**1. [Síntesis de un sistema combinacional de varias salidas a partir de una especificación verbal. Simplificación de funciones combinacionales.] Una rotonda tiene cuatro calles de entrada y cuatro de salida. La calle A aporta de media 5 coches por minuto, la B 15 coches por minuto, la C 25 coches por minuto y la D 30 coches por minuto. Cuatro sensores, uno por calle, nos indican por qué calle están circulando coches. Las calles de salida son SA, SB, SC y SD y pueden recoger 5, 10, 20 y 40 coches por minuto respectivamente. Teniendo en cuenta que sólo puede haber coches como máximo en dos calles de entrada simultáneamente, activar (poner en verde) los semáforos de las calles de salida cuando sea conveniente para que salgan tantos coches como entran. Realizar la implementación utilizando el mínimo número de puertas lógicas que sea posible.**

Para cada combinación de entradas, la suma de coches que permiten entrar debe ser igual que la suma de coches que permiten salir las salidas. Ejemplo: Si sólo entran coches por la calle D podrán entrar 30 coches por minuto, por lo que se deberán encender los semáforos de las salidas SB y SC (que permiten la salida de 10+20=30 coches/minuto).

Se especifica que sólo puede haber coches en dos calles de entrada simultáneamente. Entonces, todas las salidas aparecerán como una ‘indeterminación’ (X) en los casos en los que entren coches por más de 2 salidas.

Tabla de verdad:

| CA | CB | CC | CD | SA | SB | SC | SD |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 |
| 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 |
| 0 | 0 | 1 | 1 | 1 | 1 | 0 | 1 |
| 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 |
| 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 |
| 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 |
| 0 | 1 | 1 | 1 | X | X | X | X |
| 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 1 | 1 | 1 | 0 |
| 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 |
| 1 | 0 | 1 | 1 | X | X | X | X |
| 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 | X | X | X | X |
| 1 | 1 | 1 | 0 | X | X | X | X |
| 1 | 1 | 1 | 1 | X | X | X | X |

Mintérminos por salida:

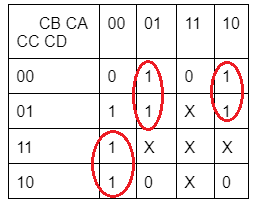
SA: ∑mi (2,3,4,5,8,9)

SB: ∑mi (1,3,4,9,10)

SC: ∑mi (1,2,9,10,12)

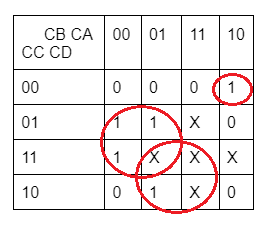
SD: ∑mi (3,5,6)

Tabla de Karnaugh y función simplificada:

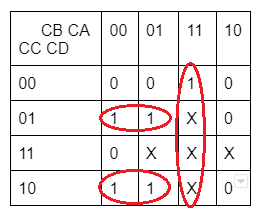
* SA:

SA = CA\*CB’\*CC’ + CA’\*CB\*CC’ + CA’\*CB’\*CD

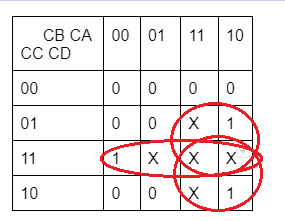
(Siendo CA’,CB’,CC’,CD’ las negaciones de las variables de entrada CA,CB,CC,CD respectivamente)

* SB:

SB = CB’\*CD + CA\*CC + CA’\*CB\*CC’\*CD’

