Synopsys DesignWare Core SuperSpeed USB 3.0 Controller

Author: Felipe Balbi <felipe.balbi@linux.intel.com>
Date: April 2017

Introduction

The *Synopsys DesignWare Core SuperSpeed USB 3.0 Controller* (hereinafter referred to as *DWC3*) is a USB SuperSpeed compliant controller which can be configured in one of 4 ways:

1. Peripheral-only configuration
2. Host-only configuration
3. Dual-Role configuration
4. Hub configuration

Linux currently supports several versions of this controller. In all likelyhood, the version in your SoC is already supported. At the time of this writing, known tested versions range from 2.02a to 3.10a. As a rule of thumb, anything above 2.02a should work reliably well.

Currently, we have many known users for this driver. In alphabetical order:

1. Cavium
2. Intel Corporation
3. Qualcomm
4. Rockchip
5. ST
6. Samsung
7. Texas Instruments
8. Xilinx

Summary of Features

For details about features supported by your version of DWC3, consult your IP team and/or *Synopsys DesignWare Core SuperSpeed USB 3.0 Controller Databook*. Following is a list of features supported by the driver at the time of this writing:

1. Up to 16 bidirectional endpoints (including the control pipe - ep0)
2. Flexible endpoint configuration
3. Simultaneous IN and OUT transfer support
4. Scatter-list support
5. Up to 256 TRBs [1] per endpoint
6. Support for all transfer types (*Control*, *Bulk*, *Interrupt*, and *Isochronous*)
7. SuperSpeed Bulk Streams
8. Link Power Management
9. Trace Events for debugging
10. DebugFS [3] interface

These features have all been exercised with many of the **in-tree** gadget drivers. We have verified both *ConfigFS* [4] and legacy gadget drivers.

Driver Design

The DWC3 driver sits on the *drivers/usb/dwc3/* directory. All files related to this driver are in this one directory. This makes it easy for new-comers to read the code and understand how it behaves.

Because of DWC3's configuration flexibility, the driver is a little complex in some places but it should be rather straightforward to understand.

The biggest part of the driver refers to the Gadget API.

Known Limitations

Like any other HW, DWC3 has its own set of limitations. To avoid constant questions about such problems, we decided to document them here and have a single location to where we could point users.

OUT Transfer Size Requirements

According to Synopsys Databook, all OUT transfer TRBs [1] must have their *size* field set to a value which is integer divisible by the endpoint's *wMaxPacketSize*. This means that *e.g.* in order to receive a Mass Storage *CBW* [5], req->length must either be set to a value that's divisible by *wMaxPacketSize* (1024 on SuperSpeed, 512 on HighSpeed, etc), or DWC3 driver must add a Chained TRB pointing to a throw-away buffer for the remaining length. Without this, OUT transfers will **NOT** start.

Note that as of this writing, this won't be a problem because DWC3 is fully capable of appending a chained TRB for the remaining length and completely hide this detail from the gadget driver. It's still worth mentioning because this seems to be the largest source of queries about DWC3 and *non-working transfers*.

TRB Ring Size Limitation

We, currently, have a hard limit of 256 TRBs [1] per endpoint, with the last TRB being a Link TRB [2] pointing back to the first. This limit is arbitrary but it has the benefit of adding up to exactly 4096 bytes, or 1 Page.

DWC3 driver will try its best to cope with more than 255 requests and, for the most part, it should work normally. However this is not something that has been exercised very frequently. If you experience any problems, see section **Reporting Bugs** below.

Reporting Bugs

Whenever you encounter a problem with DWC3, first and foremost you should make sure that:

1. You're running latest tag from Linus' tree
2. You can reproduce the error without any out-of-tree changes to DWC3
3. You have checked that it's not a fault on the host machine

After all these are verified, then here's how to capture enough information so we can be of any help to you.

Required Information

DWC3 relies exclusively on Trace Events for debugging. Everything is exposed there, with some extra bits being exposed to DebugFS [3].

