CS251 – Spring 2008

Barrel Shifters

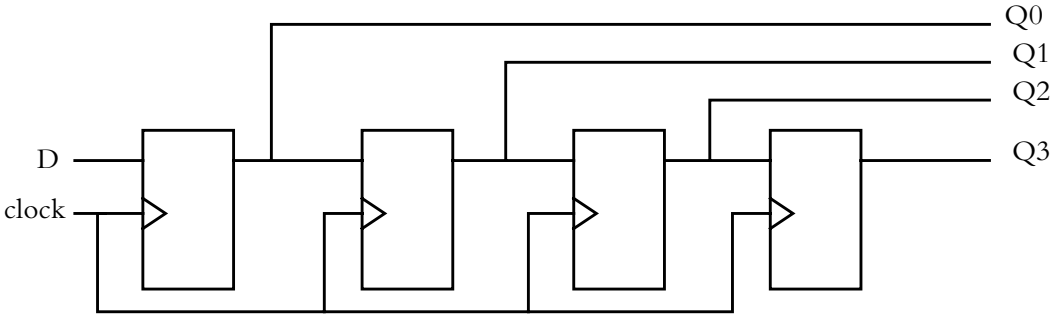
Bill Cowan

1 Introduction

By now you should well know that I am prone to harp on about the importance of the barrel shifter as a CPU component. This document is a short introduction to shifters.

2 History

The ancestor of the barrel shifter was the shift register, a version of which you saw in the assignment question on Gray codes. It consists of a sequence of D flip-flops, wired as in the diagram below. Notice that



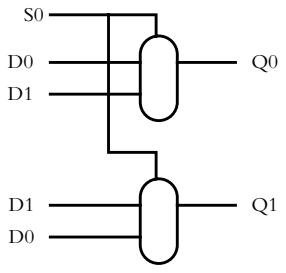
in this circuit every time the clock ticks all data moves one flip-flop to the right, with whatever data is currently on input D shifted into the first flip-flop. The disign you see is called a serial in, parallel out register because the data comes in one bit per clock tick, but can be read all at once on the outputs. Components like these are fundamental to serial buses of all kind, such as USB, because they convert a series of bits into a byte or a word.

Shift registers are available with a wide variety of features improving its base function. Examples include the following.

1. Parallel load. Load all flip-flops at once from a parallel bus.
2. Parallel clear. Clear all flip-flops to zero or one.
3. Bidirectional operation. An input determines whether data will be right or left shifted.
4. Rotation, logical or arithmetic shift. Rotation puts the shifted out bit onto the input. Logical shift fills with zeros, appropriate for unsigned data. Arithmetic shift right fills with zeros, and left fills by replicating the MSB, appropriate for TCIs.

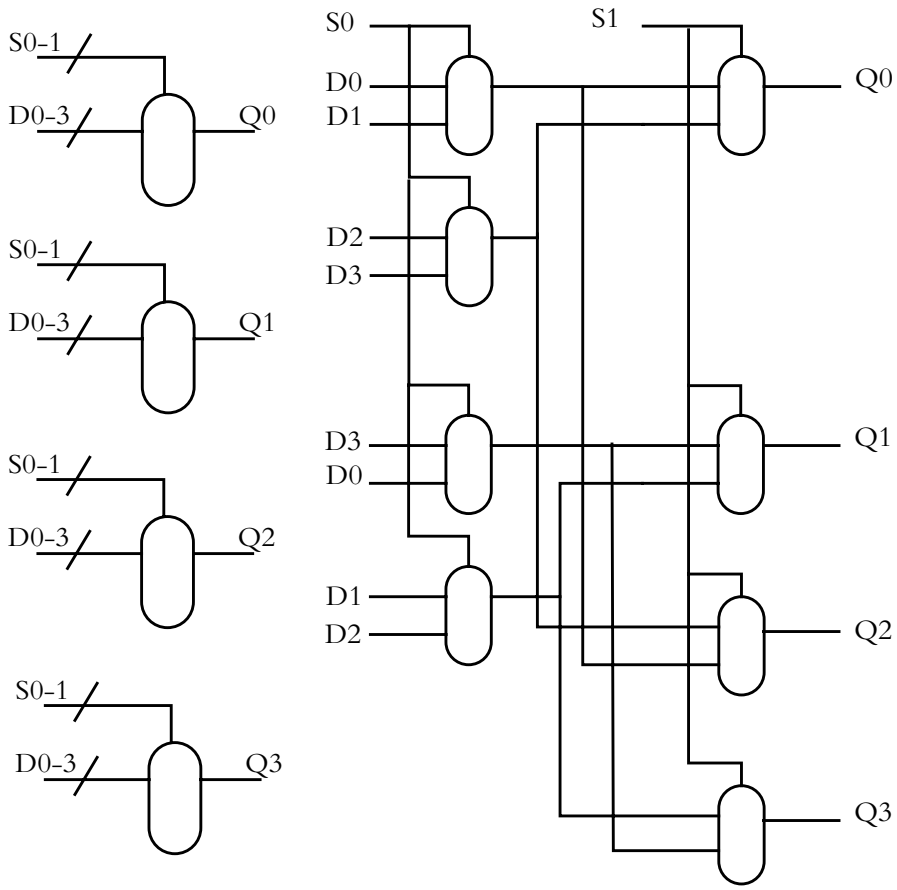
3 The Barrel Shifter

The barrel shifter is based on multiplexors: that is, it is pure combinational logic. An unsigned number input indicates how many bits to shift. A two-bit barrel shifter, is shown in the diagram to the right. The select input, S0, connects the upper data input of the multiplexor to the ouput when reset, and the lower input when set. Thus, interpreting reset as zero and set as one, S0 is the number of bits to shift the input (D0, D1). Shifting by zero bits gives (Q0, Q1) = (D0, D1), and by one bit gives (Q0, Q1) = (D1, D0).



In this simple example, the shifter is rotating its input, and for two bits right and left rotate are the same operation. As an easy exercise, change the multiplexor inputs to that the 2-bit shifter shown does logical or arithmetic shift, either right or left.

Suppose now that we have four bit data, and we want to do three operations on this data, rotate, logical shift and arithmetic shift, each one either right or left. We can create such a shifter using two selector, four- bit multiplexors, as shown immediately to the right. On the far right the top three 1/2 multiplexors implement the topmost 2/4 multiplexor. In practice, barrel shifters are usually constructed from a large number of 1/2 multiplexors. Why? The second group of three multiplexors calculates the second bit of the shifted result, the circuit being more or less independent of the one that calculated the first bit of the shifted result. But now, the remaining two bits requires only one multiplexor each, so the total number of multiplexors needed is eight insteda of twelve. This advantage increases as the number of bits increases.



The illustration is a very specific barrel shifter, which rotates the value left. It is also possible, by introducing different inputs, to do logical or arithmetic shift, and to shift right instead of left.

Nowadays barrel shifters are rarely seen as discrete components, but are combined with other components to make a CPU or more often a DSP. When they appear in block diagrams of circuits they are most often given a symbol like the one illustrated to the right. (The actual part shown is part of a LSH32, 32-bit barrel shifter. SFT0-4 gives the number of bits to be shifted; WRAP/FILL determines whether the operation is a rotation (WRAP) or a shift (FILL), and when the operation is a shift LEFT/RIGHT determines the direction of the shift.)