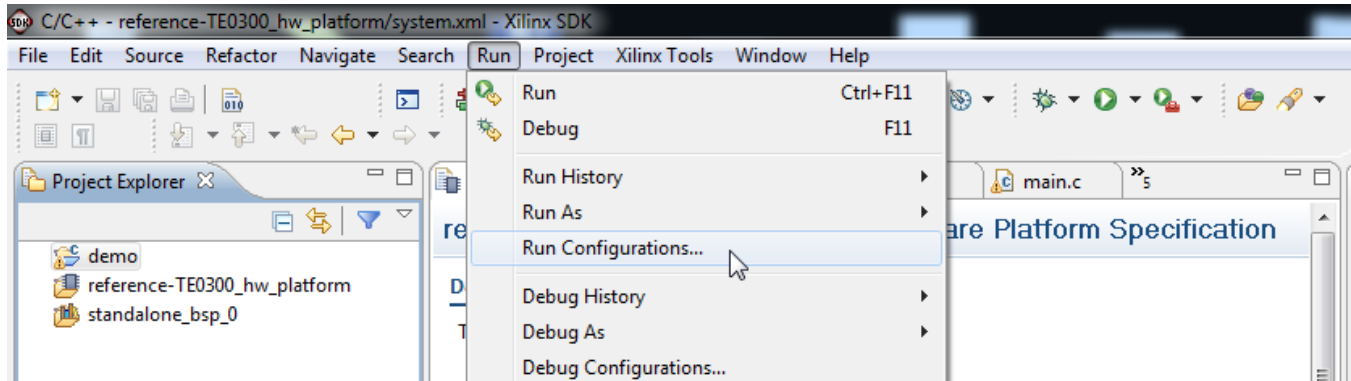


Run the demo project

Configure debug and run operation

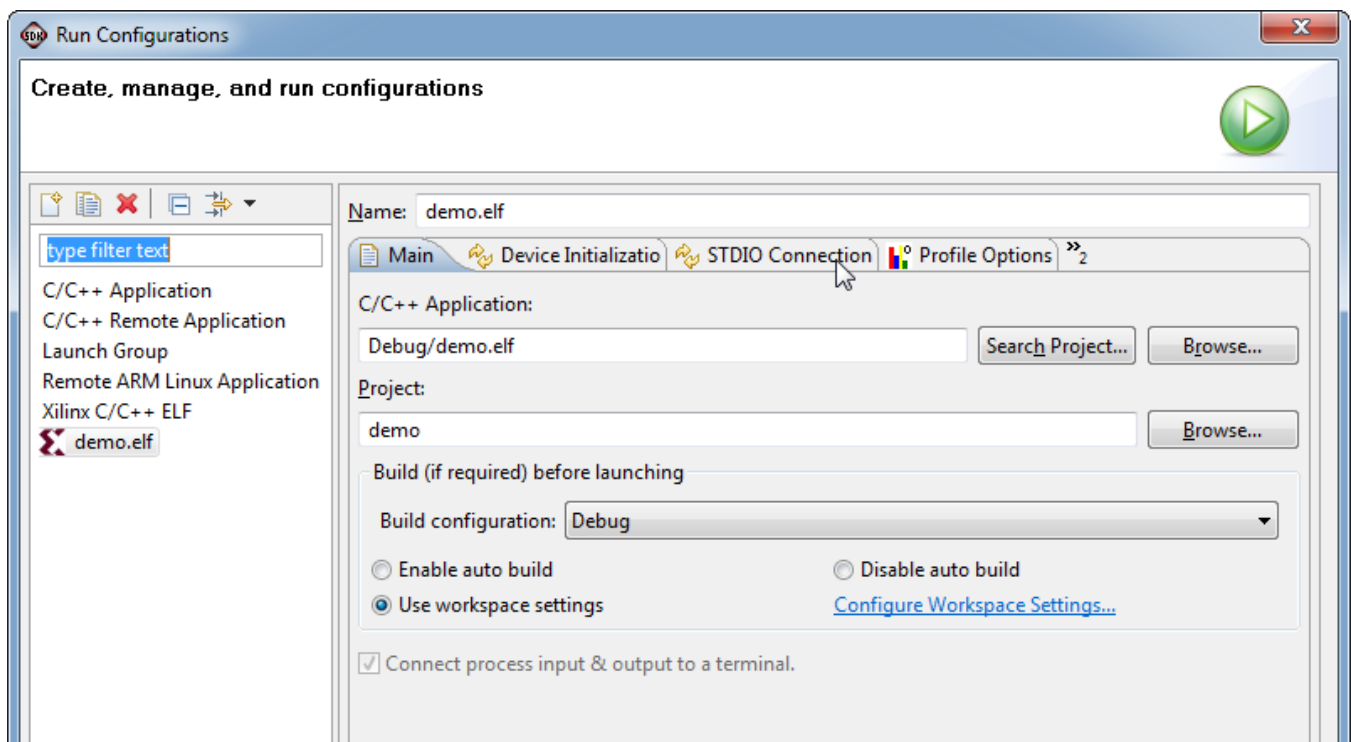
Before you could run the "demo.elf" file on MicroBlaze you should configure the "stdio output".

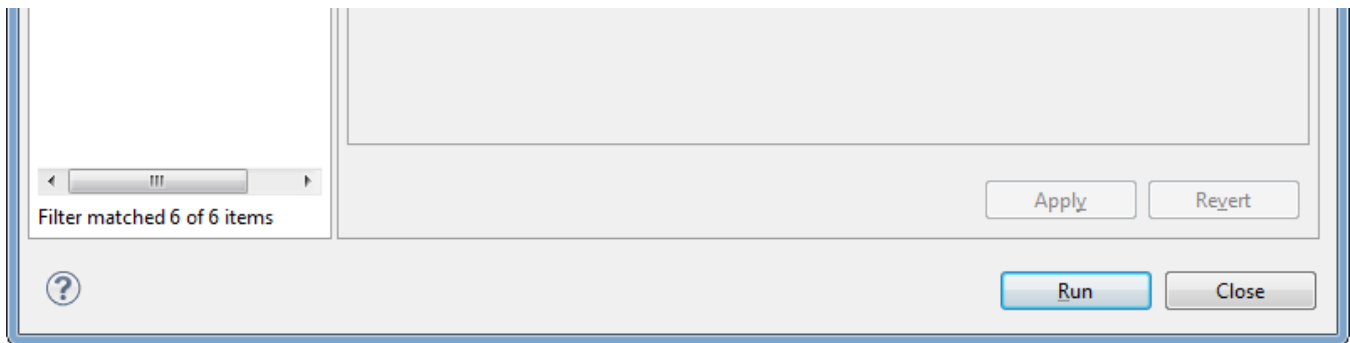
You should click "Run">"Run Configurations..."



