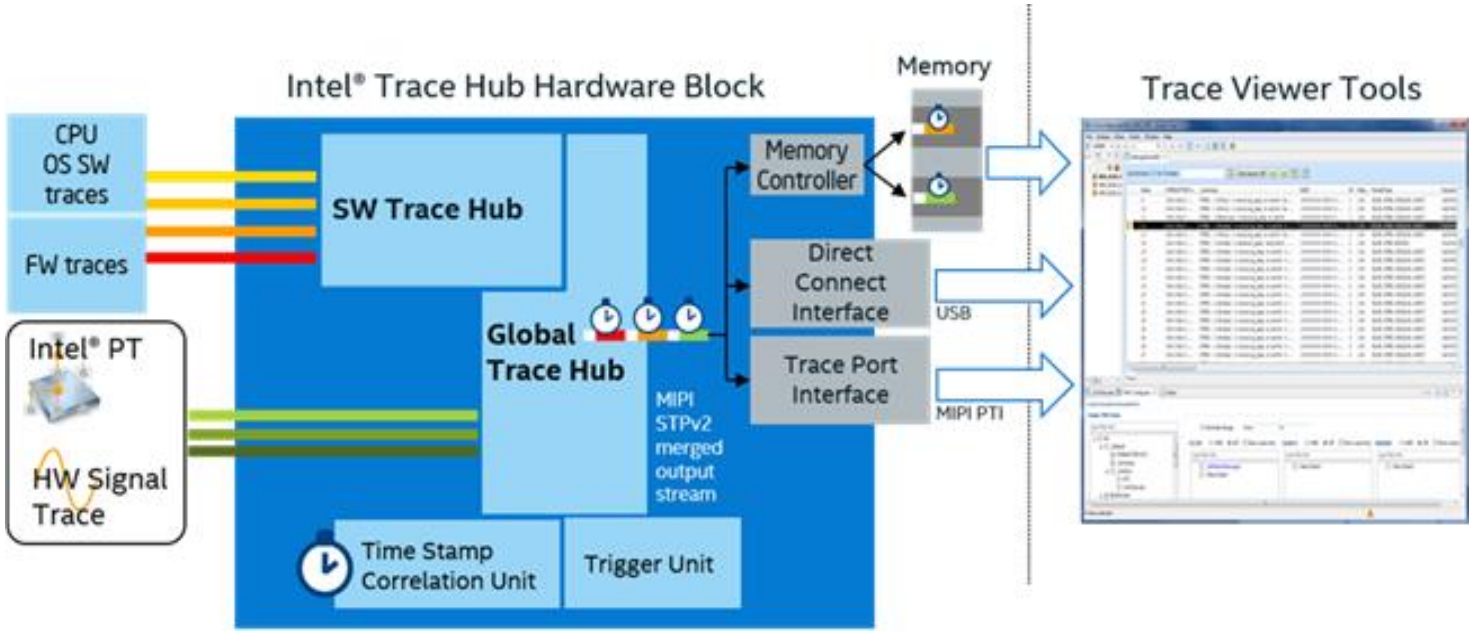
# Intel® Trace Hub usage on Android

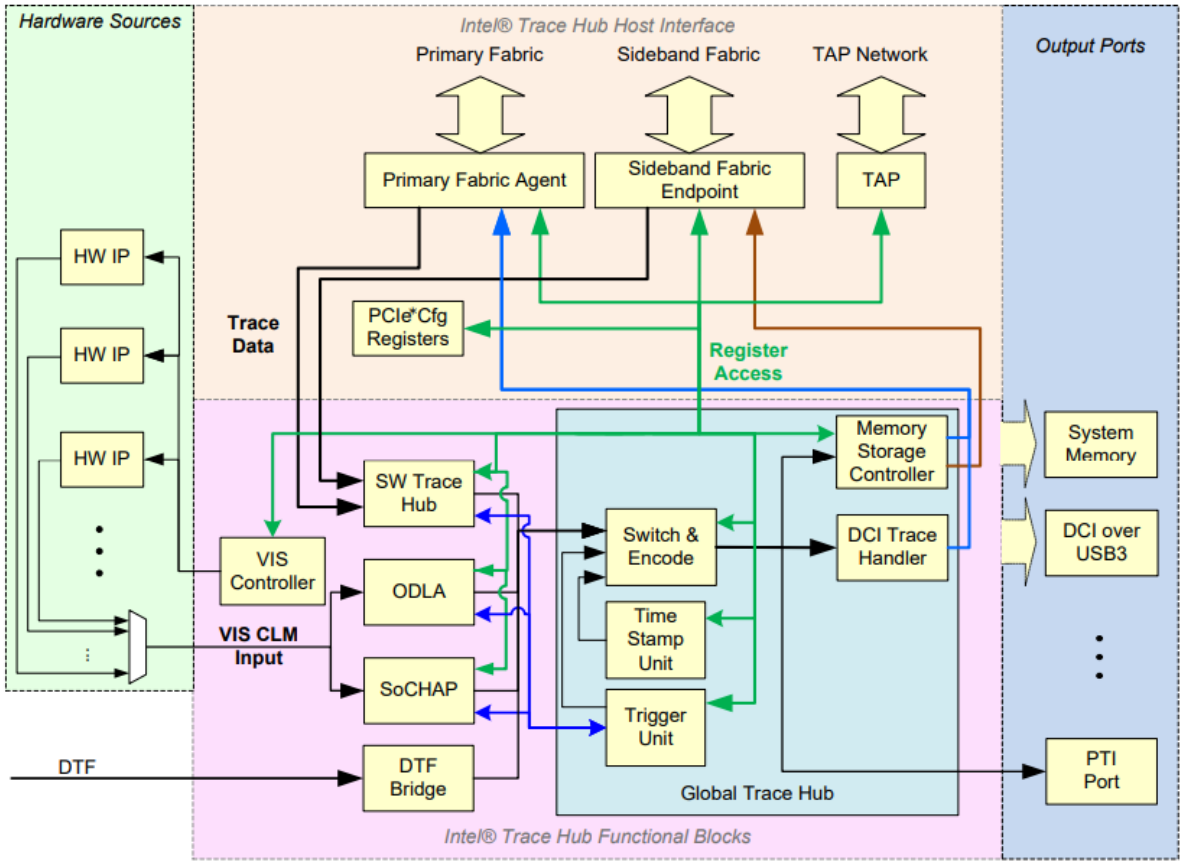
Baofeng, Tian
Aug 20th, 2019

# Intel® Trace Hub overview

Intel(R) Trace Hub (TH) is a set of hardware blocks that collect, switch and output trace data from multiple hardware, software, and firmware sources over several types of trace output ports and is intended to perform full system debugging.



# Intel® Trace Hub high-level architecture




The Intel Trace Hub functional blocks include:
the Software Trace Hub(STH) block
the On-Die Logic Analyzer (ODLA)/VIS Event Recognition (VER) block
the System-on-Chip Performance Counters (SoCHAP) block,
the Visualization of Internal Signals(VIS) Controller block
the DTF Bridge
the Global Trace Hub (GTH) block.

