

**ENCS 211**

**Section #2**

**Instructor: Dr. Hanna Bullata**

**Report for experiment 8**

**Verilog part 2**

**Sara Sameeh #1130501**

* **Abstract :**

**The aim of this experiment is using quartus to implement project (counter count at 2Hz & the result appear at seven segment ) by gathering many models (using counter , seven segment & frequency division as models ), & to implement any algorithm using Verilog language.**

* **Contents :**

Cover page \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_1

Abstract\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2

Theary\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_3

Procedure\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_5

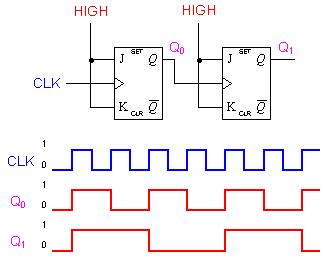
Conclusion\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_8

References\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_8

* **Theory :**

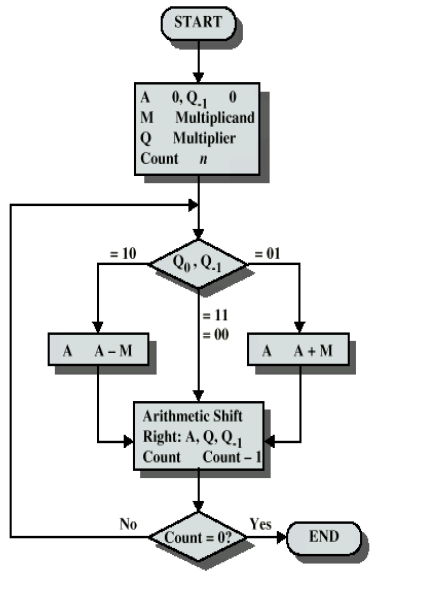
# In FBGA design, sometimes we need to change frequency, for example in our case (in this experiment) if we didn’t change it then we will not be able to see the output, to solve this problem we should take the high frequency & convert it to low frequency.

# One of components that reduce the frequency is the frequency divider which can be implemented using counter & ratio for input & output frequencies.



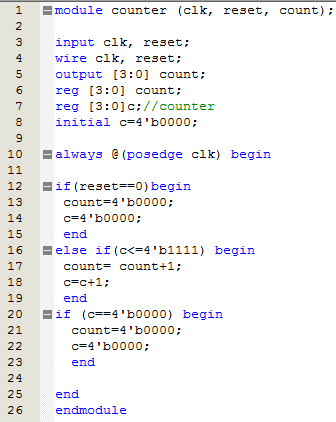
**Reducing the frequency**

## To multiply any two numbers, there is many algorithms to do this such as booth algorithm, this algorithm multiplies two signed binary numbers in two’s complement notation, it’s examines adjacent pairs of [bits](https://en.wikipedia.org/wiki/Bit) of the *N*-bit multiplier *Y* in signed [two's complement](https://en.wikipedia.org/wiki/Two%27s_complement) representation, including an implicit bit below the [least significant bit](https://en.wikipedia.org/wiki/Least_significant_bit), *y*-1 = 0. For each bit *yi*, for *i* running from 0 to *N*-1, the bits *yi* and *yi*-1 are considered. Where these two bits are equal, the product accumulator *P* is left unchanged. Where *yi* = 0 and *yi*-1 = 1, the multiplicand times 2*i* is added to *P*; and where *y*i = 1 and *y*i-1 = 0, the multiplicand times 2*i* is subtracted from *P*. The final value of *P* is the signed product. The multiplicand and product are not specified; typically, these are both also in two's complement representation, like the multiplier, but any number system that supports addition and subtraction will work as well. As stated here, the order of the steps is not determined. The algorithm is shown below:

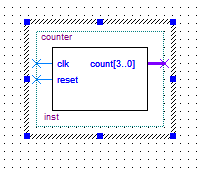


* **Procedure :**

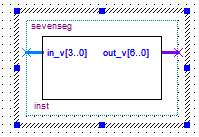
# First of all we built 4 bit counter using this Verilog code:



# Then we create symbol file 🡪 (File🡪create/update🡪create symbol file for current file).



# We also create symbol files for frequency division & seven segment as following:

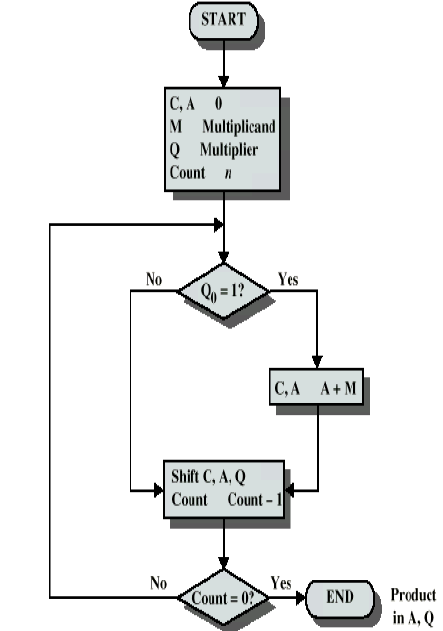
 

# Then we connect all of the previous models together as required:

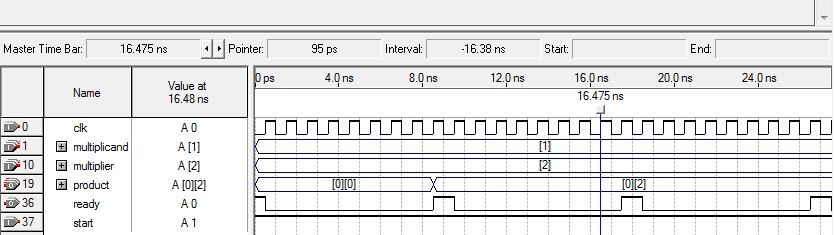
# 

**We compile & simulate it to make sure there is no error in our work then we assign the pins (clock, reset & output) using the manual of FBGA, also we change some of line in code of frequency division, we change counter to 13000000 to change the frequency, then we use FBGA & notice the output.**

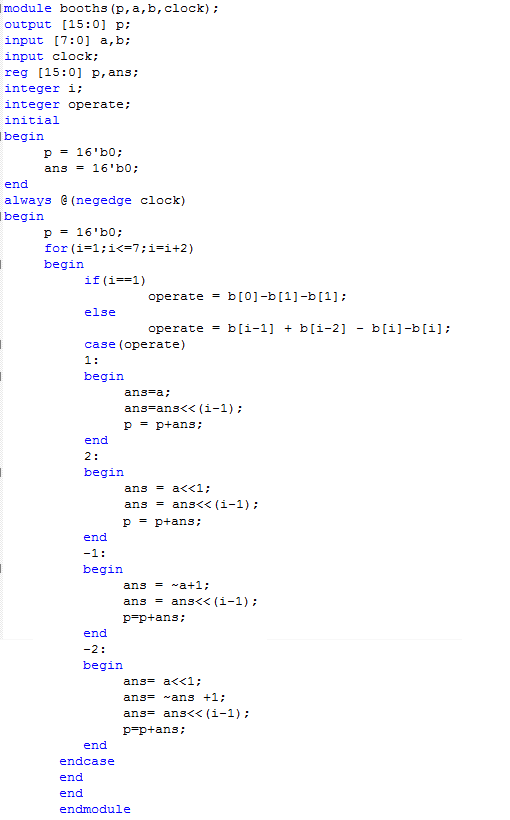
# Unsigned multiplier: to implement it we use shift & add as the following chart:



# Then we compile & simulate it, & the result was the following:



# To implement the booth algorithm we use the following Verilog code:



# After compile & simulate it to make sure that there is no error, & we got the following result:

# 

* **Conclusion :**

# In this experiment we become more familiar with quartus programming, we implements many modules like counter , frequency division & booth multiplier , also we realize the importance of FBGA which gives to user high level of flexibility to rapidly construct and test any hardware , in addition to that we learn how to change frequency to satisfy the aims of difference implementations.

* **References :**

<http://vlsicoding.blogspot.com/2013/11/verilog-code-for-4-bit-multiplier-using.html>

<https://www.google.ps/url?sa=t&rct=j&q=&esrc=s&source=web&cd=3&cad=rja&uact=8&ved=0ahUKEwjfqf3p45_JAhUCwxoKHanIBSQQFggrMAI&url=http%3A%2F%2Fhighered.mheducation.com%2Fsites%2Fdl%2Ffree%2F0070601755%2F366087%2FMB_MultiplierHDL_FPGA.pdf&usg=AFQjCNFj5cu4WnYvibdcsn4UWX5vQQUHg&sig2=EBW_9JLSbzmDCpFE2IZxOg&bvm=bv.108194040,d.d2s>