Instituto Tecnológico de Buenos Aires

22.05 Análisis de Señales y Sistemas Digitales

Trabajo práctico N°N

Grupo 3

Mechoulam, Alan	58438
Lambertucci, Guido Enrique	58009
Rodriguez Turco, Martín Sebastian	56629
Londero Bonaparte, Tomás Guillermo	58150

Profesores
Jacoby, Daniel Andres
Belaustegui Goitia, Carlos F.
Iribarren, Rodrigo Iñaki

Presentado: ??/??/20

Índice

1. Ejercicio 1

1.1. Implementación por lógica discreta

Para la lógica combinacional lo primero que se hizo fue escribir las posiciones de memoria en las que vivirá nuestro periférico en binario, teniendo en cuenta que la memoria es de 8k (8192) ocupará 0x2000 posiciones de memoria.

Address	**Dispositivo**	**Binario Comienzo**	**Binario Fin**
0x4000	RAM	010000000000	011000000000

Tabla 1: Parámetros de la memoria RAM.

De aqui se arma la tabla de verdad de los últimos 3 bits mas significativos.

$\mathbf{a_{15}}$	$\mathbf{a_{14}}$	$\mathbf{a_{13}}$	**CS**
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

Tabla 2: Tabla de verdad.

De aquí se puede solucionar el mapa de karnaugh para la siguiente configuración:

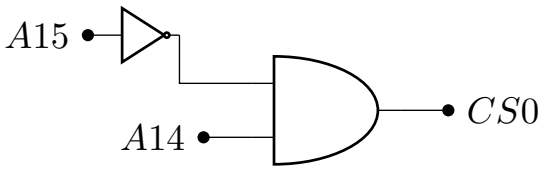

Figura 1: Lógica Discreta

.

1.2. Implementación por lógica de baja complejidad

Se utilizó el decodificador 74LS139, conectando a los pines a_{15} y a_{14} a las entradas B y A respectivamente, el CS será la salida Y_1 quedando de la siguiente manera

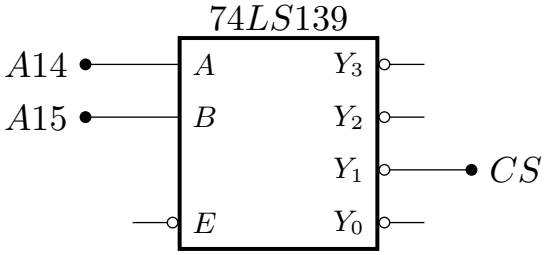

Figura 2: Lógica de baja complejidad

.

1.3. Implementación por medio de una PAL

Se utilizó una PAL como decodificador de direcciones, como se observa en la Tabla (2) es posible detectar el perisferico viendo únicamente los bits a_{15} y a_{14} asi se llega a la siguiente ecuación:

$$x_1 = a_{15} \qquad x_2 = a_{14} \tag{1}$$

$$f1 = CS \quad f1 = \bar{x_1} \ \& \ x_2 \tag{2}$$

1.4. Análisis y construcción del diagrama de tiempos

Se construyó para el microprocesador M68HC11 el diagrama de tiempos para un ciclo de lectura/escritura, usando como ejemplo la posición de memoria $2345, la cual está dentro de la hipotética región del mapa de memoria donde se aloja la memoria para la cual se diseñó el decodificador anteriormente.

