Implementing NTB Controller Using PCIe Endpoints

Kishon Vijay Abraham I

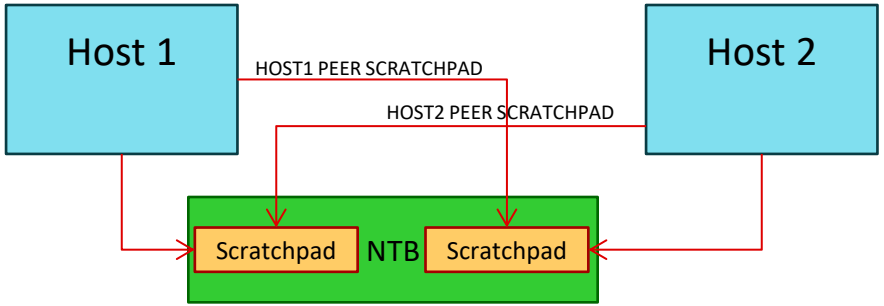
Introduction

- NTB – Non-transparent bridge
- Allows two RP to communicate with each other

- Implemented by certain processors or switches with NTB HW.
  - Intel® Xeon® Processor C5500/C3500 Series
  - IDT® 89HPES24NT6, 89HPES32NT8, 89HPES12NT12, 89HPES16NT16 etc..
- Constructs used to communicate with each other
  - Scratchpad Registers
  - Doorbell Registers
  - Memory windows

Constructs used in NTB

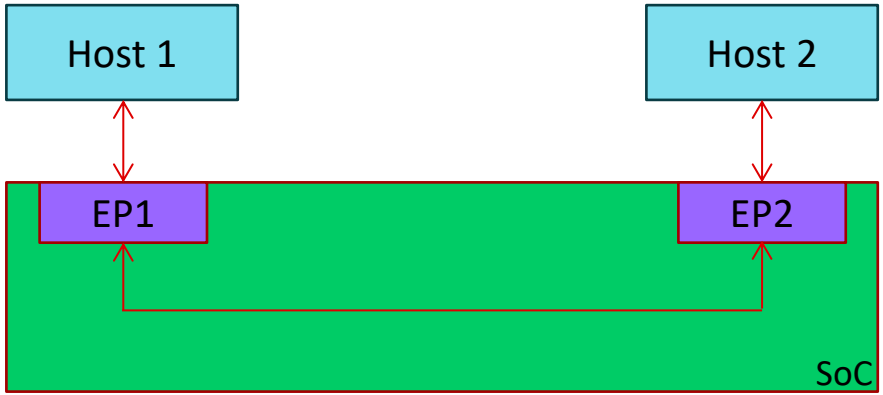
- Scratchpad Registers: Register space for each of the host (used by applications built over NTB)
  - Self Scratchpad
  - Peer Scratchpad



- Doorbell Registers: Registers to send interrupts from one side of NTB to the other
- Memory Window: Used for accessing buffers in the remote host for transferring data.

NTB Functionality using SoC

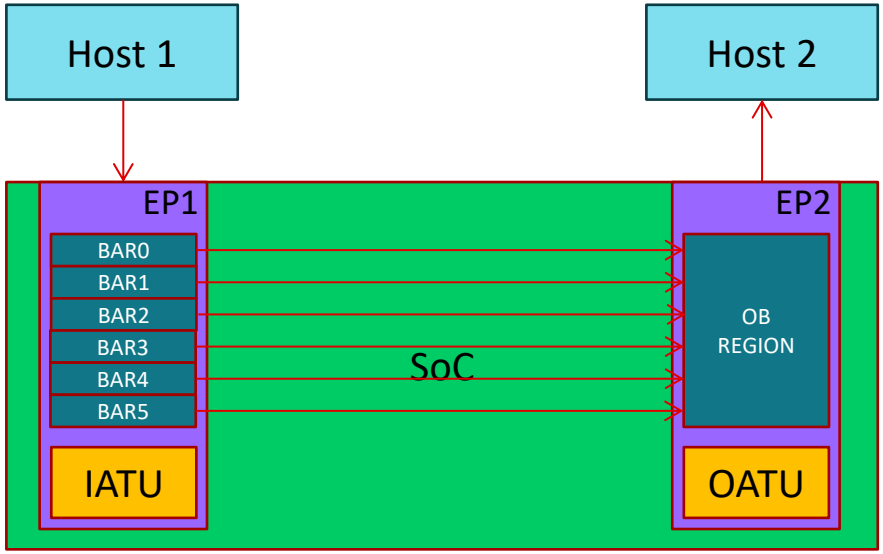
- NTB functionality can be built using SoCs with multiple EP instances.



