE 主桥数目、配置等描述。
3) 其它 SOC 外设。主要分为：
   - 快速 IO 模块：主要包括 SD 卡控制器模块 SDC、HDAudio 控制器
   - 慢速 IO 模块：主要包括 UART 串口、LPC、GPIO、I2C、QSPI、通用 SPI、RTC 等设备。
   - GMAC 模块：以太网控制器。

具体信息参见相应型号处理器的参考文档。

5.2 电源域划分

根据处理器型号不同，芯片对电源域的划分有所区别。为了实现更细粒度的功耗控制，飞腾处理器芯片内部一般实现了多个细粒度的电源域划分，各个电源域可分别关断。对电源域进行关断，一般需要先配置对应寄存器，然后执行 WFI 指令。对电源域的具体操作，可由固件的 PSCI 实现，也可由 ACPI 的控制方法实现。

5.3 处理器低功耗状态

在 ACPI 中，可使用处理器和处理器容器设备描述处理器层次，将系统中所有处理器以分层的方式描述成一个树结构。处理器容器描述一组关联的处理器；当处理器以某种方式关联，如共享 cache 或被一共同低功耗模式影响，则称该处理器属于某一容器。

在处理器层次中，每个节点拥有与该节点特定的低功耗状态。ACPI 称该电源状态为局部电源状态（Local Power State）。例如，处于最底层的 CPU 节点，可能包括的局部电源状态有：clock gate、retention 和 power down。该局部电源状态使用_LPI 对象描述。

当运行在某一处理器上的 OS 检测到该处理器没有更多工作调度时，需要选择一个低功

耗空闲状态。该状态可能不仅仅影响该处理器自身。例如，一个进入空闲状态的处理器可能是系统或处理器容器中的最后一个活动处理器，因此系统可以选择一种影响多个处理器的电源状态。为选择这样一个状态，OS 需要为处理器层次中每个受影响的级别选择一个局部电源状态。

然而，大多数硬件体系结构仅支持从 OS 到平台的单个电源状态请求；即，不能为每个层次节点发起单独的电源状态请求。因此，OS 必须组合每个层次的局部状态形成单个的组合电源状态（Composite Power State）。平台根据组合电源状态请求进行操作。

空闲状态协调：当某一处理器请求进入某一低功耗状态时，可能会影响多个处理器或处理器电源层次的电源状态，这要求协调处理器的空闲状态请求。APCI 支持两种协调机制：平台协调、OS 发起。

飞腾系统采用平台协调方式，在这种方式中，由硬件平台负责跨处理器的空闲状态协调。OSPM 针对所有处理器层次发出一个请求，该请求蕴含着每个处理器为其自身、父节点、父节点的父节点等的局部功耗状态的投票选择，平台固件综合所有处理器的选择，最终选择一个满足约束的组合目标电源状态，并将平台置于该状态。

关于处理器的低功耗状态描述和电源管理，飞腾系统遵循《ARM Functional Fixed Hardware Specification》。ACPI 表中对处理器_LPI 的描述，需遵循该规范；由平台固件实现底层硬件的具体操作，通过 psci 接口供 OS 等上层软件调用。

5.4 PCIE 控制器

对于 UEFI 等固件实现来说，PCIE 桥控制器信息要通过飞腾 Base Firmware 接口规范提供的 PCI_HOST_BRIDGE 接口动态获取。参见《飞腾 Base Firmware 接口规范》。PCIE 桥控制器配置描述方法参考《PCI Firmware Specification》。

5.5 处理器 SOC 设备

需遵循 APCI 6.2 规范中设备相关描述标准。

飞腾系统的 SOC 设备需要描述的内容有：

- **硬件标志 HID**：对于标准设备，使用标准的 ACPI HID；对于飞腾自设计的设备，使用飞腾定义的 HID，需要驱动支持。
- **内存资源**：设备在内存映射地址空间中占据的资源范围，具体信息参考相关型号处

