

# Components and abstraction levels for SoC

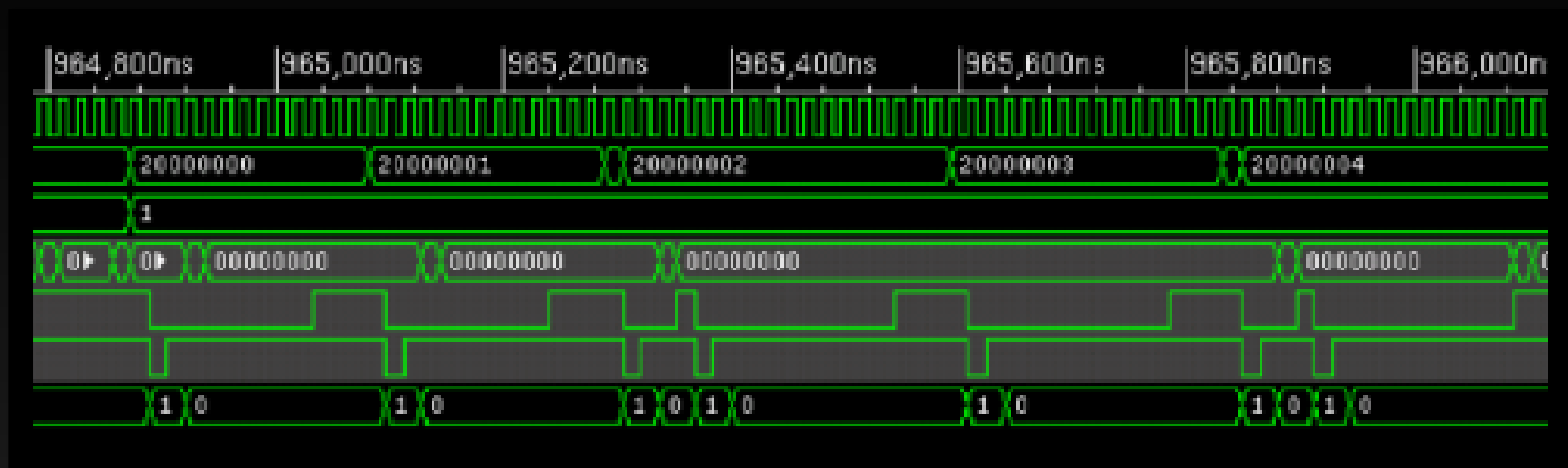
Giovanni Funchal  
STMicroelectronics  
Verimag

