Инженерный вестник Дона, №8 (2023)
ivdon.ru/ru/magazine/archive/n8y2023/8560

Анализ вычислений значений с плавающей точкой на микроконтроллерах

О.В. Игнатьева, А.Д. Сокирка, Д.С. Журавлев

Ростовский государственный университет путей сообщения

Аннотация: В статье рассмотрены методы оптимизации вычислений с плавающей точкой на микроконтроллерных устройствах. Рассмотрены аппаратный и программный способы ускорения вычислений. Приведены алгоритмы Карацубы и Шёнхаге-Штрассена для операции умножения. Предложен способ замены вычислений с плавающей точкой на целочисленные вычисления. Описано использование фиксированной точки вместо плавающей. Рассмотрен вариант использования хэш-памяти и оптимизации кода. Представлены результаты измерения вычислений на микроконтроллере AVR.
Ключевые слова: вычисления с плавающей точкой, вычисления с фиксированной точкой, микроконтроллер, AVR, ARM.

Введение

В современном мире вычисления с плавающей точкой являются одним из самых распространенных способов обработки данных [1]. Они используются во многих приложениях, включая научные и инженерные расчеты, компьютерную графику, финансы и многое другое.

Однако, вычисления с плавающей точкой требуют большого количества ресурсов, таких, как оперативная память, процессорное время и энергия [2, 3]. В микроконтроллерах, которые имеют ограниченные ресурсы, это может стать проблемой.

Анализ проблемы

Число с плавающей запятой (или число с плавающей точкой) — экспоненциальная форма представления вещественных (действительных) чисел, в которой число хранится в виде мантиссы и порядка (показателя степени) [4]. При этом, число с плавающей запятой имеет фиксированную относительную точность и изменяющуюся абсолютно. Реализация математических операций с числами с плавающей запятой в вычислительных системах может быть, как аппаратная, так и программная. На рис. 1 изображены разряды числа с плавающей запятой.

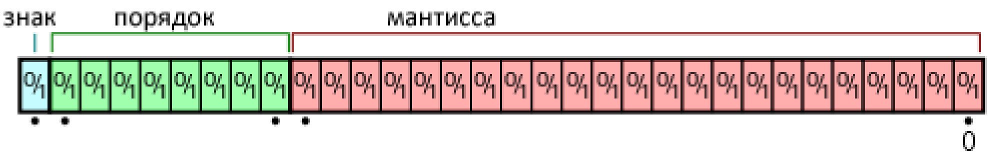

Рис. 1 – Разряды числа с плавающей запятой

Аппаратное решение

Многие современные микроконтроллеры имеют аппаратное ускорение для вычислений с плавающей точкой. Например, микроконтроллеры семейства ARM Cortex-M4 имеют FPU-блок, который может выполнять операции с плавающей точкой с высокой скоростью. Это позволяет значительно ускорить вычисления, по сравнению с программным выполнением операций [5].

Модуль с плавающей запятой (FPU, в просторечии математический сопроцессор) является частью компьютерной системы, специально разработанной для выполнения операций с числами с плавающей запятой. Типичными операциями являются сложение, вычитание, умножение, деление и извлечение квадратного корня. Некоторые FPU также могут выполнять различные трансцендентные функции, такие, как экспоненциальные или тригонометрические вычисления, но точность может быть очень низкой, так что некоторые системы предпочитают вычислять эти функции программно.

Однако, не все микроконтроллеры имеют аппаратное ускорение для вычислений с плавающей точкой. В таких случаях, можно использовать специальные библиотеки, которые реализованы на уровне аппаратуры и позволяют ускорить выполнение операций с плавающей точкой [6, 7].

Алгоритмы Карацубы и Шёнхаге-Штрассена

Выбор оптимального алгоритма может значительно повлиять на скорость вычислений с плавающей точкой [8, 9]. Например, для умножения двух чисел с плавающей точкой можно использовать стандартный алгоритм умножения, который требует выполнения нескольких операций умножения и сложения. Однако, существуют более эффективные алгоритмы, такие, как алгоритм Карацубы или алгоритм Шёнхаге-Штрассена, которые могут значительно ускорить выполнение операций умножения [10]. Сравнение вычисления умножения в столбик и при помощи алгоритма Карацубы, приведены на рис. 2.

