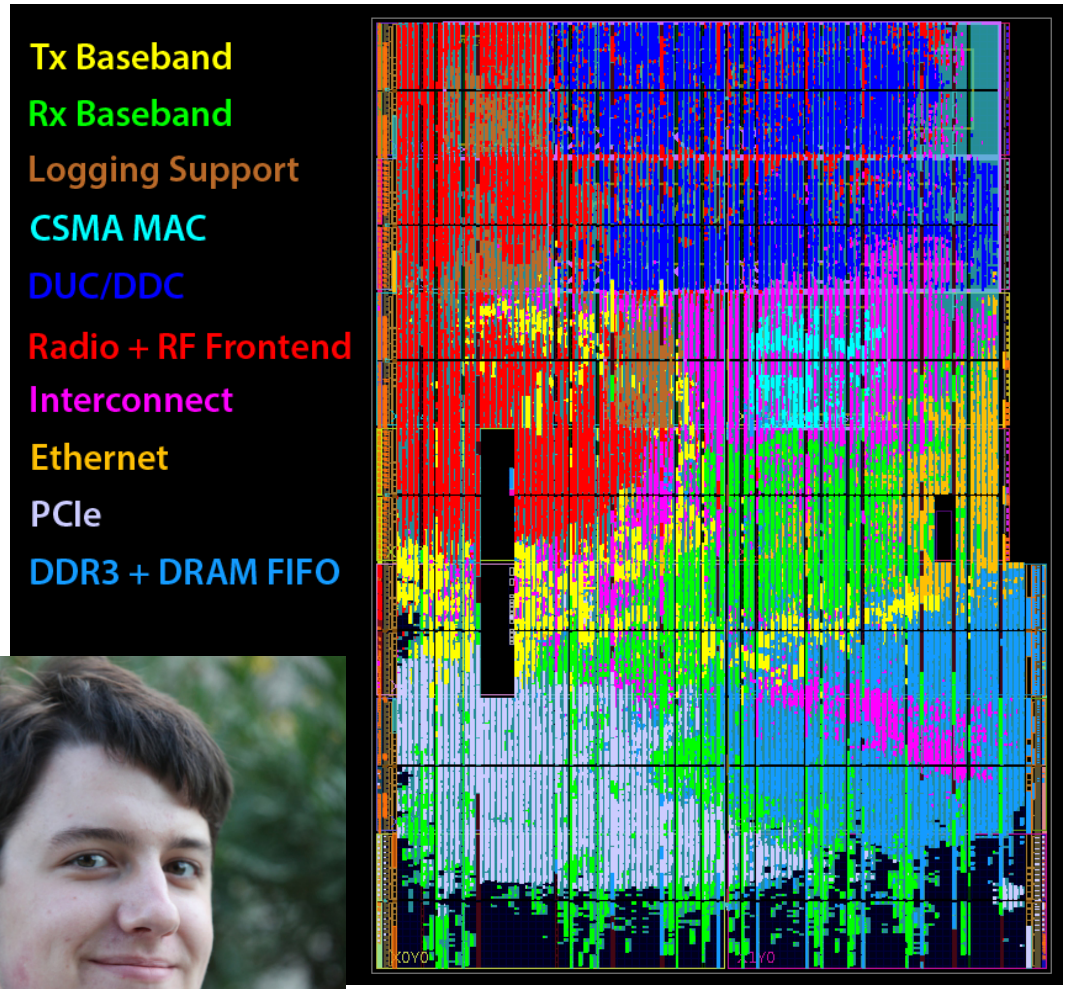
EECS151/251A Discussion

Christopher Yarp

Jan. 25, 2019

About me

- 5th Year Graduate Student
  - Advisor: John Wawrzynek
- Work in the Berkeley Wireless Research Center (BWRC)
- Prior Projects: Implementing Radio Basebands (DSP) in FPGAs
- Current Research: Design Methodologies & Tools for DSP



My Job

- To help you get the most of this course!
  - Running the discussion session
  - Running the FPGA lab sections
  - Answering questions in office hours and on Piazza

