Progress Report 3

Josh Crosby, Scott Kwok, Leo Serpe

Github: https://github.com/jcrosby3uic/ece366-project1

Summary:

For problem 1, we used a combination of XOR and AND gates to build a 1-bit full adder. To build the 4-bit full adder, we needed 4 1-bit full adders, with the carry bit cascading onto the next sequentially. For the Ripple Carry Subtractor (*RCS*), we added a NOT gate to the B input and Changed Cin from 0 to 1. All other functionality remained the same.

For problem 2, we built off of problem 1 by using 8, 4-bit Ripple Carry Adders (RCA) and computed the P&G signals within each block in parallel using two input AND/OR gates to achieve the desired faster computation logic (when compared to 32-bit RCAs). While building this CLA we realized just how much faster it was compared to a 32-bit RCA due to the shorter longest critical delay path.

For problem 3 we realized that PPA are even faster than CLAs making them particularly useful in our case. We used 2-input AND/OR gates to build the P&G logic and based it on our CLA module. The Big-O runtime of an CLA must propagate sequentially which leads to a linear runtime, while a PPA has an O(log N) runtime, the runtime of the PPA vs the CLA is asymptotically faster than the CLA runtime.

The benefit of the bonus problem is a different adder architecture that is faster than the PPA and CLA. Although it is the fastest adder implementation we have done thus far, it is the most complex and most power-hungry, so it is not necessarily the best. In this project, We as a group discussed the tradeoffs between the different adders and architectures. In our research, we have found that the KS and parallel adders are most commonly used.

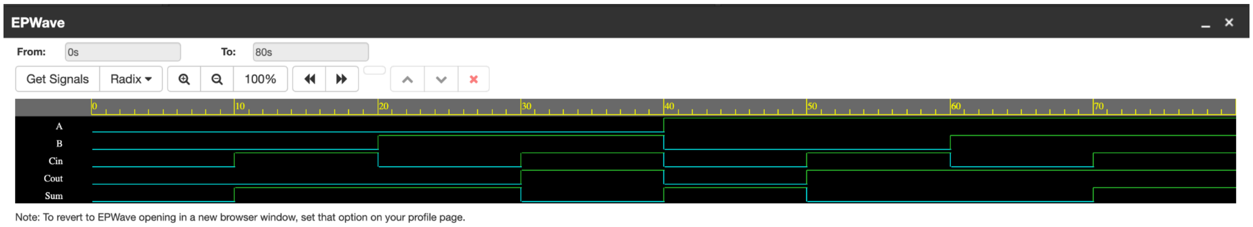
We have completed the first project and finalized all of our work. We used an iterative approach to ensure good implementation.

Team Responsibilities:

All team mates have equal responsibility to every part of the project, which includes equal contributions as well as time dedicated to finish the project in a timely manner. We will all keep checks on each other and ensure that if one group member struggles, others members will help. For the last part of our project, we would delegate for the 16-bit prefix adder.

Throughout our project we would communicate deadlines, and work together, We delegated equally so each of us could work on a problem. Josh worked on Problem 1, Scott worked on Problem 2 and Leo worked on Problem 3. If we struggled we would call each other and work out the numbers until a solution was found, or if any personal problems arose

