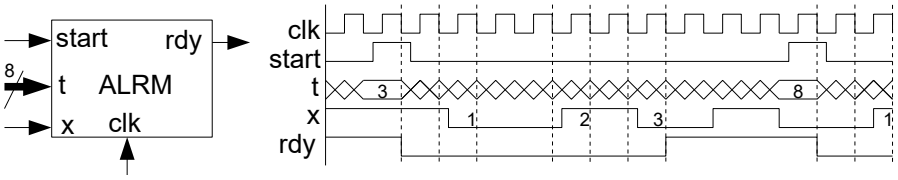
CC4301 Arquitectura de Computadores – Control
Primavera 2020 – Prof.: Luis Mateu

Pregunta 1



Use diseño modular para implementar el circuito *ALRM* de la figura. Este circuito avisa en en la salida *rdy* cuando se detectaron *t* transiciones de 1 a 0 o de 0 a 1 en la línea *x*. Funciona de la siguiente manera. Cuando *start* se pone en 1, la entrada *t* es el número de transiciones que se deben detectar. En el diagrama de tiempo *t* es 3 la primera vez que *start* se pone en 1. En tal caso la línea *rdy* pasa a 0 hasta que se completen 3 transiciones. Una transición ocurre cuando el valor de la entrada *x* cambia de un pulso de bajada del reloj al siguiente pulso de bajada. La primera transición en el diagrama está señalada por el 1, porque previamente *x* era 1 en el pulso de bajada del reloj, pero en el siguiente pulso de bajada fue 0. El número 2 señala la transición de 0 a 1, y el número 3 es la última transición (de 1 a 0). Cuando se detectan *t* transiciones, *rdy* pasa a 1, y se mantiene así hasta que *start* se pone en 1 nuevamente para contar otras 8 transiciones en el ejemplo.

Metodología: Use un registro para almacenar la cantidad de transiciones que quedan por detectar. Se inicializa en *t* cuando *start* se pone en 1 y se decrementa en 1 cada vez que se detecta una transición. El conteo termina cuando ese registro llega a 0. Use un circuito secuencial de 3 estados para controlar el conteo de transiciones. El estado A indica que el conteo de transiciones terminó y por lo tanto *rdy* está en 1. El estado B significa que el conteo está activo y el último bit visto en *x* fue 1. El estado C señala que el conteo está activo y que el último bit visto en *x* fue 0. En los estados B y C normalmente *rdy* está en 0, excepto cuando el registro de transiciones llegó a 0. El circuito secuencial tiene como entradas *start*, *x* y una línea interna que indica si el registro de transiciones es 0. Las salidas son *rdy* y una línea interna que indica si decrementar o no el registro de transiciones. Diseñe el diagrama de estados del circuito secuencial, especificando el orden en que anota las entradas y salidas. ***No se pide implementarlo*** por razones de tiempo. Sí debe implementar el resto del circuito modular. Note que cuando el registro llega a 0, no importa si continúa decrementándolo. De todas formas lo que mantiene la línea *rdy* en 1 es el estado A del circuito secuencial.

Pregunta 2

Parte a.- (2 puntos) Simplifique la siguiente fórmula booleana a su mínima expresión como suma de productos. Justifique su resultado usando álgebra de boole.

$$\overline{x}\,\overline{y}\,\overline{z}\,\overline{w} + xyzw + \overline{x}\,\overline{y}\,z\,\overline{w} + x\,\overline{y}\,z\,\overline{w} + \overline{x}\,\overline{y}\,\overline{z}\,w + \overline{x}\,\overline{y}\,z\,w$$

Ayuda: use mapas de Karnaugh para encontrar las simplificaciones convenientes.

Parte b.- (4 puntos) El siguiente es un programa en assembler x86. Escriba el programa *equivalente* 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 valor retornado.

```
inc: # int *inc(int *a);
   movl   4(%esp), %eax
   movl   %eax, %edx
.L4:
   movl   (%eax), %ecx
   movl   %ecx, (%edx)
   cmpl   $0, %ecx
   je     .L2
```

```
.L3:
   addl   $4, %edx
   cmpl   (%edx), %ecx
   je     .L3
   jmp    .L4
.L2:
   ret
```