There may be some different mapping possibilities.

Mapping directive always takes at least one argument: the object to map. Following arguments are keyword arguments (i.e. they are named) that may be specified in any order. Their meaning is tied to what kind of object you are mapping, and how you are mapping it.

Some mapping arguments may be optional.

Let's explore mappable objects, and how they may be deployed.

Tasks

as Software Task

- Mandatory arguments:
    - `run`: A Cpu to run task on
    - `stack`: A memory Segment to put stack in. Stack does not grow dynamically. This should be a segment of cacheable memory for performance reasons.
    - `desc`: A memory Segment to put the task's description in. This description won't be written to. Segment may be read-only memory.
- Optional arguments:
    - `tty` and `tty_no`: name of a segment associated to a multi_tty and the tty number in the component. They should both be defined at the same time
- Example:

```
mapper.map( "cons0",
            run   = mapper.hard.processor,
            stack = mapper.hard.cram0,
            desc  = mapper.hard.cram0)
```

as Coprocessor

A `SyntheticTask()` or `MwmrCoproc()` implementation must exist for the task in order to be able to map it as a coprocessor.

You must have a netlist with a coprocessor instanciated (see MwmrCoprocCreation) in it. Then you have to map the task on this coprocessor:

- Mandatory arguments:
    - `coprocessor`: Name of the coprocessor component.
    - `controller`: Name of the (mwmr) controller component.
- Example:

```
mapper.map( "idct0",
            coprocessor = "fifo_idct0",
            controller = "fifo_idct0_ctrl")
```

Deprecated method:

- Mandatory arguments:
  - ♦ `vci`: A VCI interconnect to connect MwMr controller to.
  - ♦ `address`: Address to map MwMr controller at.
- Example:

```
mapper.map( "cons0",
            vci = mapper.hard.vgmn,
            address = 0x71200000 )
```

Memspaces

- Mandatory arguments:
  - ♦ `buffer`: A memory Segment to put data in. This may be cacheable.
- Example:

```
mapper.map( "memsp",
            buffer = mapper.hard.cram1)
```

Locks

- Mandatory arguments:
  - ♦ `segment`: A memory Segment to put the lock in.
- Example:

```
mapper.map( "lock0",
            segment = mapper.hard.cram1)
```

MwMr channels

- Mandatory arguments:
  - ♦ `buffer`: A memory Segment to put the data buffer in. This may be cacheable.
  - ♦ `status`: A memory Segment to put dynamic status and buffer in. This must be cacheable. Software Mwmr protocol implementations do explicit cachelines flushing.
- Optional argument:
  - ♦ `lock`: A Locks dedicated hardware component, like SocLib's Locks component. Uses one lock.
- Example:

```
mapper.map( "fifo",
            lock    = mapper.hard.locks,
            status  = mapper.hard.cram1,
            buffer  = mapper.hard.cram1)
```

Processor

- Mandatory arguments:
  - ♦ `private`: A memory Segment to put the processor's private data in. This should be cacheable.

    - `shared`: A memory Segment to put shared status in. This must be uncached as it is used to implement inter-cpu communication.
- Optional arguments:
    - `tty` and `tty_no`: name of a segment associated to a multi_tty and the tty number in the component. They should both be defined at the same time
- Example:

```
mapper.map("processor",
           private = mapper.hard.cram0,
           shared  = mapper.hard.uram0)
```

OS objects

- Mandatory arguments:
    - `private`: A memory Segment where to put private data. It may be cached.
    - `shared`: A memory Segment to put shared data in. This must be uncached as it is used to implement inter-process communication.
    - `code`: A memory Segment to put all unplaced executable code in. This should be cacheable and may be read-only. If you specify a list of segments here, code will be replicated on each segment and closest copy will be used by CPUs.
- Optional arguments:
    - `tty` and `tty_no`: name of a segment associated to a multi_tty and the tty number in the component. They should both be defined at the same time
- Example:

```
mapper.map(mapper.tcg,
           private = mapper.hard.cram1,
           shared  = mapper.hard.uram1,
           code    = mapper.hard.cram1)
```