EMSoC'2008

# SoCs are everywhere

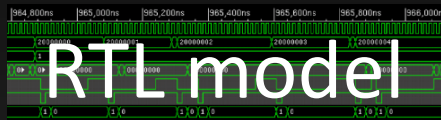


# Register Transfer Level

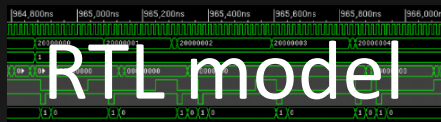


Precise model of Hardware

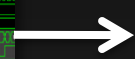
# Design flow



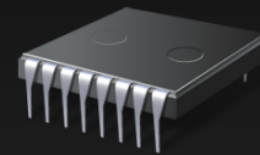
# Design flow



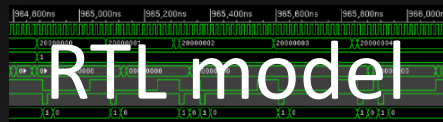
RTL model



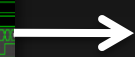
Fabrication



# Design flow



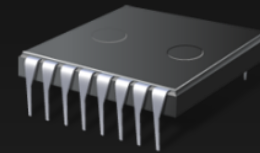
RTL model



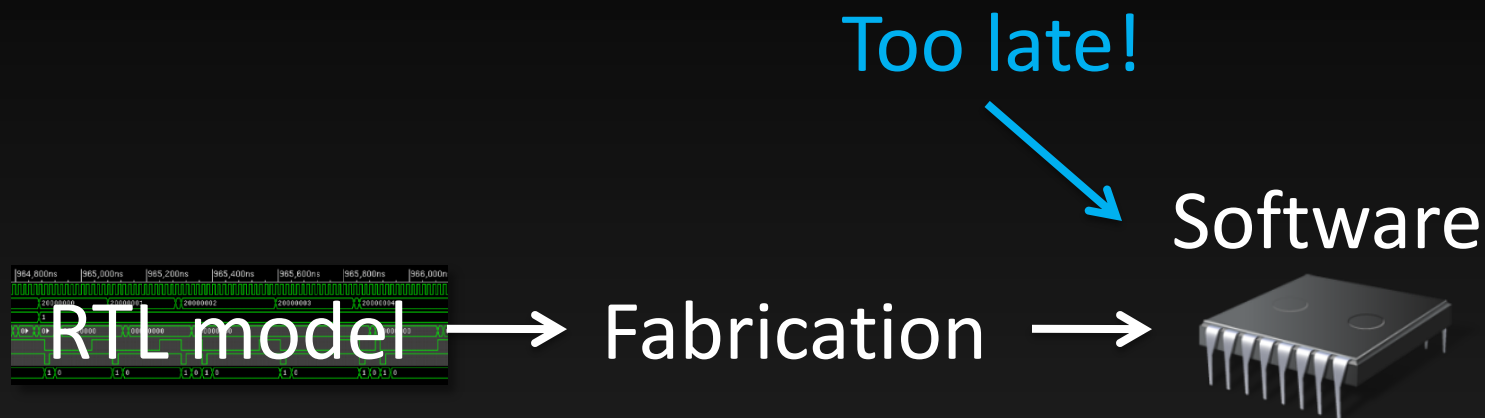
Fabrication



Software

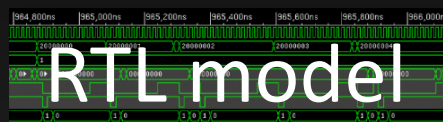


# Design flow

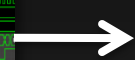


# Design flow

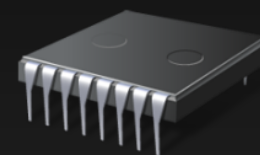
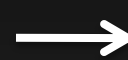
Software



