ESTRUCTURA DE COMPUTADORES

AUTOEVALUACIÓN. SOLUCIONES.

Tema 0

1. ¿Cuál es el elemento dominante de una operación AND? ¿Y de una O?

   El '0'. El '1'.

2. Para la representación de números binarios con signo ¿qué ventajas tiene el complemento a dos que hace que sea el más utilizado?

   Permite realizar restar mediante la suma. La resta de dos números en binario natural puede llevarse a cabo mediante al minuendo del sustraendo en complemento a dos. El resultado es la diferencia entre ambos en complemento a dos (la llevada, en caso de que exista, no ha de tenerse en cuenta).

3. ¿Qué es la representación BCD?

   Decimal codificado en binario (Binary Coded Decimal), es la representación de un dígito decimal mediante 4 digitos binarios (de 0 a 9), las combinaciones binarias de 10 a 15 no existen en esta representación.

4. L a notación científica permite representar de manera aproximada números muy grandes o muy pequeños con un número de dígitos fijos ¿cómo se llama la representación en base dos para computadoras?

   Coma flotante.

5. ¿Qué es el bit de paridad?

   Es un bit que se añade a un código haciendo que el número de '1's total sea par o impar según se elija paridad par o impar. Esto permite detectar errores de un bit.

6. ¿Qué es la distancia Hamming de un código?¿Cuál es la distancia necesaria para detectar y corregir errores de dos bits?

   Cuando a un conjunto de datos se le añaden unos bit de paridad siguiendo una cierta lógica, el número total de bits se incrementa, no así el número de combinaciones válidas. La distancia entre dos palabras de código válidas es el número de bits que hay que cambiar para pasar de una palabra de código válida a otra. La distancia mínima determina la distancia Hamming del código. Para detectar errores de 'd' bits necesitamos una distancia (d+1), para corregirlos la distancia ha de ser de (2d+1). Es decir, para detectar errores de dos bits

necesitamos un código con una distancia Hamming de 3; si además queremos poder corregirlos la distancia Hamming ha de ser de 5.

7. ¿Qué es el código Hamming?

   Es un código que permite, mediante la adición de ciertos bits de redundancia/paridad, detectar y corregir errores de un bit. En caso de que haya error en dos bits, la aplicación de este método no sólo no lo corregirá sino que puede introducir un tercer error.

8. ¿Cuántos bits de redundancia/paridad son necesarios para 5 bits de datos? ¿En qué posiciones se colocan dichos bits?

   Si tenemos m datos necesitamos r bits de redundancia, cumpliéndose $(m+r)\leq 2^r-1$, por lo tanto para m=5 necesitamos r=4.

9. En recepción se calculan los bits de control, si todos son cero es que no ha habido error; pero en caso de que haya un error, ¿cómo sabemos en qué bit?

   Se calcula un bit de control por cada bit de redundancia. Cada uno de estos bits tiene un peso (C8, C4, C2, C1); el número expresado por la combinación de estos bits indica el bit en el que se ha dado el error. Por ejemplo, si tenemos 4 bits de redundancia, y por lo tanto también cuatro bits de control, y calculados sus valores obtenemos C8=0, C4=1, C2=0 y C1=1, sabemos que se ha dado un erro en el bit 5.

10. Una vez obtenida la señal que indica si un bit es o no erróneo, ¿qué operación se realiza para corregirlo (invertirlo)?

    Los bits de control se introducen en un decodificador, activando a uno sólo la salida del bit erróneo, aplicando esta señal a una entrada de una puerta XOR, y el bit correspondiente a la otra entrada de la XOR, invertiremos dicho bit solo en caso de que la señal indicase error (señal a '1'), dejándolo como estaba en caso contrario (señal a '0').

Tema 1

1. ¿Qué es la UAL (Unidad Aritmético Lógica)?

   Es la parte de la CPU en la que se realizan operaciones aritméticas y lógicas.

2. ¿De qué se compone la CPU (Unidad Central de Proceso)?

   De la Ruta de Datos y la Unidad de Control.

3. ¿Qué es la Ruta de Datos?

   El conjunto formado por la UAL y el bloque de registros.

4. ¿Qué es la palabra de control?

   El conjunto de señales de selección y control necesarias para que en la Ruta de Datos se lleve a cabo la ejecución de una instrucción .

5. La UC (Unidad de Control) genera la palabra de control ¿a partir de qué información?

   A partir de la instrucción.

6. ¿Cómo genera la palabra de control la UC cableada?

