Physical Implementation: Timing constraints

Rajit Manohar

Asynchronous VLSI and Architecture (AVLSI) Group
Computer Systems Lab, Yale University

https://csl.yale.edu/~rajit/
https://avlsi.csl.yale.edu/act
So far...

• Behavioral modeling
  ❖ Message-passing programming in CHP
  ❖ Dataflow graphs
  ❖ Links and joints

• Gate-level design
  ❖ Link-joint implementations
  ❖ Controllers for dataflow components
  ❖ Syntax-directed translation

• Multiple circuit families!
  ❖ ... for channels
  ❖ ... for links
  ❖ ... for controllers
Why use different circuit families?

• Sufficiently **different costs** in energy, delay, area, and implementation effort
  - Implementation effort = effort to validate any *timing* requirements

• Basic intuition on the trade-offs
  - A circuit is robust to large uncertainty in the delay of a signal/gate **if some other circuit checks that it has changed**
    
    ⇒ this has a cost in circuits, and hence area, delay, energy
  
  - If we eliminate this cost and the circuit is **not** robust, then we must **check** that the timing requirements are met

• Examples of delay-robustness
  - Quasi delay-insensitive (QDI) / Speed-independent (SI)
    - Correct behavior is **independent** of gate delay
    - Some wire forks have a relative delay requirement
Why use different circuit families?

- Example: MOUSETRAP

**Timing Analysis**

Setup constraint: matched delay

Hold Time constraint:

*Data must be safely “captured” by Stage N before new inputs arrive from Stage N-1*

- Stage N’s “self-loop” faster than entire path through previous stage
Timing requirements

• In order of verification complexity…
  ❖ Local to a small module
  ❖ Between neighboring modules that exchange data
  ❖ Paths across multiple modules

• A common approach to describe constraints: **relative timing**
  ❖ “A signal transition x+ must happen before y-”
  ❖ “The data must change before the request goes high”
Timing constraints in ACT

- Associate an “iteration counter” with circuit

  *[ S ] // infinite loop

- Each signal transition “x+” / “x-” has an iteration counter attached to it
  - Some transitions connect iteration “i” to iteration “i+1”
  - This is determined during logic synthesis

- A timing fork
Timing constraints in ACT

- maximum delay from A to C < minimum delay from A to B

spec {
    timing a+ : c+ < b-
}
Timing constraints in ACT

- Example: bundled-data channel

```cpp
template<pint M>
defchan bd (bool d[M]; bool r, a)
{
    spec {
        /* timing fork */
        timing a- : d* < r**
    }
    ...
}
```

*The standard channel definitions in `std::channel` include timing constraints, so those will be automatically included.*

*actsim and prsim check constraints during simulation.*
Gate level specification

- Circuits specified using production rules

- Timing
  - Constraints specified in channels and/or controllers
  - Relation between one iteration to the next also specified
    - This is determined during logic synthesis
    - This can also be computed using the chip reset protocol

- Delay annotations for more accurate timing simulation

```plaintext
prs {
    [after=25] a -> b-
}
```