Design of a Linux Audio Driver Based on the SEP6200 *

YANG Liyuan *, *WANG Peng*, *WU Xiaofei*, *LING Ming*
(*National Engineering Research Center for ASIC*, *Southeast University*, *Nanjing* 210096, *China*)

Abstract: Currently, embedded devices has penetrated every aspect of people's lives, but embedded devices based on domestic CPU core SOCs are still in their infancy. SEP6200 processor designed by Southeast University National ASIC Engineering Research Center is a self-developed high-performance SOC chip using Unicore core architecture of Peking University. This new Linux audio decoder driver has been proposed and designed, which is based on the hardware of SEP6200, CS3700 and software of ALSA, ASOC architecture. We port the MPlayer to the platform, in order to achieve sound and make recording test. Experiments show that the driver system runs on the platform of SEP6200 steadily and achieve the desired results.

Key words: embedded; Unicore; Linux; SEP6200; ALSA; ASOC; audio driver
EEACC: 8510; 1210 **doi**: 10.3969/j.issn.1005-9490.2014.02.021

基于 SEP6200 的 LINUX 音频驱动设计 *

杨鲤源*, 王 鹏, 吴晓飞, 凌 明
(东南大学国家专用集成电路工程研究中心, 南京 210096)

摘 要: 当前, 嵌入式设备已经深入人们生活的各个方面, 但基于国产内核的 SOC 嵌入式设备尚处于起步阶段。SEP6200 是由东南大学国家专用集成电路工程研究中心采用北大 UNICORE 内核架构自主设计研发的高性能 SOC 芯片, 我们提出并设计了以 SEP6200 及 CS3700 芯片为硬件基础, 详细介绍了 ALSA 及 ASOC 软件架构, 并以此为基础设计了 LINUX 系统音频驱动。通过在平台上移植 MPLAYER 播放器, 以实现放音及录音测试, 实验表明该驱动系统在 SEP6200 平台上运行稳定, 达到了预期效果。

关键词: 嵌入式; Unicore; Linux; SEP6200; ALSA; ASOC; 音频驱动
中图分类号: TN713 **文献标识码**: A **文章编号**: 1005-9490(2014)02-0266-04

Now, various electronic products based on embedded technologies, including MP3, MP4, Smart Phones, Tablet PCs etc, have been widely used in our daily life. Domestic CPU core based on electronic products is still in their infancy, so the autonomous SOC (System On Chip) SEP6200 based the CPU of unicore must promote the development of China-made electronic products. Current traditional audio driver system is mostly designed and developed based on the ARM family CPU cores. Unicore CPU as a domestic self-developed processor core is different from the ARM architecture, so the audio system based on the hardware platform of SEP6200 must be re-designed and developed. The purpose of this paper is to design and implement a linux audio driver system that can run on the platform of SEP6200 and CS3700 steadily. Through porting MPlayer to the platform, both playback and recording can achieve stable results which prove that the system works properly.

1 Hardware Design

1.1 SEP6200

SEP6200[1] is independently developed by National Engineering Research Center for ASIC of Southeast University. SEP6200 uses unicore of Peking University as the kernel. It has a 32 high-performance RISC core, up to 800 MHz, supports floating point unit; has a 16-bit SRAM/NOR Flash interface, supports to boot from Nand; has 16/32-bit DDR2/DDR3 interface; maximum supports 512 MB, 800 MHz bit rate; supports MPEG

项目来源: "核、高、基"国家科技重大专项项目
收稿日期: 2013-11-09 修改日期: 2013-12-04

1/2/4/H.264/Divx/RV/AVS/VC1/VP6/7/8 and other formats; up to 1 920 × 1 080 resolution; supports color TFT LCD, supports 16,18,24 bit RGB output; integrated HDMI Transmitter, compatible with HDMI 1.3 specification. It has 1-way USB2.0 OTG controller, 2-way SDIO interface, 3-way SPI interface, 4-way UART interface, I2C interface, I2S interface, I2S interfaces, 16-way GPIO interface; 16-channel DMA, 2-channel DMA. It uses digital chips TSMC 65 nm LP process, TFBGA package.