Open "Run Configurations"

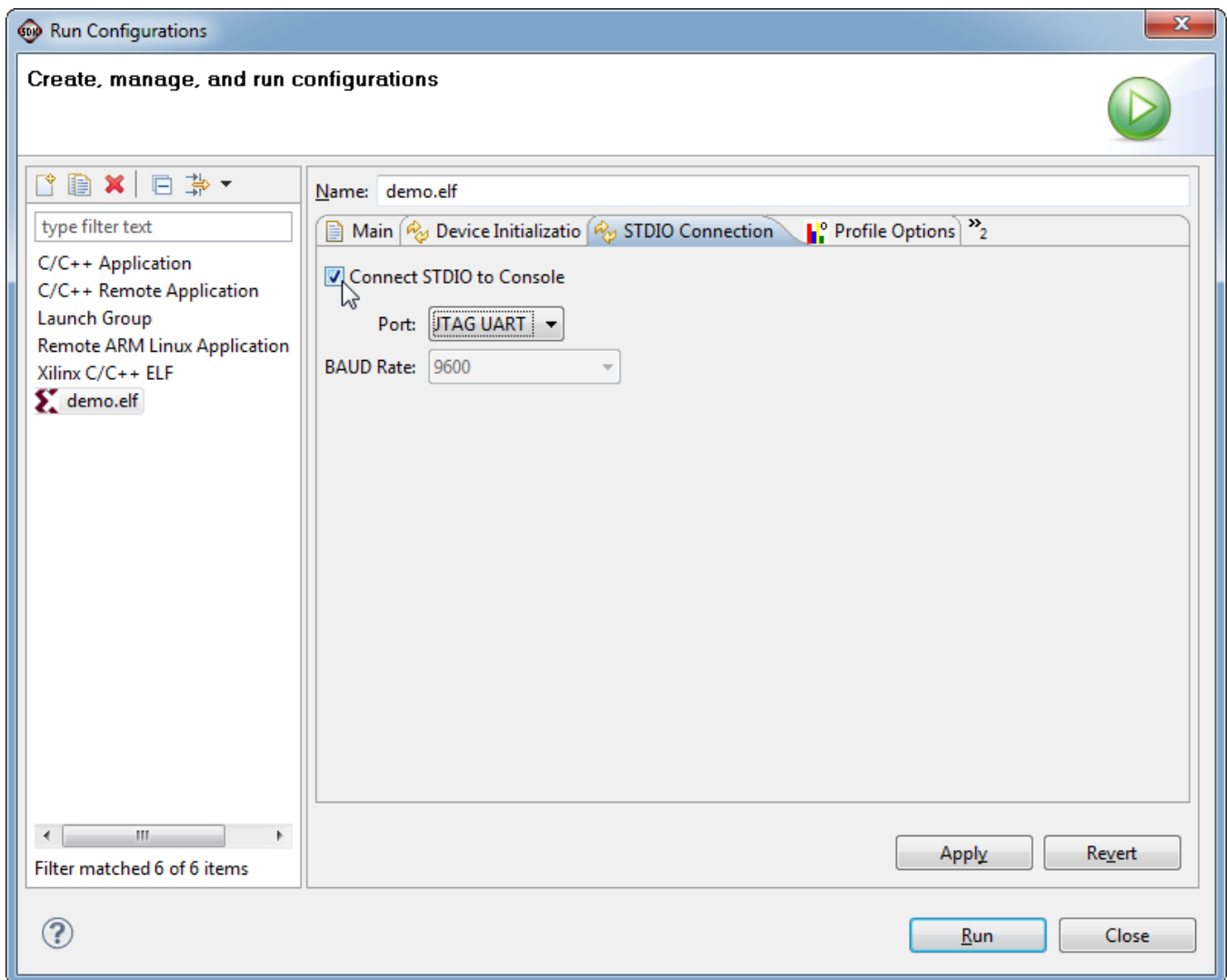
A pop-up "Run Configurations" will appear. Click "demo.elf" (if it is not already selected) and then click "SDIO Connection" tab.





"Run Configurations" opened

Check (✓) the box "Connect STDIO to Console" and select as port "JTAG UART". Then click "Apply" button.



