Programación Automática de FPGAs desde lenguajes de alto nivel para aplicaciones multimedias

Depto. de Ingeniería y Ciencia de los Computadores.

Grupo de Arquitectura Avanzada de Computadores y Computación Reconfigurable

Germán León

leon@icc.uji.es

26 de febrero de 2004

Introducción

- **Sistemas actuales: Forge,Lava,Handel C ...**
  - Declaración de tipos de datos previa.
  - Uso limitado de tipo de datos escalares (enteros y coma fija).
  - Estos tipos de datos en general no se adaptan de forma eficiente a la implementación en la arquitectura reconfigurable (AR).
- **Sistema propuesto:**
  - Inferencia de tipos (asignación y propagación de tipos de datos).
  - Tratamiento simbólico.
  - Compilación sobre arquitecturas heterogéneas.

Introducción

- **Sistemas actuales: Forge,Lava,Handel C ...**
  - Declaración de tipos de datos previa.
  - Uso limitado de tipo de datos escalares (enteros y coma fija).
  - Estos tipos de datos en general no se adaptan de forma eficiente a la implementación en la arquitectura reconfigurable (AR).
- **Sistema propuesto:**
  - Inferencia de tipos (asignación y propagación de tipos de datos).
  - Tratamiento simbólico.
  - Compilación sobre arquitecturas heterogéneas.

Introducción (II)

- Sistema de desarrollo [Lagadec2000]
  - Meta-herramientas para el rutado de circuito.
  - Síntesis de circuitos combinacionales basados en expresiones.
-

Introducción (II)

- Sistema de desarrollo [Lagadec2000]
  - Meta-herramientas para el rutado de circuito.
  - Síntesis de circuitos combinacionales basados en expresiones.
- Aplicaciones Multimedia:
  - Grano fino y muy dependiente del flujo de datos.
  - Alto Paralelismo.
  -

Introducción (II)

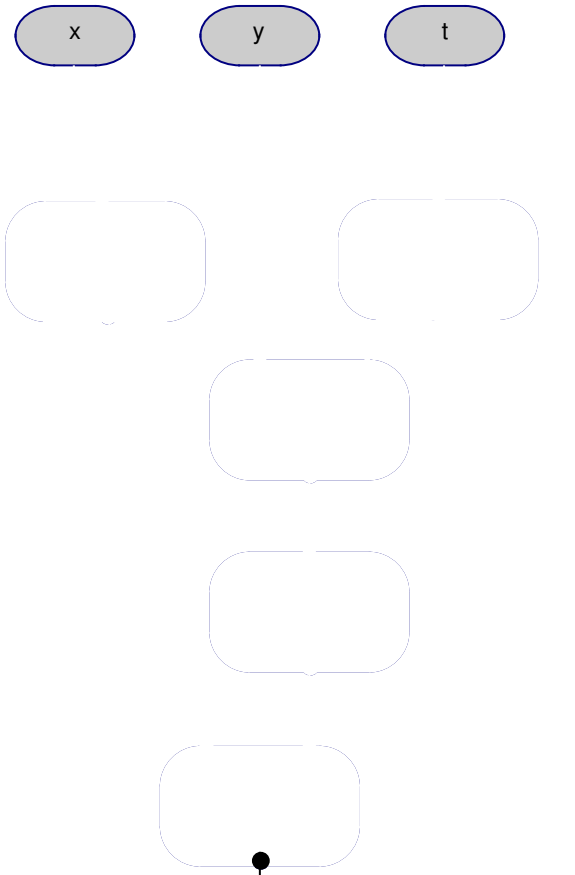
- Sistema de desarrollo [Lagadec2000]
  - Meta-herramientas para el rutado de circuito.
  - Síntesis de circuitos combinacionales basados en expresiones.
- Aplicaciones Multimedia:
  - Grano fino y muy dependiente del flujo de datos.
  - Alto Paralelismo.
  - Buscar formas para expresar las operaciones SIMD y adaptarlas a los recursos de la FPGA.