理器的说明文档。

- **中断资源**：系统设计时为设备分配的硬件中断号，具体信息参考相关型号处理器的说明文档。
- **设备电源状态**：设备支持的电源状态。
- **其它**：其它与设备相关的操作和属性。设备操作使用相关 ACPI 控制方法描述，设备专有属性使用_DSD 方法描述，具体信息根据设备类型确定。

附录 A 列出了 FT-2000/4 处理器 SOC 设备的资源需求。

附录 A FT-2000/4 处理器 SOC 设备 ACPI 描述示例

为方便固件实现对 FT-2000/4 系统 SOC 设备的 APCI 描述，表附 A-1 列出了 FT-2000/4 系统 SOC 设备的资源需求。

其中，第一列为设备名；第二列为设备在系统内存空间占用的内存资源；第三列表示设备分配的中断号、中断触发方式（电平：表示电平方式；高：表示高电平有效）；第四列表示为设备分配的 ACPI HID 名，用于操作系统驱动加载。

表 附 A-1 **FT-2000/4** 系统 SOC 设备资源表

设备	内存资源	中断资源	HID
uart0	0x000_28000000~0x000_28000FFF	38/电平/高	ARMH0011
uart1	0x000_28001000~0x000_28001FFF	39/电平/高	ARMH0011
uart2	0x000_28002000~0x000_28002FFF	40/电平/高	ARMH0011
uart3	0x000_28003000~0x000_28003FFF	41/电平/高	ARMH0011
GPIO0	0x000_28004000~0x000_28004FFF	42/电平/高	FTGP0001
GPIO1	0x000_28005000~0x000_28005FFF	43/电平/高	FTGP0001
RTC	0x000_2800D000~0x000_2800DFFF	36/电平/高	FTRT0001
I2C0	0x000_28006000~0x000_28006FFF	44/电平/高	FTI20001
I2C1	0x000_28007000~0x000_28007FFF	45/电平/高	FTI20001
I2C2	0x000_28008000~0x000_28008FFF	46/电平/高	FTI20001
I2C3	0x000_28009000~0x000_28009FFF	47/电平/高	FTI20001
WDT0	0x000_2800A000~0x000_2800BFFF	48/电平/高	GTDT 表描述

WDT1	0x000_28016000~0x000_28017FFF	49/电平/高	GTDT 表描述
GMAC0	0x000_2820C000~0x000_2820FFFF	81/电平/高	FTGM0001
GMAC1	0x000_28210000~0x000_28213FFF	82/电平/高	FTGM0001
SDC	0x000_28207C00~0x000_28207C00	52/电平/高 53/电平/高 54/电平/高	FTSD0001
HDAudio	0x000_28206000~0x000_28206FFF	55/电平/高	FTHD0001
LPC	0x000_20000000~0x000_27FFFFFF	37/电平/高	LPC0001

A.1 UART

该模块设计符合 ARM 协议规范，pl011。

对应的 Linux 驱动为：amba-pl011.c，该驱动支持 ACPI，无需修改。

具体资源需求参见表 FT-2000/4 系统 SOC 设备资源表。

UART ACPI 描述示例：

```
Device (UAR0)
{
    Name (_HID, "ARMH0011")   // 用于加载 pl011 驱动
    Name (_UID, Zero)   // _UID: Unique ID
    Name (_CRS, ResourceTemplate ()// _CRS: Current Resource Settings
    {
                Memory32Fixed (ReadWrite,
            0x28001000,             // Address Base
```

```
                0x00001000,             // Address Length
                 )
                 Interrupt (ResourceConsumer, Level, ActiveHigh, Exclusive, ,, )
                 {
                39, //中断号
                 }
    })
}
```

A.2 RTC

为该设备自定义的 APCI HID 为 FTRT0001。

具体资源需求参见表附 A-1 FT-2000/4 系统 SOC 设备资源表。

ACPI 描述示例：

```
    Device (RTC0) {
      Name (_HID, "FTRT0001")
      Name (_UID, Zero)
      Name (_CRS, ResourceTemplate ()// _CRS: Current Resource Settings
    {
            Memory32Fixed (ReadWrite,
                0x2800d000,             // Address Base
                0x00001000,             // Address Length
            )
            Interrupt (ResourceConsumer, Level, ActiveHigh, Exclusive, ,, )
            {
```

```
                0x2e, //中断号
            }
        })
    }
```

