RTLog Framework

Yet another open HDL and compiler, this time for

Relative-Timing desing

UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

What is RTLog?

- A Hardware Description Language
- Simpler than Verilog or VHDL. No overhead clauses in the syntax
- Pure RTL, does not require behavioral constructions
- Useful to describe asynchronous circuits

Keywords of RTLog

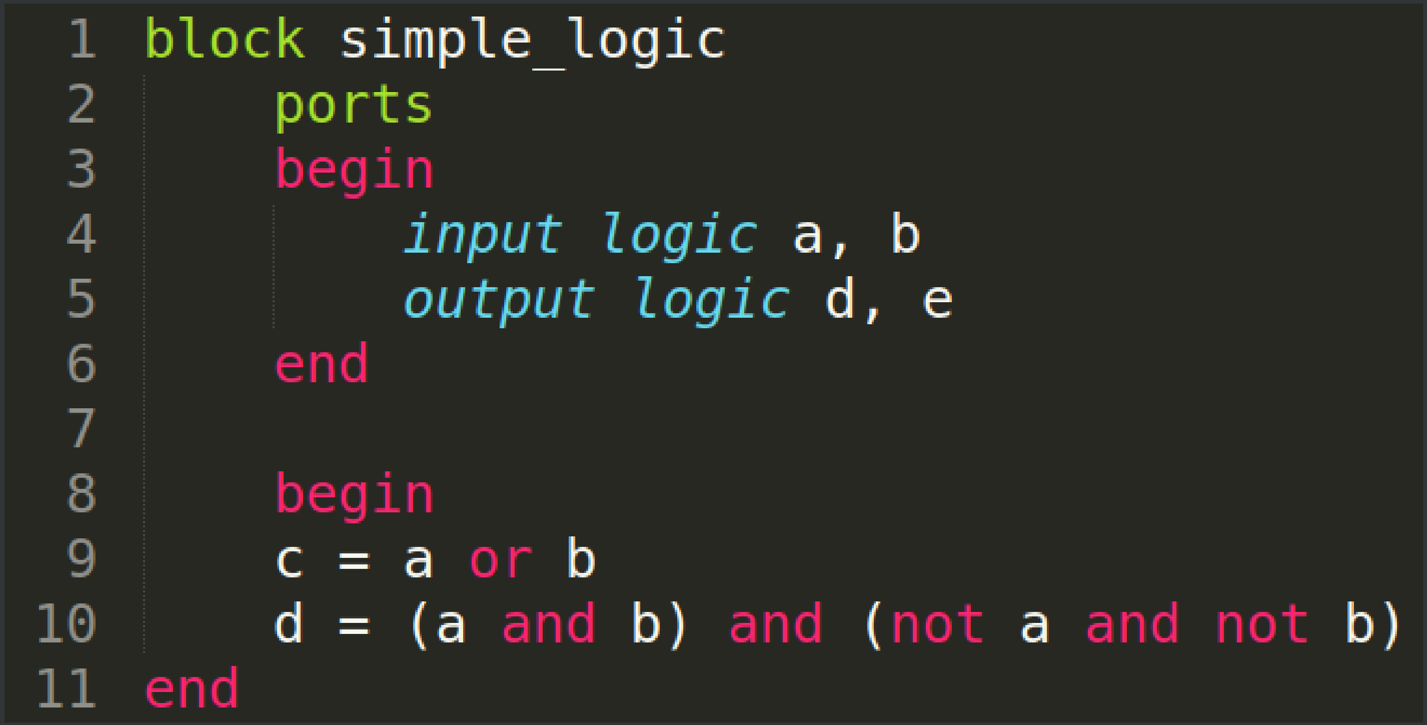
and	cat	if	natural	parameters	select
begin	constant	in	not	ports	when
block	end	for	or	reg	xor
case	else	logic	others	rep	xnor

Operations supported by RTLog

not	Logic NOT
or	Logic OR
and	Logic AND
xor	Logic XOR
+	Addition
+x	Addition with carry
-	Subtraction
-x	Subtraction with carry
>>	Right Shift
>>s	Right Shift signed
<<	Left Shift

