Open BMC Development Series (VII) Linux Device Tree

原创 dayuli Posted on 2021-12-02 18:48:24 Read 4.2k Collection 19 Likes 5

Copyright CC 4.0 BY-SA

Category Column: BMC Device Tree Article Tags: linux operating system bmc

BMC Included in 2 columns at the same time

24 articles Subscribe to our column

1. What is a device tree?

The device tree is used to store the system's device information. When the system starts, the OS can parse the device tree information and load the corresponding device resources (including loading drivers and setting key device parameters).

The emergence of device tree solves the embarrassing situation that hardware resources are hard-coded in the code and the image needs to be recompiled every time a change is made. After adopting device tree, the corresponding device can be dynamically loaded by modifying the content of device tree without recompiling the image.

Usually the extension of the device tree we modify is dts or dtsi, which is a text file and can be opened with Notepad.

I uploaded two device tree files, dts and dtsi. Students who need them can download them by themselves.

aspeed-g5.dtsi-OS Document Resources-CSDN Download

Aspeed-2500's general configuration device tree file. For more download resources and learning materials, please visit CSDN download channel.

https://download.csdn.net/download/yjj350418592/54149828

aspeed-ast2500-evb.dts-OS Document Resources-CSDN Download

For more download resources and learning materials of aspeed2500 device tree file, please visit CSDN download channel.

https://download.csdn.net/download/yjj350418592/54149527

2. The system and grammar of the device tree

2.1 Device Tree File Types

The device tree has its own separate syntax rules and a system of its own, but it is not complicated. The device tree system is mainly divided into: device tree definition dts/dtsi, device tree compilation tool dtc, and device tree binary file dtb.

2.1.1 dts

The corresponding information of the hardware will be written in a file with the suffix .dts. Each hardware can have a separate xxxx.dts file. Generally, there are a large number of dts files in the Linux source code. For the arm architecture, the corresponding dts can be found in arch/arm/boot/dts. One dts file corresponds to one ARM machie.

2.1.2 dtsi

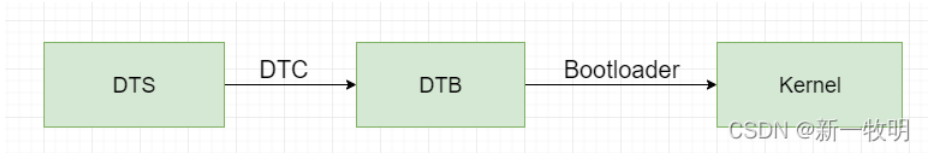
It is worth mentioning that some identical dts configurations can be abstracted into dtsi files, and then included into dts files in a similar way to C language. For the settings of the same node, the configuration in dts will overwrite the configuration in dtsi.

2.1.3 dtc

dtc is a tool for compiling dts. You can install dtc on Ubuntu by running the command apt-get install device-tree-compiler. However, the dtc tool is already included in the kernel source code scripts/dtc path.

2.1.4 dtb

dtb (Device Tree Blob), after dts is compiled by dtc, dtb file will be obtained, and dtb is loaded into the kernel through the Bootloader. Therefore, the Bootloader needs to support the device tree; the Kernel also needs to add support for the device tree;

1. 

2.2 Syntax Rules of DTS/DTSI

I will not list all the syntax rules here, but only explain a few important points. For other information, please refer to the official documentation of the device tree. Document download address:

devicetree-specification-0.3.zip-Linux Document Resources-CSDN Download

The official device tree document - devicetree-specification-v0.3, details the syntax of the device tree and for more download resources and learning materials, please visit the CSDN download channel.

https://download.csdn.net/download/yjj350418592/54148404

Syntax description:

1. dtsi is included in dts through include. If there are identical items in the file, the configuration in dts will overwrite the same configuration in dtsi.

   #include "aspeed-g5.dtsi"

   In addition, dtsi can also include .h C language header files.

