HPES
High Performance Embedded Systems

EEE4120F

University of Cape Town • iYunivesithi yaseKapa • Universiteit van Kaapstad

High Performance Embedded Systems

Lecture 18

FPGA & CPU Performance Comparison

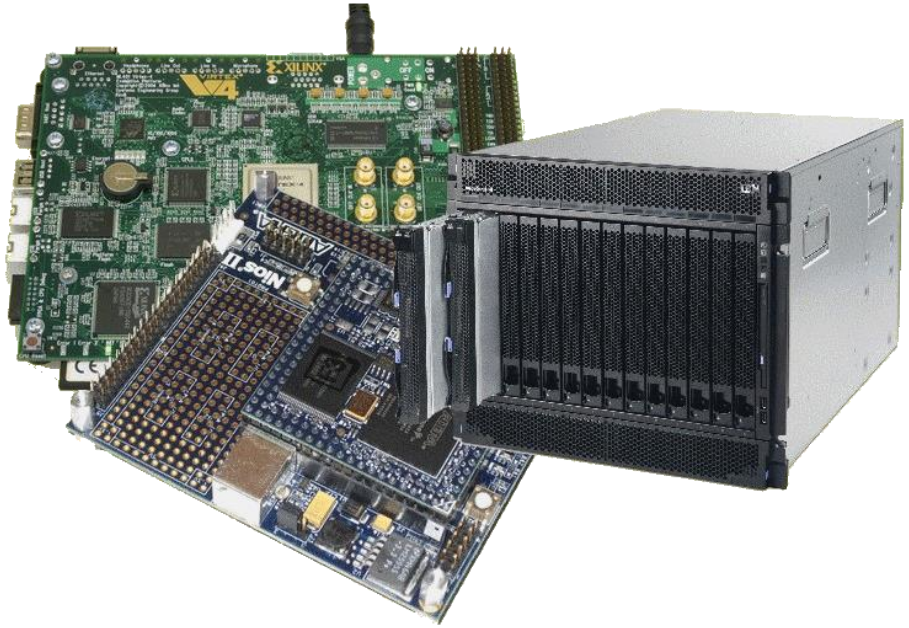
FPGA Families

Lecturer:
Simon Winberg

Click on speaker icon to hear narration

Lecture Overview

- FPGA performance evaluation
- FPGA vs CPU performance
- FPGA families
- YODA issues



