Лабораторная работа № 2
по курсу "Спецпроцессоры"
Изучение процессора CORDIC

Цель работы: разработать схему устройства, реализующего алгоритм CORDIC для чисел с фиксированной запятой в дополнительном коде (8-разрядных) в разных модификациях, исследовать его работу средствами САПР ПЛИС Quartus II.

Алгоритм CORDIC (COordinate Rotation DIgital Computer – цифровой компьютер для вращения координат) предназначен для поворота двумерного вектора на заданный угол. Данная операция может быть записана следующим образом:

$$\begin{pmatrix} x \\ y \end{pmatrix}' = \begin{pmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}, \quad (1)$$

где $\boldsymbol{a} = \begin{pmatrix} x \\ y \end{pmatrix}$ – исходный преобразуемый вектор, $\boldsymbol{a}' = \begin{pmatrix} x \\ y \end{pmatrix}'$ – вектор-результат, α – угол, на который необходимо повернуть вектор $\boldsymbol{a}$,

$$\boldsymbol{R}(\alpha) = \boldsymbol{R} = \begin{pmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{pmatrix} - \quad (2)$$

матрица вращения. Преобразование (1) может быть также записано в матричной форме:

$$\boldsymbol{a}' = \boldsymbol{R}\boldsymbol{a}, \quad (3)$$

и называется вращением Гивенса или преобразованием Гивенса.

На рис. 1 показана геометрическая интерпретация преобразования Гивенса.

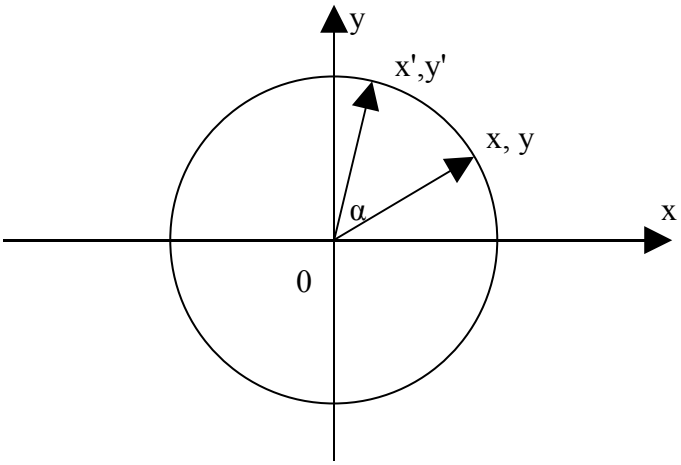

Рис. 1 – геометрическая интерпретация преобразования Гивенса

Выполнение преобразования (1) требует вычисления тригонометрических функций, умножения вычисленных значений на координаты исходного вектора и наконец вычисление сумм и разностей полученных произведений. Данные операции достаточно сложно реализуются аппаратурно, не могут выполняться параллельно, плохо конвейеризуются, следовательно, реализующее их устройство будет иметь большую аппаратурную сложность и низкую производительность.

Идея алгоритма CORDIC состоит в преобразовании (1) в множество простых, легко реализуемых аппаратурно операций, таких как сложение, вычитание, определение знака числа и двоичный сдвиг (умножение на степень 2). Для этого α раскладывается в виде:

$$\alpha = \sum_{k=0}^{\infty} \xi_k \Delta\alpha_k , \quad (3)$$

где $\xi_k \in X = \{+1, -1\}$ – операторы направления вращения,

$$\Delta\alpha_k = \operatorname{arctg} 2^{-k} - \quad (4)$$

базисные углы, выбираемые из I квадранта координатной плоскости (таблица 1).

Таблица 1. Значения некоторых базисных углов

k	$\Delta\alpha_k$	
	рад	град
0	π/4	45,00
1	0,4636	26,57
2	0,2450	14,04
3	0,1244	7,13
4	0,0624	3,58
5	0,0312	1,79
6	0,0156	0,90
7	0,0078	0,45
8	0,0039	0,22

Заметим, что (3) является записью α в позиционной системе счисления с цифрами X и весами $\Delta\alpha_k$.

Прочие свойства членов ряда (3):

$$\forall k \geq 0: \Delta\alpha_k > 0, \tag{5}$$

$$\sum_{k=0}^{\infty} \Delta\alpha_k = \alpha_{max} \approx 1{,}74 \approx 100\,^\circ, \tag{6}$$

$$\lim_{k\to\infty} \Delta\alpha_k = 0\ . \tag{7}$$

Из (5) – (7) следует, что $\forall\alpha$: $-\alpha_{\max} \leq \alpha \leq \alpha_{\max}$ может быть представлено рядом (3), если выполняется условие:

$$\forall\, k \geq 0 : \Delta\alpha_k \leq \sum_{m=k+1}^{\infty} \Delta\alpha_m\ . \tag{8}$$

Из (4) следует, что условие (8) выполняется.

