Estado del arte en las metodologías de diseño de circuitos digitales asíncronos

Rubén D. Nieto-Londoño*, Álvaro Bernal-Noreña*[§]

** Escuela de Ingeniería Eléctrica y Electrónica-Facultad de Ingeniería*
Universidad del Valle, Cali, Colombia.
[§] *e-mail: alvaro@univalle.edu.co*

(Recibido: Septiembre 30 de 2005 - Aceptado: Diciembre 2 de 2005)

Resumen

En este artículo se presenta una síntesis de las principales características de las distintas clases de circuitos digitales asíncronos. Se describen las técnicas utilizadas en el estilo de diseño digital asíncrono. También se hace una reseña del estado actual del diseño asíncrono en la industria y de las herramientas de software utilizadas en las distintas fases de diseño.

Palabras Clave: Diseño digital asíncrono, modelos de retardo acotado, modelos de retardo no acotado, Circuitos de Huffman, Micropipelines, circuitos SDI, circuitos De-Sincronizados.

Abstract

In this paper a synthesis of the main characteristics of the different classes of asynchronous digital circuits is presented. The techniques used in the style of asynchronous digital design are described. A review of both, current state of the asynchronous design in the industry and the software tools used in the different design phases, is also provided

Keywords: Digital asynchronous design, bounded delay model, unbounded delay model, Huffman circuits, micropipelines, SDI circuits, De-Synchronized circuits.

1. Introducción

El diseño asíncrono ha sido un área activa de investigación desde mediados de los años 50, cuando el foco principal eran los circuitos de relés mecánicos. En 1956, D. E. Muller y W. S. Bartky estudiaron en detalle una serie de aspectos teóricos relacionados con circuitos asíncronos; desde entonces, este campo ha tenido un número de ciclos de alto interés [1]. En años recientes ha habido un interés sin precedentes por el estilo de diseño asíncrono, tanto en el nivel académico como en el nivel industrial, lo cual se ha visto reflejado en el desarrollo de herramientas de síntesis, simulación y verificación y también en el desarrollo de circuitos integrados con una amplia variedad de aplicaciones.

Este artículo pretende despertar el interés por el

estudio del estilo de diseño digital asíncrono y de las metodologías y herramientas de diseño utilizadas en la actualidad. El artículo está organizado como sigue: la sección 2, la más extensa, presenta el marco teórico en el que se describen las características generales y los tipos de circuitos asíncronos clasificados de acuerdo con modelos de retardo. La sección 3 presenta un ejemplo particular en el que se hace una comparación de desempeño entre procesadores síncronos y asíncronos. En la sección 4 se presenta una síntesis de informes presentados por la *IST (Information Society Technologies)* sobre el estado del diseño asíncrono en la industria y sobre el estado del arte en métodos y herramientas para el diseño de sistemas VLSI digitales asíncronos. Finalmente se presentan las conclusiones de este trabajo.

2. Marco teórico

2.1. Características de los circuitos asíncronos

De acuerdo con lo planteado en [2], el diseño lógico de hoy está basado en dos aspectos: *todas las señales son binarias y el tiempo es discreto.* Asumir valores binarios sobre las señales permite usar lógica Booleana para describir y manipular construcciones lógicas. Asumir que el tiempo es discreto permite ignorar los riesgos (*glitches*) y los retardos de la realimentación que presentan los circuitos síncronos. Los circuitos asíncronos mantienen el supuesto que las señales son binarias pero remueven el supuesto que el tiempo es discreto. Lo anterior tiene varios posibles beneficios [3],[4]:

No hay retrasos de reloj (clock Skew)*:* ya que los circuitos asíncronos no tienen señal de reloj.

Bajo Consumo de Potencia: debido a que no es necesario conmutar líneas de reloj y precargar y descargar circuitos que no serán usados en alguna ruta de cálculos.

Desempeño del caso promedio en lugar del peor caso: los circuitos síncronos deben esperar a que todos los posibles cálculos se hayan llevado a cabo antes de capturar el resultado manteniendo el peor caso. Muchos circuitos asíncronos censan cuándo ha finalizado el cálculo manteniendo el caso promedio.

Facilidad de manejo de la temporización global: en microprocesadores síncronos, el reloj del sistema y por tanto el desempeño del sistema está determinado por la ruta más lenta (crítica) por lo que ésta se debe optimizar para obtener la más alta tasa de reloj. Los circuitos asíncronos operan a la velocidad de la ruta concreta de operación.

Mejor Potencial de Migración Tecnológica: en sistemas asíncronos sólo el componente más crítico influye sobre el desempeño promedio global ya que el desempeño sólo depende de la ruta activa en el instante.

Adaptación Automática a propiedades físicas: los circuitos síncronos deben asumir que se presenta la peor combinación posible de factores para determinar el reloj del sistema. En la mayoría de circuitos asíncronos se censa la finalización del cálculo y estos correrán tan rápidamente como las propiedades físicas lo permitan.

Exclusión Mutua Robusta y Manejo de entradas externas: los elementos que garantizan una correcta exclusión mutua de señales independientes y sincronización de señales externas para un reloj están sujetas a metaestabilidad. La mayoría de sistemas asíncronos pueden esperar un largo tiempo arbitrario y como no necesitan un reloj con el cual las señales se sincronicen, pueden acomodar sus entradas desde el mundo exterior.

El predominio de los sistemas síncronos causa sorpresa a pesar de todas las ventajas potenciales de los circuitos asíncronos. La razón es que el diseño asíncrono presenta varias desventajas respecto del diseño síncrono [3],[4]:

Complejidad: el paradigma del diseño con reloj tiene una simple regla fundamental; *cada etapa de procesamiento debe completar sus actividades antes que termine el periodo del reloj*. El diseño asíncrono requiere hardware extra para permitir que cada bloque realice la sincronización local para el paso de datos a otros bloques. Esta complejidad adicional resulta en circuitos más grandes y procesos de diseño más difíciles.

Punto Muerto (Deadlock): el diseño de lógica de control usando una técnica de diseño asíncrono entra en un punto muerto si se pierde un evento o si éste se introduce en forma incorrecta, por ejemplo como resultado de un ruido o radiación iónica.