Connect STDIO to Console

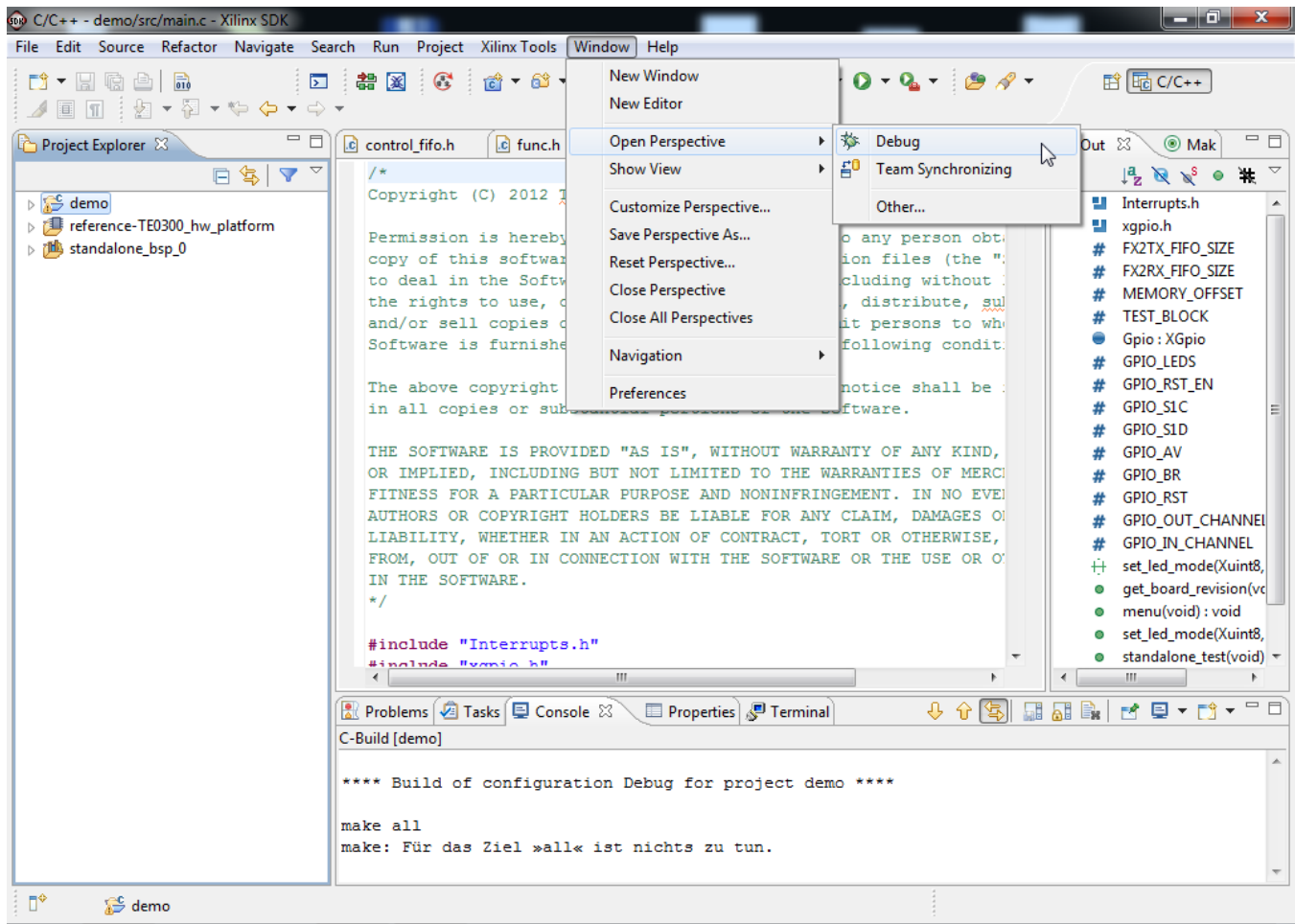
Then click "Close" button. The pop-up will close.



The Debug operation normally share these settings, so you doesn't need to repeat this procedure for debug operation but is better to check the settings (it is better to be on the safe side).

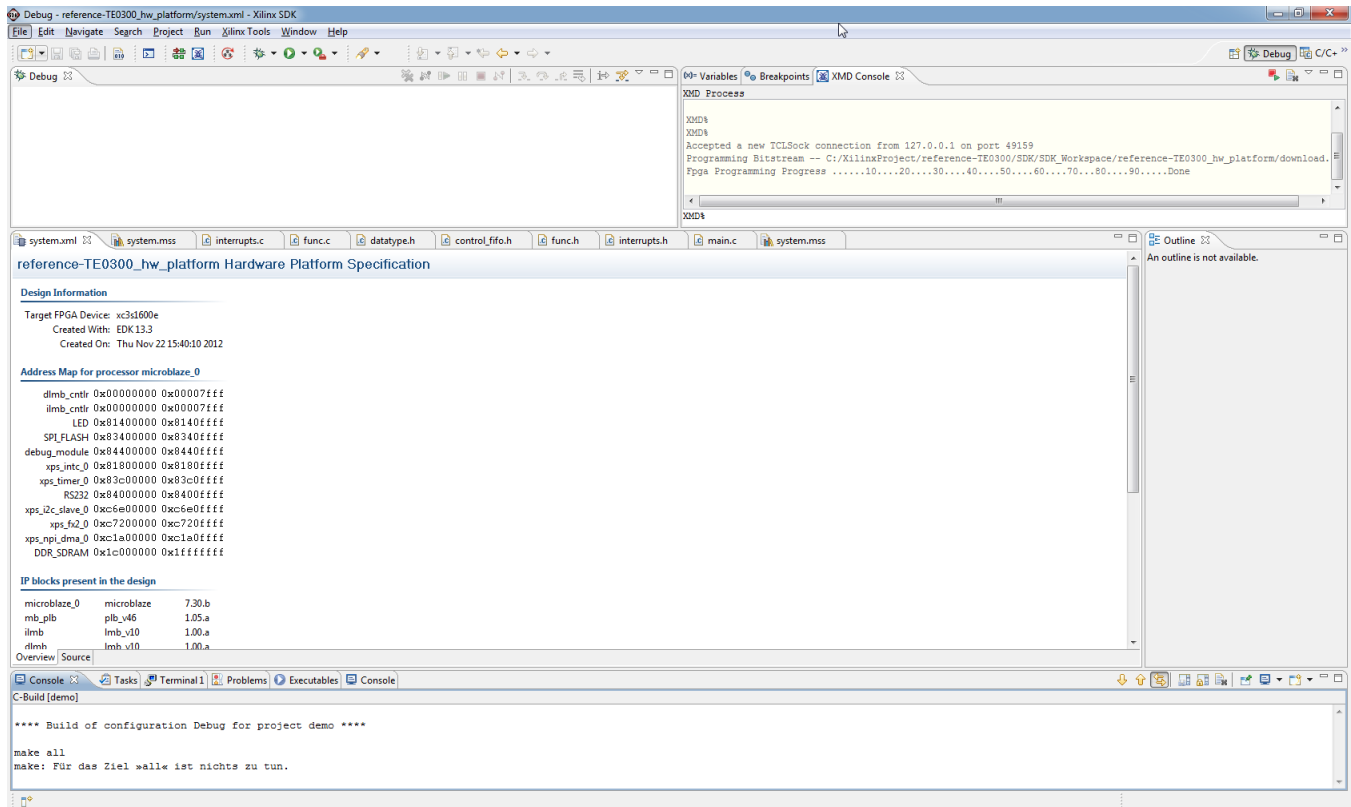
Change perspective from "C/C++" to "Debug"

To change perspective you should click "Window">"Open Perspective">"Debug".



Open Debug Perspective

The new perspective is the following.



