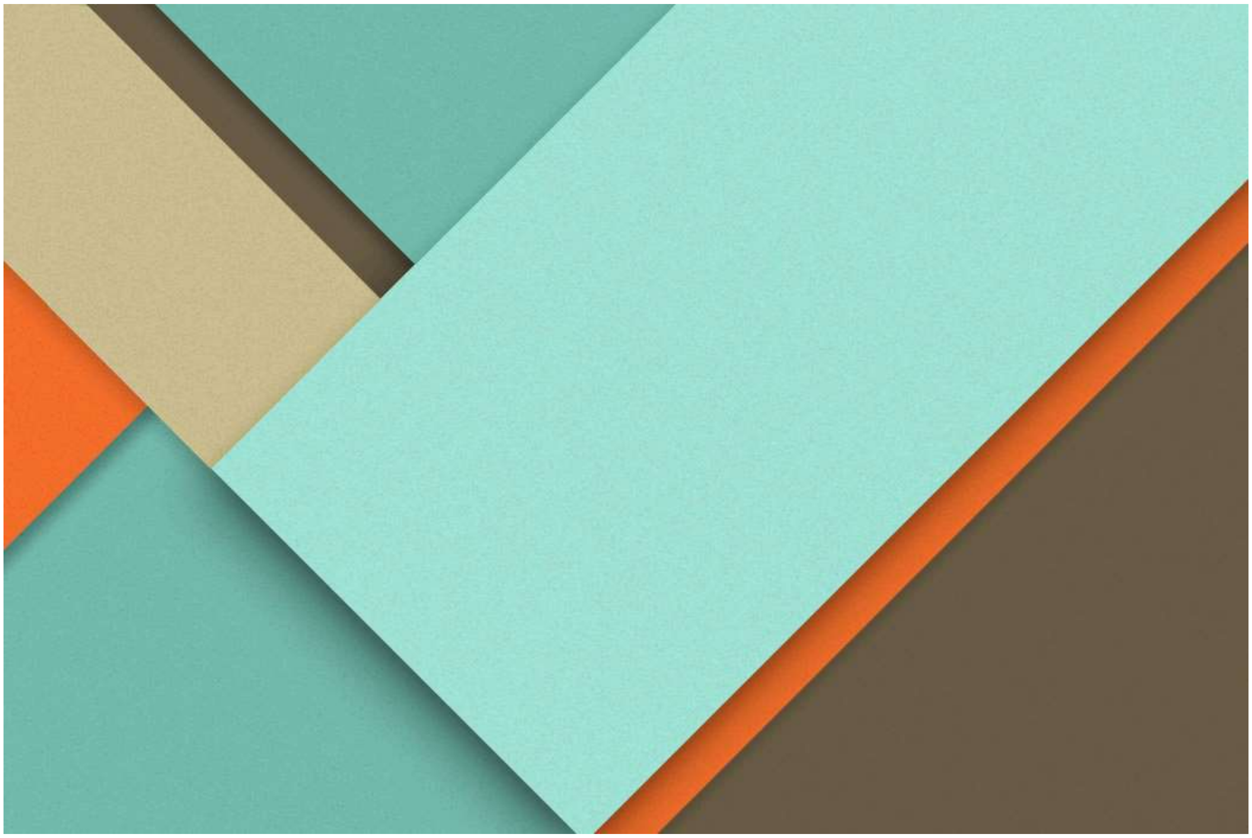


Program Counter using Verilog & VHDL

10.05.2024

—

Anish Godse
(ETRX Engg.)

