neoway

Get connected Get smart

S726

TP Debugging Guide

Issue 1.0

Notice

This document provides guide for users to use S726.

This document is intended for system engineers (SEs), development engineers, and test engineers.

THIS GUIDE PROVIDES INSTRUCTIONS FOR CUSTOMERS TO DESIGN THEIR APPLICATIONS. PLEASE FOLLOW THE RULES AND PARAMETERS IN THIS GUIDE TO DESIGN AND COMMISSION. NEOWAY WILL NOT TAKE ANY RESPONSIBILITY OF BODILY HURT OR ASSET LOSS CAUSED BY IMPROPER OPERATIONS.

THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT NOTICE DUE TO PRODUCT VERSION UPDATE OR OTHER REASONS.

EVERY EFFORT HAS BEEN MADE IN PREPARATION OF THIS DOCUMENT TO ENSURE ACCURACY OF THE CONTENTS, BUT ALL STATEMENTS, INFORMATION, AND RECOMMENDATIONS IN THIS DOCUMENT DO NOT CONSTITUTE A WARRANTY OF ANY KIND, EXPRESS OR IMPLIED.

Neoway provides customers complete technical support. If you have any question, please contact your account manager or email to the following email addresses:

Sales@neoway.com

Support@neoway.com

Website: http://www.neoway.com

Contents

About This Document

Scope

This document is applicable to the S726 series.

Audience

This document is intended for system engineers (SEs), development engineers, and test engineers.

Change History

Issue	Change	Changed By
1.0	Initial draft	Leo Shen

Conventions

Symbol	Indication
	Indicates danger or warning. This information must be followed. Otherwise, a catastrophic module or user device failure or bodily injury may occur.
	Indicates caution. This symbol alerts the user to important points about using the module. If these points are not followed, the module or user device may fail.
	Indicates instructions or tips. This symbol provides advices or suggestions that may be useful when using the module.

1 Overview

This document describes the the process of how to adapt of the touchpanel (TP) driver to S726, which is developed on the UNISOC SL8541E platform and runs Android 10 OS.

The integration of input device drivers on Android 10.0 + Kernel4.14 is different from before. The drivers are no longer compiled into the kernel in a static loading method, and must be dynamically loaded into the kernel in the form of Kernel Object (KO). Therefore, the code location and compilation rules are different from before.

- Platform: UIS8581E
- System: Android 10
- TP IC: GT1151Q
- TP interface: I2C

2 Kernel Adaptation

The TP driver file must be stored in the **bsp\modules\input\touchscreen** directory. The following uses **gt1151q** as an example to describe how to add a TP driver.

Step 1: Add the TP driver file.

Put the driver file, **goodix_gt_1151q** , provided by the TP manufacturer into **bsp\modules\input\touchscreen**.

Step 2: Add the **Android.mk** file name to the code (the highlighted content).

```
LOCAL_PATH:= $(call my-dir)
include $(CLEAR_VARS)
LOCAL_MODULE_TAGS := optional
LOCAL_MODULE := goodix_gt_1151q.ko
LOCAL MODULE CLASS := SHARED LIBRARIES
LOCAL MODULE PATH := $(TARGET OUT VENDOR)/lib/modules
LOCAL STRIP MODULE := keep symbols
LOCAL SRC FILES := $(LOCAL MODULE)
include $(BUILD PREBUILT)
ifeq ($(TARGET_BUILD_VARIANT),user)
DEBUGMODE := BUILD=no
else
DEBUGMODE := $(DEBUGMODE)
endif
#convert to absolute directory
PRODUCT_OUT_ABSOLUTE:=$(shell cd $(PRODUCT_OUT); pwd)
$(LOCAL_PATH)/goodix_gt_1151q.ko: $(TARGET_PREBUILT_KERNEL)
    $(MAKE) -C $(shell dirname $@) ARCH=$(TARGET_KERNEL_ARCH)
CROSS_COMPILE=$(KERNEL_CROSS_COMPILE) $(DEBUGMODE) KDIR=$(PRODUCT_OUT_ABSOLUTE)/obj/KERNEL
clean
    $(MAKE) -C $(shell dirname $@) ARCH=$(TARGET_KERNEL_ARCH)
CROSS_COMPILE=$(KERNEL_CROSS_COMPILE) $(DEBUGMODE) KDIR=$(PRODUCT_OUT_ABSOLUTE)/obj/KERNEL
    $(TARGET_STRIP) -d --strip-unneeded $@
```