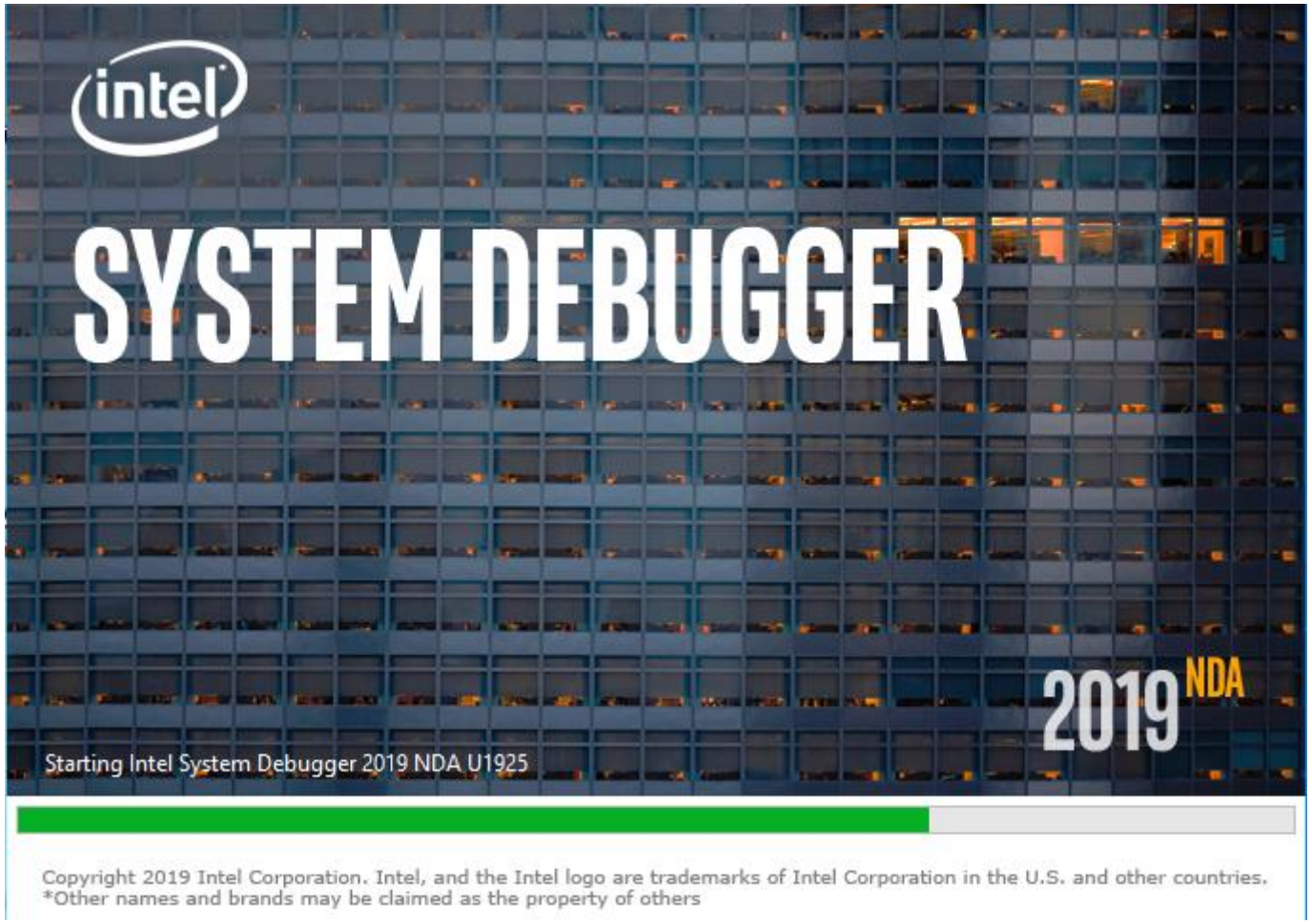
# Intel® Trace Hub – subdevice

Currently, the following Intel TH subdevices (blocks) are supported in kernel:

1. Software Trace Hub (STH), trace source, which is a System Trace Module (STM) device,
2. Memory Storage Unit (MSU), trace output, which allows storing trace hub output in system memory,
3. Parallel Trace Interface output (PTI), trace output to an external debug host via a PTI port,
4. Global Trace Hub (GTH), which is a switch and a central component of Intel(R) Trace Hub architecture.