Композиция поворотов есть поворот на суммарный угол:

$$\boldsymbol{R}(\alpha_2)\boldsymbol{R}(\alpha_1)\boldsymbol{a} = \boldsymbol{R}(\alpha_1 + \alpha_2)\boldsymbol{a}, \tag{9}$$

следовательно, (3) позволяет заменить поворот на угол α последовательностью поворотов на базисные углы:

$$R\left(\sum_{k=0}^{\infty} \xi_k \Delta\alpha_k\right) = \prod_{k=0}^{\infty} R(\xi_k \Delta\alpha_k)\ , \tag{10}$$

или, учитывая коммутативность произведений $\boldsymbol{R}$:

$$\begin{cases} \boldsymbol{a_0} = \boldsymbol{a}, \\ \boldsymbol{a_{k+1}} = \boldsymbol{R}(\xi_k \Delta\alpha_k)\,\boldsymbol{a_k}, \\ \boldsymbol{a'} = \lim\limits_{k\to\infty} \boldsymbol{a_k}, \end{cases} \tag{11}$$

где

$$a_k = \left(\prod_{m=0}^{k-1} R(\xi_m \Delta\alpha_m)\right) a\ - \tag{12}$$

преобразуемый вектор $\boldsymbol{a}$ перед выполнением k-й итерации (11), $k \geq 0$.

На рис. 2 процесс (11) продемонстрирован геометрически.

Для выполнения вращения на базисный угол преобразуем матрицу (2):

$$R(\xi_k \Delta\alpha_k) = \begin{pmatrix} \cos\Delta\alpha_k & -\xi_k \sin\Delta\alpha_k \\ \xi_k \sin\Delta\alpha_k & \cos\Delta\alpha_k \end{pmatrix} = \cos\Delta\alpha_k \begin{pmatrix} 1 & -\xi_k\, tg\,\alpha_k \\ \xi_k\, tg\,\alpha_k & 1 \end{pmatrix} =$$

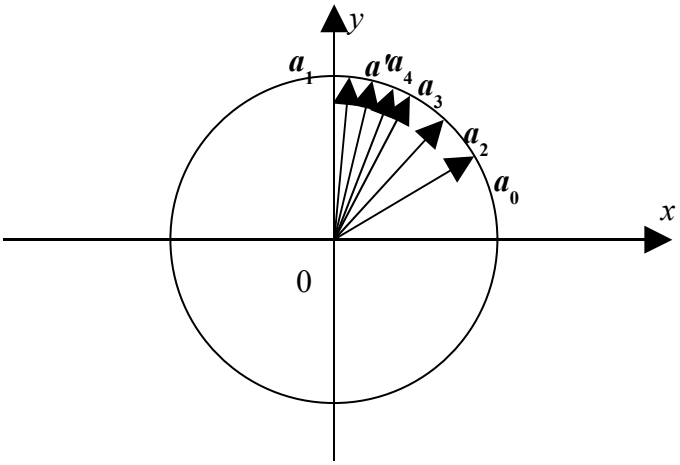

Рис. 2 – последовательность элементарных вращений

$$=\cos\Delta\alpha_k\begin{pmatrix}1 & -\xi_k 2^{-k}\\ \xi_k 2^{-k} & 1\end{pmatrix}=\cos\Delta\alpha_k R_k\ , \tag{13}$$

где:

$$R_k=\begin{pmatrix}1 & -\xi_k 2^{-k}\\ \xi_k 2^{-k} & 1\end{pmatrix}. \tag{14}$$

Обозначим:

$$\tilde{a_0}=a_0=a\ . \tag{15}$$

$$\tilde{a_{k+1}}=R_k\tilde{a_k}\ , \tag{16}$$

тогда

$$\tilde{a_k}=\left(\prod_{m=0}^{k-1}R_m\right)a=\left(\prod_{m=0}^{k-1}\frac{1}{\cos\Delta\alpha_m}R(\Delta\alpha_m)\right)a=\left(\prod_{m=0}^{k-1}\frac{1}{\cos\Delta\alpha_m}\right)\left(\prod_{m=0}^{k-1}R(\Delta\alpha_m)\right)a=$$

$$=\left(\prod_{m=0}^{k-1}\frac{1}{\cos\Delta\alpha_m}\right)a_k\ , \tag{17}$$

$$a'=\lim_{k\to\infty}a_k=\lim_{k\to\infty}\left(\prod_{m=0}^{k-1}\cos\Delta\alpha_m\right)\tilde{a_k}=\left(\prod_{m=0}^{\infty}\cos\Delta\alpha_m\right)\lim_{k\to\infty}\tilde{a_k}=\frac{1}{K}\lim_{k\to\infty}\tilde{a_k}\ , \tag{18}$$

где

$$K=\prod_{m=0}^{\infty}\frac{1}{\cos\Delta\alpha_m}=\prod_{m=0}^{\infty}\sqrt{1+2^{-2m}}\approx 1{,}646760\ \_ \tag{19}$$

коэффициент масштабирования.

