示的优先级编码器功能表，以验证实验结果。保存电路设计文件。

表 11.5　8-3 优先级编码器功能表

输　入								输　出	
I0	I1	I2	I3	I4	I5	I6	I7	Q2 Q1 Q0	Hex 显示
1	X	X	X	X	X	X	X		
0	1	X	X	X	X	X	X		
0	0	1	X	X	X	X	X		
0	0	0	1	X	X	X	X		
0	0	0	0	1	X	X	X		
0	0	0	0	0	1	X	X		
0	0	0	0	0	0	1	X		
0	0	0	0	0	0	0	1		

3. 加法器实验

设计一个全加器（FA），在此基础上将 4 个全加器串联成一个 4 位串行进位加法器。将输入、输出分别连接到 16 进制数码显示管（Hex Digital Display）进行验证。实验步骤如下。

1）设计全加器。根据全加器输出逻辑表达式设计电路图，在 Logisim 工作区中添加逻辑门、连线和标识符。为便于串联，按照图 11.38 所示方式布局电路图，将全加器输出端 F 置于右上方，进位输入 CIN 和进位输出 COUT 置于两侧。然后将或门输入端口数改为 3，将输入引脚、逻辑门的输入端和输出端、输出引脚等通过连接线相连。选中输入、输出引脚，在属性表中添加引脚标识符。选中逻辑门，在属性表中添加门标识符。点击快捷工具栏中的文本工具，在电路空白处添加描述文字。最后对电路进行仿真测试，填写表 11.6 所示的全加器功能表，以验证电路功能。保存电路设计文件，文件名为 FA。

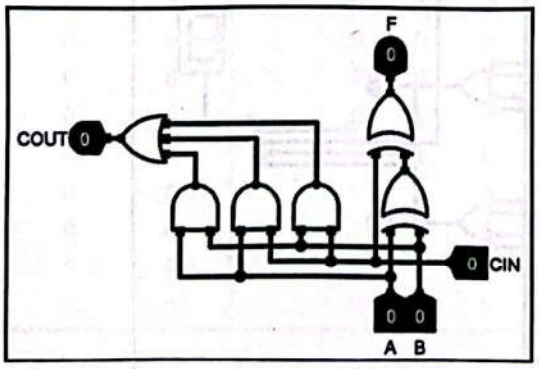

图 11.38　全加器电路图

表 11.6　全加器功能表

输　入			输　出	
A	B	Cin	F	Cout
0	0	0		
0	0	1		
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

2）设计 4 位串行进位加法器。首先新建一个 4 位串行进位加法器电路，并按图 11.39 所示进行组件布局，其中包含 4 个全加器子电路 FA、3 个分线器、输入 / 输出引脚、接地、0 常量、16 进制数字显示组件等。

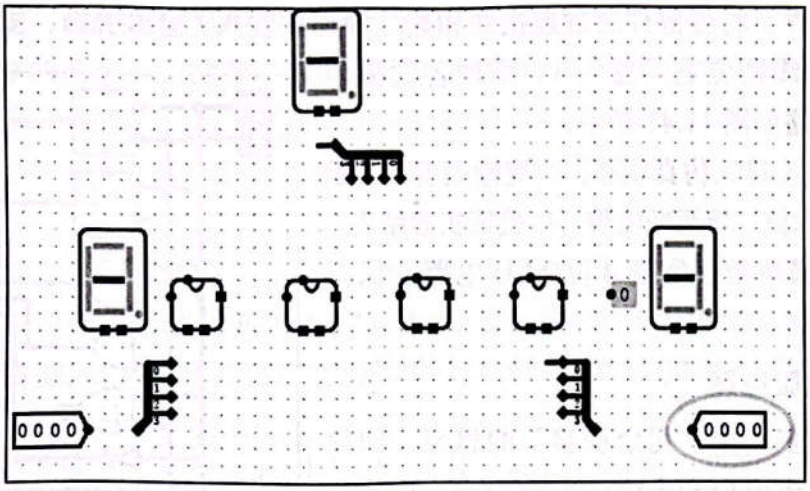

图 11.39　4 位串行加法器组件的布局图

然后将 2 个输入引脚和 1 个输出引脚的数据位宽都设置为 4，并设置分线器属性、朝向等，连接相应端口，得到 4 位串行进位加法器电路图，最终通过仿真进行功能验证。图 11.40 为电路验证示例图。保存电路设计文件。