You can refer to below link for more details:
https://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git/tree/drivers/hwtracing/intel_th?h=v5.3-rc4

# Host tool - Intel® System Debugger



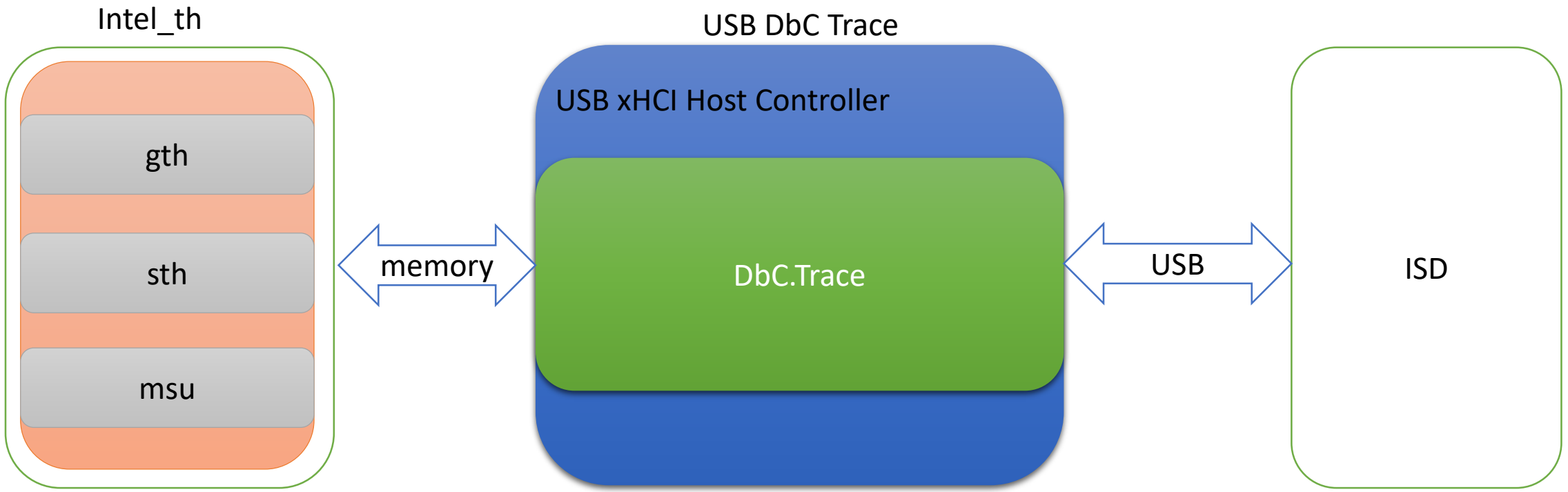

Intel® System Debugger is a modern, GUI-based architecture system software debugger. As such, it presents a view of the architecturally defined system state (processor data structures and registers) along with the software that is executing and interacting with that state (OS kernel, interrupt handlers).

It is supporting in-depth debugging and tracing of Intel® Architecture-based System Software and Embedded Applications.
It enables developers to debug and trace
Intel® Architecture based platforms system-wide,
e.g. UEFI / firmware, System-on-Chip peripheral registers,
OS kernel and drivers with full OS awareness.

# Intel® Trace Hub – use case/usage model

1. Provide New trace debug methods for Intel® Architecture-based System Software platforms.
2. Find root causes faster with full system level traces(include UEFI level tracing).
3. Navigate through instruction trace.
4. Intel Processor Trace supported and displayed with UI based ISD.

