# Using Ultra-XD for Synopsys DesignWare ARC Cores with tł

7 7

### C 1.

2.

3.

4.

5.

6.

7.

8.

<b>Je</b>	Met	taWare Debugger
	ents	
5110	Introdu	liction
		ation and Configuration
	2.1	32-bit versus 64-bit
	2.1	
	2.2	Windows Installation
	<u>.</u>	2.2.1 Windows™ USB Driver Installation
	2.3	Linux Installation
		2.3.1 Ultra-XD for ARC Driver Installation
		2.3.2 Installing Additional Required Libraries
		2.3.2.1 Centos/RHEL/Fedora Library Installation (32-bit)
		2.3.2.2 Ubuntu/Debian Library Installation (32-bit)
		2.3.2.3 Centos/RHEL/Fedora Library Installation (64-bit)
		2.3.2.4 Ubuntu/Debian Library Installation (64-bit)
	2.4	Using Ethernet with Ultra-XD
		2.4.1 Configuring Static IP for your Ultra-XD
	Debug	ging with MetaWare Debugger and Ultra-XD
	3.1	Connecting Ultra-XD to the Target
	3.2	Using the MetaWare Debugger (MDB)
		3.2.1 Getting started
		3.2.2 Trouble-shooting
		3.2.2.1 General
		3.2.2.2 Capturing Trace for Ashling Support Trouble-shooting
		3.2.2.3 Ultra-XD for ARC Diagnostic Utility
		3.2.2.3.1 Check for Ultra-XD
		3.2.2.3.2 Scan Chain Testing
		3.2.2.3.3 Memory Interface Testing
	RTT su	
	4.1	Configure RTT
	4.2	Search trace
	4.3	Saving/Logging trace
		D Firmware upgrade
	Conclu	
		dix A. Ultra-XD LEDs
		dix B. Ultra-XD Connection
	8.1	Trace Connector Pinning
	0.1	8.1.1 For Synopsys Real-time Trace v1
		8.1.2 For Synopsys Real-time Trace v2.
	8.2	JTAG Signal Timings
	8.3	Trace Signal Timings
	0.5	

Appendix C. CE Notice 9.

## 1. Introduction

This Ashling Product Brief (APB219) describes the usage of Ashling's Ultra-XD with the MetaWare Debugger (MDB). Ultra-XD is a powerful high-speed trace and run-time control debug probe for embedded development on Synopsys' DesignWare ARC<sup>™</sup> configurable RISC cores with the Real-time Trace extensions (RTT). Ultra-XD works with Synopsys' MetaWare Debugger for advanced embedded system debugging and analysis. Ultra-XD allows:

- Capture and viewing of program-flow and data-accesses in real-time, non-intrusively
- Download program from host PC to target embedded system
- Exercise program in target (go, step, halt, breakpoints, interrogate memory, registers and variables)

Synopsys' MetaWare IDE is a complete development environment for embedded C/C++ development on ARC<sup>™</sup> and includes an Eclipse based Integrated Development Environment, Compiler, and Debugging and Analysis tools.



Figure 1.Ultra-XD

Ultra-XD supports Gigabit Ethernet as well as USB2.0 High-speed host connections. This APB will look at using Ultra-XD and MetaWare Debugger with the Synopsys AXS board as a target system.



#### Figure 2. Synopsys AXS Board

## 2. Installation and Configuration

### 2.1 32-bit versus 64-bit

The Ashling drivers are available in 32-bit (x86) and 64-bit (x64) binary flavours. MetaWare 2019.12 onwards is required for 64-bit support.

### 2.2 Windows Installation

To install Ultra-XD software package, run the SETUP program located on the installation CD. Follow the on-screen instructions and the setup program will install the software in the directory of your choice. By default, the software is installed in the C:\AshlingUltra-XDforARC directory.

#### 2.2.1 Windows<sup>™</sup> USB Driver Installation

When you first connect Ultra-XD to your PC, you will get a **New USB hardware found** message and will be prompted to install the appropriate USB drivers. The Ashling Ultra-XD drivers are installed in your installation directory e.g. <*Installation path*>\usb. Direct the Windows **Hardware Installation Wizard** to this directory so that it can locate the necessary drivers and complete the installation. Windows only needs to perform this operation the first time you connect your Ultra-XD to the PC. The Ultra-XD USB driver is called libusb0.sys.

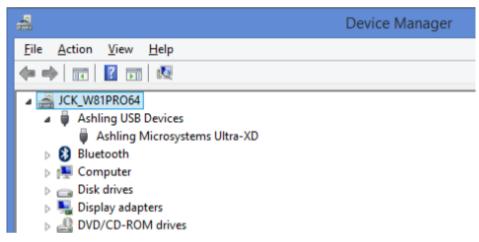


Figure 3. Ashling Ultra-XD USB Device Driver installed in Device Manager

### 2.3 Linux Installation

### 2.3.1 Ultra-XD for ARC Driver Installation

- Use the following command to extract the Ultra-XD software to the user home folder: \$ tar zxf AshlingUltra-XDforARCvX.Y.Z-[x86/x64].tar.gz -C ~/ where X.Y.Z indicates the software release number.
- 2. To ensure the current \$USER has access to the Ultra-XD device we recommend using the Linux utility udev
- 3. Ensure udev is installed and running on your system by checking for the udev deamon process (udevd) eg: \$ ps -aef | grep udev
- 4. Create a udev rules file to uniquely identify the Ultra-XD device and set permissions as required by owner/ groups. An example udev file is supplied (60-ashling.rules) which identifies Ultra-XD device (by Ashling's USB product ID and Vendor ID).
- 5. The rules file must then be copied into the rules directory (requires root permission) e.g.: \$ sudo cp ./60-ashling.rules /etc/udev/rules.d

### 2.3.2 Installing Additional Required Libraries

#### 2.3.2.1 Centos/RHEL/Fedora Library Installation (32-bit)

Install the required 32-bit libraries using the following command: \$ yum install libstdc++.i686 glibc.i686 gtk2.i686 qt5-qtbase.i686 qt5-qtbase-gui.i686 libusb.i686

#### 2.3.2.2 Ubuntu/Debian Library Installation (32-bit)

Install the required 32-bit libraries using the following command:

\$ sudo apt install libstdc++6:i386 libgtk2.0-bin:i386 qt5-default:i386 libusb-0.1-4:i386

#### 2.3.2.3 Centos/RHEL/Fedora Library Installation (64-bit)

Install the required 64-bit libraries using the following command: \$ yum install libstdc++ glibc gtk2 gt5-gtbase gt5-gtbase-gui libusb

#### 2.3.2.4 Ubuntu/Debian Library Installation (64-bit)

#### 2.4 Using Ethernet with Ultra-XD

By default, Ultra-XD powers up in DHCP mode and will acquire an IP address automatically if connected to a DHCP network. If a DHCP network is not available, then Ultra-XD will default to a Link-local address of 169.254.1.1/16. You may configure the IP address of Ultra-XD using the **Ashling Probe Configuration** dialog which is invoked when connecting to the target as shown below:

Probe Type		Ultra-XD	•				
				Jse specific debu	ug probe		
Communication	Interface	Ethernet	[	Scan for debu	ıg probes		
Network Interfa	ice	192.168.10.196	▼				
Specific IP							
Scanned Debug	Probe Details						
Serial No	IP Address	MAC Address	Default Protocol	Action	Firmware Version	Connection Speed	
		MAC Address 00:50:C2:02:27:9A	Default Protocol	Action Refresh	Firmware Version	Connection Speed	

Figure 4. Ashling Probe Configuration of Ultra-XD

Select the communication interface as **ETHERNET**. If your PC/laptop has multiple network adaptors, select the appropriate one via **Network Interfaces** and click the **Scan for Debug Probes** button. This will initiate a search for all connected Ultra-XDs. This will internally scan sing the multicast protocol (address: 226.0.01, receive-port 28007 and transmit-port 28008) to find the Ultra-XD probes connected to the network. If the probe is not listed, then you must manually enter the IP address of the probe you want to use by selecting **Use Specific Debug Probe**.

