AN13507

How to Perform Boundary Scan for LPC55(S)xx based on µTrace and Trace32

Rev. 1 — 14 November 2023 Application note

Document information

Information	Content
Keywords	AN13507, boundary scan, Boundary-Scan Description Language (BSDL)
Abstract	This application note describes how to enter the boundary scan mode, how to perform boundary scan test based on Boundary-Scan Description Language (BSDL) on LPC55(S)xx series, an overview of JTAG and boundary scan technology.

1 Overview

LPC55(S)xx is a microcontroller based on the Arm Cortex-M33 for embedded applications. It supports JTAG boundary scan. This document describes:

- How to enter the boundary scan mode.
- How to perform a boundary scan test based on Boundary-Scan Description Language (BSDL) on LPC55(S)xx series.
- An overview of JTAG and boundary scan technology.

To understand this document better, basic knowledge of JTAG and boundary scan is required.

2 JTAG and boundary scan

2.1 Introduction

JTAG/boundary scan is an interface containing four ports. The interface allows access to the special embedded logic on most chips. JTAG/boundary can provide several functions that can contain any or all the following:

- Probe-less device connectivity test.
- Logic programming for Flash memory, CPLD, and FPGA.
- Debug logic in microprocessors and microcontrollers used for software debugging, or test connections with peripheral devices at speed without embedded software.

2.2 Development history

The architecture for Test Access Port (TAP) and Boundary Scan is defined in IEEE Std 1149.1. The development history of this standard is summarized as follows:

- 1985 Joint European Test Action Group (**JETAG**) was formed.
- 1986 Joint European Test Action Group (**JETAG**) was renamed as Joint Test Action Group (**JTAG**).
- 1986-1988 JTAG Technical Subcommittee developed and published a series of proposals for a standardized form of boundary scan.
- 1988 the last of these proposals, JTAG Version 2.0, was offered to the IEEE Testability Bus Standards Committee (P1149) and was accepted by P1149. JTAG proposal became the basis of a standard within the Testability Bus family.
- 1990 From 1990, JTAG developed a supplement for correction, clarification, and enhancement.
- 1993 IEEE Std 1149.1aTM-1993
- 1994 IEEE Std 1149.1b-1994
- 2001 IEEE Std 1149.1-2001
- 2013 IEEE Std 1149.1-2013

2.3 Basic principle

Boundary scan is a method for testing interconnects on PCBs and internal IC subblocks. For boundary scan tests, more logic is added to the device. Boundary scan cells are placed between the core logic and the ports.

In the boundary scan test, each primary input and output signal on a device is supplemented with a multipurpose memory element called as a boundary scan cell. These cells are connected to a shift register, which is referred to as the boundary scan register. This register can be used to read and write port states.

In normal mode, these cells are transparent and the core is connected to those ports. In the boundary scan mode, the core is isolated from the ports and the JTAG interface controls the port signals.

Figure 1 shows the basic principle of boundary scan.

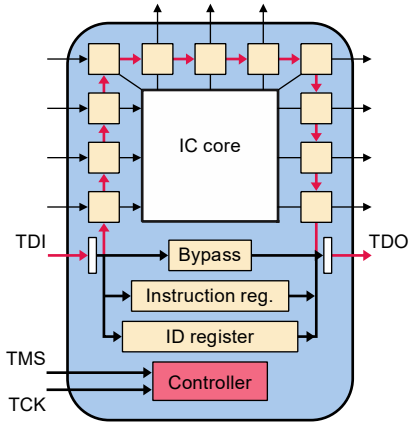

Figure 1. Boundary scan basic principle

2.4 Instruction set

Table 1 describes the boundary scan instructions defined in the IEEE Std 1149.1.

Table 1. Standard instruction set

Instruction	Mandatory/Optional	Description
BYPASS	Mandatory	TDI is connected to TDO via a single shift register.
SAMPLE	Mandatory	Takes a snapshot of the normal operation of the Integrated Circuit (IC).
PRELOAD	Mandatory	Loads data to the boundary scan register.
EXTEST	Mandatory	Applying preloaded data of the boundary scan register to the ports.
INTEST	Optional	Applying the preloaded data of the boundary scan register to the core logic.
RUNBIST	Optional	Executing a self-contained self-test of the IC.
CLAMP	Optional	Applying the preloaded data of the boundary scan register to the ports and selects the bypass register as the serial path between TDI and TDO.
IDCODE	Optional	Reading the device identification register.
USERCODE	Optional	Reading and writing a user programmable identification register.
HIGHZ	Optional	Placing the IC in an inactive drive state (for example, all ports are set to high-impedance state).

2.5 JTAG Test Access Port (TAP)

TAP is a general-purpose port that can provide access to many test support functions built into a component, including the test logic. It is composed of **a minimum of the three input connections**, which are **TCK**, **TMS**, **TDI**, and **one output connection**, which is **TDO**. An optional fourth input connection, TRST, is provided for asynchronous initialization of the test logic.

Table 2 describes TAP signals.

Table 2. TAP signal description

Signal name	I/O type	Description
TCK	Input	Providing the clock for the test logic.
TMS	Input	Value of the signal presented as TMS at the time of a rising edge at TCK determines the next state of the TAP controller. The circuit that controls test operations.
TDI	Input	Serial test instructions and data received by the test logic at TDI.
TDO	Output	Serial output for test instructions and data from the test logic.
TRST	Input	Provided for asynchronous initialization and active low.

Note: *JTAG is used on LPC55(S)xx devices for boundary scan and the production test only. It cannot be used for debugging purposes.*

2.6 BSDL

Boundary-Scan Description Language (BSDL) is based on the syntax and grammar of Very high-speed integrated-circuit Hardware Description Language (VHDL). BSDL is not a general-purpose hardware description language. It describes key aspects to implement the boundary scan within a particular component.

Table 3 lists elements contained in the BSDL file.

Table 3. BSDL elements

Element	Description
Entity Description	Statement for device name or functionality.
Generic Parameter	Description for package or pin mapping.
Logical Port Description	Description for pin types such as in, out, in-out, linkage.
Standard Use Statement	References external definitions.
Component Conformance Statement	Standards to follow.
Device Package Pin Mapping	Description for pin mapping.
Scan Port Identification	Pin description on device for JTAG TAP including TCK, TMS, TDI, and TDO.
Compliance Enable Description	Pins involved in entering boundary scan mode and level applied to the pins. It is useful for chip enter boundary scan mode.
Instruction Register Description	Instruction length and instruction code. Sometimes it includes device-specific instruction, also called as private instruction.
Register Access Description	Registers corresponding to specific instructions
Boundary-Scan Register Description	List recording the boundary scan cells and functionality of these boundary scan cells.

Download BSDL files for LPC55(S)xx through the following links:

- LPC55(S)6x: https://www.nxp.com/downloads/en/bsdl/LPC55S6X-BSDL.zip
- LPC55(S)2x: https://www.nxp.com/downloads/en/bsdl/LPC55S2X-BSDL.zip
- LPC55(S)1x: https://www.nxp.com/downloads/en/bsdl/LPC55S1X-BSDL.zip

For the latest BSDL file of a specific part, download it from NXP official website. Taking the BSDL file for LPC55(S)6x as an example, perform the following steps:

- Go to NXP official website and type the product model in the search box. In this example, it is **LPC55S69**.

- To go to the homepage of the corresponding part, click the specific part in the search result list. In this example, it is **LPC55S69JBD100**.
- To go to the **TOOLS&SOFTWARE** tab on the Introduction page of the desired part, click the **Tools and Software** quick link.
- Download BSDL file from an item named **Design Tools and Files** in the **Tools and Software** section.

2.7 More information on JTAG and boundary scan

For more information on JTAG and boundary scan, see the links below:

- Homepage for JTAG and boundary scan: https://www.jtag.com/
- IEEE Std 1149.1
  - version 1990: https://standards.ieee.org/standard/1149_1-1990.html
  - version 2001: https://standards.ieee.org/standard/1149_1-2001.html
  - version 2013: https://standards.ieee.org/standard/1149_1-2013.html

3 Building boundary scan test environment

3.1 Introduction to boundary scan test tool suite

In this application note, the boundary scan test uses the tool set from Lauterbach, which is all-in-one debug and trace solution for Cortex-M. This tool set includes the following two parts:

- **µTrace for Cortex-M**
  µTrace for Cortex-M is one of the architecture-specific products from Lauterbach. The features are as below:
  - On-chip/external flash programming, debug, trace, and **JTAG boundary scan**
  - Recommended for **single-core microcontrollers with Cortex-M**
  - Recommended for **multicore microcontrollers with solely Cortex-M** (single debug port)
  - 256 MBytes trace memory
  - **USB 3 interface to the host computer**
  - TRACE32 streaming up to 150 MBytes/s
  - TRACE32 mixed signal probe supported

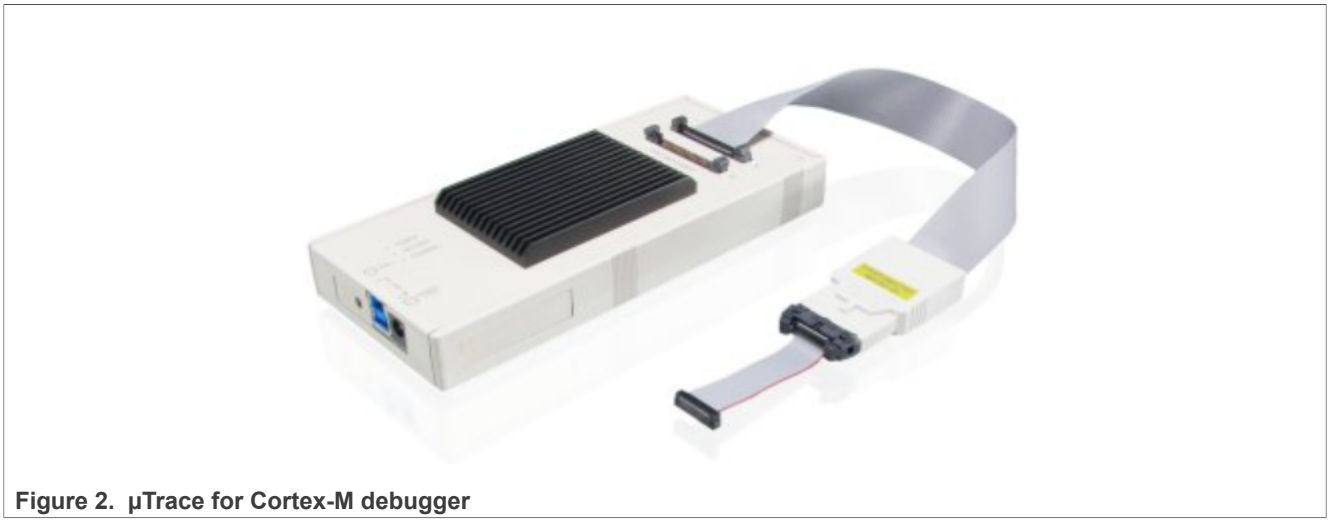

Figure 2. µTrace for Cortex-M debugger

For more information about µTrace for Cortex-M, see Figure 3.

