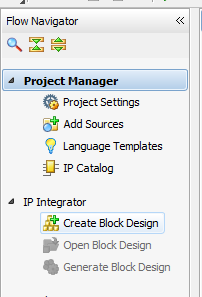
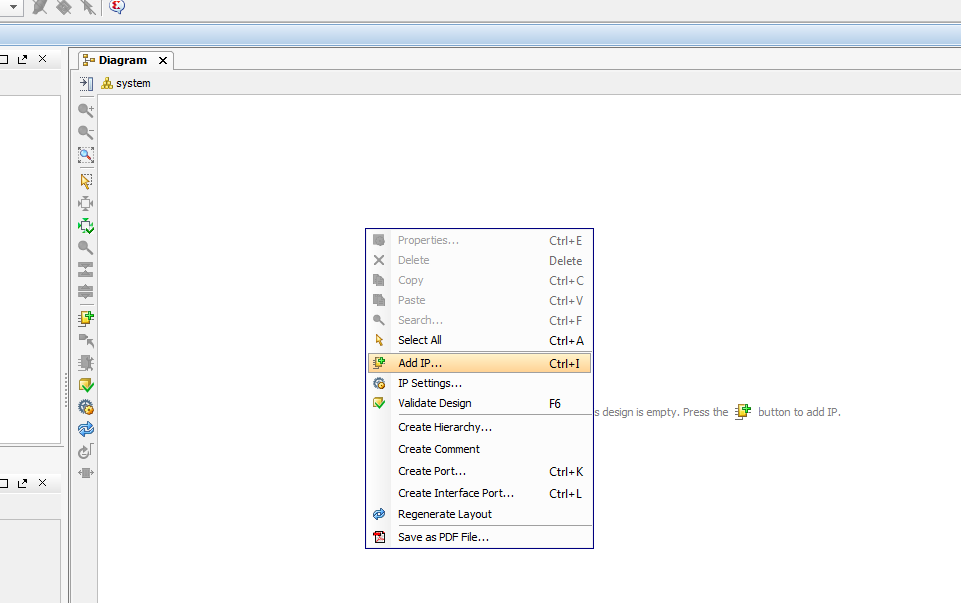
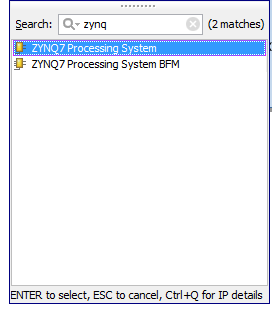
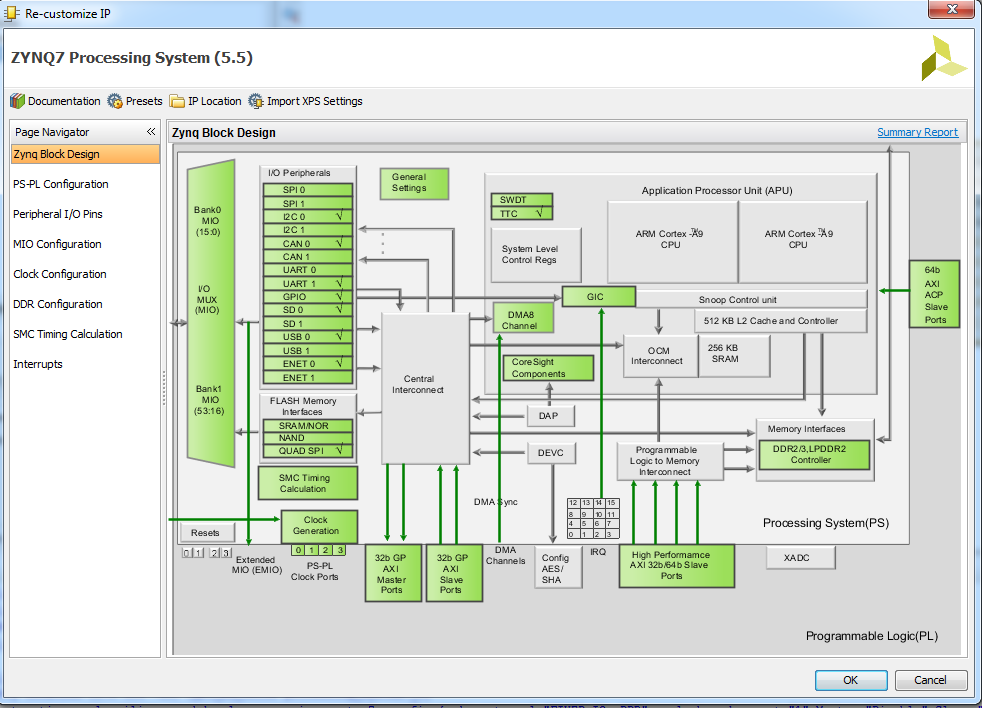
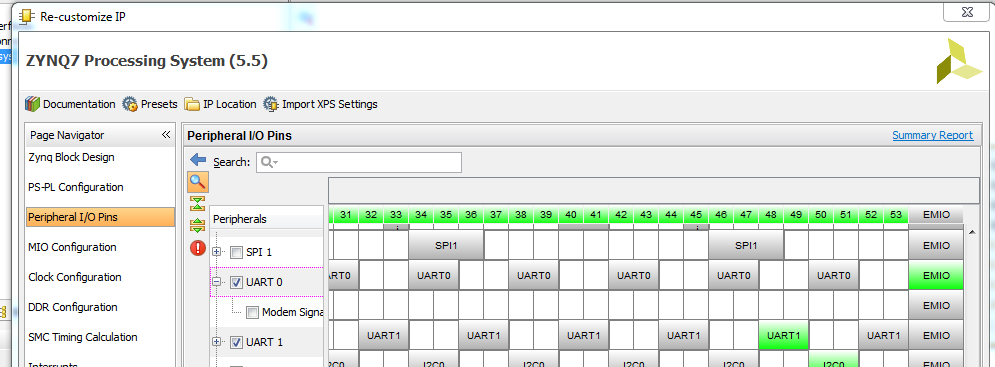
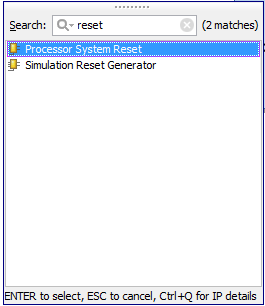
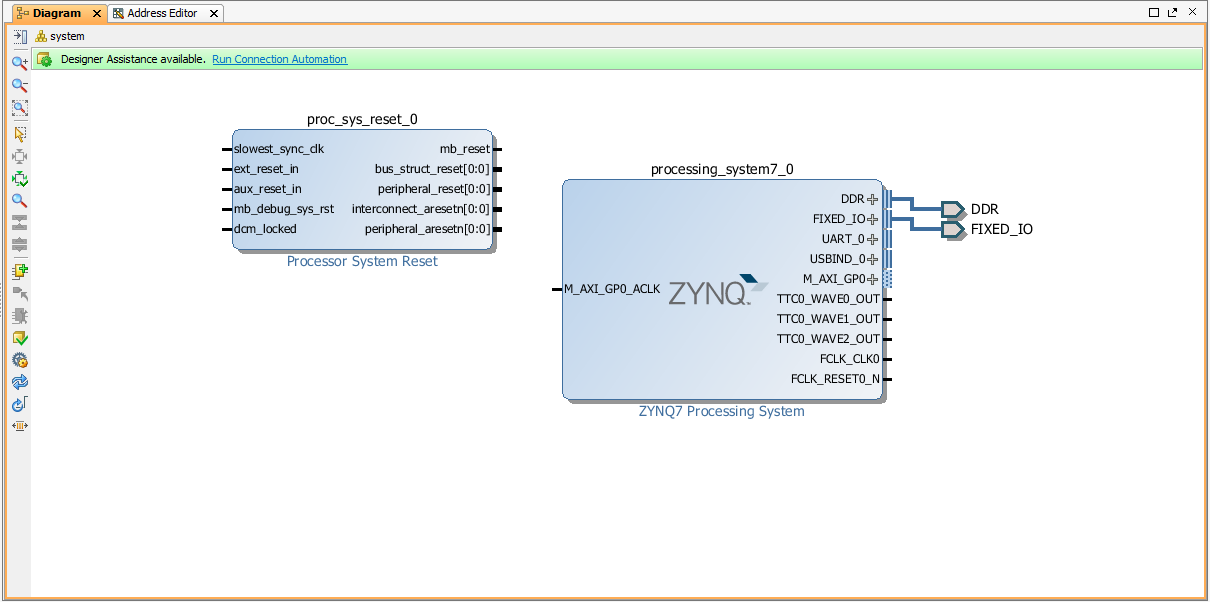
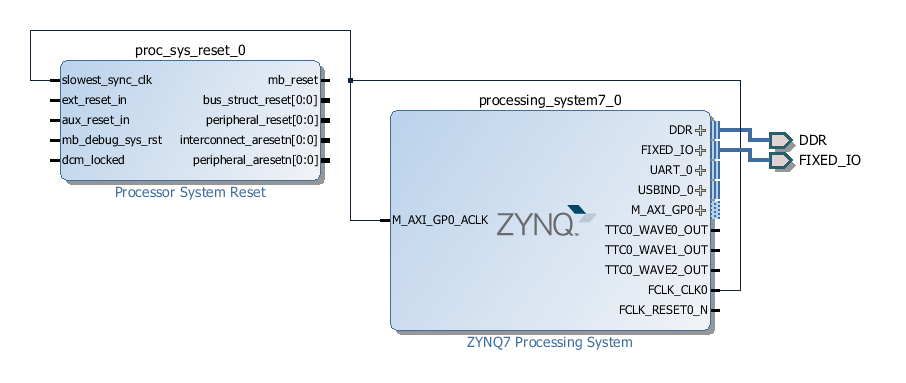
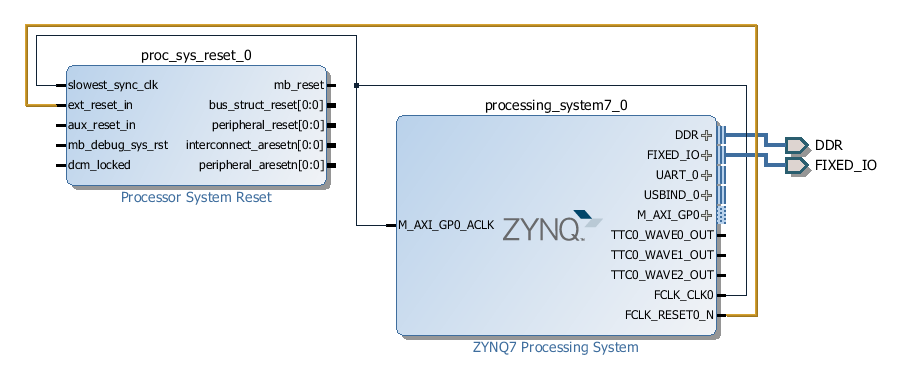
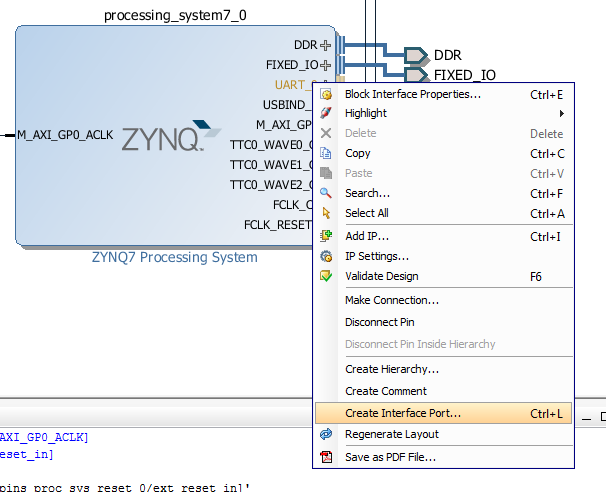
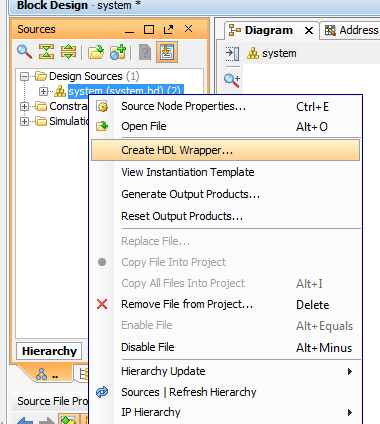
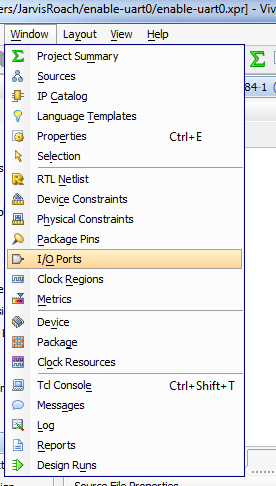
