ВЕСТНИК ТОМСКОГО ГОСУДАРСТВЕННОГО УНИВЕРСИТЕТА

2024 Управление, вычислительная техника и информатика № 69

Tomsk State University Journal of Control and Computer Science

Научная статья
УДК 621.382
doi: 10.17223/19988605/69/13

Реализация на современных ПЛИС вычислителя сигмоидной функции активации нейронных сетей табличным методом

Инна Владимировна Ушенина

Пензенский государственный технологический университет, Пенза, Россия, ivl23@yandex.ru

Аннотация. Вычисление функции сигмоида реализуется методом поразрядного отображения. В рамках этого метода аргументы и значения сигмоида представляются в двоичном коде в формате с фиксированной запятой. Каждый разряд значения сигмоида отделен от других и представляется в виде булевой функции от разрядов аргумента или ее таблицы истинности. Оцениваются возможности реализации вычислителей разрядов значений функции сигмоида на блоках программируемой логики ПЛИС. Анализируется два способа реализации: на основе таблиц истинности и на основе минимизированных булевых функций. Во всех реализованных схемах аргументы и значения функции сигмоида имеют равную друг другу разрядность. Схемы, реализованные по таблицам истинности, имеют разрядности от 6 до 11 бит. Показано, что вычислители отдельных разрядов значений функции сигмоида при 7- и 8-разрядном представлении аргумента могут размещаться всего на одном блоке ПЛИС и выполняют вычисления за наименьшее время. Предложенный вариант реализации вычислителя сигмоидной функции может использоваться в составе обученных нейронных сетей, реализуемых аппаратно.

Ключевые слова: нейронная сеть; сигмоид; ПЛИС; табличный метод.

***Благодарности*:** Исследование выполнено за счет гранта Российского научного фонда и Пензенской области № 24-21-20100, https://rscf.ru/project/24-21-20100/

***Для цитирования*:** Ушенина И.В. Реализация на современных ПЛИС вычислителя сигмоидной функции активации нейронных сетей табличным методом // Вестник Томского государственного университета. Управление, вычислительная техника и информатика. 2024. № 69. С. 124–133. doi: 10.17223/19988605/69/13

Original article
doi: 10.17223/19988605/69/13

Realization of the sigmoid activation function for neural networks on current FPGAs by the table-driven method

Inna V. Ushenina

Penza State Technological University, Penza, Russian Federation, ivl23@yandex.ru

Abstract. In the work, the sigmoid function is implemented using the bit-level mapping method. Within this method, inputs and outputs of the sigmoid function are represented in binary code in fixed-point format. Each output bit is separated from others and represented by a Boolean function of the input bits or its truth table. The possibilities of implementing sigmoid function output bit calculators on FPGA programmable logic blocks are assessed. Two implementation ways are analyzed: on the base of truth tables and on the base of minimized Boolean functions. All implemented circuits have equal bit widths of inputs and outputs to each other. The circuits based on truth tables have bit widths in the range of 6 to 11 bits. It is shown that the sigmoid output bit calculators of 7- and 8-bit inputs occupy just a single programmable logic block and make calculations in the shortest time. The proposed variant of the sigmoid function calculator can be used as a part of trained neural networks implemented in hardware.

Keywords: neural network; sigmoid function; FPGA; table-driven method.

***Acknowledgments*:** The research was carried out with the financial support of the Russian Science Foundation and Penza region under Grant № 24-21-20100, https://rscf.ru/project/24-21-20100/

***For citation*:** Ushenina, I.V. (2024) Realization of the sigmoid activation function for neural networks on current FPGAs by the table-driven method. *Vestnik Tomskogo gosudarstvennogo universiteta. Upravlenie, vychislitelnaja tehnika i informatika – Tomsk State University Journal of Control and Computer Science*. 69. pp. 124–133. doi: 10.17223/19988605/69/13

Введение

Обучение и развертывание нейронных сетей на аппаратной платформе с возможностью массивно-параллельных вычислений значительно экономит время вычислений, а также позволяет выполнять их в реальном масштабе времени. Доступной платформой, позволяющей осуществлять массивно-параллельные вычисления, являются ПЛИС.

В нейронных сетях типа «многослойный персептрон», обучающихся по алгоритму обратного распространения ошибки, в качестве функции активации нейронов часто используется сигмоид

$$s(x) = \frac{1}{1+e^{-x}}. \quad (1)$$

Эта функция имеет подходящую форму, непрерывно дифференцируема и требует небольших дополнительных аппаратных затрат на вычисление производной, нужной при обучении.

В настоящее время чаще всего используется три метода вычисления функции сигмоида на ПЛИС, каждый из которых позволяет обойти вычисления частного и экспоненты, напрямую не реализуемые на современных ПЛИС.

Наиболее распространена аппроксимация сигмоида, в частности кусочно-линейная [1–4], когда сигмоид заменяется набором аппроксимирующих отрезков. Недостатком метода является дополнительная погрешность вычислений; преимуществом при реализации на ПЛИС – небольшое количество умножителей или даже их отсутствие [1, 2].

Метод CORDIC (COordinate Rotation DIgital Computer) заключается в постепенном приближении к вычисляемому значению за счет последовательности поворотов вектора и уточнения его координат. Вычислитель такого типа содержит только регистры сдвига и устройства сложения / вычитания и не требует умножителей [5]. Метод CORDIC может обеспечить высокую точность результатов вычислений, но за счет большой латентности схемы, так как каждое уточнение результата требует дополнительной итерации вычислений.