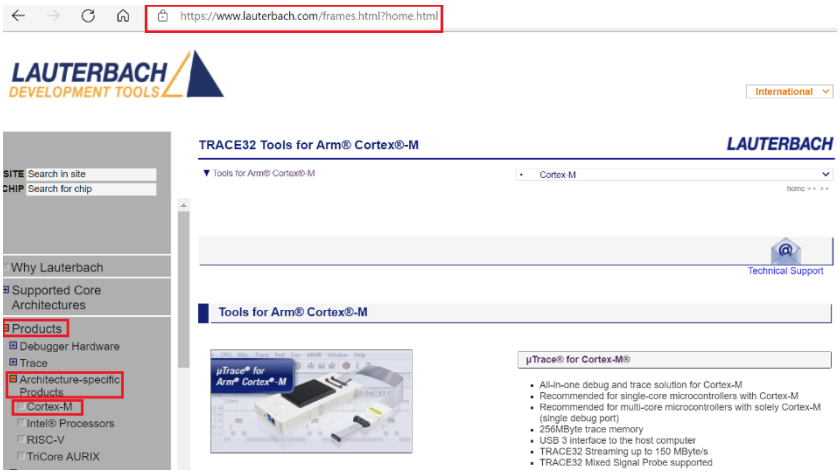

Figure 3. µTrace for Cortex-M introduction

- **TRACE32**

  TRACE32 is a simulation test tool developed by Lauterbach. It runs on a personal computer. It is used with µTrace for Cortex-M for on-chip/external Flash programming, debugging, tracing, and JTAG boundary scan. It supports various processor architectures, including standard processors such as **Arm**, MIPS, PowerPC and DSP, soft cores, and coprocessors.

  For boundary scan, TRACE32 provides GUI operations for interactive test and supports a script for automated test. To execute many commands, such as, system settings, JTAG, and BSDL, make a script contain these commands. To complete the test, execute the script. The execution is efficient and reduces the possibility of errors in the command-line mode.

  TRACE32 supports command-line input. The command-line input is at the bottom of the TRACE32 main page, starting with `B::`. To complete the operations, input commands such as system reset, system settings, BSDL file loading, and boundary scan test.

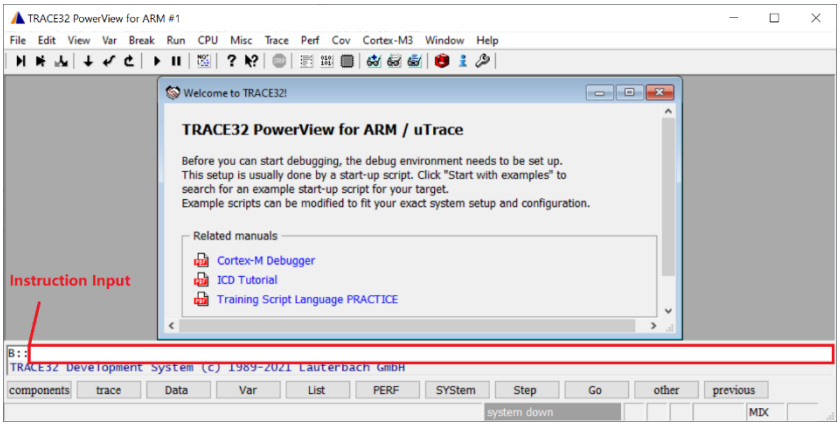

Figure 4. Main page for TRACE32 for Arm

To download TRACE32, see Figure 5.

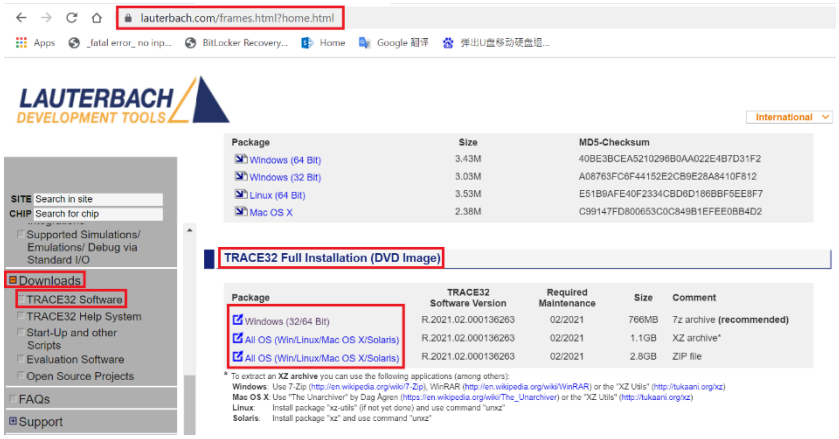

Figure 5. Download TRACE32

3.2 Hardware environment

µTrace for Cortex-M consists of:

- Universal debugger
- Debug cable

Figure 6. µTrace for Cortex-M composition

Figure 6 shows the schematic diagram for hardware connection of the entire boundary scan system.

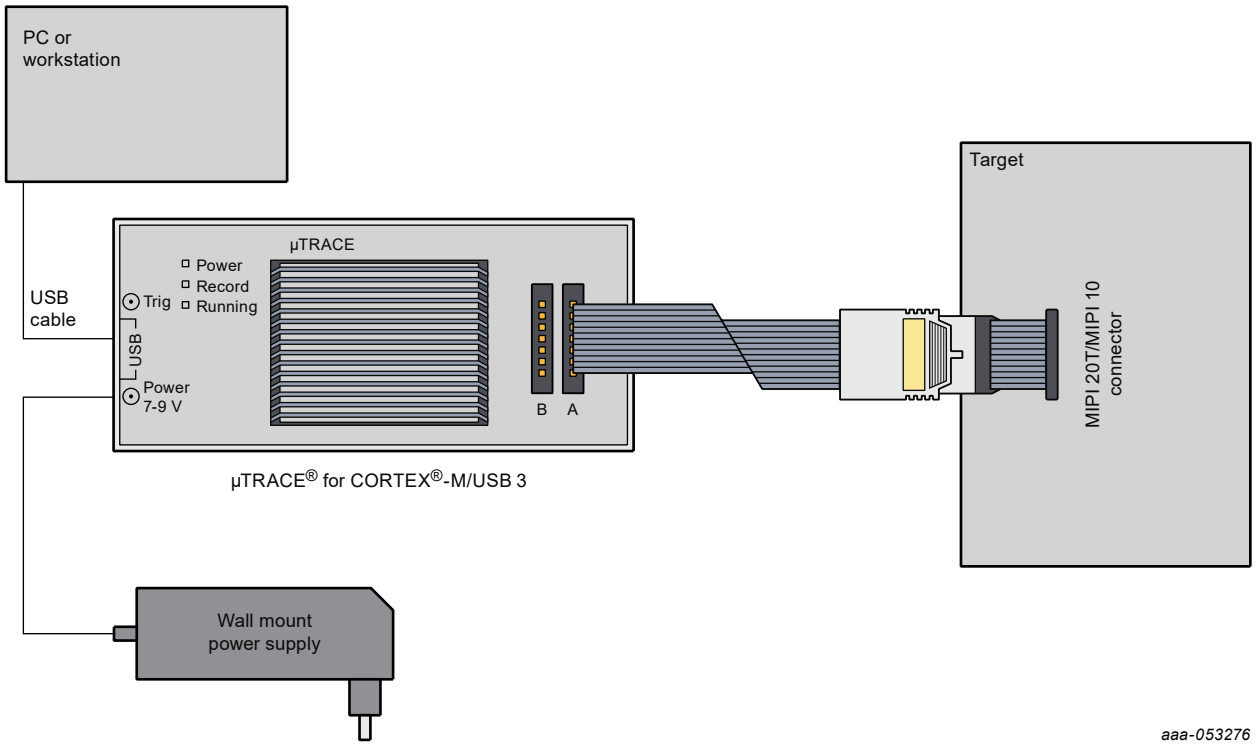

Figure 7. Schematic diagram of hardware connection

Below lists the suggestions for users:

- To prevent the debugger or target board being damaged, do not plug or unplug the debugger while the target board is powered on. The recommended sequence for powering on or off is as follows:
  - Power on: debugger > target board
  - Power off: target board > debugger
- The debugger interface has pin1. To prevent damages to the debugger or target, double check the direction.

To set up the hardware environment, perform the following steps:

1. Connect the µTrace for Cortex-M debugger to your target board with the standard JTAG interface.
2. Connect the µTrace for Cortex-M debugger to the personal computer through the USB cable, and then power on the debugger with a 5 V power adapter. Open Device Manager on the personal computer, and the Lauterbach equipment appears in Trace32 Devices, as shown in Figure 8. If not, check the connection.

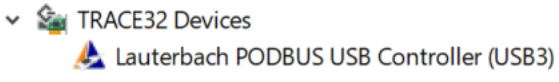

Figure 8. Lauterbach equipment appear in Device Manager

3. Power-up your target board.

3.3 Enter boundary scan mode

To let MCU series, including LPC55(S)0x, LPC55(S)1x, LPC55(S)2x, and LPC55(S)6x, enter the boundary scan mode, perform the following steps:

- Connect pins `P0_2` (TRST) and `P0_11` (SWCLK) to **POWER**, and pin `P0_12` (SWDIO) to **GND**.
- Press and hold the **ISP(PIO0_5)** button (PIO0_5 = 0).
- Press and hold the **RESET** button (Reset = 0).
- Release the **RESET** button (Reset = 1).

