DFX Calculator
[16 8 4]
(Adder, Subtractor, Multiplier)

ECE 5736 – Reconfigurable Computing

Professor: Daniel Llamocca

By: Oshin Gupta, Chandra Kasimkota, & Tahmid Uddin

6/26/2020

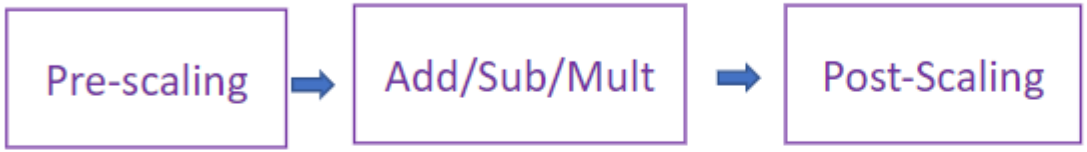
Dual Fixed Point Background

Standard Numerical Representations:

- Fixed and Floating Point:
  - Floating point features a large dynamic range at the expense of a large resource usage on the other hand fixed point requires fewer resources, but it delivers low dynamic range.
- Dual Fixed point (DFX):
  - This numerical representation is a compromise between Fixed and Floating point number system, it overcomes the limitations of Fixed and Float point numbers
- For applications such as Digital Signal Processing where a large dynamic range is required using the floating point data representation
- The DFX has Exponent and Significand, depends on the exponent value DFX can have two types of scaling: num0 and num1
- For the project DFX calculator we are using the DFX number system

DFX Process & Formulas

- As part of the pre scaling convert the DFX to FX
- Process the data for Fixed point Addition/Subtraction/Multiplication
- As part of the post scaling convert FX to DFX



- Range for $num0$: $[-2^{n-p_0-2}, 2^{n-p_0-2} - 2^{-p_0}]$
- Range for $num1$: $[-2^{n-p_1-2}, -2^{n-p_0-2} - 2^{-p_1}] \cup [2^{n-p_0-2}, 2^{n-p_1-2} - 2^{-p_1}]$

✓ Unsigned numbers with n bits:	$\frac{\lvert 2^n - 1\rvert}{\lvert 1\rvert} = 2^n - 1 \rightarrow Dynamic\ Range = 20\log_{10}(2^n - 1)\ dB$
✓ Signed numbers (2's complement): $[n\ p]$	$\frac{\lvert -2^{n-1-p}\rvert}{\lvert 2^{-p}\rvert} = 2^{n-1} \rightarrow Dynamic\ Range = 20\log_{10}(2^{n-1})\ dB$
✓ Dual fixed point (DFX) numbers: n_p0_p1	$\frac{\lvert -2^{n-2-p_1}\rvert}{\lvert 2^{-p_0}\rvert} = 2^{n-2-p_1+p_0} \rightarrow Dynamic\ Range = 20\log_{10}(2^{n-2-p_1+p_0})\ dB$

Dynamic range=20log10(2n+p0−p1−2) dB

Benefit of using DFX over Fixed & Floating Point

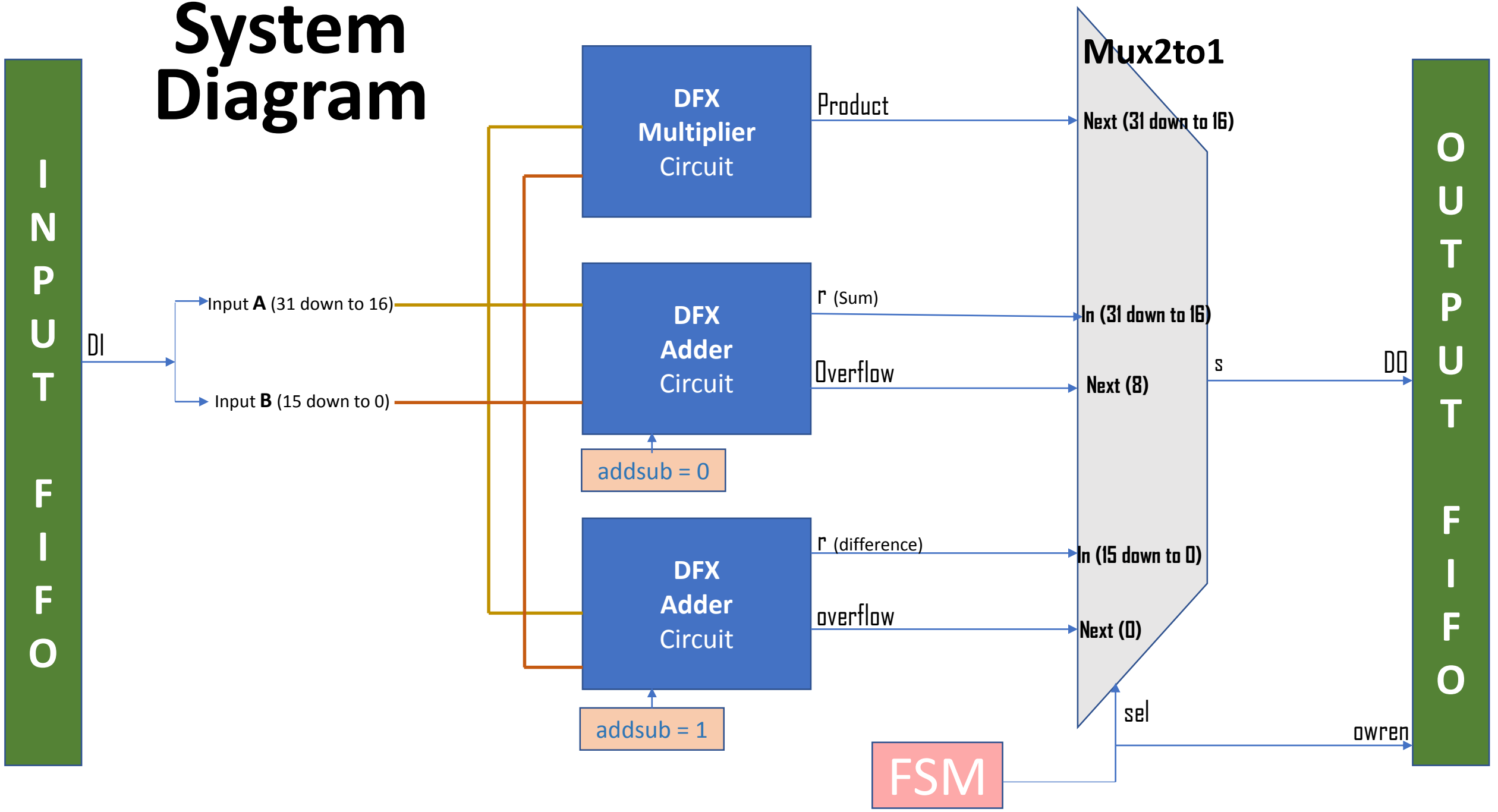
By choosing the right scaling, DFX can have similar performance to fixed-point and also having the capability of handling a wider dynamic range of the Floating point.