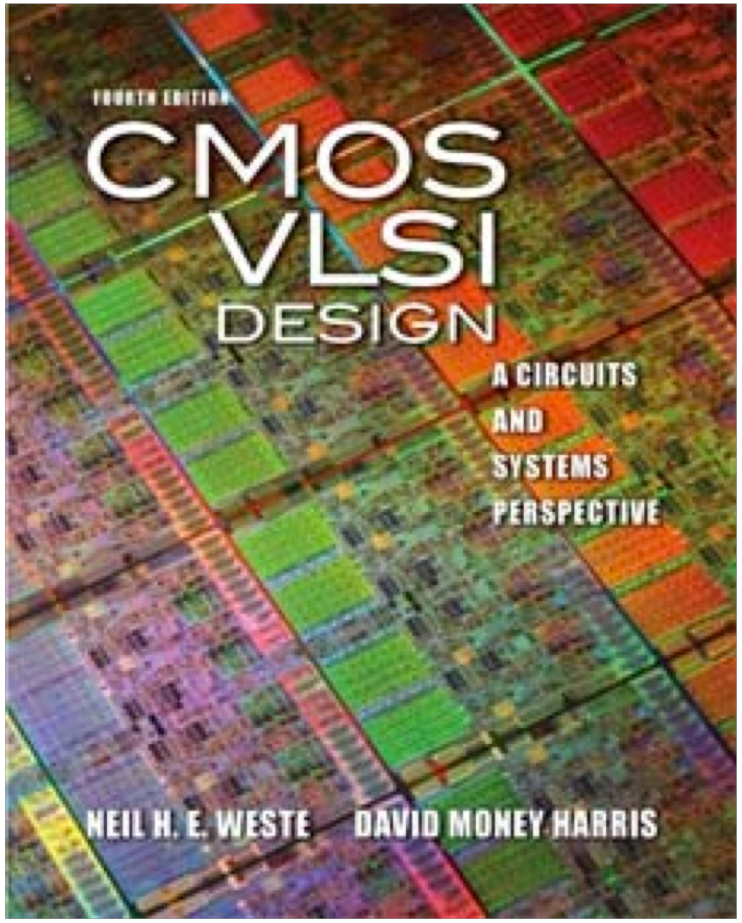
What to expect from discussion

- Review of important concepts from past week's lectures
- Answer your questions!
- More examples

- *Please give me feedback on what is helpful!*

Textbook Resources

- I may, from time to time, reference content from the Weste, Harris book mentioned in lecture
  - CMOS VLSI Design: A Circuits and Systems Perspective, 4th Ed.
- The textbook is not required but does provide additional explanation and examples



Trends in Digital Design

Scaling Laws

Moore's Law

- Number of transistors per ASIC die doubles every 1-2 years
  - Typically fueled by shrinking transistors to increase density
- General view: Moore's Law is coming to an end (or at least slowing)
  - Harder to scale the size of transistors to increase transistor density
  - Cost/transistor is not scaling as well as it used to


