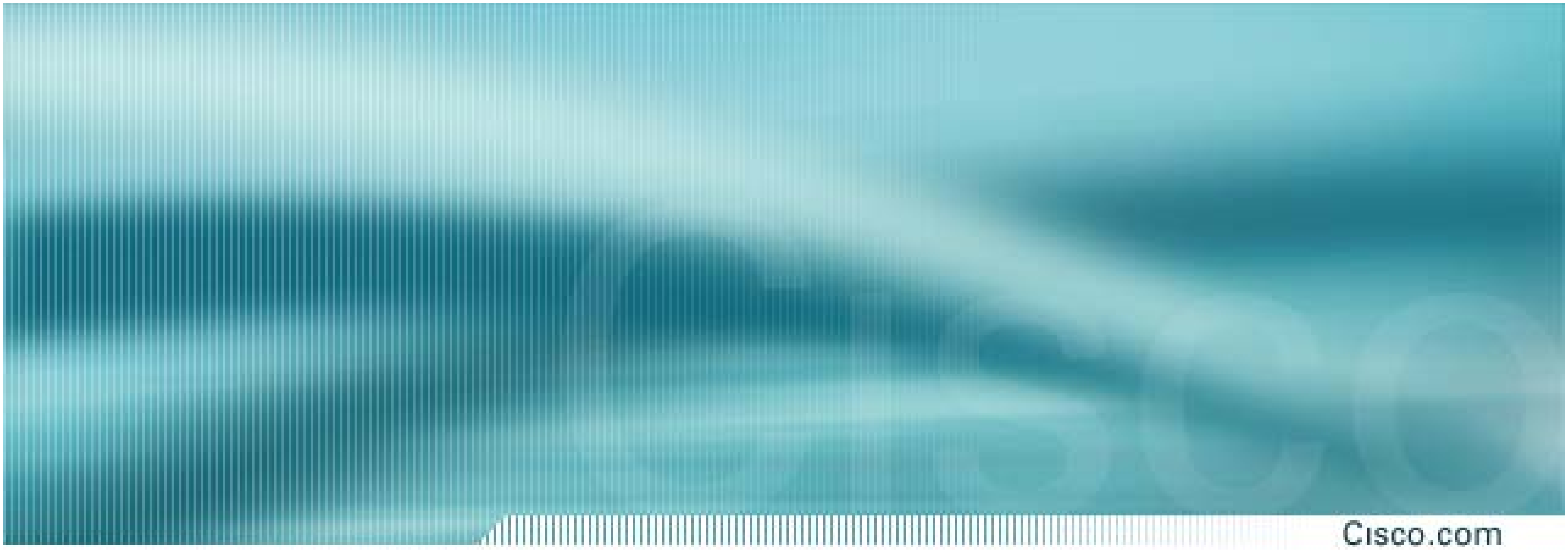
Cisco Systems



*csco_config e*VC Overview

Joseph H. Zhang / Mark Strickland

March 2006

Agenda

- **Benefits**
- **Theory of Operation**
- **Usage Model**
- **Legacy Support**

The Requirements

- Provide all three of the following
  - Have a config struct that contains all the control information (besides individual data items) for an eVC
  - Allow constraints in the config struct of a higher level unit to constrain values in the config struct of a lower level unit
  - Allow re-generation of all the configs in a branch of the unit tree during the run

The Problem

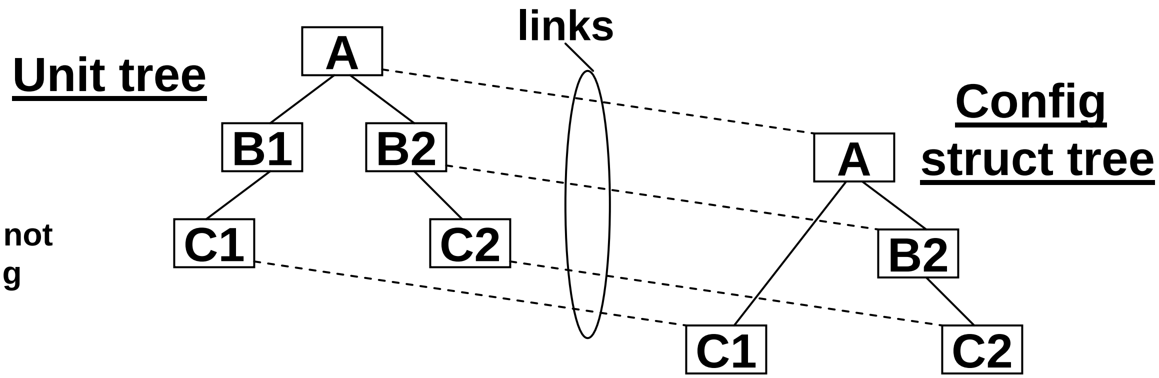
- **Specman generator uses only constraints found in or below the struct/unit being generated**

```
unit A_u {
  config : A_config_s ;
  B : B_u is instance ;
  keep B.config.x < config.x ;
};
struct A_config_s {
  x : byte ;
};
unit B_u {
  config : B_config_s ;
  reset_unit() is also {
    gen config ;
  };
};
struct B_config_s {
  x : byte ;
};
```