В рамках табличного метода вычислитель представляет собой блок оперативной памяти, хранящий заранее рассчитанные значения функции [6–8]. На его адресные входы подаются значения аргумента. Преимущества метода – меньшая погрешность вычислений по сравнению с аппроксимацией, а также высокая скорость, связанная как с отсутствием латентности, так и с тем, что время доступа к памяти, как правило, меньше времени вычисления. Однако с увеличением разрядности вычислений требуется все больший объем памяти, которой ПЛИС имеют весьма ограниченные запасы.

Разновидность табличного метода – метод поразрядного отображения [9]. В рамках этого метода аргументы и значения сигмоида представляются в двоичном коде в формате с фиксированной запятой. Каждый разряд значения сигмоида отделяется от других и представляется в виде булевой функции от разрядов аргумента или ее таблицы истинности. Разрядность аргумента влияет на размеры таблиц истинности и сложность булевых функций. Применительно к реализации на ПЛИС поразрядное отображение дает преимущество по сравнению со стандартным табличным методом: при работе с отдельными разрядами можно освободить блочную память ПЛИС и реализовать вычисления на распределенной памяти – табличных преобразователях (Look-Up Tables, LUT), которые имеются на ПЛИС в большом количестве в составе блоков программируемой логики.

В [9] и последовавших за ней работах [10–12] отдельные разряды значений функций активации вычисляются по булевым функциям, представленным в минимальной дизъюнктивной нормальной форме (МДНФ). Схемы вычислителей имеют разрядность не более 7 бит, при которой булевы функции достаточно просты и реализуются на малом количестве ресурсов. В [13] показано, что в обученной нейронной сети разрядность вычислителя функции сигмоида должна быть не менее 7–8 бит, так

как именно при этих разрядностях происходит резкое снижение среднеквадратической ошибки выходных сигналов в слоях нейронной сети. Там же показано, что обучение нейронной сети и достижение «жесткой» сходимости, то есть условия

$$\left|t_j - x_{L,j}\right| \le 2^{-4}. \quad (2)$$

требует разрядности вычислений функции сигмоида не менее 9–10 бит. Здесь t_j – целевое значение j-го нейрона выходного слоя сети, $x_{L,j}$ – значение выходного сигнала j-го нейрона выходного слоя сети, L – количество слоев нейронной сети.

Очевидно, что повышение разрядности вычислений значительно усложняет булевы функции и может сделать схемы вычислителей практически нереализуемыми из-за возросшей ресурсоемкости.

Для нейронной сети, развернутой или обучающейся аппаратно, важны также время вычислений и анализ вклада в это время различных влияющих факторов: разрядности вычислителя, времени работы логических элементов, задержек распространения сигналов по соединениям между ними. В работах [9, 11, 12] ресурсоемкость и время вычислений приводятся для 7-битных вычислителей; кроме того, функцией активации не всегда является сигмоид, а целевым устройством – ПЛИС.

В данной статье представлены и проанализированы схемы вычислителей отдельных разрядов значений функции сигмоида (далее – вычислителей) разрядностью от 6 до 11 бит, работающие методом поразрядного отображения и реализованные на блоках программируемой логики ПЛИС. Анализируется два способа реализации: на основе таблиц истинности и на основе минимизированных булевых функций в МДНФ. Для каждой схемы получены характеристики ресурсоемкости, такие как требуемое количество блоков программируемой логики, табличных преобразователей, мультиплексоров, и временны́е характеристики, включая общее время вычисления и задержку распространения сигналов по соединениям между логическими элементами.

Реализация и анализ схем вычислителей выполнены для двух ПЛИС – XC7A200 семейства Artix7 от Xilinx и GW2AR семейства Arora от Gowin Semiconductor. Ресурсоемкость и время вычисления оценивались с использованием встроенных средств пакетов Vivado IDE и Gowin EDA.

1. Ресурсы современных ПЛИС и вычисление функции сигмоида

Основной ресурс ПЛИС – массив блоков программируемой логики. Название таких блоков зависит от производителя: у Xilinx/AMD это конфигурируемые логические блоки (Configurable Logic Block, CLB), у Gowin – конфигурируемые блоки вычисления функций (Configurable Function Unit, CFU) и т.д. Для реализации комбинационных схем эти блоки содержат следующие ресурсы:

1. LUT – небольшие блоки оперативной памяти, позволяющие реализовать таблицу истинности или имитировать работу комбинационной логики при вычислении булевых функций. Входные сигналы LUT являются адресом нужной ячейки, где хранится 1 бит информации. В современных ПЛИС количество входов у LUT обычно равно четырем или шести.

2. 2-входовые мультиплексоры, которые соединены с выходами LUT.

3. Логика переноса: 2-входовые мультиплексоры и 2-входовые элементы «исключающее ИЛИ».

Поскольку вычисления выполняются отдельно для каждого разряда значения функции сигмоида, 1-битная распределенная память, т.е. LUT, и примыкающие к ней мультиплексоры могут удобно использоваться для реализации схем вычислителей.

Так как сигмоид – симметричная функция, здесь рассматривается только вычисление разрядов ее значений при неотрицательных значениях аргумента, находящихся в диапазоне [0; 8). Вычисления при значениях аргумента в диапазоне (–8; 0] могут быть выполнены аналогично или с использованием схемы вычитания, работающей согласно формуле

$$s(x) = 1 - s(|x|). \quad (3)$$

В настоящей работе этот вопрос не рассматривается.

