Universidad
Carlos III de Madrid

Design at Register Transfer Level (RTL)

Index

- Abstraction levels
- RT level digital systems structure
  - Datapath
  - Control
- Design optimization
  - Area
  - Performance
- Optimization by design
  - Serial, parallel and pipeline architectures

Digital systems

- Digital systems specification
  - Inputs (protocol)
  - Algorithm: ordered set of operations
  - Outputs (protocol)
- Design at Register Transfer Level

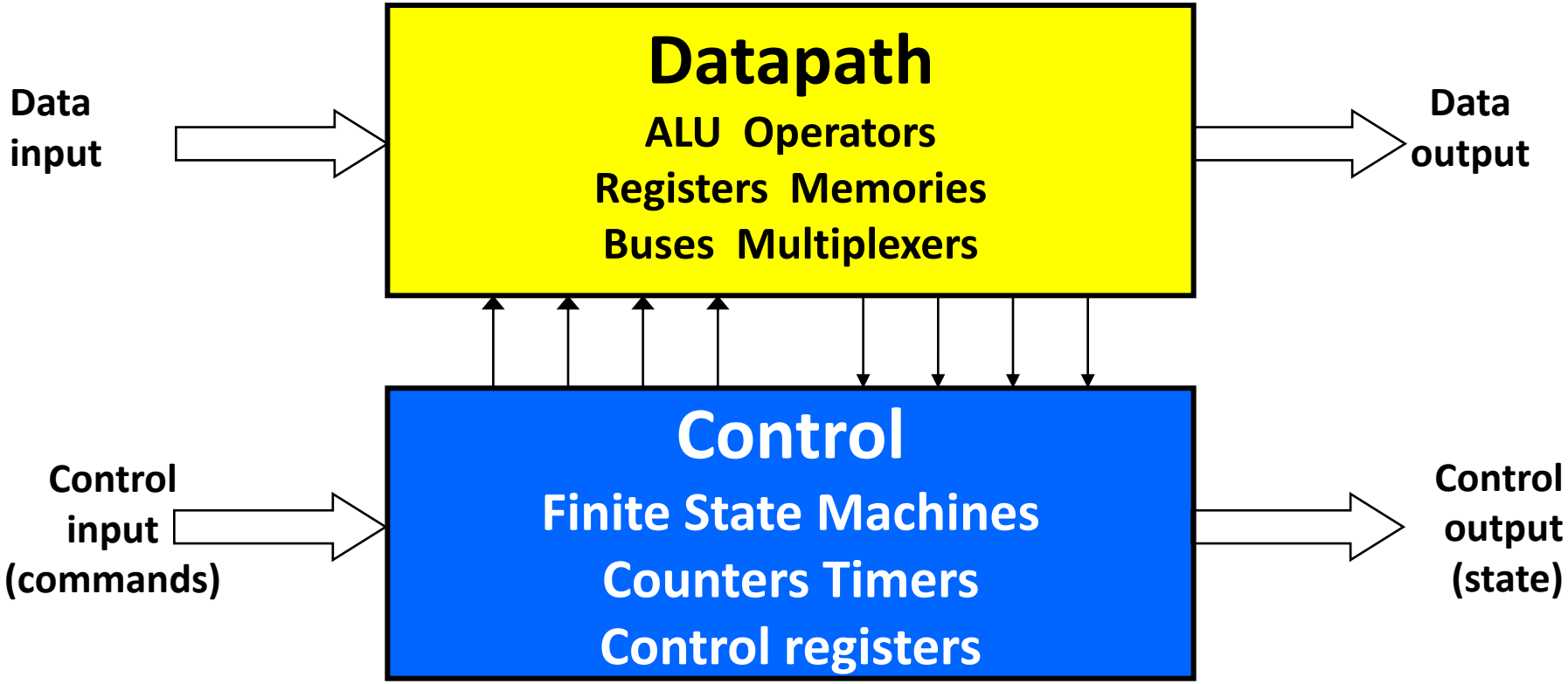
Abstraction levels

Synthesis

	Logic	RTL
Time resulution	Delays (ps)	Clock cycle
Data	Bits	Vectors Integers
Functional components	Logic functions	Operations State diagrams
Structural components	Gates Flip-flops	ALU Registers Multiplexers Counters Buses