Step 3: Add the **Kbuild** file name to the code (the highlighted content).

```
#
# Source List
#
KO_MODULE_NAME := goodix_gt_1151q
KO_MODULE_PATH := $(src)
KO_MODULE_SRC  :=
KO_MODULE_SRC += $(wildcard $(KO_MODULE_PATH)/*.c)

#
# Build Options
```

```
#
ccflags-y += -DDEBUG

#
# Final Objects
#
obj-m := $(KO_MODULE_NAME).o
# Comment it if the only object file has the same name with module
$(KO_MODULE_NAME)-y := $(patsubst $(src)/%.c,%.o,$(KO_MODULE_SRC))
```

Step 4: Modify the **Makefile** file name (the highlighted content).

```
KO_MODULE_NAME := goodix_gt_1151q
KO_MODULE_OUT := $(BSP_MODULES_OUT)/$(KO_MODULE_NAME)
KO_MODULE_KBUILD := $(CURDIR)/Kbuild
.PHONY: modules modules_install clean
modules:
    @mkdir -p $(KO_MODULE_OUT) && ln -snf $(KO_MODULE_KBUILD) $(KO_MODULE_OUT)/Kbuild
    @ln -snf $(CURDIR) $(KO_MODULE_OUT)/source
    $(MAKE) -C $(BSP_KERNEL_PATH) M=$(KO_MODULE_OUT) src=$(CURDIR) $@
modules_install:
    $(MAKE) -C $(BSP_KERNEL_PATH) M=$(KO_MODULE_OUT) $@
# Remove the out directory wholly
clean:
    @#$(MAKE) -C $(BSP_KERNEL_PATH) M=$(KO_MODULE_OUT) src=$(CURDIR) $@
    rm -rf $(KO_MODULE_OUT)
```

Step 5: Modify the TP DTS file.

In the DTS file on the device tree of the project, add the I2C device node of TP, and modify the **S726_uis8581e5h10-overlay.dts file** under the **bsp\kernel\kernel4.14\arch\arm64\boot\dts\sprd** directory as follows (the highlighted content):

```
&i2c3 {
    status = "okay";
    goodix@14 {
        compatible = "goodix,gt1x";
        reg = <0x14>;
        goodix,irq-gpio = <&ap_gpio 144 GPIO_ACTIVE_HIGH>;
        goodix,reset-gpio = <&ap_gpio 145 GPIO_ACTIVE_HIGH>;
    };
};
```

Step 6: Add the TP diver into the project.

1. Add the **.ko** file name to the **modules.cfg** file under **bsp\device\sharkl3\androidq\S726\S726**.

```
BSP_MODULES_LIST="
sample.ko
```

```
microarray_fp.ko
sprd_sensor.ko
sprd_camera.ko
sprd_cpp.ko
sprdbt_tty.ko
sprd_flash_drv.ko
flash_ic_ocp8137.ko
tcs3430.ko
stmvl53l0.ko
sprdwl_ng.ko
sprd_fm.ko
gator.ko
pvrsrvkm.ko
goodix_gt_1151q.ko
"
```

2. Add the **.ko** file name (highlighted) into the **S720.mk** file of the project under the **device\nwy\S720** directory.

```
PRODUCT_SOCKO_KO_LIST := \
    $(BSP_KERNEL_MODULES_OUT)/flash_ic_ocp8137.ko \
    $(BSP_KERNEL_MODULES_OUT)/pvrsrvkm.ko \
    $(BSP_KERNEL_MODULES_OUT)/sprdbt_tty.ko \
    $(BSP_KERNEL_MODULES_OUT)/sprd_camera.ko \
    $(BSP_KERNEL_MODULES_OUT)/sprd_cpp.ko \
    $(BSP_KERNEL_MODULES_OUT)/sprd_flash_drv.ko \
    $(BSP_KERNEL_MODULES_OUT)/sprd_fm.ko \
    $(BSP_KERNEL_MODULES_OUT)/sprd_sensor.ko \
    $(BSP_KERNEL_MODULES_OUT)/sprdwl_ng.ko \
    $(BSP_KERNEL_MODULES_OUT)/tcs3430.ko \
    $(BSP_KERNEL_MODULES_OUT)/microarray_fp.ko \
    $(BSP_KERNEL_MODULES_OUT)/goodix_gt_1151q.ko
```

Step 7: Configure data to load the TP driver upon startup.

Modify the **init.common.rc** file under the **device\nwy\S726\rootdir\root** directory for driver auto-load upon startup by adding the KO loading command.
See the highlighted part in the code below:

```
on post-fs
    # support goodix gt1151q ts
       insmod ${ro.vendor.ko.mount.point}/socko/goodix_gt_1151q.ko
```

