MT793X IoT SDK for GPIO_EINT User Guide

Version: 0.2
Release date: 2021-04-01

Version History

Version	Date	Description
0.1	2021-03-25	Initial release
0.2	2021-04-01	Add EINT API and sample code

Table of Contents

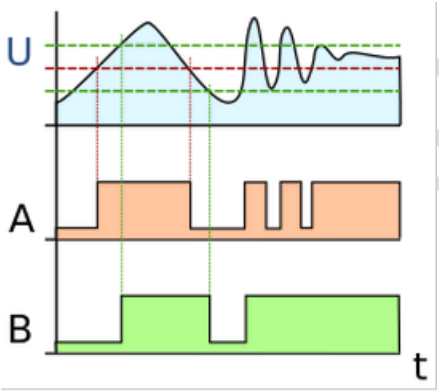
List of Figures

1 Overview

1.1 Hardware Features

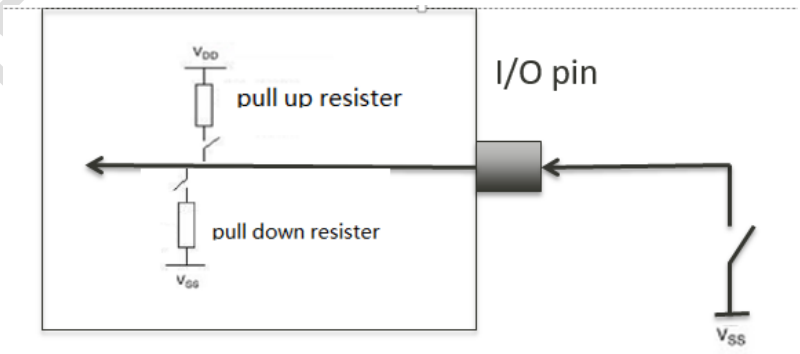
- Need to balance power consumption and speed of response
- Schmitt trigger hysteresis
- Can be pull-up, pull-down or disable pull when direction is input
- Can be push-pull when direction is output
- Driving strength: 2mA, 4mA, 6mA, 8mA

- Schmitt trigger
  - A Schmitt trigger is a comparator circuit with hysteresis implemented by applying positive feedback to the noninverting input of a comparator or differential amplifier



The output of a Schmitt trigger (B) and a comparator (A), when a noisy signal (U) is applied. The green dotted lines are the circuit's switching thresholds. The Schmitt trigger tends to remove noise from the signal.

- Pull up/down
  - Assign the input data of pin to be high/low if the pin is left unconnected
  - Only work on input data



- PUPD/R0/R1
  - Some pins have more than one pull-up/pull-down resister

- Resister is decided by different combinations of PUPD/R0/R1

21	sysrst_b	SW	**Control PAD_SYSRST_B R0 pin. R1R0=00:High-Z, R1R0=01:10kohm, R1R0=10:50kohm, R1R0=11:10kohm/50kohm.**

1.2 Register Definition

- GPIO mode control register.
- GPIO direction control register.
- GPIO pull-up/pull-down control register.
- GPIO data output register.
- GPIO data input register.
- GPIO IES Control register.
- GPIO DRV Control register.

1.3 Mapping Relationship between EINT and GPIO

EINT NUMBER	GPIO NUMBER
HAL_EINT_NUMBER_0	HAL_GPIO_8
HAL_EINT_NUMBER_1	HAL_GPIO_9
HAL_EINT_NUMBER_2	HAL_GPIO_10
…….	……
HAL_EINT_NUMBER_16	HAL_GPIO_24
HAL_EINT_NUMBER_17	**HAL_GPIO_43**
HAL_EINT_NUMBER_18	HAL_GPIO_44
HAL_EINT_NUMBER_19	HAL_GPIO_45
HAL_EINT_NUMBER_20	**HAL_GPIO_28**
HAL_EINT_NUMBER_21	HAL_GPIO_29
…….	……
HAL_EINT_NUMBER_30	HAL_GPIO_38