2. There must be a root node and only one root node. /
3. The general size of the device tree is uint32, which is 4 bytes. Usually we see the data size <2> format, which means that the field is represented by 2 units of uint32, which is 8 bytes.
4. All statements need to end with a semicolon (;).
5. Strings are mainly of the following types:

   1. Device node name
   2. Device Type
   3. Attribute Node Name Attribute Value
   4. Tag Name

6. There are usually two types of attribute value representations:

   Angle brackets < > can have multiple values, separated by spaces

   English double quotes "", can also have multiple values, separated by commas

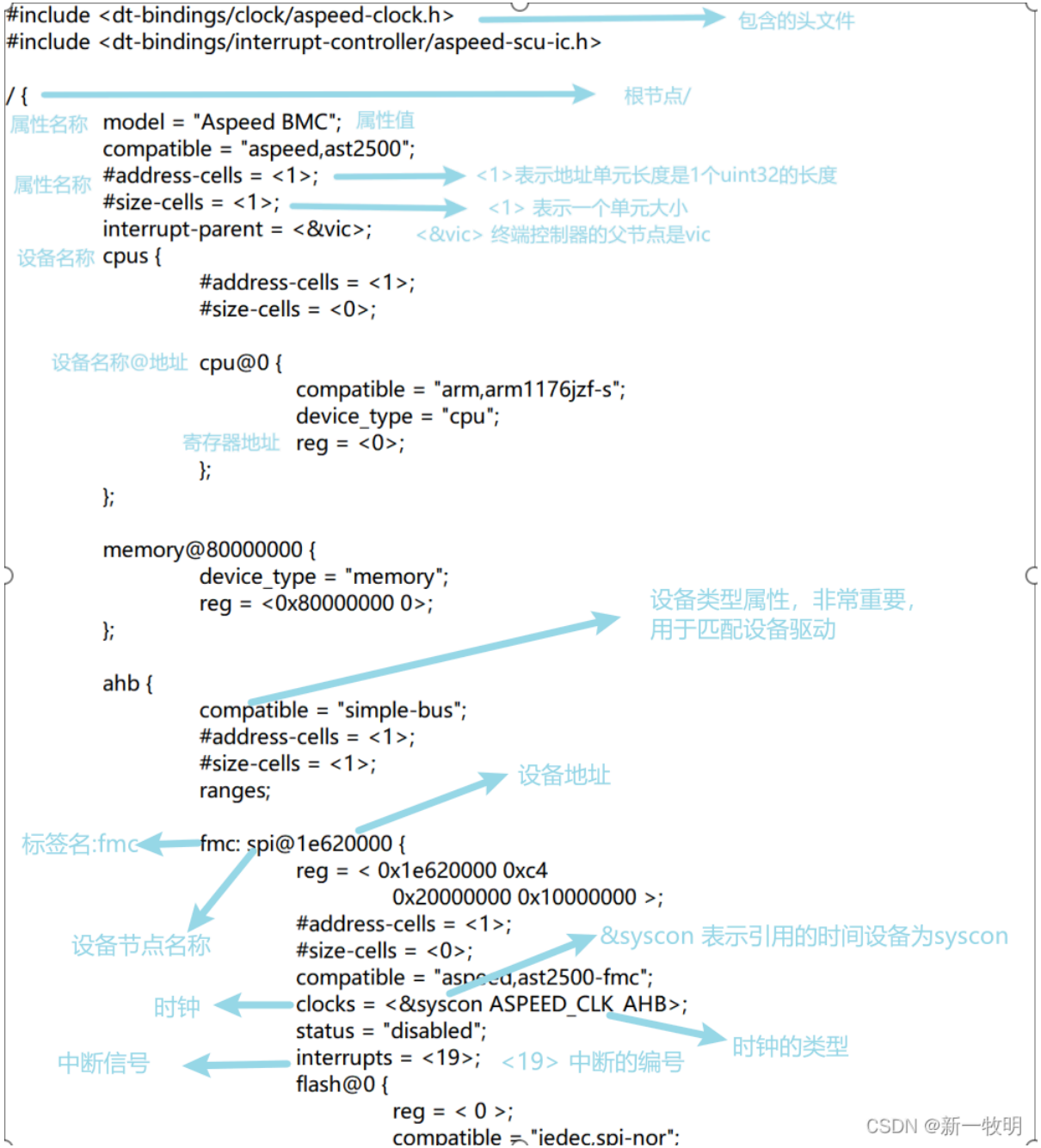
7. Several important attribute descriptions

#address-cells = <1>: word length of base address, chip select number and other absolute starting addresses, unit uint32

#size-cells = <1>: length in words, in uint32

The compatible property value is a list of strings that specifies the name of the system and contains a string of the form "<manufacturer>,<model>". It is important to specify an exact device and include the manufacturer's name to avoid namespace conflicts.

