16-bits Accumulator Design

Ruiheng Wang, Mochamad Adrian Prananda, Bohan Wang

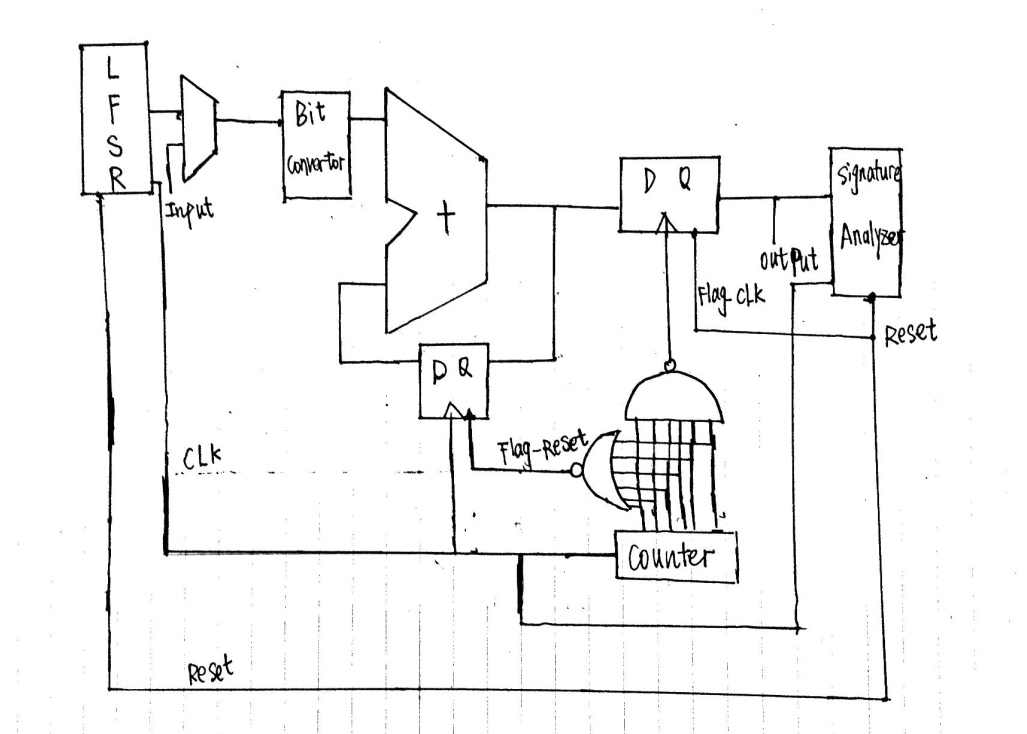
EE 476 AUT 2014

## Abstract

This project is proposed as a accumulator with 16-bits sign magnitude inputs and 22-bits two's complement output as a set of sum-of-64. It performs accumulative addition over 64 clock cycles, and outputs the result immediately at the cycle after 64. It also allows a reset signal to clear the output whenever it is fed in. The accumulator allows BIST capability that accepts pseudo-random number inputs from a LFSR, and outputs testing results through a signature analyzer. Overall schematic is tested both in waveform observation and hand calculation as well as in MATLAB and Perl scripts. Scripts and circuit control file from Bin and Alexa are also used to assist the verification. Layout of the accumulator is built by assembling all modules and the parasitic extraction is used to test the actual performance of the circuit.

## Introduction

This accumulator consists of several modules: mode-selection mux, number conversion, 22-bit Brent-Kung tree adder, output register, feedback register, counter, LFSR and signature analyzer, as shown in the block diagram. The accumulator is able to take input either from external bitstreams or pseudo-random number sequences from LFSR. Input in the form of sign magnitude will be converted to two's complement. The adder takes both the current input and the sum of last cycle and updates the accumulative sum stored in the output register. it is implemented as a modified Brent-Kung tree adder with fixed fan-out of 2 for each gate and compact area. Result from last cycle will be stored in the feedback register, while the output register is clocked by the 6-bit asynchronous counter. When it counts to 64, the counter signals the counter through combinational logic and turns on the output register at the rising edge of the clock. It is also capable of clearing the data anytime when the reset signal is HIGH. For testing,linear feedback shift register(LFSR) provides handy random input to be tested by a signature analyzer. This LFSR implements the Galois type with relatively fast speed, due to the small fan-in of XOR gate.