A.3 GPIO

具体资源需求参见表附 A-1 FT-2000/4 系统 SOC 设备资源表。

ACPI 描述示例：

```
    Device(GPI0) {
        Name(_HID, "FTGP0001")
        Name(_ADR, 0)
        Name(_UID, 0)
        Name (_CRS, ResourceTemplate ()  {
          Memory32Fixed (ReadWrite, 0x28004000, 0x1000)
        })

        Device(GP00) {
          Name(_HID, "FTGP0001")
          Name(_ADR, 0)
          Name (_DSD, Package () {
            ToUUID("daffd814-6eba-4d8c-8a91-bc9bbf4aa301"),
            Package () {
              Package () {"reg",0},
              Package () {"snps,nr-gpios",8},
          }
```

```
        })
      }
      Device(GP01) {
        Name(_HID, "FTGP0001")
        Name(_ADR, 1)
        Name (_DSD, Package () {
          ToUUID("daffd814-6eba-4d8c-8a91-bc9bbf4aa301"),
          Package () {
            Package () {"reg",0},
            Package () {"snps,nr-gpios",8},
        }
        })
      }
    }
```

A.4 I2C

具体资源需求参见表附 A-1 FT-2000/4 系统 SOC 设备资源表。

ACPI 描述示例：

```
Device (I2C0) {
    Name (_HID, "FTI20001")
    Name (_UID, Zero)
    Name (_CRS, ResourceTemplate () {
      Memory32Fixed (ReadWrite, 0x28006000, 0x1000)
      Interrupt (ResourceConsumer, Level, ActiveHigh, Exclusive) { ？？}
    })
```

```
Name (_DSD, Package () {
  ToUUID("daffd814-6eba-4d8c-8a91-bc9bbf4aa301"),
  Package () {
    Package () {"clock-frequency", 0x186a0},
    Package () {"clocks", 2},
  }
})
```

A.5 Watchdog

FT-2000/4 的 watchdog 遵循 SBSA 标准。

对应的 Linux 驱动为：

- drivers/acpi/arm64/gtdt.c
- drivers/watchdog/sbsa_gwdt.c

gtdt.c 驱动从 GTDT 表获取 watchdog 信息，初始化 watchdog platform device 信息。sbsa_gwdt.c 驱动为设备分配资源、初始化并对设备进行操作。具体实现参考源代码。

Watchdog 需要在 GTDT 表中进行描述。需要提供的参数有：

- Refresh Frame 物理地址
- Watchdog ControlFrame 物理地址
- Watchdog Timer GSIV：SBSA 通用 watchdog timer 使用的全局中断号
- Watchdog Timer Flags: 标志位，意义如下表所示。

表 附 A-2 Watchdog Timer Flags

位 0	Timer 中断模式	1：边沿触发 0：电平触发
位 1	Timer 中断 polarity	1：低有效 0：高有效
位 2	安全 Timer	1：timer 是安全的

		0：timer 是非安全的
其它位	保留	必须为 0

具体资源需求参见表附 A-1 FT-2000/4 系统 SOC 设备资源表。

A.6 GMAC

具体资源需求参见表附 A-1 FT-2000/4 系统 SOC 设备资源表。

ACPI 描述示例：

```
Device (ETH0) {
        Name (_HID, "FTGM0001")
        Name (_UID, 0)
        Name (_CRS, ResourceTemplate () {
          Memory32Fixed (ReadWrite, 0x2820C000, 0x4000)
          Interrupt (ResourceConsumer, Level, ActiveHigh, Exclusive) { 81 }
        })
        Name (_DSD, Package () {
          ToUUID("daffd814-6eba-4d8c-8a91-bc9bbf4aa301"),
          Package () {
            Package () {"interrupt-names", "macirq"},
            Package () {"clocks", 3},
            Package () {"clock-names", "stmmaceth"},
            Package () {"snps,pbl", 0x10},
            Package () {"snps,abl", 0x20},
            Package () {"snps,burst_len", 0x0e},
```

```
                Package () {"snps,multicast-filter-bins", 0x40},
                Package () {"snps,perfect-filter-entries", 0x41},
                Package () {"max-frame-size", 0x2328},
                Package () {"phy-mode", "rgmii"},
                Package () {"clock-frequency",250000000},
                Package () {"bus_id",0},
                Package () {"txc-skew-ps", 0x3e8},
                Package () {"rxc-skew-ps", 0x3e8},
              }
            })
        }
```

