PENMOUNT TOUCH CONTROLLER EMBEDDED LINUX USER GUIDE

Revision D

09/Sep/'23

PenMount®
TOUCH SCREEN SOLUTIONS

Preface

Disclaimer

The information in this document is subject to change without notice. The manufacturer makes no representations or warranties regarding the contents of this manual and specifically disclaims any implied warranties of merchantability or fitness for any particular purpose. Furthermore, the manufacturer reserves the right to revise this publication or make changes in the specifications of the product described within it at any time without notice and without obligation to notify any person of such revision.

Trademarks

AMT is the registered trademark of Apex Material Technology Corp. **PenMount** is a registered trademark of **SALT International Corp.** Microsoft and Windows are registered trademarks of Microsoft Corp. Other product names used in this manual are the properties of their respective owners and are acknowledged.

Copyright

Revision Table

Date	Revision	Changes
25/May/2020	A	Revised Release
30/Nov/2020	B	Add: PenMount K1 Driver Support
26/Jun/2023	C	Add: yocto linux kernel comparison table. Add: 3.4 Trouble shooting for RS-232 / UART interface.
09/Sep/2023	D	Document renamed. Add: 5: Qt setup.

Table of Content

1. Introduction

Touch function requires Linux kernel driver support. Most modern systems support PenMount touch controllers with its inbox driver. To determine whether additional device driver is required, please check the following items.

1. Linux kernel version
   PenMount devices are well supported after Linux kernel 3.
   *The cells that marked **grey color** in table below are systems that require additional drivers provided by PenMount.*

Controller	Interface	Product ID	Kernel Version				
			2.6	3.X	4.X	5.X	6.X
PM9000	RS-232	---		V3.2 penmount			
PM6000	USB	0x6000	generic-usb	V3.5 hid-generic	V3.18 hid-penmount		
		0x6005	generic-usb	V3.5 hid-generic			
	RS-232	---		V3.2 penmount			
P2 / RMT	USB	0x3500		V3.0 hid-multitouch			
		0x3502	generic-usb	V3.5 hid-generic			
		0x3508		3.4 hid-multitouch			
	UART / RS-232	---		V3.2 penmount			
	I2C	---					
	HID over I2C	0x3508			3.8 i2c-hid		
K1	USB	0x14E1	generic-usb (mouse mode)	3.4 hid-multitouch			
	HID over I2C	0x14E1		3.8 i2c-hid			

If using yocto, the default kernel versions are listed in the table below for reference. Please notice that the kernel version and default configurations from different sources could vary significantly. For example, Same yocto codename from Vericite and Advantech could use different kernel versions.

Yocto codename	Yocto Version	Linux kernel	Vericite DART-6UL	Advantech BSP
Jethro	2.0	4.1	3.14.52	4.1.15

Krogoth	2.1	4.4	4.1.15	4.1
Morty	2.2	4.8	4.1.15	4.9.11
Pyro	2.3	4.10	4.1.15	
Rocko	2.4	4.12	4.9.11	4.9.123
Sumo	2.5	4.15	4.9.88	5.4.24
Thud	2.6	4.18	4.14.78	
Warrior	2.7	5.0		
Zeus	3.0	5.2	5.4.3	5.4.3
Dunfell	3.1	5.4	5.4.142	5.4.142
Gatesgarth	3.2	5.8		
Hardknott	3.3	5.10	5.10.72	5.10.72
Honister	3.4	5.14		
Kirkstone	4.0	5.15		5.15.52
Langdale	4.1	5.19		6.1.1
Mickledore	4.2	6.1		

2. Linux Distribution Quirks
   - Red Hat Enterprise Linux / CENTOS 4： using kernel V2.6.9, the bug in generic-usb driver prevents from loading the PenMount 6000 USB device correctly.
   - Red Hat Enterprise Linux / CENTOS 6： using kernel V2.6.32. This version has been patched to support the hid-multitouch driver, so that PenMount P2 USB can be supported.[1]
   - Raspbian / SuSE Enterprise Server 12 SP1: Using kernel 4.X. However, these image does not include **hid-penmount** support, so that PenMount 6000 USB cannot function correctly on those systems.

3. Special Firmware Features
   - Virtual Button: This feature of PenMount P2 USB V6.0 firmware can run without any issue in Linux kernel V4.9 or later systems. Other kernel versions will require modifications of the hid-multitouch device driver.

