Examen – 19 de febrero de 2014

Instrucciones

- **Indique su nombre completo y número de cédula en cada hoja.**
- **Numere todas las hojas e indique la cantidad total de hojas que entrega en la primera.**
- **Escriba las hojas de un solo lado.**
- **Empiece cada ejercicio en una hoja nueva.**
- **Sólo se contestarán dudas de letra. No se aceptarán dudas los últimos 30 minutos del examen.**
- **Duración: 3 horas.**
- **Requisito para Aprobación: 1 problema y medio correctamente resueltos**

Problema 1

En un centro comercial se ha observado que rara vez se utiliza más de una escalera mecánica simultáneamente, por esto se decidió instalar una sola escalera y colocar un sistema de control con dos sensores. Un sensor irá colocado en la base y el otro en el tope de la escalera (Figura 1).
Se desea que cuando la escalera esté apagada y alguno de los sensores detecte una persona sobre el escalón, ésta se ponga en marcha en el sentido adecuado y permanezca en funcionamiento durante 2 minutos. Se debe evitar el funcionamiento por tiempo indefinido en un sentido (si hay un requerimiento en el otro sentido).

Se pide:
Dibujar un diagrama de estados que modele dicha realidad e implementar en un lenguaje de alto nivel todas las rutinas necesarias para controlar el sistema de escalera.
Se dispone de un reloj externo que interrumpe cada 20 segundos, dicha interrupción es atendida por la rutina **tiempo()**.
Para conocer el estado de los sensores se dispone del puerto de E/S en la dirección **SENSORES** (byte de solo lectura) cuyo bit i-esimo indica el estado del sensor i, i ∈{0,1}. El valor del bit de un sensor es cero si no detecta la presencia de una persona y uno en caso contrario.
Para controlar los motores se dispone del puerto de E/S (byte de solo esctritura) en la dirección **MOTORES**, el bit 0 (msb) controla el motor para bajar y el bit 1 para subir (1 = encender y 0 = apagar).

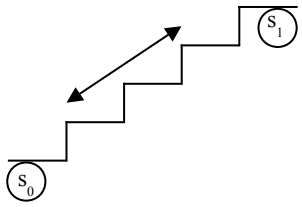
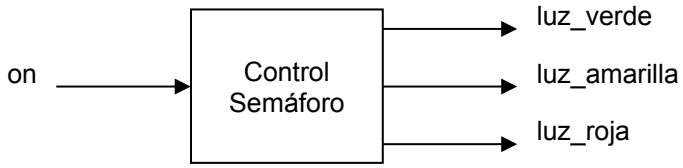


Fig. 1: Escalera

Problema 2

Deseamos implementar un circuito que permita controlar el funcionamiento de las luces de un semáforo. Dicho circuito tiene tres líneas de salida y una entrada de control.



La línea de control **on** indica el modo de funcionamiento del semáforo. Si **on** == '1', entonces el comportamiento es que secuencialmente la luz verde dura 40 segundos, la luz amarilla dura 5 segundos y la luz roja dura 35 segundos y se repite cíclicamente. Si **on** == '0' todas las luces simultáneamente titilaran (prenden y apagan).
Para prender una luz debemos colocar un '1' en la línea de salida correspondiente.

Se pide:

a) Implementar el circuito control semáforo (sin la línea de entrada **on**) utilizando compuertas básicas, flip flop D y un módulo reloj que genera una onda cuadrada de 0.2 Hz.
Sugerencia: implementar previamente un módulo contador de 4 bits.

b) Ampliar el circuito anterior para incorporar la entrada **on**.

Problema 3

Se desea utilizar una memoria ROM para almacenar el contenido de una función x=log(a) siendo a un entero sin signo de 32 bits y x codificado en punto flotante de precisión simple usando la norma IEEE 754.