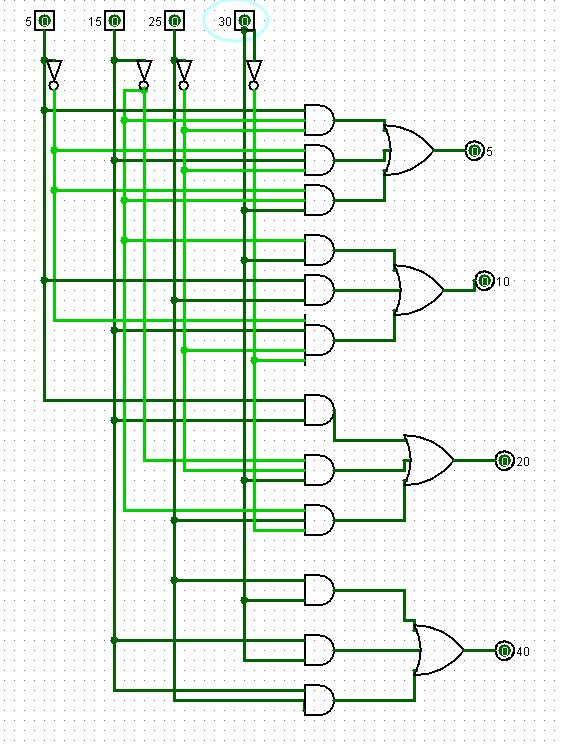
* SC: 

SC = CA\*CB + CB’\*CC’\*CD + CB’\*CC\*CD’

* SD:

SD = CC\*CD + CB\*CD + CB\*CC

Una vez tenemos la función simplificada para todas las salidas, realizamos el circuito en logisim.



**2. [Síntesis de un sistema combinacional a partir de especificación verbal. Utilización de diferentes conjuntos universales de puertas.] Dise˜nar un bloque que reciba como entrada una palabra de 4 bits representando un código BCD y genere una se˜nal llamada ERROR que deberá activarse (ponerse a un valor alto) cuando la palabra de entrada no corresponda a un valor válido BCD. Implementar la función necesaria de las dos maneras siguientes:**

**Utilizando el mínimo n´umero de puertas lógicas AND, OR y NOT que sea posible.**

**Utilizando solamente puertas NOR, si es posible, y en el menor n´umero de puertas que sea posible.**

Las palabras de entrada no válidas seran aquellas cuyo valor decimal sea mayor de 9. Elaboramos la tabla de verdad: (siendo A,B,C,D los bits recibidos)

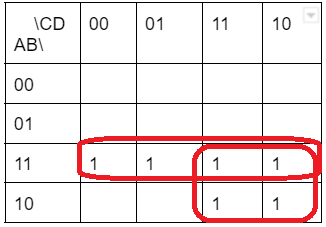
| A | B | C | D | S |
| --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 1 | 1 |
| 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 0 | 1 | 1 |
| 1 | 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 |

Debido a que debemos hacer el circuito con puertas NOR y después con NAND (apartado añadido en clase), expresaremos la función tanto en mintérminos como maxtérminos.

Mintérminos: S: ∑mi (10,11,12,13,14,15)

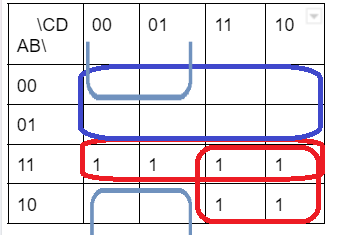
Maxtérminos: S: ΠMi (1,2,3,4,5,6,7,8,9)

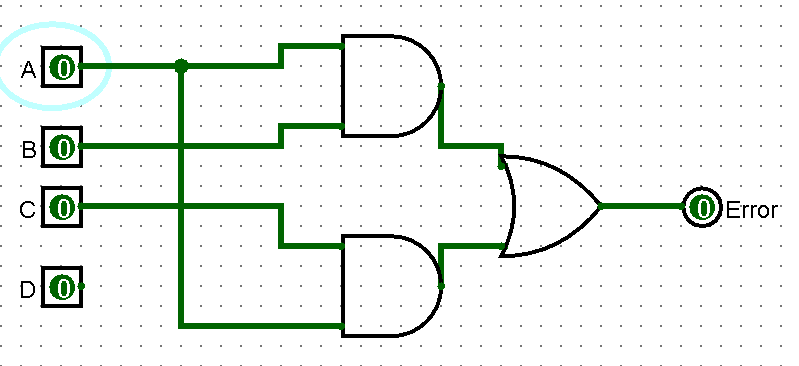
Tabla de Karnaugh y función simplificada:



S: AB + AC

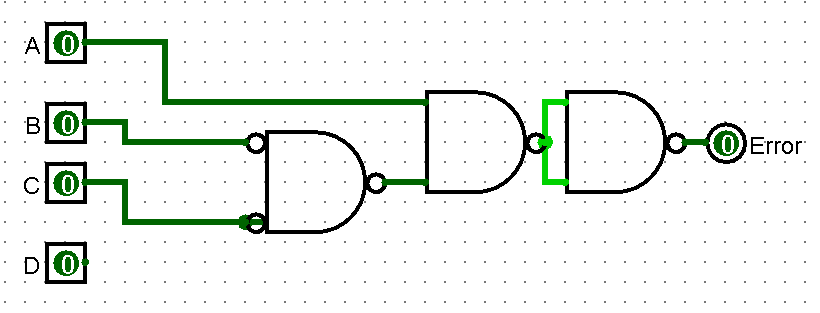
Por maxtérminos:

S: A \* (B + C)

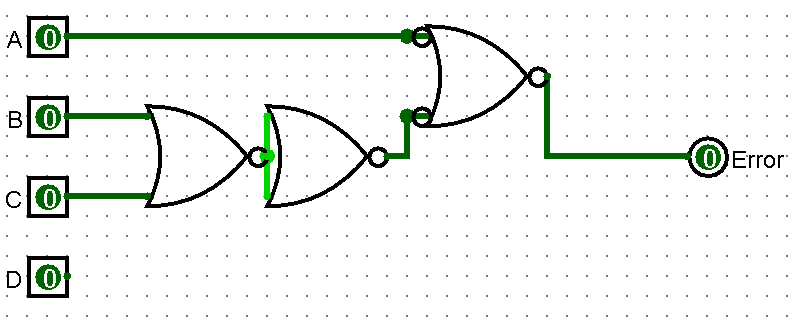
A partir de la función con mintérminos, elaboramos el circuito con puertas OR y AND. 

Elaboramos también el circuito con puertas NAND.

S: AB + AC = A\*(B+C) = A \* (B’\*C’)’ = (( A \* (B’\*C’)’ )’ )’

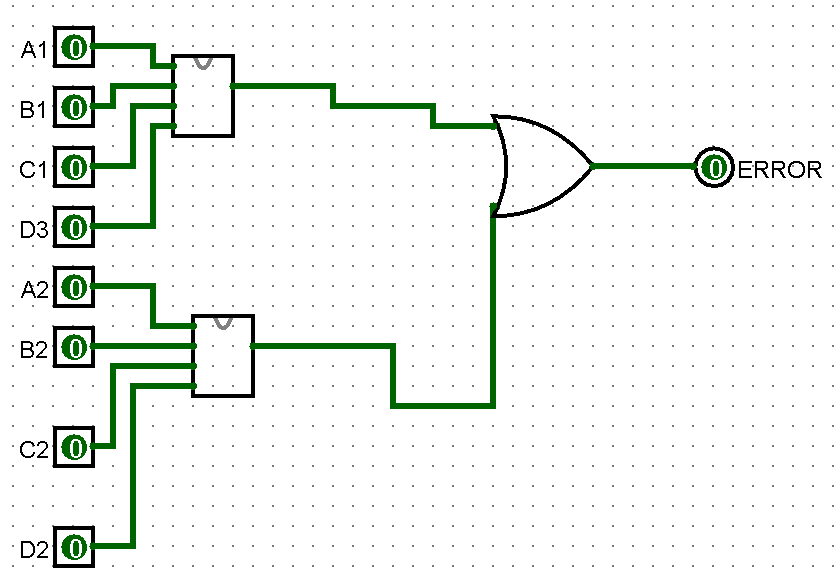


Elaboramos también el circuito con puertas NOR.



**3. [Síntesis de un sistema combinacional a partir de otro más simple usando la definición de un elemento de librería.] Utilizando el módulo dise˜nado en el ejercicio 2, dise˜nar un bloque que reciba como entrada un n´umero codificado en BCD de 2 cifras. Cuando una de las dos cifras recibidas como entrada no corresponda a un valor válido en BCD se debe activar una se˜nal de salida llamada ERROR. Unsar el módulo que se desarrolló en el ejercicio anterior.**

Utilizando la función de importar módulo integrada en el logisim, introducimos el ‘Circuito 2’ elaborado previamente. Conectamos dos de estos circuitos con una puerta OR, pues si cualquiera de los dos da un error, el la salida de ERROR se activa.



