Linaro connect
Vancouver 2018

BoF: DeviceTree

Rob Herring
Linus Walleij

Agenda

- json-schema for bindings
- Connector bindings

Linaro connect Vancouver 2018

BoF: DT: json-schema for bindings

Rob Herring

Devicetree Schema Documentation and Validation

The problem: too easy to get devicetree wrong

- Data must be encoded in very specific ways
- Toolchain provides little validation
- No checks against documented schema (aka. bindings)
    - Schemas are loosely structure prose
    - Not machine readable
- Too much manual review of bindings and dts
- Steep learning curve

Project Goals

- Define a DT schema language
  - Human friendly
  - Machine readable
  - Include binding documentation
- Better tooling
  - Validate DTS files at build time
  - Validate DT Schema files are in the correct format
  - Useful error and warning messages
- Leverage existing technology
  - Use existing schema validation framework
    - Extended to handle quirks of DT
  - Don’t write a lot of code!
  - Don’t define an entirely new language!
- Generate Specification Documentation from Schema files

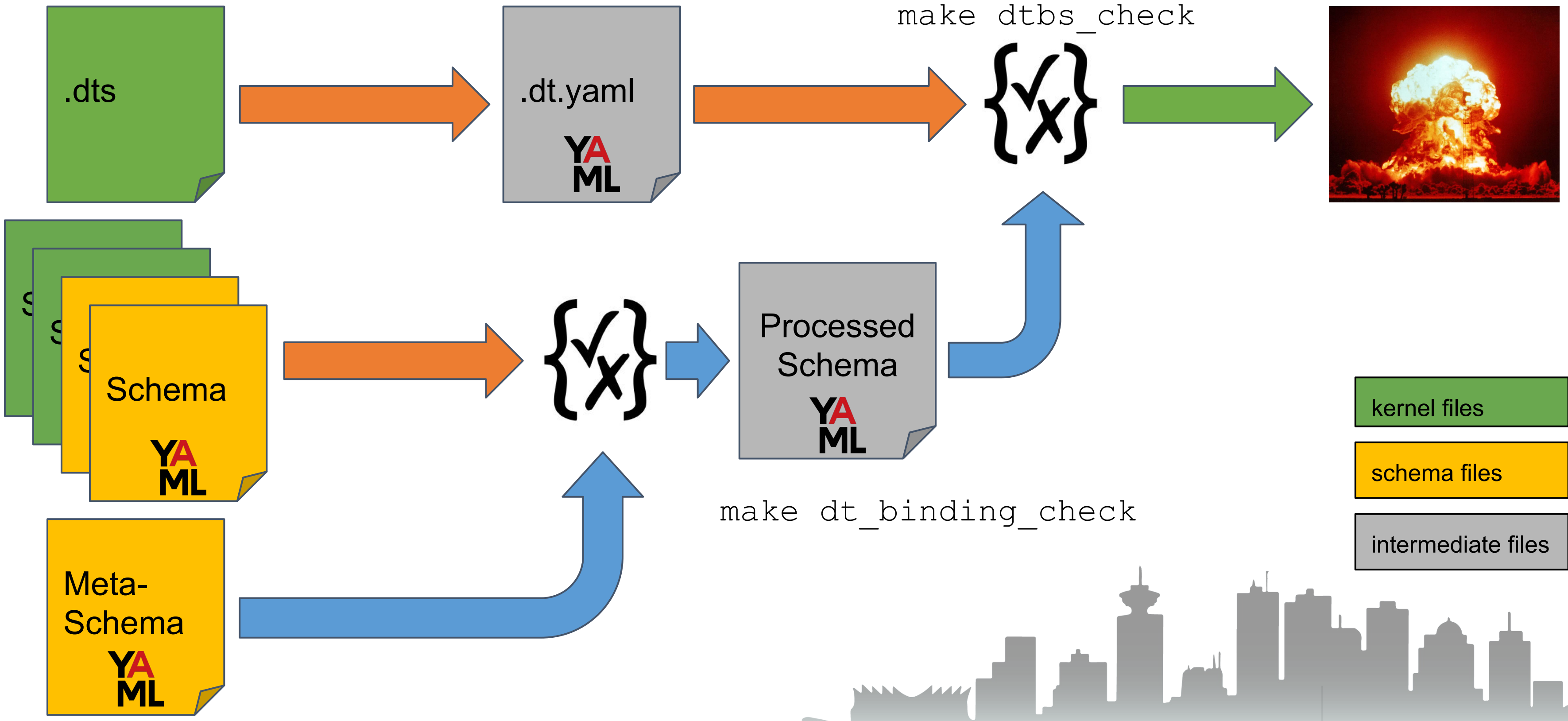
Recap from HKG18

- Session: https://connect.linaro.org/resources/hkg18/hkg18-120/
- Initial project: https://github.com/robherring/yaml-bindings
- Using json-schema for schema vocabulary
- Schema docs in YAML format (a JSON compatible subset)
- Tools written in Python 3 utilizing ruamel.yaml and jsonschema modules

Current Status/Features

- schema/tools one step install from pip (Works for Me™)
- YAML output format support in upstream dtc
- Kernel build integration
  - schema doc validation
  - dts validation with schema - core schema and some ARM board level binding schema
  - Binding examples extracted from schema and built with dtc for validation
- Validation support for size (8-bit, 16-bit, etc.) and phandle tags
- Build time performance is reasonable now, but don't have 1000s of binding schemas yet

DT Schema Flow



🟩 kernel files

🟨 schema files

⬜ intermediate files

Schema doc contents

- $id - URI with json-schema unique identifier (within a set of schemas)
- $schema - URI for meta-schema the schema adheres to
- title - A one-line description for the binding
- description - A multi-line description for the binding
- maintainers - List of email addresses for owner(s) of the binding
- select - Schema to match DT nodes. Only needed when not matching by compatible or node name.
- properties/patternProperties - dictionary of DT properties for a binding
- required - List of mandatory properties
