FLOATING POINT FUSED DOT PRODUCT UNIT

Guided by

Prof. Subir Roy

Submitted by

Swagatika Mahapatra (MT2018525)

Ravali Palakurthi (MT2018518)

CONTENTS:

INTRODUCTION:

- **Fused dot product:** In fused dot product unit, addition, multiplication and subtraction can be done in one hardware.

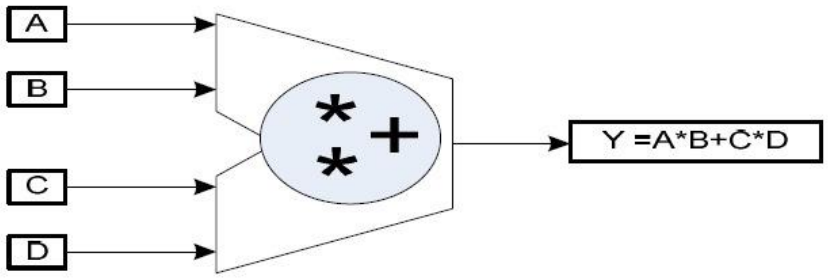

Fig1: Fused dot product

- **Difference between fixed and floating point numbers:**
  Fixed point: Fixed number of digits after and before decimal point.
  Eg: 456.88, 123.45, 456.12 etc
  Floating point: Decimal point is not fixed, it varies. Wide range of numbers.
  Eg: 4.5688, 45.688, 0.00045688 etc

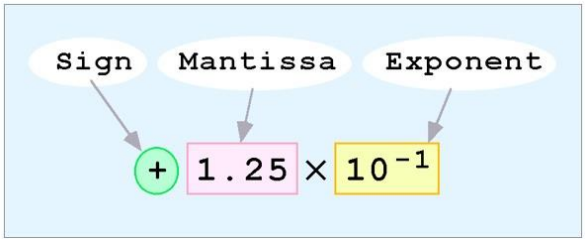

Fig2: Sign, Mantissa, Exponent

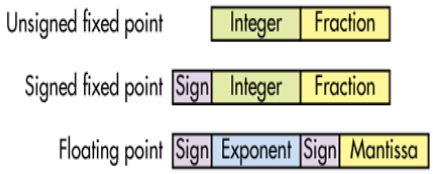

Fig3: Fixed and floating point numbers

IEEE 754 FORMAT:

Single precision:

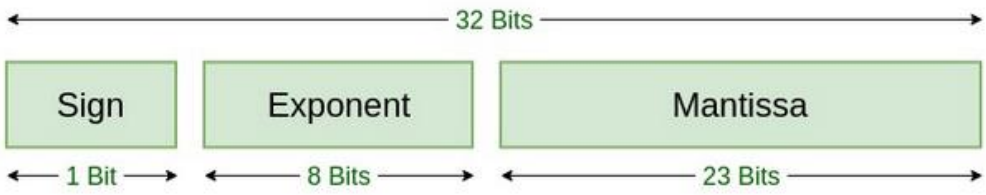

Fig4: Single precision format

- The exponent is biased by 127.(e-127)
- The normalized significand is 1.m . The binary point is before bit-22 and the one is not present explicitly.
- S = 1 → Negative number , S = 0 → Positive number
- The value of normalized number is

$$(-1)^s \times 1.m \times 2^{e-127}$$

Conversion of IEEE 754 to decimal:

Consider the following 32-bit pattern

1 1011 0110 011 0000 0000 0000 0000 0000

The value is

$$\begin{aligned} & (-1)^1 \times 2^{10110110-01111111} \times 1.011 \\ = \; & -1.375 \times 2^{55} \\ = \; & -49539595901075456.0 \\ = \; & -4.9539595901075456 \times 10^{16} \end{aligned}$$

FLOATING POINT ADDITION:

