MT793X IoT SDK for SWLA User Guide

Version: 1.1

Release date: 2022-10-06

Version History

Version	Date	Author	Description
0.1	2021-11-02	Eason Lai	Initial draft for temporarily release
1.0	2021-11-15	Eason Lai	Add custom tag and label APIs
1.1	2022-10-06	Eason Lai	Get SWLA information from flash

Table of Contents

List of Figures

1 Overview

SWLA stands for Software Logic Analyzer. The goal of SWLA is to record and view the task/ISR execution sequence and timing as shown below.

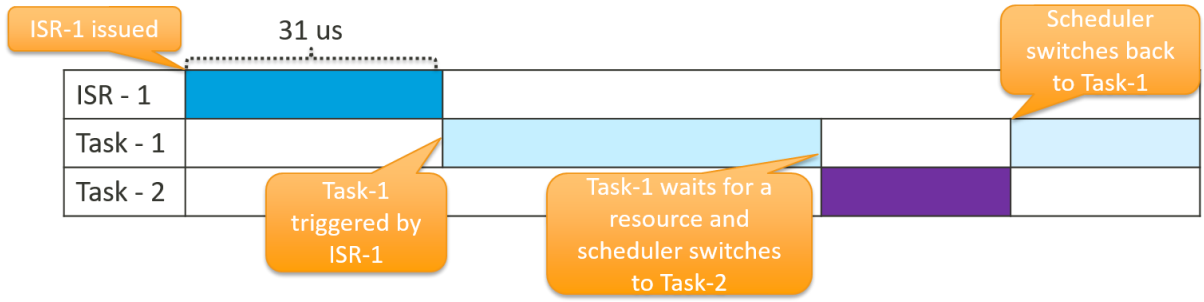

Figure 1 Schematic Diagram of SWLA

1.1 Configuration Options

You can enable or disable SWLA from the feature.mk by changing the value of **MTK_SWLA_ENABLE**.

- If you would like to enable SWLA, please change **MTK_SWLA_ENABLE to y**
- If you would like to disable SWLA, please change **MTK_SWLA_ENABLE to n**

Once SWLA is enabled, the default buffer is placed in PSRAM. You can change **MTK_SWLA_USE_SYSRAM_BUFFER** to **y** to move the SWLA buffer to SYSRAM. The default size of the buffer is 48KB, which is defined by **SWLA_BUFFER_SIZE** in **project/[board]/apps/[project]/inc/kernel_service_config.h**

1.2 How to Use SWLA

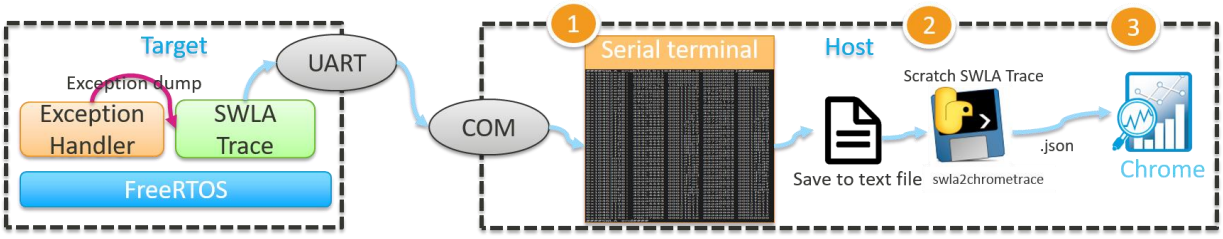

Figure 2 Process of Using SWLA

Step 1. Collect SWLA's Trace

- SWLA's trace is output through the UART log if an exception occurs
  - Start with `**####SWLA enabled[...]####**`
  - End with `**###SWLA end####**`
- Open the UART log with the serial terminal to get SWLA's trace.
- Save the whole UART log to a text file (including other system logs)

Step 2. Extract SWLA's Trace to Convert the Trace to the Chrome Trace Format

- swla2chrometrace.exe is a necessary program in this step.
- Drag the UART/exception log collected from Step 1 to swla2chrometrace.exe
  - The program scratches SWLA's trace from the log and converts the trace to the Chrome trace format.
    - `**<log file name>_swla_info.json**` is generated in the same folder, if the command is successful.
  - If the command fails, please check the FAQ section.