4. APP Support
   - Android APP： Most modern Android systems uses Linux kernel 3 and later versions, so with USB interface, touch can be plug-and-play. Additional drivers are needed if using resistive touch controllers such as PenMount 6000, which requires calibration with user mode APP.

[1] However, since the Xorg X Server version is too old in RHEL6, touch will still require additional XINPUT drivers to function correctly.

2. USB Interface Support

PenMount USB interface are supported in most Linux kernel versions.

Controller	Interface	USB Product ID	Kernel Version				
			2.6	3.X	4.X	5.X	6.X
PM6000	USB	0x6000	generic-usb	V3.5 hid-generic	V3.18 hid-penmount		
		0x6005	generic-usb	V3.5 hid-generic			
P2 / AMR	USB	0x3500		V3.0 hid-multitouch			
		0x3502	generic-usb	V3.5 hid-generic			
		0x3508		3.4 hid-multitouch			
K1	USB	0x14E1	generic-usb (mouse mode)	3.4 hid-multitouch			

2.1. Kernel Configurations

The kernel drivers must be enabled when configuring the kernel. It touch is not functioning, please check the .config file of the system and be sure that the corresponding kernel drivers are enabled.

Kernel Driver	Configuration	Description
generic-usb	CONFIG_USB_HID=y	USB Human Interface Device (full HID) support
hid-generic	CONFIG_HID_GENERIC=m	Generic HID driver
hid-penmount	CONFIG_HID_PENMOUNT=m	Penmount touch device
hid-multitouch	CONFIG_HID_MULTITOUCH=m	HID Multitouch panels

2.2. Using Customized PenMount Device Drivers

PenMount provide sources codes for systems that do not have building support of PenMount devices.

Item	Description	
Kernel Driver	hid-penmount	
Configuration	CONFIG_HID_PENMOUNT=m	
Source Code	Source Code Path	<penmount_src_dir>/linux/driver/hid/hid-penmount.c
	Copy To	<kernel_src_dir>/drivers/hid/hid-penmount.c
Kconfig File	Path	<kernel_src_dir>/drivers/hid/Kconfig
	Add New	**config** HID_PENMOUNT

	Entry	tristate "Penmount touch device" **depends on** USB_HID **---help---** This selects a driver for the PenMount USB touch controller.
Makefile File	Path	<kernel_src_dir>/drivers/hid/Makefile
	Add New Entry	obj-$(CONFIG_HID_PENMOUNT) += hid-penmount.o

For kernel 2.6, additional setup is required for PenMount P2 USB devices.

Path	Kernel	File	Modification or New Entries
<kernel_src_dir>/drivers/hid		hid-ids.h	#define USB_VENDOR_ID_PENMOUNT 0x14E1 #define USB_DEVICE_ID_PENMOUNT_P2 0x3500 #define USB_DEVICE_ID_PENMOUNT_P2_WIN8 0x3508
	~ 2.6.32	hid.h	#define IS_INPUT_APPLICATION(a) (((a >= 0x00010000) && (a <= 0x00010008)) || (a == 0x00010080) || (a == 0x000c0001) || **((a >= 0x000d0002) && (a <= 0x000d0006))**)

	2.6.33 ~ 2.6.37	hid-core.c	static const struct hid_device_id hid_blacklist[] = {
			// …
			{ HID_USB_DEVICE(USB_VENDOR_ID_PENMOUNT, USB_DEVICE_ID_PENMOUNT_P2) },
			{ HID_USB_DEVICE(USB_VENDOR_ID_PENMOUNT, USB_DEVICE_ID_PENMOUNT_P2_WIN8) },
			{ }
			}

	2.6.38 ~ 2.6.39	hid-core.c	static const struct hid_device_id hid_have_special_driver[] = {
			// …
			{ HID_USB_DEVICE(USB_VENDOR_ID_PENMOUNT, USB_DEVICE_ID_PENMOUNT_P2) },
			{ HID_USB_DEVICE(USB_VENDOR_ID_PENMOUNT, USB_DEVICE_ID_PENMOUNT_P2_WIN8) },
			{ }
			}

2.3. Enable Virtual Key

When using the virtual kernel feature of PenMount P2 USB V5.2 ~ V6.X firmware versions with Linux kernel before 4.9, by default design, these hotkeys will be filtered out by hid-multitouch device drivers. In this case, please follow the steps below to modify hid-multitouch device drivers.