An Ultra-XD probe may not be discoverable due to:

- the probe is not on the same subnet as your current host machine
- your network adapter is not configured for multi-cast
- the routers servicing your network/firewall are blocking multi-cast packets

#### 2.4.1 Configuring Static IP for your Ultra-XD

You can configure a static IP address for your Ultra-XD irrespective of whether it is using USB or Ethernet as the current communication interface. Once you have done a scan, you have the probes listed. Select **Static** from the **Default Protocol** control as shown below.

Ashling Probe Co	onfiguration		- 15	-			? ×
Probe Configur	ation						
Probe Type		Ultra-XD	- U	se specific debu	g probe		
Communicatio	n Interface	Ethernet	<b>•</b>	Scan for debu	probes		
Network Inter	face	192.168.10.196	•				
Specific IP							
Scanned Debu	g Probe Details						
Serial No	IP Address	MAC Address	Default Protocol	Action	Firmware Version	Connection Speed	
1315048	192.168.10.200	00:50:C2:02:27:9A	рнср 👻	Refresh	v1.0.0	1Gbps	
			DHCP STATIC				
	Chec	k Connection			Upgrade Firm	ware to v1.0.0	
		(	Next	Cancel			

#### Figure 5. Configure Static IP address

If you select Static, a dialog box pops up as shown below, displaying the IP Address and Gateway.

Configure Ethernet Details	<b>×</b>
Serial Number	1315007
MAC ID	00:50:C2:02:37:99
IP Address	10 . 1 . 125 . 58
Subnet Mask	255 . 255 . 255 . 0
Gateway	10 . 1 . 125 . 254
OK	Cancel

#### Figure 6. Setting Static IP details

Change the IP address and Gateway according to your settings and click OK.

## 3. Debugging with MetaWare Debugger and Ultra-XD

## 3.1 Connecting Ultra-XD to the Target

Ultra-XD is designed to connect to the Target board via the supplied Target Probe Assembly (TPA) and 38-pin MICTOR extender cable as shown below (pinning for the expected 38-pin MICTOR target connector is given in **Appendix B.**):

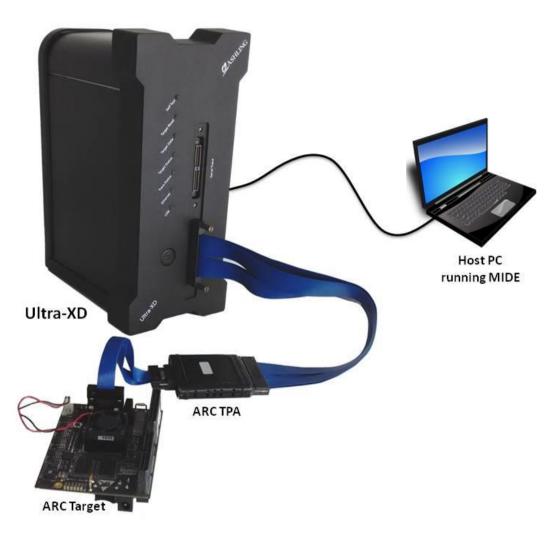


Figure 7.Ultra-XD connected to Target Board

Please note the following recommended target connection sequence:

- 1. Ensure your target and Ultra-XD are powered off.
- 2. Connect **Debug Trace** on the Ultra-XD front-panel to the target's mictor connector using the TPA as shown above.
- Power up Ultra-XD via the power button on the front-panel of Ultra-XD. The Self-test LED on Ultra-XD blinks orange during the self-test process followed by green during the final initialisation process. When successfully completed, the Self-test LED is green. See Appendix A. Ultra-XD LEDs for details on all Ultra-XD front-panel LEDs
- 4. Power up your target

## 3.2 Using the MetaWare Debugger (MDB)

#### 3.2.1 Getting started

1. Run MetaWare Debugger (MDB) and select **Debugger Options** 

	)
🏙 MetaWare Debugger	
File Display Tools Windows Help	
Command:    Project: <none loaded=""></none>	Stopped

Figure 8.MetaWare Debugger

2. Select program files by clicking on Program Options

🕱 Debugger options								
	Program & args 0 "C:\\AshlingUltra-XDforARC\\xplorer_sample_files\\I_2013_09\\smallprog\\Debug\\smallprog.elf" 🕶 🔯 🗃							
GUI Options	Source path C:\AshlingUltra-XDforARC\xplorer_sample_files\I_2013_09\smallprog\							
Semantic Inspection Target Selection	Directory translation -							
<ul> <li>Simulator Extensions</li> <li>Initialization</li> </ul>								
Breakpoints/Stepping	V Automatically execute to main if function main exists							
Peripheral displays AUX Registers	☑ Include local symbols from the ELF symbol table							
Side-effect registers	Program is already present; don't download							
laigerteset	Connect Only  E E BE							
	Verify program downloaded successfully							
	☑ Show register differences while instruction stepping							
	✓ Cache target memory							
	Read readonly from executable							
	V Prefer software breakpoints							
	V Show string contents							
	V Show possible breakpoints in source							
	Restore breakpoints from last run of same program     RTOS selection							
	RTOS (none)  (This allows debugger OS-awareness)							
Debugger Options: -hard	th=C:\AshlingUltra-XDforARC\xplorer_sample_files\_2013_09\smallprog\-loggle=include_local_symbols=1 -mexfersize=0x8000 -prop=hw_rtt=2 -profile							
Project: ultraxd	OK Cancel Help							

Figure 9.MetaWare Debugger: Program Options

3. Choose Target selection and select Ashling Ultra-XD JTAG for ARC from Choose hardware connection type.

🕱 Debugger options	
Program Options Command-Line Options	Select the target Hardware
GUI Options	Hardware connection
Semantic Inspection 	Choose hardware connection type
Simulator Extensions     Initialization	Ashling Ultra-XD JTAG for ARC
-Breakpoints/Stepping -Peripheral displays	
AUX Registers Side-effect registers	Ultra-XD ARC communication DLL
Target reset	Ultra-XD communication DLL C:\AshlingUltra-XDforARC_v16_internal\uxdarc.dll 🔹 🔞 🗃
	-Ultra-XD ARC JTAG frequencies
	JTAG frequency 5MHz    Note: stay at or below 1/2 of the ARC's operating frequency.
	Optimize JTAG speed by disabling certain status checking
	(none) • 🔞 Set timeout in milliseconds when adaptive clocking (RTCK) is used.
	50 🔹 🔞 Set reset pulse duration in milliseconds (1 to 5000). Default 50.
Debugger Options: -DLL=C	:\AshlingUltra-XDforARC_v16_internal\uxdarc.dll -memxfersize=0x8000 -prop=jtag_frequency=5MHz -prop=jtag_optimise=0
Project: ultraxd	OK Cancel Help

Figure 10.MetaWare Debugger: Target Selection

- 4. Next select the Ultra-XD communication DLL by browsing to the installation directory and selecting UXDARC.DLL (libashultraarc.so in Linux)
- 5. The Ultra-XD JTAG frequency can be set via JTAG frequency. It's best to set up your system initially with a low JTAG clock frequency, such as 1MHz. Once you have established communication with the target, you can increase the JTAG clock to the highest frequency that maintains stable and consistent operation. You should not select a JTAG frequency that is more than half of the ARC processor's clock frequency.
- 6. Click OK on debugger options and then on Debug a process or processes
- 7. You will then be presented with the **Ashling Probe Configuration** dialog as follows:

Probe Type		Ultra-XD	• 🛛 u	se specific debu	g probe		
Communicatio		Ethernet		Scan for debu	g probes		
Network Inter	ace	192.168.10.196	<b>•</b>				
Specific IP							
Serial No	J Probe Details	MAC Address	Default Protocol	Action	Firmware Version	Connection Speed	
		00:50:C2:02:27:9A	DHCP 🔻	Refresh	v1.0.0	1Gbps	
1315048	192.168.10.200		)(				

Figure 11. Probe Configuration

Ensure you are using the correct communication interface and select Next.