В работе представлены схемы вычислителей разрядностью от 6 до 11 бит. Во всех схемах аргументы и значения функции имеют равную друг другу разрядность. В представлении аргументов из

всех имеющихся разрядов три старших представляют целую часть, остальные – дробную часть. В представлении значений функции сигмоида, находящихся при неотрицательных аргументах в диапазоне [0,5; 1), все имеющиеся разряды представляют дробную часть.

Таблицы истинности для реализации схем вычислителей первым способом составлялись по таблицам аргументов сигмоида и соответствующих им значений, представленных в двоичном коде с нужной разрядностью. По полученным таблицам истинности определялись кодовые слова, записываемые в LUT. Булевы функции в МДНФ для реализации схем вычислителей вторым способом были получены по соответствующим им таблицам истинности с использованием алгоритма Куайна–МакКласки. Описание всех схем выполнено на языке VHDL.

2. Реализация вычислителей по таблицам истинности (первый способ)

Во многих ПЛИС LUT и мультиплексоры могут быть соединены, как показано на рис. 1, с получением структур, удобных для реализации вычислителей по таблицам истинности. При этом адресными входами LUT будут 4 или 6 младших разрядов аргумента функции сигмоида, а управляющими входами мультиплексоров – оставшиеся старшие разряды.

CFU ПЛИС GW2AR содержит восемь 4-входовых LUT (см. рис. 1, *a*), значит на одном CFU может быть размещена не более чем 7-разрядная таблица истинности. Для реализации 8-разрядной таблицы истинности понадобится два CFU (рис. 2), на которых размещается примитив 8-разрядного табличного преобразователя LUT8. Это возможно, так как каждый CFU содержит 8 мультиплексоров.

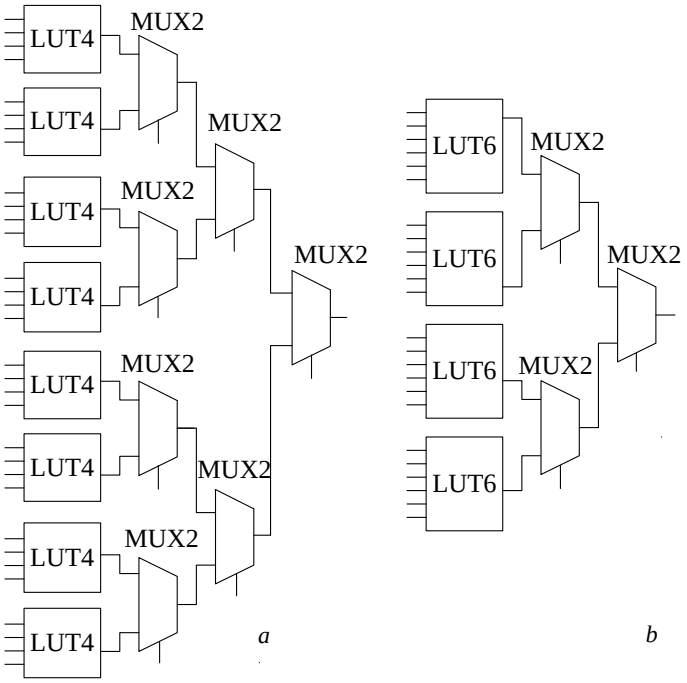

Рис. 1. LUT и мультиплексоры в блоках программируемой логики: 4-входовые LUT и 2-входовые мультиплексоры в CFU ПЛИС GW2AR (*a*) и 6-входовые LUT и 2-входовые мультиплексоры в CLB ПЛИС XC7A200 (*b*)
Fig. 1. LUTs and multiplexers in programmable logic blocks: 4-input LUTs and 2-input multiplexers in CFUs of GW2AR FPGAs (*a*) and 6-input LUTs and 2-input multiplexers in CLBs of XC7A200 FPGAs (*b*)

Для вычислителей большей разрядности нужно несколько примитивов LUT8, выходы которых должны поступать на мультиплексор. В табл. 1 представлены характеристики схем вычислителей разрядностью 6–11 бит, реализованных на GW2AR по таблицам истинности. Вычислители разрядностью 6 и 7 бит умещаются в один CFU, и задержки распространения сигнала по внутренним соединениям CFU оказываются равными нулю. 8-разрядный вычислитель реализован на двух CFU и, вероятно, соединение между ними имеет ненулевую задержку, но для примитивов (в данном случае для LUT8)

временной анализатор сообщает только общее время задержки (в данном случае 1,638 нс). Вычислители с разрядностью 9, 10 и 11 бит содержат соответственно 2, 4 и 8 примитивов LUT8, выходы которых поступают соответственно на 2-, 4- и 8-входовой мультиплексор, реализованный еще на одном CFU.