In order to capture DWC3's Trace Events you should run the following commands **before** plugging the USB cable to a host machine:

```
# mkdir -p /d
# mkdir -p /t
# mount -t debugfs none /d
# mount -t tracefs none /t
# echo 81920 > /t/buffer_size_kb
# echo 1 > /t/events/dwc3/enable
```

After this is done, you can connect your USB cable and reproduce the problem. As soon as the fault is reproduced, make a copy of files `trace` and `regdump`, like so:

```
# cp /t/trace /root/trace.txt
# cat /d/*dwc3*/regdump > /root/regdump.txt
```

Make sure to compress `trace.txt` and `regdump.txt` in a tarball and email it to me with linux-usb in Cc. If you want to be extra sure that I'll help you, write your subject line in the following format:

> **[BUG REPORT] usb: dwc3: Bug while doing XYZ**

On the email body, make sure to detail what you doing, which gadget driver you were using, how to reproduce the problem, what SoC you're using, which OS (and its version) was running on the Host machine.

With all this information, we should be able to understand what's going on and be helpful to you.

Debugging

First and foremost a disclaimer:

```
DISCLAIMER: The information available on DebugFS and/or TraceFS can
change at any time at any Major Linux Kernel Release. If writing
scripts, do **NOT** assume information to be available in the
current format.
```

With that out of the way, let's carry on.

If you're willing to debug your own problem, you deserve a round of applause :-)

Anyway, there isn't much to say here other than Trace Events will be really helpful in figuring out issues with DWC3. Also, access to Synopsys Databook will be **really** valuable in this case.

A USB Sniffer can be helpful at times but it's not entirely required, there's a lot that can be understood without looking at the wire.

Feel free to email me and Cc linux-usb if you need any help.

DebugFS

`DebugFS` is very good for gathering snapshots of what's going on with DWC3 and/or any endpoint.

On DWC3's `DebugFS` directory, you will find the following files and directories:

`ep[0..15]{in,out}/ link_state regdump testmode`

link_state

When read, `link_state` will print out one of `U0`, `U1`, `U2`, `U3`, `SS.Disabled`, `RX.Detect`, `SS.Inactive`, `Polling`, `Recovery`, `Hot Reset`, `Compliance`, `Loopback`, `Reset`, `Resume` or `UNKNOWN link state`.

This file can also be written to in order to force link to one of the states above.

regdump

File name is self-explanatory. When read, `regdump` will print out a register dump of DWC3. Note that this file can be grepped to find the information you want.

testmode

When read, `testmode` will print out a name of one of the specified USB 2.0 Testmodes (`test_j`, `test_k`, `test_se0_nak`, `test_packet`, `test_force_enable`) or the string `no test` in case no tests are currently being executed.

In order to start any of these test modes, the same strings can be written to the file and DWC3 will enter the requested test mode.

ep[0..15]{in,out}

For each endpoint we expose one directory following the naming convention `ep$num$dir` *(ep0in, ep0out, ep1in, ...)*. Inside each of these directories you will find the following files:

`descriptor_fetch_queue event_queue rx_fifo_queue rx_info_queue rx_request_queue transfer_type trb_ring tx_fifo_queue tx_request_queue`

With access to Synopsys Databook, you can decode the information on them.

transfer_type

When read, `transfer_type` will print out one of `control`, `bulk`, `interrupt` or `isochronous` depending on what the endpoint descriptor says. If the endpoint hasn't been enabled yet, it will print `--`.