Table 1. Dynamic Range Comparison. The DFX is 32 bits wide,P0 =16 and p1=4

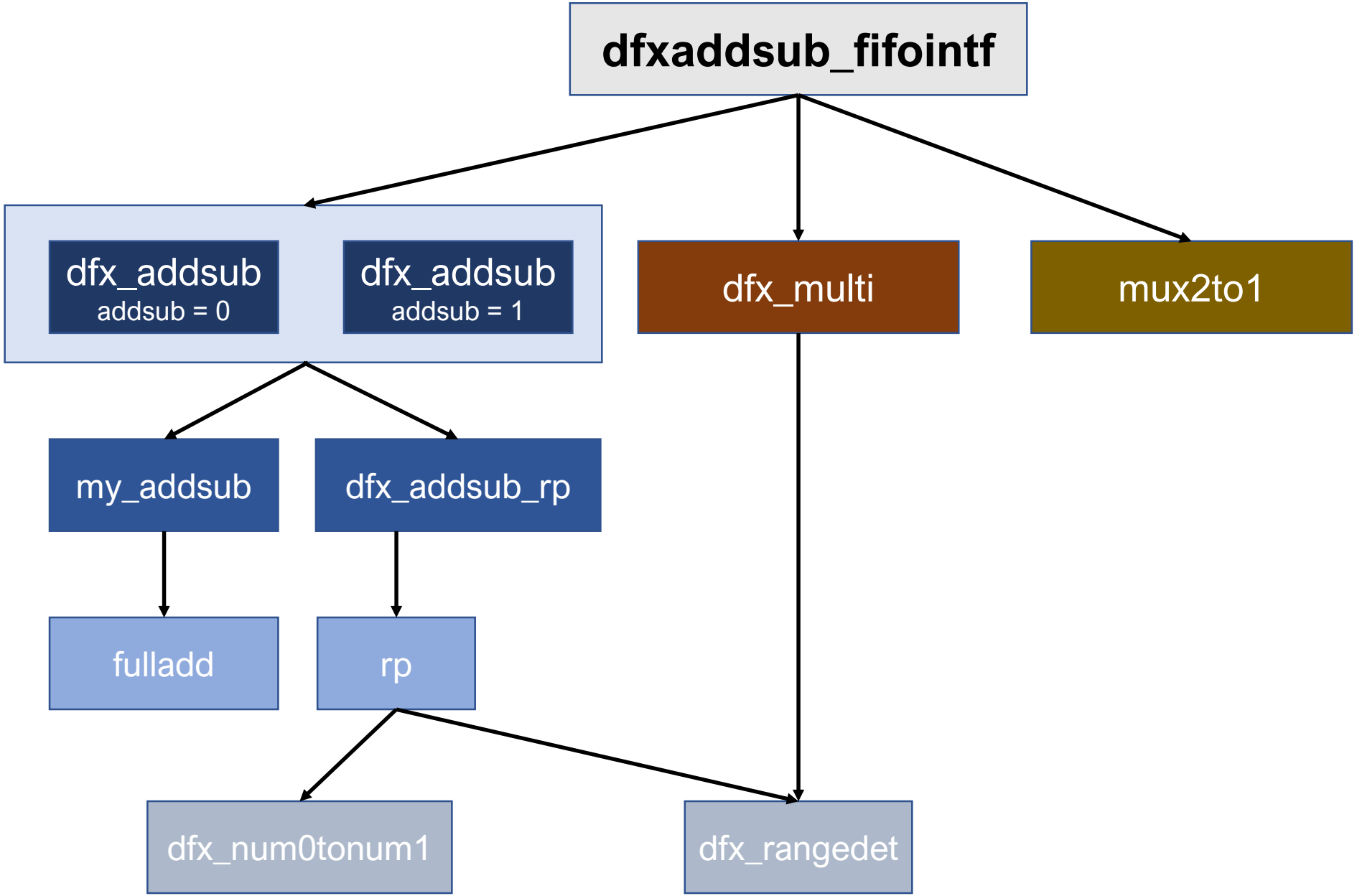
Number System	DFX	Fixed-Point	Floating-Point
Format	32-bit	32-bit	32-bit IEEE
Dynamic Range	2^{+46} ~276dB	2^{+46} ~187dB	2^{+254} ~1529dB