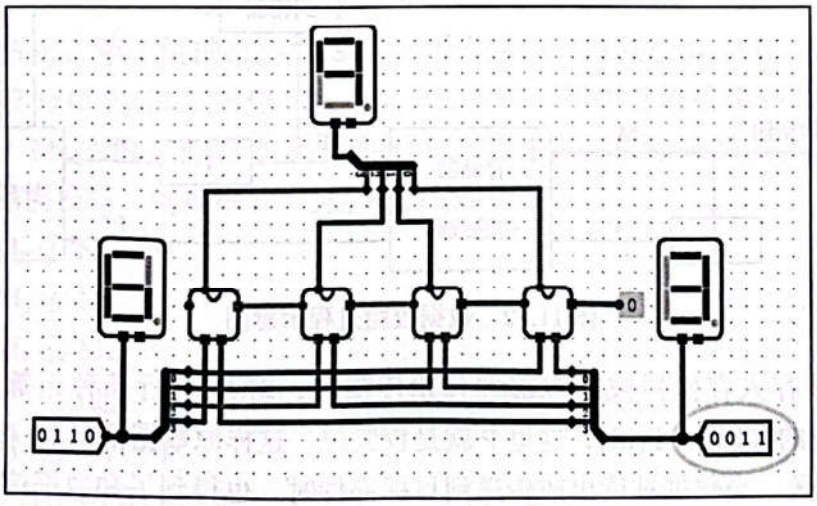

图 11.40　4 位串行加法器的电路验证图

4. 多路选择器的应用

利用多路选择器实现一个一位 ALU 电路，其功能如表 11.7 所示，其中加、减运算仅考虑 A、B 两个本位数的运算，不考虑来自低位的进位或借位，运算结果也不考虑向高位的进位或借位。要求对电路进行仿真测试，以验证电路功能。

表 11.7　一位 ALU 功能表

S3 S2 S1	功能
0 0 0	A 加 B
0 0 1	A 减 B
0 1 0	A·B
0 1 1	A+B
1 0 0	A 异或非 B
1 0 1	A 非
1 1 0	A
1 1 1	B 非

1）根据功能表设计电路图。在工作区中添加构建一位 ALU 电路的逻辑门电路、1 位加法器、1 位减法器和 8 选 1

多路选择器等组件。将各部件通过连接线相连，并连接输入 / 输出引脚、多路选择器的选择端分线器，将分线器位宽设置为 3。添加各类标识符。完成后的电路如图 11.41 所示。

