Comparators part 4

July 17, 2019

1 Interfacing comparators with the logic

Most comparators are used to interface analog signals coming from outside of the chip or from analog functions with the logic. It depends on the function how the comparator is connected to the logic. Usually there are timing constraints to be respected and in many cases the signal swing has to be converted between the supply of the analog function and the logic.

1.1 Level shift

To convert the signals from one supply domain to an other levelshifts are required.

1.1.1 Shift down

Shifting down from a high supply level (for instance 3.3V) to a low supply level (for instance 1.2V) usually is done building a 3.3V inverter but supplying it with the low voltage rail. So the input can handle 3.3V while the output is in the 1.2V domain.

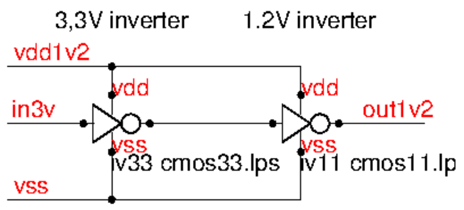

Fig.1 shift down from 3.3V to 1.2V

1.1.2 Shift up

Shifting from a low supply domain to a high supply domain typically requires a PMOS half-latch and NMOS switches. This kind of level shift is requires the least area. The disadvantage is that it becomes undefined if the supply on the input driver side gets lost.

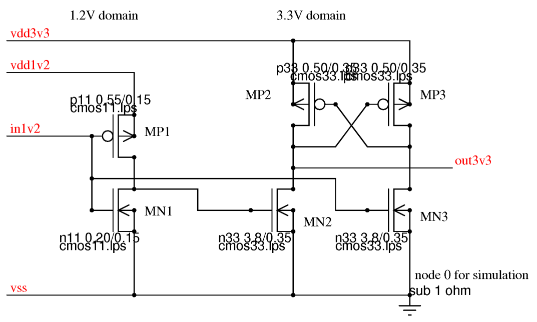

Fig2 shift up level shift without latch function

Note that the NMOS transistors MN2 and MN3 are significantly stronger than the PMOS transistors MP2 and MP3. There are two reasons for it:

- MN2 and MN3 are driven with a lower gate voltage (coming from the 1.2V domain) than the PMOS transistors MP2 and MP3 (driven from the 3.3V domain)
- MN2 and MN3 must be stronger than MP2 and MP3 under all circumstances. Worst case usually are low 1.2V supply, high 3.3V supply, simulation corner sf (slow NMOS, fast PMOS)

If the 1.2V supply vdd1v2 is lost the gates of MN2 and MN3 are not driven. The previously 'on' PMOS will remain conducting as long as the gate doesn't float up. If the gate of the conducting PMOS floats up (for instance due to leakage) the transistor turns off an the outputs of the level shift become undefined.

Shift up level shift with latch function The level shift is based on the shift up level shift without latch but has two additional hold transistors. These two hold transistors are added to maintain the last valid state even if the low voltage driver looses its supply. This way floating gates at the output of the level shift are prevented.

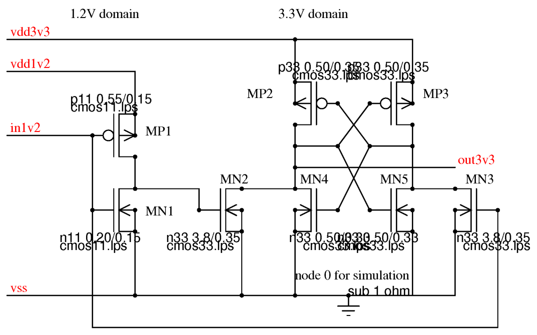

Fig.3: Level shift with latch function.

If the supply of the 1.2V domain gets lost the transistors MP2, MP3, MN4, MN5 store the last state of the level shift while MN2 and MN3 are not driven.