Evaluating Performance

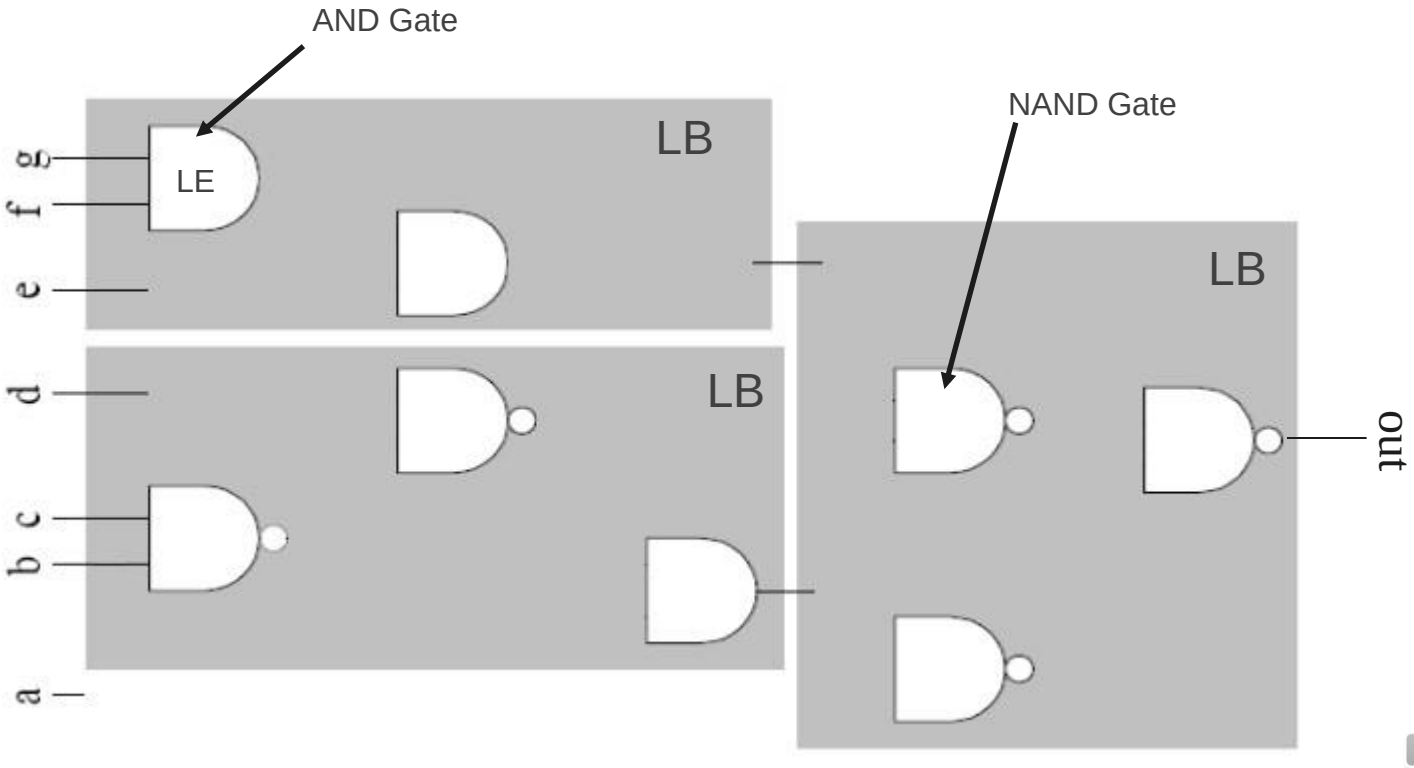
Evaluating synthesis (simplified) of an FPGA design

… a useful exercise, which you could be tested on, and which is more relevant to PLDs…
but which is unlikely to be used for even remotely complex FPGAs designs.

HDL to FPGA execution & LE cost (1)

In order to implement an HDL design, the design need to be decomposed and mapped to the physical LBs on the FPGA and the interconnects need to be appropriately configured.

In this explanatory scenario, we are using a very basic FPGA that has three logic blocks (LBs), each logic block having only a few logic elements. The logic elements in this example are just AND gates and OR gates.



HDL to FPGA execution & LE cost (2)

Let us consider that we want to map the following example logic functions to the FPGA. In other words, we want to figure out how the FPGA will be set up so that it will complete the desired operation when input is applied.

Example logic functions are:

x = AND(e,f,g)