8. The Ashling Ultra-XD for ARC Configuration dialog box will appear as follows:

🌉 Ashling Ultra-XD for ARC Configuration	?	$\times$
Multi-core Support Configuration Devices On Scan Chair 1		
Device 1     ARC Device IR Width :4	$\sim$	^
		÷
Target Configuration Target Options ARC JTAG/Trace Target (using TPA-UX ARC cJTAG/Trace Target (using TF	(D- <b>A</b> R	
Enable RTCK Support Do Not Issue	TAP/R	ST* R
Trace Configuration Trace from Power-up Clear existing Enable Autolock	Trace	
OK Cancel		

Figure 12. Ultra-XD for ARC Configuration

- If your target system has multiple devices on the JTAG scan-chain then you need to specify the configuration of your scan-chain using the Multi-core Support Configuration group. Specify the number of Devices on Scan Chain and type of each device (i.e. ARC core or not); for devices that are not ARC cores you will need to specify the JTAG Instruction Register IR Width (typically, this is 4 bits). Up to a total of 128 devices are supported. By default, Ultra-XD will connect to and debug the first ARC core in the scan chain.
- Select the appropriate Target Options based on whether your target device uses JTAG or cJTAG
- Trace Configuration supports the following options:
  - **Trace from Power-up** allows you to trace from power-up and it requires that your boot-code enables the ARC core's RTT port out of reset via the appropriate sequence of RTT register writes. In this mode, Ultra-XD will wait until the target power is detected as on and immediately start tracing.
  - Clear existing Trace allows you to remove any existing trace before the new trace is captured.
  - Enable Autolock. High-speed trace data is routed from your ARC device pins to the mictor trace connector on your target system. Due to PCB tracking issues etc., there may be skews between trace data lines and the trace clock. These can cause setup/hold time violations and reduced eye width of trace data, which in turn can corrupt trace data being captured by Ultra-XD. To overcome this problem, Ultra-XD provides a mechanism named AUTOLOCK which allows automatic skew adjustment of trace data/clock lines to provide better integrity of parallel trace data captured. This option is off by default. Ashling recommend that you enable Autolock when using DDR. Autolock is not used (this setting is ignored) when using SDR.
- 9. Click **OK** and Ultra-XD will connect to your target. Run-time control debugging using Ultra-XD (run, stop, restart and step etc.) is now possible from the Debugger user-interface

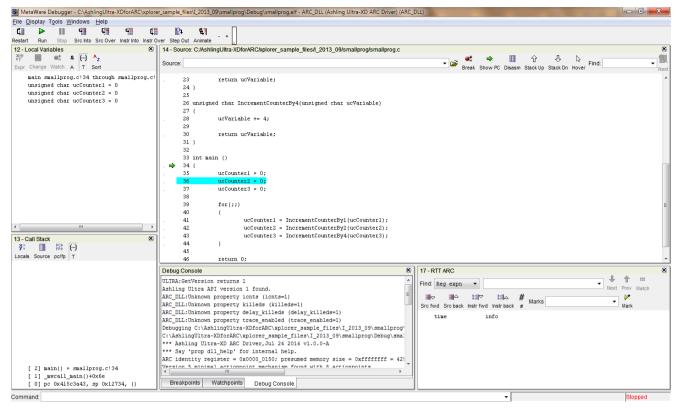


Figure 13.Debugger Interface

## 3.2.2 Trouble-shooting

#### 3.2.2.1 General

This section outlines some trouble-shooting tips when getting an error on target connection:

- 1. Ensure you have selected the correct options in the Ashling Ultra-XD for ARC Configuration as previously outlined.
- 2. Make sure the Ultra-XD is powered up and properly connected to both the host PC and the target as previously outlined
- 3. Make sure your target board is powered up
- 4. Check your MetaWare **Debugger Options**|Target Selection:
  - a. Make sure the Ultra-XD communication DLL is correct
  - b. Try using a lower JTAG clock frequency, such as 1MHz. Once you have established communication with the target, you can increase the JTAG clock to the highest frequency that maintains stable and consistent operation. You should not select a JTAG frequency that is more than half of the ARC processor's clock frequency.
  - c. Toggle the Optimise JTAG Access to see if this helps (recommended setting is on)
- Ashling supply a separate Ultra-XD for ARC Diagnostic Utility (details in the next section) which allows you to test your Ultra-XD can properly communicate with your ARC based target including basic memory and testing (this can be used before using MetaWare MDB).
- 6. If you require support from Ashling then:
  - a. Log extra debug messages in MDB by adding the following entries to the **Debugger Options**|Command-Line Options:
    - -prop=trace\_messages=1 -prop=trace\_board\_interface=1
    - and capture to a text-file
  - b. Take a screen-shot of your MetaWare debugger settings
  - c. Send the text-file/screen-shots to Ashling at: support@ashling.com

#### 3.2.2.2 Capturing Trace for Ashling Support Trouble-shooting

The ash\_save\_trace\_to\_bin command-line option allows you to capture Ultra-XD trace to a binary file for trouble-shooting issues with Ashling Technical Support.

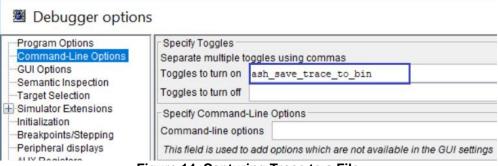


Figure 14. Capturing Trace to a File

The trace binary file used is called trace\_data.bin and is created in your Ultra-XD driver installation directory (e.g. C:\AshlingUltra-XDforARC\). Make sure you have write permissions for this directory (especially users using the Linux versions of the Ashling drivers).

You can also load trace from that file using the  $ash_load_trace_from_bin$  command-line option and this will be displayed in MDB.

#### 3.2.2.3 Ultra-XD for ARC Diagnostic Utility

The Ultra-XD for ARC Diagnostic Utility allows you to test your Ultra-XD can properly communicate with your ARC based target. It's available under Windows™ (UXDARC.EXE) and Linux (UXDARC) and installed as per the instructions in chapter 1 (note the Linux version requires that the GTK2 libraries and QT5 libraries are installed on your PC, see www.gtk.org)

tra-XD and Target Configuration hling Ultra-XD Interface DLL Location		JTAG Frequency	Connect
C:\AshlingUltra-XDforARC\uxdarc.dll	Browse	12 MHz 🔹	Discourse
		<ul> <li>Optimized JTAG Access</li> </ul>	Disconnect
		✓ Autoincrement JTAG Address	Reset Target

Figure 15. Ashling Ultra-XD for ARC Diagnostic Utility

Before using the utility, you need to tell it what Ashling Ultra-XD Interface DLL Location MetaWare Debugger driver to use (UXDARC.DLL for Windows or libashultraarc.so for Linux).

