

Assignment

To design a 16-bit Barrel shifter for your microprocessor.

Description

The shifter is an essential element for many microprocessor operations. It may be used to align or scale data, manipulate bits and bytes, or it may be used in an automatic or program-controlled shift-and-add multiply function. In the baseline machine, you are only required to implement a Logical Shift, which shifts data in a register (Rdest) as specified by a signed count operand (two's complement). A positive count specifies a left shift; a negative count specifies a right shift. You must support a shift amount contained in a register or the immediate field of the instruction. All bits shifted out of the destination register are lost. All destination bits not mapped from the original operand are filled with zeros.

Other common shift functions, which could be added to the baseline processor, are arithmetic shifts, arithmetic shift including the carry bit, byte swap, and rotate. In right arithmetic shifts, the high-order bits are filled with the original sign bit.

To shift the contents of a register one bit per clock cycle, an ordinary shift register, which can be assembled from cascaded D-latches, could be used. However if there is a need to shift data by an arbitrary number of bit positions within one clock cycle, a dedicated, programmable shifter is required.

Two frequently used approaches are (i) Barrel Shifters, and (ii) Logarithmic Shifters. While the barrel shifter implements the whole shifter as an array of pass transistors or multiplexers, the logarithmic shifter uses a staged approach. In general, barrel shifters are appropriate for small shift widths and logarithmic shifters are more suitable for shifters of large width. Both types of shifters can be implemented using full CMOS transmission gates or NMOS pass gates. NMOS pass gate designs can benefit from the addition of level restorers, which help to avoid problems associated with V_{th} drops. Shifters should be disabled in low-power designs when they are not in use.

Barrel Shifter

A simple 4x4, transmission-gate barrel shifter is shown in the Rabaey text (Figure 11-37, page 595). It consists of an array of transmission gates, with the number of rows equal to the bit width of the data words, and there is one column for each shift possibility. In this case, both are set equal to four. The input to the shifter is the value to be shifted, a literal <6:0> and the shift amount <3:0>. Weste and Eshraghian's Table 8.7 shows how different types of shifters can be implemented through the appropriate choice of literals. To do left shifts or rotates, the shift control inputs must be swapped. The control wires are routed diagonally through the array. A major advantage of this shifter is that a signal has to pass through at most one transmission gate, regardless of the size of the shifter. The shifter delay does, nevertheless, grow linearly with the size of the shifter, as the capacitance at the buffer inputs increases with additional columns.

The layout size of this shifter is dominated by the number and pitch of wires, rather than by transistor area. The size of the shifter grows linearly with the number of shift positions allowed. A decoder is required to select one of the shift lines. This decoder is more easily implemented in the controller (CAD8) should you choose to implement a barrel shifter. A good barrel shifter is likely to be the fastest, smallest, and easiest design for 16 bits.

Logarithmic Shifters

In logarithmic shifters, the total shift value is decomposed into shifts over powers of two. A shifter with a maximum shift width of M consists of $\log_2 M$ stages, where the i^{th} stage either shifts over 2^i or passes the data unchanged. Rabaey (Figure 11-38, page 597) shows a multiplexor-based logarithmic shifter. One could also use tri-state buffer multiplexors or logic gate multiplexors instead of transmission gates. A circuit like this with many transmission gates in series would benefit from having buffers inserted along the path - perhaps every three stages or so. This example is hard wired to give arithmetic right and left shifts (by sign extension for right shift, and making the least significant bits zero for left shift.). Minor changes will convert it to a logical shift, or a small amount of logic can replace the hard wiring to allow several kinds of shifts.

Logarithmic shifters have intrinsic decoding, as the shift bits are used directly to control the muxes or transmission gates. Small barrel shifters, however, may be more compact than logarithmic shifters. Logarithmic shifters are always better when the word size is large and there are many shift possibilities.

Procedure

1. Create Library with IBM_PDK

You need to give a specific name for your library (eg, CAD5, Shifter, Shifter16, etc...)

2. Schematic Design

You need to implement a barrel shifter which achieves the required functionality - both left and right logical shifts from 0 to 15 bits. You will also have to design the decoder in this CAD assignment. Keep in mind that the shift amount is a 2's complement amount - not signed magnitude! Some of the shifters presented in class or in the text assume a signed magnitude shift amount. Provide buffering for any array-wide control signals.

3. NCVerilog

Run NCVerilog on the 16-bit shifter and verify that it functions as expected (shift left and right by up to 15 bits.)

4. Pre-layout HSpice

As with earlier designs, build a worst case path schematic for device sizing purposes prior to starting layout. You may need to add some significant capacitance to emulate the wire capacitance. Sizing up device widths will help to some extent.

5. Layout

Make sure that the shifter is bit-slice width-matched with the rest of your datapath structures. You may find the layout of the shifter to be metal intensive, especially if you are implementing the pass gate implementation, so think first of how to run metal lines.

6. Design Verification

Run DRC, LVS on the entire shifter. Extract the parasitics.

7. Analog Simulation

Find the delays through your shifter using HSpice with parasitics.

Requirements

You should have the following in your group's **cad5** directory.

Barrel Shifter

- Schematic of the entire shifter, including the **drivers/buffers/decoder**.
- Layout of the shifter. This should include hierarchical layout of the drivers/buffers/decoder.
- Clean DRC report file.
- Clean LVS report file.
- PEX (Calibre view)
- NCVerilog waveform showing a fixed input (10101010) with all possible shift amounts (single file).
- NCVerilog waveform (the same as above but with delay added)
- HSpice printouts showing critical path delay (after pex, rise/fall).

README

- Full path to your designs.
- Briefly explain how you implement your shifter.
- Worse case delay numbers (rise/fall) and explain the delays, i.e., which path was critical in determining the delays (start/end points).
- Estimate all input pins capacitance, and explain how you estimate these numbers.
- Other comments.

Deadline

You need to turn in CAD5 by **Thursday, Oct. 21st, 2010, 11:00 pm**.