Building HSR DANH on ZedBoard using FMC-GbE-RJ45

Version 0 Revision 2

March 29, 2019 (Jan. 7, 2019)

Future Design Systems
www.future-ds.com / contact@future-ds.com

Copyright © 2019 Future Design Systems, Inc.

Abstract

This manual describes details of building HSR DANH on Avnet ZedBoard using Future Design Systems FMC-GbE-RJ45 board.

Table of Contents

1 HSR DANH

As shown in Figure 1, HSR (High-availability Seamless Redundancy) DANH (Dual Attached Node with HSR) supports two HSL Links and is built using MAC[3] and HSR[4].

- Link-A port: it is one of two HSR Ethernet ports
- Link-B port: it is one of two HSR Ethernet ports

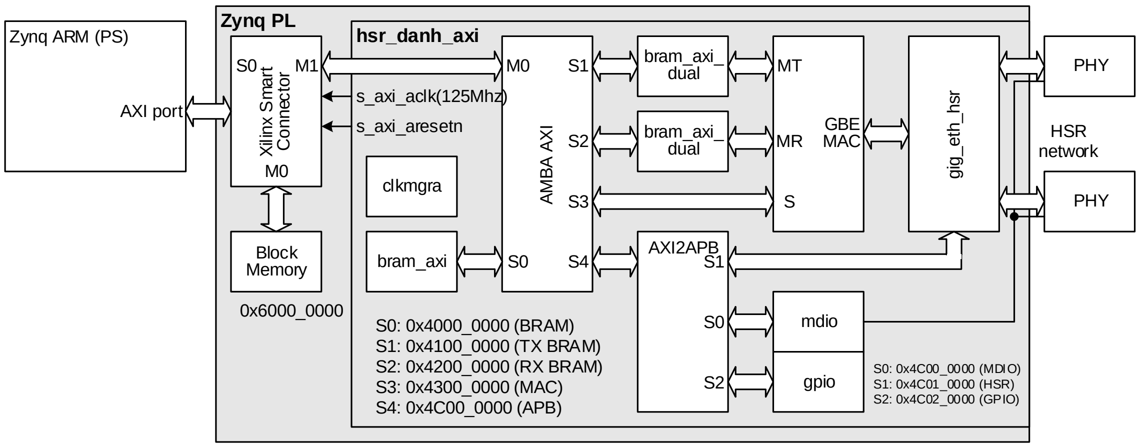

Figure 1: HSR DANH structure

ARM processor in the PS region can access blocks in the PL region through following addresses.

- S0: 0x4000_0000 (BRAM)
- S1: 0x4100_0000 (TX BRAM)
- S2: 0x4200_0000 (RX BRAM)
- S3: 0x4300_0000 (MAC)
- S4: 0x4C00_0000 (APB)
- P0: 0x4C00_0000 (MDIO)
- P1: 0x4C01_0000 (HSR)
- P2: 0x4C02_0000 (GPIO)

HSR DAMH is a kind of Ethernet MAC supporting dual-Ethenrt ports for HSR feature, where a processor send an Ethernet packet and it is duplicated and sent through two HSR port after adding HSR header as shown in Figure 2. HSR packet received from the HSR Link-A/B is forwarded to the upstream and the other HSR Link depending on MAC address and other information. More detaails should be referred to HSR specification[1].

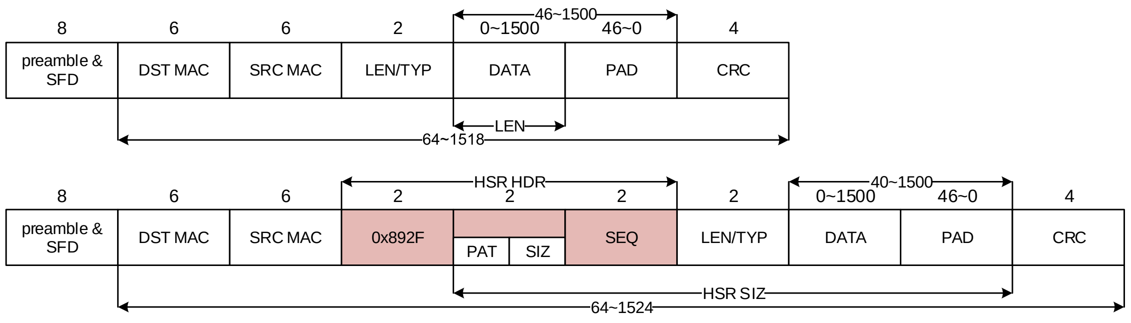

Figure 2: Normal packet and HSR packet

This document describes a details of building DANH using Future Design Systems HSR RTL[3] on the following hardware borads.

