LEVEL 2

KENNY NISANTH

February 2024

1 Introduction

At this level, you will learn System Design in Verilog. This level is divided into two parts namely Level-2.A and Level-2.B. At this level, test benches are also provided. Here in this level, we use all the basics discussed in LEVEL 1.

2 Pre-requisites

Basic knowledge of Digital System Design. Verilog HDL Language. Even if you don't know you will learn knowledge after reading the concepts

3 Tools/Requirements

Xiling Vivado To simulate and observe the behaviour of different logic we need the Xilinx Vivado tool. Vivado is software for synthesising and analysing hardware description language designs, superseding Xilinx ISE with additional features for system-on-a-chip development and high-level synthesis.

4 Order to learn

4.1 PART A

1) half-adder 2) full-adder 3) mux2to1 4) mux4to1 5) mux16to1 6) JKFF

7) DFF 8) TFF 9) bin2gray 10) gray2bin 11) enc4to2 12) dec2to4 13) compare 14) seg7

4.2 PART B

1) bin2bcd 2) minority detector 3) priority 4) moduloNbitcounter 5) parity generator 6) gcd

7) Factorial 8) ALU 9) Ringcounter 10) synchronous counter 11) bidirectional shift register

5 CONCEPT

HALF ADDER A half-adder is a basic digital circuit used in computer arithmetic to perform the addition of two binary digits. It has two inputs, typically labelled A and B, representing the two binary digits to be added. The half-adder produces two outputs: the sum (S) and the carry (C).

The sum output represents the least significant bit of the addition result, while the carry output indicates whether there is a carry-out to the next higher-order bit. The half-adder does not consider any carry-in from previous stages of addition.

The truth table for a half-adder is as follows:

"' A — B — S — C ———————— 0 — 0 — 0 — 0 0 — 1 — 1 — 0 1 — 0 — 1 — 0 1 — 1 — 0 — 1 "'

The sum output (S) is the result of the XOR operation on the inputs A and B, while the carry output (C) is the result

2 TO 4 DECODER A 2-to-4 decoder is a combinational circuit with two inputs and four outputs. It is primarily used in digital electronics to decode binary inputs into one of four possible output combinations. The two input lines represent the binary input, and based on the combination of these inputs, one of the four output lines is activated.

In a 2-to-4 decoder:

When both inputs are low (0, 0), the first output is activated while the other three outputs remain inactive. When the inputs are (0, 1), the second output is activated. When the inputs are (1, 0), the third output is activated. When both inputs are high (1, 1), the fourth output is activated. This type of decoder finds applications in various digital systems where data needs to be distributed selectively based on specific input conditions, such as in-memory addressing, data routing, and control signal generation.

4 TO 2 ENCODER A 4 to 2 encoder is a digital circuit that takes in four input signals (usually represented by four binary digits) and produces a two-bit binary output based on the active input signal. Essentially, it converts a four-bit binary number into a two-bit binary number.

The operation of a 4 to 2 encoder is such that only one input line is active at a time, and the encoder identifies which input is active and encodes it into a two-bit binary output. The output is typically in the form of two bits, where one bit indicates the most significant bit of the active input, and the other bit indicates the least significant bit.

This type of encoder is commonly used in digital systems where multiple inputs need to be encoded into a smaller number of output bits, such as in multiplexers, memory address decoding, or data compression schemes.

7 SEGMENT A 7-segment display is a type of electronic display device used to represent numbers and sometimes letters. It consists of seven individually controllable segments arranged in a pattern that can display numerals from 0 to 9 (and some letters). Each segment can be individually activated or deactivated to create different characters. 7-segment displays are commonly used in digital clocks, electronic meters, and other devices where numerical information needs to be displayed in a compact and easy-to-read format. They are simple and cost-effective, making them popular for various applications where basic numerical output is required.