RTL model

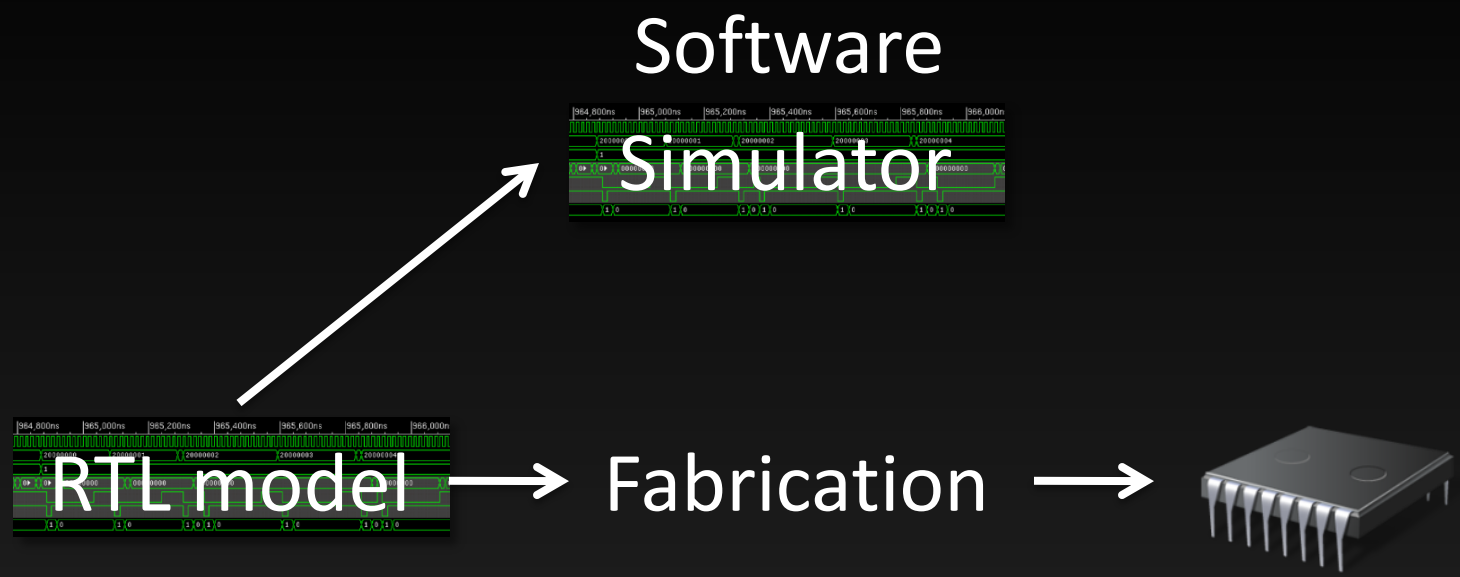


Fabrication

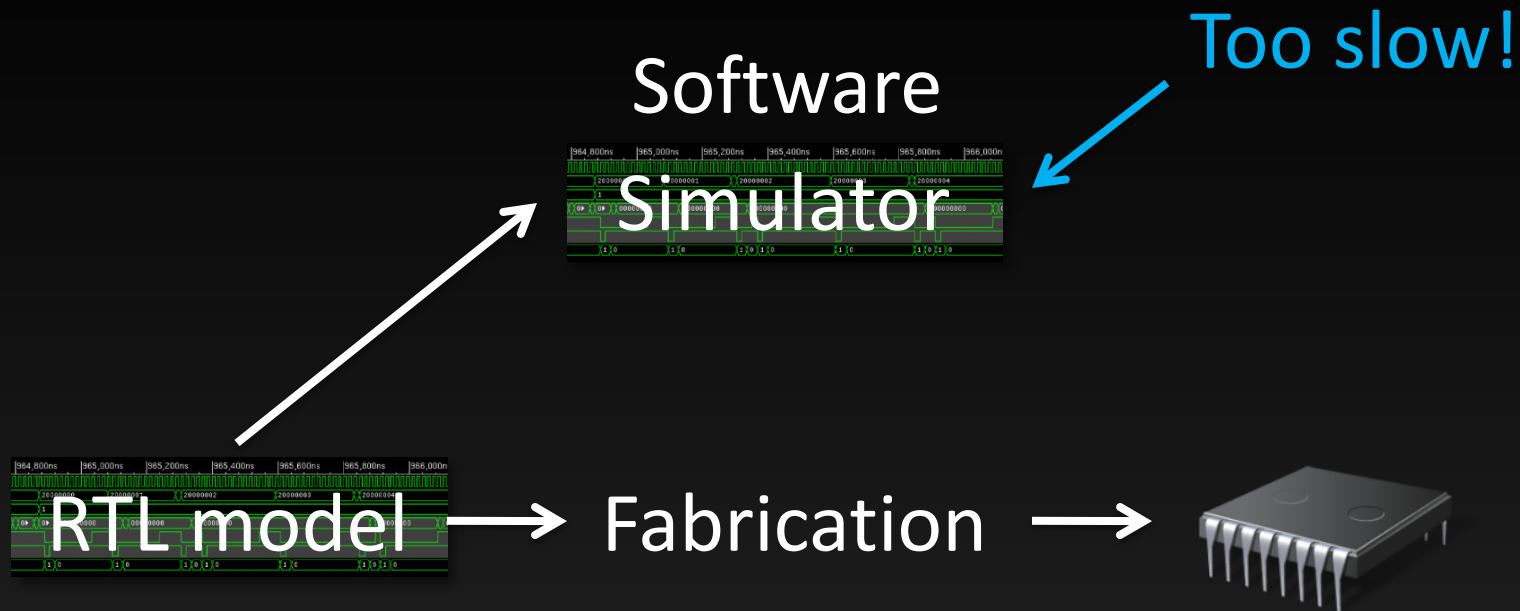




# Design flow



# Design flow

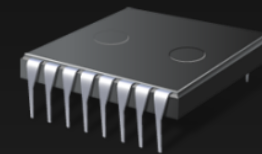
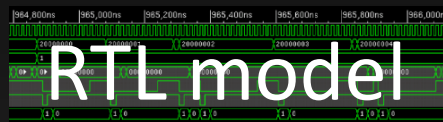