EE Times

HOME NEWS PERSPECTIVES DESIGNLINES VIDEOS RADIO EDUCATION

DESIGNLINES | SOC DESIGNLINE

Path to 2 nm May Not Be Worth It

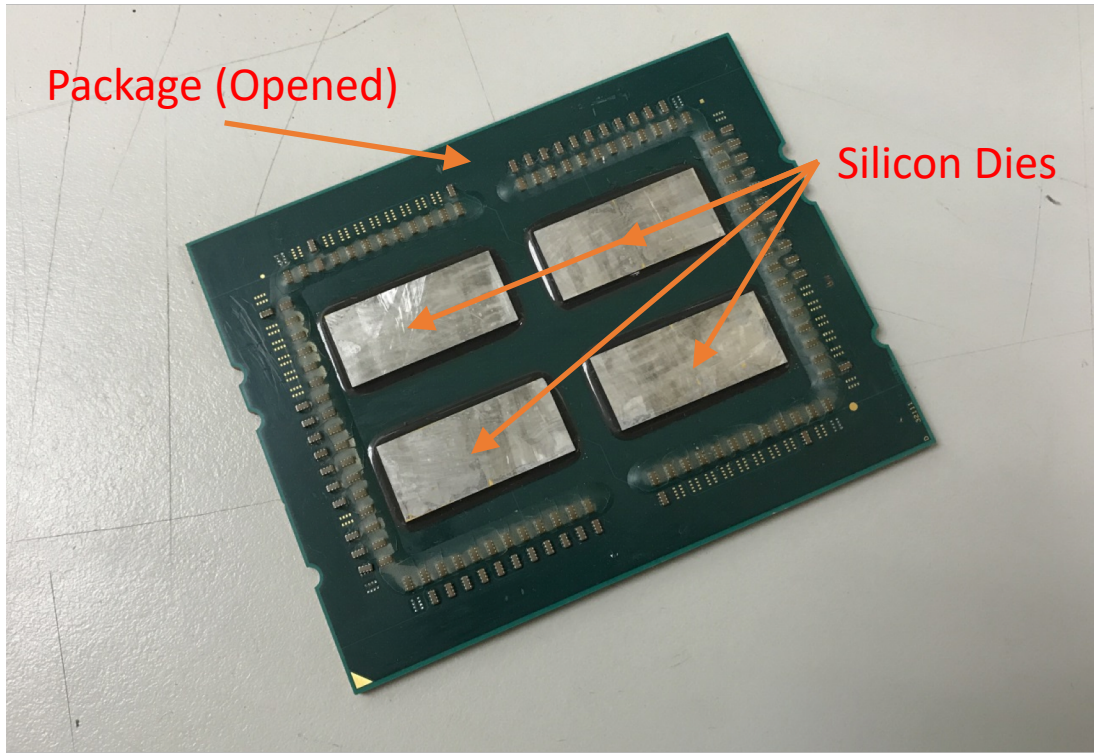
Diminishing returns may evaporate at 5 nm

By Rick Merritt, 03.23.18 5

https://www.eetimes.com/document.asp?doc_id=1333109

Other ways of increasing transistor count

- If it is harder to scale transistor sizes, what can we do?
- Increase the size of the die!
  - Well … we could do this up to a point. Undesirable to make dies much larger
    - Large dies = poor yield = higher cost
- Put more dies in a package!
  - AMD Threadripper (Released)
  - Intel 48-Core Cascade Lake (Announced)



AMD Threadripper “De-lidded”
https://www.extremetech.com/computing/253248-amd-threadripper-delidded-multi-core-surprise-hood

Dennard Scaling & Frequency Scaling

Dennard Scaling

- Voltage can be reduced as transistors are physically scaled down in size
  - Lower power
- Capacitance also scaled down
  - Lower power, higher speed
- Keeps power density constant, linearly improves delay (Weste, Harris 256)

