Dual-Rail Combinational Circuit Design

The design process for NCL combinational circuits is similar to Boolean circuits, where a Karnaugh map, or other simplification technique, can be utilized to determine the simplified sum-of-product (SOP) expressions for each output. However, SOP expressions for both the function's 1 and 0 outputs are needed. **The *0s* refer to a signal's *rail*^{0} and the *1s* refer to a signal's *rail*^{1}**. After expressions for the outputs have been obtained, an assessment must be made to ensure that the circuit is input-complete. If not, the missing input(s) must be added to the appropriate product term(s). The output equations must then be partitioned into sets of four or fewer variables to be mapped to the 27 NCL gates, while ensuring that the resulting circuit is observable. To minimize area and delay, partitioning should be performed such that the minimal number of sets is obtained, which will occur when the maximum number of product terms are grouped into each set.

Take for example the design of a partial product (PP) generating component for the most significant bit of an unsigned Booth2 multiplier, which is only required to be input-complete with respect to input, MR_1. Figure 1 shows the Karnaugh map for this component, along with the optimal coverings. Since this design must be input-complete with respect to MR_1, the coverings should not eliminate MR_1 from the corresponding product term; hence some of the coverings are 2-coverings instead of 4-coverings. The SOP equations are derived directly from the K-map coverings as follows:

$PP^1 = MR_2^{\,1}MR_1^{\,1} + MR_1^{\,1}MR_0^{\,1}MD_{i-1}^{\,1} + MR_2^{\,1}MR_1^{\,0}MR_0^{\,1} + MR_2^{\,1}MR_1^{\,0}MD_{i-1}^{\,0}$;

$PP^0 = MR_2^{\,0}MR_1^{\,0} + MR_1^{\,0}MR_0^{\,0}MD_{i-1}^{\,1} + MR_2^{\,0}MR_1^{\,1}MR_0^{\,0} + MR_2^{\,0}MR_1^{\,1}MD_{i-1}^{\,0}$.

Since each product term contains MR_1, the circuit is input-complete with respect to MR_1. The equations can be partitioned into four sets of 4 variables as highlighted above, resulting in the optimized circuit shown in Figure 2. The yellow and blue terms each map to a TH54w32 gate and the green and pink terms each map to a TH54w22 gate. Since no product terms were divided, the circuit is observable.

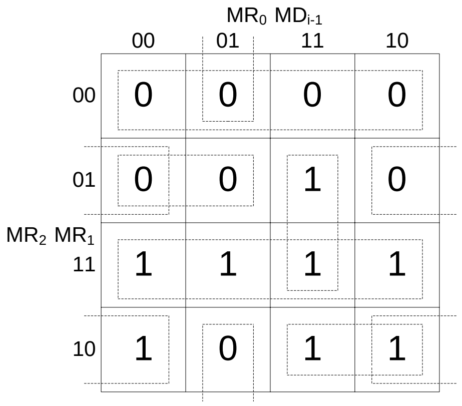

Figure 1. K-map for Booth2 PP generation component.

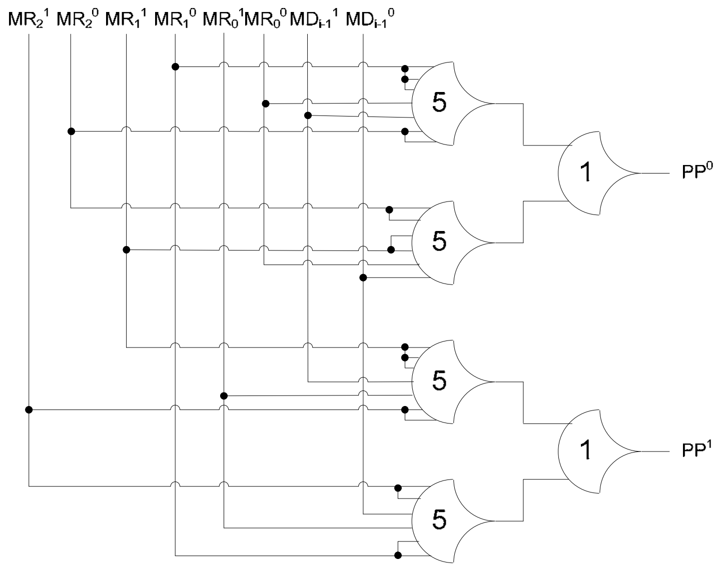

Figure 2. NCL Booth2 PP generation circuit.