- Release the **ISP** button (PIO0_5 = 1).
- Disconnect pins P0_2 (TRST) and P0_11 (SWCLK) from **POWER**, and pin P0_12 (SWDIO) from **GND**.

The method for LPC55(S)3x to enter boundary scan mode is different from other LPC55(S)xx including 0x, 1x, 2x, and 6x. To make LPC55(S)3x enter boundary scan mode, perform the following steps:

- Apply a **rectangular** signal with a frequency of 1 MHz (greater than or equal to 500 kHz) and a **duty cycle** of **50 %** to P0_3 (**TCK**), which can be provided by a signal generator.
- Connect pins P0_2 (TRST) and P0_4 (TMS) to **POWER**.
- Hold the **RESET** pin **low** for more than 5.5 uS, and then release the **RESET** pin **high**.
- Remove the rectangular signal from P0_3 (TCK). This step is crucial.

Figure 9 and Figure 10 show the timing of LPC553x entering boundary scan. As shown in Figure 9, the duration of the **RESET negative pulse** is **181 ms**, which satisfies the condition of **greater than 5.5 us**. As shown in Figure 10, the frequency of **TCK** is **1 MHz**, which satisfies the condition of **greater than 500 kHz**.

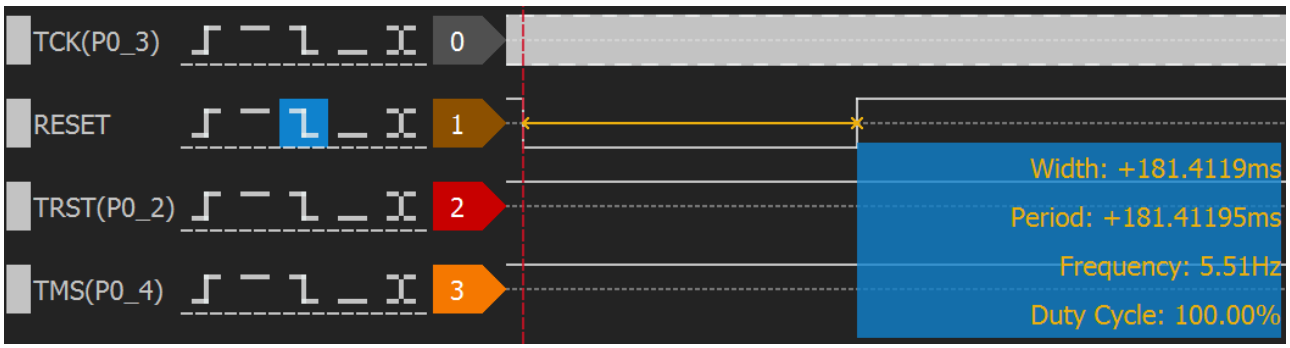

Figure 9. Duration of RESET negative pulse

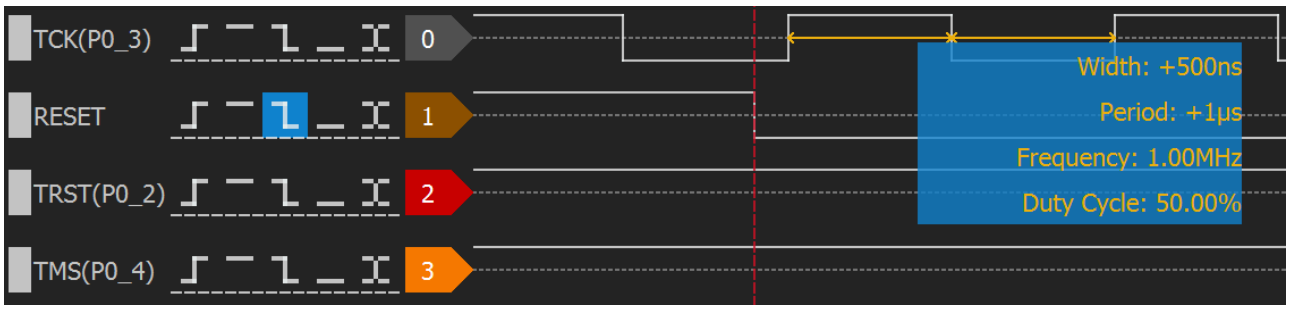

Figure 10. TCK frequency

After the above sequence to enter the boundary scan mode is executed, use a hardware debugger such as Jlink to check the byte addressed at 0x50020004 to determine whether the LPC55(S)xx actually enters the boundary scan mode or not. If the byte addressed at 0x50020004 is 0x07, it indicates that the chip is already in boundary scan mode. Below takes LPC55S36 as an example to describe this process.

- Use the Jlink debugger to connect the **SWD** port on the LPC55S36 target board to the USB port on the personal computer.
- Open the **J-Link Commander**. Figure 11 shows the operation interface.

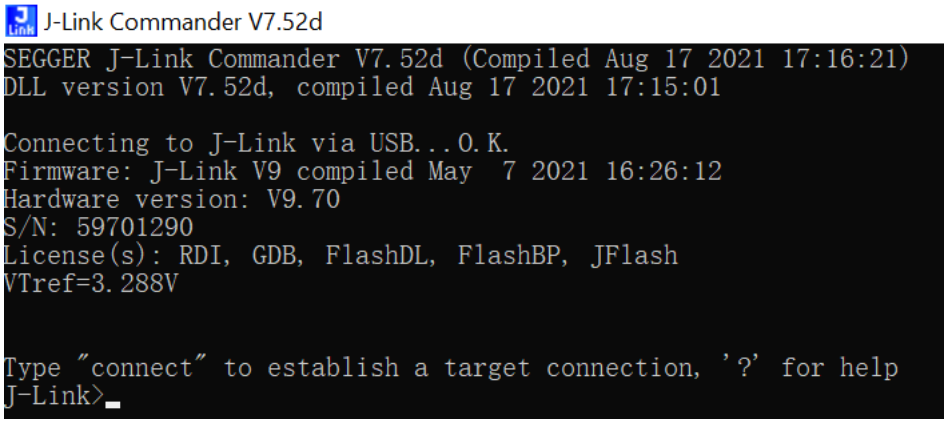

Figure 11. J-Link commander

- Enter **connect** to establish a connection between J-Link Commander and LPC55S36.
- Enter **?** to pop up the **Target device settings** dialog box, as shown in Figure 12. Select **Cortex-M33**.

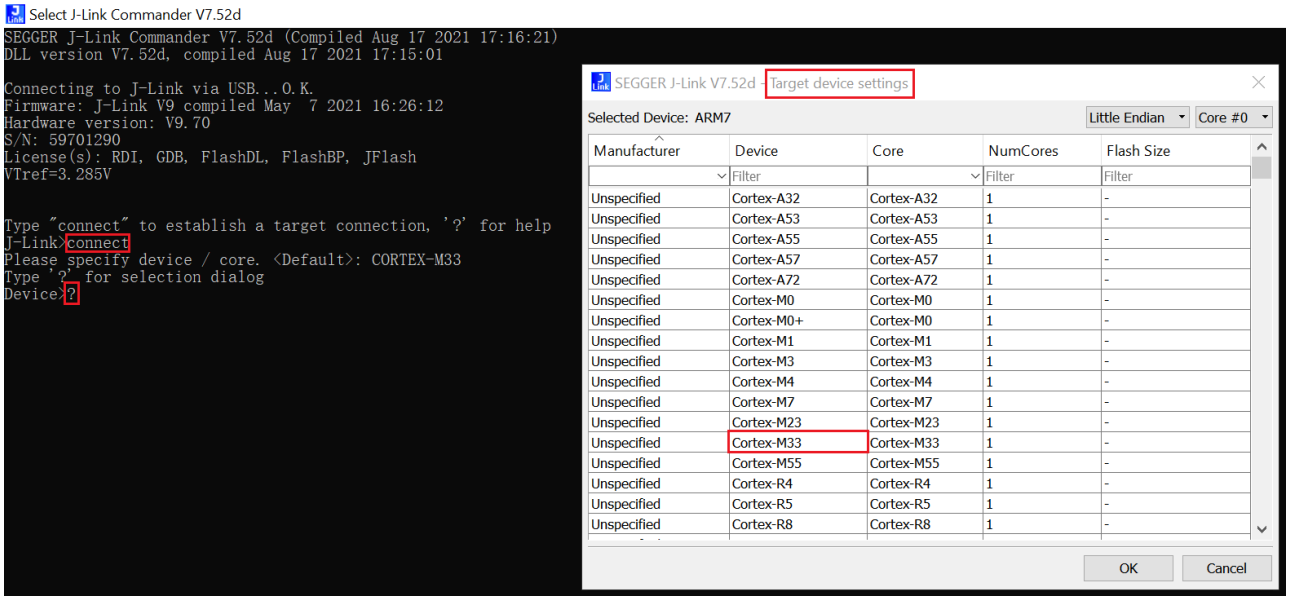

Figure 12. Establish debug connection and perform device setting

- Select the debug interface. In this example, it is **SWD**, as shown in Figure 13.

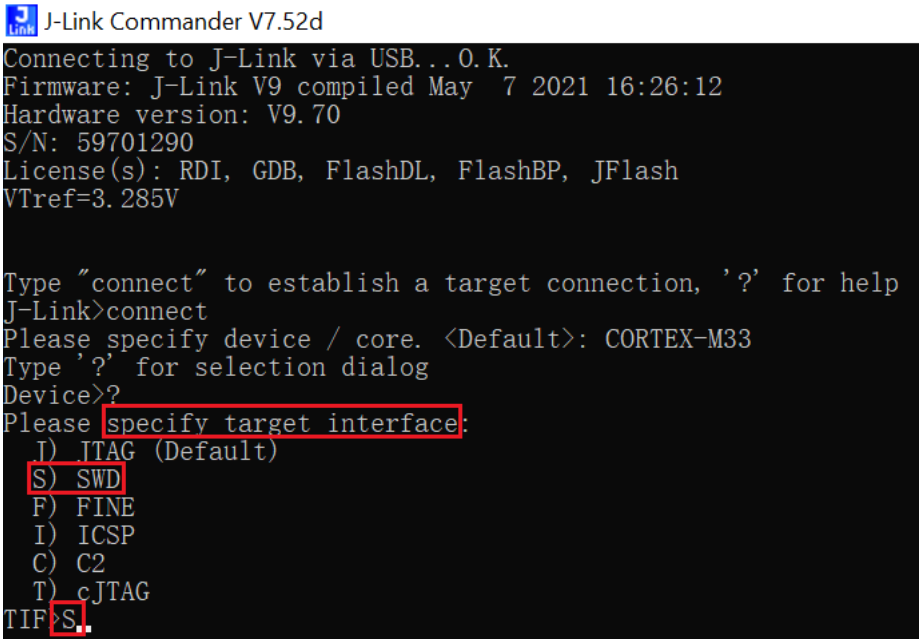

Figure 13. Specify the target interface as SWD

- Specify target interface speed. In this example, use the default value. To continue, press the **enter** button.