Too Complex

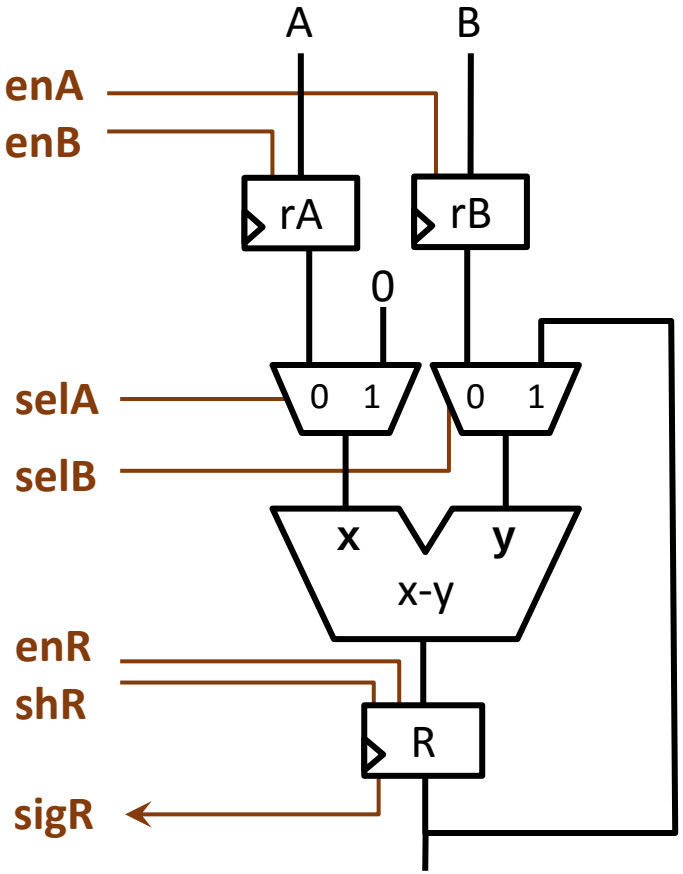
Digital systems structure



Sample algorithm

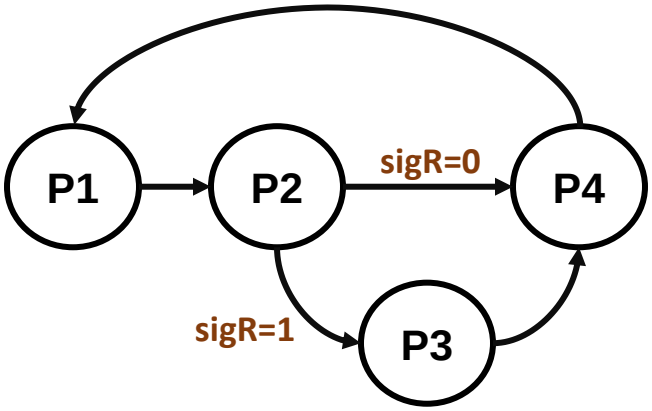
- Function:
  - R = abs(A-B)/2
- Algorithm:
  1. **R <= A-B**
  2. **If R<0 then R <= -R**
  3. **R <= R/2**

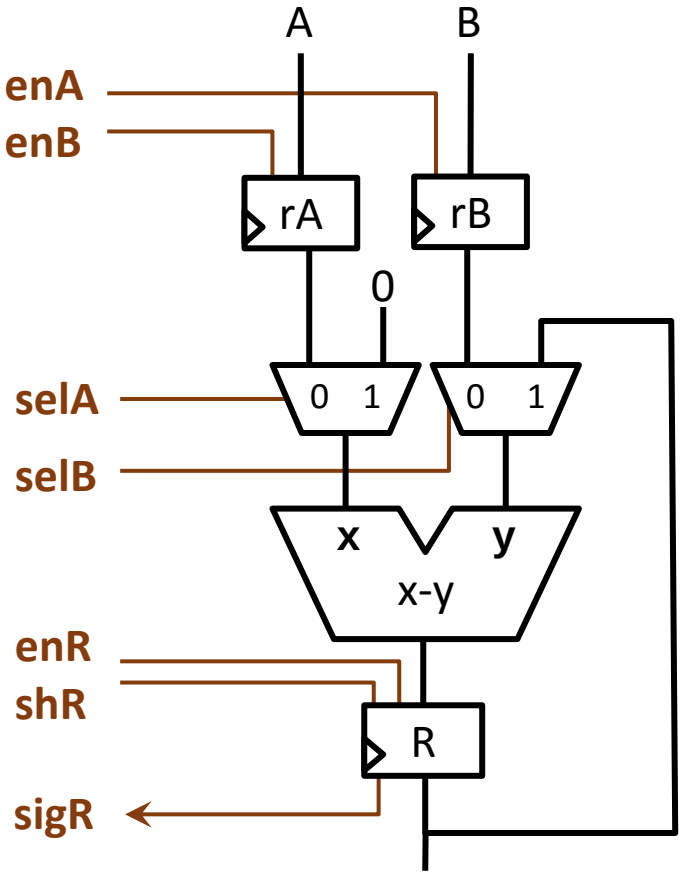
Signals	Function
enA, enB, enR	Register load enable
selA, selB	Mux select
shR	R shift
sigR	R sign



Algorithm control

Step	Operation	enA enB	selA selB	enR	shR
P1	Load operators	1	X	0	0
P2	R <= rA – rB	0	0	1	0
P3	R <= 0 - R	0	1	1	0
P4	R <= R/2	0	X	1	1





Datapath description

```
-- Registers y and substraction
process (Reset, Clk)
begin
  if Reset = '1' then
    rA <= (others => '0');
    rB <= (others => '0');
    R <= (others => '0');
  elsif Clk'event and Clk='1' then
    if enA='1' THEN
      rA  <= A;
    end if;
    if enB='1' THEN
      rB  <= B;
    end if;
    if enR='1' THEN
      if shR='1' THEN
        R <= R/2;
      else
        R  <= mA - mB;
      end if;
    end if;
  end if;
end process;
```

```
-- rR sign (MSB)
sigR <= R(R'high);

-- Multiplexers
mA <= rA when selA='0' else
      (others=>'0');
mB <= rB when selA='0' else
      R;
```

Circuit synthesis

- Synthesis: transform a circuit described at RTL into a circuit in the Logic Level (netlist)
- After synthesis, analyze:
  - Area: area taken by the circuit cells (ASIC) or amount of resources used (FPGA)
  - Performance: circuit processing speed
  - Power

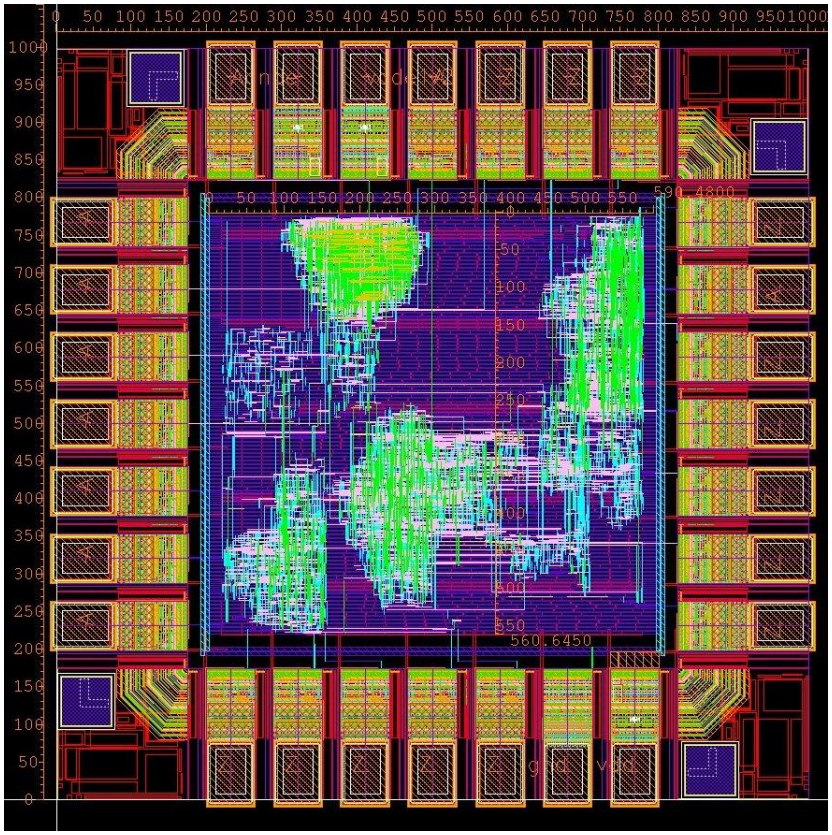
Bigger circuit → More Area → Higher cost

