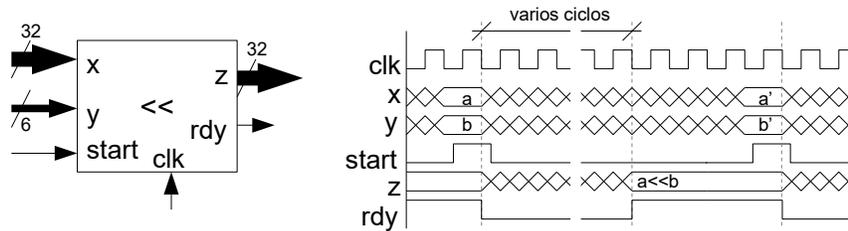


Pregunta 1

Parte a.- (4 puntos) Se desea implementar un desplazador (*shifter*) que utilice menos transistores que el desplazador en cascada visto en cátedra. Haga un diseño de desplazador a la izquierda usando un circuito serial que desplace de a un bit por ciclo del reloj. Utilice componentes modulares como sumadores, restadores, multiplexores, registros, **desplazador en 1 bit**, etc. (pero *no* el desplazador que es capaz de desplazar de a varios bits en un solo ciclo).



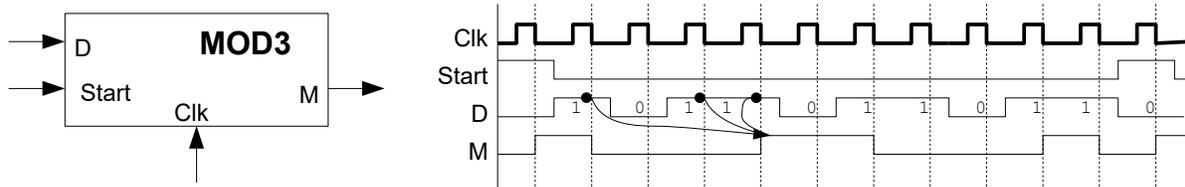
Parte b.- (2 puntos) Simplifique la siguiente fórmula booleana a su mínima expresión como suma de productos. Justifique su resultado recurriendo específicamente a las 2 reglas de simplificación de fórmulas vistas en clase de cátedra (como en la tarea 1).

$$\bar{x}\bar{y}\bar{z} + x\bar{y}\bar{z} + x y z + \bar{x} y \bar{z}$$

Ayuda: Puede usar mapas de Karnaugh para encontrar las simplificaciones convenientes. Pero lo relevante en esta pregunta es justificar con las 2 reglas de simplificación. Sin esa justificación no tendrá puntaje.

Pregunta 2

I. (3 puntos) El circuito secuencial *MOD3* debe indicar en la salida *M* cuando la cantidad de 1s vistos en su entrada *D* es múltiplo de 3 (sin contar el ciclo actual). *Start* se pone en 1 para indicar el inicio de una secuencia de bits. Diseñe un diagrama de estados que especifique completamente el comportamiento de *MOD3* y que funcione de acuerdo al diagrama de tiempo de la figura. *No implemente el circuito.*



Ayuda: Se puede resolver con solo 3 estados. Un estado para cuando la cantidad de 1s módulo 3 es 0, otro estado para cuando la cantidad de 1s módulo 3 es 1 y el último para cuando la cantidad de 1s módulo 3 es 2.

II. (3 puntos) La función de abajo está en assembler Risc-V y recibe 2 parámetros en *a0* y *a1*. Escriba el programa *equivalente* a *incognito* en C sin usar la instrucción **goto** de C. Preocúpese de *reproducir* en C todos los aspectos del programa original en assembler, en particular el tipo de los parámetros y el valor retornado.

```

incognito:
    addi    a2, a0, 4
    lw     a3, 0(a0)
    j      L2
L1:
    lw     a4, 0(a2)
    addi   a2, a2, 4
    beq    a3, a4, L2
    mv     a3, a4
    sw     a3, 0(a0)
    addi   a0, a0, 4
L2:
    blt    a2, a1, L1
    ret
    
```