INTRODUCTION :

- **Program Counter is a very important part of any microprocessor or microcontroller. Its job is to store the address of the next instruction to be executed. It is basically a combination of an up counter and PIPO register. For general instruction, PC simply increase the count by 1 after every instruction gets executed. For Branch instructions the PC directly loads the address of the next instruction to be executed. These are the 2 modes of operation of a PC. These 2 modes are selected using 2 control signals i.e increment & load.**

- **We have designed a 4-bit program counter capable of counting from 0000 to 1111 in binary, (0 to F) in hexadecimal, thereby enabling the execution of basic programs in digital systems**

METHODOLOGY :

- **The design methodology involves a combination of digital logic design principles and sequential circuit techniques. We used hardware description language (HDLs) such as VHDL for the design, simulation, and synthesis of the program counter.**

- **The implementation is simulated using software tools like Xilinx Vivado, and then synthesized for deployment on programmable logic devices such as FPGAs.**

- **We have implemented our design on Spartan-7 FPGA - Boolean board.**

Overview

- **This project involves use of VHDL and Hardware implementation.**
- **We are going to observe PC logic with a Seven Segment Display on FPGA.**

Goals

1. **Understand the concept of Hardware Description Language (VHDL)/verilog.**
2. **Implementation of logic on FPGA.**

Specifications

Components Used :

Sr.No	Component Name	Description
1	Boolean FPGA Board - (XC7s50csga324-1)	Used for Logic testing
2	Connecting wire	Used for code uploading

Sr.No.	Software Used	Description
1	Vivado Software	Used for Verilog/VHDL programming.

Working :

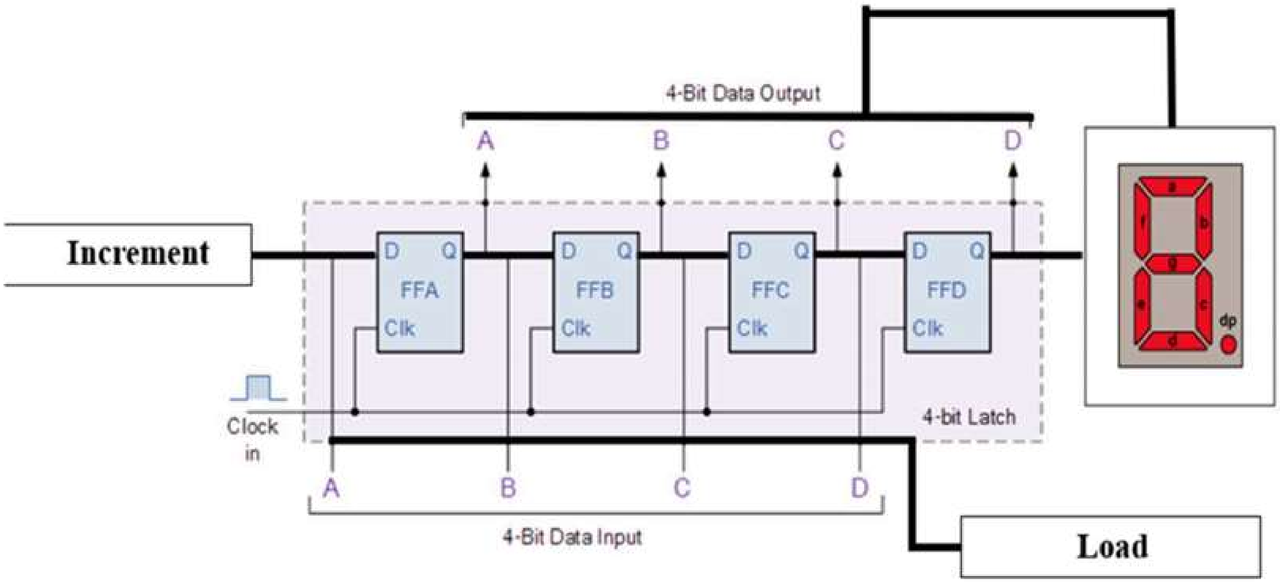
- **A 4-bit program counter operates as the memory address register within a microprocessor, orchestrating the sequential execution of instructions. Representing memory addresses from 0000 to 1111 in binary, it advances on each clock cycle, synchronizing with the processor's timing.**