Area

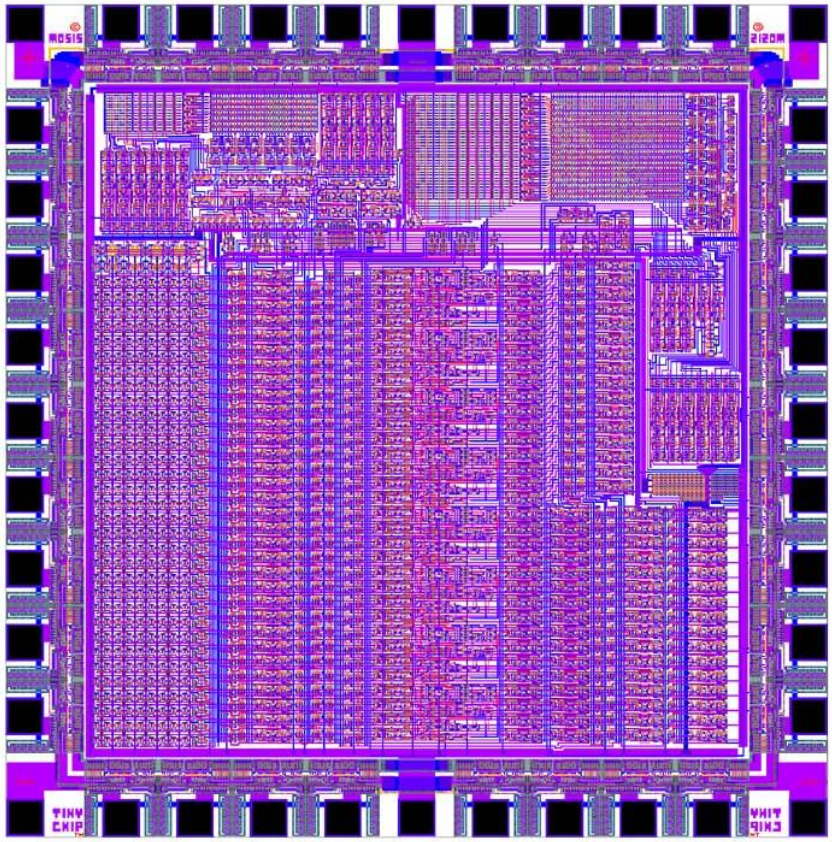
- FPGA (Field Programmable Gate Array)
  - Resources: LUTs, flip-flops, RAM blocks, pins…
  - Before Place & Route (implementation), approximate estimations
  - Cannot achieve 100% occupation due to routing
- ASIC (Application Specific Integrates Circuit)
  - No. gates, flip-flops, memories ➔ area estimation
  - Interconnection area
  - Before Place & Route, approximate estimations
  - Pins/pads: *core limited*, *pad limited.* Cell area may be irrelevant

Core and pad limited circuits

- **Pad limited**



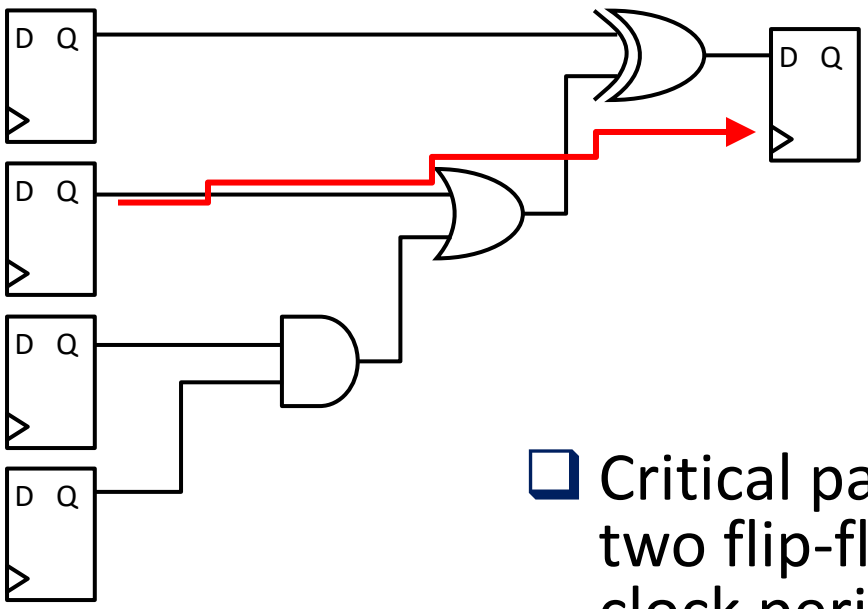
- **Core limited**



Performance

- Clock frequency
  - $f_{Clk} = 1 / T_{Clk}$
- *Throughput or data rate* : no. data per time unit
  - N = clock cycles required to process a single datum
  - $T_{dat} = N \times T_{Clk}$ ➔ Single datum processing time
  - $R = 1 / (N \times T_{Clk}) = f_{Clk} / N$ ➔ Data per second
- Do not confuse performance with clock frequency

Clock period: critical path



- Critical path: slower path between two flip-flops, which forces the clock period
  - Flip-flop times: setup time, hold time
  - Gate delays
  - Interconnection delays
- $T_{Clk} > T_{critical\_path}$

Design optimization

- Area and performance are opposite: higher performance usually imply higher area
- Requirements
  - For a given frequency ➔ get the lowest area
  - For a given area ➔ get the highest frequency
- Tool guided optimization
  - Temporal restrictions: clock frequency
  - Area or performance optimization
- Optimization by design
  - Series implementation
  - Parallel implementation
  - Pipeline