Ref: IEEE document/1515822

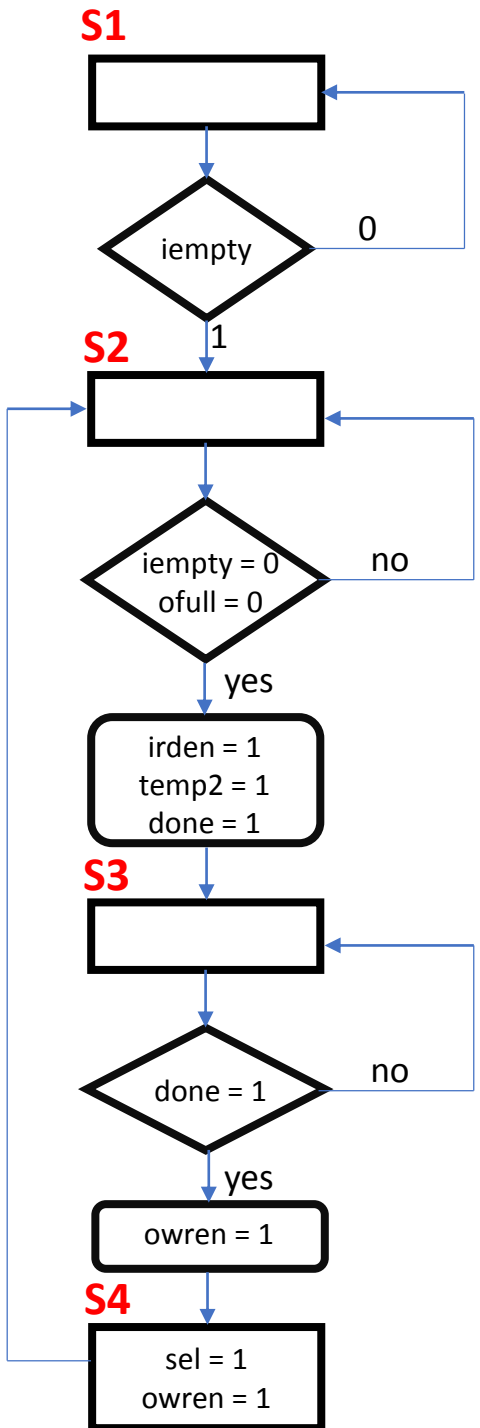
System Diagram



File Hierarchy



FSM



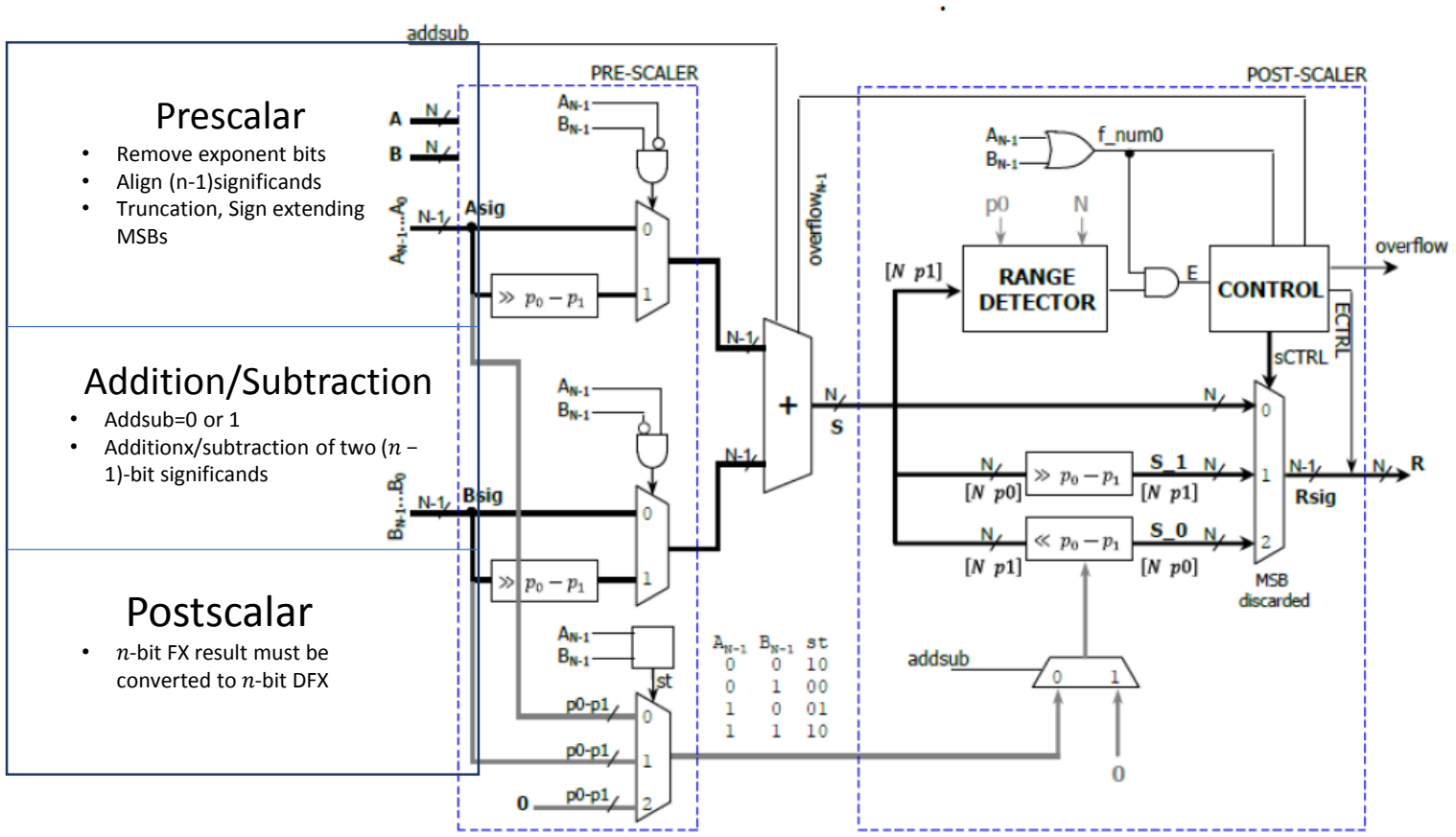
State Description
S1: If there is data to be read, it will start execution
S2: Input data is written into memory
S3: Read the first byte of multiplexor
S4: Read second byte of multiplexor