# Intel Trace Hub – linux use case



# Intel® Trace Hub – Android problems & solution

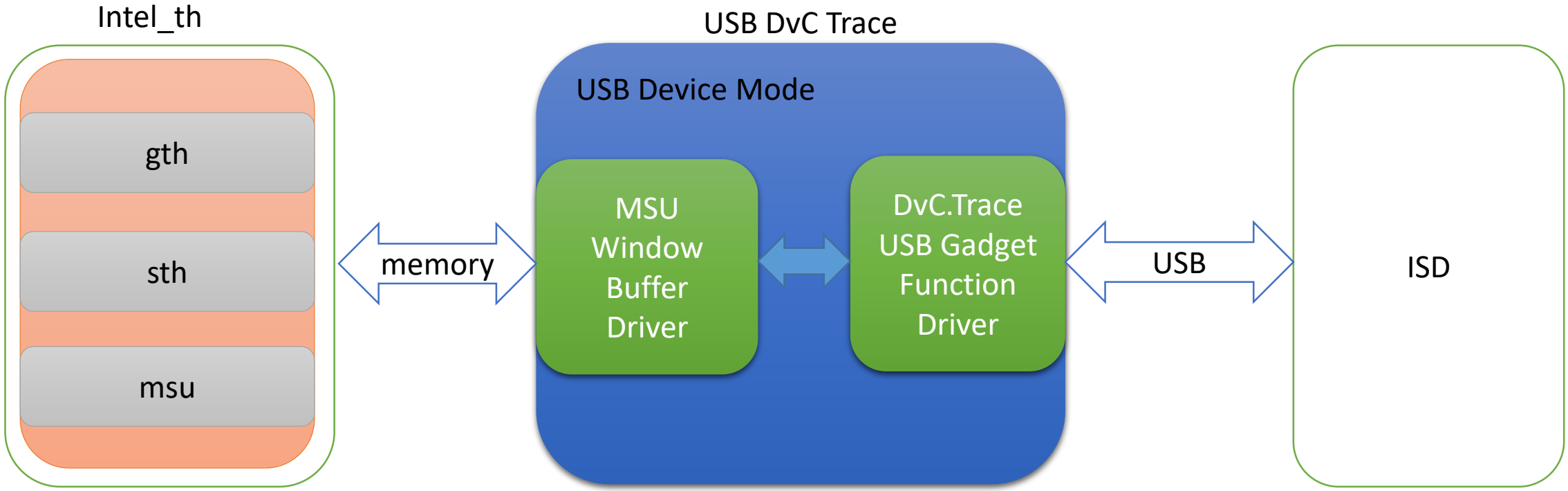
## Status in upstream kernel

In Linux use case, DUT runs in USB host mode, it connect to ISD via USB xHCI DbC.Trace.
This is also what the Intel Trace Hub kernel driver support in upstream kernel.