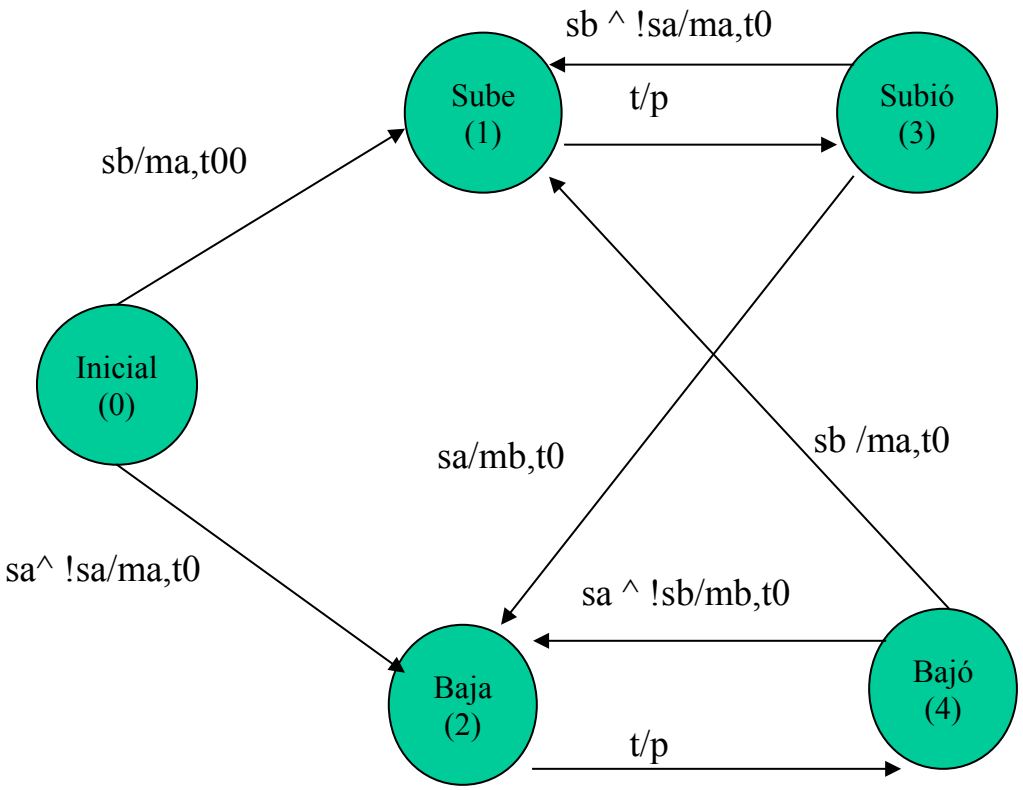
Se pide:

1. Determinar el tamaño y la organización de la ROM especifiando el significado de sus entradas y salidas. Construya la memoria requerida en base a ROMs de 1Gx16.
2. Interesa que los accesos a la memoria construída sean balanceados entre las memorias que la componen. El balanceo debe ser tal que en 4 accesos de direcciones consecutivas respondan al menos una única vez cada una de las memorias componentes. ¿Cumple su circuito estas características? Si es asi, justifique. Si no, indique las modificaciones que permitirían realizarlo y justifique.

Solución Problema 1

Entradas relevantes:
- Sensor_bajo(sb)
- Sensor_alto(sa)
- Timeout(t)

Salidas relevantes:
- Motor_arriba(ma)
- Motor_abajo(mb)
- Para_motor(p)
- Reset_timer(t0)



```
int estado;
int timer;

void main(){
      estado = 0;                              // inicializo variables
      timer = -1;
      out(MOTORES, 0);
      int tmp = 0;
      instalar_int_timer();                    // instalo int.

      for(;;){

            while((tmp=in(SENSORES)&3) == 0);     // polling en el sensor
            switch(estado){

                  case 0,3:             // estado inicial o "subió"
                        if(tmp & 1){          // hay alguien abajo
                              estado = 2;
                              out(MOTORES,1);
                              timer = 0;
                              break;
                        }
                        else{
                              estado = 1;
                              out(MOTORES,2);
                              timer = 0;
                              break;
                        }

                  case 4:                     // estado "bajó"
                        if(tmp & 2){          // hay alguien arriba
                              estado = 1;
                              out(MOTORES, 2);
                              timer = 0;
                              break;
                        }
                        else{
                              estado = 2;
                              out(MOTORES, 1);
                              timer = 0;
                              break;
                        }

                  default: break;

            }

      }


}
```

```
void tiempo(){

      if(timer >= 0){                  // si estoy contando...

            timer++;
            if(timer == 6){            // y pasaron al menos 2 minutos...
                  out(MOTORES, 0);// detengo la escalera,
                  timer = -1;          // reseteo el timer,
                  if(estado == 1){// y modifico el estado
                        estado = 3;
                  }
                  else{
                        estado = 4;
                  }
            }

      }
}
```

