第十一章作业参考答案

11.2 11.6 11.10 11.12 11.16 11.18 11.22

11.2

11.2 在相加两个无符号数时，最终一级的进位输出表示溢出。但在相加两个 2 的补码形式的有符号数时，溢出的检测稍微复杂一些。推导出溢出与两个输入及输出的最高有效位之间关系的布尔表达式。

设$A = A_nA_{n-1}...A_1$、$B = B_nB_{n-1}...B_1$是两个补码形式有符号数，n位，A_n和B_n是最高位（或者说符号位），$A_{n-1}...A_1$和$B_{n-1}...B_1$是数据位。令S_n为输出的最高有效位，其实也就是输出的符号位。当符号位A_n、B_n相同但与输出最高有效位不同时产生最终的溢出。

$\therefore$ 最终的溢出$C = A_nB_nS_n' + A_n'B_n'S_n$.

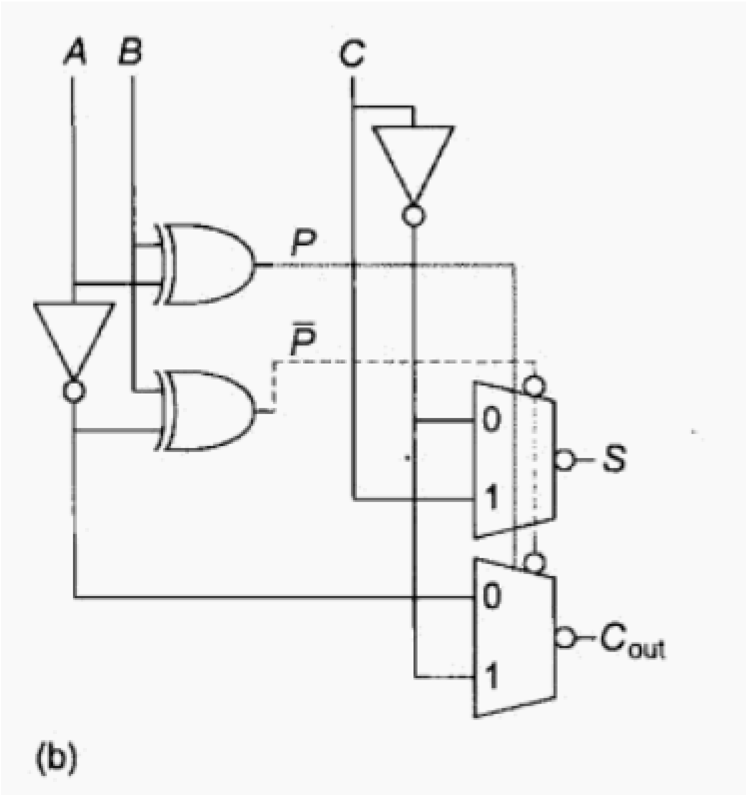
11.6

11.6 图 11.26(b) 中具有可变长度模块的增量进位加法器，对于 16 位的加法它需要 5 级 2 阶的组 PG 单元。对于 32 位的加法器它需要多少级？对于 64 位加法呢？

根据其递增的规律，32位加法器需要8级，64位加法器需要11级。

11.10

11.10 写出图 11.6(b) 所示电路中 C_{out} 的布尔表达式，简化该式以证明该传输管电路计算的确实是多数函数。



$P = A \oplus B, C_{out} = (PC^{'} + P^{'}A^{'})^{'} = (AB^{'}C^{'} + A^{'}BC^{'} + A^{'}B^{'})^{'} = MAJ(A, B, C)$

11.12

11.12　设计一个计算 $A-B=k$ 的比较器，画出它的原理图。

$A - B = A + B^{'} + 1$,所以参照图11.47，B_i后面加上反相器，$C_0 = 1$，其余电路不变即可。当然，按照$A = K + B$设计也行，此时$C_0 = 0$。

11.16

11.16　找出数 0 ~ 15 的 4 位二进制反射格雷码值。

0:0000;1:0001;2:0011;3:0010;4:0110;5:0111;6:0101;7:0100;8:1100;9:1101;10:1111;11:1110;12:1010;13:1011;14:1001;15:1000。

