

# Using RealView Debugger BCD files

## Introduction

This documentation describes how to install and use BCD files in the ARM RealView Debugger. Specifically, this documentation covers the BCD files for i.MX6-series included in the register header package.

It is assumed that version 4.1 of the RealView Debugger is being used.

## Install BCD files.

Extract the zip file containing the BCD files, or otherwise gain access to the files.

The BCD files are organized into subdirectories by the i.MX6-series variant. Copy all of the .bcd files for the chip you are using into the RealView Home Directory.

On Windows, this directory is located here:

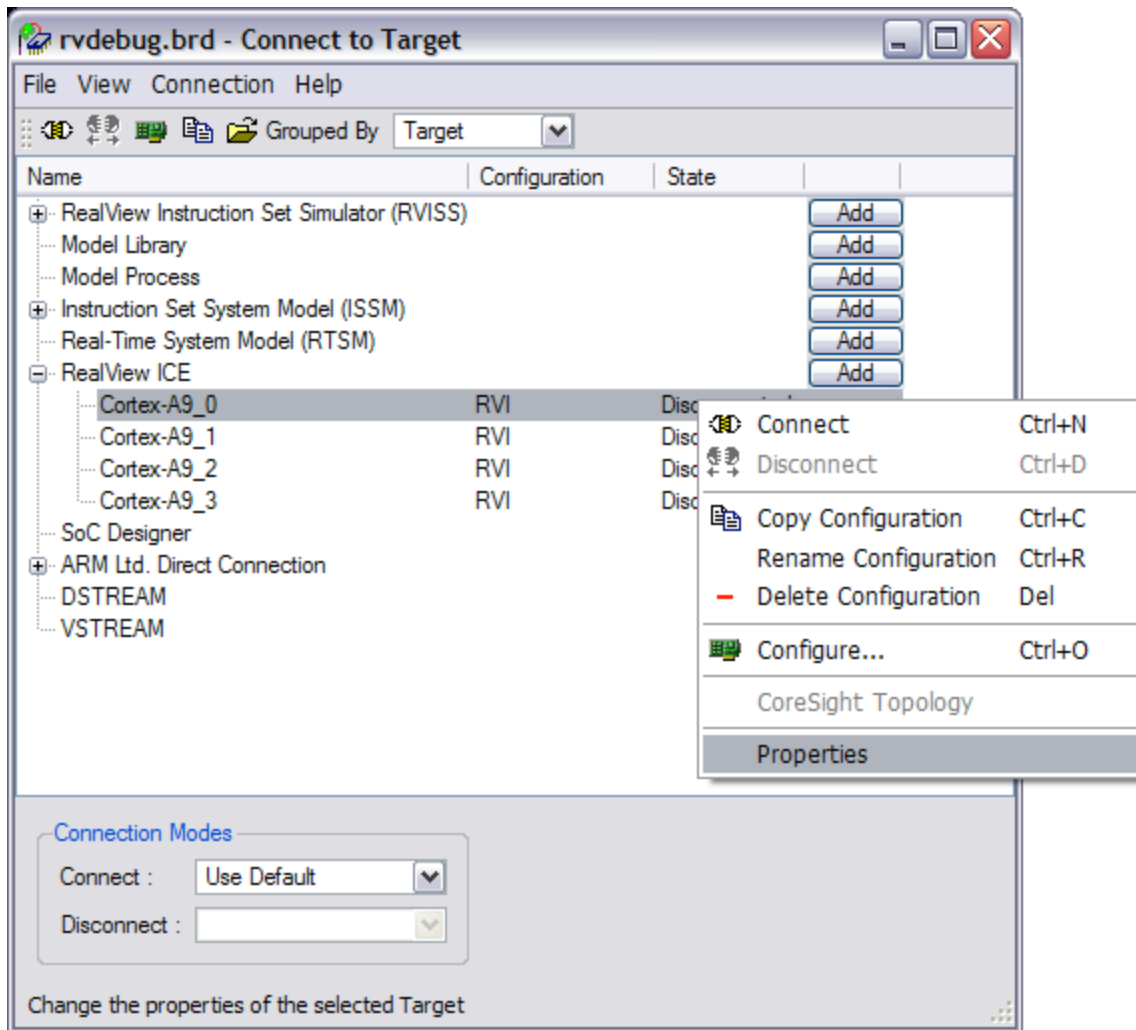
`C:\Documents and Settings\<username>\Application Data\ARM\rvdebug\4.1.2`

You can copy the BCD files for all three i.MX6-series chips into this directory, if you prefer. Any time you add or remove files from this directory, you should restart the RealView Debugger.