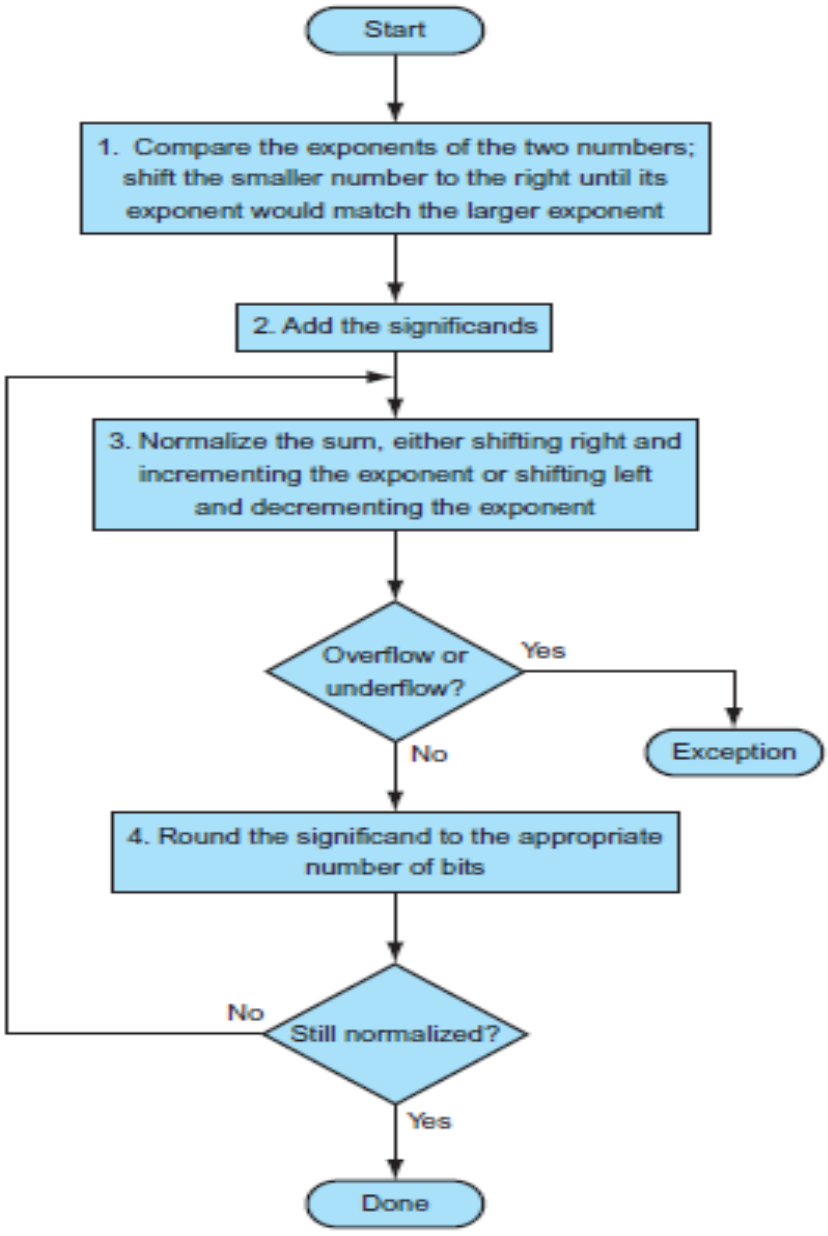

Fig5: Floating point addition

FLOATING POINT MULTIPLICATION:

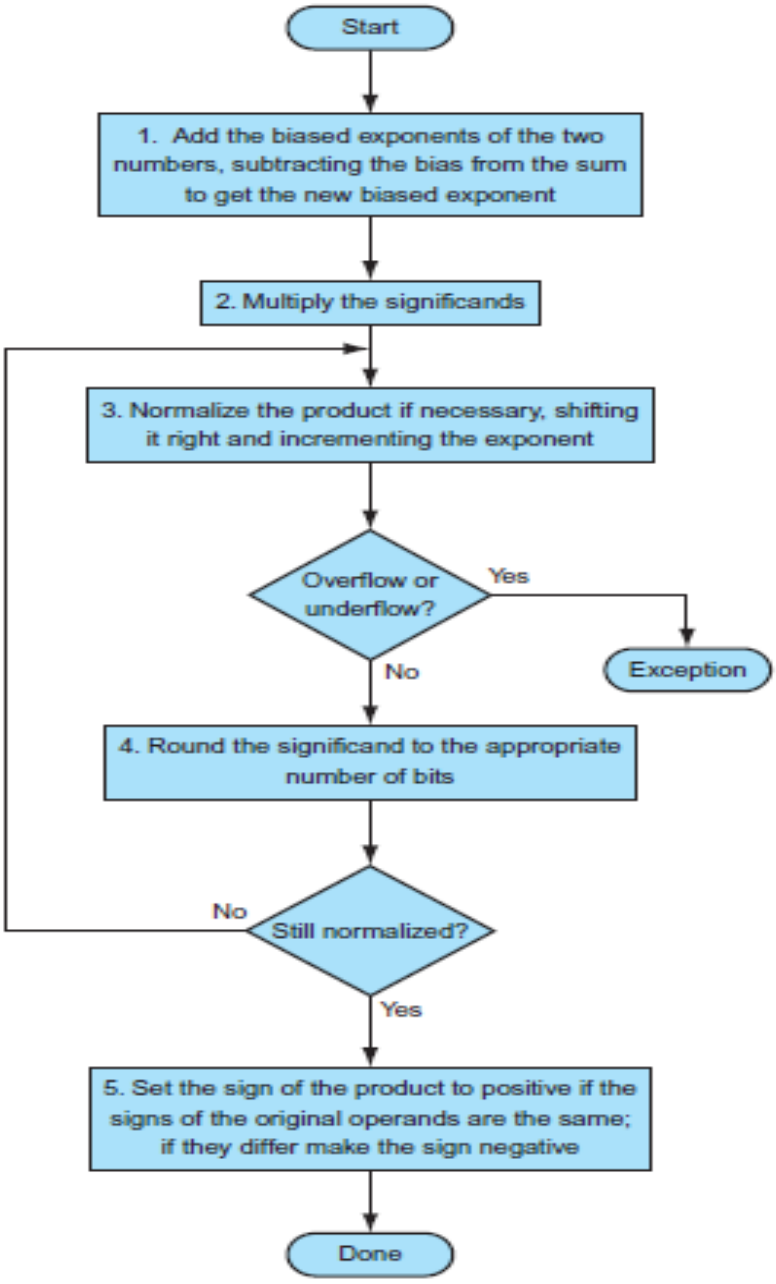

Fig6: Floating point multiplication

IMPLEMENTED DESIGN:

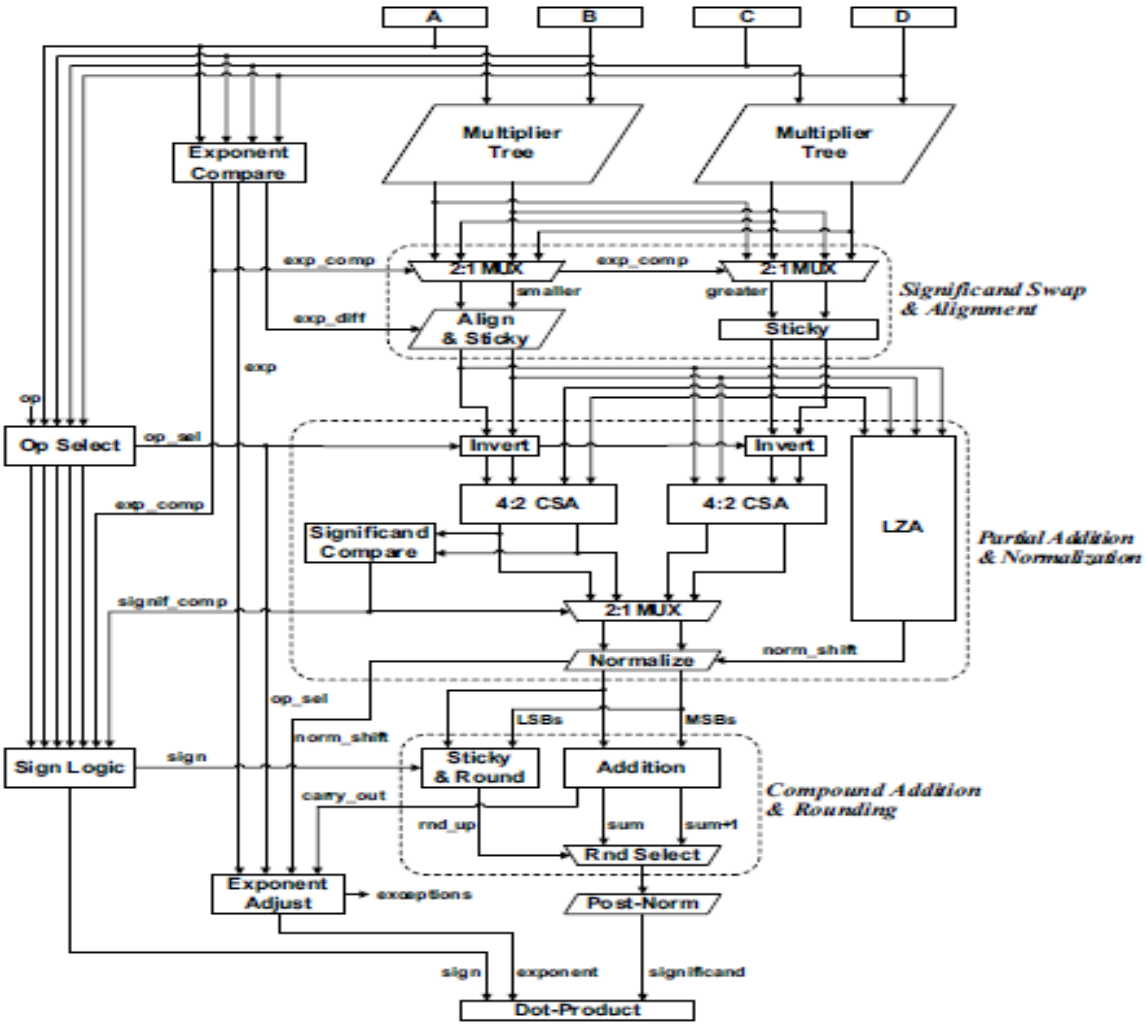

Fig7: Enhanced floating point fused dot product unit

KOGGE STONE ADDER:

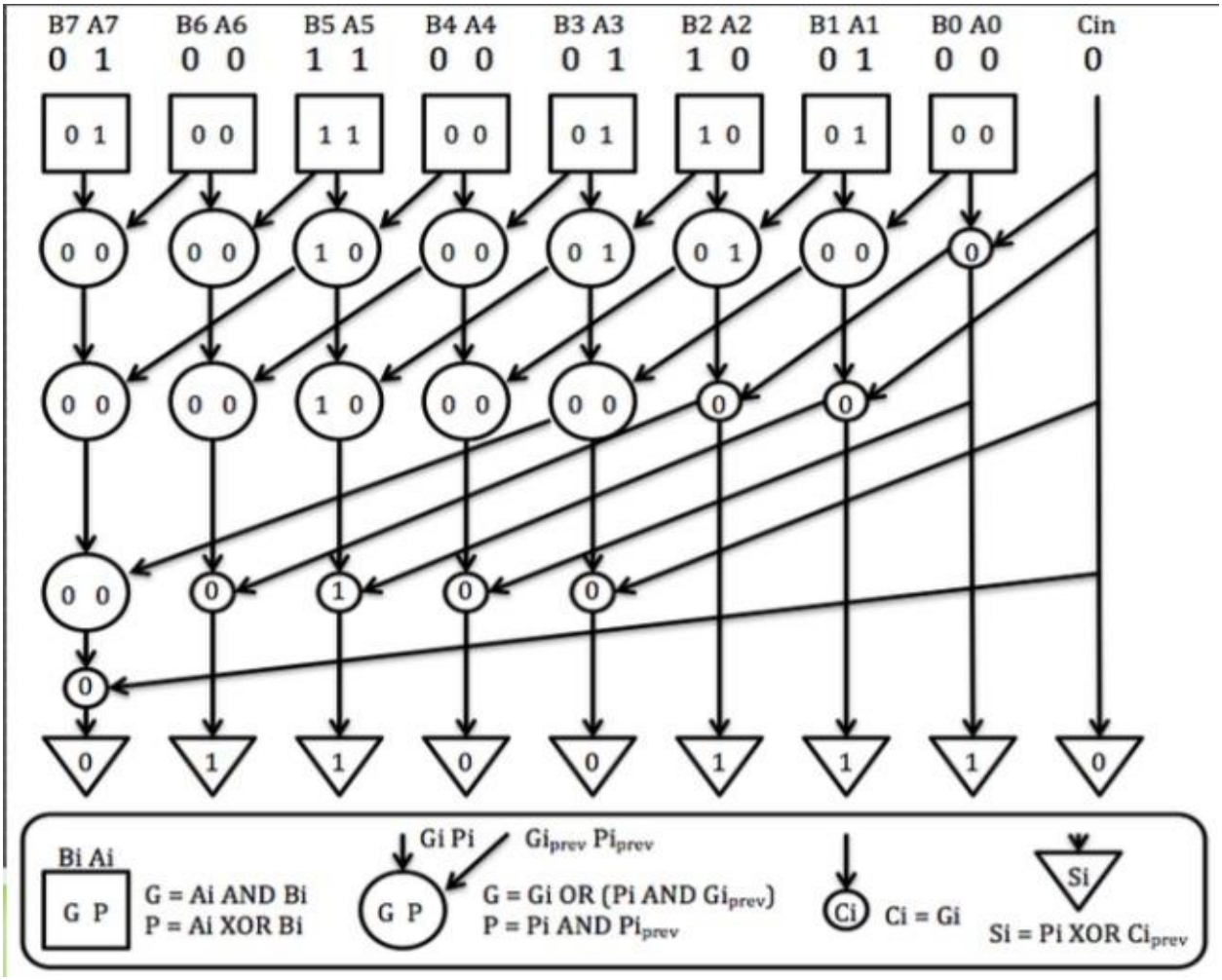

Fig8: Kogge stone adder architecture for 8 bit

Code for carry generate:

```
module grCarryGen(p1,g1,p0,g0,P,G);
input p1,g1,p0,g0;
output P,G;
wire op1;
and a1(P,p1,p0);
and a2(op1,p1,g0);
or o1(G,op1,g1);
endmodule
```