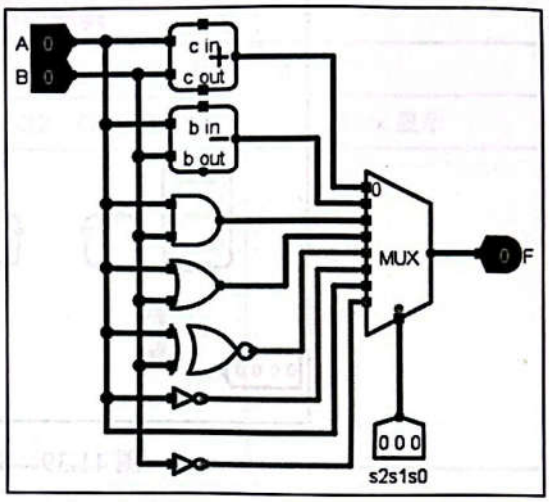

图 11.41　一位 ALU 的电路图

2）仿真测试。进入仿真状态，改变输入引脚的赋值，记录输出引脚值，填写功能表，以验证实验结果。

3）扩展数据位数，实现 4 位 ALU 电路。保存电路设计文件。

5. 汉明码校验电路

数据校验大多采用“冗余校验”的思想，即除原数据信息外，还增加若干位附加的编码，这些新增编码称为校验位。图 11.42 给出了一般情况下的处理过程。

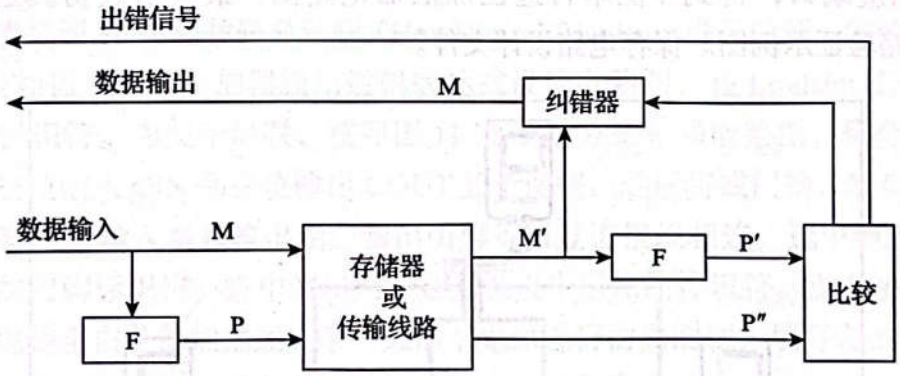

图 11.42　数据校验过程示意图

当数据 M 被存入存储器或从源部件开始传输时，对 M 进行某种运算（用函数 F 来表示），以产生相应的代码 P=F(M)，这里 P 就是校验位。这样原数据信息 M 和相应的校验位 P 一起被存储或传送。当数据被读出或传送到目标部件时，也得到了和数据信息一起被存储或传送的校验位，可用于检错和纠错。假定读出后的数据为 M′，通过同样的运算 F 对 M′ 也得到一个新的校验位 P′=F(M′)，假定原来被存储的校验位 P 取出后其值为 P″，将校验位 P″ 与新生成的校验位 P′ 进行比较运算，生成一个故障字，根据故障字可以确定是否发生了差错。

最简单的数据检错方法是奇偶校验，通过判断数据 M 中 1 的个数是否发生了奇偶性变化来进行检错。若发生奇偶变化，则故障位 $S=P'\oplus P''=1$。

汉明码（Hamming Code，也译为海明码）的主要思想是，将数据按某种规律分成若干组，对每组进行相应的奇偶检测，以提供多位校验信息，得到相应的故障字，并根据故障字对发生的错误进行定位和纠正。汉明校验码实质上就是一种多重奇偶校验码。

对于只能对单个位出错的情况进行定位和纠错的单纠错码（SEC），进行汉明校验的主要思想如下：将需要进行检 / 纠错的数据分成 i 组，每组对应 1 位校验位，共有 i 位校验位，因

此，故障字为 i 位。若故障字为 0，则表示无错，否则故障字的数值就是出错位在码字中的位置编号。除去 0 的情况，i 位故障字的编码个数为 2^i-1，因此构造的码字最多有 2^i-1 位，例如，当 i=3 时，码字可以有 7 位，其中 3 位为校验位，4 位为数据位。为了方便判断码字中出错的是校验位还是数据位，可将校验位的位置编号设为 2 的幂次，即校验位排在第 1（001），2（010），4（100），…的位置上，其余位置上为数据位。这样，当故障字中只有一位为 1 时，说明是校验位出错，否则就是数据位出错。例如，当 i=3 时，假设校验码为 $P_3P_2P_1$，数据信息为 $M_4M_3M_2M_1$，则码字排列为 $P_1P_2M_1P_3M_2M_3M_4$。通常把上述由数据位和校验位构成的码字称为汉明码。图 11.43 给出了 7 位汉明码的故障字和出错情况的对应关系。

