Project 46: Asynchronous Multiplier

A Comprehensive Study of Advanced Digital Circuits

By: Abhishek Sharma , Ayush Jain , Gati Goyal, Nikunj Agrawal

Documentation Specialist: Dhruv Patel & Nandini Maheshwari

Contents

1 Project Overview

The Asynchronous Multiplier project aims to design and implement a digital circuit that performs multiplication of binary numbers without relying on a global clock signal, thereby reducing power consumption and latency. The project involves creating input registers to store the numbers, generating partial products, and accumulating them using an asynchronous adder. Key techniques such as waveform logic, null convention logic, and bundled data protocols will be explored to ensure efficient operation. The design will be implemented using HDL (VHDL or Verilog) and tested through simulation tools like ModelSim or Vivado, with potential implementation on FPGA. Challenges include maintaining timing integrity and managing circuit complexity, while the expected outcome is a functional asynchronous multiplier with documentation detailing the design process and possible future optimizations.

2 Asynchronus Multiplier

2.1 Basic Concept of Asynchronus multiplier

The basic concept of an asynchronous multiplier revolves around performing multiplication without the use of a global clock signal. Instead of relying on synchronized clock edges to control operations, asynchronous circuits operate based on the completion of tasks, enabling different parts of the circuit to run at their own pace. Here are the key elements of an asynchronous multiplier:

- **Partial Product Generation:** When multiplying two binary numbers, partial products are generated based on the multiplicand and the bits of the multiplier. Each bit of the multiplier determines whether to include a shifted version of the multiplicand in the product.
- **Asynchronous Logic:** The design utilizes asynchronous logic gates that respond to changes in input signals rather than clock signals. This allows for a more flexible operation, as components can process data independently and signal when they are ready to proceed to the next stage.
- **Carry Accumulation:** The partial products are then accumulated using an asynchronous adder, such as a carry-save adder, which allows for the efficient addition of multiple values without the need for synchronous carry propagation.
- **Data Validity:** To ensure that the data is valid and correctly processed, control signals are used to indicate when the data is ready. This can involve techniques like bundled data, where data is sent along with a control signal to signify its validity.
- **Output Generation:** Finally, the accumulated product is stored in an output register for use. The output is generated as soon as the computation is complete, without waiting for a clock cycle.

2.2 Architecture of synchronus multiplier

The architecture of a synchronous multiplier typically involves several key components that work together in a synchronized manner, controlled by a clock signal. The major elements include:

- **Registers:**Input Registers store the multiplicand and multiplier values.Output Register stores the final result after the multiplication is completed.
- **Partial Product Generators:**These circuits generate partial products by multiplying the individual bits of the multiplicand with bits of the multiplier.This operation often involves combinational logic such as AND gates.
- **Adders (Carry Propagate Adder or Carry Save Adder):**Partial products are added sequentially or in parallel using adders.Adders like carry-save adders (CSA) are often used for faster computation by reducing carry propagation delays.
- **Control Unit:**The control unit is responsible for synchronizing the operations based on the clock signal.It controls the generation of partial products, the addition process, and ensures proper data flow between the registers and other components.

- **Shift Registers:**Shift registers are used to shift the bits of the partial products during the multiplication process.In many designs, the multiplicand or multiplier is shifted to align bits for addition in each clock cycle.
- **Clock Signal:**A clock signal ensures that each step of the multiplication process is carried out in a synchronized manner, with one step performed in each clock cycle.

2.3 Working of Asynchronus multiplier

The working of an asynchronous multiplier involves a series of steps that enable the multiplication of two binary numbers without relying on a global clock signal. Here's a breakdown of the key processes involved:

- **Input Preparation**The two binary numbers to be multiplied, typically referred to as the multiplicand (A) and the multiplier (B), are fed into input registers.These registers may be designed to hold the values temporarily until the multiplication process begins. **Partial Product Generation**Each bit of the multiplier (B) is examined to determine if it is 0 or 1.For each bit that is 1, a corresponding shifted version of the multiplicand (A) is generated:

  1. If the least significant bit (LSB) of B is 1, A is taken as is.
  2. If the next bit is 1, A is shifted left by one position, and so on.

  This results in a series of partial products based on the bits of the multiplier.

- **Asynchronous Logic Gates**The partial products are generated using asynchronous logic gates that activate based on the input signals. Unlike synchronous circuits, these gates do not wait for a clock signal; they respond immediately when inputs change.
- **Partial Product Accumulation**The generated partial products are accumulated using an asynchronous adder, such as a carry-save adder (CSA). A CSA allows for the efficient addition of multiple binary numbers by delaying carry propagation until all partial products have been generated.Each stage of addition can proceed independently, enabling faster accumulation.
- **Data Validity and Control Signals** Control signals indicate the validity of data throughout the process. For example:Once a partial product is ready, a signal is sent to indicate its availability.When the accumulation of partial products is complete, a signal denotes that the final product is ready for output.
- **Output Generation** Once the accumulation is finished, the result is stored in an output register. The output register holds the final product of the multiplication, which can then be read or used by other parts of the system.
- **Completion Signaling** After the multiplication is complete, a completion signal is sent to indicate that the result is ready for use. This allows other components in the system to respond accordingly without needing to synchronize with a clock cycle.