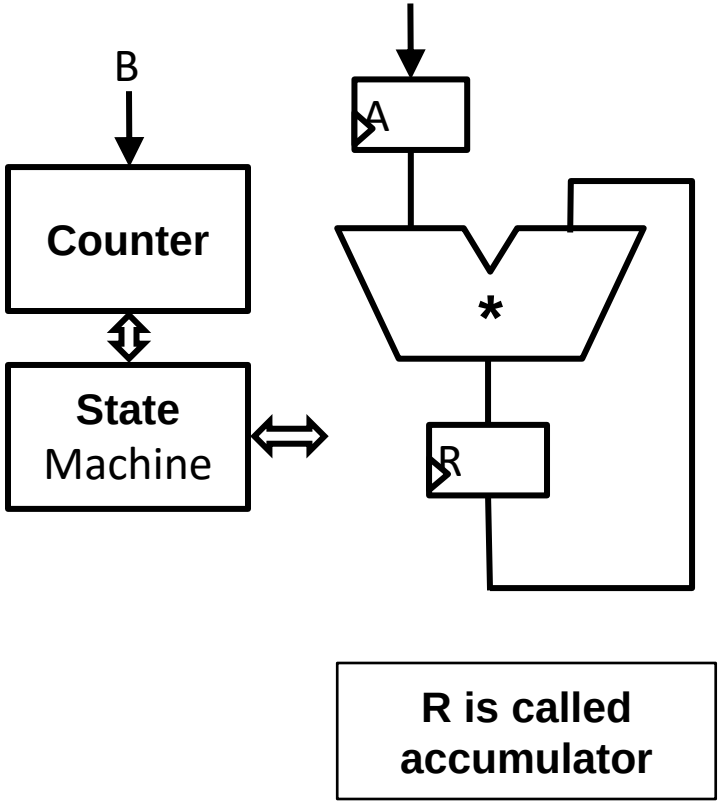
Example: exponentiation circuit

- $R = A^B$ (B three bit: 0 – 7)
- Required operator: multiplier
- Binary codification: $B = b_2 b_1 b_0$

$$A^B = \boldsymbol{A}^{(4b_2+2b_1+b_0)} = \underbrace{\boldsymbol{A}^{4b_2}}_{\substack{A^4 \text{ if } b_2=1 \\ 1 \text{ if } b_2=0}} \cdot \underbrace{\boldsymbol{A}^{2b_1}}_{\substack{A^2 \text{ if } b_1=1 \\ 1 \text{ if } b_1=0}} \cdot \underbrace{\boldsymbol{A}^{b_0}}_{\substack{A \text{ if } b_0=1 \\ 1 \text{ if } b_0=0}}$$

Serial implementation (I)

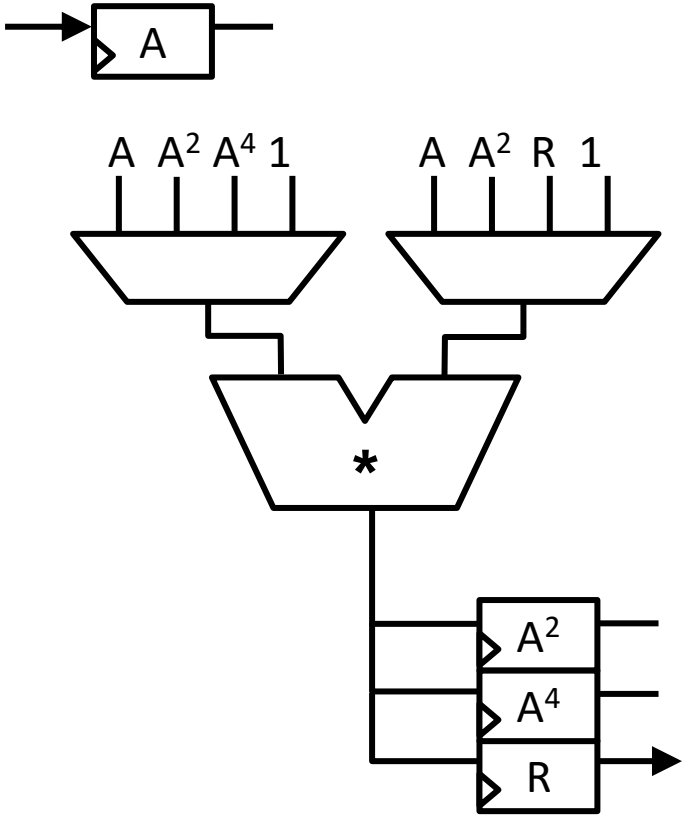
- Multiply A, B times
  - R <= R*A
  - R initialized to 1
- Control needs a counter
- Small area
- High clock frequency
- Variable number of clock cycles



Serial implementation (II)

$$A^B = A^{4b_2} \cdot A^{2b_1} \cdot A^{b_0}$$

1) $A^2 <= A*A$
2) $A^4 <= A^2*A^2$
3) $R <= A^2*A$
4) $R <= A^4*R$



- Small area
- High clock frequency
- 4 clock cycles

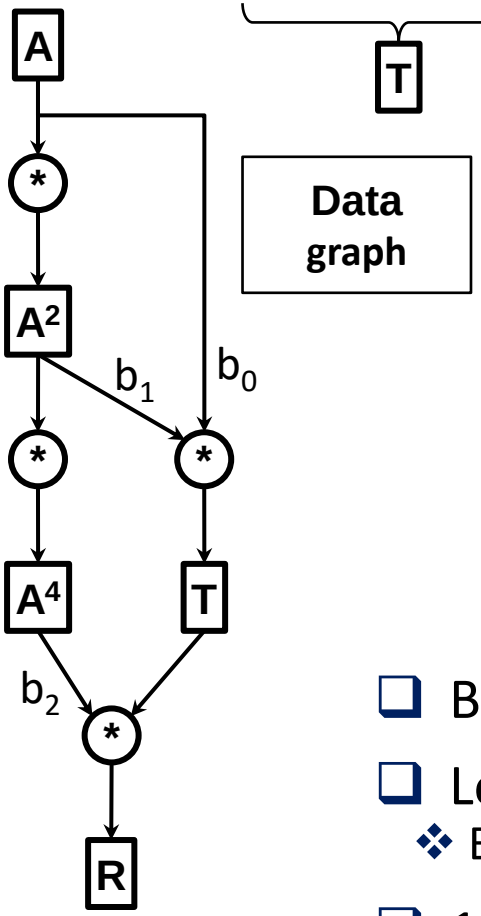
ALU + register file
Typical microprocessor structure