1.2 Architecture Design

CS3700 is a low power, high-performance audio decoder chip. It is often used for portable multimedia electronic products. It can provide high quality audio, while eliminating the use of large-capacitor headphones. It not only can perform basic D/A and A/D conversion function, but also audio controlling and digital signal processing functions. It supports I2S[2] bus data format. The audio system hardware framework shown in Fig. 1.

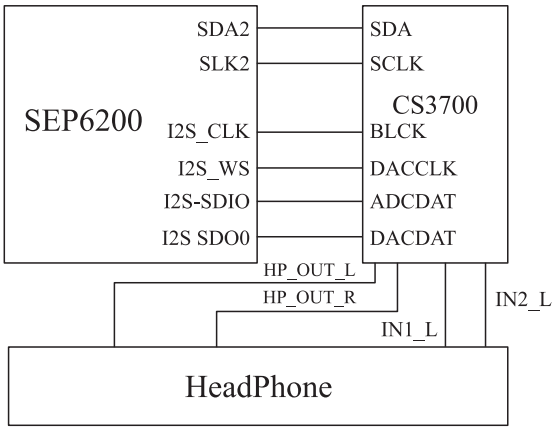


Fig. 1　Hardware architecture

SEP6200 is connected with the audio decoder chip CS3700 via I2C and I2S bus. The principle of playing audio: SEP6200 transmits control signal to the CS3700 through I2C, configuring some register value of CS3700, while the audio signal is transmitted through I2S. CS3700 completes audio signal D/A conversion. The analog signal is played back through a microphone. Recording has used the same principle. It is the inverse process of playback.

2 Audio Software System

2.1 ALSA system architecture

Before ALSA (Advanced Linux Sound Architecture)[3] audio system, OSS (Open Sound System)[4] audio system has been developed for a long time and is very mature. But it is still a commercial product that is not completely open source, and has been largely lost in the Linux mainline updates. ALSA just to make up the deficiencies of OSS. It is a completely open source audio system, compatible with GPL. It is an alternative sound card driver architecture.

Compared with OSS, ALSA in addition provides a set of kernel driver modules, and also provides a rich interface function library ALSA-lib[5] to simplify the preparation of applications specifically. Compared to the original ioctl-based programming interface provided by OSS, ALSA-lib library is more convenient and more abundant in use. In linux2.6 kernel, ALSA has become the default sound subsystem, used to replace OSS in the linux2.4.

ALSA's main features are as follows:

(1) Supporting all audio interfaces from consumer sound cards to professional sound equipment;

(2) Modular kernel driver;

(3) Supporting for symmetric multiprocessing (SMP) and multithreading;

(4) Providing a rich application development library to simplify application development;

(5) Supporting OSS API, compatible with OSS applications.

ALSA system can be roughly divided into the driver package (alsa-driver) and development package (alsa-lib). The alsa-driver[6] is divided into the core layer and the hardware-related underlying hardware layer. Usually, we only need to port the underlying hardware layer. The realization of the core layer can isolate from underlying hardware. It belongs to the ALSA standard framework, and does not require developers to implement their own transplants. Framework is shown in Fig. 2.

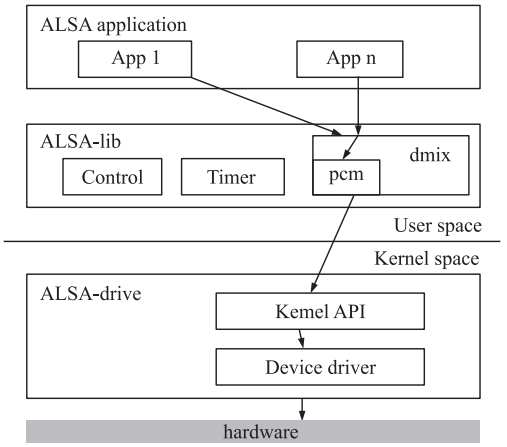


Fig. 2　ALSA architecture

2.2 ASOC Driver Design

The audio system uses ASOC (ALSA System on Chip)) framework. ASOC[7] is the development and evolution of ALSA audio system in SOC. So it still belongs to ALSA in essence. But it separates the CPU-related code framework, making ASOC framework for better portability.