Hardware/ Software Tasks

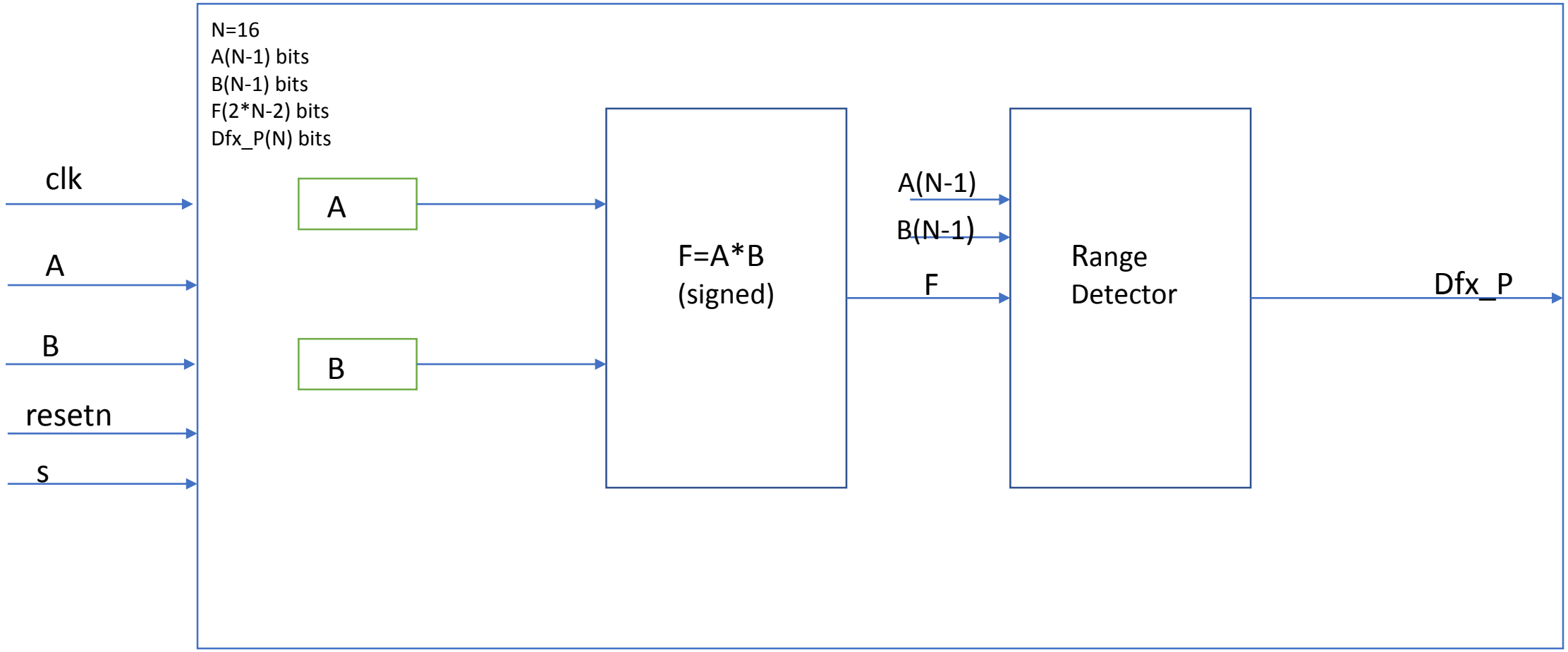
Hardware
Input data is read from axi_wdata.
Digital Systems perform processes. (DFX Multiplier/ DFX Adder).
Output data of Digital Systems is sent to Multiplexer (2 to 1).
FSM will trigger output FIFO to read first set of data and then the next set.
Data is output on axi_rdata.

Software
Will take User input.
Provide input into AXI Interface.
Read output from AXI Interface.
Parse the data according to operation.
Determine if there is overflow.
Print results to User.