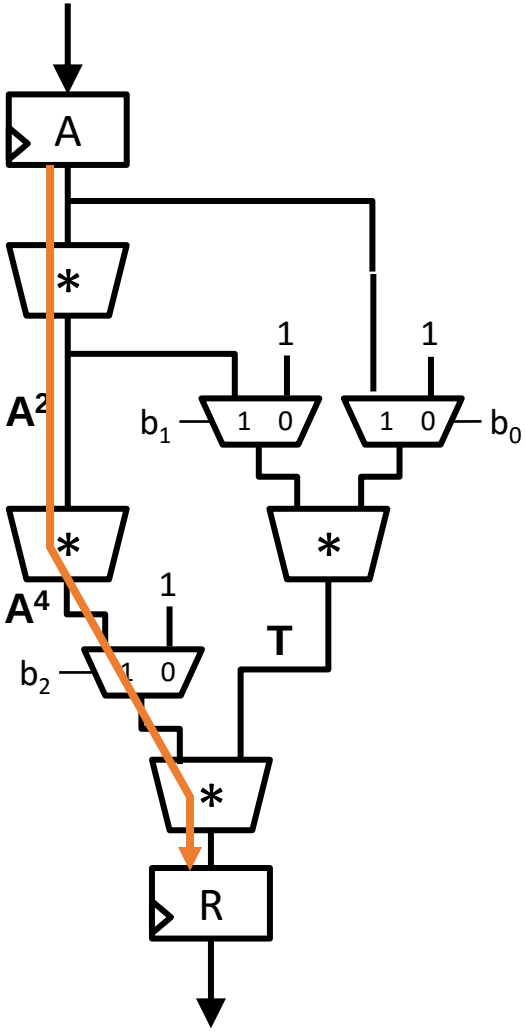
Can you re-order the algorithm to use lower number of registers?

Yes

Parallel implementation

$$A^B = A^{4b_2} \cdot \underbrace{A^{2b_1} \cdot A^{b_0}}_{T}$$

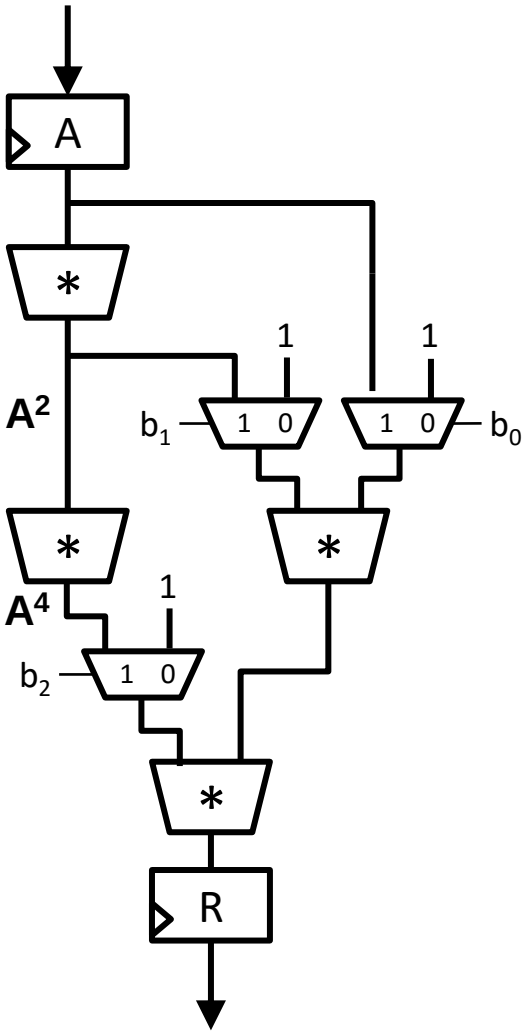




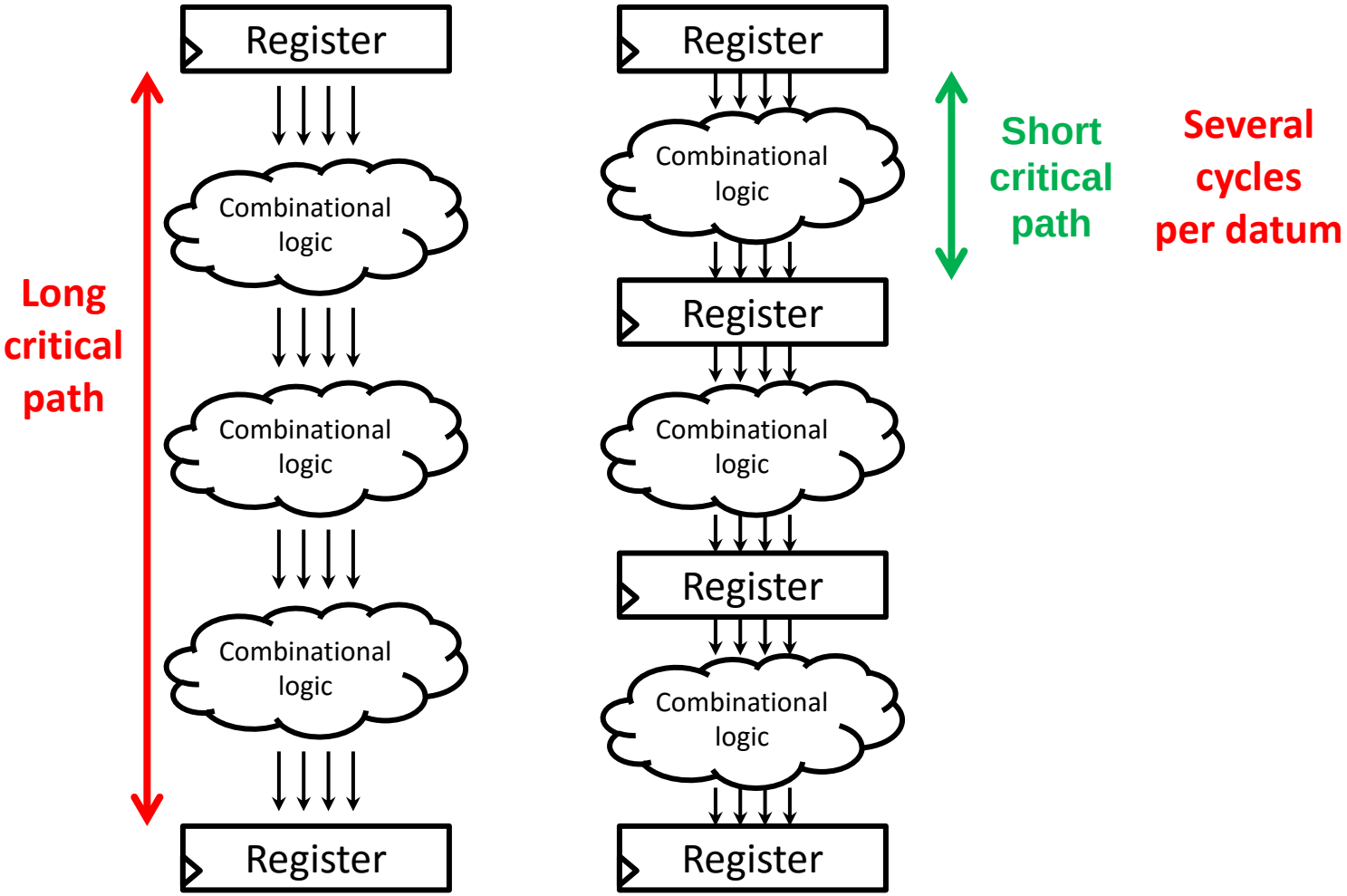
- Big area
- Low clock frequency
  - Big critical path
- 1 clock cycle