2.3.1. Kernel Before 3.14

Please modify hid-multitouch.c source file and find the following function:

```
static int mt_input_mapping(struct hid_device *hdev, struct hid_input *hi,
                struct hid_field *field, struct hid_usage *usage,
                unsigned long **bit, int *max)
{
```

```
        /* Only map fields from TouchScreen or TouchPad collections.
        * We need to ignore fields that belong to other collections
        * such as Mouse that might have the same GenericDesktop usages. */
        if (field->application != HID_DG_TOUCHSCREEN &&
           field->application != HID_DG_PEN &&
           field->application != HID_DG_TOUCHPAD) {
                return -1;
        }
}
```

Since the HID Top Level Collection Usage Page of Virtual Keyis HID_UP_KEYBOARD, it will be ignored. Please change the code and return 0, instead of -1, if the target device is PenMount.

```
static int mt_input_mapping(struct hid_device *hdev, struct hid_input *hi,
                struct hid_field *field, struct hid_usage *usage,
                unsigned long **bit, int *max)
{
        /* Only map fields from TouchScreen or TouchPad collections.
        * We need to ignore fields that belong to other collections
        * such as Mouse that might have the same GenericDesktop usages. */
        if (field->application != HID_DG_TOUCHSCREEN &&
           field->application != HID_DG_PEN &&
           field->application != HID_DG_TOUCHPAD) {
                if (hi->input->id.vendor == USB_VENDOR_ID_PENMOUNT) {
                        return 0;
                } else {
                        return -1;
                }
        }
}
```

2.3.2. Kernel 3.15 to 4.9

Please open hid-multitouch source file, and find the following array.

```
static const struct hid_device_id mt_devices[] = {

        /* Generic MT device */
        { HID_DEVICE(HID_BUS_ANY, HID_GROUP_MULTITOUCH, HID_ANY_ID, HID_ANY_ID) },

        /* Generic Win 8 certified MT device */
        {  .driver_data = MT_CLS_WIN_8,
                HID_DEVICE(HID_BUS_ANY, HID_GROUP_MULTITOUCH_WIN_8,
                        HID_ANY_ID, HID_ANY_ID) },
        { }
};
```

Please add a new entry before “Generic MT device”:

```
static const struct hid_device_id mt_devices[] = {

        /* PenMount devices */
```

```
        { .driver_data = MT_CLS_EXPORT_ALL_INPUTS,
                MT_USB_DEVICE(USB_VENDOR_ID_PENMOUNT,
                        0x3508) },

        /* Generic MT device */
        { HID_DEVICE(HID_BUS_ANY, HID_GROUP_MULTITOUCH, HID_ANY_ID, HID_ANY_ID) },

        /* Generic Win 8 certified MT device */
        {  .driver_data = MT_CLS_WIN_8,
                HID_DEVICE(HID_BUS_ANY, HID_GROUP_MULTITOUCH_WIN_8,
                        HID_ANY_ID, HID_ANY_ID) },
        { }
};
```

2.3.3. Kernel 4.10 and Later

Kernel 4.10 and later versions can support the PenMount virtual key functions correctly without any modification required.

3. RS-232 / UART Interface Support

The PenMount RS-232 interface are supported after kernel V3.2.

Controller	Interface	USB Product ID	Kernel Version				
			2.6	3.X	4.X	5.X	6.X
PM9000	RS-232	---		V3.2 penmount			
PM6000	RS-232	---		V3.2 penmount			
P2 / RMT	RS-232	---		V3.2 penmount			

3.1. Enable Kernel Support

The kernel drivers must be enabled when configuring the kernel. If touch is not functioning, please check the .config file of the system and be sure that the corresponding kernel drivers are enabled.

Kernel Driver	Configuration	Description
serport	CONFIG_SERIO=y	Serial I/O support
	CONFIG_SERIO_SERPORT=m	Serial port line discipline
serio-penmount	CONFIG_TOUCHSCREEN_PENMOUNT=m	Penmount serial touchscreen

Although some of the Linux kernels prior to V3.2 may include penmount kernel driver, these are for products that already phased out. In this case, please use the customized PenMount kernel driver.

Please also notice that the inbox driver after kernel V3.2 supports dual touch only. For multitouch support, please use the customized device driver provided by PenMount.

3.2. Using Customized Kernel Driver

PenMount provides customized kernel driver that supports Linux kernel prior to V3.2.