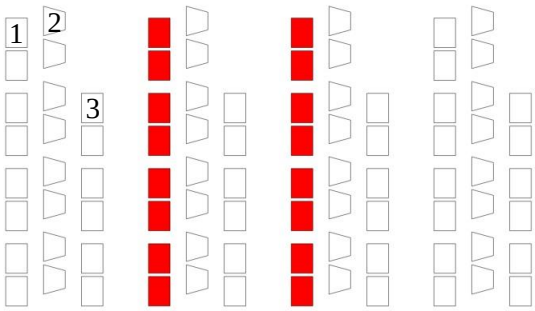

Рис. 2. Результат размещения на двух CFU 8-разрядной таблицы истинности. Фрагмент изображения кристалла GW2AR взят из редактора FloorPlanner среды Gowin EDA. Показаны 4 CFU. *1* – LUT4, *2* – MUX2, *3* – регистры. Занятые LUT выделены редактором. Занятые мультиплексоры редактором не выделяются

Fig. 2. Result of mapping an 8-bit truth table on two CFUs. The piece of GW2AR device image has been taken from Gowin EDA FloorPlanner. Four CFUs are shown. *1* – LUT4, *2* – MUX2, *3* – registers. The occupied LUTs are colored by FloorPlanner. The occupied multiplexers remain uncolored in FloorPlanner

Таблица 1

Характеристики вычислителей, реализованных по таблицам истинности на ПЛИС GW2AR

Разрядность, бит	Кол-во CFU	Общее время вычисления, нс	Задержка по соединениям, нс
6	1	0,780	0
7	1	0,864	0
8	2	1,638	–
9	5	1,887	0,403
10	9	2,369	0,679
11	17	3,092	0,645

CLB ПЛИС XC7A200 содержат четыре 6-входовых LUT и три мультиплексора (см. рис. 1, *b*), значит один CLB позволит реализовать не более чем 8-разрядную таблицу истинности. На рис. 3 представлен результат размещения на ресурсах CLB ПЛИС XC7A200 8-разрядной таблицы истинности, которая реализована на четырех примитивах LUT6 и трех мультиплексорах. Для реализации 9-, 10- и 11-разрядных таблиц истинности потребуется соответственно 3, 5 и 9 CLB, из которых на одном CLB реализован 2-, 4- и 8-разрядный мультиплексор, а остальные реализуют 8-разрядные таблицы истинности от младших разрядов, как показано на рис. 3.

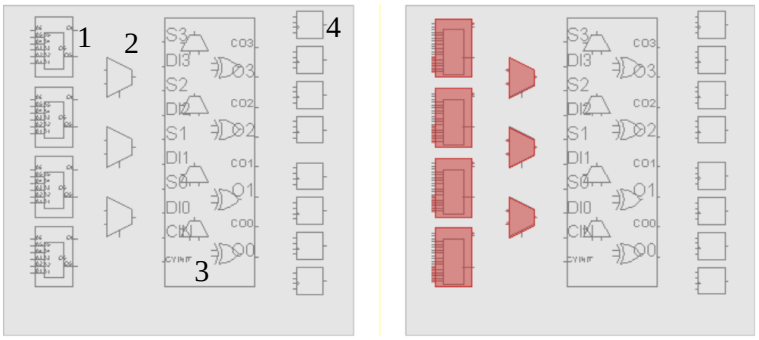

Рис. 3. Результат размещения на одном CLB 8-разрядной таблицы истинности. Фрагмент изображения кристалла XC7A200 взят из редактора FloorPlanner Vivado IDE. Показано 2 CLB. *1* – LUT6, *2* – MUX2, *3* – логика переноса, *4* – регистры. Занятые LUT и мультиплексоры выделены редактором

Fig. 3. Result of mapping an 8-bit truth table on a single CLB. The piece of XC7A200 device image has been taken from Vivado IDE FloorPlanner. Two CLBs are shown. *1* – LUT6, *2* – MUX2, *3* – carry logic, *4* – registers. The occupied LUTs and multiplexers are colored by FloorPlanner

Таблица 2

Характеристики вычислителей, реализованных по таблицам истинности на ПЛИС XC7A200

Разрядность, бит	Кол-во CLB	Общее время вычисления, нс	Задержка по соединениям, нс
6	1	0,199	0
7	1	0,257	0
8	1	0,324	0
9	3	1,134	0,55
10	5	1,294	0,710
11	9	1,231	0,5

Характеристики схем вычислителей с разрядностью 6–11 бит, реализованных на ПЛИС XC7A200 по таблицам истинности, представлены в табл. 2. Вычислители разрядностью 6–8 бит занимают один CLB, причем выходы LUT соединяются только с входами мультиплексоров. Задержки, вносимые этими соединениями в работу вычислителя, равны нулю. Задержки распространения сигналов по соединениям, указанные в табл. 2 для вычислителей разрядностью более 8 бит, соответствуют соединениям между CLB.

3. Реализация вычислителей по минимизированным булевым функциям (второй способ)

Из табл. 1 и 2 видно, что, когда вычислители перестают помещаться в одном блоке программируемой логики, увеличение точности на один разряд приводит к увеличению ресурсоемкости более чем в два раза. На примере 9-разрядных вычислителей проверим, удастся ли снизить ресурсоемкость схем, если реализовывать их на основе минимизированных булевых функций в МДНФ, представленных в табл. 3. Старший, 9-й разряд аргумента функции сигмоида в табл. 3 обозначен буквой *a*, 8-й разряд – буквой *b* и т.д. Первый после запятой разряд значения функции сигмоида обозначен *s*1, второй – *s*2 и т.д. При неотрицательных значениях аргумента *s*1 всегда равен единице. Разряд *s*2 представляют 6 импликант; разряд *s*3 – 12; далее количество импликант постепенно нарастает и доходит до 64 для разряда *s*9.

