Обзор Sa-C

Sa-C – высокоуровневый алгоритмический язык, предназначенный для создания приложений обработки графических данных на ПЛИС. Разработан в университете штата Колорадо в рамках проекта Cameron. По структуре и синтаксису язык является C-подобным, поддерживает высокоуровневый (на уровне цикла) и низкоуровневый (на уровне инструкций) параллелизм.

Особенности языка

- Разнообразные типы данных: целые со знаком, без знака, с фиксированной точкой, у пользователя имеется возможность задавать размерность типа (9,12, 16, 32 и.т.д. бит):

```
fix12.4;
int16;
```

- Если имя объявляемой переменной совпадает с именем уже существующей, предыдущая переменная более не рассматривается:

```
int12  x  = 1;
……..
uint18 x = x+1;
```

- Отсутствует поддержка динамических структур данных и рекурсивных вызовов;
- Поддерживаются многомерные массивы, размер которых может задаваться как статически, так и динамически:

```
int14 M [:,6];
```

- Отсутствуют операции взятия адреса (&) и разыменовки (*);
- Введена новая система обращения к массивам в цикле for – генераторы:
  - scalar – позволяет получить линейную последовательность скалярных величин из массива,
  - array-element –получает один элемент из массива,
  - array-slice – получение подмассива минимальной размерности (например, вектора от матрицы),
  - array-window – получает прямоугольный подмассив из исходного массива;
- Цикл for состоит из 3 частей:
  - тело цикла,
  - одно или более возвращаемое значение,
  - один или более генератор;
- Цикл может возвращать значение редукции, вычисленной в его теле (sum, product, min, max, median, mean, histogram):

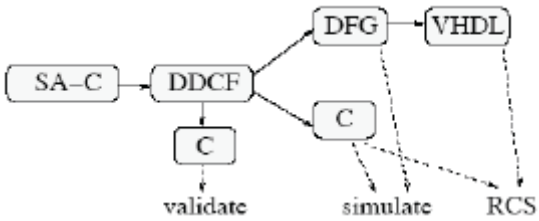
```
uint8 R [:,:] =
 for window W[3,3] in A{
     uint8 med = array_median(W)
 }return (array(med));
```

- В теле цикла могут вводиться “nexified” (с ключевым словом next) переменные. От значений таких переменных на текущем шаге цикла зависят вычисления на следующем;
- Поддержка оптимизации:

- построение шаблона доступа к массивам,
- циклы без “nexified” переменных полностью параллельны.

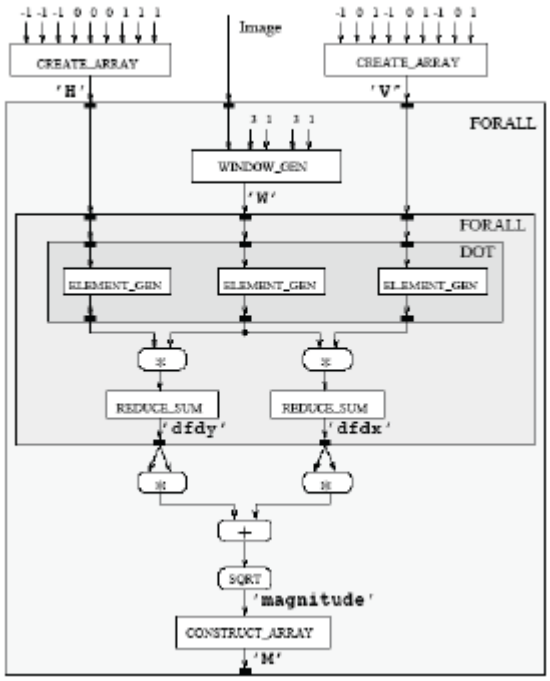
Принципы работы транслятора Sa-C

Трансляция с Sa-C в язык VHDL осуществляется за несколько этапов. На первом этапе строится граф зависимости по данным и потока управления (DDCF, Data Dependent Control Flow), который используется для оптимизации. Далее следует построение графа потоков данных (DFG), который может быть оттранслирован в VHDL-код. Также на основе DDCF или DFG графов можно сгенерировать код на C для отладки, либо симуляции.