Item	Description	
Kernel Driver	serio-penmount	
Configuration	CONFIG_TOUCHSCREEN_PENMOUNT=m	
Source Code	Source Code Path	<penmount_src_dir>/linux/driver/serial/penmount.c
	Copy To	<kernel_src_dir>/drivers/input/touchscreen/penmount.c
Kconfig File	Path	<kernel_src_dir>/drivers/input/touchscreen/Kconfig
	Add New Entry	Not required
Makefile File	Path	<kernel_src_dir>/drivers/input/touchscreen/Makefile
	Add New	Not required

	Entry	

3.3. Loading the PenMount Serial kernel driver

The PenMount RS-232/UART kernel driver is a SERIO device driver, which requires additional utility to launch manually.

There is a public utility called inputattach which supports various PenMount RS-232 / UART devices. PenMount also provides it's own utility. Please select one for compilation.

Utility	Source Code Path	Example (Suppose device on /dev/ttyS1)
pmsAttach	<penmount_src_dir>/linux/driver/serial/pmsAttach	pmsAttach 9000 /dev/ttyS1 19200
		pmsAttach 6000 /dev/ttyS1 19200
		pmsAttach PCI /dev/ttyS1 38400
inputattach	https://github.com/flosse/linuxconsole/	inputattach –baud 19200 –pm9k /dev/ttyS1
		inputattach –baud 19200 –pm6k /dev/ttyS1
		inputattach –baud 38400 –pm3k /dev/ttyS1

3.4. Trouble Shooting

1. PenMount RS-232 / UART touchscreen not working. Here is a quick check list.

Item	Action	Command
Kernel Driver	Check if kernel driver module exists.	ls /lib/modules/`uname -r`/kernel/drivers/input/touchscreen/penmount*
Use inputattach	Please don't forget that RS-232 / UART device is not PnP, you will need to use launch it manually.	inputattach <mode> <port>
Check event device	Check if PenMount event device is created.	cat /proc/bus/input/devices | grep PenMount -b8 -a8
Test input event	Use helper utilities such as evtest to check if touch is responding. If not, please double check the connection cables, baudrate settings are setup correctly.	evtest

When PenMount kernel device is running correctly, messages could found in dmesg.

```
[   88.588595] serio: Serial port ttyS0
[   88.617997] input: PenMount Serial TouchScreen as /devices/pnp0/00:08/tty/ttyS0/serio2/input/input5
```

2. Running the inputattach utility returns with error “can't set line discipline”.
   When this happens, it indicates that the Linux kernel does not include serport support, please reconfigure the kernel with CONFIG_SERIO_SERPORT first.

4. I2C Interface Support

For PenMount P2 I2C interface, please use the customized PenMount kernel driver. For PenMount K1, it is supported in Linux kernel after 5.1.

<table>
<tr><th rowspan="2">Controller</th><th rowspan="2">Interface</th><th rowspan="2">Product ID</th><th colspan="5">Kernel Version</th></tr>
<tr><th>2.6</th><th>3.X</th><th>4.X</th><th>5.X</th><th>6.X</th></tr>
<tr><td rowspan="2">P2 / RMT</td><td>I2C</td><td>---</td><td colspan="5"></td></tr>
<tr><td>HID over I2C</td><td>0x3508</td><td></td><td colspan="4">V3.8
i2c-hid</td></tr>
<tr><td rowspan="2">K1</td><td>HID over I2C</td><td>0x14E1</td><td></td><td colspan="4">V3.8
i2c-hid</td></tr>
<tr><td>I2C</td><td>ili251x</td><td colspan="3"></td><td colspan="2">V5.1
Ili210x</td></tr>
</table>

4.1. Using Customized PenMount P2 Driver

Item		Description
Kernel Driver	penmount_i2c	
Configuration	CONFIG_TOUCHSCREEN_PENMOUNT_I2C=m	
Source Code	Source Code Path	<penmount_src_dir>/linux/driver/i2c/penmount-i2c.c
	Copy To	<kernel_src_dir>/drivers/input/touchscreen/penmount-i2c.c
Kconfig File	Path	<kernel_src_dir>/drivers/input/touchscreen/Kconfig
	Add New Entry	**config** PENMOUNT_TOUCHSCREEN_I2C tristate "Penmount I2C touchscreen" **depends on** I2C **---help---** This selects a driver for the PenMount I2C touch controller.
Makefile File	Path	<kernel_src_dir>/drivers/input/touchscreen/Makefile
	Add New Entry	obj-$(CONFIG_TOUCHSCREEN_PENMOUNT_I2C) += penmount-i2c.o

4.2. Interrupt Setup

The PenMount P2 I2C device driver can operate with Interrupt driven mode or polling mode. In most cases, it is recommended using interrupt driven mode by connecting the INT PIN to a selected GPIO pin on board.