Implementación en AR

- RA de grano fijo : *registros*, *interconexiones* y *LUTs*.
- .
-
-

Implementación en AR

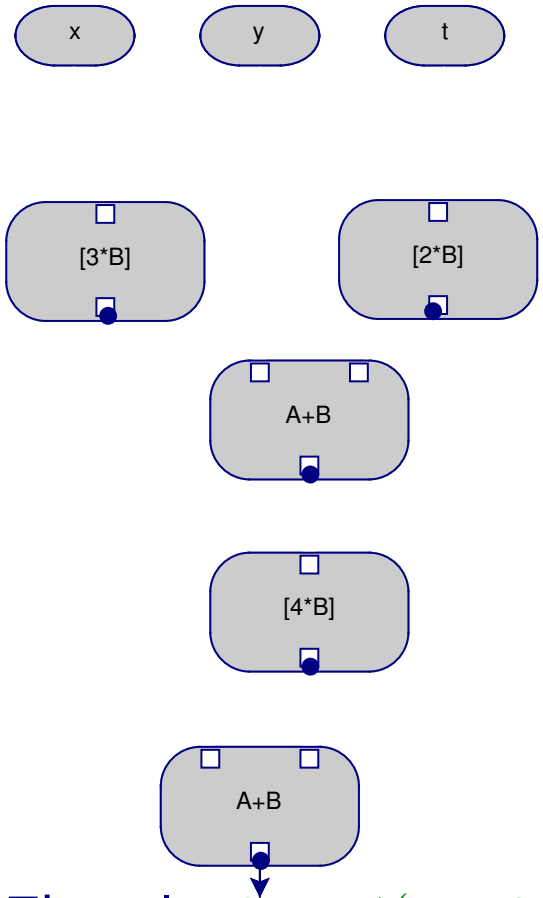
- RA de grano fijo : *registros*, *interconexiones* y *LUTs*.
- .
-
-



Ejemplo: $3x + 4(y + 2t)$

Implementación en AR

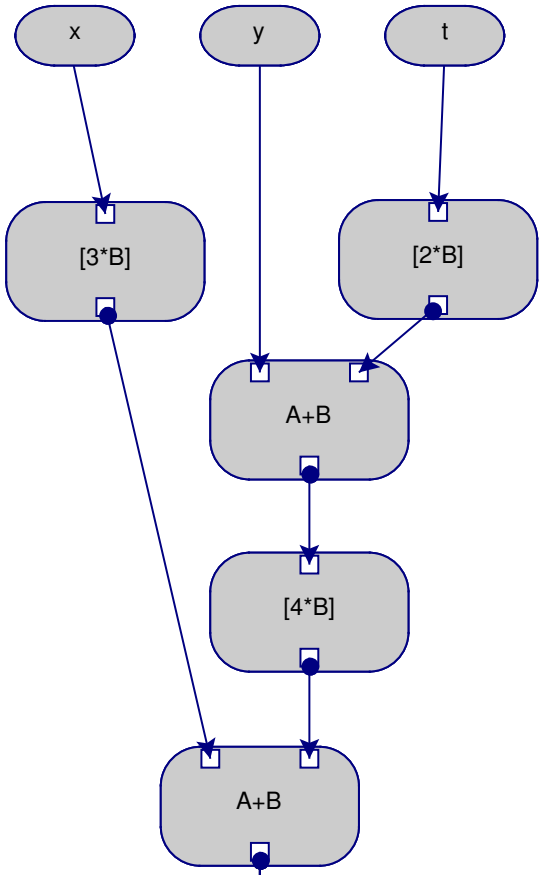
- RA de grano fijo : *registros*, *interconexiones* y *LUTs*.
- Cada operador se transforma en una LUT .
-
-



Ejemplo: $3x + 4(y + 2t)$

Implementación en AR

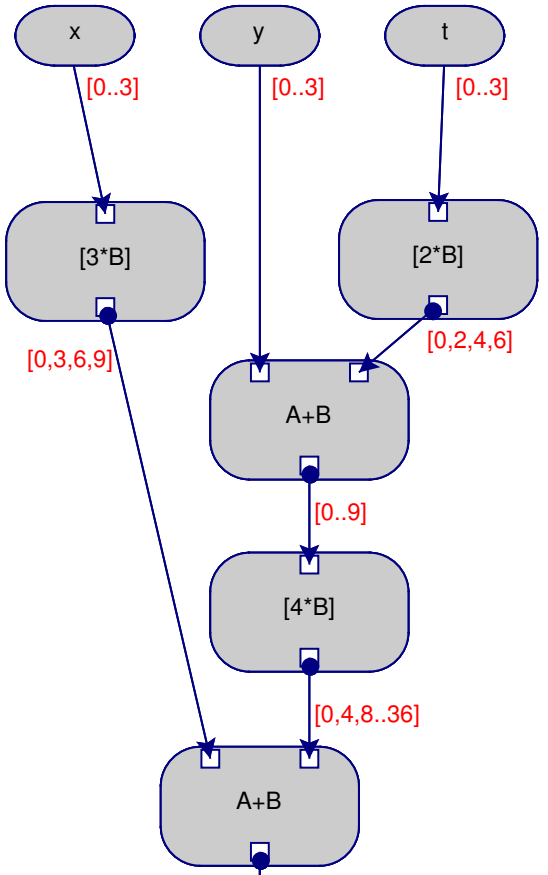
- RA de grano fijo : *registros*, *interconexiones* y *LUTs*.
- Cada operador se transforma en una LUT .
- Se obtiene la interconexión de operadores según las dependencias de datos
-