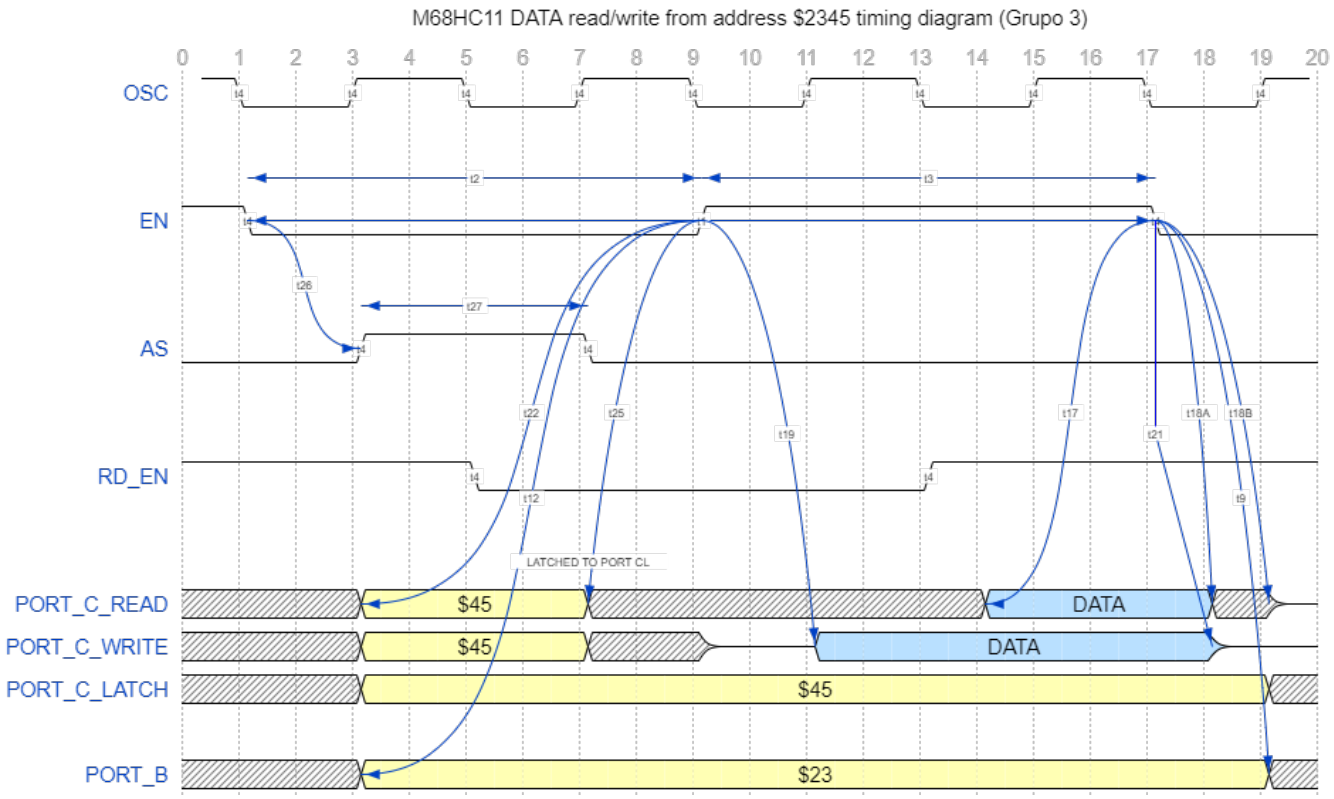

Figura 3: Ciclo de lectura/escritura de *DATA* en la dirección de memoria *$2345*

.

Para el análisis de tiempos se tiene en cuenta una frecuencia característica de 2 MHz. Dado esto, se obtiene un rise time de las señales de $t_4 = 20\ ns$ y un periodo entre ciclos de lectura/escritura de $t_1 = 500\ ns$, por lo que los tiempos en alto y bajo de la señal **E** de enable serán de $t_3 = 230ns$ respectivamente.

1.4.1. Primera mitad del ciclo de escritura/lectura

El comienzo del ciclo de lectura o escritura comienza con el flanco descendente de la señal de enable. Un tiempo $t_{26} = 53\ ns$ después se activa la señal **AS** de address strobe, lo cual indica que se utilice el bus de address entero para cargar la parte baja y alta de la dirección de memoria en los puertos C y B del M68HC11 respectivamente. Esta señal se desactiva luego de un tiempo $t_{27} = 96\ ns$ activando el latch que retendrá la parte baja de la dirección de memoria. De esta manera se logra multiplexar la parte baja del bus de address, o puerto C, para leer o escribir datos al igual que retener la parte baja de la dirección del mapa de memoria.

El puerto C tiene la dirección de memoria por un tiempo válido de $t_{t22} = 88\ ns$ como mínimo y el puerto B por un tiempo de $t_{12} = 94\ ns$ como mínimo, que corresponde con el flanco ascendente de la señal de enable y marca la mitad del ciclo de lectura/escritura.

1.4.2. Segunda mitad del ciclo de escritura/lectura

Lectura: En el caso de la lectura, el tiempo de setup para que el periférico coloque el dato a su salida y lo mantenga estable antes del flanco descendente de la señal de enable es de $t_{17} = 30\ ns$ y debe ser mantenido estable por $t_{18A} = 10\ ns$ pasado dicho flanco. Luego pasa a hiZ el puerto C pasados $t_{18B} = 83\ ns$ de dicho flanco.