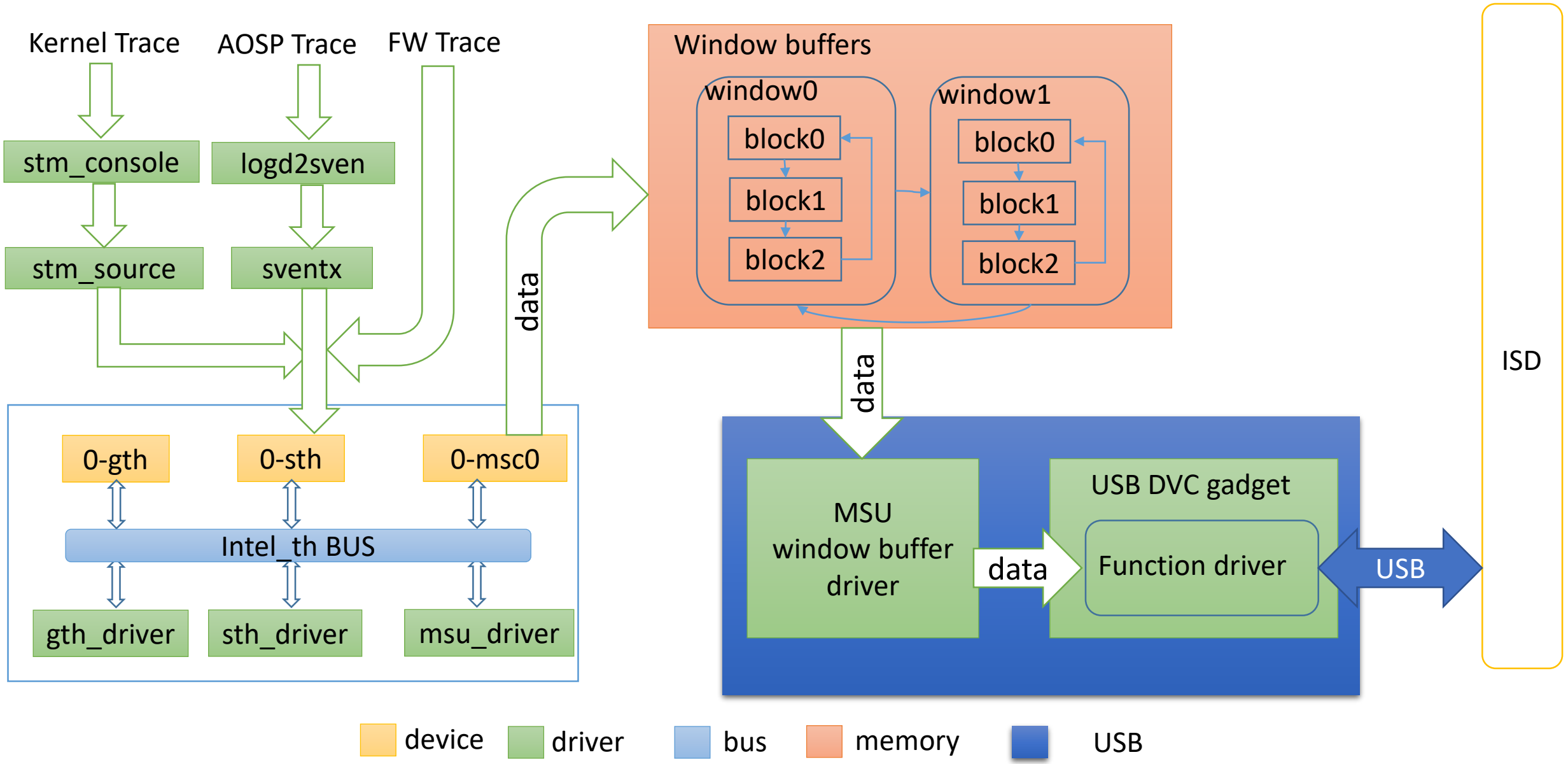
## Android specific solution

- In Android platforms, DUT need to run in USB device mode to support ADB.
  So the DbC.Trace can not be used for trace.
- In USB spec version: 31_072715, USB Debug Class Rev 1 0 Final.pdf, USB device debug capability(DvC) is defined. DvC.Trace runs at USB device mode.
- USB DvC.Trace driver is implemented. It's a bridge between Intel Trace Hub linux driver and the host tool – ISD.