Ejemplo: $3x + 4(y + 2t)$

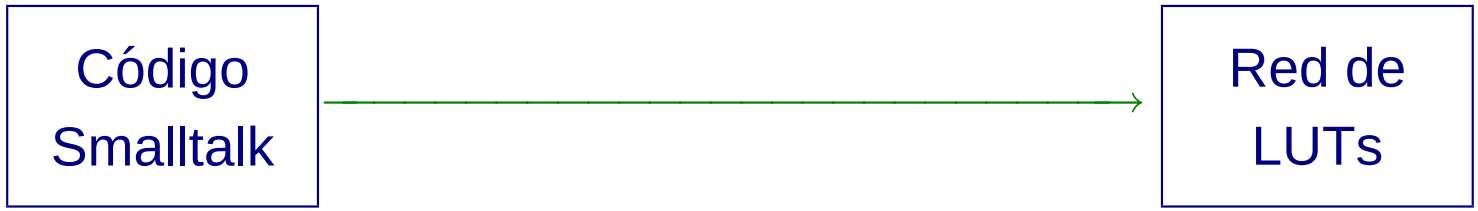
Implementación en AR

- RA de grano fijo : *registros*, *interconexiones* y *LUTs*.
- Cada operador se transforma en una LUT .
- Se obtiene la interconexión de operadores según las dependencias de datos
- Asignación y propagación de tipos de datos.



Ejemplo: $3x + 4(y + 2t)$

Modelo de ejecución



Modelo de ejecución

- Compilación
  - Esquema Algorítmico ⇒ Grafo de Flujo de Datos(GFD).
  - Análisis de realimentaciones para estructuras iterativas.

Modelo de ejecución

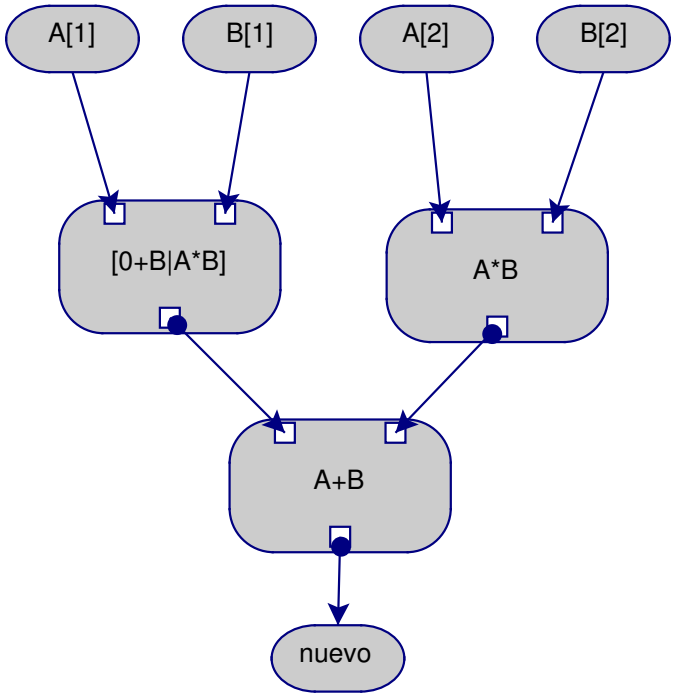
- Compilación
  - Esquema Algorítmico $\Rightarrow$ Grafo de Flujo de Datos(GFD).
  - Análisis de realimentaciones para estructuras iterativas.
- Cálculo de LUTs
  - Se evalúa para cada operador su conjunto de entrada y se calcula el conjunto de salida.
    - Independencia del tipo de datos de cada operando.
    - Permite buscar una codificación óptima.

Compilación y Síntesis lógica

- Obtención del GFD.
- Asignación y propagación de tipos de datos.
- Optimización del Grafo.
- Obtención de la Red de LUTs, interconexión y registros.
- Transformación a un *netlist* (p.e. Bliff, VHDL)
  - Simplificación de LUT: Utilizando el *SIS* en un Cluster.
  - Integracción en VHDL: Especificación de multiplexores y registros, Traducción de las LUT simplificadas.