***Difícil Verificación*:** para un diseño asíncrono, la verificación es difícil debido al comportamiento no determinista de los elementos de arbitración, y no es fácil detectar el punto muerto sin una exploración exhaustiva del espacio de estados.

***Testibilidad*:** las pruebas para las fallas de fabricación en sistemas asíncronos es el principal obstáculo debido al comportamiento no determinista de los elementos de arbitración.

***Falta de familiaridad de los diseñadores con las técnicas de diseño asíncrono**:* debido a que las técnicas de diseño síncronas, principalmente, han sido usadas y enseñadas ampliamente en universidades por más de dos décadas.

2.2. Metodologías de diseño asíncrono

Una forma de distinguir el amplio espectro de diseños asíncronos es entendiendo los diferentes modelos de retardo y operación. Cada circuito físico tiene un retardo inherente, sin embargo, ya que los circuitos síncronos procesan entradas en cada pulso de reloj, ellos pueden ser considerados como operadores instantáneos que calculan un nuevo resultado en cada ciclo de reloj. Por otro lado, ya que los circuitos asíncronos no tienen reloj, estos se pueden considerar como elementos de cálculo dinámico a través del tiempo, por lo tanto, un modelo de retardo no define de forma eficiente el comportamiento dinámico de un circuito asíncrono. Hay dos modelos fundamentales de retardo, el *modelo de retardo puro* y el *modelo de retardo inercial.* Un retardo puro puede retardar la propagación de una forma de onda pero no la altera. Un modelo inercial puede alterar la forma de la onda atenuando los glitches. Los retardos también se caracterizan por sus modelos de tiempo: en un *modelo de retardo fijo* se asume que el retardo es fijo; en un *modelo de retardo acotado*, un retardo puede tener cualquier valor en un intervalo de tiempo dado. En un modelo de *retardo no acotado*, un retardo puede tener cualquier valor finito [1].

Un modelo de circuito se define en términos de modelos de retardos para las líneas individuales y los componentes. Dado un modelo de circuito, es importante caracterizar la interacción del circuito con su ambiente. El circuito y el ambiente forman un sistema cerrado llamado *Circuito Completo.* Si al ambiente se le permite responder a las salidas del circuito sin restricciones de tiempo, los dos interactúan en un modo de entrada/salida. El ejemplo más común es el *modo fundamental*, donde el ambiente debe esperar para que un circuito se estabilice antes de responder a las salidas del circuito; tal requerimiento se puede ver como el tiempo de sostenimiento (*hold time*) para un latch o un flip-flop. Dados los modelos para un circuito y su ambiente, los circuitos asíncronos se pueden clasificar en una jerarquía, tal como se describe en [5] y se resume en la figura 1.

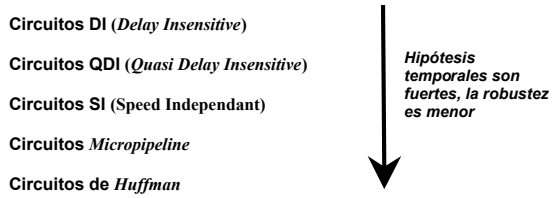

Figura 1: Clasificación Jerárquica de los Circuitos Asíncronos

Tradicionalmente han existido dos modelos importantes en el mundo de los circuitos asíncronos [6], [7]:

- **El modelo de Huffman**, en el cual el circuito se descompone en un circuito combinacional y un conjunto de hilos de realimentación. Además se asocia un elemento de retardo acotado a cada interconexión entre puertas (modelo de retardo de hilo acotado[1]).

- **El modelo de Muller**, el circuito se descompone en un conjunto de puertas arbitrariamente interconectadas y se asocia a un elemento de retardo no acotado pero finito a cada salida de puerta (modelo de retardo de puerta no acotado[2]).

1 Traducción del termino en Inglés "unbounded wire delay"
2 Traducción del término en Inglés "unbounded gate delay"
3 Traducción del término en inglés "bounded delay"

2.2.1. Modelos de Huffman o de Retardo Acotado[4]

2.2.1.1. Circuitos de Huffman en Modo Fundamental

La metodología de diseño basada en el modelo de Huffman (en honor de su inventor David A. Huffman) clásicamente produce ecuaciones lógicas que implementan la llamada *Tabla de Flujo* que es una especificación similar a las Tablas de Transición en las Máquinas de Estados. Este modelo es altamente costoso de analizar y menos fiable que otros ante variaciones en el proceso de fabricación [6]. El modelo básico de Huffman, según [8],[9] se resume a continuación:

i. ***Retardos de Realimentación***: en cada línea de realimentación se coloca un elemento de retardo, cuya salida define una variable de entrada secundaria; se supone que todas las compuertas y las demás líneas tienen un retardo de propagación igual a cero.

ii. ***Incertidumbre de retardos***: en un cambio de estado en el que toman parte dos o más variables secundarias, éstas pueden cambiar de valor en cualquier orden arbitrario.

iii. ***Modo Fundamental:*** una señal de entrada primaria puede cambiar de valor entre los niveles 0 y 1 únicamente después que el circuito haya alcanzado un estado estable.

iv. ***Cambios en una sola entrada***: se supone que las señales de entrada primarias cambian de valor de una en una.

2.2.1.2. Extensiones de Circuitos de Huffman a Modo No Fundamental

Existen también técnicas para extender el modelo de Huffman a circuitos no fundamentales que permiten nuevas transiciones de las señales antes de lo que permite el Modo Fundamental. Una alternativa que combina aspectos del mundo síncrono (simplicidad y eficiencia) con aspectos del mundo asíncrono (velocidad y modularidad), son los llamados circuitos ***auto-sincronizados*** en los cuales se genera un reloj para cada máquina de estados que es independiente del resto. Existe una lógica global de control de reloj que produce pulsos siempre y cuando el estado de alguna salida deba cambiar y que controla los elementos de memoria implicados en la realimentación. Tal técnica permite además el uso de métodos de codificación típicos de circuitos síncronos. En este sentido se han presentado diversos trabajos como los de Nowick, Yun y Dill [10],[11] que usan una forma restringida de tabla de flujo llamada máquina de estados ***"Burst Mode".*** El estilo de diseño "*Burst Mode*" se desarrolló con base en los trabajos realizados previamente en los laboratorios Hewlett Packard y la Universidad de Stanford por Davis, Stevens y Coates [1],[12]. Ladd y Birmingham [13], utilizan un sistema de comunicación en el que la validez de un dato se define en referencia a una señal de reloj local libre de riesgos. Tanto las entradas como las salidas se mantienen con el uso de registros. Rosenberger y otros [14],[15], usan un tipo especial de flip-flop con señal de terminación llamado Q-flop que asegura que el circuito se ha estabilizado cuando se produce el pulso de reloj.