This constraint will not be applied …

… if config is re-generated at this level

Separate Config Tree From Unit Tree

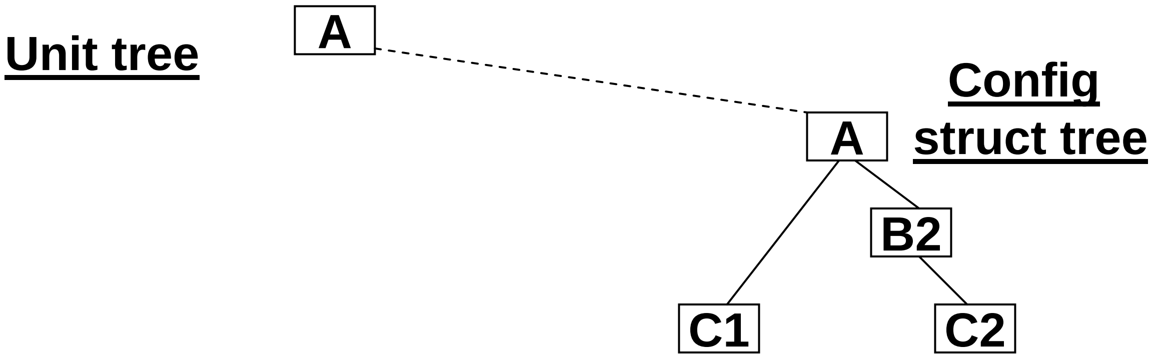


assume B1 does not
require a config

- Macros for specifying definition and hierarchy
- Links updated on reset/regeneration