Solución Problema 2

a) Para resolver el problema, primero construiremos un contador de 4 bits, y a partir de él, generaremos las salidas necesarias con lógica combinatoria. A pesar de que no se exige la utilización de la metodología del curso para la resolución del problema, apicarla parcialmente es de utilidad para poder visualizar las solución. La tabla de verdad que resuelve el problema es la siguiente:

Estado Actual	**Próximo Estado**	**Salidas**
q3 q2 q1 q0	q3 q2 q1 q0	lv la lr
0 0 0 0	0 0 0 1	1 0 0
0 0 0 1	0 0 1 0	1 0 0
0 0 1 0	0 0 1 1	1 0 0
0 0 1 1	0 1 0 0	1 0 0
0 1 0 0	0 1 0 1	1 0 0
0 1 0 1	0 1 1 0	1 0 0
0 1 1 0	0 1 1 1	1 0 0
0 1 1 1	1 0 0 0	1 0 0
1 0 0 0	1 0 0 1	0 1 0
1 0 0 1	1 0 1 0	0 0 1
1 0 1 0	1 0 1 1	0 0 1
1 0 1 1	1 1 0 0	0 0 1
1 1 0 0	1 1 0 1	0 0 1
1 1 0 1	1 1 1 0	0 0 1
1 1 1 0	1 1 1 1	0 0 1
1 1 1 1	0 0 0 0	0 0 1

Donde se utiliza que el ciclo general de las luces es 16 ciclos de reloj (8 en verde, 1 en amarillo y 7 en rojo), por lo cual sólo se deben adecuar las salidas del contador.

q3q2\q1q0	00	01	11	10
00	0	0	0	0
1	0	0	1	0
11	1	1	0	1
10	1	1	1	1

q3(t+1)=q3'.q2.q1.q0+q3.q2'+q3.q1'+q3.q0'

q3q2\q1q0	00	01	11	10
00	0	0	1	0
01	1	1	0	1
11	1	1	0	1
10	0	0	1	0

q2(t+1)=q2'.q1.q0+q2.q1'+q2.q0'

q3q2\q1q0	00	01	11	10
00	0	1	0	1
01	0	1	0	1
11	0	1	0	1
10	0	1	0	1

q1(t+1)=q1'.q0+q1.q0'

q3q2\q1q0	00	01	11	10
00	1	0	0	1
01	1	0	0	1
11	1	0	0	1
10	1	0	0	1

q0(t+1)=q0'

q3q2\q1q0	00	01	11	10
00	1	1	1	1
01	1	1	1	1
11	0	0	0	0
10	0	0	0	0

lv=q3'