- Future Design Systems, FMC-GbE-RJ45[5]
- Avnet, ZedBoard[6]

2 HSR DANH on ZedBoard

Figure 3 shows how HSR DANH is implemented on ZedBoard.

- Zedboard: FPGA boarad to implement HSR design
- FMC-GbE-RJ45: 3 port Ethernet board

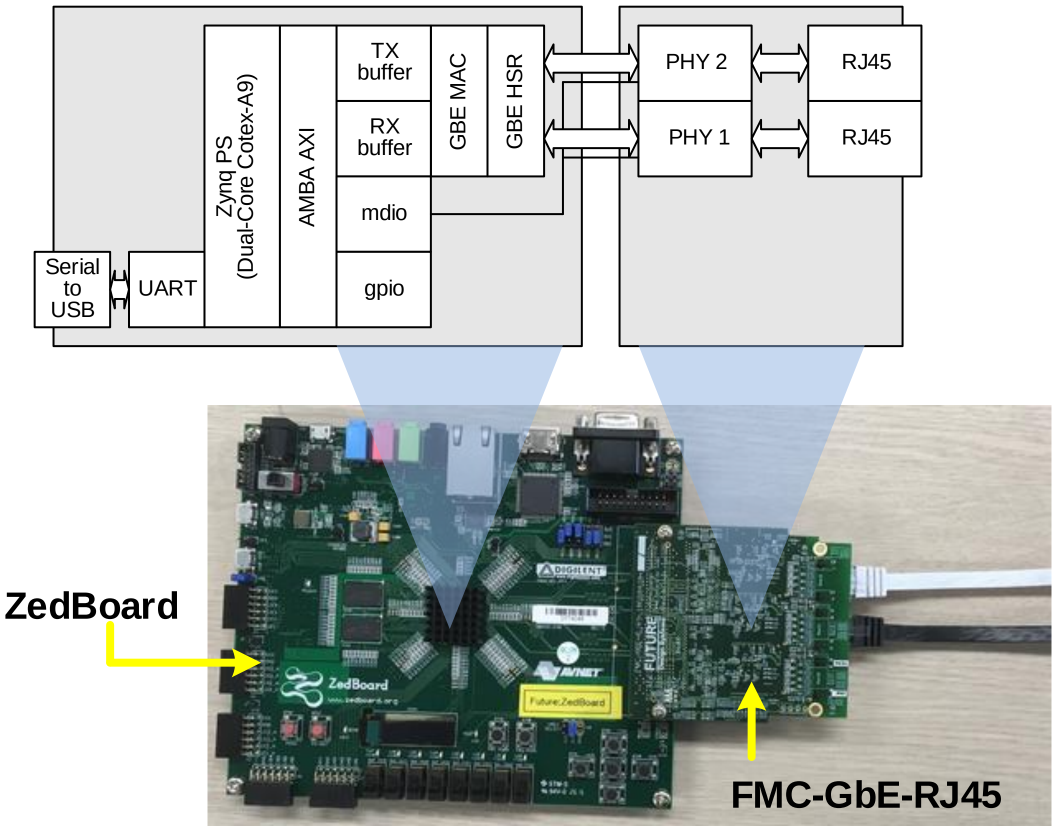

Figure 3: HSR DANH implementation on ZedBoard

2.1 Directory structure

'hw' directory contains hardware building project. 'sw.arm' directory contains software building project. 'bootgen' directory contains project to prepare bitstream that contains both hardware and software.

directory				remarks
hw	Hardware building projects			
	design	'hsr_danh_axi' building		
		verilog		
	bench	Test-bench		
		verilog		
	sim	simulation		
		modelsim.vivado		
	syn	Preparing 'hsr_danh_axi.edn'		
		vivado.zedboard.lpc		
	gen_ip	This project requires 'syn/vivado.zedboard.lpc/hsr_danh_axi.edn' and prepares 'hsr_danh_axi.xpr'		
		zedboard.lpc		
	impl	This project requires		