**4. [PRACTICA OPTATIVA. Síntesis de un sistema combinacional de varias sali- ´ das a partir de especificación verbal utilizando puertas NOR.] Dise˜nar un circuito de control de la máquina trituradora que describimos a continuación. Esta máquina consiste en un embudo que en su parte inferior tiene un sensor de nivel que detecta cuando la máquina está medio llena (sensor S1) y otro en la parte superior que detecta cuando está totalmente llena (sensor S2). Es decir, los sensores detectan el nivel de los elementos a triturar. La máquina dispone al final del embudo de dos motores trituradores M1 y M2. Cuando el embudo está lleno del todo, tienen que entrar en funcionamiento ambos trituradores, cuando se encuentra medio lleno, sólo tiene que funcionar uno cualquiera de ellos, mientras que si el nivel de llenado no llega a la mitad o si el embudo está vacío, ambos motores se han de parar. Dicha máquina tiene un mecanismo de emergencia a través de un conmutador de triturado de tal forma que cuando está conectado la máquina opera seg´un su contenido, mientras que si está desconectado, la máquina debe pararse. Realiza el dise˜no y la implementación del circuito de control descrito utilizando el mínimo n´umero de puertas que sea posible y utilizando ´unicamente puertas NOR.**

Si el interruptor está desactivado, todas las salidas estarán siempre apagadas.

Si el interruptor está activado y el S1 detecta algo, se activará sólo el M1.

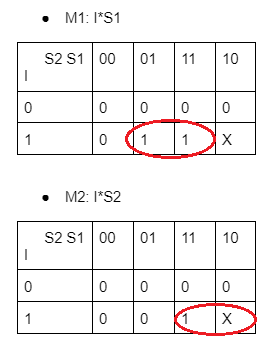
Si el interruptor está activado y ambos sensores detectan algo, se activarán ambos motores.

Si sólo detecta algo el S2, consideraríamos este caso imposible, y lo trataríamos como una indeterminación.

Tabla de verdad:

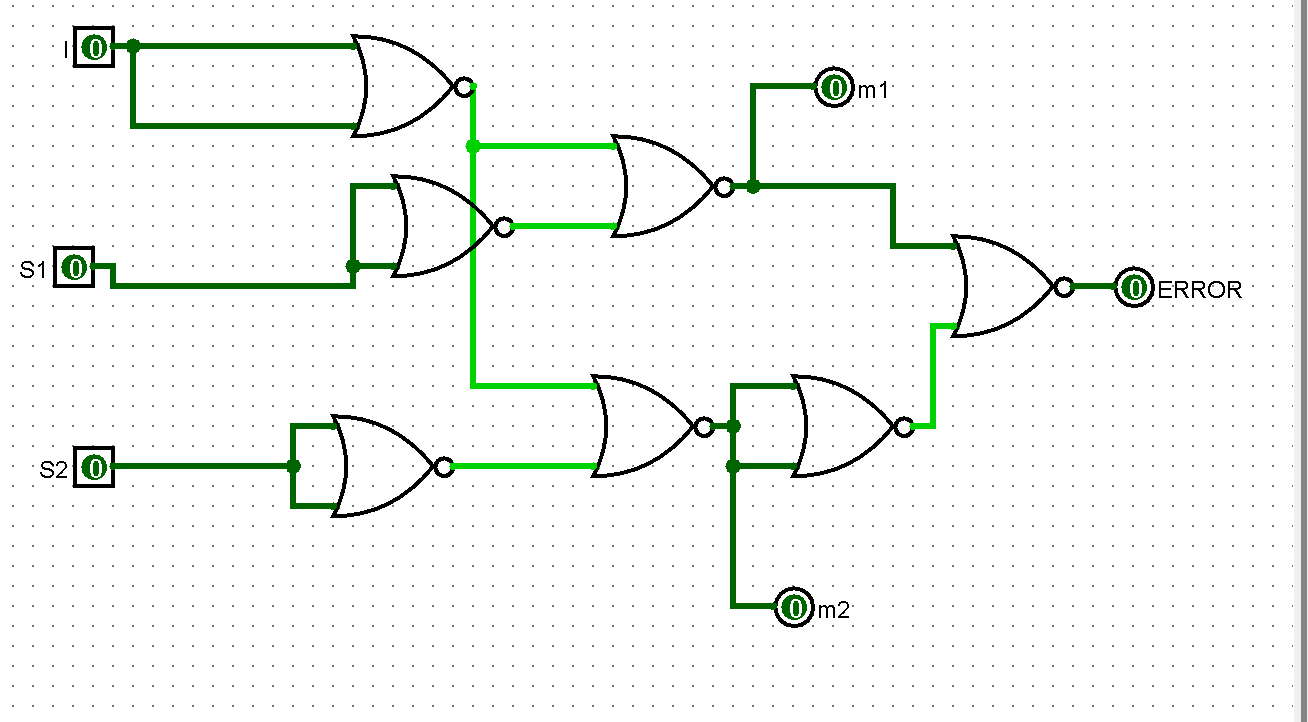
| I | S2 | S1 | M1 | M2 |
| --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | X | X |
| 1 | 1 | 1 | 1 | 1 |

