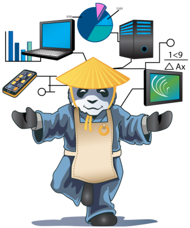
How to passthrough your integrated device to a VM on ARM

Julien Grall
julien.grall@citrix.com

Xen Developper Summit 2015



Platform Device

- Directly integrated in the ARM SOC
- Non-discoverable device
- Described via Device Tree

Device Tree - 1

- Tree data structure
- Each node describe the physical devices in a system

Device Tree - 2

```
/dts-v1/;

/ {
        model = "XENVM-4.2";
        compatible = "xen,xenvm-4.2", "xen,xenvm";
        interrupt-parent = <&gic>;
        #address-cells = <2>;
        #size-cells = <2>;

        cpus {
                #address-cells = <1>;
                #size-cells = <0>;

                cpu@0 {
                        device_type = "cpu";
                        compatible = "arm,cortex-a15";
                        reg = <0>;
                };
        };

        gic: int-controller@2c001000 {
                compatible = "arm,cortex-a15-gic", "arm,cortex-a9-gic";
                #interrupt-cells = <3>;
                #address-cells = <0>;
                interrupt-controller;
                reg = <0 0x2c001000 0 0x1000>,
                     <0 0x2c002000 0 0x100>;
        };
};
```

Device path

- Every node has a unique path
  - Concatenation of parent nodes separated by **/**
- Examples
  - **cpu@0**: */cpus/cpu@0*
  - **int-controller@2c001000**: /int-controller@2c001000

DOM0

- Special case of passthrough
- Every device are assigned by Xen to DOM0 at boot

Problems with platform device passthrough

- Device can depends on other devices
  - PHY
  - Clock
  - ...
- Some node properties should be dropped
- Reset is device specific

Requirements

- Hardware with IOMMU
  - SMMUv1 and SMMUv2 supported
- Device path of the device to passthrough
- Know the dependencies of the device

Example used

- Midway server from Calxeda
- SMMUv1 supported
- Second network card

Example used - DTS

```
/ {

        soc {
                #address-cells = <1>;
                #size-cells = <1>;
                compatible = "simple-bus";
                interrupt-parent = <&intc>;

[...]

                mac0: ethernet@fff50000 {
                        compatible = "calxeda,hb-xgmac";
                        reg = <0xfff50000 0x1000>;
                        interrupts = <0 77 4 0 78 4 0 79 4>;
                        dma-coherent;
                        #stream-id-cells = <2>;
                };

                mac1:  ethernet@fff51000 {
                        compatible = "calxeda,hb-xgmac";
                        reg = <0xfff51000 0x1000>;
                        interrupts = <0 80 4 0 81 4 0 82 4>;
                        dma-coherent;
                        #stream-id-cells = <2>;
                };
        };
};
```

Mark the device for passthrough

- Used to notify Xen the device will be passthrough
- Property **xen,passthrough** used in DT node
- Example for U-Boot
  - *fdt set /soc/ethernet@fff51000 xen,passthrough*

Why?

- Disabling the device from DOM0 during runtime is complex
- Xen need to tell DOM0 not using the device at boot
  - Property **status = "disabled"** used DT node
  - OS already knows the property

Xen DTS

```
/ {

        soc {
                #address-cells = <1>;
                #size-cells = <1>;
                compatible = "simple-bus";
                interrupt-parent = <&intc>;

[...]

                mac0: ethernet@fff50000 {
                        compatible = "calxeda,hb-xgmac";
                        reg = <0xfff50000 0x1000>;
                        interrupts = <0 77 4 0 78 4 0 79 4>;
                        dma-coherent;
                        #stream-id-cells = <2>;
                };

                mac1: ethernet@fff51000 {
                        compatible = "calxeda,hb-xgmac";
                        reg = <0xfff51000 0x1000>;
                        interrupts = <0 80 4 0 81 4 0 82 4>;
                        dma-coherent;
                        xen,passthrough;
                        #stream-id-cells = <2>;
                };
        };
};
```

DOM0 DTS

```
/ {

        soc {
                #address-cells = <1>;
                #size-cells = <1>;
                compatible = "simple-bus";
                interrupt-parent = <&intc>;

[...]

                mac0: ethernet@fff50000 {
                        compatible = "calxeda,hb-xgmac";
                        reg = <0xfff50000 0x1000>;
                        interrupts = <0 77 4 0 78 4 0 79 4>;
                        dma-coherent;
                        #stream-id-cells = <2>;
                };

                mac1: ethernet@fff51000 {
                        compatible = "calxeda,hb-xgmac";
                        reg = <0xfff51000 0x1000>;
                        interrupts = <0 80 4 0 81 4 0 82 4>;
                        dma-coherent;
                        status = "disabled";
                        #stream-id-cells = <2>;
                };
        };
};
```

Partial Device Tree - What is it?