		‘gen_ip/zedboard.lpc/hsr_danh_axi.xpr’ and prepares ‘zed_board_wrapper.bit’ and ‘zed_bd_wrapper_sysdef.hdf’		
		zedboard.lpc		
sw.arm	Software building projects			
	fsbl[1]	This project prepares boot loader.		
	eth_send_receive	This project compiles user application program to test the hardware by sending and receiving packets		
	mem_test	This project compiles a memory testing program.		
bootgen	This project prepares SD Card image to boot the ZedBoard. It requires ‘sw.arm/fsbl/fsbl_o.elf’ and ‘hw/impl/zedboard.lpc/zed_b_wrapper.bit’ and user program.			

2.2 Testing structure

Figure 4 shows hardware structure to simulate a HSR system, where several HSR nodes are connected to build ring and each HSR node consists of ‘hsr_danh_axi’ and ‘tester’.

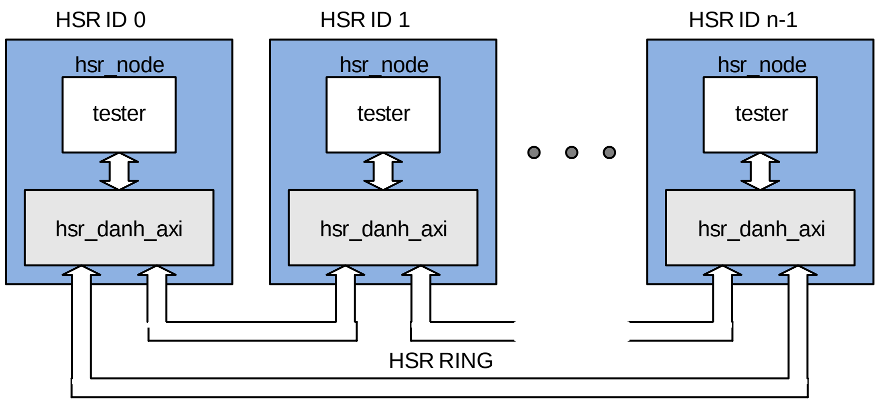

Figure 4: Testing structure

HSR nodes are connected through GMII instead of PHY and each ‘tester’ in the HSR node is assigned a unique MAC address related to node id such that ‘48’hF0_12_34_56_78_{node_id[7:0]’.

2.3 Simulation

Simulation requires Mentor Graphics ModelSim and Xilinx Vivado suites.

[1] FSBL: First Stage Boot Loader

- ✧ HDL simulator: Mentor Graphics ModelSim
- ✧ FPGA related library: Vivado 2017.4 for Zynq

It requires following environment variables and refer to ‘Makefile’ and ‘sim_define.v’ in ‘sim/modelsim.vivado’.

- ‘XILINX_VIVADO’ environment variable to point Vivado installation directory, such as ‘/opt/Xilinx/Vivado/2017.4’ for Linux or ‘C:/Xilinx/Vivado/2017.4’ for Windows.
- ‘FIP_HOME’ environment variable in the “Makefile” to point directory containing hardware blocks, such as ‘../../../../../FIP’.
- ‘FPGA_TYPE’ environment variable in the “Makefile” to specify FPGA type, such as ‘z7’ for Zynq 7000.
- ‘BOARD_ZED’ environment variable in “hw/sim/modelsim.vivado/sim_define.v” to specify FPGA board.

Do as follows to simulate this HSR system.

1. Make sure all necessary files are ready.
    - ✧ Go to ‘FIP/mem_axi/bram_simple_dual_port/z7/vivado.2017.4’ and run ‘make. (It takes time).
2. Go to ‘hw/sim/modelsim.vivado’ directory
3. Run ‘make’ for Linux
    - ✧ If any errors occurs, have a close look at the message.
    - ✧ Reasons of most of errors may be come from mismatches including path or environment variables.
    - ✧ Simulation version specific options may cause problems.
4. Invoke VCD waveform viewer to check simulation if simulation completes without any errors.
    - ✧ GTKwave would be a good choice

‘NUM_OF_HSR_NODE’ in ‘sim_define.v’ specifies the number of HSR nodes to simulate.

Figure 5 shows an example case that NODE 0 is only enabled to send a packet to NODE 1.

