Rev.1, 8/2023

S32G LLCE CAN Linux Driver Test

by John Li (nxa08200)

本文说明S32G LLCE CAN Linux驱动的快速测试方法。

历史	说明	作者
V1	● 创建本文	● John.Li

目录

1 参考资料

一般来讲，都是将 LLCE CAN 布署在实时且支持功能安全的 M7 核上，所以请参考第四章，Linux 驱动说明部分，将 Linux LLCE CAN 驱动从 Linux 中去掉。(DTS disabled 即可)。

不过有一些小众用法，可能会用到 Linux LLCE CAN 驱动，本文说明其测试方法。

1.1 参考资料

以 S32G3 RDB3 为例：

序号	资料	说明	如何获取
1	S32G3 BSP35 Linux Release. ● S32G3_LinuxBSP_35.0_User_Manual.pdf ● …	根据文档构建 Yocto Linux 编译环境	www.nxp.com/s32g 通过 bundle 下载 Linux 相关文档
2	S32G3 LLCE MCAL RTD ● S32G_LLCE_GATEWAY_1.0.5_D2212.exe	RTD MCAL	www.nxp.com/s32g 通过 bundle 下载安装包，安装后 S32G3 的 FW 位于：C:\NXP\S32G_LLCE_1_0_5\firmware \llce_bin\s32g3\bin\ghs\enablement dte.bin，frpe.bin，ppe_rx.bin，ppe_tx.bin
3	S32G_Kernel_BSP32_V*.pdf	应用手册	通过 NXP 社区下载： https://community.nxp.com/t5/ NXP-Designs-Knowledge-Base/ S32G-Linux-BSP-customization-doc/ta-p/1399902

1.2 版本匹配说明

参考文档《S32G3_LinuxBSP_35.0_User_Manual.pdf》说明：

12.1 Prerequisites

LLCE drivers rely on LLCE firmware, which is a separate NXP product available through https://www.nxp.com/. For more information, and to find out how to obtain LLCE Firmware, please contact NXP marketing department.

NOTE

LLCE Logging capability is only available using the advanced version of the LLCE firmware.
All Linux LLCE drivers are compatible with LLCE firmware:

- Product ID: S32G_LLCE_1.0.4
- Firmware archive: S32G_LLCE_GATEWAY_1.0.4_D2204.exe

而实际是 S32G3 的 bundle release 里的 LLCE 版本为 1.0.5，实测下来两个版本的 FW 都可以工作，建议使用 1.0.5。

2 环境搭建

2.1 使用 Yocto 编译

参考文档《S32G3_LinuxBSP_35.0_User_Manual.pdf》说明：

12.2.1 Enabling LLCE CAN from Yocto

LLCE CAN support can be enabled directly from Yocto using the following steps:
1. Download LLCE firmware (see 12.1)
2. Unpack the package to a folder of your choice and locate the following files in subdirectory firmware/llce_bin/s32g3/bin/ghs/enablement, which are required by Yocto:
• dte.bin
• frpe.bin
• ppe_tx.bin
• ppe_rx.bin
3. Enable LLCE CAN feature and specify the location where firmware binaries can be found. You can copy the .bin files
to a common firmware location (e.g. /path/to/firmware/binaries/folder) or use unpack subdirectory

above directly. This step requires to add the following lines in <builddirectory>/conf/local.conf file.

```
DISTRO_FEATURES_append = " llce-can "

NXP_FIRMWARE_LOCAL_DIR = "/path/to/firmware/binaries/folder"
```

另外，《S32G3_LinuxBSP_35.0_User_Manual.pdf》增加了以下说明：

12.2 Manual compilation

The source code of the LLCE drivers is part of the Linux kernel repository and can be compiled and deployed
manually. The drivers are conditionally included in the build by several Kconfig switches. The relevant ones are placed
under menuconfig menu:
_Networking support
_CAN bus subsystem support

_CAN Device Drivers
_Platform CAN drivers with Netlink support
_NXP LLCE CAN Support
After the compilation step, the locations of the LLCE drivers can be obtained using:

```
$ find -name "*llce*.ko"
./drivers/net/can/llce/llce_can.ko
./drivers/net/can/llce/llce_logger.ko //logger功能需要advance LLCE FW支持，本文不涉及
./drivers/mailbox/llce-mailbox.ko
./drivers/mfd/llce-core.ko
```

本文不再重复说明使用 Yocto 编译的方法，而是使用以下 Standalone 的方法。

2.2 使用 Standalone 编译

关于 S32G Linux BSP standalone 的编译与镜像烧写方法请参考文档《S32G_Kernel_BSP32_V*.pdf》，BSP35 与之类似。

1. 增加 LLCE CAN 的支持：

可以如前所说的 make s32cc_defconfig 后 make menuconfig，增加 LLCE CAN 的支持，或者直接编辑：

vi .config

```
CONFIG_CAN_LLCE=y
CONFIG_CAN_LLCE_CONTROLLER=m
CONFIG_LLCE_CORE=m
CONFIG_NXP_LLCE_MBOX=m
```

```
make -j8 //编译 Linux 内核，输出为：
OBJCOPY arch/arm64/boot/Image
…
 LD [M]  drivers/net/can/llce/llce_can.ko
 LD [M]  drivers/mailbox/llce-mailbox.ko
 LD [M]  drivers/mfd/llce-core.ko
```

