Multipliers, Iterative Algorithms, Negative Numbers

Leveraging Adders for a Greater Purpose

Akilesh Praveen & Ashwath Krishnan

UMD

January 23, 2020

Agenda

1. Announcements
   - Project 3
2. Multipliers
   - Multiplication in Binary
   - Limitations
3. Iterative Algorithms
   - Iterative Algorithms

Announcements

Project 3

- Project 3's been released, and the name of the game is multipliers.
- You'll find everything you need under the 'week 4' section on the course website
- Today's lecture will give you all the background knowledge that you need to know about implementing a multiplier
- More info to come at the end of lecture

Some Reminders

- As the projects become more involved, we'd like to remind you to reach out and attend office hours if you are struggling.
- Additionally, here's a reminder that all projects can be group projects, but you'll have to let us know **4 days** prior to the project's due date if you'll be working in a group. (Max size of 3)

Multipliers

A Question

How does multiplication work in binary?

Multiplication Table

- Multiplication works the same way in binary as it does in base-10!
- Just like single digit multiplication in base-10 goes up to 9×9, single digit multiplication in binary goes up to 1×1
- Here's the basic multiplication table for binary:

	0	**1**
0	0	0
1	0	1

Multiplication

- Before you get very excited, this is only a small bit of what we're after.
- Now, we know how to multiply one-bit numbers (Yay!)
  - note that `AND` is all you need to represent this operation
- In binary, multiplication is the same as it's always been for us in decimal
- Let's go through an example now

Multiplication

- Again, we will look to the 'long' way of doing this computation for inspiration

$$\begin{array}{r} 1101 \\ \times\, 1011 \\ \hline \end{array}$$

Multiplication

$$
\begin{array}{r}
1101 \\
\times\ 1011 \\
\hline
1101 \\
11010 \\
0000\,00 \\
1101\,000 \\
\hline
10001111
\end{array}
$$

What can we gather?

- What exactly can we gather from doing this computation out longhand?
  - Binary multiplication is even easier than decimal; we only have two choices: 1 or 0
  - Go from right to left, multiplying out the individual numbers
  - After some analysis, this appears to be repeated additions (which are somewhat similar each time)
  - We will implement multiplication the same way!
  - By leveraging repeated sets of Full Adders, we can easily perform these repeated additions.
- Q: But Aki, what if we have too many carries in a single column for a FA to add???
  - A: We're screwed

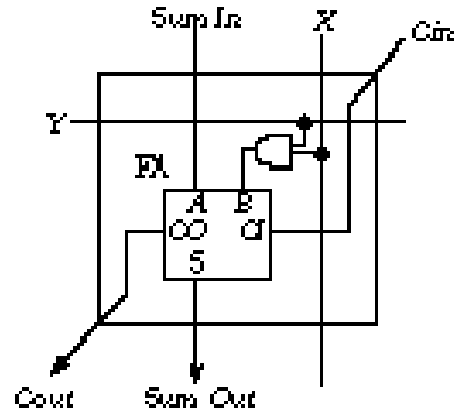
Partial Additions

- Remember how some columns had us adding more than the maximum that a Full Adder could handle?
  - Let's try and address that
- In other words, we want to cut down this big computation into chunks juust small enough so that Full Adders can handle each chunk of it
- Formally, this sort of black magic is called 'Partial Additions'- look to the whiteboard for a demonstration
  - I've also got a video up on the course website explaining this stuff in more detail if you want more clarification

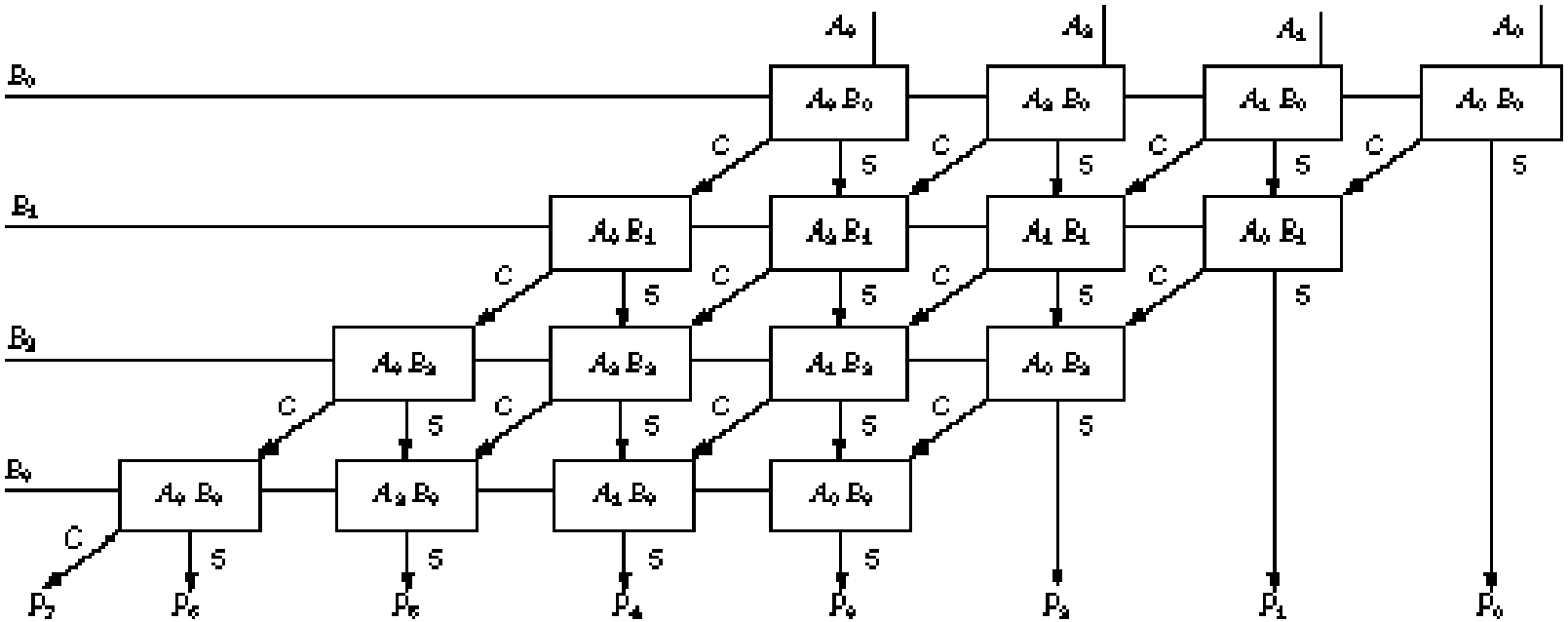
Bringing it all Together

- We know how to multiply one-bit numbers
- We know how to add said numbers together with Full Adders
- We've come up with a clever way to make sure we don't have more carries than we can handle
- Now we just need to implement it (and repeat it a bunch!)

The Actual Circuit



(a) Basic building block



(b) 4 × 4 multiplier structure

Limitations

- Addition, Subtraction (check out the two's complement video on the course website), and Multiplication- we've done it all.
- What about division? Exponentiation?
- The Answer: **It's harder**
- One Solution: Iterative algorithms

Iterative Algorithms

Iterative Algorithms

- Why would we want to use this fancy sounding method?
- Reusing adders will help us- saving gates at the cost of speed
- This is where we begin bridging the gap between the cave-person computer science we're doing and what's actually in computers these days

Division

- Consider division for a moment- how would we do a classic division?
  - No, I'm not whipping out some long division example for you guys
  - Q: Is it possible to convert this to binary, though?
  - A: We can try!