IL2234: Digital Systems Design and Verification Using HDL

Homework 1

Fall 2025

Name: Andreas S. Pedersen
E-Mail: aspasp@kth.se
Date: September 13, 2025

Contents

Question 1

Model using HDL a look-up table using a 1-to-16 de-multiplexer that can act as a 4-bit one-hot (active-high outputs) decoder. The design should take as input a 4-bit binary value and output the one-hot equivalent.

Solution

The testbench output of the module is shown below.

Q1 - Testbench Output

```
[10] Starting simulation...
[010] in=1 sel=0 out=1
[020] in=1 sel=1 out=2
[030] in=1 sel=2 out=4
[040] in=1 sel=3 out=8
[050] in=1 sel=4 out=16
[060] in=1 sel=5 out=32
[070] in=1 sel=6 out=64
[080] in=1 sel=7 out=128
[090] in=1 sel=8 out=256
[100] in=1 sel=9 out=512
[110] in=1 sel=10 out=1024
[120] in=1 sel=11 out=2048
[130] in=1 sel=12 out=4096
[140] in=1 sel=13 out=8192
[150] in=1 sel=14 out=16384
[160] in=1 sel=15 out=32768
```

Question 2

Model using HDL a design that uses 16-to-1 multiplexers and implements a 4-bit binary to BCD (Binary-coded decimal) encoder.

Solution

The testbench output of the module is shown below.

Q2 - Testbench Output

```
[10] Starting simulation...
[010] binary=0000 bcd=0000 carry=0
[020] binary=0001 bcd=0001 carry=0
[030] binary=0010 bcd=0010 carry=0
[040] binary=0011 bcd=0011 carry=0
[050] binary=0100 bcd=0100 carry=0
[060] binary=0101 bcd=0101 carry=0
[070] binary=0110 bcd=0110 carry=0
[080] binary=0111 bcd=0111 carry=0
[090] binary=1000 bcd=1000 carry=0
[100] binary=1001 bcd=1001 carry=0
[110] binary=1010 bcd=0000 carry=1
[120] binary=1011 bcd=0001 carry=1
[130] binary=1100 bcd=0010 carry=1
[140] binary=1101 bcd=0011 carry=1
[150] binary=1110 bcd=0100 carry=1
[160] binary=1111 bcd=0101 carry=1
```

Question 3

Model a design using HDL that implements an 8-bit unsigned carry select adder (CSA). The CSA should be built using 4-bit adders. The design of the adder is given to you.

Solution

The testbench output of the module is shown below.

Q3 - Testbench Output

```
[24810] A=248 B=0 sum=248 carry=0
[24820] A=248 B=1 sum=249 carry=0
[24830] A=248 B=2 sum=250 carry=0
[24840] A=248 B=3 sum=251 carry=0
[24850] A=248 B=4 sum=252 carry=0
[24860] A=248 B=5 sum=253 carry=0
[24870] A=248 B=6 sum=254 carry=0
[24880] A=248 B=7 sum=255 carry=0
[24890] A=248 B=8 sum=0 carry=1
[24900] A=248 B=9 sum=1 carry=1
```

Question 4

Model a design using HDL that can count the number of 1s ina 4-bit binary number. For example, the 4'b0101 should output 3'b010.

Q4 - Testbench Output

```
[10] Starting simulation...
[010] A=0000 B=0
[020] A=0001 B=1
[030] A=0010 B=1
[040] A=0011 B=2
[050] A=0100 B=1
[060] A=0101 B=2
[070] A=0110 B=2
[080] A=0111 B=3
[090] A=1000 B=1
[100] A=1001 B=2
[110] A=1010 B=2
[120] A=1011 B=3
[130] A=1100 B=2
[140] A=1101 B=3
[150] A=1110 B=3
[160] A=1111 B=4
```

Question 5

Model using HDL a design that uses a 16-to-1 multiplexer and implements a 4-bit binary to gray encoder.

Solution

Model a design using HDL that can count the number of 1s ina 4-bit binary number. For example, the 4'b0101 should output 3'b010.

