УДК 519.6

МОДЕЛИРОВАНИЕ ИНКРЕМЕНТНЫХ АЛГОРИТМОВ РЕШЕНИЯ СИСТЕМ ЛИНЕЙНЫХ АЛГЕБРАИЧЕСКИХ УРАВНЕНИЙ ДЛЯ РЕАЛИЗАЦИИ НА FPGA

Гомозов О.В., Ладыженский Ю.В.
Донецкий национальный технический университет

Во многих инженерных, научно-исследовательских задачах технических и гуманитарных наук используются СЛАУ – системы линейных алгебраических уравнений, позволяющий описать ту или иную модель, процесс, явление. Особенностью этих систем является их порядок – количество уравнений, который может достигать сотен или даже тысяч, при этом задачи требуют быстрого и точного решения, но также являются очень трудоемкими. В связи с этим имеются специализированные алгоритмы, производящие вычисления на цифровых устройствах, ориентированных на подобные задачи. Одним из перспективных направлений являются итерационные алгоритмы, называемые инкрементными[1]. В докладе исследуются их особенности, возможности, а также проводится анализ аппаратной реализации.

Каноническое представление СЛАУ имеет вид $A \cdot \vec{x} = \vec{b}$, его можно привести к другому виду, а именно, эквивалентной системе линейных дифференциальных уравнений, через интегрирование: $\frac{dx_i}{dt} = b_i - \sum_{i,j=1}^{n} a_{ij} x_i$, $i, j = 1,2....n$, $n - порядок\ СЛАУ$, и обозначить левую часть через $\varepsilon_i(x)$, тогда $\varepsilon_i(x) = b_i - \sum_{i,j=1}^{n} a_{ij} x_i$. Применяя метод Эйлера[1] имеем обобщенные формулы для инкрементных алгоритмов:

$$
\begin{aligned}
& x_i^{k+1} = x_i^k + \Delta x_i^{k+1}; \\
& \Delta x_i^{k+1} = 2^{-p} \cdot sign(\varepsilon_i^k), \\
& \varepsilon_i^{k+1} = \varepsilon_i^k + \Delta \varepsilon_i^{k+1}; \\
& \Delta \varepsilon_i^{k+1} = -\sum_{j=1}^{n} a_{ij} \Delta x_j^{k+1}, \ где\ x_j^0 = 0, \varepsilon_i^0 = b_i, \qquad (1)
\end{aligned}
$$

где a_{ij} *– коэффициент матрицы* A,

b_i *– свободный член*,

x_i *– неизвестное*,

Δx_i *– приращение неизвестного*,

ε_i *– невязка*,

$\Delta \varepsilon_i$ *– приращение невязки*,

i *– номер неизвестного*,

k *– номер итерации*,

p *– разрядность величин*

Важным понятием в данных методах является невязка $\varepsilon_i(x)$, определяющая отклонение текущего значения неизвестного x_i от истинного на числовой оси. Поэтому эта величина является критерием условия сходимости [2]:

$$\sum_{i=1}^{n}(\varepsilon_i^k)^2 \le E\text{ps} - \textit{погрешность вычисления} \quad (2)$$

Вторым критерием окончания решения может быть ограничение на максимальное количество итераций Im. Псевдокод алгоритма представлен на рис. 1.

```
BEGIN PROGRAM
k := 0
SET p, eps, Im
FOR i := 1 TO N
        ε_i := b_i
        x_i := 0
END FOR
REPEAT
cr := 0
FOR i := 1 TO N
        Δx_i := 2^(-p) · sign(ε_i)
        Δε_i := 0
        FOR j := 1 TO N
                Δε_i := Δε_i + a_ij Δx_j
        END FOR
        ε_i := ε_i − Δε_i
        x_i := x_i + Δx_i
        cr := cr + (ε_i)^2
END FOR
        k := k + 1
UNTIL (cr < eps) OR (k > Im)
END PROGRAM
```

Рисунок 1 – Псевдокод обобщенного инкрементного алгоритма

Формулы, приведенные в (1) показывают, что инкрементные алгоритмы ориентированы на использование простых операций сложения, а трудоемкое умножение заменяется операцией сдвига, так как приращение Δx_i является степенью двойки.