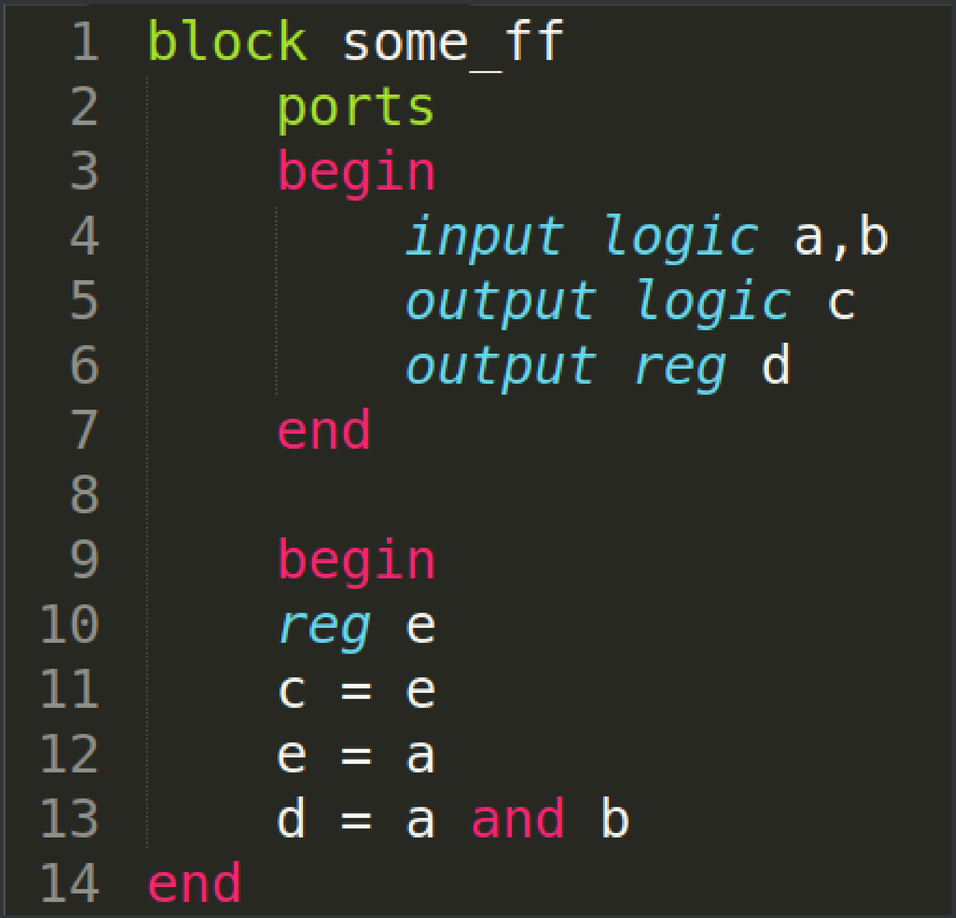
*	Multiplication
*x	Signed Multiplication
==	Compare equal
!=	Compare defferent
<	Compare smaller
<=	Compare smaller or equal
>	Compare greater
>=	Compare greater or equal
cat	Concatenation
rep	Replication