Q5 - Testbench Output

```
[10] Starting simulation...
[010] Binary=0000 Gray=0000
[020] Binary=0001 Gray=0001
[030] Binary=0010 Gray=0011
[040] Binary=0011 Gray=0010
[050] Binary=0100 Gray=0110
[060] Binary=0101 Gray=0111
[070] Binary=0110 Gray=0101
[080] Binary=0111 Gray=0100
[090] Binary=1000 Gray=1100
[100] Binary=1001 Gray=1101
[110] Binary=1010 Gray=1111
[120] Binary=1011 Gray=1110
[130] Binary=1100 Gray=1010
[140] Binary=1101 Gray=1011
[150] Binary=1110 Gray=1001
[160] Binary=1111 Gray=1000
```

Question 6

Model using HDL a design that can be used to implement an arithmetic right shift. Your design has to be parametric and have N-bit input, where $N \geq 4$. It should also allow to shift up to 3 positions. Do **not** use the `<<<`, `>>>` operators.

Draw a schematic of your design. Assume a value for the parameter $N = 5$.

Solution

The functionality can be implemented using an array of N-to-1 multiplexers. Denoting the input and output bit-vectors d_i and d_o respectively, and the 2-bit select line for s, the circuit schematic when $N = 5$ is shown in [Figure 1].

The output from the testbench of the implemented circuit is shown below.

Q6 - Testbench Output

```
[10] Starting simulation...
[010] Input=10101010 Control=0 Output=10101010
[020] Input=10101010 Control=1 Output=01010101
[030] Input=10101010 Control=2 Output=00101010
[040] Input=10101010 Control=3 Output=00010101
```

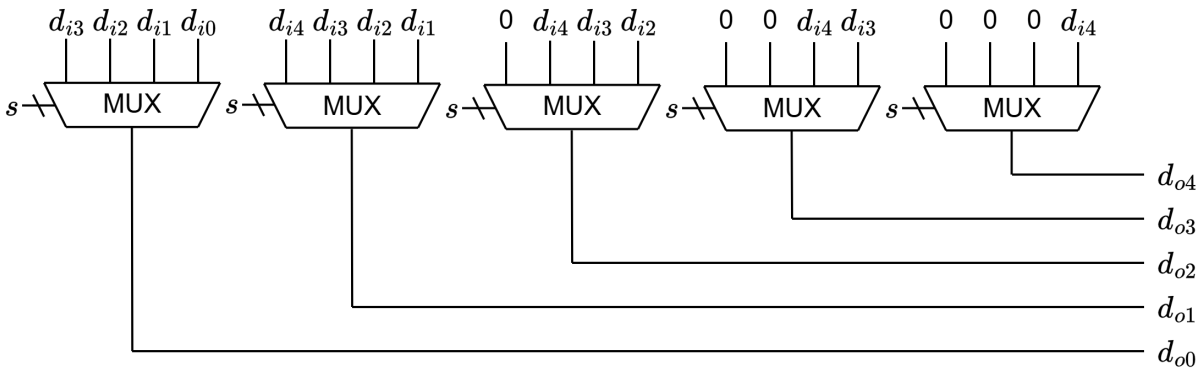


Figure 1: $N = 5$ right shifter implemented using an array of multiplexers.

Question 7

Model using HDL a signed N-bit multiplier. The design has to be parametric with 2 inputs of N-bit, where $N > 3$. Your design should use half adders and/or full adders.

Draw the schematic of your design. Assume a value for the parameter $N = 5$.