Parallel

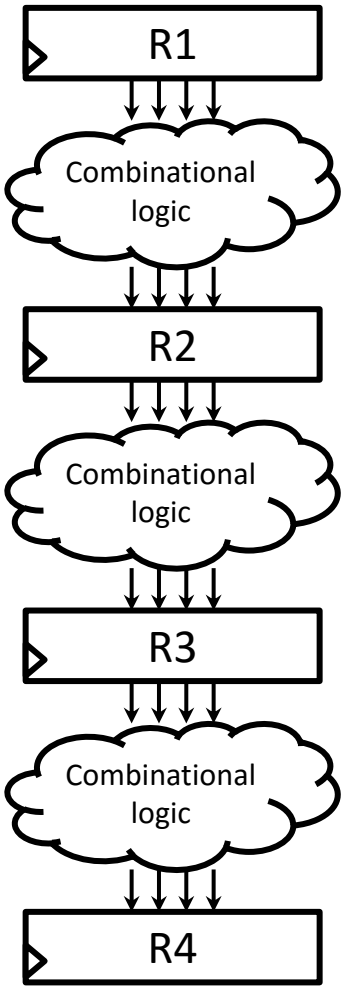
```
process (Reset, Clk)
  variable A2,A4,T: ...
begin
  if Reset = '1' then
    R <= (others => '0');
  elsif Clk'event and Clk='1'
then
    A2 := A*A;
    A4 := A2*A2;
    if b(0)='0' then
      T := 1;
    else
      T := A;
    end if;
    if b(1)='1' then
      T := T*A2;
    end if;
    if b(2)='1' then
      T := T*A4;
    end if;
    R <= T;
  end if;
end process;
```

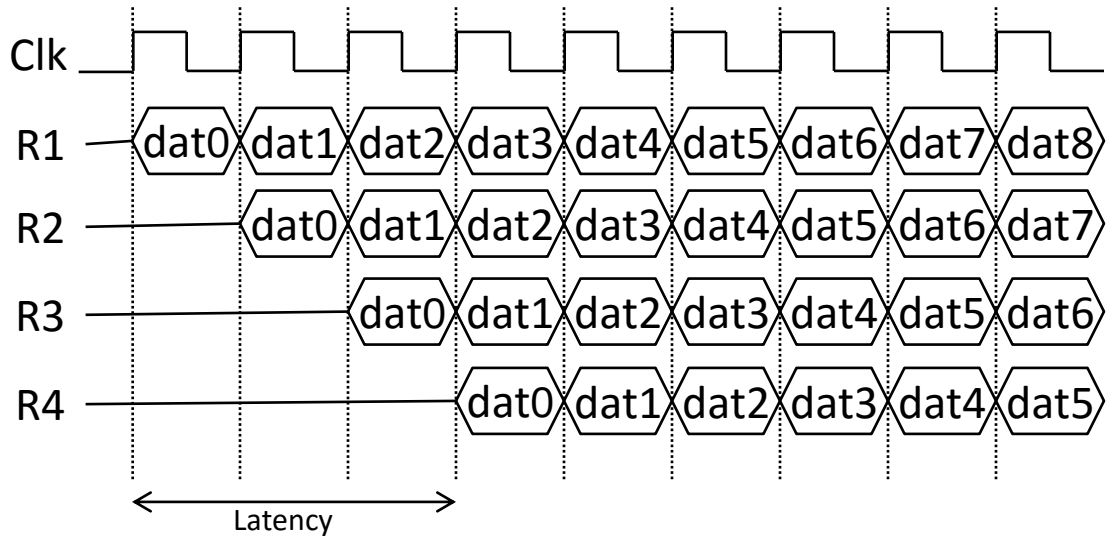


Pipeline


Register
Combinational logic
Combinational logic
Combinational logic
Register
Long critical path
Register
Combinational logic
Register
Combinational logic
Register
Combinational logic
Register
Short critical path
Several cycles per datum