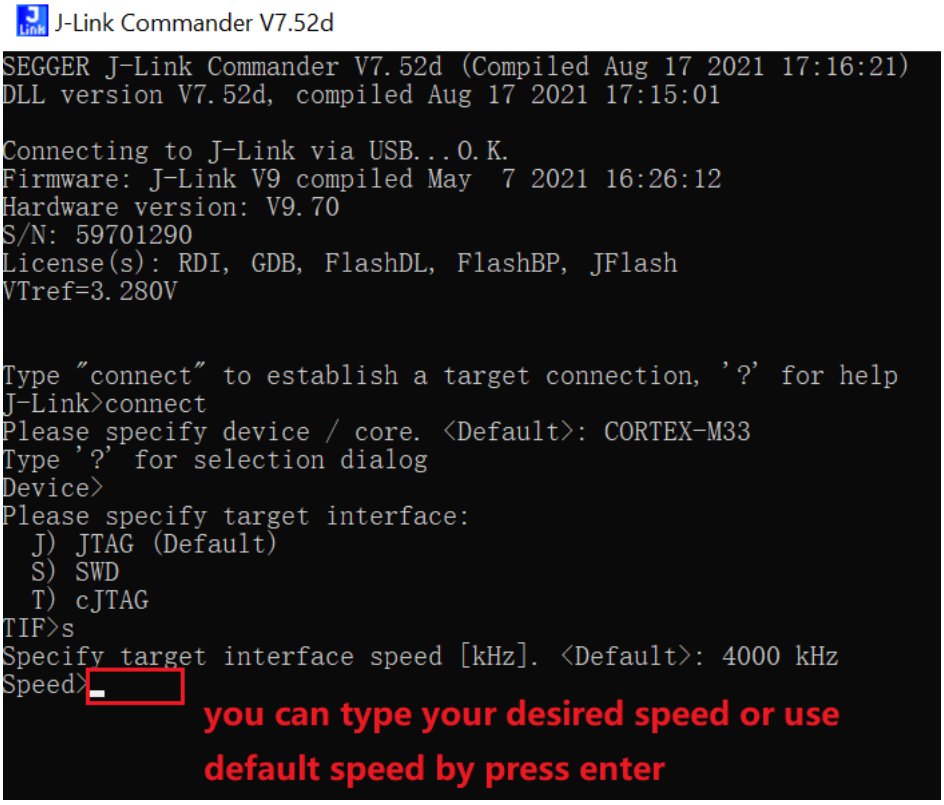

Figure 14. Specify target interface speed

- In the normal mode, check the byte value at address `0x50020004`. It is `0x01`.

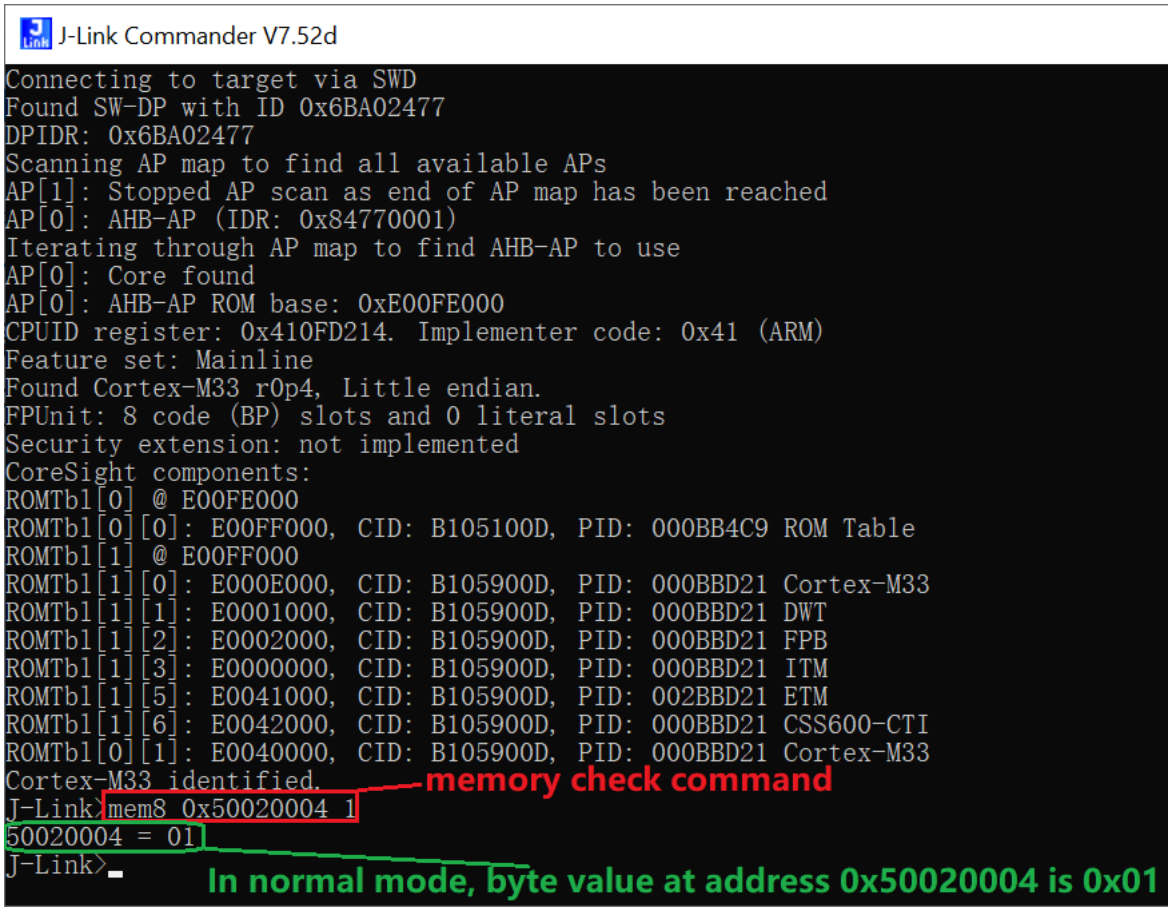

Figure 15. Check byte value at address 0x50020004 in normal mode

- Let LPC55S36 enter boundary scan mode and check the byte value at address **0x50020004**. Now, in **boundary scan mode**, it is `0x07`.

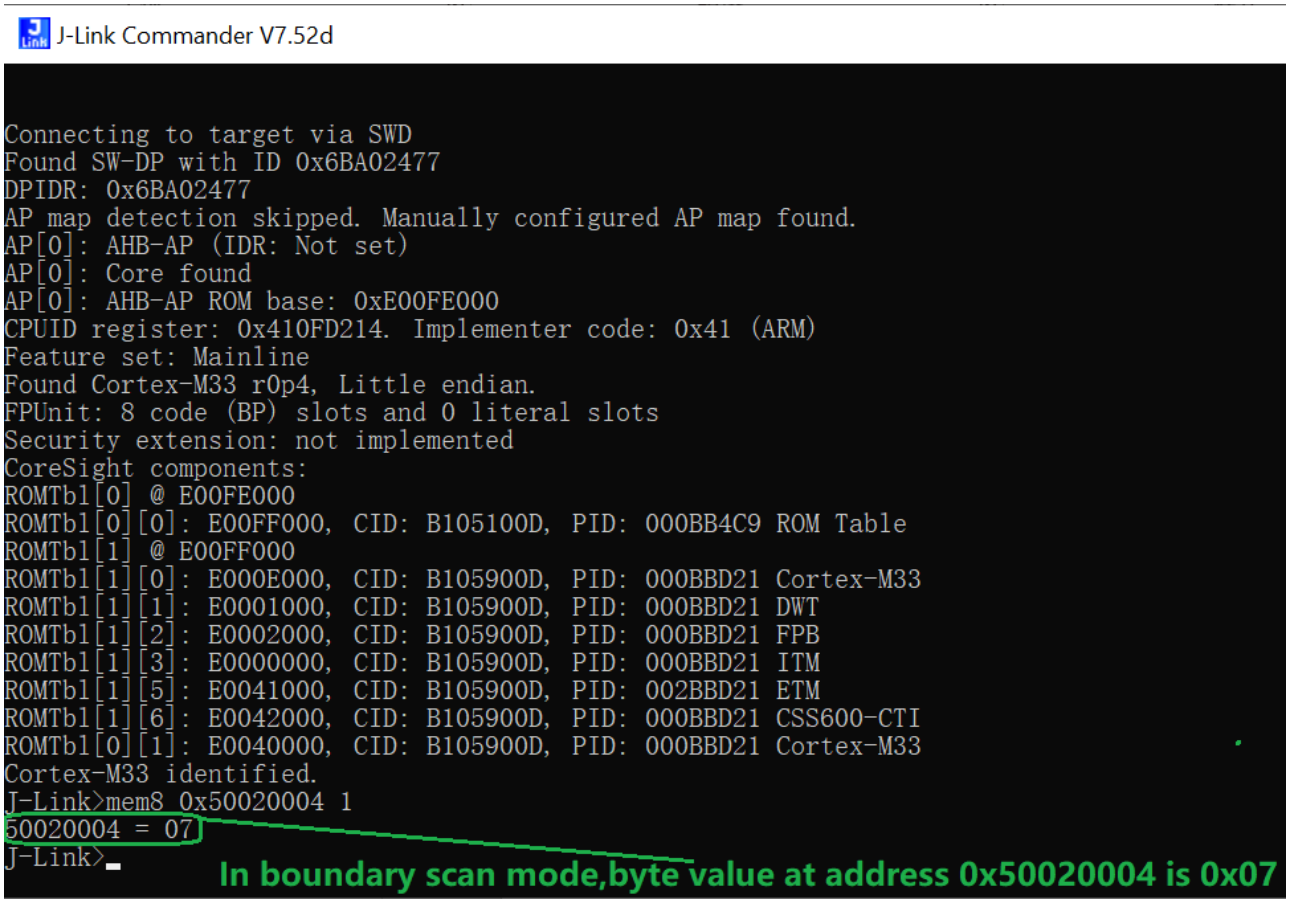

Figure 16. Check byte value at address `0x50020004` in boundary scan mode

4 Interactive boundary scan test

To perform a boundary scan test using µTrace for Cortex-M debugger and TRACE32 software, follow the steps below:

1. Open the TRACE32 software and choose **ARM32 USB**.

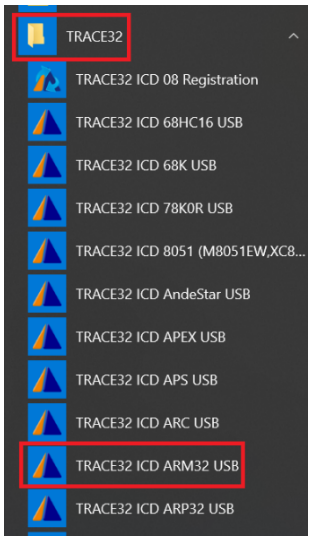

Figure 17. Opening TRACE32 for ARM32

2. Figure 18 shows the main page of TRACE32 for ARM32. If the status bar at the bottom of the main page shows **power down** instead of **system down**, check the power supply of the debugger and the connection with the JTAG interface of the target board.

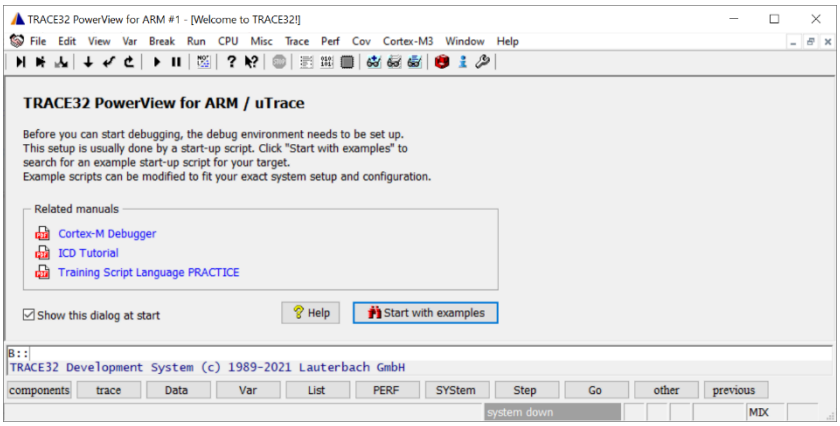

Figure 18. Main page of TRACE32 for ARM32

3. Click **CPU** -> **System Settings…** in the menu bar, and the system setting dialog appears. Perform system settings as shown in Figure 19.

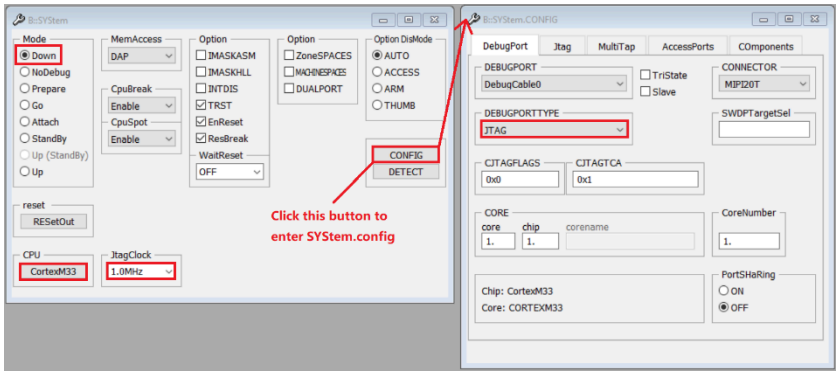

Figure 19. System setting

4. Type in below commands, with each command followed by entering:

```
BSDL.RESet
BSDL.ParkState Select-DR-Scan
BSDL.state
```

5. The **BSDL.state** window appears, as shown in Figure 20. Click the **FILE** button and load the BSDL file you want to validate.

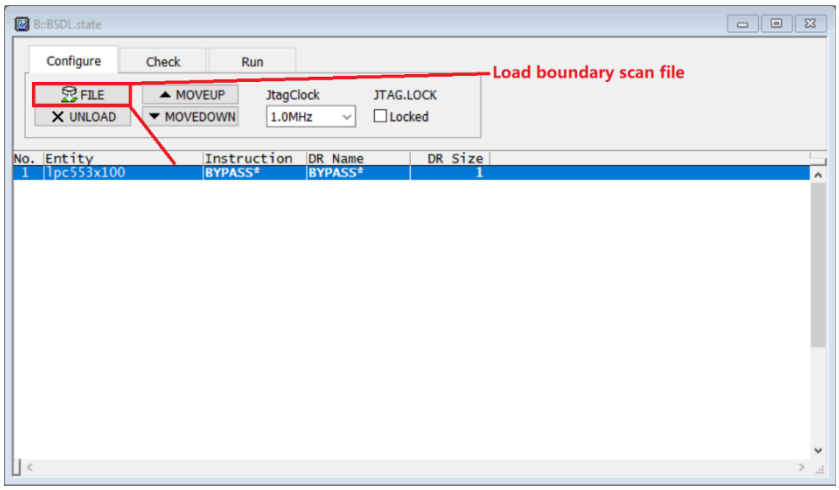

Figure 20. BSDL.state window

6. After loading the file, type in command as below:

```
BSDL.SOFTRESET
```

7. Switch to the **Check** tab of the **BSDL.state** window. To see whether both results can pass, click **BYPASSall** and **IDCODEall**, as shown in Figure 21 and Figure 22. Double-click the entity name in Figure 22, and the IDCODE test result can be seen in the **BSDL.SET** window, as shown in Figure 23.

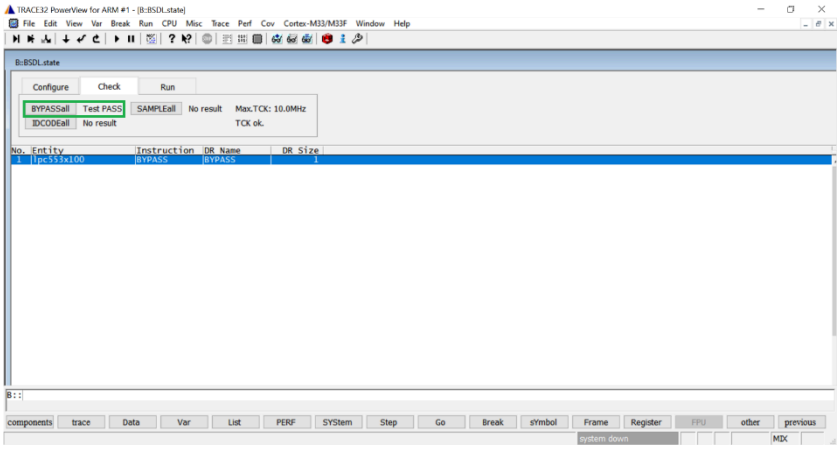

Figure 21. Checking BYPASS

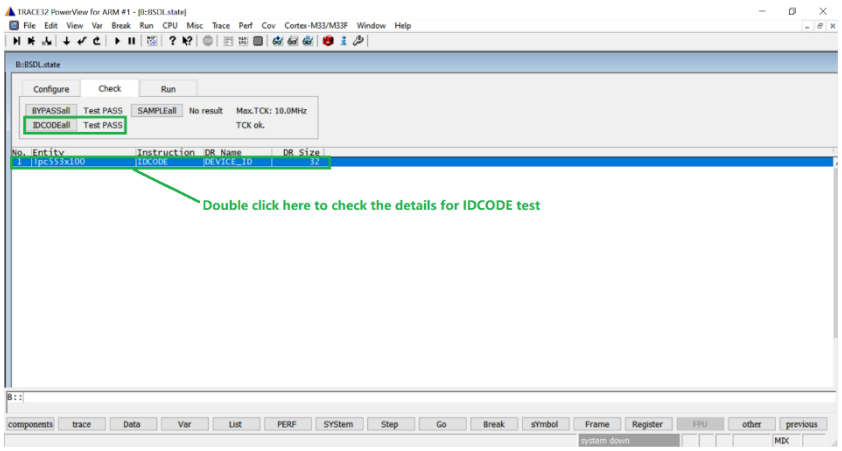

Figure 22. Check IDCODE

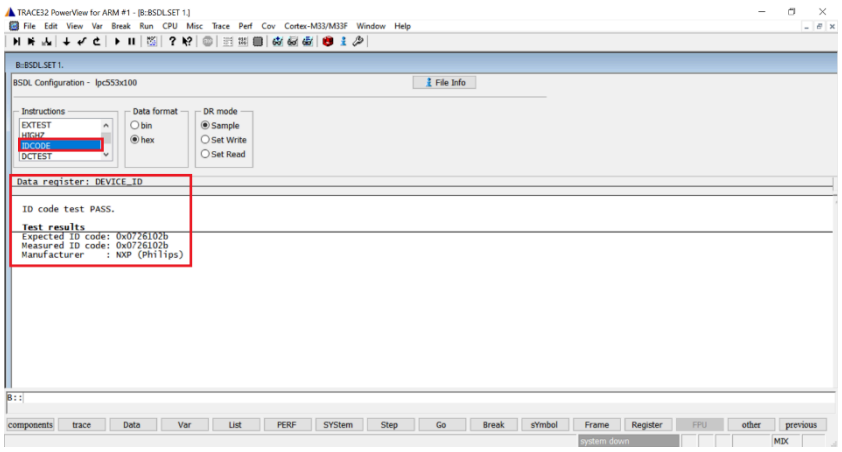

Figure 23. IDCODE test result

8. Click the **SAMPLEall** button, and **No result** becomes **Test done**. Double-click the entity name, as shown in Figure 24, and the SAMPLE test result can be seen in the **BSDL.SET** window, as shown in Figure 25.