Tabla de Karnaugh y función simplificada:



Utilizando las leyes de Morgan, podemos elaborar ambos circuitos exclusivamente con puertas NOR.

Como ejercicio añadido en clase, incluímos una salida adicional de ‘error’ cuando está activado el S2 pero no el S1. Podemos realizar este cambio de forma sencilla añadiendo una puerta AND conectada a ambas salidas del circuito (estando negada la S1). Cuando esté activado el M2 pero no el M1, se activará la señal.



**5. [ PRACTICA OPTATIVA. Síntesis de un sistema combinacional de varias ´ salidas a partir de especificación verbal] Sea un sistema que controla la temperatura de una sala de ordenadores. El sistema tiene cuatro entradas que son las siguientes:**

**Las variables t1 y t0 indican la temperatura de la habitación en cada momento según la tabla de la figura.**

**La variable D toma valor 1 si es de día y valor 0 si es de noche.**

**La variable A toma valor 1 si el sistema está en modo ahorro de energía (para mantener la temperatura) y valor 0 en caso contrario.**

**El sistema tiene dos salidas. Una activa el aire acondicionado (E) y otra la calefacción (C). Se entiende que cada una de estas salidas está a 1 si el aparato correspondiente está encendido y 0 en caso contrario. El sistema funciona del siguiente modo:**

**Si el sistema está en modo ahorro de energía sólo enfría si T ≥ 35 y sólo calienta si es de día y T < 15.**

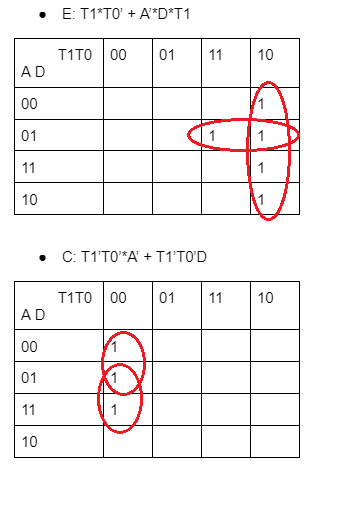
**Si no está en modo ahorro de energía, el sistema debe garantizar que la temperatura se mantiene entre 15 y 34º durante la noche y entre 15 y 24º durante el día. Implementar el sistema utilizando el mínimo n´umero posible de puertas.**

Observando los datos del enunciado, realizamos la tabla de verdad de ambas salidas:

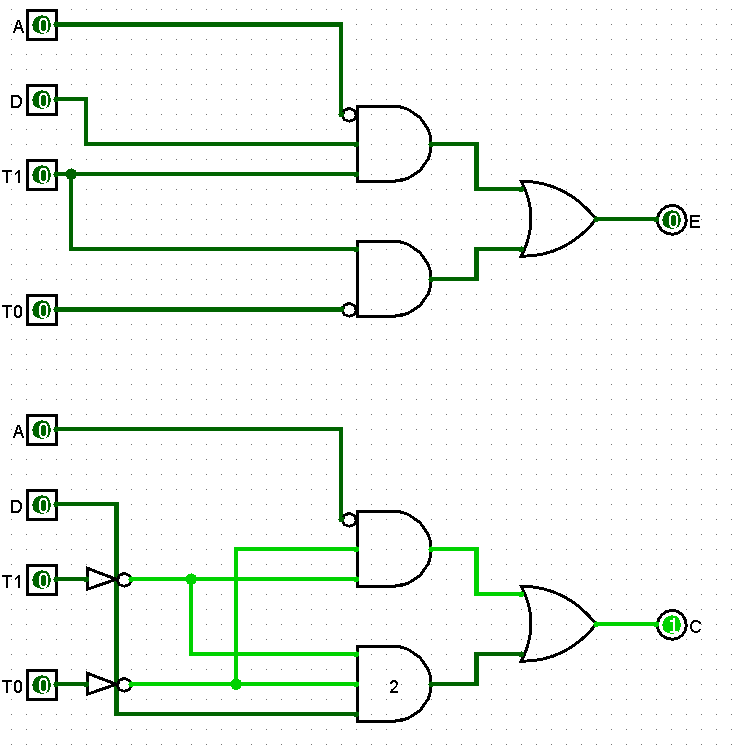
| A | D | T1 | T0 | E | C |
| --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 0 | 1 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 | 0 |
| 0 | 0 | 1 | 1 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 1 |
| 0 | 1 | 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 0 | 0 |
| 1 | 0 | 1 | 0 | 1 | 0 |
| 1 | 0 | 1 | 1 | 0 | 0 |
| 1 | 1 | 0 | 0 | 0 | 1 |
| 1 | 1 | 0 | 1 | 0 | 0 |
| 1 | 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 1 | 0 | 0 |

E: ∑mi (2,6,7,10,14). C: ∑mi (0,4,12)

Tabla de Karnaugh y función simplificada:

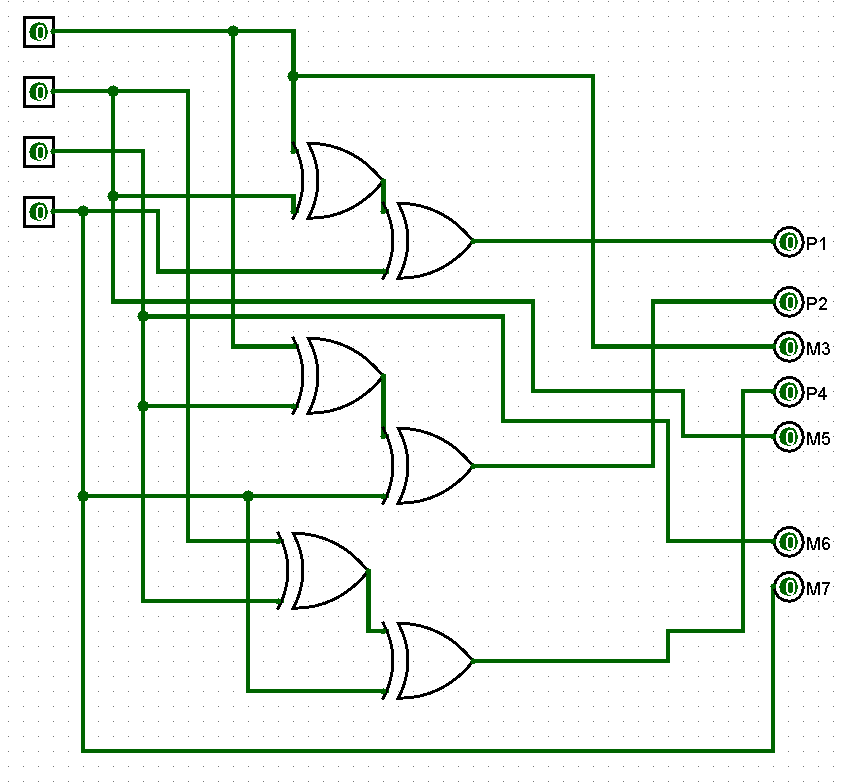


A partir de estas funciones, elaboramos el circuito.