Escritura: Para el caso de la escritura, el puerto C tiene un delay máximo para contener el dato a escribir de $t_{19} = 128\ ns$ y un tiempo de hold de $t_{21} = 33\ ns$ como mínimo, por lo cual el tiempo de escritura deberá ser como máximo de $t_3 + t_{21} - t_{19} = 143\ ns$.

Finalmente, el address se mantendrá por un tiempo de t_9 tras el flanco descendente de la señal de enable, por lo que el tiempo válido de lectura de la dirección de memoria en un ciclo de $t_1 = 500ns$ será de $t_1 - t_{26} + t_9 = 480\ ns$.

2. Ejercicio 2

Para este ejercicio se implemento el decodificador de direcciones con un integrado 74LS138 y lógica combinacional. Valiendonos de que únicamente se utilizaran 4 periféricos, es posible asignar cantidades de memoria por demás a los mismos. Quedando de la siguiente manera la tabla de direcciones.

Adress	Dispositivo	Binario Comienzo	Binario Fin
C000	ROM16K	1100000000000000	1111111111111111
A800	Entrada 8 Bits	1010100000000000	1011111111111111
A000	Salida 8 Bits	1010000000000000	1010011111111111
2000	RAM4K	0010000000000000	0010111111111111

Tabla 3: Tabla de dispositivos y direcciones.

Basta con conectar los siguientes bits con el decoder

$$a_{14} \implies C \tag{3}$$
$$a_{15} \implies B \tag{4}$$
$$a_{11} \implies A \tag{5}$$

así también conectar las salidas Y_0 Y_1 con una compuerta or, al igual que las Y_6 Y_7 con otra. quedando definidos los chip select de la siguiente manera.

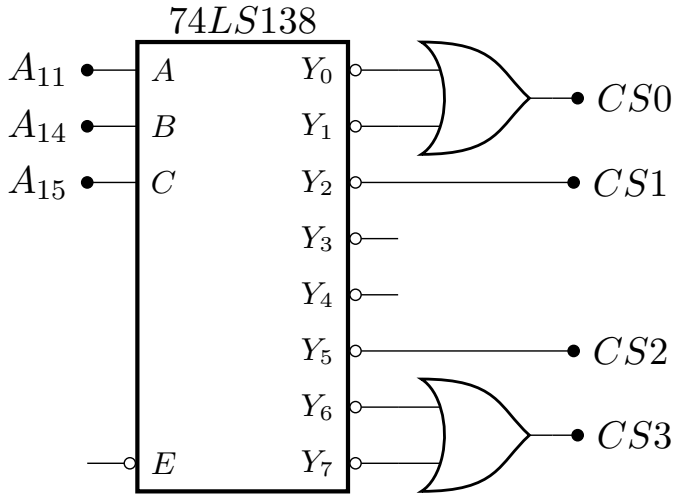

Figura 4: Diagrama en bloques

.

3. Ejercicio 3

3.1. Single-Chip mode y Expanded mode

Los microprocesadores cuentan con una cantidad finita de accesos que, dependiendo de que tipo de pin sea, puede adoptar un carácter unidireccional o bidireccional. Para que la MCU tenga la posibilidad de interactuar con diferentes periféricos, es necesario que este tenga la habilidad de poder comunicarles cuando se los requiere. Esta conexión recibe el nombre de **chip-select**.

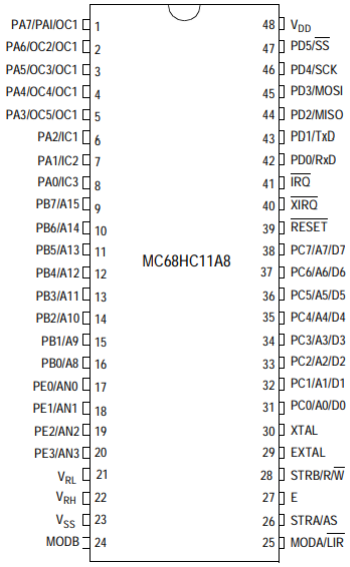

Figura 5: Microprocesador HC11

El **HC11** posee un bus de datos de 8 bits y un puerto dedicado a indicar direcciones, el **address-bus**. Este puerto recibe el nombre de **Puerto B**. Este esta conformado por 8 bits. En single-chip-mode este puerto nos permitiría conectarnos con hasta 8 periféricos, es decir que nos provee con solo 8 lineas de chip-select. Para poder acceder a memorias y chips adicionales se introduce el **expanded-mode**.