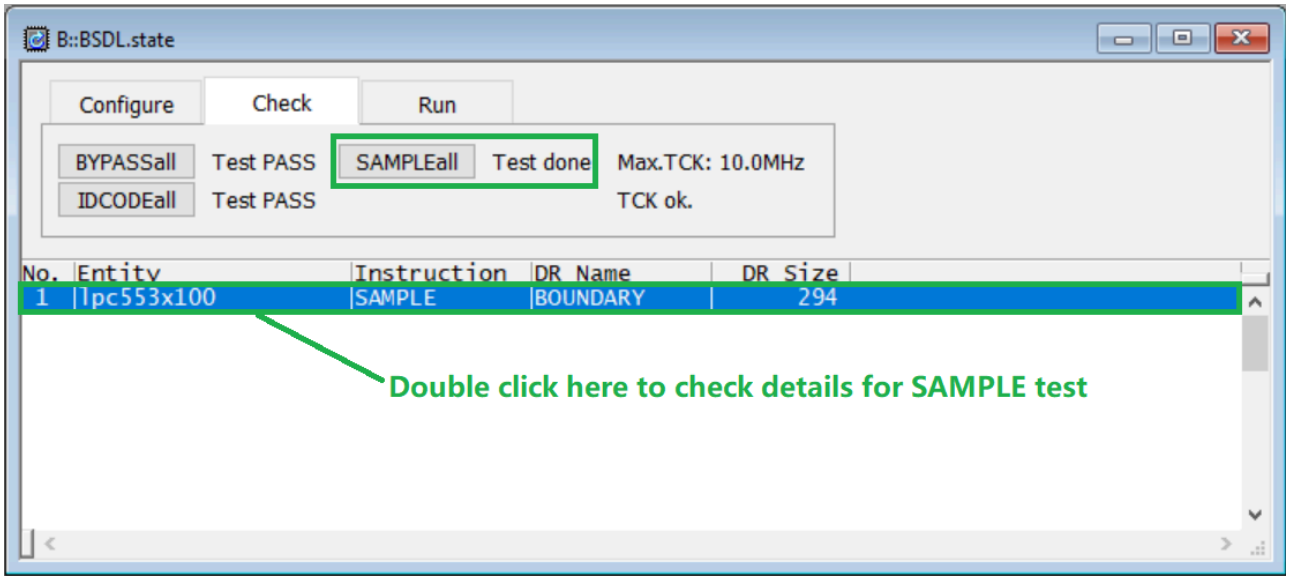

Figure 24. Checking SAMPLE

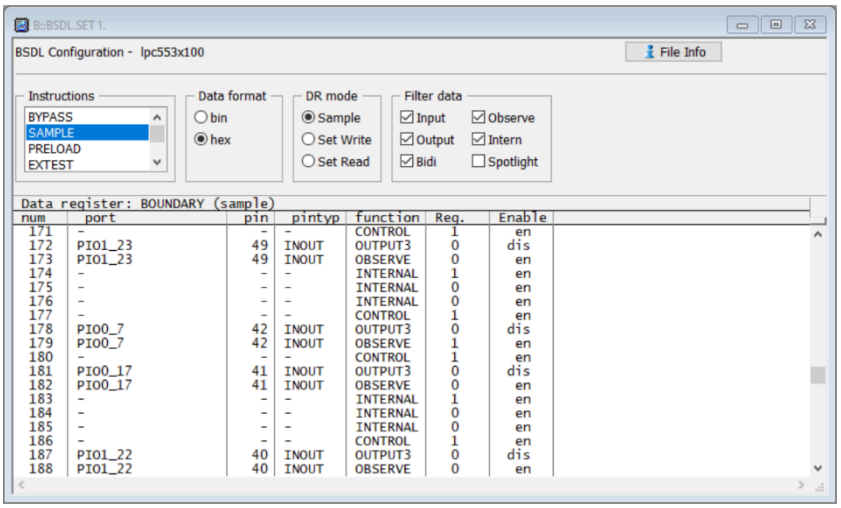

Figure 25. SAMPLE test results

To judge whether the SAMPLE test passes or not, take pin `P0_7` as an example to perform the following steps:

a. Apply high level to this pin. Use the `BSDL.RUN` command to run the **SAMPLE** test. Search the line where the port is `PIO0_7` and function is **OBSERVE**. Read the register value in the **Reg.** column and it is **1**, as shown in Figure 26.

b. Apply a low level to this pin. Check the sample result and it is **0**, as shown in Figure 27.

Therefore, the pin `P0_7` passes SAMPLE test.

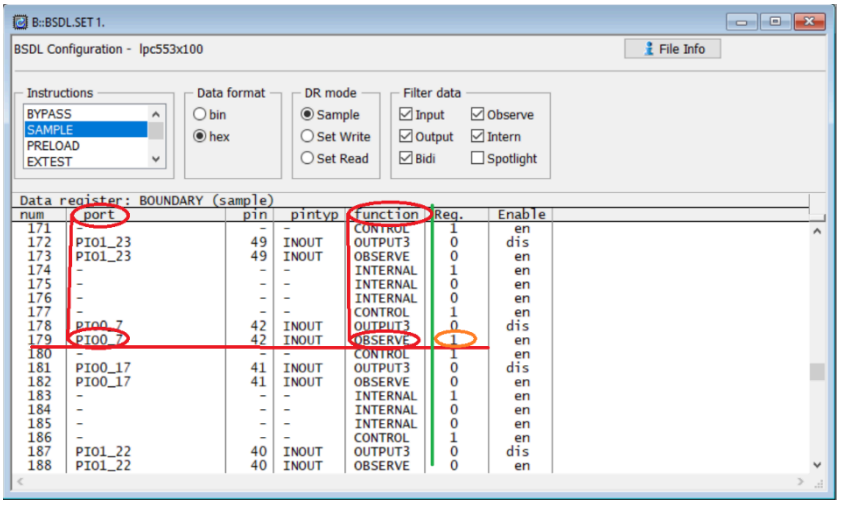

Figure 26. Sample high level on P0_7

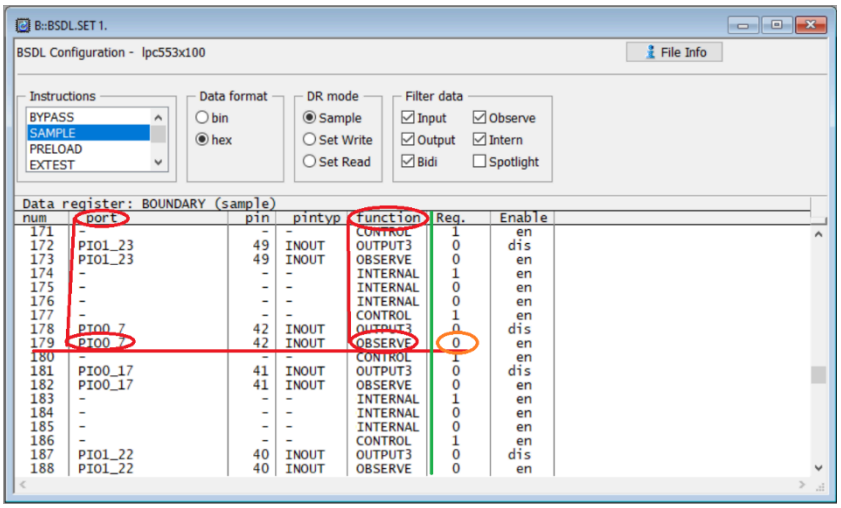

Figure 27. Sample low level on P0_7

Use the above method to traverse all the IO pins defined in the BSDL file. If all the IO pins pass the test, the SAMPLE test of BSDL passes.

9. Enter the `BSDL.SET` command on the **TRACE32** command line, and the **BSDL.SET** window appears. In the **Instructions** field, click **EXTEST** and in the **DR mode** field, choose **Set Write**, as shown in Figure 28.

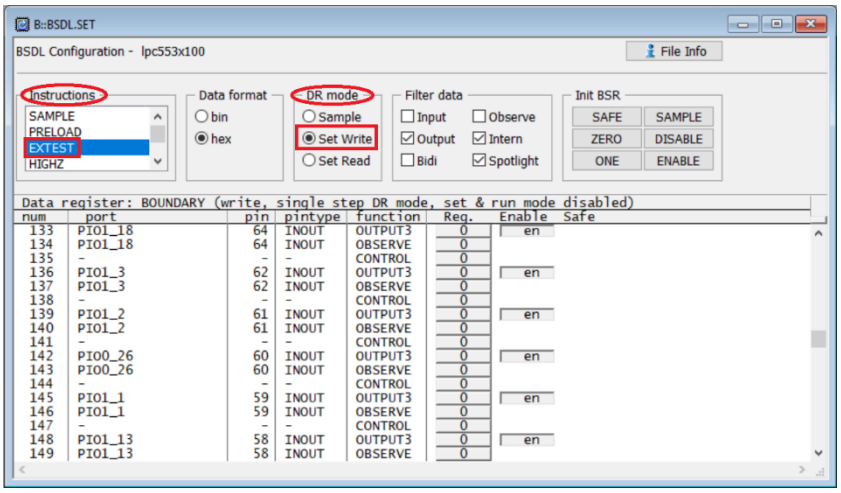

Figure 28. EXTEST settings in the BSDL.SET window

Switch to the **BSDL.state** window and check **SetAndRun** and **TwoStepDR**, as shown in Figure 29.