1.2 Unsynchronized interface:

Directly connecting a comparator output to the logic without any synchronization is very unusual. It only is done for applications that must be functional even if the system clock isn't running anymore. Typical application of this kind are under voltage lock out, wake up or system reset. Unsyncronized interfaces must be described carefully in the logic design because most logic synthesis tools automatically insert a synchronization at every interface! It is strongly recommended to check manually each interface that is unsynchronized intentionally. (I have already seen wake up inputs that didn't end STOP mode because some logic synthesis added a synchronization automatically. So it was possible to send the CPU into STOP mode, but waking it up again didn't work because in STOP mode no more clock was running.)

Most asynchronous logic functions (for instance asynchronous reset) require a certain minimum pulse time. This is needed to guarantee that setup and hold times of flip flops aren't violated.

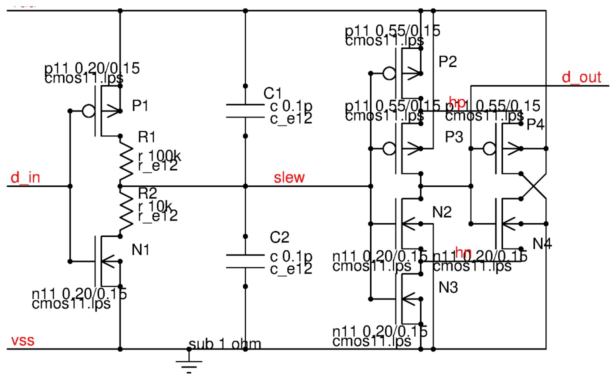

Fig.4: Minimum pulse generator

The minimum pulse generator provides either no pulse at all or a pulse with a certain minimum HIGH or LOW time. The resistors can of course be replaced by current sources as well. (In many cases this is cheaper).

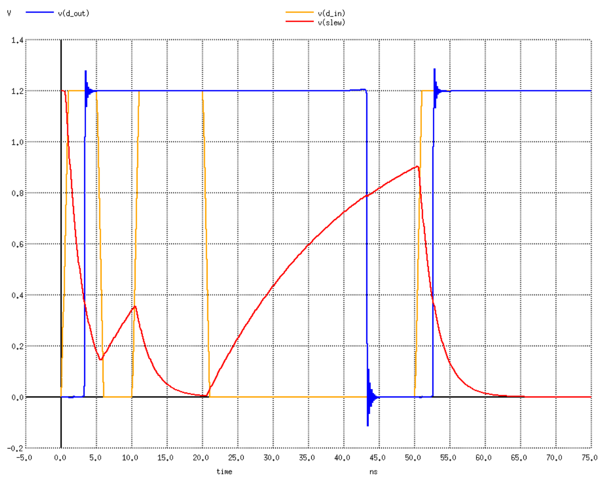

Fig.5: Simulation of the minimum pulse width generator

In the plot above a short pulse from 6ns to 10ns will not trigger a change of signal v(d_out) while a long drop of v(d_in) from 20ns to 50ns switches v(d_out). The hysteresis of the 6 transistor schmitt trigger can be seen comparing v(slew) rising crossing v(d_out) and v(slew) falling crossing v(d_out). The hysteresis is about 400mV. The minimum LOW pulse width is determined by the capacitors and R2. It is about 2.6ns. For a HIGH pulse the minimum pulse width is determined by R1 and the capacitors. It is about 26ns.

Synchronized interfaces: This is the standard. A comparator output can change exactly at the clock edge. Reading such a signal leads to violations of setup and hold times of the flip flops inside the logic. Such violations can have two effects:

- The flip flop doesn't detect the signal as expected. (Best case it will lead to a detection one clock edge later. For many applications this isn't a problem - but for testing it is a severe one because the whole pattern fails depending on spread of the delay!)
- The flip flop starts to oscillate as a ring oscillator when the data input has an edge that is coincident with the clock edge. Usually these oscillations end after a few cycles. But for a counter this can already be a disaster.

The most common approach to prevent such situations is double buffering with two flip flops. The second flip flop can either be driven from the same clock or from a inverted clock.