Tree in Pre-run Generation

Benefits	Theory	Usage	Legacy



Unit tree

A

Config struct tree

A

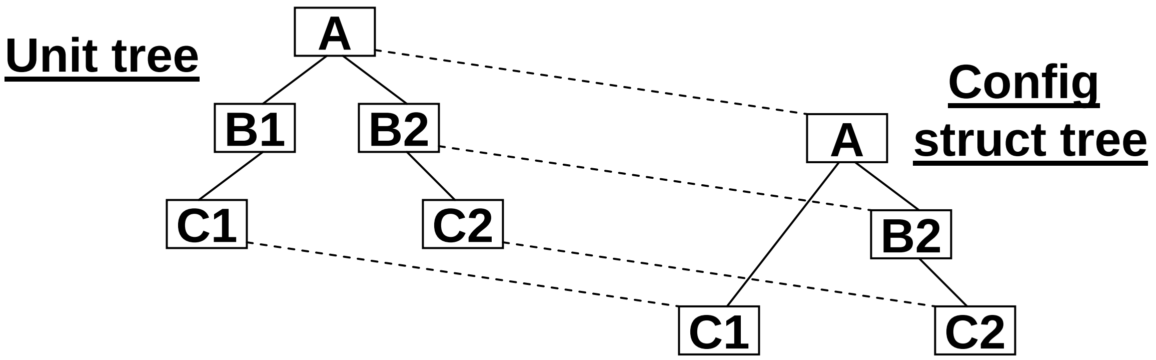
B2

C1

C2

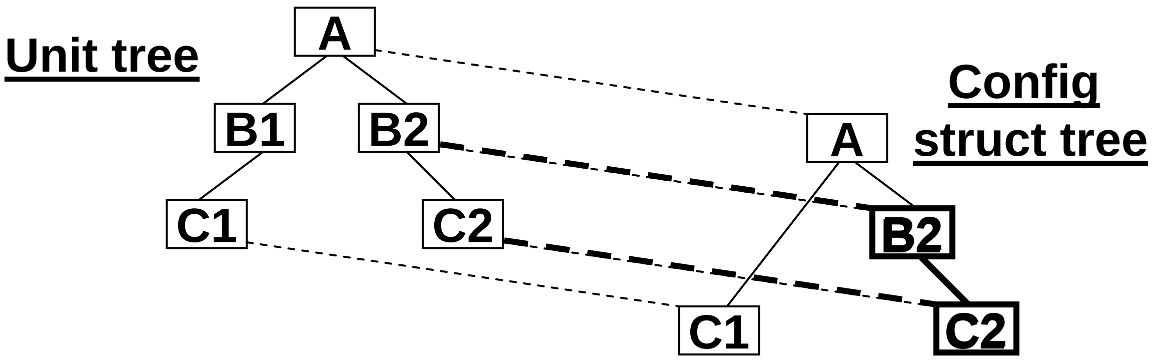
1. **Entire config tree starting with A is generated when unit A is generated.**

Tree in Pre-run Generation



2. As each unit in the unit tree is generated, it establishes the link to its config struct

Branch Being Regenerated



- **If B2 config is regenerated, its leaf C2 is also regenerated and then their links are updated**

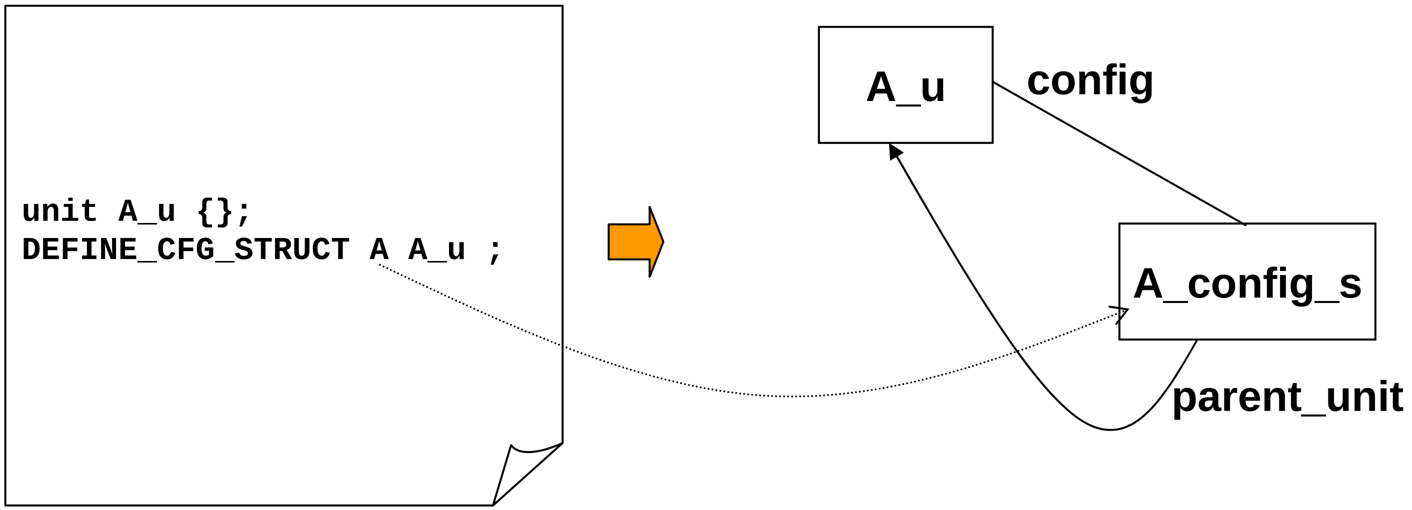
Usage

- Description organized by typical user
  - eVC creator who is developing the units that make up a component
  - testbench integrator who is building an environment by combining the components
  - testcase writer who is adding direction to an existing environment

DEFINE_CFG_STRUCT

Benefits	Theory	Usage	Legacy
	eVC Creator	TB Integrator	Testcase Writer

- **See Word doc for details**



```
unit A_u {};
DEFINE_CFG_STRUCT A A_u ;
```

Constraints Based on Config Fields

Benefits	Theory	Usage	Legacy
	eVC Creator	TB Integrator	Testcase Writer

- Constrain unit fields based on config fields
  - keep unit_field < read_only(config_field);
- These constraints go in the unit, not in the config struct
- In general, do not duplicate config information in the unit

