SourcePoint Intel

Getting Started Guide for the

AAEON UP Xtreme i11

Revision 1.00

Contents

Introduction

ASSET InterTech, Inc. reserves the right to make changes to its products or to discontinue any product or service without notice, and advises its customer to obtain the latest version of relevant information to verify, before placing orders, that the information being relied on is current.

ASSET InterTech, Inc. assumes no liability for applications assistance, customer product design, software performance, or infringement of patents or services described herein. Nor does ASSET InterTech warrant or represent that any license, either express or implied, is granted under any patent right, copyright, or other intellectual property right of ASSET InterTech covering or relating to any combination, machine, or process in which such products or services might be or are used.

While every precaution has been taken in the preparation of this document, ASSET InterTech assumes no responsibility for errors or omissions. Neither is any liability assumed for damages resulting from the use of the information contained herein.

Contacting ASSET InterTech:
Phone: (888) 694-6250 toll free in the U.S.
+1 (949) 517-4150 outside the U.S.

For Sales contact emails and telephone numbers:
https://www.asset-intertech.com/contact-us/

For Support contact emails and telephone numbers:
https://www.asset-intertech.com/support/

Revision History

Revision Number	Description	Date
1.00	Original document	November 28, 2021

Welcome!

Thank you very much for your SourcePoint Intel Individual or Institutional License purchase! We appreciate you acquiring our best-in-class debugger, and hope you enjoy using it. We strive to deliver the most powerful, easy-to-use and polished product as possible. So, please feel free to share your feedback directly at our contact information above, or via your favorite social media outlet.

As with any new tool, mastering SourcePoint takes an investment in terms of time and effort. JTAG-based debug is a fairly specialized area, and low-level “on the metal” firmware development on x86 platforms is even more so. So, in your use of the tool, you may encounter behavior that seems non-intuitive or even wrong. You may be encountering a tool corner case, a limitation inherent in JTAG or DCI, or even a bug. If so, try a few different options as may be referenced in the Troubleshooting section of this Guide, and if it persists, give us a call. We are here to serve.

Which Board and Cables to Purchase?

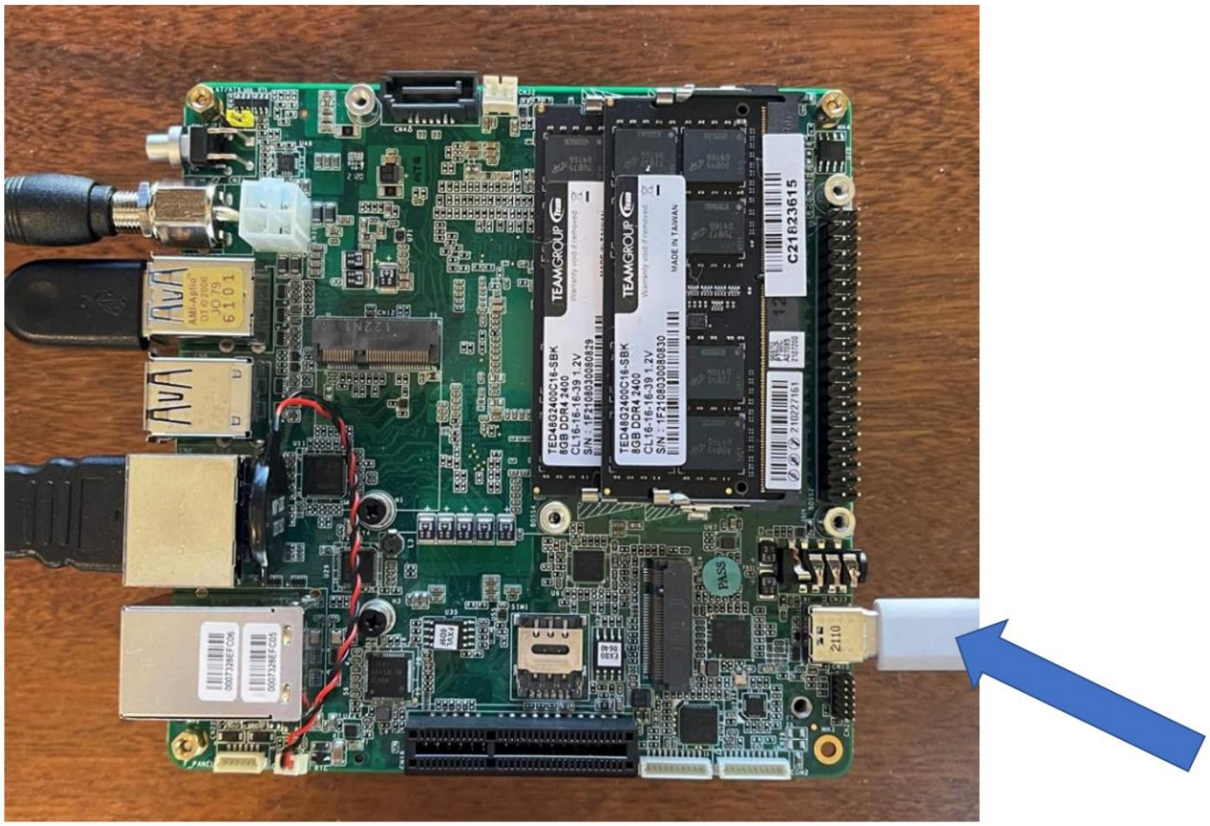
The board supported by your SourcePoint Individual or Institutional license is the AAEON UP Xtreme i11 board. It comes in three flavors: i3, i5 and i7. All three flavors are supported by your license. The Celeron version of the board is not supported by your license; it may work (we haven't tried it) but we recommend the i3, i5 or i7 to get the most value out of your purchase. This is a Tiger Lake (TGL) UP3 CPU that supports all the latest Intel debug and trace features such as Intel Processor Trace (Intel PT), Intel Trace Hub (ITH), Architectural Event Trace (AET), and others.

WARNING:

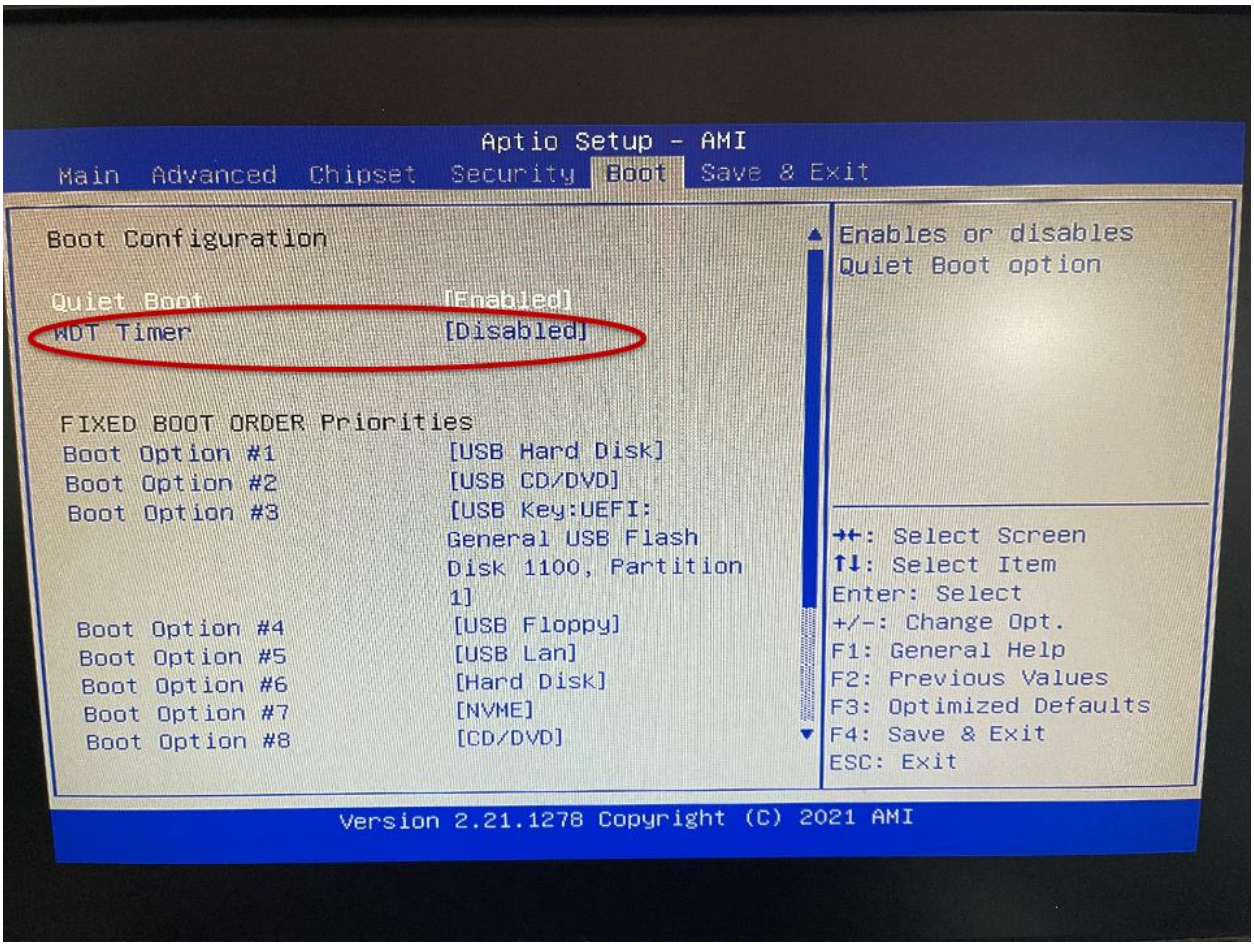
Do ***NOT*** plug a regular USB cable into the target and attempt to use DCI. Specialty cables, with VBUS snipped, are required; using a regular USB cable may possibly fry your target, or worse. Purchase a DCI Type-A/C cable from designintools.intel.com or mouser.com. This target has its Type-C port enabled for DCI. If you have a debug host with a Type-A port, purchase the part # ITPDCIAMCM1MU (1.0 meter) cable. The longer 1.8 meter ITPDCIAMCM2MU will work as well. If your host has a Type-C port, purchase the ITPDCIC2CD2U1M. Using Type A/C hubs have been seen to work, but are not warranted.Type A/C adapters have been seen not to work.

BIOS Settings

The AAEON UP Xtreme i11 board, as of November 19, 2021, comes equipped with an AMI Aptio BIOS that is based upon typical Intel Customer Reference Board (CRB) BIOS. As of the time of this writing, this is Release 1.3, and can be found as a standalone image here: https://downloads.up-community.org/download/up-xtreme-i11-uefi-bios-v1-3/. Luckily, the platform comes with all the necessary hardware hooks and firmware straps to support Intel Direct Connect Interface (DCI) out of the box. The USB Type C port on the board is the port of interest:



There is only one BIOS setting change that must be made: to disable the board's Watchdog Timer (WDT). Go into the BIOS Boot menu, and set WDT Timer -> Disabled. If you don't, run-control will be successful, but the board will power-cycle every 30 seconds; putting a real crimp in your debugging!

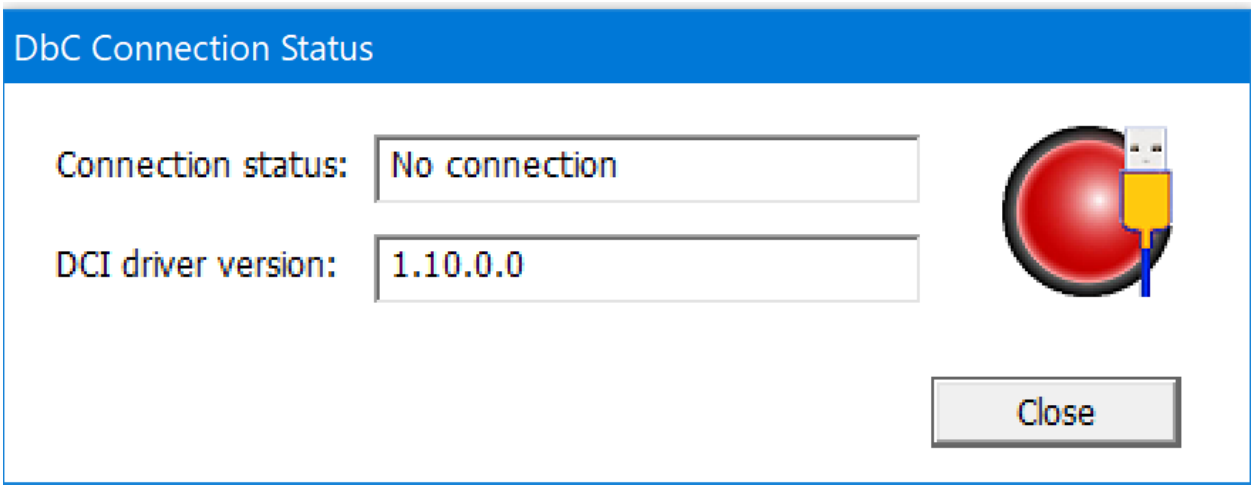


And that's all that's needed. All the Debug Settings in the CRB Advanced menu can be safely ignored. You are now ready to test your connection, and then launch SourcePoint and begin debugging.

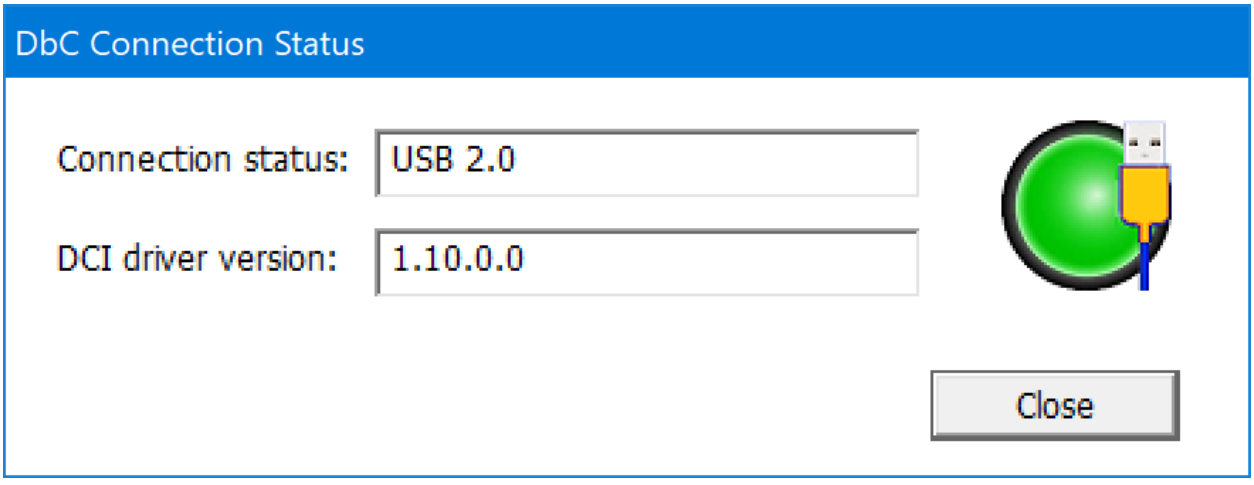
DbCStatus.exe: The "Green Ball" indicates a Good Connection

Luckily, there is a convenient application in the SourcePoint install directory that will tell you that the DCI driver is successfully installed on your computer, and it is possible to make a connection between SourcePoint and the target.