Code for carry propagate:

```
module grCarryPropagate(p1,g1,g0,G);
input p1,g1,g0;
output G;
wire op1;
and a2(op1,p1,g0);
or o1(G,op1,g1);
endmodule
```

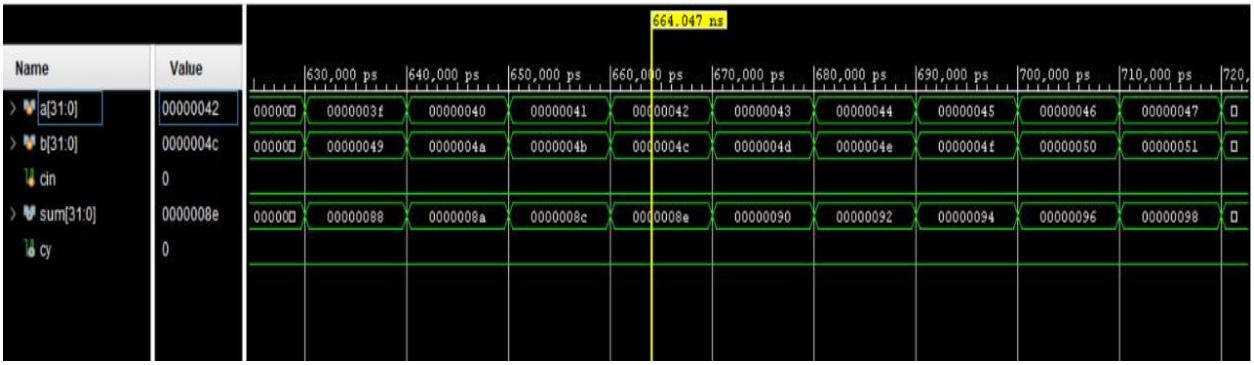

Fig9: simulation results of 32 Bit Kogge stone adder

DADDA MULTIPLIER:

- Three stages in a multiplier:
  1. Partial product generation
  2. Partial product reduction
  3. Carry propagation addition
- Dadda is a tree multiplier.
- Weight of carry is one bit higher than the weight of sum.

Bits in a multiplier (N)	*Number of stages needed*
3	*1*
4	*2*
5 to 6	*3*
7 to 9	*4*
10 to 13	*5*
14 to 19	*6*
20 to 28	*7*
29 to 42	*8*
43 to 63	*9*
64 to 94	*10*