Frequency Scaling

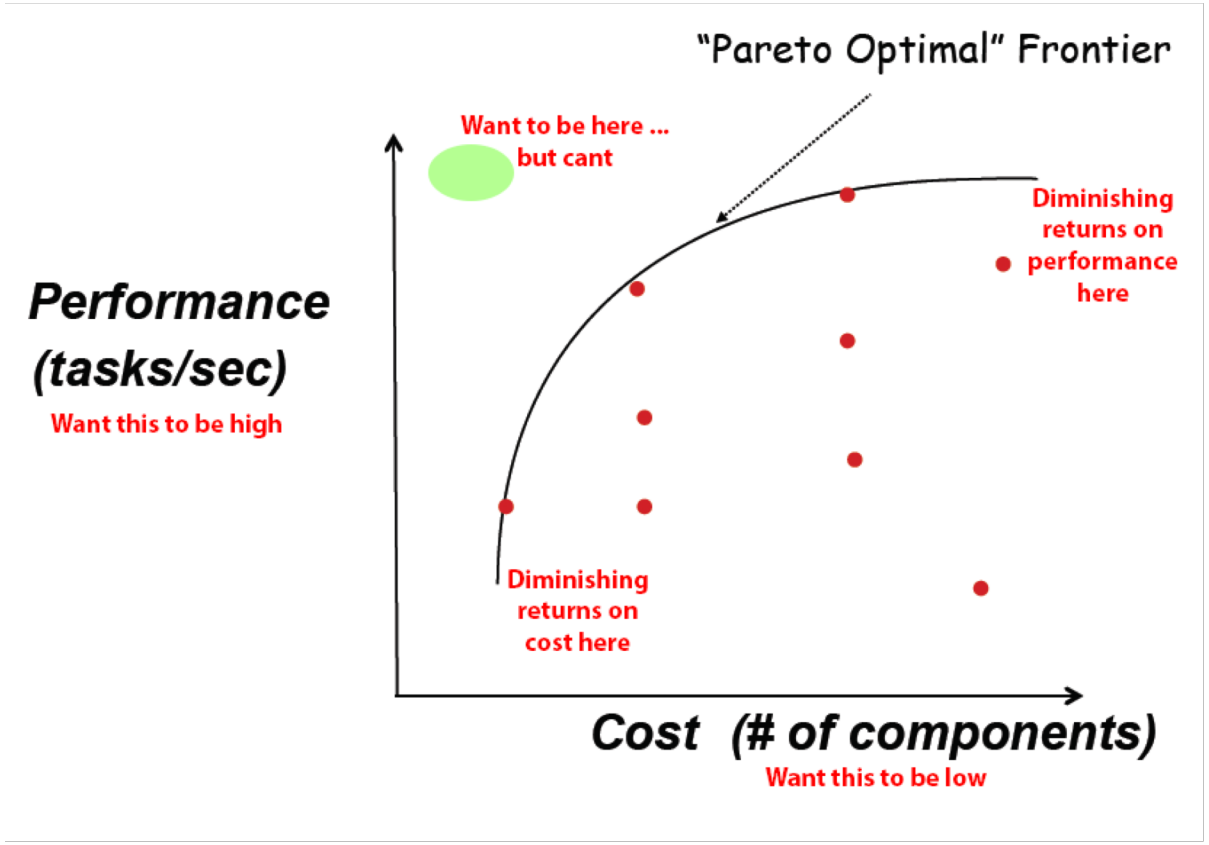
- Keep voltage constant as transistors are scaled (Weste, Harris 256)
  - Delay quadratically decreases (YAY!!)
  - Power density cubically increases (NO!!)
  - Generally stopped around 3-4 GHz
    - Encountered the “power wall” – cannot dissipate that much power

What do these laws mean for performance?

- Moore's Law
  - Performance may increase for a given chip area
    - Relies on parallelism
  - Why not 2x when doubling number of transistors?
    - Limited opportunities for parallelism
    - Overhead
  - Does your 4 core processor work 2x as fast as your last 2 core processor?
- Frequency Scaling:
  - Performance did mostly scale with frequency
    - A 1.6 GHz processor performed approx. 2x better than 800 MHz processor without any change to architecture!
  - Don't really see this today

Design Space, Tradeoffs, and the “Pereto Optimal” Frontier

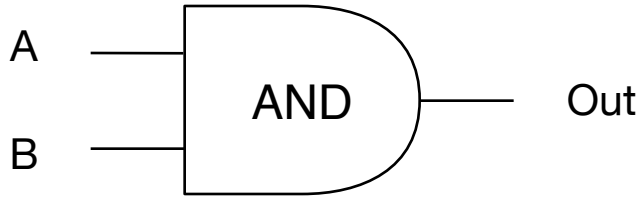
- We often have competing objectives when designing hardware
  - High performance
  - Low power
  - Low cost
- We usually can’t get everything we want - we need to make some tradeoffs
- The “Perato Optimal” frontier represents the edge of the tradeoff space
  - Can’t to go beyond the “Perato Optimal” frontier