BINARY TO GRAY Binary to Gray code conversion is a process used in digital electronics to transform binary numbers into Gray code. The Gray code, named after Frank Gray, is a binary numeral system where two consecutive values differ in only one-bit position. This property makes Gray code useful in applications such as rotary encoders and digital communications, where reducing errors during transitions between consecutive values is important.

The conversion from binary to Gray code involves XORing each bit of the binary number with the bit immediately preceding it, resulting in a new bit sequence that represents the Gray code equivalent. This process eliminates adjacent bit changes, reducing the likelihood of errors during transitions.

Overall, binary to Gray code conversion is a fundamental operation in digital electronics, enabling more reliable and efficient communication and signal processing.

COMPARATOR A comparator is a device or circuit that compares two voltages or currents and produces an output based on their relative magnitudes. It typically has two input terminals for the voltages or currents to be compared and one output terminal that provides a binary indication of which input is larger. Comparators are commonly used in electronic circuits for tasks such as signal conditioning, level detection, and waveform shaping. They are essential components in many electronic systems, ranging from simple voltage detectors to complex control systems.

D-FLIP FLOP A D-flip flop is a type of digital electronic circuit used in digital logic and sequential logic circuits. It is a one-bit memory element that stores the state (0 or 1) based on the input signal, often referred to as the "D" input.

The D-flip flop has a data input (D), a clock input (CLK), and outputs for both the current state (Q) and its complement (Q'). When the clock input transitions from low to high (or high to low, depending on the specific type of flip flop), the D-flip flop samples the input data and stores it in its memory element. This stored value is then output on the Q output.

D-flip flops are commonly used in applications where synchronization of data is required, such as in-memory elements, counters, and state machines. They are a fundamental building block in digital electronics and are widely used in various digital systems.

FULL ADDER A D-flip flop is a type of digital electronic circuit used in digital logic and sequential logic circuits. It is a one-bit memory element that stores the state (0 or 1) based on the input signal, often referred to as the "D" input.

The D-flip flop has a data input (D), a clock input (CLK), and outputs for both the current state (Q) and its complement (Q'). When the clock input transitions from low to high (or high to low, depending on the specific type of flip flop), the D-flip flop samples the input data and stores it in its memory element. This stored value is then output on the Q output.

D-flip flops are commonly used in applications where synchronization of data is required, such as in-memory elements, counters, and state machines. They are a fundamental building block in digital electronics and are widely used in various digital systems.

GRAY TO BINARY Grey to binary conversion is a process used to convert a Gray code, which is a binary numeral system where two consecutive values differ in only one bit, into standard binary code. The Gray code is commonly used in telecommunications and digital communication systems to reduce errors during transitions between consecutive values.

To convert Gray code to binary, you can use various methods such as the binary-reflected Gray code algorithm or the Gray to the binary conversion table. These methods involve systematically analyzing each bit of the Gray code and determining the corresponding bits in the binary representation.

Overall, Gray to binary conversion is essential for accurately interpreting and processing Gray code values in digital systems and communication protocols. of the AND operation on the same inputs.

JK FLIP FLOP A JK flip-flop is a type of digital electronic circuit that stores one bit of information. It has two inputs, J and K, and two outputs, Q and its complement Q. The flip-flop's state changes (or toggles) based on the input signals and its current state.

The behavior of a JK flip-flop is determined by its truth table, which specifies how the outputs change in response to different combinations of inputs. The flip-flop has three possible states: "set" (Q=1, Q=0), "reset" (Q=0, Q=1), and "toggle" (Q and Q change to their complement).

JK flip-flops are widely used in digital circuits for various applications such as counters, shift registers, and memory elements. They are versatile and can be cascaded together to create more complex sequential circuits.

MUX 16X1 A 16x1 multiplexer, often abbreviated as "MUX 16x1," is a digital circuit that selects one of 16 input signals and passes it to the output based on the value of the select lines. It has 16 input lines and 1 output line.

The select lines determine which input is routed to the output. MUX 16x1 is commonly used in digital systems for various purposes, such as data routing, signal selection, and digital communication. It's a fundamental building block in digital design and is widely used in electronic circuits and systems.