There are several ways to configure the interrupt pin.

Option	Actions	Description	Example
Modify PenMount	Define the PENMOUNT_I2C_GPIO_IRQ	Define the GPIO pin	#define GPIO_TO_PIN(bank, gpio) (32 * (bank) + (gpio))

Driver Source Code			GPIO_TO_PIN(3,19)
	Implement the penmount_i2c_init_gpio ()	Configure GPIO to be input, pull high, and low level trigged.	
Use Device Tree Overlays	Configure the "interrupt-parent" "Interrupts" items	Define the GPIO pin and configure to be input, pull high, and low level trigged.	interrupt-parent = <&gpio3>; interrupts = <19 8>; /* IRQ_TYPE_LEVEL_LOW */

4.3. Auto launch the device driver

The PenMount P2 I2C driver is a I2C client driver that requires additional configurations to be launched correctly. There are two possible options.

Option	Modifications	Path (Example)	Configurations (Example)

Modify platform settings directly.	i2c_board_info[]	(S3C2440) <kernel_src_dir>/kernel/arch/arm/mach-s3c2440/mach-s3c2440.c	Find friendly_arm_i2c_devices[] and add the following entry.
			#include <linux/i2c.h>
			
			static struct i2c_board_info friendly_arm_i2c_devices[] __initdata = {
			{ **I2C_BOARD_INFO("penmount_i2c", 0x38)** },
			};
			
			static void __init mini2440_machine_init(void) {
			i2c_register_board_info(0, friendly_arm_i2c_devices, ARRAY_SIZE(friendly_arm_i2c_devices));
			s3c_i2c0_set_platdata(NULL);
			}

Use DeviceTree Overlays	compatible reg	PenMount P2	compatible = "penmount,penmount_i2c"; reg = <0x38>; interrupt-parent = <&gpio>; interrupts = <24 8>;
		PenMount K1	compatible = "hid-over-i2c"; reg = <0x41>; hid-descr-addr = <0x0001>; interrupt-parent = <&gpio>; interrupts = <24 8>;

5. Qt Setup

Qt is a C++ application development framework primarily used for developing graphical interfaces in embedded systems. Currently, there are main versions of Qt, which are Qt 5 and Qt 6. They differ in their level of support for touchscreens, as outlined below:

Plugin	configure	Qt 5.0 ~ 5.5	QT5.6 ~ Qt6	Resistive Touch Calibration	Multi-Touch Support
tslib	-tslib	V	V	V	
evdevtouch	-evdev	**V**	V		V
libinput	-libinput		**V**	V	V

For resistive touchscreens, additional calibration is required, so in the Qt framework, additional configurations with tslib or libinput are needed for proper functionality.

5.1. Recompile Qt

Generally, Linux SDKs provided for embedded systems include Qt libraries. However, in some cases, to save space, they may not include touchscreen support. In such cases, you'll need to recompile Qt from source with the necessary support. Below are some commonly used Qt version source code download links for reference.

Qt Version	Source Code link
5.9	https://download.qt.io/archive/qt/5.9/5.9.9/single/qt-everywhere-opensource-src-5.9.9.tar.xz
5.12	https://download.qt.io/archive/qt/5.12/5.12.11/single/qt-everywhere-src-5.12.11.tar.xz
5.15	https://download.qt.io/official_releases/qt/5.15/5.15.10/single/qt-everywhere-opensource-src-5.15.10.tar.xz
6.2	https://download.qt.io/official_releases/qt/6.2/6.2.5/single/qt-everywhere-opensource-src-6.2.5.tar.xz
6.5	https://download.qt.io/archive/qt/6.5/6.5.2/single/qt-everywhere-src-6.5.2.tar.xz

5.1.1. Compile Qt5 / Qt6

Qt5 have provided support for resistive touchscreen calibration through tslib and libinput. Please check if your Qt5 installation includes the following files:

Plugin	Path	Calibration Utility
libinput	$QT_PLUGINS_DIR/generic/libqlibinputplugin.so	pm-qcalib
tslib	$QT_PLUGINS_DIR/generic/libqtslibplugin.so	ts_calibrate