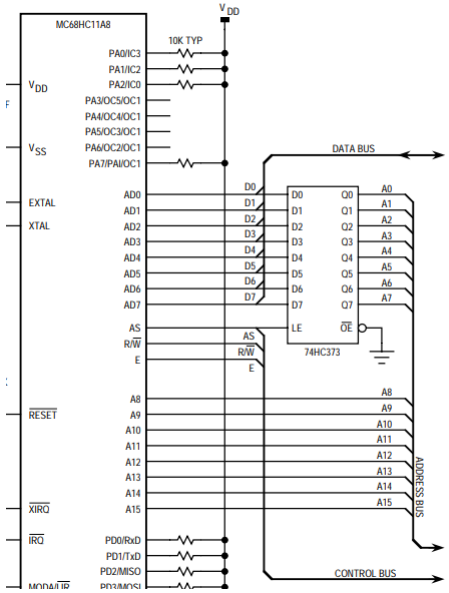

Figura 6: Microprocesador HC11 en modo expandido

El modo expandido del **HC11** utiliza su bus de datos como un bus de address de manera temporal, este se encarga de escribir la parte baja de los nuevos 16 bits de address. Sin embargo, durante la comunicación con los periféricos, el bus de datos cumple la función de intercambiar datos o de dar instrucciones. Es por eso que se introduce un nuevo

bloque a la salida del **Puerto C**, un bloque del tipo latch. Este se encarga de "recordar" la parte de baja que fue escrita por el bus de datos y dejarlo libre. Esto significa que ahora hemos conseguido duplicar la cantidad de linea de chip-select disponibles consiguiendo 16 lineas. Sin embargo es posible aumentar aún más el número de perifericos a los que se puede acceder si implementamos un *decoder/demux* tal que el bus de address sea la entrada del mismo y el chip select sea su salida. La implementación más burda y directa de este decoder nos permitirá acceder a $2^16 = 65536$ periféricos. No obstante, en la practica se utilizan diferentes circuitos de lógica de digital que utilizan de forma más inteligente el bus de address. Además debemos considerar que si se utilizasen las 65536 lineas de chip-select antes mencionadas, el área que ocuparían sería excesivamente grande.

3.2. Fanout

Una vez resuelto el problema de **cómo** realizar la interconexión *perifericos-cpu* cabe preguntarse el efecto que tendrán todas esas lineas conectadas a la cpu. Si se le demanda a la cpu por sobre su capacidad de **Fanout** se deben añadir buffers que añadirán retraso y por ende bajaran la velocidad máxima del sistema, pero sera capaces de soportar más cargas.

4. Ejercicio 4

4.1. Introducción

Se pidió investigar soluciones para la interconexión de un sistemas TTL de 5V con uno de 3.3V. Para esto proponemos dos soluciones.
La primera es la utilización de TTL[1] y LVTTL[2] tal que los voltajes queden de la siguiente manera siendo compatibles entre sí.

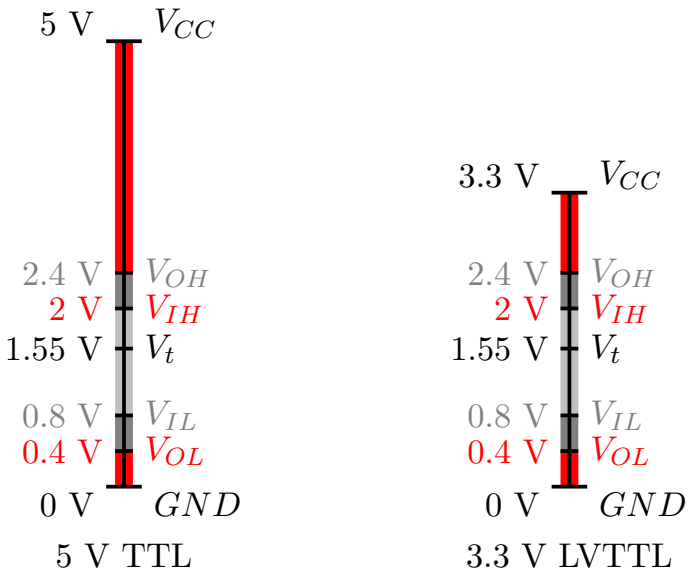

Figura 7: Niveles Lógicos TTL y LVTTL

.