Making Copies of Config Fields

Benefits	Theory	Usage	Legacy
	eVC Creator	TB Integrator	Testcase Writer

- Copy the config field in the unit if that field is necessary for subtyping the unit

```
unit A_u {};
DEFINE_CFG_STRUCT A A_u ;

type A_color_t : [RED, BLUE] ;
extend A_config_s {
  color : A_color_t ;
};
extend A_u {
  !color : A_color_t ;
  assign_cfg_shadow_values() is also {
    color = config.color ;
  };
};
extend RED A_u ...
```

Add subtyping field to config

Also add field to unit with "!" and extend assign_cfg_shadow_values() to assign value from the config field

Methodology – Config Mode

Benefits	Theory	Usage	Legacy
	eVC Creator	TB Integrator	Testcase Writer

Macro creates

```
type A_config_mode_t : [] ;
extend A_config_s {
  config_mode : A_config_mode_t ;
};
```

Suggested usage example

```
extend A_config_mode_t : [MODE1] ;
extend MODE1 A_config_s {
  keep x in [1..4] ;
  keep y in [6..9] ;
};
```

- **This is a suggestion only – the package does not require or enforce the use of the config mode**

Methodology – Mode Type

Benefits	Theory	Usage	Legacy
	eVC Creator	TB Integrator	Testcase Writer

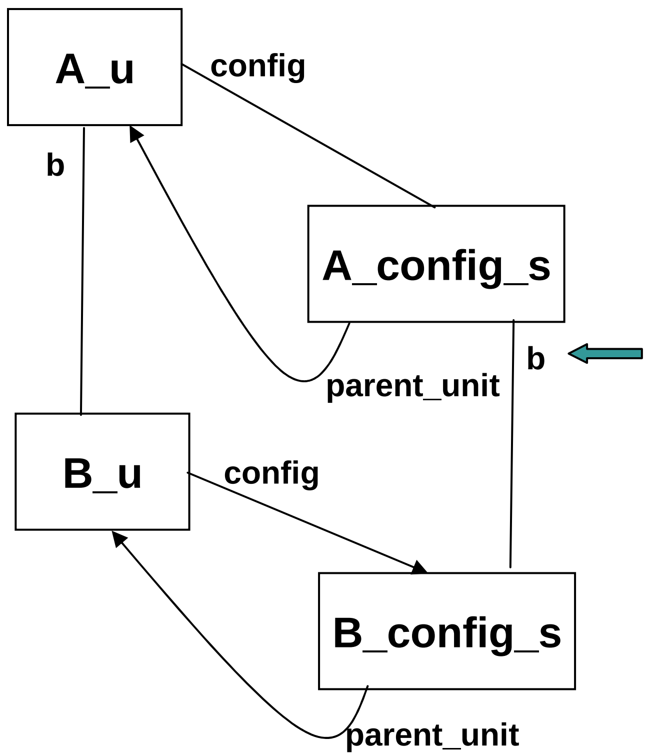
- **The eVC creator may want to create additional similar fields to represent orthogonal configuration patterns**

```
type A_config_mode_b_t : [BMODE1] ;
extend A_config_s {
  config_mode_b : A_config_mode_b_t ;
};
extend BMODE1 A_config_s {
  keep a < b ;
};
```

INSTANCE_CFG

- **See Word doc for details**

```
unit B_u {};
DEFINE_CFG_STRUCT B B_u ;
unit A_u {b : B_u is instance;};
DEFINE_CFG_STRUCT A A_u ;
INSTANCE_CFG begin
  config_inst_name = b,
  child_config_type = B_config_s,
  parent_config_type = A_config_s,
  parent_unit_type = A_u,
  child_unit_type = B_u
end ;
```