Debug perspective opened

Use the "demo" project with the XMD UART

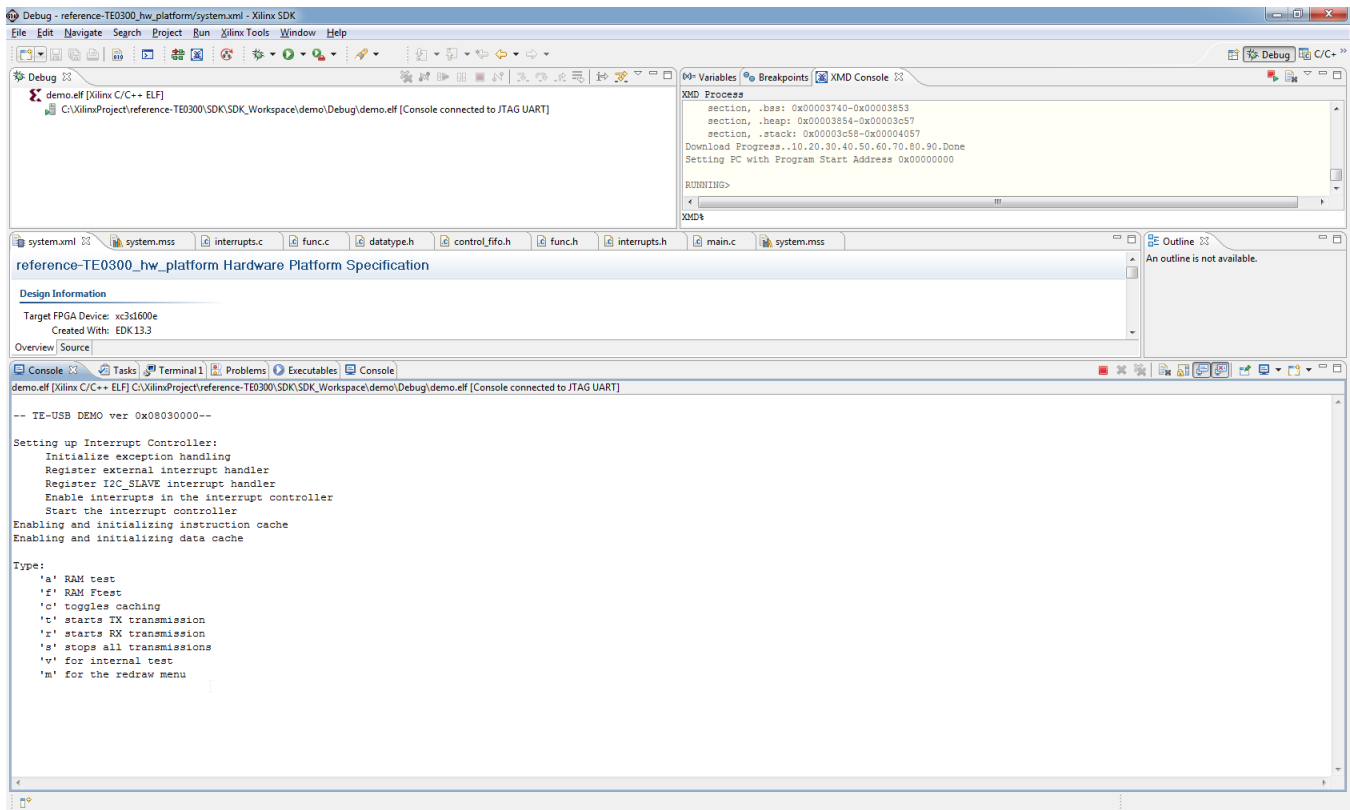


Demo program (running on MicroBlaze) will work even in case the UART port is left unconnected: it is not necessary to use a USB/Uart converter or Uart port on a PC, if you are using XMD UART HDL block.

With this application, you can test the PC USB JTAG FPGA communication using a simulated UART (XMD_UART) on JTAG/USB connection.

GUI procedure

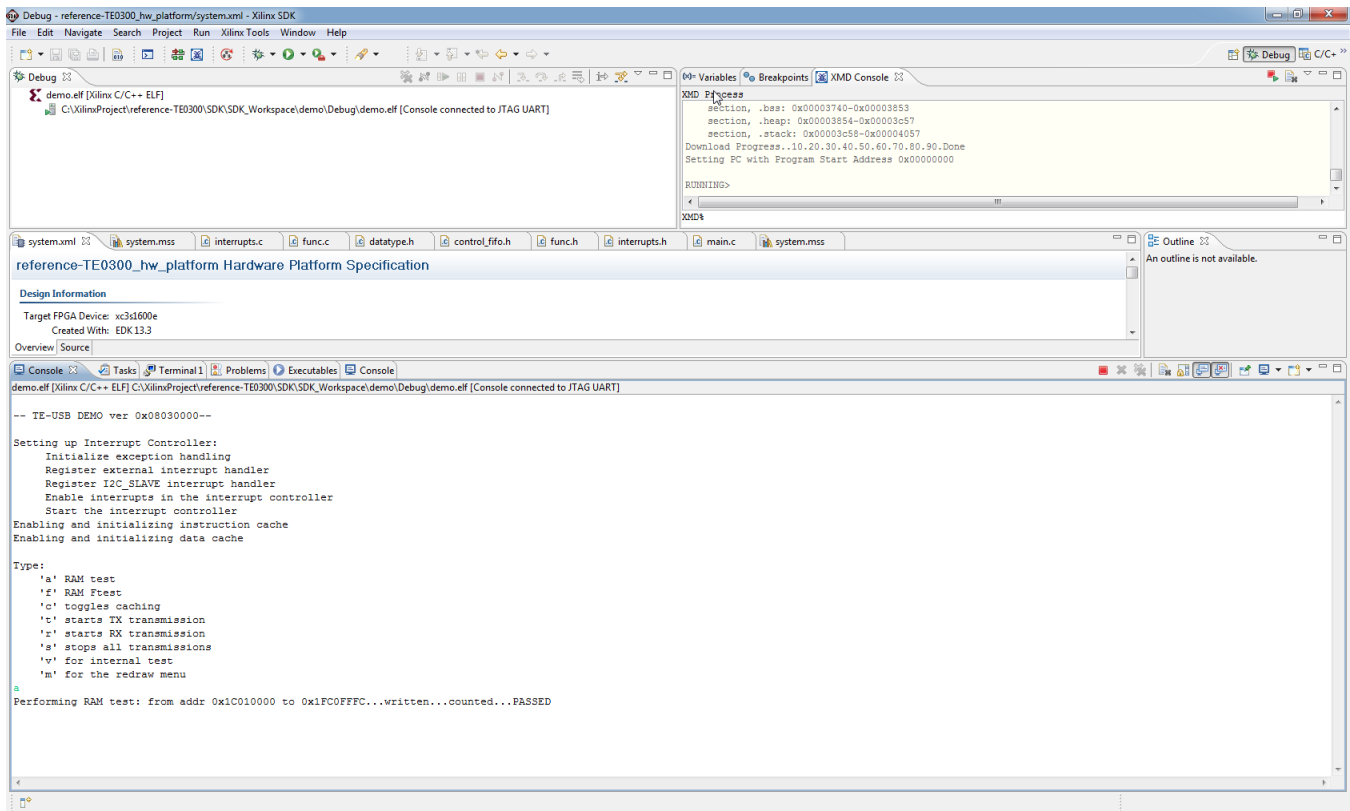
In this simple case you can simply click "Run" > "Run"