Here is a picture as an example for your reference.


#include <dt-bindings/clock/aspeed-clock.h>
#include <dt-bindings/interrupt-controller/aspeed-scu-ic.h>
包含的头文件
/ {
根节点/
属性名称
model = "Aspeed BMC";
属性值
compatible = "aspeed,ast2500";
属性名称
#address-cells = <1>;
<1>表示地址单元长度是1个uint32的长度
#size-cells = <1>;
<1> 表示一个单元大小
interrupt-parent = <&vic>;
<&vic> 终端控制器的父节点是vic
设备名称
cpus {
#address-cells = <1>;
#size-cells = <0>;
设备名称@地址
cpu@0 {
compatible = "arm,arm1176jzf-s";
device_type = "cpu";
寄存器地址
reg = <0>;
};
};
memory@80000000 {
device_type = "memory";
reg = <0x80000000 0>;
};
设备类型属性，非常重要，
用于匹配设备驱动
ahb {
compatible = "simple-bus";
#address-cells = <1>;
#size-cells = <1>;
ranges;
设备地址
标签名:fmc
fmc: spi@1e620000 {
reg = < 0x1e620000 0xc4
0x20000000 0x10000000 >;
#address-cells = <1>;
设备节点名称
#size-cells = <0>;
&syscon 表示引用的时间设备为syscon
compatible = "aspeed,ast2500-fmc";
时钟
clocks = <&syscon ASPEED_CLK_AHB>;
status = "disabled";
时钟的类型
中断信号
interrupts = <19>;
<19> 中断的编号
flash@0 {
reg = < 0 >;
compatible = "jedec,spi-nor";

CSDN @新一牧明

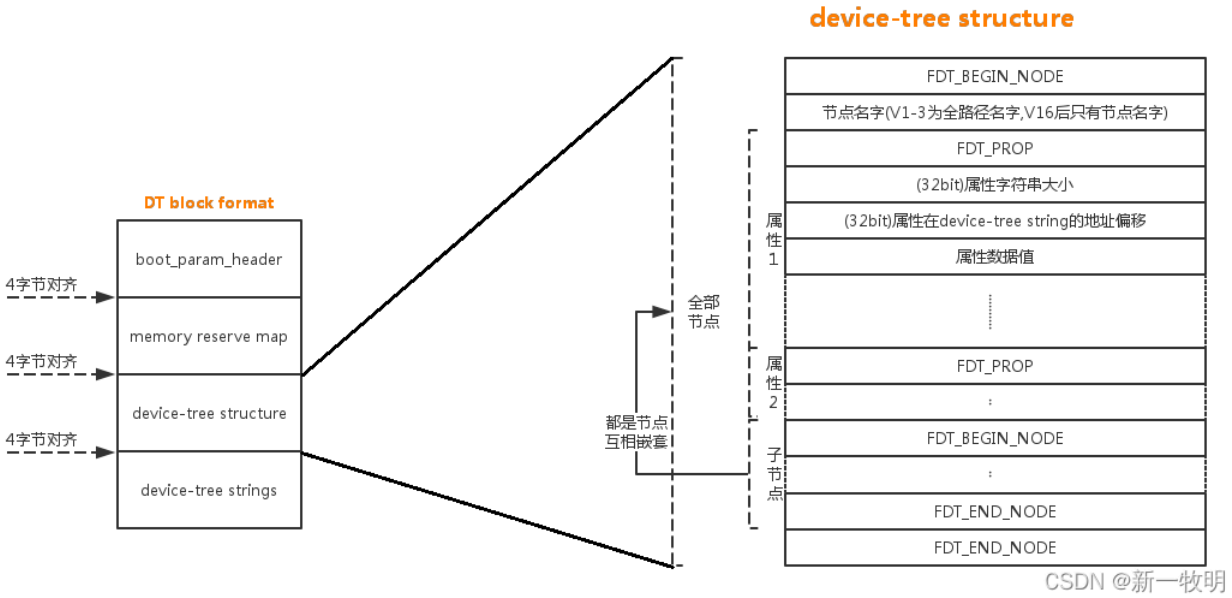
```
vic: interrupt-controller@1e6c0080 {
        compatible = "aspeed,ast2400-vic";
        interrupt-controller;
        #interrupt-cells = <1>;
        valid-sources = <0xfefff7ff 0x0807ffff>;
        reg = <0x1e6c0080 0x80>;
};
```

标签名vic　　中断控制器的地址　　中断控制器　　中断控制器的单元长度为1个uint32　　寄存器

CSDN @新一牧明

2.3 dtb file format

2.3.1 The file format of Dtb is as follows:



CSDN @新一牧明

Please refer to the following image file, which details the format of dtb and the corresponding files.