trb_ring

When read, `trb_ring` will print out details about all TRBs on the ring. It will also tell you where our enqueue and dequeue pointers are located in the ring:

```
buffer_addr,size,type,ioc,isp_imi,csp,chn,lst,hwo
000000002c754000,481,normal,1,0,1,0,0,0
000000002c75c000,481,normal,1,0,1,0,0,0
000000002c780000,481,normal,1,0,1,0,0,0
000000002c788000,481,normal,1,0,1,0,0,0
000000002c78c000,481,normal,1,0,1,0,0,0
000000002c754000,481,normal,1,0,1,0,0,0
000000002c75c000,481,normal,1,0,1,0,0,0
000000002c784000,481,normal,1,0,1,0,0,0
000000002c788000,481,normal,1,0,1,0,0,0
000000002c78c000,481,normal,1,0,1,0,0,0
000000002c790000,481,normal,1,0,1,0,0,0
000000002c758000,481,normal,1,0,1,0,0,0
000000002c780000,481,normal,1,0,1,0,0,0
000000002c788000,481,normal,1,0,1,0,0,0
000000002c790000,481,normal,1,0,1,0,0,0
000000002c758000,481,normal,1,0,1,0,0,0
000000002c780000,481,normal,1,0,1,0,0,0
000000002c784000,481,normal,1,0,1,0,0,0
000000002c788000,481,normal,1,0,1,0,0,0
000000002c78c000,481,normal,1,0,1,0,0,0
000000002c754000,481,normal,1,0,1,0,0,0
000000002c758000,481,normal,1,0,1,0,0,0
000000002c780000,481,normal,1,0,1,0,0,0
000000002c784000,481,normal,1,0,1,0,0,0
000000002c78c000,481,normal,1,0,1,0,0,0
000000002c790000,481,normal,1,0,1,0,0,0
000000002c758000,481,normal,1,0,1,0,0,0
```

```
000000002c780000,481,normal,1,0,1,0,0,0
000000002c788000,481,normal,1,0,1,0,0,0
000000002c790000,481,normal,1,0,1,0,0,0
000000002c758000,481,normal,1,0,1,0,0,0
000000002c780000,481,normal,1,0,1,0,0,0
000000002c788000,481,normal,1,0,1,0,0,0
000000002c790000,481,normal,1,0,1,0,0,0
000000002c758000,481,normal,1,0,1,0,0,0
000000002c780000,481,normal,1,0,1,0,0,0
000000002c788000,481,normal,1,0,1,0,0,0
000000002c790000,481,normal,1,0,1,0,0,0
000000002c758000,481,normal,1,0,1,0,0,0
000000002c780000,481,normal,1,0,1,0,0,0
000000002c788000,481,normal,1,0,1,0,0,0
000000002c790000,481,normal,1,0,1,0,0,0
000000002c758000,481,normal,1,0,1,0,0,0
000000002c780000,481,normal,1,0,1,0,0,0
000000002c788000,481,normal,1,0,1,0,0,0
000000002c790000,481,normal,1,0,1,0,0,0
000000002c758000,481,normal,1,0,1,0,0,0
000000002c780000,481,normal,1,0,1,0,0,0
000000002c788000,481,normal,1,0,1,0,0,0
000000002c790000,481,normal,1,0,1,0,0,0
000000002c758000,481,normal,1,0,1,0,0,0
000000002c780000,481,normal,1,0,1,0,0,0
000000002c788000,481,normal,1,0,1,0,0,0
000000002c790000,481,normal,1,0,1,0,0,0
000000002c758000,481,normal,1,0,1,0,0,0
000000002c780000,481,normal,1,0,1,0,0,0
000000002c78c000,481,normal,1,0,1,0,0,0
000000002c784000,481,normal,1,0,1,0,0,0
000000002c788000,481,normal,1,0,1,0,0,0
000000002c78c000,481,normal,1,0,1,0,0,0
000000002c754000,481,normal,1,0,1,0,0,0
000000002c758000,481,normal,1,0,1,0,0,0
000000002c780000,481,normal,1,0,1,0,0,0
000000002c788000,481,normal,1,0,1,0,0,0
000000002c790000,481,normal,1,0,1,0,0,0
000000002c758000,481,normal,1,0,1,0,0,0
000000002c780000,481,normal,1,0,1,0,0,0
000000002c758000,481,normal,1,0,1,0,0,0
000000002c780000,481,normal,1,0,1,0,0,0
000000002c78c000,481,normal,1,0,1,0,0,0
000000002c75c000,481,normal,1,0,1,0,0,0
000000002c78c000,481,normal,1,0,1,0,0,0
000000002c780000,481,normal,1,0,1,0,0,0
000000002c754000,481,normal,1,0,1,0,0,0
000000002c788000,481,normal,1,0,1,0,0,0
000000002c754000,481,normal,1,0,1,0,0,0
000000002c780000,481,normal,1,0,1,0,0,0
000000002c788000,481,normal,1,0,1,0,0,0
000000002c78c000,481,normal,1,0,1,0,0,0
000000002c790000,481,normal,1,0,1,0,0,0
000000002c754000,481,normal,1,0,1,0,0,0
000000002c758000,481,normal,1,0,1,0,0,0
000000002c75c000,481,normal,1,0,1,0,0,0
000000002c780000,481,normal,1,0,1,0,0,0
000000002c784000,481,normal,1,0,1,0,0,0
000000002c788000,481,normal,1,0,1,0,0,0
000000002c78c000,481,normal,1,0,1,0,0,0
000000002c790000,481,normal,1,0,1,0,0,0
000000002c754000,481,normal,1,0,1,0,0,0
000000002c758000,481,normal,1,0,1,0,0,0
000000002c75c000,512,normal,1,0,1,0,0,1        D
0000000000000000,0,UNKNOWN,0,0,0,0,0,0       E
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
```

```
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
```

```
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
0000000000000000,0,UNKNOWN,0,0,0,0,0,0
00000000381ab000,0,link,0,0,0,0,0,1
```