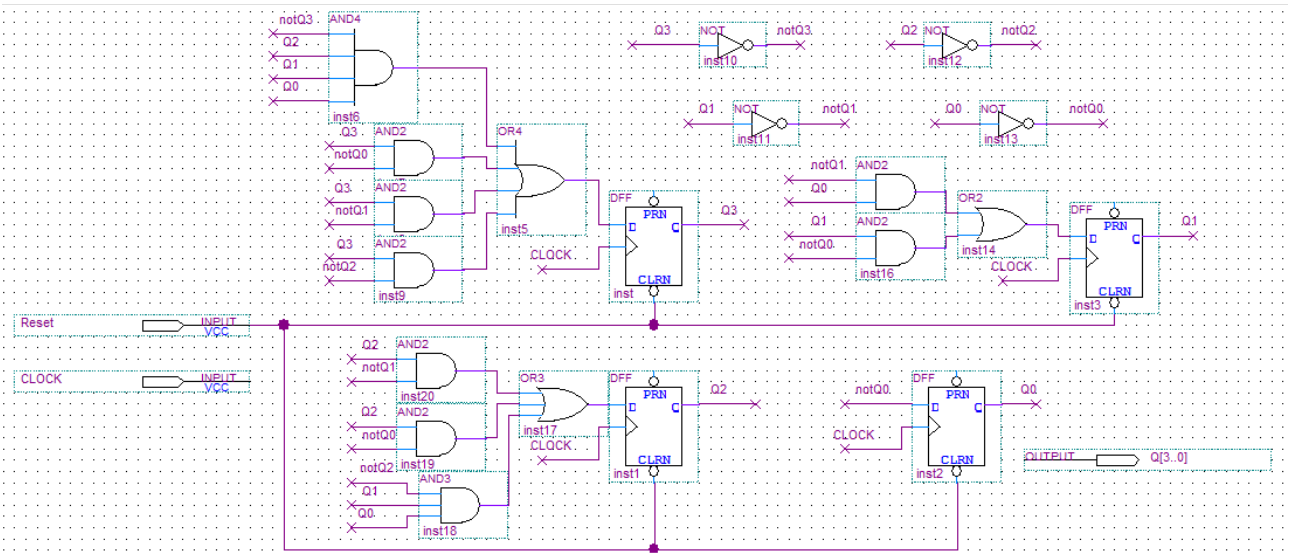
q3q2q1q0	00	01	11	10
00	0	0	0	0
01	0	0	0	0
11	0	0	0	0
10	1	0	0	0

la=q3.q2'.q1'.q0'

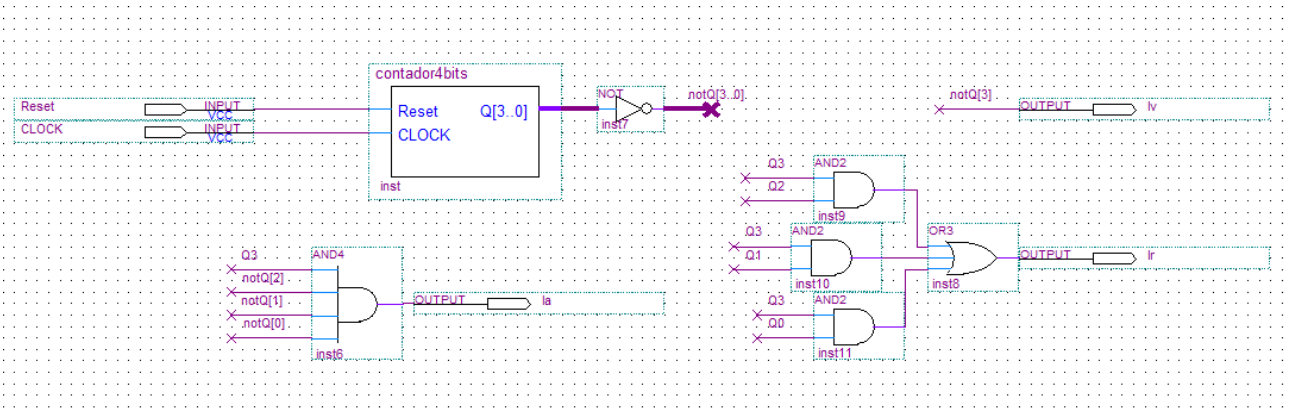
q3q2\q1q0	00	01	11	10
00	0	0	0	0
1	0	0	0	0
11	1	1	1	1
10	0	1	1	1