Fig10:Reference table of Wallace tree multiplier

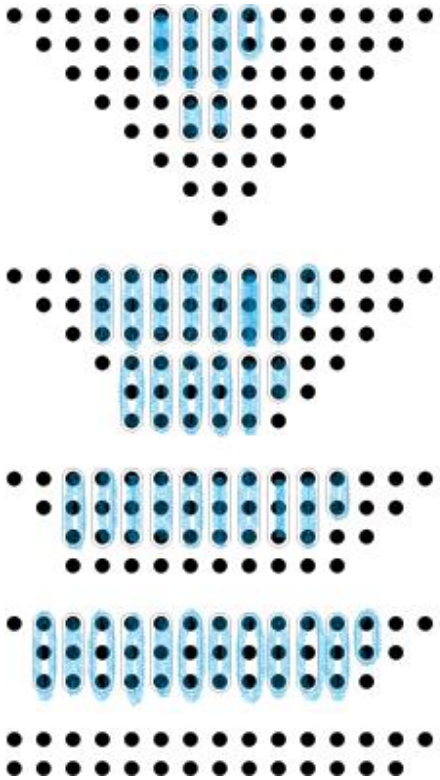

Fig11: Tree diagram of 8 bit Dadda multiplier

Simulation results of 24 bit Dadda multiplier