I/O Ports and Simple Logic

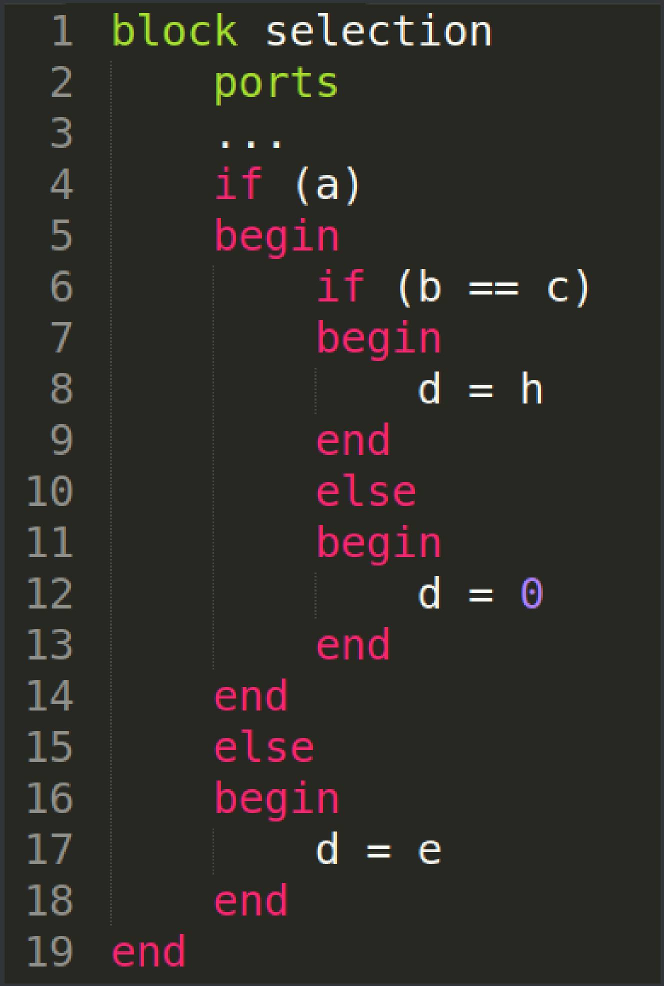


```
block simple_logic
    ports
    begin
        input logic a, b
        output logic d, e
    end

    begin
    c = a or b
    d = (a and b) and (not a and not b)
end
```

Registers and Signal Selection

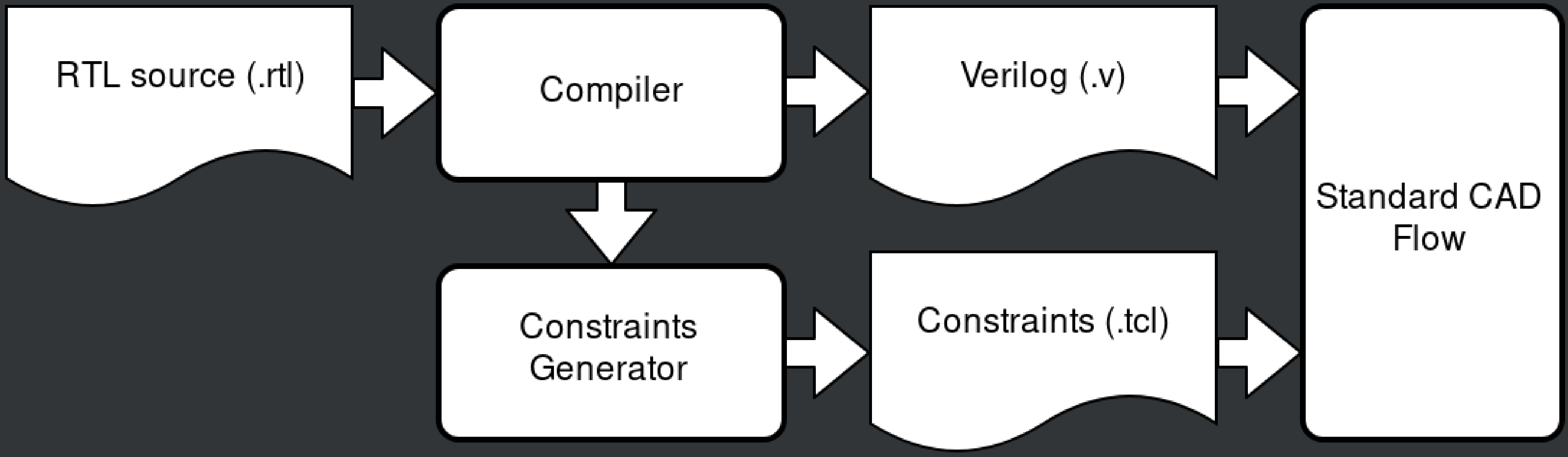


```
block some_ff
    ports
    begin
        input logic a,b
        output logic c
        output reg d
    end

    begin
    reg e
    c = e
    e = a
    d = a and b
end
```