# Intel® Trace Hub – Software architecture/stack



# Intel® Trace Hub – data/module flow



# Intel® Trace Hub – new added modules for DvC

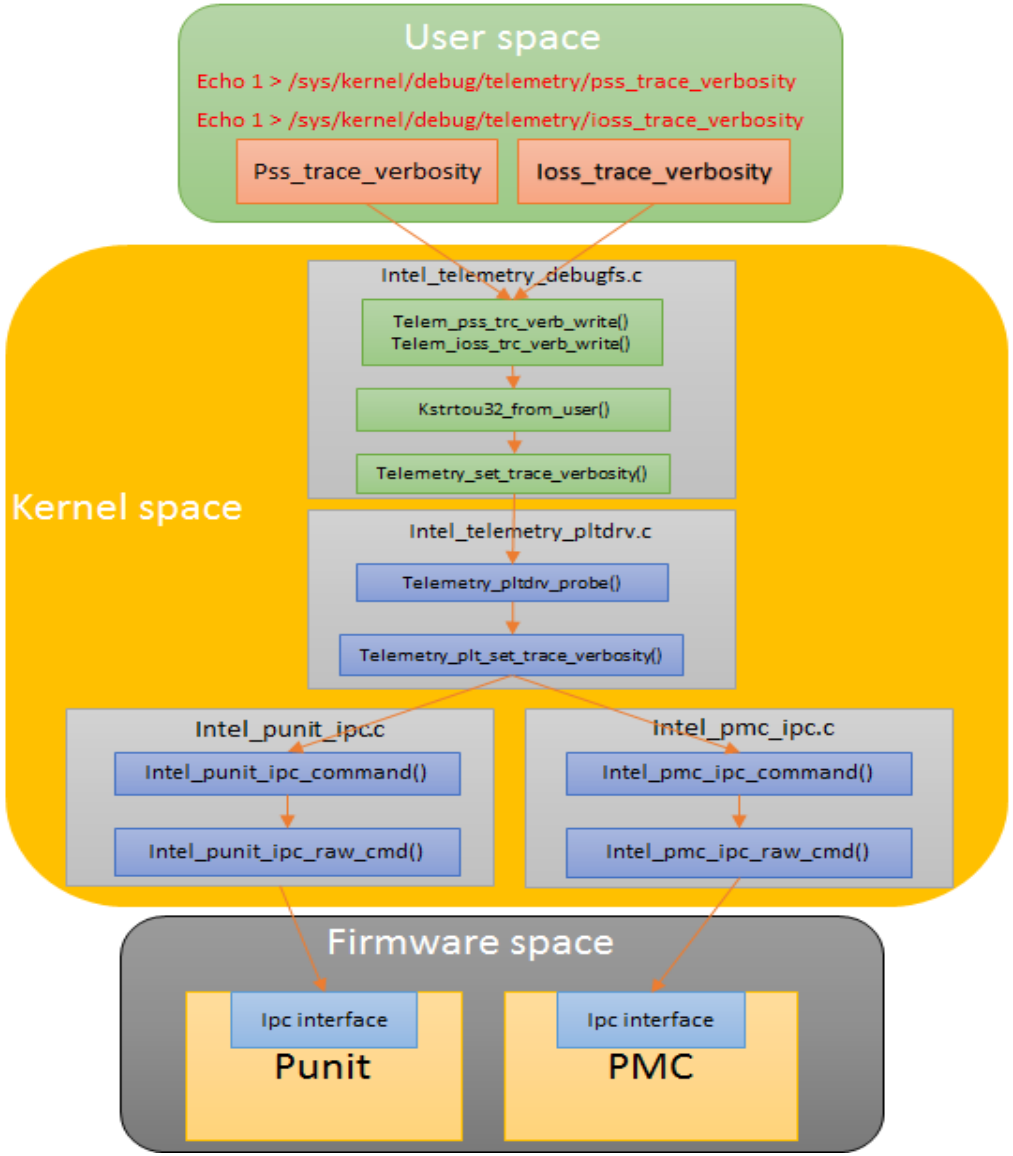
In USB debug class spec: https://www.usb.org/sites/default/files/
documents/usb_debug_class_rev_1_0_final_0.pdf,
DvC(Debug Capability on the USB device, Device Capability) is defined.
This patch implement standard USB DvC gadget driver from above spec.

This patch register a buffer driver to Intel Trace Hub (intel_th) to
get the trace data from system memory. And provide a registration of
buffer users. The USB DvC function will register a user to it to get
trace data.