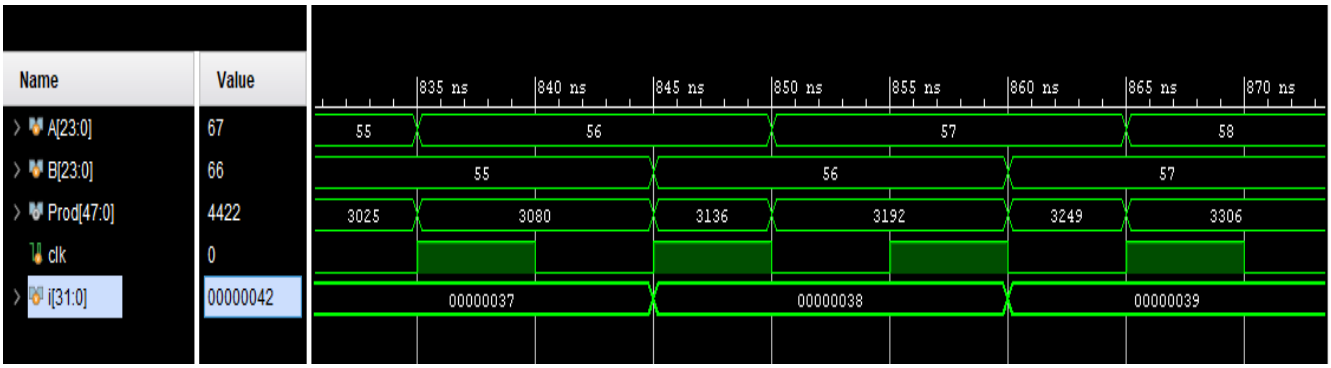

Fig12: Simulation results

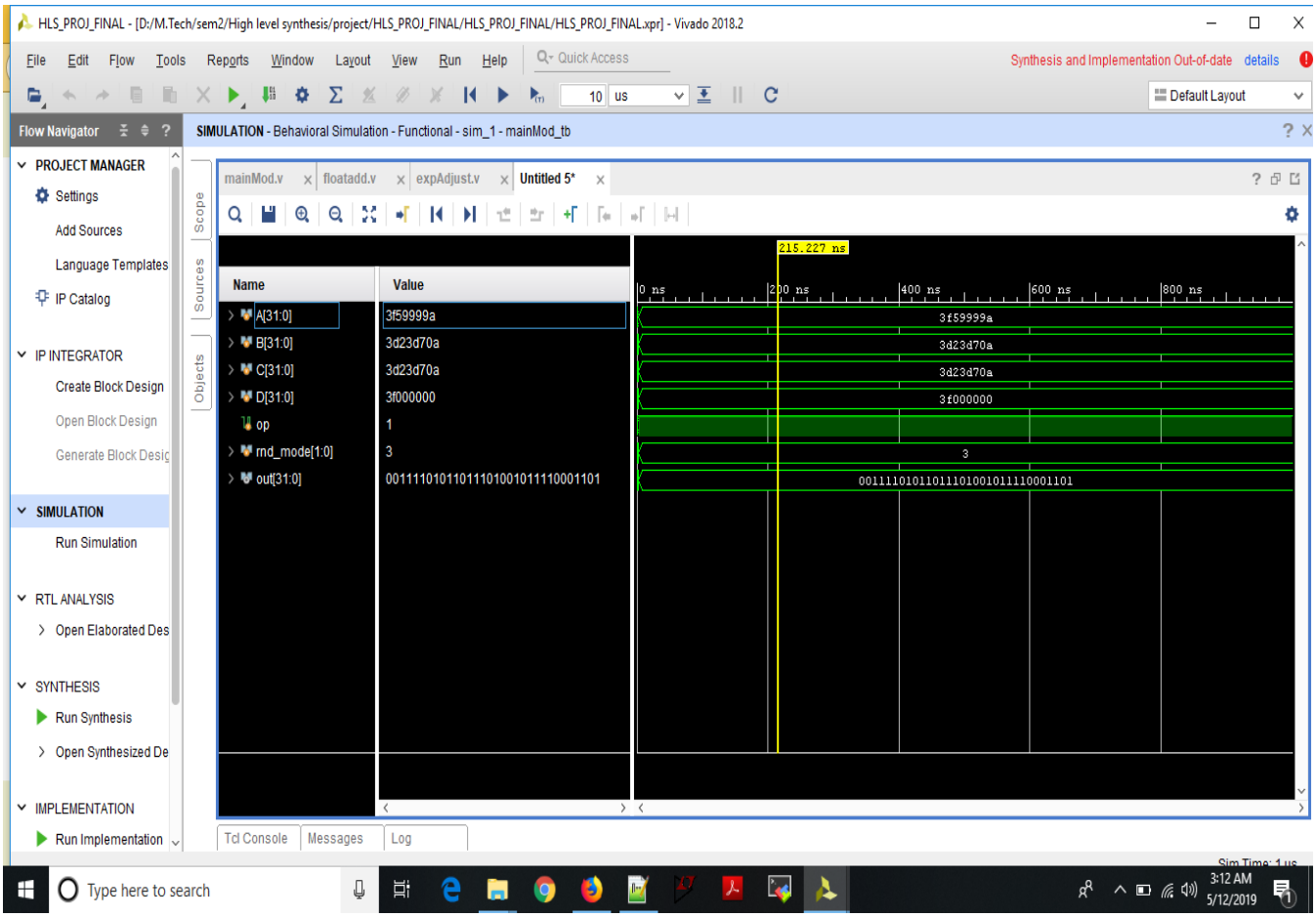

Fig.13 Simulation Result Of Fused Dot Product Unit

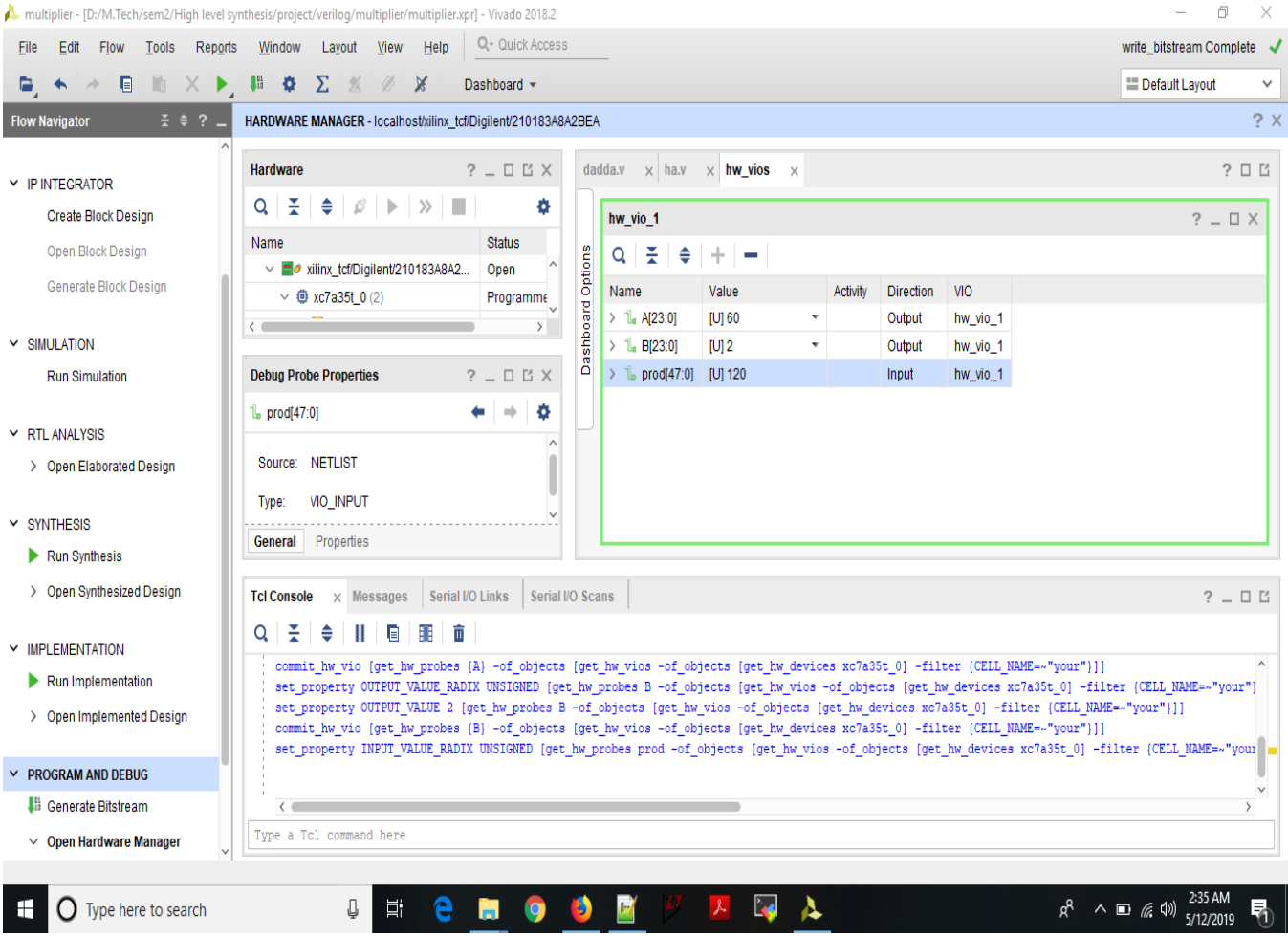

