DISEÑO DE SIMULADORES DIGITALES USANDO SDLC++

V. G. RUIZ
Departamento de Arquitectura de Computadores y Electrónica. Universidad de Almería. 04210, Almería. España. Tel: +34 950 215711, Fax: +34 950 215486, E-mail: vruiz@gogh.ualm.es

SDLC++ (Simulador Digital en Lenguaje C++) es un programa escrito en lenguaje C++ pensado para simular circuitos electrónicos digitales, descritos jerárquicamente y que en última instancia pueden entenderse como redes de puertas lógicas. Permite describir y simular tanto el comportamiento funcional como temporal de circuitos de muy alta complejidad (por ejemplo, microprocesadores) de forma sencilla y eficiente.

1. Introducción

A la hora de simular en una computadora un sistema digital, existen fundamentalmente dos alternativas: (1) diseñar un programa que exprese su comportamiento lógico a lo largo del tiempo lo que no implica un conocimiento previo de la arquitectura y (2) describir el sistema expresando su estructura en función de elementos funcionales más simples. La principal ventaja de la segunda aproximación radica en que no es necesario conocer el funcionamiento del circuito simulado.

La mayor dificultad a la hora de diseñar modelos de simulación reales radica en que los circuitos son intrínsecamente paralelos. Si los analizamos a nivel de puertas lógicas, en cualquier instante todas ellas están funcionando simultáneamente. Por esta razón, muchos lenguajes de simulación son concurrentes (VHDL, Verilog). Con ellos, los sistemas digitales son abordados como un conjunto de tareas o procesos que se ejecutan en paralelo. El inconveniente de estos lenguajes es que son pesados de ejecutar sobre máquinas monoprocesadoras y el sistema operativo debe soportarlos (típicamente como procesos o hilos). Si tenemos en cuenta que en algunas simulaciones, aunque tratándose de procesos muy elementales (puertas lógicas), existen cientos de miles o millones de ellos, otro tipo de alternativas que necesiten menos requerimientos computacionales pueden ser interesantes.

SDLC++ ejecuta programas secuenciales escritos en C++ [3]. Todos los elementos digitales simulados se implementan usando objetos. Se construye una lista con todos ellos y se ejecutan en serie, dedicando una porción de la CPU a cada uno de ellos de forma periódica. De esta forma, la sensación es que todos se ejecutan en paralelo. No existen por tanto cambios de contexto entre procesos o hilos lo que junto con la alta eficiencia de los programas escritos en C++ hace posible simular circuitos de muy alta complejidad.

2. Estructura de SDLC++

Desde un punto de vista didáctico y práctico, el diseño y simulación de un sistema digital a partir de otros más sencillos es una gran ventaja ya que permite ver a los elementos constituyentes como cajas negras. En este sentido, SDLC++ es una biblioteca de circuitos digitales diseñados a partir de otros más elementales. La descripción jerárquica se realiza fácilmente usando objetos.

El núcleo de SDLC++ es un fichero escrito en C++ (compilable) que contiene los objetos que simulan las puertas lógicas. El modelo usado permite conocer el comportamiento lógico y los tiempos de establecimiento y propagación. Las cantidades de CPU y memoria consumidas son muy pequeñas lo que garantiza la ejecución de muchos objetos de este tipo en un corto espacio de tiempo sobre computadoras con recursos modestos.

La siguiente capa de circuitos se monta utilizando únicamente las puertas y su descripción es independiente del modelo usado para éstas, por lo que si éste es modificado por ejemplo para contemplar efectos de fan-out, el resto de circuitos no tienen que ser modificados. Así, se han diseñado (des)codificadores, (des)multiplexores, cerrojos, registros, contadores, memorias, ficheros de registros, ALUs, [1] hasta llegar a una implementación de la versión reducida del microprocesador MIPS R2000 (versión multiciclo) descrito en [2]. Con él es posible ejecutar programas escritos en código máquina MIPS, conocer los tiempos de ejecución y encontrar el estado de cualquier línea del microprocesador en cualquier instante de tiempo. El software SDLC++ puede descargarse desde http://gogh.ualm.es/vruiz/docencia/sdlc++.tar.gz.

3. Simulación de un contador asíncrono de 8 bits

A continuación presentamos un ejemplo sencillo de simulación (ver figura 1). Se muestra el esquema lógico del circuito y su descripción en SDLC++. Como puede apreciarse, se trata de definir un conjunto de clases que contienen una serie de objetos privados que declaran los elementos funcionales más simples de los que consta cada circuito y una única función `run()`, que lanza la simulación.

Tras la declaración del circuito que se desea simular, se escribe un programa en C++ (ver figura 2) que declara un objeto de la clase diseñada y que lo ejecuta de forma iterativa (en el ejemplo son 2500 iteraciones). Después de compilar y ejecutar, se genera la salida ASCII mostrada donde podemos ver la salida del reloj y del contador a lo largo de tiempo. Puede verse cómo las transiciones de alto (255) a bajo (0) provocan la conmutación de los flip-flops JK del contador, que son activos en los flancos de bajada. Para apreciar además con más facilidad qué está ocurriendo a lo largo del tiempo, se adjunta también un cronograma que es simplemente el contenido del fichero ASCII representado con el programa gnuplot. Evidentemente, la salida del simulador puede ser cualquier otra, sólo tenemos que acomodarla a nuestras necesidades.