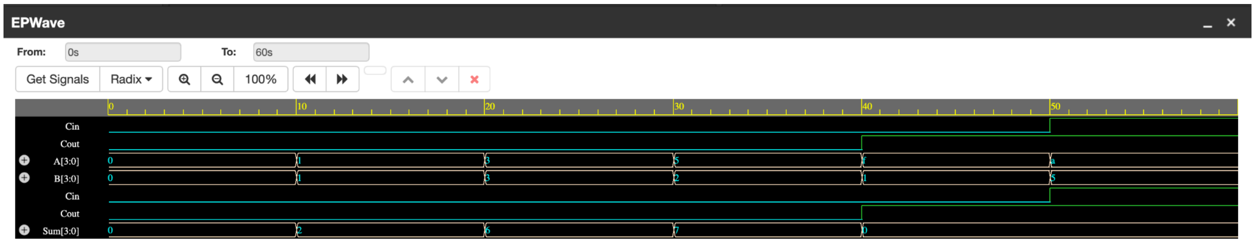
Results


EPWave

From: 0s To: 80s

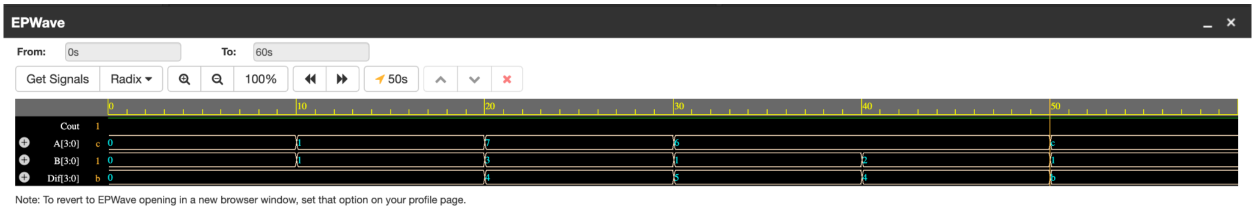
Note: To revert to EPWave opening in a new browser window, set that option on your profile page.

Figure 1: *1-bit adder waveform*


EPWave

From: 0s To: 60s

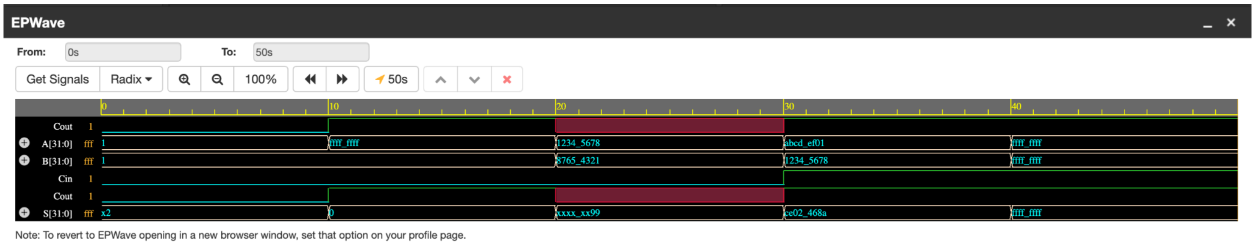
Figure 2: *4-bit RCA waveform*


EPWave

From: 0s To: 60s

Note: To revert to EPWave opening in a new browser window, set that option on your profile page.

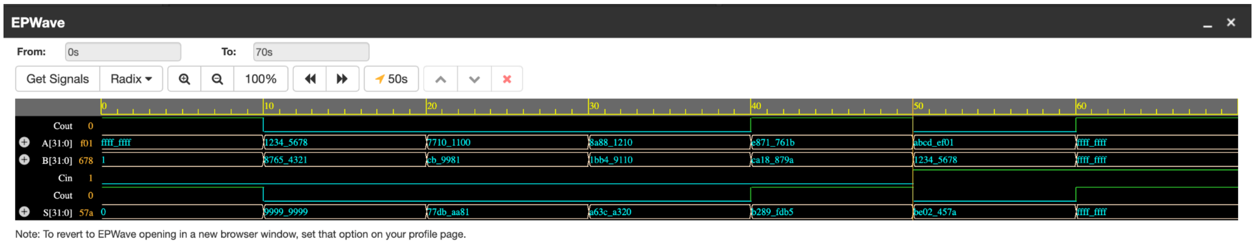
Figure 3: *4-bit RCS waveform*


EPWave

From: 0s To: 50s

Note: To revert to EPWave opening in a new browser window, set that option on your profile page.

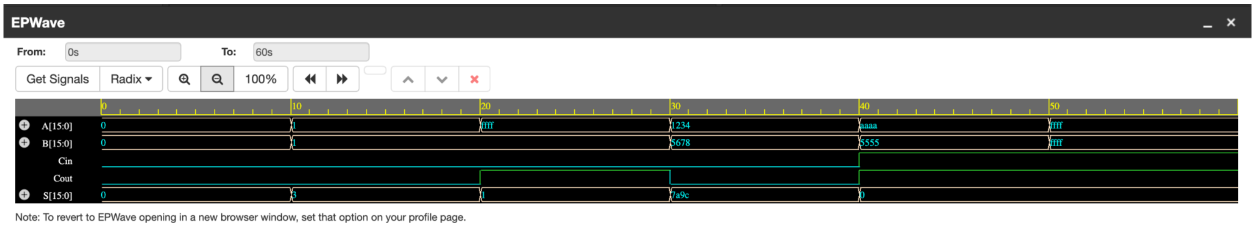
Figure 4: *32-bit CLA waveform*



From: 0s To: 70s

Note: To revert to EPWave opening in a new browser window, set that option on your profile page.

Figure 5: *32-bit PPA waveform*



From: 0s To: 60s

Note: To revert to EPWave opening in a new browser window, set that option on your profile page.

Figure 6: *16-bit Kogge-Stone Adder waveform*