Таблица 3

МДНФ булевых функций для вычисления разрядов значений сигмоида с точностью 9 бит

Разряд	Функция
*s*2	$a + b + cd + ce + cf + cghi$
*s*3	$a + b + \bar{c}di + \bar{c}dh + \bar{c}dg + \bar{c}df + \bar{c}de + c\bar{d}\bar{e}\bar{f}\bar{g} + c\bar{d}\bar{e}\bar{f}\bar{h} + c\bar{d}\bar{e}\bar{f}\bar{i} + defgi + defgh$
*s*4	$a + bc + bde + \bar{b}\bar{c}\bar{d}ei + \bar{b}\bar{c}\bar{d}eh + \bar{b}\bar{c}eg + \bar{b}\bar{c}ef + \bar{b}d\bar{e}\bar{f}\bar{g}\bar{h}\bar{i} + c\bar{e}\bar{f}\bar{g} + c\bar{e}\bar{f}\bar{h} + c\bar{e}\bar{f}\bar{i} + cd\bar{e} +$ $+bdfgh + cd\bar{f} + cd\bar{g} + \bar{b}\bar{d}efgh + cdh\bar{i} + \bar{b}\bar{c}ehi$
…	…
*s*9	$abc + abd + abe + acd\bar{e} + a\bar{b}\bar{c}\bar{f}ghi + a\bar{b}\bar{c}efh + a\bar{b}defg\bar{i} + a\bar{b}d\bar{e}\bar{h}\bar{i} + \bar{b}cd\bar{e}ghi + a\bar{b}\bar{c}de\bar{f}\bar{g}i + bcde\bar{f}\bar{g}i + a\bar{c}d\bar{e}\bar{f}gh +$ $+a\bar{c}efg\bar{h}\bar{i} + a\bar{c}d\bar{e}gh\bar{i} + b\bar{c}defhi + bcd\bar{e}\bar{f}h\bar{i} + bc\bar{e}gh\bar{i} + bcde\bar{g}h + bcd\bar{e}gh + \bar{b}\bar{c}d\bar{f}h + \bar{b}\bar{c}d\bar{f}g + a\bar{b}\bar{c}d\bar{e}\bar{f}g\bar{i} +$ $+a\bar{c}d\bar{e}\bar{f} + acd\bar{f}g + acd\bar{e}\bar{f}g\bar{i} + a\bar{c}e\bar{f}g\bar{h} + b\bar{c}d\bar{e}h + a\bar{b}d\bar{g}\bar{h}i + b\bar{c}d\bar{e}f\bar{g} + a\bar{c}fgh\bar{i} + a\bar{b}\bar{c}d\bar{e}f + a\bar{b}\bar{c}de\bar{f}hi + a\bar{b}d\bar{f}g\bar{h}\bar{i} +$ $+bef\bar{g}hi + a\bar{c}de\bar{f}\bar{g} + b\bar{c}e\bar{f}g\bar{h} + a\bar{c}d\bar{f}\bar{g}i + cd\bar{e}\bar{f}\bar{g}hi + acd\bar{f}i + \bar{c}d\bar{e}\bar{f}g\bar{i} + bef\bar{g}h\bar{i} + bcef\bar{g}h + a\bar{b}d\bar{e}\bar{f}gh + a\bar{b}d\bar{e}gh\bar{i} +$ $+a\bar{c}e\bar{f}g\bar{h} + a\bar{b}\bar{e}\bar{f}gh\bar{i} + a\bar{b}d\bar{e}\bar{f}g\bar{i} + a\bar{b}d\bar{e}efh\bar{i} + a\bar{c}d\bar{e}egh\bar{i} + bde\bar{f}gh\bar{i} + bde\bar{f}gh\bar{i} + a\bar{b}d\bar{e}\bar{f}ghi + bcd\bar{e}\bar{f}h\bar{i} +$ $+a\bar{b}\bar{c}e\bar{f}g\bar{h}i + a\bar{b}d\bar{e}\bar{f}ghi + acd\bar{f}h + \bar{c}d\bar{e}\bar{f}hi + a\bar{d}e\bar{g}\bar{h}i + bd\bar{e}\bar{f}gh + a\bar{c}e\bar{f}h\bar{i} + a\bar{b}\bar{c}e\bar{f}g\bar{i} + a\bar{c}d\bar{f}g\bar{h}i + bcd\bar{e}\bar{f}g\bar{h} + a\bar{d}\bar{e}\bar{f}ghi$

Vivado IDE синтезирует схемы по булевым функциям, используя минимальное количество ресурсов. На рис. 4, *a–c* показаны синтезированные схемы вычисления разрядов *s*2–*s*5, которые помещаются в один CLB ПЛИС XC7A200 и реализованы только на LUT. Так, вычисление *s*2 может быть выполнено всего на двух LUT: LUT6, расположенный на рис. 4, *a* слева, вычисляет функцию $d + e + f + ghi$, а LUT6, расположенный справа, умножает эту функцию на *c* и складывает ее с *a* и *b*. Вычисление *s*3 требует трех LUT; вычисление *s*4 – четырех. Вычисление *s*5 выполняется такой же схемой, что и *s*4.

Схема, синтезированная для разряда *s*7, представлена на рис. 4, *d*. Схема для разряда *s*6 отличается от представленной на рис. 4, *d* тем, что два из шести LUT обрабатывают пять, а не шесть входных сигналов. Разряды *s*8 и *s*9 вычислить, используя менее трех CLB, не удалось.