1. ‘tester’ in NODE 0 prepares a packet targeted to NODE 1.
2. ‘gig_eth_hsr’ in NODE 0 receives the packet and forward it to LINK-A and B after inserting HSR header.
3. NODE 1 receives the packet through LINK-B and forwards to the Upstream Link since MAC hit occurs.
4. NODE 2 receives the packet through LINK-A and forwards to the LINK-B.
5. NODE 1 receives the packet from NODE 2 through LINK-A, but discard it since it has been received already. (see step 3)

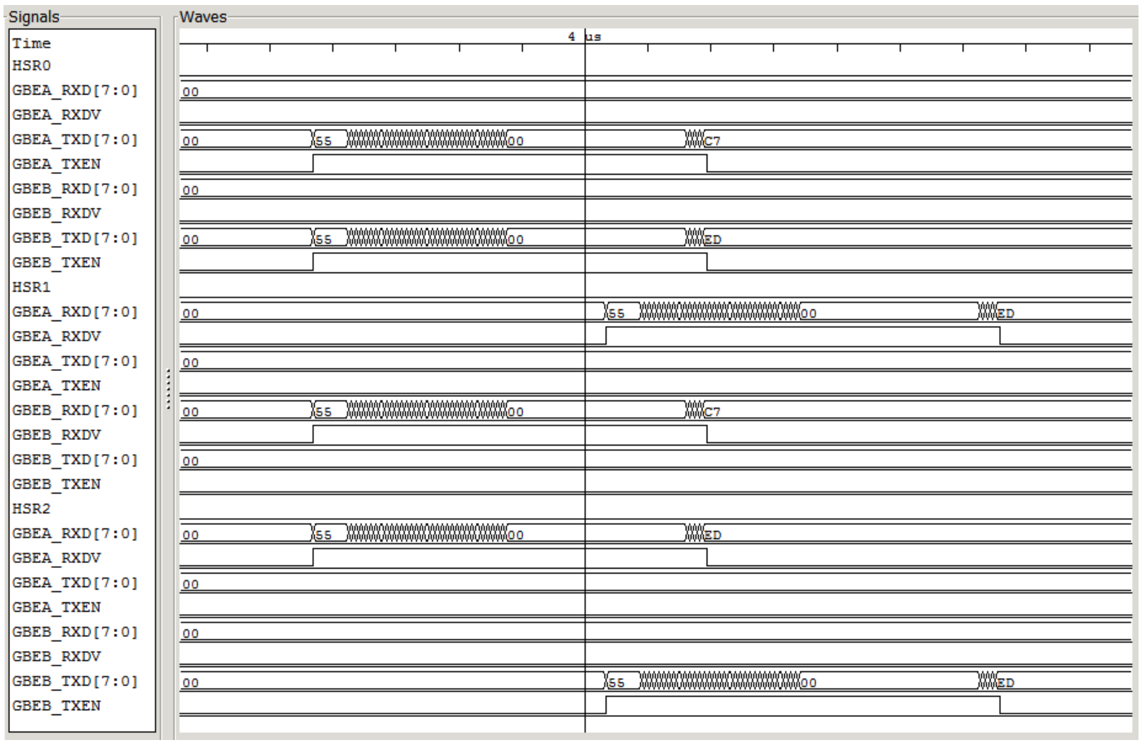

Figure 5: Simulation case NODE 0 generates a packet

3 Building system

3.1 Preparing bit files for HW and SW

Implementation requires Vivado and SDK suites.

✧ FPGA related library: Vivado 2017.4 for Zynq
✧ ARM related library: SDK

It requires following environment variables and refer to 'Makefile' and 'syn_define.v' in 'pnr/vivado.zedboard.lpc'.

- 'XILINX_VIVADO' environment variable to point Vivado installation directory, such as '/opt/Xilinx/Vivado/2017.4' for Linux or 'C:/Xilinx/Vivado/2017.4' for Windows.
- 'XILINX_SDK' environment variable in the "Makefile" to point SDK installation directory, such as '/opt/Xilinx/SDK/2017.4' for Linux or 'C:/Xilinx/SDK/2017.4'.
- 'FIP_HOME' environment variable in the "Makefile" to point 'gig_eth_hsr' directory, such as '../../../../FIP'.

