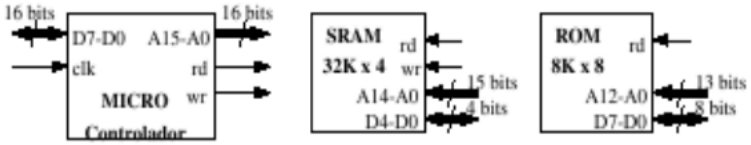
Auxiliar 7
Rodrigo Cánovas
24 de Octubre del 2008

1. Problema 1

El bus de datos de un microcontrolador es de 8 bits y su bus de direcciones de 16 bits. Esto permite direccionar hasta un máximo de 64 KB de memoria. Despúes del encendido, el microcontrolador parte ejecutando la instrucción que se encuentre en la dirección hexadecimal 0xff00 (casi al final de los 64 KB), por lo que se requiere que en esa dirección haya una memoria ROM que se encargue de cargar e inicializar el sistema operativo. Esto es conflictivo porque dado que 64 KB es poca memoria se desea por otra parte que todo corresponda a memoria RAM.
Para resolver el conflicto se usa una técnica que consiste en que justo despúes del encendido el microcontrolador ve una memoria RAM ubicada en [0, 56 KB[y una memoria ROM de 8 KB en [56, 64 KB[. Un programa de la ROM se encarga de cargar e inicializar el sistema operativo en la RAM, ***sin escribir*** en la memoria ROM. La interfaz en hardware de la memoria ROM y RAM hace que cuando se escribe la primera vez en alguna dirección asignada a la memoria ROM (i.e. Rango [56, 64 KB[ ), ésta se reemplaza por 8 KB de memoria RAM adicionales (no inicializada), y asi la primera escritura se realiza en memoria RAM. De ahí en adelante el microprocesador ve que los 64 KB son memoria RAM. No hay forma de acceder nuevamente al contenido de la memoria ROM sin apagar el sistema.
Se le pide a Ud. que diseñe e implemente este sistema a partir de las componentes de la figura. Ud. dispone además de un Latch, que se garantiza que después del encendido su valor es 0, y el resto de las componentes modulares usuales (ands, ors,nots, decodificadores, reloj, etc...).



2. Problema 2

Se desea agregar la siguiente instrucción a M32:

```
LDW++ [<reg-dir>], <val> , <reg-dest>
```

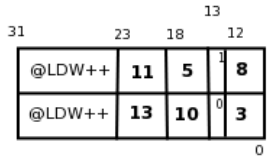
Esta instrucción carga en el registro reg-dest la palabra ubicada en la dirección especificada por el registro reg-dir, y luego, incrementa este último registro en val . A continuación se dan dos ejemplos de uso de esta instrucción indicando al lado cómo operan:

```
LDW++ [R5], 8 , R11          ; R11<-MEM[R5], R5<-R5+8
LDW++ [R10], R3 , R13          ; R13<-MEM[R10], R10<-R10+R3
```

Estas dos instrucciones usan los mismos formatos de instrucción ya existentes en M32:

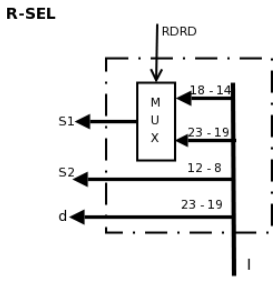
LDW++ [R5] , 8 , R11

LDW++ [R10] , R3 , R13



i. Explique por qué no se puede implementar esta instrucción con el actual diseño de M32. Es decir explique por qué no existe ninguna secuencia de señales que pueda aplicar la unidad de control para implementar LDW++.

ii. Dado que el diseño de R-SEL es el siguiente:



¿Cómo modificaría este circuito para que se pueda implementar LDW++? Agregue señales de control si es necesario.

iii. Utilizando el nuevo R-SEL, indique ciclo por ciclo las señales de control necesarias para ejecutar LDW++ (no incluya la fase de fetch ni la fase de decodificación). Indique además para cada ciclo las transferencias entre registros que se realizan.