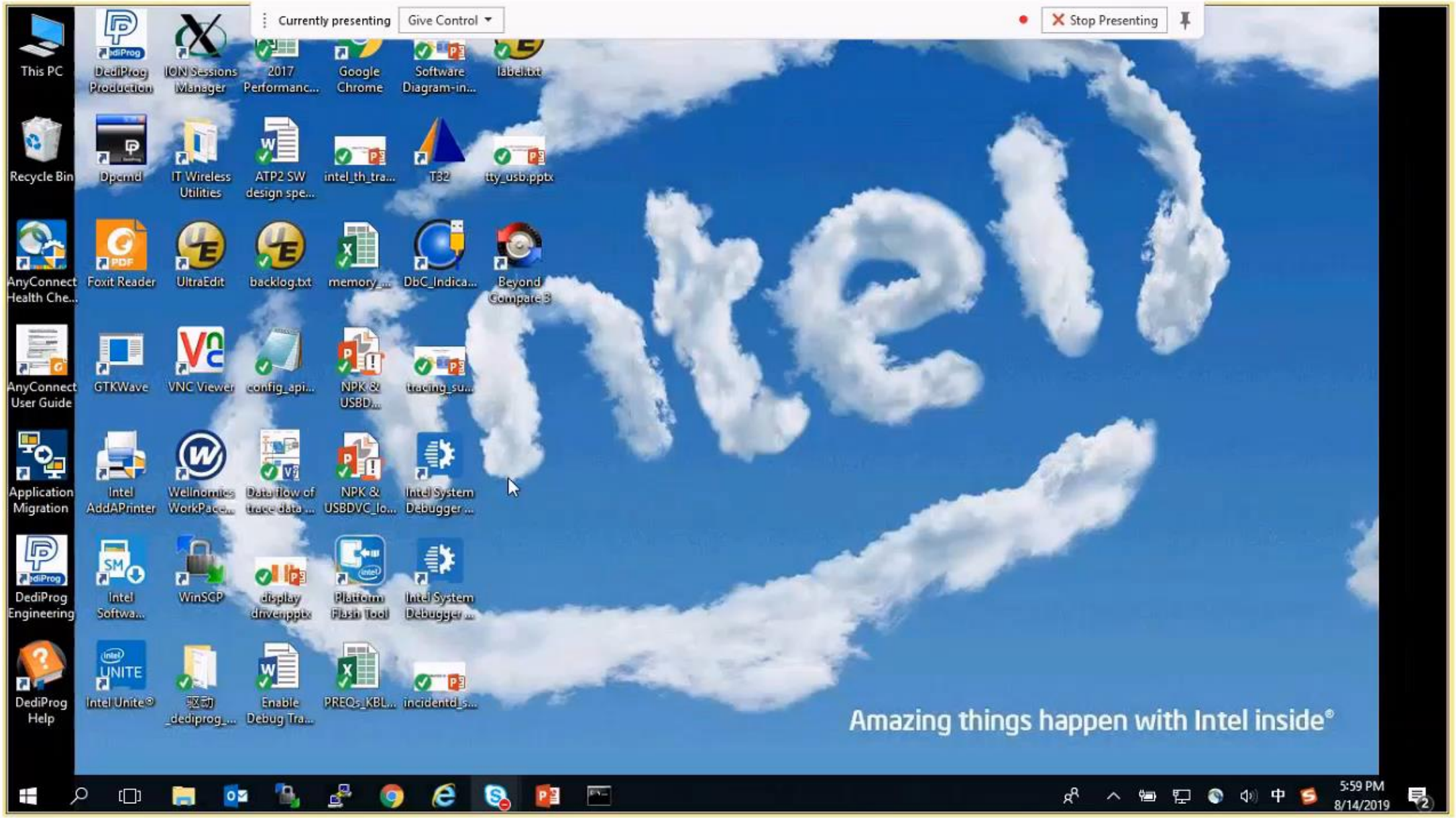
This patch also implement a USB function for USB DvC, and register
it to USB subsystem. It fills descriptors for USB enumeration,
and setup a USB interface with one IN endpoint for data tranferring
to USB host. The standard USB debug class command-"SET_TRACE" is supported.
USB host send this command to start or stop transfer,denpends on the
"wIndex" value of the command.

Signed-off-by: Tian Baofeng <baofeng.tian@intel.com>
---
.../ABI/testing/configfs-usb-gadget-dvctrace | 8 +
MAINTAINERS | 9 +
drivers/usb/gadget/Kconfig | 14 +
drivers/usb/gadget/function/Makefile | 2 +
drivers/usb/gadget/function/f_dvctrace.c | 516 ++++++++++++++++++++
drivers/usb/gadget/function/u_dvctrace_buf.c | 445 +++++++++++++++
drivers/usb/gadget/function/u_dvctrace_buf.h | 26 +
include/linux/usb/debug.h | 184 +++++++
include/uapi/linux/usb/ch9.h | 1 +
9 files changed, 1205 insertions(+)

# An example for trace level control - FW



# Demo of USB DvC.Trace with ISD



# Reference


1. https://www.kernel.org/doc/html/v4.17/trace/intel_th.html
2. https://software.intel.com/sites/default/files/intel-trace-hub-developers-manual-2.1.2.pdf
3. https://software.intel.com/en-us/intel-system-studio-2019-system-debug-user-guide-introducing-the-intel-system-debugger

# Q/A