Figure 3 Convert SWLA's Trace to Chrome Trace Format

Step 3. Open the JSON file with Chrome

- Open Chrome with the URL, chrome://tracing
- Drag the JSON file generated from Step 2 and drop the file into Chrome

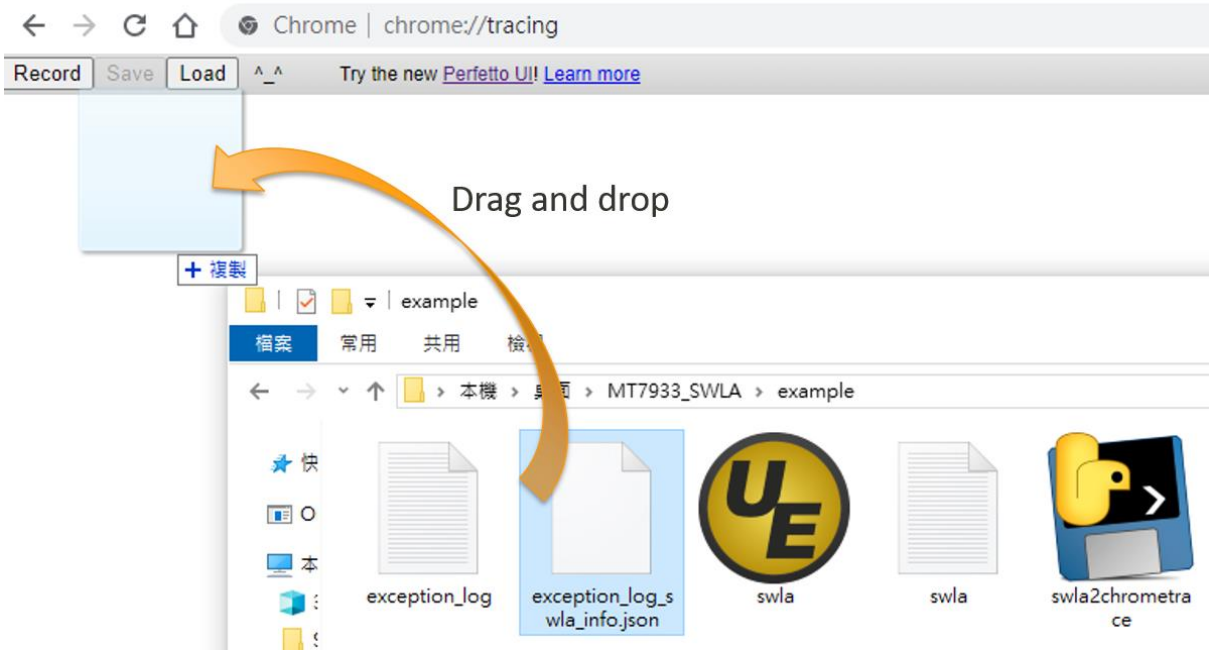

Figure 4 Open the JSON File with Chrome

2 Custom Tags and Labels

SWLA allows you to indicate special regions or events on the view of tasks/ISRs execution as shown below.

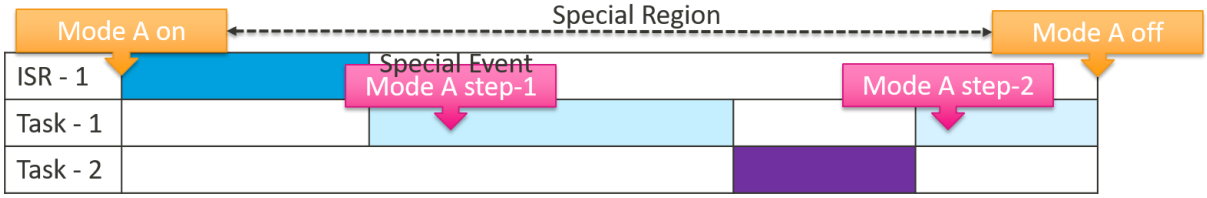

Figure 5 Schematic Diagram of SWLA Tags and Labels

To indicate a special region or event, you can utilize the API, **SLA_CustomLogging.** There are two things you need to know:

- The API increases an overhead within 5us each call.
- The custom tag or label is marked within 3 letters to decrease the overhead and trace size.

```
void SLA_CustomLogging(uint8_t  *pxCustomLabel,
SA_ACTION_T xAction)
     - xAction : SA_START/SA_STOP for custom
logging start/end
      Logging the certain event duration and it is
not influenced by others
     - xAction : SA_LABEL for one shot custom
logging
            Logging the certain event happen
```

2.1 Examples

2.1.1 Custom Tags

A special region is created by using the API, **SLA_CustomLogging**, twice. You need to call the API twice and pass **SA_START** and **SA_STOP** into the **xAction** parameter. If the tag, SA_START or SA_STOP, is not paired, the tag is ignored and disappears on the visualization view. Here is an example of tasks/ISRs sequence and timing in the custom tag, **slB**.

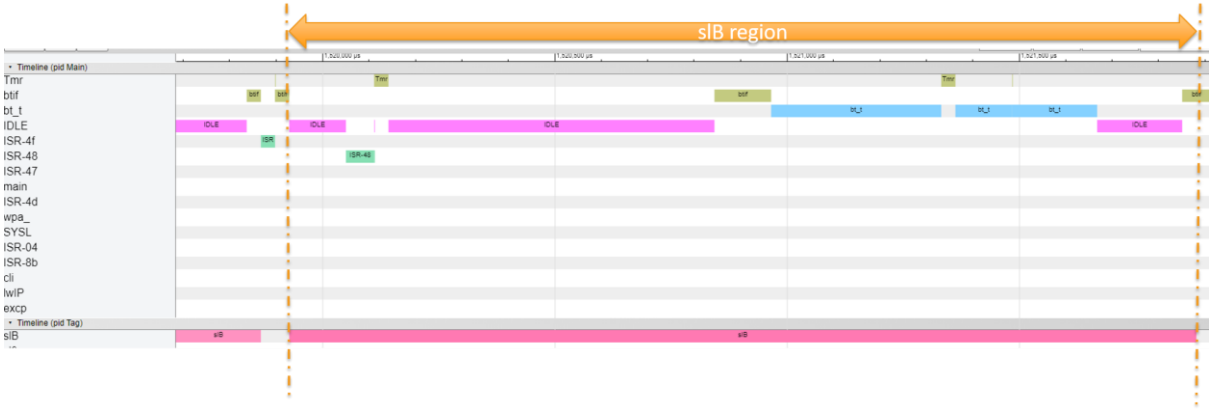

Figure 6 Example of Tasks/ISRs Sequence and Timing

```
#ifdef MTK_SWLA_ENABLE
SLA_CustomLogging("slB",
SA_START);
#endif /* MTK_SWLA_ENABLE */

//… The region user would
like to mark

#ifdef MTK_SWLA_ENABLE
SLA_CustomLogging("slB",
SA_STOP);
#endif /* MTK_SWLA_ENABLE */
```

2.1.2 Custom Labels

A special event is also created by the API, **SLA_CustomLogging.** You can easily mark an event on the visualization of system execution. Here is an example of **rs2 to rs8** custom labels.