A.7 SDC

具体资源需求参见表附 A-1 FT-2000/4 系统 SOC 设备资源表。

ACPI 描述示例：

```
Device (SDC0) {
            Name (_HID, "FTSD0001")
            Name (_UID, 0)
            Name (_CRS, ResourceTemplate () {
              Memory32Fixed (ReadWrite, 0x28207C00, 0x100)
              Interrupt (ResourceConsumer, Level, ActiveHigh, Exclusive)
        { 52,53,54}
            })
}
```

A.8 HDAudio

具体资源需求参见表附 A-1 FT-2000/4 系统 SOC 设备资源表。

ACPI 描述示例：

```
Device (HDA0) {
        Name (_HID, "FTHD0001")
        Name (_UID, 0)
        Name (_CRS, ResourceTemplate () {
          Memory32Fixed (ReadWrite, 0x28206000, 0x1000)
          Interrupt (ResourceConsumer, Level, ActiveHigh, Exclusive) { 55}
        })
}
```

A.9 LPC

具体资源需求参见表附 A-1 FT-2000/4 系统 SOC 设备资源表。

ACPI 描述示例：

```
Device (LPC1)
{
    Name (_HID, "LPC0001")   // _HID: Hardware ID
    Name (_UID, Zero)   // _UID: Unique ID
    Name (_CRS, ResourceTemplate ()   // _CRS: Current Resource Settings
    {
        Memory32Fixed (ReadWrite,
            0x20000000,             // Address Base
            0x08000000,             // Address Length
        )
        Interrupt (ResourceConsumer, Level, ActiveHigh, Exclusive, ,, )
        {
```

```
                0x00000025,
            }
        })
    }
```

附录 B FT-2000/4 处理器 CPU ACPI 描述

飞腾 FT-2000/4 为 4 核处理器，分为 2 个 cluster，每个 cluster 包含 2 个处理器核。处理器采用 SCP 调频进行动态功耗管理。为说明处理器所在的调频时钟域，在 ACPI 表中，增加了相关描述内容。CPU 描述示例如下，其中的 clock-name 和 clock-domain 描述了用于调频的 scpi 时钟名和时钟域 ID，供 scpi 驱动使用。

```
Device (CLU0) { // Cluster0 state
    Name(_HID, "ACPI0010")
    Name(_UID, 1)

    Device(CPU0) { //    Cluster 0, Cpu 0
      Name(_HID, "ACPI0007")
      Name(_UID, 0)
      Name (_DSD, Package () {
            ToUUID("daffd814-6eba-4d8c-8a91-bc9bbf4aa301"),
            Package () {
                Package () {"clock-name","c0"},
                Package () {"clock-domain",0},
            }
      })
  }

    Device(CPU1) { //    Cluster 0, Cpu 1
      Name(_HID, "ACPI0007")
      Name(_UID, 1)
      Name (_DSD, Package () {
            ToUUID("daffd814-6eba-4d8c-8a91-bc9bbf4aa301"),
            Package () {
                Package () {"clock-name","c0"},
                Package () {"clock-domain",0},
              }
      })
```

```
    }

}

Device (CLU1) { // Cluster1 state
    Name(_HID, "ACPI0010")
    Name(_UID, 2)

    Device(CPU2) { //    Cluster 0, Cpu 2
        Name(_HID, "ACPI0007")
        Name(_UID, 2)
        Name (_DSD, Package () {
                ToUUID("daffd814-6eba-4d8c-8a91-bc9bbf4aa301"),
                Package () {
                    Package () {"clock-name","c1"},
                    Package () {"clock-domain",1},
                }
        })
    }
    Device(CPU3) { // Cluster 0, Cpu 3
        Name(_HID, "ACPI0007")
        Name(_UID, 3)
        Name (_DSD, Package () {
                ToUUID("daffd814-6eba-4d8c-8a91-bc9bbf4aa301"),
                Package () {
                    Package () {"clock-name","c1"},
                    Package () {"clock-domain",1},
                }
        })
    }
}
```