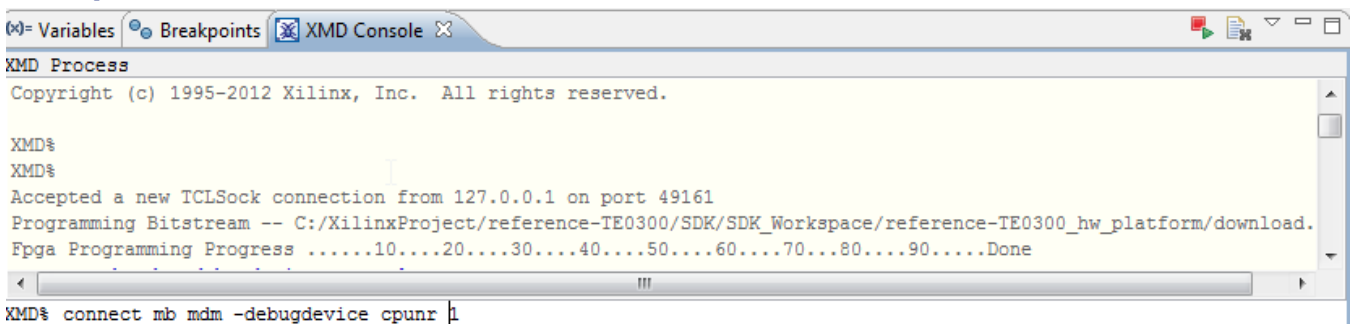
demo.elf downloaded and connected to Jtag-Uart

In the console the menu of demo.elf should appear. If the menu doesn't appear you have probably set RS232 instead of debug (mdm) and/or set incorrectly "Stdio output".

If you write the character "a" the RAM test should start.

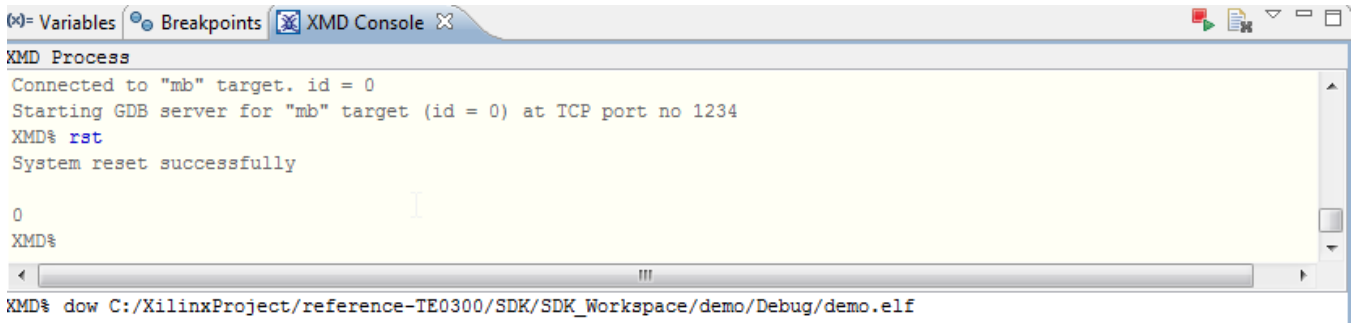


XMD procedure



In the XMD console you should write "connect mb mdm -debugdevice cpunr 1".

In the XMD console you should write "rst" and then click "return" on the keyboard.



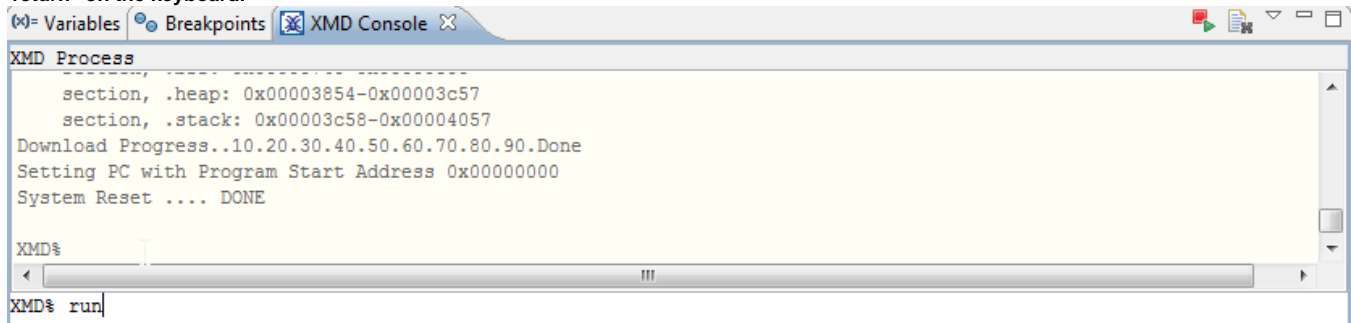
The screenshot shows the XMD Console window with the following text:

```
XMD Process
Connected to "mb" target. id = 0
Starting GDB server for "mb" target (id = 0) at TCP port no 1234
XMD% rst
System reset successfully

0
XMD%

XMD% dow C:/XilinxProject/reference-TE0300/SDK/SDK_Workspace/demo/Debug/demo.elf
```

In the XMD console you should write "dow C:/XilinxProject/reference-TE0300/SDK/SDK_Workspace/demo/Debug/demo.elf" and then click "return" on the keyboard.



