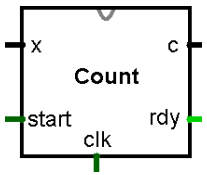
CC3301 Programación de software de sistemas – Tarea 6 – Primavera 2024 – Profesores Mateu/Ibarra/Urrea



Parte a. (3 puntos) El circuito *Count* de la derecha calcula y entrega en su salida *c* la cantidad de cifras hexadecimales distintas en su entrada *x*, excluyendo la cifra 0. Por ejemplo si *x* es 0x0**3**10f1**1f** el resultado *c* debe ser 3. El cálculo comienza cuando se detecta que *start* es 1 justo en el momento en que la entrada *clk* pasa de 1 a 0. En ese instante la salida *rdy* debe ir a 0 y permanecer en 0 mientras se realiza el cálculo durante varios ciclos del reloj. Una vez terminado el cálculo, el resultado debe aparecer en *c* y *rdy* debe ir a 1. Debe mantener las salidas *c* y *rdy* constantes hasta que *start* sea 1 nuevamente cuando *clk* pasa de 1 a 0.

El circuito se encuentra parcialmente construido en el módulo *Count* del archivo *count.circ*. Complete el circuito siguiendo la ayuda que se encuentra en el mismo circuito. **Pruebe que su solución funciona** correctamente seleccionando el módulo *Test-Count* y simulando el circuito con *control-r* y *control-k*. Solo obtendrá el puntaje de esta parte si se enciende la luz verde.

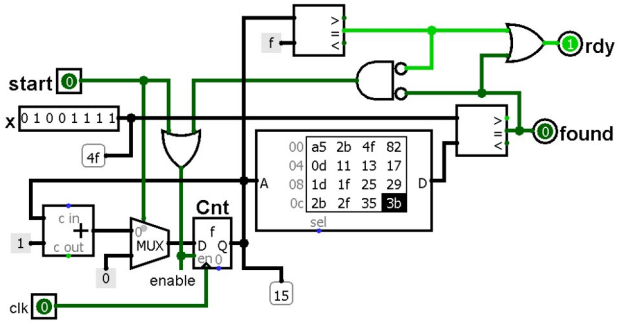
Como resultado de esta pregunta Ud. debe entregar el circuito *count.circ* en donde completó la implementación del módulo *Count*. No puede modificar los demás módulos. Puede regular la velocidad de la simulación en *Simular → Seleccionar Frecuencia del reloj*.

Ayuda: Para entender cómo resolver esta parte y cómo verificar que funciona correctamente, vea este video en donde explico la solución de un problema similar de un control de arquitectura de computadores. La solución del problema que sale en el video está en el circuito *max1s.circ* de los archivos adjuntos. También le será de mucha utilidad ver los videos de las cátedras del jueves 10 y martes 15 la clase auxiliar del viernes 11 de octubre, todas sobre circuitos.

Parte b. (1 punto) Considere el circuito de arriba en la siguiente columna. Complete la siguiente tabla con los valores que toma cada señal hasta el ciclo 6 del reloj:

ciclo	start	x (hex)	Cnt (hex)	enable	rdy	found
1	0	4f	f	0	1	0
2	**1**	4f				
3	**0**	4f				
4	0	4f				
5	0	4f				
6	0	4f				

Parte c. (1 punto) La tabla de arriba en la siguiente columna muestra un extracto del estado inicial de un *caché* de 4 KB (2^{12} bytes) de un grado de asociatividad con 256 líneas de 16 bytes. Por ejemplo en la línea 6b del caché (en hexadecimal) se almacena la línea de memoria que tiene como etiqueta 86b (es decir, la línea que va de la dirección 86b0 a la dirección 86bf).



Circuito

línea cache	*etiqueta*	*contenido*
9c	79c	
6b	86b	
34	a34	

Tabla

Un programa accede a las siguientes direcciones de memoria (en hexadecimal): 7**9c**0, 8**6b**4, a**34**c, 5**6b**0, 7**9c**4, a**34**8, d**34**0, 7**9c**8, 8**6b**8. Indique qué accesos a la memoria son aciertos en el caché, cuáles son desaciertos y rehaga la figura mostrando el estado final del cache. Observe que el caché no está vacío inicialmente y por lo tanto el acceso 7**9c**0 es un acierto. Base su respuesta en el ejemplo que aparece al principio de este video.

Parte d. (1 punto) La tabla de la derecha muestra las instrucciones Risc-V ejecutadas por un programa. Haga un diagrama que muestre el ciclo en que se ejecuta cada etapa de las instrucciones, considerando una arquitectura (i) en pipeline con etapas *fetch*, *decode* y *execute*, y (ii) superescalar, con 2 pipelines con las mismas etapas de (i). Suponga que el salto en E no ocurre y el salto en F sí ocurre (y no hay ningún tipo de predicción de saltos). Base sus respuestas en los ejemplos que aparecen a partir del minuto 57:13 de este video.

```
A add  a1,a2,a3
B andi t4,a1,7
C addi t5,t4,1
D add  t6,t4,a4
E blt  a1,t4,I
F beq  t5,a1,P
G add  ...
H sub  ...
I xor  ...
J andi ...
...
P addi a3,a2,1
Q sub  t7,a4,t5
```

Instrucciones

Baje *t6.zip* de U-cursos y descomprímalo. Contiene el circuito *count.circ*, que Ud. debe completar, y el circuito *max1s.circ* con la solución del ejemplo del video.

Entrega

Entregue por medio de U-cursos un archivo *arq.zip* con el circuito *count.circ* modificado con su solución de la parte *a*, y las soluciones de las partes *b*, *c* y *d* en el formato de su elección (por ejemplo foto legible de su solución en papel). La parte *a* es binaria, se otorga 0 o todo el puntaje, pero en las demás partes se otorga puntaje de acuerdo a lo logrado. Se descontará medio punto por día de atraso (excluyendo sábados, domingos, festivos o recesos).