ASOC is mainly composed of three parts:

(1) Codec driver This section is only concerned about Codec itself, characteristics associated with the CPU platform has nothing to with this part[8];

(2) Platform driver This part is related to the CPU platform, and has nothing to do with Codec. it mainly deals with two problems: DMA engine and PCM, I2S control integrated in SOC[9];

(3) Board-level driver (also known as machine driver) This part binds the platform-driver and Codec driver, describing the board-level hardware features specifically[10].

These three parts are ASOC core layers, they come true by kernel/sound/soc-core. c in the kernel source. In the case of sound card driver based on ASOC framework, ALSA-libs as well as a series of utility are still available.

2.3 ASOC Driver Registration

Firstly, it enters by the function of snd_soc_init( ) in the ASOC driver core layer. Then, it calls the platform driver registration function platform_driver_register( ). Secondly, it probes the driver by the function of soc_probe( ). after probed the driver, it register card driver for the core layer[11], achieving by the function of snd_soc_register_card( ). After registering the core sound card, it begins to detect the hardware in three parts:

(1) Board-level detection By the function of sep0611_board_probe( ) to complete, it is responsible for setting the Codec's gpio, enabling the codec, checking the headphone plug, setting the speaker and headphone;

(2) I2S detection By the function of sep0611_i2s_probe( ) to complete, it is responsible for setting i2s, dma initialization, application for dma interruption;

(3) Codec detection By the function of cs3700_probe( ) to complete, functions called by it include i2c_add_driver( ), cs3700_i2s_probe and cs3700_init( ).

Function cs3700_init( ) mainly completes for three things, function snd_soc_new_pcms( ) registers the stream of pcm; snd_soc_init_card( ) registers card, which includes two devices: control device and pcm device; sep0611_cs3700_init( ) completes cs3700's hardware initialization.

```
Static void sep0611_cs3700_init(struct snd_soc_codec * codec)
{
  cs3700_write_reg(0x00,0xFFFF);
};
```

An important structural body sep0611_i2s_dai needs detailed explanation. All operations to codec of I2S are with this structure as objects, including the probe, suspend, resume and other operations. It specifies the minimum and maximum number of channels, sample rate and data format when it reads and writes data.

```
struct snd_soc_dai sep0611_i2s_dai = {
  .name = "SEP0611-I2S",
  .id = 0,
  .probe = sep0611_i2s_probe,
  .suspend = sep0611_i2s_suspend,
  .resume = sep0611_i2s_resume,
  .ops = &sep0611_i2s_dai_ops,
  .playback = {
    .channels_min = 2,
    .channels_max = 2,
    .rates = SEP0611_I2S_RATES,
    .formats = SEP0611_I2S_TX_FMTS,
      },
  .capture = {
    .channels_min = 2,
    .channels_max = 2,
    .rates = SEP0611_I2S_RATES,
    .formats = SEP0611_I2S_RX_FMTS,
    },
};
```

Contents with "sep0611" is closely related to SOC chip, they are underlying driver that must be implemented by yourself.

2.4 Process of Writing Data

Applications write data by calling API function and_pcm_write( ) in ALSA-lib, then wait the function snd_pcm_wait( ) to be called by ALSA-lib. After the bottom can be written, it comes into the ALSA-driver underlying driver by calling the poll system, poll signal enters the sleep queue blocking process[12]. Interrupt signal of hardware triggers interrupt handler registered in ALSA-driver layer. Interrupt function then calls the related functions in ALSA-driver to determine whether to write data. ALSA-driver needs to do two things at this time: first, call the underlying hardware drivers to get the hardware current

data size; secondly, determine the size of the idle data area. After completed, if it satisfied the condition, awaken the sleep queue, awaken the poll signals, and finally return to the ALSA-lib. After ALSA-lib signal is received, the data will be written into buffer.