   Los bits de la instrucción se introducen en un decodificador compuesto por puertas lógicas, a cuya salida se genera la palabra de control correspondiente a dicha instrucción.

7. La UC microprogramada tiene las palabras de control almacenadas en la memoria de microprograma, ¿cómo se direcciona esta memoria?

   Si a cada instrucción le corresponde una palabra de control, esta se almacena en la dirección de memoria correspondiente al código de operación (campo que identifica la instrucción). Es decir, se utiliza el código de operación para direccionar la memoria de control.

8. En caso de tratarse de una UC de ciclo múltiple la ejecución de una instrucción necesitará, por lo general, de la ejecución de un microprograma (secuencia de microinstrucciones o palabras de control). La secuenciación de dichas microinstrucciones puede ser implícita o explícita. ¿Cómo funciona y qué supone cada uno de estos mecanismos de secuenciación?

   En el secuenciamiento explícito, cada palabra de control tiene la dirección de la siguiente así como un bit que indica si esta era la última microinstrucción del microprograma o no. Esto implica que no pocos bits de cada palabra de control, una parte considerable de la memoria de control, se utilicen para la secuenciación. Por otro lado, las microinstrucciones pueden almacenarse en cualquier dirección de memoria, no siendo necesario que se encuentren agrupadas. Así, aunque las posiciones de memoria sean más que las direccionables mediante el código de operación (supongamos un código de operación de 5 bits, que da lugar a un máximo de 32 instrucciones; si cada una requiere un microprograma de, digamos, 4 microinstrucciones, es necesaria una memoria de control de 128 posiciones), podemos seguir utilizándolo para direccionar la memoria, poniendo en las primeras posiciones la primera microinstrucción de cada uno de los posibles microprogramas.

   En el caso del secuenciamiento implícito, las microinstrucciones se almacenan en orden. Así la siguiente microinstrucción estará en la dirección actual + 1. Cada

palabra de control tendrá un bit que indique si era o no la última del microprograma, pero nada más. Sin embargo, si cada instrucción requiere varias microinstrucciones y estas tienen que estar en orden, ya no es posible direccionar la primera microinstrucción directamente con el código de operación. En este caso es imprescindible añadir una memoria ROM o un PLA (array lógico programable) que haga la función de decodificador, generando, a partir del código de operación, la dirección de la primera microinstrucción del microprograma correspondiente.

9. ¿Qué es la ejecución en canalización o pipeline?

   Consiste en dividir la ejecución de una instrucción en varias etapas, para hacer que varias instrucciones puedan ejecutarse de manera simultánea. Requiere la división en etapas de similar retardo (ya que la más lenta marcará el nuevo periodo de reloj) y la adición de registros que mantengan los datos entre etapas. La ejecución de una instrucción por si sóla tardará más que sin canalización, pero la ventaja se ve una vez llena la canalización, cuando todas las etapas puedan trabajar a la vez, reduciéndose el tiempo de ejecución por instrucción.

10. ¿Qué es el riesgo de datos? ¿Cómo evitarlo?

    El riesgo de datos surge de la ejecución en canalización: una etapa de una instrucción puede recuperar un operando que se habría modificado en la instrucción anterior, antes de que se ejecute la etapa de escritura del resultado de dicha instrucción. Así se accedería a un dato sin actualizar dando lugar a resultados inesperados.

    Hay varias formas de evitarlo, nosotros hemos visto una software y dos hardware:

    La solución software consisten en añadir retardos cuando sean necesario introduciendo en el código instrucciones NOP (not an operation), que no hacen nada, pero tardan tanto como otra instrucción. Requiere un conocimiento detallado del programa y de la canalización. Además introduce un retardo innecesario (a menos que en lugar de introducir instrucciones NOP se pueda reordenar el código y evitar el riesgo de datos).

    La solución hardware consiste en añadir la lógica que detecta el riesgo de datos, parando la ejecución de la canalización en caso necesario. Libera al programador de detectar el riesgo de datos y de introducir las instrucciones NOP, pero en cuanto al retardo la penalización es la misma que en la solución software.

    Otra solución hardware hace uso de la misma lógica que la anterior: cuando una instrucción quiere coger un operando modificado en la instrucción inmediatamente anterior, este no está aún reescrito, pero sí calculado; de manera que se añade un multiplexor a la salida de UAL, que lleva el dato de la salida directamente a la entrada cuando la lógica detecta riesgo de datos.

