Arquitectura de Computadoras

Parcial 2012

- *El parcial consta de 5 preguntas que se deben responder por escrito en hojas aparte. Completar TODAS las hojas con el nombre y el número de cédula. Numerarlas y escribir el total en la primer hoja.*
- *Utilizar una única carilla e iniciar cada respuesta en una hoja nueva.*
- *No se puede utilizar material de ningún tipo.* ***Apagar celulares.***
- *Se dispone de cartillas Intel 8086 y MIC-1.*
- *El parcial dura 1 hora y media.*
- *Se contestarán preguntas hasta 20 minutos antes de la hora de finalización.*

Pregunta 1

Represente mediante un diagrama por cada elemento cómo se disponen en memoria en 8086 cada una de las estructuras dadas a continuación. (Nota: asuma que todas se ubican a partir de la dirección 0x300 de memoria)

```
short entero;
char letra;
short arregloEntero[2];
struct letrasEnteros{
      char letras[2];
      short entero;
} letrasEnteros;
letrasEnteros arregloRegistros[2];
```

Pregunta 2

a) ¿Qué se gana y qué se pierde en la jerarquía de memoria según qué tan cerca se encuentra de la CPU?
b) Enumere los 3 tipos de organización para memorias caché e identifique para cada una las partes de la dirección. ¿Cuál es la utilidad del componente TAG? ¿Por qué siempre abarca los bits más significativos?

Pregunta 3

a) Explique cómo puede ser utilizada una ROM para resolver una función lógica arbitraria.
b) Explique cómo puede ser utilizado un multiplexor para resolver funciones lógicas generales.

Pregunta 4

a) Dibuje el circuito de un Flip-Flop (Latch) R-S y explique por qué tiene la propiedad de recordar entradas.
b) A partir del Flip-Flop (Latch) de la parte anterior, construya un Flip-Flop tipo D sincrónico.

Pregunta 5

Explique qué realizan las siguientes instrucciones en Assembler 8086. Para cada una de ellas, indique los modos de direccionamiento de cada uno de los operandos.

```
cmp byte ptr ES:[BX], 0
pop AX
mov DL, DS:[0x30]
```

Respuestas

Pregunta 1

Respuesta:
Cada casillero en las tablas representa un byte.
short entero:

Dirección	Dato
0x300	entero(parte baja)
0x301	entero(parte alta)

char letra:

Dirección	Dato
0x300	letra

short arregloEntero[2]:

Dirección	Dato
0x300	entero[0](parte baja)
0x301	entero[0](parte alta)
0x302	entero[1](parte baja)
0x303	entero[1](parte alta)

struct letrasEnteros:

Dirección	Dato
0x300	letras[0]
0x301	letras[1]
0x302	entero(parte baja)
0x303	entero(parte alta)

letrasEnteros arregloRegistros:

Dirección	Dato
0x300	arregloRegistros[0].letras[0]
0x301	arregloRegistros[0].letras[1]
0x302	arregloRegistros[0].entero(parte baja)
0x303	arregloRegistros[0].entero(parte alta)
0x304	arregloRegistros[1].letras[0]
0x305	arregloRegistros[1].letras[1]
0x306	arregloRegistros[1].entero(parte baja)

0x307	arregloRegistros[1].entero(parte alta)

Pregunta 2

Respuesta:

1. Cuanto más cerca de la CPU, se gana velocidad de acceso a la memoria y se pierde tamaño. Cuanto más lejos de la CPU, lo opuesto.
2. Las tres organizaciones de caché son: Direct mapping, Fully Associative y N-Way Associative.
   Direct mapping: [TAG][LÍNEA][BYTE]
   Fully Associative: [TAG][BYTE]
   N-Way Associative: [TAG][SET][BYTE]
   El Tag sirve para verificar si el contenido en una línea dada es el correspondiente a la dirección buscada, ya que si coincide la línea o el set y luego además coincide el tag, existe un caché hit y se busca dentro del bloque el byte indicado por byte.
   El TAG se ubica en los bits más significativos de forma tal de permitir que bloques contiguos de memoria (que comparten los bits más significativos) puedan ubicarse en la caché simultáneamente ocupando líneas distintas.

