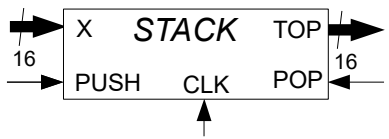
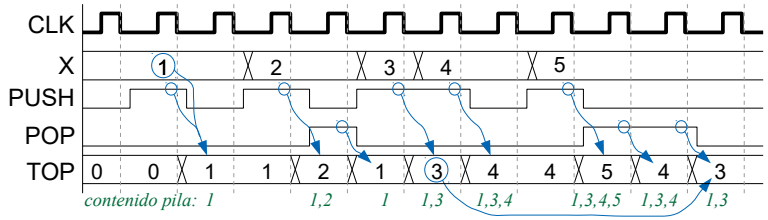
Pregunta 1



El circuito STACK de la figura es una pila de hasta 8 números. La salida TOP de STACK muestra el tope de la pila. Cuando la entrada PUSH se coloca en 1 se apila la entrada X en el tope de la pila. Cuando la entrada POP se pone en 1 se desapila un elemento. Si PUSH y POP permanencen en 0, la pila conserva su contenido. El siguiente diagrama de tiempo muestra un ejemplo de uso. Las operaciones ocurren en este orden: *push 1, push 2, pop, push 3, push 4, push 5, pop, pop*.



Implememente STACK usando el banco de registros de M16. Por simplicidad considere que al inicio todos los registros y flip-flops data valen 0 y por lo tanto la pila está vacía. No se preocupe por el desborde de la pila o una operación POP con la pila vacía. No hay operaciones PUSH y POP simultáneas.

Ayuda: Use un registro SP de 3 bits como puntero a la pila, es decir almacena el número del registro que contiene el tope de la pila. Inicialmente SP contiene 0. En una operación PUSH incremente SP en 1 y almacene X en el registro indicado por SP. En una operación POP decremente SP en 1. No necesitará usar la entrada *s2* del banco de registros ni la salida *Rs2*.

Pregunta 2

Parte a. (3 puntos) Diseñe una interfaz para M16 que conecte el circuito STACK de la pregunta 1 con el bus de M16. Su interfaz debe incluir un puerto de entrada/salida en la dirección *0xe000* para operar STACK. La escritura de un valor *x* en este puerto se debe traducir en la operación PUSH, es decir activar la entrada PUSH y dirigir el valor *x* a la entrada X de STACK. Al leer el mismo puerto se debe entregar el tope de la pila y desapilar ese elemento, es decir dirigir la salida TOP de STACK al bus de datos de M16 y activar la entrada POP para desapilarlo.

Parte b. (1,5 puntos) Programe en C las funciones *push* y *pop* con los encabezados del cuadro de al lado. La función *push* debe usar el puerto de salida de la parte *a* para apilar *x* en el circuito STACK. La función *pop* debe usar el puerto de entrada para entregar el tope de la pila en el circuito STACK y desapilarlo.

```
void push(int x);
int pop( );
```

Parte c. (1,5 puntos) Reprograme las funciones *push* y *pop* en assembler de M16.

Las preguntas 1, 2.*a*, 2.*b* y 2.*c son* independientes entre sí, es decir Ud. puede responder cualquiera de ellas sin haber respondido las otras.

Pregunta 3

Parte i.- (1,5 puntos) Simplifique la siguiente fórmula algebraica usando un mapa de Karnaugh: $a\bar{b}\bar{c}+abc+\bar{a}\bar{b}c+ab\bar{c}+\bar{a}b\bar{c}$

No necesita justificar con algebra de boole las simplificaciones.

Parte ii.- (3 puntos) Considere el diseño de M16. El registro de instrucción IR tiene cargada la instrucción *OR R3, -3, R5* (R5 es el registro de destino). Evalúe de manera independiente si cada una de estas 5 transferencias entre registros se puede realizar en *un solo ciclo del reloj*:

a) R5 ← R3 + (-3) b) PC ← (-3)	c) AR ← T d) R3 ← R5 + 2	e) R5 ← R5 & 2

Si la transferencia se puede realizar indique cuáles son la señales de control requeridas para llevarla a cabo. Si no se puede realizar, explique por qué.

Parte iii.- (1,5 puntos) La figura muestra un extracto del contenido de un *cache* de 4 KB (2^{12} bytes) de 1 grado de asociatividad con 256 líneas de 16 bytes. El computador posee un bus de direcciones de 16 bits. Por ejemplo en la línea 0f (en hexadecimal) se almacena la línea de memoria que tiene como etiqueta 20f (es decir, la línea que va de la dirección 20f0 a la dirección 20ff).

línea cache	*etiqueta*	*contenido*
c9	6c9	
0f	20f	
8b	78b	

Un programa accede a las siguientes direcciones de memoria: 20f8, 48b4, 90f0, 20f0, 6c90, 90f8, 30f0. Indique qué accesos a la memoria son aciertos en el cache, cuales son desaciertos y rehaga la figura mostrando las etiquetas finales del cache.