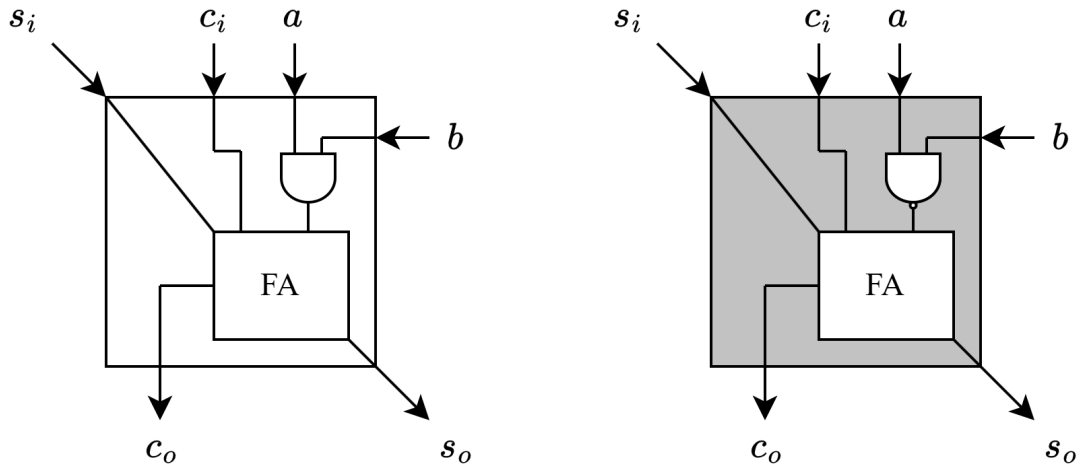
Solution

To implement signed multiplication the Baugh-Wooley multiplication algorithm is used. A Baugh-Wooley multiplier is implemented as a grid of "Baugh-Wooley cells". Two types of cells are used, white and gray [Figure 2]. The complete circuit diagram is seen in [Figure 3].

The testbench iterates through all 8-bit input values and confirms the correct result is produced by the multiplier. An example snippet is shown below.

Q7 - Testbench Output

```
[655270] a=127 b=118 prod=14986
[655280] a=127 b=119 prod=15113
[655290] a=127 b=120 prod=15240
[655300] a=127 b=121 prod=15367
[655310] a=127 b=122 prod=15494
[655320] a=127 b=123 prod=15621
[655330] a=127 b=124 prod=15748
[655340] a=127 b=125 prod=15875
[655350] a=127 b=126 prod=16002
[655360] a=127 b=127 prod=16129
Testbench succesfully completed!
```



(a) Baugh-Wooley multiplier white-cell.

(b) Baugh-Wooley multiplier gray-cell.

Figure 2: Baugh-Wooley multiplier basic cell construction.