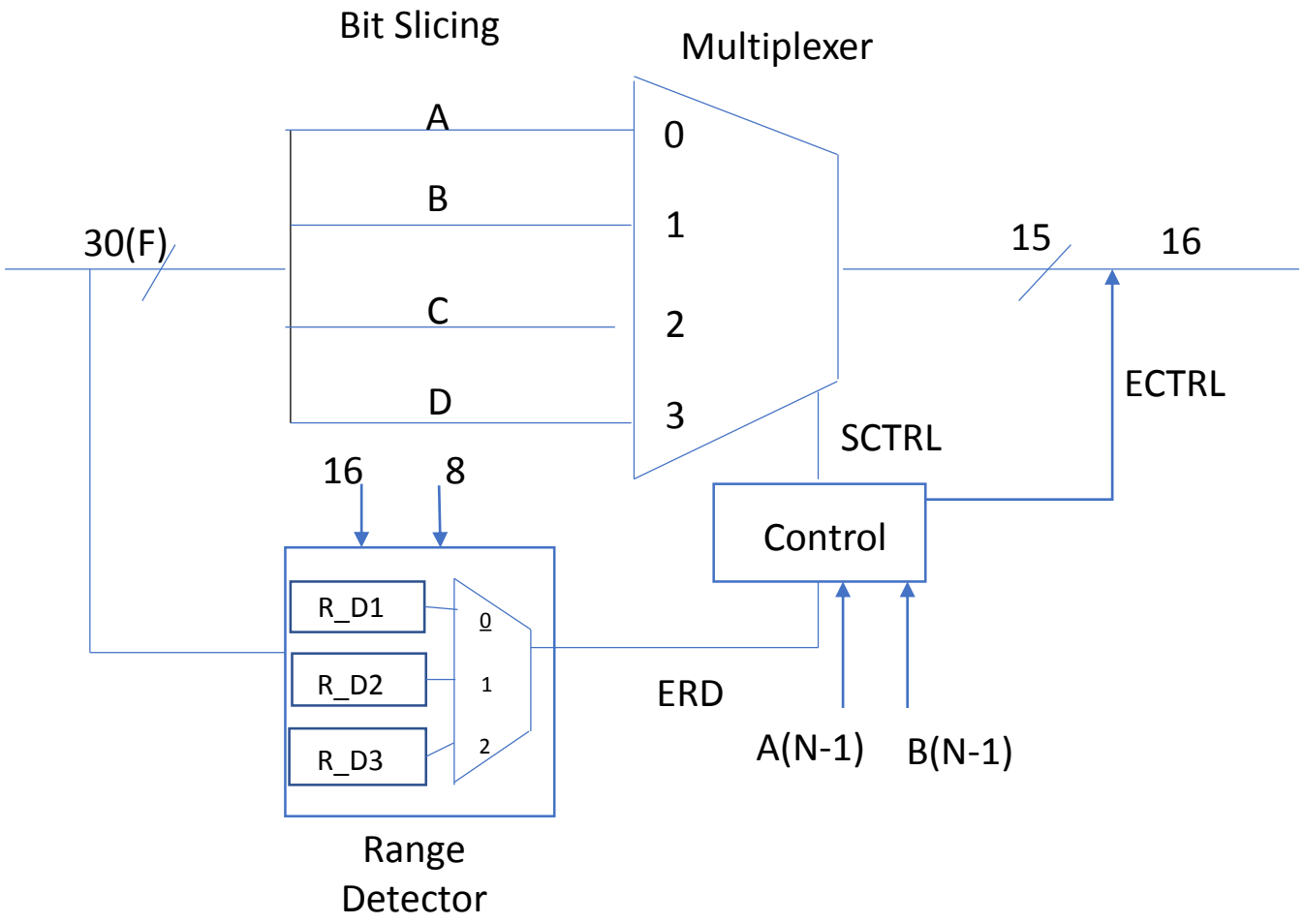
DFX Adder/ Subtractor



DFX Multiplier

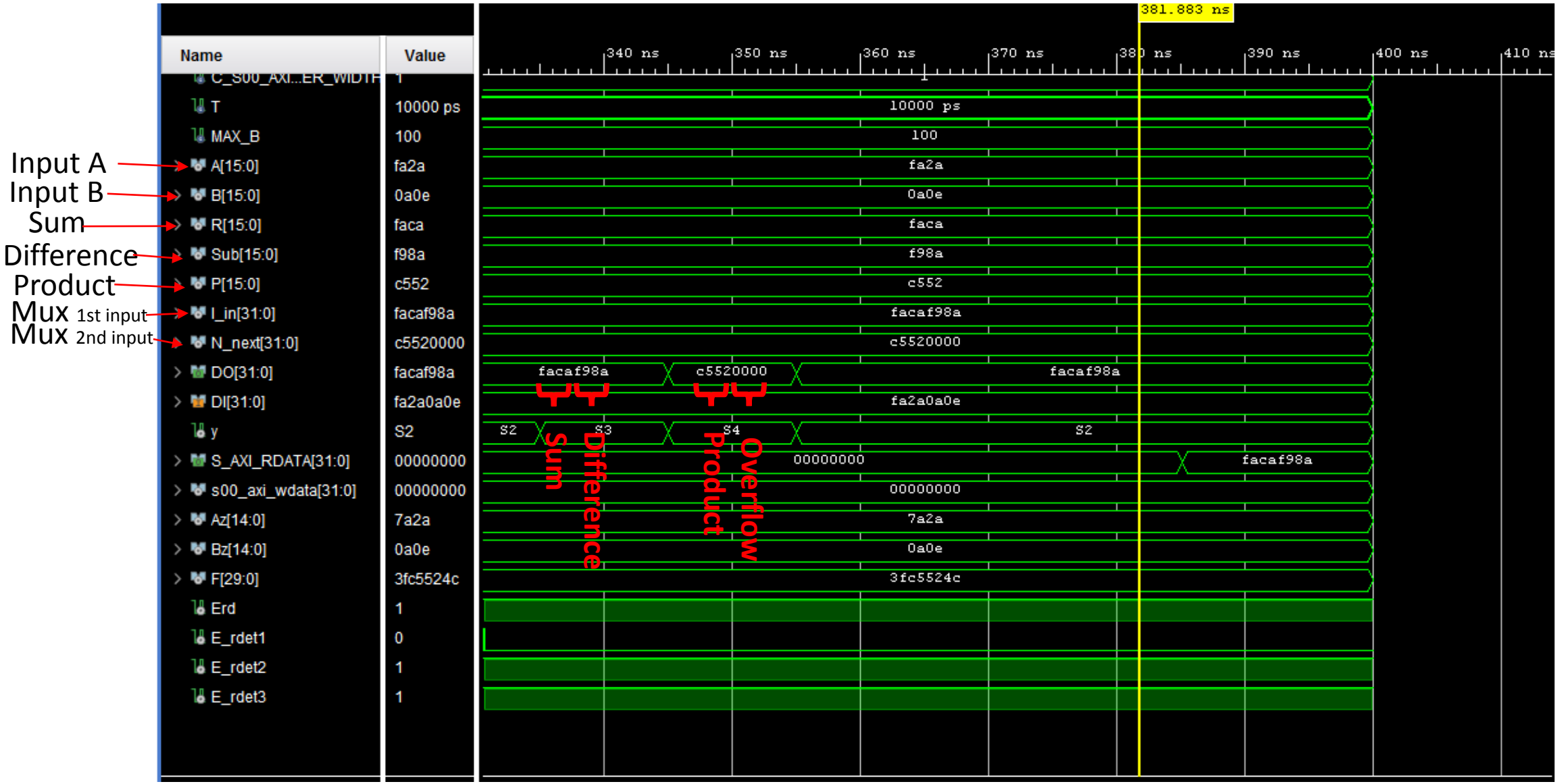


DFX Multiplier Range Detector+ Bit slicing