- examples - List of examples in dts syntax

properties schema

- The part used in validation of DTs
- Contains list of DT properties for a binding
- Can also be child nodes with their own DT properties
- A subset of json-schema is allowed and checked by the meta-schema
- Common properties only need to define binding specific constraints (e.g. number of items, valid values, etc.)
- Vendor specific properties need to reference base type

Common Property Examples

```
clock-frequency:
  minimum: 100
  maximum: 200

reg:
  items:
    - description: the first register range
    - description: the 2nd register range
```

Vendor Property Examples

```
vendor,uint32-prop:
  allOf:
    - $ref: "/schemas/types.yaml#/definitions/uint32"
    - minimum: 100
      maximum: 200
      description: A vendor uint32 property
vendor,string-prop:
  allOf:
    - $ref: "/schemas/types.yaml#/definitions/string"
    - enum: [ foo, bar, baz ]
      description: A property with meaningless strings
```

Gotchas

- YAML is indentation sensitive and doesn’t like tabs
- json-schema keywords are case sensitive
- Validator handling of unknown json-schema keywords is to ignore
- Constraints dependent on other properties can’t be expressed (e.g. 2 interrupts if compatible A or 1 interrupt if compatible B)
- Using allOf/oneOf/anyOf results in vague error messages
- Only one binding per doc
-

Running

- pip3 install git+https://github.com/robherring/yaml-bindings.git@master \
  --process-dependency-links
- Install libyaml and its headers
- make allmodconfig
- make dt_binding_check
- make ~~DTC=/path/to/yaml-enabled/dtc~~ dtbs_check

Demo

Next Steps

- dtc update and dtb building rework pending for v4.20
- What to do with yaml-binding project?
  - Keep separate
  - Integrate a subset into kernel
- Convert bindings - Help wanted!

Resources

- Schema/Tools repo
  - https://github.com/robherring/yaml-bindings
- kernel repo with DT schema
  - https://github.com/robherring/linux/tree/yaml-bindings
-

Linaro connect
Vancouver 2018

BoF: DT: connector bindings

Linus Walleij

Connector Bindings

- Problem space: non-discoverable daughterboards on random electronic connectors
- The same daughterboards are used with ACPI-based systems with the same connectors
- How do we enable users to use their daughterboards?
- Example: 96Boards Secure96 with TPM chip for all DT-based 96Boards but also desired for the ACPI-based developer box
- If the whole world is DT, overlays and connector nexi solves this in theory
- FWNODE a common point between DT and ACPI?