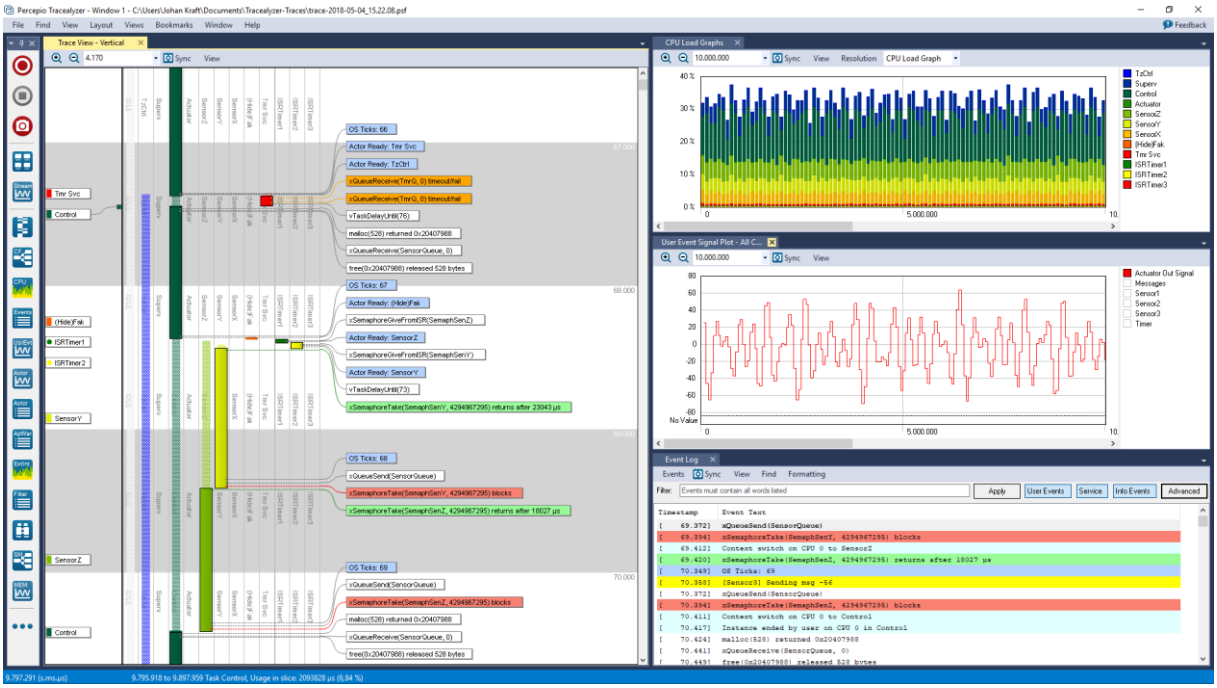
Tracealyzer streaming over Arm ITM using IAR Embedded Workbench

Application Note PA023, 2018-06-29



As of version 4.1.5, Tracealyzer supports ITM tracing via IAR Embedded Workbench and I-Jet debug probes, targeting ARM Cortex-M3, M4 and M7 MCUs and all real-time operating systems supported by Percepio's trace recorder library (v4.1 or later). These include Amazon FreeRTOS, Micrium µC/OS-III, and soon also SafeRTOS and ThreadX. This application note describes how to set up Tracealyzer together with IAR Embedded Workbench for ITM-based trace streaming using an IAR I-Jet debug probe, and also presents some performance measurements from this setup.