The **JTAG Frequency** allows you to specify the JTAG frequency used by Ultra-XD for target access. It's best to set up your system initially with a low JTAG clock frequency, such as 1MHz. Once you have established communication with the target, you can increase the JTAG clock to the highest frequency that passes **Memory Interface Testing** (see section 0). *Please Note:* You should not select a JTAG frequency that is more than half of the ARC processor's clock frequency.

The **Optimised JTAG Access** allows you to optionally turn on/off checking of the JTAG status registers by Ultra-XD between successive JTAG Read and Write operations. Ashling recommend that you turn this option on (i.e. disable JTAG status register checking) as it will improve performance. However, it is strongly recommended that you verify this will work properly for your target (see section 0).

Autoincrement JTAG Address allows you to specify that the JTAG interface on your target ARC device does not support auto-incrementing of addresses.

Once you are happy with your settings, press **Connect**. You will then be presented with the **Ashling Probe Configuration** dialog as follows:

Probe Type		Ultra-XD	• U:	se specific deb	ug probe		
Communicatio Network Inter		Ethernet		Scan for deb			
Specific IP							
canned Debu	) Probe Details						
		MAC Address	Default Protocol	Action	Firmware Version	Connection Speed	
Serial No	IP Address	MAC Address					
	IP Address 192.168.10.200	00:50:C2:02:27:9A	DHCP	Refresh	v1.0.0	1Gbps	

Figure 16. Ashling Probe Configuration

Ensure you are using the correct Ultra-XD and **Communication Interface** and select **Next**. The **Ashling Ultra-XD for ARC Configuration** dialog box will appear as follows:

Ashling Ultra-XD for ARC Configuration ?	$\times$
Multi-core Support Configuration Devices On Scan Chair 1  V	
Device 1   ARC Device IR Width 4	^
	÷
Target Configuration Target Options ARC JTAG/Trace Target (using TPA-UXD-AI ARC cJTAG/Trace Target (using TF	4
Enable RTCK Support Do Not Issue TAP/	'RST* R
Trace Configuration Trace from Power-up Clear existing Trace Enable Autolock	ce
OK Cancel	

Figure 17. Ashling Ultra-XD for ARC Configuration

This dialog allows you to:

- If your target system has multiple devices on the JTAG scan-chain then you need to specify the configuration
  of your scan-chain using the Multi-Core Support Configuration group. Up to a total of 1024 devices are
  supported. Specify the number of Devices on Scan Chain and type of each device (i.e. ARC core or not);
  for devices that are not ARC cores you will need to specify the JTAG Instruction Register IR Width (typically,
  this is 4 bits).
- The Target Configuration options allow to configure for use with JTAG or cJTAG targets

Click **OK** when you are complete. The Ashling diagnostic utility will now attempt to check for Ultra-XD as outlined in the following sections.

#### 3.2.2.3.1 Check for Ultra-XD

This test verifies that Ultra-XD is connected and attempts to communicate with and configure Ultra-XD for operation with an ARC target. Progress is indicated in the **Status Messages** window.

#### 3.2.2.3.2 Scan Chain Testing

The Scan Chain Details group shows you the Total Number of ARCs and the Total IR Length in your target (as specified via the Ashling Ultra-XD for ARC Configuration dialog) and allows you to perform the following tests:

#### Scan Chain Loopback Test

This test puts the TAP controller into bypass mode (by writing the bypass instruction to the IR register) and sends out a data pattern (using the DR register) on the JTAG scan-chain (via TDI) and verifies the pattern is returned okay (via TDO). This test verifies basic JTAG communication between Ultra-XD and your target and that your target JTAG scan-chain is functional. Results from the test are shown in the **Status Messages** window. If this test fails then:

- o Check that Ultra-XD is correctly connected to your target using the supplied TPA.
- Check that your target is correctly powered.
- Rerun the test (no effect) at lower frequencies (JTAG Frequency). Once you have established communication with the target, you can try to increase the JTAG clock to the highest frequency that passes the Scan Chain Loopback Test. You should not select a JTAG frequency that is more than half of the ARC processor's clock frequency.

#### Note: Scan Chain Loopback Test is limited to 512 cores

#### Read ARC ID Reg(s)

This test attempts to read the Auxiliary Identity register of each ARC on the scan-chain. Results from the test are shown in the **Status Messages** window. If the test fails then:

- Ensure settings in the Ashling Ultra-XD for ARC Configuration dialog match your target.
- Ensure the Scan Chain Loopback Test is passing (see previous section)

The following screen-shot shows the results of a successful test:

	Astin	ng ulua-Ab	) for ARC Diagnostic Utility	
Ultra-XD and Target Con Ashling Ultra-XD Interfac	-		JTAG Frequency	Connect
C:\AshlingUltra-XDforAR		Browse	12 MHz	Disconnect
			Autoincrement JTAG Address	Reset Target
Scan Chain Details	1			Scan Chain Loopback Test
Total Number of ARC's Total IR Length	4			Read ARC ID Reg(s)
Memory Interface Testing ARC core Memory Start Address	1 <b>•</b>	Memory End /	Address 0xFFFF	Test Memory Access
Status Messages Initializing connection Connected successfully ABC: 001 ID	register: 00000051			
Reading of ARC ID regi	ster(s) completed. st passed.			
Scan chain loopback te				
Scan chain loopback te				
Scan chain loopback te				
Scan chain loopback te				
Scan chain loopback te				

Figure 18. Ultra-XD Diagnostic Utility showing Scan Chain Loopback Test Results

#### 3.2.2.3.3 Memory Interface Testing

This option allows you to test target memory access via the Ultra-XD JTAG interface. Specify the **ARC** core (for Multi-core systems) and the target memory location to use via the **Memory Start Address** and **Memory End Address** controls. *Please Note:* Ensure these addresses are word aligned i.e. A1 and A0 are 0. For example:

- 0x0000 to 0x00FF is valid
- 0x0000 to 0x00F1 is not valid

Press the **Test Memory Access** button to begin the test. Progress is indicated in the **Status Messages** window. Note: the Ultra-XD driver generates random patterns for writing to memory. If this operation fails then:

- Ensure that Scan Chain Testing passes (see previous section).
- Rerun the test with Optimised JTAG Access off and at lower frequencies (JTAG Frequency). Once you
  have established communication with the target, you can try to enable Optimise JTAG Access and increase
  the JTAG clock to the highest frequency that passes Test Memory Access. You should not select a JTAG
  frequency that is more than half of the ARC processor's clock frequency.

## 4. RTT support

When used in conjunction with Ultra-XD, MDB supports real-time capture, reconstruction and display of trace, based on RTT information emitted from the target's trace port. The emitted trace data is captured by Ultra-XD, time stamped (optional), and transferred to the host PC. MDB will reconstruct the program flow based on the trace data, and display it in a readable format in its custom trace views.

## 4.1 Configure RTT

To configure RTT, right click RTT ARC window and select Configure producer and RTT capture....