Simply run 'RunAll.sh' or do as follows to implement this HSR system, where step 1 to 3 for hardware and step 4 to 5 for software

1. Make sure all necessary files are ready.
   ✧ Go to ‘FIP/mem_axi_dual/bram_true_dual_port/z7/vivado.2017.4’ and run ‘make’. (It takes time).
2. Go to ‘hw/syn/vivado.zed board.lpc’ directory and run ‘make’
   ✧ ‘hsr_danh_axi.edn’ should be ready
3. Go to ‘hw/gen_ip/zedboard.lpc’ directory and run ‘make’
   ✧ ‘hsr_danh_axi.xpr’ should be ready
4. Go to ‘hw/impl/zedboard.lpc’ directory
   ✧ ‘zed_bd_wrapper.bit’ and ‘zed_bd_wrapper_sysdef.hdf’ should be ready
5. Go to ‘sw.arm/fsbl’ directory and run ‘make’
   ✧ ‘fsbl_0.elf’ should be ready
6. Go to ‘sw.arm/eth_send_receive’ directory and run ‘make’
   ✧ ‘eth_send_receive.elf’ should be ready
7. Go to ‘bootgen’ directory and run ‘make’
   ✧ ‘BOOT.bin’ should be ready

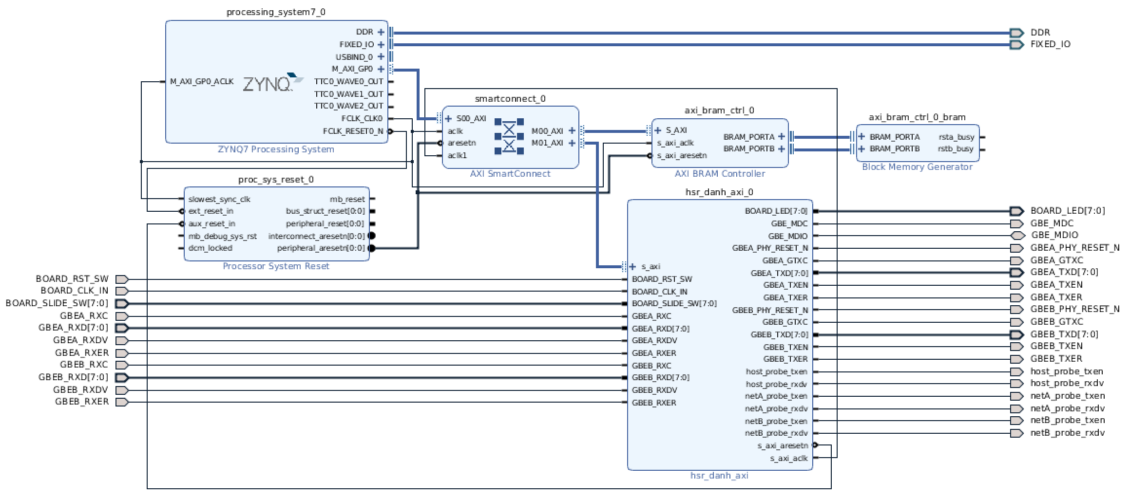

Figure 6: Vivado Block Design Diagram

3.2 Prepare SD Card

ZedBoard can be configured using SD Card. . <u>The boot configuration DIP switches should be set properly; consult manual for details.</u>

1. Prepare ZedBoard with FMC-GbE-RJ45
   ✧ Port 1 for HSR Link-A
   ✧ Port 2 for HSR Link-B
   ✧ For this DANH, Upstream link is not used (i.e., port 0)
2. Go to ‘bootgen’ directory
3. ‘BOOT.bin’ should be ready
4. Copy ‘BOOT.bin’ to the SD Card
5. Insert SD Card to the ZedBoard

- ✧ Make sure ZedBoard is turned off
- ✧ Make sure Boothing mode selection switches are properly set as shwon in Figure 7.

6. Turn on ZedBoard
   - ✧ Configuration DONE LED should be on.
   - ✧ Two LEDs should be on as shown in Figure 7; one for HSR ready and the other for DANH indicator.

