ELEC4320 Homework 3

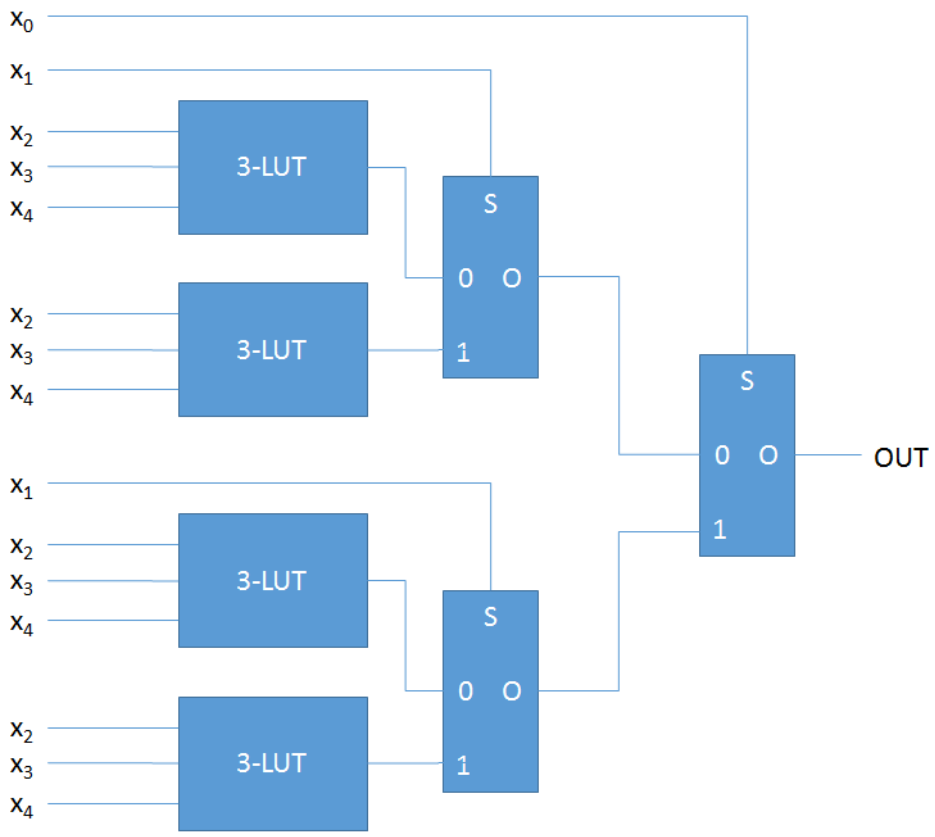
Dhesant Nakka — 20146587

December 8, 2015

Question 1

a) q1_mux.png

b)

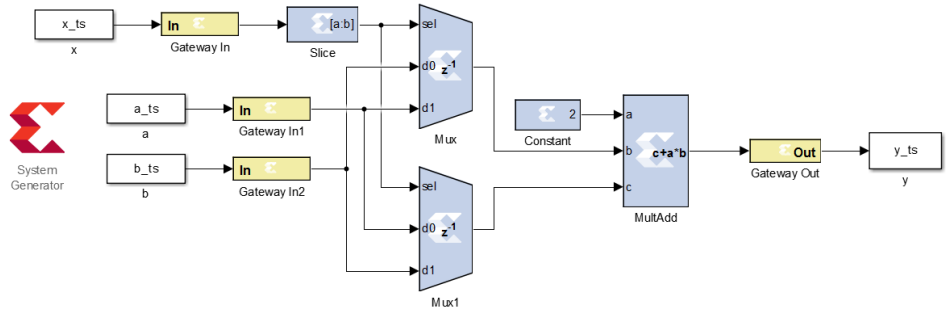


2-input MUX's are created using a 3-LUT that use the following truth table.