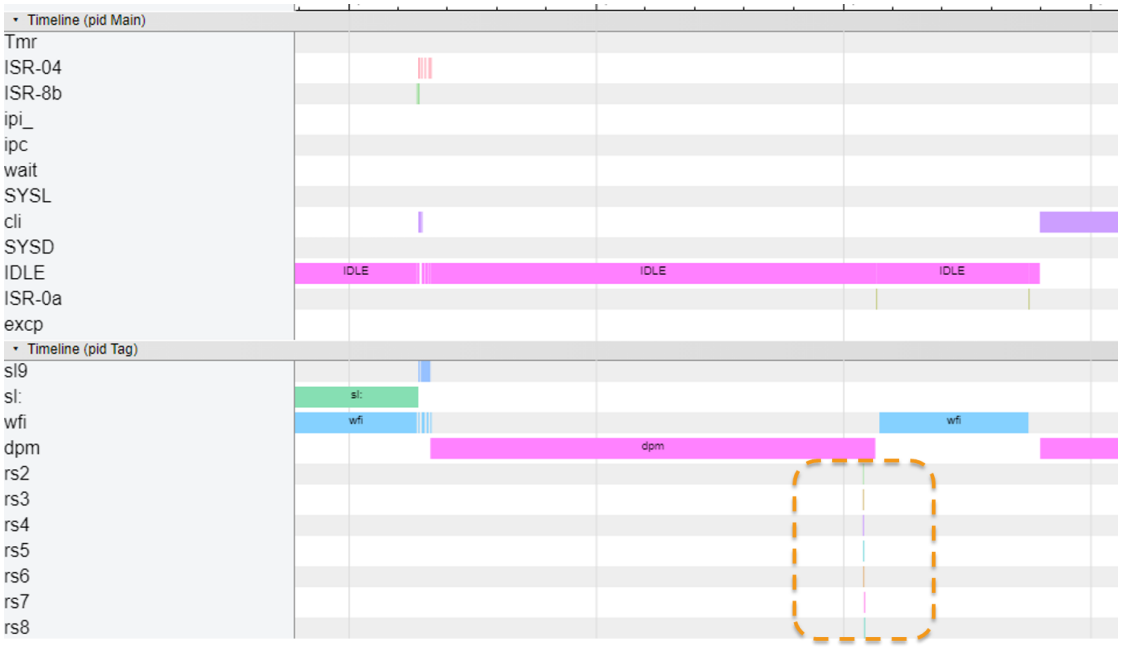

Figure 7 Example of rs2 to rs8 Custom Labels

```
char  lbl[] = "rs2";
for (i = 0; i < N; ++i) {
  #ifdef MTK_SWLA_ENABLE
  SLA_CustomLogging(lbl, SA_LABEL);
  lbl[2]++;
  #endif /* MTK_SWLA_ENABLE */
  // … The event user would like to mark
  (void) callback[i]();
}
```

3 Acquire SWLA information from flash

Please reference to **MT793X IoT SDK for System Log Developers Guide** to enable **Save Log to Flash** feature. If this feature is enabled, the latest 4KB data of SWLA will be saved to flash if exception occurs. Users can follow **MT793X IoT SDK for System Log Developers Guide** to get binary image by the flash tool and check the task/ISR execution sequence before the exception occurs.

3.1 Extract SWLA from the binary image

Please use Memory Extractor to extract the SWLA data from the binary image readback by the flash tool as shown in Figure 8 MemExtractor extract SWLA data-1, Figure 9 MemExtractor extract SWLA data-2, and Figure 10 MemExtractor extract SWLA data-3. Once the extraction is successful, users can find a file named swla_sla_mem.bin. Drag this file and drop on the **swla2chrometrace.exe** and follow the steps in the **section 1.2 How to Use SWLA** to visualize SWLA data on Chrome browser.