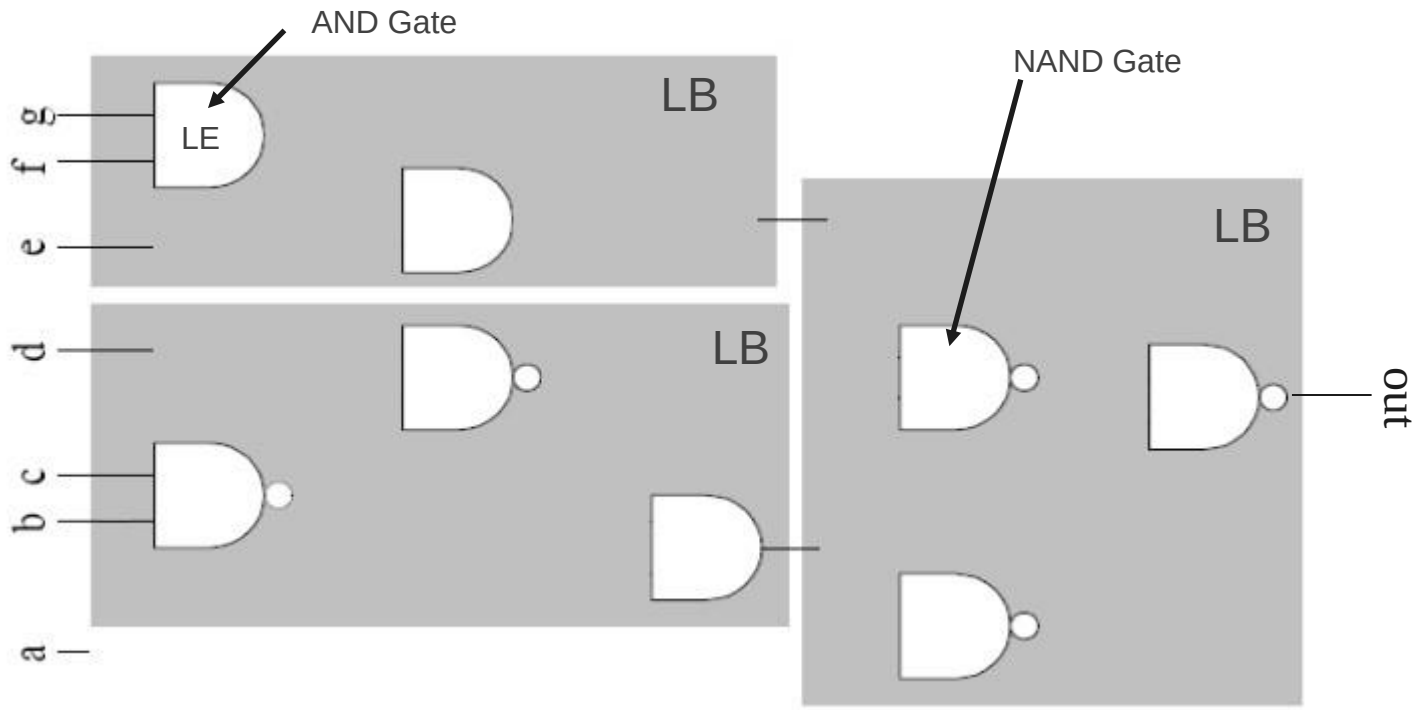
y = AND(b,NAND(NAND(b,c),d))

out = NAND((NAND(x,y),NAND(a,y))

Note that x and y are intermediate wires

(1)

(2)



HDL to FPGA execution & LE cost (3)

Example logic functions are:

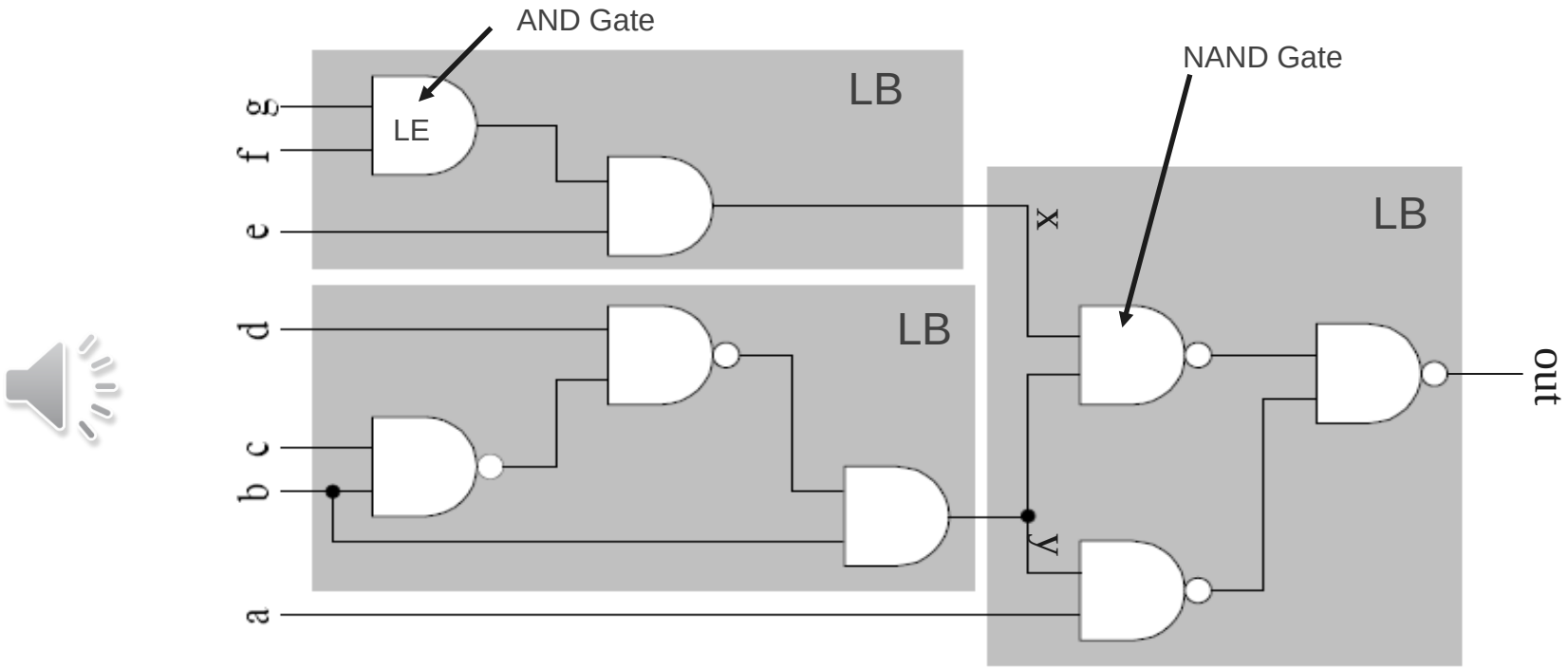
x = AND(e,f,g)

