System devicetree basics

Martí Bolívar | 2022-11-01

Goals for today

Using nRF5340 as an example

- How to define CPUs in system DT
- How to define memory maps in system DT

Issues:

- **Better support for multi-core AMP SoCs #51833**

- **System devicetree support #51830**

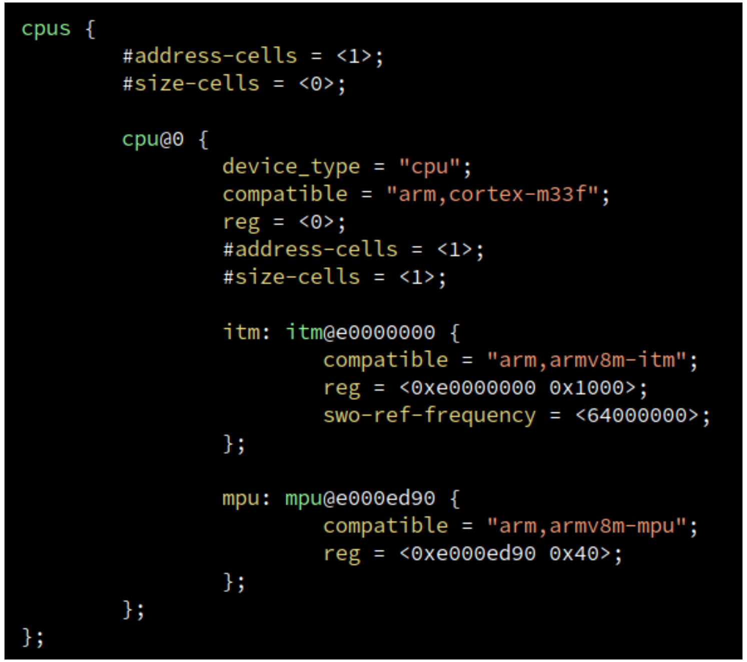
nRF5340 summary

- Dual-core AMP SoC
- “Application” core: Cortex-M33 with TrustZone
  - Your app goes here
  - Trusted firmware M goes here if you want
- “Network” core: Cortex-M33 without TZ
  - Bluetooth / etc. controller app goes here
- Network core can “see” application core memory for IPC, etc.

System DT intro

- Evolving specification
- Mainly worked on by Xilinx, but also Linaro’s Devicetree Evolution (DTE) group
- Meets regularly with DT spec / Linux DT maintainers
- Tweaks to the spec to get Zephyr adoption should be OK
  - There already have been some ☺

Regular DT: /cpus

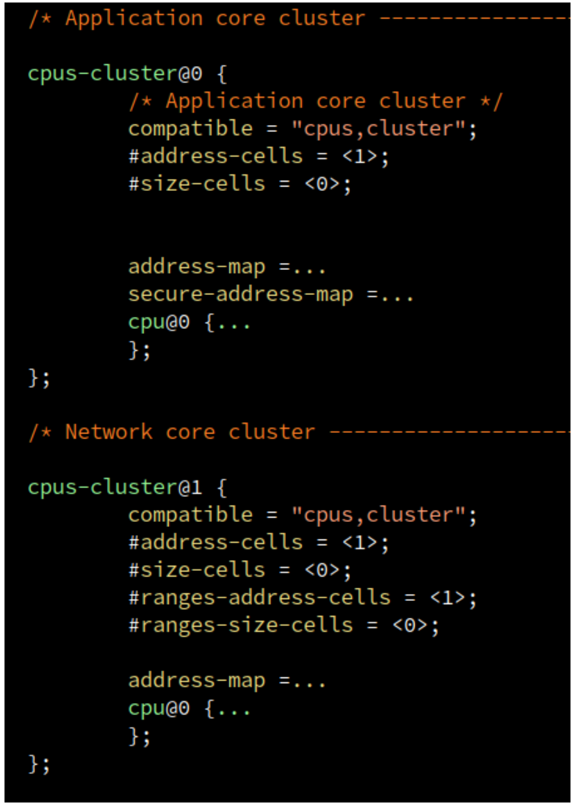


```
cpus {
        #address-cells = <1>;
        #size-cells = <0>;

        cpu@0 {
                device_type = "cpu";
                compatible = "arm,cortex-m33f";
                reg = <0>;
                #address-cells = <1>;
                #size-cells = <1>;

                itm: itm@e0000000 {
                        compatible = "arm,armv8m-itm";
                        reg = <0xe0000000 0x1000>;
                        swo-ref-frequency = <64000000>;
                };

                mpu: mpu@e000ed90 {
                        compatible = "arm,armv8m-mpu";
                        reg = <0xe000ed90 0x40>;
                };
        };
};
```