1. El formato de instrucción es una caja rectangular que representa los bits del código binario de la instrucción. Los campos y número de bits de estos dependen del formato de instrucción. Un campo que siempre aparece es el *código de operación* o *instrucción* ¿Qué representa ese conjunto de bits?

   Es el código máquina (combinación de ceros y unos) que identifica a dicha instrucción.

2. Un campo que puede aparecer o no es el campo de dirección, de hecho puede no haber ninguno, haber uno, dos o incluso tres campos de dirección. ¿Qué diferencia habrá en un programa que realiza una operación si se utilizan instrucciones con más o menos campos de dirección?

   Cuantos más campos de dirección más compacto, más corto, es el programa resultante.

3. ¿Qué es la dirección efectiva?

   Es la dirección en la que se encuentra el operando (el dato).

4. ¿Qué es el campo de *modo* y cómo se relaciona con el campo *dirección*?

   El campo modo es un conjunto de bits que indica un modo de direccionamiento, haciendo que la información en el campo dirección se interprete de una u otra manera.

5. Si el direccionamiento es directo, el campo dirección contiene la dirección efectiva, ¿y si el direccionamiento es indirecto?

   En ese caso, el campo dirección contiene la dirección de memoria en la que se encuentra la dirección efectiva (es decir, contiene la dirección de la dirección del dato).

6. En el microprocesador 8085 un par de registros (generalmente la pareja HL) puede utilizarse como puntero a memoria ¿qué modo de direccionamiento se está utilizando en este caso?

   Registro indirecto.

7. Los tipos de instrucciones se clasifican en tres: las de transferencia de datos (mueven datos entre registros, de memoria a registro o viceversa), de procesamiento de datos (realizan operaciones aritméticas, lógicas o de desplazamiento) y de control de secuencia. ¿Para qué sirven estas últimas?

   En general la ejecución de un programa es secuencial: cada vez que se ejecuta una instrucción se incrementa el contador de programa haciendo que apunte a la siguiente posición, donde estará la siguiente instrucción. Cuando se quiere modificar la secuencia de ejecución de las instrucciones para ejecutar una

subrutina o crear un bucle por ejemplo, son necesarias instrucciones de control de secuencia: instrucciones de salto, bifurcación, llamada y retorno de subrutina princpalmente.

8. Las instrucciones lógicas permiten modificar los bits deseados (ponerlos a '1' o a '0') manteniendo el resto como estaban. Para ello es necesario una palabra con una combinación concreta de ceros y unos. ¿Cómo se denomina dicha palabra?

   Máscara.

9. Las instrucciones PUSH y POP introducen y extraen respectivamente un elemento de la pila, pero ¿qué es la pila? ¿Y el SP (puntero de pila)?

   La pila es una zona de la memoria ubicada generalmente en la parte final de la memoria. Es una zona de tamaño dinámico que según se llena crece hacia valores más bajos. Es una pila LIFO, es decir, el último elemento en entrar es el primero en salir. El SP puede indicar la posición del último elemento (TOS, Top Of Stack), o la primera posición libre. Se actualiza automáticamente con cada instrucción PUSH y POP.

10. ¿Qué es el diagrama de flujo?

    Es un tipo de diagrama que representa de manera gráfica los pasos y la evolución de un algoritmo o proceso, permitiendo su rápida comprensión así como la detección de errores.

Tema 3

1. ¿Qué son las subrutinas?

   Es una porción de código que realiza una operación en base a unos valores dados como parámetros y que puede ser invocado desde cualquier parte del código, incluso desde sí misma, evitando el código redundante y permitiendo la división del problema así como el encapsulado del código.

2. Si durante la ejecución secuencial de un programa se encuentra una instrucción de salto, la ejecución del programa seguirá a partir de la dirección que en ella se indique. ¿Qué diferencia hay entonces entre una instrucción de salto y una llamada a subrutina?

   Si se produce un salto, se introduce en el PC (contador de programa) la nueva dirección a partir de la cual la ejecución continúa de manera secuencial. En el caso de la llamada a subrutina sin embargo, el valor del PC en el momento de la llamada se guarda en la pila, recuperándose al terminar la ejecución de la

subrutina (con la instrucción RET). Es decir, tras la ejecución del código de la subrutina el programa sigue en el punto en el que estaba; en el caso de un salto no.

3. La instrucción RET no toma parámetros, por lo que la dirección de retorno ha de guardarse en un lugar conocido; pero además sabemos que una llamada a subrutina puede darse durante la ejecución de otra subrutina. ¿Cuál es la solución para almacenar la dirección de retorno permitiendo además el anidamiento de subrutinas?