2.2.2. Modelo de Muller

El trabajo de Muller [16],[17] introdujo el primer modelo formal de circuito asíncrono basándose en el concepto de *Diagramas de Transiciones de Estados (STD).* Su modelo de circuito consta de un conjunto de puertas lógicas con retardos no acotados. Cada puerta tiene asociada una expresión que describe una función lógica completamente especificada [6]. Muller mostró que el comportamiento del circuito se puede describir de forma equivalente con un tipo de autómata finito llamado Diagrama de Transiciones de Estados. Los circuitos diseñados con base al modelo de Muller se denominan *independientes de la Velocidad* (ver sección 2.1.2.5.).

Se han propuesto distintos métodos para la síntesis de circuitos con retardos no acotados de puerta basándose en los trabajos de Muller: Armstrong y otros [18] proponen el uso de un código auto-sincronizado para la codificación de los estados de entrada y salida del circuito y la utilización de un protocolo de petición/confirmación para la sincronización entre circuitos que cooperen. Un ejemplo de este tipo de código es el de doble rail

implementado con dos señales. Las metodologías más extendidas en el campo de la síntesis asíncrona son las basadas en la utilización de Redes de Petri [19] como herramientas de descripción del comportamiento de un circuito. Se destacan los Grafos de Señales [20] y los Grafos de Transición de Señales STGs [21] que son redes de Petri interpretadas en las que cada transición representa un cambio de valor en una de las señales del circuito. Así mismo destacan los Diagramas de Cambios (CDs) propuestos en [22], que son una forma similar a los STGs. Las técnicas basadas en STGs tienen actualmente gran número de seguidores. De ellos, se pueden comentar trabajos como los de Lavagno [23], Moon y otros [24].

2.2.3. Circuitos Insensibles a los Retardos (*DI: Delay Insensitive*)

Un circuito *Delay Insensitive* (*DI*) es aquel que está diseñado para operar correctamente, sean cuales sean los retardos en sus compuertas y líneas de interconexión. Este tipo de circuitos asume el modelo de retardo de línea y de compuerta no acotados. El concepto de circuito insensible a los retardos fue sugerido por Clark y Molnar en el trabajo *Macromodules* [25][4]. Los sistemas DI han sido formalizados por Udding [26] y Dill [27]. Las clases de circuitos DI que han sido construidos son limitadas, de hecho, se ha probado que la mayoría de circuitos útiles están restringidos a una clase de compuertas y operadores muy sencillos; el elemento C (del que se habla más adelante) es un ejemplo [28].

2.2.4. Circuitos Casi Insensibles a los Retardos (*QDI: Quasi-Delay Insensitive*)

Un circuito casi insensible a retardos, (*quasi-DI*) o QDI, es un circuito insensible a retardos excepto que requiere bifurcaciones isocrónicas (la diferencia del retardo entre las ramas de la bifurcación es inferior al mínimo retardo de puerta). En contraste, en un circuito DI los retardos en las diferentes ramas bifurcadas es completamente independiente y puede variar considerablemente. La motivación de los circuitos QDI es que ellos constituyen el más delicado compromiso para la insensibilidad al retardo puro requerido para construir circuitos prácticos usando compuertas simples [28].

En diferentes documentos publicados por Martin y otros [29],[30],[31] se minimiza la hipótesis de que todos los hilos del circuito tengan retardos no acotados y se realiza el supuesto de que ciertas bifurcaciones en los hilos son isocrónicas. Los circuitos QDI entonces, no reconocen retardos en los operadores o en los alambres, excepto para bifurcaciones isocrónicas para las cuales el diseñador asume que tienen retardos similares sobre las diferentes ramas. Los circuitos asíncronos QDI son los más conservativos en términos de retardos; debido a eso, su dependencia de los retardos es muy pequeña, siendo los más robustos contra las variaciones de parámetros físicos; esta robustez, por lo tanto, permite el intercambio de energía y ajustes de voltaje en forma sencilla [32].

2.2.5. Circuitos independientes de la velocidad (*Speed Independent, SI*)

Los circuitos SI fueron introducidos por David Muller en la década de 1950. Los circuitos SI asumen que los retardos de las interconexiones son insignificantes, solamente las compuertas del circuito exhiben retardos apreciables. Las bifurcaciones se asumen como isocrónicas [33]. Si en un circuito los retardos en las líneas se asumen como cero (o en la práctica, menores que el mínimo retardo de compuerta), y el circuito exhibe una correcta operación, sean cuales sean los retardos en cualquiera de los elementos del circuito, entonces se dice que el circuito es independiente de la velocidad [34]. El supuesto de que el retardo de las líneas es cero es válido para circuitos pequeños. En general, los circuito SI y QDI son considerados como equivalentes desde el punto de vista de diseño. Sin embargo, la figura 2 permite observar la diferencia entre ambos tipos de modelos.

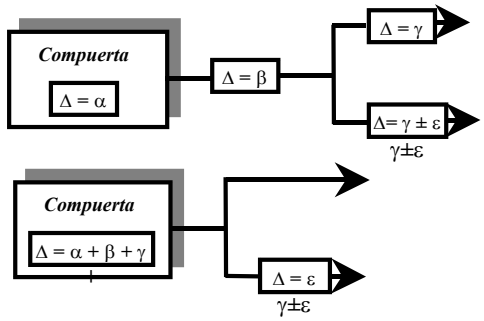

Figura 2: Modelos QDI (arriba) y SI (abajo) [2]

[4] Nota del autor: *"Este artículo contiene numerosas referencias al trabajo de Charles Molnar, quien ha inspirado a muchas de las personas que hoy son consideradas pioneras en el campo del diseño de los circuitos asíncronos "*

2.2.6. Circuitos *Scalable Delay-Insensitive (SDI)*

El modelo SDI es un modelo de retardo no acotado que no asume un límite superior sobre los retardos de compuertas y líneas de interconexión. Diferente a los modelos DI o QDI, el modelo SDI asume que existe una relación de retardo relativa y acotada entre cualquiera de dos componentes. El modelo SDI se define como sigue [36],[37]:

- ***Modelo Scalable Delay Insensitive***:
  Un componente se refiere a una compuerta o a una línea de interconexión entre dos compuertas. Sean ***d1*** y ***d2*** para los componentes ***C1*** y ***C2***. El retardo relativo ***D*** de ***C1*** a ***C2*** se expresa como ***D=d1/d2***. Sea ***De*** el retardo relativo estimado por el diseñador, y sea ***Da*** el retardo relativo observado a través del tiempo de vida del sistema. La relación de variación relativa ***R*** se expresa como ***R=Da/De***. Luego, el modelo SDI asume que ***R*** está acotada para cualquiera de dos componentes, es decir, ***1/K < R < K***, donde **K** es una constante y se llama relación de variación máxima.