/cpus:

- Address space size (32 vs 64 bit)
- Number of cores, potentially SMP
- Private peripherals on each core

System DT: /cpus-cluster@N



```
/* Application core cluster ---------------

cpus-cluster@0 {
        /* Application core cluster */
        compatible = "cpus,cluster";
        #address-cells = <1>;
        #size-cells = <0>;


        address-map =...
        secure-address-map =...
        cpu@0 {...
        };
};

/* Network core cluster -------------------

cpus-cluster@1 {
        compatible = "cpus,cluster";
        #address-cells = <1>;
        #size-cells = <0>;
        #ranges-address-cells = <1>;
        #ranges-size-cells = <0>;

        address-map =...
        cpu@0 {...
        };
};
```

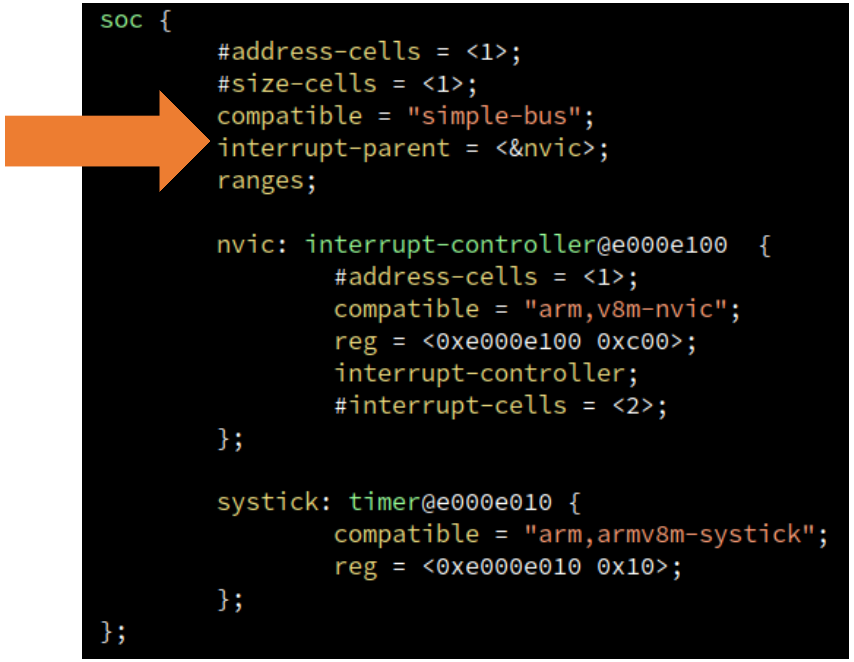
/cpus-cluster@N:

- Like /cpus, but one per "cluster"
- **address-map, secure-address-map**

Other notes:

- nRF5340 has two "clusters": app, net

Regular DT: simple-bus

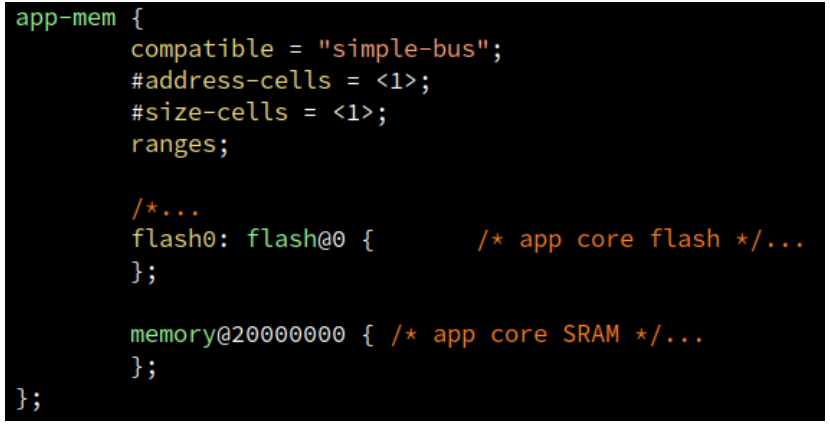


```
soc {
        #address-cells = <1>;
        #size-cells = <1>;
        compatible = "simple-bus";
        interrupt-parent = <&nvic>;
        ranges;

        nvic: interrupt-controller@e000e100  {
                #address-cells = <1>;
                compatible = "arm,v8m-nvic";
                reg = <0xe000e100 0xc00>;
                interrupt-controller;
                #interrupt-cells = <2>;
        };

        systick: timer@e000e010 {
                compatible = "arm,armv8m-systick";
                reg = <0xe000e010 0x10>;
        };
};
```