y = AND(b,NAND(NAND(b,c),d))

out = NAND((NAND(x,y),NAND(a,y))

Note that x and y are intermediate wires

This is a view of how this logic circuit would get mapped to contain the required logic function (or combinational logic circuit). It happens that there is just enough logic elements to satisfy this design.

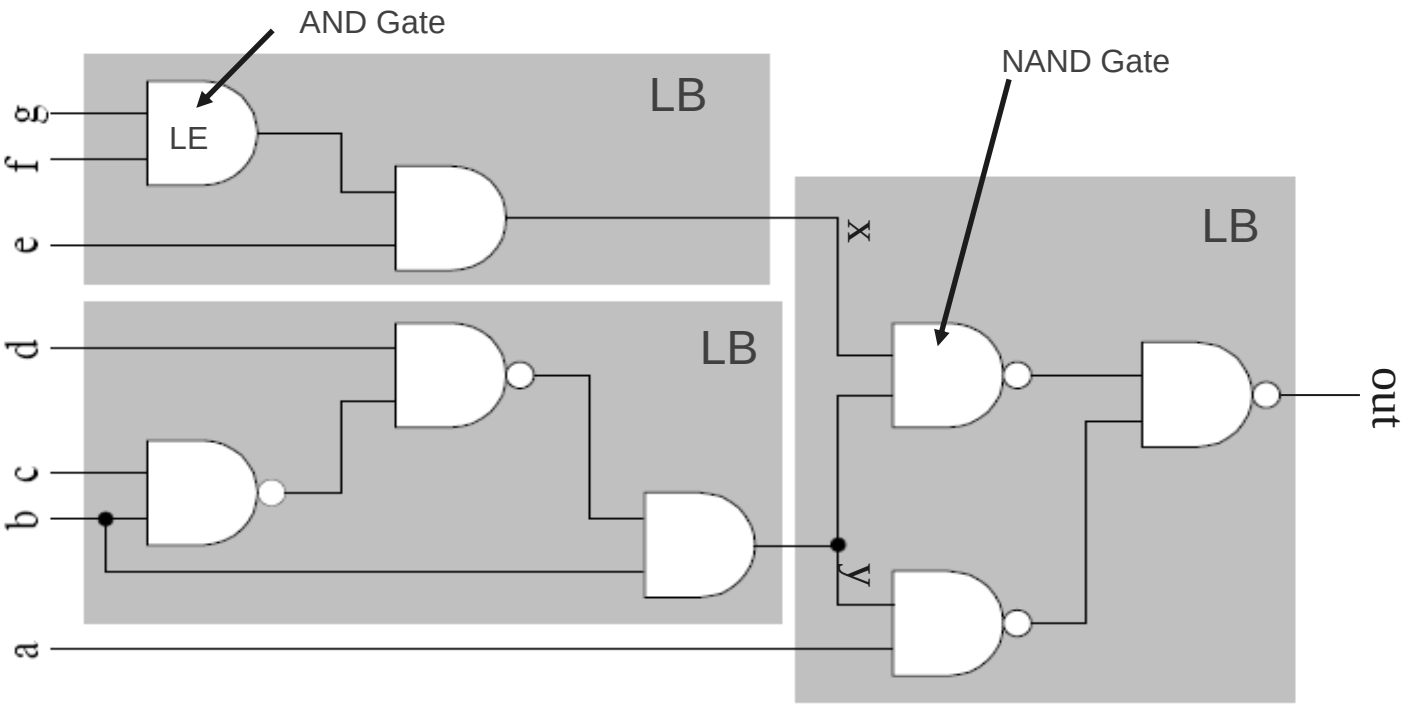


But what is the execution time cost of this circuit?

You can listen to some of my discussion about working this out by listening to the next sound clip …

