1. A) Interconnects
    1. A.1) Vgmn
    2. A.2) LocalCrossbar
2. B) VCI Initiators
    1. B.1) Xcache
3. C) Processors
    1. C.1) Mips
4. D) VCI Targets
    1. D.1) MultiRam
    2. D.2) MultiTty
    3. D.3) Locks

A) Interconnects

A.1) Vgmn

- functionality : a generic VCI compliant micro-network
- Mandatory arguments:
    - `instance name`
- Optional arguments:
    - `min_latency` (one-way)
- Example:

```
my_vgmn = Vgmn("my_vgmn", 10)
```

- Defined ports:
    - `getBoth()`, `getInit()` and `getTarget()`: local ports, allocated on demand

A.2) LocalCrossbar

- functionality : a VCI compliant crossbar interconnect
- Mandatory arguments:
    - `instance name`
- Example:

```
my_lc = LocalCrossbar("lc0")
```

- Defined ports:
    - `getBoth()`, `getInit()` and `getTarget()`: local ports, allocated on demand
    - `upstream`, bidirectional port to upper-level interconnect

B) VCI Initiators

B.1) Xcache

- functionality: a direct mapping cache controler (separated instruction & data cache)
- Mandatory arguments:
    - `instance name`
- Optional arguments:
    - `dcache_lines` : number of lines in data cache
    - `dcache_words` : number of words per line in data cache
    - `icache_lines` : number of lines in instruction cache

- ♦ `icache_words` : number of words per line in instruction cache

- Example:

```
my_cache = Xcache( "my_cache",
                   dcache_lines = 32,
                   dcache_words = 8,
                   icache_lines = 32,
                   icache_words = 8 )
```

- Defined ports:
  - ♦ `cache`: to the CPU
  - ♦ `vci`: to the VCI micro-network

C) Processors

C.1) Mips

- Functionality : a MIPS R3000 micro-processor
- Mandatory arguments:
  - ♦ `name`
- Example:

```
my_proc = Mips("my_proc")
```

- Defined ports:
  - ♦ `cache`: to the cache's cache port
  - ♦ `irq[n]`: interrupt line (0 <= n < 6)

D) VCI Targets

D.1) MultiRam

- !Mandatory arguments:
  - ♦ `name`
- Optional arguments:
  - ♦ a list of segments, allocated with Segment()
- Example:

```
my_ram = MultiRam("my_ram", seg1, seg2, seg3)
```

- Defined ports:
  - ♦ `vci`: to the micro-network

D.2) MultiTty

- functionality: a TTY controler (up to 256 TTYs)
- Mandatory arguments:
  - ♦ `instance name`
  - ♦ an ordered list ot names (one name per emulated terminal)
- Example:

```
my_tty = MultiTty("my_tty_controler", "TTY0", "TTY1", "TT2")
```

- Defined ports:
  - ♦ `vci`: to the micro-network
  - ♦ `irq[n]`: interrupt line (0 <= n < nb of ttys)

D.3) Locks

- functionality : a locks controler
- Mandatory arguments:
  - `instance name`
- Example:

```
my_locks = Locks("my_locks_controler")
```

- Defined ports:
  - `vci`: to the micro-network