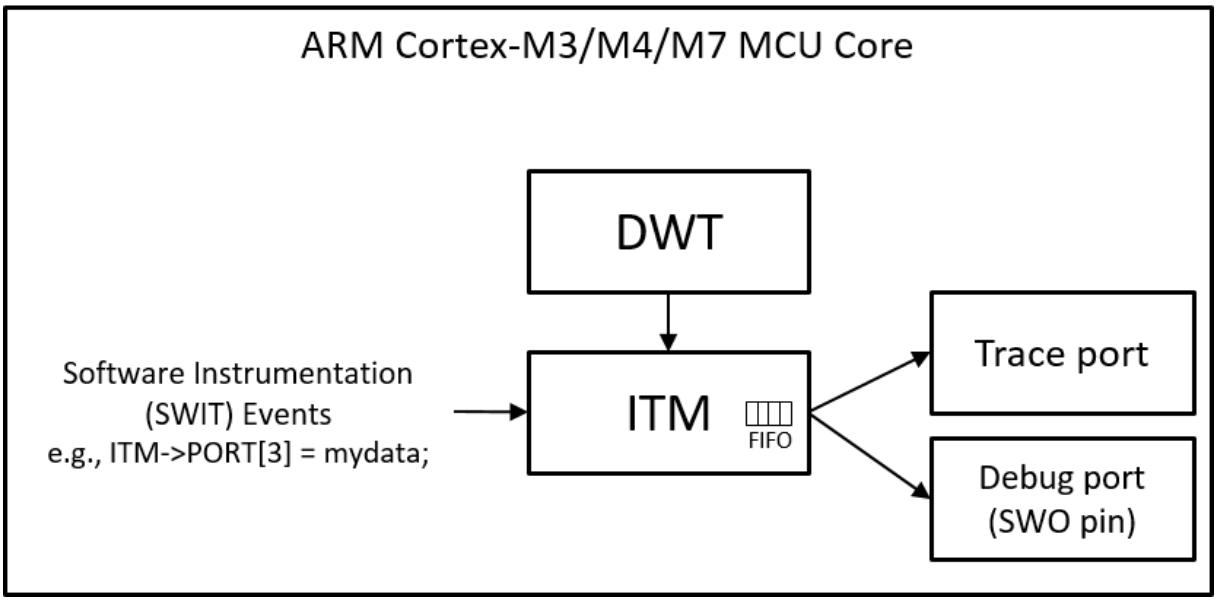
The ITM interface (short for Instrumentation Trace Macrocell) allows the Tracealyzer recorder library to transmit software-generated trace data via the debug probe, either using the SWO pin (Serial Wire Output) which is supported by the regular I-Jet model, or over the ETM trace port if you have an IAR I-Jet Trace. However, SWO on a regular I-Jet gives more than sufficient speed.

With a fast debug probe like IAR I-Jet, ITM allows for high data rates and the target-side overhead is minimal. Unlike some other streaming solutions supported by Tracealyzer, with ITM streaming the data is written directly instead of first stored in a temporary RAM buffer. This means that the RAM usage of the recorder library is much lower in this setup.

We measured the performance of our setup in an experiment and the results are pretty impressive. In this experiment, we used a Renesas S7G2 with an Arm Cortex-M7 core running at

240 MHz, IAR Embedded Workbench v7.71.1 (Renesas Synergy version) and an IAR I-Jet configured at 24 MHz SWO frequency.

With this setup and the compiler optimization level set to “high”, sending 16 bytes of data (4 writes of 32 bit each) took only 36 clock cycles (150 ns). With the compiler optimization level set to “low” the write time was a bit longer, 88 clock cycles, but this is still quite fast (367 ns).



Achieving so fast write times however requires that the ITM FIFO has room for the data, otherwise the write operation will block while waiting for the previous data to be transmitted. We measured this by writing the 16-byte bursts without any delay and the write time then increased to 3 266 clock cycles (13,6 µs) due to this blocking. Still, in this extreme case we managed to transmit over 73 400 events per second of 16 byte each, resulting in a throughput of 1 175 KB/s. This is many times higher data rate than typically needed for RTOS tracing with Tracealyzer.

However, this was a stress test. In real applications where the events are naturally separated in time by application code, there is typically no such blocking. This since the trace port typically has time to flush the ITM FIFO between the writes. However, this assumes that events are separated by a minimum time, corresponding to the ITM blocking time. To check this, we added a delay of about 3300 clock cycles in between the 16-byte bursts to simulate application code running between traced events. In this case, the write times stayed consistently at 36 resp. 88 clock cycles, depending on optimization level.

