

Оптимизация сумматора для получения дополнительного кода

Логические формулы ячейки суммирования двух битов X_i и Y_i с переносом C_i имеют вид:

$$R_i = (X_i \text{ xor } Y_i) \text{ xor } C_i$$

$$C_{(i+1)} = (X_i \text{ and } Y_i) \text{ or } ((X_i \text{ xor } Y_i) \text{ and } C_i)$$

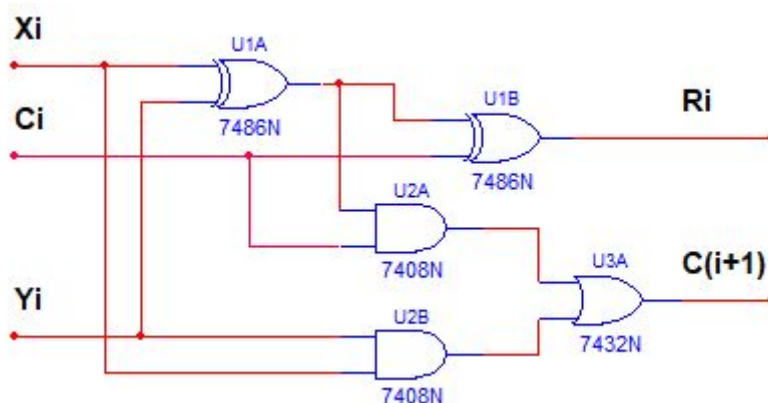
где:

i – индекс разряда

R_i – результат суммирования битов

$C_{(i+1)}$ – перенос в следующий разряд

Реализация такой ячейки имеет вид:



Для получения дополнительного кода, один из операндов сумматора, например Y_i должен равняться нулю во всех ячейках. Если $Y_i = 0$, то:

$$R_i = (X_i \text{ xor } 0) \text{ xor } C_i$$

$$C_{(i+1)} = (X_i \text{ and } 0) \text{ or } ((X_i \text{ xor } 0) \text{ and } C_i)$$

или

$$R_i = (X_i \text{ xor } 0) \text{ xor } C_i$$

$$C_{(i+1)} = 0 \text{ or } ((X_i \text{ xor } 0) \text{ and } C_i)$$

Следовательно, таблица истинности для операции **xor**, должна быть модифицирована, то есть, в ней отсекается столбец для которого $Y_i = 0$

X/Y	0	1
0	0	1
1	1	0

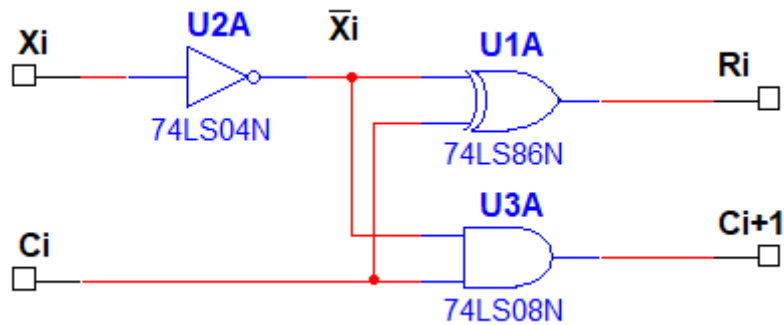
В этом случае логические формулы упрощаются и имеют вид:

$$R_i = X \text{ xor } C_i$$

$$C_{(i+1)} = X_i \text{ and } C_i$$

Синтез ячейки для получения дополнительного кода

Кроме оптимизированной ячейки сумматора, ячейка преобразователя в дополнительный код должна вначале инвертировать входной бит, а только потом выполнить его сложение с нулем. Реализация такой ячейки будет иметь вид:



Для создания преобразователя в дополнительный код осталось только вспомнить, что перенос в самую младшую ячейку должен быть равен единице, (поскольку дополнительный код это инверсия его битов плюс единица), а перенос из самого старшего бита просто игнорируется.