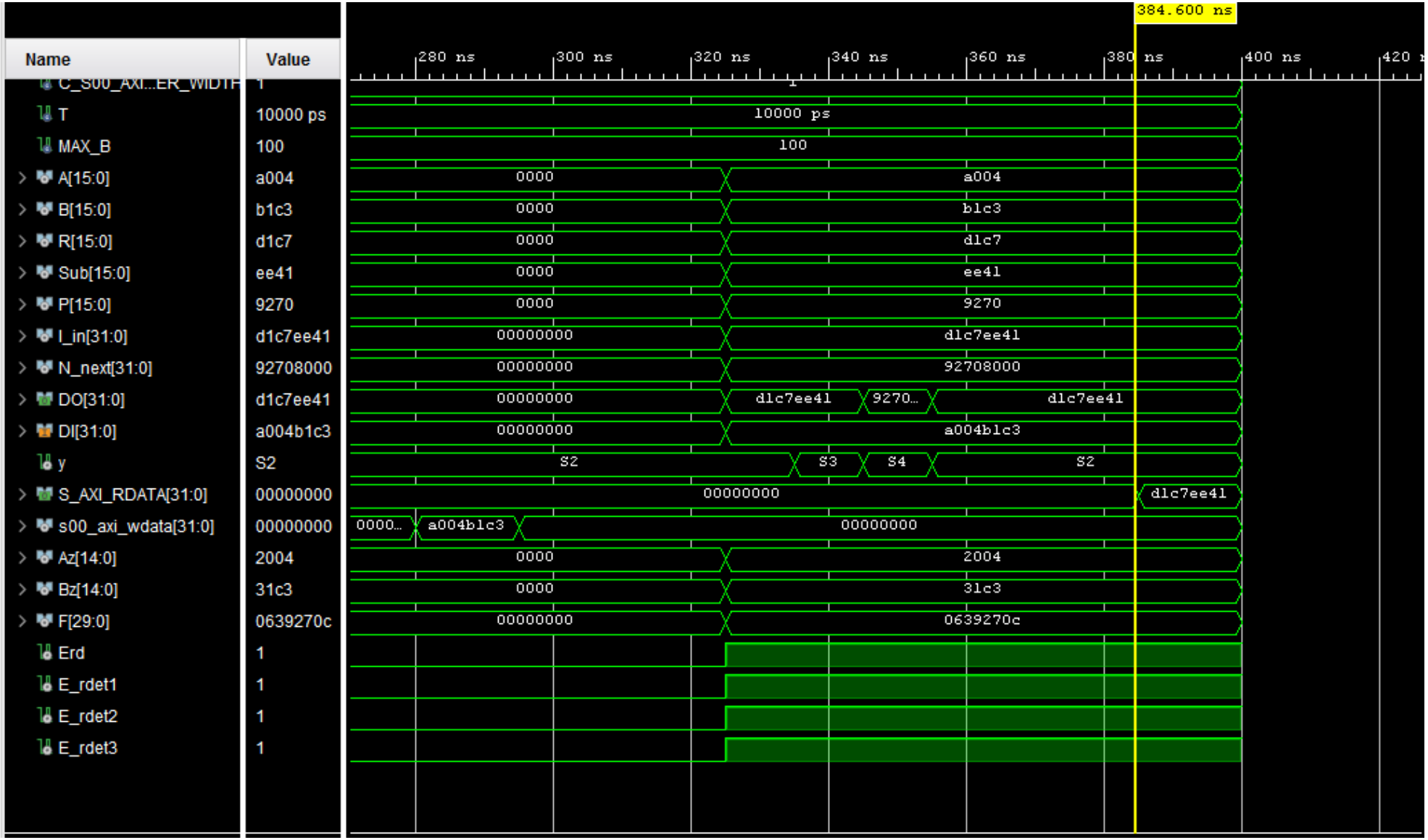


A(N-1)	B(N-1)	ERD	SCTRL	ECTRL
0	0	0	01 (B)	0
0	0	1	11 (D)	1
0	1	0	00 (A)	0
0	1	1	01 (B)	1
1	0	0	00 (A)	0
1	0	1	01 (B)	1
1	1	0	10 (C)	0
1	1	1	00 (A)	1