2 Driver Introduction

2.1 Driver Architecture

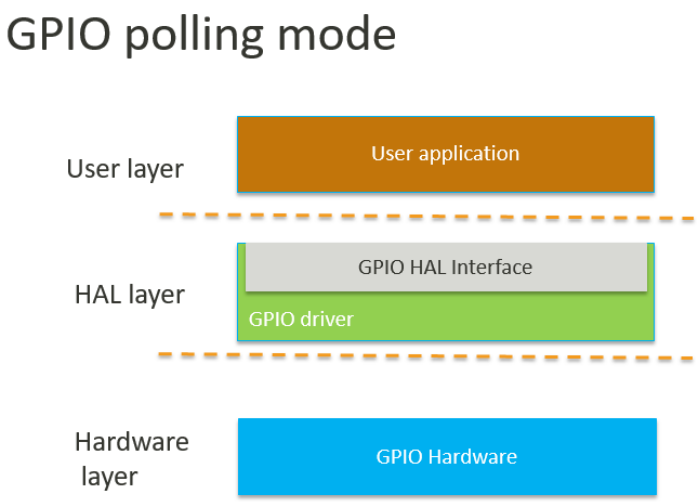


Figure 1. GPIO Polling Mode

2.2 Driver API Reference

API	Introduction
hal_gpio_init()	Initialize basic function. Must be called before used
hal_gpio_deinit()	De-initialize the GPIO module to return to the initial state. Must be called if the module is not used
hal_pinmux_set_function()	Set pinmux of target pin
hal_pinmux_get_function()	Get pinmux value of target pin
Hal_gpio_set_analog()	Set the analog function for the target pin

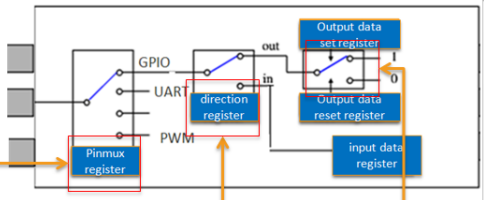
API	Introduction
hal_gpio_toggle_pin	Toggle the output data for the targer pin when the direction of the pin is output.
hal_gpio_set_driving_current	Set the driving current for the target pin
hal_gpio_get_driving_current	Get the driving current for the target pin
hal_gpio_set_direction()	Set direction of target pin
hal_gpio_get_direction()	Get direction of target pin
hal_gpio_set_output()	Set output data of target pin
hal_gpio_get_output()	Get output data of target pin
hal_gpio_get_input()	Get input data of target pin
hal_gpio_pull_up()	GPIO mode + input + disable pull may lead to leakage
hal_gpio_pull_down()	
hal_gpio_disable_pull()	
hal_gpio_set_high_impedance()	Avoid leakage in case of GPIO mode + input +no pull
hal_gpio_clear_high_impedance()	Out of high-z state

API	Introduction
hal_gpio_set_pupd_register()	Pull configuration if pin has PUPD/R0/R1 Notes: Only work on pin that has PUPD/R0/R1
hal_eint_init	Initialize trigger type of specified EINT
hal_eint_set_debounce_time	Set debounce time(ms)
hal_eint_register_callback	Register specified EINT IRQ callback
hal_eint_set_trigger_mode	Set trigger type of specified EINT
hal_eint_deinit	Uninstall specified EINT IRQ callback
hal_eint_mask	Disable specified EINT
hal_eint_unmask	Enable specified EINT

2.3 Sample Code

```
/** @defgroup hal_gpio_enum Enum
  * @{
  */

/** @brief This enum defines the GPIO direction. */
typedef enum {
    HAL_GPIO_DIRECTION_INPUT  = 0,              /**<  GPIO input direction. */
    HAL_GPIO_DIRECTION_OUTPUT = 1               /**<  GPIO output direction. */
} hal_gpio_direction_t;


/** @brief This enum defines the data type of GPIO. */
typedef enum {
    HAL_GPIO_DATA_LOW  = 0,                     /**<  GPIO data low. */
    HAL_GPIO_DATA_HIGH = 1                      /**<  GPIO data high. */
} hal_gpio_data_t;

/** @brief This enum defines the function index of GPIO. */
typedef enum {
    HAL_GPIO_MODE_0 = 0,                        /**<  GPIO mode 0. */
    HAL_GPIO_MODE_1 = 1,                        /**<  GPIO mode 1. */
    HAL_GPIO_MODE_2 = 2,                        /**<  GPIO mode 2. */
    HAL_GPIO_MODE_3 = 3,                        /**<  GPIO mode 3. */
    HAL_GPIO_MODE_4 = 4,                        /**<  GPIO mode 4. */
    HAL_GPIO_MODE_5 = 5,                        /**<  GPIO mode 5. */
    HAL_GPIO_MODE_6 = 6,                        /**<  GPIO mode 6. */
    HAL_GPIO_MODE_7 = 7                         /**<  GPIO mode 7. */
} hal_gpio_mode_t;
```