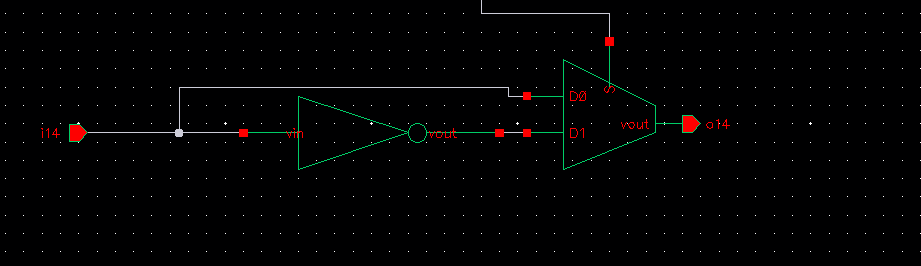


**Fig.1 Block Diagram**

## Implementation

1. *Number Conversion*

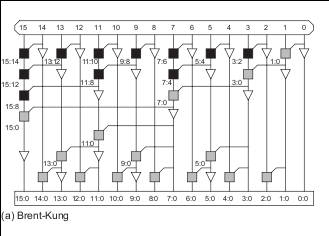
Input in the form of sign magnitude must be converted to two’s complement for the simplicity of binary addition. If the number is positive(the first bit is zero), the number remains the same. If the number is negative, i.e., the sign bit is 1, all bits except the sign one will be inverted and the number is incremented by 1. Below is a snapshot of a bit conversion module. It consists of 16 identical modules , with the sign bit as the select signal and the input “1” gated with inverter.