2. 将 standalone 的镜像，驱动模块与 FW 拷贝到 Sdcard 中：

- Copy arch/arm64/boot/Image to sdcard boot fat partition
- Copy

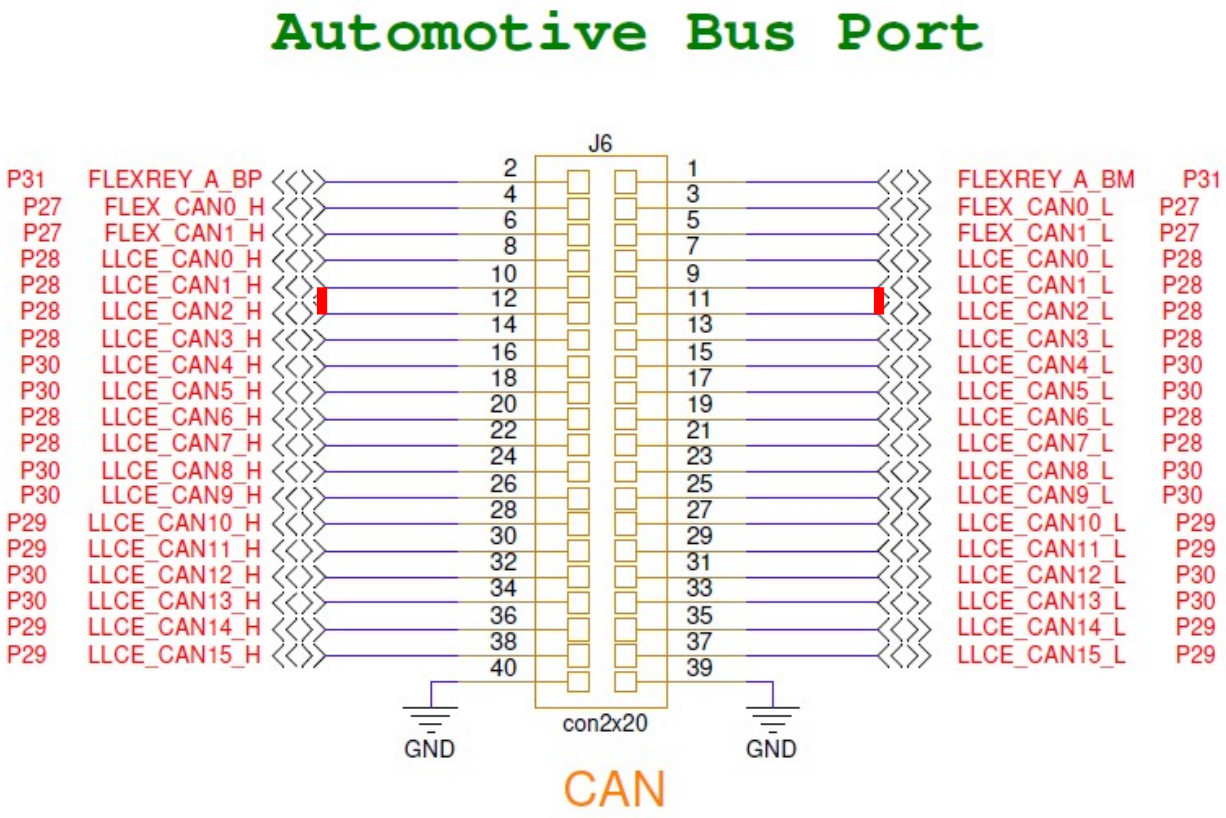
```
 LD [M]  drivers/net/can/llce/llce_can.ko
 LD [M]  drivers/mailbox/llce-mailbox.ko
 LD [M]  drivers/mfd/llce-core.ko
to sdcard rootfs /home
```

- Copy:
  C:\NXP\S32G_LLCE_1_0_5\firmware\llce_bin\s32g3\bin\ghs\enablement*.bin
  to
  sdcard rootfs /lib/firmware

3 测试

3.1 硬件连接

使用S32G3 RDB3板，设置为Sdcard启动，J6接口连接测试线束，然后将LLCECAN1与LLCECAN2连接，如下图：



注意， CAN的连接方式是H to H， L to L。

3.2 测试方法

启动 RDB3 板，在终端输入：

```
root@s32g399ardb3:/home# ls
llce-core.ko  llce-mailbox.ko  llce_can.ko  root
root@s32g399ardb3:/home# insmod llce-core.ko
root@s32g399ardb3:/home# insmod llce-mailbox.ko
root@s32g399ardb3:/home# insmod llce_can.ko
root@s32g399ardb3:/home# lsmod
Module                  Size  Used by
llce_can               24576  0
llce_mailbox           28672  15
llce_core              16384  0
root@s32g399ardb3:/home# dmesg |grep llce
[   26.745218] llce_core 43ff8000.llce: Successfully loaded LLCE firmware
[   33.048706] llce_mb 43a00000.llce_mb: LLCE firmware version: GGEE_3A
[   33.048725] llce_mb 43a00000.llce_mb: LLCE firmware: logging support disabled
root@s32g399ardb3:/home# ip link set up llcecan1 type can bitrate 500000 dbitrate 5000000 fd on
root@s32g399ardb3:/home# ip link set up llcecan2 type can bitrate 500000 dbitrate 5000000 fd on
root@s32g399ardb3:/home# ifconfig
```