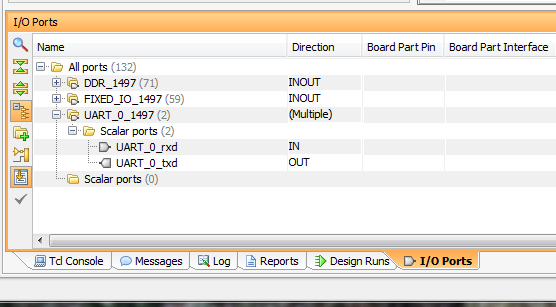
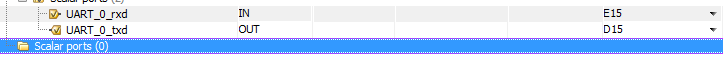
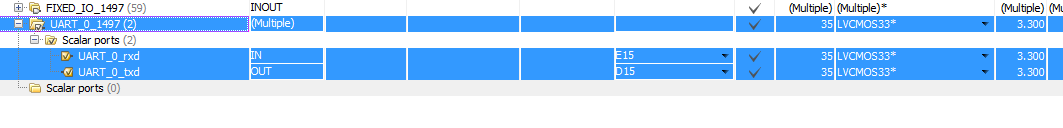
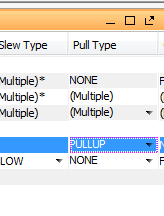
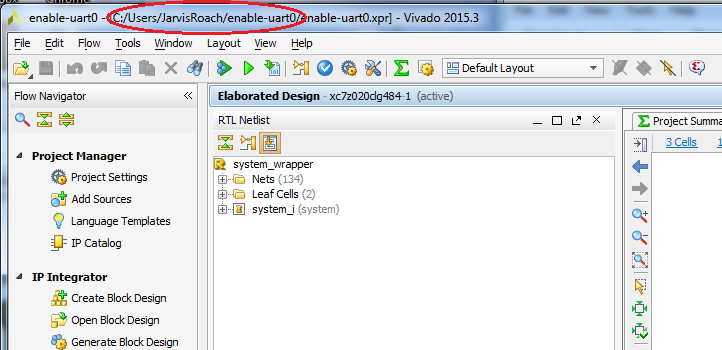
## Create new FPGA bitstream

By default, the ZC702 dev board does make the UART0 signals externally available. However, it is possible to route those signals to the PMOD1\_0 and PMOD1\_1 pins through EMIO with an FPGA design.

The following steps were performed using Vivado 2015.3 running on Windows on a 8G DDR3 1333MHz  
Intel(R) Core(TM) i7 CPU Q 870 @ 2.93GHz.

1. Create new project, enable-uart0.
   1. “Create New Project” from Quick Start on the start-up window.
   2. “Create a New Vivado Project” popup.
      1. Click “Next”.
   3. “Project Name” popup
      1. Change “Project name:” to “enable-uart0”.
      2. Click “Next”.
   4. “Project Type” popup.
      1. Make sure “RTL Project” is selected.
      2. Select “Do not specify sources at this time”.
      3. Click “Next”.
   5. “Default Part” popup.
      1. Click on “Boards” from the “Select:” row towards the top of the window.
      2. Select “ZYNQ-7 ZC702 Evaluation Board” from the “Display Name” column.
      3. Click “Next”.
   6. “New Project Summary” popup.
      1. Double check that it is an RTL project with the name ‘enable-uart0’ that will be created.
      2. Confirm the default board is the ZC702.
      3. Click “Finish”.
2. In “Flow Navigator”, click “IP Integrator”->”Create Block Design .
3. “Create Block Design” popup.