Navigate to `C:\Program Files (x86)\Arium\SourcePoint 7.12.XX` (where XX is your current SourcePoint release), and launch the `DbCStatus.exe`. You should see the red ball, indicating that there is no connection:



Ensure that the Type-C cable is firmly connected to both the host and target, power up the UP Xtreme i11. You should hear two beeps, and in a moment the ball should turn green:

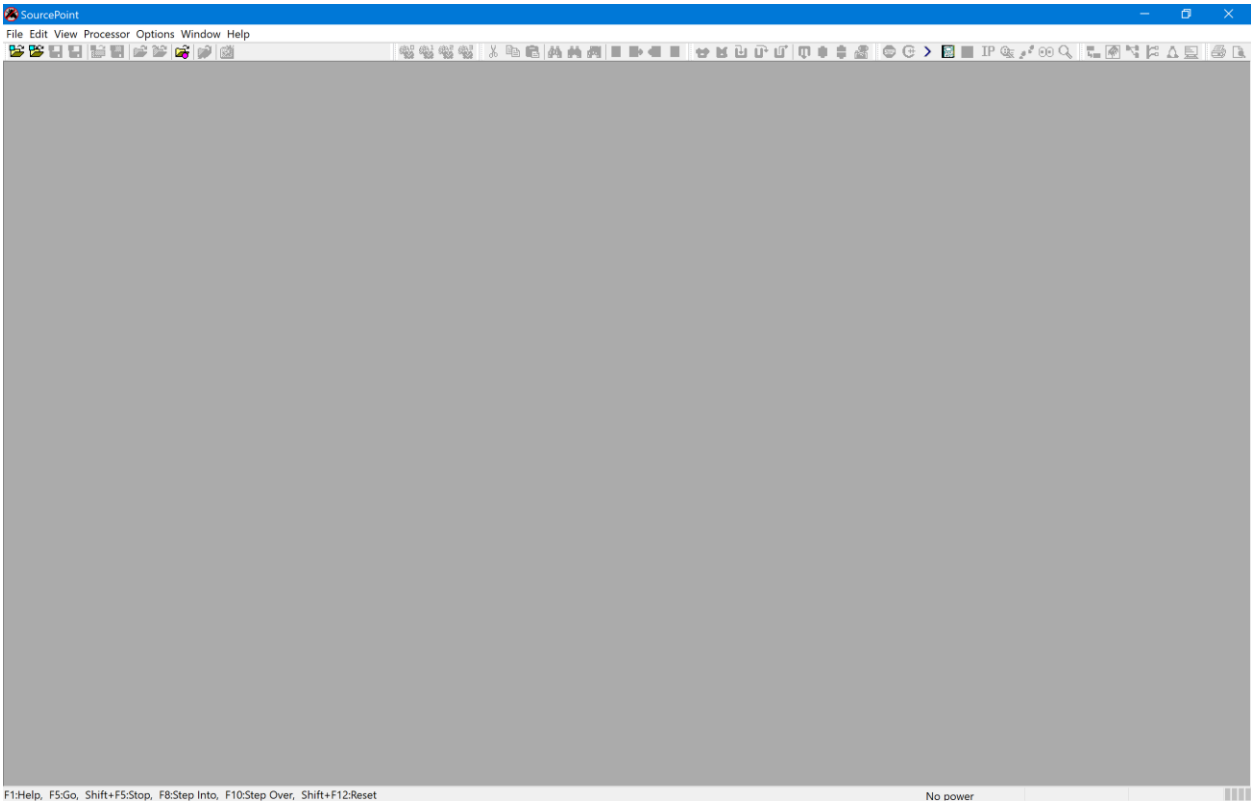


Let the platform boot to the UEFI shell. Congratulations! You have a working DCI connection. It's smooth sailing from here.

Is the ball still red? Try cycling power on the target (don't just power cycle with the reset button; physically remove the power plug, then plug it back in again). Still red? See our Troubleshooting section.

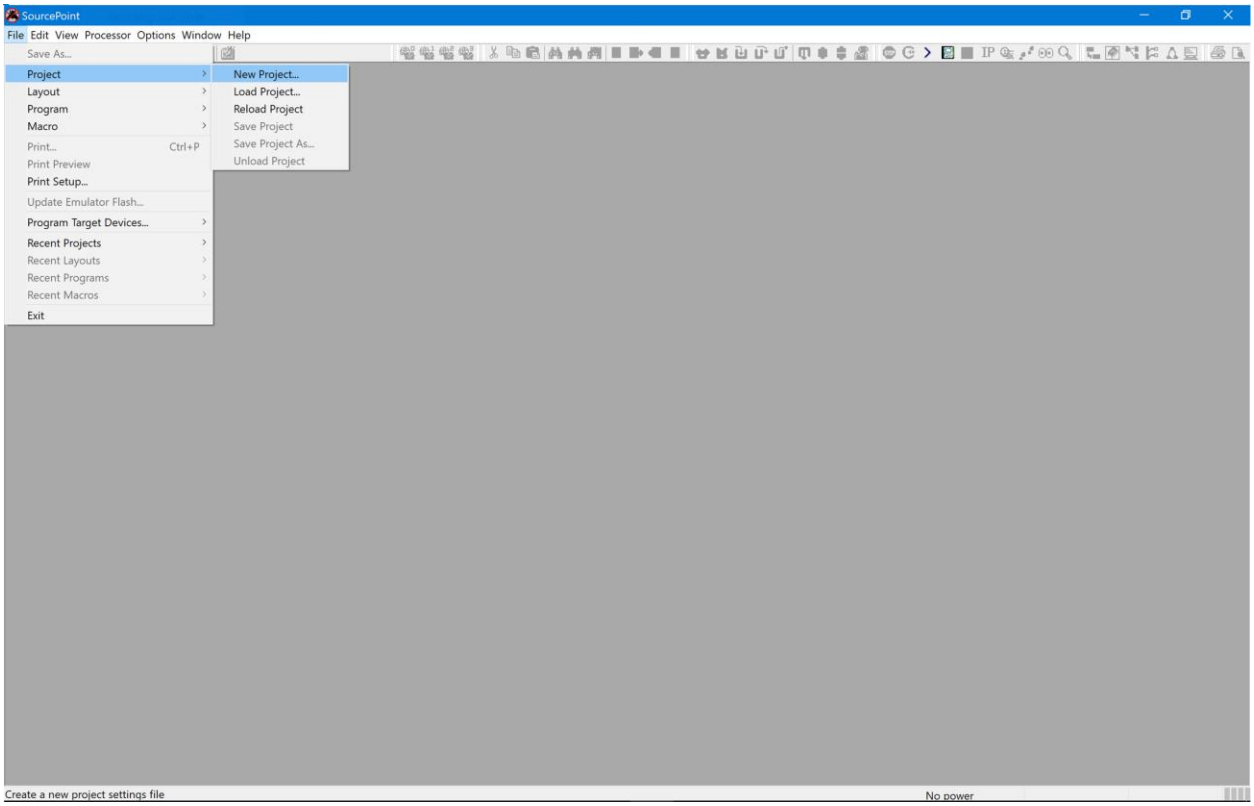
Getting Started with SourcePoint

When you launch SourcePoint for the first time, you will see the main screen, mostly grey:

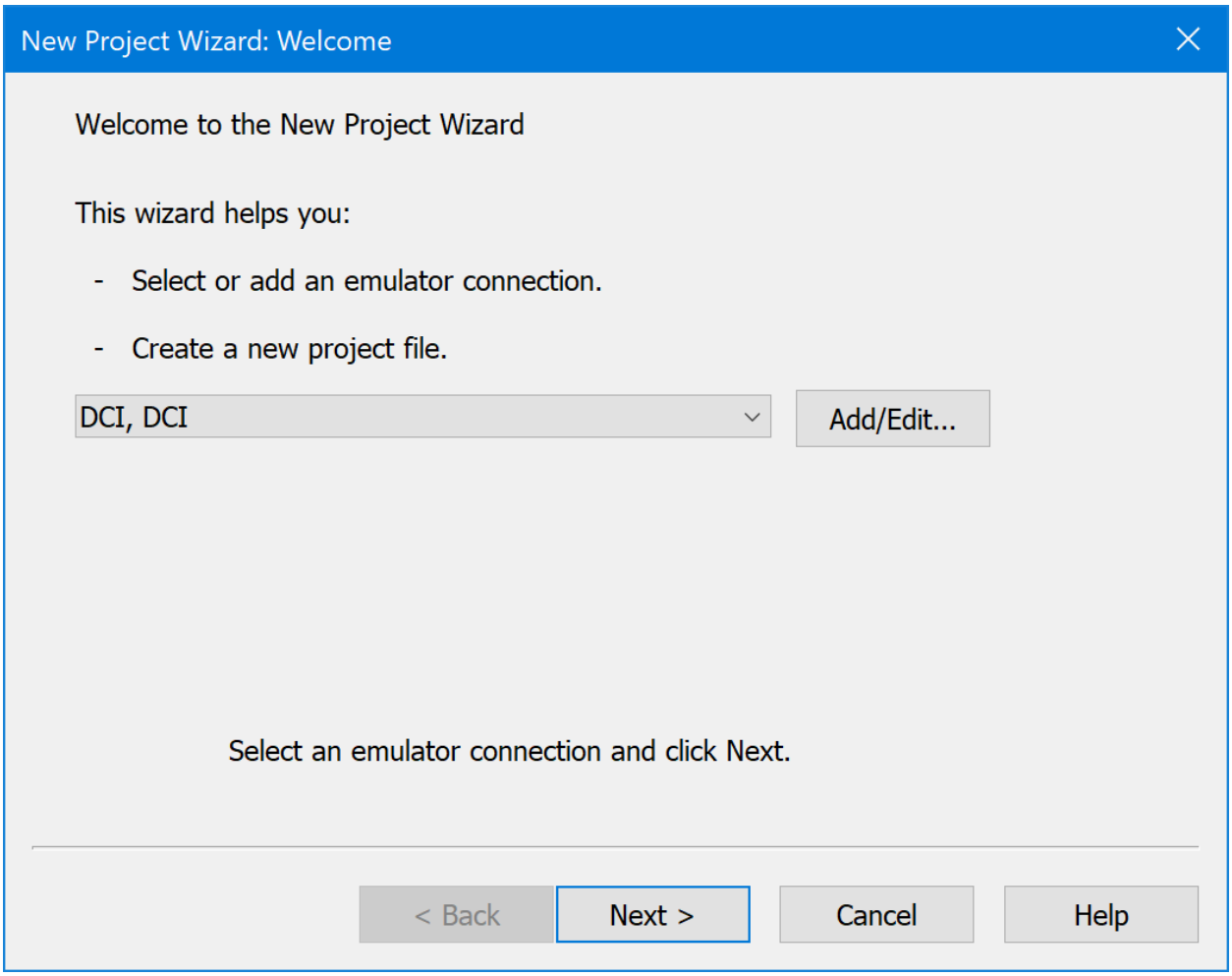


SourcePoint uses Projects (files with suffix .prj) as containers for your debugging session. You can create as many Projects as you want, with all your own preferences saved. Often, once you have the SourcePoint Project configured to your liking, you'll save it and use it repeatedly during your separate debugging sessions. Other users may wish to save a separate Project for each separate debugging session. That's really a matter of user preference and what you're debugging – it's your choice.

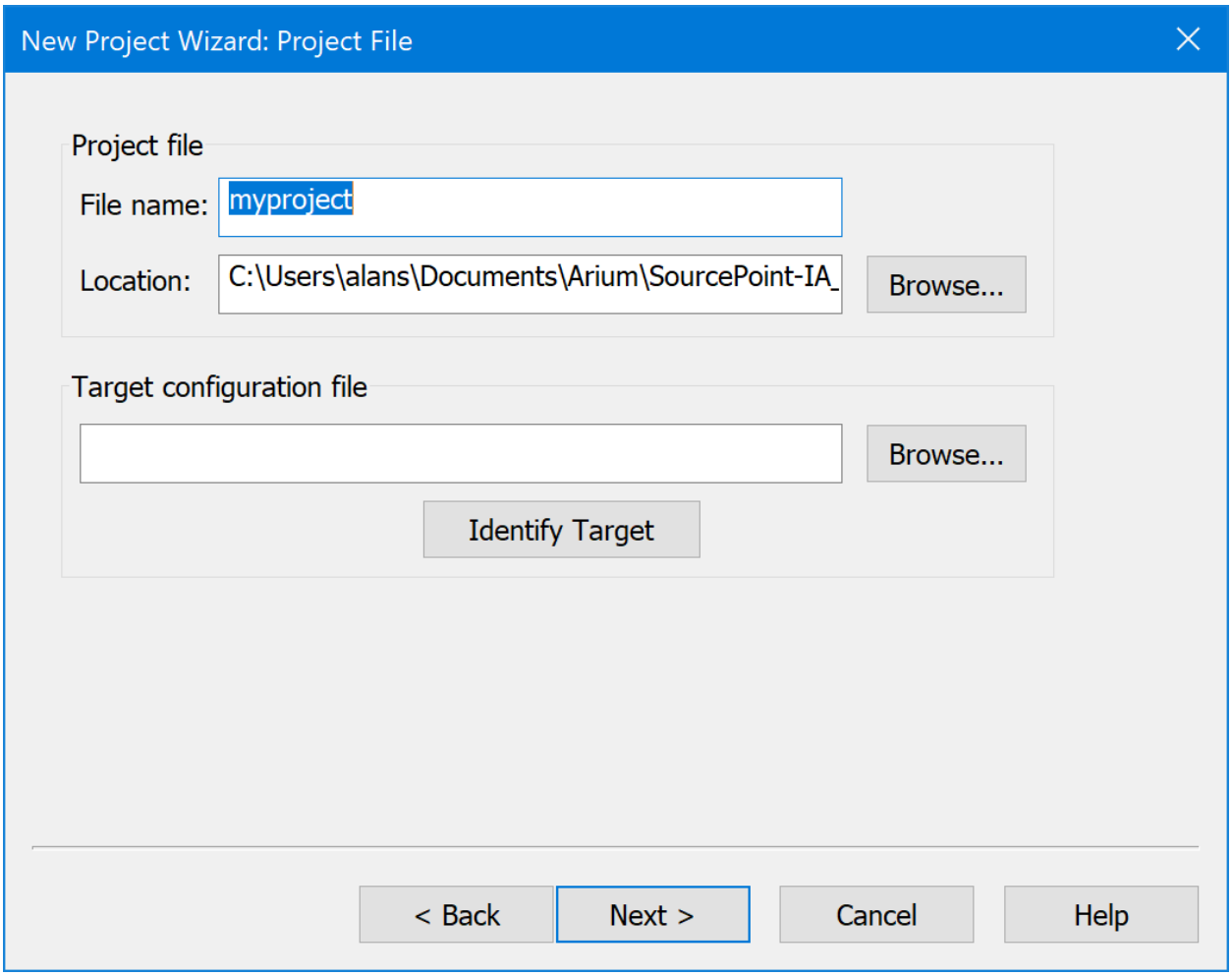
Now it's time to create the Project. Under `File > Project…` select `New Project`:



You'll be presented by the New Project Wizard (NPW). The emulator connection should be via DCI:

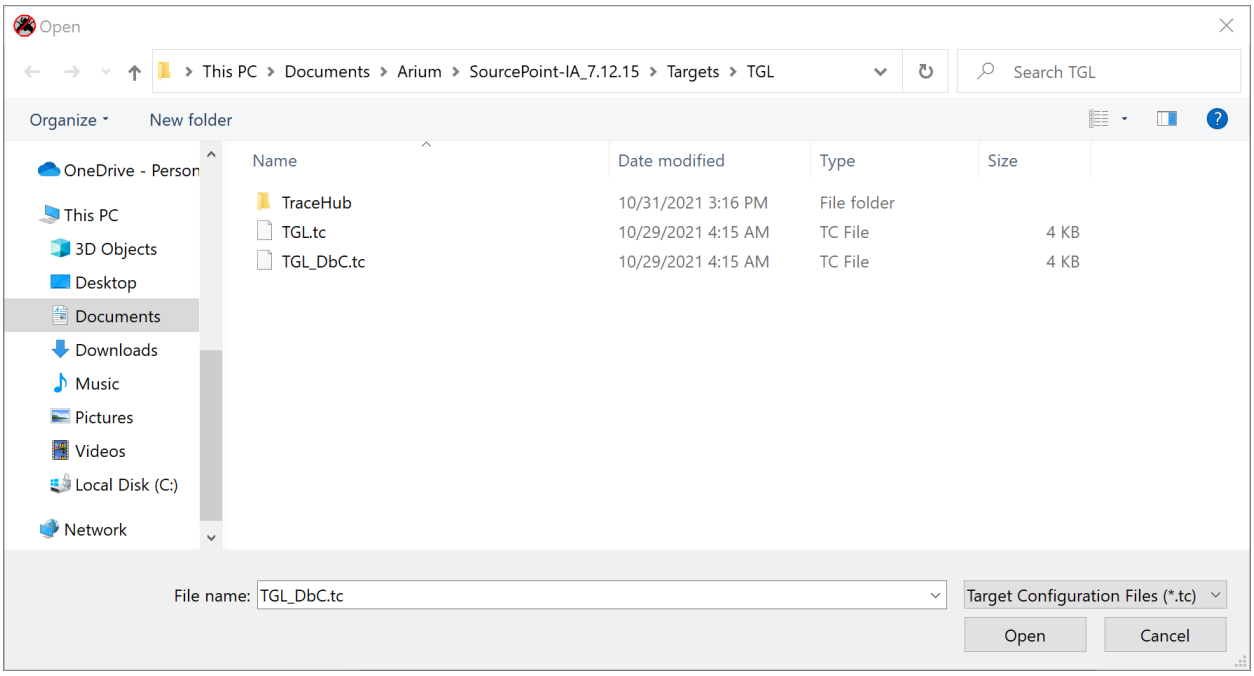


After hitting `Next`, you'll be prompted for the Project Name, Location to store the Project, and the location of the Target Configuration file:

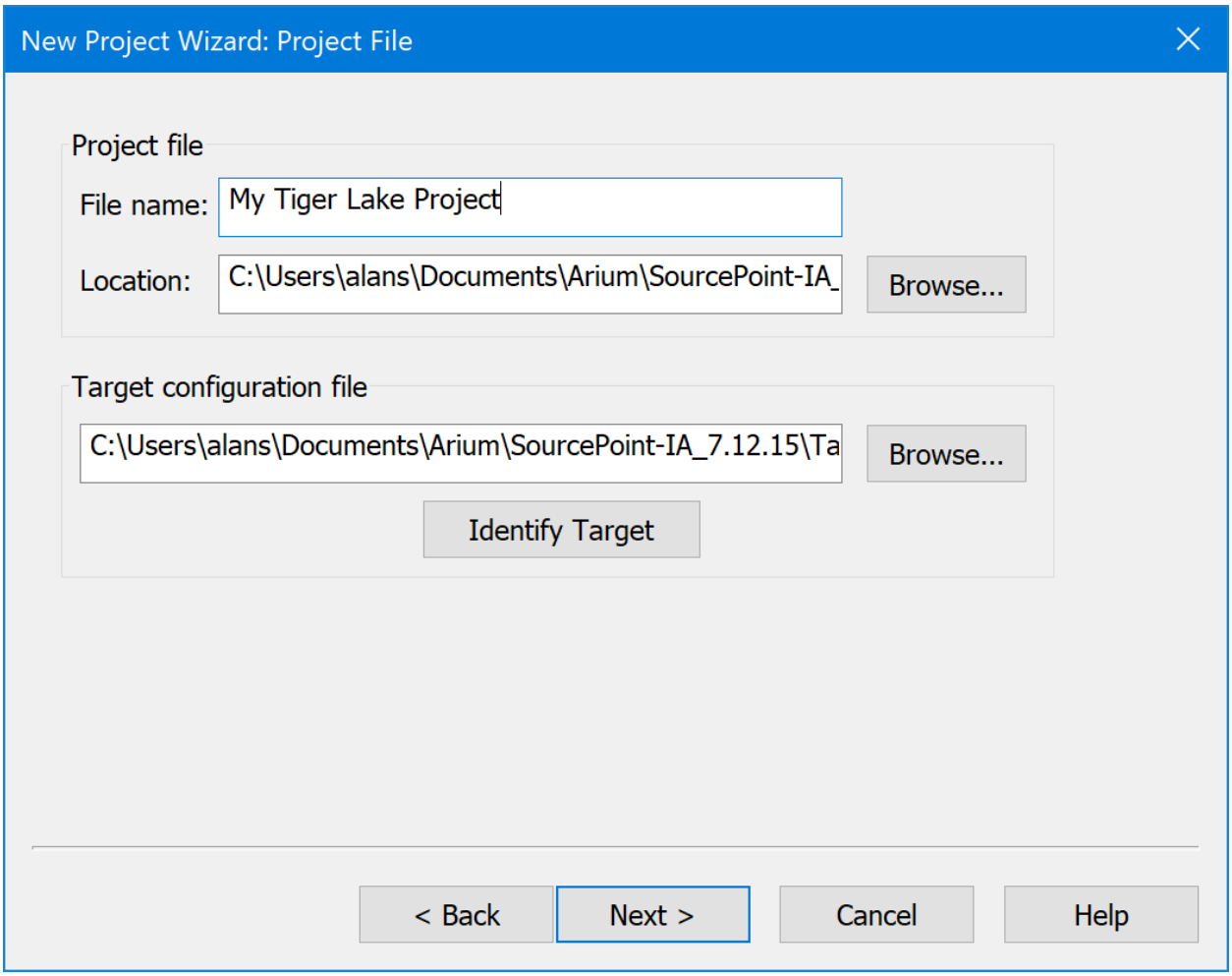


Note that the Target Configuration (TC) files located in `C:\My Documents\Arium\Targets` are used in conjunction with the `jtag-devices.xml` file to define the specific silicon and SourcePoint settings necessary to ensure a successful DCI connection.

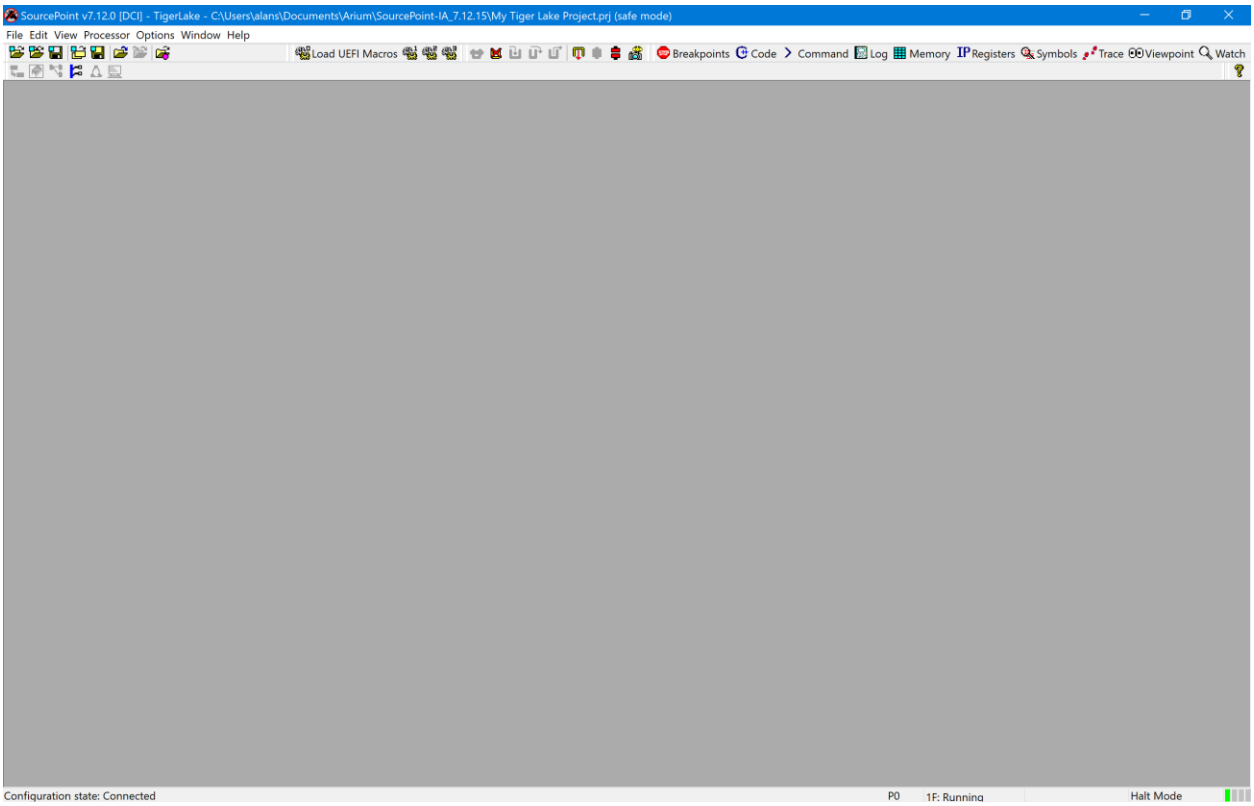
You can do an `Identify Target` to automatically select the TC file of interest, or you can manually select the Tiger Lake DbC TC file (`TGL_DbC.tc`) and hit Open:



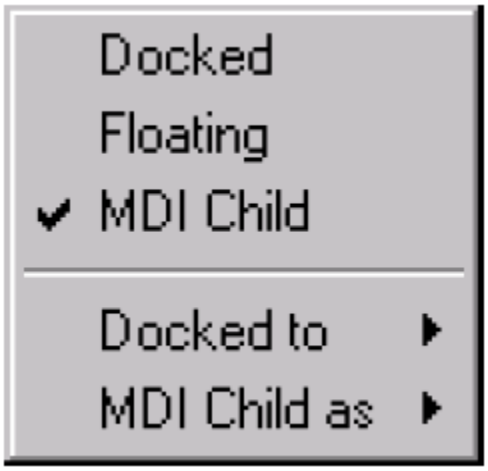
Be sure to select the DbC version! Then, your screen should look something like this:



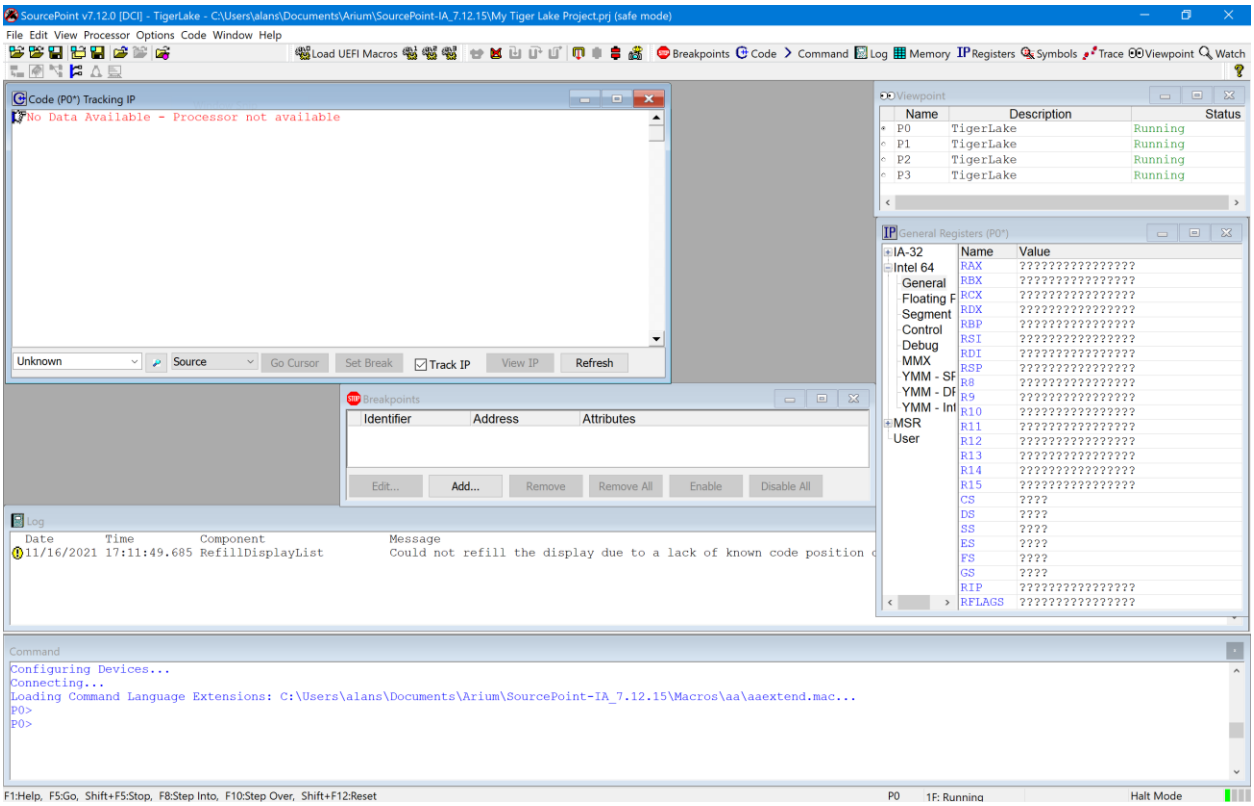
Hit `Next`, then `Finish`, and SourcePoint should successfully connect to the target. You should see "`JtagTest: Successful operation`" followed by "`Configuration state: Connected`" in the Status bar at the bottom left:



Now the fun part begins. Click on the buttons at the top to set up the Viewpoint, Code, Command, Registers and other windows to your own preference. Move the windows around and resize them to take best advantage of your available screen real estate. You can right-click in the title bar of each window to change its type and, for example, to dock the window to the bottom, right side, etc.

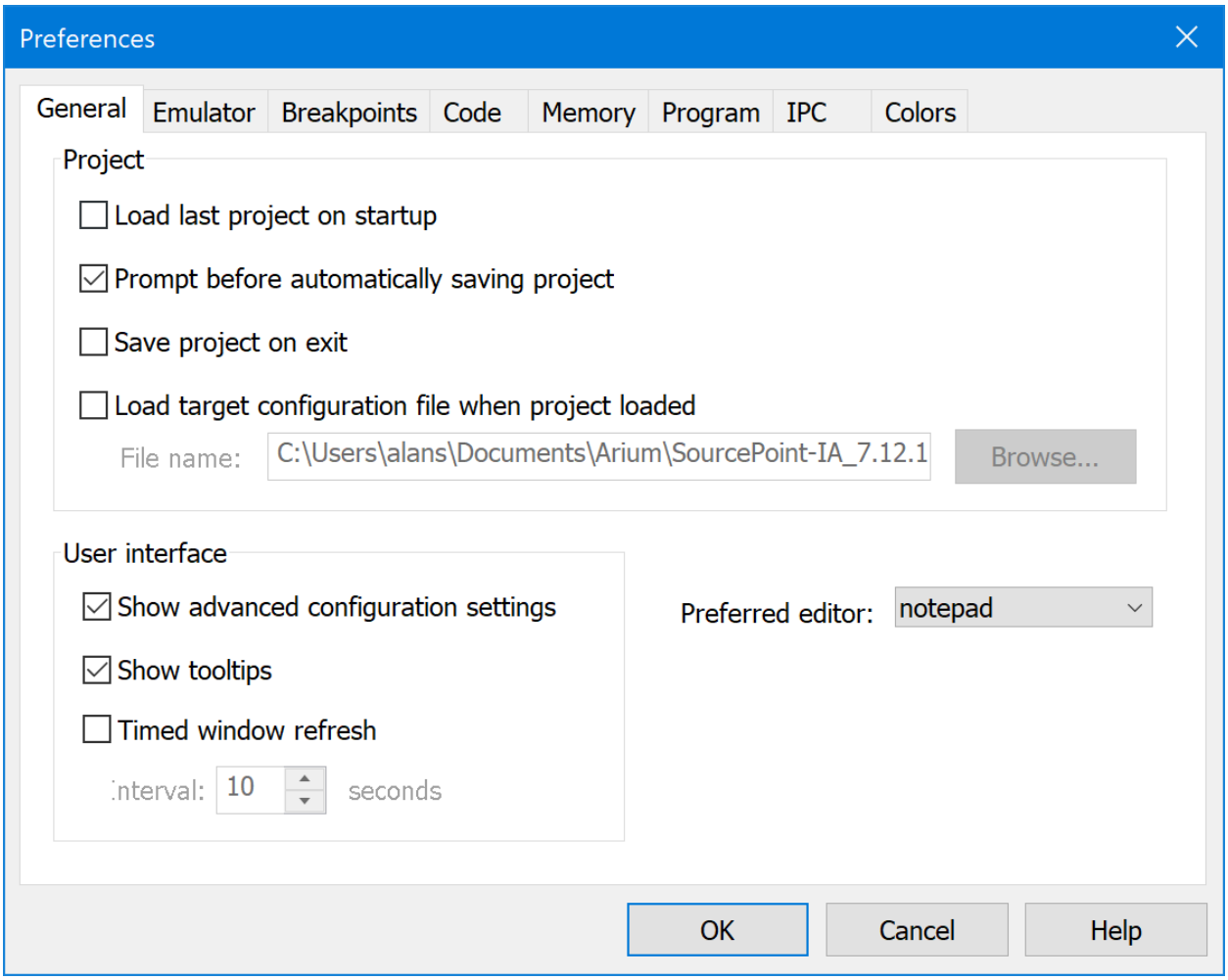


A sample layout is below:

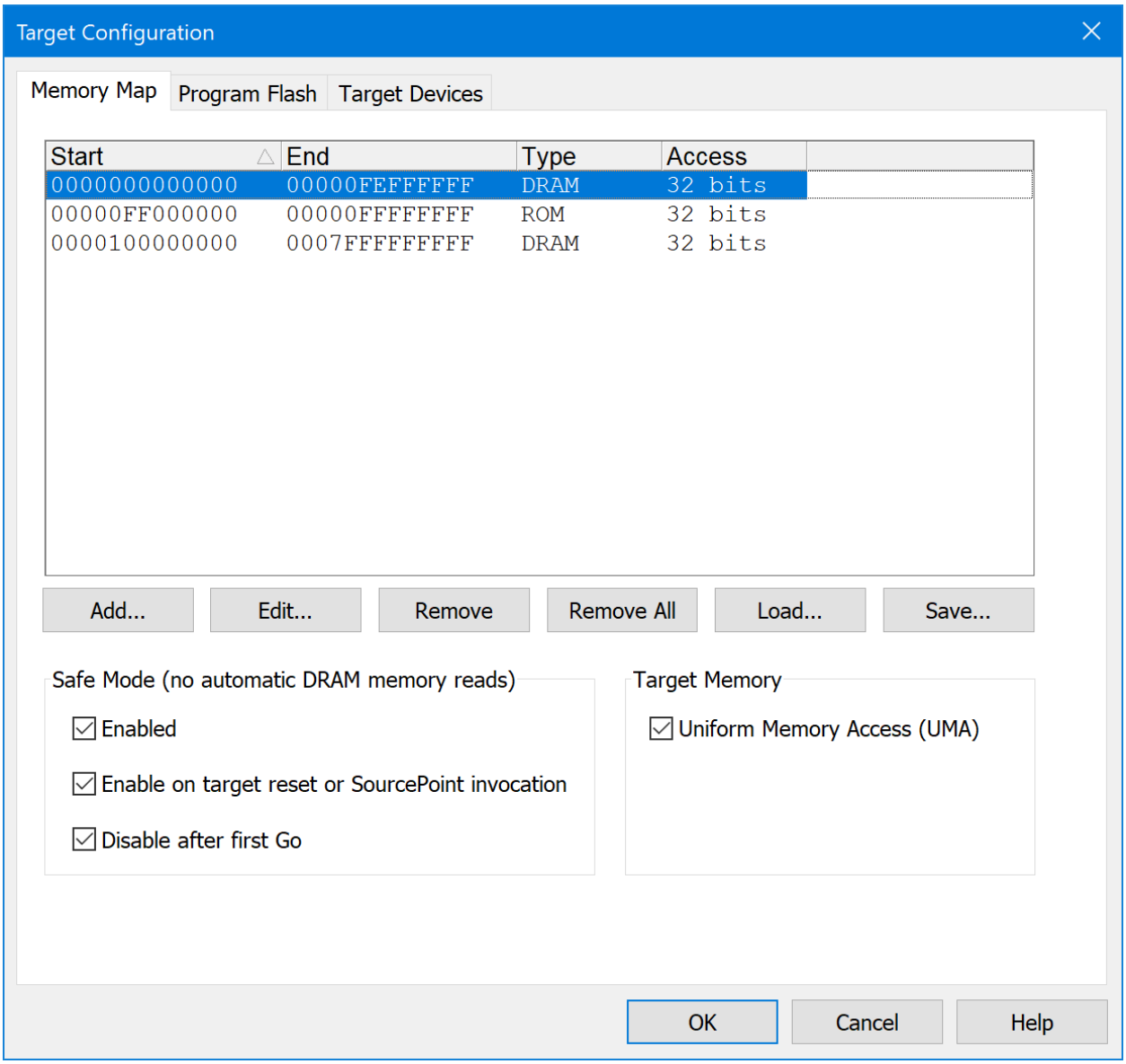


Power Tip: the window layout can be saved as a separate file to your Project. That way, when you create a new Project, you can just Load the layout file separately, without having to create and move the windows around again. Choose `File > Layout > Save Layout…` to create a .lyt file, and then do a `Load Layout…` to save yourself time every time you create a new Project.

Power Tip: Exploring `Options > Preferences…` and some of the other menu items may give you some more labor-saving ideas. For instance, I like to ensure that both `Load last project on startup` and `Save project on exit` are disabled. This gives me more control on entry and exit from the application:



Now, for the AAEON UP Xtreme i11 board, there is one important thing to do to ensure that SourcePoint recognizes the memory layout and can perform some of the advanced Trace functions: this is to adjust the Memory Map. This will also eliminate some extraneous “address translation” errors in the Log window, in case you happened to notice those (they are generally, but not always, harmless). Go into `Options > Target Configuration`, and you should see the default mapping:



For this Tiger Lake board, you need to change these default settings (note that this is an excerpt out of your Project .prj file):

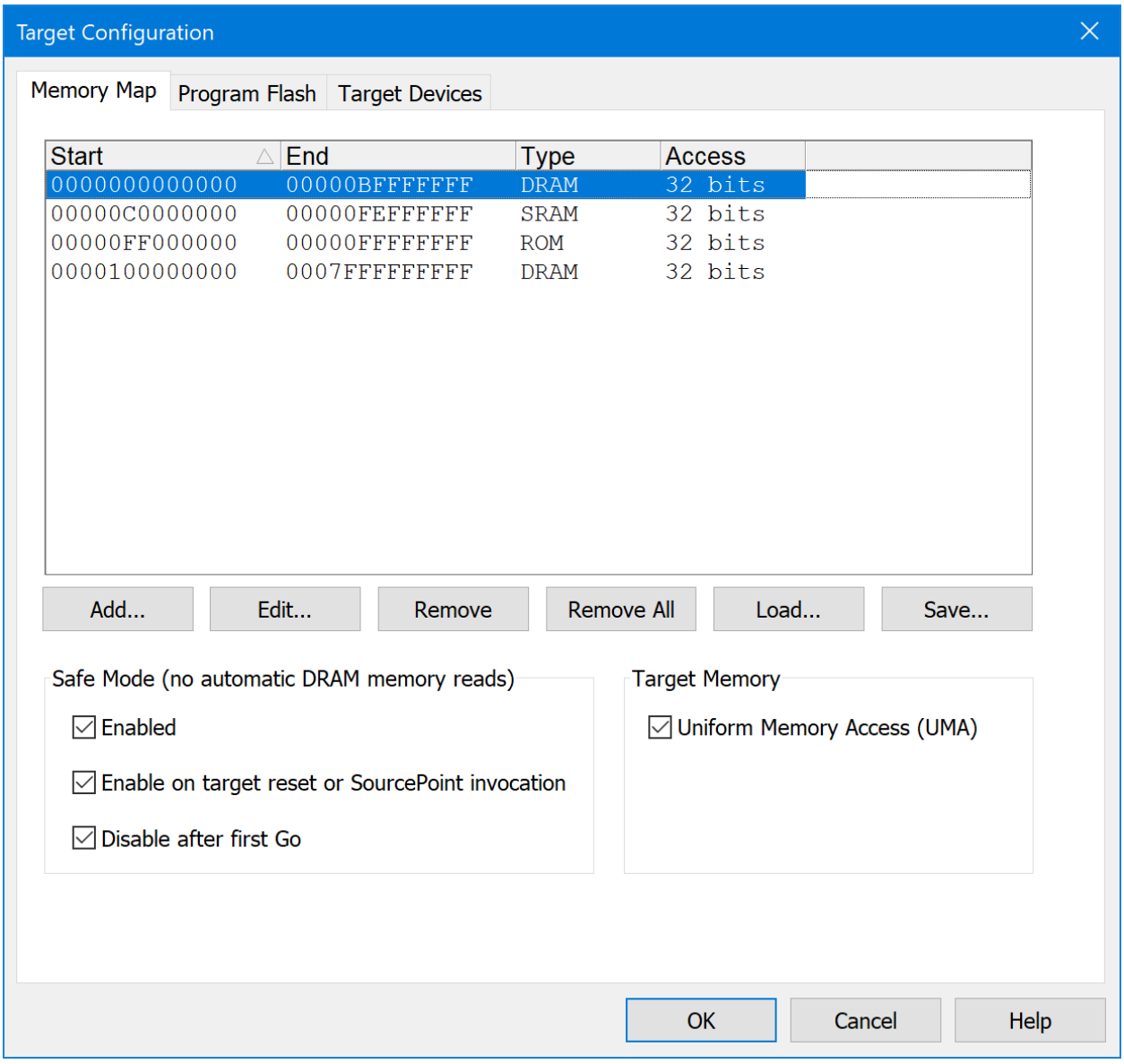
```
[MemoryMap]
Range0=0-feffffff
Range0_AccessWidth=32 bits
Range0_Processor=All
Range0_Type=DRAM
Range1=ff000000-ffffffff
Range1_AccessWidth=32 bits
Range1_Processor=All
Range1_Type=ROM
```

```
Range2=100000000-7fffffffff
Range2_AccessWidth=32 bits
Range2_Processor=All
Range2_Type=DRAM
```

to:

```
[MemoryMap]
Range0=0-bfffffff
Range0_AccessWidth=32 bits
Range0_Processor=All
Range0_Type=DRAM
Range1=c0000000-feffffff
Range1_AccessWidth=32 bits
Range1_Processor=All
Range1_Type=SRAM
Range2=ff000000-ffffffff
Range2_AccessWidth=32 bits
Range2_Processor=All
Range2_Type=ROM
Range3=100000000-7fffffffff
Range3_AccessWidth=32 bits
Range3_Processor=All
Range3_Type=DRAM
```

The easiest way to do this is to `Edit` the first entry in the Memory Map, changing 0FEFFFFFF to 0BFFFFFFF, and then do an `Add` to add in `0C0000000 0FEFFFFFF SRAM 32 bits`. The new Memory Map will look like this:

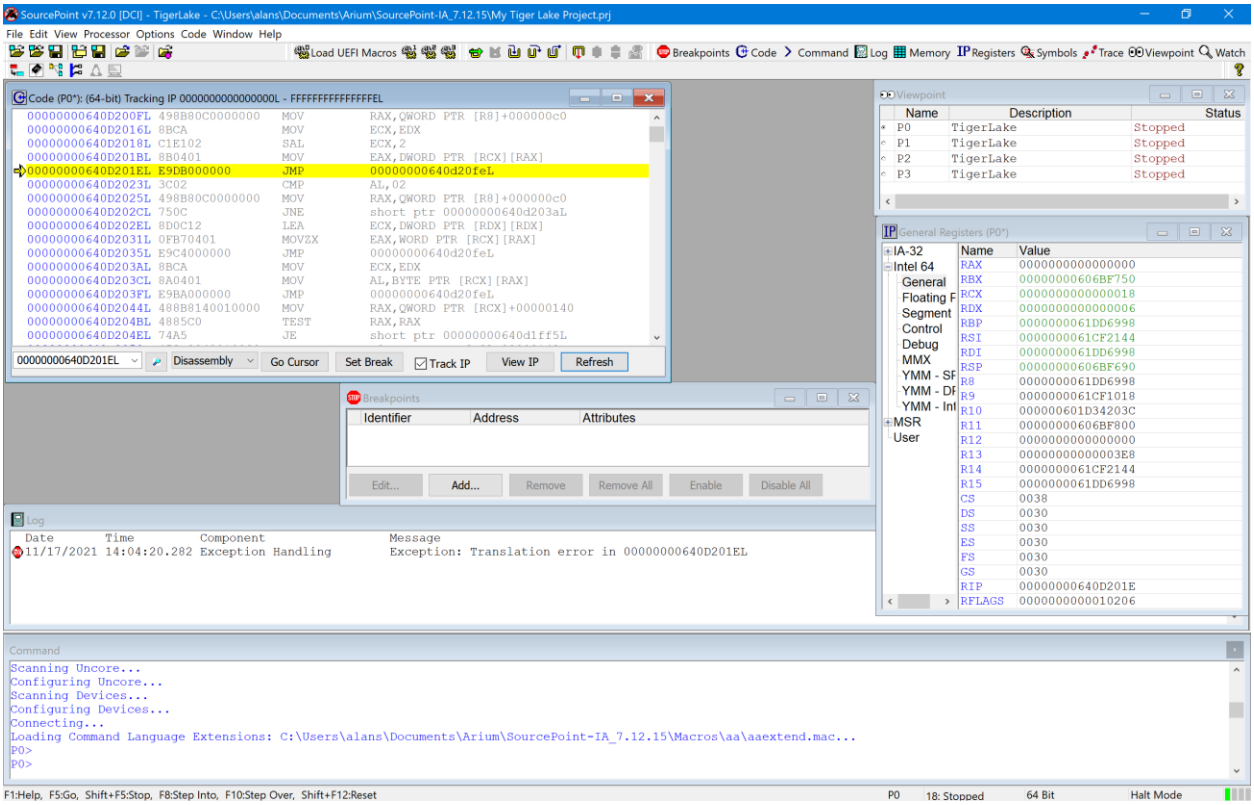


Next, it is necessary to disable the platform TCO Timer, so that you can halt at the reset vector successfully, without the target going into limbo. Click on `File > Macro > Configure Macros…` and click on the `Event Macros` tab. Click on Browse and select the `Macros\Intel\ADL_TCO_Timer_Disable.mac`. This is the one you need to choose for the `Reset (after)` event.

This is a good point to do a `Project > Save Project…` That way, you don't have to start all over, if for some reason your project gets messed up.

At this point, you are ready to fully begin your debug session. Many of the operations can be accessed via the toolbar at the top of the screen. You can issue a `Stop` on the

target, `Step Into`, `Reset` it and halt at the reset vector, set some breakpoints, `Go` to the breakpoint, and so on.



Hit the `Refresh` button in the Code window after the first Stop. This is only necessary once; the Code window will automatically refresh with all run-control operations (stop, go, single-step, etc.) afterwards.

Refer to the SourcePoint User Guide in your install directory for detailed instructions on using all of the tool's features.

Congratulations, you have mastered SourcePoint's basic capabilities, and are using run-control. Many users are content to just use these basic operations, because run-control by itself is very powerful. However, iff you wish to master the product and use some of its more advanced features, read on.

Advanced Topics: Using Trace

Trace is by far one of the most useful debugging utilities for triaging the most difficult, hard-to-reproduce bugs. Fortunately, the Tiger Lake CPU is equipped with all the latest-and-greatest trace logic, and SourcePoint supports them all.

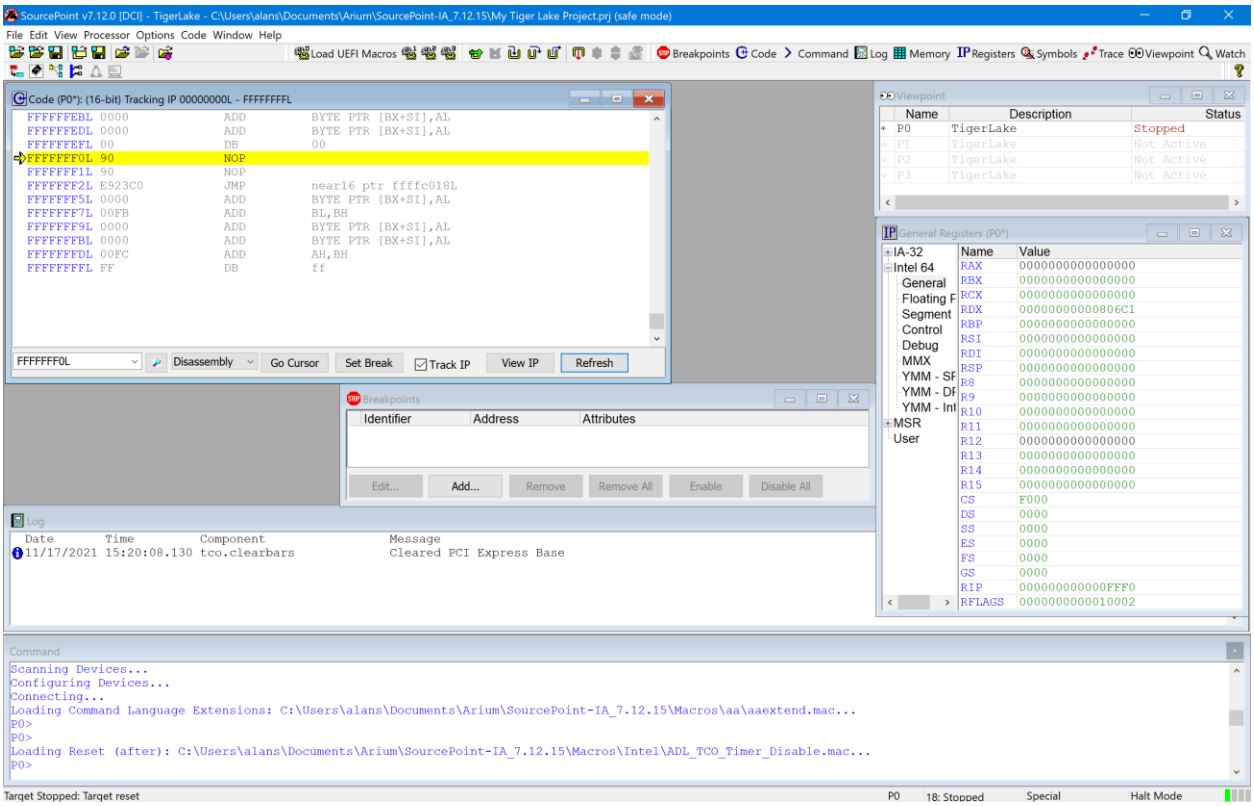
Let's look at a few of them, and how to configure their use in SourcePoint.