Logic Design

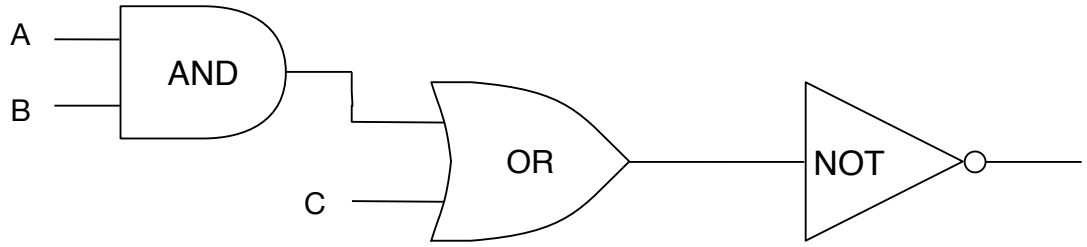
Combinational Logic

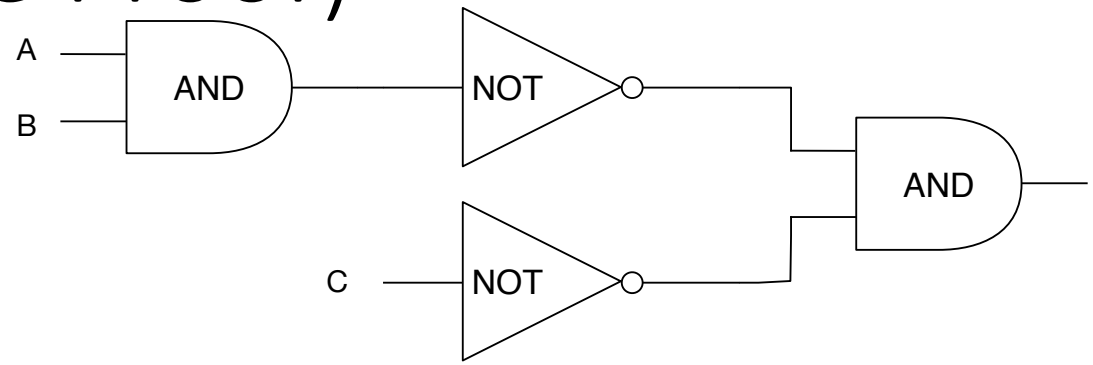
- Logic where the outputs only depend on the current inputs
  - Do not depend on any previous inputs
- Can be expressed using a Truth Table
  - Enumerate all possible inputs
  - Define the outputs



A	B	Out
0	0	0
0	1	0
1	0	0
1	1	1

Proving (or Disproving) Equivalence with Truth Tables (Exhaustive Proof)



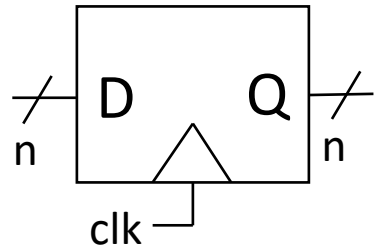


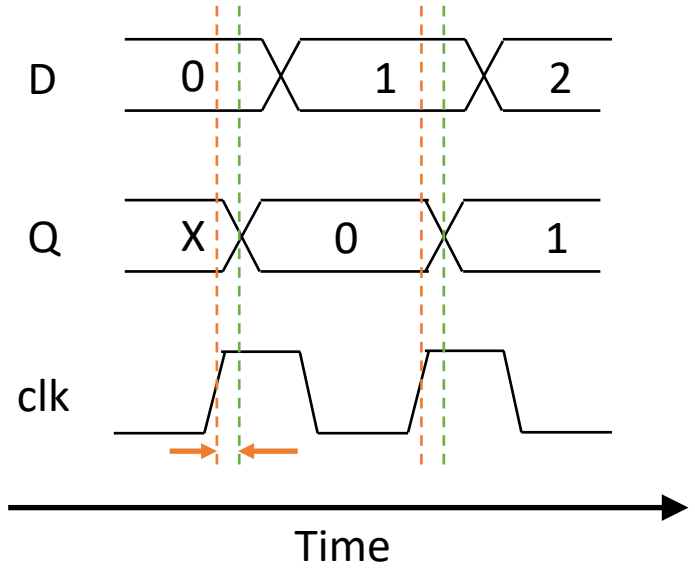
C	B	A	A&&B	(A&&B)||C	Out
0	0	0	0	0	**1**
0	0	1	0	0	**1**
0	1	0	0	0	**1**
0	1	1	1	1	**0**
1	0	0	0	1	**0**
1	0	1	0	1	**0**
1	1	0	0	1	**0**
1	1	1	1	1	**0**

