`Vgmn`

◊ functionality : a generic VCI compliant micro-network
◊ Mandatory arguments:
· `instance name`
◊ Optional arguments:
· `min_latency`
◊ Example:

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

`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 )
```

`Mips`

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

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

`MultiRam`

◊ Mandatory arguments:
· `name`
◊ Optional arguments:
· a list of segments, allocated with Segment()
◊ Example:

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

`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")
```

`Locks`

◊ functionality : a locks controler

◊ Mandatory arguments:

- `instance name`

◊ Example:

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

◊ See RamLocks