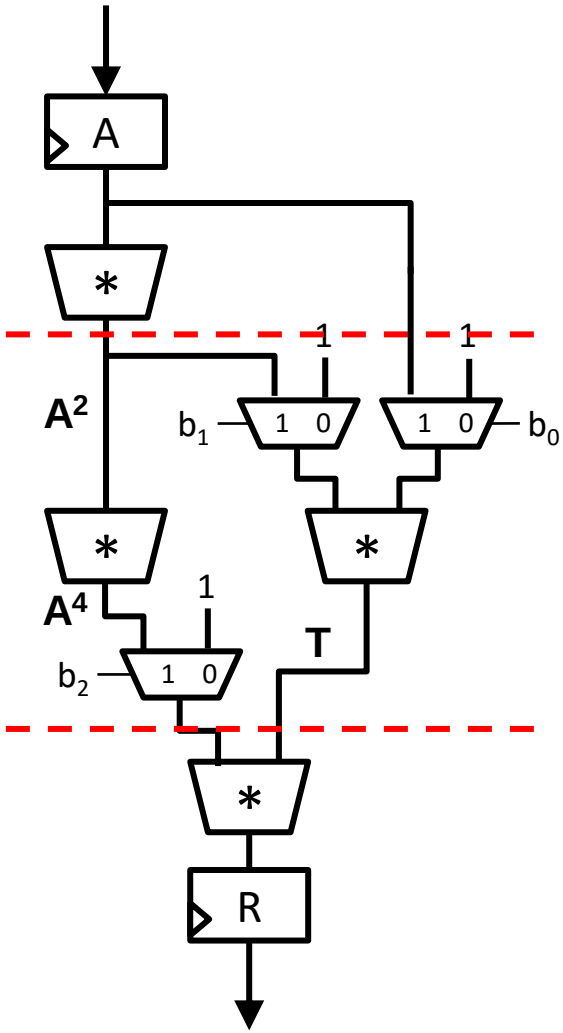
Pipeline





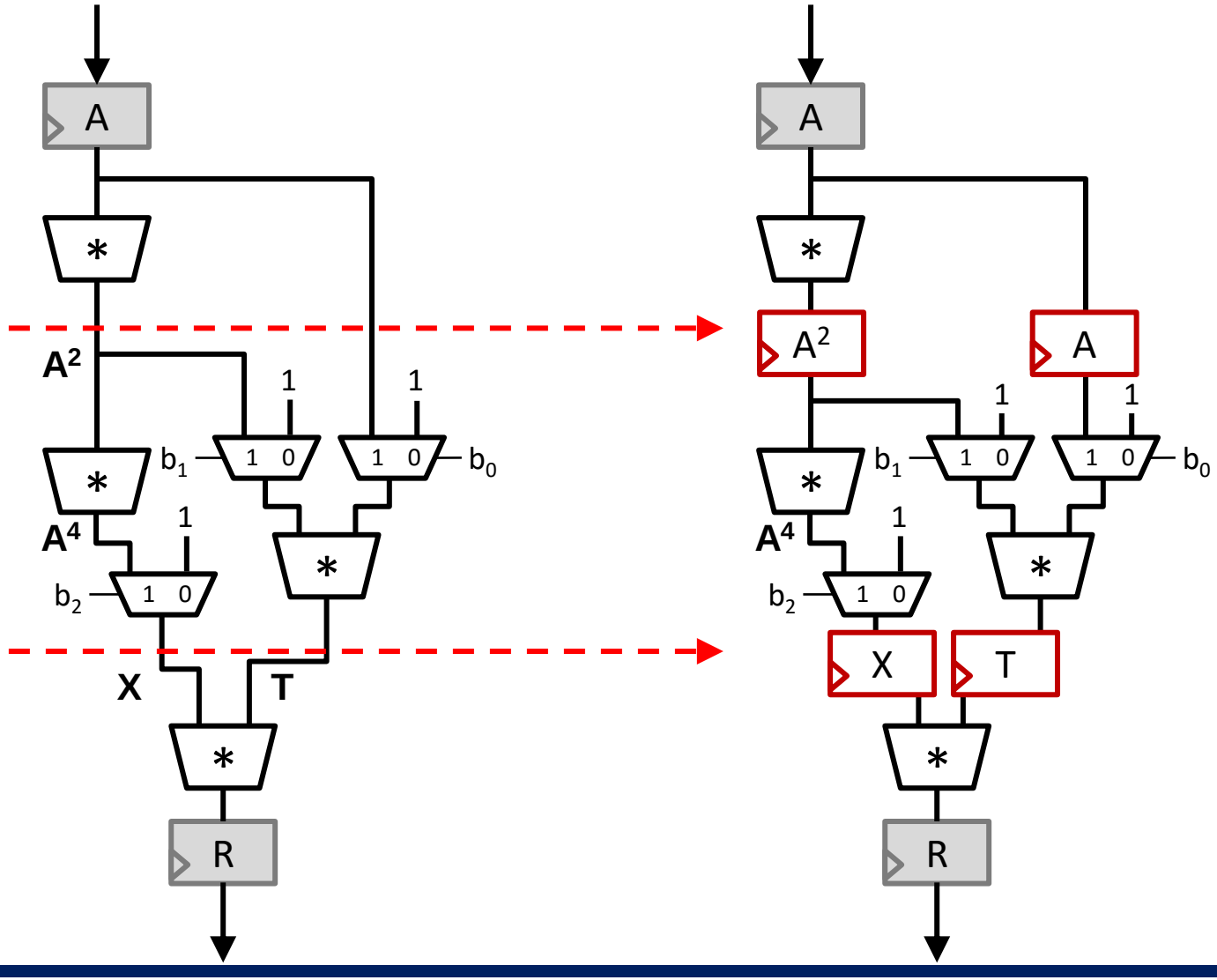
- Biggest area
- High clock frequency
  - Short critical path
- 1 datum every clock cycle
- Latency