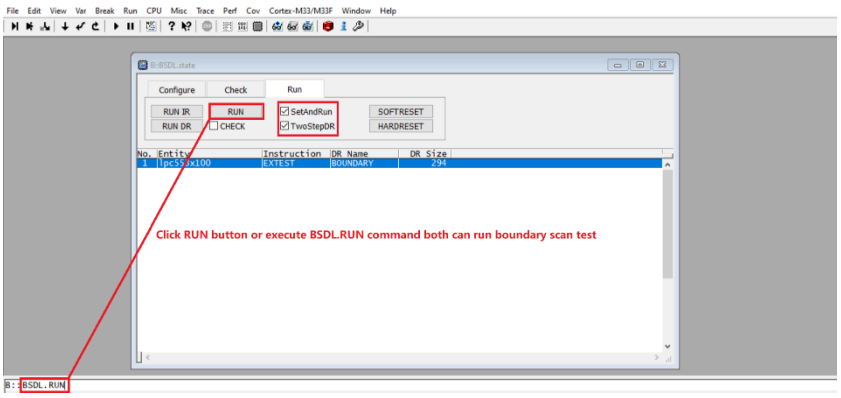

Figure 29. EXTEST settings in BSDL.state window

Switch back to the **BSDL.SET** window. Taking the pin `P0_7` as an example to describe how to perform **EXTTEST**. Enable output of the pin `P0_7` by clicking the button in the **Enable** column. Toggle the output logic state 0 or 1 of this pin by clicking the button in the **Reg.** column, as shown in Figure 30. Use a multimeter to measure whether the logic state really toggles on this pin.

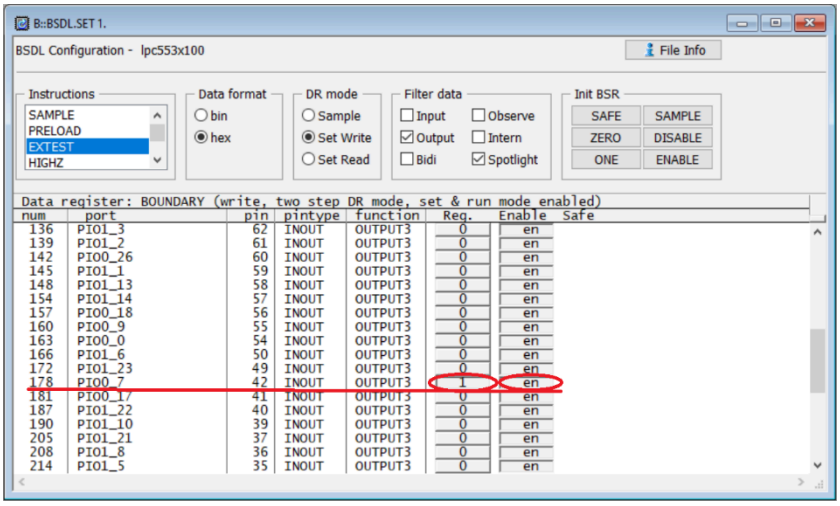

Figure 30. Enabling and toggling signal level

Use the above method to traverse all the IO pins defined in the BSDL file. If all the IO pins pass the test, the EXTTEST test of BSDL passes.

10. Use **PRELOAD** test with **EXTTEST**. To enable and preset driving buffers to 1 or 0 of all IO pins, click **ONE** or **ZERO** and then **ENABLE** in the **Init BSR** field, as shown in Figure 31.

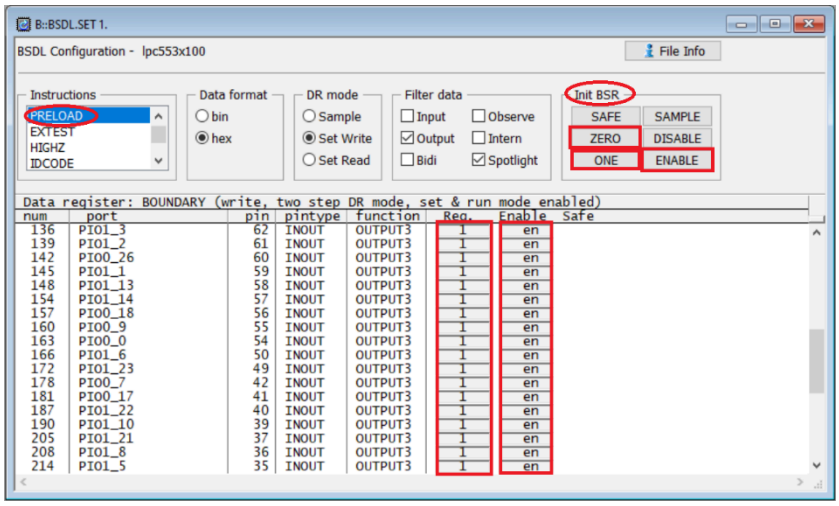

Figure 31. Preload IO output status

Run the **PRELOAD** test and then **EXTTEST**. Use a multimeter to measure whether the logic state of all IO pins matches the preload values.

Use the above method to traverse all the IO pins defined in the BSDL file. If all the IO pins pass the test, the PRELOAD test of BSDL passes.

11. Choose **HIGHZ** in the **Instructions** field in the **BSDL.SET** window and then run the **HIGHZ** test. All IO pins defined in the BSDL file are in the high-impedance state. Taking 3.3 V logic as an example, if an intermediate level, such as 1.65 V, is applied to a pin in a high-impedance state, use a multimeter to measure the pin. The measurement is 1.65 V and the pin in the high-impedance state is regarded as an open circuit due to its high impedance. It does not cause a significant voltage drop to the external driving source.

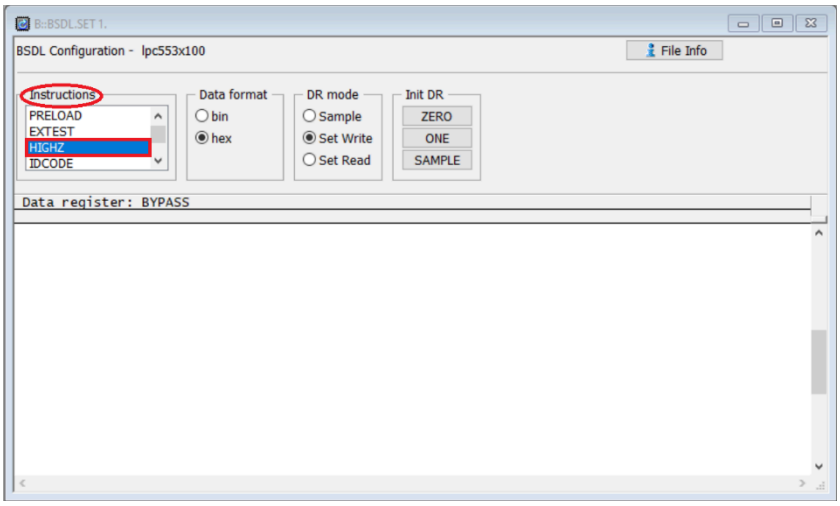

Figure 32. HIGHZ settings

After the **HIGHZ** test starts, all pins are in a high impedance state. Then select a pin and apply an intermediate level. Taking 1.64 V as is selected, use a multimeter to measure the pin level. The **HIGHZ** test result of `P0_7` pin is as shown in Figure 33.

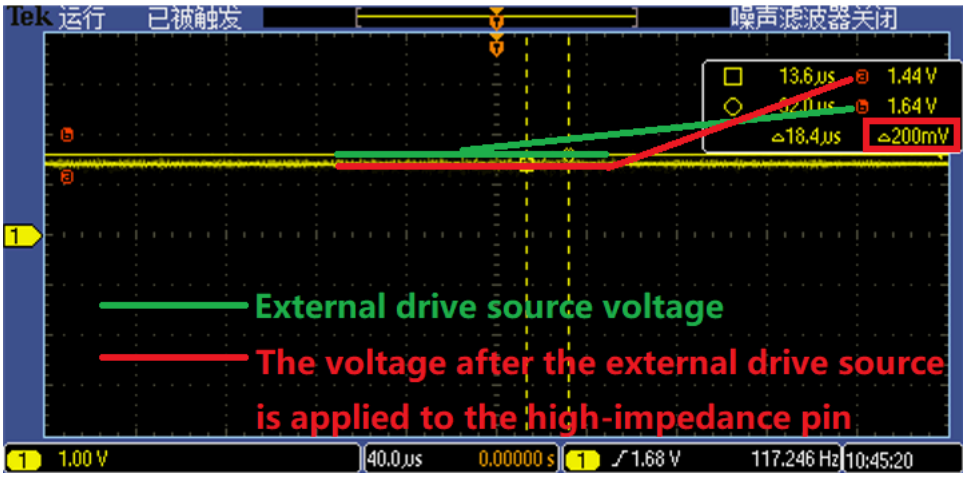

Figure 33. HIGHZ test result

As shown in Figure 33, when the external driving source is applied to the pin `P0_7`, it only causes a 200 mV voltage drop. `P0_7` is indeed in a high impedance state.

Use the above method to traverse all the IO pins defined in the BSDL file. If all the IO pins pass the test, the **HIGHZ** test of BSDL passes.

5 Let LPC55(S)xx pass boundary scan SAMPLE test

After LPC55(S)xx is reset, the digital input function of some pins is disabled. If the SAMPLE test is directly performed on these pins, these pins do not pass the test.

For LPC55(S)xx, the DIGIMODE bit of the IOCON register controls the digital input function, as shown Figure 34.

8 DIGIMODE	Select Digital mode 0 - Disable digital mode. Digital input set to 0. 1 - Enable Digital mode. Digital input is enabled.

Figure 34. DIGIMODE bit in IOCON register

Take LPC55(S)3x as an example to describe how to enable the digital input function of the pin `P0_1` when the chip is in boundary scan mode. The setting methods of other pins are similar. The detailed steps are described as follows:

1. Open **Jlink** commander.
2. For LPC55(S)3x, the clock enable the bit of `IOCON` is bit 13 in the `AHBCLKCTRL0` register with address `0x40000200`. If this bit is set to **1**, the clock for IOCON is enabled.