C	B	A	A&&B	!(A&&B)	!C	Out
0	0	0	0	1	1	**1**
0	0	1	0	1	1	**1**
0	1	0	0	1	1	**1**
0	1	1	1	0	1	**0**
1	0	0	0	1	0	**0**
1	0	1	0	1	0	**0**
1	1	0	0	1	0	**0**
1	1	1	1	0	0	**0**

Registers (State Elements)

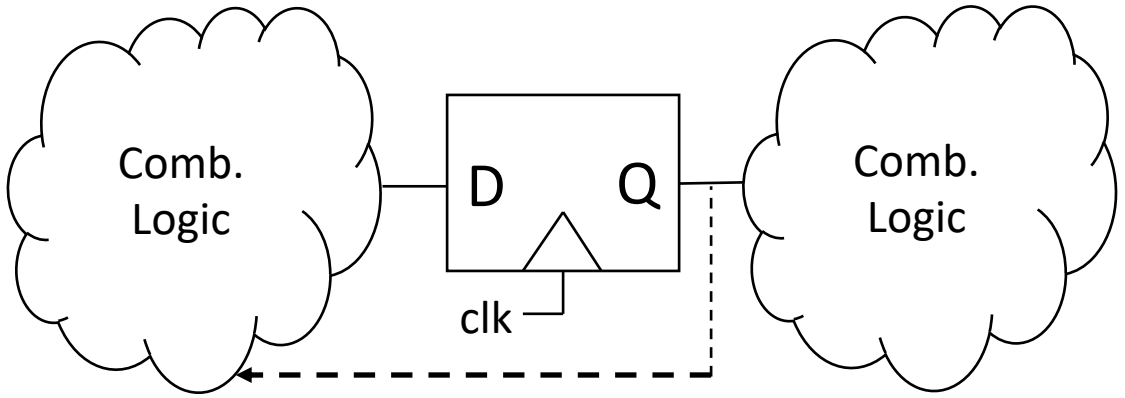
- Combinational logic is great but what about making decisions based on past inputs?
- We need a way to store information!
- Registers (FlipFlops) act as a storage element
  - Move the input to the output at a the 0 to 1 transition of a “load” line
    - There is some delay doing this
  - Hold the output until the next 0 to 1 transition of the “load” line
  - The “load” line is typically connected to the clock (clk)





Register Transfer Level (RTL)

- Can split your design into combinational logic blocks and state elements (sequential logic)



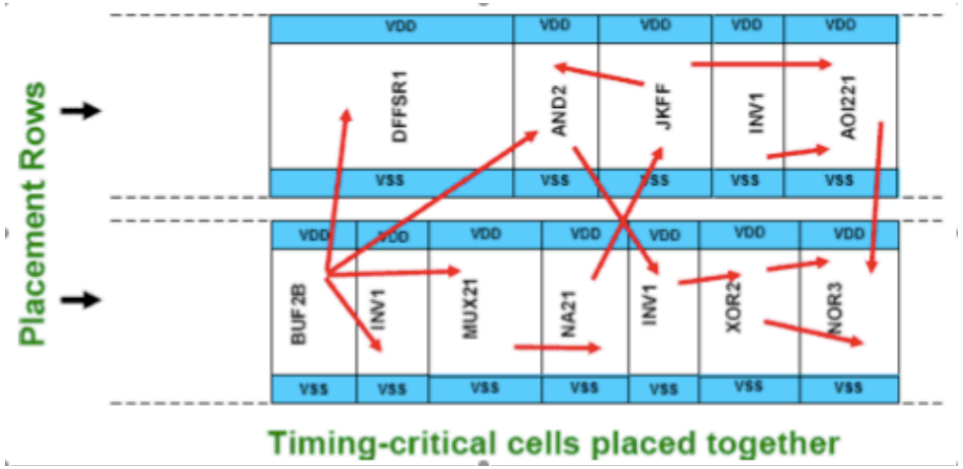
You may even have feedback from registers!