..

```
llcecan1: flags=193<UP,RUNNING,NOARP>  mtu 72
        unspec 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00  txqueuelen 10  (UNSPEC)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

llcecan2: flags=193<UP,RUNNING,NOARP>  mtu 72
        unspec 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00  txqueuelen 10  (UNSPEC)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

root@s32g399ardb3:/home# candump any,0:0,#FFFFFFFF >log &
[1] 372
root@s32g399ardb3:/home# cansend llcecan2 15575555##3001122334455667788 99aabbcceedd112233
root@s32g399ardb3:/home# ls
llce-core.ko  llce-mailbox.ko  llce_can.ko  log  root
root@s32g399ardb3:/home# vi log

 llcecan1  15575555  [20]  00 11 22 33 44 55 66 77 88 99 AA BB CC EE DD 11 22 33 00 00
 llcecan2  15575555  [20]  00 11 22 33 44 55 66 77 88 99 AA BB CC EE DD 11 22 33 00 00
```

4 LLCE CAN Linux 驱动说明

4.1 DTS

Arch\arm64\boot\dts\freescale\s32gxxxa-rdb.dtsi

```
…
&llce_can0~15 {
    status = "okay";
```

```
};
```

Arch\arm64\boot\dts\freescale\s32g.dtsi

```
                llce_can0: llce_can0 {
                        compatible = "nxp,s32g-llce-can";
                        …
                        pinctrl-0 = <&llce_can0_pins>;
                        status = "disabled";
                };
    llce_can0_pins: llce_can0_pins {
            llce_can0_grp0 {
                pinmux = <S32CC_PINMUX(43, FUNC0)>;
                input-enable;
                slew-rate = <S32CC_SLEW_208MHZ>;
            };
```

及 LLCE CAN1~15 配置。

```
    llce: llce@43a00000 {
                compatible = "nxp,s32g-llce-core";
firmware-name = "dte.bin", "ppe_rx.bin",
                        "ppe_tx.bin", "frpe.bin";
…
    llce_mb: llce_mb@43a00000 {
                        compatible = "nxp,s32g-llce-mailbox";
…
    llce_can_core: llce_can_core {
                        compatible = "nxp,s32g-llce-can-core";
llce_can_logger0: llce_can_logger0 {
                        compatible = "nxp,s32g-llce-can-logger";
```

对应源代码包括：

Drives\mfd\llce-core.c

Drivers\mailbox\llce-mailbox.c

Drivers\net\can\llce\llce_can.c，llce_can_core.c，llce_logger.c。

4.2 源代码

- Drives/mfd/llce-core.c

```
llce_core_probe
|-> init_core_clock
|-> map_sram_nodes，init_sram_nodes
|-> llce_load_fw_images
of_property_count_strings(dev->of_node, "firmware-name");
for (i = 0; i < core->nfrws; i++) {
                        ret = of_property_read_string_index(dev->of_node,
                                                            "firmware-name", i,
                                                            &img_name);
ret = request_firmware(&fw->fw_entry, img_name, dev);
}
|    |-> _request_firmware
|    |    |-> fw_get_filesystem_firmware
static const char * const fw_path[] = {
            fw_path_para,
            "/lib/firmware/updates/" UTS_RELEASE,
            "/lib/firmware/updates",
            "/lib/firmware/" UTS_RELEASE,
            "/lib/firmware"
};
len = snprintf(path, PATH_MAX, "%s/%s%s",
                                    fw_path[i], fw_priv->fw_name, suffix);
/* load firmware files from the mount namespace of init */
                        rc = kernel_read_file_from_path_initns(path, fw_priv->offset,
                                                               &buffer, msize,
                                                               file_size_ptr,
                                                               READING_FIRMWARE);
            ret = start_llce_cores(dev, core);
            |-> llce_flush_fw
           |-> llce_cores_kickoff
```

```
        dev_info(dev, "Successfully loaded LLCE firmware\n");
```

- Drivers\mailbox\llce-mailbox.c

```
llce_mb_probe
|-> llce_init_chan_map
|->init_llce_mem_resources
|->init_llce_irq_resources
|->map_llce_status
|->map_llce_shmem
|->init_hif_config_chan
|   |->llce_hif_startup
|->init_core_clock
|->get_fw_version
|->print_fw_version//dev_info(dev, "LLCE firmware version: %s [%s]\n", ver_str, extra);
|->llce_platform_init
```

- Drivers\net\can\llce\llce_can.c

```
|-> llce_can_probe
|->init_llce_can_dev
|->init_completion
|->register_devlink_params
|->llce_init_can_priv
|->init_llce_chans
|->enable_llce_napi
|->llce_can_interfaces_set
|->register_candev
```