# Design flow

Simpler, faster

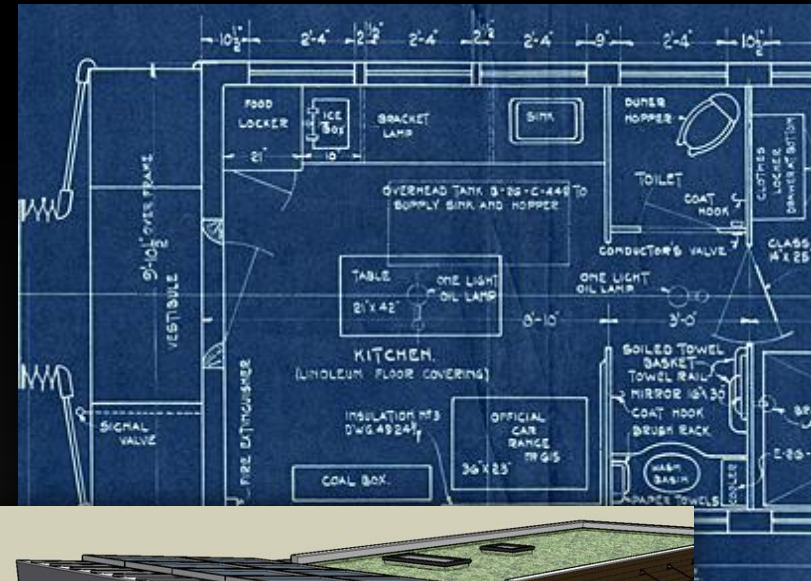


Software



# RTL models

- Late (hw fab)
- Precise
- Slow

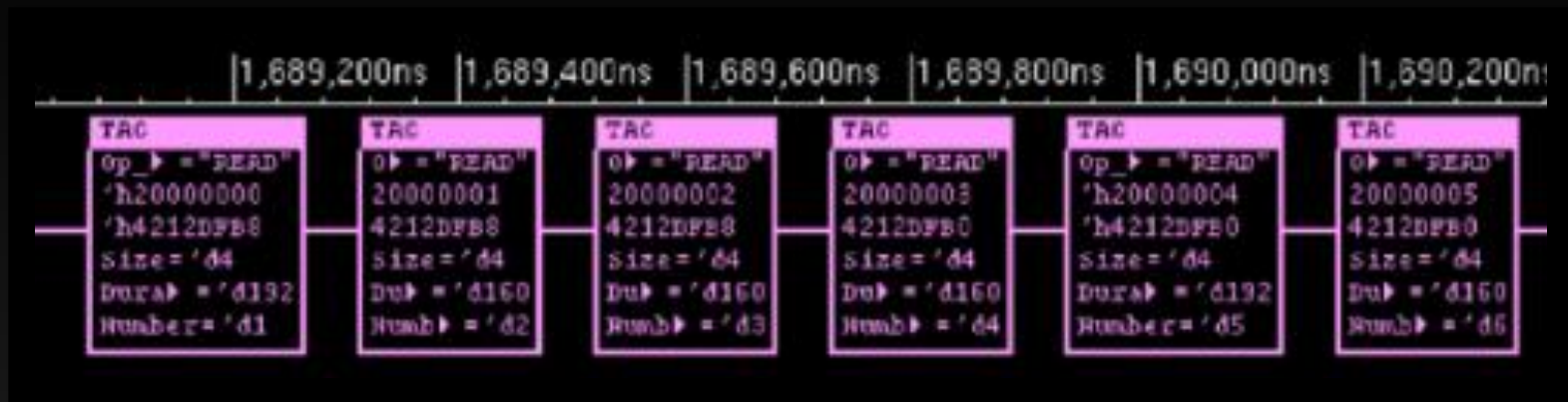


# TLM models

- Early (sw dev)
- Less precise
- Fast