DTB format detailed explanation.jpg-OS document resources-CSDN download

For more download resources and learning materials, please visit the CSDN download channel.

https://download.csdn.net/download/yjj350418592/54166100

2.3.2 File header boot_param_header

The structure is defined as follows:

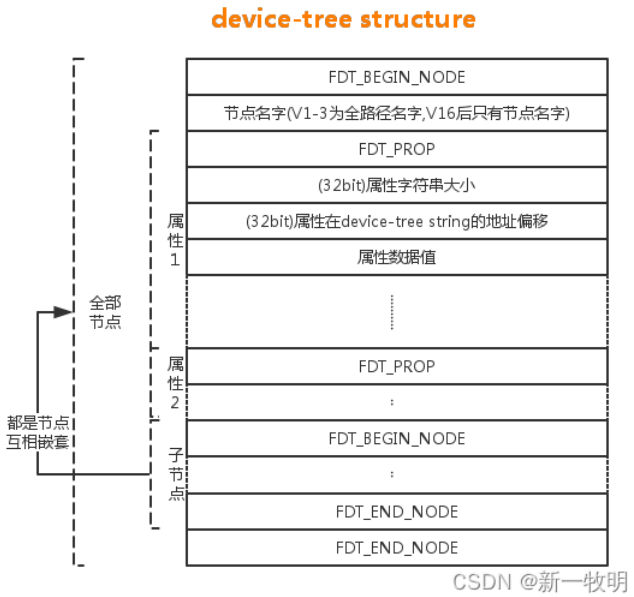
```
struct boot_param_header {
    u32 magic;----------Used to mark the dtb file header, equal to OF_DT_HEADER="0xd00dfeed"
    u32 totalsize;----------dtb file size
    u32 off_dt_struct;--------DT structure offset
    u32 off_dt_strings;-------DT strings offset
    u32 off_mem_rsvmap;-------memory reserve map offset
    u32 version;--------------version number
    u32 last_comp_version;----The earliest compatible version number
    /* version 2 fields below */
    u32 boot_cpuid_phys;------physical CPU id
    /* version 3 fields below */
    u32 size_dt_strings;------size of the strings block
    /* version 17 fields below */
    u32 size_dt_struct;-----------size of the DT structure block
};
```

2.3.3 Memory reserve map

This section stores a list of reserved memory mappings, each of which consists of a pair of 64-bit physical address and size.

2.3.4 device-tree structure&strings

Since some attributes (such as compatible) exist under most nodes, in order to reduce the size of the dtb file, it is necessary to specify only one storage location for these attribute strings. In this way, the attribute of each node can be obtained by finding the location of the attribute string according to the location. Therefore, dtb lists device-tree strings separately for storage. The following figure shows the format of the device-tree structure, with nodes nested in nodes.



The above macro definition is as follows

#define FDT_MAGIC 0xd00dfeed /* 4: version, 4: total size */

#define FDT_TAGSIZE sizeof(uint32_t)

#define FDT_BEGIN_NODE 0x1 /* Start node: full name */

#define FDT_END_NODE 0x2 /* End node */

#define FDT_PROP 0x3 /* Property: name off, size, content */

#define FDT_NOP 0x4 /* nop */

#define FDT_END 0x9

#define FDT_V1_SIZE (7*sizeof(uint32_t))

#define FDT_V2_SIZE (FDT_V1_SIZE + sizeof(uint32_t))

#define FDT_V3_SIZE (FDT_V2_SIZE + sizeof(uint32_t))

#define FDT_V16_SIZE FDT_V3_SIZE

#define FDT_V17_SIZE (FDT_V16_SIZE + sizeof(uint32_t))

3. Compilation and decompilation of device tree

3.1 Summary of dtc usage

DTC Compilation:

The dtc compiler can generate dts files into dtb files, and can also generate dtb files into dts files.

grammar:

dtc [-I input-format] [-O output-format] [-o output-filename] [-V output_version] input_filename