Now let's consider the design of input-complete optimized 2-input fundamental Boolean logic functions. Since each of these functions requires 2 dual-rail inputs, each output rail equation will consist of at most 4 input variables, such that the combinational logic for each rail will require only 1 NCL gate. The canonical expressions for an OR function, $Z = X + Y$, are $Z^0 = X^0Y^0$ and $Z^1 = X^1Y^1 + X^0Y^1 + X^1Y^0$. Z^0 directly maps to a TH22 gate and Z^1 directly maps to a THand0 gate. Similarly, canonical expressions for an AND function, $Z = X \bullet Y$, are $Z^0 = X^0Y^0 + X^0Y^1 + X^1Y^0$ and $Z^1 = X^1Y^1$. Z^0 directly maps to a THand0 gate and Z^1 directly maps to a TH22 gate. The optimized XOR function, $Z = X \oplus Y$, is a bit more complex. The canonical expressions are $Z^0 = X^0Y^0 + X^1Y^1$ and $Z^1 = X^1Y^0 + X^0Y^1$, which both directly map to a THxor0 gate. However, two transistors can be eliminated for each rail of Z (when using static gates) by adding the two *don't care* terms, representing the cases when both rails of either X or Y are simultaneously asserted. The new equations follow: $Z^0 = X^0Y^0 + X^1Y^1 + X^0X^1 + Y^0Y^1$ and

$Z^1 = X^1Y^0 + X^0Y^1 + X^0X^1 + Y^0Y^1$, both of which now map to a TH24comp gate. The inverse logic functions, NAND, NOR, and NXOR, can easily be attained by exchanging the output rails of the AND, OR, and XOR functions, respectively.

Finally, let's design an optimized NCL full adder, whose truth table is shown in Figure 3, where *X* and *Y* denote the input addends and C_i denotes the carry input. *S* and C_o denote the *sum* and *carry* outputs, respectively. The K-map for the C_o output is shown in Figure 4, yielding: $C_o^0 = X^0Y^0 + C_i^0X^0 + C_i^0Y^0$ and $C_o^1 = X^1Y^1 + C_i^1X^1 + C_i^1Y^1$, both of which directly map to a TH23 gate. Since C_o is not input-complete with respect to any inputs, *S* must be input-complete with respect to all inputs.

The K-map for *S*, based on *X*, *Y*, C_i, and the intermediate output C_o, is shown in Figure 5, with essential prime implicants covered. This covering yields: $S^0 = C_o^1X^0 + C_o^1Y^0 + C_o^1C_i^0 + X^0Y^0C_i^0$ and $S^1 = C_o^0X^1 + C_o^0Y^1 + C_o^0C_i^1 + X^1Y^1C_i^1$, both of which directly map to a TH34W2 gate. Checking input-completeness, the *carry* output requires at least two inputs to be generated and the *sum* output requires either the carry output and one more input, or all three inputs to be generated; so all three inputs are needed to generate the *sum* output. Therefore, the circuit as a whole is input-complete even though C_o is not. Furthermore, the *sum* output, and therefore this circuit, is inherently input-complete since it is impossible to determine the value of *S* without knowing the value of all 3 inputs, *X*, *Y*, and C_i. The resulting optimized NCL full adder circuit is shown in Figure 6.

X	Y	Ci	Co	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

Figure 3. Truth table for full adder.

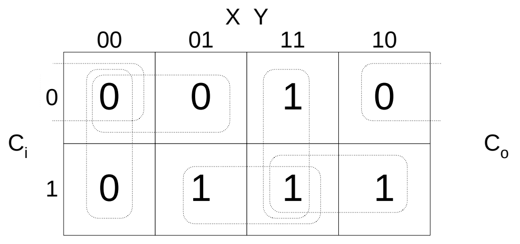

Figure 4. K-map for C_o output of full adder.

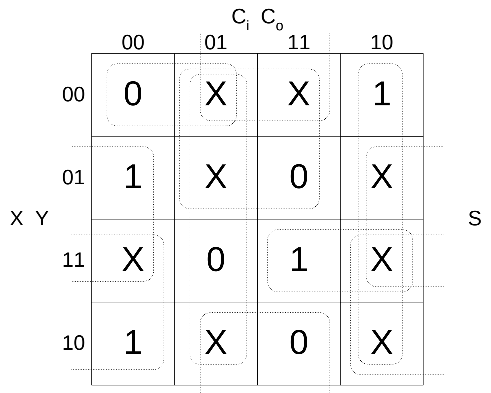

Figure 5. K-map for *S* output of full adder.

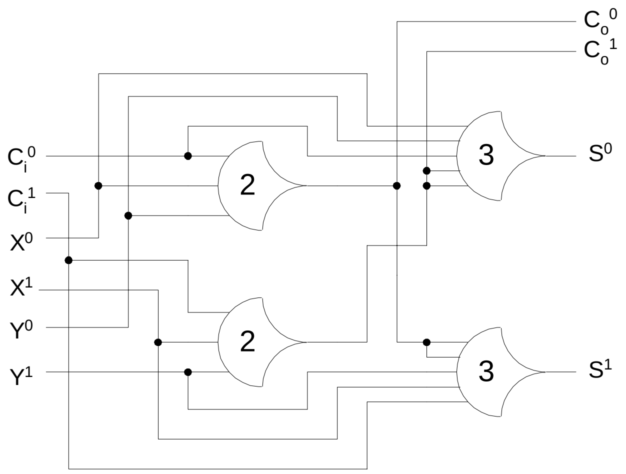

Figure 6. Optimized NCL full adder.