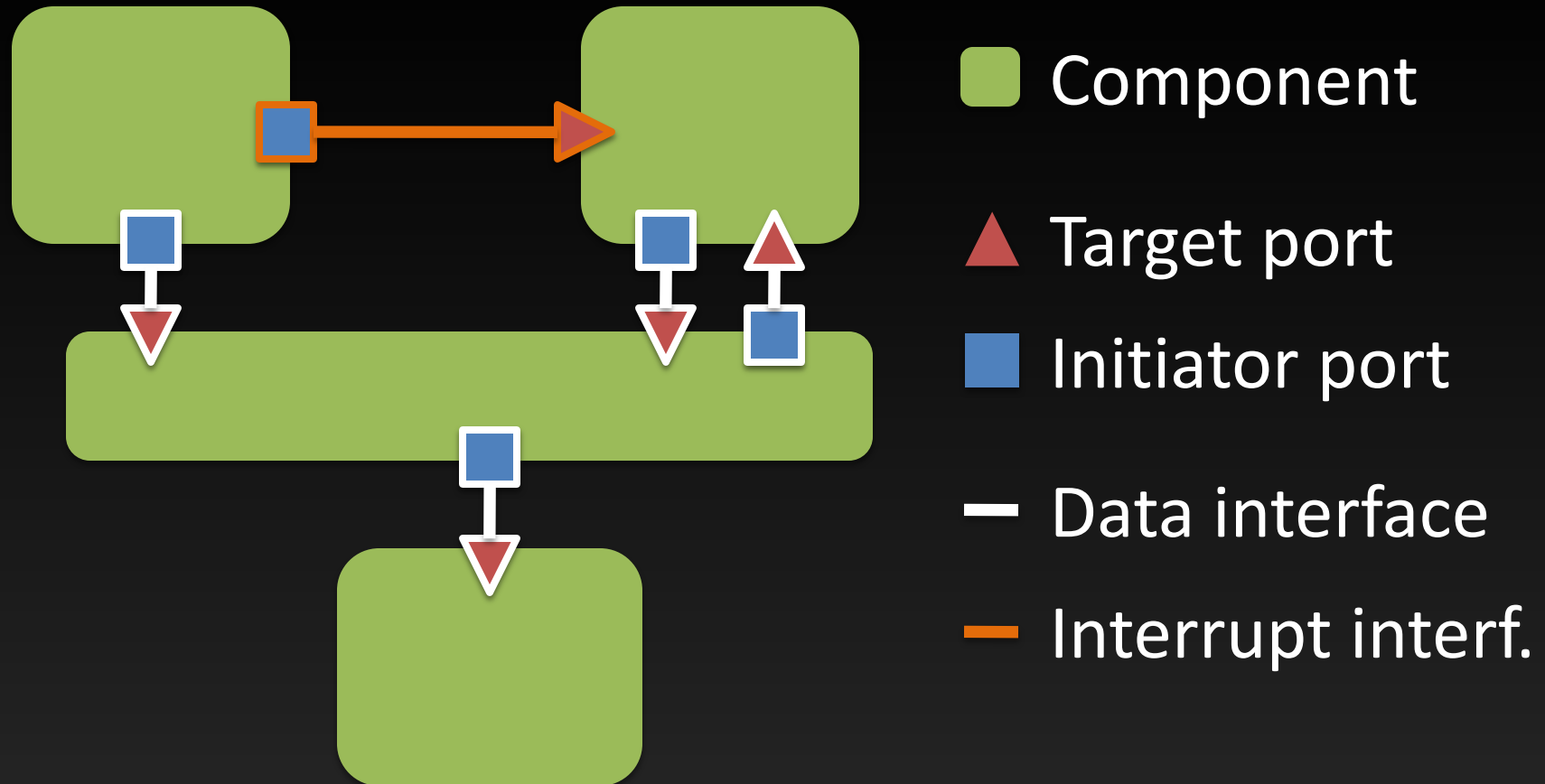


# Transaction Level Modeling

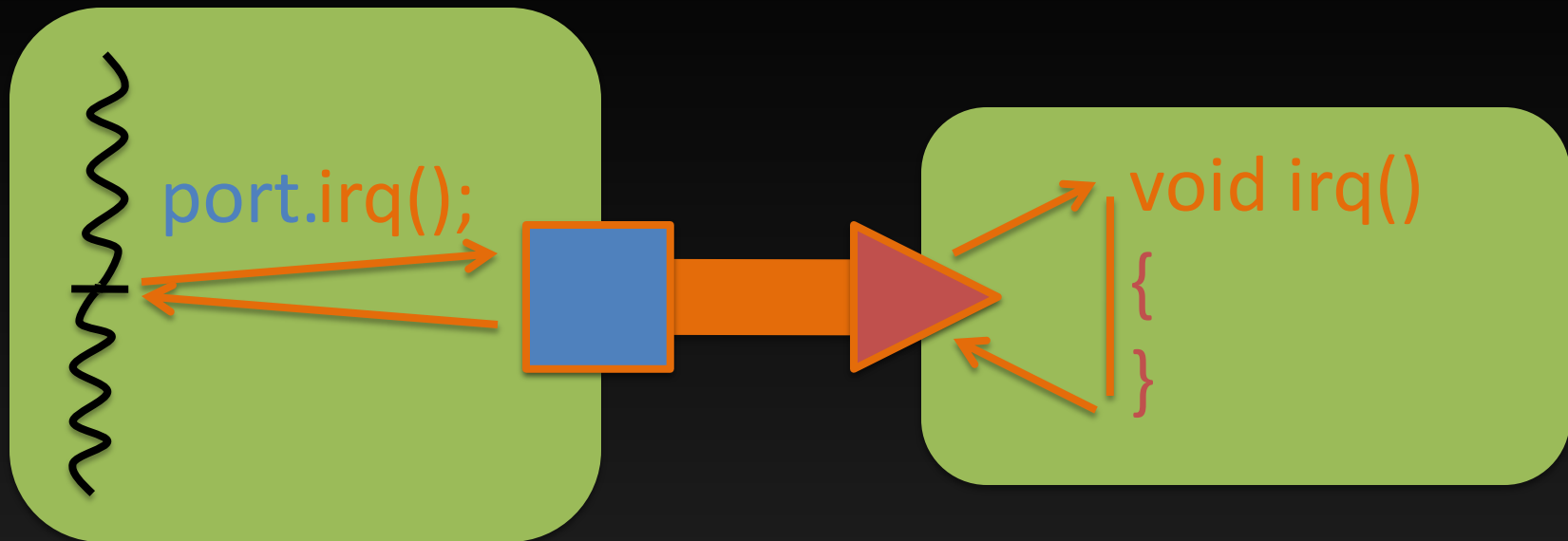


Abstract transfers of data

# TLM concepts



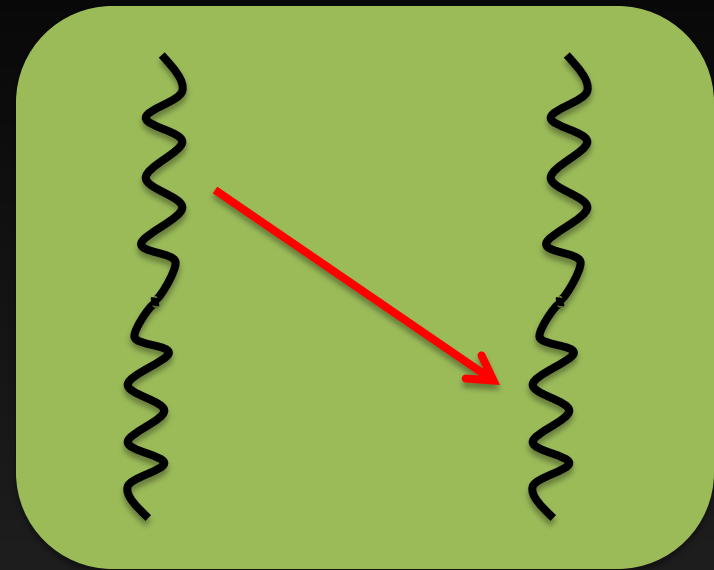
# Interface function calls



Interface function call = transaction

# IEEE 1666 Std SystemC

- Processes
- Events
- Scheduler





# Processes

- Waiting
  - Ready
  - Running
- Event
- Scheduler
- 
- ```
graph TD; Waiting -- Event --> Ready; Ready -- Scheduler --> Running;
```