- **During the instruction fetch phase, it accesses the next instruction's memory location, incrementing its value to move to the subsequent instruction. However, in branching scenarios, such as jumps or conditional branches, the counter may be modified to redirect execution to different memory locations. This alteration is managed by control logic, temporarily overriding the default increment operation.**

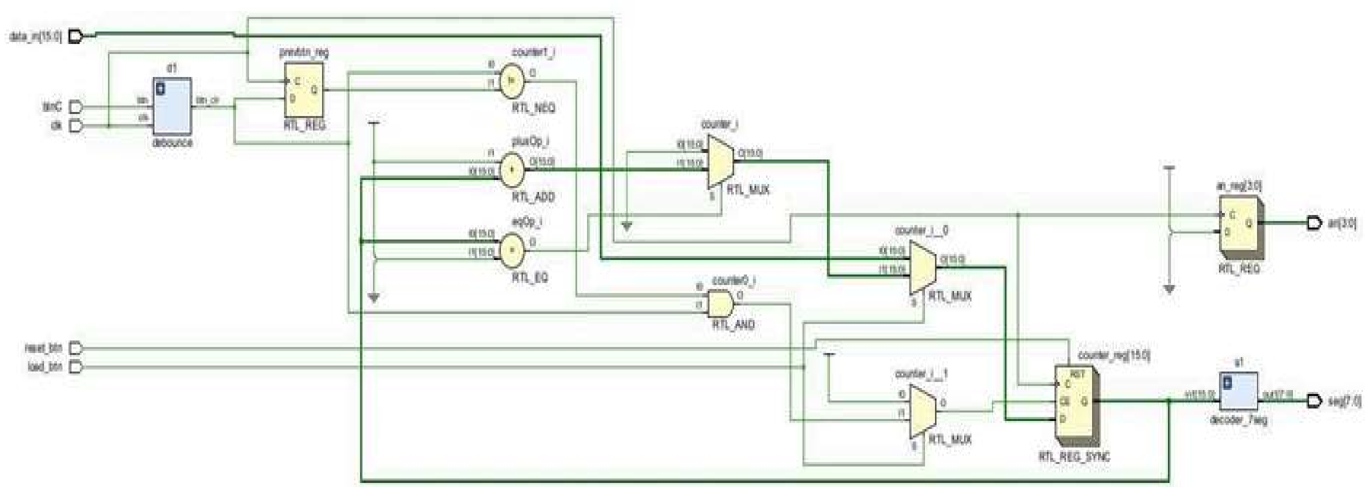
- **Upon power-up or reset, the counter initializes to a predetermined value, typically 0000, establishing the starting point for program execution. Its output serves as the address input for accessing program memory, ensuring the processor fetches instructions in the correct sequence for execution.**

- **As part of the microprocessor's architecture, the 4-bit program counter plays a fundamental role in maintaining instruction flow and program execution order.**