Intel Trace Hub: SVEN

Event tracing on the TGL platform is accomplished by the Intel Trace Hub (ITH). Fortunately, using DCI, events supported by the ITH can be streamed directly out of system reset. The one limitation that exists is that some events (like Port IN/OUT tracing) happen so frequently at some points of the boot process that they overwhelm the capacity of the USB 2.0 (DbC2) connection and event processing, and thus cause buffer overflows – but these should be rare as long as the events collected are relatively close to the debug point of interest.

System Visibility Event Nexus (SVEN) is a means by which, within the AMI BIOS, printf statements are output through the ITH, as opposed to the serial port. This speeds up the boot time, and reduces the likelihood of port latency contributing to the masking of bugs.

The first thing to do is to configure the ITH. Reset the target and halt at the reset vector, address FFFFFFF0:

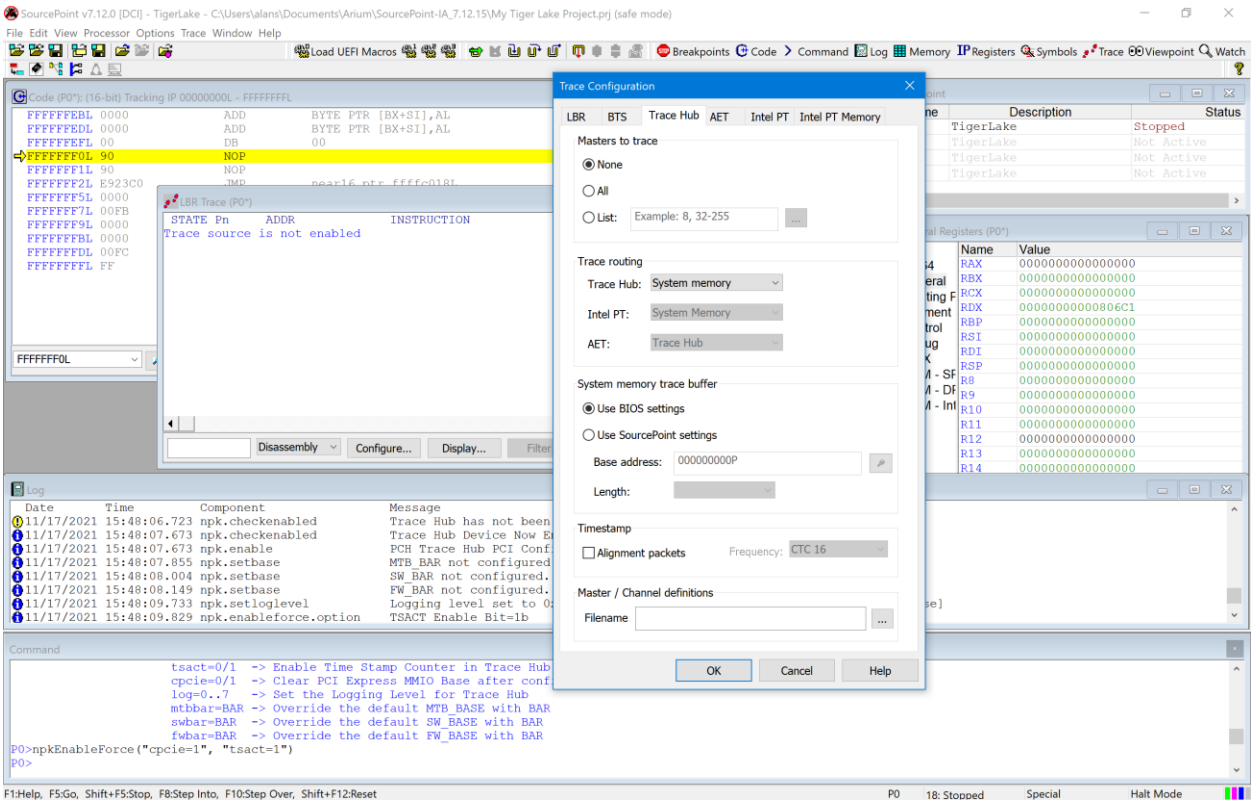


The next step is to load in the ITH macros. This is accomplished by `File > Macro > Load Macros…` and selecting `C:\My Documents\Arium\SourcePoint-IA_7.12.15\Macros\Intel\TraceHub.mac`. Then, in the Command window, at the `P0>` prompt, type:

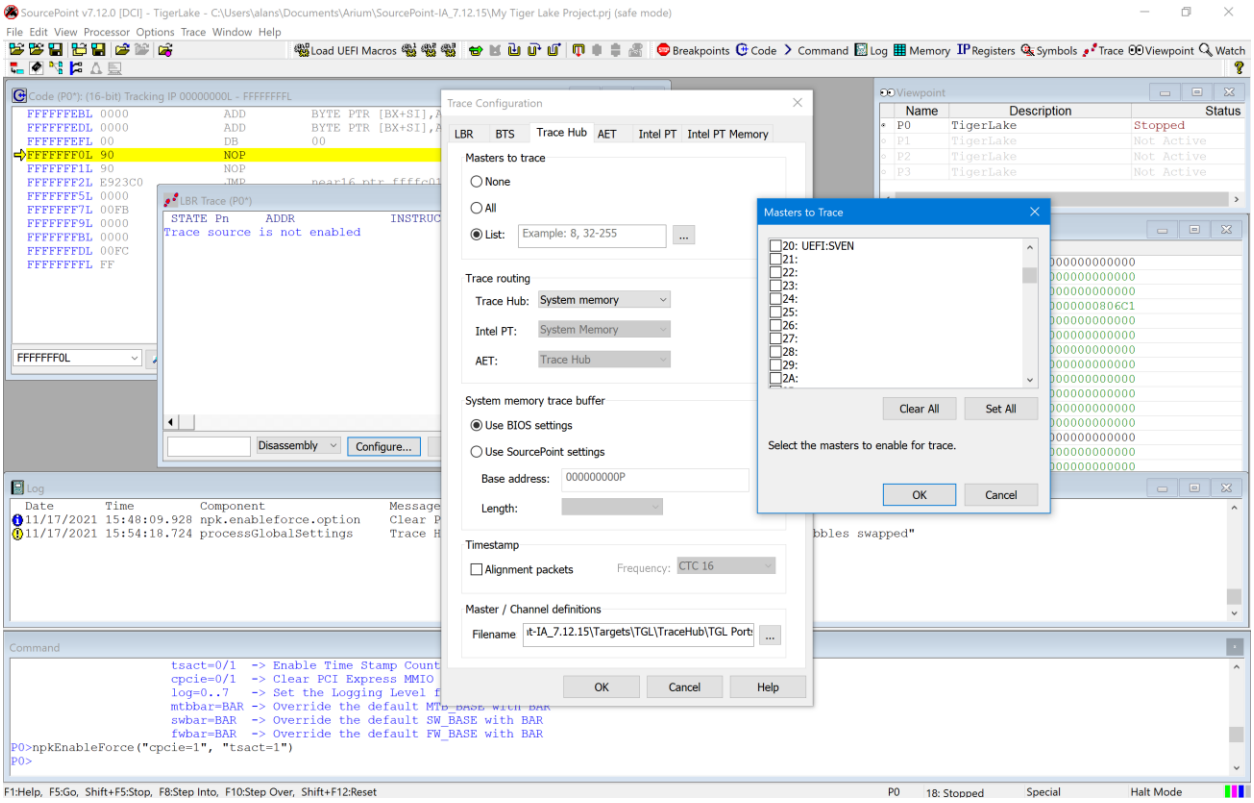
```
npkEnableForce("cpcie=1", "tsact=1")
```

Power tip: If you want to know more about what this macro can do, type `npkEnableForce_Help()`.

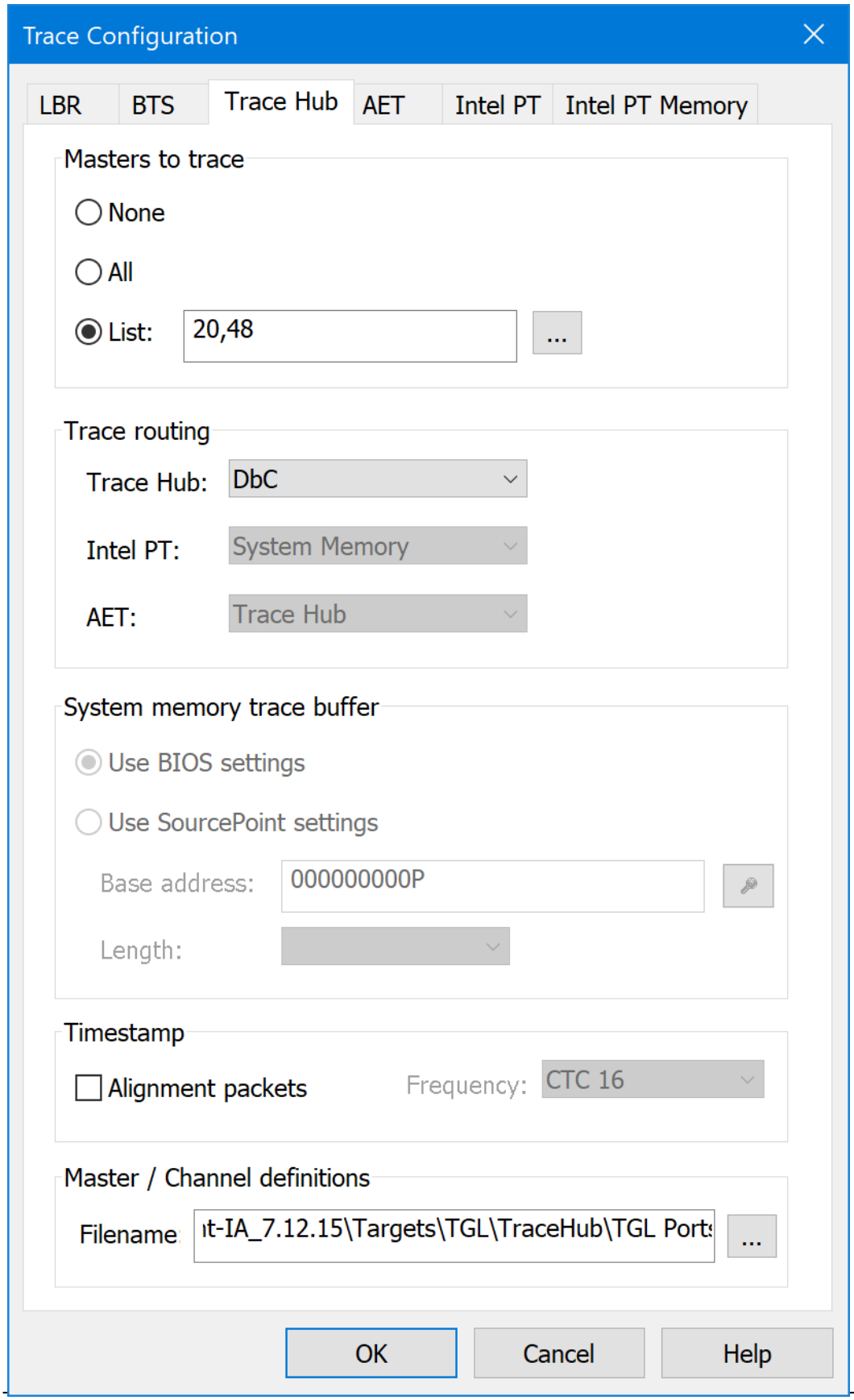
Now, it's time to set up the ITH, and let it know what you want to trace. Click on the `Trace` button in the toolbar at the top, click on the `Configure…` button, and then click on the `Trace Hub` tab:



Go to the bottom and click select the `C:\MyDocuments\Arium\SourcePoint-IA_7.12.15\Targets\TGL\TraceHub\TGL Ports.xml` file to define the `Master / Channel definitions`. Click `OK`, and then go back in again, click on `List` under `Masters to trace` and select the ellipses (…) to look at the choices available:

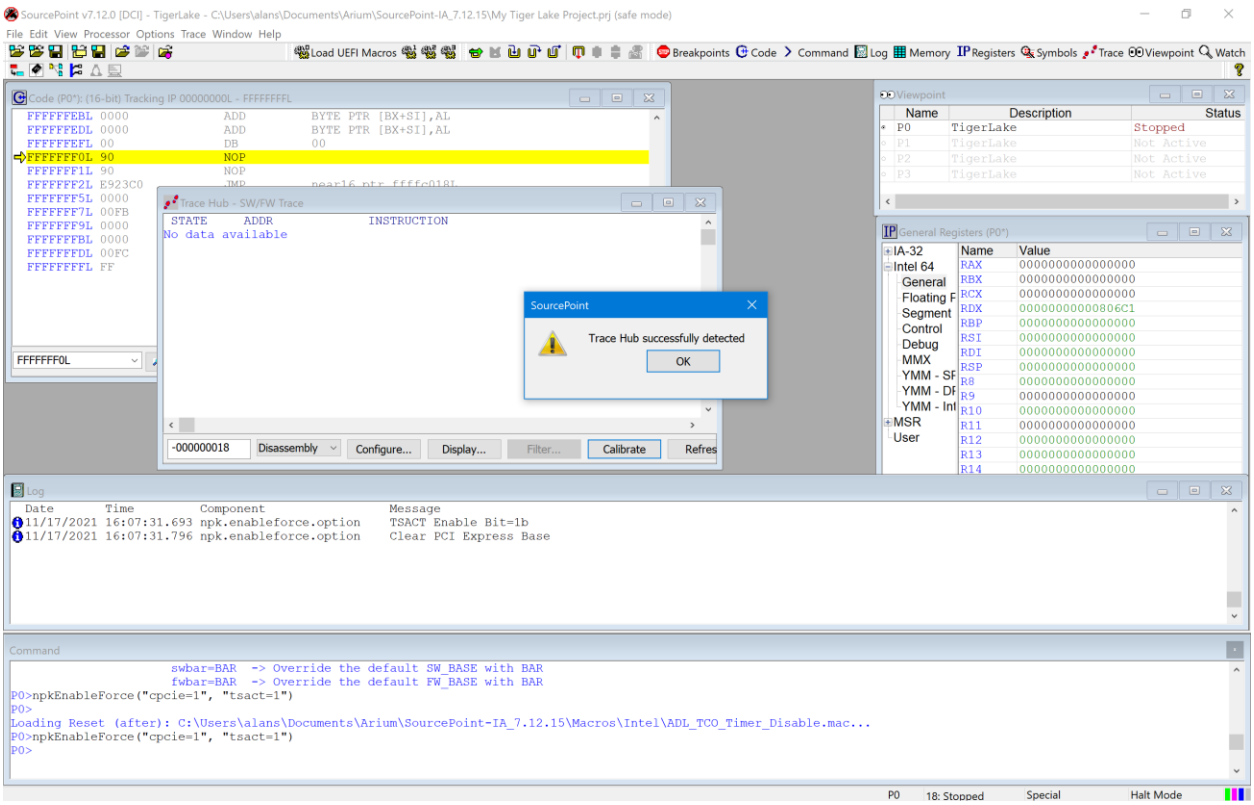


For now, select 20: `(UEFI:SVEN)` and `48 (UEFI:SVEN)`. Also, under `Trace Routing`, select `Trace Hub` and set to `DbC`:


Trace Configuration
LBR
BTS
Trace Hub
AET
Intel PT
Intel PT Memory
Masters to trace
None
All
List:
20,48
...
Trace routing
Trace Hub:
DbC
Intel PT:
System Memory
AET:
Trace Hub
System memory trace buffer
Use BIOS settings
Use SourcePoint settings
Base address:
000000000P
Length:
Timestamp
Alignment packets
Frequency:
CTC 16
Master / Channel definitions
Filename:
lt-IA_7.12.15\Targets\TGL\TraceHub\TGL Port
...
OK
Cancel
Help

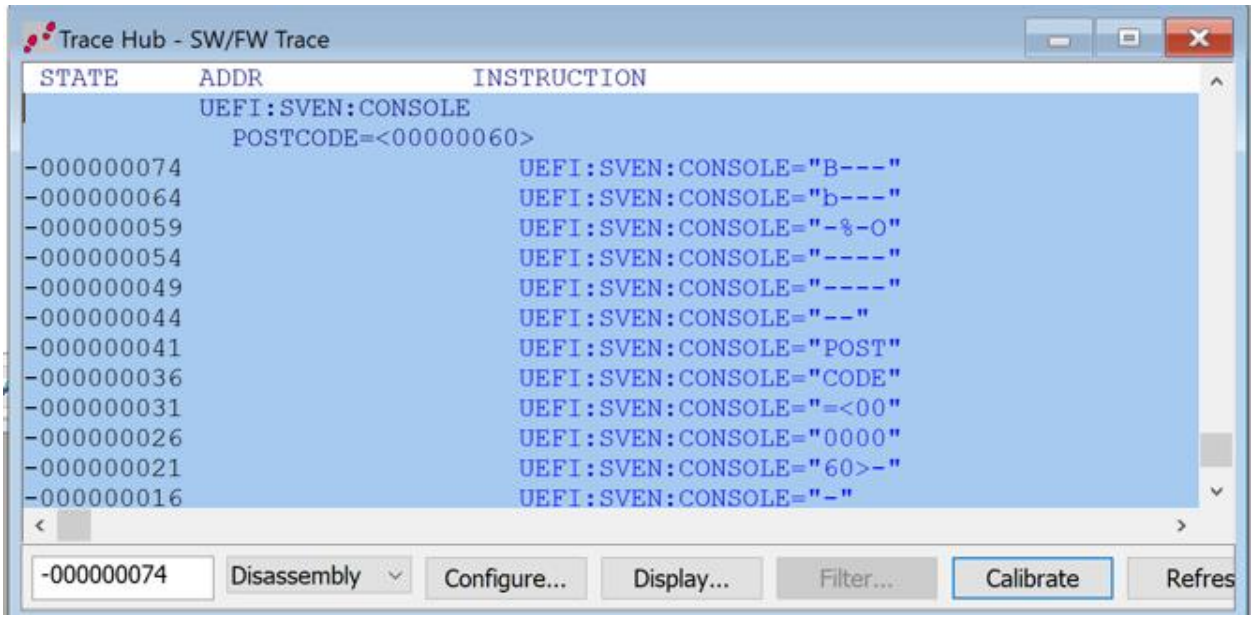
Hit `OK`.