- EP1 and EP2 should be configured such that transactions from one host is routed to the other host.

Endpoint Configuration in SoC

- Inbound Transactions (Implemented using BARs and Inbound ATU)
- Outbound Transactions (Implemented using OB regions and Outbound ATU)



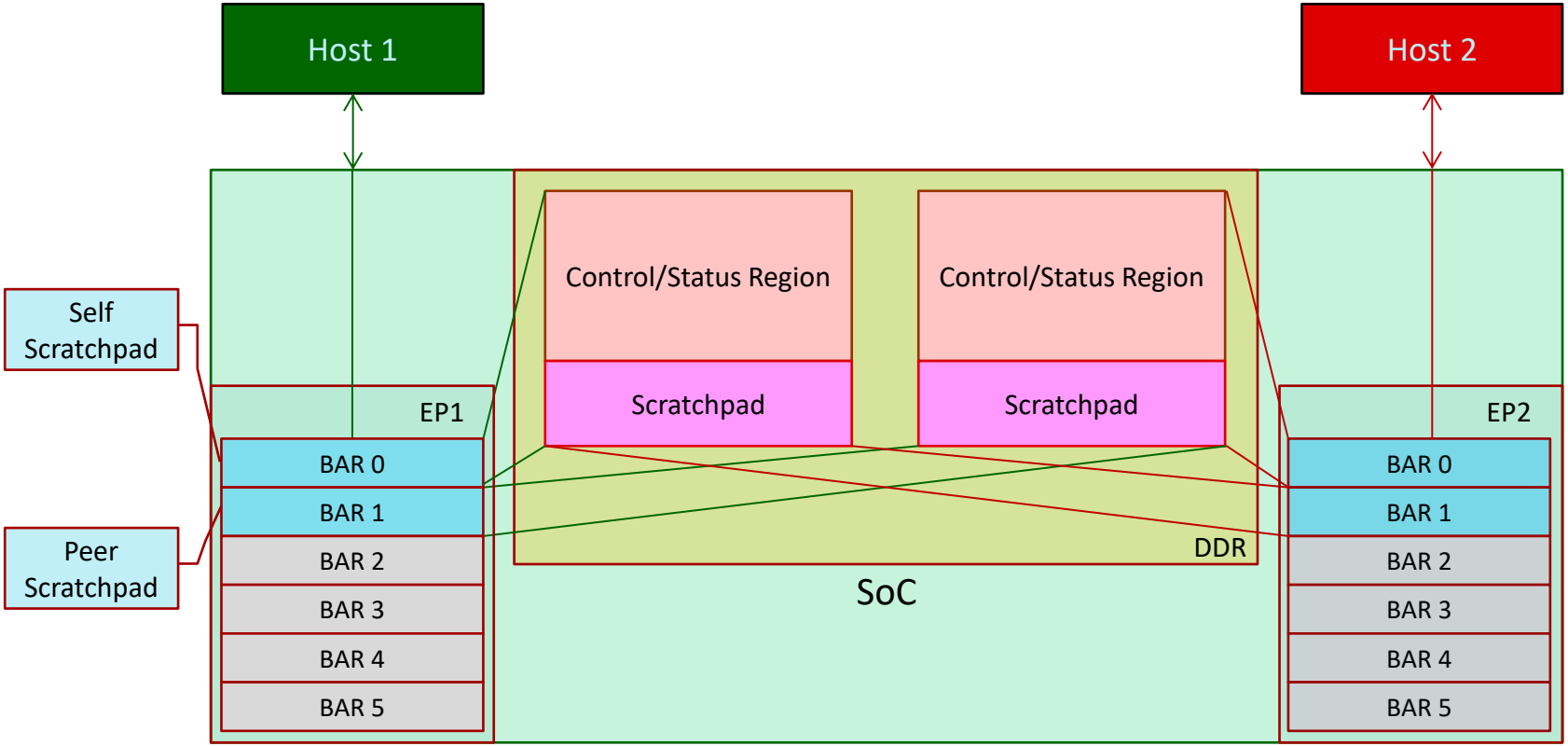
Constructs to be used in NTB

- Control/Status Region: Register space for configuring the endpoint controller
  - Hosts write into this region for configuring the OB ATU
  - Endpoint controller can populate SPAD offset, number of memory windows etc..
- Scratchpad Registers: Register space for each of the host (used by applications built over NTB)
  - Self Scratchpad
  - Peer Scratchpad
- Doorbell Registers: Registers to send interrupts from one side of NTB to the other
- Memory Window: Used for accessing buffers in the remote host for transferring data.