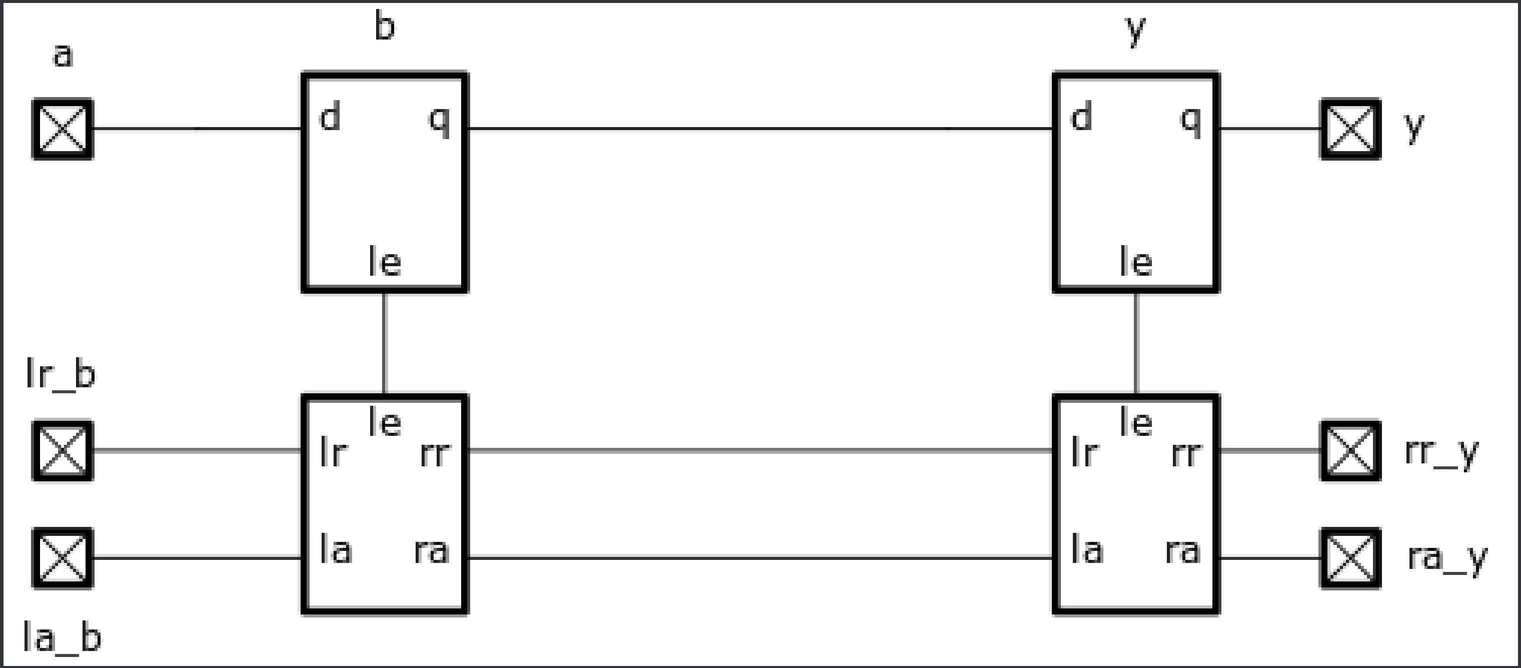
```
block selection
    ports
    ...
    if (a)
    begin
        if (b == c)
        begin
            d = h
        end
        else
        begin
            d = 0
        end
    end
    else
    begin
        d = e
    end
end
```

RTLog desing flow



What about asynchronous circuits?