   La solución consiste en guardar las direcciones de retorno en la pila, ya que al tratarse de una pila LIFO se deshará en el orden contrario al que se creo, justo como las subrutinas anidadas. Además el SP (puntero de pila) se actualiza de manera automática, haciendo que el último elemento esté siempre disponible sin indicar ninguna dirección.

4. ¿Qué es el bloque de activación?

   Es la porción de memoria de la pila que contiene el espacio para los resultados y los parámetros de la subruina, así como la dirección de retorno y la copia del valor original del registro en el que se guardará el SP.

5. ¿Por qué es necesario el puntero al bloque de activación para acceder a los parámetros, espacio de resultados, etc. mediante un direccionamiento indexado?

   Porque la pila se modificará durante la ejecución de la subrutina, haciendo que cambie el valor del SP. Para que la distancia entre el SP y un parámetro sea fija es necesario copiar a un registo el valor que este tiene en un momento dado, para poder indexar respecto de un valor fijo durante la ejecución de la subrutina.

6. ¿Qué diferencia hay entre una llamada a subrutina y una interrupción (hardware)?

   Una llamada a subrutina forma parte del código, por lo tanto se sabe cuándo va a suceder y además es síncrona con el reloj del sistema. Una interrupción es imprevisible, puede suceder en cualquier parte del código y en cualquier momento (de manera asíncrona).

7. Un *trap* es un tipo de interrupción generada por la CPU ante una condición de error, ¿por qué esta es síncrona mientras que la interrupción hardware es asíncrona?

   Porque los traps son interrupciones causadas por la CPU ante una condición de error derivada de una instrucción, lo que hace que se den de manera síncrona. Es más, un análisis detallado del código permitiría detectarlas. No así las interrupciones hardware, que son consecuencia de una causa externa, no relacionada con el código, por lo que pueden suceder en cualquier momento.

8. Cuando se produce una llamada a subrutina es necesario guardar la dirección de retorno (el valor del PC); ante una interrupción hay que guardar además la información del bloque de registros o al menos el acumulador y los flags de estado, ¿a qué se debe esta diferencia?

   A que la llamada a subrutina, siendo parte del código, sabemos cuándo se da, en qué condiciones; sin embargo el hecho de que una interrupción pueda darse en cualquier momento implica que la siguiente instrucción pudiera depender de la situación actual (digamos por ejemplo un salto condicionado); en cuyo caso la modificación durante la ejecución de la rutina de servicio a al interrupción (RSI) de los registros o flags de estado pueden afectar a la correcta ejecución del programa a su regreso de la RSI. Para evitar que eso ocurra han de guardarse estos valores antes de ejecutar la RSI, y recuperarlos tras su ejecución.

9. ¿Qué tres formas existen para identificar la causa de la interrupción?

   1)Que las interrupciones sean multinivel (cada dispositivo una entrada), 2)que haya una línea única que indique la solicitud de interrupción por parte de alguno seguido de un sondeo (*polling*) o 3) que sean vectorizadas (además de la señal de interrupción, ponen en el bus de datos un vector que las identifica).

10. Una interrupción puede indicar la dirección de su rutina de servidio a la interrupción (RSI) o simplemente un vector que la identifique. En el segundo caso ¿dónde se encuentra la RSI?

    En este caso la CPU consultaría una tabla en la que conserva las distintas interrupciones (su identificador) y la dirección de la RSI correspondiente.

Tema 4

1. Define los métodos de acceso SAM, RAM y CAM.

   SAM: memoria de acceso secuencial. Para llegar a un dato es necesario pasar por todos los anteriores (entre la posición actual y la buscada).

   RAM: memoria de acceso aleatorio. Se accede al dato mediante la dirección.

   CAM: memoria direccionable por contenido. Se indica parte de la información buscada, se compara simultáneamente en toda la memoria, devolviendo la información así como la posición en la que está.

2. ¿Cuál de estos tiempos es mayor, el tiempo de acceso (T_A) o el tiempo de ciclo de memoria (T_C)?

   El tiempo de acceso es el tiempo que transcurre desde que se pone la dirección del dato hasta que este está disponible; el tiempo de ciclo es el tiempo mínimo que ha de transcurrir entre dos solicitudes a memoria. El tiempo de ciclo es mayor que el de acceso.