/soc

- Name is just a convention!
- "simple-bus": DT spec definition
- interrupt-parent: simple interrupt tree
- ranges: simple addressing

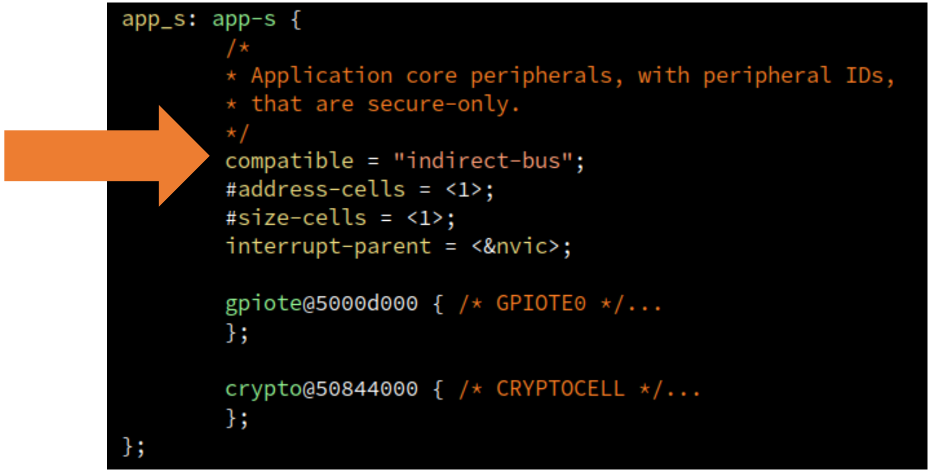
System DT: simple-bus still works



```
app-mem {
        compatible = "simple-bus";
        #address-cells = <1>;
        #size-cells = <1>;
        ranges;

        /*...
        flash0: flash@0 {       /* app core flash */...
        };

        memory@20000000 { /* app core SRAM */...
        };
};
```

- "simple-bus" still available in system DT
- Example: application core memory is visible to all CPU clusters, with trivial addressing

System DT: indirect-bus



```
app_s: app-s {
        /*
        * Application core peripherals, with peripheral IDs,
        * that are secure-only.
        */
        compatible = "indirect-bus";
        #address-cells = <1>;
        #size-cells = <1>;
        interrupt-parent = <&nvic>;

        gpiote@5000d000 { /* GPIOTE0 */...
        };

        crypto@50844000 { /* CRYPTOCELL */...
        };
};
```

indirect-bus:

- Contains HW not visible to all clusters
- Otherwise similar to simple-bus
- Addressability defined via "address-map" type properties in CPU clusters

System DT: address map usage

```
/* Application core cluster ---------------------------------------- */

cpus-cluster@0 {
        /* Application core cluster */
        compatible = "cpus,cluster";
        #address-cells = <1>;
        #size-cells = <0>;


        address-map =
                <0x0 &app_core_components 0x0 0x01000000>,
                <0xe0000000 &app_ppb 0xe0000000 0xffffffff>,
                <0x40000000 &app_ns 0x40000000 0x01000000>,
                <0x40000000 &app_us 0x0 0x01000000>;

        secure-address-map =
                <0x0 &app_core_components 0x0 0x01000000>,
                <0xe0000000 &app_ppb 0xe0000000 0xffffffff>,
                <0x50000000 &app_s 0x50000000 0x01000000>,
                <0x40000000 &app_ns 0x40000000 0x01000000>,
                <0x50000000 &app_us 0x0 0x01000000>;

        cpu@0 {
                #address-cells = <1>;
                #size-cells = <1>;
                device_type = "cpu";
                compatible = "arm,cortex-m33f";
                reg = <0>;

                mpu@e000ed90 {
                        compatible = "arm,armv8m-mpu";
                        reg = <0xe000ed90 0x40>;
                        arm,num-mpu-regions = <8>;
                };
        };
};
```

- Specify which indirect-buses are visible to each CPU cluster using address-map
- For Armv8-M, use secure-address-map for the secure view of the world
- <0xINDIRECT_BUS_BASE &indirect_bus 0xCPU_CLUSTER_BASE 0xSIZE>;
- Fixed at SoC level, defines all “possible” visible hardware

System DT → DT process

1. Pick a CPU cluster and address map property
2. Rewrite addresses in any indirect-bus nodes reachable from address-map
3. Change reachable indirect-bus nodes to simple-bus nodes
4. Delete unreachable indirect-bus nodes
5. Rename chosen CPU cluster to /cpus
6. Delete other CPU clusters