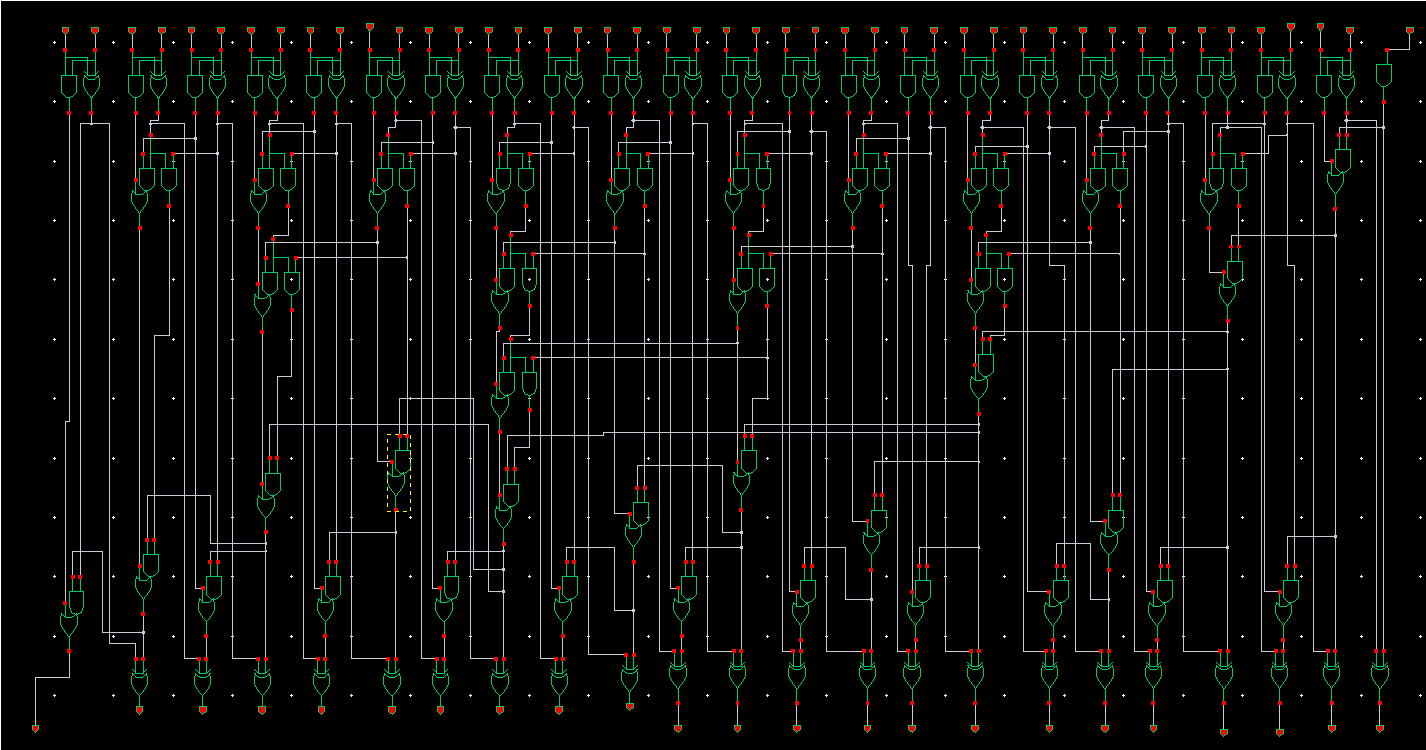
**Fig.2 Bit conversion unit**

1. *Adder*

The group choose the Brent-Kung Adder, a modified carry skip adder that computes prefixes of 2-bit groups, which in turn are used to find the ones of 4-bit groups, and so on. The prefixes fan back down to compute the carries-in to each bit[1]. This concept can be shown from the propagate and generate operation diagram below. One of the advantage of Brent-Kung is the limited