Пример небольшой программы на Sa-C, выделяющей контуры объекта на изображении, и её DDCF:

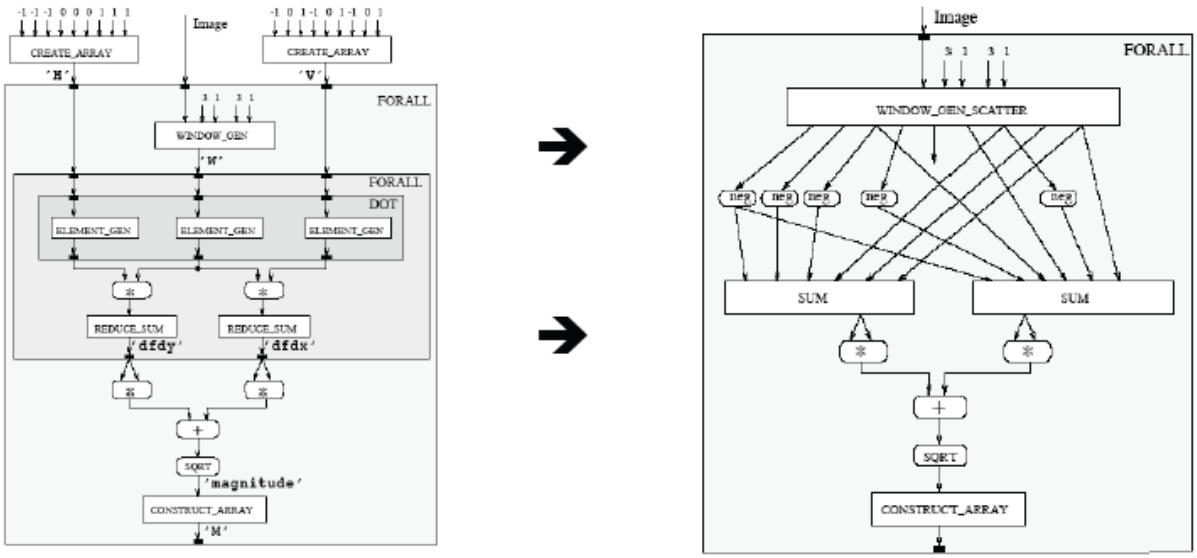
```
/* Prewitt Edge Detector Code */
int16[:,:] main (uint8 Image[:,:]) {
  int16 H[3,3] = {{ -1, -1, -1 },
                  { 0, 0, 0 },
                  { 1, 1, 1 }};
  int16 V[3,3] = {{ -1, 0, 1 },
                  { -1, 0, 1 },
                  { -1, 0, 1 }};
  int16 M[:,:] =
    for window W[3,3] in Image {
      int16 dfdy, int16 dfdx =
        for h in H dot w in W dot v in V
          return (sum (h*w), sum (v*w));
      int16 magnitude =
        sqrt (dfdy*dfdy+dfdx*dfdx);
      } return (array (magnitude));
  } return (M);
}
```



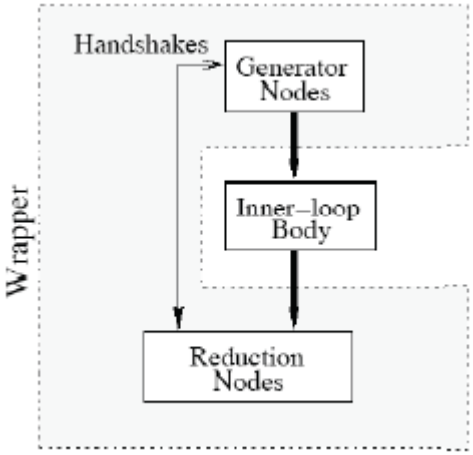
Закрашенные прямоугольники на схеме обозначают входные и выходные порты.
Оптимизация кода производится 3 путями:

- Автоматическая трансформация полученного DDCF-графа;
- Пользователь может ввести прототипы функций для внешних VHDL модулей;
- Транслятор поддерживает директивы, специфичные для оптимизации кода.

В DFG вершинами являются операторы, ребрами – потоки данных. Существуют вершины двух типов – простые, такие как арифметические и логические операторы, и комплексные, например, генераторы, редукции. На рисунке показан DFG, для нашего примера, полученный из оптимизированного DDCF.



На последнем этапе DFG транслируется в VHDL. Простые вершины графа непосредственно заменяются на единичные предложения на VHDL и формируют тело цикла, комплексные вершины транслируются путем выбора необходимого компонента в библиотеке и формируют «обертку»:



Производительность

В качестве теста на производительность была использована программа решения матричного уравнения вида Ax = b, с матрицей A размером 8x8 элементов. Оттранслированный в VHDL Sa-C код был зашит в микросхему Xilink XCV-1000-BG560-4, аналогичная программа на C запускалась при помощи тестовой платформы на базе процессора Pentium III 800 МГц:

seconds (800 MHz P3)	microseconds (AMS)	Frequency	Logic Blocks
$2.3x10^{-6}$	$0.5x10^{-6}$	25.2 MHz	35%