# Processes

- Waiting
- Ready
- Running

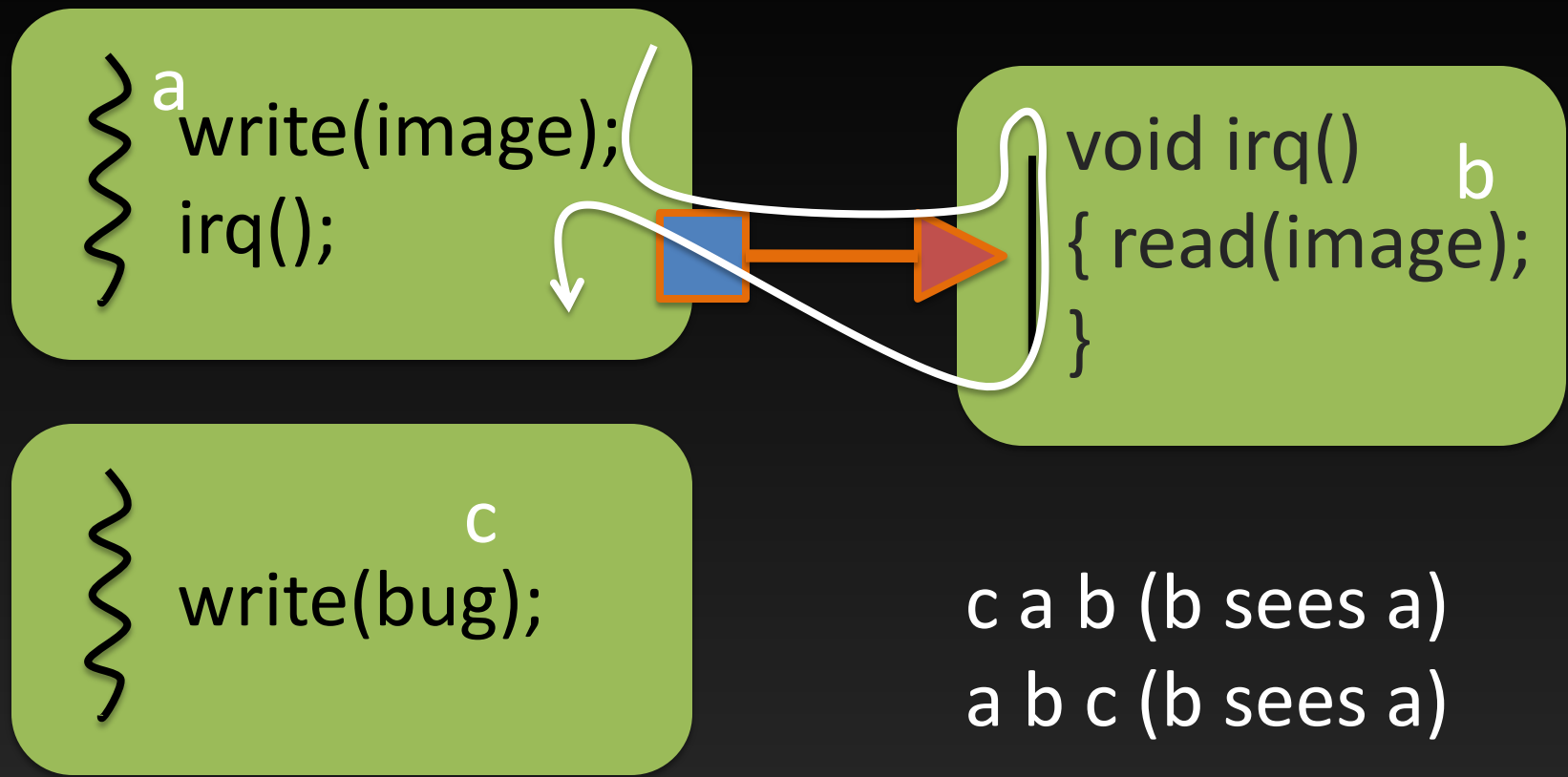


Process must  
yield control  
(explicitly)

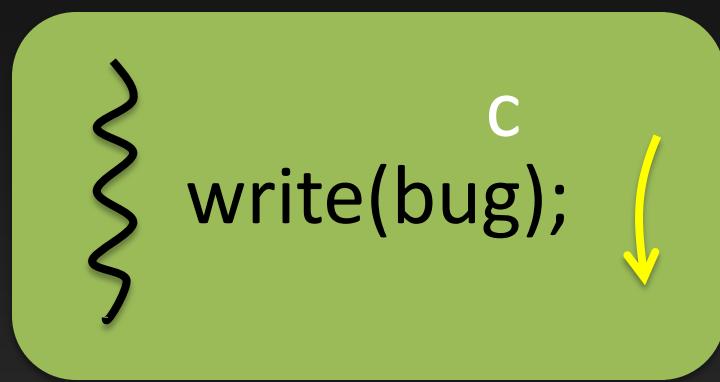
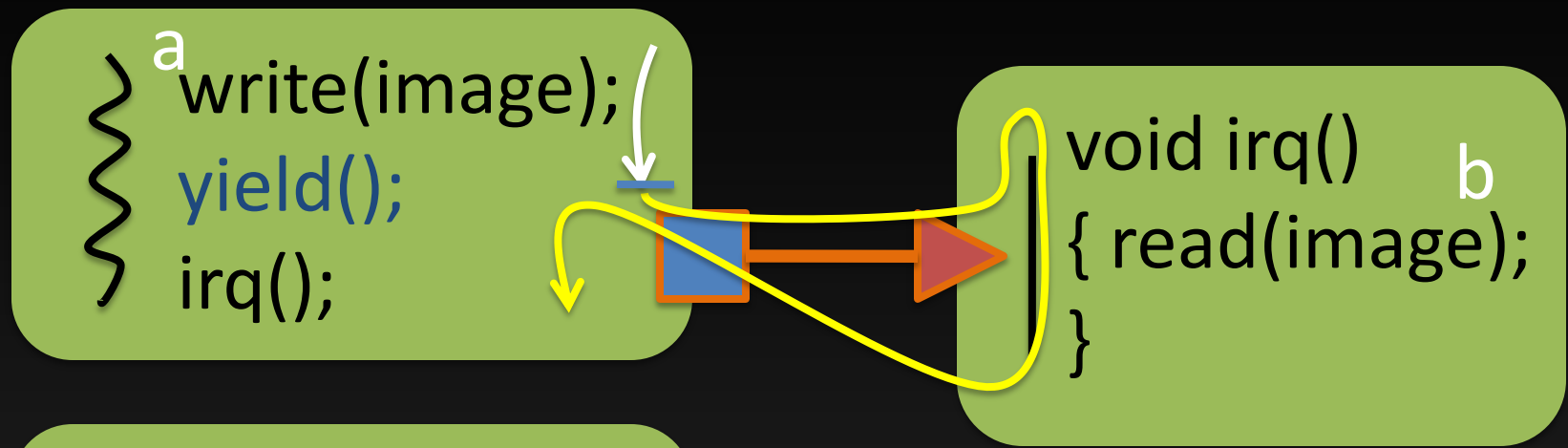
Process must  
yield control  
(explicitly)

When?