- Allow the user to pass additional nodes to the guest DT
- Everything under the following nodes will be copied:
  - **/passthrough**
  - **/aliases**

Skeleton DTS

```
/ {
        /* #*cells are here to keep DTC happy */
        #address-cells = <2>;
        #size-cells = <2>;

        aliases {
                /* List of your aliases */
        };

        passthrough {
                compatible = "simple-bus";
                ranges;
                #address-cells = <2>;
                #size-cells = <2>;
                /* List of your nodes */
        };
};
```

How to write it?

- Interrupt numbers:
  - Same as the hardware
  - Only **SPIs** are supported
- MMIO regions:
  - Need to find a hole in the guest layout
  - Guest layout defined in *xen/include/public/arch-arm.h*
  - The layout may change between Xen versions
- Device specific properties:
  - Not all the properties can be copied

Example HW DTS

```
/ {

        soc {
                #address-cells = <1>;
                #size-cells = <1>;
                compatible = "simple-bus";
                interrupt-parent = <&intc>;

[...]

                mac0: ethernet@fff50000 {
                        compatible = "calxeda,hb-xgmac";
                        reg = <0xfff50000 0x1000>;
                        interrupts = <0 77 4 0 78 4 0 79 4>;
                        dma-coherent;
                        #stream-id-cells = <2>;
                };

                mac1:  ethernet@fff51000 {
                        compatible = "calxeda,hb-xgmac";
                        reg = <0xfff51000 0x1000>;
                        interrupts = <0 80 4 0 81 4 0 82 4>;
                        dma-coherent;
                        #stream-id-cells = <2>;
                };
        };
};
```

Example partial DTS

```
/dts-v1/;

/ {
        #address-cells = <2>;
        #size-cells = <2>;

        aliases {
                net = &mac0;
        };

        passthrough {
                compatible = "simple-bus";
                ranges;
                #address-cells = <2>;
                #size-cells = <2>;
                mac0:  ethernet@10000000 {
                        compatible = "calxeda,hb-xgmac";
                        reg = <0 0x10000000 0 0x1000>;
                        interrupts = <0 80 4 0 81 4 0 82 4>;
                        ~~dma-coherent;~~
                        ~~#stream-id-cells = <2>;~~
                };
        };
```

What to add?

- **device_tree = "path"**
  - Path to the partial device tree.
  - Only trusted device tree should be passed
- **dtdev = ["DTpath1", "DTpath2", ...]**
  - List of device to passthrough
  - Used to setup the SMMU
  - Only device protected by SMMU should be list

What to add? - 2

- **irqs = [irq1, irq2, ...]**
  - List of interrupts to route
- **iomem = ["START,NUM[@GFN]", ...]**
  - List of MMIO to assign
  - **START**: Start frame of the I/O region
  - **NUM**: Number of 4K pages to assign
  - **GFN**: Start frame in the guest layout (optional)

Example guest configuration

```
device_tree = "/root/guest-midway.dtb"
dtdev = [ "/soc/ethernet@fff51000" ]
irqs = [ 112, 113, 114 ]
iomem = [ "0xfff51,1@0x10000" ]
```

Reset

- Require a specific reset code per device
  - See whether we can share with VFIO
- DOM0 must be able to remove a device
  - Interrupts can't be shared between domains

Clock

- Clock may be shared between multiple devices
    - Can't passthrough the clock to the guest
- How to handle the clock in the guest?

DT improvement

- MMIO/Interrupts needs to be described in the DT and Xen cfg
    - More works for the user
- Introduce Xen bindings the DT to specify HW mapping?

Further reading

- `http://www.devicetree.org/Device_Tree_Usage`
- `http://xenbits.xen.org/docs/unstable/misc/arm/passthrough.txt`

Fin