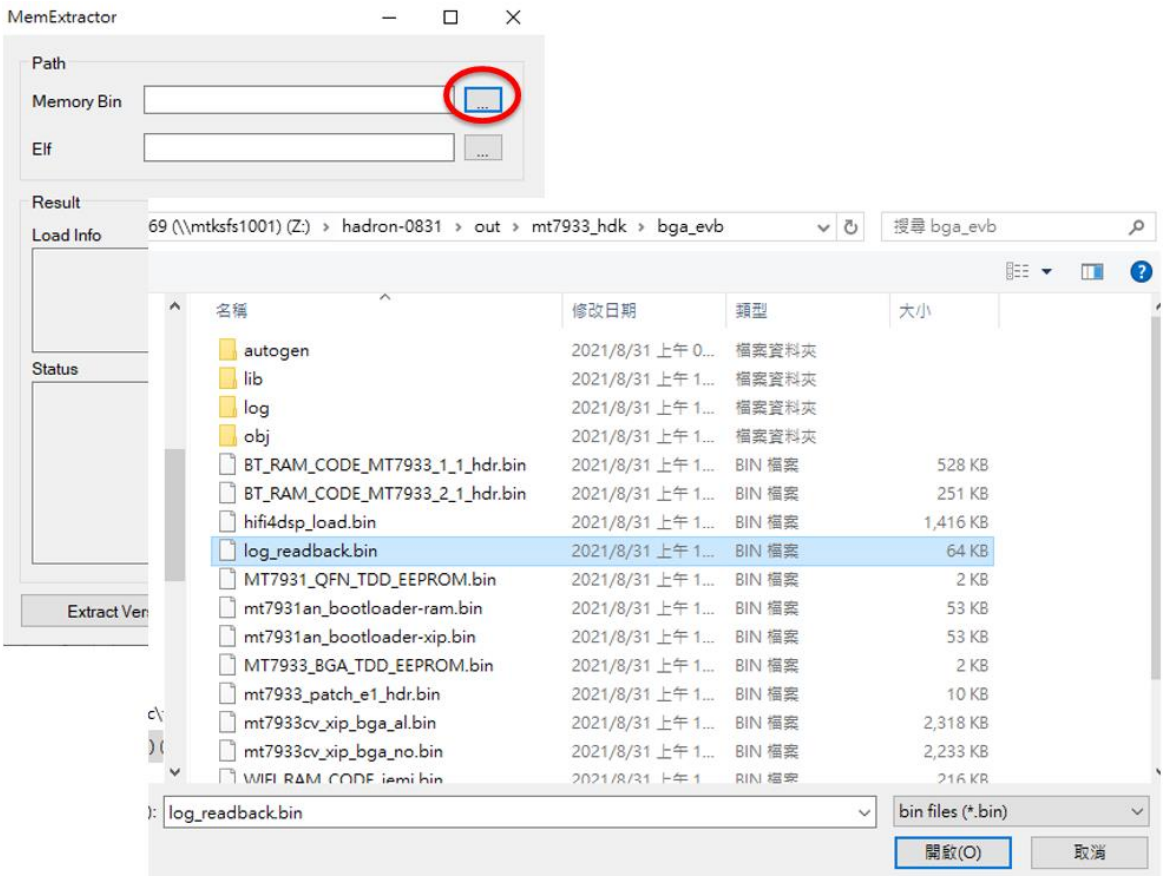

Figure 8 MemExtractor extract SWLA data-1

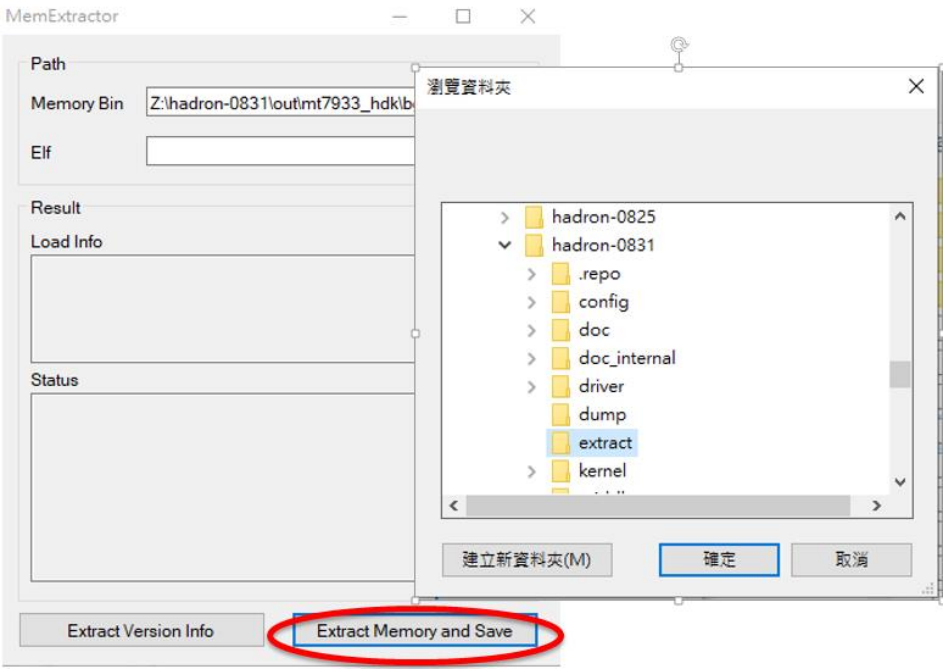

Figure 9 MemExtractor extract SWLA data-2

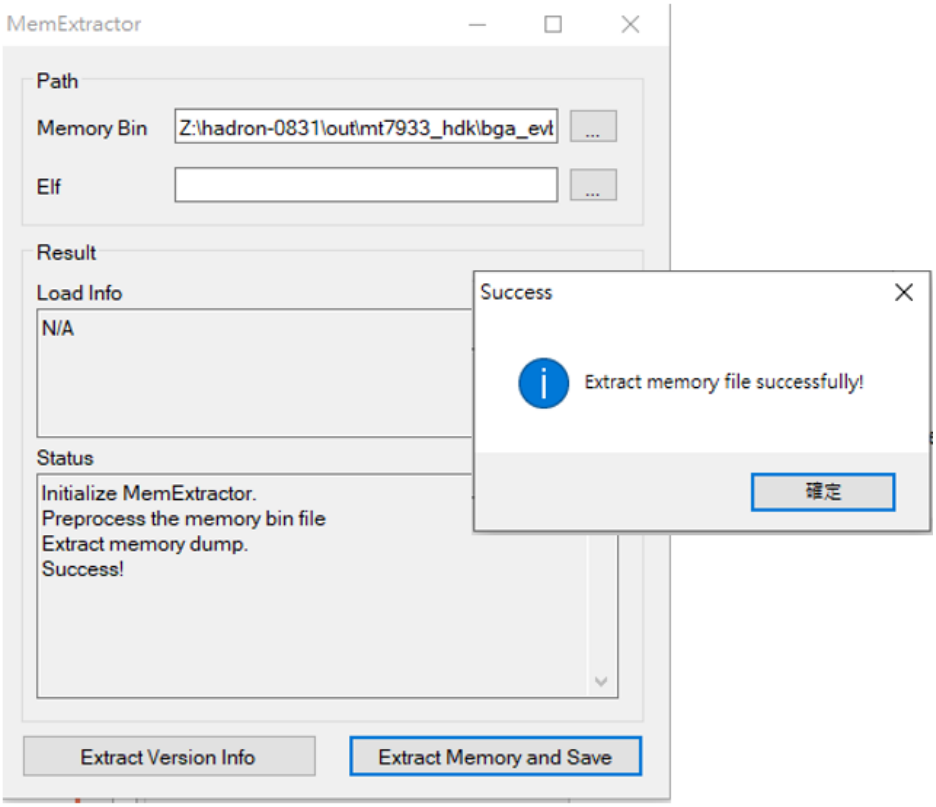

Figure 10 MemExtractor extract SWLA data-3

4 Appendix

4.1 Tips

Chrome trace UI

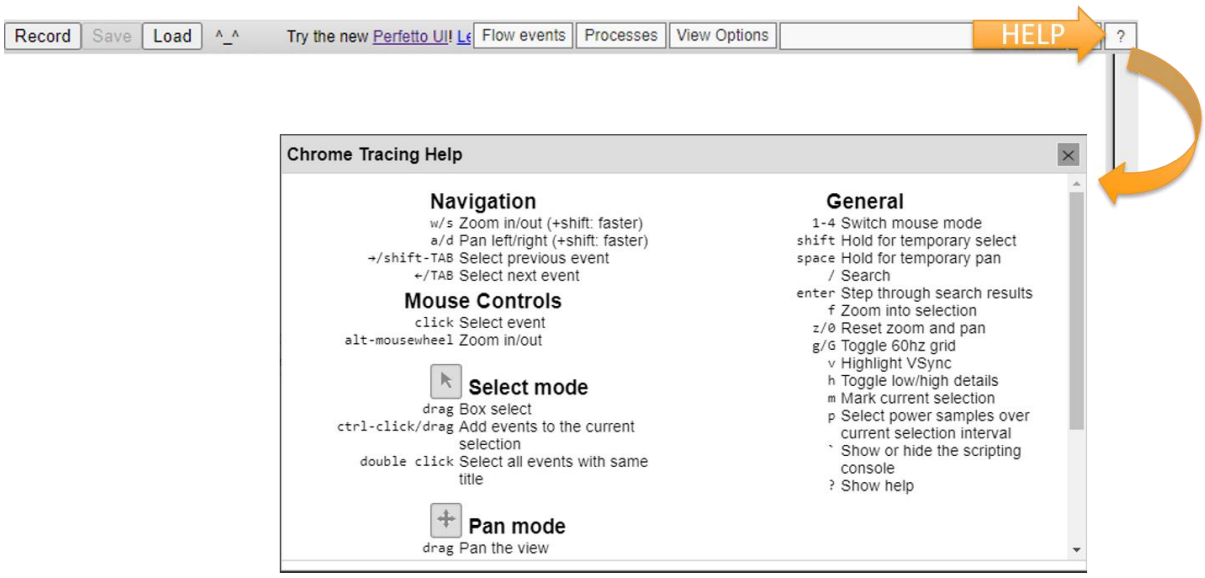

Figure 11 Chrome Trace UI Hotkeys

