CoreSight Access Tool for SoC600 (CSAT600) User Guide

Version 1.0

Non-Confidential

Issue 01

102584_0100_01_en

CoreSight Access Tool for SoC600 (CSAT600) User Guide

Release information

Document history

Issue	Date	Confidentiality	Change
0100-01	21 September 2021	Non-Confidential	Initial release

Proprietary Notice

This document is protected by copyright and other related rights and the practice or implementation of the information contained in this document may be protected by one or more patents or pending patent applications. No part of this document may be reproduced in any form by any means without the express prior written permission of Arm. No license, express or implied, by estoppel or otherwise to any intellectual property rights is granted by this document unless specifically stated.

Your access to the information in this document is conditional upon your acceptance that you will not use or permit others to use the information for the purposes of determining whether implementations infringe any third party patents.

THIS DOCUMENT IS PROVIDED "AS IS". ARM PROVIDES NO REPRESENTATIONS AND NO WARRANTIES, EXPRESS, IMPLIED OR STATUTORY, INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF MERCHANTABILITY, SATISFACTORY QUALITY, NON-INFRINGEMENT OR FITNESS FOR A PARTICULAR PURPOSE WITH RESPECT TO THE DOCUMENT. For the avoidance of doubt, Arm makes no representation with respect to, and has undertaken no analysis to identify or understand the scope and content of, patents, copyrights, trade secrets, or other rights.

This document may include technical inaccuracies or typographical errors.

TO THE EXTENT NOT PROHIBITED BY LAW, IN NO EVENT WILL ARM BE LIABLE FOR ANY DAMAGES, INCLUDING WITHOUT LIMITATION ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL, PUNITIVE, OR CONSEQUENTIAL DAMAGES, HOWEVER CAUSED AND REGARDLESS OF THE THEORY OF LIABILITY, ARISING OUT OF ANY USE OF THIS DOCUMENT, EVEN IF ARM HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

This document consists solely of commercial items. You shall be responsible for ensuring that any use, duplication or disclosure of this document complies fully with any relevant export laws and regulations to assure that this document or any portion thereof is not exported, directly

Arm Limited. Company 02557590 registered in England.

110 Fulbourn Road, Cambridge, England CB1 9NJ.

(LES-PRE-20349|version 21.0)

Confidentiality Status

This document is Non-Confidential. The right to use, copy and disclose this document may be subject to license restrictions in accordance with the terms of the agreement entered into by Arm and the party that Arm delivered this document to.

Unrestricted Access is an Arm internal classification.

Product Status

The information in this document is Final, that is for a developed product.

Feedback

Arm® welcomes feedback on this product and its documentation. To provide feedback on the product, create a ticket on https://support.developer.arm.com

To provide feedback on the document, fill the following survey: https://developer.arm.com/documentation-feedback-survey.

Inclusive language commitment

Arm values inclusive communities. Arm recognizes that we and our industry have used language that can be offensive. Arm strives to lead the industry and create change.

This document includes language that can be offensive. We will replace this language in a future issue of this document.

To report offensive language in this document, email terms@arm.com.

Contents

1. Overview

CoreSight Access Tool for SoC600 (CSAT600) provides access to Arm Debug Interface Architecture Specification ADIv6.0 or CoreSight SoC-600 targets. The CSAT600 tool is used to interact with CoreSight SoC-600 targets at a CoreSight architecture level. This level of interaction is useful when trying to debug target behavior at a low debug architectural level.

This user guide provides information on how to use the CSAT600 tool, and how the CSAT600 tool commands compare to the commands in the original CSAT tool. The guide also explains what CSAT600 tool commands are available and how to use them, and provides examples demonstrating common CSAT600 tool use cases.

Before you begin

To use the CSAT600 tool, you need:

- Arm Development Studio 2019.0 or later.
- A working knowledge of the Arm Debug Interface Architecture Specification ADIv6.0.
- If you are working with an additional probe which does not have built-in CSAT600 support, a vendor provided probe definition file and RDDI library file.
- A CoreSight SoC-600 target.

The CSAT600 tool differs from the original CoreSight Access Tool (CSAT). This is because CSAT only works with Arm Debug Interface Architecture Specification ADIv5.2 or earlier CoreSight targets. Wherever possible, the command syntax of the CSAT600 tool mimics the command syntax of CSAT.

2. How to use a CSAT600

The CSAT600 tool is only available in Arm Development Studio 2019.0 or later.

To use the CSAT600 tool:

1. Open a command prompt to **<Arm Development Studio installation directory>*
2. Run:

```
csat -cs600
```

You should see the following output in the command prompt:

```
**********************************
**  Welcome to CSAT for SoC600  **
**********************************

%>
```

Note

To run the original CSAT tool, run `csat` in an Arm Development Studio bin directory command prompt.

3. CSAT600 command list

For Arm Development Studio 2019.0, the following commands listed on this page are available for the CSAT600 tool. Additional commands will be available in future Arm Development Studio releases. You should be aware that:

- Most commands have a command alias that is used to execute the same operation. Any command aliases are in () beside the full command name.
- In the command's Example section, "…" denotes that executing the command provides further output. The further output is not shown in this user guide.

autodetect (auto)

Autodetect which CoreSight devices are present on the target.

To run this command, you must have a debug probe connection. See the instructions at connect (con) for details on how to connect to your debug probe.

Syntax

```
autodetect (auto) [enum-aps] [read-rom]
```

Where:

```
enum-aps
```

Enumerates the autodetected Access Ports (APs).

```
read-rom
```

Returns the contents of the read ROM table.

Example

```
%> autodetect
Detecting platform...
------------+---------------------------------+---------
 Device No. | DTSL Device Name (& Address)    | AP No.
------------+---------------------------------+---------
          0 |                      ARMCS-DP   |

% > auto enum-aps
Detecting platform...
------------+---------------------------------+---------
 Device No. | DTSL Device Name (& Address)    | AP No.
------------+---------------------------------+---------
          0 |                      ARMCS-DP   |
          1 |         CSMEMAP_0 (0x00000000) | 0
          2 |         CSMEMAP_1 (0x00000000) | 1

% > autodetect read-rom
Detecting platform...
------------+---------------------------------+---------
 Device No. | DTSL Device Name (& Address)    | AP No.
------------+---------------------------------+---------
```

```
          0 |                         ARMCS-DP  |
          1 |        CSMEMAP_0 (0x00000000) | 0
          2 |      Cortex-A53_0 (0x80410000) | 0
...
```

cfgbox (cfb)

Get or set the debug probe configuration.

