System Device Tree
Bus Firewalls and Interconnects

Oct 2020

XILINX.

Bus Firewalls – XMPU, XPPU

- **In collaboration with ST Microelectronics**
- **Description and configuration of XMPU-like devices**
- **Hardware description**
  - Firewall Controllers
  - Firewall property to link a device with its firewall controller
  - Bus Master IDs
- **Firewall Configuration**
  - Firewallconf
  - Firewallconf-default

```
amba_xppu: indirect-bus@1 {
        compatible = "indirect-bus";

        lpd_xppu: xppu@ff990000 {
                compatible = "xlnx,xppu";
                reg = <0xff990000 0x1000>;
                #firewall-cells = <0x0>;
        };

        pmc_xppu: xppu@f1310000 {
                compatible = "xlnx,xppu";
                reg = <0xf1310000 0x1000>;
                #firewall-cells = <0x0>;
        };
};


mmc0: sdhci@f1050000 {
        bus-master-id = <0x243>;
        firewall-0 = <&pmc_xppu>;
};

domain@1 {
        /* 1: block */
        firewallconfig = <&linux1 1 0>;
};
```

Bus-Firewalls in System Device Tree: Goals

- describe bus firewall controllers
- describe which device is protected by which controller
- describe a new ID space to configure the firewalls
- describe domains-based firewall configurations in System Device Tree

Bus-Firewall Controllers

- **a node describing a firewall controller**
  - there can be multiple controllers in a system
- **a #firewall-cells property to define firewall-specific extra information**
  - #firewall-cells can be 0
- **a firewall property under each device node linking to the appropriate firewall controller**
  - it links to the firewall controller protecting accesses *to the device MMIO regions*

Bus Master IDs

- Bus mastering devices are identified by bus-firewalls using IDs. Their transactions are marked with a device ID. These IDs are used to configure bus-firewalls.
- We shall call these IDs "Bus Master IDs"
- We shall advertise them on device tree using a new property: bus-master-id

```
dev0: device@0 {
        bus-master-id = <&lpd_xppu 0x212>;
```

Example

```
amba_xppu: indirect-bus@1 {
        compatible = "indirect-bus";
        #address-cells = <0x2>;
        #size-cells = <0x2>;

        lpd_xppu: xppu@ff990000 {
            compatible = "xlnx,xppu"
            #firewall-cells = <0x0>;
            reg = <0x0 0xff990000 0x0 0x1000>;
        };
        pmc_xppu: xppu@f1310000 {
            compatible = "xlnx,xppu"
            #firewall-cells = <0x0>;
            reg = <0x0 0xf1310000 0x0 0x1000>;
        };
    };

    amba {
        ethernet0: ethernet@ff0c0000 {
            bus-master-id = <&lpd_xppu 0x234 &pmc_xppu 0x234>;
            firewall-0 = <&lpd_xppu>;
        };

        can0: can@ff060000 {
            firewall-0 = <&lpd_xppu>;
        };

        mmc0: sdhci@f1050000 {
            bus-master-id = <&lpd_xppu 0x243 &pmc_xppu 0x243>;
            firewall-0 = <&pmc_xppu>;
        };

        serial0: serial@ff000000 {
            firewall-0 = <&lpd_xppu>;
        };
    };
```

System Device Tree Hardware Description Example

```
cpus_r5: cpus-cluster@0 {
        #address-cells = <0x1>;
        #size-cells = <0x0>;
        #cpus-mask-cells = <0x1>;
        compatible = "cpus,cluster";

        bus-master-id = <&lpd_xppu 0x0 &pmc_xppu 0x0 &lpd_xppu 0x1 &pmc_xppu 0x1>;
    };

    amba {
        ethernet0: ethernet@ff0c0000 {
            bus-master-id = <&lpd_xppu 0x234 &pmc_xppu 0x234>;
            firewall-0 = <&lpd_xppu>;
        };

        can0: can@ff060000 {
            firewall-0 = <&lpd_xppu>;
        };

        mmc0: sdhci@f1050000 {
            bus-master-id = <&lpd_xppu 0x243 &pmc_xppu 0x243>;
            firewall-0 = <&pmc_xppu>;
        };

        serial0: serial@ff000000 {
            firewall-0 = <&lpd_xppu>;
        };
    };
```