MUX 2X1 A MUX 2x1, short for multiplexer 2x1, is a digital logic component that selects one of two input signals based on a select input. It has two data inputs, typically labelled as D0 and D1, and one select input, usually labelled as S. The select input determines which data input is passed to the output. When S is low (0), the output equals D0; when S is high (1), the output equals D1. MUX 2x1s are commonly used in digital circuits for data routing and selection purposes. They are often found in applications such as data transmission, signal processing, and control systems.

6 MUX 4X1

A MUX 4x1, or multiplexer 4x1, is a digital logic circuit that selects one of four input signals and routes it to a single output line based on the control inputs. It has four data inputs, one output, and two control inputs, typically labelled as "select lines." The output of the MUX is determined by the combination of the select lines, which act as binary selectors to choose which input is passed through to the output. MUX 4x1 is commonly used in digital electronics for data routing and selection applications, such as in microcontrollers, data switches, and communication systems.

T-FLIP FLOP A T flip-flop, also known as a toggle flip-flop, is a type of digital circuit used in electronics and digital logic design. It has one input, typically labelled "T" or "toggle," and two outputs: the normal output (Q) and its complement (Q').

The behaviour of a T flip-flop is such that when the T input is triggered, the output state toggles or switches to its opposite state. If the flip-flop is in the set state (Q = 1), toggling the T input will reset it (Q = 0), and vice versa. Essentially, it acts as a single-bit memory cell that alternates between two states based on input pulses.

T flip-flops are commonly used in synchronous digital systems for various applications such as counters, frequency dividers, and frequency synthesizers. They are simple yet versatile components that play a crucial role in digital circuitry.

ALU An Arithmetic Logic Unit (ALU) is a fundamental component of a central processing unit (CPU) in a computer. Its primary function is to perform arithmetic and logical operations on input data provided by the CPU's registers or memory. The ALU can perform operations such as addition, subtraction, multiplication, division, AND, OR, XOR, and NOT.

Typically, an ALU consists of combinational logic circuits that execute arithmetic operations (such as addition and subtraction) and logical operations (such as AND, OR, and XOR). It may also include additional circuits for handling special operations like shifting, rotating, or comparing data.

The ALU takes input from the CPU's registers or memory, processes the data according to the specified operation, and then produces output results. These results can be stored back in registers, memory, or used as inputs for subsequent operations.

In modern CPUs, the ALU is a crucial component that operates at high speeds and is optimized for performance and efficiency. It plays a central role in executing the instructions of a computer program and performing the necessary calculations and data manipulation required for various tasks.

BIDIRECTIONAL SHIFT REGISTER A bidirectional shift register is a type of digital circuit that can shift its contents both left and right. It consists of a series of flip-flops connected in a chain, with each flip-flop storing one bit of data. The shift register also includes control logic to enable shifting in either direction.

When shifting data to the right, the contents of the register are moved sequentially, with the rightmost bit being shifted out and the remaining bits shifting to the right. Conversely, when shifting data to the left, the contents move in the opposite direction.

Bidirectional shift registers are commonly used in digital systems for various applications such as data transmission, storage, and processing. They provide a flexible means of manipulating data by enabling both left and right shifts, allowing for efficient data manipulation and processing in electronic devices.

BINARY TO BCD Binary-coded decimal (BCD) is a binary encoding scheme used to represent decimal numbers in digital systems. In a binary to BCD converter, the input is a binary number, typically in a 4-bit format, and the output is its equivalent BCD representation, which is expressed in decimal digits.

The conversion process involves breaking down the binary number into groups of four bits, which correspond to decimal digits ranging from 0 to 9. Each group of four bits is then converted into its decimal equivalent using a lookup table or combinational logic. The resulting BCD digits are then combined to form the final BCD representation of the input number.