序号 / 分组 / 含义	1	2	3	4	5	6	7	故障字	正确	出错位 1 2 3 4 5 6 7
	P_1	P_2	M_1	P_3	M_2	M_3	M_4			
第3组				√	√	√	√	S_3	0	0 0 0 1 1 1 1
第2组		√	√			√	√	S_2	0	0 1 1 0 0 1 1
第1组	√		√		√		√	S_1	0	1 0 1 0 1 0 1

图 11.43　7 位汉明码的故障字和出错情况的对应关系

如图 11.43 所示，第 1 组的故障位 S_1 由校验位 P_1 和数据位 M_1、M_2、M_4 生成，第 2 组的故障位 S_2 由校验位 P_2 和数据位 M_1、M_3、M_4 生成、第 3 组的故障位 S_3 由校验位 P_3 和数据位 M_2、M_3、M_4 生成。假设在终部件得到的数据位 M′ 为 $M_4M_3M_2M_1$，校验位 P″ 为 $P_3P_2P_1$，每组采用偶校验，则根据 M′ 得到 P′ 的每一位如下：

$$P_1' = M_1 \oplus M_2 \oplus M_4$$

$$P_2' = M_1 \oplus M_3 \oplus M_4$$

$$P_3' = M_2 \oplus M_3 \oplus M_4$$

故障字 S=P′ ⊕ P″，因此，根据 P′ 和 P″ 得到故障字的每一位如下：

$$S_1 = M_1 \oplus M_2 \oplus M_4 \oplus P_1$$

$$S_2 = M_1 \oplus M_3 \oplus M_4 \oplus P_2$$

$$S_3 = M_2 \oplus M_3 \oplus M_4 \oplus P_3$$

因此，在终部件的汉明码检 / 纠错电路只要根据所得到的数据位 $M_4M_3M_2M_1$ 和校验位 $P_3P_2P_1$ 形成的码字，按图 11.43 所示的方式划分成 3 组，每组按照上述偶校验方式，得到每一组的故障位 S_i，由故障位构成的故障字 $S_3S_2S_1$ 的值就能确定码字中哪一位发生了错误。图 11.44 给出了 7 位汉明码检 / 纠错电路的原理图，由 3 个偶校验器、一个 3-8 译码器和 7 个异或门构成。

在图 11.44 中，DU[1:7] 是 4 位数据位 $M_4M_3M_2M_1$ 和 3 位校验位 $P_3P_2P_1$ 构成的 7 位汉明码，码字 DU[1:7]=$P_1P_2M_1P_3M_2M_3M_4$。例如，当输入 DU[1:7] 为 1000001 时，说明数据位 $M_4M_3M_2M_1$ 为 1000，检验位 $P_3P_2P_1$ 为 001，根据上述公式得到故障字的各位如下：