3. La gran mayoría de las memorias principales son memorias de semiconductor de acceso aleatorio. Las memorias secundarias pueden ser magnéticas, ópticas o magneto-ópticas. Los discos duros (HDD) magnéticos están siendo sustituidos por discos de estado sólido (SSD) en los portátiles, ¿qué tipo de memoria es esta? ¿Qué ventajas/inconvenientes presentan frente a los HDD?

   Son memorias de semiconductor. Estos discos pesan menos, son más rápidos y al no tener partes móviles son también más silenciosos y robustos que los discos duros magnéticos (HDD). La desventaja frente a los HDD es el precio.

4. Las memorias de solo lectura (ROM) tienen un contenido inalterable. Las memorias PROM son una variante que permiten una única escritura por parte del usuario. Compuestas por una matriz fija de puesrtas AND y una matriz programable de puertas OR ¿para qué se utiliza este tipo de memorias?

   Para prototipos.

5. Define los términos, dinámico/estático y volátil/no volátil en referencia a las características de una memoria.

   Dínamico: la información se pierde con el paso del tiempo, necesita un refresco periódico. Estático: la información se mantiene en el tiempo.

   Volátil: la información se pierde ante una falta de suministro eléctrico. No volátil: la información se mantiene en ausencia de alimentación.

6. ¿Cómo se interpreta la información del bus de direcciones para una organización 2D (o lineal) y para una organización 3D (o por coincidencia)?

   En la organización 2D los bits del bus de datos decodificados sirven para indicar una posición de memoria, una fila, en la que se encuentra el dato (los n bits del dato).

   En la organización 3D unos bits de la dirección sirven para indicar la fila; el resto la columna. El dato está en las coordenadas indicadas, un bit del dato en cada una de las n capas.

7. Explica el concepto *jerarquía de memorias*.

   En un computador coexisten diferentes memorias, las más rápidas son también más caras, y por lo tanto de menor tamaño. Las diferentes memorias forman una

jerarquía en función de los tres parámetros interrelacionados: tamaño, precio y velocidad. En el nivel 0 de la jerarquía están los registros (los más caros, rápidos y de menor capacidad de almacenamiento), en el nivel 1 la caché, en el 2 la memoria principal y en el 3 la memoria secundaria (más barata, lenta y de gran capacidad de almacenamiento).

8. ¿Qué es el principio de localidad?

   El principio de localidad (en el que se basa el concepto de memoria caché), hace referencia al hecho de que en un periodo corto de tiempo las referencias a memoria de un programa se agrupan en un entorno, haciendo uso de los datos e instrucciones contenidos en una pequeña fracción de la memoria.

9. ¿Qué es la memoria caché?

   Es una memoria de menor capacidad que la memoria principal (debido a su precio), pero más rápida. Surge del principio de localidad, ya que volcando un bloque de información de la memoria principal a la caché, durante un tiempo (habrá que ir cambiando el contenido de la caché según se necesite) del programa tendrá la información que necesita disponible en una memoria mucho más rápida que la memoria principal.

10. ¿Cómo se define la tasa de acierto (en relación con la memoria caché)?

    Tasa de acierto indica el porcentaje de veces que se accede a la caché y seencuentra la información buscada. Se calcula como el número de veces que se accede a la caché y se encuentra la información buscada, dividido por el número total de veces que se accede a la caché.

11. Si la correspondencia es directa, a cada bloque de memoria le corresponde una línea de caché, lo que hace más fácil buscarla en la caché, pero también impone una clara limitación ¿cuál?

    Puede ser que habiendo espacio libre en la caché se vea en la necesidad de sacar un bloque para poder meter otro al que casualmente le corresponde la misma línea de caché.

12. La dirección total de un dato en memoria principal se divide, en el caso de que la correspondencia sea directa, en: *etiqueta*, *línea* y *palabra* (o *byte*). El último localiza el dato dentro del bloque y la *línea* indentifica una de las posibles líneas de la caché; ¿para qué es necesaria la *etiqueta*?

Ya que la memoria principal (MP) es mayor que la caché (MC), y el tamaño de bloque es igual en ambas, a varios bloques de MP les corresponde la misma línea de MC. La etiqueta sirve para diferenciar entre los posibles bloques de MP a los que les corresponde la misma línea de MC.

13. Si el método de correspondencia es totalmente asociativo, cualquier bloque de memoria principal puede alojarse en cualquir línea de caché; en caso de que la caché esté llena, hay que decidir qué bloque se desaloja para meter en nuevo. Para ello pueden seguirse diferentes algoritmos de sustitución, define los siguientes: FIFO, LRU y LFU.