Note that the bottleneck here is the trace port, i.e. the 24 MHz SWO frequency. This is not affected by the compiler optimization level or the relatively high clock frequency of the Renesas S7G2 MCU. Similar throughput is possible also on slower MCUs by lowering the SWO prescaler.

Requirements

To follow the instructions in this document, you need the following:

- **Target system:** Any Arm Cortex-M processor with ITM support and a trace-enabled debug connector. A full ETM trace port can be used but is not required. SWO is used in this guide.
    - ITM is supported by practically all Arm Cortex-M3, M4 or M7 MCUs, and is expected to be available on future high-end Cortex-M MCUs (e.g. M33).
    - ITM is not supported by ARM Cortex-M0, M0+ or M23 MCUs.
- **RTOS:** An RTOS supported by Percepio's trace recorder library, such as
    - FreeRTOS (v7.3 or newer)
    - Micrium µC/OS-III (v3.04 or newer)
- **Development tools**
    - IAR Embedded Workbench for ARM/Renesas Synergy
    - IAR I-Jet or I-Jet Trace
    - Percepio Tracealyzer v4.1.5 or later, with a license matching your RTOS.

Target-side setup

The trace recorder library needed by Tracealyzer is included with the application and found in the application directory, under "Tracealyzer 4/<RTOS Name>/TraceRecorder". To integrate this in your IAR project in streaming mode, follow the getting started guide provided in the Tracealyzer User Manual (see "Integrating the recorder" and "Setting up streaming").

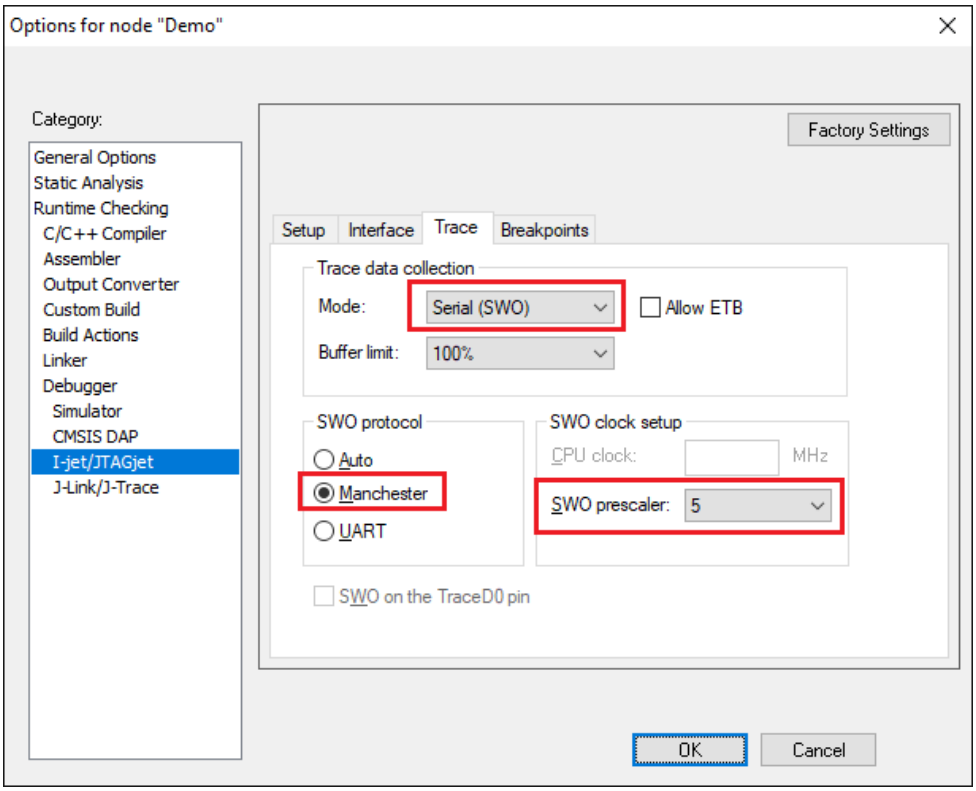
Make sure to include the "stream port" found in "TraceRecorder/streamports/ARM_ITM". Note that is a generic solution for ITM tracing that works with any IDE and debug probe, assuming they have Trace/SWO support and an ability to output the ITM data to a binary file.