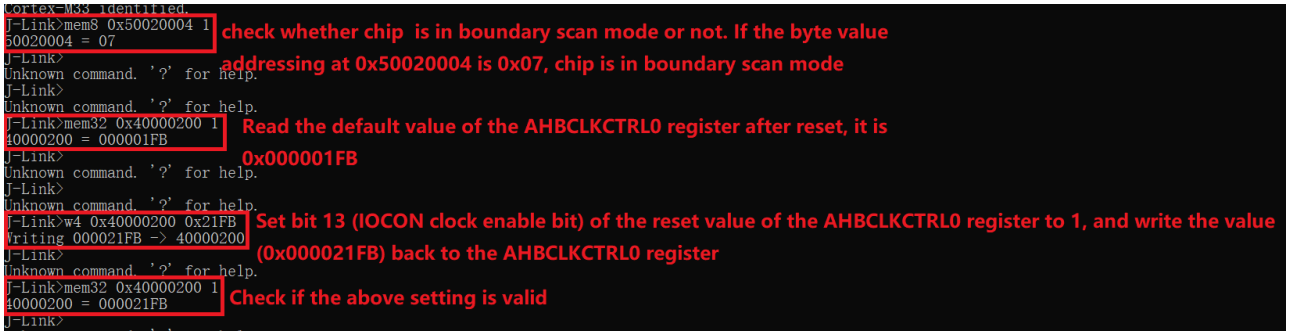

Figure 35. Enabling IOCON clock

3. The address of the `IOCON` register corresponding to the `P0_1` pin is `0x40001004`. To enable the digital input function, set bit 8 (DIGIMODE) of this register to **1**. To set the pin as floating input, pulldown input, and pullup input, set the bit field `[4:5]` to `0x00`, `0x01`, and `0x02` as required.

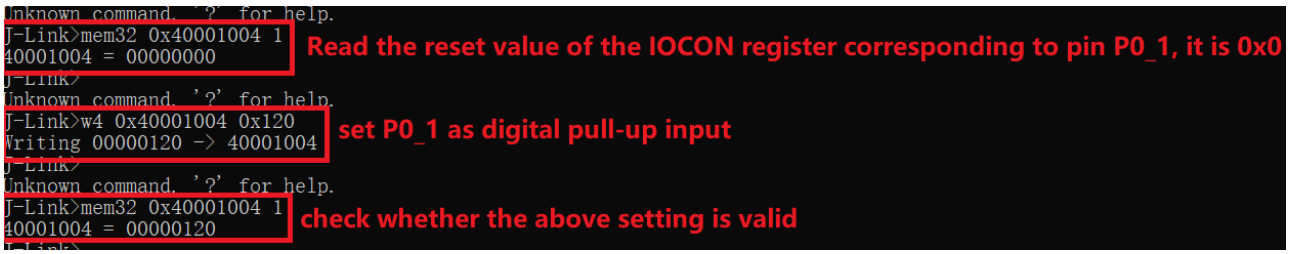

Figure 36. Setting P0_1 pin as digital pull-up input

6 Automated boundary scan test

Through the above introduction to the interactive boundary scan test, some interactive testing steps are not conducive for quick test. To improve test efficiency, TRACE32 supports a practice script. To perform an automated boundary scan test, write the script program.

On the main page of TRACE32, the **File** menu provides three submenu items related to script, **New Script**, **Open Script...**, and **Run Script...**. They are used to create, open, and run script.

A script example used to automate the boundary scan test is described as below.

```
;System setup
SYStem.Mode Down                    ;Disables the debug mode.
SYStem.CPU  CortexM33               ;Tells TRACE32 the exact CPU type
                                    ;used on your target, CPU core of
```

```
                                ;LPC553x is Cortex-M33.
SYStem.CONFIG.DEBUGPORTTYPE JTAG ;Specifies which probe cable shall
                                ;be used, here, JTAG is selected
SYStem.JtagClock    1MHz        ;Selects JTAG frequency (TCK)
;BSDL Settings
BSDL.RESet                      ;Initialize the boundary scan engine
BSDL.ParkState Select-DR-Scan   ;Set PartState as Select-DR-Scan
BSDL.state                      ;Open BSDL.state window
;Configure boundary scan chain
BSDL.FILE lpc553x100.bsdl       ;your BSDL file name
;Check boundary scan chain
BSDL.SOFTRESET
IF !BSDL.CHECK.BYPASS()         ;BYPASS Test
(
  BSDL.BYPASSall
  PRINT %ERROR "Bypass test failed"
  ENDDO
)
IF !BSDL.CHECK.IDCODE()         ;IDCODE Test
(
 BSDL.IDCODEall
 PRINT %ERROR "ID code test failed"
 ENDDO
)
;Perform SAMPLE test
BSDL.SAMPLEall
;Perform EXTTEST
;Pin output settings, you can add other pin output settings
BSDL.SET 1. PORT PIO0_7 0       ;Set PIO0_7 output as 0
BSDL.RUN DR                     ;Only apply data register settings
                                ;to the boundary scan chain
BSDL.SET 1. IR EXTEST           ;Only apply instruction register
                                ;settings to the boundary scan chain
BSDL.RUN                        ;BSDL run
;Perform HIGHZ test
BSDL.SET 1. IR HIGHZ            ;Only apply instruction register
                                ;settings to the boundary scan chain
BSDL.RUN                        ;BSDL run
```

7 Note about the source code in the document

Example code shown in this document has the following copyright and BSD-3-Clause license:

Copyright 2023 NXP Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. Neither the name of the copyright holder nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED

TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

8 Revision history

Table 4 summarizes the revisions to this document.

Table 4. Revision history

Revision number	Release date	Description
1	14 November 2023	• Updated images to svg format • Updated Section 3.2
0	07 January 2022	Initial public release

Legal information

Definitions

Draft — A draft status on a document indicates that the content is still under internal review and subject to formal approval, which may result in modifications or additions. NXP Semiconductors does not give any representations or warranties as to the accuracy or completeness of information included in a draft version of a document and shall have no liability for the consequences of use of such information.

Disclaimers

Limited warranty and liability — Information in this document is believed to be accurate and reliable. However, NXP Semiconductors does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information. NXP Semiconductors takes no responsibility for the content in this document if provided by an information source outside of NXP Semiconductors.

In no event shall NXP Semiconductors be liable for any indirect, incidental, punitive, special or consequential damages (including - without limitation - lost profits, lost savings, business interruption, costs related to the removal or replacement of any products or rework charges) whether or not such damages are based on tort (including negligence), warranty, breach of contract or any other legal theory.

Notwithstanding any damages that customer might incur for any reason whatsoever, NXP Semiconductors' aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the Terms and conditions of commercial sale of NXP Semiconductors.

Right to make changes — NXP Semiconductors reserves the right to make changes to information published in this document, including without limitation specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

Suitability for use — NXP Semiconductors products are not designed, authorized or warranted to be suitable for use in life support, life-critical or safety-critical systems or equipment, nor in applications where failure or malfunction of an NXP Semiconductors product can reasonably be expected to result in personal injury, death or severe property or environmental damage. NXP Semiconductors and its suppliers accept no liability for inclusion and/or use of NXP Semiconductors products in such equipment or applications and therefore such inclusion and/or use is at the customer's own risk.

Applications — Applications that are described herein for any of these products are for illustrative purposes only. NXP Semiconductors makes no representation or warranty that such applications will be suitable for the specified use without further testing or modification.

Customers are responsible for the design and operation of their applications and products using NXP Semiconductors products, and NXP Semiconductors accepts no liability for any assistance with applications or customer product design. It is customer's sole responsibility to determine whether the NXP Semiconductors product is suitable and fit for the customer's applications and products planned, as well as for the planned application and use of customer's third party customer(s). Customers should provide appropriate design and operating safeguards to minimize the risks associated with their applications and products.

NXP Semiconductors does not accept any liability related to any default, damage, costs or problem which is based on any weakness or default in the customer's applications or products, or the application or use by customer's third party customer(s). Customer is responsible for doing all necessary testing for the customer's applications and products using NXP Semiconductors products in order to avoid a default of the applications and the products or of the application or use by customer's third party customer(s). NXP does not accept any liability in this respect.

Terms and conditions of commercial sale — NXP Semiconductors products are sold subject to the general terms and conditions of commercial sale, as published at https://www.nxp.com/profile/terms, unless otherwise agreed in a valid written individual agreement. In case an individual agreement is concluded only the terms and conditions of the respective agreement shall apply. NXP Semiconductors hereby expressly objects to applying the customer's general terms and conditions with regard to the purchase of NXP Semiconductors products by customer.

Export control — This document as well as the item(s) described herein may be subject to export control regulations. Export might require a prior authorization from competent authorities.

Suitability for use in non-automotive qualified products — Unless this document expressly states that this specific NXP Semiconductors product is automotive qualified, the product is not suitable for automotive use. It is neither qualified nor tested in accordance with automotive testing or application requirements. NXP Semiconductors accepts no liability for inclusion and/or use of non-automotive qualified products in automotive equipment or applications.

In the event that customer uses the product for design-in and use in automotive applications to automotive specifications and standards, customer (a) shall use the product without NXP Semiconductors' warranty of the product for such automotive applications, use and specifications, and (b) whenever customer uses the product for automotive applications beyond NXP Semiconductors' specifications such use shall be solely at customer's own risk, and (c) customer fully indemnifies NXP Semiconductors for any liability, damages or failed product claims resulting from customer design and use of the product for automotive applications beyond NXP Semiconductors' standard warranty and NXP Semiconductors' product specifications.

Translations — A non-English (translated) version of a document, including the legal information in that document, is for reference only. The English version shall prevail in case of any discrepancy between the translated and English versions.

Security — Customer understands that all NXP products may be subject to unidentified vulnerabilities or may support established security standards or specifications with known limitations. Customer is responsible for the design and operation of its applications and products throughout their lifecycles to reduce the effect of these vulnerabilities on customer's applications and products. Customer's responsibility also extends to other open and/or proprietary technologies supported by NXP products for use in customer's applications. NXP accepts no liability for any vulnerability. Customer should regularly check security updates from NXP and follow up appropriately.

Customer shall select products with security features that best meet rules, regulations, and standards of the intended application and make the ultimate design decisions regarding its products and is solely responsible for compliance with all legal, regulatory, and security related requirements concerning its products, regardless of any information or support that may be provided by NXP.

NXP has a Product Security Incident Response Team (PSIRT) (reachable at PSIRT@nxp.com) that manages the investigation, reporting, and solution release to security vulnerabilities of NXP products.

NXP B.V. — NXP B.V. is not an operating company and it does not distribute or sell products.

Trademarks

Notice: All referenced brands, product names, service names, and trademarks are the property of their respective owners.

NXP — wordmark and logo are trademarks of NXP B.V.

Contents

For more information, please visit: https://www.nxp.com

Date of release: 14 November 2023
Document identifier: AN13507