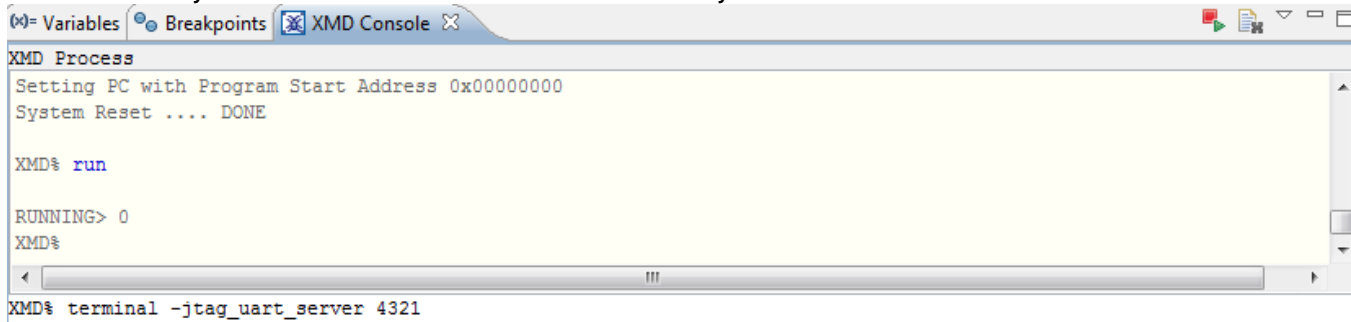
The screenshot shows the XMD Console window with the following text:

```
XMD Process
-----
section, .heap: 0x00003854-0x00003c57
section, .stack: 0x00003c58-0x00004057
Download Progress..10.20.30.40.50.60.70.80.90.Done
Setting PC with Program Start Address 0x00000000
System Reset .... DONE

XMD%

XMD% run
```

In the XMD console you should write "run" and then click "return" on the keyboard.



The screenshot shows the XMD Console window with the following text:

```
XMD Process
Setting PC with Program Start Address 0x00000000
System Reset .... DONE

XMD% run

RUNNING> 0
XMD%

XMD% terminal -jtag_uart_server 4321
```

In the XMD console you should write "terminal -jtag_uart_server 4321" and then click "return" on the keyboard.

After this you should open some terminal emulators (because you want to input/output some characters with the XMD UART), such as

- Microsoft / Hilgraeve HyperTerminal (usually included in Windows before Vista START MENU > All programs > Accessories > Communications > Hyper Terminal).
- [ClearTerminal](#) (very easy)



Connect using the following settings:

- No Host address
- Port Number: 4321
- TCP/IP connection type (client)



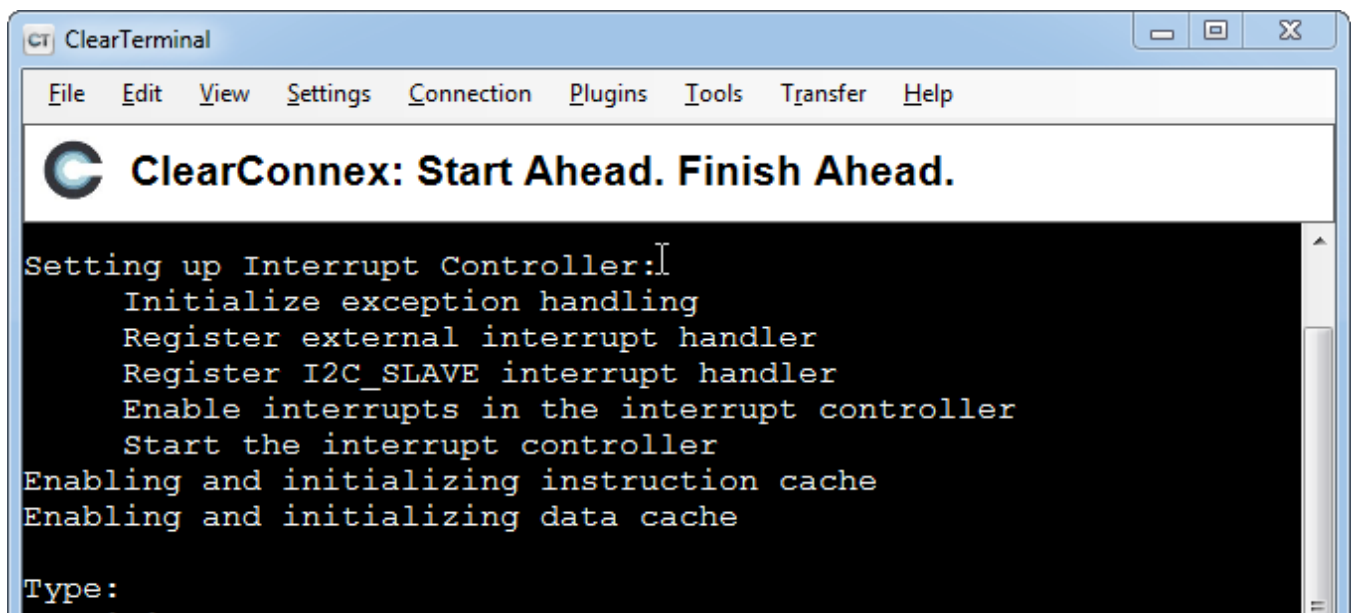
The UART settings (not required by ClearTerminal) are:

- bits per seconds: 115,200
- data bits: 8
- parity: none
- stop bits: 1
- flow control: none (otherwise you will not be able to enter commands)

In the XMD console you should write "stop" and then click "return" on the keyboard.

In the XMD console you should write "run" and then click "return" on the keyboard.

After this two further step the menu of "demo.elf" should appear in the terminal emulator.