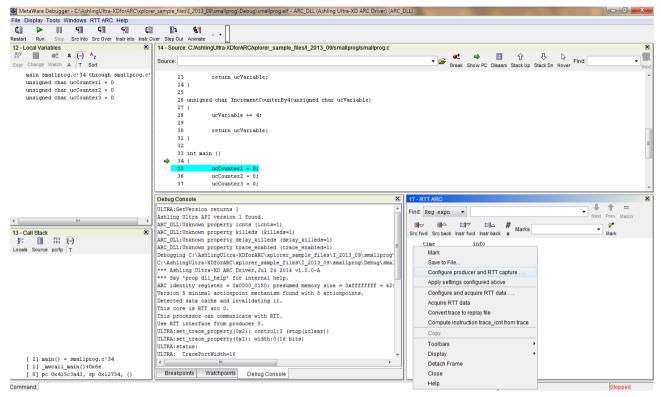


Figure 19. RTT Configuration for capturing

#### The **ARC RTT configuration** window will now be shown as in the following screenshot:

Configure	producer and RTT	capture			-						×
Trace	e PC e memory write e aux register write e core register write e debug status	Trace	e memory rea aux register r hpoint enab	ead	0xff v X						
Type0	Disabled •	Capture	Range 👻	Type1	Disabled 👻	Capture	Range 💌	Туре	Disabled	•	
Start			- 🗙	Start			- 🗙	LSW		Ŧ	×
Stop			- X	Stop			- 🗙	MSW		Ŧ	×
Type2	Disabled •	Capture	Range 🔻	ТуреЗ	Disabled -	Capture	Range 🔻	Туре	Disabled	•	
Start		, .	- 🗙	Start		, .	- ×	LSW		-	×
Stop			- 🗙	Stop			- 🗙	MSW		-	$\mathbb{X}$
Type4 Start Stop	Disabled -	Capture	Range	Type5 Start Stop	Disabled 🔻	Capture	Range				
Туреб	Disabled •	Capture	Range 🔻	Type7	Disabled 👻	Capture	Range 💌				
Start			- 🗙	Start			- 🗙				
Stop			- 🗙	Stop			- 🗙				
🔽 Confi	igure producer-inde	pendent	settings								
System	embus Trace m	nemory ad	dress 0x10_0	0000	▼ X Trace mer	nory size	0×10000	- 2	K 🗌 Wrap		
Oltra	🔲 Wrap 📃 🛙	DDR_EN (	lock Divide	1 -							
📰 Enabl	le time stamps 🔲	Trace deb	ug transactio	ns	Producer select mas Producer freeze enat			×			
ОКС	ancel										

Figure 20.RTTconfiguration selection

Note: DDR\_EN above enables support for Double-Data Rate (DDR, i.e. RTT trace data is emitted from the ARC core on the rising and falling edge of the trace clock). By default, RTT trace data is only emitted on the rising edge (i.e. Single-Data Rate or SDR). The Clock Divide specifies the trace clock ratio relative to the processor clock (e.g. when the value is two then it is half the frequency of the processor clock).

This dialog allows you to select the Trace sources and Address filter options. CPU trace can originate from up to six sources as follows:

- 1. PC
- : The instruction executed
- Memory write
   Memory read
   Auxiliary write
   Auxiliary read
   Control of the instruction of the instructio : Writing to memory from a core register.
- - : Reading memory to a core register.
- : Load value from an auxiliary register. 6. Core write : A value written to a core register as the result of a non-load instruction.

Filtering allows you to selectively include or exclude parts of your application from tracing and an Address filter can be used along with any of these six sources and can be specified using Type0 to Type7 in the UI. Options include a Range or a Trigger where:

- 1. A Trigger begins tracing when the Start address is executed and stops tracing when the Stop address is executed.
- 2. A Range implies a filter and trace information is only emitted when the programming is executing between the Start and Stop addresses

Data filter can be configured to use:

1. Memory write 2. Memory write

3. Memory write/read

- : the value written to memory (32 or 64 bits)
- : the value read from memory (32 or 64 bits)
- : the value written or read from memory.
- : the value written to the aux register : the value read from the aux register
- 4. Auxiliary write 5. Auxiliary read
- 6. Auxiliary write/read
- 7. Core write

: the value written /read from aux register.

8. Core read

- : the value written to the core register (32 or 64 bits).
- : the value read from the core register (32 or 64 bits).

32 bit data is specified using LSW and 64 bit data using LSW and MSW. To acquire trace data, right click over the RTT ARC window and select the Configure and acquire RTT data option as shown below:

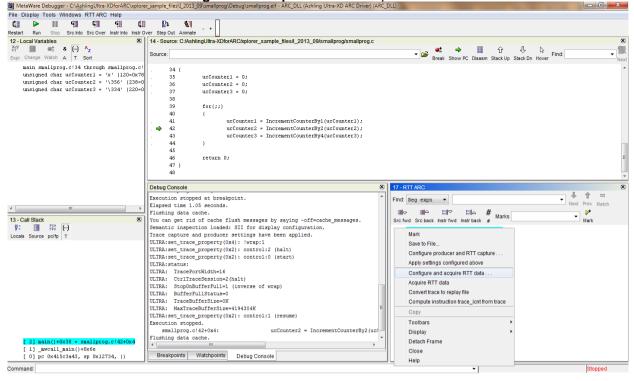


Figure 21.Acquiring RTT data

For more details on the dialog options, refer to the MetaWare Debugger Users Guide for ARC (Debugger\_Guide.pdf). This brings up the **Configure the acquisition** dialog as follows:

Acquire RTT o	
	ice (otherwise, it is overwritten) iy messages: 1_000_000_000 -> 💽
Note that a	e e, and write data to file rtt_capture0.bin ny acquisition from hardware trace from the hardware.
Acquire only	100000 👻 🙆 packets
From sav	ed disk files
	ce to replay file while acquiring after acquisition
Project: <i>ultraxd</i>	OK Cancel Help

Figure 22. Configure the acquisition

This dialog allows you to acquire from the Ultra-XD or previously saved trace files.

#### Once captured, trace may be viewed as below:

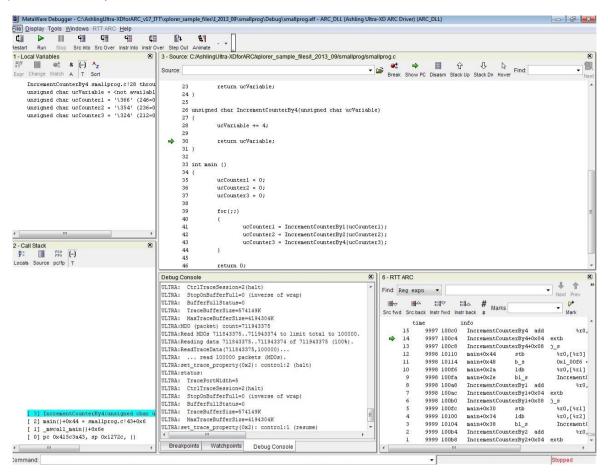


Figure 23. RTT View

### 4.2 Search trace

You can search trace frames captured via the **Find** button and specify only the entries that match an expression using **Match** as shown below:

	9_ITT\xplorer_sample_files\1_2013_09\smallprog\Debug\smallprog.elf - ARC_DLL (Ashling Ultra-XD ARC Driver) (ARC_DLL)	
le Display Tools Windows Source Help 1월 🕨 💵 대표 대표 대회	N. +=	
and a set the first test date		
start Run Stop Src Into Src Over Instr Into Instr O	6-RTT ARC	
	Find: Reg expn	
xpr Change Watch A T Sort		
<pre>main smallprog.c!34 through smallprog.c unsigned char ucCounter1 = '.' (46=0x2e</pre>	Src fwd Src back instr fwd instr back # Marks	
unsigned char uccounter1 = '.' (40=0x2e unsigned char ucCounter2 = '\' (92=0x5c	N New York Street Stree	
unsigned char ucCounter3 = '\270' (184=)	time info 43 184140 000100fc -> write r31	
annen ar seastan and a statistic state same a statistic state same	45 104140 0001000 - 001000 134 42 184141 100a8 IncreantCounterByl add %r0,*r0,1	
	41 184141 0000002e -> write r0	
	40 184142 100ac IncrementCounterByl+0x04 extb %r0,%r0	
	39 184142 0000002e -> write r0	
	38 184143 100b0 IncrementCounterBy1+0x08 j_s [%blink]	
	a)         37         184144         100fc         main+0x30         stb         %r0,[%r1]           36         184144         2e -> write mem [0x011000]         %r0,[%r1]	
	35 184145 10100 main+0x34 1db %r0,[%r2]	
	34 184145 5a <- read mem [0x011001] -> write r0	
	33 184146 0000005a -> write r0	
	32 184147 10104 main+0x38 bl_s IncrementCounterBy2 = smallprog.c!21	
	31 184147 00010106 -> write r31 30 184148 100b4 IncrementCounterBy2 add %r0,%r0,2	
4 m	29 184148 000005c -> write r0	
	28 184149 100b8 IncrementCounterBy2+0x04 extb \$r0, \$r0	
Local Variables	27 184149 0000005c -> write r0	
Call Stack	26 184150 100bc IncrementCounterBy2+0x08 j_s [%blink]	
¥: Pp (~)	25 184151 10106 main+0x3a stb %r0,[%r2]	
cals Source pc/fp T	Source Source RTT ARC	
	Debug Console X 7 - Source: C/AshlingUltra-XD/orARC_V0.0.19_ITT/xplorer_sam	The Block Data
	ULTRA:Reading data 15560229581556154029 of 1556498670 (65%).	
	Source:	ow PC Disasm
	ULTRA: read 131072 packets (MDOs).	on to block
	ULIRA:Reading data 15561540501556255101 01 1556495670 (754).	
	ULTRA: ReadTraceData (1556154030,131072) 29 ULTRA: read 131072 packets (MDOs). 30 return ucVariable;	
	ULTRA:Reading data 15562851021556416173 of 1556498670 (91%). 31 )	
	ULTRA: ReadTraceData (155625102, 131072) 32	
	ULTRA: read 131072 packets (MDOs). 33 int main ()	
	ULTRA:Reading data 15564161741556498669 of 1556498670 (100%). 34 (	
	ULIRA:ReadTraceData(1556416174,82496) 35 ucCounter1 = 0; ULIRA: mad \$2405 making (MDa) 36 ucCounter2 = 0;	
	oblikk lead 62456 packets (hbos).	
	ULTRA:set_trace_property(0x2): control:2 (halt) 37 uccounters = 0; ULTRA:status: 38	
	ULTRA: TracePortWidth=8 39 for(;;)	
	ULTRA: CtrlTraceSession=2(halt) 40 {	
	ULTRA: StopOnBufferFull=0 (inverse of wrap) 41 ucCounterl = IncrementCo	
	ULTRA: BufferFullStatus=0 42 ucCounter2 = IncrementCo ULTRA: TransBufferSica=0002070W 43 ucCounter3 = IncrementCo	
<pre>[ 2] main()+0x30 = smallprog.c!41+0x6</pre>	OLIKA: IFACEBUILEFSIZE=20920/0K	uncerby4(ucco
<pre>[ 1] mwcall main()+0x6e</pre>	ULTRA: MaxTraceBufferSize=4194304K 44 } ULTRA:set_trace_property(0x2): control:1 (resume) + 45	
[ 0] pc 0x415c3a43, sp 0x12728, ()		

Figure 24. Trace Window

e Display Tools Windows RTT ARC Help	
start Run Stop Src Into Src Over Instr Into Instr C	
Local Variables	
<sup>gy</sup> 🔟 🐠 & () <sup>A</sup> z	Find Reg expn v Increment
pr Change Watch A T Sort	Next Prev hoMatch
main smallprog.c!34 through smallprog.c	
unsigned char ucCounter1 = '.' (46=0x2e	Src fwd Src back Instr fwd Instr back # Mark
unsigned char ucCounter2 = '\' (92=0x5c	
unsigned char ucCounter3 = '\270' (184=)	649006 5 10104 main+0x38 bl_s IncrementCounterBy2 = smallprog.ct21
	649004 6 100b4 IncrementCounterBy2 add %r0,%r0,2
	649002 7 100b8 IncrementCounterBy2+0x04 extb %r0,%r0
	649000 8 100bc IncrementCounterBy2+0x08 j_s [%blink]
	648994 12 1010e main+0x42 bl_s IncrementCounterBy4 = smallprog.c!28
	648992 13 100c0 IncrementCounterBy4 add %r0,%r0,4
	648990 14 100c4 IncrementCounterBy4+0x04 extb %r0,%r0
	648988 15 100c8 IncrementCounterBy4+0x08 j_s [%blink]
	648978 7 1010e main+0x42 bl_s IncrementCounterBy4 = smallprog.c!28
	648976 8 100c0 IncrementCounterBy4 add %r0,%r0,4
	648974 9 100c4 IncrementCounterBy4+0x04 extb %r0,%r0
	648972 10 100c8 IncrementCounterBy4+0x08 j_s [%blink] 648965 14 100fa main+0x2e bl s IncrementCounterBy1 = smallprog.c!14
	648965 14 100fa main+0x2e bl_s IncrementCounterBy1 = smallprog.c!14 648963 15 100a8 IncrementCounterBy1 add %r0,%r0,1
4 III	649951 15 100a0 IncrementConterBy14004 extb \$10,570
	648959 17 100b0 IncrementCounterly1000 is [Vilink]
ocal Variables Local Variables	648952 5 1010e main+0x42 bl s IncrementCounterBy4 = smallprog.c!28
Call Stack	
	648948 7 100c4 IncrementCounterBy4+0x04 extb %r0,%r0
als Source pc/fp T	Source Source RTT ARC
	Debug Console 8 7 - Source: C:/AshlingUltra-XDforARC V0.0.19 ITT/xplorer sample files/I 2013
	ULTRA:Reading data 15560229581556154029 of 1556498670 (65%).
	ULIRA:ReadTraceData(1556022958,131072)
	ULTRA: read 131072 packets (MDOs).
	ULTRA:Reading data 15561540301556285101 of 1556498670 (78%). 28 ucVariable += 4;
	DLIAA:ReadiraceData(1556154050,1310/2)
	ULIKA: Fead ISID/2 packets (NDOS).
	ULIKA:Reading data 155625510215564161/5 DI 15564966/0 (914).
	ObinAckeddiraceData (1556265102,1510/2)
	obina: read 1510/2 packets (mbos).
	Chikekeeling data issterii/eisstersters ti isstersters (itse).
	011AR. Keddilacebaca (15504101/4, 02450)
	ULTRA: read 82496 packets (MDOs). 36 uccounter2 = 0; ULTRA:set trace property(0x2): control:2 (halt) 37 ucCounter3 = 0;
	UIRA:se_ciade_property(uz): Concors2 (naic) 38
	ULTRA: TracePortWidth=8 39 for(;;)
	ULTRA: Independent data ULTRA: Curlinger Series (Later data ULTRA: Curlinger Series and Curlinger Series (Later data ULTRA: Curlinger Series and Curlinger Series (Later data ULTRA: Curlinger Series and Curlinger Series
	ULTRA: StopOnBufferFull=0 (inverse of wrap)
	ULTRA: BufferFullStatus=0 42 ucCounter2 = IncrementCounterBy2(ucCo
	ULTRA: TraceBufferSize=2092070K 43 ucCounter3 = IncrementCounterBy4(ucCo
<pre>[ 2] main()+0x30 = smallprog.c!41+0x6</pre>	ULTRA: MaxTraceBufferSize=4194304K = 44 )
<pre>[ 2] main()+0x30 = smallprog.c'41+0x6 [ 1] _mwcall_main()+0x6e [ 0] pc 0x415c3a43, sp 0x12728, ()</pre>	ULTRA: MaxTraceBufferSize=4194304K 44 } ULTRA:set trace property(0x2): control:1 (resume) 45