To run this command, you must have a debug probe connection. See the instructions at connect (con) for details on how to connect to your debug probe.

Syntax

```
cfgbox (cfb) [< item_name > [< item_value >]]
```

Where:

```
item_name
```

Debug probe configuration item to be configured.

```
item_value
```

Value to set the debug probe configuration item to.

Example

```
% > cfgbox
Linked_SRST_TRST     : 0
SRSTOnConnect        : 0
UserOutputPin  s     : 000000
UseDeprecatedSWJ     : 0
...

% > cfb Linked_SRST_TRST
Linked_SRST_TRST: 0
Description: Set TRUE if the target hardware has these two signals physically
 linked.
Type: Boolean
Default Value: 0
Minimum Value: 0
Maximum Value: 1
Enum Values: 0 - False, 1 - True
Access: Read-Write

% >
> cfb Linked_SRST_TRST 1
Config item Linked_SRST_TRST was set to 1
```

cfgtplate (cfg)

Get or set debug probe configuration items for the current CoreSight device.

To run this command, you must have both an active debug probe connection, and an active CoreSight device connection. See the instructions at `connect (con)` for details on how to connect to your debug probe. See the instructions at devopen (dvo, device) for details on how to connect to a Coresight device.

Syntax

```
cfgtplate (cfg) [< item_name > [< item_value >]]
```

Where:

`item_name`

Debug probe configuration item to be configured.

`item_value`

Value to set the debug probe configuration item to.

Example

```
%> cfgtplate
CTI_SYNCH_START      : Boolean : 0
ALLOW_EXECUTION_WITHOUT_T_BIT : Boolean : 0
POST_RESET_STATE     : Enum    : 1
CORESIGHT_DP_MEMSPACE : Boolean : 0
...

%> cfg POST_RESET_STATE
POST_RESET_STATE: 1
Description: Determines if the core should halt or run after reset
Type: Enum
Default Value: 1
Minimum Value: 0
Maximum Value: 1
Enum Values: 0 - Running, 1 - Stopped
Access: Read-Write

%> cfg POST_RESET_STATE 1
Config item POST_RESET_STATE was set to 1
```

chain (chn)

Set or autodetect the JTAG scan chain and clock frequency.

To run this command, you must have a debug probe connection. See the instructions at `connect (con)` for details on how to connect to your debug probe.

Syntax

```
chain (chn) [dev=device_name1,...,device_nameN | dev=auto] [clk=< freqHz > | clk=A]
```

Where:

`dev`

Use this to specify one or more devices on the JTAG scan chain. Alternatively, set to auto to autodetect all of the devices on the JTAG scan chain.

`clk`

Value to set the debug adapter clock frequency to in Hz or A to use adaptive clocking.

Example

```
% > chain dev=ARMCS-DP
------------+--------------------------------+---------
 Device No. | DTSL Device Name (& Address)   | AP No.
------------+--------------------------------+---------
          0 |                      ARMCS-DP  |

% > chn dev=ARMCS-DP clk=10000000

------------+--------------------------------+---------
 Device No. | DTSL Device Name (& Address)   | AP No.
------------+--------------------------------+---------
          0 |                      ARMCS-DP  |

% > chain dev=auto

Detecting scanchain...

------------+--------------------------------+---------
 Device No. | DTSL Device Name (& Address)   | AP No.
------------+--------------------------------+---------
          0 |                      ARMCS-DP  |
```

connect (con)

Connect to a probe via TCP or USB.

It is possible to connect to a probe without specifying an address (TCP or USB option). This is relevant to additional probes where the connection address is specified in RDDI library file and the capability “ConnectionAddressRequired” is set to False in the probe definition file. See the instructions at loadprobes(load) for the details on how to load additional probes. See the instructions at listprobes (probes) for details on how to list available probes.

Syntax

```
connect (con) [< probe type >:]TCP:< hostname > | TCP:< ip address > | USB [config
file]
```

Where:

`probe type`

Specify the probe type you wish to connect to or use to change the current probe type. To change the probe type, place the probe `type` in front of the `TCP` or `USB` option. See the instructions at `loadprobes (load)` and `listprobes (probes)` for the details on how to load and list additional available probes respectively.

`TCP`

Use this option if you are connecting to your debug probe using an Ethernet connection. You must specify either the hostname or the IP address.

`USB`

Use this option if you are connecting to your debug probe through USB.

`config file`

Specify the System Description File (SDF) from a platform configuration used when connecting to the target. If an SDF is not supplied, you must run a CoreSight device discovery command. See instructions at `autodetect (auto)` or `chain (chn)` for details on discovering CoreSight devices.

Example

```
% > connect TCP:255.255.255.255
Connecting to TCP:255.255.255.255 ...
Connected to: DSTREAM
Base H/W: V2 Rev C-00
TurboTAP Rev: 0.16
DSTREAM Probe V1 Rev B-00
Firmware: 5.4.0, Build 5
Configuration file: C:\Users\<user>\AppData\Local\Temp
\csat_scanchain_devices5451335090165749803.sdf

% > con USB
Connecting to USB ...
Connected to: DSTREAM
Base H/W: V2 Rev C-00
TurboTAP Rev: 0.16
DSTREAM Probe V1 Rev B-00
Firmware: 5.4.0, Build 5
Configuration file: C:\Users\<user>\AppData\Local\Temp
\csat_scanchain_devices713844846634569063.sdf

% > con TCP:myDSTREAM C:\work\target.sdf
Connecting to TCP:myDSTREAM ...
Connected to: DSTREAM
Base H/W: V2 Rev C-00
TurboTAP Rev: 0.16
DSTREAM Probe V1 Rev B-00
Firmware: 5.4.0, Build 5
Configuration file: C:\work\target.sdf

% > connect DSTREAM-ST:TCP:myDSTREAM-ST
Probe type was set to DSTREAM-ST.
Connecting to TCP:myDSTREAM-ST ...
Connected to: DSTREAM-ST
Base H/W: V2 Rev A-06
FPGA build 0x0014, Debug 1V8, Trace 1V8
Firmware: 5.4.0, Build 5
Configuration file: C:\Users\< user >\AppData\Local\Temp
\csat_scanchain_devices713844846634569063.sdf
```

devclose (dvc)

Close the connection to the CoreSight device.