附录 C FT-2000/4 处理器 PCIE ACPI 描述

目前，飞腾 FT-2000/4 的 PCIE 配置为单根模式，最多可配置为 6 个 PCI 控制器。PCIE 的 ACPI 描述示例如下，其中 PCI 中断路由表中描述了 6 个 PCI 控制器的路由配置，_CRS 方法中描述了 PCI 的 IO、MEM32、MEM64 空间等。

```
Device(PCI0)
 {
    Name(_HID, EISAID("PNP0A08")) // PCI Express Root Bridge
    Name(_CID, EISAID("PNP0A03")) // Compatible PCI Root Bridge
    Name(_SEG, Zero) // PCI Segment Group number
    Name(_BBN, 0)       // PCI Base Bus Number
    Name(_CCA, 1)       // Initially mark the PCI coherent (for JunoR1)

    // Root Complex
    Device (RP0) {
        Name(_ADR, 0x00000000)       // Dev 0, Func 0
    }
  // PCI Routing Table
  Name(_PRT, Package() {
    ROOT_PRT_ENTRY(0, 0, LNKA),     // INTA
    ROOT_PRT_ENTRY(0, 1, LNKB),     // INTB
    ROOT_PRT_ENTRY(0, 2, LNKC),     // INTC
    ROOT_PRT_ENTRY(0, 3, LNKD),     // INTD

    ROOT_PRT_ENTRY(1, 0, LNKA),     // INTA
    ROOT_PRT_ENTRY(1, 1, LNKB),     // INTB
    ROOT_PRT_ENTRY(1, 2, LNKC),     // INTC
    ROOT_PRT_ENTRY(1, 3, LNKD),     // INTD
    ROOT_PRT_ENTRY(1, 3, LNKD),     // INTD

    ROOT_PRT_ENTRY(2, 0, LNKA),     // INTA
    ROOT_PRT_ENTRY(2, 1, LNKB),     // INTB
    ROOT_PRT_ENTRY(2, 2, LNKC),     // INTC
```

```
    ROOT_PRT_ENTRY(2, 3, LNKD),     // INTD

    ROOT_PRT_ENTRY(3, 0, LNKA),     // INTA
    ROOT_PRT_ENTRY(3, 1, LNKB),     // INTB
    ROOT_PRT_ENTRY(3, 2, LNKC),     // INTC
    ROOT_PRT_ENTRY(3, 3, LNKD),     // INTD

    ROOT_PRT_ENTRY(4, 0, LNKA),     // INTA
    ROOT_PRT_ENTRY(4, 1, LNKB),     // INTB
    ROOT_PRT_ENTRY(4, 2, LNKC),     // INTC
    ROOT_PRT_ENTRY(4, 3, LNKD),     // INTD

    ROOT_PRT_ENTRY(5, 0, LNKA),     // INTA
    ROOT_PRT_ENTRY(5, 1, LNKB),     // INTB
    ROOT_PRT_ENTRY(5, 2, LNKC),     // INTC
    ROOT_PRT_ENTRY(5, 3, LNKD),     // INTD
})
    // Root complex resources
Method (_CRS, 0, Serialized) {
    Name (RBUF, ResourceTemplate () {
        WordBusNumber ( // Bus numbers assigned to this root
            ResourceProducer,
            MinFixed, MaxFixed, PosDecode,
            0,    // AddressGranularity
            0,    // AddressMinimum - Minimum Bus Number
            255, // AddressMaximum - Maximum Bus Number
            0,    // AddressTranslation - Set to 0
            256   // RangeLength - Number of Busses
        )

        DWordMemory ( // 32-bit BAR Windows
            ResourceProducer, PosDecode,
```

```
    MinFixed, MaxFixed,
    Cacheable, ReadWrite,
    0x00000000,                 // Granularity
    0x58000000,                 // Min Base Address
    0x7FFFFFFF,                 // Max Base Address
    0x00000000,                 // Translate
    0x28000000                  // Length
)

QWordMemory ( // 64-bit BAR Windows
    ResourceProducer, PosDecode,
    MinFixed, MaxFixed,
    Cacheable, ReadWrite,
    0x00000000,                 // Granularity
    0x1000000000,               // Min Base Address
    0x1FFFFFFFFF,               // Max Base Address
    0x0000000000,               // Translate
    0x1000000000                // Length
    0x0000000000,               // Translate
    0x1000000000                // Length
)

 DWordIo ( // IO window
    ResourceProducer,
    MinFixed,
    MaxFixed,
    PosDecode,
    EntireRange,
    0x00000000,                 // Granularity
    0x00000000,                 // Min Base Address
    0x00efffff,                 // Max Base Address
    0x50000000,                 // Translate
```

```
            0x00f00000,                 // Length
            ,,,TypeTranslation
          )
        }) // Name(RBUF)
        Return (RBUF)
      } // Method(_CRS)
```