lr=q3.q2+q3.q1+q3.q0

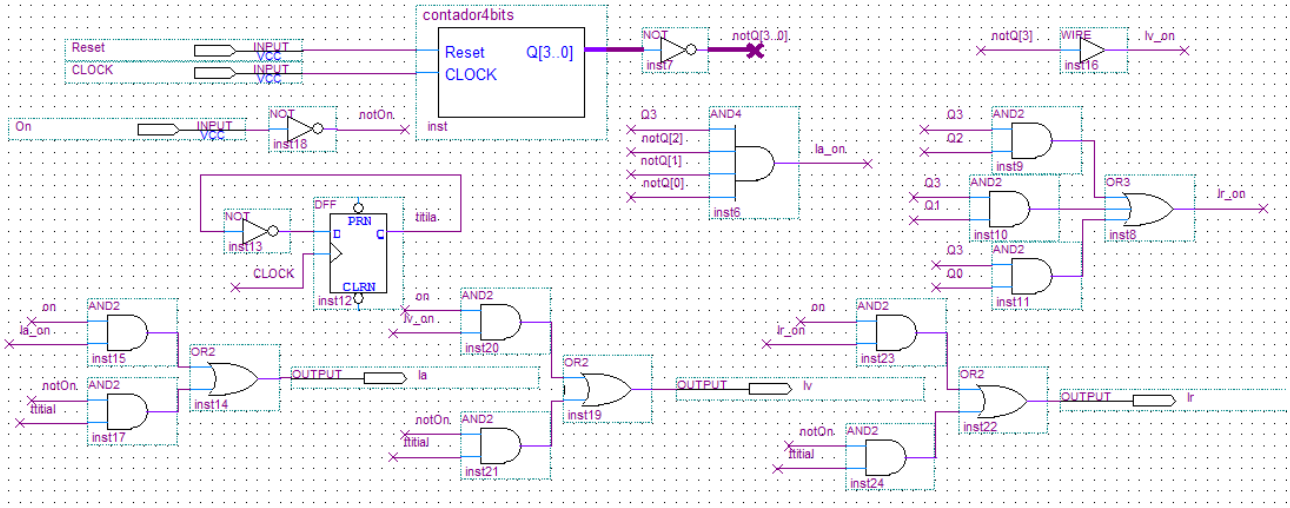
El circuito del contador resulta de la siguiente manera:



Y el circuito general:



b) Para esta parte tendremos en cuenta que la salida a generar en caso de que la entrada 'on' valga cero, es muy sencilla de generar, simplemente se debe usar un contador de 1 bit para generar una onda cuadrada con semiperíodo 0,2 Hz, y conectar dicha salida a las luces.