area, logarithmic stages of 2log2N-1, and limited fanout of 2 at each. One of the modification of the adder is the integration of a “carry-in” bit that performs the increment of number conversion from sign-magnitude to two’s complement. Additional six bits are used to prevent the overflow given that the maximum possible accumulative sum will not exceed 22 bits.



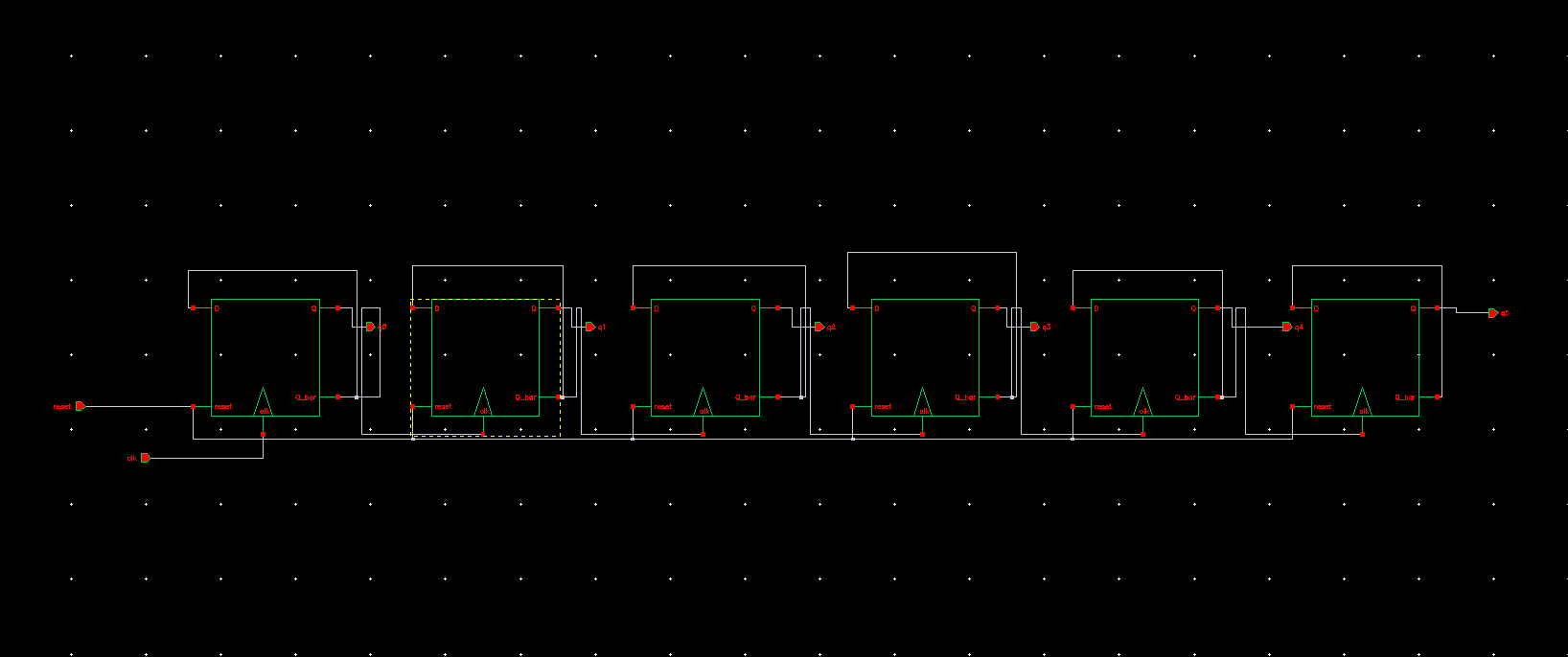
**Fig.3 16 bit Brent\_Kung PG networks[1] (Source: W & H)**