Otherwise,

if you're impatient and want to just get into the process of manually working out performance of a simple circuit, then skip to the next slide.



LE cost = 8 and LB cost = 3

HDL to FPGA execution & LE cost

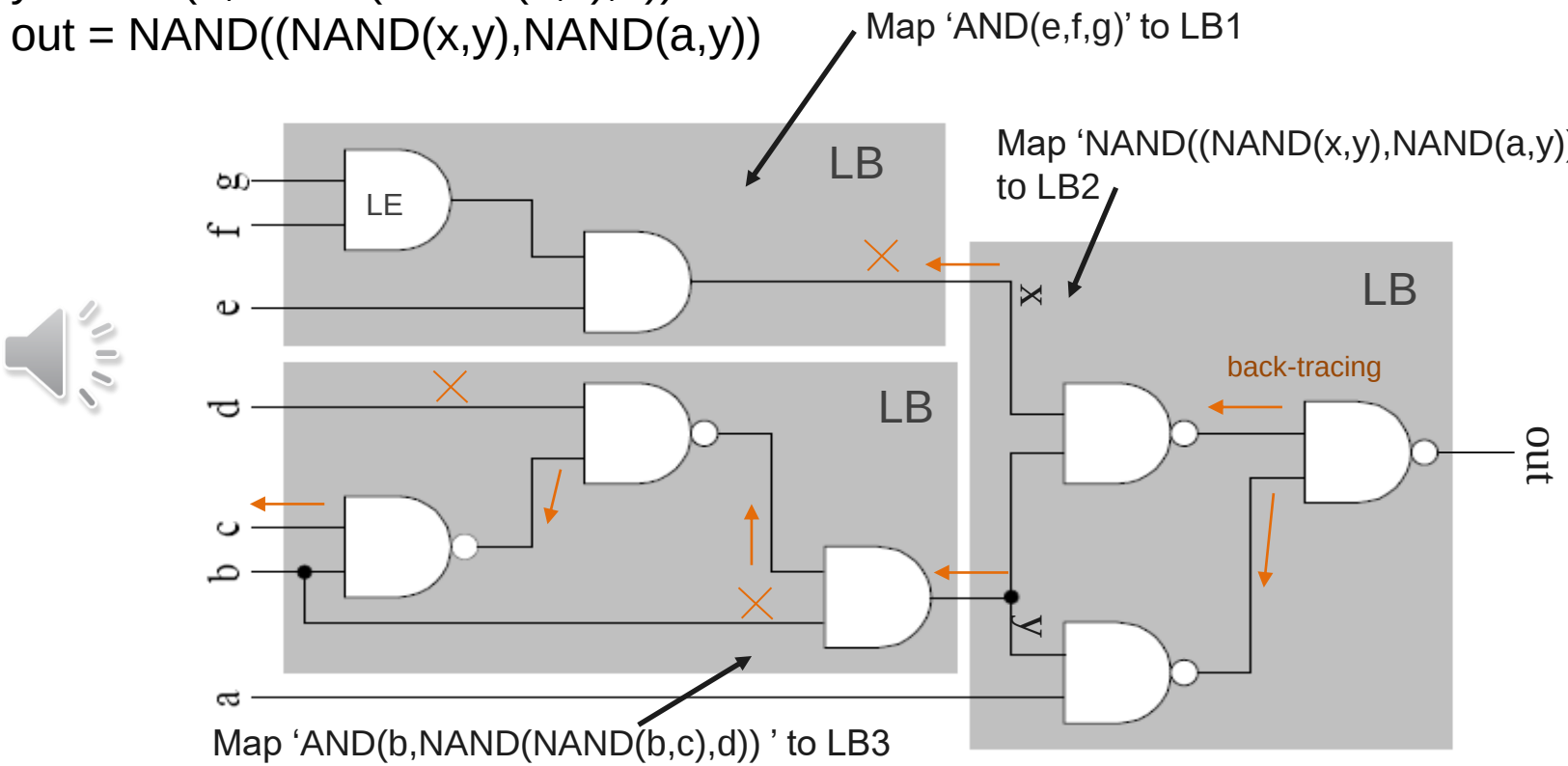
In order to implement an HDL design, the design need to be decomposed and mapped to the physical LBs on the FPGA and the interconnects need to be appropriately configured.