**Prácticas optativas** **relacionadas con bits de paridad:**

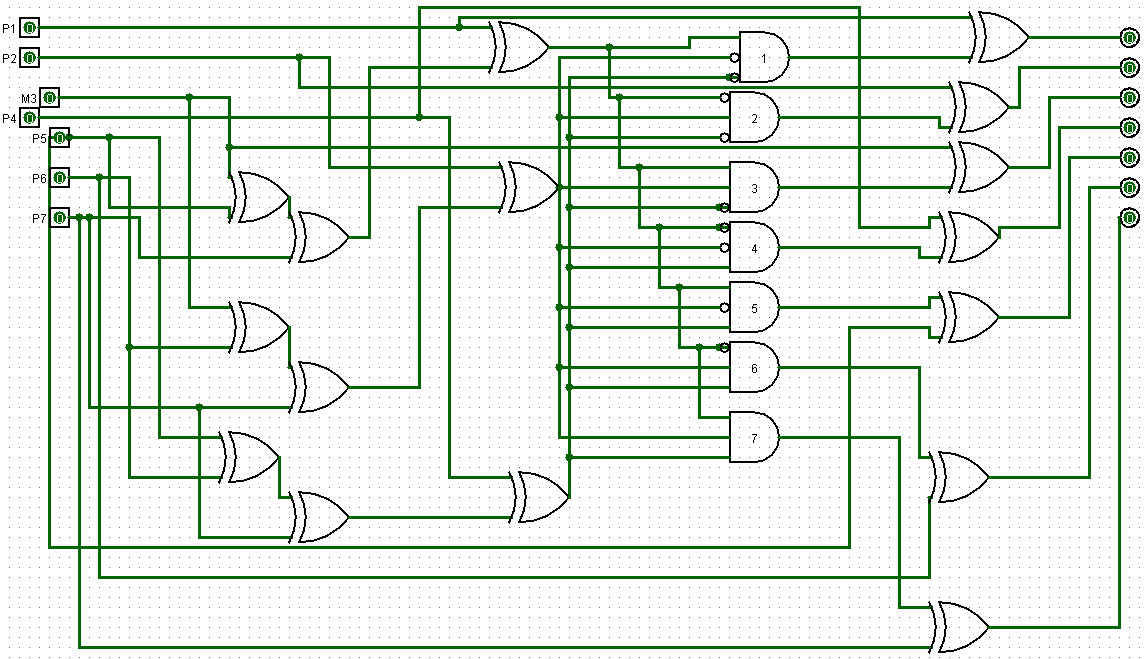
1. **Recibir un mensaje de 4 bits y generar sus 3 bits de paridad.**



La comprobación de paridad se realiza con puertas XOR. Estas puertas devolverán ‘1’ si la suma de las entradas es impar, y ‘0’ si es par. Para sumar los 3 bits que forman el bit de paridad, utilizamos 2 puertas XOR.

Las salidas M3, M5, M6 y M7 están conectadas directamente a las entradas.

1. **Al recibir un mensaje de 4 bits de información y 3 de paridad, comprobar si es correcto y, de no serlo, corregirlo.**



Primero, se calculan los bits de paridad correctos que se corresponden al mensaje recibido mediante el mismo sistema del previo ejercicio. Estos bits se comparan con los bits de paridad recibidos en la entrada mediante una puerta XOR (devuelve 0 si son iguales, 1 si hay error y es necesario corregirlo).

Luego, se utiliza un decodificador que recibe los bits de paridad que necesitan ser intercambiados. Estos bits luego se hacen pasar por un XOR que intercambia su resultado: para los bits erróneos, la salida de la NAND será 1, y se cumple que a⊕1 = a’.

Entonces, la salida será el mensaje introducido con los bits corregidos. Este programa no puede corregir errores en 2 o más bits de forma correcta.