There are two methods to measure the interval between two tasks

- ***Ctrl + left click*** to select two execution tasks, and then click ***m*** to measure the interval.
- ***Drag-select the task/ISR/tag***, and then click ***m*** to measure the interval.

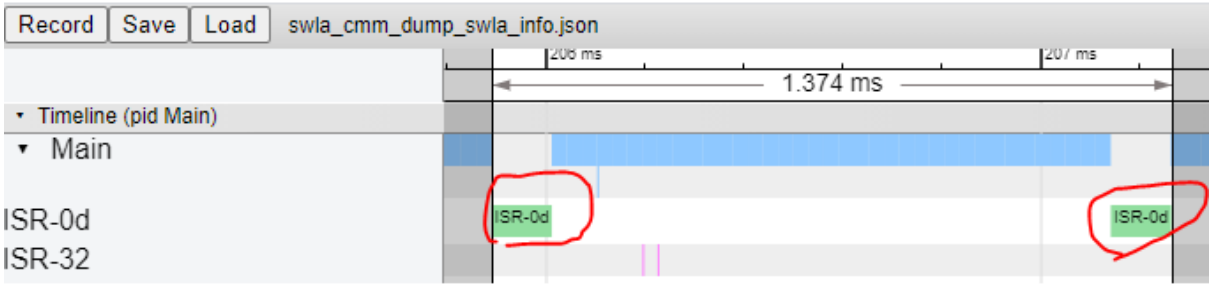

Figure 12 Measure Interval Between Tasks

4.2 FAQ

4.2.1 Timestamp Resolution

In the MT793X, the timestamp resolution of SWLA is 0.0000305175 second.

4.2.2 What Is an `excp` Task?

`**excp**` is a reserved term for labeling an exception.

4.2.3 Fail to Parse SWLA Data

Some serial terminals may cause SWLA data parsing failures. SWLA's traces should be 16-byte aligned. If the traces are not 16-byte aligned, you may encounter this problem. Please collect the exception log again.