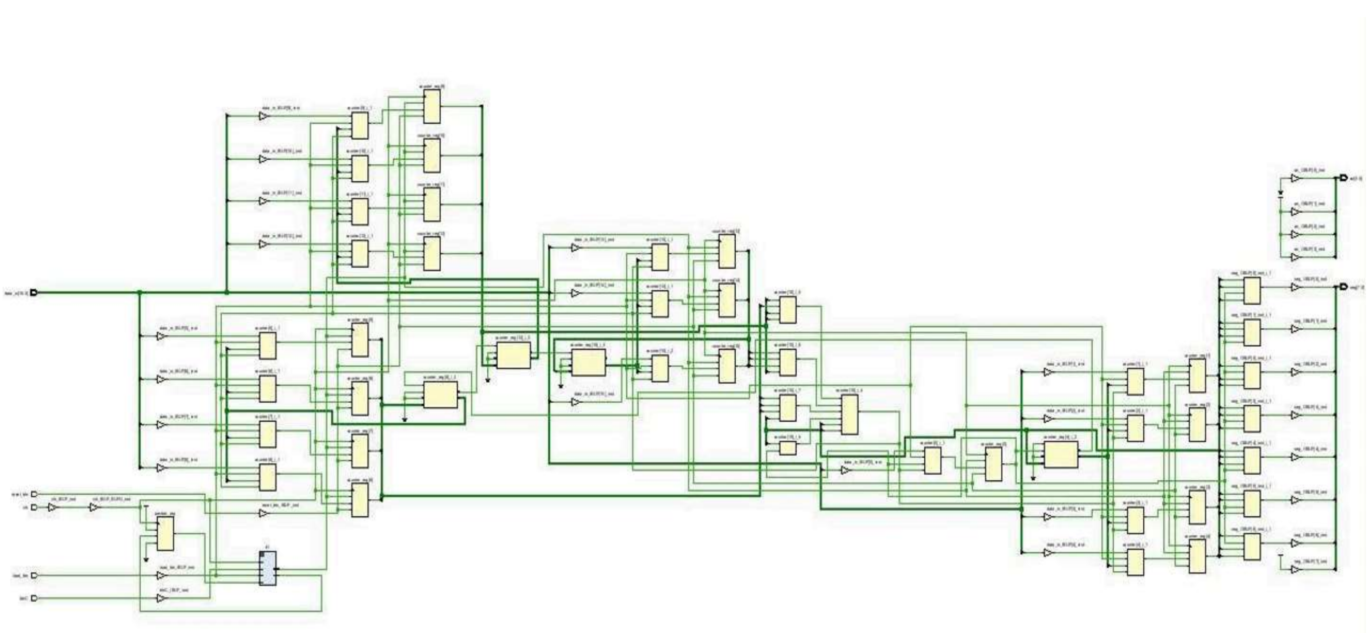
Block Diagram :



Schematic :



Synthesized Design and Schematic :



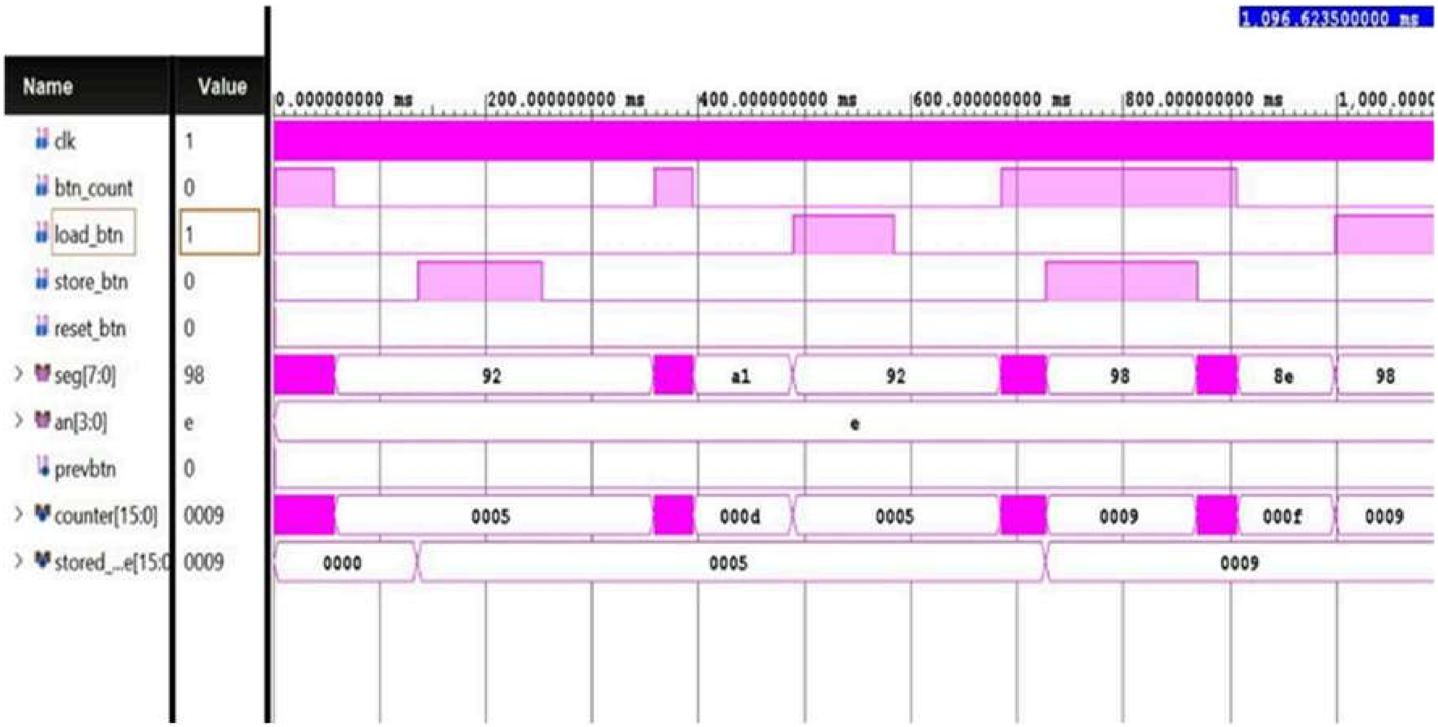
Results :

