Laboratory work 3 for TSTE18 Digital Arithmetic Multiplication

Oscar Gustafsson

September 13, 2017

The purpose of this laboratory work is to realize different arithmetic operations with the help of a computer. The recommended programming languages to use are **Matlab**, **VHDL**, and **Verilog**. However, if you feel more comfortable with using another language feel free to do so, but make sure you have confirmed with the course responsible that they can understand that programming language and have a fair chance of running your instances. **It is OK to use different programming languages for different parts/labs.**

It is required that you use a digit-based representation format for all your data, i.e., arrays, rather than integer types. Feel free to write conversion functions for easy use, but the main realization should use a digit-based representation in the appropriate radix.

The laboratories are **nominally individual**. However, as there are a few more students than can fit in a lab, it is OK to use the smallest number of pairs possible per lab. Note that each student is only allowed to work in pairs a minimum number of times. Accordingly, even though you happen to be working in a pair, you must make sure that you understand everything you submit. Also, note that the pairings will be decided in the lab, based on the number of students attending. Hence, there is no point in planning the pairs beforehand as some students may choose to do the labs at home. **All source codes will be cross-correlated, so please write your own and do not get "inspired"**.

The reporting should consist of (emailed to `oscar.gustafsson@liu.se`):

- Source code
- Example run, showing the usage of the realization
- Some non-trivial examples showing the correctness of the realization
- Where applicable, make sure that all relevant intermediate results are also shown in the examples
- Where applicable, drawings (hand-written is OK scanned/photographed and emailed or handed in on paper)

For Matlab, an .m-file with the runs and the output log would be fine. For VHDL/Verilog a .do-file setting up the windows etc and providing stimuli or a testbench may be appropriate.

> It is OK to have separate functions for the different cases.

1 Booth-encoding

Realize a function can convert an array of binary numbers to a modified Booth-encoded representation for radix-4.

> To obtain an array of binary numbers from an integer in Matlab you can use:
> `dec2bin(57)-48`
> where 48 is the magic number to convert the ASCII numbers for 0 and 1 to 0 and 1.

2 Partial product array generation

Realize a function that can generate the partial product arrays for:

- unsigned multiplication of two binary arrays
- two's complement multiplication of two binary two's complement arrays using modified Baugh-Wooley sign-handling
- multiplication where modified radix-4 Booth encoding is used for one of the operands and two's complement for the other, use the function realized above

The partial product array should only contain binary digits, i.e., 0 or 1.

3 Complete multiplier

Realize a function that perform a summation of all the binary partial products generated in any of the three cases above and encapsulate it into three different multiplier functions taking two binary arrays as input. It is OK to use high-level functions to accumulate the results, e.g., `sum` in Matlab. However, the output should be bits and the multiplication must be performed by generating and accumulating the partial products, so no use of `*`.