Trace Events

DWC3 also provides several trace events which help us gathering information about the behavior of the driver during runtime.

In order to use these events, you must enable `CONFIG_FTRACE` in your kernel config.

For details about how enable DWC3 events, see section **Reporting Bugs**.

The following subsections will give details about each Event Class and each Event defined by DWC3.

MMIO

It is sometimes useful to look at every MMIO access when looking for bugs. Because of that, DWC3 offers two Trace Events (one

for dwc3_readl() and one for dwc3_writel()). `TP_printk` follows:

```
TP_printk("addr %p value %08x", __entry->base + __entry->offset,
              __entry->value)
```

Interrupt Events

Every IRQ event can be logged and decoded into a human readable string. Because every event will be different, we don't give an example other than the `TP_printk` format used:

```
TP_printk("event (%08x): %s", __entry->event,
              dwc3_decode_event(__entry->event, __entry->ep0state))
```

Control Request

Every USB Control Request can be logged to the trace buffer. The output format is:

```
TP_printk("%s", dwc3_decode_ctrl(__entry->bRequestType,
                                 __entry->bRequest, __entry->wValue,
                                 __entry->wIndex, __entry->wLength)
)
```

Note that Standard Control Requests will be decoded into human-readable strings with their respective arguments. Class and Vendor requests will be printed out a sequence of 8 bytes in hex format.

Lifetime of a `struct usb_request`

The entire lifetime of a `struct usb_request` can be tracked on the trace buffer. We have one event for each of allocation, free, queueing, dequeueing, and giveback. Output format is:

```
TP_printk("%s: req %p length %u/%u %s%s%s ==> %d",
      __get_str(name), __entry->req, __entry->actual, __entry->length,
      __entry->zero ? "Z" : "z",
      __entry->short_not_ok ? "S" : "s",
      __entry->no_interrupt ? "i" : "I",
      __entry->status
)
```

Generic Commands

We can log and decode every Generic Command with its completion code. Format is:

```
TP_printk("cmd '%s' [%x] param %08x --> status: %s",
      dwc3_gadget_generic_cmd_string(__entry->cmd),
      __entry->cmd, __entry->param,
      dwc3_gadget_generic_cmd_status_string(__entry->status)
)
```

Endpoint Commands

Endpoints commands can also be logged together with completion code. Format is:

```
TP_printk("%s: cmd '%s' [%d] params %08x %08x %08x --> status: %s",
      __get_str(name), dwc3_gadget_ep_cmd_string(__entry->cmd),
      __entry->cmd, __entry->param0,
      __entry->param1, __entry->param2,
      dwc3_ep_cmd_status_string(__entry->cmd_status)
)
```