The implemented 4-bit program counter successfully counts from 0 to F in Hexadecimal, incrementing on each clock cycle when the increment button is pressed. When the load button is pressed it loads the input data value in the output signal.
Output signals accurately reflect the current count value. The design meets the specified objectives and performs reliably within the defined operational parameters.

Simulation:

1. **when increment button is pressed (PC is counting from 0 to F)**

2. **When the load button is pressed it gives input data value in output.**



Codes used :

- **VHDL code link :**

https://docs.google.com/document/d/e/2PACX-1vSGCKwNQ5VJpds4qnynmQ0Y_F-Q1xmaDVNs7yVB3q7SJkn_sUcv5hftGdZJLz9BxeFNpJ-H16SfPhla/pub

- **Verilog code link :**

https://docs.google.com/document/d/e/2PACX-1vS842TDmz2B6PPhfI2jAV6Og2vjPLWkJmj-wUihD71DfNq3KsZjofL1JW0Q0ZuvmxyXwEAk3z8syEAn/pub

- **Sample Output video of our project :**

https://drive.google.com/file/d/1i78sPInkYVmyKubj1UIyee_uBzZF3wh7/view?usp=sharing

- **Specifications of FPGA Boolean Board :**

https://drive.google.com/file/d/1wsLa2MUKY-FIN-vpQPwIY4Mj76Y5rAK9/view?usp=sharing

Applications :

1. **Microcontrollers and Microprocessors: In microcontrollers and microprocessors, a 4-bit program counter is often used to sequence through program instructions stored in memory. It helps fetch the next instruction address during each clock cycle.**

2. **Digital Signal Processors (DSPs): DSPs require program counters to execute instructions sequentially, so a 4-bit counter is suitable for simple DSP applications or as part of a larger address generation system.**

3. **Embedded Systems: In embedded systems with limited memory and simple instruction sets, a 4-bit program counter can efficiently manage program flow and execute basic operations.**

4. **State Machines: In finite state machines (FSMs), a 4-bit counter can serve as a state register, cycling through states based on external inputs or clock pulses.**

5. **Instruction Decoding and Control: The output of a 4-bit program counter is often used to select instructions from memory, enabling instruction decoding and control logic to execute the desired operations; this logic can be used for control purpose.**

6. **Address Generation: In memory-mapped systems, a 4-bit program counter generates memory addresses for accessing data or instructions stored in memory.**

Conclusion :

The design and implementation of the 4-bit program counter demonstrate its essential role in digital systems. The project achieves its objectives of providing a reliable execution. A 4-bit program counter may be suitable for simple digital systems and microcontrollers with limited memory and basic instruction sets; more complex applications typically require larger address spaces and more sophisticated addressing schemes. The choice of program counter width depends on the specific requirements of the application, including memory size, instruction set architecture, and performance goals.