Figure 25. Trace showing search results

## 4.3 Saving/Logging trace

You can log/save the captured trace data to a text file using the RTT ARC Save to File menu as shown below:

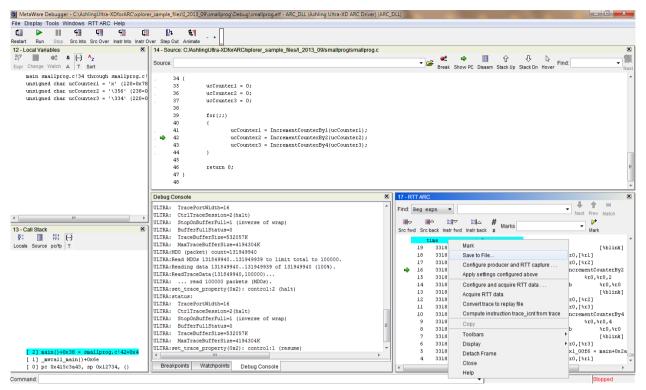


Figure 26. Log Trace Data

## 5. Ultra-XD Firmware upgrade

Ashling Probe Configuration allows users to manually upgrade the firmware of the Ultra-XD debug probe. Manual firmware upgrade can be done through by **Upgrade Firmware to vx.y.x** (where x.y.z denotes the latest version). Upgrading when prompted is strongly recommended as newer versions of MDB (or MIDE) may not function with older versions of firmware.

Probe Type	Interface	Ultra-XD Ethernet		lse specific det	oug probe		
letwork Interfa		192.168.10.196		Scan for deb	oug probes		
Specific IP							
icanned Debug							
Serial No	IP Address	MAC Address 00:50:C2:02:27:9A	Default Protocol	Action	Firmware Version	Connection Speed	
	103 109 10 300		DHCP 🔻	Refresh	VI.0.0	1Gbps	
1315048	192.168.10.200	00.50.C2.02.21.5K					

Figure 27. Firmware Upgrade

## 6. Conclusion

This APB shows the debugging capabilities of Ultra-XD debug/trace probe when used in-conjunction with MDB. Powerful features such as real-time trace capture are easily configured and used from within MDB's user-interface. These features allow real-time, non-intrusive debug and analysis of your ARC processor based embedded application, thus helping you to achieve on-time delivery to market. We hope you like it! Please send your feedback to hugh.okeeffe@ashling.com

# 7. Appendix A. Ultra-XD LEDs

LED	State	Meaning				
	Off	No Power connected to Ultra-XD				
	Orange Blinking	Power-On-Self-Test (POST) in progress				
Self-test	Green Blinking	Power-On-Self-Test (POST) OK. Ultra-XD booting in progress				
	Green Permanent	Ultra-XD booted OK				
	Red	Internal error				
Target Reset	Off	Target reset not asserted				
	Red Blink	Target reset asserted				
	Off	No traffic between Ultra-XD and Target				
Target Data	Green	Traffic between Ultra-XD and Target (e.g. TDI).				
	Red	Traffic between Target and Ultra-XD (e.g. TDO)				
	Orange	Traffic in both directions (normal operation)				
	Off	Target reference voltage not detected.				
Target Status	Green	Target reference voltage detected				
	Red Blink	Target status changed e.g. go/stop or stop/go				
	Off	Ultra-XD Trace not configured				
Trace Status	Green	Ultra-XD Trace configured (Ready)				
	Red	Ultra-XD Trace buffer full				
	Orange	Trace in progress				
	Off	Ethernet not selected				
Ethernet	Green	Ethernet selected				
Linemet	Red	Ethernet error				
	Orange Blink	Ethernet transactions ongoing				
	Off	USB not selected				
USB	Green	USB selected				
	Red	USB error				
	Orange Blink	USB transactions ongoing				

The following table describes the LEDs on the Ultra-XD:

Table 1.Ultra-XD LEDs

## 8.1 Trace Connector Pinning

### 8.1.1 For Synopsys Real-time Trace v1

Ultra-XD is designed to connect to a 38-pin MICTOR on your target board as shown in the following table. Voltages in the range 0.9V to 3.6V are supported.

#	JTAG/Trace Pinning	cJTAG/Trace Pinning	Comment
1	-	-	No connect
2	-	-	No connect
3	-	-	No connect
4	-	-	No connect
5	GND	GND	Required
6	МСКО	МСКО	Required
7	EVTI	EVTI	No connect
8	EVTO	EVTO	No connect
9	/RESET	/RESET	Optional. Required for target reset assertion (via software debugger) and detection support. This bi- directional signal is an open-drain output with an internal 470K $\Omega$ pull-up to an internal voltage equivalent to VREF. In addition, there is a 3.3 $\Omega$ series resistor. See note 1.
10	-	-	No connect
11	TDO	TDO not required for cJTAG	Required for JTAG
12	VREF_TRACE	VREF_TRACE	Required
13	RTCK	RTCK	Optional. Required for adaptive clocking support for target systems that provide a returned TCK (RTCK). When selected, Ultra-XD will wait for RTCK before sending a subsequent TCK pulse.
14	VREF_JTAG	VREF_JTAG	Required
15	TCK	тск	Required. See note 1.
16	MDO7	MDO7	Required for 8-bit and 16-bit trace support
17	TMS	TMS	Required. See note 1.
18	MDO6	MDO6	Required for 8-bit and 16-bit trace support
19	TDI	TDI not required for cJTAG	Required for JTAG. See note 1.
20	MDO5	MDO5	Required for 8-bit and 16-bit trace support
21	/TRSTN	/TRSTN	JTAG reset. Ultra-XD will pull this pin high to enable JTAG. See note 1.
22	MDO4	MDO4	Required for 8-bit and 16-bit trace support
23	MDO15	MDO15	Required for 16-bit trace support
24	MDO3	MDO3	Required for 4-bit, 8-bit and 16-bit trace support
25	MDO14	MDO14	Required for 16-bit trace support
26	MDO2	MDO2	Required for 4-bit, 8-bit and 16-bit trace support
27	MDO13	MDO13	Required for 16-bit trace support
28	MDO1	MDO1	Required for 4-bit, 8-bit and 16-bit trace support
29	MDO12	MDO12	Required for 16-bit trace support
30	MDO0	MDO0	Required for 4-bit, 8-bit and 16-bit trace support
31	MDO11	MDO11	Required for 16-bit trace support
32	-	-	No connect
33	MDO10	MDO10	Required for 16-bit trace support
34	-	-	No connect
35	MDO9	MDO9	Required for 16-bit trace support
36	MSEO1	MSEO1	Required for 4-bit, 8-bit and 16-bit trace support
37	MDO8	MDO8	Required for 16-bit trace support
38	MSEO0	MSEO0	Required for 4-bit, 8-bit and 16-bit trace support

#### Table 2. RTT v1 38-way MICTOR pinning for JTAG and cJTAG

**Note 1:** To provide a defined state on the debug-input pins to the ARC core when the Ultra-XD isn't connected, pull-up resistors should be fitted to TDI, TMS, TCK, /TRSTN, /RESET pins on the target board (typically  $10K\Omega$ ).