Now, and this is very important, you need to calibrate the ITH. The Trace window should be right-clicked and set to `Trace Hub – SW/FW Trace` (if it isn't already). Click on the `Calibrate` button, and you should be rewarded with it being successfully detected:

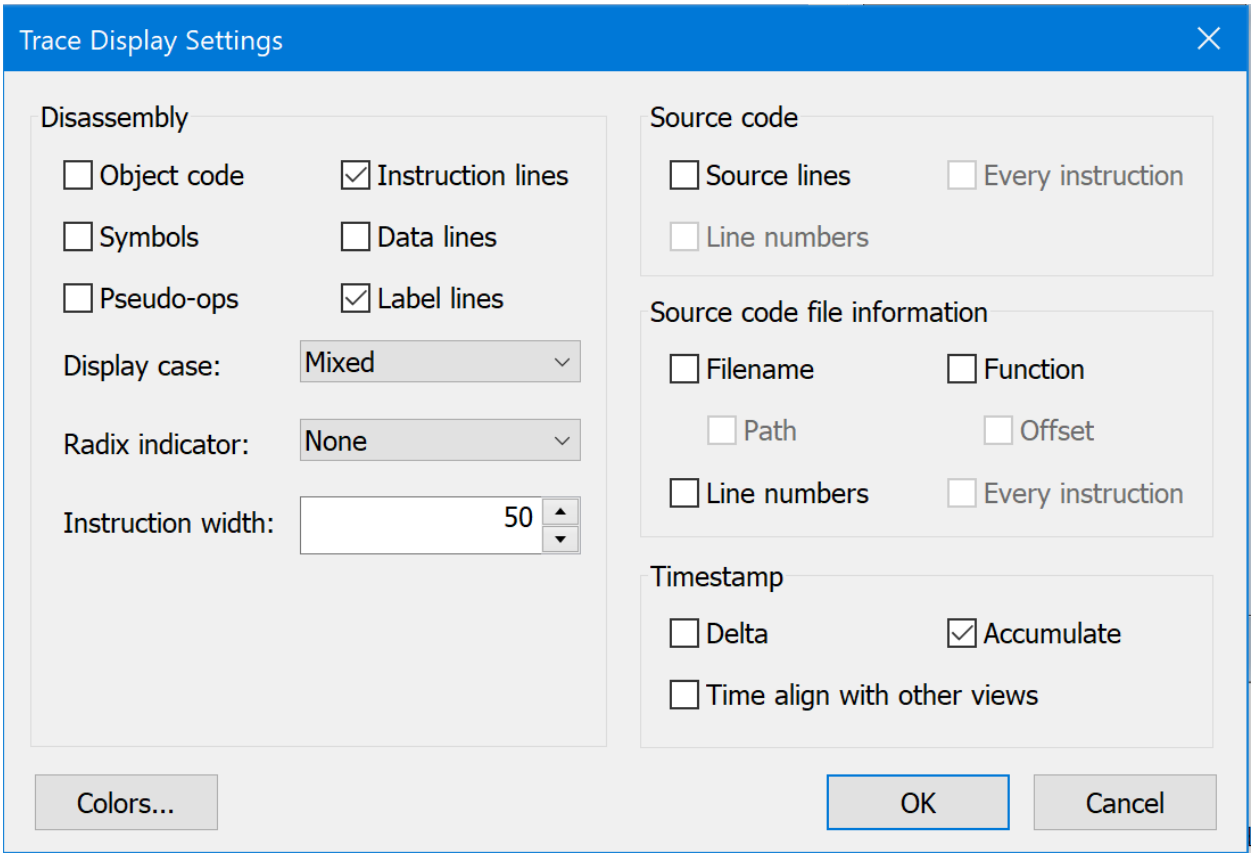


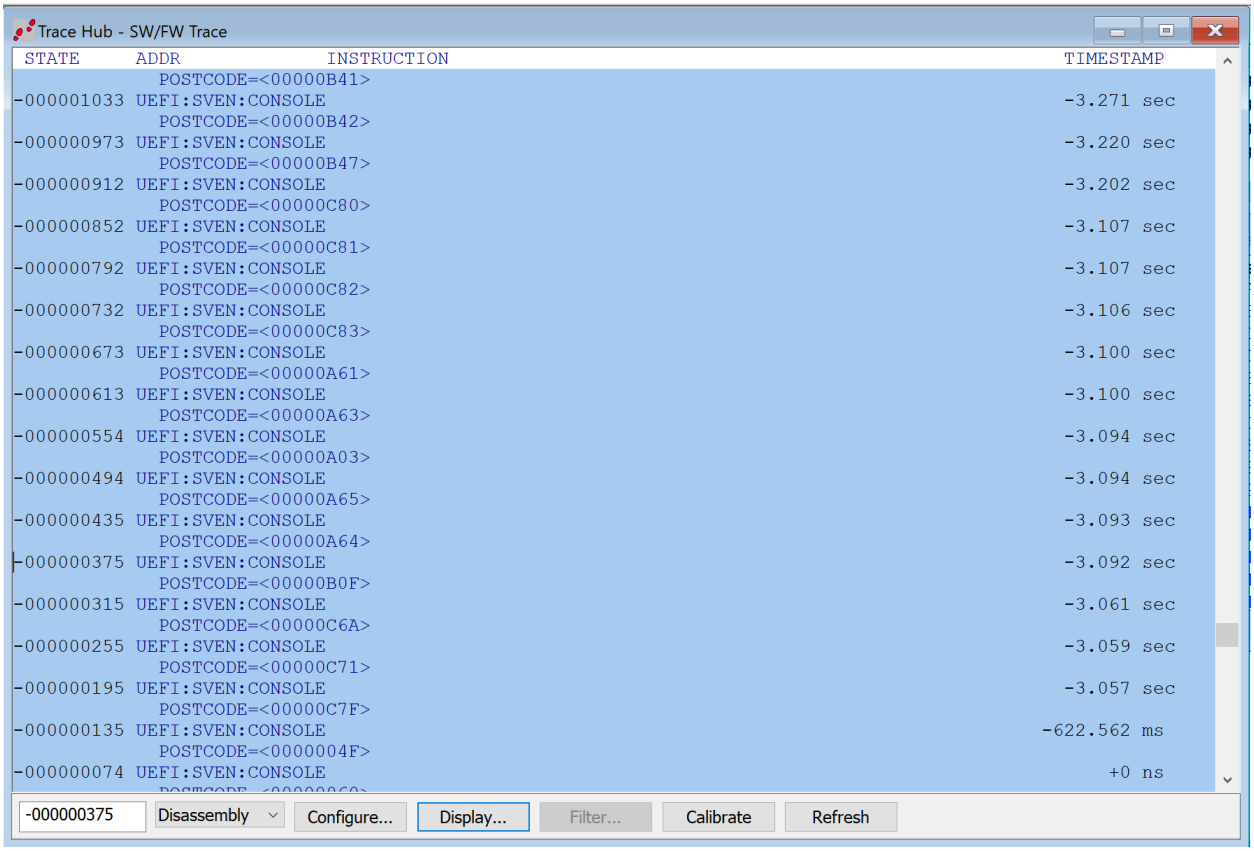
Power tip: The `Calibrate` button only needs to be used once per project. The `npkEnableForce("cpcei=1", "tsact=1")` command must be re-issued after each reset.

Then, hit the `Go` button in the top toolbar, and boot to the UEFI shell. Then hit `Stop`. You will see the SVEN printf data in all its glory:



Actually, it looks better if you select the `Display…` button, and de-select Data lines:

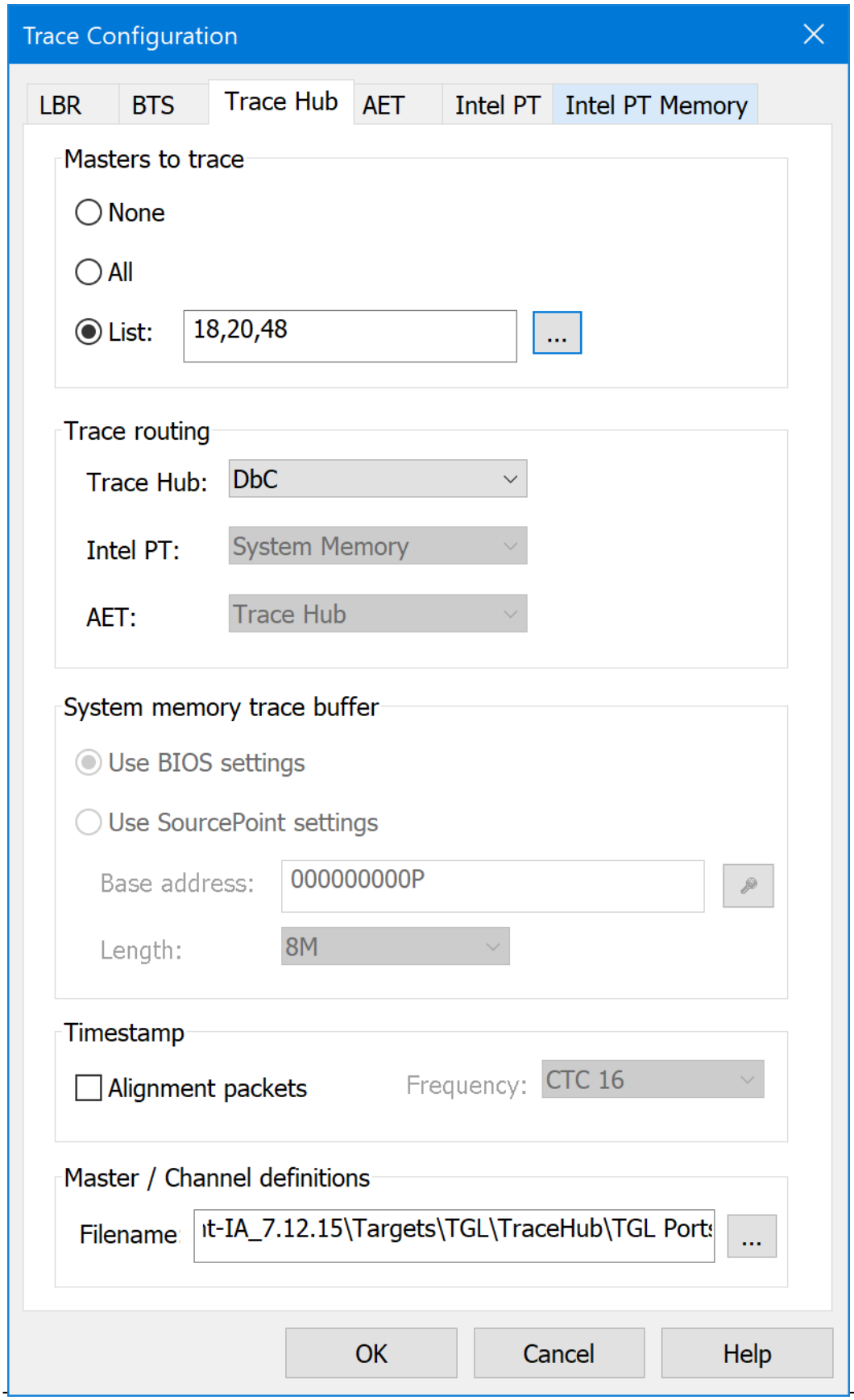




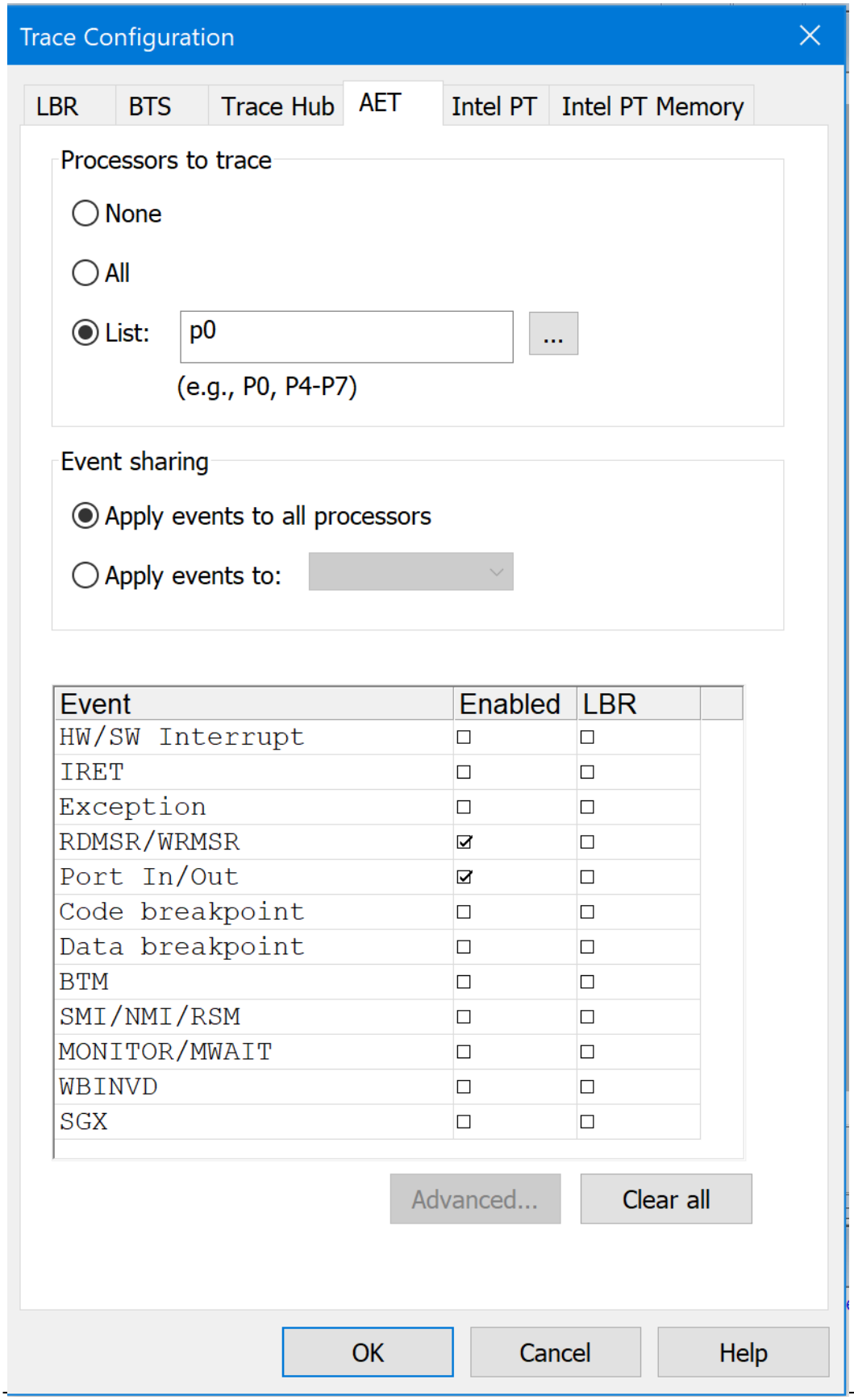
This BIOS is not particularly verbose, so all we see here are the POST codes. But, you get the idea.