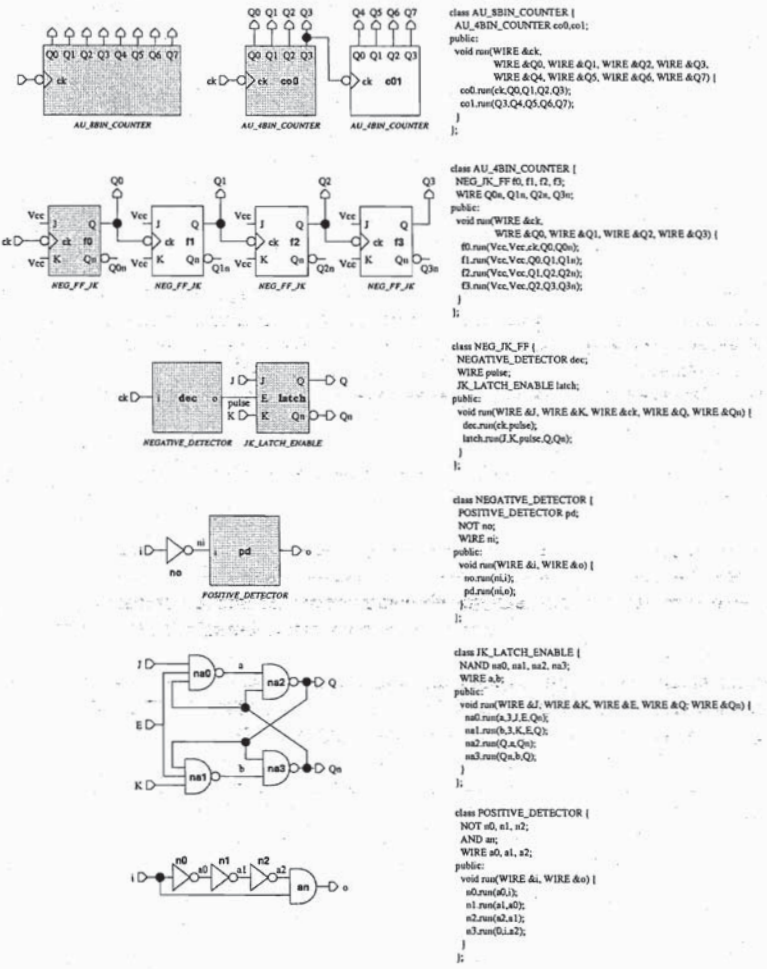

Figura 1: Implementación de un contador binario asíncrono de 8 bits.

```
void main() {
 int iters=2500;
 WIRE ck, Q0, Q1, Q2, Q3, Q4, Q5, Q6, Q7;
 AU_8BIN_COUNTER co;
 CLOCK clock(7);
 while(iters--) {
  clock.run(iter,ck);
  co.run(ck,Q0,Q1,Q2,Q3,Q4,Q5,Q6,Q7);
  printf("%3d %3d %3d %3d %3d %3d %3d %3d %3d\n",
         ck,Q0,Q1,Q2,Q3,Q4,Q5,Q6,Q7);
 }
}
```

```
255   0   0 255   0   0   0   0 255
255   0 255 255 255   0 255 255 255
255   0 255 255 255   0 255 255 255
255   0 255   0 255   0 255 255   0
255   0 255   0 255   0 255 255   0
255   0 255   0 255   0 255 255   0
255   0 255   0 255   0 255 255   0
  0   0 255   0 255   0 255 255   0
  0   0 255   0 255   0 255 255   0
  0   0 255   0   0   0 255 255   0
  0 255 255   0   0   0 255 255   0
  0 255 255   0   0   0 255 255   0
  0 255 255   0   0 255 255 255   0
```

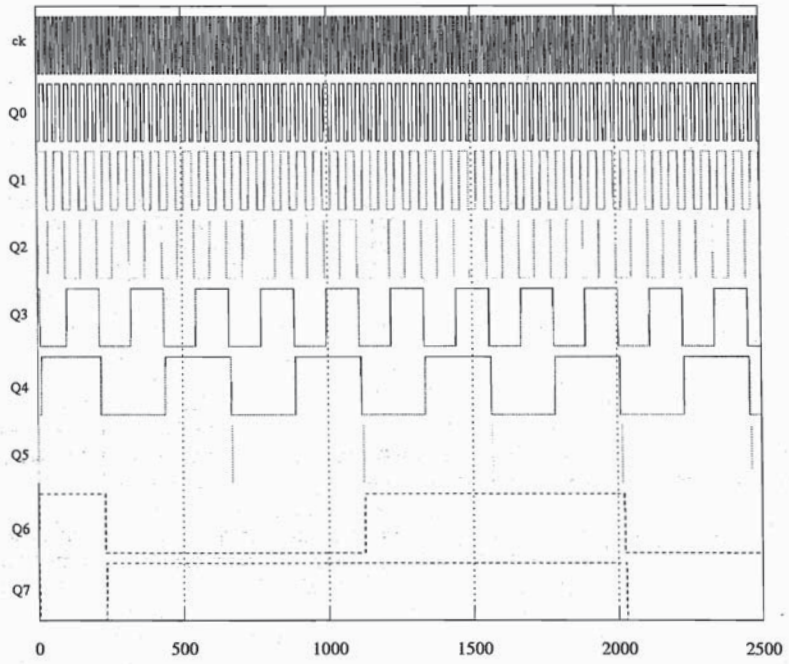

Figura 2: Simulación del contador.

Referencias

[1] Floyd, T. L. *Principios Digitales, 6ª ed.* Prentice-Hall. 1997.

[2] Patterson, D. A. y Hennessy, J. L. *Organización y Diseño de Computadores. La Interfaz Hardware/Software.* McGraw-Hill. 1995.

[3] Stroustrup, B. *El Lenguaje de Programación C++.* Adison Wesley. 1997.