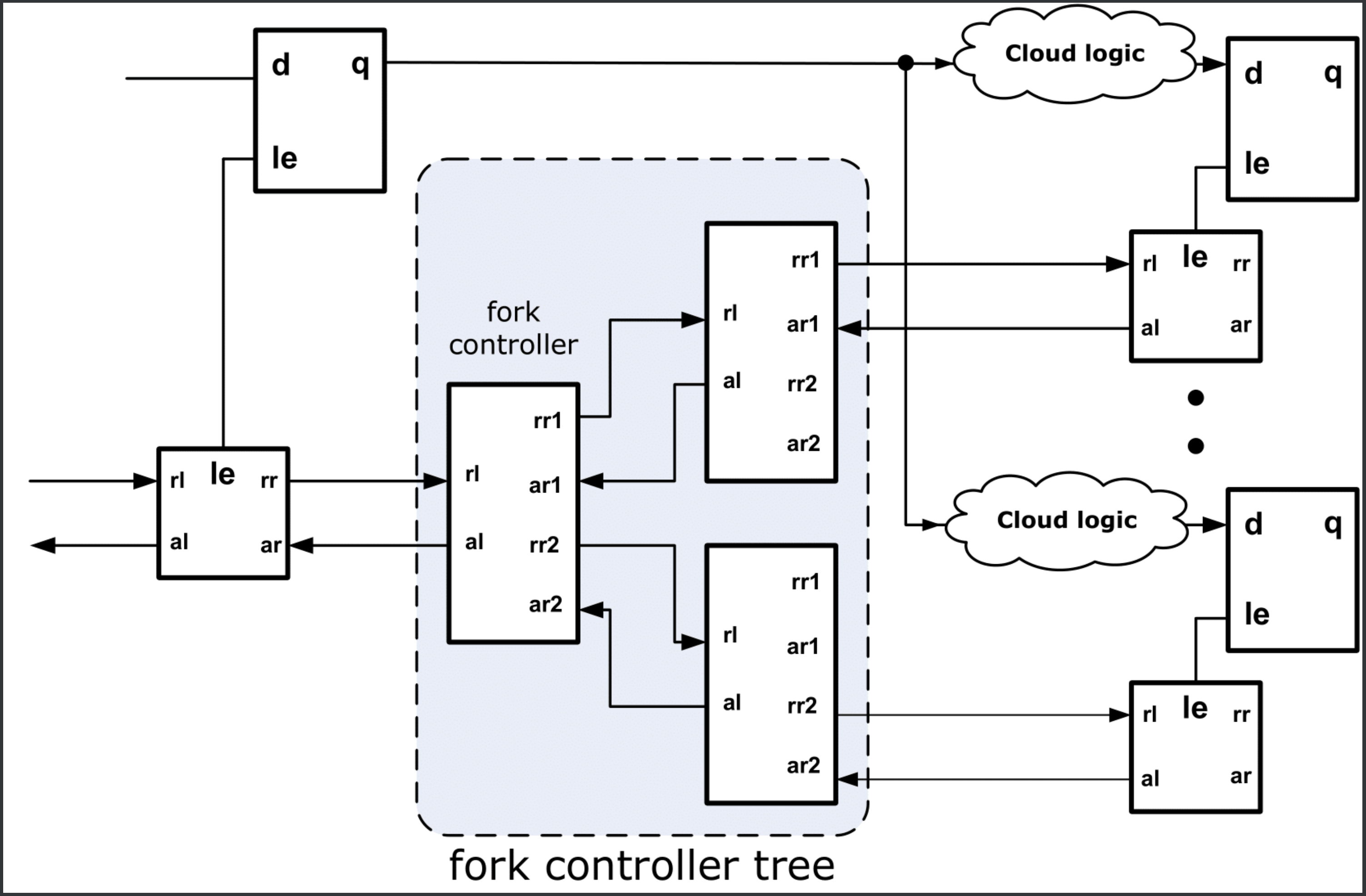
RTlog facilitates the implementation of asynchronous pipeline stages using a 4-phase handshake protocol