La figura 3 ilustra el orden de cambio de señales garantizado en los modelos QDI y SDI. Lo que el modelo QDI garantiza es que el cambio de la señal ***t*** precede el cambio en las señales ***t1*** y ***t2***. De otro lado, el modelo SDI supone que los retardos estimados para las rutas ***CC1*** y ***CC2*** son ***d1*** y ***d2***, respectivamente. Entonces, se garantiza que el cambio de señal en ***t1*** precede a ***t2*** si ***d2 > K.d1***.

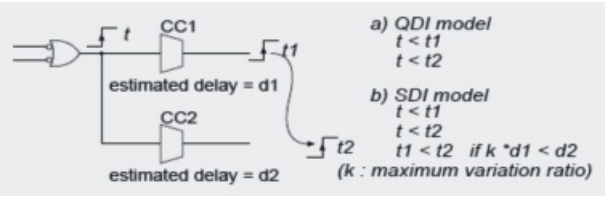

Figura 3: Orden de señal garantizada (a) Modelo QDI, (b) Modelo SDI [35]

La metodología de diseño SDI consiste en dividir un sistema completo en partes de razonable tamaño, diseñar cada parte basados en un modelo de retardo más optimista, y diseñar las partes de interconexión o las partes globales basados en el modelo DI. El modelo SDI es muy realista y ofrece una base amigable para el diseño de procesadores asíncronos de alto desempeño razonablemente confiable [35].

2.2.6. Micropipelines

La metodología de micropipelines [38] propuesta por Ivan Sutherland pretendía ser en principio una alternativa asíncrona a los pipelines elásticos síncronos (la cantidad de datos dentro de ellas puede variar). Posteriormente se observó que también sirven como un potente método para implementar cálculos segmentados. En general, un micropipeline se compone de un *datapath* con retardos acotados en cada etapa y está gobernado por un circuito de control insensible a retardos [6].

Los procesadores *pipeline* tienen una alta velocidad debido a que sus etapas separadas operan concurrentemente. Estos procesan y almacenan datos alternando, a lo largo del circuito; los elementos de memoria y los elementos de procesamiento. Los pipelines pueden ser sincronizados por reloj o manejados por eventos. Algunos pipelines son ***inelásticos*** (donde la cantidad de datos en ellos es fija). Las tazas de velocidad de entrada y de salida de un pipeline inelástico son idénticas. Otros pipelines son ***elásticos*** (la cantidad de datos en ellos puede variar); la taza de velocidad de entrada y la de salida pueden diferir momentáneamente debido a un almacenamiento interno (*buffering*). Un pipeline elástico se asemeja a una memoria FIFO (*first-in-first-out*). Las *FIFOs* pueden ser manejadas por reloj o por eventos, su propiedad importante es que ellas son elásticas. El nombre de ***micropipeline*** se ha asignado a una forma particularmente simple de *pipeline elástico* manejado por eventos, con o sin procesamiento interno. El nombre "micro" se debe a que los micropipelines contienen circuitería muy simple, a que son útiles en longitudes muy cortas y a que son adecuados para *layouts* en microelectrónica.

Los micropipelines utilizan los llamados ***elementos C de Muller*** para efectos de control de flujo de datos. Un elemento *C de Muller*, actúa como el elemento ***AND*** para eventos; se utiliza el símbolo lógico estándar AND con una "C" dentro de éste para su representación. La figura 4 muestra el símbolo de circuito y la implementación a nivel de transistores

para dos entradas.

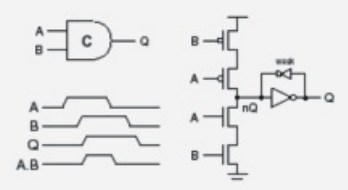

Figura 4: El elemento C de Muller

Los micropipelines, además de los elementos C de Muller, también utilizan circuitos controladores de eventos que implementan operaciones elementales. La figura 5 muestra los símbolos para los circuitos más conocidos.

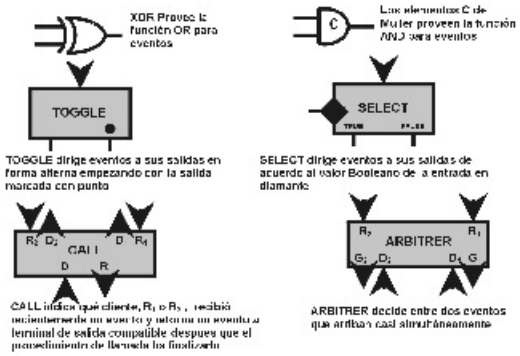

Figura 5: Módulos Lógicos para Control de Eventos [38]

Sutherland describe varios diseños como elementos de memoria llamados *registros controlados de eventos*, los cuales responden simétricamente a transiciones de subida y de bajada sobre las entradas. Tales pipelines han sido utilizados por muchos investigadores en el diseño de microprocesadores asíncronos. Sutherland, Sproull, Molnar y otros diseñaron, en los laboratorios de Sun icrosystems, el *"Counterflow microprocessor"* basado en micropipelines [39]. Los Micropipelines también forman la base del microprocesador *Manchester ARM* (desarrollado por Furber y el grupo AMULET) [40],[41].