Для вычисления ξ_k обозначим угол между текущим вектором $\boldsymbol{a_k}$ и искомым $\boldsymbol{a'}$ через $\tilde{\alpha_k}$. После поворота текущего вектора этот угол уменьшается на величину поворота:

$$\tilde{\alpha_{k+1}}=\tilde{\alpha_k}-\xi_k\Delta\alpha_k\ , \tag{20}$$

Данный угол должен из двух возможных значений принять то, что ближе к нулю:

$$\tilde{\alpha_{k+1}}=\begin{cases}\tilde{\alpha_k}-\Delta\alpha_k, если\ \tilde{\alpha_k}>0\\ \tilde{\alpha_k}+\Delta\alpha_k, если\ \tilde{\alpha_k}\le 0\end{cases}=\tilde{\alpha_k}-sign(\tilde{\alpha_k})\Delta\alpha_k\ , \tag{21}$$

то есть:

$$\xi_k=sign\,\tilde{\alpha_k}\ . \tag{22}$$

Таким образом, преобразование (1) заменяется итерационным алгоритмом:

$$
\begin{cases}
\begin{pmatrix} x \\ y \end{pmatrix}' = \frac{1}{K} \lim_{k\to\infty} \begin{pmatrix} \tilde{x}_k \\ \tilde{y}_k \end{pmatrix}, \\
K = \prod_{m=0}^{\infty} \sqrt{1+2^{-2m}}, \\
\begin{pmatrix} \tilde{x}_{k+1} \\ \tilde{y}_{k+1} \end{pmatrix} = \begin{pmatrix} 1 & -\xi_k 2^{-k} \\ \xi_k 2^{-k} & 1 \end{pmatrix} \begin{pmatrix} \tilde{x}_k \\ \tilde{y}_k \end{pmatrix}, \\
\begin{pmatrix} \tilde{x}_0 \\ \tilde{y}_0 \end{pmatrix} = \begin{pmatrix} x \\ y \end{pmatrix}, \\
\xi_k = sign\,\tilde{\alpha}_k, \\
\tilde{\alpha}_0 = \alpha, \\
\tilde{\alpha}_{k+1} = \tilde{\alpha}_k - \xi_k \Delta\alpha_k, \\
\Delta\alpha_k = arctg\,2^{-k}.
\end{cases}
\qquad (23)
$$

Сходимость алгоритма (23) следует из соотношения (8). На практике при вычислении (1) используют конечное число итераций (23), зависящее от необходимой точности и, как правило, связанное с разрядностью обрабатываемых значений. Если не учитывать деление на K, то все операции (23) являются простыми и легко реализуются аппаратурно (значения $\Delta\alpha_k$ являются константами, могут быть вычислены заранее и, например, храниться в ПЗУ). Что касается деления на K, то, во-первых, K является константой, не зависящей от входных значений x, y, α, поэтому деление на K может быть реализовано достаточно просто способами, которые будут рассмотрены ниже, а, во-вторых, в некоторых приложениях достаточно найти направление результирующего вектора без сохранения его модуля.

В основе одной итерации (23) лежит преобразование:

$$
\tilde{a}_{k+1} = R_k \tilde{a}_k \,, \qquad (24)
$$

называемое не масштабированной итерацией CORDIC.

Операционное устройство, выполняющее k-ю итерацию преобразует входные значения $\tilde{x}_k$, $\tilde{y}_k$ и $\tilde{\alpha}_k$ в выходные $\tilde{x}_{k+1}$, $\tilde{y}_{k+1}$ и $\tilde{\alpha}_{k+1}$ (рис. 3). При этом существует два основных способа реализации (23): последовательный и конвейерный. В первом случае, существует всего одно операционное устройство, которое в качестве входного значения принимает также номер итерации k и на котором выполняются последовательно все итерации (23). Во втором случае количество операционных устройств равно количеству выполняемых операций, а номер итерации k является внутренним параметром операционного устройства. Последовательное и конвейерное устройства различаются аппаратурными затратами и производительностью. В любом случае, процессор CORDIC, то есть устройство, реализующее алгоритм (23) имеет три входа – x, y, α и два выхода – $\tilde{x} = Kx'$ и $\tilde{y} = Ky'$ (рис. 4).