    FIFO: se extrae el que más tiempo llevé en memoria caché

    LRU: se extrae el que más tiempo lleve sin ser usado

    LFU: se extrae el que menos veces se haya referenciado

14. Si un dato es modificado en la caché, ha de actualizarse en memoria principal. A este respecto la estrategia elegida puede ser la escritura inmediata (*write through*) o la post-escritura (*write back*). Descríbelas indicando sus ventajas e inconvenientes.

    Escritura inmediata: cada vez que se modifica un dato en MC, se modifica también en MP. La pega es que genera mucho tráfico entre ambas.

    Post-escritura: se mantiene un bit de *modificado* por cada línea de MC, que se activa cuando se modifica algún dato de esa línea. Cuando se saca un bloque de caché se comprueba este bit, solo en caso de que esté activado se copia la información a MP. Evita el excesivo tráfico de la escritura inmediata, pero obliga a los dispositivos de E/S a acceder a MP a través de la MC.

15. ¿Qué es la memoria virtual?

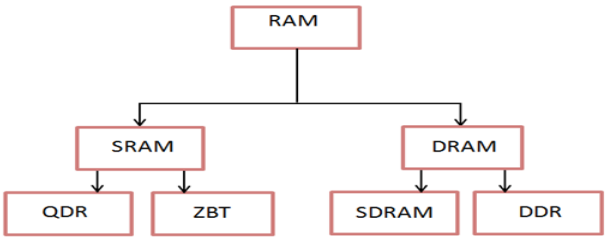
    Cuando el microprocesador puede direccionar una memoria mayor que la memoria física de la que dispone el sistema, se trabaja como si tal memoria existiera, aunque no es así, de ahí el nombre de memoria virtual. Bloques de datos (denominados *páginas*) se vuelcan desde la memoria secundaria (MS) a memoria principal (MP), y las direcciones virtuales se pasan a direcciones físicas mediante el mecanismo de *mapeado*. Cuando el dato buscado no se encuentra en MP se produce un *fallo de página*, la *página* deseada se vuelca de MS a un *marco de página* en MP, proceso de *paginación*.

16. El mapeado es el mecanismo por el que las direcciones virtuales se traducen a direcciones físicas. ¿Qué campos componen estas direcciones y en qué consiste este proceso de mapeado?

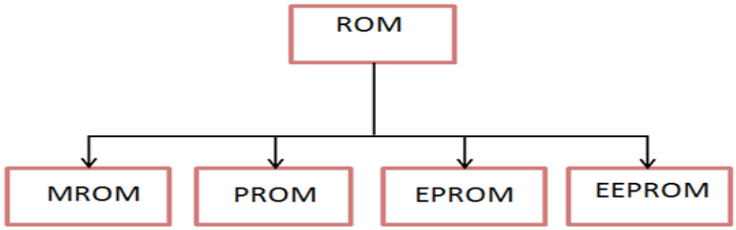
    Cada direción tiene dos partes, una de ellas común: la *dirección virtual* consta del *número de página* y del *desplazamiento dentro de la página*; la dirección virtual del *número de marco* y del desplazamiento dentro del mismo (igual al desplazamiento en la página). Una *tabla de páginas* tiene tantas entradas como

páginas, y en cada una de ellas guarda la dirección de esa página en MS (dirección fija), un *bit de presencia/ausencia* que indica si esa página está o no en MP, y en caso de que esté también el número de marco de página en el que se encuentra. En el proceso de *mapeado* se mira en la entrada correspondiente de la tabla de páginas si dicha página está en MP, en caso afirmativo se sustituyen los bits que indican la página en la dirección virtual por los del marco de página correspondiente, consiguiendo así la dirección física. (Si no se produciría un *fallo de página* y habría que traerla de MS).

17. Familia de memorias RAM.



18. Familia de memorias ROM.



19. Se pretende diseñar una memoria de 4kB con circuitos integrados de 2kB, ¿cuántos C.I.s hacen falta? ¿Cuántos chip select (CS) hay que generar? ¿Qué bits del bus de direcciones utilizarías para ello?

$$\frac{4k \times 8}{2k \times 8} = 2 \times 1 = 2 \text{ circuitos integrados}$$

2 CS, 1 bit

Para direccionar 4k hacen falta 12 bits en el bus de direcciones A[11 . . 0]; para direccionar 2k hacen falta 11 → el bit más significativo (A11) servirá para generar los CS.