Interconnect Bindings

- **Documentation/devicetree/bindings/interconnect/interconnect.txt**
  - interconnect controllers and #interconnect-cells
  - interconnects property for consumers

```
snoc: interconnect@580000 {
    compatible = "qcom,msm8916-snoc";
    #interconnect-cells = <1>;
    reg = <0x580000 0x14000>;
    clock-names = "bus_clk", "bus_a_clk";
    clocks = <&rpmcc RPM_SMD_SNOC_CLK>,
         <&rpmcc RPM_SMD_SNOC_A_CLK>;
};

sdhci@7864000 {
    ...
    interconnects = <&pnoc MASTER_SDCC_1 &bimc SLAVE_EBI_CH0>;
    interconnect-names = "sdhc-mem";
};
```

Interconnect Bindings vs Firewall Bindings

- **Similar concepts**
- **Firewall controller and Interconnect controller are separate hardware blocks**
  - how to distinguish between the two controllers?
  - they are configured independently (e.g. different DTSi files)
  - QoS and firewalls are configured independently
  - it would be good to keep QoS and firewall data separate
- **Using Interconnect Bindings for Firewalls**
  - firewall controller -> interconnect controller
  - #firewall-cells -> #interconnect-cells
  - bus-master-id -> interconnects
  - firewall-0 -> **missing**
  - there is no distinction between a device as bus master or as slave
- **we need a way to describe firewall interactions as master and as slave**

Interconnect Bindings vs Firewall Bindings

```
 amba_xppu: indirect-bus@1 {
   compatible = "indirect-bus";
   #address-cells = <0x2>;
   #size-cells = <0x2>;

   lpd_xppu: xppu@ff990000 {
       compatible = "xlnx,xppu"
       #interconnect-cells = <0x0>;
       reg = <0x0 0xff990000 0x0 0x1000>;
   };

   pmc_xppu: xppu@f1310000 {
       compatible = "xlnx,xppu"
       #interconnect-cells = <0x0>;
       reg = <0x0 0xf1310000 0x0 0x1000>;
   };
};
```

```
amba {
    ethernet0: ethernet@ff0c0000 {
        interconnects = <&lpd_xppu 0x234 &pmc_xppu 0x234>;
        firewall-0 = <&lpd_xppu>; // XXX
    };

    can0: can@ff060000 {
        firewall-0 = <&lpd_xppu>; // XXX
    };

    mmc0: sdhci@f1050000 {
        interconnects = <&lpd_xppu 0x243 &pmc_xppu 0x243>;
        firewall-0 = <&pmc_xppu>; // XXX
    };

    serial0: serial@ff000000 {
        firewall-0 = <&lpd_xppu>; // XXX
    };
};
```

Interconnect Bindings for Bus Firewalls

- **Reuse the same controller for both QoS and Firewall configurations**
- **Add two new consumer properties**
  - interconnects-master (and #interconnect-master-cells)
  - interconnects-slave (and #interconnect-slave-cells)
- **Specify both master and slave firewall interactions**
- **Keep the QoS data and Firewall data separately**

Interconnect Bindings for Bus Firewalls

```
amba_xppu: indirect-bus@1 {
  compatible = "indirect-bus";
  #address-cells = <0x2>;
  #size-cells = <0x2>;

  lpd_xppu: xppu@ff990000 {
      compatible = "xlnx,xppu"
      #interconnect-cells = <0x4>;
      #interconnect-master-cells = <0x1>;
      #interconnect-slave-cells = <0x0>; >;
      reg = <0x0 0xff990000 0x0 0x1000>;
  };

  pmc_xppu: xppu@f1310000 {
      compatible = "xlnx,xppu"
      #interconnect-cells = <0x4>;
      #interconnect-master-cells = <0x1>;
      #interconnect-slave-cells = <0x0>;
      reg = <0x0 0xf1310000 0x0 0x1000>;
  };
};
```

```
amba {
    ethernet0: ethernet@ff0c0000 {
        interconnects-master = <&lpd_xppu 0x234 &pmc_xppu 0x234>;
        interconnects-slave = <&lpd_xppu>;
                                          0   1
interconnects = <&lpd_xppu MASTER 0x234 SLAVE 0x0
                 &pmc_xppu MASTER 0x234 0x0 0x0
                 &pmc_xppu MASTER 0x234 0x0 0x0
                 &pmc_xppu MASTER 0x234 0x0 0x0

    };

    can0: can@ff060000 {
        interconnects-slave = <&lpd_xppu>;
    };

    mmc0: sdhci@f1050000 {
        interconnects-master = <&lpd_xppu 0x243 &pmc_xppu 0x243>;
        interconnects-slave = <&pmc_xppu>;
    };

    serial0: serial@ff000000 {
        interconnects-slave = <&lpd_xppu>;
    };
};
```

Adaptable.
Intelligent.

XILINX.