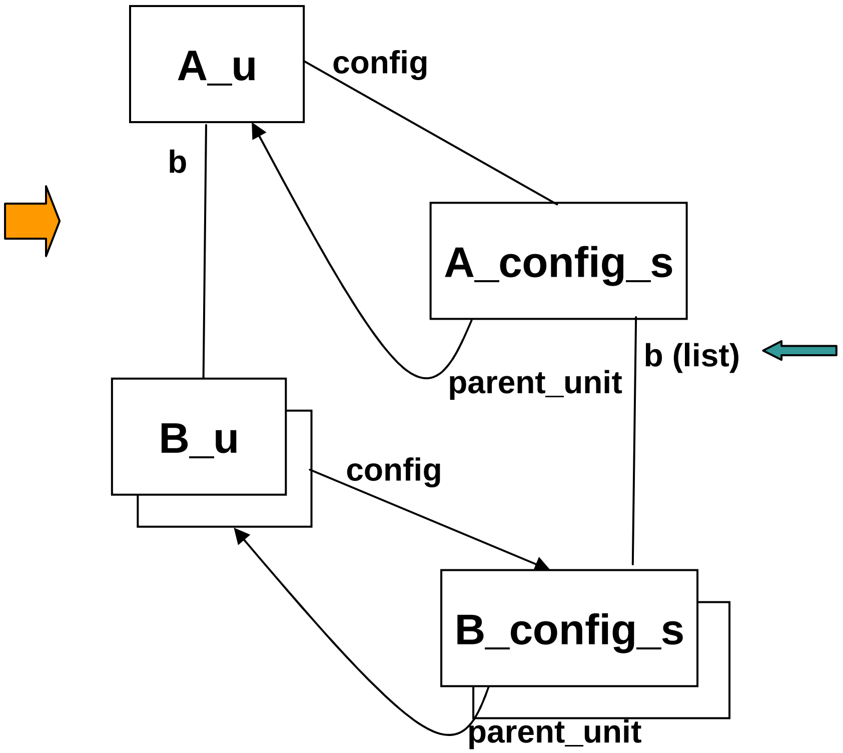


INSTANCE_CFG_LIST

Benefits	Theory	Usage	Legacy
	eVC Creator	TB Integrator	Testcase Writer

- **See Word doc for details**

```
unit B_u {};
DEFINE_CFG_STRUCT B B_u ;
unit A_u {b : list of B_u is instance;};
DEFINE_CFG_STRUCT A A_u ;
INSTANCE_CFG_LIST begin
  config_inst_name = b,
  child_config_type = B_config_s,
  parent_config_type = A_config_s,
  parent_unit_type = A_u,
  child_unit_type = B_u
end ;
```

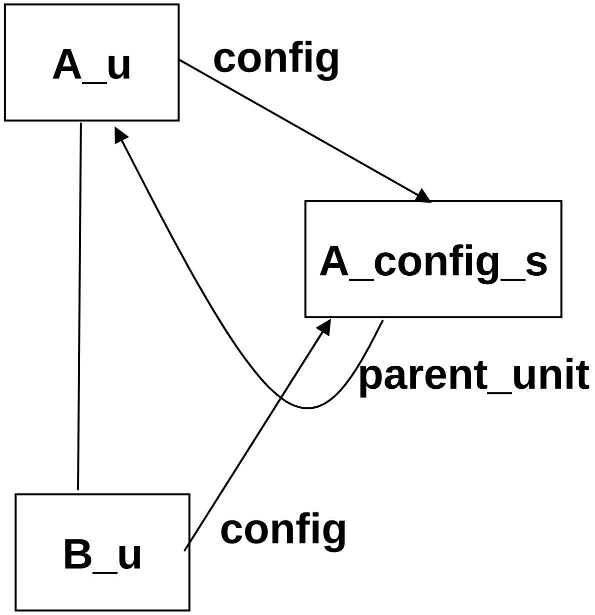


USE_PARENT_CFG

Benefits	Theory	Usage	Legacy
	eVC Creator	TB Integrator	Testcase Writer

- **See Word doc for details**

```
unit B_u {};
DEFINE_CFG_STRUCT B B_u ;
unit A_u {b : B_u is instance;};
DEFINE_CFG_STRUCT A A_u ;
USE_PARENT_CFG begin
  parent_config_type = A_config_s,
  parent_unit_type = A_u,
  child_unit_type = B_u,
  child_unit_path = b
end ;
```