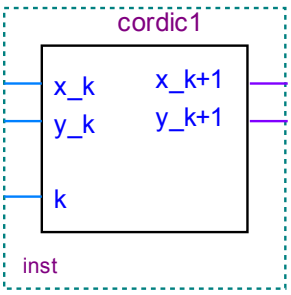

Рис. 3. Блок для реализации одной итерации CORDIC

Рассмотрим еще одну задачу, связанную с вращением вектора на плоскости: Выполнить поворот исходного вектора ***a*** таким образом, чтобы он совместился с

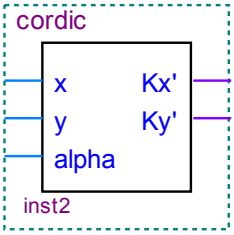

Рис. 4. Процессор CORDIC

горизонтальной осью координат, то есть $y' = 0$, при этом x' станет равным модулю исходного вектора $\boldsymbol{a}$.

Из (1) следует, что:

$$y' = x\sin\alpha + y\cos\alpha, \tag{25}$$

следовательно, для решения задачи необходимо решить уравнение:

$$x\sin\alpha + y\cos\alpha = 0 \tag{26}$$

относительно sinα и cosα, после чего построить из полученных значений матрицу и выполнить умножение матрицы на вектор, что является сложной задачей, трудно реализуемой аппаратурно. С другой стороны, вращение вектора, как было показано выше, реализуется алгоритмом CORDIC, пригодным для аппаратурной реализации. Чтобы применить алгоритм CORDIC для решения новой задачи, разложим неизвестный угол α в виде (3) и рассмотрим, каким образом меняется вторая компонента при выполнении одной итерации (23):

$$\tilde{y}_{k+1} = \tilde{y}_k + \xi_k 2^{-k} \tilde{x}_k \ . \tag{27}$$

Поскольку для приближения к оси x необходимо производить вращение по часовой стрелке, когда $\boldsymbol{a_k}$ находится в верхней полуплоскости, то есть при $\tilde{y}_k > 0$ и, наоборот, против часовой стрелки, когда $\tilde{y}_k \le 0$, а направление вращения определяется исключительно оператором ξ_k, нужно определять на каждой итерации $\xi_k = -\text{sign}\ \tilde{y}_k$.

Обозначив через α_k угол, на который уже повернут вектор $\boldsymbol{a}$ к началу k-й итерации, получаем алгоритм:

$$\begin{pmatrix} x \\ y \end{pmatrix}' = \begin{pmatrix} \sqrt{x^2 + y^2} \\ 0 \end{pmatrix} = \frac{1}{K} \lim_{k\to\infty} \begin{pmatrix} \tilde{x}_k \\ \tilde{y}_k \end{pmatrix},$$

$$K = \prod_{m=0}^{\infty} \sqrt{1 + 2^{-2m}},$$

$$\begin{pmatrix} \tilde{x}_{k+1} \\ \tilde{y}_{k+1} \end{pmatrix} = \begin{pmatrix} 1 & -\xi_k 2^{-k} \\ \xi_k 2^{-k} & 1 \end{pmatrix} \begin{pmatrix} \tilde{x}_k \\ \tilde{y}_k \end{pmatrix},$$

$$\begin{pmatrix} \tilde{x}_0 \\ \tilde{y}_0 \end{pmatrix} = \begin{pmatrix} x \\ y \end{pmatrix}, \tag{28}$$

$$\xi_k = -\,sign\,\tilde{y}_k,$$

$$\alpha_0 = 0,$$

$$\alpha_{k+1} = \alpha_k + \xi_k \Delta\alpha_k,$$

$$\Delta\alpha_k = arctg\, 2^{-k},$$

$$\lim_{k\to\infty} \alpha_k = arctg \frac{y}{x}.$$

Приведенный алгоритм называется векторным режимом или режимом приложения алгоритма CORDIC, а алгоритм (23) – режимом вращения или режимом вычисления.

Порядок выполнения работы :

1. Синтезировать схему арифметического устройства для выполнения одной итерации алгоритма CORDIC. При синтезе данной схемы и всех последующих возможно использование языка VHDL.
2. Разработать схему последовательного процессора CORDIC.
3. Разработать схему конвейерного процессора CORDIC. Сравнить производительность полученного устройства с предыдущим.
4*. Разработать схему последовательного или конвейерного процессора CORDIC в режиме векторизации.

Содержание отчета :

Протокол работы должен содержать схему арифметического устройства и схему синхронизации CORDIC, временные диаграммы работы устройства на одной итерации и при выполнения преобразований в целом в двух основных режимах. При реализации устройства средствами языка VHDL приводится текст программы.