Различные модификации основного алгоритма представлены в виде различных вариантов формулы нахождения приращения неизвестного. Существует 4 метода [2]:

Метод №1:

$$\Delta x_i^{k+1} = 2^{-p} \cdot sign(\varepsilon_i^k),\ где\ \ p - разрядность \tag{3}$$

Метод №2:

$$\Delta x^{k+1} = 2^{-r} \cdot sign(\varepsilon^k), где \quad r = 1,2 \ldots p-1, p \tag{4}$$

Метод №3:

$$\Delta x^{k+1} = 2^{-\frac{p}{S}} \cdot sign(\varepsilon^k), где\ S = r, \frac{r}{2}, \frac{r}{3}, \ldots\ldots 1, \tag{5}$$

$$каждые\ 2^{\frac{p}{r}}\ итерации\ выбирается\ новое\ S$$

$$(r - количество\ участков, на\ которые\ разбивается\ p)$$

Метод №4:

$$\Delta x^{k+1} = 2^{-z} \cdot U_i \cdot sign(\varepsilon_i^k); \quad z = \min_{i=1..n} m_i;$$

$$2^{-m_i} \le \varepsilon_i^k < 2^{-m_i+1}; \quad U_i = \begin{cases} 1, \varepsilon_i^k \ge 2^{-z} \\ 0, \varepsilon_i^k < 2^{-z} \end{cases} \tag{6}$$

Основным критерием быстродействия алгоритмов является количество итераций k, зависящее от разрядности данных и характера матрицы A. Для исследования этих алгоритмов, в частности, алгоритма (3), была составлена абстрактная модель устройства на языке VHDL в среде Active-HDL. Модель обрабатывает числа с фиксированной запятой. В качестве тестовых были взяты два вида матриц: Dom – матрицы с доминирующими элементами на главной диагонали и Gil – плохо обусловленные матрицы Гильберта, известные тем, что большинство итерационных алгоритмов расходятся при их решении или же решение является ошибочным из-за неустойчивости к точности коэффициентов матрицы A. На рис. 2 - 4 представлены временные диаграммы решения матрицы Dom, где приняты следующие обозначения: A – матрица коэффициентов, B – вектор свободных членов, Xx – вектор найденных неизвестных, k- количество итераций, N – порядок СЛАУ, P – разрядность данных, X – истинные значения неизвестных.