Architectural Event Trace

Click on the `Trace` button in the top toolbar, and select `Configure…` again. Within the `Trace Hub` tab, add '18' to the list of `Masters to trace`.


Trace Configuration
LBR
BTS
Trace Hub
AET
Intel PT
Intel PT Memory
Masters to trace
None
All
List:
18,20,48
...
Trace routing
Trace Hub:
DbC
Intel PT:
System Memory
AET:
Trace Hub
System memory trace buffer
Use BIOS settings
Use SourcePoint settings
Base address:
000000000P
Length:
8M
Timestamp
Alignment packets
Frequency:
CTC 16
Master / Channel definitions
Filename:
t-IA_7.12.15\Targets\TGL\TraceHub\TGL Port
...
OK
Cancel
Help

Then click on the `AET` tab, select `p0` as `Processors to trace`, and select `RDMSR/WRMSR` and `Port In/Out` as events to trace:

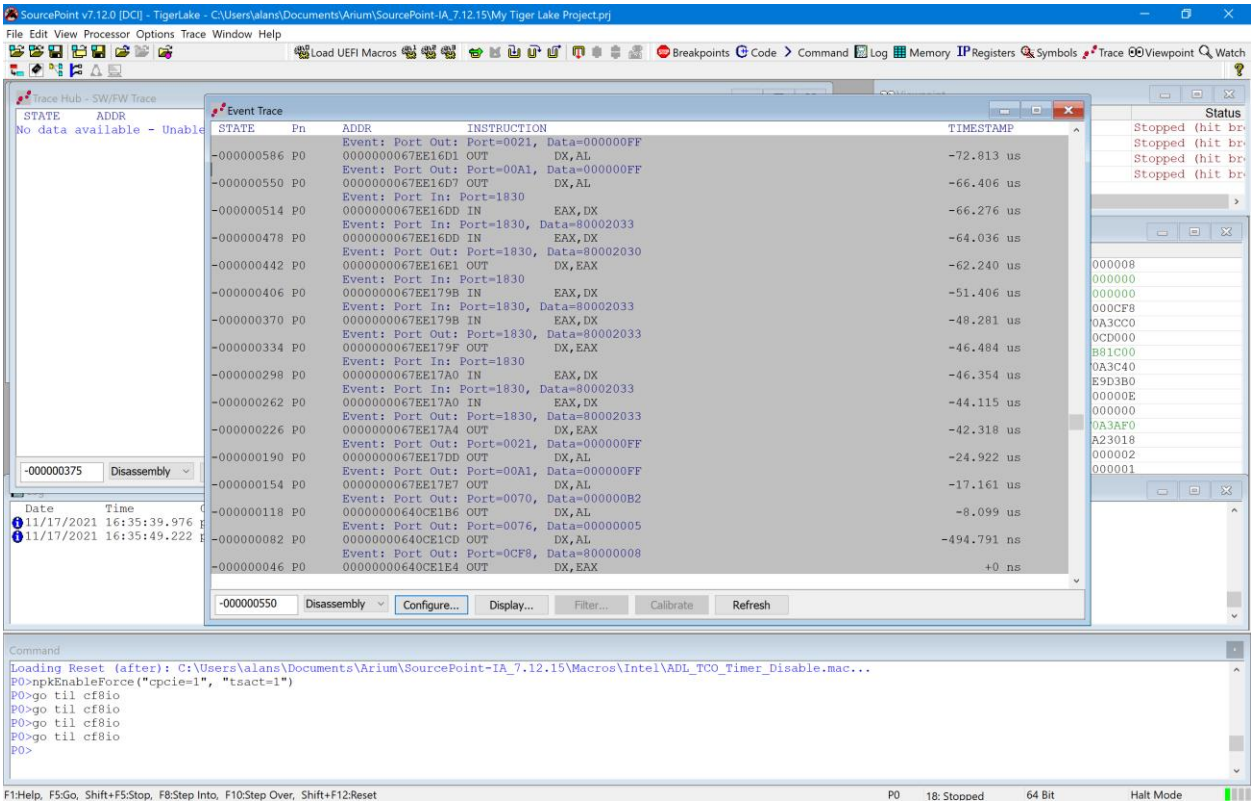

Trace Configuration
LBR
BTS
Trace Hub
AET
Intel PT
Intel PT Memory
Processors to trace
None
All
List:
p0
...
(e.g., P0, P4-P7)
Event sharing
Apply events to all processors
Apply events to:
Event
Enabled
LBR
HW/SW Interrupt
IRET
Exception
RDMSR/WRMSR
Port In/Out
Code breakpoint
Data breakpoint
BTM
SMI/NMI/RSM
MONITOR/MWAIT
WBINVD
SGX
Advanced...
Clear all
OK
Cancel
Help

Now, the trick here is not to try to capture too much data, otherwise you'll overwhelm the ITH packet processing. So, you should debug only close to your event or code of interest. One way to demonstrate this is to use the `Command` window to simulate a break on any read/write of, say, port x'CF8', the PCI CONFIG_ADDRESS. This is conveniently done by issuing at the `Command` window `P0>` prompt:

```
go til cf8io
```

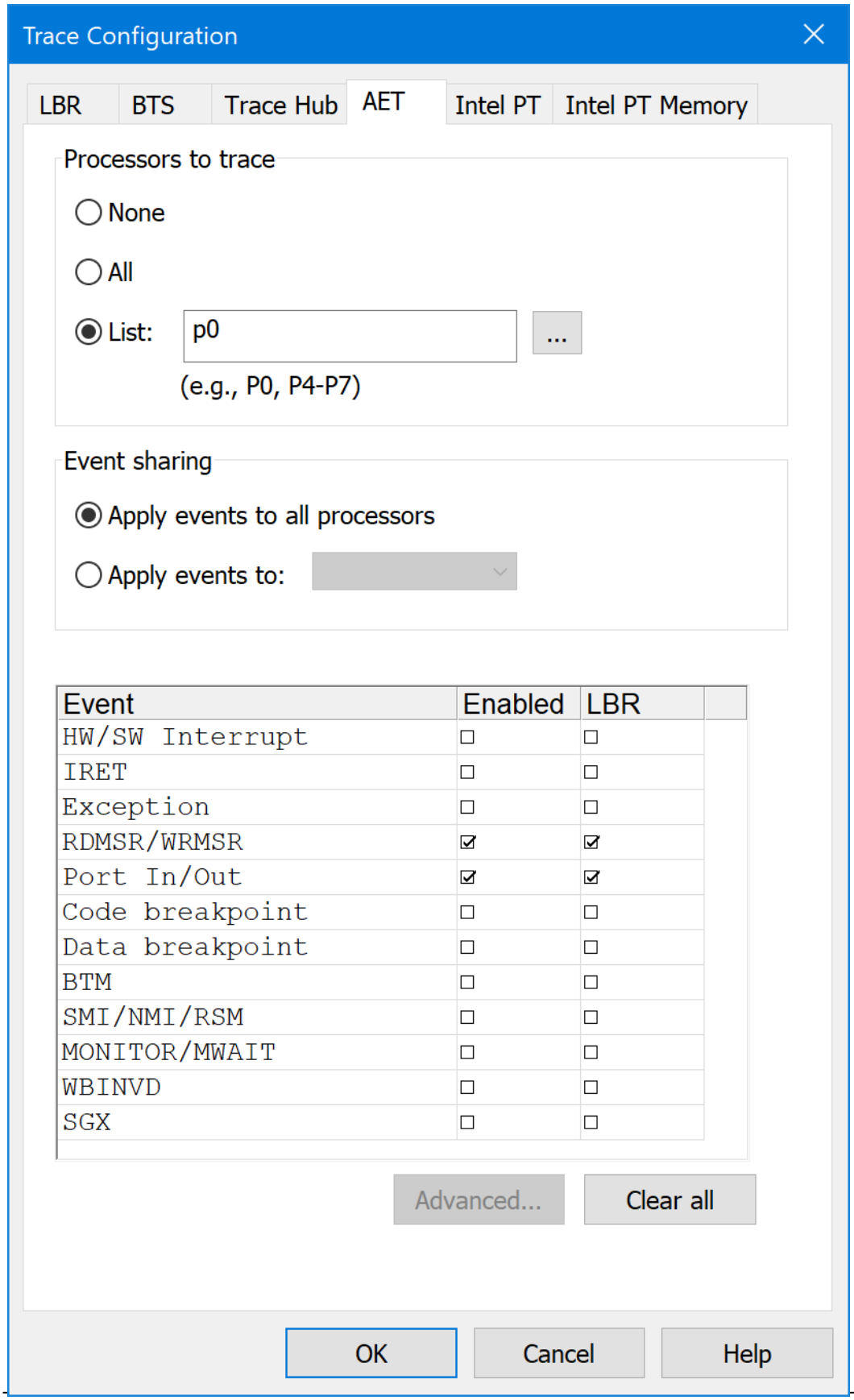
This will run the target until the next IN or OUT to CF8. This happens fairly frequently, so you'll not likely overflow any buffers.

After issuing the command, you'll see something like this:



Scrolling up a little, you'll see a mix of `Port In/Out` and `RDMSR/WRMSR`. All timestamped, and with a CYCLE that can be used to correlate with other trace sources.

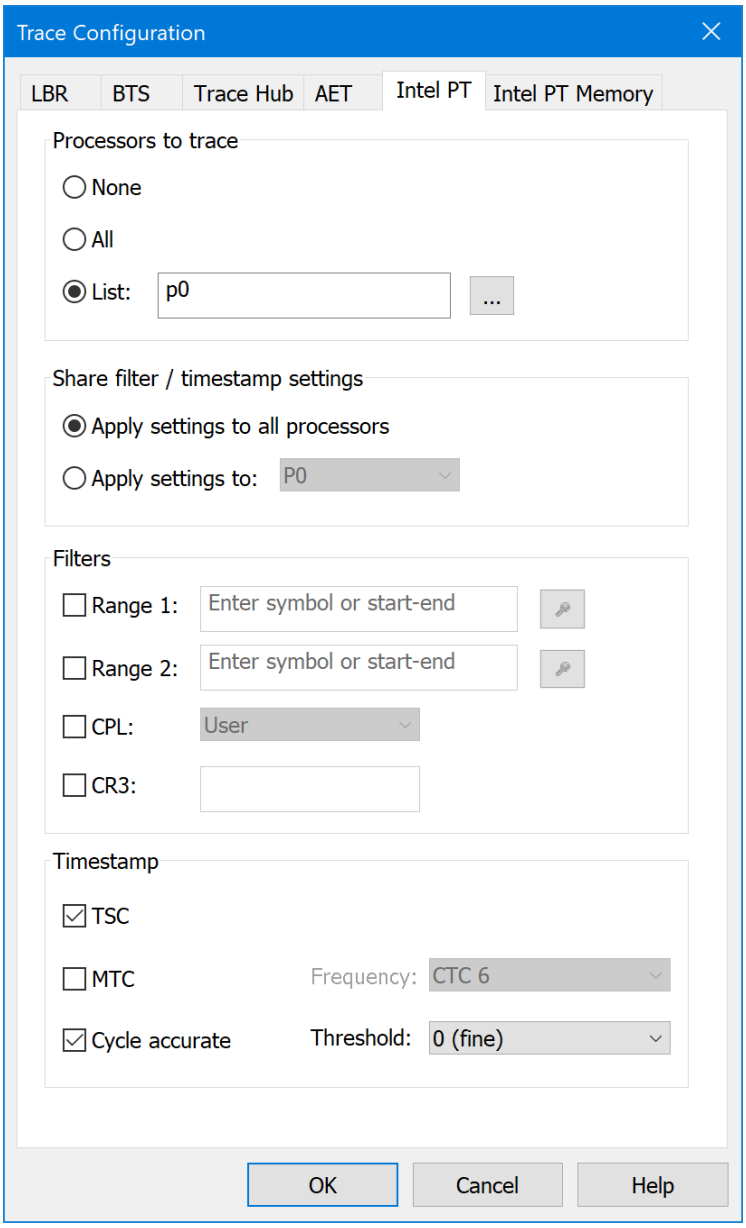
Power tip: The Last Branch Record (LBR) stack associated with each event can be captured as well. This is a very powerful debugging utility, especially when troubleshooting code execution leading up to events before system memory is initialized and Intel Processor Trace is available.


Trace Configuration
LBR
BTS
Trace Hub
AET
Intel PT
Intel PT Memory
Processors to trace
None
All
List:
p0
...
(e.g., P0, P4-P7)
Event sharing
Apply events to all processors
Apply events to:
Event
Enabled
LBR
HW/SW Interrupt
IRET
Exception
RDMSR/WRMSR
Port In/Out
Code breakpoint
Data breakpoint
BTM
SMI/NMI/RSM
MONITOR/MWAIT
WBINVD
SGX
Advanced...
Clear all
OK
Cancel
Help

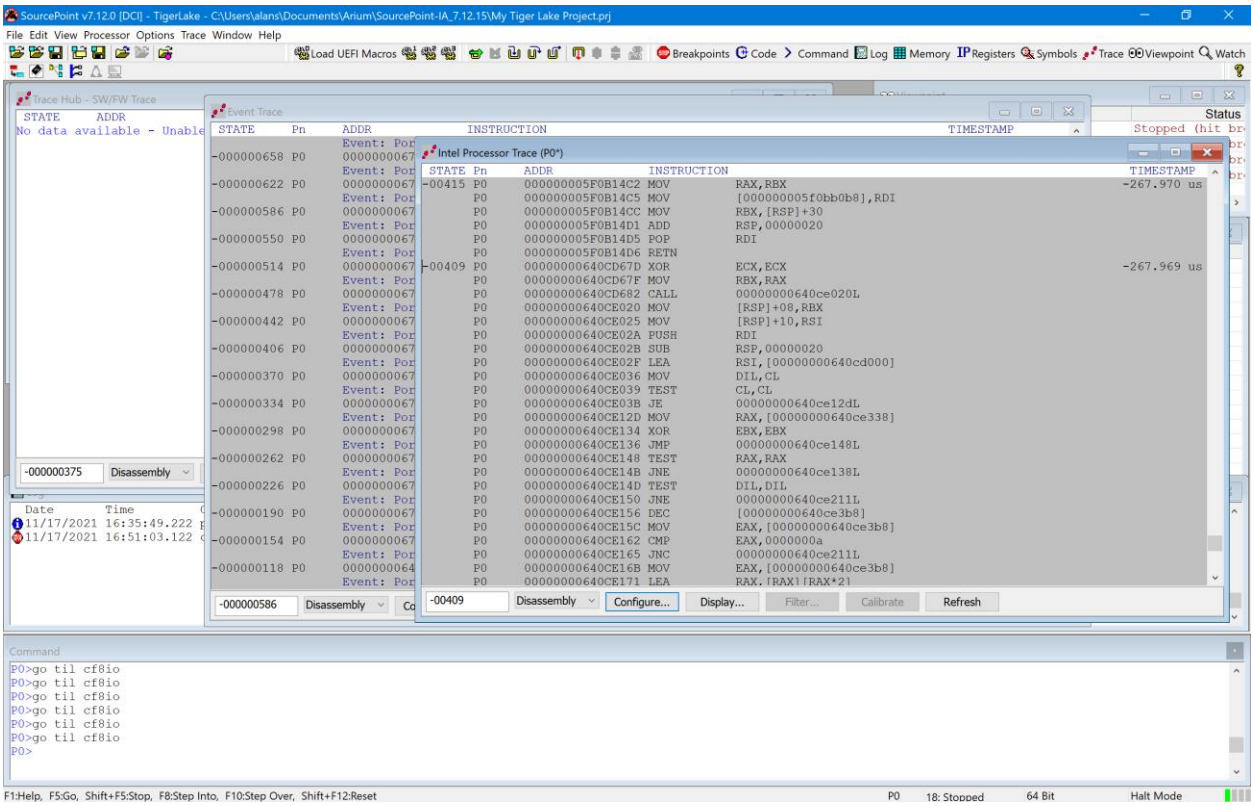
Intel Processor Trace

Intel PT is available only after system memory is initialized.

It's easy to set up. Click on the `Trace` button in the top toolbar, click the `Configure…` button, click on the `Intel PT` tab, put `p0` in `Processors to trace`, and click on `TSC` and `Cycle accurate` under the `Timestamp` heading:



That's all. Then use the `go til cf8io` trick to capture some instruction trace data:



There's a lot more that you can see and do with these Trace utilities. SourcePoint can use Intel PT to display a Call Chart and Call Tree. You can open up Code Tracking windows that update dynamically as you walk through the code, showing you exactly where you are and the interaction between code and events. When you have source and symbols available, the firmware flow becomes much more intuitive and visual. Indulge your curiosity and imagination.