La figura 6 muestra una cadena de elementos C de Muller con inversores interpuestos, la cual conforma la lógica requerida para controlar los micropipelines. Según la figura, una señal de solicitud y una señal de reconocimiento pasa entre etapas adyacentes de este control. Las líneas de datos también pasan entre etapas. Para cada interfaz entre etapas las señales de solicitud y reconocimiento y los datos siguen exactamente un protocolo de dos fases. Cada etapa en la figura 6 sigue la siguiente regla:

IF *el predecesor y el sucesor difieren de estado*
THEN *copiar el estado del predecesor*
ELSE *mantenga el estado presente*

Esta regla de estado de etapa hace estable el sistema de control cuando todas las etapas están en el mismo estado y cuando las etapas alternas están en estados opuestos.

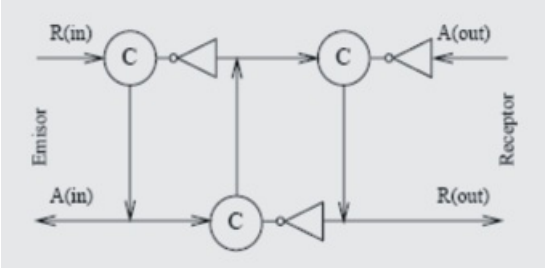

Figura 6: Circuito de Control para Micropipeline [6],[38]

Hay dos tipos de circuitos micropipelines: ***sin procesamiento*** *y* ***con procesamiento***. Los primeros cuentan con registros controlados por eventos para ruta de datos y elementos C de Muller para control. La figura 7 muestra un micropipeline de cuatro etapas, entre ellas se conforma un protocolo de dos fases de datos acotados. Los micropipelines con procesamiento usan lógica combinatoria entre los registros controlados por eventos mientras que los micropipelines sin procesamiento omiten tal lógica.

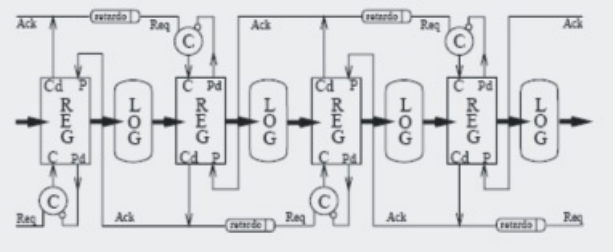

Figura 7: Micropipeline con Procesamiento [6],[38]

Las investigaciones sobre micropipelines asíncronos y "*datapaths*" están avanzando en muchas direcciones. Se han propuesto nuevos esquemas de pipelines asíncronos, algunos enfatizan la baja potencia [42],[43],[44] mientras que otros enfatizan el alto desempeño [44],[45],[46],[47]. También se han propuesto algunas generalizaciones a las estructuras de pipelines asíncronos: *rings* (anillos) [48], *multi-rings* [49] y *2-dimensional micropipelines* [50]. Se han establececido estructuras micropipelines de baja potencia usando escalamiento adaptativo de fuentes de voltaje [51].

2.2.8. Método de De-sincronización

Se trata de una metodología alterna para derivar circuitos asíncronos a partir de circuitos síncronos optimizados mediante el reemplazo del árbol de distribución de reloj por una red de *handshake*. La novedad de esta metodología es que no se requiere de un conocimiento del diseño asíncrono, en particular, esta metodología parte de una especificación HDL sintetizable o una lista de nodos a nivel de compuertas, para conseguir las ventajas claves de la asincronicidad. La idea esencial de la aproximación por de-sincronización es comenzar desde un circuito síncrono completamente sintetizado (o diseñado manualmente), y luego reemplazar directamente la red de reloj global por un conjunto de circuitos de *handshake* locales (figura 8).

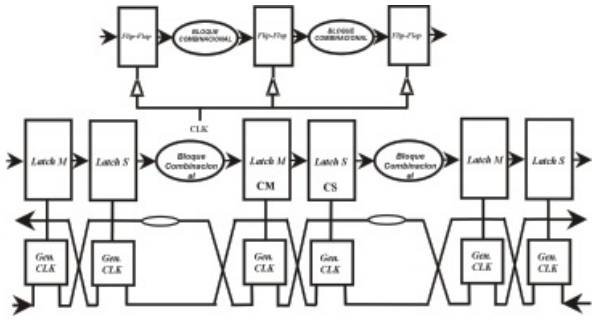

Figura 8: Circuito síncrono (arriba) y de-sincronizado (abajo) [34]

Datos de diseños reales muestran que en circuitos *ASIC* de alto desempeño la red de reloj disipa entre el 30 y el 50% de la potencia dinámica total. El método reduce en forma considerable este consumo ya que los relojes se generan localmente sin los peligrosos efectos de los retrasos de reloj (*clock skew*). Más aún, la emisión electromagnética (EMI) también se reduce drásticamente ya que los elementos de memoria (*latches* o *flip-flops*) no conmutan en forma simultánea [52], [53], [54].

3. Procesadores asíncronos

En 1988 se diseñó el primer microprocesador completamente asíncrono del mundo, el *CAM* (*Caltech Asynchronous Microprocessor*) es un microprocesador RISC (*Reduced Instruction Set Computing*) de 16 bits con 23.000 transistores. Una década después se diseñó el *MiniMIPS*, una versión asíncrona del microprocesador MIPS R3000 de 32 bits [32]. Un procesador recientemente diseñado es el *Lutonium* compatible con el microcontrolador 8051 asíncrono orientado al bajo consumo de energía [55],[56]. La tabla 1 permite comparar los resultados de desempeño de varios procesadores síncronos con su contraparte síncrona, el *Lutonium* [57]. En la tabla 2 se observa el desempeño de algunos procesadores comerciales similares al procesador *MiniMIPS* [32].

