Description File

A soclib description file is a python-parseable file, ending in ".sd".

In fact, it is not so pythonic, using python syntax is just to avoid to write a new parser from scratch and yet having a human-readeable file.

It's currently used by two tools:

- Soclib-CC, for compiling a platform with automatic template instanciation
- DSX, for complete netlist generation

It contains:

- Source-file references
- C++ class template instanciation information
- C++ class instanciation information
- Netlist information
- DSX extensions (if needed)

Later, when IP-Xact (Spirit) packaging will be ready, there will be a conversion tool from this format to IP-Xact.

Module

Arguments to those declarations are:

name (mandatory)
: the reference name from anywhere else in Soclib Cc. This name is preferably of the form `'type:name'` where type may be:
    ◊ caba
    ◊ common
    ◊ tlmt

classname (mandatory)
: the C++ class name to instanciate, with full namespace, but no template parameters

tmpl_parameters
: the template parameters, if any. This must be a list of parameters (see below)

header_files
: header files to include when using this component, paths relative to the sd

implementation_files

implementation files to compile, paths relative to the sd

ports
: ports in the module

instance_parameters
: needed parameters to instanciate the module. This must be a list of parameters (see below)

Signal

Arguments are:

name, classname, tmpl_parameters, header_files, implementation_files
: as above

accepts
: a dictionnary of port names associated to maximum connection count.

sample: Vci signal description?

PortDecl

Arguments are:

name, classname, tmpl_parameters, header_files, implementation_files
: as above

signal
: connectable signal name (in soclib-cc)

sample: Vci ports description?

Parameters

Instance Parameters

In any parameter, you may specify a default value using `default=`

- `parameter.IntTab`
  - arguments: name
  - usage: an IntTab for Vci indexes
  - example: `parameter.IntTab('index')`
  - sample: vci_ram.sd, line 23?
- `parameter.Module`
  - arguments: name, typename = module type (name declared to soclib-cc)
  - usage: passing another module reference, like a mapping table or another component
  - example: `parameter.Module('mt', typename = 'common:mapping_table')`
  - sample: vci_vgmn.sd, line 18?, vci_ram.sd, line 25?
- `parameter.String`
  - arguments: name
  - example: `parameter.String('input_file', default = 'input.txt')`
  - sample: fifo_reader.sd, line 20?
- `parameter.Int`
  - arguments: name
  - example: `parameter.Int('min_latency', default = 3)`

        - sample: vci_vgmn.sd, line 19?
- `parameter.Bool`
        - arguments: name
        - example: `parameter.Bool('use_atomic_operations', default = False)`
- `parameter.StringArray`
        - arguments: name
        - example: `parameter.StringArray('tty_names')`
        - sample: vci_multi_tty.sd, line 28?

Template parameters

- `parameter.Module`
        - arguments: name
        - example: `parameter.Module('iss_t')`
        - sample: iss_wrapper.sd, line 11?
- `parameter.Bool`
        - arguments: name
        - example: `parameter.Bool('use_atomic_operations', default = False)`
        - sample: vci_target_fsm.sd, line 11?
- `parameter.Type`
        - arguments: name
        - example: `parameter.Type('word_t')`
        - sample: fifo_reader.sd, line 24?

Common warnings

ModuleDeprecationWarning: Deprecation

Some modules may be marked as deprecated with a friendly warning until they are removed. You'll get this kind of message when using such a module:

```
ModuleDeprecationWarning at /Users/nipo/projects/soclib/soclib/lib/mips/metadata/mipsel.sd:10:
    Module common:mips deprecated: "Please migrate to Mips32"
```

The message enclosed in quotes at the end is the one specified in the mips.sd file?.

PartialNameWarning : Lazyness in declaration

```
PartialNameWarning at /Users/nipo/projects/soclib/utils/lib/python/soclib_desc/specialization.py
    Short name vci_timer is deprecated, please use a full name with caba:, tlmt: or common:
```

Modules have dependencies on other modules. Dependencies can be:

- Ports, example:

```
Port('caba:vci_initiator', 'p_vci')
```

- Uses, i.e. uses of another module as sub-module. Example:

```
Uses('caba:vci_target_fsm')
```

- Module parameters, example:

```
tmpl_parameters = [
       parameter.Module('vci_param',  default = 'caba:vci_param'),
```

```
            ],
```

When such construct are used, you should write the full name of the module. Here, it is `caba:vci_initiator`, `caba:vci_target_fsm`, `caba:vci_param`.

Previously, soclib-cc supported to implicitly use the abstraction level from the parent module, but this has limitations with multi-abstraction level simulations, therefore it is deprecated.

InvalidComponentWarning: Invalid Definition

```
InvalidComponentWarning at /Users/nipo/projects/soclib/soclib/module/verification_component/avci
    Invalid component caba:avci_filter, it will be unavailable. Error: "NoSuchComponent(`vci_pa
```

This is most probably a syntax or usage error. As parsing of the `.sd` file cant be done, module is disabled.