2.4 RTL Code

Listing 1: Asynchronus Multiplier

```
1
2  module async_multiplier (
3      input  logic [7:0] a, // 8-bit input 'a'
4      input  logic [7:0] b, // 8-bit input 'b'
5      output logic [15:0] product // 16-bit output 'product'
6  );
7
8      always_comb begin
9          product = a * b;
10     end
11
12 endmodule
```

2.5 Testbench

Listing 2: Asynchronus multiplier

```
1
2  module tb_async_multiplier;
3
4      logic [7:0] a, b;
5      logic [15:0] product;
6
7      // Instantiate the multiplier module
8      async_multiplier uut (
9          .a(a),
10         .b(b),
11         .product(product)
12     );
13
14     // Test stimulus
15     initial begin
16         $display("Starting test...");
17
18         // Test case 1
19         a = 8'd12;
20         b = 8'd10;
21         #10;
22         $display("a = %0d, b = %0d, product = %0d", a, b, product);
23
24         // Test case 2
25         a = 8'd15;
26         b = 8'd20;
27         #10;
28         $display("a = %0d, b = %0d, product = %0d", a, b, product);
29
30         // Test case 3
31         a = 8'd255;
32         b = 8'd255;
33         #10;
34         $display("a = %0d, b = %0d, product = %0d", a, b, product);
35
36         $finish;
37     end
```

```
38
39 endmodule
```