Lifetime of a `TRB`

A `TRB` Lifetime is simple. We are either preparing a `TRB` or completing it. With these two events, we can see how a `TRB` changes over time. Format is:

```
TP_printk("%s: %d/%d trb %p buf %08x%08x size %s%d ctrl %08x (%c%c%c%c:%c%c:%s)",
      __get_str(name), __entry->queued, __entry->allocated,
      __entry->trb, __entry->bph, __entry->bpl,
      ({char *s;
      int pcm = ((__entry->size >> 24) & 3) + 1;
      switch (__entry->type) {
      case USB_ENDPOINT_XFER_INT:
      case USB_ENDPOINT_XFER_ISOC:
              switch (pcm) {
              case 1:
                      s = "1x ";
                      break;
              case 2:
                      s = "2x ";
                      break;
```

```
                case 3:
                        s = "3x ";
                        break;
                }
        default:
                s = "";
        } s; }),
        DWC3_TRB_SIZE_LENGTH(__entry->size), __entry->ctrl,
        __entry->ctrl & DWC3_TRB_CTRL_HWO ? 'H' : 'h',
        __entry->ctrl & DWC3_TRB_CTRL_LST ? 'L' : 'l',
        __entry->ctrl & DWC3_TRB_CTRL_CHN ? 'C' : 'c',
        __entry->ctrl & DWC3_TRB_CTRL_CSP ? 'S' : 's',
        __entry->ctrl & DWC3_TRB_CTRL_ISP_IMI ? 'S' : 's',
        __entry->ctrl & DWC3_TRB_CTRL_IOC ? 'C' : 'c',
      dwc3_trb_type_string(DWC3_TRBCTL_TYPE(__entry->ctrl))
)
```

Lifetime of an Endpoint

And endpoint's lifetime is summarized with enable and disable operations, both of which can be traced. Format is:

```
TP_printk("%s: mps %d/%d streams %d burst %d ring %d/%d flags %c:%c%c%c%c%c:%c:%c",
        __get_str(name), __entry->maxpacket,
        __entry->maxpacket_limit, __entry->max_streams,
        __entry->maxburst, __entry->trb_enqueue,
        __entry->trb_dequeue,
        __entry->flags & DWC3_EP_ENABLED ? 'E' : 'e',
        __entry->flags & DWC3_EP_STALL ? 'S' : 's',
        __entry->flags & DWC3_EP_WEDGE ? 'W' : 'w',
        __entry->flags & DWC3_EP_TRANSFER_STARTED ? 'B' : 'b',
        __entry->flags & DWC3_EP_PENDING_REQUEST ? 'P' : 'p',
        __entry->flags & DWC3_EP_END_TRANSFER_PENDING ? 'E' : 'e',
        __entry->direction ? '<' : '>'
)
```

Structures, Methods and Definitions

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\usb\(linux-master)(Documentation)(driver-api)(usb)dwc3.rst, line 687)

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/usb/dwc3/core.h
   :doc: main data structures
   :internal:
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\usb\(linux-master)(Documentation)(driver-api)(usb)dwc3.rst, line 691)

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/usb/dwc3/gadget.h
   :doc: gadget-only helpers
   :internal:
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\usb\(linux-master)(Documentation)(driver-api)(usb)dwc3.rst, line 695)

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/usb/dwc3/gadget.c
   :doc: gadget-side implementation
   :internal:
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\usb\(linux-master)(Documentation)(driver-api)(usb)dwc3.rst, line 699)

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/usb/dwc3/core.c
   :doc: core driver (probe, PM, etc)
   :internal:
```

[1] (*1*,*2*,*3*) Transfer Request Block

[2] Transfer Request Block pointing to another Transfer Request Block.

[3] (*1*,*2*) The Debug File System

[4] The Config File System

[5] Command Block Wrapper