Set GPIO0 to output high



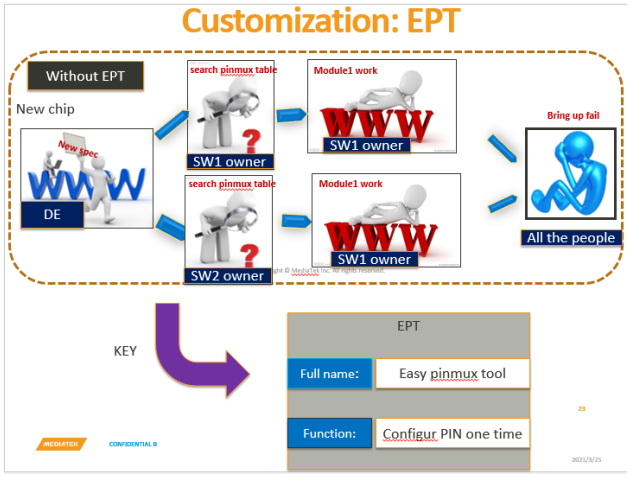

```
void gpio_application(void)
{
  hal_gpio_status_t ret;
  hal_pinmux_status_t ret_pinmux_status;
  ret = hal_gpio_init(HAL_GPIO_0);
   /*  set the pin to work in GPIO mode  */
  ret_pinmux_status = hal_pinmux_set_function(HAL_GPIO_0, HAL_GPIO_0_GPIO0);
  ret = hal_gpio_set_direction(HAL_GPIO_0, HAL_GPIO_DIRECTION_OUTPUT);
  ret = hal_gpio_set_output(HAL_GPIO_0, HAL_GPIO_DATA_HIGH);
  ret = hal_gpio_deinit(HAL_GPIO_0);
}
```

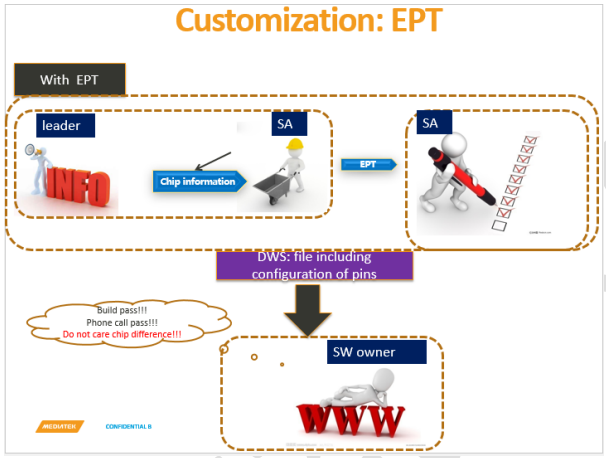
Steps to be used as EINT:

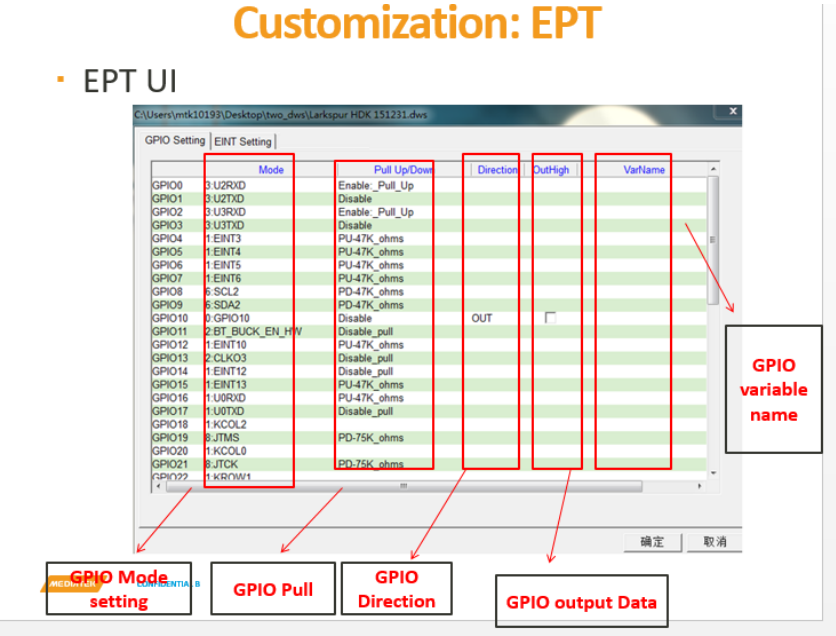
```
hal_pinmux_status_t pinmux_status;
hal_eint_config_t eint_config = {0};
uint32_t duration_count;

hal_eint_number_t _Tst_EINT_num = HAL_EINT_NUMBER_0;

printf("ut_hal_eint enter \r\n");


eint_config.trigger_mode = HAL_EINT_EDGE_FALLING;
eint_config.debounce_time = 0;

// Init test GPIO
hal_gpio_init(_Tst_GPIO_PIN);

pinmux_status = hal_pinmux_set_function(_Tst_GPIO_PIN, MT7933_PIN_8_FUNC_CM33_GPIO_EINT0);

if (HAL_PINMUX_STATUS_OK != pinmux_status) {
    printf("ut_hal_eint set pinmux function error, pin = %d \r\n", _Tst_GPIO_PIN);
    return UT_STATUS_ERROR;
}

hal_eint_mask(_Tst_EINT_num);

if (HAL_EINT_STATUS_OK != hal_eint_register_callback(_Tst_EINT_num, callback, NULL)) {
    printf("ut_hal_eint register callback error \r\n");
    return UT_STATUS_ERROR;
}

if (HAL_EINT_STATUS_OK != hal_eint_init(_Tst_EINT_num, &config)) {
    printf("ut_hal_eint eint init error \r\n");
    return UT_STATUS_ERROR;
}

if (config.trigger_mode == HAL_EINT_LEVEL_LOW) {
    // Change debounce time at runtime
    if (HAL_EINT_STATUS_OK != hal_eint_set_debounce_time(_Tst_EINT_num, debounce_time_ms)) {
        printf("ut_hal_eint set debounce time error \r\n");
        return UT_STATUS_ERROR;
    }
}
hal_eint_unmask(_Tst_EINT_num);

// Clean up test env
hal_eint_deinit(_Tst_EINT_num);
```