**Fig. 4 Modified 23-bit Brent Kung tree**

1. *Sequential/Logical Design*

Since the accumulator’s purpose is to output the last sum-of-64 every 64 clock cycles, we need to have a control logic that controls when to output the sum and when to reset the sum to 0. This particular part involves a couple of parts: counter, reset logic, and latching logic. For the counter, we used an asynchronous clock divider that counts up as shown below.



**Fig. 5 6-bits up counter**

The second stage of the logic stage would be a nand gate logic that gates all of the counter bits with the inverted clock. The output of the nand logic will go to the clock of the output register. The purpose of this nand gate is to capture the output data on every 64th clock cycle. Whenever the clock is high and all the counter bits hit 64th cycle, we capture the data. The other stage is the nor logic that gates the bits of the counter and the inverted clock signal. Whenever the clock is high, and the counter bits are low, we reset the system and get ready to capture a new data cycle.

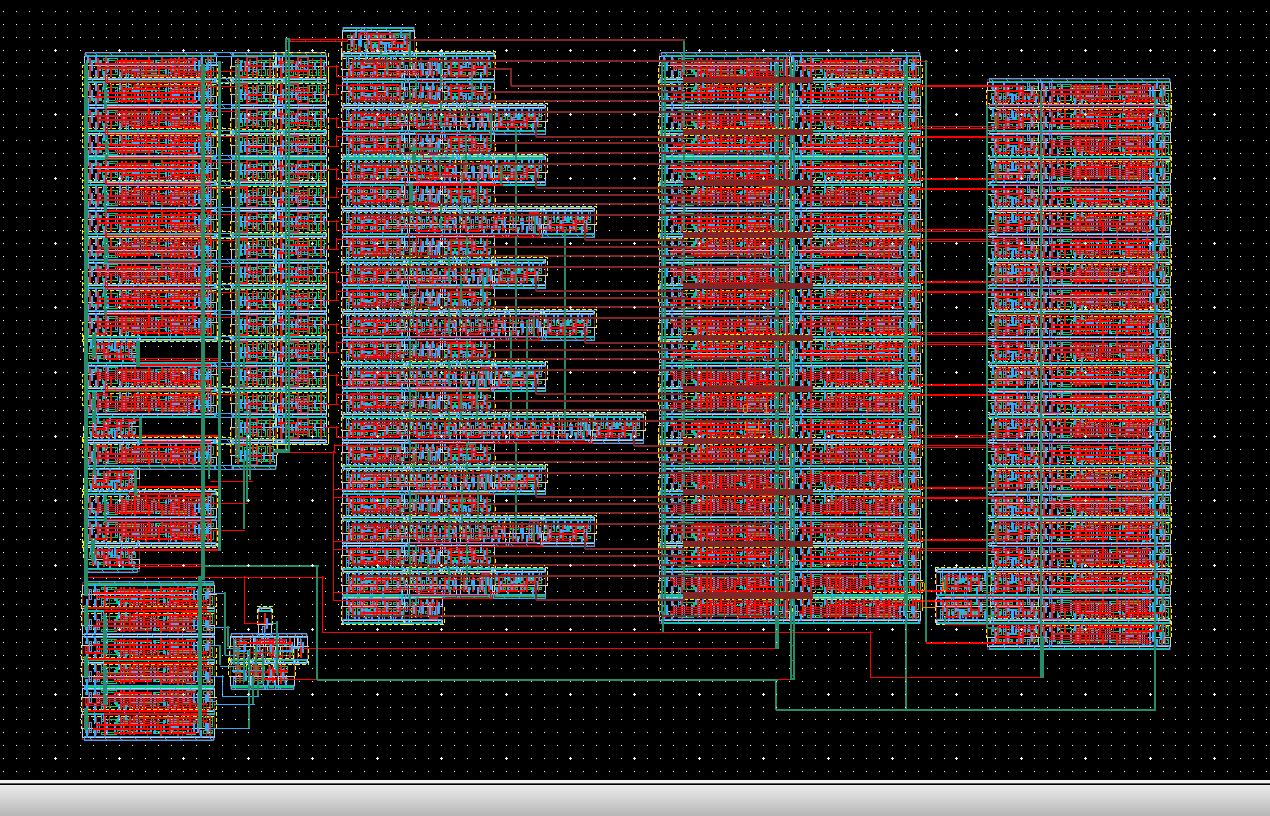
1. *BIST*

This accumulator is capable of self-testing by generating pseudo-random numbers from LFSR and verifying results in signature analyzer. The seed, 16’b100000000000000, is provided by the reset signal connected to the MSB at the beginning of the a 64-clock cycle time interval. The group chose the Galois LFSR due to the small XOR gate fan-in and thus faster speed, and the design is based on the characteristic polynomial of 16 bits, x^{ 16 }+x^{ 14 }+x^{ 13 }+x^{ 11 }+1. The signature analyzer is implemented as a basic LFSR with XORs in each input. Output will be mixed into the pseudo-random patterns based on the characteristic polynomial and the result will form unique signature sequences given the input is correct.

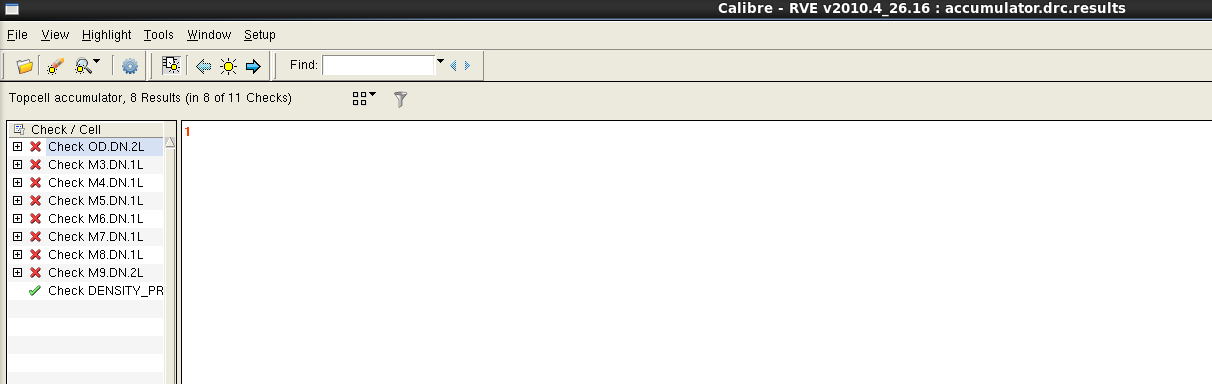
## Results

1. *Layout and DRC, LVS Results*

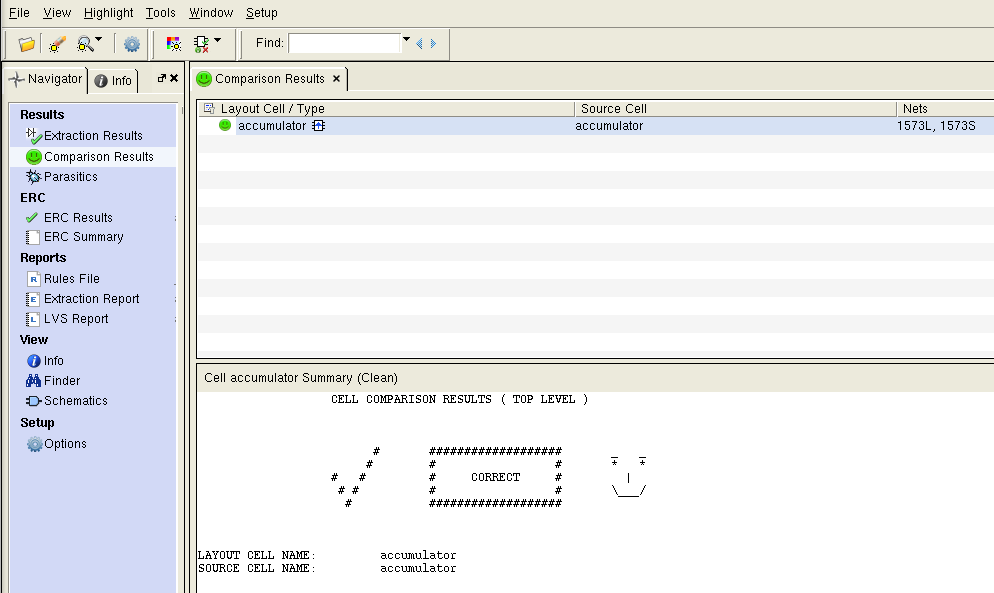
The group has assembled the layout and passed DRC and LVS tests.