Binary to BCD converters are commonly used in digital systems where decimal arithmetic is required, such as in digital clocks, calculators, and electronic instrumentation. They provide a means of efficiently converting binary numbers into a format that is compatible with decimal-based operations and displays.

FACTORIAL Factorial is a mathematical function denoted by the symbol "!" and is represented by the product of all positive integers up to a given number. For example, the factorial of 5 (written as 5!) is equal to $5 \times 4 \times 3 \times 2 \times 1$, which equals 120.

The factorial function is often used in mathematics and computer science to solve problems involving permutations, combinations, and probability. It is particularly useful in combinatorial problems where the number of possible arrangements or combinations of objects needs to be calculated.

In programming, factorial calculations are commonly implemented using iterative or recursive algorithms. Iterative algorithms involve looping through each integer from 1 to the given number and multiplying them together. Recursive algorithms, on the other hand, break down the factorial calculation into smaller subproblems until a base case is reached.

Factorials grow rapidly with increasing input values, leading to large results for relatively small inputs. For instance, 10! equals 3,628,800, and 20! equals 2,432,902,008,176,640,000. As a result, factorial calculations can quickly become computationally intensive for large input values.

GCD The Greatest Common Divisor (GCD) is a mathematical concept used to find the largest positive integer that divides two or more numbers without leaving a remainder. It is denoted by "gcd(a, b)" for two numbers a and b.

The GCD is calculated by finding the common factors of the given numbers and selecting the largest one. These common factors are the integers that evenly divide both numbers without any remainder.

There are several algorithms to calculate the GCD, with the most common ones being the Euclidean algorithm and the prime factorization method. The Euclidean algorithm iteratively reduces the problem of finding the GCD to simpler subproblems until a solution is found, while the prime factorization method involves finding the prime factors of each number and determining their common factors.

The GCD has various applications in mathematics and computer science, including simplifying fractions, solving linear Diophantine equations, and optimizing algorithms for efficiency. It is an essential concept in number theory and is widely used in various computational tasks.

MINORITY DETCTOR A minority detector is a digital circuit or algorithm designed to determine the minority value in a set of binary inputs. The inputs can be considered as a group of binary bits, and the minority detector identifies which value occurs less frequently within the set.

The operation of a minority detector involves comparing the number of occurrences of each binary value within the input set and selecting the value that occurs less frequently as the minority value. This process can be implemented using various techniques, such as counting the number of ones and zeros or using logical operations to identify the minority value directly.

Minority detectors have applications in various digital systems, including error detection and correction, data transmission, and fault tolerance. They are used to identify anomalies or discrepancies within binary data sets and enable systems to take appropriate action based on the detected minority value.

MOD N COUNTER A Mod-N counter is a type of digital counter that counts in a cyclic sequence of N states, where N is a positive integer. This counter can be implemented using flip-flops or other digital logic elements, and it generates output signals representing each state in the sequence.

The counter starts counting from 0 and progresses through the sequence of states until it reaches the maximum count value (N-1). Upon reaching the maximum count value, the counter resets to 0 and continues counting from there, creating a continuous cyclic pattern.

Mod-N counters are commonly used in digital systems for various applications such as frequency division, timing, and sequencing. They provide a straightforward way to generate repetitive patterns of digital signals and are often employed in clock generation circuits, event sequencing, and control systems.

The modulus (N) of the counter determines the number of states in the sequence, and it can be adjusted to meet specific requirements of the digital system. For example, a Mod-10 counter would cycle through ten states (0 through 9), while a Mod-16 counter would cycle through sixteen states (0 through 15).

PARITY GENERATOR A parity generator is a digital circuit that calculates and appends a parity bit to a group of binary data bits. The purpose of the parity bit is to enable error detection in data transmission or storage systems.

The parity generator calculates the parity bit based on the number of binary ones in the data. There are two common types of parity: even parity and odd parity. In even parity, the parity bit is set to make the total number of ones in the data, including the parity bit, an even number. In odd parity, the parity bit is set to make the total number of ones an odd number.