3 Results

3.1 Simulation

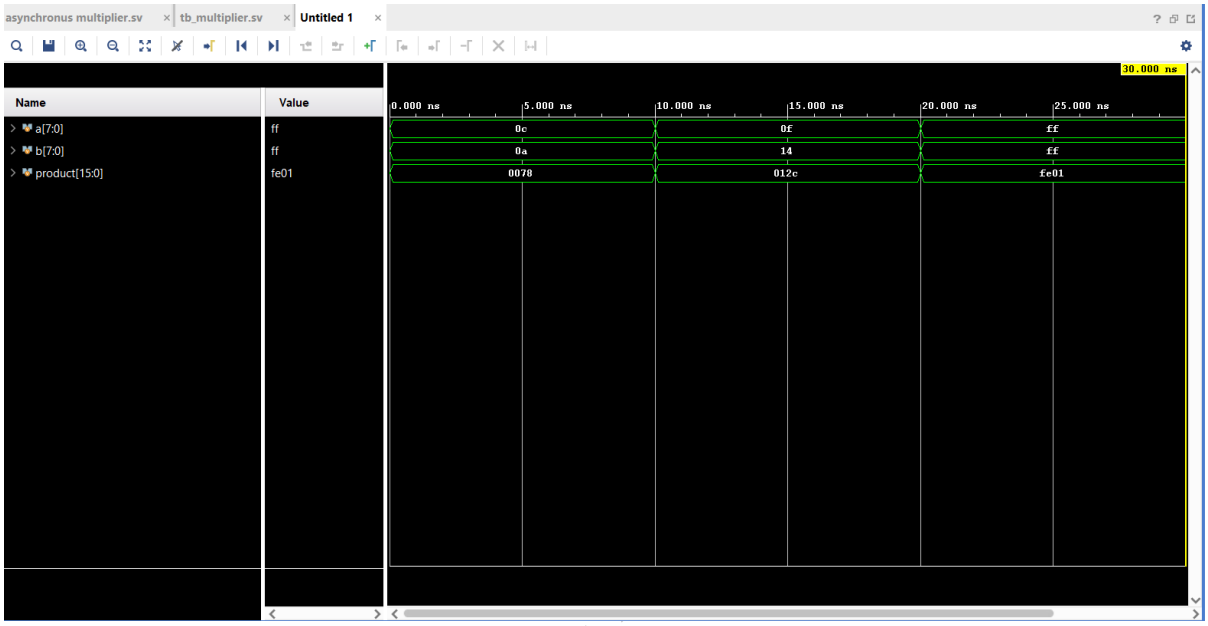

Figure 1: Simulation of Synchronus Multiplier

3.2 Schematic

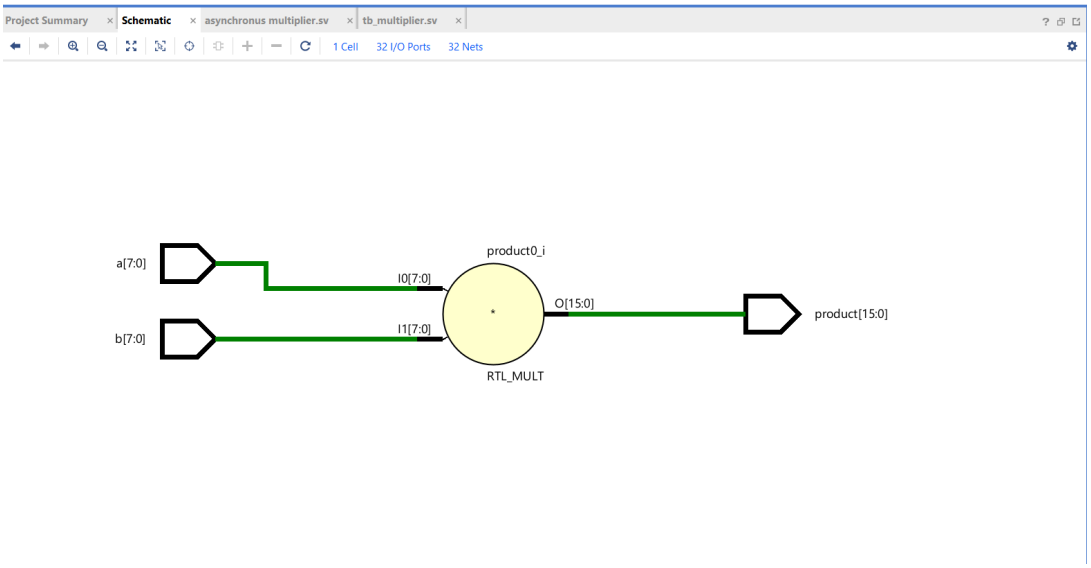

Figure 2: Schematic of Synchronus Multiplier

3.3 Synthesis Design

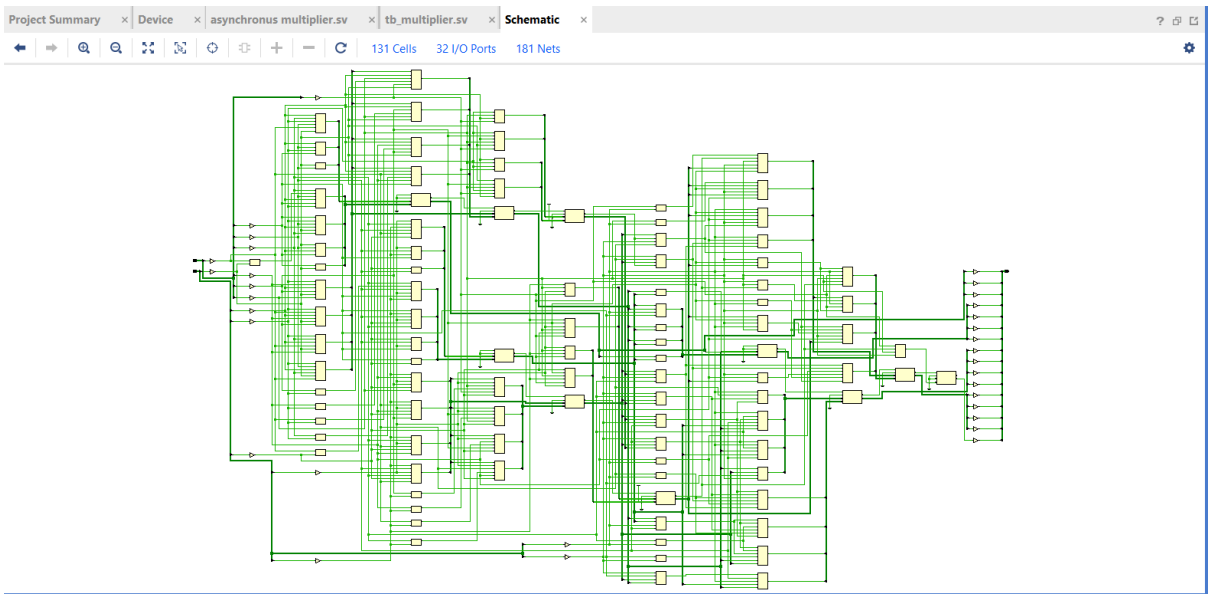

Figure 3: Synthesis Design of Synchronus Multiplier

4 Advantages of Asynchronus Multiplier

- **Lower Power Consumption:** Operates without a global clock, reducing dynamic power usage.
- **Reduced Latency:** Components can process data independently, leading to faster execution times.
- **Scalability:** Easier to scale for larger bit-width multipliers without significant design changes.
- **Flexible Timing:** Allows for varied operational speeds and can adapt to different workload demands.
- **Simplified Routing:** Potentially less complex routing than synchronous designs due to the absence of a clock distribution network.
- **Increased Reliability:** Reduced risk of timing errors and glitches associated with clock skew in synchronous systems.