**Fig. 6 Layout for the accumulator system**



**Fig. 7 DRC Result (Clean without Density Errors)**

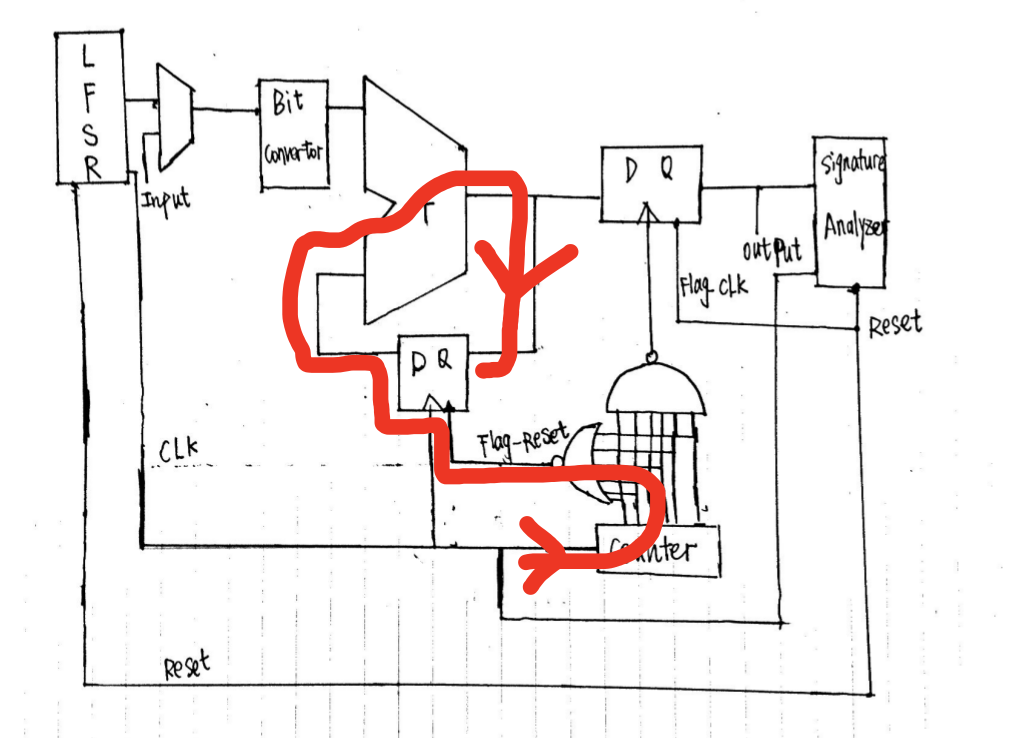


**Fig. 8 LVS Result**

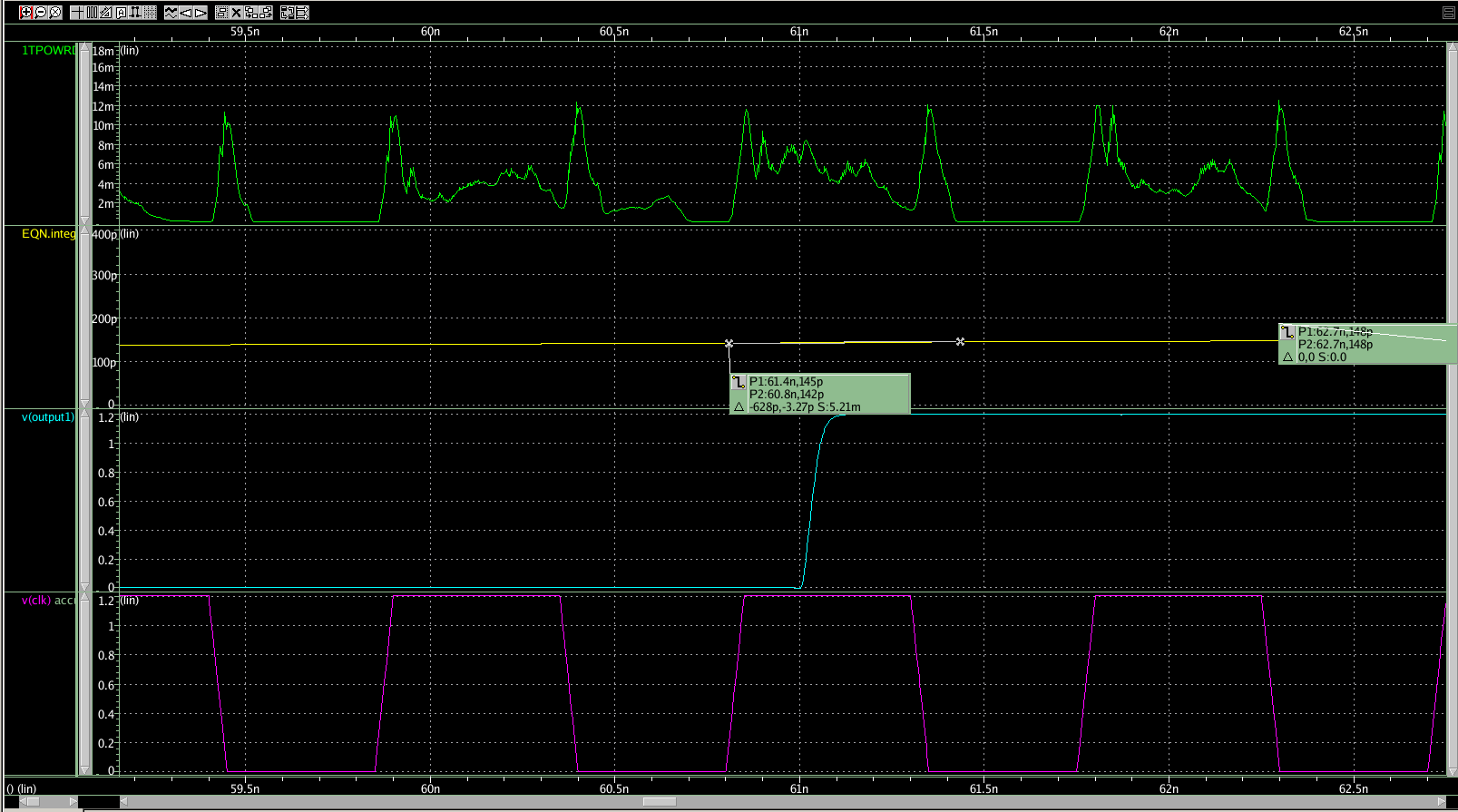
1. *Performance*

|  |  |
| --- | --- |
| Critical path delay | 1.31ns |
| Energy consumption | 620pJ |

After a few tries, we found that the reset on the layout is not triggered properly. This information helps us discover the critical path that is located throught the counter, NOR logic gate, feedback register, the adder. The following figure shows the system's critical path. Energy consumption is done by probing the overall power in HSPICE. In Custom WaveView, energy is calculated by integrating the power consumption over the one clock cycle.



