TD

{{#switchcategory:MSP430=<McuHitboxHeader/>|C2000=<McuHitboxHeader/>|Stellaris=<McuHitboxHeader/>|TMS570=<McuHitboxHeader/>|MCU=<McuHitboxHeader/>|MAVRK=<MAVRKHitboxHeader/>|<HitboxHeader/>}}

Contents

Introduction

This page describes usage of TD utility. TD utility is delivered as part of Code Composer Studio (CCS) and used to decode hardware trace data into a human readable text format. This is a command line utility is typically used in a standalone or scripting environment.

Location

This utility gets installed in CCS_DIR/ccsv5/ccs_base/emulation/analysis/bin directory as part of Code Composer Studio installation. To use this utility, open a command line shell and change directory to CCS_DIR/ccsv5/ccs_base/emulation/analysis/bin.

Parameters

Required

-bin <Full path to binary file>
This parameter is used to specify the trace binary file.

-app <Full path to out file>
This parameter is used to specify full path of the application location. Trace decoders use application information to decompress and output the trace information.
If using software messages from STM, this parameter needs to be specified with CPU qualification (i.e. *CPU=file)*. More than one of such entries can be specified with comma(,) seperations e.g. CPU0=file0,cpu1=file1,...

It is sufficient to give just the name after the slash (e.g., C66XX_0) as CPU name, it is not required to use full CPU name as formed in CCS (e.g., Blackhawk XDS560v2-USB Mexxanine Emulator_0/C66XX_0).

If there is only one processor generating STM SW messages, CPU qualification could be skipped

-procid <proc>
This parameter is used to specify processor or trace type. Valid options - 64x, 64x+, 66x, 55x, 55x+, arm7, arm9, arm11, cortexa8, and stm.

Note: If you are using CCSv4, this parameter name would be **-cpuid.**

-rcvr <receiver name>

This parameter is used to specify receiver type used for trace collection. Valid options - ETB, 560T, 560V2, and Pro

Required for STM (Optional for DSP and ARM/Cortex trace)

-devicekey DeviceID
This parameter is used to define the device ID or device key for identifying STM topology in the device. This is a unique ID (TAP ID) and can be read via R0 of ICEPICK or retreived from the device TRM.

A trace device XML file path could also be used by the advanced users .

Examples:

Device id for

OMAP4430 ES2	-- 0x1B85202F
OMAP4430 ES2.1	-- 0x3B95C02F
OMAP4460	-- 0x0B94E02F
OMAP4470	-- 0x0B97502F
C6678	-- 0x0009E02F
C6670	-- 0x0009D02F
Turbo-C6670	-- 0x0b94102F
AM335x	-- 0x0B94402F
AM386x	-- 0x0B96802F
AM387x	-- 0x0B8F202F
DM816x	-- 0x0B81E02F

Note: If you are using CCSv4, this parameter name would be **-deviceid**.

-dcmfile <Full path to the metadata file>
This parameter is used to pass additional metadata to help decode. Default is none.
For STM ETB, you could create a *foo.dcm* configuration file by copying the text in the box below into a text file and saving it with a name like *foo.dcm* or whatever you prefer.

```
STM_data_flip=1
STM_Buffer_Wrapped=0
HEAD_Present_0=0
HEAD_Pointer_0=
HEAD_Present_1=1
HEAD_Pointer_1=4
```

STM_data_flip=1 --> This field tells decoder that the data is flipped in receiver packing. This is 1 if using ET. Otherwise, its 0.

STM_Buffer_Wrapped=0 --> This field tells decoder that the buffer is wrapped or not. 1 is wrapped, 0 for buffer not wrapped.

HEAD_Present_0=0 --> STM ETB pointer references to efficiently detect start of the buffer (if buffer wrapped). Default to 0 unless you know the HEAD information.

HEAD_Pointer_0=0 --> STM ETB pointer references to efficiently detect start of the buffer (if buffer wrapped). Default to 0 unless you know the HEAD information.

HEAD_Present_1=0 --> STM ETB pointer references to efficiently detect start of the buffer (if buffer wrapped). Default to 0 unless you know the HEAD information.

HEAD_Pointer_1=0 --> STM ETB pointer references to efficiently detect start of the buffer (if buffer wrapped). Default to 0 unless you know the HEAD information.

Optional

-output <Output file name>

This parameter is used to specify output file name. Defaul is stdout.

-columns <coloum names>
Comma seperated column names to include in the output. Default is all the columns.You can use td *-coulminfo* with all the required parameters to get a list of supported columns.

-seekpos <% position>
Trace decode start potition in % with respect to the trace data binary file. Default is 0.

-samples <count>
Total number of requested output samples. Default is max available.

-format <output format>
Output format id. One of the following id can be specified. Default is CSV_NO_TPOS_QUOTE.

CSV_QUOTE,

CSV,

CSV_SEMICOLON,

CSV_SEMICOLON_NO_TPOS,

CSV_NO_TPOS_QUOTE,

CSV_NO_TPOS,

CSV_SEMICOLON_NO_TPOS,

LOG

Note: TPOS --> Trace position information, which is an decoder specific and unique internal identification for trace output samples.

-deviceidfile < .dim file >
This is a CCS or cToolsTrace scripting generated metadata file to help decode context data. Default is none.