Tabla 1: Comparación de desempeño de algunos procesadores síncronos con el microprocesador asíncrono Lutonium [57]

Microprocesador	Tecnología del Proceso	Frecuencia	Consumo de Potencia	MIPS/mW	Año de Fabricación
Pentium Pro	0,6 μm	150 MHz	23 W	0,0065	1995
Celeron	0,25 μm	400 MHz	23,4 W	0,017	1999
Athlon (K7)	0,25 μm	700 MHz	45 W	0,019	1999
DEC21364	0,18 μm	1 GHz	100 W	0,047	2000
SH7708	0,5 μm	60 MHz	0,6 W	0,1	1994
Lutonium	0,18 μm	200 MHz	0,1 W	1,8	Asíncrono, 1,8 V, 2004
Lutonium	0,18 μm	4 MHz	0,17 mW	23	Asíncrono, 0,5 V, 2004

Tabla 2: Comparación de desempeño del Procesador MiniMIPS con otros Procesadores Comerciales [32]

Microprocesador	Tamaño de Word	Tecnología [/μm]	Frecuencia [/MHz]	Potencia por bit	Energía [/10^{-10} J]	Et^2 * [/10^{-26} Js^2]
MiniMIPS (Simulado)	32	0,6	280	0,219	7,8	1,0
MiniMIPS (fabricado)	32	0,6	180	0,125	7	2,1
R3000 (cpu)	32	1,2	25			
R3000A (cpu)	32	1,0	33			
VR3600 (cpu+fpu)	32	0,8	40			
R4600	64	0,64	150	0,0719	4,8	2,1
21064	64	0,6	200	0,469	23,5	2,1
R4400	64	0,6	150	0,234	15,6	7,0
SH7708	16/32	0,5	60	0,018	3	8,3
P6	32	0,6	150	1,8	120	52

Según [31],[32], la habilidad de los circuitos integrados QDI para correr a muy bajos voltajes, aún ligeramente debajo del umbral de voltaje del transistor, permite la gestión de energía por instrucción, ***E***, en relación al tiempo de ciclo, ***t***, a través del ajuste de la fuente de voltaje. Cuando la energía y el tiempo deban ser optimizados, surge la pregunta sobre qué exactamente optimizar, y qué tan

importante es la energía comparada con el tiempo y viceversa. Estos dos factores se combinan en la métrica Et^2, la cual es independiente del voltaje para circuitos CMOS. Debido a esta independencia de la fuente de voltaje, se puede escoger entre diseños de circuitos conociendo el menor Et^2 a un voltaje dado dentro de condiciones normales de operación; esto significa que si la fuente de voltaje puede ser ajustada, el circuito con menor Et^2 puede obtener un tiempo de ciclo más bajo que uno con mayor Et^2 con el mismo consumo de energía, o en forma equivalente, una menor energía con el mismo tiempo de ciclo.

4. Estado del diseño asíncrono en la industria

De acuerdo con [58] en reportes sucesivos de la *IST **(Information Society Technologies)*** sobre el **"Estado del Diseño Asíncrono en la Industria"** se hace un reconocimiento de las compañías comprometidas, o que muestran interés en la tecnología asíncrona. Las compañías involucradas se dividen en:

- aquellas con modelos de negocios primarios basados en tecnología asíncrona
- aquellos que venden productos que incorporan tecnología asíncrona
- aquellos con una significativa presencia de investigación en circuitos asíncronos
- quellas afiliadas al ***ACiD-WG*** (*Working Group on Asynchronous Circuit Design*) y quienes siguen la pista de la tecnología pero no tienen un esfuerzo visible en investigación
- aquellas que advierten un interés en la tecnología asíncrona, pero que no tienen un perfil significativo en la comunidad asíncrona

En [58] se describe el estado del arte en métodos y herramientas para el diseño de sistemas VLSI digitales asíncronos. El reporte está pensado inicialmente para que sea utilizado por compañías miembros y no miembros del Grupo de Trabajo sobre Diseño de Circuitos Asíncronos (*ACiD-WG, http://www.bcim.lsbu.ac.uk/ccsv/ACiD-WG/)),* quienes son conscientes de sus beneficios potenciales, pero necesitan conocer más acerca de los métodos de diseño y herramientas antes de comprometer recursos. El reporte ayuda a resaltar deficiencias en las aproximaciones existentes y así a proveer el ímpetu para el posterior desarrollo de herramientas. El reporte contiene las autoevaluaciones recibidas de varios desarrolladores de herramientas. Según este informe, *"las metodologías de test para circuitos asíncronos están todavía en su etapa de infancia"*. El documento presenta un anexo que identifica metodologías y herramientas de diseño. Se listan un total de 22 herramientas/metodologías en orden alfabético. Cuatro de ellas (ACK, Balsa, Tangram y TAST) están principalmente ocupados con la compilación de silicio, y dos (LARD y VHDL++) con simulación. Otros dos (BUTLER y NCL *technologies*) explotan librerías de componentes de propósito especial. Cinco más (3D, ATACS, MINIMALIST, Petrify y SIS) realizan síntesis lógica, tres más (di2pn, Pipefitter and VSTGL) se usan como parte de Petrify. Finalmente, seis (CADP, CCS, Firemaps, Transyst, Veraci y XDI) soportan verificación formal.

5. Conclusiones

El estado del arte indica que los circuitos asíncronos son un área creciente de investigación que ya tiene un impacto significativo en el diseño de circuitos integrados comerciales. Hay signos muy significativos que llevan a pensar que los circuitos asíncronos mantendrán su característica de primer orden en el campo investigativo. Hay serios esfuerzos por parte de los departamentos de investigación de algunas compañías como Sun Microsystems e Intel en incrementar el desempeño de los dispositivos. Philips, por su parte, tiene como objetivo principal reducir el consumo de potencia basándose en técnicas de diseño asíncrono.

El incipiente desarrollo de herramientas para *test* y verificación formal de los circuitos asíncronos constituyen una de las principales desventajas respecto del estilo de diseño síncrono. Sin embargo, existen notables esfuerzos por parte de la comunidad académica e industrial en desarrollar herramientas de

diseño como lo evidencian los resultados publicados en los journals de IEEE.

La implementación asíncrona de circuitos de alta complejidad ha mostrado mejores resultados en cuanto al desempeño que su contraparte síncrona, por este motivo varias de las metodologías de diseño asíncrono están siendo aplicadas a arquitecturas tradicionalmente síncronas.

6. Referencias bibliográficas

[1] Davis, A. Nowick S. M. "An Introduction to Asynchronous Circuit Design". Computer Science University of Utah, Salt Lake City, UT 84112; Computer Science, Columbia University, New York, NY 10027. September 19, 1997.

[2] Hauk, S. "Asynchronous Design Metodologies: An Overview". Department of Sciencie and Engineering, University of Washington, Seattle, WA 98195. Proceedings of IEEE, Vol 83, No. 1, pp. 69-93, January 1995.