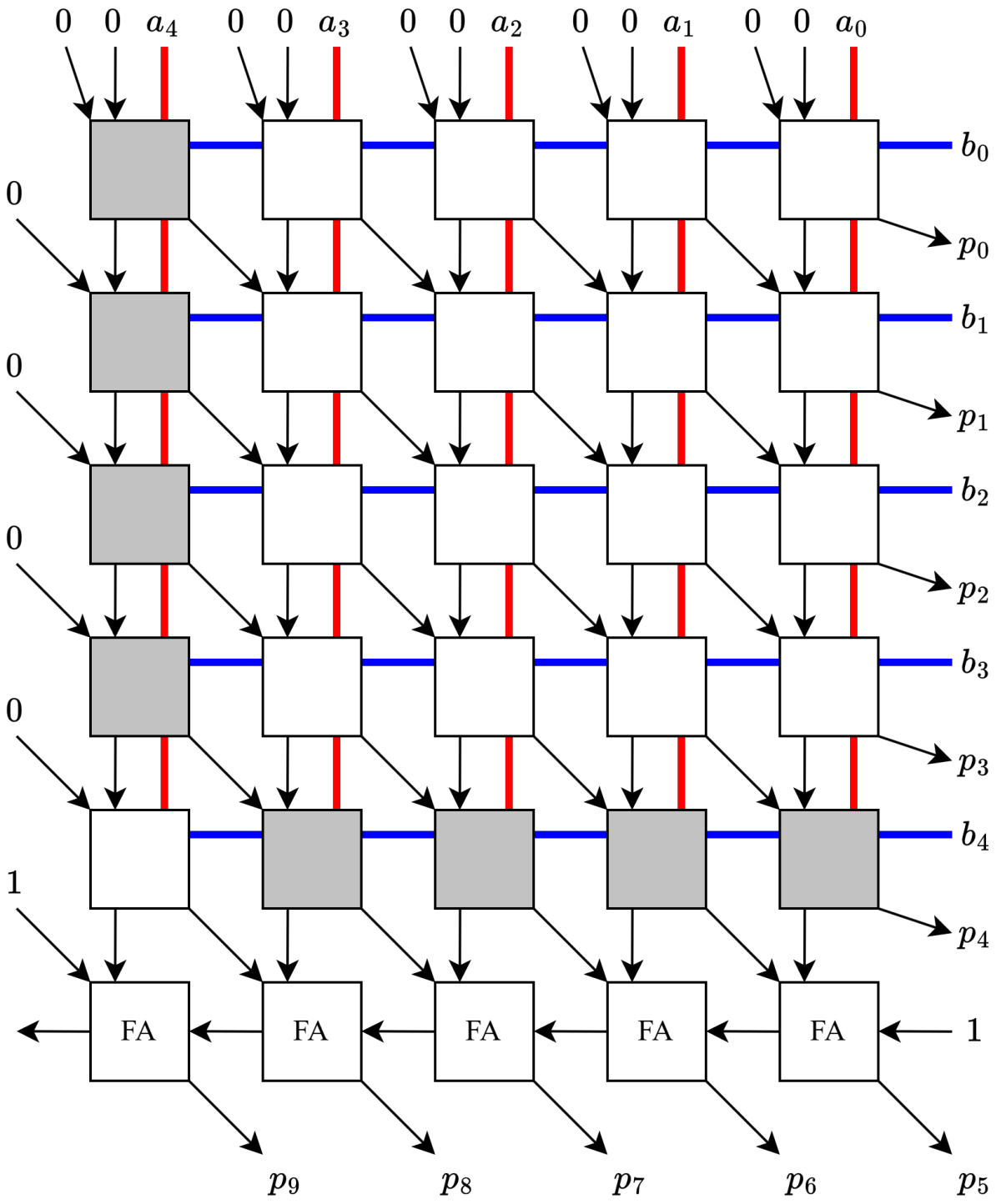


Figure 3: Block diagram of a $N = 5$ Baugh-Wooley multiplier implemented using basic Baugh-Wooley white- and gray-cells. Inputs are denoted a, b, and the product is denoted p.

Question 8

Model using HDL a design that can multiply 6 N-bit numbers and add the results. The design should be parametric with parameter N. You must ensure that your design does not overflow or underflow and assign the correct bit width for all intermediate and output signals. The design should implement the following equation:

$$\text{out} = \sum_{k \in 0,2,4} X_k \cdot X_{k+1}$$

Draw a schematic of your design. Assume a value for the $N =$ parameter.

Solution

The module is implemented using the previous Baugh-Wooley multiplier and multiple carry-lookahead adders. The schematic is shown in [Figure 4] and the testbench output is shown below.

Q7 - Testbench Output

```
[010] a=0 b=0 c=0 d=0 e=0 f=0 res=0 expected=0
[020] a=1 b=2 c=3 d=4 e=5 f=6 res=44 expected=44
[030] a=10 b=20 c=5 d=8 e=100 f=2 res=440 expected=440
[040] a=-1 b=10 c=-5 d=4 e=1 f=-1 res=-31 expected=-31
[050] a=127 b=127 c=127 d=127 e=127 f=127 res=48387 expected=48387
[060] a=-128 b=-128 c=-128 d=-128 e=-128 f=-128 res=49152 expected=49152
```

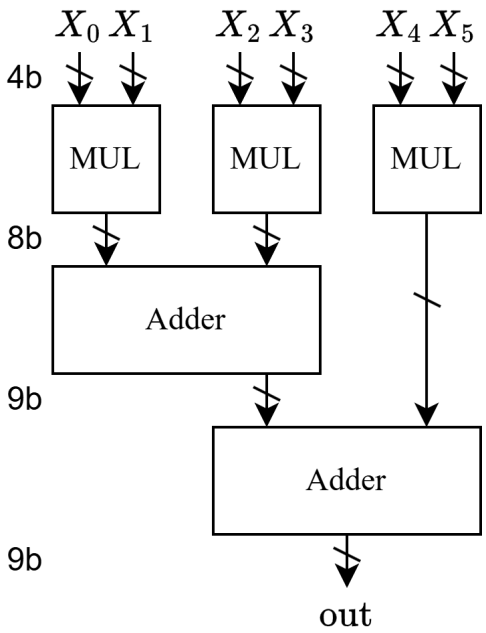


Figure 4: Block diagram of the 6 number multiply/accumulate module with $N =$.