When calling vTraceEnable to initialize the recorder, use the parameter "TRC_START" which starts the tracing directly. It is not yet possible to control the recording from the Tracealyzer application, as there is only one-way communication when using ITM. However, since you can trace over practically unlimited periods in the streaming mode, you can record your entire debug sessions.

Host-side setup – IAR Embedded Workbench

In IAR Embedded Workbench, start by opening Options… for your project and check the following settings:

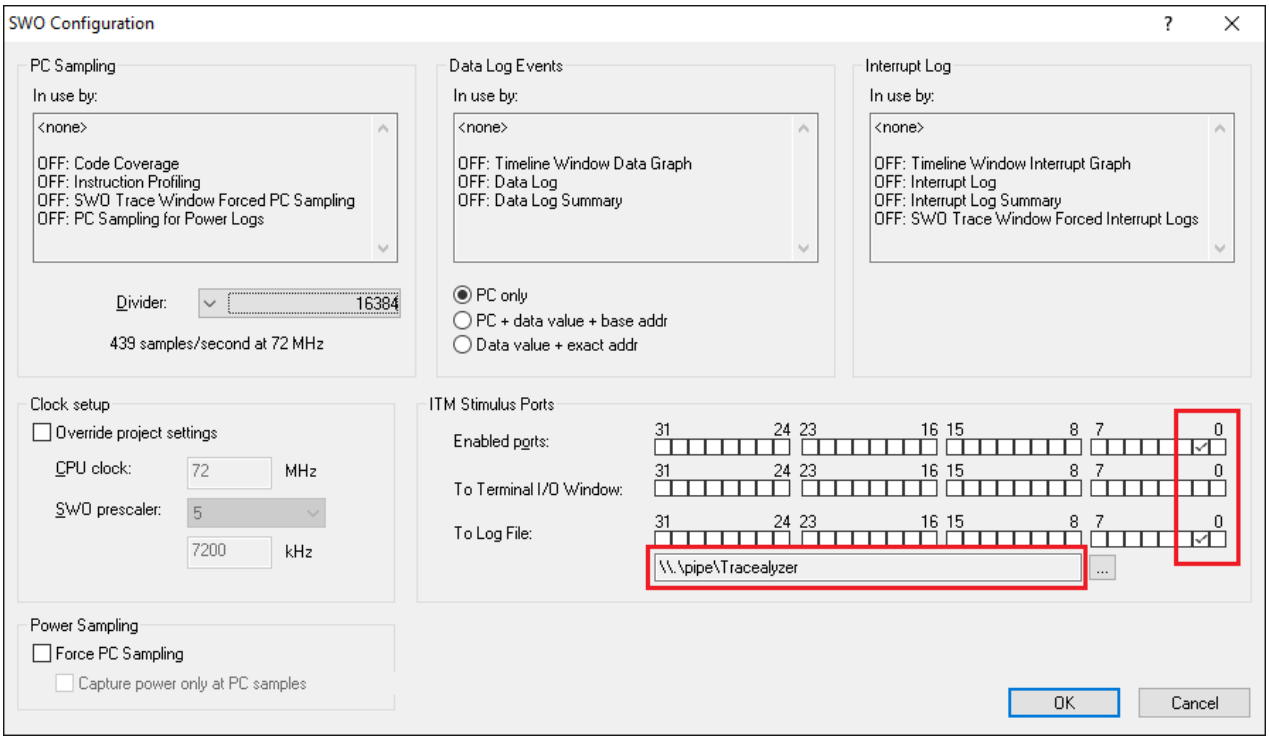
- Debugger -> Setup -> Driver: I-Jet/JTAGjet
- Debugger -> I-Jet/JTAGjet -> Interface: SWD
- Debugger -> I-Jet/JTAGjet -> Trace -> Trace data collection -> Mode: SWO
- Debugger -> I-Jet/JTAGjet -> Trace -> SWO protocol: Manchester
- Debugger -> I-Jet/JTAGjet -> Trace -> SWO clock setup -> SWO prescaler: (see below)



In my case, the "auto" setting for the SWO prescaler set a prescaler of 4, which resulted in a slightly too high SWO frequency as I got occasional errors in the data transfer, shown in Tracealyzer as Dropped events (indicated in red).