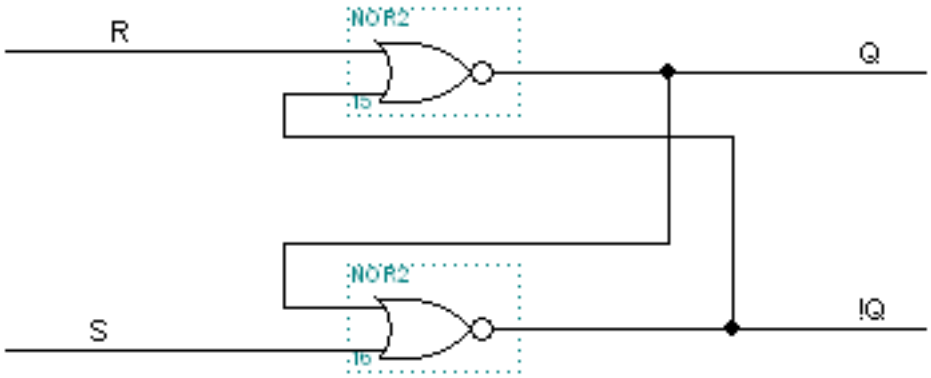
Pregunta 3

Respuesta:

1. Sea F la función lógica arbitraria, de n bits de entrada y m bits de salida. Una ROM de 2^n * m computará F cargando en cada posición X de la ROM, el dato F(X). Así, poniendo como entrada en la ROM la tira de n bits, X, la salida será la tira de m bits que conforma F(X).
2. Se utiliza un multiplexor de 2^n x m donde en las 2^n entradas se coloca el valor (m bits) que toma la función para cada posible combinación de entradas y las n entradas se conectan con el selector del multiplexor.

Pregunta 4

Respuesta:
Parte A -

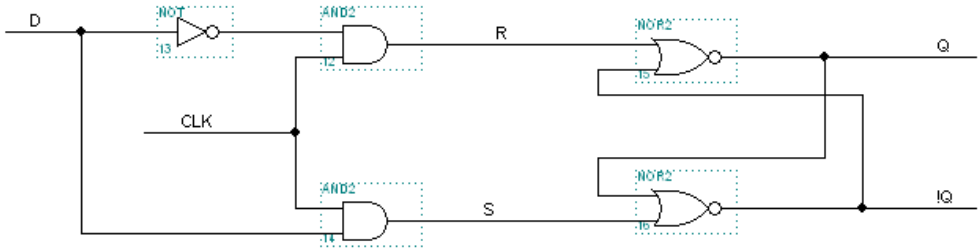


La tabla de verdad del latch RS es la siguiente:

R	S	Q(t+1)
0	0	Q(t)
0	1	1
1	0	0
1	1	-

Aceptando que a lo sumo una de las entradas R y S puede valer uno en cada instante de tiempo, veamos:
si R=1, tenemos que Q=0(sin importar el valor de !Q), por lo que las entradas del segundo NOR serán (0,0) por lo que !Q será 1 Si luego R pasa a ser 0, la entrada del primer NOR pasa de (1,1) a (0,1), con lo que su salida no cambia(y por tanto tampoco las entradas o salidas del segundo NOR).
Algo análogo pasa si S pasa de ser 1 a 0.
Por lo que para todo tiempo la salida Q presentará cuál fue la última entrada que valió uno, de la siguiente forma: Si R dejó de ser uno de último, Q valdrá cero, y si S dejó de ser uno de último, Q valdrá uno.

Parte B)



Pregunta 5

Respuesta:

```
cmp byte ptr ES:[BX], 0
```

Esta instrucción efectúa la resta entre el byte contenido en la dirección ES * 16 + BX, con 0, y actualiza el registro de estado (flags) con los valores correspondientes según el resultado de la resta.
Los modos de direccionamientos son:

ES:[BX] -> Indirecto por registro.
0 -> Inmediato.

```
pop AX
```

Esta instrucción carga la palabra de 16 bits en la dirección SS * 16 + SP en AX y aumenta en 2 el valor de SP.
El modo de direccionamiento es directo a registro.

```
mov DL, DS:[0x30]
```

Esta instrucción carga el byte contenido en la dirección DS * 16 + 0x30 en DL.
Los modos de direccionamiento son:

DL -> Directo a registro
[0x30] -> Directo a memoria