La segunda es la implementación de un Level Shifter mediante un transistor MOS y dos resistencias (8), esta configuraicón permite la traducción de los niveles lógicos en ambos sentidos.

[1] Transistor Transistor Logic
[2] Low Voltage Transistor Transistor Logic

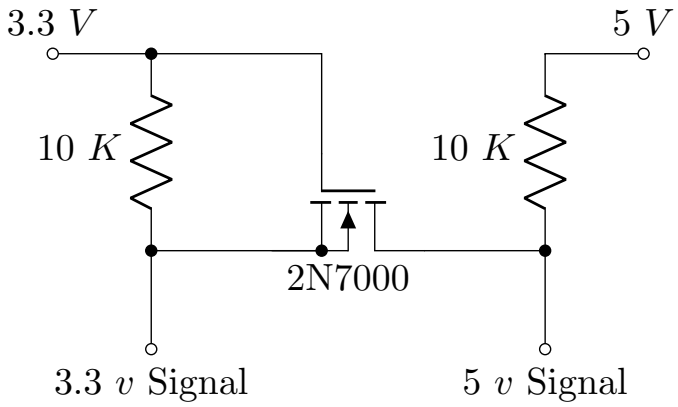

Figura 8: Level Shifter

.

5. Ejercicio 5

Se quiere construir el diagrama de tiempos del MC68HC11 para el programa de la Tabla (4).

	org	$C000	
	ldaa		#$A5
L1	staa		$4000
	jmp		L1

Tabla 4: Programa a implementar.

Para esto, se construye la Tabla (5) donde se descompone a cada instrucción en los ciclos que la componen.

Instruction	Cycle	Address	Data
LDAA	1	$C000	$86
	2	$C001	$A5
STAA	3	$C002	$B7
	4	$C003	$40
	5	$C004	$00
	6	$4000	$A5
JMP	7	$C005	$7E
	8	$C006	$C0
	9	$C007	$02

Tabla 5: Descomposición en ciclos del programa a implementar.

Finalmente, se construye el diagrama de tiempos teniendo en cuenta el modo extendido del MC68HC11 en el cual el bus de address está compuesto por el puerto C para los ocho bits menos significativos y el puerto B para los ocho bits más significativos. A su vez, el puerto C está multiplexado de manera tal que funcione como bus de datos en el semiciclo bajo de la señal de enable.

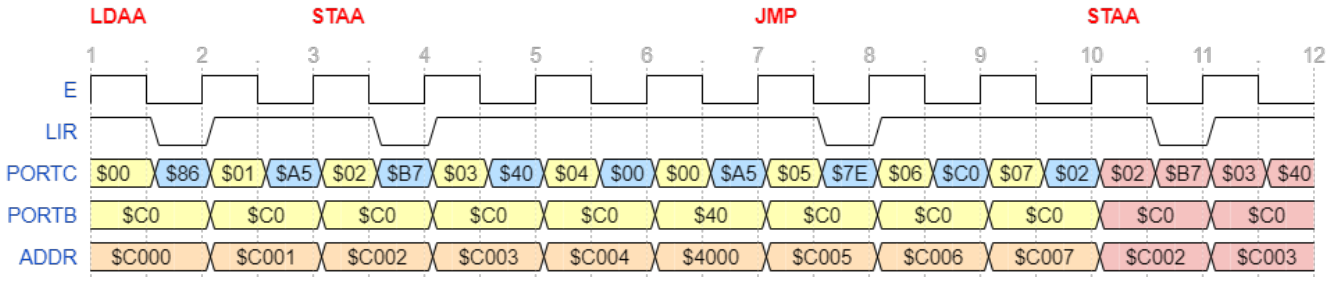

Figura 9: Diagrama de tiempos del programa de la Tabla 4.

La señal de ADDR indica el valor del bus de address visto como la concatenación del puerto C latcheado y el puerto B. La señal de LIR es una señal activa baja de ayuda al momento de debuggear y marca el primer semiciclo negativo de cada ciclo de cada nueva instrucción. Esto es útil debido a que, como se puede ver en la Figura (9), la señal LIR tendrá un valor bajo cuando el puerto C retiene el OPCODE, el cual identifica qué instrucción ejecutará el M68HC11.