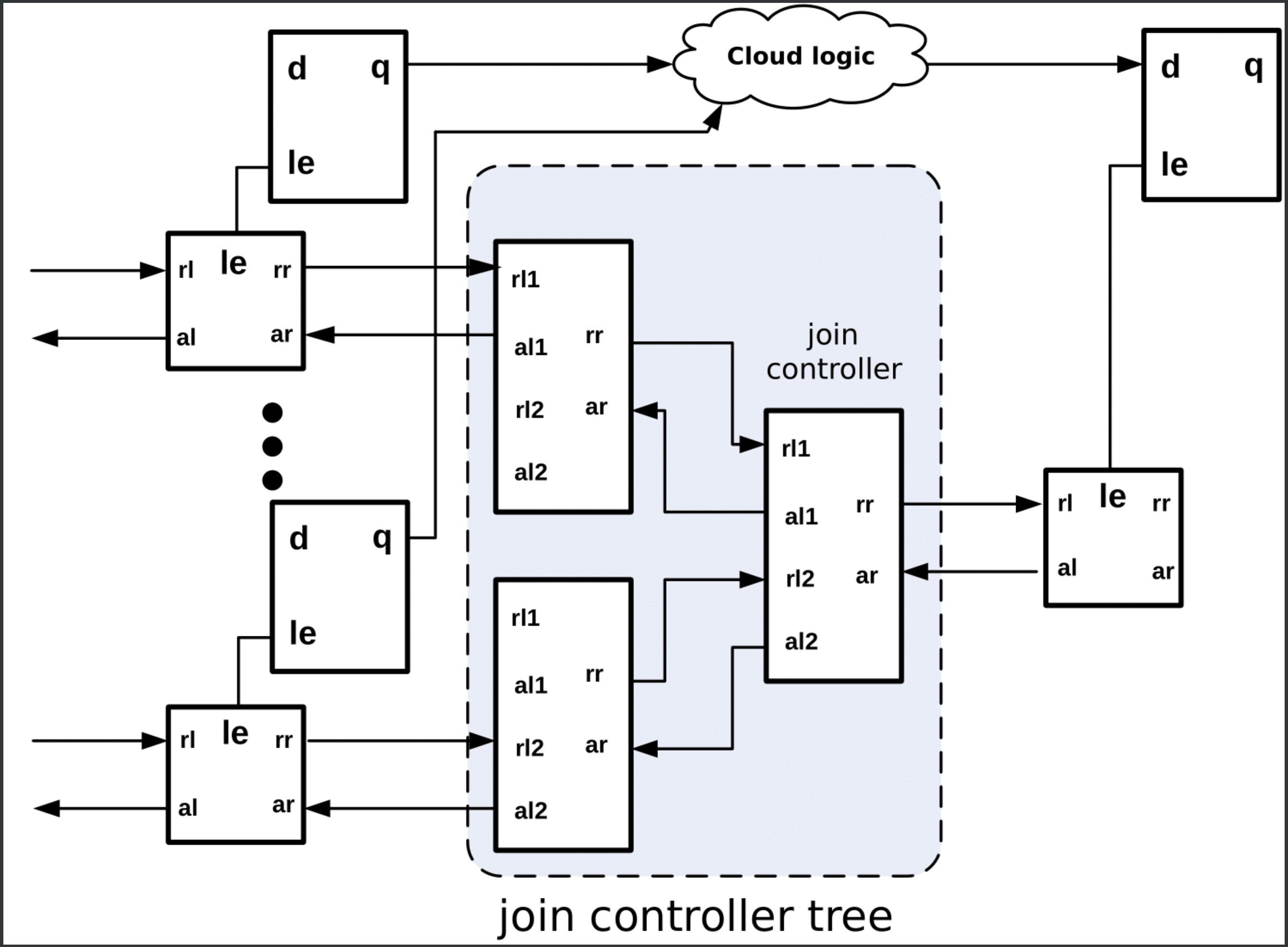
Relative timing circuit generation

- Find all register-to-register paths
- Replace the registers by latches
- For each latch, add a templated linear controller
- Wire request and acknowledge signals
- Add input and output handshake ports
- Connect ports to the corresponding latches