But if increasing the prescaler to 5 instead of auto (4), it worked fine and the data transfer is still really fast (24 MHz instead of 30 MHz). But I recommend that you start with a slightly higher value to ensure stable transfer, perhaps 6-8, then try decreasing the prescaler to improve performance.

Now, start a debug session to enable the SWO Configuration dialog, found under the I-Jet menu, and open this dialog.



Under *ITM Stimulus Ports*, check second checkbox from the right to enable Port 1. If you want to change the ITM port used, you need to update this setting also in the trace recorder library (see TRC_CFG_ITM_PORT, defined in trcStreamingPort.h).

To make IAR Embedded Workbench output the ITM data to Tracealyzer, enable the “To Log File” option for this ITM port. As “log file” we use a *Named Pipe*, a mechanism for inter-process communication in Windows, which is created and read by Tracealyzer. A named pipe can be accessed like a regular file by using a path like “\\.\pipe\MyPipeName”.

To specify a named pipe in “To Log File”, follow these three steps:

1. Make sure IAR Embedded Workbench is closed.
2. In the “settings” directory of your IAR v8.x project, locate the file named <projectname>.dnx. For IAR v7.x projects, the file is called <projectname>.dni.
3. Save a backup copy of the .dni or .dnx file and then open the original file in a text editor.
4. Locate the “ItmLogFile” entry and change its value to the full path of the named pipe (e.g. \\.\pipe\Tracealyzer), like shown below.
5. Save the file and start IAR Embedded Workbench again.
6. Open SWO Configuration and verify that new path is shown under “To Log File”.

The relevant entry in the settings file looks like this:

```
<SWOManager>
    <SamplingDivider>8192</SamplingDivider>
    <OverrideClock>1</OverrideClock>
    <CpuClock>168000000</CpuClock>
    <SwoClock>0</SwoClock>
    <DataLogMode>0</DataLogMode>
    <ItmPortsEnabled>63</ItmPortsEnabled>
    <ItmTermIOPorts>1</ItmTermIOPorts>
    <ItmLogPorts>2</ItmLogPorts>
    <ItmLogFile>\\.\pipe\Tracealyzer</ItmLogFile>
    <PowerForcePC>0</PowerForcePC>
    <PowerConnectPC>1</PowerConnectPC>
</SWOManager>
```

<project>.dnx, if using v8.x or later

```
ItmPortsEnabled=2
ItmTermIOPorts=0
ItmLogPorts=0
ItmLogFile=\\.\pipe\Tracealyzer
PowerForcePC=0
PowerConnectPC=0
```