Example:

x = AND(e,f,g)
y = AND(b,NAND(NAND(b,c),d))
out = NAND((NAND(x,y),NAND(a,y))

Animation showing back-tracing
Applied to determine execution time.

In this example, LE are the gates and LBs are the grey blocks they are within.



Costing: 3 LBs, 8 LEs (assuming LBs have LEs that are AND or NAND gates)

Timing calculations

- The previous slide didn't show whether the connections were synchronized (i.e., a shared clock) or asynchronous – since they are all logic gates and, no clocks imply it is probably asynchronous.
- Determining the timing constraints for synchronous configurations are generally easier. Because everything is related to the clock speed. Still, you need to keep in mind cascading calculations.
- For asynchronous use, the implementation could run faster. But can also become a more complicated design. And make it more difficult to work out the timing…

Async Timing calculations

- Keep in mind that the propagation delays for the various gates / LUTs may be different. In the previous example circuit we could assume each AND takes 6ns to stabilise, and the NANDs 10ns to stabilise.
- So, time taken to compute output *out* is =

MAX OF (time to compute x, time to compute y) + 2x10ns

= (2x10ns+6ns) + 20ns = **46ns** = pretty fast!! *Or is it??*

Compared to a 1GHz CPU using just registers (and no mem access)?

Try this calculation for yourself ...

(assume each instruction takes on avg. 3 clocks due to pipeline, data dependencies, etc, as worst case performance on a RISC processor)

Comparing to CPU speed

CPU running at 1GHz → each clock 1ns period
Assume each instruction takes ~ 3 clocks each due to pipeline etc
CODE:

```
int doit ( unsigned a, b, c, d, e, f, g ) {
 unsigned x = AND(e,f,g);
 unsigned y = AND(b,NAND(NAND(b,c),d))
 out = NAND((NAND(x,y),NAND(a,y))
  return out;
}
```

But some of these Can't be done as just 1 RISC instruction.

```
unsigned t1 = AND(e,f);
unsigned x = AND(t1,g);
unsigned t1 = NAND(b,c)
unsigned t2 = NAND(t1,d)
unsigned y = AND(b,t2)
t1 = NAND(x,y)
t2 = NAND(a,y)
out = NAND(t1,t2)
```

→ 1 instruction, i.e. AND *t1*,*e*,*f*

in all 8 instructions → 8 x 3 clocks ea.
= **24 ns** **(**assuming all registers pre-loaded)
A speed-up of **1.92** over the FPGA case

Digital Clock Manager (DCM) blocks

- An important element included in FPGA designs nowadays are Digital Clock Manager (DCM) blocks.
- These are used to eliminate* clock distribution delays.
- They can also increase or decrease the frequency of the clock.
- We will look at PLLs later in the course.

* or at least greatly reduce these clock distribution delays

FPGA Families

EEE4120F

Providing a broad variety of different performance, costing, size and tolerance options.

