Lab 2 report

Yihao Wang 7410178057

1. Design Description

In my design, I used four 8bits Booth multipliers and CSA/RCA addition structure to generate (xr/4^k), an 8bits Booth multipliers for (xr/4^k)*n generation and CSA/RCA addition structure for t generating t. Then, the comparison between t and t – n is achieved by 9bits signed number subtraction (0 < t < 2n) and I used RCA for subtraction (t – n = t + n’ + 1). The mod results (s) is selected by a 7bits 2 to 1 MUX based on the sign bit of t – n.

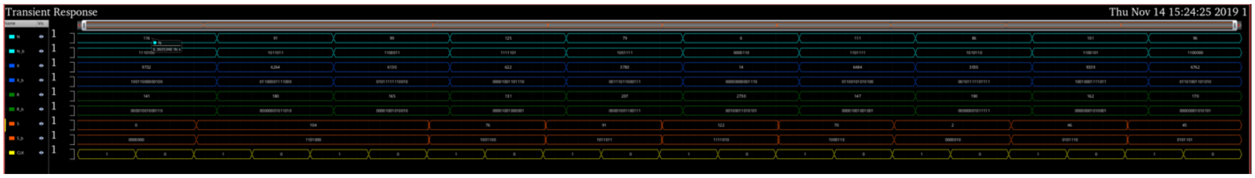
2. Python Scripts

```
import random
import os

## Coversion between decimal number and binary number
def dec_to_bin (num,width):
    the_num = num
    bin_str = ''
    if (num < 0 or num >= 2 ** width):
            print('the parameter must be [0,' + str(2 ** width - 1) + ']!')
    else:
        for num in range(width):
            bin_str = str(the_num % 2) + ' ' + bin_str
            the_num = int(the_num / 2)
        return bin_str

## Creates vector file with randomly generated test patterns
def random_generator(extension = 'vec', x_width = 14, r_width = 14, n_width = 7, number = 30, clk = 10, delay  = 2， unit = 'ns', slope = 0.01, vih = 1.8, vil = 0):
    time = delay
    os.chdir(input('Please input your work directory: '))
    file_name = input('input the file name: ')
    the_file = open(file_name + '.' + extension,'w')
    the_golden = open(file_name + '_golden' + '.' + 'txt', 'w')
    count = x_width - 1
    print('radix', file = the_file, end = ' ')
    for i in range(x_width + r_width + n_width) :
        print('1', file = the_file, end = ' ')
    print('\nio', file = the_file, end = ' ')
    for i in range(x_width + r_width + n_width) :
        print('i', file = the_file, end = ' ')
    print('\nvname', file = the_file, end = ' ')
    for i in range(x_width) :
        print('X' + str(count) + '_D', file = the_file, end = ' ')
        count -= 1
    count = r_width - 1
    for i in range(r_width) :
        print('R'+ str(count) + '_D', file = the_file, end = ' ')
        count -= 1
    count = n_width - 1
    for i in range(n_width) :
        print('N' + str(count) + '_D', file = the_file, end = ' ')
        count -= 1
    print('\ntunit ' + str(unit), file = the_file)
    print('slope ' + str(slope), file = the_file)
    print('vih ' + str(vih), file = the_file)
```

```python
    print('vil ' + str(vil), file = the_file)
    for num in range(number) :
        print(time, end = ' ', file = the_file)
        n = 0
        while n < 3:
            n = int(random.random() * (2 ** n_width))
        x = n ** 2
        while (x <=0) or (x >= n ** 2) :
            x = int(random.random() * (2 ** x_width))
        r = int((4 ** n_width)/n)
        ## Golden file is used to give the correct mod value in order to verify the waveform
        print(str(time) +': ', end = '', file = the_golden)
        print('x=' + str(x), end = ' ', file = the_golden)
        print('n=' + str(n), end = ' ', file = the_golden)
        print('mod= ' + str(x - n * int(x / n)), file = the_golden)

        print(dec_to_bin(x ,x_width), end = ' ', file = the_file)
        print(dec_to_bin(r, r_width), end = ' ', file = the_file)
        print(dec_to_bin(n, n_width), end = '\n', file = the_file)
        time += clk
    print('Successfully Generated!')
    the_file.close()
    the_golden.close()
```

3. Test Patterns Form

x(dec)	n(dec)	r(dec)	s(dec)	x(bin)	n(bin)	r(bin)	s(bin)
9732	**116**	**141**	**104**	**10011000000100**	**1110100**	**00001001000110**	**1101000**
6264	**91**	**180**	**76**	**01100001111000**	**1011011**	**00000001011010**	**1001100**
6130	**99**	**165**	**91**	**01011111110010**	**1100011**	**00001001010010**	**1011011**
622	**125**	**131**	**122**	**00001001101110**	**1111101**	**00001001000001**	**1111010**
3783	**79**	**207**	**70**	**00111011000111**	**1001111**	**00001001100111**	**1000110**



4. Related Parameters

Height: 431.1um

Width: 110.7um

Area: 47722.77um^2

Fastest Clock Frequency: 125MHz

Average Power Consumption: 5.501E-3 W