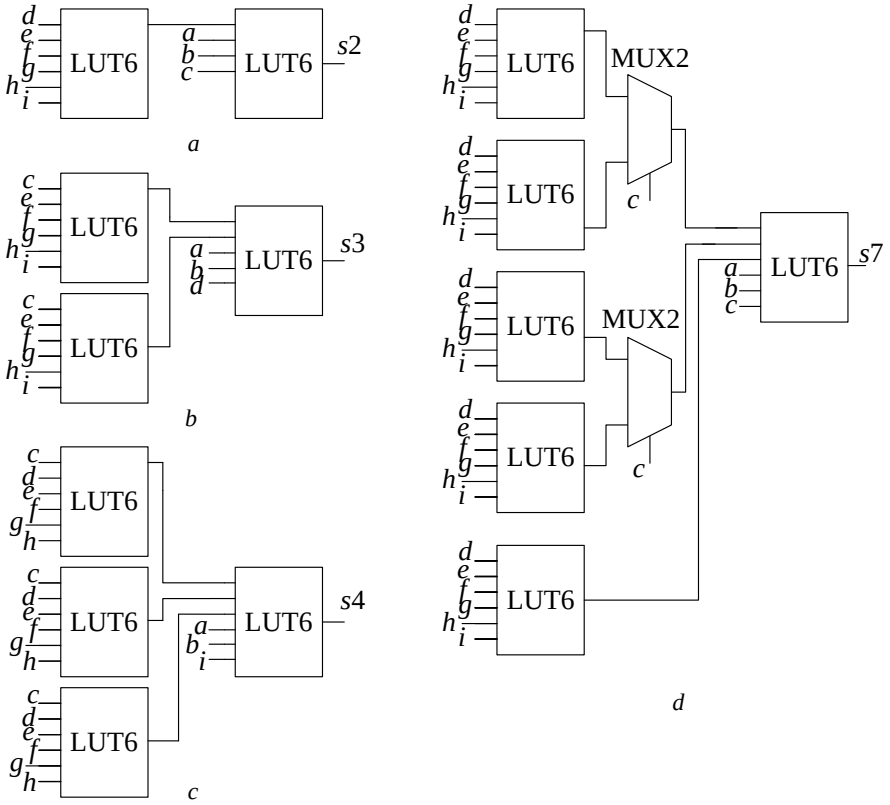

Рис. 4. Схемы вычисления разрядов *s*2–*s*7 значений функции сигмоида по минимизированным булевым функциям, синтезированные для ПЛИС XC7A200

Fig. 4. Circuits for calculating the sigmoid *s*2–*s*7 bits using the minimized Boolean functions, synthesized for the XC7A200 FPGA

Характеристики вычислителей разрядов *s*2–*s*7, синтезированных по булевым функциям, представлены в табл. 4. Общую ресурсоемкость вычислителей разрядов *s*2–*s*9 удалось снизить с 24 до 14 CLB, т.е. на 42% по сравнению со схемами, выполненными по таблицам истинности.

Таблица 4

Характеристики вычислителей, реализованных по булевым функциям в МДНФ на XC7A200

Разряд	Ресурсоемкость			Время вычисления	
	LUT6	MUX2	CLB	Общее	Задержка распространения по соединениям, нс
*s*2	2	0	1	0,509	0,315
*s*3	3	0	1	0,509	0,315
*s*4	4	0	1	0,695	0,501
*s*5	4	0	1	0,695	0,501
*s*6	6	2	2	0,682	0,214
*s*7	6	2	2	0,779	0,285

Соединения между LUT внутри CLB ПЛИС XC7A200 могут быть реализованы только с использованием внешних трассировочных ресурсов. Поэтому задержки по соединениям для вычислителей разрядов *s*2–*s*5, размещающихся в одном CLB, отличны от нуля. Тем не менее общее время вычисления в схемах, представленных на рис.4, оказывается на 31–55% меньше, чем в схемах, реализованных по таблицам истинности, за счет снижения задержек, вносимых логическими элементами.

Для ПЛИС GW2AR реализация вычислителей по булевым функциям возможна менее чем на пяти CFU для разрядов *s*2–*s*7. Схемы вычислителей разрядов *s*2 и *s*3 на ПЛИС GW2AR (рис. 5) отли-

чаются от тех, что были получены для XC7A200. Вычислители *s*4–*s*7 реализованы в соответствии со схемами, представленными на рис. 4, *c*, *d*.

Результаты реализации и временного анализа вычислителей *s*2–*s*7 на ПЛИС GW2AR приведены в табл. 5. По сравнению с первым способом удалось добиться снижения количества используемых CFU на 42%, что совпадает с результатом, достигнутым на ПЛИС XC7A200. Заметим, что для вычисления *s*6 требуется три CFU, а для *s*7 – четыре, так как в схеме для *s*6 два из пяти LUT обрабатывают пять, а не шесть переменных, а значит, требуют меньше ресурсов.

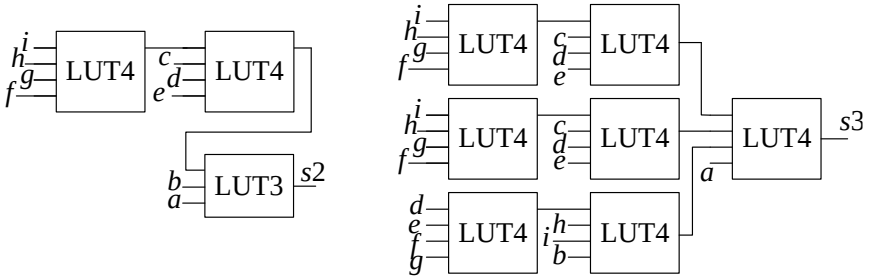

Рис. 5. Схемы вычисления разрядов *s*2 и *s*3 значений функции сигмоида по минимизированным булевым функциям, полученные для ПЛИС GW2AR