The operation of a parity generator involves counting the number of ones in the data bits and determining whether the parity bit should be set to 0 or 1 to achieve the desired parity. This process can be implemented using combinational logic circuits, such as XOR gates, to perform the necessary calculations.

Parity generators are commonly used in communication systems, storage devices, and memory systems to detect errors caused by data corruption during transmission or storage. By comparing the received parity bit with the calculated parity from the received data, errors can be detected and corrected if necessary.

PRIORITY Priority refers to the relative importance or precedence assigned to tasks, events, or elements within a system or process. In various contexts, priority determines the order in which tasks are performed, resources are allocated, or decisions are made.

In task scheduling and resource management, priority is often used to determine which tasks or processes are executed first. Tasks with higher priority are typically given precedence over those with lower priority, ensuring that critical or time-sensitive operations are completed in a timely manner.

Priority can also be applied in decision-making processes to determine the order in which options are considered or actions are take RING COUNTERn. For example, in a queuing system, customers with higher priority may be served before those with lower priority, or in emergency response situations, priority may be given to individuals in critical condition.

In computer systems and networking, priority is used to manage the flow of data packets, with higher-priority packets being given preferential treatment in terms of transmission and processing. This helps ensure that important data is delivered promptly and efficiently, even in congested or high-traffic environments.

Overall, priority plays a crucial role in various aspects of systems design, operation, and management, helping to optimize performance, allocate resources effectively, and meet the requirements of critical tasks or events.

RING COUNTER A ring counter is a type of digital counter circuit that consists of a ring of flip-flops connected in a closed loop configuration. Unlike traditional counters where each flip-flop represents a distinct binary state, in a ring counter, only one flip-flop is set to the high state (logic 1) at any given time, with the rest being in the low state (logic 0).

The operation of a ring counter involves shifting the high state from one flip-flop to the next in a cyclic manner, creating a rotating sequence of states. Each clock pulse causes the high state to move to the next flip-flop in the ring, effectively shifting the position of the active bit.

Ring counters are commonly used in digital systems for applications such as timing generation, frequency division, and sequencing. They provide a simple and efficient way to generate repetitive patterns of digital signals and are often employed in control circuits, communication systems, and signal-processing applications.

One advantage of ring counters is that they require fewer flip-flops compared to traditional binary counters, making them more compact and cost-effective for certain applications. However, they are limited by the fixed sequence of states and may not be suitable for all counting requirements.

SYNCHRONOUS COUNTER A synchronous counter is a type of digital counter circuit where all flip-flops are clocked simultaneously by the same clock signal. In contrast to asynchronous counters, where the clock inputs of flip-flops are connected in a cascaded fashion, synchronous counters ensure that all flip-flops change state simultaneously, typically on the rising or falling edge of the clock signal.

The operation of a synchronous counter involves a systematic counting sequence, where each flip-flop represents a bit in the counter's binary output. When the clock signal transitions, the flip-flops store their new state based on the inputs they receive, typically from the outputs of other flip-flops or external control logic.

Synchronous counters offer several advantages over asynchronous counters, including better timing control, reduced chance of glitches or hazards, and simplified design and analysis. By ensuring that all flip-flops change state simultaneously, synchronous counters can achieve more predictable and reliable performance, especially in high-speed digital systems.

Synchronous counters are commonly used in applications requiring precise timing, such as clock generation, frequency division, event counting, and digital signal processing. They are an essential component of many digital systems and play a crucial role in various electronic devices and equipment.

7 What have I done?

At this level, I have implemented the above concepts using Verilog and verified their outputs. I have written all codes according to the given module syntax in Xilinx Vivado and performed a simulation to observe the functionalities of logic gates according to given input signals. Also, obtained RTL Schematics for each logic gate, and for further use, I am making a collection of each logic gate code on Github under project "FPGAspeaks".

8 For more information

https://github.com/KENNYNISATH/FPGAspeakshttps://github.com/KENNYNISATH/FPGAspeaks