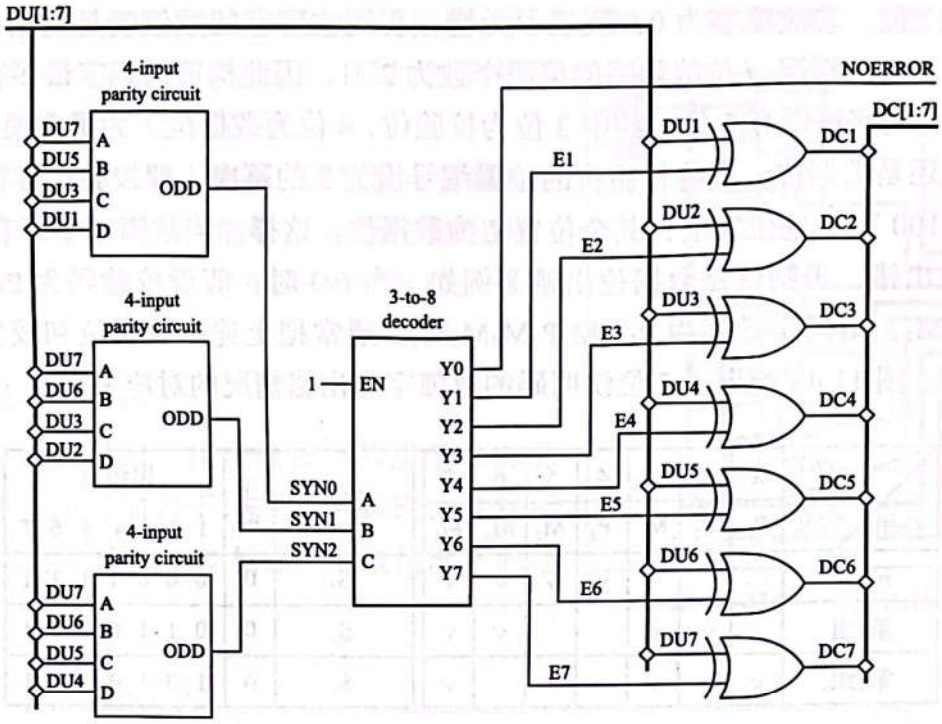

图 11.44　7 位汉明码检 / 纠错电路的原理图

$S_1 = M_1 \oplus M_2 \oplus M_4 \oplus P_1 = 0 \oplus 0 \oplus 1 \oplus 1 = 0$

$S_2 = M_1 \oplus M_3 \oplus M_4 \oplus P_2 = 0 \oplus 0 \oplus 1 \oplus 0 = 1$

$S_3 = M_2 \oplus M_3 \oplus M_4 \oplus P_3 = 0 \oplus 0 \oplus 1 \oplus 0 = 1$

因此，故障字 $S_3S_2S_1$ 为 110，说明码字中第 6 位（对应 DU[6]，即 M_3）发生错误。在图 11.45 中，A 组偶校验结果为 $S_1=0$，B 组偶校验结果为 $S_2=1$，C 组偶校验结果为 $S_3=1$，对应位置编号为 $S_3S_2S_1=110$，3-8 译码器输入端 C、B、A 分别为 1、1、0，输出 Y6 为 1，其余输出为 0，Y6 与 DU[6] 异或生成 DU[6] 的相反值，使得出错位得到纠正。

汉明码校验电路的实验步骤如下。

1）添加 3-8 译码器子电路。在 Project 菜单下选择 Add Circuit，设置子电路名称为 3-8 译码器，根据图 11.35 中的电路图实现一个 3-8 译码器，子电路外观如图 11.45 所示。