Método para síntesis de una AR paralela

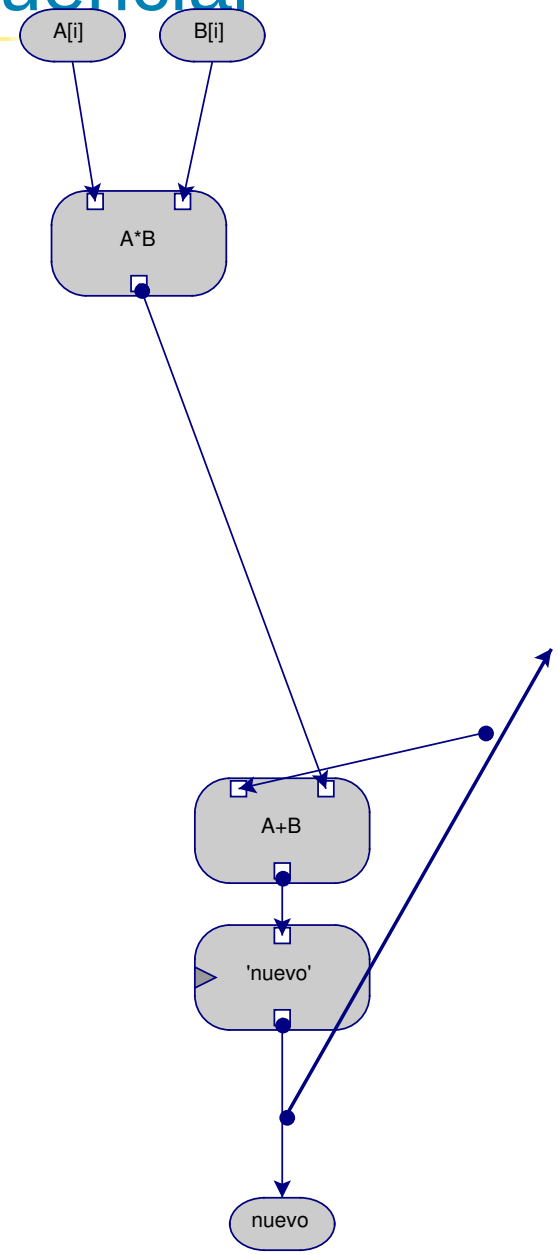
```
nuevo:=0.
(1 to: 2) collect:[:i| nuevo:=nuevo+((A at:i)*(B at:i))]
```



Método para síntesis de una AR secuencial

```
nuevo:=0.
(1 to: 20) do:[:i
   nuevo:=nuevo+((A at:i)*(B at:i))]
```

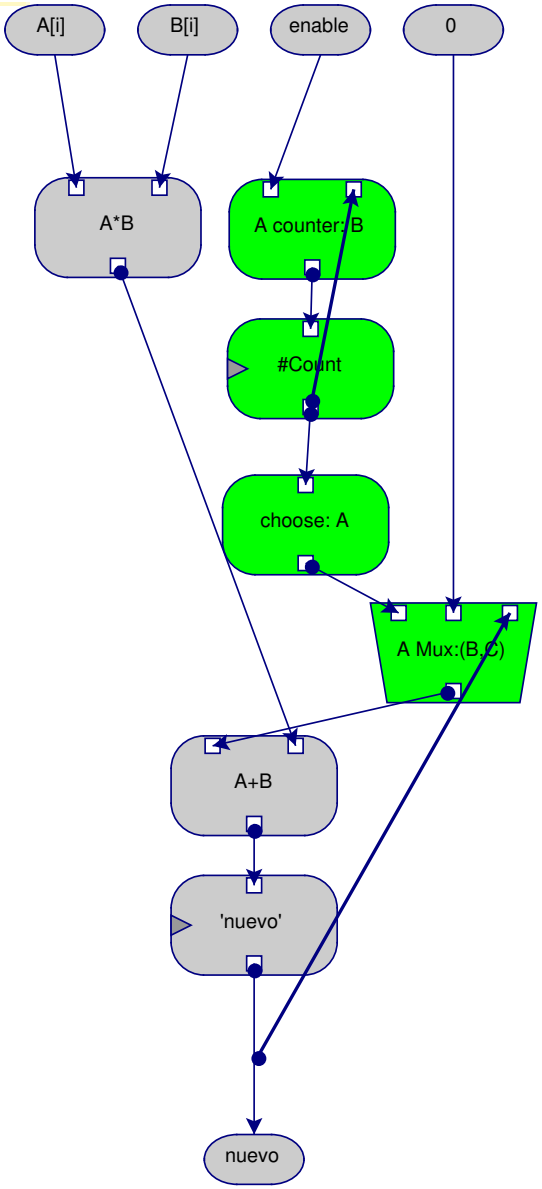
- Especifica un circuito secuencial síncrono
- Cada ciclo de reloj se deben introducir nuevos datos



Método para síntesis de una AR secuencial

```
nuevo:=0.
(1 to: 20) do:[:i
   nuevo:=nuevo+((A at:i)*(B at:i))]
```

- Especifica un circuito secuencial síncrono
- Cada ciclo de reloj se deben introducir nuevos datos



Trabajo en desarrollo

- Evaluar codificaciones internas entre las interconexiones intermedias.
- Análisis de resultado en algoritmos con operación y tipos no convecionales (p.e. Campos de Galois).
- Implementar mensajes que determinen el desenrollado de un bucle dependiendo de los recursos asociado a una RA específica.
- Incluir optimizaciones en el cálculo de la LUT.
- Incluir lógica difusa en la inferencia de tipos.