## Add register sets to your connection

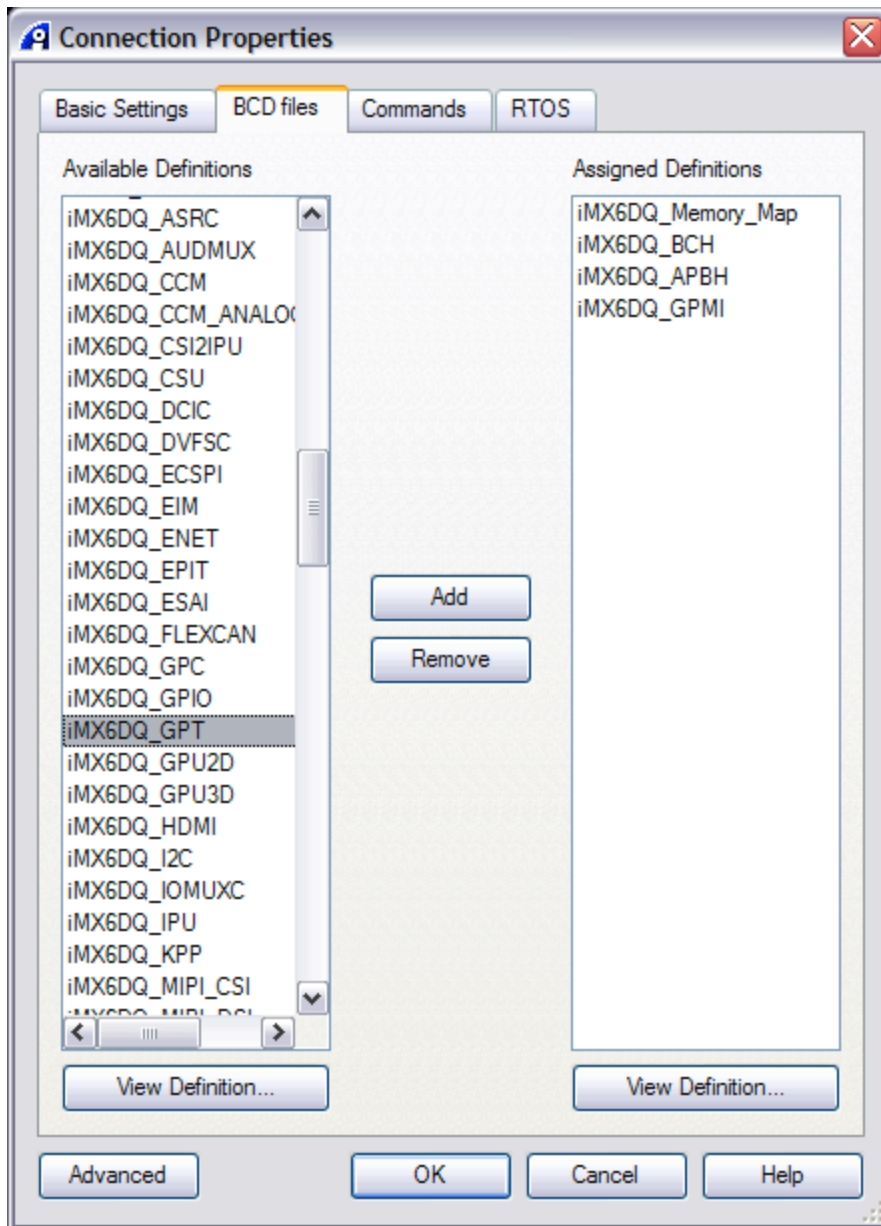
To actually use the BCD files in order to view registers, you need to add them to your connection.

First, open the **Connection Properties** window for the i.MX6 connection you are using. You can do this by right-clicking on the connection in the **Connect to Target** window, as shown below.



Select the **Properties** item from the context menu.