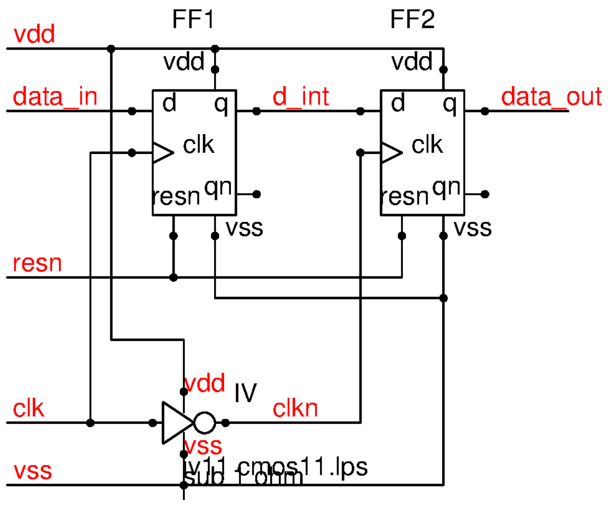

Fig.6: Typical clock synchronization circuit

Even if there is a setup time or hold time violation at FF1 and FF1 may either start to oscillate or have a very wide spread of delay times the second flip flop stabilizes the timing again. For a synchronous logic synchronizing with only one stage isn't good enough. The second flip flop will provide a stable output signal. Hitting such a condition in simulation is extremely difficult (I tried hundreds of runs but didn't get it in simulation). For this reason a little hardware setup using a standard CMOS 4035 shift register is used to demonstrate the effect. (Thousands of random timings in measurements is much faster than simulating it.)

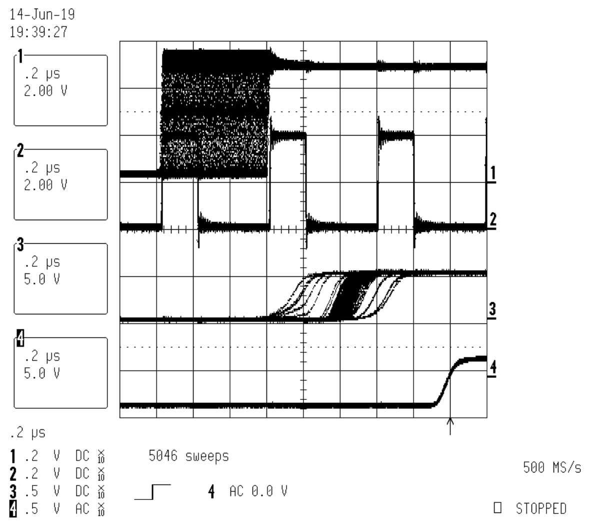

Fig.7: Demonstration of the behavior of a double synchronization

Trace 1 is the asynchronous input signal of the synchronizer. The clock is shown on trace 2. After the first flip flop the sampled signal has a very wide timing spread due to setup and hold time violations (trace 3) . After the second fip flop the signal is stable (trace 4).

One of the drawbacks of the synchronization is the delay added to the signal path. Using clock and inverted clock as shown in the circuit of figure 6 the worst case delay is one clock period + the propagation delay of the flip flops. Sometimes the circuit is implemented using clk (instead of clkn) for the second flip flop as well. (This is the case in the example measurement using the 4035 shift register as a synchronizer.) In this case the delay time can reach 2 clock periods + the propagation delay of the flip flops.

To work properly the propagation delay of the flip flops must be less than half the clock period using the implementation shown in the circuit above. If both flip flops work on the same edge of clk the requirements for the propagation delay relax by factor 2.

The delay isn't critical for most applications. In extreme cases (for instance PWM control of a switched mode power supply or fast ADCs) even this short delay may in fact matter! (the duty cycle of the PWM becomes quantisized. But using a digital regulation loop this is the case anyway.) In a critical application it may make sense to build a self contained asynchronous design and only provide the status information to the logic via a synchronizer.