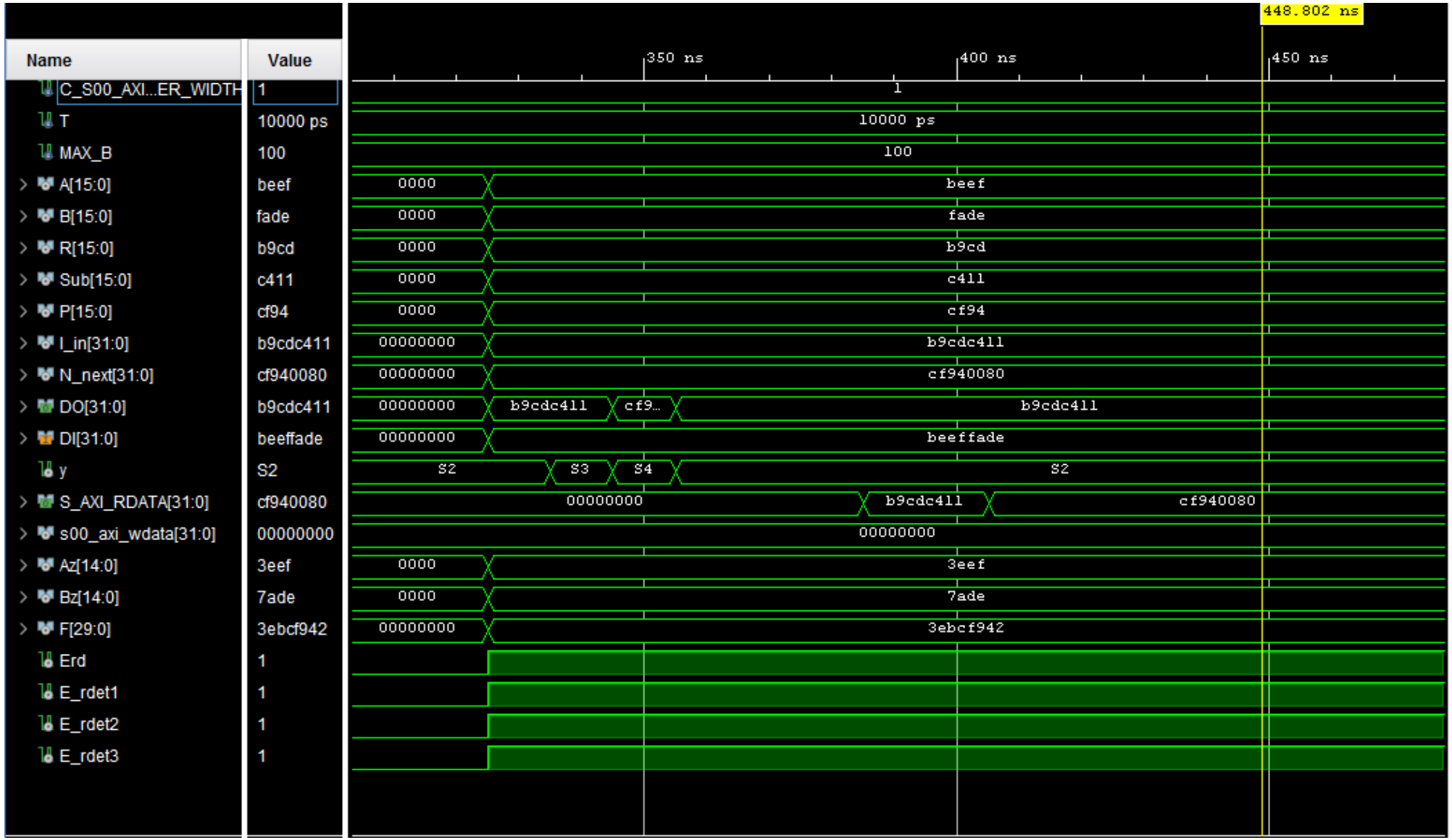
Test bench Simulation : Input A = FA2A ; Input B = 0A0E



Test bench Simulation : Input A = A004 ; Input B = B1C3



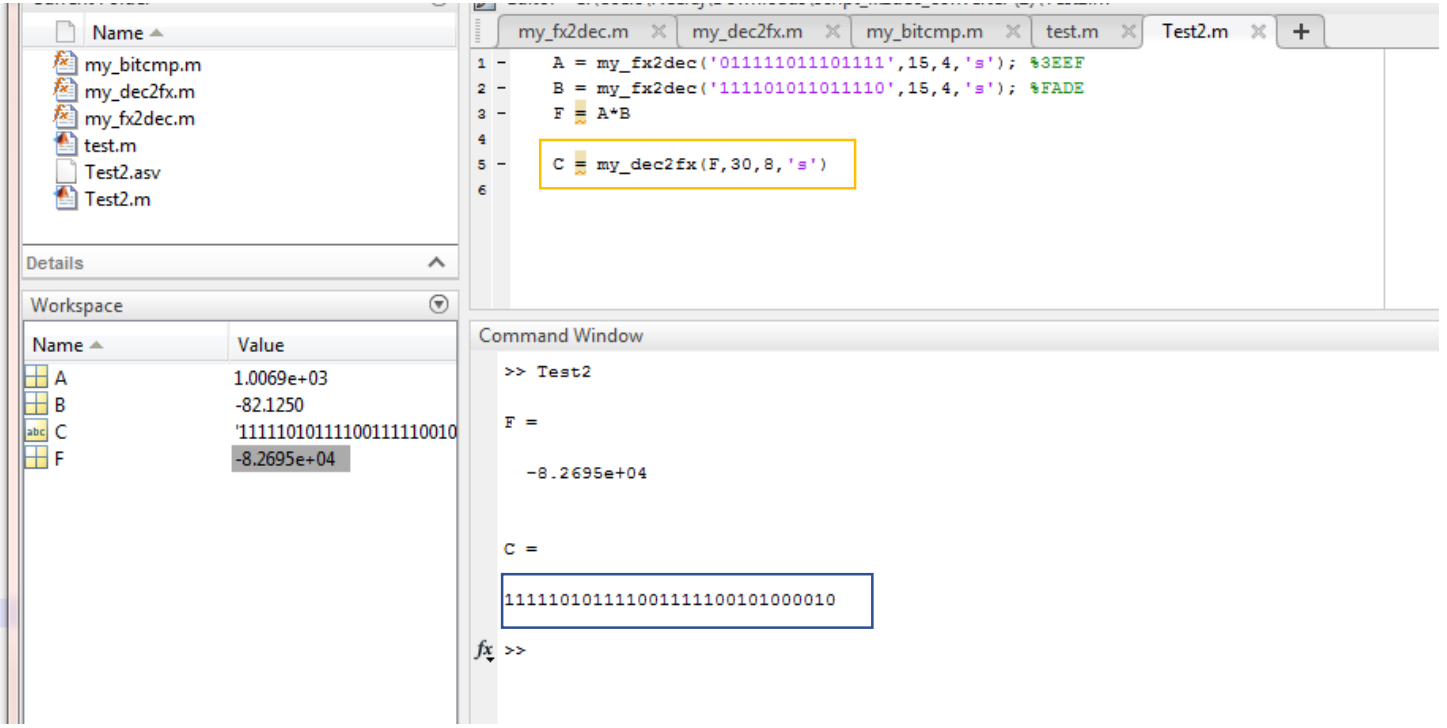
Test bench Simulation : Input A = BEEF ; Input B = FADE