When the **Connection Properties** window appears, select the **BCD files** tab. The window should look like this:



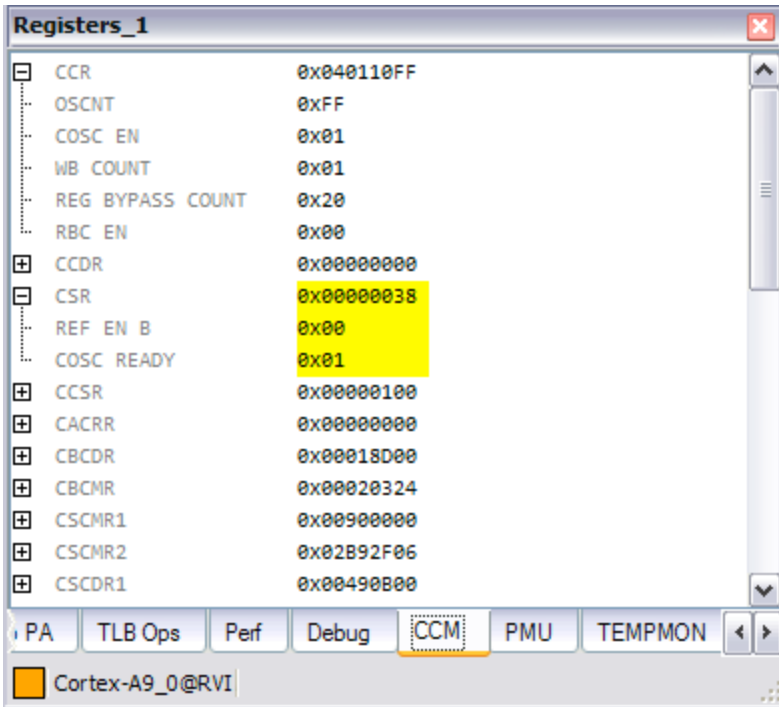
Scroll the list on the left side of the pane until the iMX6 items are available. Double-click the items on the left to add to your connection. Double-clicking an item on the right-hand list will remove it from your connection.

*Note: you always need to have the **iMX6DQ\_Memory\_Map** (or similar name for other chips) item added to you connection. Otherwise, memory configuration scripts may not work.*

The registers are broken into separate definitions because there are so many peripherals available on the i.MX6-series chips. Having all definitions in one file could make the registers window very confusing. This arrangement gives you control over exactly which peripherals' registers are available for access.

Keep in mind that RVD only supports up to 32 BCD files per connection.

Close the **Connection Properties** window by clicking **OK**. Now you can connect to the target and view the registers you have selected. The registers window will look something like this:



To change available registers, you must disconnect, edit properties, and reconnect.

## Known issues

When connecting, you will see some warnings about overlapping memory regions. This is expected. As of right now, there is no known way to avoid this warning.

RVD has a maximum of 8192 lines per register window, and a few of the register definitions are larger than this.

- IPU is larger than 8192 lines.
- PCIe causes a crash when loading.
- RVD complains about "Register not found" for a few of the definitions.
- Duplicate definitions in some register sets.
- A few registers cause "precise aborts".


















Many components require that their clock is enabled in CCM, otherwise the debugger will lose track of the target state when you select the component's register tab. This can be accomplished easily by always running the debugger init script (assuming it enables all clocks) before accessing registers.

## Component status

Only MX6DQ has been tested so far.

Component	Status	Notes
APBH	✓	
ASRC	✓	
AUDMUX	✓	
BCH	✓	
CCM	✓	
CCM_ANALOG	✓	
CSI2IPU	✓	

CSU	✓	
DCIC	⚠	registers not found
DVFS	⚠	precise aborts
ECSPI	✓	
EIM	✓	
ENET	✓	
EPIT	✓	
ESAI	⚠	precise aborts
FLEXCAN	⚠	registers not found (FLEXCAN1 only)
GPC	✓	
GPIO	✓	
GPMI	✓	
GPT	✓	
GPU2D	✓	
GPU3D	✓	
HDMI	⚠	duplicate bits
I2C	✓	
IOMUXC	✓	
IPU	✗	reg window is too large
KPP	✓	
MIPI_CSI	⚠	precise aborts
MIPI_DSI	✓	
MIPI_HSI	✓	
MLB150	✓	duplicate bit
MMDC	⚠	lots of register not found warnings
OCOTP	✓	
PCIE	✗	causes debugger to lose track of target
PMU	✓	
PWM	✓	
ROMC	✓	
SATA	✓	

SDMA_ARM		registers are shown OK, but seems to prevent any BCD files afterwards from working
SDMA_BP		
SDMA_CORE		
SJC		
SPDIF		
SRC		
SSI		duplicate bits
TEMPMON		
UART		precise abort
USB_ANALOG		
USBC		duplicate bits
USBNC		
USB_PHY		registers not found (USB_PHY1 only)
USDHC		duplicate bits, many registers not found
VDOA		
VPU		
WDOG		
XTALOSC24M	