20. Se pretende diseñar una memoria de 4kB con circuitos integrados de 2kx4, ¿cuántos C.I.s hacen falta? ¿Cuántos chip select (CS) hay que generar? ¿Qué bits del bus de direcciones utilizarías para ello?

$$\frac{4k \times 8}{2k \times 4} = 2 \times 2 = 4 \text{ circuitos integrados}$$

2 CS, 1 bit

Para direccionar 4k hacen falta 12 bits en el bus de direcciones A[11 . . 0]; para direccionar 2k hacen falta 11 → el bit más significativo (A11) servirá para generar

los CS. Aunque hay 4 C.I.s como han de trabajar de dos en dos (ya que cada uno solo tiene 4 bits, la mitad de la palabra), no se necesitan más CS que en ejercicio anterior.

Tema 5

1. Pon algún ejemplo de periféricos de entrada, de salida y de entrada salida.

   Teclado (entrada), altavoz (salida), pantalla táctil (entrada/salida).

2. ¿Por qué es necesario el controlador de periférico?

   Los periféricos son diferentes entre sí en cuanto a electromecánica, tipo de información y temporización, lo que hace necesario el uso de controladores de periférico que permiten al procesador ver de forma simplificada una amplia gama de dispositivos de E/S.

3. Describe los métodos de sincronización *strobing* y *handshaking*.

   Strobing: una línea indica una petición (de lectura o escritura según la petición provenga del destino o de la fuente). Si la solicitud la hace la fuente, pone los datos, espera un rato y supone que el destino los ha leído (no se comprueba). Si la solicitud la hace el destino, espera un rato y lee lo que haya en el bus de datos (no se tiene certeza de que la fuente haya puesto los datos solicitados).

   Handshaking: tiene dos líneas que permite un diálogo, una confirmación por parte del otro interlocutor, así se evitan los problemas de strobing. Se hace la solicitud y se espera a la confirmación del otro (un mecanismo de time-out evita que se quede indefinidamente a la espera si hubiera algún problema y el otro no contestase).

4. Para la transferencia de datos entre los dispositivos de E/S y memoria puede utilizarse la E/S programada (sin interrupciones) o la E/S mediante interrupciones. ¿Qué ventaja supone esta segunda opción?

   Libera a la CPU de estar comprobando el estado del dispositivo. La CPU realiza una solicitud y sigue con otra cosa; cuando el dispositivo esté listo avisará mediante una interrupción.

5. Describe el denominado Acceso Directo a Memoria (DMA).

   El acceso directo a memoria permite liberar a la CPU de la transferencia entre dispositivos de E/S y memoria. Ante una solicitud de transferencia de información, el controlador de DMA solicita el control de los buses, la CPU se los concede

(durante este tiempo la CPU puede seguir realizando tareas, pero ha de retrasar aquellas que requieran el uso de los buses).

6. ¿Cómo sabe la CPU si la transferencia de datos mediante DMA se ha completado correctamente?

   Al terminar la transferencia, el controlador de DMA avisa a la CPU mediante una interrupción y esta comprueba, mirando el estado de los registros del controlador, si la transferencia se ha realizado correctamente.

7. Describe el funcionamiento de un monitor de cristal líquido (LCD).

   Unas moléculas de cristal líquido se colocan entre dos polarizadores cruzados. Si no hay tensión aplicada la moléculas están reviradas, haciendo girar la onda 90º; sin embargo estas moléculas se ordenan al aplicar una tensión, lo que impide que la onda gire, y es por tanto bloqueada por el segundo polarizador. Si la tensión es máxima no pasará nada, si es nula pasará todo y tensiones intermedias harán que pase solo parte de la onda.

8. Un disco duro contiene varios discos o platos, que se dividen en pistas y sectores. ¿Cómo se llaman los tiempos medios para acceder a cada uno de ellos?

   El tiempo medio que requiere la cabeza para desplazarse hasta la pista es tiempo medio de búsqueda (*average seek time*); el tiempo medio de rotación para llegar al sector se denomina latencia rotacional media (*average rotation latency*).

9. ¿Cómo se determina qué tecla se ha pulsado en un teclado?

   El teclado está compuesto internamente por una matriz de filas y columnas, el microprocesador aplica una tensión en una de las filas de dicha matriz cada vez y simultáneamente inspecciona todas las columnas mediante un multiplexor. Si alguna tecla ha sido pulsada, la tensión de la fila se propagar por la columna en la intersección con la tecla, permitiendo al microprocesador determinar las coordenadas (fila y columna) y por lo tanto la tecla pulsada.