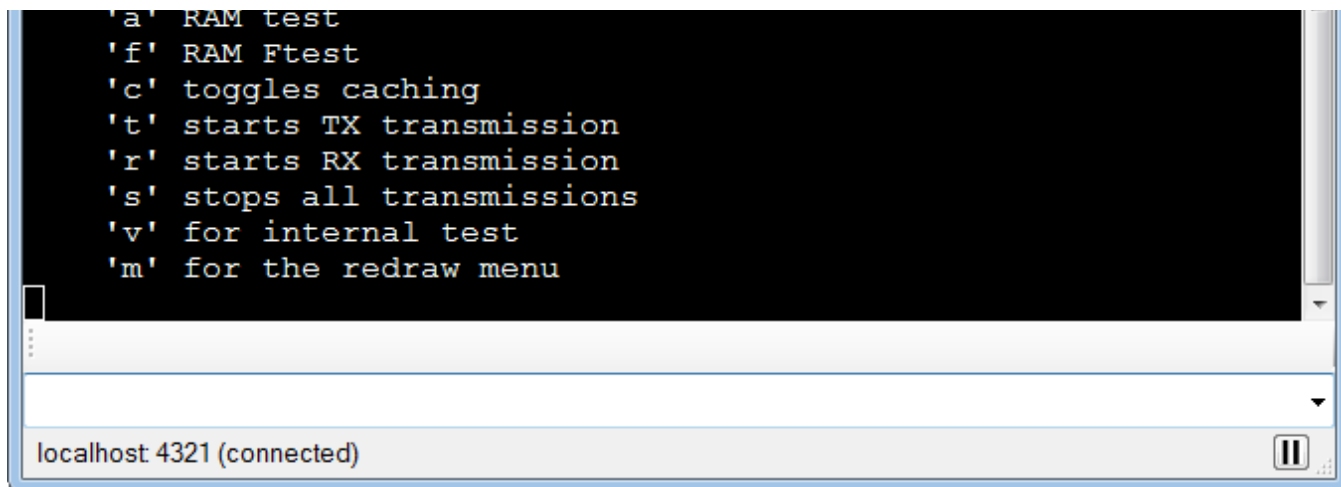
The screenshot shows a window titled "ClearTerminal" with a menu bar containing "File", "Edit", "View", "Settings", "Connection", "Plugins", "Tools", "Transfer", and "Help". Below the menu bar is a header with the ClearConnex logo and the text "ClearConnex: Start Ahead. Finish Ahead.". The main area is a black terminal window with white text listing initialization steps. A vertical scrollbar is on the right side of the terminal area.

```
CT ClearTerminal
File Edit View Settings Connection Plugins Tools Transfer Help

ClearConnex: Start Ahead. Finish Ahead.

Setting up Interrupt Controller:
  Initialize exception handling
  Register external interrupt handler
  Register I2C_SLAVE interrupt handler
  Enable interrupts in the interrupt controller
  Start the interrupt controller
Enabling and initializing instruction cache
Enabling and initializing data cache

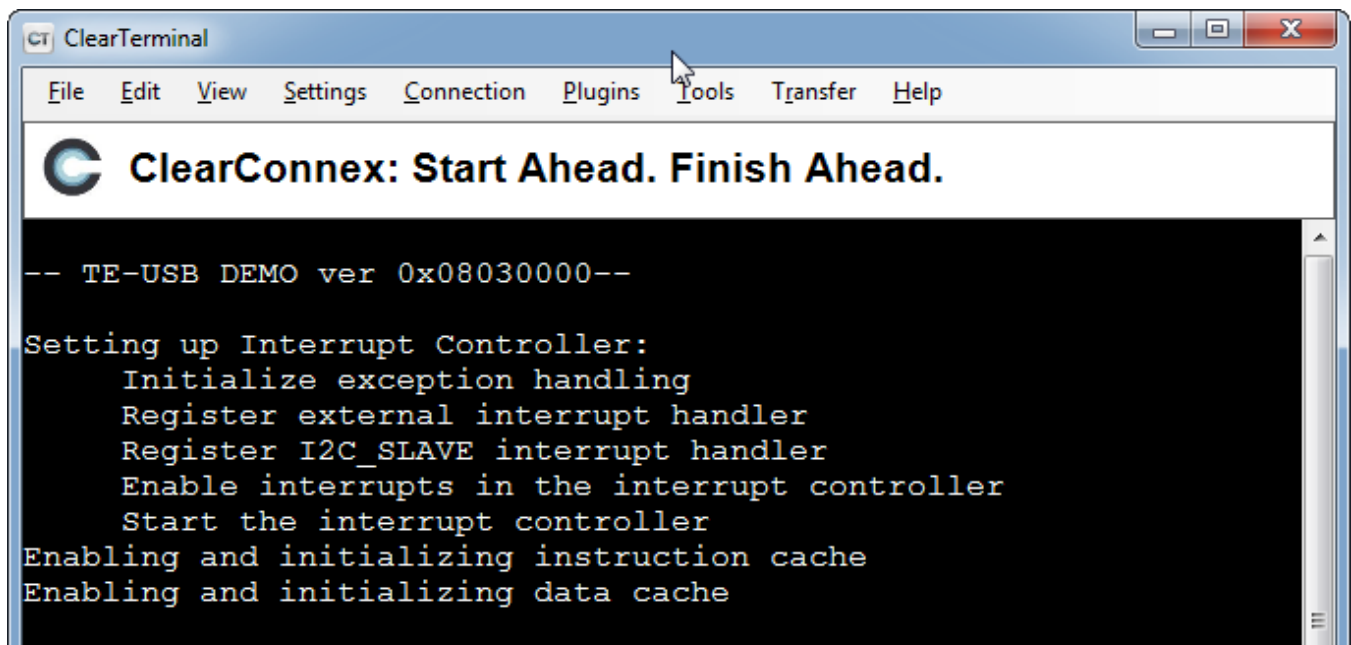
Type:
```



After this two further step the menu of "demo.elf" should appear in the terminal emulator.

In the console the menu of demo.elf should appear. If the menu doesn't appear you have probably set RS232 instead of debug (mdm) and/or set incorrectly "Stdio output".

If you write the character "a" the RAM test should start.



```
Type:
'a' RAM test
'f' RAM Ftest
'c' toggles caching
't' starts TX transmission
'r' starts RX transmission
's' stops all transmissions
'v' for internal test
'm' for the redraw menu
Performing RAM test: from addr 0x1C010000 to 0x1FC0FFFC...write
n...counted...PASSED

localhost 4321 (connected)
```


If you write the character "a" the RAM test should start.

Use the *demo* project without the XMD UART

To use the *demo* project without the XMD UART, you need to use "RS232" instead of "debug_module" as standard in/out port. Otherwise the application running on the Microblaze processor freezes if you disconnect the XMD. To accomplish that you need to set up the Microblaze "Software Platform Settings".

- In the dialog window select "OS and libraries" in the left window and pick "RS232" as a stdout and stdin interface.
- Then rebuild the software and download again the project to the FPGA.

The UART is then redirected to external pins, which are defined in the *data/system.ucf* file.