Fig.14 FPGA Implementation Of Dadda Multiplier

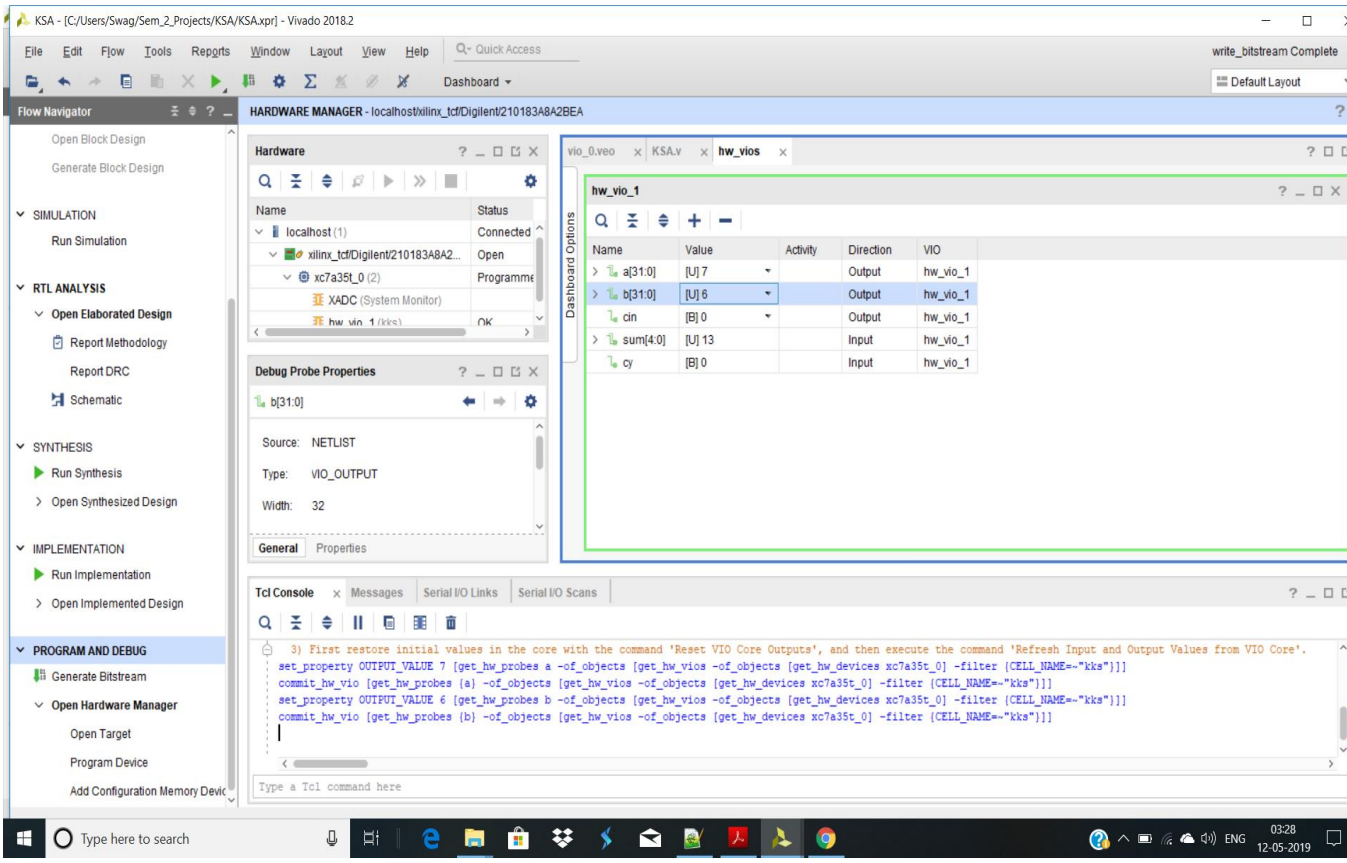

Fig.15 FPGA Implementation Of Kogge Stone Adder

Utilization Report

No of LUTs Used = 11729

On Chip Power = 0.126W

Dynamic Power = 0.117W