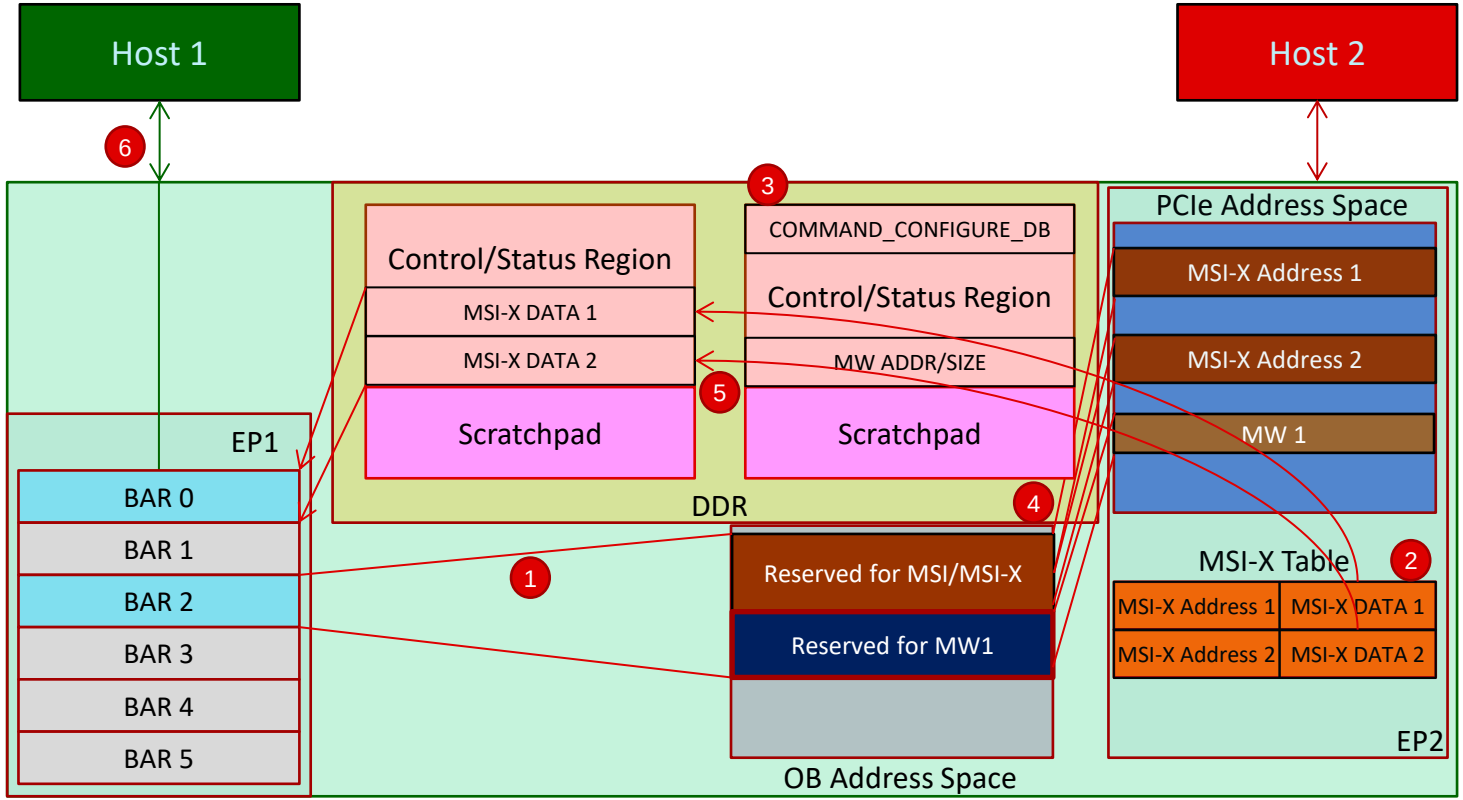
Modeling constructs

- Six 32 bit BARs for 5 regions (Control status region, Self scratchpad region, peer scratchpad region, doorbell region, memory window)
  - Platform supporting only 64 bit BARs cannot provide all the regions
  - Platforms having reserved BARs cannot support all the regions