<project>.dni, if using v7.x or earlier

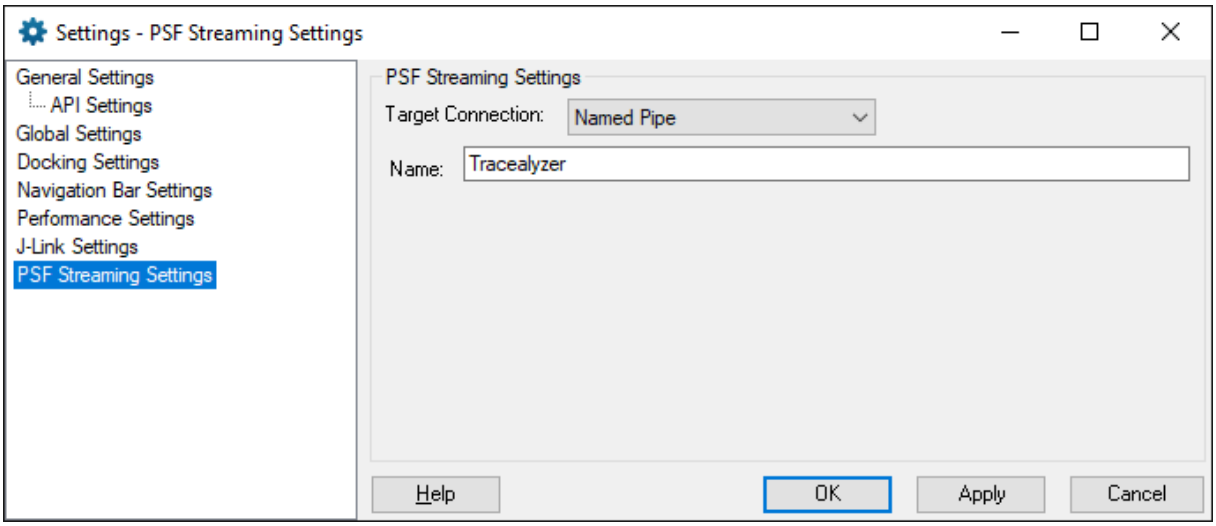
Note this procedure is only required once per project, as you can easily disable the “To Log File” option using the checkbox, but the path remains unchanged.

IAR Embedded Workbench should now be correctly configured for Tracealyzer streaming.

Host-side Setup - Tracealyzer

Streaming data from a named pipe requires Tracealyzer v4.1.5 or later, so if you have an older version you need to update it. To see your Tracealyzer version, select Help -> About.

To configure Tracealyzer to read from the named pipe, select File -> Settings, or use the Recording Settings link on the welcome screen. Select PSF Streaming Settings and change the “Target Connection” setting to “Named Pipe”, like shown below.



Enter the same pipe name as specified in the IAR settings for ItmLogFile. However. you don’t need to provide the full path, the name of the pipe is sufficient (e.g. “Tracealyzer”). Save the settings.

To record a trace, begin in Tracealyzer by selecting Start Recording (the big red button in the main navigation bar). Tracealyzer has now created the named pipe and is listening for data.

Next, start a debug session in IAR Embedded Workbench and run the application. Tracealyzer should now receive and display the data live. Stop the recording to enable all 30+ views in Tracealyzer.

Note that you need to start the Tracealyzer recording *before* you start your IAR debug session, otherwise the named pipe won't be available when IAR Embedded Workbench attempts to write the data.

During tracing, you may stop on break points, single-step and resume execution without interfering with the resulting trace. This since the event timestamps are set based on the target system cycle counter, so halting the system "stops the time" for the recorder library.

You don't need to have the Event Log or SWO Trace views open in IAR Embedded Workbench, the data is forwarded to Tracealyzer anyway, as long as a debug session is running with the To Event Log option enabled´.

Troubleshooting

If Tracealyzer doesn't show any data, you may check the following:

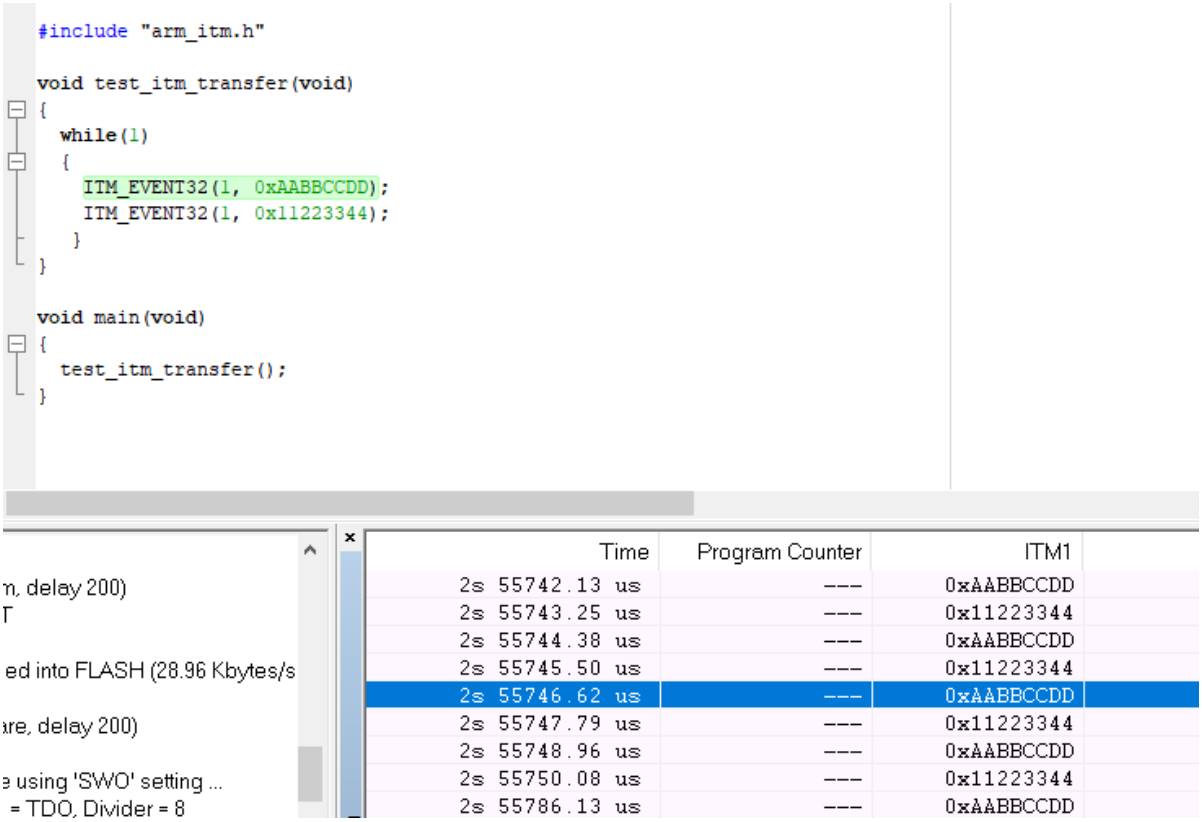
Step 1. To ensure that the trace recorder library has been properly integrated in your target system, put a breakpoint in itm_write(), found in trcStreamingPort.c, and verify that it is called from the trace macros inside the RTOS functions. If not, consult the general troubleshooting guide in the Tracealyzer User Manual.

Step 2. To ensure the data is transferred correctly between the target system and IAR Embedded Workbench, make a simple test program like shown in the screenshot below. This writes a known byte pattern to the ITM port, so you can check the integrity of the data. Use the function ITM_EVENT32 (defined in "arm_itm.h"), where the first parameter is the ITM port.

Start a debug session and open the Event Log in IAR Embedded Workbench, found in the I-Jet menu. Then run the test program for a while and stop. Check carefully that you get the right data, without any random anomalies. If the data looks good, you can try decreasing the prescaler and repeat the analysis. If you get errors in the data, increase the prescaler and try again.

Step 3. If you don't get any data at all, check the debug cable between the IAR I-Jet and the debug connector on the board. Make sure the cable is in good shape and that the connectors are fully inserted.

Step 4. Double-check the previous steps of this guide, if you have not done so already.



Step 5. Try setting the CPU clock manually in SWO Configuration, under Clock Setup. The clock frequency of your target system can be found by inspecting the global variable SystemCoreClock, once the low-level setup has completed. This variable is defined in Arm's CMSIS library, which is often available by default.

Learning more

To learn more about Tracealyzer and ARM ITM, we recommend the following resources:

- Downloading Percepio Tracealyzer: https://percepio.com/download (registration required)
- Tracealyzer User Manual (Help -> User Manual)
- Tracealyzer "Getting Started" portal - https://percepio.com/gettingstarted
- About ITM trace - https://percepio.com/2016/06/09/arm-itm/

For technical support and other questions, please contact support@percepio.com.