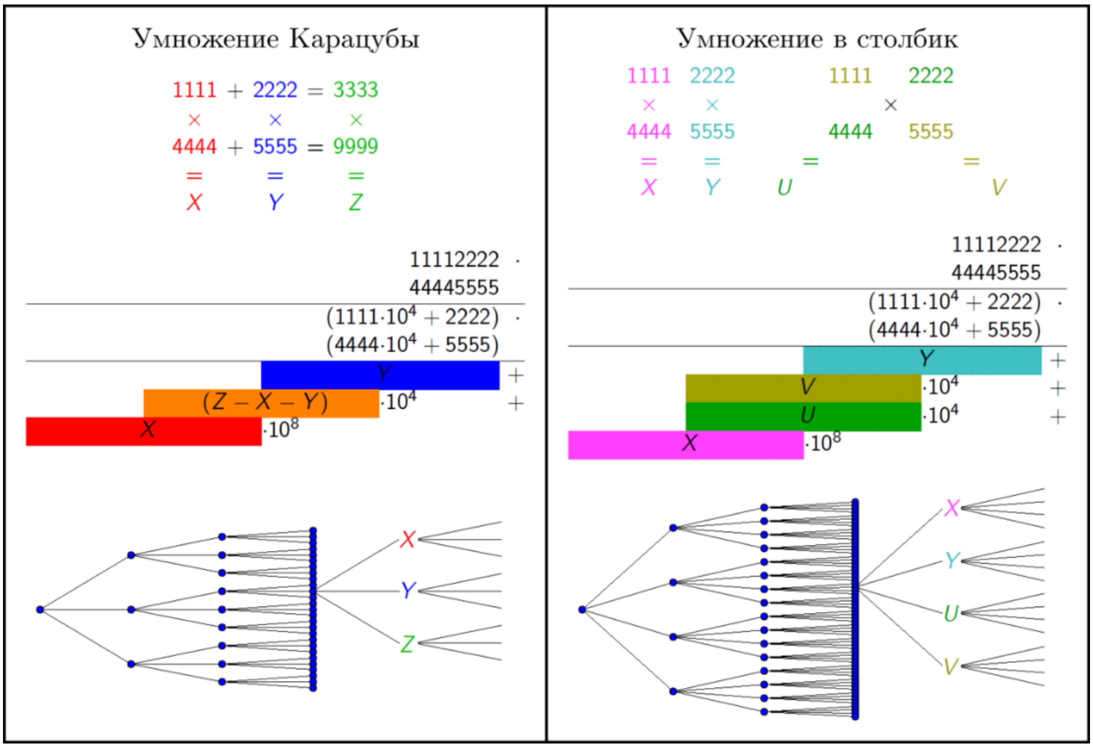

Рис.2 – Сравнение умножения в столбик и алгоритма Карацубы

На рис. 3 представлен алгоритм Шёнхаге-Штрассена.

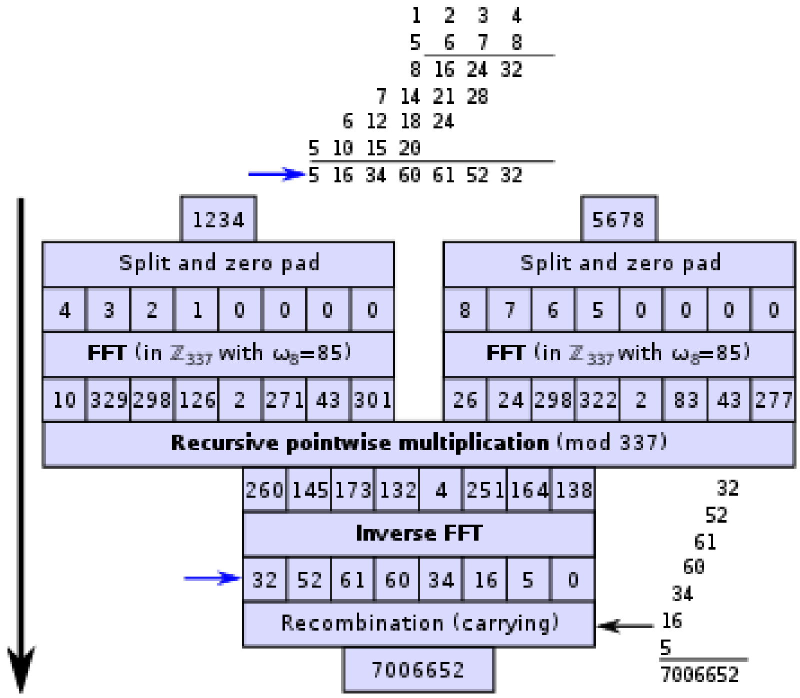

Рис. 3 – Алгоритм Шёнхаге-Штрассена

Способ замены типа вычислений

В качестве одного из способов ускорения вычислений с плавающей точкой на микроконтроллере используют вместо вычислений вещественных значений, преобразуя такие вычисления в целочисленные.

Такого результата можно добиться несколькими способами. Во-первых, вместо использования числа 0.3 следует использовать аналогичное целочисленное значение, например, 3 или 30. Однако данный подход будет удобен для человека, привыкшего пользоваться десятичной системой счисления.