   1. Change “Design name:” to “system”.
   2. Click “OK”.
4. Add Zynq 7000 block.
   1. In “Diagram” window, right click and select “Add IP…”.
   2. In the resulting popup, type “Zynq” in the search window, and double-click “ZYNQ7 Processing System”.  
      
5. Apply board configuration defaults.
   1. In the “Diagram” Window, click on “Run Block Automation”.
   2. In the “Run Block Automation” pop-up.
      1. Select “Apply Board Preset”.
      2. Click “OK”.
6. Map UART0 to EMIO.
   1. Double click the ZYNQ block component in the “Diagram” window.
   2. Wait until the “Re-customize IP” window pops up. 
   3. Select “Peripheral I/O Pins” in the “Page Navigator” on the left hand side.
   4. Scroll down to “UART0” and select it.
   5. Confirm that “EMIO” is highlighted for UART0.
   6. Click “OK” at the bottom of the “Re-customize IP” pop-up.
7. Add reset controller.
   1. In “Diagram” window, right click and select “Add IP…”.
   2. Search for “reset” and double click on “Processor System Reset”.  
      The “Diagram” window should look something like this now: 
8. Connect FCLK\_CLK0, M\_AXI\_GPO\_ACLK from ZYNQ block to “slowest\_sync\_clk” on the reset block.



1. Connect “FCLK\_RESET0\_N from ZYNQ to “ext\_reset\_n” on reset block.
2. Create “UART\_0” port.
   1. Right click on “UART\_0” in the Zynq block.
   2. Select “Create Interface Port…”. 
   3. Confirm “VLNV:” is “xilinx.com:interface:uart\_rtl:1.0”
   4. Click “OK” on “Create Interface Port” pop-up.
3. Create wrapper for block diagram.
   1. In “Sources->Hiearchy”, right click on “system” and select “Create HDL Wrapper”
   2. Click “OK” in “Create HDL Wrapper” pop-up.
4. In “Flow Navigator” click on “Open Elaborated Design”.
   1. If the “Elaborate Design” window pops up, click “OK” to dismiss, no action needed.
5. Click on “Window”->”I/O Ports” to open the I/O port pin planning window. 
6. Assign UART0 pins.
   1. In the “I/O Ports” tabbed window (on the bottom), expand “UART\_0\_####” and lower level “Scalar ports” to display “UART\_0\_rxd” and “UART\_0\_txd”. 
   2. Enter “E15” for “UART\_0\_rxd” in the “Site” column.
   3. Enter “D15” for “UART\_0\_txd” in the “Site” column. 
7. Change I/O standard used by UART0 pins.
   1. In the “I/O Ports” tab, change “UART\_0\_####” “I/O Std” to “LVCMOS33”. 
8. Add pullup to UART0 RX pin.
   1. In the “I/O Ports” tab, change “UART\_0\_rxd” “Pull Type” to I/O Std” to “PULLUP”. 
9. Run synthesis.
   1. “Flow Navigator” -> “Run Synthesis”.
   2. If the “Save Project” window pops up, click “Save”.
   3. If the “Save Constraints” window pops up, enter “system” for “File name:” and click “OK”.
   4. Wait until synthesis completes, which took 43 seconds on my PC.
      1. The “Synthesis Completed” window will pop up when complete.
10. Run implementation.
    1. In the “Synthesis Completed” pop-up, make sure “Run Implementation” is selected, then click “OK”.
    2. Wait until implementation completes, which took 1 minute and 10 seconds on my PC.
       1. The “Implementation Completed” window will pop up when complete.
11. Generate bitstream.
    1. In the “Implementation Completed” pop-up, select “Generate Bitstream”.
    2. Click “OK”.
    3. Wait for bitstream to be generated, which took 46 seconds on my PC.
       1. The “Bitstream Generation Completed” window will pop up when complete.
       2. Click “Cancel” on the “Bitstream Generation Completed” pop-up.
12. The bitstream, “system\_wrapper.bit”, should be located in the “enable-uart0.runs\impl\_1\” sub-directory from the project root directory, which for Windows is typically at “C:\Users\<your\_username>\enable\_uart0” and for Linux at “~/enable\_uart0”.
    1. The project root directory can be found at the top left of the Vivado window. 

At this point you have an FPGA bitstream that can be used to program your ZC702 board to route UART0’s RX and TX pins to PMOD1\_0, which is pin J63.1 on the board, and PMOD1\_1, pin J63.3, respectively. Follow the directions in “Debug How-To” for instructions on how to use this bitstream file to build a new BOOT.BIN that will enable UART0 for debugging.