- Solution is to pack the regions without impacting functionality
  - BAR0 -> Control/Status Region + Self Scratchpad Region
  - BAR1 -> Peer Scratchpad Region
  - BAR2 -> Doorbell Region + Memorywindow1
  - BAR3..BAR5 -> Additional memory window

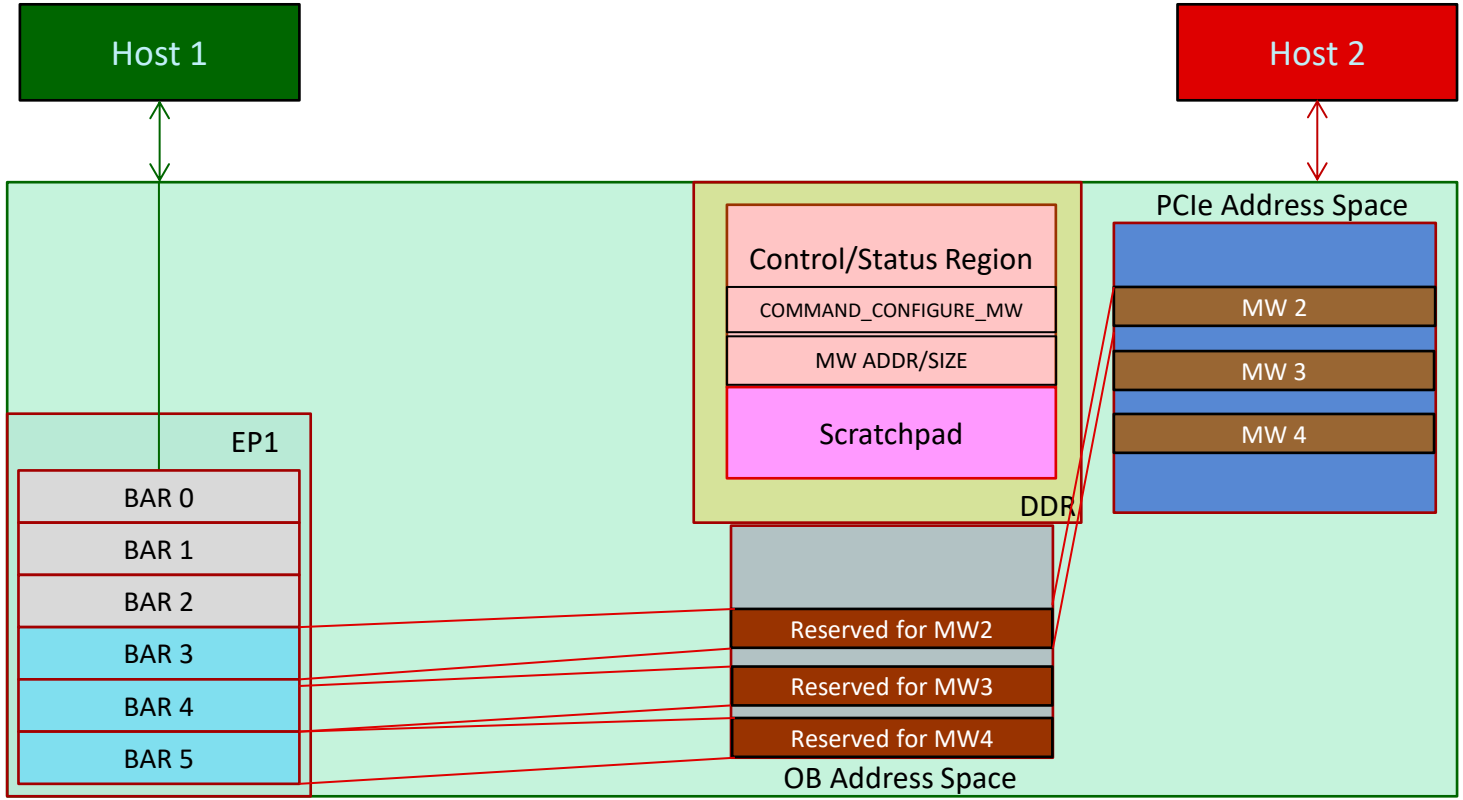
Control, Status and Scratchpad Registers