- This abstraction covers all digital logic design
- Does *not* cover every electronic circuit you could make
  - Combinational loops are not allowed
    - When one of the inputs to your combinational logic is the output
      - Ex: inverter wired to itself
- This abstraction helps us when using HDL languages like Verilog!
  - Describe combinational sections
  - Describe sequential (state) sections

ASICs vs. FPGAs

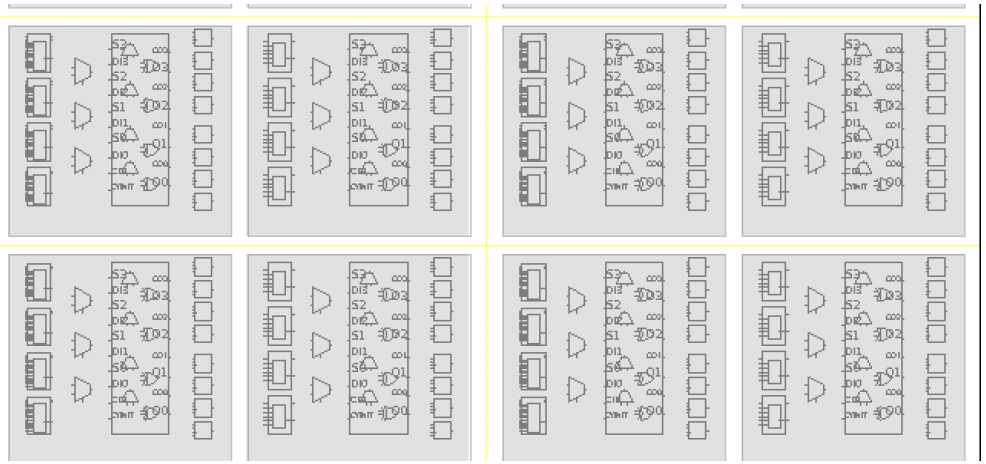
ASIC vs FPGAs

ASIC



- Flexibility in placing standard cells during design
- Can place exactly the cells you need

FPGA



- Arrays of General Logic Resources
  - Lookup Tables
  - Registers
  - Multiplexers
  - Memory
  - DSP Blocks
  - Interconnect Network
- Programming the FPGA configures these general resources to implement your HW design

ASIC vs FPGAs

ASIC

- No unused logic -> you placed exactly what you needed
- Inflexible after manufacturing -> only configurability is what you designed in
- Design iteration time: months - years
- High Fixed Manufacturing Cost (NRE – Non Reoccurring Engineering)
  - Designing and verifying
    - Limited flexibility -> better get the right design
    - Expensive to manufacture again -> avoid needing to fix things and manufacture again
  - Mask production (used during manufacturing)
  - Setting up the production line
- Low Incremental Manufacturing Cost
  - Once the design is done and the production line is set up, producing more chips is not very expensive
- Better sell a lot of chips to amortize the NRE!

FPGA

- Generality -> unused logic in some applications
- Remains Flexible -> can change design later (reprogram FPGA)
- Design iteration time: minutes - hours
- Medium Fixed Cost (NRE)
  - Still HW design
    - More things to consider than SW
    - Relatively slow design tools
- Medium Incremental Cost
  - FPGAs are general -> need larger die area to accommodate additional logic -> more cost/die
- Good for lower volumes or when reconfigurability is required

Questions