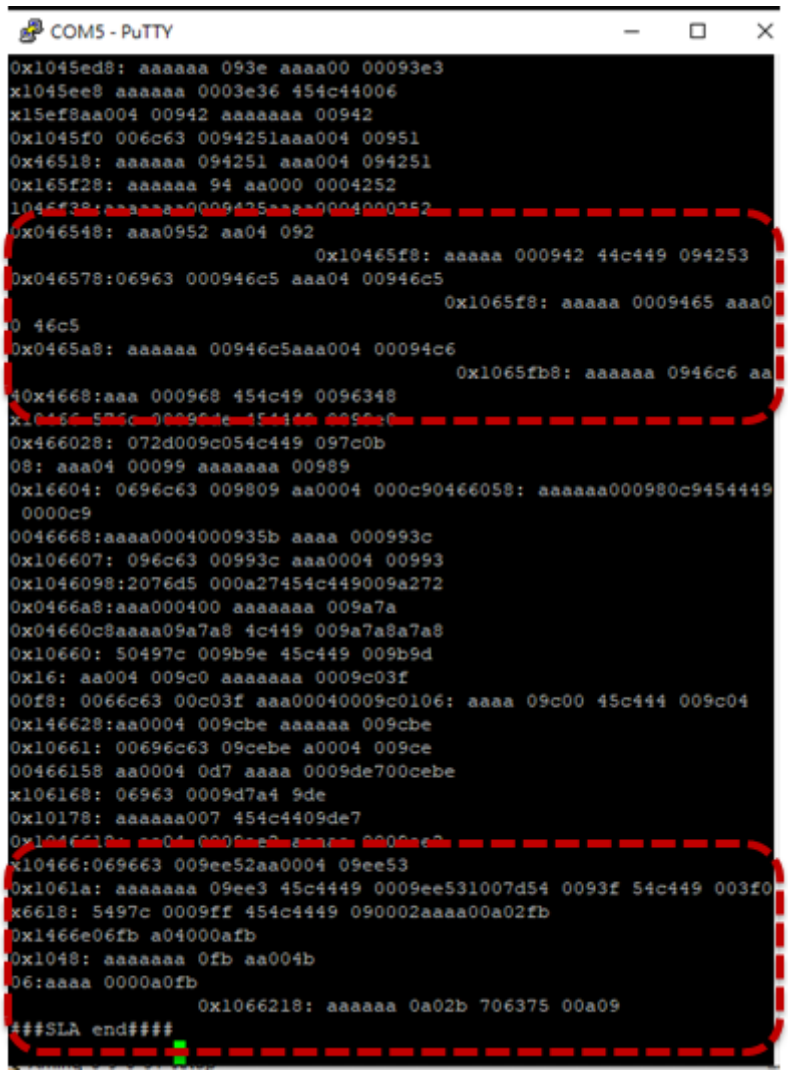

Figure 13 Example of SWLA's Trace Not 16-Byte Aligned

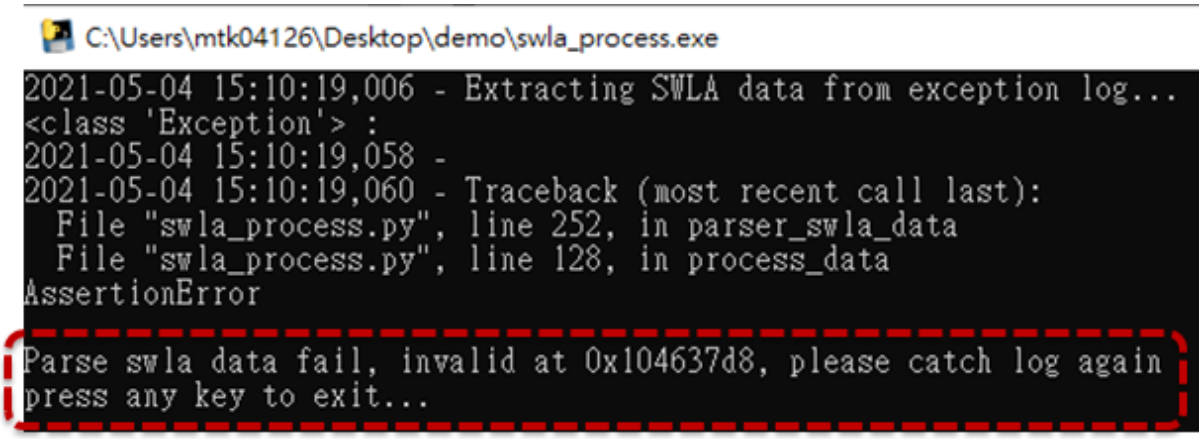

Figure 14 Example of SWLA Parsing Failure

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.