-decodetime
Prints time before and after the trace decode. No other functional imlications.

Examples

```
td -procid 64x+ -bin trace.bin -app may_app.out -rcvr 560T -format CSV_NO_TPOS_QUOTE -columns "Program Address,Cycles,Trace Status" -timestamp abs -seekpos 10 -samples 100000
```

```
td -procid 64x+ -bin trace.bin -app may_app.out -rcvr 560T -format CSV_NO_TPOS -timestamp delta
```

```
td -procid 64x+ -bin trace.bin -app may_app.out -rcvr ETB -format CSV_SEMICOLON -columns "Program Address,Cycles,Trace Status" -timestamp abs -seekpos 10 -samples 100000
```

```
td -procid 66x -bin trace.bin -app may_app.out -rcvr ETB -format CSV_SEMICOLON -columns "Program Address,Cycles,Trace Status" -timestamp abs -seekpos 10 -samples 100000
```

```
td -procid cortexa8 -bin c:/temp/ETM_etbdata.bin -app C:/Examples/bin/etm_csetb_A8.c6a816x.out -rcvr ETB -dcmfile c:/temp/ETM_etbdata.dcm -format CSV_SEMICOLON -columns "Program Address,Cycles,Trace Status" -timestamp abs
```

```
td -procid stm -bin trace.bin -app CortexA9_0=CortexA9File.out,CortexM3_0=CortexM3File.out -rcvr 560V2STM -devicekey 0x0B81E02F
```

```
td -procid stm -bin trace.bin -app CortexA9_0=CortexA9File.out -rcvr 560V2STM -deviceid device_omap4430_ES2.xml
```

```
td -procid stm -bin trace.bin -app CortexA9_0=CortexA9File.out,CortexM3_0=CortexM3File.out -rcvr 560V2STM -deviceidfile foo.dim
```

```
td -procid stm -bin trace.bin -app CortexA8_0=CortexA8File.out -rcvr 560V2STM -devicekey 0x0B81E02F
```

```
td -procid stm -bin trace.bin -app C66X_0=corepac0.out -rcvr ETB -devicekey 0x0009D02F
```

Output

The out of TD is a text file or stdout with the decoded trace infirmation. Here is an example of the decoded trace output from the TD.

```
Program Address; Load Address; Program Data; Cycles; Trace Status
==========================================================
--------------; --------------; --------------; --------; Start of trace
00802368; 00802368; 00000427; 000001; PC collection off, Timing collection off
008022b0; 008022b0; 0000dc0d; 000005; PC collection on, Timing collection off
008022b2; 008022b2; 00002401; 000001; ----
008022b4; 008022b4; 0000dc05; 000001; ----
008022b6; 008022b6; 00002c6e; 000002; ----
```

```
008022b8; 008022b8; 0191c028; 000001; ----
008022c0; 008022c0; 01804068; 000001; ----
008022c4; 008022c4; 023c22e4; 000001; ----
```

Related

- CToolsLib
- Embedded Trace Buffer or ETB

{{ 1. switchcategory:MultiCore= - For technical support on MultiCore devices, please post your questions in the C6000 MultiCore Forum - For questions related to the BIOS MultiCore SDK (MCSDK), please use the BIOS Forum Please post only comments related to the article **TD** here.	Keystone= - For technical support on MultiCore devices, please post your questions in the C6000 MultiCore Forum - For questions related to the BIOS MultiCore SDK (MCSDK), please use the BIOS Forum Please post only comments related to the article **TD** here.	C2000=*For technical support on the C2000 please post your questions on The C2000 Forum. Please post only comments about the article* ***TD*** *here.*	DaVinci=*For technical support on DaVincoplease post your questions on The DaVinci Forum. Please post only comments about the article* ***TD*** *here.*	MSP430=*For technical support on MSP430 please post your questions on The MSP430 Forum. Please post only comments about the article* ***TD*** *here.*	OMAP35x=*For technical support on OMAP please post your questions on The OMAP Forum. Please post only comments about the article* ***TD*** *here.*	OMAPL1=*For technical support on OMAP please post your questions on The OMAP Forum. Please post only comments about the article* ***TD*** *here.*	MAVRK=*For technical support on MAVRK please post your questions on The MAVRK Toolbox Forum. Please post only comments about the article* ***TD*** *here.*	*For technical s please post you questions at http://e2e.ti.cor Please post on comments abo article* ***TD*** *here.* }}

Links

Amplifiers & Linear
Audio
Broadband RF/IF & Digital Radio
Clocks & Timers
Data Converters

DLP & MEMS
High-Reliability
Interface
Logic
Power Management

Processors

- ARM Processors
- Digital Signal Processors (DSP)
- Microcontrollers (MCU)
- OMAP Applications Processors

Switches & Multiplexers
Temperature Sensors & Control ICs
Wireless Connectivity

{{#switchcategory:MSP430=<McuHitboxFooter/>|C2000=<McuHitboxFooter/>|Stellaris=<McuHitboxFooter/>|TMS570=<McuHitboxFooter/>|MCU=<McuHitboxFooter/>|MAVRK=<MAVRKHitboxFooter/>|<HitboxFooter/>}}

Retrieved from "https://processors.wiki.ti.com/index.php?title=TD&oldid=129709"

This page was last edited on 11 January 2013, at 17:43.