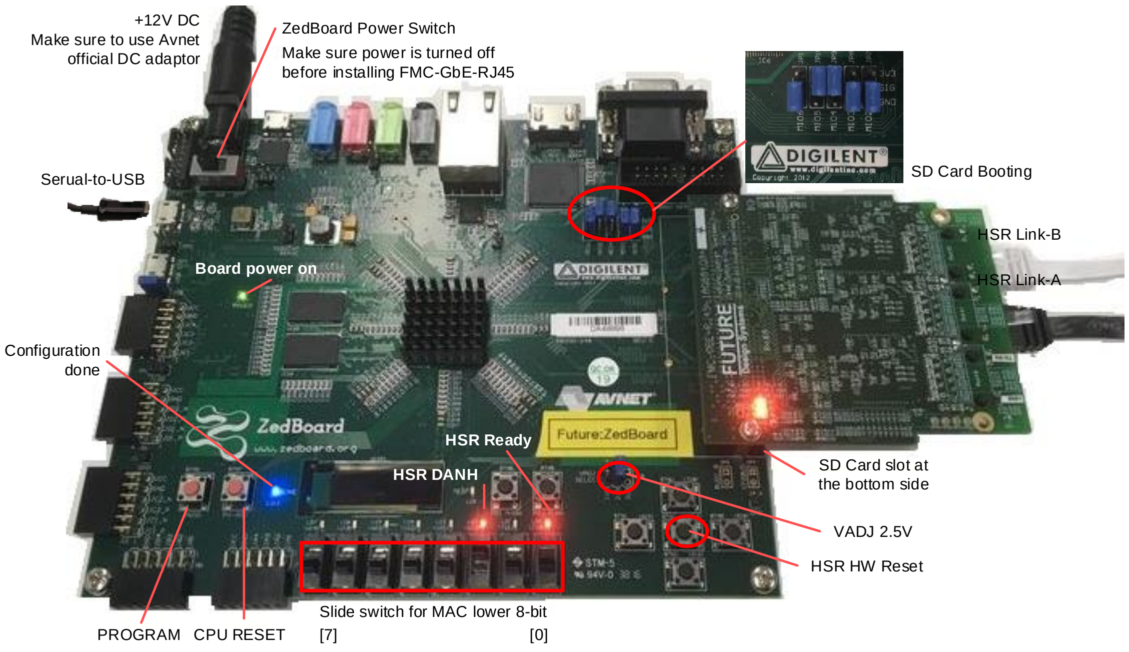

Figure 7: ZedBoard and FMC-GbE-RJ45

4 Testing with two DANHs

4.1 Setup

When all has been done as shown in Figure 7, a HSR system shown in Figure 8 can be set up using two DANH. In addition to this, USB-to-Serial connections are required to interact with ARM processor. Make sure that FMC-GbE-RJ45 port 1 and port 2 are connected (do not use port 0).