#### 8.1.2 For Synopsys Real-time Trace v2.

v2 supports both ARC Trace (ARCT) and DesignWare SoC Trace (DWT) and was released in Q4' 2018 with single and dual x8-bit NEUX AUX port support. Ultra-XD supports v2 and is designed to connect to a 38-pin MICTOR on your target board as shown in the following table. Voltages in the range 0.9V to 3.6V are supported.

#	JTAG/Trace Pinning	cJTAG/Trace Pinning	Comment
1	-	-	No connect
2	-	-	No connect
3	-	-	No connect
4	-	-	No connect
5	GND	GND	Required
6	MCKO	МСКО	Required
7	EVTI	EVTI	No connect
8	EVTO	EVTO	No connect
9	/RESET	/RESET	Optional. Required for target reset assertion (via software debugger) and detection support. This bi- directional signal is an open-drain output with an internal $470$ K $\Omega$ pull-up to an internal voltage equivalent to VREF. In addition, there is a 3.3 $\Omega$ series resistor. See note 1.
10	-	-	No connect
11	TDO	TDO not required for cJTAG	Required for JTAG
12	VREF_TRACE	VREF_TRACE	Required
13	RTCK	RTCK	Optional. Required for adaptive clocking support for target systems that provide a returned TCK (RTCK). When selected, Ultra-XD will wait for RTCK before sending a subsequent TCK pulse.
14	VREF_JTAG	VREF_JTAG	Required
15	TCK	ТСК	Required. See note 1.
16	MDO_A7	MDO_A7	Required for single/dual x8 AUX support
17	TMS	TMS	Required. See note 1.
18	MDO_A6	MDO_A6	Required for single/dual x8 AUX support
19	TDI	TDI not required for cJTAG	Required for JTAG. See note 1.
20	MDO_A5	MDO_A5	Required for single/dual x8 AUX support
21	/TRSTN	/TRSTN	JTAG reset. Ultra-XD will pull this pin high to enable JTAG. See note 1.
22	MDO_A4	MDO_A4	Required for single/dual x8 AUX support
23	MDO_B7	MDO_B7	Required for dual x8 AUX support
24	MDO_A3	MDO_A3	Required for single/dual x8 AUX support
25	MDO_B6	MDO_B6	Required for dual x8 AUX support
26	MDO_A2	MDO_A2	Required for single/dual x8 AUX support
27	MDO_B5	MDO_B5	Required for dual x8 AUX support
28	MDO_A1	MDO_A1	Required for single/dual x8 AUX support
29	MDO_B4	MDO_B4	Required for dual x8 AUX support
30	MDO_A0	MDO_A0	Required for single/dual x8 AUX support
31	MDO_B3	MDO_B3	Required for dual x8 AUX support
32	MSEO_B1	MSEO_B1	Required for dual x8 AUX support
33	MDO_B2	MDO_B2	Required for dual x8 AUX support
34	MSEO_B0	MSEO_B0	Required for dual x8 AUX support
35	MDO_B1	MDO_B1	Required for dual x8 AUX support
36	MSEO_A1	MSEO_A1	Required for single/dual x8 AUX support
37	MDO_B0	MDO_B0	Required for dual x8 AUX support
38	MSEO_A0	MSEO_A0	Required for single/dual x8 AUX support

#### Table 3. RTT v2 38-way MICTOR pinning for JTAG and cJTAG

**Note 1:** To provide a defined state on the debug-input pins to the ARC core when the Ultra-XD isn't connected, pull-up resistors should be fitted to TDI, TMS, TCK, /TRSTN, /RESET pins on the target board (typically  $10K\Omega$ ).

## 8.2 JTAG Signal Timings

Number	Characteristic	Up to 33 MHz		Up to 50 MHz		Up to 100 MHz		
		Min	Max	Min	Max	Min	Max	Unit
1	TCK Cycle Time ( <b>Tc</b> )	30		20		10		ns
2	TCK Duty Cycle	40	60	45	55	45	55	%
3	Rise and Fall Times (20%–80%)	0	3	0	1.5	0	1.5	ns
4	/TRSTN Setup Time to TCK Falling Edge	(0.30)Tc		(0.15)Tc		(0.15)Tc		ns
5	/TRSTN Assert Time	(0.30)Tc		2Tc		2Tc		ns
6	TMS, TDI Data Setup Time	(0.20)Tc	—	(0.15)Tc	—	(0.15)Tc	—	ns
7	TMS, TDI Data Hold Time	(0.10)Tc		(0.15)Tc		(0.15)Tc		ns
8a	TCK Low to TDO Data Valid (Easy Timing)	(– 0.10)Tc	(0.20)T c	(– 0.10)Tc	(0.20)Tc	(– 0.10)Tc	(0.20)Tc	ns
8b	TCK Low to TDO Data Valid High speed support	_	Tc-4	-	Tc-4	I	Tc-4	ns
8c	TCK Low to TDO hold time (high speed support)	1	-	1	-	1	-	ns

#### Table 4. JTAG Timings

## 8.3 Trace Signal Timings

When running in single data rate (SDR) mode (no clock divide enabled in the ARC core Real-Time Trace (RTT)) then the below timings apply i.e. MDO timing is relative to MCKO falling:

Number	Characteristic	Min	Max	Unit
1	MCKO Cycle Time ( <b>Tco</b> )	5	—	ns
2	MCKO Duty Cycle	40	60	%
3	Output Rise and Fall Times	0	3	ns
4	MCKO low to MDO Data Valid	(–0.10) Tco	(0.20) Tco	ns
5	MCKI Cycle Time ( <b>Tci</b> )	5	—	ns
6	MCKI Duty Cycle	40	60	%
7	Input Rise and Fall Times	0	3	ns
8	MDI Setup Time	(0.20) Tci		ns
9	MDI Hold Time	(0.10) Tci	—	ns

#### Table 5. Trace Timings in single-data rate mode (MDO timing relative to MCKO falling)

When running in double data rate (DDR) mode (i.e. MCKO clock is divided down from the core clock) then the timings for MDO from MCKO are source synchronous (e.g. MDO is driven out on MCKO rising edge).

## 9. Appendix C. CE Notice

The **CE** mark on the back of this Ashling product indicates its compliance with the EMC (Electromagnetic Compatibility) Directive of the European Union (Directive 2004/108/EC). In accordance with this directive, this Ashling product has been tested to the following technical standards:

- EN 61326-1:2006: Electrical equipment for measurement, control and laboratory use.
- Equipment classification: Class B (domestic and light industrial)

To ensure the continued compliance of your Ashling product with the EMC directive (and to ensure that your product can be used without causing interference to, or being affected by other electronic equipment), please note the following:

- This Ashling product is intended for use in the development and test of electronic systems in a development laboratory, by suitably trained staff.
- This Ashling product has been designed to be used with a target system. It should be noted that there may
  be exposed electronic circuitry on the target system, thus when handling the target please note that it is
  possible that electrostatic discharges (ESD) can potentially cause damage to the target or, due to the cabling
  connection, to the Ultra-XD itself. Please exercise all the normal precautions required for electrostatic
  sensitive devices when handling the target system including the use of a workbench equipped to control
  static electricity and an anti-static wrist strap, properly connected to the workbench.
- This product is designed for use with a Personal Computer or Laptop whose electromagnetic emission and susceptibility performance comply with the EMC Directive.
- This product is designed for use with an external 12V DC supply whose electromagnetic emission and susceptibility performance comply with the EMC Directive.

Doc: APB219-Ultra-XD(with MW).docx