10. El puerto paralelo tiene un conector de 25 pines (DB25) por el que se transmiten datos en paralelo, ¿para la conexión de qué dispositivo era mayormente utilizado (aunque hoy en día está siendo sustituido por el USB)?

    Para la impresora, es por ello que se conoce también como puerto de impresora.

1. Los buses son líneas que permiten la transferencia de información. Las prestaciones de estos pueden disminuir si se le conectan muchos dispositivos, ¿cuál es la solución a este problema?

   Añadir más buses. Crear una jerarquía de buses.

2. ¿Qué diferencia hay entre una arquitectura tradicional de bus y una arquitectura de altas prestaciones?

   En una arquitectura tradicional se añade un bus de expansión del que cuelgan todos los dispositivos; en una arquitectura de altas prestaciones se añade un bus de alta velocidad para los dispositivos más rápidos, el resto se conectan al bus de expansión.

3. ¿Por qué es necesario el arbitraje entre dispositivos que quieren acceder a un bus?

   Al ser el bus un medio compartido, puede darse que varios dispositivos quieran hacer uso de él al mismo tiempo. El arbitraje es necesario para gestionar quién accede en cada momento.

4. ¿En qué se diferencian el arbitraje centralizado y el distribuido?

   En el arbitraje centralizado, un único dispositivo hace de árbitro, gestionando las asignaciones de bus. En el distribuido cada uno de los dispositivos puede a su vez ser árbitro y los propios dispositivos acuerdan quién lo usa en cada instante.

5. ¿Qué diferencia hay entre un bus dedicado y uno multiplexado?

   Un bus dedicado transfiere direcciones, datos o bus señales de control. En uno multiplexado el mismo conjunto de líneas eléctricas sirven para direcciones y datos (multiplexadas en el tiempo).

6. ¿Y entre uno síncrono y uno asíncrono?

   El bus síncrono trabaja de manera síncrona con el reloj del sistema (los valores de los buses se leen en los flancos de reloj). En un bus asíncrono existen señales que indican la disponibilidad de la información contenida en el bus, haciendo que funcione a la velocidad que los dispositivos puedan.

7. ¿Y entre uno serie y uno paralelo?

   En un bus serie los datos se envían bit a bit, en serie, uno detrás de otro, mientras que en uno paralelo hay una línea por cada bit de información. Así si los datos van en bytes, por ejemplo, un bus serie enviaría los ocho bits uno detrás del otro por una única línea, mientras que en el bus serie habría ocho líneas que permitirían enviar los ocho datos a la vez.

8. El método de arbitraje puede ser centralizado o distribuido, y para cada uno de ellos, la estrategia puede ser *Daisy-Chain*, *encuesta* o *petición independiente*. Describe brevemente estas tres estrategias de arbitraje.

   Daisy-Chain: ante una petición se sondean todos los dispositivos uno a uno.

   Encuesta: cada dispositivo tiene un número que lo identifica, ante una solicitu de bus se inicia una cuenta hasta que el solicitante detecta su identificador.

   Petición independiente: cada dispositivo tiene una línea de petición de bus, por lo que no es necesario hacer nada para identificar quién solicita el bus.

9. Si en un bus síncrono se conecta un dispositivo más lento que los demás, una solución es añadir una señal (WAIT) que permita solicitar otro ciclo de reloj cuando sea necesario, convirtiendo así el bus en un bus semisíncrono.
10. El protocolo PCI define un bus AD para direcciones (Address) y datos (Data), es por lo tanto un bus multiplexado.
11. El arbitraje en el bus PCI es centralizado.
12. ¿Cuáles son las ideas iniciales de diseño del bus USB?

    La simplicidad y el bajo costo.

13. ¿El bus USB permite una comunicación serie o paralela?

    Serie. El bus USB sólo tiene 4 línea (5 en algunos modelos): dos de alimentación y otros dos para la transferencia de datos (de forma diferencial).

14. El bus USB se configura con una topología de árbol, ya que a cada concentrador pueden conectarse varios dispositivos de E/S u otros concentradores. ¿Cómo direcciona los datos de la CPU a un dispositivo en concreto?

    El concentrador no direcciona, repite los datos a todos los dispositivos. Como en los datos va el identificador, sólo el aludido responde.

15. ¿Cómo puede un dispositivo final comunicarse con otro?

    No puede hacerlo directamente, siempre a través del concentrador raíz.