2）设计 4 位偶校验器，并生成子电路，如图 11.46 和图 11.47 所示。

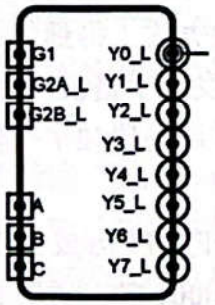

图 11.45　3-8 译码器的外观图

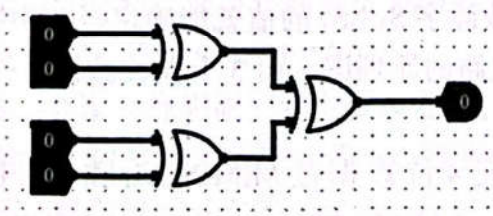

图 11.46　4 位偶校验器电路的原理图

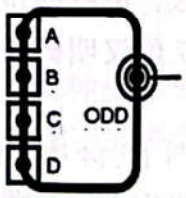

图 11.47　4 位偶校验器的外观图

3）根据图 11.44 所示原理图实现 7 位汉明码检 / 纠错电路，实现后的电路如图 11.48 所示。

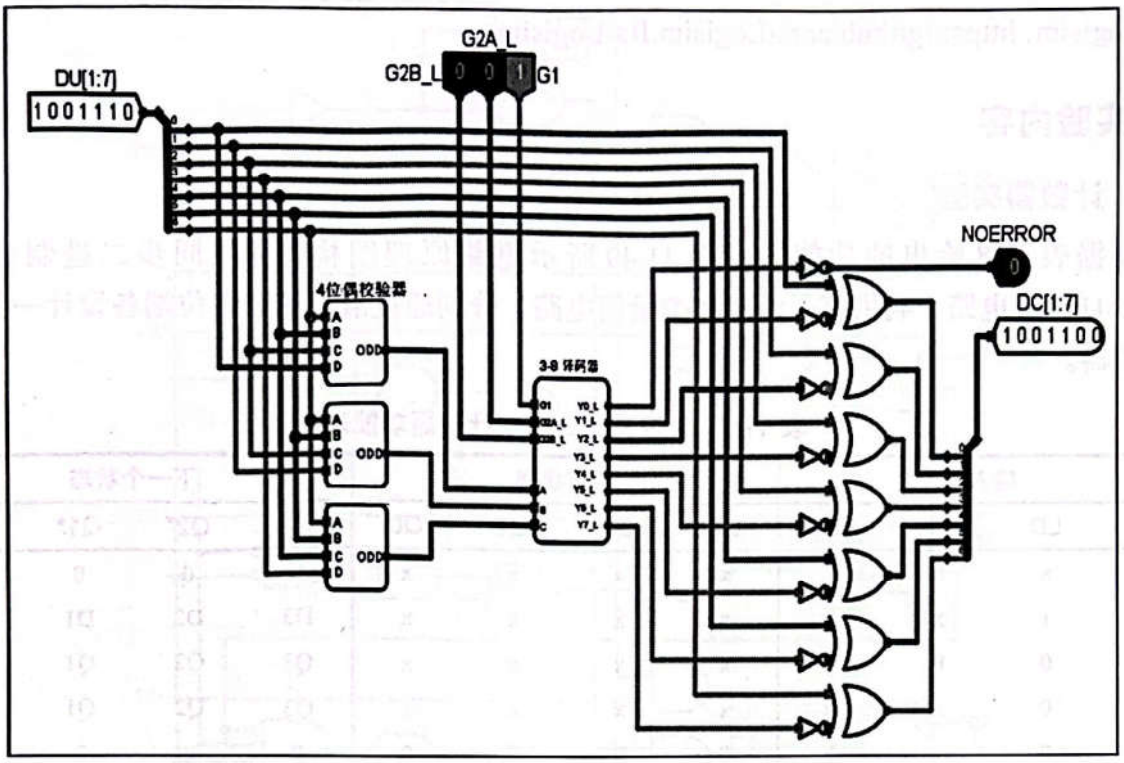

图 11.48　7 位汉明码纠错电路图

4）在 DU[1:7] 处分别输入 1000001 和 1001110 等 7 位二进制位串，以验证电路的正确性。保存电路设计文件。

四、思考题

1. 组合逻辑电路的一般设计步骤是什么？
2. 测试电路功能有哪几种方式？
3. 如何利用 Logisim 提供的 LED 矩阵显示“NJUCS”五个字符。
4. 简要说明 4 位二进制补码加法器溢出检测电路的设计思路。
5. 如何修改图 11.41 中的电路以产生进位标志 CF、溢出标志 OF、符号标志 SF 和零标志 ZF？

实验 3：同步时序电路设计

一、实验目的

1. 掌握时序逻辑电路设计的基本方法。
2. 掌握计数器和移位寄存器的构建方法。
3. 熟悉计数器和移位寄存器的应用。
4. 掌握寄存器堆的设计方法。