The FPGA Manufacturers

- The 'Big 2' (most commonly used)
  - Xilinx, now owned by AMD, around 2984 employees (in Xilinx division)
  - Intel FPGA (i.e. Intel's acquisition of Altera) (guessing from, Altera stats) around 2500 employees*
- The others pretty big ones…
  - Actel (Microsemi Corp) – around 2200 employees*
  - Lattice Semiconductor Corp – around 700 employees*

* These stats are a couple of years old.

Audio annotations end at this point

Please look over the following slides to become a bit more familiar with these FPGA manufactures, and types of FPGAs that are available.

Image source: Max Pixel CC0
https://www.maxpixel.net/Memo-Paper-Sticky-Note-Post-it-Office-Note-150262

About the FPGA Families

- Xilinx
  - Focusing on high performance and high capacity
    - Vertex family (such as Vertex 7)
    - Virtex UltraScale+
      - Extremely (some might say insanely) high-performance
      - 9 million logic cells, up to 1.5 terabits/sec DDR4 bandwidth and up to 4.5 terabits/sec transceiver bandwidth
  - Provides lower-cost options with high capacity (e.g. Spartan 6 family)
  - Range of variations, e.g. low power options, economy (lower capacity) models.

Note that the top performance FPGA changes over time and is not necessarily consistently one or other of the manufacturers

About the FPGA Families

- Altera
  - Stratix: higher performance and density models (e.g. Startix-10)
  - Arria: mid-range, lower-power, but also lower performance and denisity compared to Stratix.
  - Cyclone: lowest cost option, also aimed at low power, cost sensitive and mobile applications

About the FPGA Families

- Actel
  - Focuses on providing the lowest power, and widest range of small packages
  - IGLOO : low power, small footprint
  - SmartFuson : Mixed FPGA and ARM processor
  - RTAX/RTSX : radiation tolerant and very high reliability.

About the FPGA Families

- Lattice
  - Range of options (low power; high performance; small package)
  - Own specialized development tools
  - (of these four, this one is the only firms not in California; they are currently in Oregon)

About the FPGA Families

- Others
  - Achronix – focusing on building the fastest FPGAs (not necessarily highest capacity)
  - Tabula – unique FPGA technology 'SpaceTime', focusing on highest capacity and memory capabilities

Memory jogger…

Q: Name a high-capacity FPGA family.

A: Xilinx Vertex (e.g. ver 7+) / Altera Stratix (ver 10+)

Q: Which of the following is a FPGA manufacturer ?

(a) Acrobatics

(b) Geometrix

(c) Achronix

Note that the producers are constantly bringing out new versions so this slide may get stale quite quickly.

End of Lecture

Onwards to distributed and shared memory architecture models …

References & Acknowledgements

- References sources used include:
  - Todman, Timothy J., et al. "Reconfigurable computing: architectures and design methods." Computers and Digital Techniques, IEE Proceedings-. Vol. 152. No. 2. IET, 2005.
  - Stavinov, Evgeni. *100 Power Tips for FPGA Designers*. Evgeni Stavinov, 2011.
- Acknowledgement
  - Thanks to John-Philip Taylor (TA) for time taken to review slides and spotting typos and inaccuracies.

Disclaimers and copyright/licensing details

I have tried to follow the correct practices concerning copyright and licensing of material, particularly image sources that have been used in this presentation. I have put much effort into trying to make this material open access so that it can be of benefit to others in their teaching and learning practice. Any mistakes or omissions with regards to these issues I will correct when notified. To the best of my understanding the material in these slides can be shared according to the Creative Commons "Attribution-ShareAlike 4.0 International (CC BY-SA 4.0)" license, and that is why I selected that license to apply to this presentation (it's not because I particularly want my slides referenced but more to acknowledge the sources and generosity of others who have provided free material such as the images I have used).

Image sources:

man working on laptop – flickr

measuring tape – Wikimedia Open Commons https://commons.wikimedia.org (public domain)

scroll, video reel – Pixabay http://pixabay.com/ (public domain)

References: Verilog code adapted from
http://www.asic-world.com/examples/verilog