Для машины более предпочтительным вариантом станет использование ограничений в виде степени 2. Так, например, диапазон от 0 до 1 будет эквивалентен диапазону от 0 до 255.

В таблице 1 представлены замеры математических операций с разными типами данных для микроконтроллера семейства AVR с тактовой частотой 16МГц.

Таблица №1

Скорость выполнения операций с разными типами данных

Тип данных	Время выполнения, мкс		
	Сложение и вычитание	Умножение	Деление, остаток
int8_t	0.44	0.625	14.25
uint8_t	0.44	0.625	5.38
int16_t	0.89	1.375	14.25
uint16_t	0.89	1.375	13.12
int32_t	1.75	6.06	38.3
uint32_t	1.75	6.06	37.5
float	8.125	10	31.5

Таким образом, теоретически данное решение может предоставить ускорение работы в 18.46 раза при в случае с вычислением операций сложения или вычитания.

Использование фиксированной точки

Еще одним способом увеличения скорости вычислений с плавающей точкой на микроконтроллере является использование значений с фиксированной точкой, вместо значений с плавающей точкой [11, 12].

Фиксированная точка – это формат представления чисел, при котором дробная часть числа имеет фиксированную позицию [13]. Например, если мы используем формат с фиксированной точкой с 4 битами для дробной части, то число 3.25 будет представлено как 0011.0100.

Использование фиксированной точки может значительно ускорить вычисления с плавающей точкой. Это связано с тем, что операции с фиксированной точкой могут выполняться с помощью целочисленных операций, которые работают быстрее, чем операции с плавающей точкой.

Однако, использование фиксированной точки также имеет свои недостатки. В частности, это ограничивает диапазон значений, которые можно представить. Кроме того, использование фиксированной точки может привести к потере точности в результате вычислений.

В таблице 2 приведены временные интервалы на различные операции для значений с фиксированной точкой.

Таблица № 2

Скорость выполнения операций с фиксированной точкой

Операция	Описание операции	Время выполнения, мкс
Val = 1.1	Присваивание float	0.75
Val1 = CONVERT(1.1)	Конвертация float в fixed (число)	0.75
Val2 = CONVERT(Val)	Конвертация float в fixed (переменная)	14.9
Val3 = Val1 + Val4	Сложение float	8.25
Val5 = Val1 + Val2	Сложение fixed	2.0
Val3 = Val1 * Val4	Сложение float	10.3
Val5 = FIXMUL(Val1, Val2)	Сложение fixed	6.68
Val3 = RECONVERT(Val5)	Конвертация fixed в float	13.37

Использование кэш-памяти

Кэш-память – это специальный вид памяти, который используется для временного хранения данных, которые часто запрашиваются. Использование кэш-памяти может значительно ускорить доступ к данным, поскольку они находятся в более быстрой памяти, чем оперативная память.

Для вычислений с плавающей точкой, которые требуют доступа к большим объемам данных, можно использовать кэш-память для временного хранения результатов промежуточных вычислений. Это может ускорить выполнение операций и снизить нагрузку на оперативную память.

Оптимизация кода

При работе с вычислениями с плавающей точкой, важно обратить внимание на оптимизацию кода, который выполняет операции с плавающей точкой.

Оптимизация кода – это процесс улучшения программного кода с целью повышения его производительности [14]. Существует множество способов оптимизации кода, таких, как удаление лишних операций, использование инлайн-функций и другие.

Например, можно использовать оптимизированные библиотеки математических функций, которые были специально разработаны для микроконтроллеров.

Выводы

Вычисления с плавающей точкой являются неотъемлемой частью многих приложений, но при работе с микроконтроллерами они могут стать проблемой из-за ограниченных ресурсов. В этой статье было рассмотрено несколько способов повышения скорости вычисления значений с плавающей точкой на микроконтроллерах, включая использование аппаратного ускорения, оптимизированных алгоритмов, фиксированной точки, кэш-памяти и оптимизации кода. Каждый из этих методов имеет свои преимущества и недостатки, и выбор конкретного метода зависит от требований к приложению и доступных ресурсов микроконтроллера.

Литература

1. Ющенко Р.А., Ющенко А.К. Особенности арифметики с плавающей запятой в современных компьютерах // Компьютерная математика. 2016. № 1. С. 80-92.

2. Bailey D.H., Borwein J.M. High-precision arithmetic in mathematical physics // Mathematics. 2015. 3(2). Pp. 337-367.

3. Молчанов И.Н. Машинная математика. Проблемы и перспективы // Кибернетика и системный анализ. 2004. № 6. С. 65-72.

4. Goldberg D. What Every Computer Scientist Should Know About Floating-Point Arithmetic //ACM Computing Surveys. 1991. Vol. 1, N 23. Pp. 5 – 48.

