Name: R.MEGHANA

Affiliation (Institution/Company): NITTE MEENAKSHI INSTITUTE OF TECHNOLOGY,BANGALORE.

Email: meghanarayanki@gmail.com

Phone: 7406879355

Title of Internship Program Undertaken: Design and Verification using Verilog – DV08

Assignment No and Title: 1)4-Bit Serial Adder

2)APB Protocol

**Important Instructions:**

1. All assignment results should be computer screenshot or computer typed. Handwritten and scanned copies shall not be considered for evaluation
2. Due date for all assignment submission is 1 Week from the last date of internship
3. All assignment questions should be captured along with solutions/answers.
4. Code snippets, simulation results should be captured properly
5. Use only the JPEG image format for capturing the simulation results and name/label the results appropriately.
6. The description of answers should be short and crisp. Provide only the required information, answered copied or cut and pasted from google shall not be considered.

|  |
| --- |
| **C:\Users\LOHITH\AppData\Local\Temp\Temp1_png2jpg.zip\Screenshot (61).jpg**  **Solution:**  **Brief Explanation Of 4-Bit Serial Adder With Parallel Load**  **=>It consists of two 4-bit shift registers with parallel load, a full adder, and a D-type flip-flop for storing carry-out. In order to load data (4-bit) to these shift registers initially, shift capability of the registers should be enabled and loading mode should be enabled. Loading of numbers to these shift registers will occur in one clock cycle. After this, shifting mode should be enabled to perform the arithmetic operation. The addition of numbers stored in these shift registers requires 4 clock cycles. Starting with LSB (least significant bit), at each cycle one bit of first number and one bit of second number are being added. The sum is stored at the MSB (most significant bit) of first shift register as in circuit above. Carry-out output produced after each cycle is fed-back to the full adder as a carry-in of the next significant bit. For this purpose one D-type flip-flop is used as a temporary storage element. The LSB of the data in second shift register is fed as the input to the MSB of the same data. Hence rotation operation of this second shift register happens.**  **=>Here initially the two shift registers are loaded with data 1011 and 1000 respectively. Initially Cin is set as 0. When reset=1 data is fed parallely to these shift registers and output is 0. When reset=0, shift operation is performed be these registers. A variable count is declared because we just have to shift data 4 times in order to calculate sum of a 4-bit number. Therefore when count<4, enable=1 the circuit adds the LSB bits. When count>0 and enable=0 the same output value is retained.**  **1)Code** C:\Users\LOHITH\AppData\Local\Temp\Temp1_vlsi assignment jpeg.zip\Screenshot (62).jpg  C:\Users\LOHITH\AppData\Local\Temp\Temp1_vlsi assignment jpeg.zip\Screenshot (63).jpg |
| C:\Users\LOHITH\AppData\Local\Temp\Temp1_vlsi assignment jpeg.zip\Screenshot (64).jpg  C:\Users\LOHITH\AppData\Local\Temp\Temp1_vlsi assignment jpeg.zip\Screenshot (65).jpg |

|  |
| --- |
| **2)Test Bench**  **C:\Users\LOHITH\AppData\Local\Temp\Temp1_vlsi assignment jpeg.zip\Screenshot (67).jpg**  **3)Simulation and waveform**  **C:\Users\LOHITH\AppData\Local\Temp\Temp1_vlsi assignment jpeg.zip\Screenshot (70).jpg**  C:\Users\LOHITH\AppData\Local\Temp\Temp1_vlsi assignment jpeg.zip\Screenshot (69).jpg |
| C:\Users\LOHITH\AppData\Local\Temp\Temp1_png2jpg.zip\Screenshot (60).jpg  **Solution:**  **Brief Explaination**  **=>The APB is part of AMBA 3 protocol family. It provides a low-cost interface that is optimized for minimal power consumption and reduced interface complexity. The APB interfaces to any peripherals that are low bandwidth and has unpipelined protocol. All signal transitions are only related to the rising edge of the clock to enable the integration of APB peripherals easily into any design flow. Every transfer takes at least two cycles. It can interface with the AMBA AHB-Lite and AMBA AXI. You can use it to provide access to the programmable control registers of peripheral devices.**  **=>PCLK is the clock signal. PRESETn is a active low signal. PSEL is used to select a particular peripheral from multiple peripherals that are connected to a APB interface. If PWRITE=1, it performs write operation else it performs read operation. PREADY is the output from DUT used for putting wait states for your transactions.**  **=>Working of APB can be analyzed in 3 states: 1)Idle state:(PSEL=0 and PENABLE=0) 2)Setup state:(PSEL=1 and PENABLE=0) 3)Access State:( PSEL=1 and PENABLE=1)**  **=>APB Write Cycle: APB is in idle state initially. When PSEL is made high, then APB enters into Setup state. Now control signal is provided i.e, we assert PWRITE and we are giving address for data transfer. This PWRITE and PADDR should be high till we enter access state. Now when PENABLE is also made high, APB enters into Access state. Now data transfer from PWDATA to PADDR takes place when PREADY is high. If PREADY is low then we introduce wait states.**  **=>APB Read Cycle: It is same as APB Write cycle but here PWRITE is low and when PREADY is high, data is put on the PRDATA bus.**  **=>Read\_enable is made high when PSEL,PREADY,PENABLE are high and PWRITE is low whereas, Write\_enable is made high when PSEL,PREADY,PENABLE and PWRITE are all high.**  **=>Here function check\_wr\_rd\_data() is used to check whether the data written from PWDATA to PADDR or the data read from PADDR to PRDATA is same or no. If data is correct, it prints “good data matched” else “bad data matched” is printed.**    **1)Code**  C:\Users\LOHITH\AppData\Local\Temp\Temp1_vlsi assignment jpeg.zip\Screenshot (46).jpg  C:\Users\LOHITH\AppData\Local\Temp\Temp1_vlsi assignment jpeg.zip\Screenshot (47).jpg  **2)Test Bench**  C:\Users\LOHITH\AppData\Local\Temp\Temp1_vlsi assignment jpeg.zip\Screenshot (48).jpg  C:\Users\LOHITH\AppData\Local\Temp\Temp1_vlsi assignment jpeg.zip\Screenshot (49).jpg  C:\Users\LOHITH\AppData\Local\Temp\Temp1_vlsi assignment jpeg.zip\Screenshot (50).jpgC:\Users\LOHITH\AppData\Local\Temp\Temp1_vlsi assignment jpeg.zip\Screenshot (51).jpg  **3)Simulation and waveform**  **C:\Users\LOHITH\AppData\Local\Temp\Temp1_vlsi assignment jpeg.zip\Screenshot (52).jpg**  C:\Users\LOHITH\AppData\Local\Temp\Temp1_vlsi assignment jpeg.zip\Screenshot (53).jpg  C:\Users\LOHITH\AppData\Local\Temp\Temp1_vlsi assignment jpeg.zip\Screenshot (56).jpg  C:\Users\LOHITH\AppData\Local\Temp\Temp1_vlsi assignment jpeg.zip\Screenshot (57).jpg  C:\Users\LOHITH\AppData\Local\Temp\Temp1_vlsi assignment jpeg.zip\Screenshot (58).jpg |