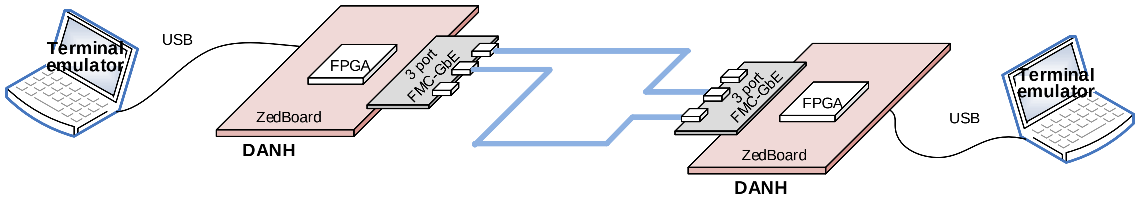

Figure 8: Two DANH conceptual

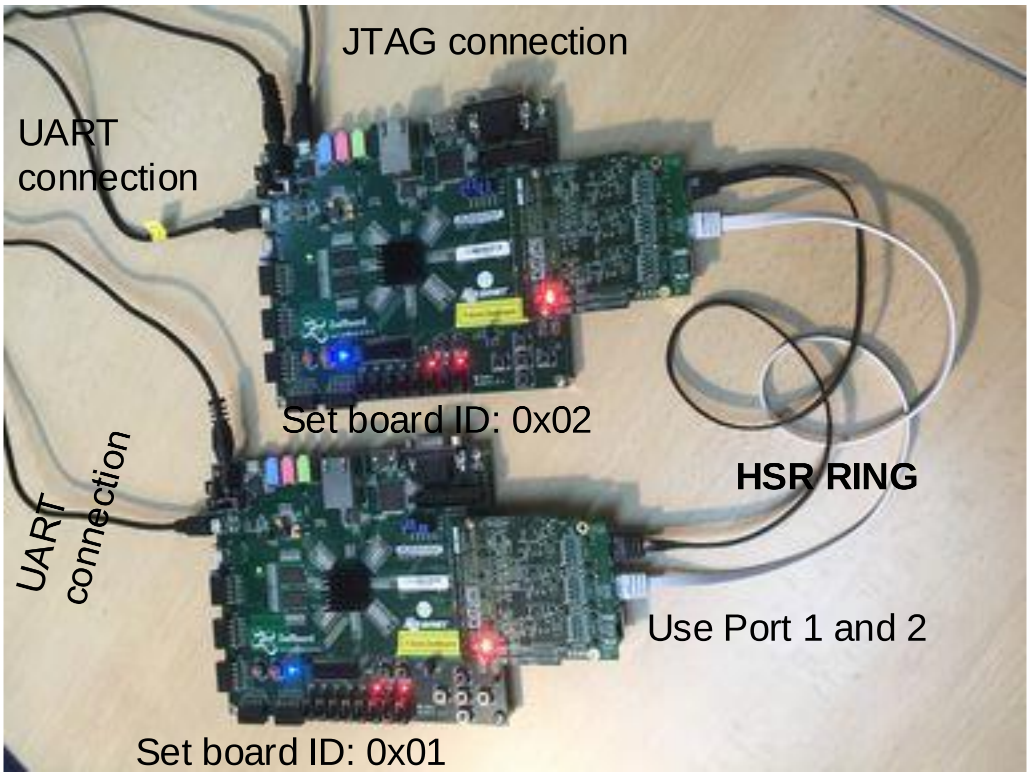

Figure 9: Two DANH with two ZedBoard

The serial port used following setings.

- baud rate: 115,200
- bit-width: 8-bit
- parity: no parity
- stop bit: 1-bit

One of following terminal emulator can be used.

- Teraterm for Windows
- Picocom[2] for Linux
  - $ sudo picocom -b 115200 /dev/ttyACM0
- GTKterm for Linux
  - $ gtkterm -s 115200 -p /dev/ttyACM1

4.2 Running

Following shows a typical case of testing with two ZedBoard and each row shows user interfaction with FDS-Monitor.

Terminal A	Terminal B	Remarks
monitor> **mac_init**	monitor> **mac_init**	initialize

[2] Use 'Ctl-AX" to exit from picocom.

monitor> **mac_addr -r** MAC 0x021234567801 HSR 0x021234567801	monitor> **mac_addr -r** MAC 0x021234567802 HSR 0x021234567802	check mac and hsr address. MAC and HSR should be the same.
pkt_snd -b *0x0213456788902*		send a packet to the other
	pkt_rcv -v 3 ETH mac dst: 0x02123456782 ETH mac src: 0x021234567801 ETH type leng: 0x0001 [60] 02:12:....	receive a packet
pkt_snd -b 0x021234567801 -r	**pkt_rcv -v 3 -r**	send and receive packets One sends packets continously. The other receives packets. Try to dis-connect network cable in order to check HSR recovery.

4.3 Changing bit and elf

When new FPGA design and ARM code are ready, download them through JTAG.

1. Go to ‘sw.arm/eth_send_receive’ directory
2. Run ‘make’ to compile software
3. Run ‘make debug’ to download bitstream and ELF.

References

[1] IEC 62439-3, Industrial communication networks –High availability automation networks –Part 3: Parallel Redundancy Protocol (PRP) and :q

[2] High-availability Seamless Redundancy (HSR), Edition 2.0, 2012-07.

[3] Future Design Systems, Gigabit Ethernet Media Access Controller, FDS-TD-2018-10-001, 2018. (company confidential)

[4] Future Design Systems, High-availability Seamless Redundancy Controller on Gigabit Ethernet, TD-2018-10-002, 2018. (company confidential)

[5] Future Design Systems, FMC-GbEGbE -RJ45 User Manual- Avnet ZedBoard, TD-2018-10-004, 2018.

[6] Avnet, ZedBoard (Zynq Evaluation and Development) Hardware User’s Guide, Jan. 2014.

Revision history

- ☐ 2019.03.29: Updated.
- ☐ 2019.03.03: Updated by Ando Ki.
- ☐ 2019.01.07: Document started by Ando Ki (adki@future-ds.com)

– End of document –