



U-Boot简介

2251730 刘淑仪 2024/11/20

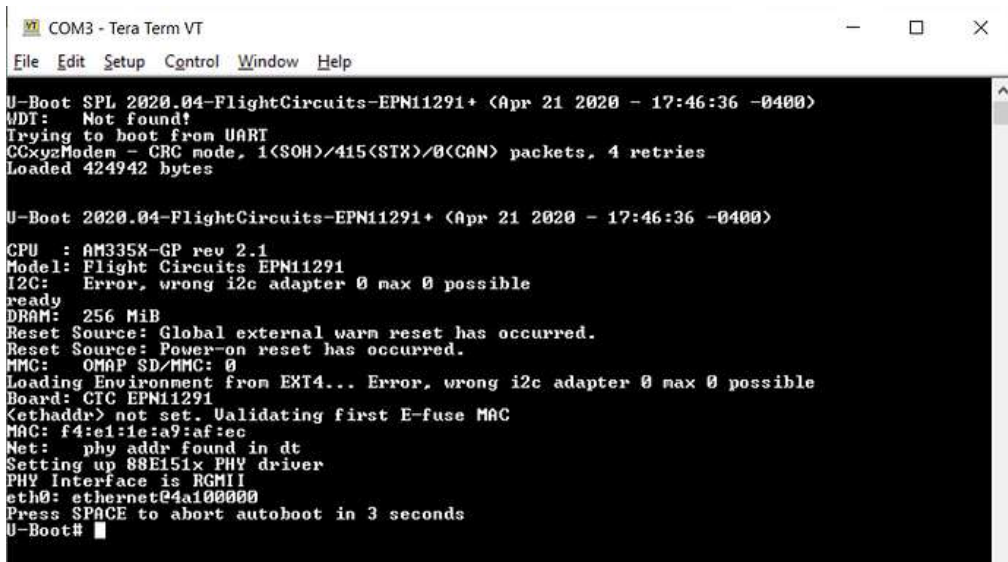
U-Boot简介

Introduction to U-Boot

U-Boot (Universal Boot Loader) 是一种广泛应用于嵌入式系统的开源Bootloader，支持多种架构，如ARM、x86、PowerPC、MIPS等。U-Boot功能强大，支持从多种存储设备（如NAND、NOR、SD卡、eMMC等）加载操作系统内核。

U-Boot的优势

- **开源和灵活性：**可以根据具体硬件平台定制。
- **跨平台支持：**支持多种处理器和硬件架构。
- **强大的生态系统：**广泛的社区支持和文档。



```
COM3 - Tera Term VT
File Edit Setup Control Window Help

U-Boot SPL 2020.04-FlightCircuits-EPN11291+ (Apr 21 2020 - 17:46:36 -0400)
WDT: Not found!
Trying to boot from UART
CCxyzModem - CRC mode, 1(SOH)/415(STX)/0(CAN) packets, 4 retries
Loaded 424942 bytes

U-Boot 2020.04-FlightCircuits-EPN11291+ (Apr 21 2020 - 17:46:36 -0400)

CPU : AM335X-GP rev 2.1
Model: Flight Circuits EPN11291
I2C: Error, wrong i2c adapter 0 max 0 possible
ready
DRAM: 256 MiB
Reset Source: Global external warm reset has occurred.
Reset Source: Power-on reset has occurred.
MMC: OMAP SD/MMC: 0
Loading Environment from EXT4... Error, wrong i2c adapter 0 max 0 possible
Board: CTC EPN11291
<ethaddr> not set. Validating first E-fuse MAC
MAC: f4:e1:1e:a9:af:ec
Net: phy addr found in dt
Setting up 88E151x PHY driver
PHY interface is RGMII
eth0: ethernet@4a100000
Press SPACE to abort autoboot in 3 seconds
U-Boot#
```

U-Boot的工作过程

1. 上电初始化阶段 (ROM Code/Primary Bootloader)

- 在嵌入式系统中，CPU上电后首先执行存储在片上ROM中的固化代码。
- 该代码完成初步的硬件初始化（如时钟、DDR、外设等），并从指定的存储介质中加载U-Boot的第一部分（通常是SPL或MLO）。

2. SPL阶段 (Secondary Program Loader)

- SPL** 是U-Boot的精简版，用于在资源有限的环境下完成基本初始化（如DDR内存、串口等），并加载完整的U-Boot镜像。
- SPL的任务是将U-Boot的核心部分从存储设备加载到DDR，并跳转执行。



U-Boot的工作过程

3. U-Boot主阶段

- **硬件初始化**：初始化更多的外设（如网络、USB、显示设备等）。
- **环境变量加载**：从存储设备（如EEPROM、Flash）中读取环境变量（包括启动配置，如启动命令、存储设备路径等）。
- **用户交互**：通过串口提供交互式命令行，允许用户修改环境变量、加载操作系统、调试硬件等。

4. 加载并启动操作系统

- U-Boot根据配置加载内核镜像（如Linux kernel）及相关资源（如设备树文件、initramfs）。
- 最后跳转到内核的入口点，启动操作系统。



U-Boot的主要功能

1. **加载内核**: 支持多种文件系统 (如 FAT、EXT4 等) 和协议 (如 TFTP、NFS) 加载内核。
2. **环境变量管理**: 支持动态修改、保存和加载环境变量。
3. **引导选项**: 支持多种启动模式 (网络启动、从不同存储设备启动等) 。
4. **硬件调试和测试**: 可用于调试硬件和验证系统功能。
5. **命令行工具**: 提供丰富的命令用于管理设备、配置网络、读写存储设备等。

典型启动过程示例

假设从eMMC加载Linux内核的典型U-Boot启动流程:

上电 -> ROM Code -> 加载SPL。

SPL初始化内存 -> 加载完整U-Boot。

U-Boot加载内核、设备树 -> 跳转到Linux内核。

这种流程保证了嵌入式系统在资源受限的环境下, 能够可靠地完成从硬件上电到操作系统加载的全过程。

典型启动过程示例

假设从eMMC加载Linux内核的典型U-Boot启动流程：

- 上电 -> ROM Code -> 加载SPL。
- SPL初始化内存 -> 加载完整U-Boot。
- U-Boot加载内核、设备树 -> 跳转到Linux内核。

这种流程保证了嵌入式系统在资源受限的环境下，能够可靠地完成从硬件上电到操作系统加载的全过程。