Data read is controlled by a DMA transfer[13]. ALSA-lib map memory applied by hardware to user space through the mechanism of mmap. After the application write data to a memory space, hardware is able to read the data directly. When there is data in map memory, DMA performs related actions, transmitting data to codec. Codec begins to play a sound decoding after reading data. These are the details writing and playing the audio on the layer of a user process. Reading and recording process is similar, just the API functions called by applications are different.

3 Conclusion

After experimental tests, in linux2. 6. 32 system, using SEP6200 and CS3700 as the hardware platform, this audio driver based on ASOC architecture works properly.

Two-channel differential input, and mono input have achieved recording; single-channel stereo output and two-channel output have achieve a normal playback. At present, the driver has been successfully applied in embedded system based on SEP6200 and CS3700, and stable, reaching the expected results.

Refereces:

[1] SEP6200 User Manual.

[2] Zhou Peng, Wang Cheng, , et al. Design of WM8976 Audio Driver Based on ALSA[J]. Journal of WUT(Information and Management engineering), 2011, 33(4): 517-520.

[3] Zhou Peng, Wang Cheng, et al. Design of WM8976 Audio Driver Based on ALSA [J]. Journal of WUT (Information and Management Engineering), 2011, 33(4): 517-520.

[4] Jaya Kumar, Liam Girdwood. ALSA, OLPC Audio and ASoC/DAPM. 2006. https://foss.in/2006/cfp/slides/ALSA _ and _ OLPC _ audio _82. pdf.

[5] Liam Girdwood. Alsa for System on Chip. 2006. https://www.embedded-kernel-track.org/2006/ASoC.pdf.

[6] Sreekrishnan Venkateswara. Essential Linux Device Driver[M]. 1st Edition. New Jersey: Prentice Hall, 2008.

[7] Ding Yong, Zhou Yu, Du Sidan. on Linux 2.6 Based Embedded ASoC Audio Driver and Its Implementation[J]. Computer Applications and Software, 2010, 27(4): 267-270.

[8] <linux-2.6.32>/Documentation/sound/alsa/soc/Codec.txt.

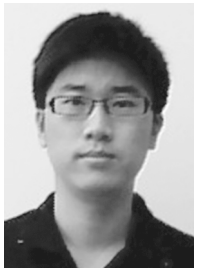
[9] <linux-2.6.32>/Documentation/sound/alsa/soc/platform.txt.

[10] <linux-2.6.32.>/Documentation/sound/alsa/soc/machine.txt.

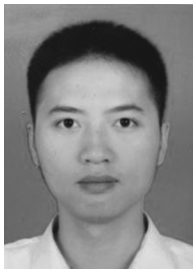
[11] Qian Hong, Hu Chen. Design and Implementation of Audio Driver Based on Audio Codec'97 for Embedded System [J]. Chinese Journal of Electron Devices, 2006, 29(02): 500-504.

[12] Corbet J. Alessandro Rubini Linux Device Drivers[M]. Sebastpol: OReilly and Associates, 2005: 137-142.

[13] Yu Yue, Yao Guoliang. Design and Implementation of Audio Driver for Embedded Linux System [J]. Chinese Journal of Electron Devices, 2008, 31(2): 709-711.



杨鲤源(1988-),男,硕士研究生,主要研究方向为嵌入式系统设计,Linux 驱动设计,SOC 异构计算;



王　鹏(1988-),男,硕士研究生,主要研究方向为 Linux 应用开发,DDR 控制器调度算法研究;

吴晓飞(1990-),男,硕士研究生,主要研究方向为嵌入式系统设计,GPU 访存特性研究

凌　明(1972-),男,副教授,主要研究方向为嵌入式实时操作系统、SOC 设计方法学、SOC 存储子系统设计。