To run this command, you must have both an active debug probe connection, and an active CoreSight device connection. See the instructions at `connect (con)` for details on how to connect to your debug probe. See the instructions at `devopen (dvo, device)` for details on how to connect to a Coresight device.

Syntax

```
devclose (dvc) [< device no >]
```

Where:

```
device no
```

CoreSight device number to close the connection to. Use the `autodetect (auto)`, `chain (chn)`, or `list (l)` commands to determine the device number.

Example

```
%> devclose
Disconnected from device no. 2

%> dvc 2
Disconnected from device no. 2
```

devopen (dvo, device)

Open a connection to a CoreSight device.

To run this command, you must have a debug probe connection with an SDF or a debug probe connection and have discovered CoreSight devices. See the instructions at `connect (con)` for details on how to connect to your debug probe. See the instructions at `autodetect (auto)` or `chain (chn)` for details on discovering CoreSight devices.

Syntax

```
devopen (dvo, device) < device no >
```

Where:

```
device no
```

CoreSight device number to open a connection to. Use the `autodetect (auto)`, `chain (chn)`, or `list (l)` commands to determine the device number.

Example

```
% > devopen 2
Connected to device no. 2: Cortex-A53_0, JTAG ID: 0x1ba06477, version 0x00000006
Msg returned from device: Cortex-A53 Template

% > dvo 0
Connected to device no. 0: ARMCS-DP, JTAG ID: 0x1ba06477, version 0x00000006
Msg returned from device: ARM-DP Template using Rv-Msg.
```

```
% > device 2
Connected to device no. 2: Cortex-M3, JTAG ID: 0x3ba00477, version 0x00000006
Msg returned from device: Cortex M3 template
```

disconnect (dcn)

Disconnect from the debug probe.

To run this command, you must have a debug probe connection. See the instructions at `connect (con)` for details on how to connect to your debug probe.

Syntax

```
disconnect (dcn)
```

Example

```
%> disconnect
Disconnected from TCP:255.255.255.255

% > dcn
Disconnected from USB
```

dpregread (drr)

Read a register from a DP, AP, or other device.

To run this command, you must have a debug probe connection with an SDF or a debug probe connection and have discovered CoreSight devices. See the instructions at `connect (con)` for details on how to connect to your debug probe. See the instructions at `autodetect (auto)` or `chain (chn)` for details on discovering CoreSight devices.

Syntax

```
dpregread (drr) [< device >.]<id>
```

Where:

`device`

Specify an offset for the currently connected device or override the currently connected device and specify the `device` explicitly. Optionally, device can specify one of the following:

- A device number as displayed by the `list (l)` command.
- The device name from the device list.
- The number of a memory access port.
- Just `DP` for a Debug Port.

Note

If the target contains multiple Debug Ports (DPs), the DP alias cannot be used and the device number must be used instead.

`id>`

Either specify a textual alias for AP or DP registers as specified by the Arm CoreSight System-on-Chip SoC-600 Technical Reference Manual or a raw CoreSight register identifier (ID). This ID is the offset specified in the TRM for the CoreSight device divided by 4.

Example

```
% > dpregread AP0.CSW
Device no. 2 is active.
Reading from device no. 2: CSMEMAP
AP0:0x340 : 0x03000052

%> dpregread CSMEMAP.CSW
Device no. 2 is active.
Reading from device no. 2: CSMEMAP
CSMEMAP:0x340 : 0x03000052

%> dpregread DP.DPIDR
Device no. 1 is active.
Reading from device no. 1: ARMCS-DP
DP:0x2080 : 0x00000000

%> dpregread device2.TAR
Device no. 2 is active.
Reading from device no. 2: CSMEMAP
device2:0x341 : 0x77441100

%> dpregread device2.0x340
Device no. 2 is active.
Reading from device no. 2: CSMEMAP
device2:0x340 : 0x03000052

%> drr TAR
Reading from device no. 2: CSMEMAP
device2:0x341 : 0x77441100
```

dpregwrite (drw)

Write a register from a DP, AP, or other device.

To run this command, you must have a debug probe connection with an SDF or a debug probe connection and have discovered CoreSight devices. See the instructions at `connect (con)` for details on how to connect to your debug probe. See the instructions at `autodetect (auto)` or `chain (chn)` for details on discovering CoreSight devices.

Syntax

```
dpregwrite (drw) [< device >.]<id> < value >
```

Where:

`device`

Specify an offset for the currently connected device or override the currently connected device and specify the device explicitly. Optionally, device can specify one of the following:

- A device number as displayed by the `list (l)` command.
- The device name from the device list.
- The number of a memory access port.
- Just DP for a Debug Port.

Note

If the target contains multiple Debug Ports (DPs), the DP alias cannot be used and the device number must be used instead.

`id`

Either specify a textual alias for AP or DP registers as specified by the Arm CoreSight System-on-Chip SoC-600 Technical Reference Manual or a raw CoreSight register identifier (ID). This ID is the offset specified in the TRM for the CoreSight device divided by 4.

`value`

32 bit number value to write to the specified register.

Example

```
% > dpregwrite AP0.TAR 0xEFC4AFC0
Device no. 1 is active.
Writing to device no. 1: CSMEMAP_0


% > dpregwrite device44.TAR 256
Device no. 44 is active.
Writing to device no. 44: CSMEMAP_2


% > drw device44.0x341 0xF00F
Device no. 44 is active.
Writing to device no. 44: CSMEMAP_2
```

exit (x)

Close the CSAT600 program.

Syntax

```
exit (x)
```

Example

```
%> exit
Disconnected from TCP:255.255.255.255

% > x
Disconnected from TCP:255.255.255.255
```

help (h)

List all the available CSAT600 commands, or display the help information for a specific command.