# The problem

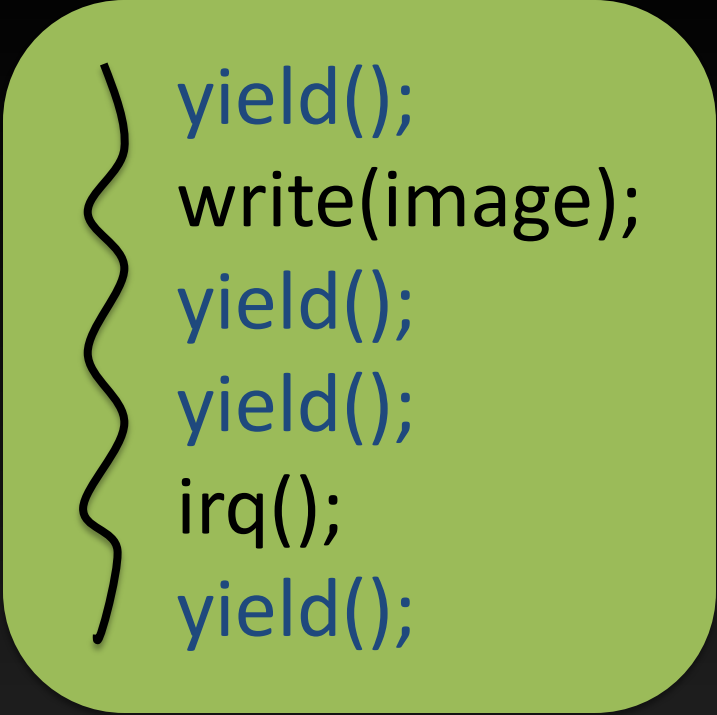


# The problem



- c a b (b sees a)
- a b c (b sees a)
- a c b (b sees c)

# Initial approach



```
yield();  
write(image);  
yield();  
yield();  
irq();  
yield();
```

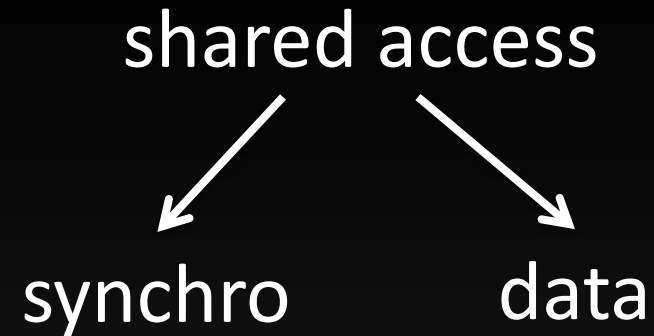
It is enough to yield  
before and after every  
shared access

Poor performance

At STMicro: TAC1

# Second approach

```
} x = write(image);  
} if(x) yield();  
} x = irq();  
} if(x) yield();
```

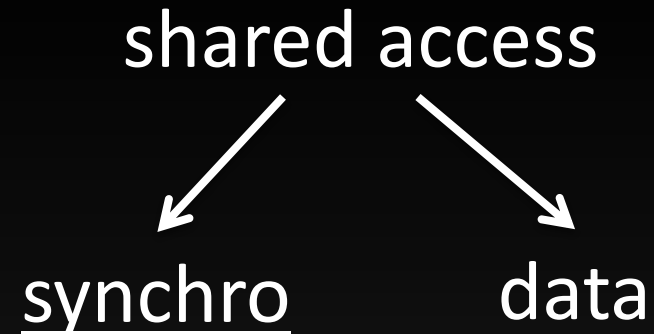


At STMicro: TAC2  
(state-of-the-art)

# Second approach

```
} x = write(image);  
} x = irq();  
} if(x) yield();
```

```
} write(image);  
} yield();  
} irq();
```

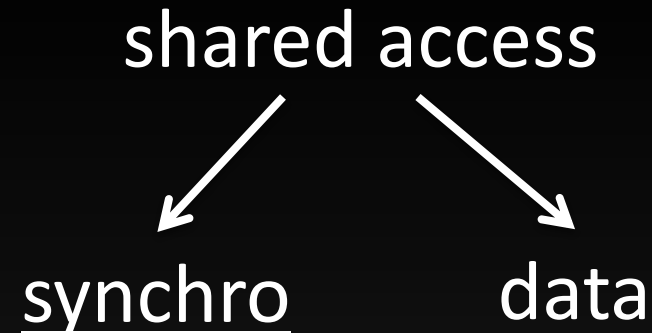


At STMicro: TAC2  
(state-of-the-art)



# Best known approach

```
x = write(image);  
if(irq(check)) {  
    yield();  
    x = irq(perform);  
}
```



At STMicro: TAC3  
(Jerome Cornet)

# Perspectives of my thesis

- Identify the exact meaning of “yield”
- Guidelines (contracts) for writing components so that the assembly “works”
- Better understanding of TLM vs RTL