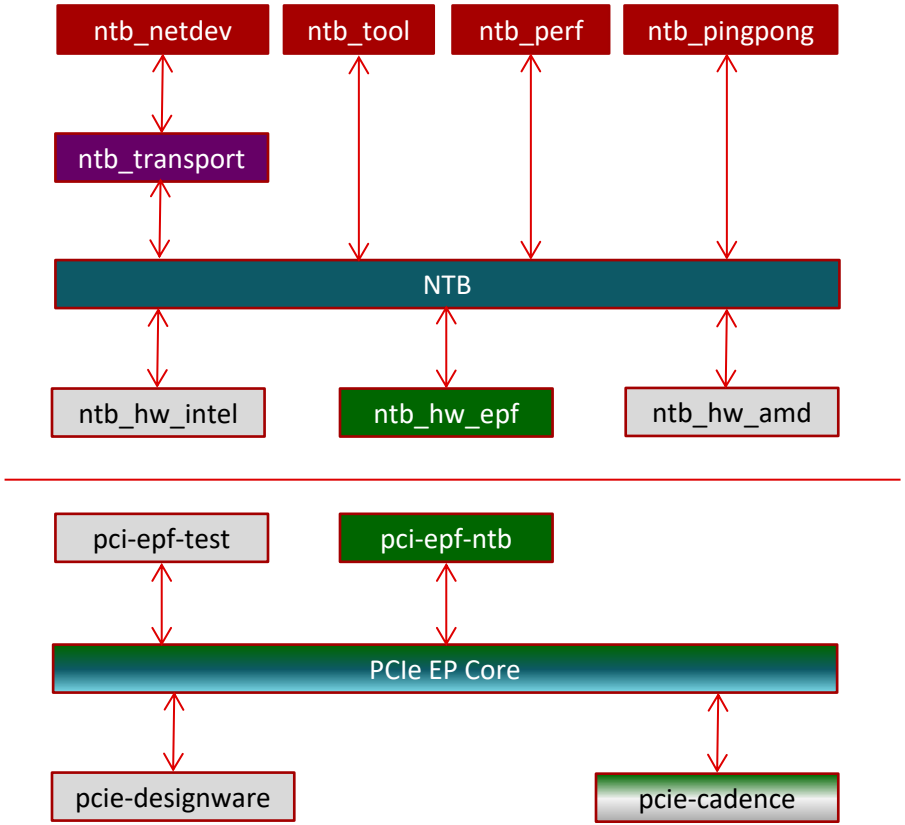
Doorbell Registers/MemoryWindow1

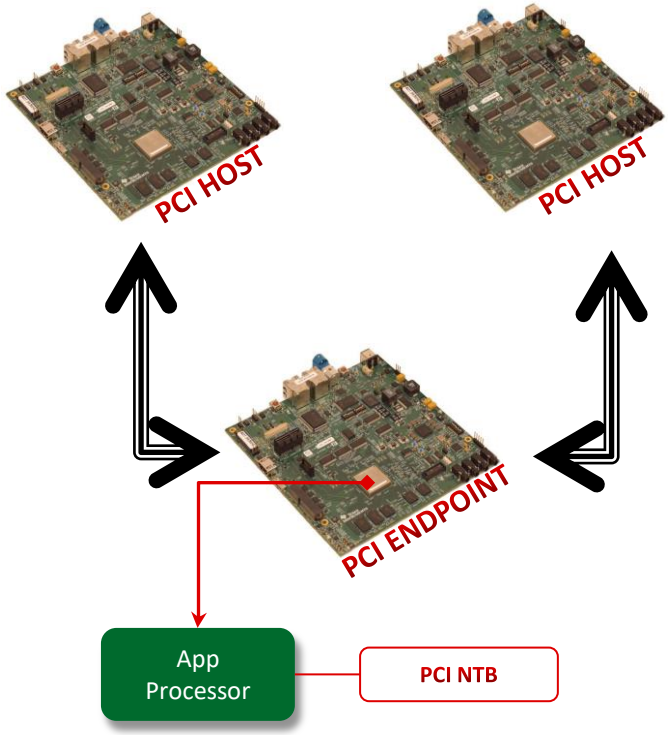


Additional Memory Window



NTB SW Architecture





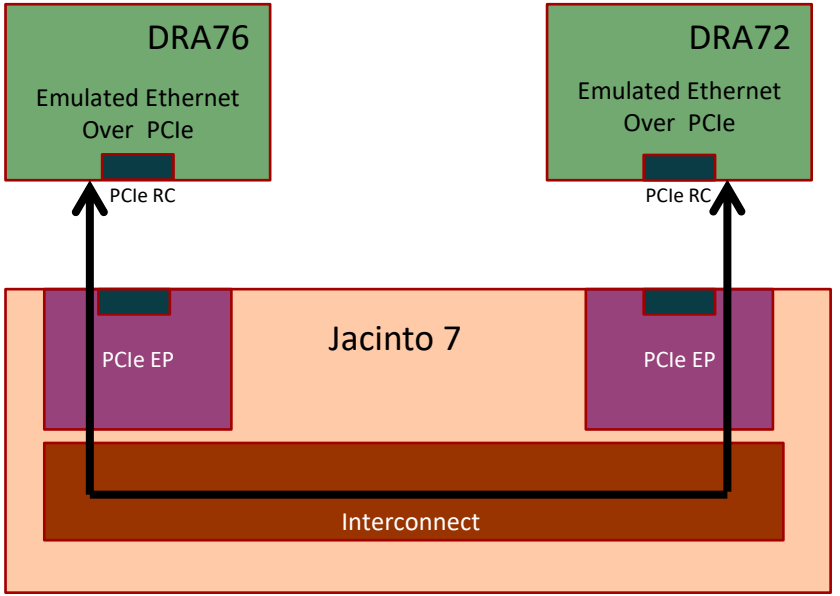
Device Tree Vs Configfs

```
mount -t configfs none /sys/kernel/config
cd /sys/kernel/config/pci_ep/
mkdir functions/pci_epf_test/func1
echo 0x104c > functions/pci_epf_test/func1/vendorid
echo 0xb00d > functions/pci_epf_test/func1/deviceid
echo 1 > functions/pci_epf_test/func1/msi_interrupts
echo 16 > functions/pci_epf_test/func1/msix_interrupts
ln -s functions/pci_epf_test/func1 controllers/d800000.pcie-ep/
echo 1 > controllers/d800000.pcie-ep/start
```

```
  epf_bus {
                compatible = "pci-epf-bus";
                ntb {
                        compatible = "pci-epf-ntb";
                        epcs = <&pcie0_ep>, <&pcie1_ep>;
                        epc-names = "primary", "secondary";
                        vendor-id = /bits/ 16 <0x104c>;
                        device-id = /bits/ 16 <0xb00d>;
                        spad-count = <32>;
                        num-mws = <4>;
                        mws-size = <0x100000>, <0x100000>,
<0x100000>, <0x100000>;
                };
        };
echo 1 > controllers/d800000.pcie-ep/start
echo 1 > controllers/d000000.pcie-ep/start
```

Status

- Demonstrate Ethernet emulation over PCIe between 2 DRA7x boards using J7 as NTB
  - Performed ping test and iperf test
  - Some more cleanup pending and should post the patches in a couple of weeks



References

- **Using Non-transparent Bridging in PCI Express Systems** By Jack Regula
- https://events.static.linuxfound.org/sites/events/files/slides/Linux%20NTB_0.pdf
- https://github.com/jonmason/ntb/wiki
- Linux Kernel: Documentation/ntb.txt
- Linux Kernel: Documentation/PCI/endpoint/

Happy Hacking