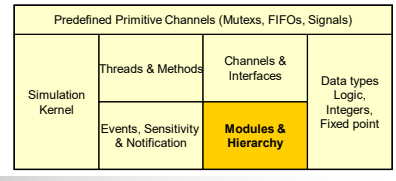
Plan

- Ch1 – Overview of System Design Using SystemC
- Ch2 – Overview of SystemC
- Ch3 – Data Types
- Ch4 – Modules
- Ch5 – Notion of Time
- Ch6 – Concurrency
- Ch7 – Predefined Channels
- Ch8 – Structure
- Ch9 – Communication
- Ch10 – Custom Channels and Data
- Ch11 – Transaction Level Modeling

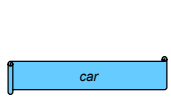
POLYTECH NICE SOPHIA
UNIVERSITÉ CÔTE D'AZUR

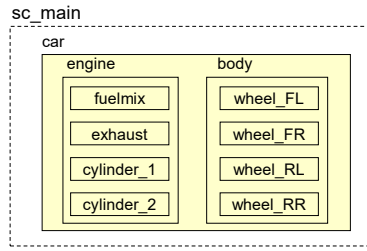
Structure

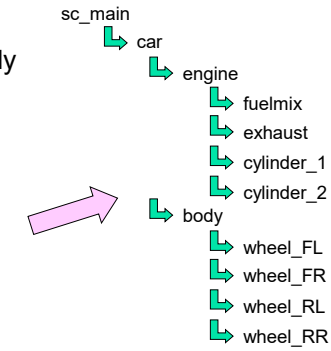


Module Hierarchy

- Systems require partitioning and hierarchy
  - complexicity
  - better understanding
  - project management
- Implementations
  - Direct or Indirect top-level
  - Direct or Indirect sub-module header-only
  - Direct or Indirect sub-module







Direct or Indirect top-level

Direct top module

```
int sc_main(int argc, char* argv[])
{
      Wheel Wheel_FL("Wheel_FL");
      Wheel Wheel_FR("Wheel_FR");
      …
      sc_start();
      return 0;
}
```

sub-design instances are instantiated and initialized in one statement

Indirect top module

```
int sc_main(int argc, char* argv[])
{
      Wheel *wheel_FL;
      Wheel *wheel_FR;
      …
      wheel_FL = new Wheel("Wheel_FL");
      wheel_FR = new Wheel("Wheel_FR");
      …
      sc_start();

      delete wheel_FL;
      delete wheel_FR;
      …
      return 0;
}
```

pointer declaration

instance creation

instance deletion

Direct or Indirect sub-module header-only

Direct sub-module header-only

Body.h

```
SC_MODULE(Body)
{
    // Sub-module instances
    Wheel Wheel_FR;
    Wheel Wheel_FL;
    Wheel Wheel_RR;
    Wheel Wheel_RL;

    // Constructor
    SC_CTOR(Body)
        : Wheel_FL("Wheel_FL"),
          Wheel_FR("Wheel_FR"),
          Wheel_RL("Wheel_RL"),
          Wheel_RR("Wheel_RR")
    {
        …
    }
};
```

declaration

initialization

Indirect sub-module header-only

Body.h

```
SC_MODULE(Body)
{
    // Sub-module instances
    Wheel *wheel_FR;
    Wheel *wheel_FL;
    Wheel *wheel_RR;
    Wheel *wheel_RL;

    // Constructor
    SC_CTOR(Body)
    {
        wheel_FL = new Wheel("Wheel_FL");
        wheel_FR = new Wheel("Wheel_FR");
        wheel_RL = new Wheel("Wheel_RL");
        wheel_RR = new Wheel("Wheel_RR");
        …
    }
    // Destructor
    ~Body()
    {
        delete wheel_FR;
        …
    }
};
```

pointer declaration

instance creation

Direct or Indirect sub-module With Separate Compilation

Direct sub-module

Body.h

```
SC_MODULE(Body)
{
    // Sub-module instances
    Wheel Wheel_FR;
    Wheel Wheel_FL;
    Wheel Wheel_RR;
    Wheel Wheel_RL;

    // Constructor
    SC_HAS_PROCESS(Body);
    Body(sc_module_name nm);
};
```

declaration only

Body.cpp

best approaches for IP !
pre-compiled object files

```
// Constructor
Body::Body(sc_module_name nm)
        : Wheel_FL("Wheel_FL"),
          Wheel_FR("Wheel_FR"),
          Wheel_RL("Wheel_RL"),
          Wheel_RR("Wheel_RR"),
sc_module(nm)
{
        …
}
```

Indirect sub-module

Body.h

```
SC_MODULE(Body)
{
    // Sub-module instances
    Wheel *wheel_FR;
    Wheel *wheel_FL;
    Wheel *wheel_RR;
    Wheel *wheel_RL;

    // Constructor
    SC_HAS_PROCESS(Body);
    Body(sc_module_name nm);
};
```

pointer declaration

instance creation

Body.cpp

```
// Constructor
Body::Body(sc_module_name nm)
        : sc_module(nm)
{
        wheel_FL = new Wheel("Wheel_FL");
        wheel_FR = new Wheel("Wheel_FR");
        wheel_RL = new Wheel("Wheel_RL");
        wheel_RR = new Wheel("Wheel_RR");
        …
}
```

Conclusion

Level	Allocation	+	-
Main	Direct	Least code	Inconsistent with other levels
Main	Indirect	Dynamically configurable	Involves pointers
Module	Direct header-only	- All in one file - Easier to understand	Requires sub-module headers
Module	Indirect header-only	- All in one file - Dynamically configurable	Involves pointers
Module	Direct sub-module (separate compilation)	Hides implementation	Requires sub-module headers
Module	Indirect sub-module (separate compilation)	- Hides sub-module headers and implementation - Dynamically configurable	Involves pointers