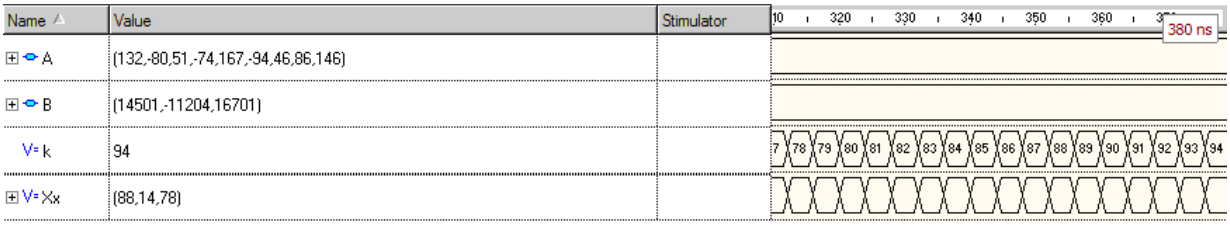

Рисунок 2 – Моделирование N=3,P=16,Dom,X=88,14,78

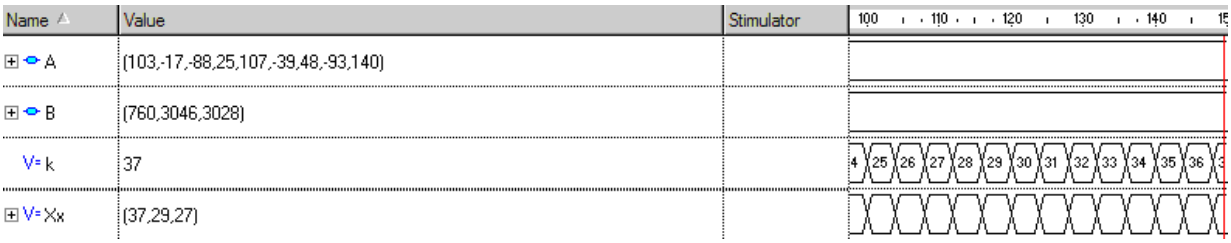

Рисунок 3 – Моделирование N=3,P=32,Dom,X=36,30,29

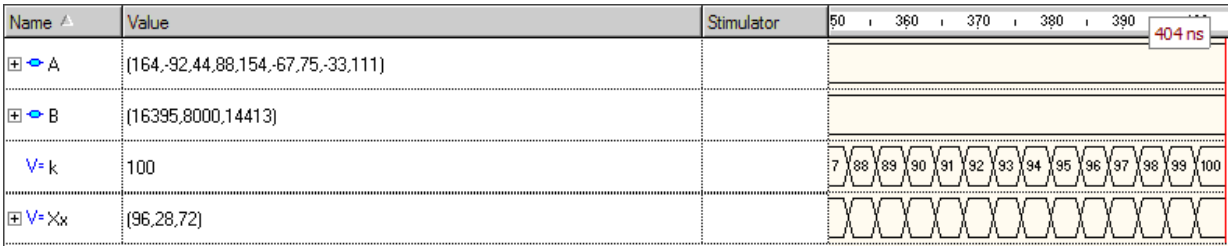

Рисунок 4 – Моделирование N=3,P=64,Dom,X=96,28,72

Как видно из рисунков, подобная модель решает СЛАУ быстро и с погрешностью, не превышающей младшего разряда, даже при малой разрядности данных. Однако в случае с матрицей Гильберта возрастает погрешность вычисления, связанная с ошибкой округления входных данных. Для противодействия этому эффекту возможно масштабирование коэффициентов матрицы и свободных членов к более высоким порядкам, и соответственное увеличение разрядности данных, что может уменьшить чувствительность алгоритмов к входным данным, результат представлен на рис. 5. При этом также возрастает и количество итераций, однако относительная погрешность решения остается практически допустимой.

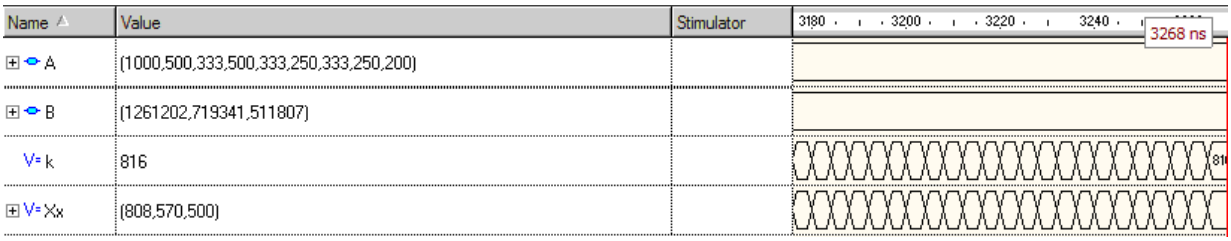

Рисунок 5 – Моделирование N=3,P=32,Gil,X = 809,585,480

Модификации (4)-(6) алгоритма способны уменьшить количество итераций, и увеличить скорость сходимости алгоритма [3]. Стоит также отметить, что исходя из формул (1), в инкрементных методах возможно параллельное вычисление каждого неизвестного и его промежуточных данных на каждой итерации.

Для аппаратной реализации вышеприведенных алгоритмов оптимальными являются интегральные схемы FPGA[4] таких фирм как Xilinx, Altera, Lattice и т.д. В их состав входят отдельные программируемые блоки памяти и шины данных. Такая структура дает возможность сконфигурировать и запрограммировать микросхемы, наиболее эффективно используя достоинства инкрементных алгоритмов, в частности, распараллеливание вычислений [4]. Реализация на FPGA является дешевым, расширяемым и масштабируемым решением.

Проведенные исследования показывают, что инкрементные методы обладают следующими свойствами:

- высокая скорость решения задач и простота арифметических операций;
- возможность эффективной реализации на базе микросхем FPGA;
- устойчивость и отсутствие ограничений для многим классам задач.

Литература

[1] Б.Н.Малиновский, В.П.Боюн, Л.Г.Козлов. Алгоритмы решения систем линейных алгебраических уравнений, ориентированные на структурную реализацию. – “Управляющие системы и машины”. Вып. №5. 1977, с.79-84.

[2] Боюн В.П., Козлов Л.Г., Малиновский Б.Н., Третьяков С.И. Устройства для решения систем линейных алгебраических уравнений. Автор. свид. № 543943 – БИ, 1977, № 3.

[3] Куркчи В.А., Гомозов О.В., Ладыженский Ю.В. Исследование инкрементных методов решения СЛАУ. – Донецк, ДонНТУ: Всеукраинская научно-техническая конференция “КМиТ”, 2009, с. 190-193.

[4] Максфилд К. Проектирование на ПЛИС. – М.: ”Додека-XXI”, 2007, 408с.