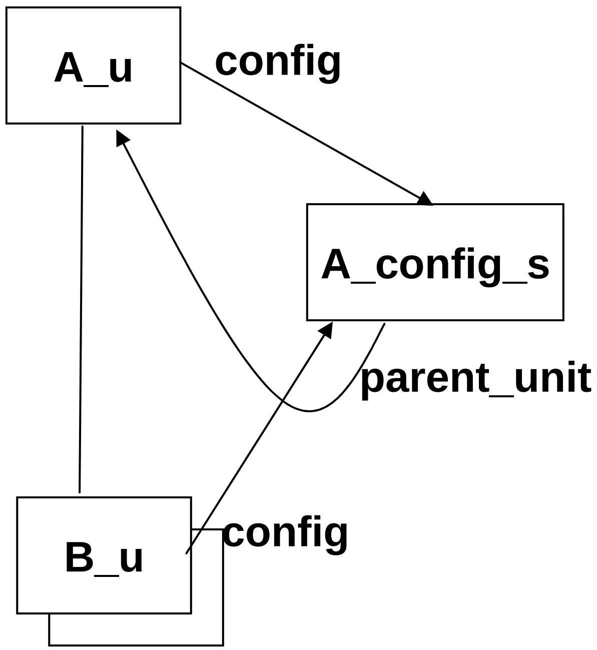


USE_PARENT_CFG_LIST

Benefits	Theory	Usage	Legacy
	eVC Creator	TB Integrator	Testcase Writer

- See Word doc for details

```
unit B_u {};
DEFINE_CFG_STRUCT B B_u ;
unit A_u {b : list of B_u is instance;};
DEFINE_CFG_STRUCT A A_u ;
USE_PARENT_CFG_LIST begin
  parent_config_type = A_config_s,
  parent_unit_type = A_u,
  child_unit_type = B_u,
  child_unit_path = b
end ;
```



Hierarchical Config Constraints

Benefits	Theory	Usage	Legacy
	eVC Creator	TB Integrator	Testcase Writer

- **Constraints relating values in a lower level of the config tree to those in a higher level should always be in the higher level config struct**
  - **Do not constrain config values from the unit tree**

Instancing Subtyped Units

- **For INSTANCE_CFG make sure subtypes are consistent**

```
extend HOT A_u {
  b : BIG B_u;
};
INSTANCE_CFG begin
  config_inst_name = b,
  child_config_type = BIG B_config_s,
  parent_config_type = HOT A_config_s,
  parent_unit_type = HOT A_u,
  child_unit_type = BIG B_u
end ;
```

Because unit is instanced as BIG subtype…

…must have the BIG subtype used for the child_config_type

Instancing Subtyped Units

Benefits	Theory	Usage	Legacy
	eVC Creator	TB Integrator	Testcase Writer

- For USE_PARENT_CFG, constrain config field that defines subtype

```
extend HOT A_u {
  b : BIG B_u;
};
USE_PARENT_CFG begin
  parent_config_type = HOT A_config_s,
  parent_unit_type = HOT A_u,
  child_unit_type = BIG B_u,
  child_unit_path = b
end;
extend HOT A_config_s {
  keep B_size == BIG ;
};
```

Because unit is instanced as BIG subtype…

…must constrain the config field for B_size to BIG

REGEN_CFG_BRANCH

Benefits	Theory	Usage	Legacy
	eVC Creator	TB Integrator	Testcase Writer

- See the Word doc for details
- REGEN_CFG_BRANCH sys.A.config.b
  - To regenerate a branch of the config tree, must pass a full path from sys to the macro

    Reason: Specman generator will not use all of the hierarchical constraints unless all the units that contain them are in the gen context

REGEN_CFG_BRANCH

Benefits	Theory	Usage	Legacy
	eVC Creator	TB Integrator	Testcase Writer

- The full path must contain the root of the config tree
  - Do not use the path through the unit tree to the unit that contains the config to be regenerated

    Reason: The config reference in the unit is constrained to point to the existing config tree, so it will not change under regen.

REGEN_CFG_BRANCH Placement

Benefits	Theory	Usage	Legacy
	eVC Creator	TB Integrator	Testcase Writer

- Testbench integrator
  - may call REGEN_CFG_BRANCH in a method triggered by reset_unit() (in a tb-specific, non-reusable file)
- Testcase writer
  - may call REGEN_CFG_BRANCH in the body of a test that does OIR

Test Writing Methodology

Benefits	Theory	Usage	Legacy
	eVC Creator	TB Integrator	Testcase Writer

- **constrain everything besides sequence data in config struct**
  - **extend XXX a_config_s { keep yyy == 5 };**
- **use config_mode whenever possible**
  - **extend XXX a_config_s {**
    keep config_mode in [MODEA, MODEB];
    };

Legacy Support

- No restrictions on legacy code co-existing with code using csco_config.
- Recommendations if OIR (regeneration) required for legacy code

  1. Edit legacy code, move config fields into csco_config structs
  2. Create a csco_config struct external to the legacy units and copy the config fields into this new struct

Cisco Systems®

Empowering the Internet Generation