If your Qt5 installation does not include the aforementioned support, you will need to recompile Qt5.

```
./configure \
-prefix $QTDIR \
-opensource \
-confirm-license \
-release -shared \
-platform linux-g++-64 \
-xplatform linux-arm-gnueabihf-g++ \
-optimized-qmake \
-pch \
-qt-sql-sqlite \
-qt-libjpeg \
-qt-zlib \
-qt-libpng \
-qt-freetype \
-no-opengl \
-no-openssl \
-no-cups \
-no-glib \
-skip webkit \
-skip webkit-examples \
-nomake examples \
-nomake tests \
-no-xcursor -no-xfixes -no-xrandr -no-xrender \
-no-separate-debug-info \
-tslib \
-I$TSLIB_DIR/include \
-L$TSLIB_DIR/lib
```

```
make
```

```
sudo make install
```

You can refer to the following link for additional settings:

https://wiki.qt.io/Building_Qt_5_from_Git#Configuring_and_Building

https://wiki.qt.io/Building_Qt_6_from_Git#Configuring_and_Building

5.2. Resistive Touchscreen Configuration

If not configured specifically, Qt will use the default input handler to drive the touchscreen, and only libinput provides a calibration mechanism. Resistive touchscreens may experience inaccurate touch positions before calibration, so additional settings are required for most resistive screens to perform calibration.

Qt Version	Default Handler
Qt 5.0 ~ 5.5	evdevtouch
Qt 5.6 ~ 6	libinput

5.2.1. Example of Setting Up tslib for Qt5/Qt6

If you are using Qt versions 5.6 and later, the default libinput support should allow you to calibrate resistive touchscreens directly with the pm-qcalib program.

However, if you need to use tslib with the ts_calibrate positioning program, you'll need to follow the instructions in this section to set up tslib with Qt. Here are the main steps to follow, but please adjust the values based on your specific setup. The example below assumes that the PenMount input device is named /dev/input/event4:

Platform	Setting	Description	Notes
x11	QT_QPA_PLATFORM	export QT_QPA_PLATFORM=xcb	By default, X Window driver is used, and PenMount X Window driver and calibration program are used. No additional setup of tslib is required.[2]
wayland	QT_QPA_PLATFORM	export QT_QPA_PLATFORM=wayland	By default libinput is used , the calibration procedure can be matched with weston-calibrator. No additional setup of tslib is required.
eglfs	QT_QPA_PLATFORM	export QT_QPA_PLATFORM=eglfs	
	QT_QPA_EGLFS_NO_LIBINPUT	（**Required in Qt5.6 ~ 6**）Set not to use the default libinput. export QT_QPA_EGLFS_NO_LIBINPUT=1	
	QT_QPA_EGLFS_TSLIB	（**Qt5.5~**）Specify the input handler to use tslib to handle input. export QT_QPA_EGLFS_TSLIB=1	Select One
	QT_QPA_GENERIC_PLUGINS	（**Qt5.0~**）Specifies to load the tslib plugin. export QT_QPA_GENERIC_PLUGINS=tslib:/dev/input/event4 Requirement： $QT_PLUGIN_DIR/plugins/generic/libqtslibplugin.so	
linuxfb	QT_QPA_PLATFORM	export QT_QPA_PLATFORM=linuxfb	
	QT_QPA_FB_NO_LIBINPUT	（**Required in Qt5.6 ~ 6**）Set not to use the default libinput. export QT_QPA_FB_NO_LIBINPUT=1	

[2] If QT_QPA_GENERIC_PLUGINS=tslib is set under the xcb platform, the Qt program will receive repeated input events, resulting in abnormal actions. Please check and remove the settings.

	QT_QPA_FB_TSLIB	（**Qt5.5~**）Specify the input handler to use tslib to handle input. export QT_QPA_FB_TSLIB=1	Chose One
	QT_QPA_GENERIC_PLUGINS	（**Qt5.0~**）Specifies to load the tslib plugin. export QT_QPA_GENERIC_PLUGINS=tslib:/dev/input/event4 Requirements： $QT_PLUGIN_DIR/plugins/generic/libqtslibplugin.so	

For platforms that need to use the tslib calibration program, additional settings are required as follows:

Plugin	Setting	Description	Notes
tslib	TSLIB_TSDEVICE	The event device path for the touch screen. export TSLIB_TSDEVICE =/dev/input/event4	
	TSLIB_CALIBFILE	(Optional) Calibration data file path for the touch screen. export **TSLIB_CALIBFILE** =/etc/pointercal	
	TSLIB_CONFFILE	(Optional) Touch screen settings file path. export **TSLIB_CONFFILE**=/etc/ts.conf Please also set the grab_events in ts.conf module_raw input **grab_events=1**	

For more configuration methods, please refer to the following links:

https://doc.qt.io/qt-5/embedded-linux.html

https://doc.qt.io/qt-6/embedded-linux.html