5 Disadvantages of Asynchronus multiplier

- **Complex Design:** Asynchronous circuits can be more challenging to design and debug due to the lack of a global clock, requiring careful consideration of timing and data flow.
- **Timing Uncertainty:** Ensuring correct timing and synchronization can be difficult, leading to potential timing issues and race conditions.
- **Limited Tool Support:** Fewer design and simulation tools are available for asynchronous circuits compared to their synchronous counterparts.
- **Increased Development Time:** The complexity of asynchronous designs may lead to longer development and testing cycles.
- **Higher Area Utilization:** Asynchronous circuits may require more components and interconnections, potentially resulting in larger chip area.

- **Less Predictable Performance:** Performance can vary based on input conditions and circuit states, making it harder to guarantee consistent timing.

6 Applications of Asynchronus multiplier

- **Low-Power Devices:** Used in battery-operated devices like mobile phones, wearables, and IoT devices where energy efficiency is essential.
- **Digital Signal Processing (DSP):** Employed in audio, video, and image processing systems that require fast and efficient multiplication operations.
- **Embedded Systems:** Utilized in microcontrollers and FPGA designs where asynchronous circuits can reduce power consumption and increase performance.
- **Cryptography:** Applied in cryptographic algorithms that require high-speed arithmetic operations, benefiting from the reduced latency of asynchronous designs.
- **High-Performance Computing:** Integrated into specialized computing units that demand efficient data processing and lower power usage, such as in AI and machine learning applications.
- **Communication Systems:** Used in modems and communication hardware where quick multiplication is necessary for signal modulation and demodulation.
- **Real-Time Systems:** Employed in systems that require immediate response times, such as robotics and real-time data processing applications.

7 Conclusion

In conclusion, asynchronous multipliers represent a significant advancement in digital circuit design, offering unique advantages such as lower power consumption, reduced latency, and flexible timing. By operating without a global clock, they allow for independent processing of components, making them suitable for applications in low-power devices, digital signal processing, embedded systems, and high-performance computing. However, their design complexity, potential timing challenges, and limited tool support present hurdles that must be addressed. Despite these disadvantages, the continued exploration and implementation of asynchronous multipliers can lead to more efficient and responsive digital systems, particularly in scenarios where speed and power efficiency are critical. As technology evolves, the relevance of asynchronous designs is likely to grow, paving the way for innovative applications in the rapidly advancing fields of electronics and computing.

8 FAQs

1. What is an asynchronous multiplier?

An asynchronous multiplier is a digital circuit that performs multiplication of binary numbers without relying on a global clock signal. It allows different components to operate independently based on the completion of tasks.

2. What are the advantages of using an asynchronous multiplier?

Key advantages include lower power consumption, reduced latency, flexible timing, scalability for larger bit-width multipliers, simplified routing, and increased reliability due to fewer timing errors.

3. What are the disadvantages of asynchronous multipliers?

Disadvantages include complex design and debugging, timing uncertainty, limited tool support, increased development time, higher area utilization, and less predictable performance compared to synchronous designs.

4. In which applications are asynchronous multipliers commonly used?

Asynchronous multipliers are used in low-power devices, digital signal processing, embedded systems, cryptography, high-performance computing, communication systems, and real-time systems.

5. How do asynchronous multipliers improve power efficiency?

By eliminating the need for a global clock, asynchronous multipliers reduce dynamic power consumption, allowing components to activate only when needed and minimizing idle power usage.

6. Can asynchronous multipliers handle large bit-width numbers?

Yes, asynchronous multipliers can be designed to handle large bit-width numbers, often with improved performance characteristics compared to synchronous multipliers due to their scalable architecture.

7. What design techniques are commonly used in asynchronous multipliers?

Common techniques include waveform logic, null convention logic (NCL), and bundled data protocols, which facilitate efficient data processing and control without relying on clock signals.

8. How do you test and validate an asynchronous multiplier?

Testing typically involves simulation using HDL (VHDL or Verilog) tools like ModelSim or Vivado, followed by implementation on hardware platforms like FPGAs for practical validation of performance and functionality.

9. Are asynchronous multipliers suitable for all applications?

While they offer several advantages, asynchronous multipliers may not be the best choice for all applications, particularly where design simplicity, predictability, and robustness are priorities, as in some high-speed synchronous systems.

10. What is the future of asynchronous multipliers?

As technology advances, asynchronous multipliers are likely to see increased adoption, especially in low-power and high-performance applications, as researchers continue to explore new designs and optimize existing techniques.