**Calculation of Multiplication on next slide

Multiplication Verification (Manual and Matlab)

```
A=BEEF (1011 1110 1110 1111)
B=FADE (1111 1010 1101 1110)

A(n-1): 1,  B(n-1):1

F= A*B (signed multiplication)
 =(3EEF)*(7ADE)
=3EBCF942  % 11111010111100111111001010000010

 The number is in format[30 8]
 For bit slicing, [30 8],E=1, 18 downto 4

 FX{30 8]->DFX [16 8 4]
 since num1, therefore,

 3EBCF942->1100111110010100 (CF94)
           CF94
```



Softwar
e

```
//--------------------------------INPUT DATA HERE-------------------------------------
   //Enter Input A & Input B as 1 word.
   //Example: Input A = FA2A ; Input B = A0A0E ===> input = 0xFA2A0A0E

//  input= 0xFA2A0A0E; // no overflow
//  input= 0xA004B1C3; // has adder overflow
//  input= 0xBEEFFADE; // has subtracter overflow


//--------------------------------AXI WRITE DATA--------------------------------------

    MYDFXADDSUBINTR_mWriteMemory(baseaddr + 0*4, (input));
    xil_printf ("Input A = %04X, Input B = %04X\r\n", (input&0xFFFF0000)/0x10000, input&0x0000FFFF);

//--------------------------------AXI READ DATA---------------------------------------
    Mem32Value1 = MYDFXADDSUBINTR_mReadMemory(baseaddr);
    Mem32Value2 = MYDFXADDSUBINTR_mReadMemory(baseaddr);

    sum = (Mem32Value1&0xFFFF0000)/0x10000; //Parsing for solutions
    diff = Mem32Value1&0x0000FFFF;
    prod = (Mem32Value2&0xFFFF0000)/0x10000;
    of = Mem32Value2&0x0000FFFF; //Overflow

//-------------------------------PRINTING SOLUTIONS---------------------------------
    if (of == 0x00000100) //If Addition has OVERFLOW
    {
        xil_printf ("Sum = Overflow, Difference = %04X, Product = %04X\r\n", diff, prod);
    }

    if (of == 0x00000001) //If Subtraction has OVERFLOW
    {
        xil_printf ("Sum = %04X, Difference = Overflow, Product = %04X\r\n", sum, prod);
    }

    if (of == 0x00000000) //If NO OVERFLOW
    {
        xil_printf ("Sum = %04X, Difference = %04X, Product = %04X\r\n", sum, diff, prod);
    }
```

User will input the date here in the format of 0xaaaabbbb.

The output data is then read and parsed accordingly.

Depending on whether or not there is Overflow, the results will be printed.

SDK TERMINAL

Condition	SDK Terminal
Calculations with no Overflow	AXI4-Full DFX Calculator Peripheral: Test Peripheral: Base address is 0x7AA00000 Input A = FA2A, Input B = 0A0E Sum = FACA, Difference = F98A, Product = C552
Calculations with Overflow in Addition	AXI4-Full DFX Calculator Peripheral: Test Peripheral: Base address is 0x7AA00000 Input A = A004, Input B = B1C3 Sum = Overflow, Difference = EE41, Product = 9270
Calculations with Overflow in Subtraction	AXI4-Full DFX Calculator Peripheral: Test Peripheral: Base address is 0x7AA00000 Input A = BEEF, Input B = FADE Sum = B9CD, Difference = Overflow, Product = CF94

Project Responsibilities

Oshin Gupta	**-**Creation of DFX Multiplier & Testbench (Digital System) -Assistance in Bus Interface -Circuit Block Diagrams & Test Bench runs in presentation
Chandra Kasimkota	-Research of background for using User inputs in SDK Terminal -High level diagrams in HW/report -Background of DFX slides of presentation
Tahmid Uddin	-Bus Interface between circuits & AXI Full Interface -Software Implementation -System Diagrams in presentation

Conclusion

- Challenges
  - Multiplier
- Next steps
  - Adding the Divider Circuit
  - Adding User Inputs
  - Add on-the-fly reconfigurable portion to set P0 & P1