Fig. 5. Circuits for calculating the sigmoid *s*2 and *s*3 bits using the minimized Boolean functions, received for the GW2AR FPGA

Таблица 5

Характеристики вычислителей, реализованных по булевым функциям в МДНФ на GW2AR

Разряд	Ресурсоемкость			Время вычисления	
	LUT4	MUX2	CFU	Общее	Задержка распространения по соединениям, нс
*s*2	3	0	1	1,378	0,005
*s*3	7	0	1	1,623	0,148
*s*4	16	12	2	1,690	0,270
*s*5	16	12	2	1,690	0,270
*s*6	22	14	3	2,312	0,598
*s*7	26	18	4	2,288	0,617

Что касается временны́х характеристик, достигнуть существенного выигрыша при переходе ко второму способу реализации вычислителей не удалось. Для разрядов *s*6 и *s*7 время вычисления оказалось даже больше, чем при реализации вычислителей первым способом. Это частично объясняется тем, что при реализации по таблицам истинности пути, по которым проходят данные, пролегают через три CFU, но на двух из них размещен примитив LUT8, и вносимая им задержка минимизирована производителем. То есть произвольно относительно друг друга могут быть размещены LUT8 (на двух CFU) и мультиплексор (на одном CFU), между которыми – одно соединение. При переходе ко второму способу часть путей проходит через три CFU, которые могут располагаться произвольно относительно друг друга, и задержка распространения по соединениям оказывается выше. Также задержка распространения через цепочку из трех примитивов LUT6–мультиплексор–LUT6 при втором способе (см. рис. 4, *d*) оказывается больше, чем через LUT8-мультиплексор при первом.

Реализация вторым способом вычислителей с разрядностью 10 бит позволяет добиться меньшей экономии ресурсов и времени. Из-за усложнения булевых функций общее снижение ресурсоемкости при переходе ко второму способу вычисления разрядов *s*2–*s*6 составляет около 30%. Время вычисления оказывается меньше при использовании второго способа только для разрядов *s*2 и *s*3; вычисление остальных разрядов обоими способами требует примерно равного времени.

Заключение

Вычислители отдельных разрядов значений функции сигмоида при 7- и 8-разрядном представлении аргумента могут размещаться всего на одном блоке программируемой логики и выполняют

вычисления за наименьшее время. Это создает хорошую перспективу для их использования в составе обученных нейронных сетей, реализуемых аппаратно.

Вычисления с точностью 9–10 бит, нужные при обучении нейронных сетей, требуют больше ресурсов и времени. При этом вычисление старших разрядов значений функции сигмоида по представляющим их минимизированным булевым функциям, а не по таблицам истинности, требует меньше логических ресурсов и может сократить время вычислений.

Список источников

1. Alippi C., Storti-Gajani G. Simple approximation of sigmoidal functions: realistic design of digital neural networks capable of learning // Proc. 1991 IEEE International Symposium on Circuits and Systems (ISCAS). Singapore, 1991. V. 3. P. 1505–1508.
2. Amin H., Curtis K.M., Hayes-Gill B.R. Piecewise linear approximation applied to nonlinear function of a neural network // IEE Proc. – Circuits, Devices and Systems. 1997. V. 144 (6). P. 313–317. doi: 10.1049/ip-cds:19971587
3. Tatas K., Gemenaris M. High-Performance and Low-Cost Approximation of ANN Sigmoid Activation Functions on FPGAs // Proc. 12th International Conference on Modern Circuits and Systems Technologies (MOCAST). Athens, Greece, 2023. P. 1–4.
4. Шашев Д.В., Шатравин В.В. Реализация сигмоидной функции активации с помощью концепции перестраиваемых вычислительных сред // Вестник Томского государственного университета. Управление, вычислительная техника и информатика. 2022. № 61. С. 117–127. doi: 10.17223/19988605/61/12
5. Chen H., Jiang L., Luo Y., Lu Z., Fu Y., Li L., Yu Z. A CORDIC-Based Architecture with Adjustable Precision and Flexible Scalability to Implement Sigmoid and Tanh Functions // Proc. 2020 IEEE International Symposium on Circuits and Systems (ISCAS). Seville, Spain, 2020. P. 1–5.
6. Sartin M.A., da Silva A.C.R. Approximation of hyperbolic tangent activation function using hybrid methods // Proc. 8th International Workshop on Reconfigurable and Communication-Centric Systems-on-Chip (ReCoSoC). Darmstadt, Germany, 2013. P. 1–6.
7. Saranya S., Elango B. Implementation of PWL and LUT based approximation for hyperbolic tangent activation function in VLSI // Proc. 2014 International Conference on Communication and Signal Processing. Melmaruvathur, India, 2014. P. 1778–1782.
8. Xie Y., Raj A.N.J., Hu Z., Huang S., Fan Z., Joler M. A twofold lookup table architecture for efficient approximation of activation functions // IEEE Transactions on Very Large Scale Integration (VLSI) Systems. 2020. V. 28 (12). P. 2540–2550. doi: 10.1109/TVLSI.2020.3015391
9. Tommiska M.T. Efficient digital implementation of the sigmoid function for reprogrammable logic // IEE Proc. – Computers and Digital Techniques. 2003. V. 150 (6). P. 403–411. doi: 10.1049/ip-cdt:20030965
10. Li X.J., Li L. IP core based hardware implementation of multi-layer perceptrons on FPGAs: a parallel approach // Advanced Materials Research. 2012. V. 433. P. 5647–5653. doi: 10.4028/www.scientific.net/AMR.433-440.5647
11. Yang T., Wei Y., Tu Z., Zeng H., Kinsy M.A., Zheng N., Ren P. Design Space Exploration of Neural Network Activation Function Circuits // IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems. 2019. V. 38 (10). P. 1974–1978. doi: 10.1109/TCAD.2018.2871198
12. Rajput G., Raut G., Chandra M., Vishvakarma S.K. VLSI implementation of transcendental function hyperbolic tangent for deep neural network accelerators // Microprocessors and Microsystems. 2021. V. 84. Art. 104270. doi: 10.1016/j.micpro.2021.104270
13. Holt J.L., Hwang J.N. Finite precision error analysis of neural network hardware implementations // IEEE Transactions on Computers. 1993. V. 42 (3). P. 281–290. doi: 10.1109/12.210171