3 Customization: EPT

3.1.1 EPT Tool Introduction







3.1.2 How to Use EPT

1. Configure pin by EPT

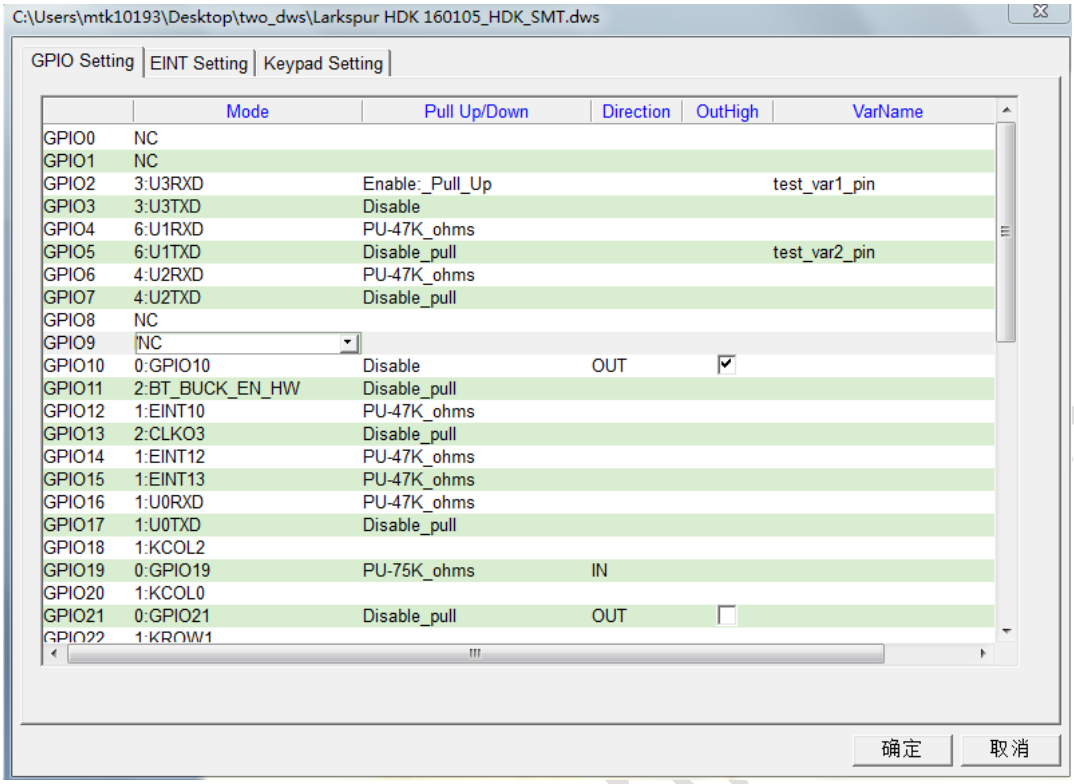
Mode: select by drop-down menu of the “Mode” column

Pull Up/Down: select by the drop-down menu(pull/pull/down/disable pull)

Direction: select by the drop-down menu(input/output)

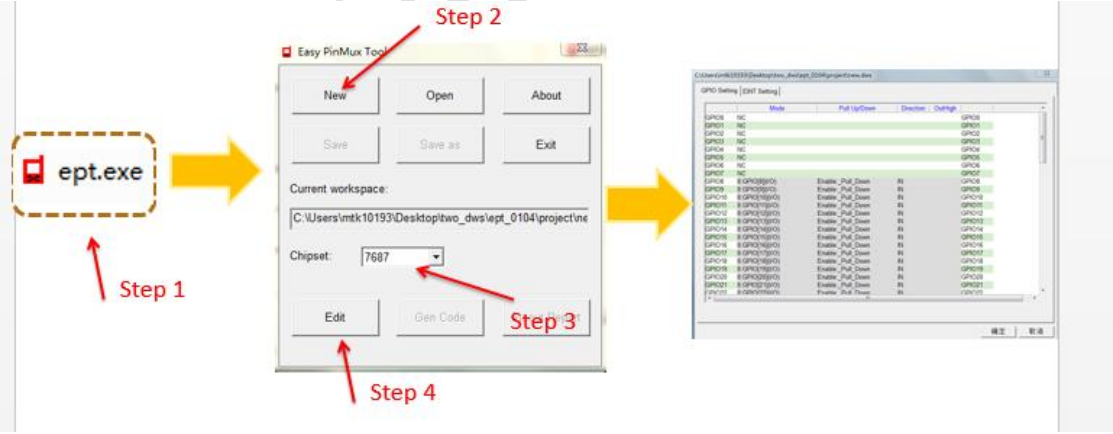
OutputHigh: set output data to HIGH if this item is ticked

VarName: select by the drop-down menu to set variable to target pin



2. Create one new DWS file

1. Double click ept.exe(/tools/mtk/ept_tool/)
2. Click “**new**” and select the workspace
3. Select chipset
4. Click “**Edit**” to open UI