Question 9

Model using HDL an unsigned multiplier that can be used to multiply two 16-bit numbers. The multiplier should be built usnig components of multiple 4-bit multipliers. The 4-bit multiplier can be modelled behaviorally by using the (*) operator.

Draw a schematic of your design.

Solution

We first construct a 8-bit multiplier from four 4-bit multipliers and three 16-bit adders as shown in [Figure 5]. Denoting the 8-bit inputs a, b, they are split into 4-bit halves a_L, a_H, b_L, b_H. The halves are fed into the 4-bit multipliers and their products are bit-shifted and added to produce the final 16-bit product. The same approach is taken to create the 16-bit multiplier, but instead of using behaviorally modelled 4-bit multipliers and 16-bit adders, it uses the 8-bit multiplier from [Figure 5] and 32-bit adders. The block diagram is shown in [Figure 6].

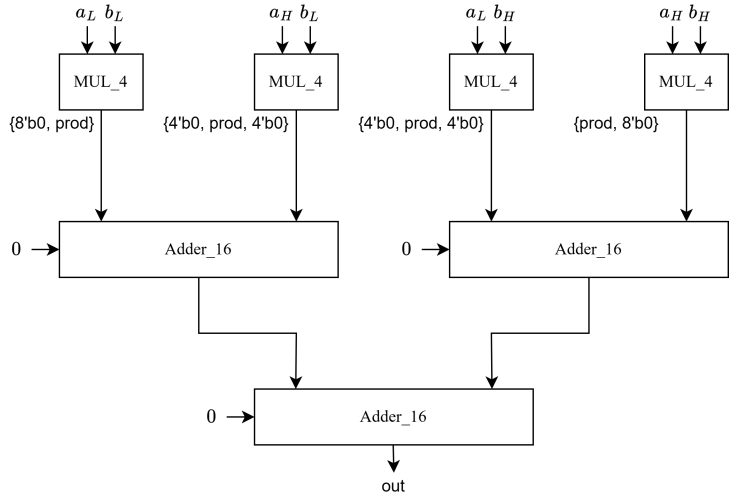


Figure 5: Block diagram of 8-bit multiplier made from 4-bit multiplier and 16-bit adder modules.

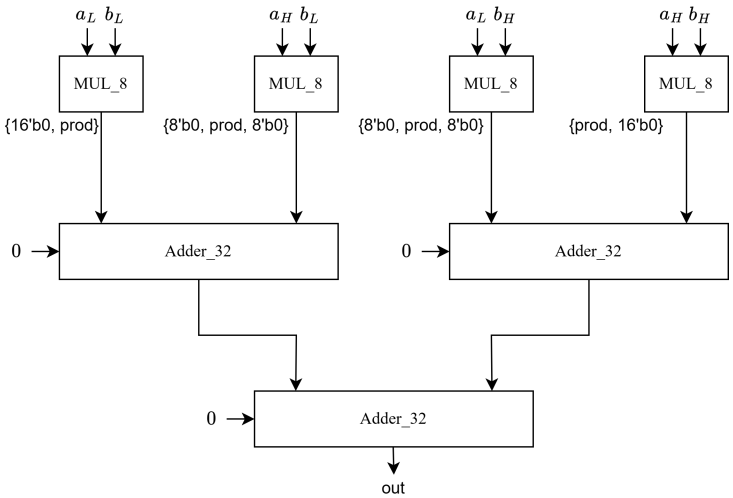


Figure 6: Block diagram of 16-bit multiplier made from 8-bit multiplier and 32-bit adder modules.