5. Floating-point unit demonstration on STM32 microcontrollers. 2016. URL: st.com/resource/en/application_note/an4044-floating-point-unit-demonstration-on-stm32-microcontrollers-stmicroelectronics.pdf

6. Раков И.Е. Повышение точности вычислений с плавающей точкой при помощи современных пакетов для научных вычислений // Сборник статей по итогам Международной научно-практической конференции «Теории, школы и концепции устойчивого развития науки в современных условиях». 2020. С. 9-15.

7. Кулямин, В.В. Стандартизация и тестирование реализаций математических функций, работающих с числами с плавающей точкой // Программирование. 2007. Вып. 33, №3. С. 44-72.

8. Стригунов В.В., Тарасенко Р.С. О точности вычислений с плавающей точкой // ТОГУ-СТАРТ: Фундаментальные и прикладные исследования молодых. Тихоокеанский государственный университет. 2020. С. 179-185.

9. Bjorndalen J.M., Anshus O.J. Trusting floating point benchmarks – are you benchmarks really data independent? // Proceeding of the 8th international

conference on Applied parallel computing: state of the art in scientific computing, 2006. Pp. 178-188.

10. Bürgisser P., Clausen M., Shokrollahi A. Algebraic Complexity Theory. Berlin: Springer-Verlag. 1997. 618 p.

11. Павлов О.В., Кейно П.П., Русак М.А., Малафеев И.В., Скрипниченко Ю.С. Повышение производительности микроконтроллеров за счёт использования типов данных с фиксированной точкой // Цифровая экономика. 2022. № 4 (20). С. 26-36.

12. Jian Chu, Jun W., Xie-he H., Sheng C. Optimization of block-floating-point realizations for digital controllers with finite-word-length considerations // Journal of Zhejiang University-SCIENCE A. 2003. pp. 651-657.

13. IEEE Standard for Floating-Point Arithmetic - Redline, in IEEE Std 754-2019 (Revision of IEEE 754-2008) - Redline, vol., no., pp.1-148, 22 July 2019. URL: ieeexplore.ieee.org/document/8866810.

14. Степович-Цветкова Г.С. Классификация способов оптимизации программного кода // Сборник научных трудов по материалам XIV Международной научно-практической конференции «Актуальные вопросы научных исследований». 2017. С. 5-8.

References

1. YUshchenko R.A., YUshchenko A.K. Komp'yuternaya matematika. 2016. № 1. Pp. 80-92.

2. Bailey D.H., Borwein J.M. High-precision arithmetic in mathematical physics. Mathematics. 2015. 3(2). Pp. 337-367.

3. Molchanov I.N. Kibernetika i sistemnyj analiz. 2004. № 6. Pp. 65-72.

4. Goldberg D. ACM Computing Surveys. 1991. Vol. 1, N 23. Pp. 5 – 48.

5. Floating point unit demonstration on STM32 microcontrollers. 2016. URL: st.com/resource/en/application_note/an4044-floating-point-unit-demonstration-on-stm32-microcontrollers-stmicroelectronics.pdf

6. Rakov I.E. Sbornik statej po itogam Mezhdunarodnoj nauchno-prakticheskoj konferencii «Teorii, shkoly i koncepcii ustojchivogo razvitiya nauki v sovremennyh usloviyah». 2020. Pp. 9-15.

7. Kulyamin, V.V. Programmirovanie. 2007. Vol. 33, №3. C. 44-72.

8. Strigunov V.V., Tarasenko R.S. TOGU-START: Fundamental'nye i prikladnye issledovaniya molodyh. Tihookeanskij gosudarstvennyj universitet. 2020. Pp. 179-185.

9. Bjorndalen J.M., Anshus O.J. Proceeding of the 8th international conference on Applied parallel computing: state of the art in scientific computing, 2006. Pp. 178-188.

10. Bürgisser P., Clausen M., Shokrollahi A. Algebraic Complexity Theory. Berlin: Springer-Verlag. 1997. 618 p.

11. Pavlov O.V., Kejno P.P., Rusak M.A., Malafeev I.V., Skripnichenko YU.S. Cifrovaya ekonomika. 2022. № 4 (20). Pp. 26-36.

12. Jian Chu, Jun W., Xie-he H., Sheng C. Journal of Zhejiang University-SCIENCE A. 2003. pp. 651-657.

13. IEEE Standard for Floating-Point Arithmetic - Redline, in IEEE Std 754-2019 (Revision of IEEE 754-2008) - Redline, vol., no., pp.1-148, 22 July 2019. URL: ieeexplore.ieee.org/document/8866810.

14. Stepovich-Cvetkova G.S. Sbornik nauchnyh trudov po materialam XIV Mezhdunarodnoj nauchno-prakticheskoj konferencii «Aktual'nye voprosy nauchnyh issledovanij». 2017. Pp. 5-8.