References

1. Alippi, C. & Storti-Gajani, G. (1991) Simple approximation of sigmoidal functions: realistic design of digital neural networks capable of learning. *Proceedings 1991 IEEE International Symposium on Circuits and Systems (ISCAS)*. 3. pp. 1505–1508.
2. Amin, H., Curtis, K.M. & Hayes-Gill, B.R. (1997) Piecewise linear approximation applied to nonlinear function of a neural network. *IEE Proceedings-Circuits, Devices and Systems*. 144(6). pp. 313–317. DOI: 10.1049/ip-cds:19971587
3. Tatas, K. & Gemenaris, M. (2023) High-Performance and Low-Cost Approximation of ANN Sigmoid Activation Functions on FPGAs. *Proceedings of the 12th International Conference on Modern Circuits and Systems Technologies (MOCAST)*. Athens. Greece. pp. 1–4.
4. Shashev, D.V. & Shatravin, V.V. (2022) Implementation of the sigmoid activation function using the reconfigurable computing environments. *Vestnik Tomskogo gosudarstvennogo universiteta. Upravlenie, vychislitelnaya tehnika i informatika – Tomsk State University Journal of Control and Computer Science*. 61. pp. 117–127. DOI: 10.17223/19988605/61/12
5. Chen, H., Jiang, L., Luo, Y., Lu, Z., Fu, Y., Li, L. & Yu, Z. (2020) A CORDIC-Based Architecture with Adjustable Precision and Flexible Scalability to Implement Sigmoid and Tanh Functions. *Proceedings 2020 IEEE International Symposium on Circuits and Systems (ISCAS)*. Seville. Spain. pp. 1–5.
6. Sartin, M.A. & da Silva, A.C.R. (2013) Approximation of hyperbolic tangent activation function using hybrid methods. *Proceedings 8th International Workshop on Reconfigurable and Communication-Centric Systems-on-Chip (ReCoSoC)*. Darmstadt. Germany. pp. 1–6.

7. Saranya, S. & Elango, B. (2014) Implementation of PWL and LUT based approximation for hyperbolic tangent activation function in VLSI. *Proceedings 2014 International Conference on Communication and Signal Processing.* Melmaruvathur. India. pp. 1778–1782.
8. Xie, Y., Raj, A.N.J., Hu, Z., Huang, S., Fan, Z. & Joler, M. (2020) A twofold lookup table architecture for efficient approximation of activation functions. *IEEE Transactions on Very Large-Scale Integration (VLSI) Systems*. 28(12). pp. 2540–2550. DOI: 10.1109/TVLSI.2020.3015391
9. Tommiska, M.T. (2003) Efficient digital implementation of the sigmoid function for reprogrammable logic. *IEE Proceedings-Computers and Digital Techniques*. 150(6). pp. 403–411. DOI: 10.1049/ip-cdt:20030965
10. Li, X.J. & Li, L. (2012) IP core-based hardware implementation of multi-layer perceptrons on FPGAs: a parallel approach. *Advanced Materials Research*. 433. pp. 5647–5653. DOI: 10.4028/www.scientific.net/AMR.433-440.5647
11. Yang, T., Wei, Y., Tu, Z., Zeng, H., Kinsy, M.A., Zheng, N. & Ren, P. (2019) Design Space Exploration of Neural Network Activation Function Circuits. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*. 38(10). pp. 1974–1978. DOI: 10.1109/TCAD.2018.2871198
12. Rajput, G., Raut, G., Chandra, M., & Vishvakarma, S.K. (2021) VLSI implementation of transcendental function hyperbolic tangent for deep neural network accelerators. *Microprocessors and Microsystems*. 84. pp. 104270. DOI: 10.1016/j.micpro.2021.104270
13. Holt, J.L. & Hwang, J.N. (1993) Finite precision error analysis of neural network hardware implementations. *IEEE Transactions on Computers*. 42(3). pp. 281–290. DOI: 10.1109/12.210171

Информация об авторе:
Ушенина Инна Владимировна – доцент, кандидат технических наук, доцент кафедры «Программирование» Пензенского государственного технологического университета (Пенза, Россия). E-mail: ivl23@yandex.ru

Авторы заявляет об отсутствии конфликта интересов.

Information about the author:
Ushenina Inna V. (Candidate of Technical Sciences, Associate Professor, Penza State Technological University, Penza, Russian Federation). E-mail: ivl23@yandex.ru

The author declares no conflicts of interests.

Received 20.07.2024; accepted for publication 02.12.2024

Поступила в редакцию 20.07.2024; принята к публикации 02.12.2024