Pipeline

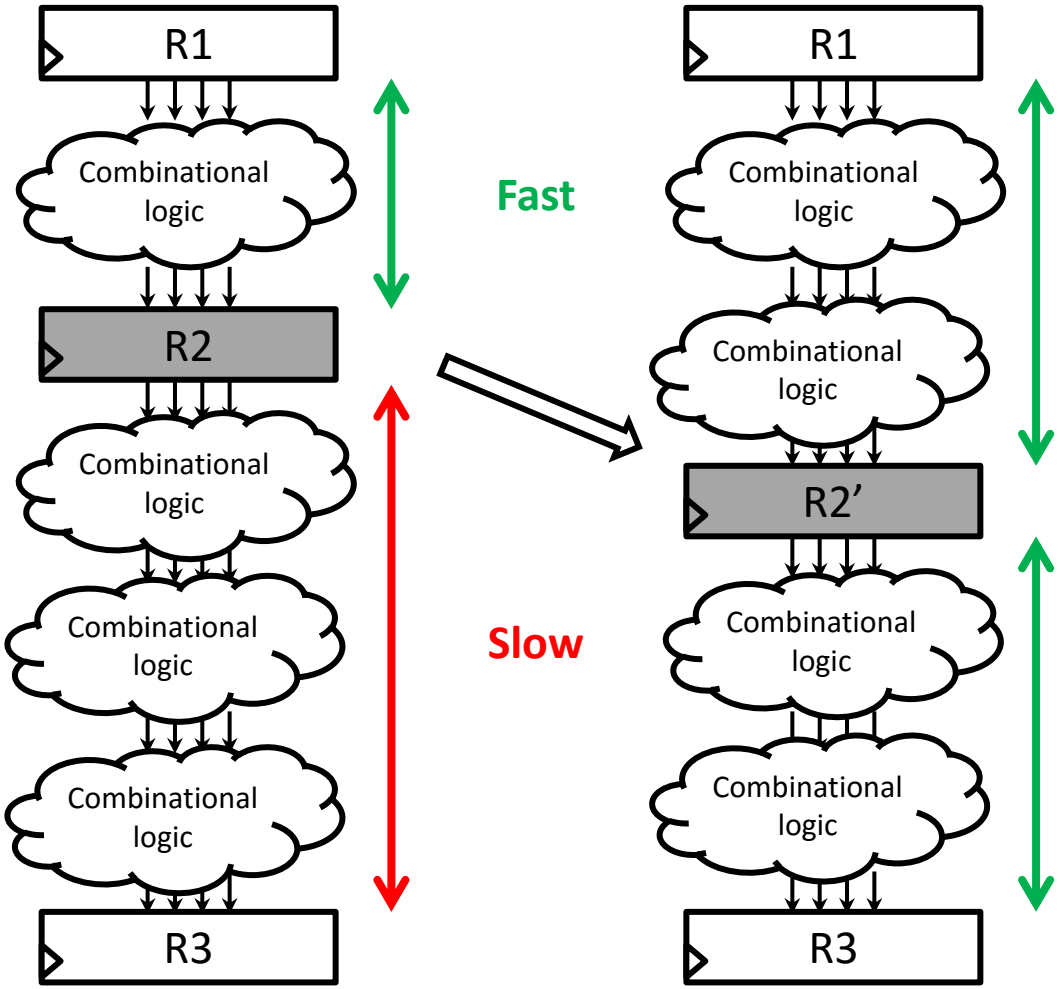


- In order to improve the critical path, the pipes should involve balanced delays
  - One multiplier and one mux
- Minimizing the number of inserted registers
  - Cutting after the multiplexers b1 y b0 requires an additional register

Pipeline

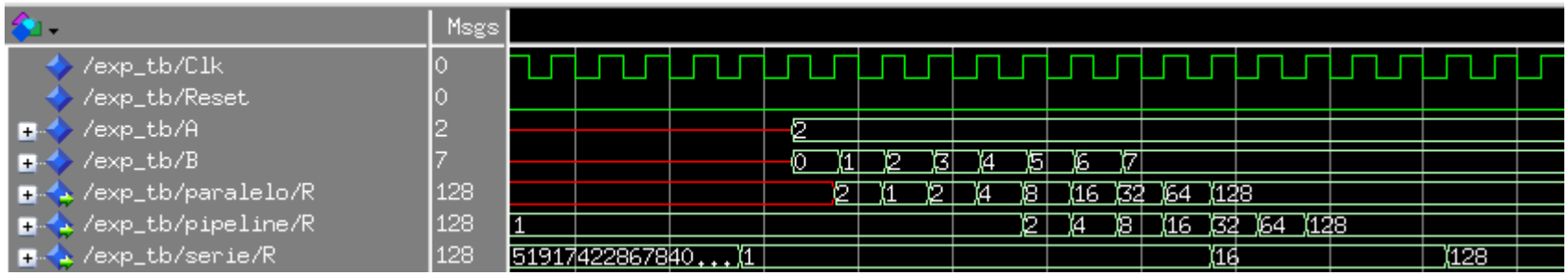


Retiming



- Improve the critical path
- It can be implemented with CAD tools

Exponentiation circuit : simulation



- Latency
- Number of cycles per datum

Exponentiation circuit: synthesis

	Serial	Parallel	Pipeline	Constraints
FF	285	116	315	Synthesis for speed Multiplier with LUTs
LUT	3,421	4,717	4,633	
T_{Clk} (ns)	22.33	30.12	20.03	
Freq (Mhz)	44,78	33.20	49.94	

	Serial	Parallel	Pipeline	Constraints
FF	278	115	290	Synthesis for area Multiplier with LUTs
LUTs	3,399	4,669	4,633	
T_{Clk} (ns)	22.33	30.12	20.025	
Freq (Mhz)	44.78	33.20	49.94	

	Serial	Parallel	Pipeline	Constraints
FF	280	115	294	Synthesis for speed Multipliers auto
LUTs	610	739	675	
MULT18x18	12	19	19	
Tclk (ns)	21.62	27.47	20.025	
Freq (Mhz)	46.24	36.40	49.94	

Comparison

	Serial	Parallel	Pipelined
Area (FF/LUT)	↓↓ 280/3500	↑↑ 115/4700	↑↑↑ 300/4600
Clock Frequency	↑↑ 45 MHz	↓↓ 33 MHz	↑↑↑ 50 MHz
Throughput	↓↓ 9 Mdata/s	↑↑ 33 Mdata/s	↑↑↑ 50 Mdata/s
Latency	Yes (5)	No	Yes (4)

Advised for reducing area

Advised for high performance

Conclusions

- The structure of a digital system at RT level
  - Datapath
  - Control
- Design optimization
  - Area
  - Performance
- Optimizing by design
  - Serial architecture
  - Parallel architecture
  - Pipelined architecture