DAT093
Introduction to Electronic System Design
Using a FSM (Finite State Machine) for the serial adder/subtractor

Introduction

A FSM can in many cases be a good way to implement a sequential design. The FSM clarifies the flow through the structure.
We will look at how this can be applied to the serial adder/subtractor with saturation control.
As stated in the document Some comments on starting a process a good way to start a process-based design is to use a start signal that first goes from zero to one (1) when we initialize the run and then we wait for the signal to go low (0) before we move on in the flow, *Figure 1*.

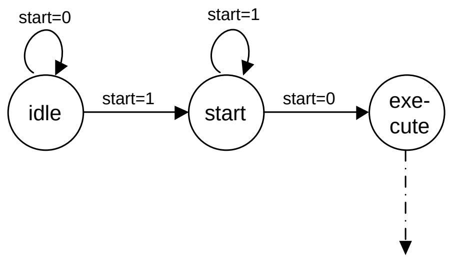

Figure 1 State machine implementtation for starting a process

FSM for the serial adder/subtractor

We can design the FSM for the serial adder/subtractor according to *Figure 2*.
Let´s look at the states.

idle state

We start in the idle state where we only wait for the start signal.

init state

When the start signal goes to one (1) we go to the init state. In this state we set up the initial values for the signals in the calculation. This includes how the b signal should be treated depending on whether we do addition or subtraction and we give the initial signal values for the LSB to the one-bit full adder. We should also set up a counter to keep track on what bit we are calculating at the moment.

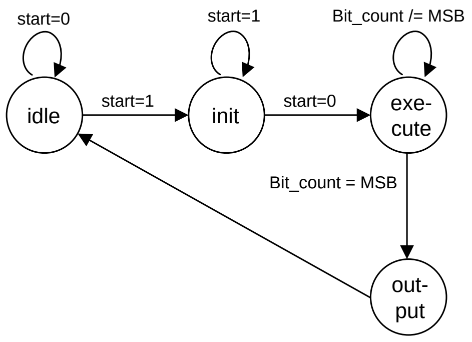

Figure 2 FSM for the serial adder/subtractor

execute state

When the start signal goes low (0) again we move on to the execute state where we do the calculation (addition or subtraction) bit by bit.
We update the bit count and for each clock pulse we do the calculation for a new bit, moving for LSB to MSB.

output state

When we have reached MSB we move to the output state where we check if the output should be saturated or not and write the result of the calculation to the output y.