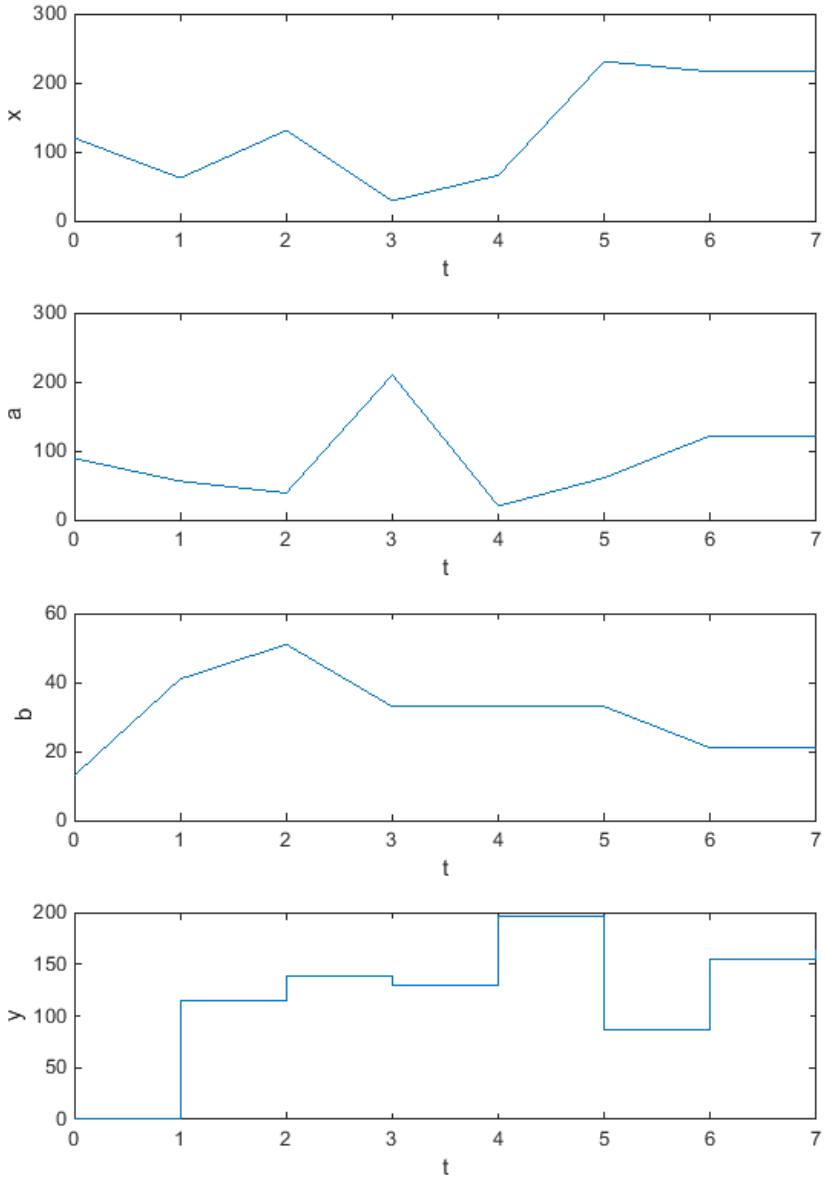
S	A	B	O
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

Question 2

System Generator schematic diagram:



Simulink simulation results:


x
300
200
100
0
0 1 2 3 4 5 6 7
t
a
300
200
100
0
0 1 2 3 4 5 6 7
t
b
60
40
20
0
0 1 2 3 4 5 6 7
t
y
200
150
100
50
0
0 1 2 3 4 5 6 7
t

Question 3

a) i) The longest operation is the two compound adders A1 and A2 in C-step1, requiring a total time of 6 ns. Therefore, the required clock period should be 6ns.

Moving the A2 adder block from C-step1 to C-step2 would satisify the new clock period of 4ns, since the output of A2 is not needed until C-step3 (see Figure 1);