**Fig. 9 Critical Path**

**Fig. 10 Energy measurement**

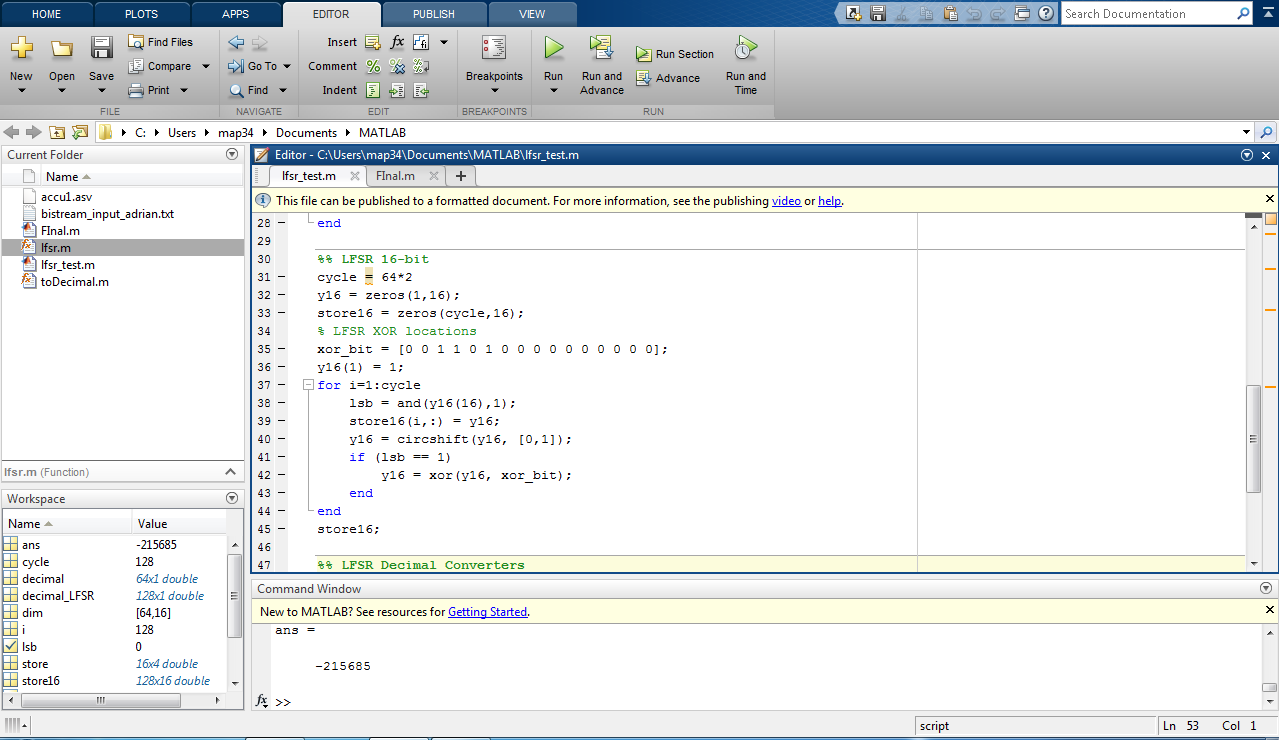
1. *Verification*

To test the results, the group uses the various scripting tools to check the functionality during the design. Verilog is first used to test the top-level design with only 4 bit inputs. The realization of an top-level accumulator can be shown from the output waveform in that it computes and stores the accumulative sum and outputs the sum in the last cycle.

verilog.png

**Fig. 11 verilog waveform verification process**

Later the group develop Perl script based on Alexa’s in order to generate multiple cycle random inputs for a more thorough testing. On the other hand, MATLAB script is also developed to compute binary sum of random inputs. The group both do hand calculation and compare results between MATLAB and circuit simulation to verify the functionality. LFSR is also useful for circuit self-testing. It is also done by MATLAB, realized by a column of pseudo-number generator and shifting of current pattern. Final verification is done by comparing the control file and Bin’s results from the waveform viewer. By using the D/A converter tool in Silicon explorer, the group is able to group 22 bits as a bus output and observe them in hexadecimal.



**Fig.12 Matlab Verification Example Code (LFSR)**



**Fig.13 Waveform Output**

## Conclusion

Our project is to build an accumulator that outputs a sum-of-64 every 64 clock cycles given the input data. The process of the accumulator includes implementing an adder (tree adder), capturing and feedback registers, counter, control logic, and muxes. We chose to build a Brent-Kung tree adder for the reason that it has a small fanout and power consumption. After we verify our tests using random number generators and LFSR, we can verify that the system behaves well. The only concern is the control logic that uses a 7-input NOR gate to reset the system since this logic contributes a lot to out critical path delay. The hardest part of this project is to design a functional system, in which it requires many verification tests, such as Matlab and Verilog scripting. Due to time constraints, we also faced a difficult time in layouting such as finding appropriate wire tracks that connect one module to the other. However, we have an overall functional design.

**V. Appendix**

* Cadence library name: "final\_project" under map34
* Spice Working Directory: "project" under map34