Syntax

`help (h) <command>`

Where:

`command`

Command name to display help content for.

Example

```
% > help
autodetect (auto)        : Autodetect which CoreSight devices are present on the
 target.
cfgbox     (cfb)         : Configure the DSTREAM probe.
cfgtplate  (cfg)         : Get or set DSTREAM configuration items for the current
 CoreSight device.
...

% > h autodetect
Command: autodetect
        Autodetect which CoreSight devices are present on the target.
Aliases:
        auto
Usage:
        autodetect [enum-aps] [read-rom]
        auto [enum-aps] [read-rom]
```

list (l)

List the available CoreSight devices.

To run this command, you must have a debug probe connection with an SDF or a debug probe connection and have discovered CoreSight devices. See the instructions at `connect (con)` for details on how to connect to your debug probe. See the instructions at `autodetect (auto)` or `chain (chn)` for details on discovering CoreSight devices.

Syntax

`list (l)` Example

```
% > list
```

```
------------+--------------------------------+---------
 Device No. | DTSL Device Name (& Address)   | AP No.
------------+--------------------------------+---------
          0 |                       ARMCS-DP |
          1 |          CSMEMAP_0 (0x00000000) | 0
          2 |       Cortex-A53_0 (0x80410000) | 0
          3 |            CSCTI_0 (0x80420000) | 0
...
```

listprobes (probes)

List available probe types.

Run this command to list built-in and additional probes supported by CSAT600. See the instructions at `loadprobes (load)` for the details on how to load additional probes.

Note

The current probe is marked with '*'.

Syntax

```
listprobes (probes)
```

Example

```
%> listprobes
* DSTREAM          built-in
  DSTREAM-HT        built-in
  DSTREAM-PT        built-in
  DSTREAM-ST        built-in
  MyProbe           from c:\work\probes.xml
  RealView ICE      built-in
```

loadprobes (load)

Load an additional probe from a probe definition file.

To run this command, you must have a probe vendor provided probe definition file and RDDI library file. A probe definition file is an XML file which defines the probe name and RDDI library file. The XML file might also contain configuration items and capabilities. The RDDI library file might be provided in both Windows and Linux variants.

Syntax

```
loadprobes (load) <xml file>
```

Where:

`xml file`

Path to the probe definition XML file.

Example

log log

Control logging.

Syntax

```
log on | off | < filename >
```

Where:

`on`

Turns on logging.

`off`

Turns off logging.

`filename`

Path and filename to save the logging to.

Example

```
% > log on
Logging is enabled with logfile: C:\Users\<user>\AppData\Local\Temp
\csat7441679604713525243.log

% > log on C:\Users\< user >\AppData\Local\Temp\csat.log
Logging is enabled with logfile: C:\Users\< user >\AppData\Local\Temp\csat.log

% > log off
Logging is disabled.
```

memread (mr)

Read memory from the specified address.

To run this command, you must have a debug probe connection with an SDF or a debug probe connection and have discovered CoreSight devices. See the instructions at `connect (con)` for details on how to connect to your debug probe. See the instructions at `autodetect (auto)` or `chain (chn)` for details on discovering CoreSight devices.

Syntax

```
memread (mr) <address> < number of words to read > [rule=< memory operation modifiers
>]
```

Where:

`address`

Address to read from.

`number of words to read`

Number of words to read starting at the specified address. A word is 32 bits or 4 bytes.

`rule` Used to set the rule parameter of a memory access operation. The rule value varies depending on the DTSL device type being used and the associated debug probe functionality that uses it. This parameter is most useful for Memory Access Port (MEMAP) accesses where it can be used to modify specific parts of the AP's `CSW` register. The rule parameter might have more specialist uses for some core device accesses in certain circumstances.

For an AHB device, the rule field maps to the 5 bits of the `HPROT` field.

For an AXI-AP device, the encoding of the rule field is more complex, and consists of:

- `Mode (4 bits) << 0x10`
- `Domain (3 bits) << 0x8`
- `Ace bit << 0x7`
- `PROT (3 bits) << 0x4`
- `CACHE ( 4bits)` See the Arm Debug Interface Architecture Specification ADIv6.0 for details of what effect these values have on the accesses using the above memory buses.

Example

```
%> memread 0x80540000 8
Reading from device no. 1: CSMEMAP_0
0x80540000 : 0x00000000
0x80540004 : 0x00000000
0x80540008 : 0x00000000
0x8054000c : 0x00000003
0x80540010 : 0x00000001
0x80540014 : 0x00000000
0x80540018 : 0x00000000
0x8054001c : 0x00000000


%> mr 0x80540000 8 rule=0
Reading from device no. 1: CSMEMAP_0
0x80540000 : 0x00000000
0x80540004 : 0x00000000
0x80540008 : 0x00000000
0x8054000c : 0x00000003
0x80540010 : 0x00000001
0x80540014 : 0x00000000
0x80540018 : 0x00000000
0x8054001c : 0x00000000
```

memwrite (mw)

Write memory to the specified address.

To run this command, you must have a debug probe connection with a SDF or a debug probe connection and have discovered CoreSight devices. See `connect (con)` for details on how to connect to your debug probe. See `autodetect (auto)` or `chain (chn)` for details on discovering CoreSight devices.

Syntax

```
memwrite (mw) <address> <data>...[dataN]* [rule=< memory operation modifiers >]
```

Where:

`address`

Address to write to.

`data...dataN`

Data to write starting at the specified address.

`rule` Used to set the rule parameter of a memory access operation. The `rule` value varies depending on the DTSL device type that is being used and the associated debug probe functionality that uses it. This parameter is most useful for Memory Access Port (MEMAP) accesses where it can be used to modify specific parts of the CSW register of the AP. The rule parameter might have more specialist uses for some core device accesses in certain circumstances.

For an AHB device, the rule field maps to the 5 bits of the HPROT field.

For an AXI-AP device, the encoding of the rule field is more complex, and consists of:

- `Mode (4 bits) << 0x10`
- `Domain (3 bits) << 0x8`
- `Ace bit << 0x7`
- `PROT (3 bits) << 0x4`
- `CACHE ( 4bits)`

See the Arm Debug Interface Architecture Specification ADIv6.0 for details of what effect these values have on the accesses using the above memory buses.