11.18

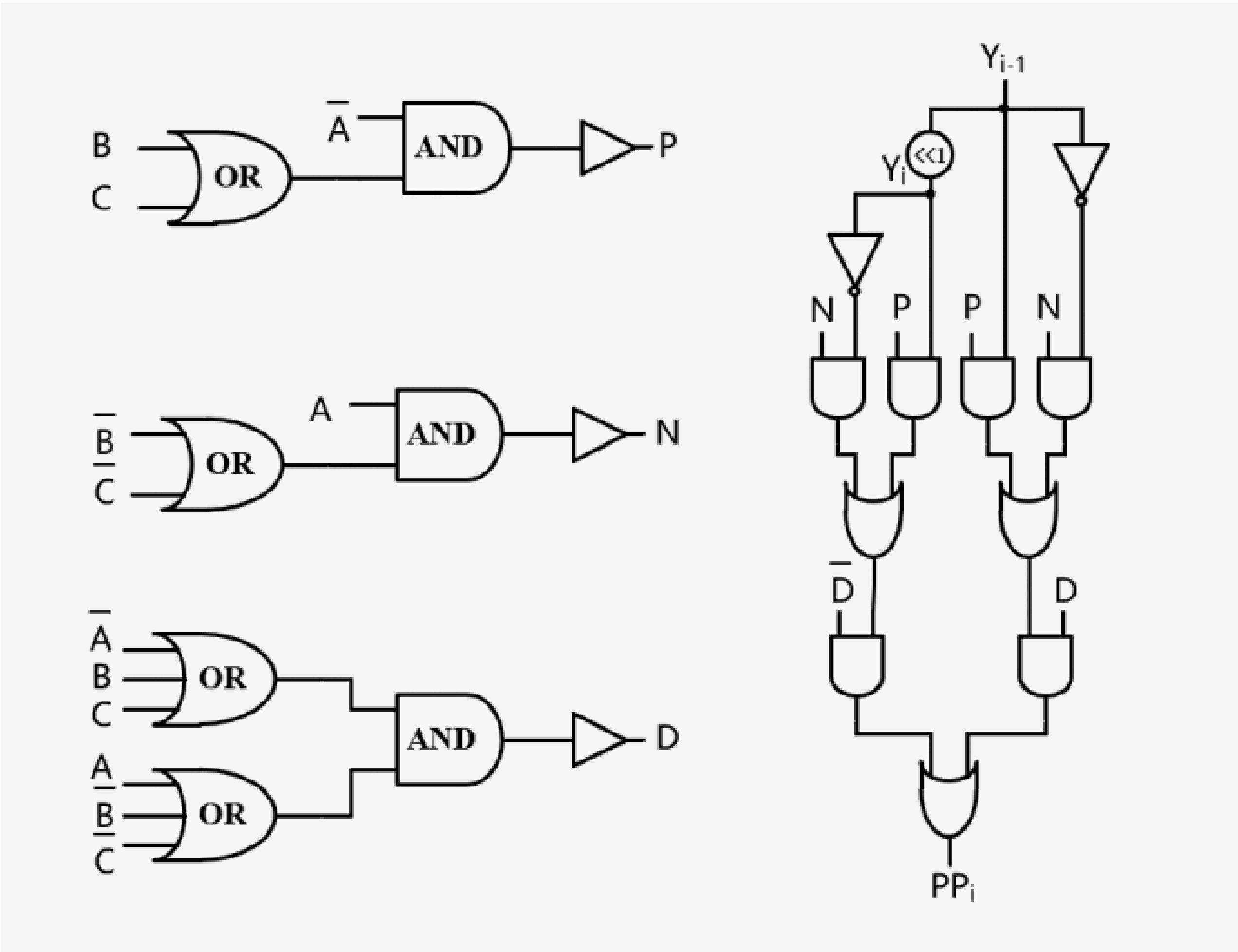
11.18　表 11.12 和图 11.80 显示了一种采用 SINGLE(单倍)、DOUBLE(双倍)和 NEG(负)选择信号的基 4 波兹编码方式，另一种编码方式采用 POS(正)、NEG(负)和 DOUBLE(双倍)信号。对于倍数 Y 和 $2Y$，POS 为 1，对于倍数 $-Y$ 和 $-2Y$，NEG 为 1；对于倍数 $2Y$ 和 $-2Y$，DOUBLE 为 1。设计一个采用这一编码方式的波兹编码器和选择器。

$A=x_{2i+1}$	$B=x_{2i}$	$C=x_{2i-1}$	**部分积PP**	POS	NEG	DOUBLE
0	0	0	0	0	0	0
0	0	1	Y	1	0	0
0	1	0	Y	1	0	0
0	1	1	2Y	1	0	1
1	0	0	-2Y	0	1	1
1	0	1	-Y	0	1	0
1	1	0	-Y	0	1	0
1	1	1	0	0	0	0

根据真值表：

$$POS = A^{'}(B+C)\ NEG = A(B^{'}+C^{'})\ POS = A^{'}BC + AB^{'}C^{'}\ PP_i = (Y_iPOS + Y_i^{'}NEG)DOUBLE^{'} + (Y_{i-1}POS + Y_{i-1}^{'}NEG)DOUBLE$$

编码器与选择器见：



ps：有同学好奇，取-Y/-2Y是取补码，也就是反码加一，为什么电路没体现+1？这是因为这一步的+1放到了下一行的最低有效位，避免当前行发生进位传播（其实也就是避免反码加一操作可能带来的进位传播），这个课本有说。

11.22

11.22 写出一个前置计算的表达式，它决定在一个 N 位的输入位串中样式 10 第二次出现的位置。例如，输入为 010010 时，返回值应为 010000。

参考11.37. 我们令$X_{i:j}$为A的第i至j-1位没有出现一组10，$W_{i:j}$为A的第i至j-1位只出现一组10，那么按位预计算：

$$X_{i:i} = (A_i A'_{i-1})', where X_{1:1} = 1.$$

$$W_{i:i} = A_i A'_{i-1}, where W_{1:1} = 0.$$

组逻辑： $X_{i:j} = X_{i:k} X_{k-1:j}$ $W_{i:j} = W_{i:k} X_{k-1:j} + X_{i:k} W_{k-1:j}$

输出逻辑： $Y_i = W_{i:i} W_{i-1:1}$