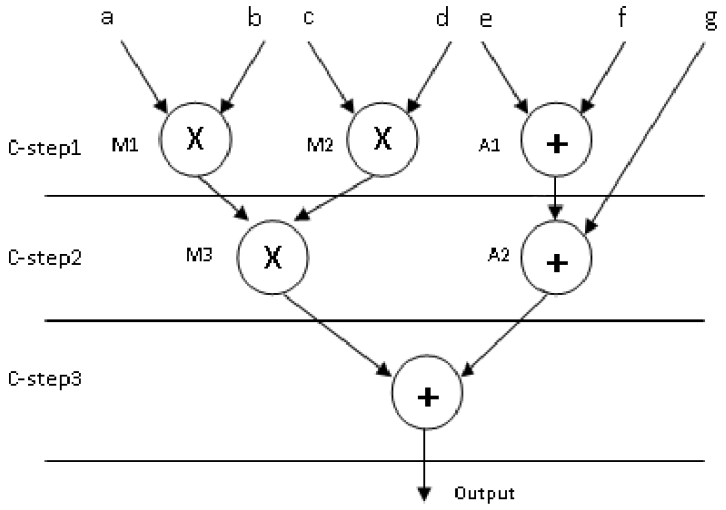


Figure 1: The new design schematic to meet the 4ns clock period requirement

ii) According to the new design, in addition the 2 multipliers and 1 adder needed to implement the design, the design will also need 4 multiplexers to control which input goes to the multiplier and adder blocks, and 3 registers (as seen in Figure 3), to store the output from each of the multiplier and adder blocks between each cycle. The assignment can be seen in Figure 2. The final schedule can be seen in Table 1. The final hardware diagram can be seen in Figure 3.

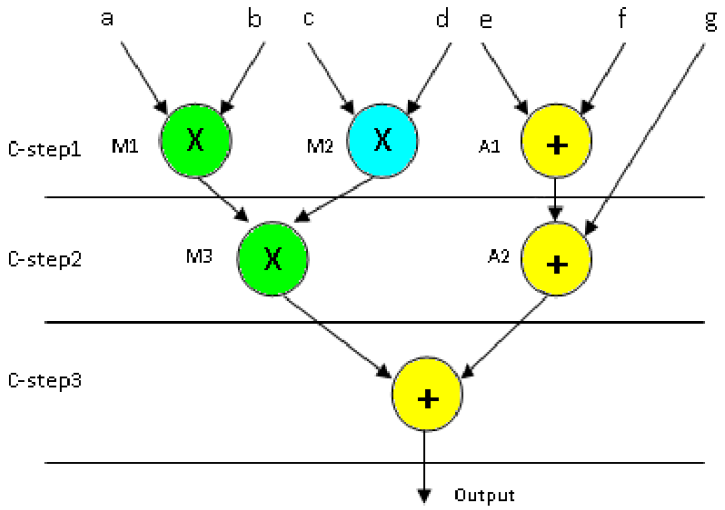


Figure 2: Assignment of the operations to different hardware block

Step	Mult 1	Mult 2	Add 1
C-step1	A × B	C × D	E + F
C-step2	Mult 1 × Mult 2		G + Add 1
C-step3			Mult 1 + Add 1

Table 1: Scheduling of block operations

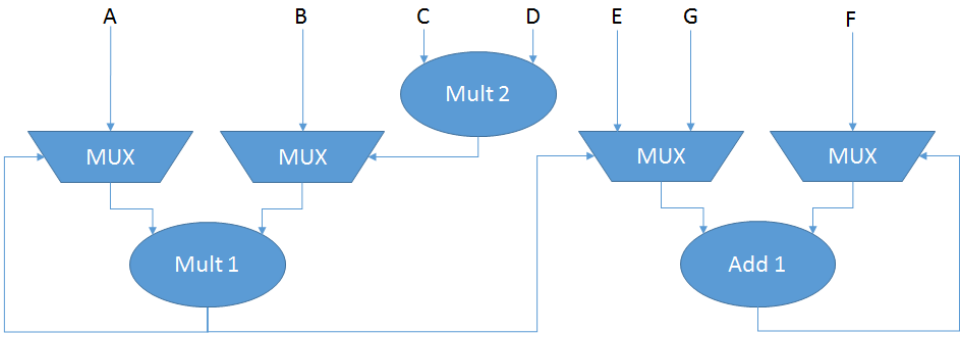


Figure 3: Block overview of the final implementation

b)

```
#include "ap_cin.h"

typedef uint4 t_data4;
typedef uint5 t_data5;
typedef uint8 t_data8;

t_data5 A[10];
t_data8 B[10], C[10];
t_data4 i;

for (i = 0; i < 10; ++i) {
  C[i] = A[i] + B[i];
}
```

The new code uses less area because of the reduced width of all the integers. Instead of having to create a 32-bit adder on the FPGA, which will require a lot of LUT's, all that is needed now is at 8-bit LUT adder, since the output value will never exceeed the 8-bit limit of 255. Also, the comparator needed to check the flow of the loop can be made smaller, since it is now a 4-bit integer instead of a 32-bit integer. In addition, since the arrays are also a lot smaller, they use less memory space than before, allowing them to be reshaped to fit into less memory.