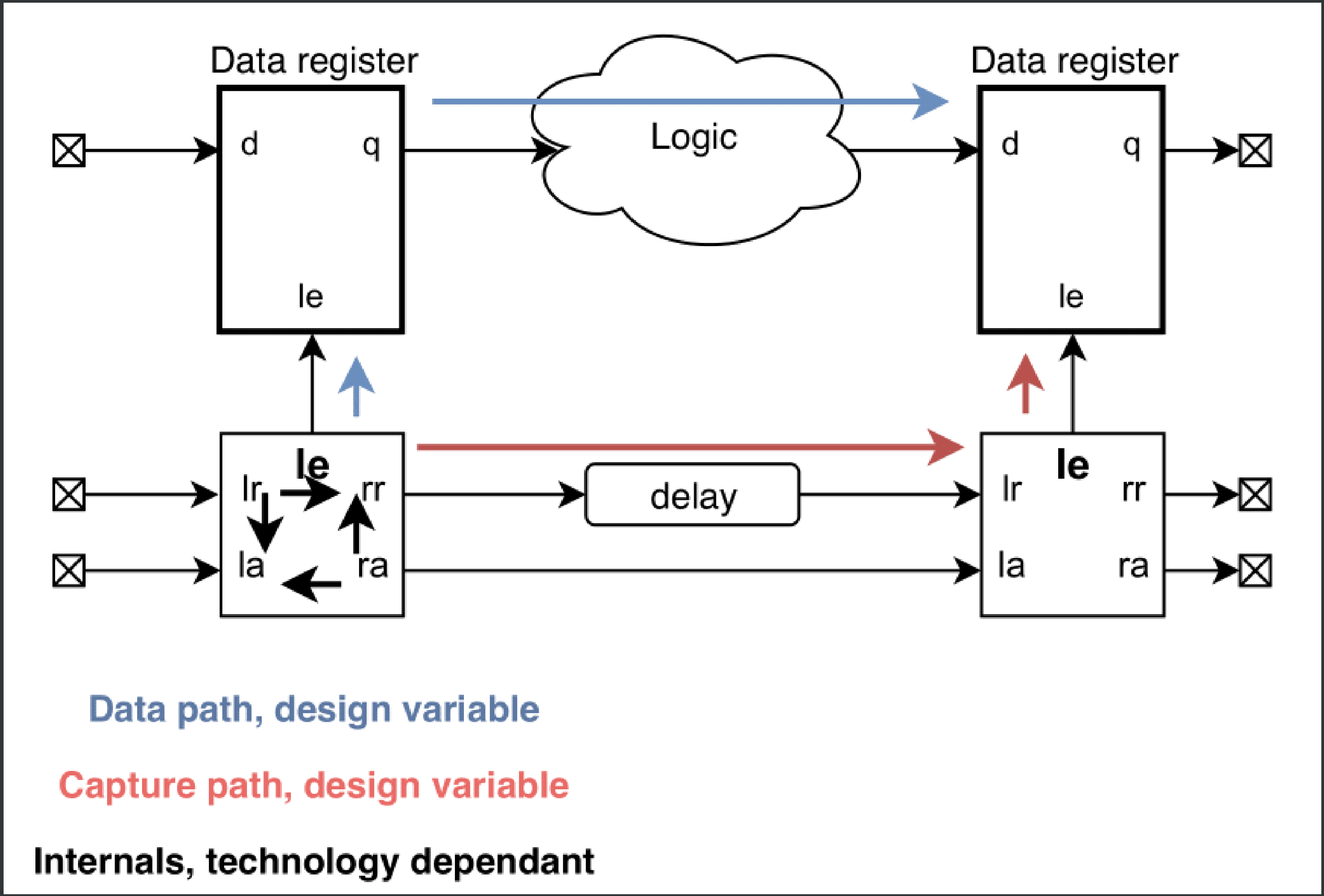
Fork and join controllers



Fork and join controllers



Relative timing constraints


Data register
d
q
le
Logic
Data register
d
q
le
le
lr
rr
la
ra
delay
le
lr
rr
la
ra
Data path, design variable
Capture path, design variable
Internals, technology dependant

Future work

- Language extension (*case* statement and two-dimensional array)
- Integration with synthesis and STA OpenSource tools
- Iterative PrimeTime runs for performance and power optimizations

Thank you for listening

https://github.com/VLSI-UTN-FRBA/RTLog