附录 D FT-2000/4 处理器笔记本 EC 描述

对于 FT-2000/4 处理器的飞腾笔记本，采用嵌入式控制器（Embedded Controller，EC），实现系统电源控制，以及键盘、电池等外设的管理。EC 连接到处理器的 LPC 总线上，EC 下连接了键盘、触摸板、电池、温度传感器等设备。

EC 相关的 ACPI 描述如下:

通过 EC 标准接口访问 EC 下的资源，EC 的命令和数据端口由 ec_command_reg、ec_data_reg 描述。由于 ARM64 平台没有专门的 IO 指令，对这两个端口的访问实际以常规的内存访问方式进行，访存地址为 LPC 的基址+端口号。需要对内核中相关部分进行修正。对 EC 内部空间的访问需要依据 EC 规范的要求对这两个端口进行操作。通过这两个端口访问的 EC 资源包括电池（BAT1）、温度传感器（SEN3）、风扇（FAN0），具体的信息在对应设备下的_DSD 中描述。

EC 还可能包含键盘控制器，以 i8042 标准键盘控制器为例，单独通过另外两个端口 i8042_command_reg、i8042_data_reg 进行访问。

```
Device (LPC1)
{
    Name (_HID, "LPC0001")  // _HID: Hardware ID
    Name (_UID, Zero)  // _UID: Unique ID
    Name (_CRS, ResourceTemplate ()  // _CRS: Current Resource Settings
    {
        Memory32Fixed (ReadWrite,
            0x20000000,          // Address Base
            0x08000000,          // Address Length
        )
        Interrupt (ResourceConsumer, Level, ActiveHigh, Exclusive, ,, )
        {
            0x00000025,
        }
    })
    Name(_DSD, Package()
```

```
{
        ToUUID("daffd814-6eba-4d8c-8a91-bc9bbf4aa301"),Package(0x02){
                Package(0x02){"ec_command_reg",0x66}, //EC 命令端口
                 Package(0x02){"ec_data_reg",0x62},    //EC 数据端口
        }
 })

 Device(KBC) {
            Name(_HID, "KBCI8042")
            Name(_UID, 0)
            Name(_DSD, Package()
            {
                 ToUUID("daffd814-6eba-4d8c-8a91-bc9bbf4aa301"),Package(0x02){
                      Package(0x02){"i8042_command_reg",0x64},
                      Package(0x02){"i8042_data_reg",0x60},
                      Package(0x02){"kbc_backlight_up",0xe001},    //背光亮度热键+
                      Package(0x02){"kbc_backlight_down",0xe002},//背光亮度热键-
                      Package(0x02){"kbc_touchpad_switch,0xe003},//触摸板开关热键
                      Package(0x02){"kbc_wifi_switch,0xe004},           //wifi 开关热键
                      Package(0x02){"kbc_lcd_backlight,0xe005},               //LCD 背光时能热键
                 }
            })
        }

    Device (BAT1)
    {
        Name (_HID, "FTEC0001")    // _HID: Hardware ID
        Name (_UID, Zero)    // _UID: Unique ID
        Name (_CRS, ResourceTemplate ()    // _CRS: Current Resource Settings
        {
        })
       Name(_DSD, Package()
```

```
{
  ToUUID("daffd814-6eba-4d8c-8a91-bc9bbf4aa301"),Package(0x02){
      Package(0x02){"ec_battery_q_event",0xB3}, //电池 Q-event
      Package(0x02){"ec_battery_status",0x3c}, //电池状态