 The UART settings are:

- bits per seconds: 115,200
- data bits: 8
- parity: none
- stop bits: 1
- flow control: none (otherwise you will not be able to enter commands)

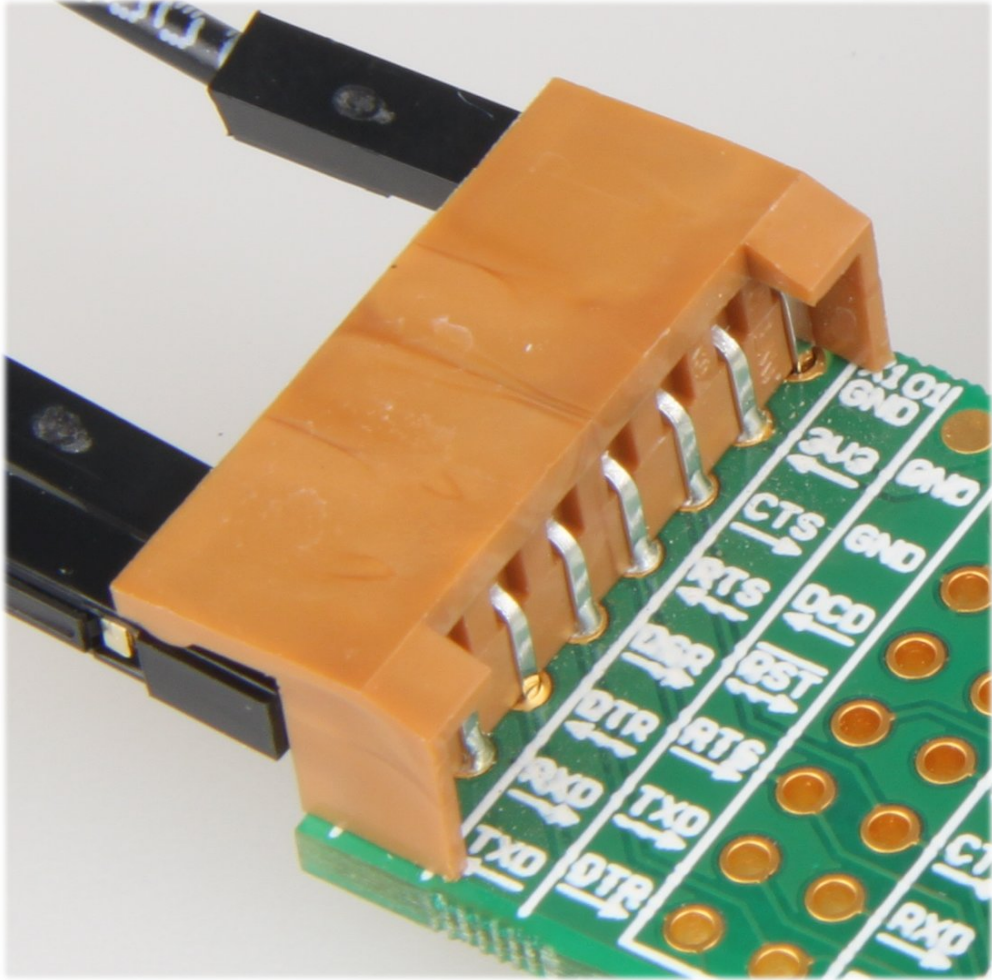
The following snippet shows the case of the TE0300 series modules:

Module RS232 constraints*
Net fpga_0_RS232_RX_pin LOC=B13;
Net fpga_0_RS232_TX_pin LOC=B14;
Please refer to the table below for other module series relevant to this application note.

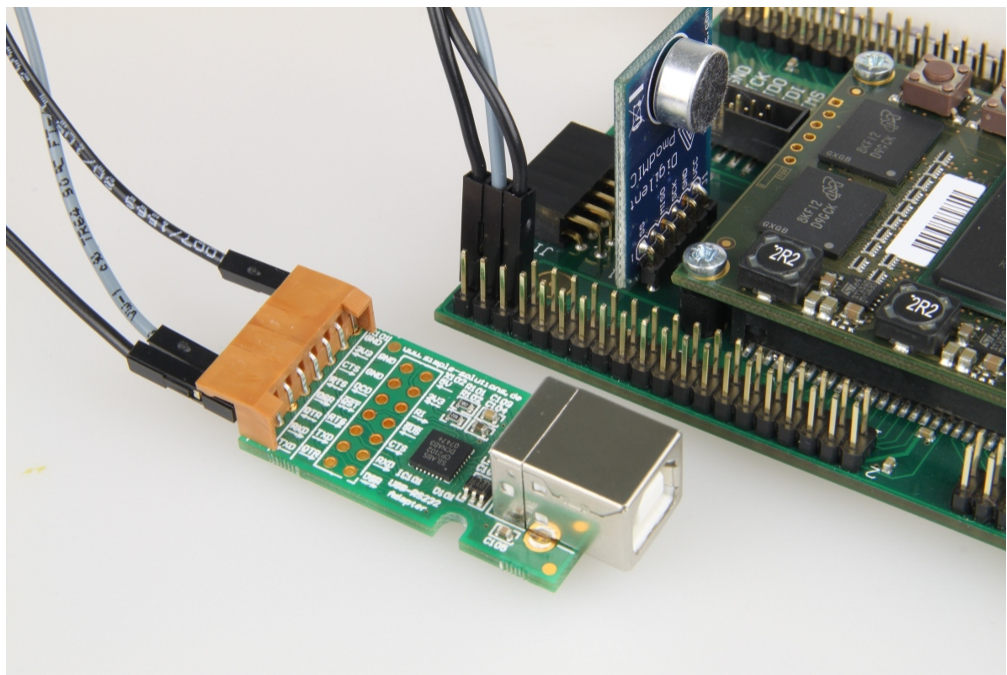
TE series	RS232_RX FPGA ball	RS232_RX module pin	RS232_TX FPGA ball	RS232_TX module pin
TE0300	R6	J5-29	P6	J5-31
TE0320	V17	J5-IO18	W17	J5-IO19
TE0630	Y7	J5-29	AB7	J5-31

TE0303	It doesn't apply	J1-33	It doesn't apply	J1-34
TE0304	It doesn't apply	J1-3	It doesn't apply	J1-2
TE0323	It doesn't apply	J4-35	It doesn't apply	J4-37
host (PC)	TX	TX	RX	RX

Location of UART pins examples.



Sample UART to USB virtual COM port converter.



Sample UART to USB virtual COM port converter: signal detail for TE0320 and TE0323.