The video at https://www.asset-intertech.com/wp-content/uploads/2021/11/UP-Xtreme-i11-Getting-Started.mp4 shows some of these capabilities. The SourcePoint User Guide also provides a very thorough, comprehensive review of the tool. And visit our SourcePoint Academy for helpful "How To" content.

Troubleshooting Tips

At some point, you'll run into something strange. We're the first to admit that JTAG-based run-control and trace are not always deterministic. JTAG is a 30-year hardware protocol, and when something goes astray at a very low level, SourcePoint tries to (but sometimes doesn't) recover gracefully. There will be times that the board will power cycle on its own. Or the firmware thinks that a thread is running but gets out of sync with the SourcePoint software, which thinks it's halted. Or the DbCStatus.exe ball stays red instead of turning green, while you swear you have a good DbC connection. Sometimes you have no choice but to quit SourcePoint and power cycle the target. That usually clears up the one-of's. But if the issue is repeatable, we ask that you collect as much information as you can, and open a ticket with us at https://www.asset-intertech.com/support/. We'll respond as soon as possible.

In the meantime, here are a few errata that we've noticed on the UP Xtreme i11, and the steps needed to mitigate.

Firmware gets out of sync with software

On the host PC using DCI, functionality is roughly partitioned between software (SourcePoint application with its GUI) and firmware (lower-level run-control primitives). Broad-brushing it, SourcePoint software on the host communicates with the firmware, that encapsulates JTAG traffic into packets which are sent to the PCH, which in turn performs the JTAG mastering function.

Somewhere along the line, the firmware may get out of sync with the software. You may see symptoms like:

In the Viewpoint window, the threads are shown as Running, whereas the Status Bar at the bottom right shows Stopped.

P0 in the Viewpoint window is Running, with one or more threads below it are in the Stopped state.

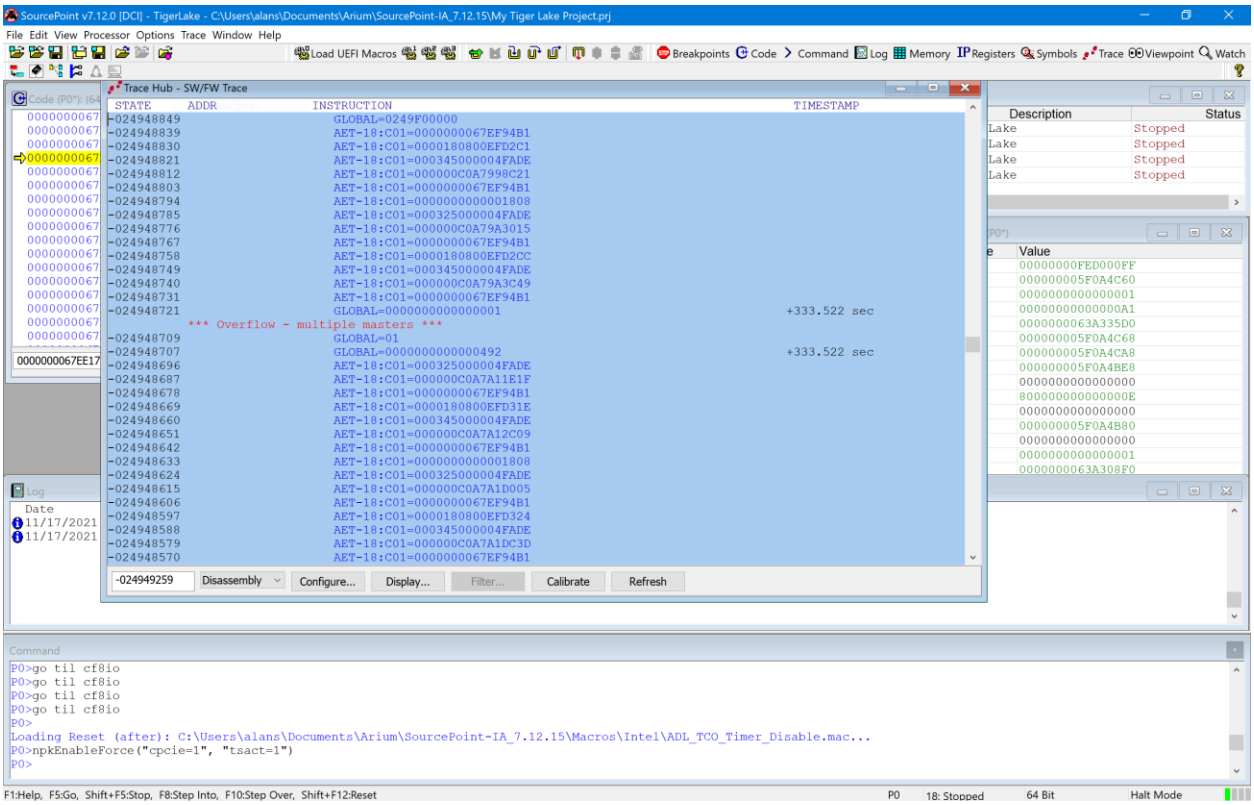
If this happens, you'll likely have to quit SourcePoint, kill the AssetDCI process (see below), power-cycle the target, then start over. Sorry. Then please enter `aalog = 20987` in the `Command` window, and try to reproduce. We're extremely interested in these cases, so capture the verbose logs in the `Log` window and send to us.

Trace buffer overflows

DCI traffic processing has its limitations. When you try to collect too much trace data, the trace buffer overflows, causing aberrant behavior.

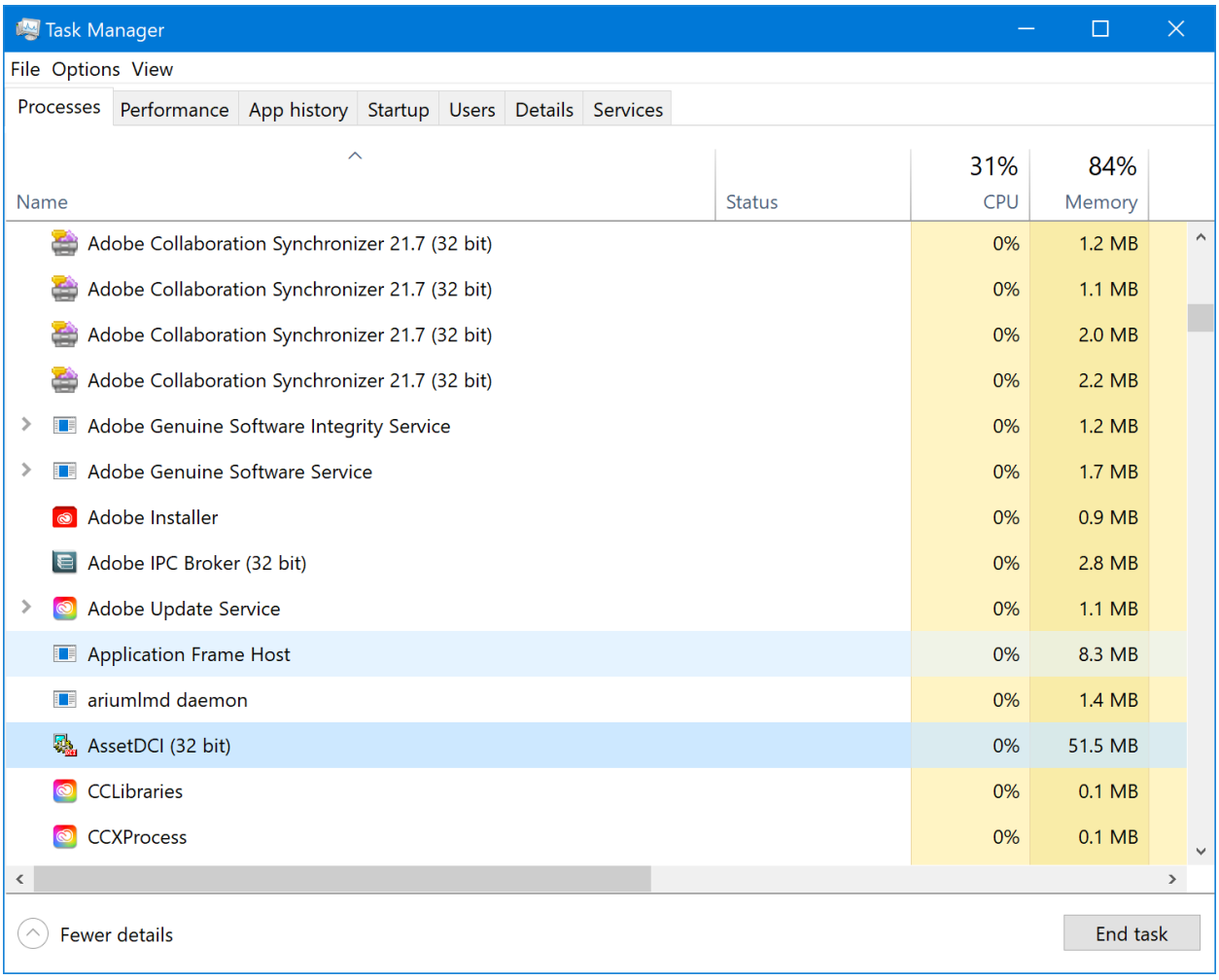
Although the trace data is highly compressed, some trace sources, in particular with AET, running through the Trace Hub can exceed the capacity of the USB 2.0 connection. In theory running at 480Mbps, in practicality SourcePoint can only process trace data at approximately 100Mbps. Beyond that, we collect ~ 20kB of trace data before a buffer in SourcePoint overflows, and we don't recovery gracefully.

You'll see these symptoms of this occurrence in the SW/FW Trace window:



A few of these overflows are no big deal. But, if you're tracing a huge amount of data, SourcePoint may spin, as it tries to process all that data, and deal with the mess. Sometimes, after maybe a few minutes, it recovers. Sometimes, you end up in limbo.

The only solution at this point is to quit SourcePoint, do an `End task` on the AssetDCI Background process, power cycle the target, and start over:

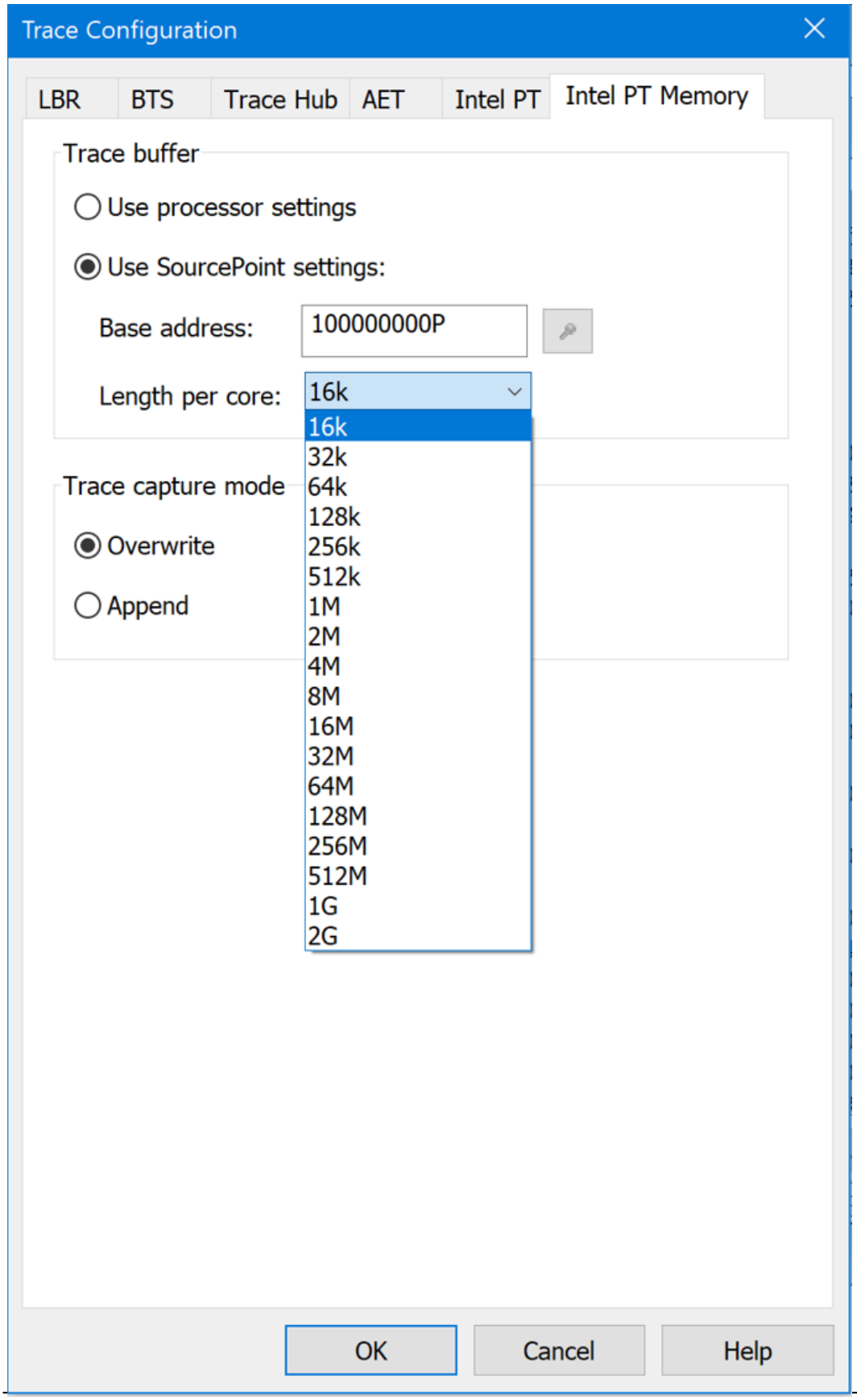


Ultimately, we're working on improving the performing of the AssetDCI driver, and behaving more graciously when overflow is encountered. But, in the interim, it is key to ensure that the trace data collected is relatively "sparse". Focus your debugging in the specific area of interest. Don't try to collect all Port IN/OUT from reset through to the UEFI shell. At 750K I/Os per second, you'll swamp the host debugger processing, and you can't deal with all that data anyway.

Note that this only happens with AET – that is, you can only overflow by collecting too much AET data. It does not apply to LBR, SW/FW Trace, SVEN, or Intel Processor Trace. It may take some trial and error to limit the scope of your event data collection.

Intel Processor Trace – Slow!

When you configure Intel PT, you can specify the size of the buffer in system memory to collect trace data:


Trace Configuration
LBR
BTS
Trace Hub
AET
Intel PT
Intel PT Memory
Trace buffer
Use processor settings
Use SourcePoint settings:
Base address:
100000000P
Length per core:
16k
16k
32k
64k
128k
256k
512k
1M
2M
4M
8M
16M
32M
64M
128M
256M
512M
1G
2G
Trace capture mode
Overwrite
Append
OK
Cancel
Help

Note that Intel PT directs instruction trace data to system memory. Although highly compact and efficient, we are not streaming over DCI to the host in this case; we buffer the code execution data until the platform is halted, at which point SourcePoint uses JTAG over DCI to collect the trace data out of system memory, reconstruct and display it. JTAG operates at fairly low speeds, and for large buffer sizes the transfer of all that data can be slow. It starts to become noticeable beyond a buffer size of 64kB. If you try to collect 1GB of data from system memory over JTAG, you can enjoy a cup of coffee and cake while this is in process. SourcePoint will display as Not Responding, and you'll get the "grey screen" if you're too persistent. SourcePoint will eventually recover, but in the interest of time and frustration, it's probably best not to try to save huge collections of instruction trace. Focus your debug efforts in the vicinity of the bug.

My board is not booting – what now?

Once in a while, especially during an intense debug session, we have found that the target goes into la-la land. You get to the UP splash screen, and then it just stops. Or the screen stays black. SourcePoint run-control continues to work, but it won't boot all the way up to the UEFI shell. Quitting SourcePoint, unplugging the DCI cable, killing the AssetDCI process – all are good steps in this instance, when you need to recover.

But then, once in a while, you wait the needed 20 seconds; and it doesn't boot. The screen stays blank or frozen.

Don't panic! For reasons we're not sure of just yet, after about 60 seconds, the board "wakes up" and should boot all the way to the UEFI shell.

There's only one thing: it has gone back to the factory settings, so you need to reset the WDT timer, as per BIOS Settings section in this manual.

Then, you'll be back in business.

Conclusion

Thank you for getting this far! We hope that you have enjoyed the ride, and are using the power of SourcePoint successfully in your debugging and learning journeys.

Feel free to browse the SourcePoint Academy at https://www.asset-intertech.com/sourcepoint-academy/ for helpful reference guides, help material and "how to" videos.

If you ever have any questions, please call, email or open a Support Case here: https://www.asset-intertech.com/support/. We'll be glad to help!