Example

```
%> memwrite 0x80540004 1
Writing to device no. 1: CSMEMAP_0
Wrote 4 bytes.


%> mw 0x80540004 0 rule=0
Writing to device no. 1: CSMEMAP_0
Wrote 4 bytes.
```

setprobe (probe)

Set the active probe type.

To use this command for an additional probe, you must have loaded the additional probe's definition XML file. See `loadprobes (load)` for details on how to load an additional probe from a probe definition XML file.

Note

By default, the active probe is set to DSTREAM. In order to connect to a different probe, the probe type must be changed accordingly using this command. See the instructions at `listprobes (probes)` for the details on how to list the available probe types.

Syntax

```
setprobe (probe) < probe type >
```

Where:

```
probe type
```

set the probe type to be used. Use the `listprobes (probes)` to view the available probe types.

Example

```
%> setprobe DSTREAM
Probe type was set to DSTREAM.

%> setprobe MyProbe
Probe type was set to MyProbe.
```

unloadprobes (unload)

Remove previously loaded additional probe from the probe definition file.

To run this command, you must have loaded a probe vendor provided probe definition file and RDDI library file. See the instructions for `loadprobes (load)` for details on how to load additional probes. See the instructions at `listprobes (probes)` for the details on how to list the available probe types.

Syntax

```
unloadprobes (unload) <xml file>
```

Where:

```
xml file
```

Path to the probe definition XML file.

Example

```
%> unloadprobes c:\work\probes.xml
Probe MyProbe was unloaded.
```

4. Command comparison between CSAT600 and CSAT

CSAT600 and CSAT use the same command names and formats where possible. The following table shows the command differences between CSAT600 and CSAT.

Note

`<device no>` used in CSAT600 has the same meaning as `<devid>` used in CSAT.

CSAT600 Command	CSAT Command	Differences
`chain (chn) [dev= device_name1,...,device_nameN | dev=auto] [clk=<freqHz> | clk=A]`	`chain (chn) [dev=auto | dev=? |dev=DEVICE_NAME{,DEVICE_NAME}*] [clk=<FreqHz> | clk=A]`	CSAT600 does not allow the dev=? to display the current setup of the scan chain.
`connect (con) [<probe type>:]TCP:<hostname> | TCP:<ip address> | USB [config file]`	`connect (con) TCP:<hostname> | TCP:<ip address> | USB | USB:<serial_no>`	CSAT600 does not allow connecting to USB using the serial number of a debug probe. CSAT does not allow specifying a `<probe type>` or a `<config file>` or (SDF).
`devopen (dvo, device) <device no>`	`devopen (dvo) <devid>`	CSAT does not allow using the device command alias.
`exit (x)`	`exit`	CSAT does not allow using the x command alias.
`help (h)`	`trace help`	CSAT lists the available trace functions. CSAT does not have an equivalent help command that lists all functions available. CSAT600 prints a list of all available commands.

5. Worked examples for CSAT600

Here are some examples of using CSAT600 to perform particular tasks. Because CSAT600 is a flexible tool with many applications, these example are here to show you the kind of command call order and possible output that you might see from the commands.

Example 1 Manually specify the scan chain

This example shows how to manually specify the scan chain using the `chain (chn)` command. In the CSAT600 tool, only the devices on the scan chain are specified. The CSAT600 tool does not specify all the CoreSight devices. For the `chain (chn)` command, JTAG scan chain devices are listed in the order in which they appear on the scan chain.

Optionally, for the `connect (con)` command, you can use the Platform Configuration Editor (PCE) tool to generate a System Description File (SDF). If an SDF is not specified, the `chain (chain)` and `autodetect (auto)` commands produce a temporary SDF for the debug probe connection. Use the SDF generated by PCE or the `connect (con)` command for subsequent connections to the target.

```
C:\Program Files\Arm\Development Studio 2019.0\bin>csat -cs600

**********************************

**  Welcome to CSAT for SoC600  **

**********************************



% > con TCP:255.255.255.255
Connecting to TCP:255.255.255.255 ...
Connected to: DSTREAM-ST
Base H/W: V1 Rev A-05
FPGA build 0x0014, Debug 1V8, Trace 1V8
Firmware: 5.0.0, Build 7
Configuration file: C:\Users\< user >\AppData\Local\Temp
\csat_scanchain_devices3560597998775515521.sdf

% > chain dev=ARMCS-DP
------------+--------------------------------+---------
 Device No. | DTSL Device Name (& Address)   | AP No.
------------+--------------------------------+---------
          0 |                      ARMCS-DP  |

% > disconnect
Disconnected from TCP:255.255.255.255

% > exit
```

