# Amjad Hamed ENG2025 Coursework Report –Parallel Multiplier

Student ID: 230733071

Module: Digital Electronics

Title: Structural VHDL Design of a 4x4-bit Parallel Multiplier

EDAPlayground Link: https://www.edaplayground.com/x/SUYt

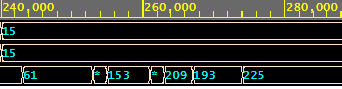
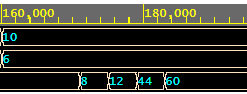
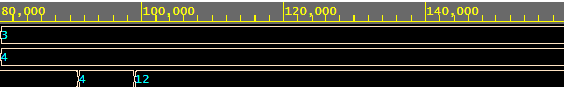
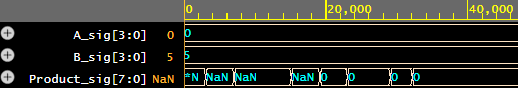
## Introduction: Design Process & Choices

In this design, a parallel multiplier is created using VHDL with structural and RTL-level designs for the basic gates and arithmetic components. The goal is to design a multiplier that can multiply two 4-bit binary numbers and produce an 8-bit result.

* Basic Gate Design:
  + AND gates are used to produce the product terms.
  + Full Adders (FAs) and Half Adders (HAs) are used for the addition steps. The FAs and HAs are designed using structural-level VHDL for clarity and modularity.
  + The final parallel adder uses a ripple-carry adder with carry-save addition to combine the product terms.
* Why Ripple-Carry Adder: A ripple-carry adder is used in the final row of the adder since it simplifies the design while ensuring correct functionality without the need for a carry-lookahead adder (CLA).
* Choice of RTL or Structural Design for XOR Gate: The XOR gate was implemented at the structural level using NOT, AND, and OR gates, as this approach simplifies debugging and provides the best understanding.

## Implementation Details

* The parallel multiplier works by first computing all product terms using AND gates. These terms are then added together using a combination of half adders and full adders.
* Delay Consideration: The ripple-carry adder used in the final addition stage introduces a delay proportional to the number of bits. For a 4-bit multiplier, the adder's delay increases linearly as the carry signal moves from between bits. While the ripple-carry adder is easy to implement, its delay impacts the overall speed of the multiplier, even more so for larger bits. For a 4-bit design, this delay is manageable, but for larger designs, more advanced adders like carry-lookahead adders (CLA) would be more much more efficient.
* The testbench is designed to test the multiplier with input values ranging from 0 to 15. The EPWave tool was used to visualise the simulation results.
* Testbench Setup:
  + Stimuli: Input values like A = 0, B = 5 A = 15, and B = 15 are applied to test multiplication for all combinations in the 0-15 range.
  + Simulation Results: The results were captured using EPWave, where the product is displayed for each multiplication.



## Conclusion

This design fulfils the required functionality of multiplying two 4-bit numbers and producing an 8-bit result. The choice of structural and RTL design ensured clear, understandable code. The testbench successfully verified the correctness of the design, and the simulation results confirmed the accuracy of the multiplication.