      Package(0x02){"ec_battery_remain_percent",0x21},  //电池剩余百分比电量
      Package(0x02){"ec_battery_voltage_l8",0x2a},  //电池电压低 8 位
      Package(0x02){"ec_battery_voltage_h8",0x2b},  //电池电压高 8 位
      Package(0x02){"ec_battery_current_l8",0x2c},  //电池电流低 8 位
      Package(0x02){"ec_battery_current_h8",0x2d},  //电池电流高 8 位
      Package(0x02){"ec_battery_avg_current_l8",0x2e},  //电池平均电流低 8 位
      Package(0x02){"ec_battery_avg_current_h8",0x2f},  //电池平均电流高 8 位
      Package(0x02){"ec_battery_remain_l8",0x26},  //电池剩余电量低 8 位
      Package(0x02){"ec_battery_remain_h8",0x27},  //电池剩余电量高 8 位
      Package(0x02){"ec_battery_temp_l8",0x28},  //电池温度低 8 位
      Package(0x02){"ec_battery_temp_h8",0x29},  //电池温度高 8 位
      Package(0x02){"ec_battery_design_charge_l8",0x38},  //电池设计充满电量低 8 位
      Package(0x02){"ec_battery_design_charge_h8",0x39},  //电池设计充满电量高 8 位
      Package(0x02){"ec_battery_design_voltage_l8",0x3a},  //电池设计电压低 8 位
      Package(0x02){"ec_battery_design_voltage_h8",0x3b},  //电池设计电压高 8 位
      Package(0x02){"ec_battery_charge_l8",0x24},  //电池充满电量低 8 位
      Package(0x02){"ec_battery_charge_h8",0x25},  //电池充满电量高 8 位
      Package(0x02){"ec_power_q_event",0xB4}, //电源 Q-event
      Package(0x02){"ec_power_status",0xb0}, //电源状态
  }

Device (SEN3)
{
    Name (_HID, "FTEC0002")  // _HID: Hardware ID
    Name (_UID, Zero)  // _UID: Unique ID
    Name (_CRS, ResourceTemplate ()  // _CRS: Current Resource Settings
    {
    })
```

```
        Name(_DSD, Package()
        {
            ToUUID("daffd814-6eba-4d8c-8a91-bc9bbf4aa301"),Package(0x02){
                    Package(0x02){"ec_cpu_temp",0x98},    //EC CPU 温度
                    Package(0x02){"ec_gpu_temp",0x9A},    //EC GPU 温度
         }
      }

      Device (FAN0)
      {
          Name (_HID, "FTEC0003")    // _HID: Hardware ID
          Name (_UID, Zero)    // _UID: Unique ID
          Name (_CRS, ResourceTemplate ()    // _CRS: Current Resource Settings
          {
          })
            Name(_DSD, Package()
             {
                  ToUUID("daffd814-6eba-4d8c-8a91-bc9bbf4aa301"),Package(0x02){
                       Package(0x02){"ec_cpu_fan_speed_l8",0xf3},    //EC CPU 风扇转速低 8 位
                       Package(0x02){"ec_cpu_fan_speed_h8",0xf2},    //EC CPU 风扇转速高 8 位
                       Package(0x02){"ec_gpu_fan_speed_l8",0xf5},    //EC GPU 风扇转速低 8 位
                       Package(0x02){"ec_gpu_fan_speed_h8",0xf4},    //EC GPU 风扇转速高 8 位
             }
      }
}
```