Example 2 Target autodetection

This example shows how to autodetect the target and read its ROM table. With a CoreSight SoC-600 target, the CSAT600 tool presents all components in a linear sequence, no matter how the Access Ports (APs) are structured. Each device detected is accessible using a device number (Device No.). CSAT600 logging is also enabled to capture a log of the autodetection process.

```
C:\Program Files\Arm\Development Studio 2019.0\bin>csat -cs600
```

```
**********************************

**  Welcome to CSAT for SoC600  **

**********************************



% > con TCP:255.255.255.255
Connecting to TCP:255.255.255.255 ...
Connected to: DSTREAM-ST
Base H/W: V1 Rev A-05
FPGA build 0x0014, Debug 1V8, Trace 1V8
Firmware: 5.0.0, Build 7
Configuration file: C:\Users\< user >\AppData\Local\Temp
\csat_scanchain_devices1844853173103486319.sdf

% > log on C:\Users\< user >\AppData\Local\Temp\log.txt
Logging is enabled with logfile: C:\Users\< user >\AppData\Local\Temp\log.txt

% > autodetect read-rom
Detecting platform...

------------+--------------------------------+---------
 Device No. | DTSL Device Name (& Address)   | AP No.
------------+--------------------------------+---------
          0 |                      ARMCS-DP  |
          1 |         CSMEMAP_0 (0x00000000) | 0
          2 |      Cortex-A53_0 (0x80410000) | 0
          3 |           CSCTI_0 (0x80420000) | 0
          4 |           CSPMU_0 (0x80430000) | 0
          5 |           CSETM_0 (0x80440000) | 0
          6 |      Cortex-A53_1 (0x80510000) | 0
          7 |           CSCTI_1 (0x80520000) | 0
          8 |           CSPMU_1 (0x80530000) | 0
          9 |           CSETM_1 (0x80540000) | 0
         10 |           CSTMC_0 (0x80800000) | 0
         11 |            CSTPIU (0x80820000) | 0
         12 |           CSTMC_1 (0x80830000) | 0
         13 |           CSTMC_2 (0x80840000) | 0
         14 |           CSTMC_3 (0x80850000) | 0
         15 |             CSSTM (0x80860000) | 0
         16 |           CSCTI_2 (0x80870000) | 0
         17 |           CSCTI_3 (0x80880000) | 0
         18 | CSATBReplicator_0 (0x80890000) | 0
         19 | CSATBReplicator_1 (0x808A0000) | 0
         20 |       CSTFunnel_0 (0x808B0000) | 0
         21 |       CSTFunnel_1 (0x808C0000) | 0
         22 |         CSMEMAP_1 (0x00000000) | 1

% > log off
Logging is disabled.

% > disconnect
Disconnected from TCP:255.255.255.255

% > exit
```

Example 3 Reading CoreSight component registers using the MEMAP APB

This example shows how to read and set CoreSight component registers using the Memory AP (MEMAP) APB. The debug probe connection is using a previously generated SDF.

```
C:\Program Files\Arm\Development Studio 2019.0\bin>csat -cs600
```

```
**********************************

**  Welcome to CSAT for SoC600  **

**********************************



% > con TCP:255.255.255.255 C:\Users\< user >\AppData\Local\Temp\AMIS FPGA.sdf
Connecting to TCP:255.255.255.255 ...
Connected to: DSTREAM-ST
Base H/W: V1 Rev A-05
FPGA build 0x0014, Debug 1V8, Trace 1V8
Firmware: 5.0.0, Build 7
Configuration file: C:\Users\< user >\AppData\Local\Temp\AMIS FPGA.sdf

% > list
------------+--------------------------------+---------
 Device No. | DTSL Device Name (& Address)    | AP No.
------------+--------------------------------+---------
          0 |                     ARMCS-DP  |
          1 |         CSMEMAP_0 (0x00000000) | 0
          2 |      Cortex-A53_0 (0x80410000) | 0
          3 |           CSCTI_0 (0x80420000) | 0
          4 |           CSPMU_0 (0x80430000) | 0
          5 |           CSETM_0 (0x80440000) | 0
          6 |      Cortex-A53_1 (0x80510000) | 0
          7 |           CSCTI_1 (0x80520000) | 0
          8 |           CSPMU_1 (0x80530000) | 0
          9 |           CSETM_1 (0x80540000) | 0
         10 |           CSTMC_0 (0x80800000) | 0
         11 |            CSTPIU (0x80820000) | 0
         12 |           CSTMC_1 (0x80830000) | 0
         13 |           CSTMC_2 (0x80840000) | 0
         14 |           CSTMC_3 (0x80850000) | 0
         15 |             CSSTM (0x80860000) | 0
         16 |           CSCTI_2 (0x80870000) | 0
         17 |           CSCTI_3 (0x80880000) | 0
         18 | CSATBReplicator_0 (0x80890000) | 0
         19 | CSATBReplicator_1 (0x808A0000) | 0
         20 |       CSTFunnel_0 (0x808B0000) | 0
         21 |       CSTFunnel_1 (0x808C0000) | 0
         22 |         CSMEMAP_1 (0x00000000) | 1

% > dvo 1
Connected to device no. 1: CSMEMAP_0

% > mr 0x80540000 8
Reading from device no. 1: CSMEMAP_0
0x80540000 : 0x00000000
0x80540004 : 0x00000000
0x80540008 : 0x00000000
0x8054000c : 0x00000003
0x80540010 : 0x00000001
0x80540014 : 0x00000000
0x80540018 : 0x00000000
0x8054001c : 0x00000000

% > mw 0x80540004 0
Writing to device no. 1: CSMEMAP_0
Wrote 4 bytes.

% > mr 0x80540004 1
Reading from device no. 1: CSMEMAP_0
```

```
0x80540004 : 0x00000000

% > dvc
Disconnected from device no. 1

% > dcn
Disconnected from TCP:255.255.255.255

% > x
```