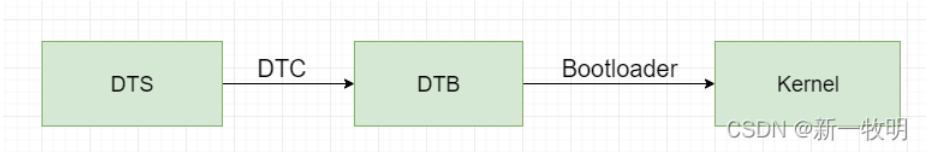
Parameter Description

input-format:
- "dtb": "blob" format
- "dts": "source" format.
- "fs" format.

output-format:
- "dtb": "blob" format
- "dts": "source" format
- "asm": assembly language file

output_version:
defines the version of "blob", which is indicated in the field of the dtb file. It supports 1 2 3 and 16. The default is 3. There are many feature changes in version 16.

3.2 Compile dts file to dtb file



As can be seen from the above figure, the file type used by the OS boot file is dtb, not dts, so we need to compile the dts file into dtb.

Use the compilation command dtc:

./dtc -I dts -O dtb -o test.dtb test.dts

3.3 Decompile dtb file to dts file

Use the compilation command dtc:

./dtc -I dts -O dtb -o test.dtb test.dts

The dts file obtained by decompiling with dtc will no longer have the content of include dtsi, and will be completely replaced with the corresponding device tree content in the new dts. Sometimes this decompilation is often used to get a complete file for device tree viewing and searching.

4. Modify the dtb device tree command

In addition to modifying the dts and then compiling with dtc to generate dtb, you can also use tool commands to directly modify the device tree.

The command to modify dtb is fdt.

4.1 uboot supports fdt

The modification is in the boot program, so uboot needs to support fdt. To support fdt in uboot, you only need to add a macro definition.

#define CONFIG_OF_LIBFDT /* Device Tree support */

Recompile uboot to support fdt.

4.2 Usage of fdt

U-Boot>fdt

fdt - flattened device tree utility commands

Usage:

fdt addr <addr> [<length>] - Set the fdt location to <addr> Set the fdt address

fdt move <fdt> <newaddr> <length> - Copy the fdt to <addr> and make it active

fdt resize - Resize fdt to size + padding to 4k addr

fdt print <path> [<prop>] - Recursive print starting at <path> Print the node of the specified path in the device tree

fdt list <path> [<prop>] - Print one level starting at <path> Print the first level node of the specified path in the device tree

fdt set <path> <prop> [<val>] - Set <property> [to <val>] Sets a property value

fdt mknode <path> <node> - Create a new node after <path> Create a node on the path

fdt rm <path> [<prop>] - Delete the node or <property>

fdt header - Display header info print fdt header information

fdt bootcpu <id> - Set boot cpuid Set boot cpuid

fdt memory <addr> <size> - Add/Update memory node Add or update memory node

fdt rsvmem print - Show current mem reserves Displays the current reserved memory

fdt rsvmem add <addr> <size> - Add a mem reserve

fdt rsvmem delete <index> - Delete a mem reserves

fdt chosen [<start> <end>] - Add/update the /chosen branch in the tree<start>/<end> - initrd start/end addr

NOTE: Dereference aliases by omiting the leading '/', eg fdt print ethernet0.

5. Recommended good reference documents

1. Refer to Wei Dongshan's explanation of the device tree.

2. Refer to the official documentation of the device tree: Specifications - DeviceTree

3. Linux device tree documentation: \linux-aspeed\Documentation\devicetree\

Finally: Liking is a virtue, following is fate, collecting is affirmation, you can reward me as you like, your encouragement is part of the goodness in my world, I love you!

about Us Careers Business Cooperation Seeking coverage 400-660-0108 kefu@csdn.net Online Customer Service Working hours 8:30-22:00
Public Security Registration Number 11010502030143 Beijing ICP No. 19004658 Beijing Internet Publishing House [2020] No. 1039-165
Commercial website registration information Beijing Internet Illegal and Harmful Information Reporting Center Parental Control
Online 110 Alarm Service China Internet Reporting Center Chrome Store Download Account Management Specifications
Copyright and Disclaimer Copyright Complaints Publication License Business license
©1999-2025 Beijing Innovation Lezhi Network Technology Co., Ltd.