[3] Bainbridge, W. J. Thesis: "Asynchronous System-on-Chip Interconnect" University of Manchester, Faculty of Science & Engineering. Department of Computer Science. March 2000.

[4] Ozdag R. O. Thesis "Template Based Asynchronous Design". Faculty of the Graduate School University of Southern California, November 2003.

[5] Vivet, P. "Une Methodologie de Conception de circuits Integres Quasi-Insensibles aux delais: application a l´etude et a la realization d´un processeur RISC 16-bit asynchrone". Institut National Polytechnique de Grenoble. France, Juin 2001.

[6] Peña, Bazurro., M. A.. Tesina: "Síntesis de circuitos asíncronos mediante la traducción de sincronización a redes de Petri". Universitate Politécnica de Catalunya. Departament d´Arquitectura de Computadores, Barcelona, marzo 1995.

[7] Liu, J. Ph.D. Thesis: "Arithmetic and Control Components for an Asynchronous System". University of Manchester, Faculty of Science and Engineering, Department of Computer Science, 1997.

[8] Nelson V., Nagle T,, Irwin D., "Analisis y Diseño de Circuitos Lógicos Digitales" . Primera Edición, Ed. Prentice-Hall.1996. ISBN: 968-880-706-0.

[9] Tinder, R. F. "Digital Engineering Design, A Modern Approach". Ed. Prentice-Hall. 1.994. ISBN: 0-13-2117077-X.

[10] Nowick, S.M., Dill, D.L. "Automatic Synthesis of Locally-Clocked Asynchronous State Machines''. IEEE International Conference on Computer-Aided Design (ICCAD), Santa Clara, CA 1991.

[11] Yun, K.Y., Dill, D.L., Nowick, S.M. "Synthesis of 3D Asynchronous State Machines"'. IEEE International Conference on Computer Design (ICCD), Cambridge, MA. 1992.

[12] Davis, A., Coates, B., Stevens, K. "The Post Office Experience: Designing a Large Asynchronous Chip". Proceedings of the 26th Annual Hawaii International Conference on Systems Sciences, Vol. I, pp. 409-418, 1993.

[13] Ladd, M., Birmingham, W. P. "Synthesis of multiple-input change asynchronous finite state machines". Proceedings. ACM/IEEE Design Automation Conference, pp. 309314, 1991.

[14] Rosenberger, F. U., Molnar, C. E.; Fang, T. P. "Synthesis of delay-insensitive modules". Chapel Hill Conference on VLSI. pags 67-86, May 1985.

[15] Rosenberger, F. U., Molnar, C. E., Chaney, T. J., Fang, T. P. "Q-modules Internally clocked delay-insensitive modules". IEEE Transactions on Computers Volume 37, IEEE Computer Society, Washington, DC, USA Issue 9 September, 1988. Pages: 10051018, ISSN:0018-9340.

[16] Muller, D. E., Bartfy, W. C. "A theory of asynchronous circuits". Annals of Computing Laboratory of Harvard University, pp. 204-243, 1959.

[17] Miller, R. E. "Sequential Circuits and Machines: Volume 2 of Switching Theory." Wiley, 1965

[18] Armstrong, D. B., Friedman, A. D., Menon, P. R. "Design of Asynchronous Circuits Assuming Unbounded Gate Delays". IEEE Transactions on Computers, C-18, 12, pp.1110-1120, December, 1969.

[19] Murata, T. "Petri Nets: Properties, Analysis and Applications", Proceedings of the IEEE, vol.

77(4), , pp. 541-580, April, 1989.

[20] Rosenblum, L.Y., Yakovlev, A.V. "Signal Graphs: from self-timed to timed ones". Proceedings. of the Int. Workshop on Timed Petri nets, 199-207, IEEE Computer Society, 1985.

[21] Chu, T. A. Ph..D. thesis: "Synthesis of Self-timed VLSI Circuits from Graph-Theoretic Specifications", M.I.T. Tech. Rep. MIT/LCS/TR-393, June 1987.

[22] Kishinevsky, M.A., Kondratyev, A. Y., Taubin, A. R., Varshavsky, V. "On Self-timed Behavior Verification". Proceedings of TAU´92, March 1992.

[23] Lavagno, L. Ph.D. thesis: "Synthesis and Testing of Bounded Wire Delay Asynchronous Circuits from Signal Transition Graphs".U.C. Berkeley, 1992.

[24] Moon, C.W., Stephan P. R., Brayton R.K. "Synthesis of hazard-free asynchronous circuits from graphical specifications". IEEE International Conference on Computer-Aided Design, pages 322-325, November 1991.

[25] Clark, W. A. "Macromodular computer systems". Proceedings of the Spring Joint Computer Conference, AFIPS, April 1967.

[26] Udding, J. T. "A formal model for defining and clasifying delay-insensitive circuits and systems" . Distributed Computing, pp 1 (4): 197-204, 1986.

[27] Dill, D. L. "Trace Theory for Automatic Hierarchical Veri_cation of Speed_Independent Circuits" MIT Press, Cambridge, MA,

[28] Davis, A., Nowick, S. M. "An Introduction to Asynchronous Circuit Design". UUCS,Computer Science. University of Utah, Columbia University, Salt Lake City, U.T, New York, NY 10027. September 19, 1997.

[29] Burns, S., Martin, A.J. "A synthesis method for self-timed VLSI circuits". Proceedings of the International Conference on Computer Design, 1987.

[30] Martin, A. J. "Formal program transformations for VLSI synthesis". E.W. Dijkstra, editor. Formal Development of Programs and Proofs UT Year of Programming Series. Addison-Wesley, 1990.

[31] Martin, A. J. "Programming in VLSI From communicating processes to delay-insensitive circuits". C.A.R Hoare, editor. Developments in Concurrency and Communications, págs 1-64. UT Year of Programming Series, Addison-Wesley, 1990.

[32] Martin, A. J., Nystrom M., Wong C. G. "Three Generations of Asynchronous Microprocessors" California Institute of Technology. IEEE. 0740-7445/03. November-December 2003.

[33] Bardsley, A. Thesis: "Implementing Balsa Handshake Circuits". Department of Computer Science, Faculty of Sciencie & Engineering, University of Manchester, 2000.