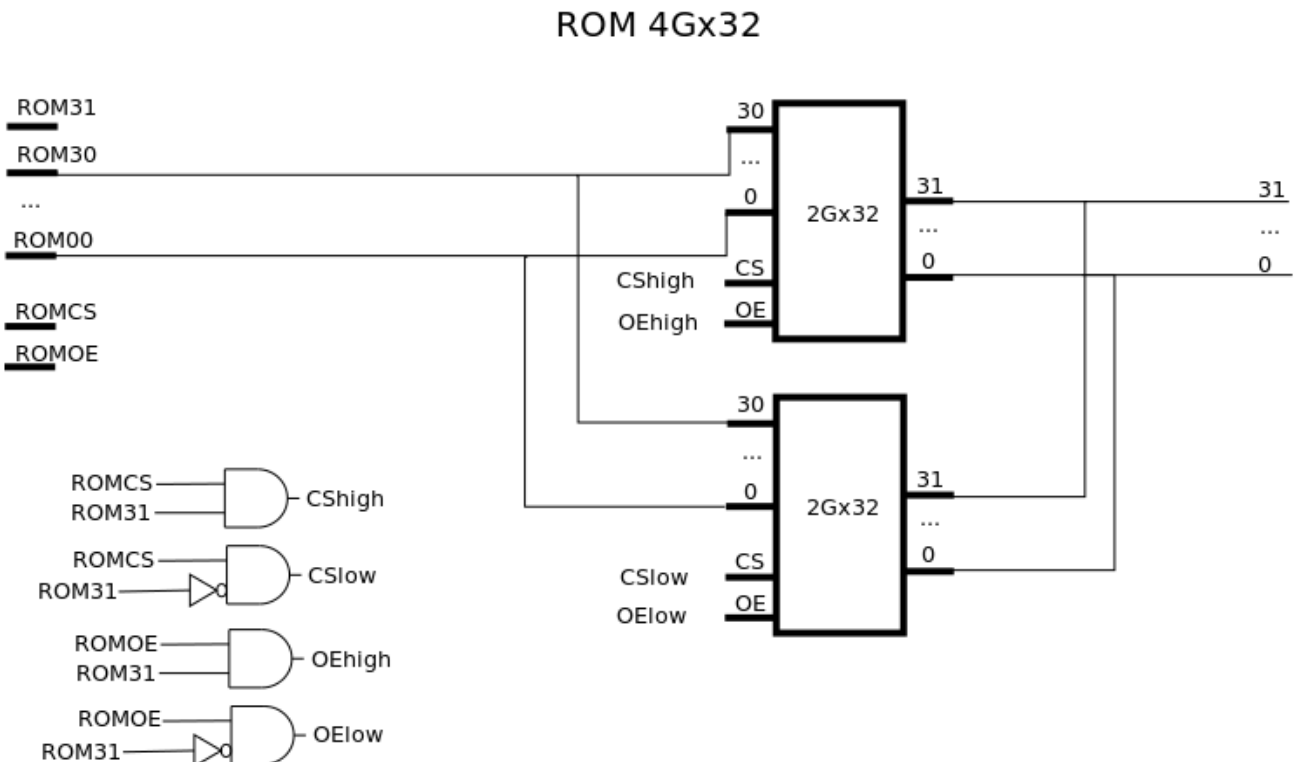


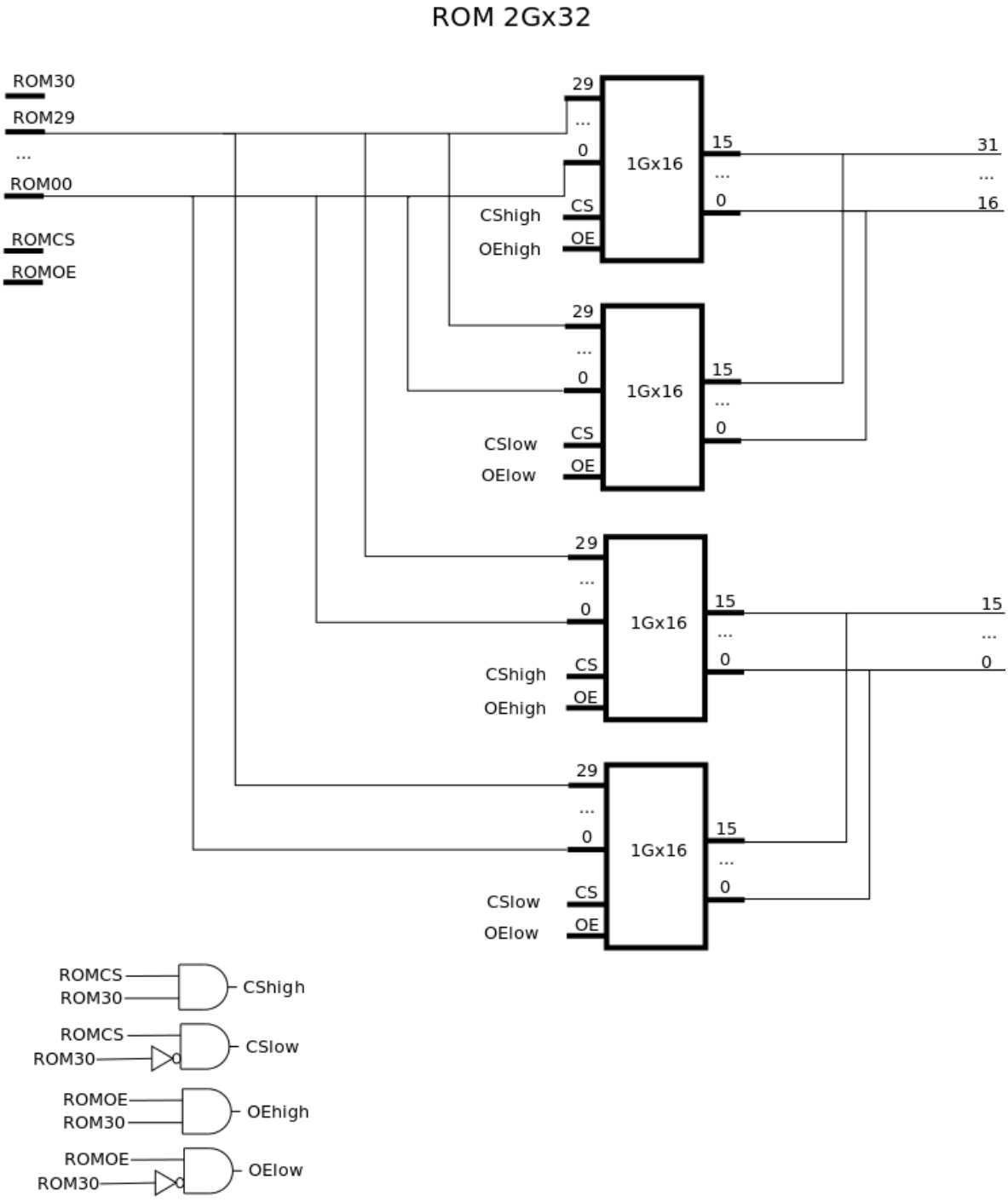
Solución Problema 3

Parte A

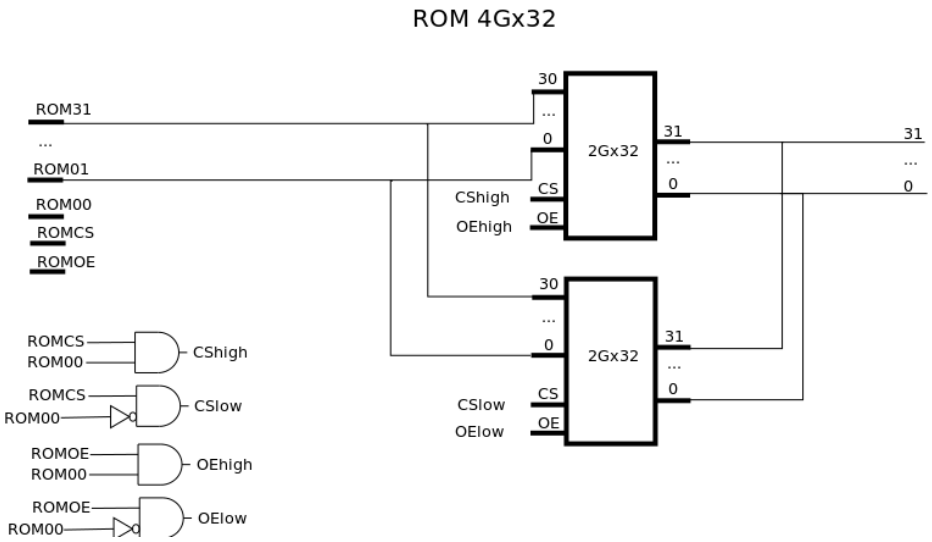
El circuito para implementar la memoria ROM tendrá 32 líneas de entradas de datos para los 32 bits del número entero a y tendrá 32 líneas de salida para el número en punto flotante de precisión simple x. En base a esto, podemos saber que el tamaño de la ROM resultante es de 4Gx32.

Para implementarla debemos utilizar 8 memorias de 1Gx16.




ROM 2Gx32
ROM30
ROM29
...
ROM00
ROMCS
ROMOE
29
...
0
CShigh
OEhigh
CS
OE
1Gx16
15
...
0
31
...
16
CSlow
OElow
15
...
0

ROMCS
ROM30
CShigh
ROMCS
ROM30
CSlow
ROMOE
ROM30
OEhigh
ROMOE
ROM30
OElow



Parte B

La memoria construida no permite el balanceo entre las diferentes memorias internas. Esto puede ver si se accede a las direcciones 0x00000000, 0x00000001, 0x00000002 y 0x00000003 que siempre utilizan la misma memoria de 2Gx32 (dado que el bit más significativo es 0).

Para corregir esto debemos utilizar los bits más significativos para las direcciones y dejar los menos significativos para la elección de las memorias ROM. De esta forma, los 2 bits menos signficativos de la dirección son los que terminan eligiendo la pareja adecuada de memorias 1Gx16.

Como en 4 direcciones consecutivas siempre se dan las 4 posibles combinaciones en los bits menos significativos, sabemos que se acceden todas las memorias internas.