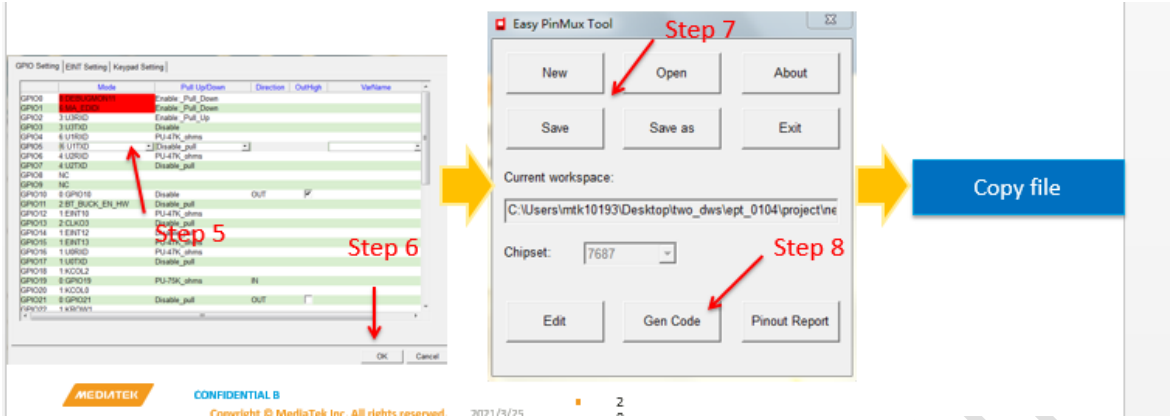


5. Configure the pin with EPT(items changed are highlighted in red)
6. Click “OK” after configuration is done
7. Click Save” to save configuration
8. Click “Gen Code”. The generated file path:

   the generated .h files is located in /tools/mtk/ept_tool/output/
9. Copy the generated file to the target path

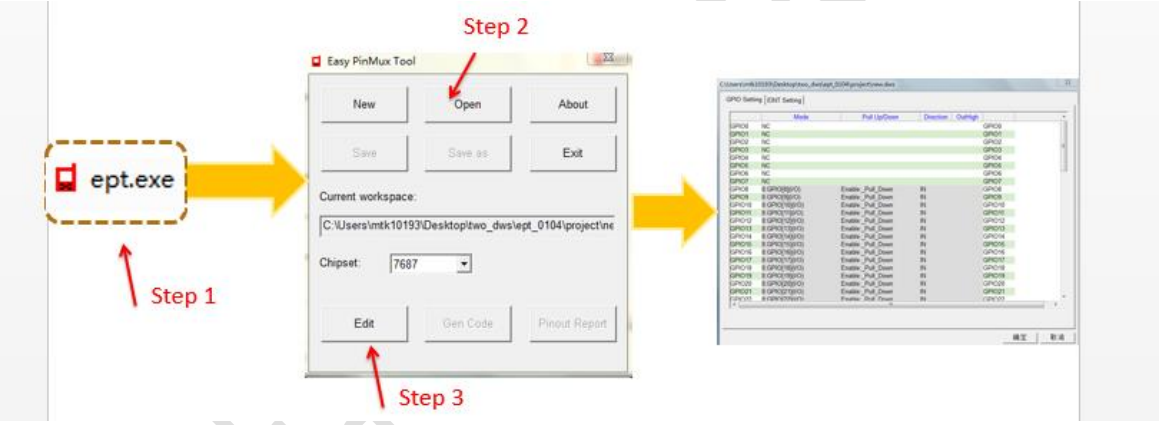
Copy .h file of the generated files to \project\<board>\apps\<application>\inc

Copy .c file of the generated files to \project\<board>\apps\<application>\src



2. Open one existing DWS file：

1. Double click ept.exe (/tools/mtk/ept_tool/)
2. Click “Open” and select the workspace
3. Click “Edit” to open UI



4. Click “OK” after configuration
5. Click “Save” to save configuration
6. Click “Gen Code”

   The generated file path: the generated .h files is located in in /tools/mtk/ept_tool/output/
7. Copy the generated file to the target path