[34] Janin, L. Thesis: "Simulation and Visualisation for Debugging Large Scale Asynchronous Handshake Circuits". pp 23,24, University of Manchester, Faculty of Science & Engineering. Department of Computer Science, 2004.

[35] Yamazaki, A., Ryu H., Yoneda T. "Verification of Scalable-Delay-Insensitive asynchronous circuits" . IEICE TRANS. INF. & SYST., VOL. E00D, NO. 1 JANUARY 1995

[36] Takamura, A., Kuwako, M., Imai, M., Fujii, T. "TITAC-2: An asynchronous 32-bit microprocessor based on Scalable-Delay Insensitive". Graduate School of Information Science and Engineering, Tokyo Institute of Technology. Research Center for advanced Science and Technology, University of Tokyo. Japan. 1997

[37] Nanya T., Takamura A., Kuwako, M., Imai, M., Ozawa, M., Ozcan, M., Morizawa, R., Nakamura, H. "Scalable-Delay Insensitive Design: A high-Performance approach to dependable asynchronous Systems". Research Center for advanced Science and Technology, University of Tokyo. Japan. 1998

[38] Sutherland, I. E. "Micropipelines" Association for Computing Machinery, Inc. (ACM) Volume 32, number 6, June 1989.

[39] Molnar, C. E., Sproull, R. F., Sutherland, I. E. "The counterfow pipeline processor architecture". IEEE Design, Test of Computers, 11(3): pp. 48-59. Sun Microsystems Laboratories, Inc. and Institute for Biomedical Computing, Washington University. SMLI TR-94-25. April, 1994

[40] Furber, S. B., Day, P., Garside, J. D., Paver, N. C., Wood, J. V. "A micropipelined ARM".

Proceedings of the IFIP TC10/WG 10.5 International Conference on Very Large Scale Integration, p.211-220, September 07-10, 1993.

[41] Furber, S, B, Day, P., Garside, J. D., Paver, N.C., Wood, J. V. "The Design and Evaluation of an Asynchronous Microprocessor" Proceedings of the1994 IEEE International Conference on Computer Design: VLSI in Computer & Processors, pp.217-220, October 10-12, 1994.

[42] Farnsworth, C., Eduards, D. A., Sikand, S. S. "Utilising dynamic logic for low power consumption in asynchronous circuits". Proceedings of the International Symposium on Advanced Research in Asynchronous Circuits and Systems (Async 94), pages 186-194. IEEE Computer Society Press, November 1994.

[43] Amulet Group "Power Reduction for System Technology" PREST Project Deliverable D1.2. Report on Asynchronous Design Techniques/Arithmetic Styles Department of Computer Science, University of Manchester, Oxford Rd. Manchester, M13 9PL.

[44] Furber, S. B., Liu, J. "Dynamic logic in four_phase micropipelines". Proceedings of the International Symposium on Advanced Research in Asynchronous Circuits and Systems (Async96), pages 11-16. IEEE Computer Society Press, November 1996.

[45] Day, P., Woods, J. V. "Investigation into micropipeline latch design styles". IEEE Transactions on VLSI Systems, 3(2): 264-272. June 1995.

[46] Yun, K. Y., Beerel, P. A., Arceo, J. "High_performance asynchronous pipeline circuits". Proceedings of the International Symposium on Advanced Research in Asynchronous Circuits and Systems (Async96), pages 17-28. IEEE Computer Society Press, November 1996.

[47] Molnar, C. E., Jones, I. W., Coates, B., Lexau, J. "A fifo ring oscillator performance experiment". Proceedings of the International Symposium on Advanced Research in Asynchronous Circuits and Systems (Async97), IEEE Computer Society Press, April 1997.

[48] Williams, T. E. Ph.D Thesis "Self-timed rings and their application to division". Technical Report CSL-TR-91-482, Computer Systems Laboratory, Stanford University, 1991.

[49] Sparso, J., Staunstrup, J. "Design and performance analysis of delay insensitive multi-ring structures" Proceedings of the Twenty-Sixth Annual Hawaii International Conference on System Sciences, volume I, pages 349-358. IEEE Computer Society Press, January 1993.

[50] Gopalakrishnan, G. "Micropipeline wavefront arbiters using lockable C-elements" IEEE Design and Test, 1(4):55-64, Winter 1994.

[51] Nielsen, L. S., Niessen, C., Sparso, J., Van Berkel, K. "Low-Power Operation Using Self-Timed Circuits and Adaptive Scaling of the Supply Voltage". IEEE Transactions on VLSI, 2(4):7, 1994.

[52] Cortadella, J., Kondratyev, A., Lavagno, L., Sotiriou, C. "A concurrent model for de-synchronization". Univ. Politécnica de catalunya, Barcelona, Spain; Cadence Design Systems, San Jose; USA; Politenico di Torino, Italy; FORTH, Crete, Greece. 2003

[53] Aspida, IST-2002-37796. "Asynchronous open-Source Processor Ip of the DLX Architecture". August 6, 2003.

[54] Sotiriou, Ch. P., Lavagno, L. "De-Synchronization: Asynchronous Circuits from Synchronous Specifications" Foundation for Research and Technology Ellas (FORTH), Heraklion, Crete, Greece; Politeccnico di Torino, Torino, Italy. 2003

[55] Martin, A. J., Nyström, M., Papadantonakis, K., Penzes, P. I., Prakash, P., Wong, C. G., Chang, J., Ko, K. S., Lee, B., Ou, E., Pugh, J., Talvala, E-V., Tong, J. T., Tura, A. "The Lutonium: A Sub-Nanojoule Asynchronous 8051 Microcontroller". 9th IEEE International Symposium on Asynchronous Systems & Circuits, 2003.

[56] Talvala, E-V. Thesis: "Designing the Port Interface Unit for the Lutonium Asynchronous Microcontroller". California Institute of Technology, Pasadena, California, June 2003.

[57] Siemers, C. "Configurable Computing". V1.01, SS 2005. Institut für Informatik der Technischen Universität Clausthal. Http://www.in.tuclausthal.de/~techinf/ConfComp/ConfCompSkript_V1_01_SS2005.pdf

[58] Edwards, Doug A., Toms, W. B. "Design, Automation and Test for Asynchronous Circuits and Systems" Information Society Technologies (IST) Programme Concerted Action Thematic Network Contract IST-1999-29119, 3rd Edition June 2004.