Given the CMOS circuit with pull-down shown here, we can generate the corresponding function that this CMOS circuit represents which is NOT(F) = (A+B)C + D.

So F = ((A+B)C + D) the whole thing negated. This function can be expressed as a truth table which describes the output that the function produces for any set of inputs.

All of the possible input combinations are enumerated, and then the value of F is specified for each of those combinations. For this function, F, for example, if A=0 B=0 C=0 and D=0, then (A+B)C = 0 that ORed with D = 0, and finally the entire thing is negated, so F = 1. For A=0 B=0 C=0 and D=1, (A+B)C = 0 + D=1 gives us 1 and the whole thing negated is F=0.

In the same way, we can complete the rest of the truth table, and we get.

1 0 1 0 0 0 1 0 0 0 1 0 0 0 Given a truth table, one can be asked if the function F that this truth table defines, can be implemented using a single CMOS gate.

To figure that out, we first convert the truth table to a logical equation for F.

Given this truth table, the function F can be expressed as a sum of products by listing all terms that make F=1 and ORing them together. For this truth table F = NOT(A) NOT(B) NOT(C) + NOT(A) B NOT(C) + A NOT(B) NOT(C). In order to determine if this function can be implemented as a single CMOS circuit, the function F needs to be manipulated through logic simplifications in order to see if NOT(F) can be written as a function of just the inputs A, B, and C (without any negations of those inputs) together with ANDs and ORs.

We start with our function for F. Factoring out NOT(C), that equals NOT(C) (NOT(A) NOT(B) + NOT(A) B + A NOT(B)). Rearranging the terms and repeating the first one continues to produce the same function. So F = NOT(C) (NOT(A) NOT(B) + A NOT(B) + NOT(A) B + NOT(A) NOT(B)). Noting that A X + NOT(A) X = X, this reduces to: F = NOT(C) (NOT(B) + NOT(A)). We are almost there, we now want to convert F to NOT(F) by negating the entire thing, so NOT(F) = NOT (NOT(C) (NOT(B) + NOT(A)).

To simplify this we make use of De Morgan's Law which states that NOT(AB) = NOT(A) + NOT(B).

And that NOT(A+B) = NOT(A) AND NOT(B). So NOT(F) = C + NOT(NOT(B) + NOT(A)), and finally NOT(F) = C + (B A). Now that we have found that NOT(F) = C + BA, we know that F can be implemented as a single CMOS gate because we were able to express NOT(F) as a function that consists purely of our non-negated inputs, A, B, and C, together with ANDs and ORs. The pull-down for this circuit would basically implement NOT(F) so it would be series(A,B) in parallel with C.

The pull-up would be the inverse of that which is parallel(A,B) in series with C.