   Copy .h file of the generated files to \project\<board>\apps\<application>\inc

   Copy .c file of the generated files to \project\<board>\apps\<application>\src

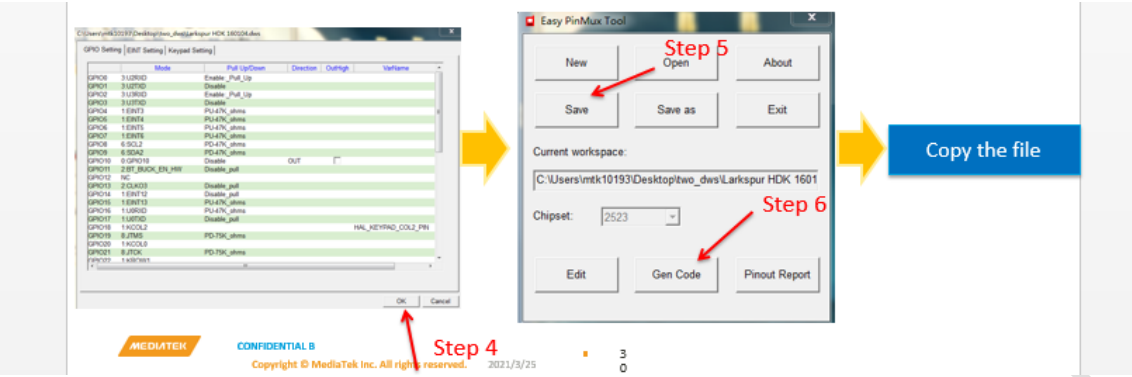


Exhibit 1 Terms and Conditions

Your access to and use of this document and the information contained herein (collectively this "Document") is subject to your (including the corporation or other legal entity you represent, collectively "You") acceptance of the terms and conditions set forth below ("T&C"). By using, accessing or downloading this Document, You are accepting the T&C and agree to be bound by the T&C. If You don't agree to the T&C, You may not use this Document and shall immediately destroy any copy thereof.

This Document contains information that is confidential and proprietary to MediaTek Inc. and/or its affiliates (collectively "MediaTek") or its licensors and is provided solely for Your internal use with MediaTek's chipset(s) described in this Document and shall not be used for any other purposes (including but not limited to identifying or providing evidence to support any potential patent infringement claim against MediaTek or any of MediaTek's suppliers and/or direct or indirect customers). Unauthorized use or disclosure of the information contained herein is prohibited. You agree to indemnify MediaTek for any loss or damages suffered by MediaTek for Your unauthorized use or disclosure of this Document, in whole or in part.

MediaTek and its licensors retain titles and all ownership rights in and to this Document and no license (express or implied, by estoppels or otherwise) to any intellectual propriety rights is granted hereunder. This Document is subject to change without further notification. MediaTek does not assume any responsibility arising out of or in connection with any use of, or reliance on, this Document, and specifically disclaims any and all liability, including, without limitation, consequential or incidental damages.

THIS DOCUMENT AND ANY OTHER MATERIALS OR TECHNICAL SUPPORT PROVIDED BY MEDIATEK IN CONNECTION WITH THIS DOCUMENT, IF ANY, ARE PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, WHETHER EXPRESS, IMPLIED, STATUTORY, OR OTHERWISE. MEDIATEK SPECIFICALLY DISCLAIMS ALL WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, FITNESS FOR A PARTICULAR PURPOSE, COMPLETENESS OR ACCURACY AND ALL WARRANTIES ARISING OUT OF TRADE USAGE OR OUT OF A COURSE OF DEALING OR COURSE OF PERFORMANCE. MEDIATEK SHALL NOT BE RESPONSIBLE FOR ANY MEDIATEK DELIVERABLES MADE TO MEET YOUR SPECIFICATIONS OR TO CONFORM TO A PARTICULAR STANDARD OR OPEN FORUM.

Without limiting the generality of the foregoing, MediaTek makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does MediaTek assume any liability arising out of the application or use of any product, circuit or software. You agree that You are solely responsible for the designing, validating and testing Your product incorporating MediaTek's product and ensure such product meets applicable standards and any safety, security or other requirements.

The above T&C and all acts in connection with the T&C or this Document shall be governed, construed and interpreted in accordance with the laws of Taiwan, without giving effect to the principles of conflicts of law.