Example 4 Changing a DSTREAM configuration item

This example shows how to autodetect a target and change the SRSTOnConnect DSTREAM configuration item.

```
C:\Program Files\Arm\Development Studio 2019.0\bin>csat -cs600
**********************************
**  Welcome to CSAT for SoC600  **
**********************************

%> con TCP:255.255.255.255
Connecting to TCP:255.255.255.255 ...
Connected to: DSTREAM-ST
Base H/W: V2 Rev A-06
FPGA build 0x0014, Debug 1V8, Trace 1V8
Firmware: 5.3.0, Build 4
Configuration file: C:\Users\< user >\AppData\Local\Temp
\csat_scanchain_devices3925139137141746493.sdf

% > chain dev=auto
Detecting scanchain...
------------+--------------------------------+---------
 Device No. | DTSL Device Name (& Address)   | AP No.
------------+--------------------------------+---------
          0 |                      ARMCS-DP  |

%> auto read-rom
Detecting platform...
------------+--------------------------------+---------
 Device No. | DTSL Device Name (& Address)   | AP No.
------------+--------------------------------+---------
          0 |                      ARMCS-DP  |
          1 |         CSMEMAP_0 (0x00000000) | 0
          2 |      Cortex-A53_0 (0x80410000) | 0
          3 |           CSCTI_0 (0x80420000) | 0
          4 |           CSPMU_0 (0x80430000) | 0
          5 |           CSETM_0 (0x80440000) | 0
          6 |      Cortex-A53_1 (0x80510000) | 0
          7 |           CSCTI_1 (0x80520000) | 0
          8 |           CSPMU_1 (0x80530000) | 0
          9 |           CSETM_1 (0x80540000) | 0
         10 |           CSTMC_0 (0x80800000) | 0
         11 |            CSTPIU (0x80820000) | 0
         12 |           CSTMC_1 (0x80830000) | 0
         13 |           CSTMC_2 (0x80840000) | 0
         14 |           CSTMC_3 (0x80850000) | 0
         15 |             CSSTM (0x80860000) | 0
         16 |           CSCTI_2 (0x80870000) | 0
         17 |           CSCTI_3 (0x80880000) | 0
         18 | CSATBReplicator_0 (0x80890000) | 0
         19 | CSATBReplicator_1 (0x808A0000) | 0
         20 |       CSTFunnel_0 (0x808B0000) | 0
         21 |       CSTFunnel_1 (0x808C0000) | 0
         22 |         CSMEMAP_1 (0x00000000) | 1
```

```
% > device 2
Connected to device no. 2: Cortex-A53_0, JTAG ID: 0x1ba06477, version 0x00000006
Msg returned from device: Cortex-A53 Template

% > help cfgbox
Command: cfgbox
        Configure the DSTREAM probe.
Aliases:
        cfb
Usage:
        cfgbox [< item_name > [< item_value >]]
        cfb [< item_name > [< item_value >]]

% > cfgbox
Linked_SRST_TRST      : 0
SRSTOnConnect         : 0
UserOutputPin  s      : 000000
UseDeprecatedSWJ      : 0
DSTREAMCS20           : 0
TResetOnInitConnect   : 1
AllowTRST             : 1
ResetHoldTime         : 100
TRSTOnConnect         : 1
PROBE                 : 90112
RvcHash               : 1362632482
MinimalConnect        : 0
nTRSTHoldTime         : 10
PowerUpGPR            : 1
GdbConnCmdSeq         :
SessionPause          : 0
nTRSTPostResetTime    : 10
UserOut_P5            : 0
TRSTPostResetTime     : 10
UserOut_P4            : 0
SWJEnable             : 0
JtagClockFreq         : 7500000
SWOMode               : 0
VCC                   : 805306368
JTAGAutoMaxFreq       : 200000000
ProbeMode             : 1
PostResetDelay        : 1000
nSRSTHighMode         : 1
ClusterDescription    :
UserOut_DBGRQ         : 0
CONNECTOR             : ARM JTAG 20 (JTAG)
nSRSTLowMode          : 0
TRSTHoldTime          : 10
AllowConInReset       : 0
SWOBaudRate           : 0
UserOut_P3            : 0
UserOut_P2            : 0
DoSoftTAPReset        : 1
UserOut_P1            : 0
DoSoftTRST            : 1
AP_V3_ADDR_IDX_MAP    :
AllowICELatchSysRst   : 1
AllowICETAPReset      : 1
nTRSTHighMode         : 0
LVDSProbeMode         : 1
SResetOnInitConnect   : 0
ResetOperation        : 0
nTRSTLowMode          : 0
FPGARegOffset         : 0
PythonScript          :
FPGARegValue          : 24576
UserOut_P6_COAX       : 0
JtagClockType         : 2
ScriptTimeout         : 1000
TCKOnIdle             : 0
PowerFilterTime       : 100
```

```
ScanChainJtagFreqs    : 7500000T

% > cfb SRSTOnConnect 1
SRSTOnConnect: 1

% > cfb SRSTOnConnect
SRSTOnConnect: 1

% > dvc 2
Disconnected from device no. 2

% > dcn
Disconnected from TCP:255.255.255.255

% > x
Disconnected from TCP:255.255.255.255
```

Example 5 Reading and writing registers using the dpregread and dpregwrite commands

This example performs various register reads and writes using the `dpregread` and `dpregwrite` commands.

The example accesses the below registers:

- Advanced eXtensible Interface (AXI) Access Port 0 Transfer Access Register (AP0.TAR)
- Advanced Peripheral Bus (APB) Access Port 1 Transfer Access Register (AP1.TAR)
- AXI Access Port 0 Identification Register (AP0.IDR)
- Advanced High-performance Bus (AHB) for Cortex-M Access Port 2 Identification Register (AP2.IDR or device44.`0x341`)
- Advanced High-performance Bus (AHB) for Cortex-M Transfer Access Register (device44.TAR or CSMEMAP_2.TAR)

Note

The example assumes that you have:

- Started CSAT600 (csat -cs600).
- A debug probe connection is in place. See `connect (con)`.
- Discovered CoreSight devices. See `autodetect (auto)` or `chain (chn)`.

```
% > list
------------+--------------------+-----------------+--------------+--------
 Device No. |    Device Name     |   Device Type   | Base Address | AP No.
------------+--------------------+-----------------+--------------+--------
          0 |         ARMCS-DP_0 |        ARMCS-DP |         None |
          1 |          CSMEMAP_0 |          AXI-AP |   0x00000000 | 0
          2 |          CSMEMAP_1 |          APB-AP |   0x00000000 | 1
          3 |            CSTMC_0 |           CSTMC |   0x80010000 | 1
          4 |            CSCTI_0 |           CSCTI |   0x80020000 | 1
          5 |             CSTPIU |          CSTPIU |   0x80030000 | 1
          6 |        CSTFunnel_0 |       CSTFunnel |   0x80040000 | 1
          7 |            CSTMC_1 |           CSTMC |   0x80070000 | 1
          8 |              CSSTM |           CSSTM |   0x80100000 | 1
          9 |           CSCTI_1 |           CSCTI |   0x80110000 | 1
         10 | CSATBReplicator_0 | CSATBReplicator |   0x80120000 | 1
         11 |       CSTFunnel_1 |       CSTFunnel |   0x80130000 | 1
         12 |           CSTMC_2 |           CSTMC |   0x80140000 | 1
         13 |       CSTFunnel_2 |       CSTFunnel |   0x80150000 | 1
         14 |           CSCTI_2 |           CSCTI |   0x80160000 | 1
         15 |      Cortex-A72_0 |      Cortex-A72 |   0x82010000 | 1
         16 |           CSCTI_3 |           CSCTI |   0x82020000 | 1
         17 |           CSPMU_0 |           CSPMU |   0x82030000 | 1
         18 |           CSETM_0 |           CSETM |   0x82040000 | 1
         19 |      Cortex-A72_1 |      Cortex-A72 |   0x82110000 | 1
         20 |           CSCTI_4 |           CSCTI |   0x82120000 | 1
         21 |           CSPMU_1 |           CSPMU |   0x82130000 | 1
         22 |           CSETM_1 |           CSETM |   0x82140000 | 1
         23 |       CSTFunnel_3 |       CSTFunnel |   0x820C0000 | 1
         24 |             ELA_0 |             ELA |   0x820D0000 | 1
         25 |      Cortex-A53_0 |      Cortex-A53 |   0x83010000 | 1
         26 |           CSCTI_5 |           CSCTI |   0x83020000 | 1
         27 |           CSPMU_2 |           CSPMU |   0x83030000 | 1
         28 |           CSETM_2 |           CSETM |   0x83040000 | 1
         29 |      Cortex-A53_1 |      Cortex-A53 |   0x83110000 | 1
         30 |           CSCTI_6 |           CSCTI |   0x83120000 | 1
         31 |           CSPMU_3 |           CSPMU |   0x83130000 | 1
         32 |           CSETM_3 |           CSETM |   0x83140000 | 1
         33 |      Cortex-A53_2 |      Cortex-A53 |   0x83210000 | 1
         34 |           CSCTI_7 |           CSCTI |   0x83220000 | 1
         35 |           CSPMU_4 |           CSPMU |   0x83230000 | 1
         36 |           CSETM_4 |           CSETM |   0x83240000 | 1
         37 |      Cortex-A53_3 |      Cortex-A53 |   0x83310000 | 1
         38 |           CSCTI_8 |           CSCTI |   0x83320000 | 1
         39 |           CSPMU_5 |           CSPMU |   0x83330000 | 1
         40 |           CSETM_5 |           CSETM |   0x83340000 | 1
         41 |       CSTFunnel_4 |       CSTFunnel |   0x830C0000 | 1
         42 |             ELA_1 |             ELA |   0x830D0000 | 1
         43 |        ARMCS-DP_1 |        ARMCS-DP |         None |
         44 |         CSMEMAP_2 |        AHB-AP-M |   0x00000000 | 0
         45 |         Cortex-M3 |       Cortex-M3 |   0xE000E000 | 0
         46 |             CSDWT |           CSDWT |   0xE0001000 | 0
         47 |             CSFPB |           CSFPB |   0xE0002000 | 0
         48 |             CSITM |           CSITM |   0xE0000000 | 0
         49 |           CSETM_6 |           CSETM |   0xE0041000 | 0
         50 |       CSTFunnel_5 |       CSTFunnel |   0xE0042000 | 0
         51 |             CSSWO |           CSSWO |   0xE0043000 | 0
         52 |           CSCTI_9 |           CSCTI |   0xE0044000 | 0
         53 | CSATBReplicator_1 | CSATBReplicator |   0xE0045000 | 0

% > dpregread AP0.TAR
Device no. 1 is active.
Reading from device no. 1: CSMEMAP_0
AP0:0x341 : 0x00000FF0

% > dpregwrite AP0.TAR 0xEFC4AFC0
Device no. 1 is active.
Writing to device no. 1: CSMEMAP_0

% > dpregread AP0.TAR
Device no. 1 is active.
Reading from device no. 1: CSMEMAP_0
AP0:0x341 : 0xEFC4AFC0

% > dpregread AP1.TAR
Device no. 2 is active.
```

```
Reading from device no. 2: CSMEMAP_1
AP1:0x341 : 0x830C0FCC

% > dpregread AP0.IDR
Device no. 1 is active.
Reading from device no. 1: CSMEMAP_0
AP0:0x37F : 0x14770004

% > dpregread AP2.IDR
Device no. 44 is active.
Reading from device no. 44: CSMEMAP_2
AP2:0x37F : 0x24770011

% > dpregwrite device44.TAR 256
Device no. 44 is active.
Writing to device no. 44: CSMEMAP_2

% > drr CSMEMAP_2.TAR
Device no. 44 is active.
Reading from device no. 44: CSMEMAP_2
CSMEMAP_2:0x341 : 0x00000100

% > drw device44.0x341 0xF00F
Device no. 44 is active.
Writing to device no. 44: CSMEMAP_2

% > drr CSMEMAP_2.TAR
Device no. 44 is active.
Reading from device no. 44: CSMEMAP_2
CSMEMAP_2:0x341 : 0x0000F00F
```

Example 6 Adding additional probe

This example shows how to load, set, and connect to an additional probe called MyProbe.

Note

The example assumes that you have started CSAT600 (csat -cs600).

```
%> loadprobes c:\work\probes.xml
Parsing file c:\work\probes.xml...
Probe MyProbe was loaded.

% > listprobes
* DSTREAM         built-in
   DSTREAM-HT      built-in
   DSTREAM-PT      built-in
   DSTREAM-ST      built-in
   MyProbe         from c:\work\probes.xml
   RealView ICE    built-in

% > setprobe MyProbe
Probe type was set to MyProbe.

% > listprobes
   DSTREAM         built-in
   DSTREAM-HT      built-in
   DSTREAM-PT      built-in
   DSTREAM-ST      built-in
* MyProbe         from c:\work\probes.xml
   RealView ICE    built-in

% > connect MyProbeAddress
```

```
Connecting to MyProbeAddress ...
Starting debug server...
Debug server started successfully.
Connected to: MyProbe
Configuration file: C:\Users\< user >\AppData\Local\Temp
\csat_scanchain_devices5986504949331606589.sdf

% > disconnect
Disconnected from MyProbeAddress

% > setprobe DSTREAM
Probe type was set to DSTREAM.

% > connect MyProbe:MyProbeAddress
Probe type was set to MyProbe.
Connecting to MyProbeAddress ...
Starting debug server...
Debug server started successfully.
Connected to: MyProbe
Configuration file: C:\Users\< user >\AppData\Local\Temp
\csat_scanchain_devices2078644656863206288.sdf

% > disconnect
Disconnected from MyProbeAddress
```