

МЕТОДЫ ПРОГРАММНОЙ И АППАРАТНОЙ СОРТИРОВКИ ЧИСЕЛ

Д.В. Кутузов, А.В. Осовский, А.Д. Киселева

В статье изложены основные методы сортировки чисел. Описаны варианты программной и аппаратной реализации, проведен анализ наиболее популярных решений данной задачи. В работе приведены схемы, основанные на анализе патентно-технической информации за последние годы.

Алгоритмы сортировки – одна из наиболее исследованных областей информатики, в которой, однако, не исключены открытия, так как, несомненно, существуют еще неизвестные методы, основанные на новых принципах и идеях. Можно с уверенностью утверждать, что любой инженер, специализирующийся в области информационных технологий, сталкиваясь с подобными задачами, пытаясь реализовать известные методы или разработать свои.

Определяя формально термин «сортировка», можно сказать, что это процесс упорядочения множества элементов, на котором заданы отношения порядка $<$, $>$, или отношения частичного порядка¹ $\geq$, $\leq$. Или, проще говоря, упорядочение множества элементов по возрастанию или убыванию. В большинстве задач сортировка сводится к упорядочению массива двоичных кодов, которые для наглядности могут быть представлены числами.

Алгоритмы сортировки имеют и большое практическое применение. Их можно встретить почти везде, где речь идет об обработке и хранении больших объемов информации. Многие задачи обработки данных решаются намного проще, если данные упорядочены.

В данной работе приводится обзор наиболее популярных программных и аппаратных методов сортировки. Здесь и далее предполагается, что числа или символы в виде двоичных кодов находятся в оперативной памяти ЭВМ или элементах памяти устройства сортировки.

Наиболее популярным методом программной сортировки является сортировка простым обменом (пузырьковая сортировка), что объясняется простотой алгоритма для понимания и реализации. Свое название этот способ упорядочивания данных получил благодаря схожести с процессом движения пузырьков воздуха в воде. Алгоритм состоит в повторяющихся проходах по сортируемому массиву, каждый из которых включает в себя попарное сравнение и обмен элементов в случае обнаружения инверсии. Отсортированный массив выступает признаком успешного завершения работы алгоритма, проходящего по нему от начала до конца. По итогам каждого прохода, как минимум, один элемент массива устанавливается на верную позицию, что обуславливает равенство между количеством элементов и числом необходимых циклических проходов.

Другим простым способом сортировки считается сортировка выбором². В соответствии с этим методом в сортируемом массиве поэтапно осуществляется выбор минимального (максимального) элемента и вставка его в начало (конец) последовательности. Реализация алгоритма требует выполнения двух циклов обработки данных. Внешний цикл выполняет проход по элементам, назначая минимум (максимум) и устанавливая на верную позицию найденный внутренним циклом, по поиску минимума (максимума) элемент. В завершение работы алгоритма происходит обмен двух последних элементов.

Не менее известным представителем класса простых сортировок является сортировка включением. Её алгоритм не только прост в реализации, но и эффективен на небольших, частично отсортированных наборах данных. Согласно данному методу первоначально упо-

рядочиваются два первых элемента последовательности. Далее на каждом шаге необходимо осуществлять выбор элемента и его вставку на нужную позицию в отсортированной части массива, без нарушения отношения порядка. Ощутимое преимущество сортировки включением состоит в том, что элементы с одинаковыми ключами не переставляются: если список элементов сортируется с использованием двух ключей, то после завершения сортировки вставкой он по-прежнему будет упорядочен по двум ключам.

Алгоритм сортировки, подразумевающий сравнение элементов, расположенных на различных расстояниях друг от друга, называют сортировкой Шелла. В основу метода положена сортировка включениями. По мере реализации алгоритма первоначально осуществляется сравнение и сортировка между собой ключей, отстоящих друг от друга на некотором расстоянии R , после чего процедура повторяется для некоторых меньших значений R . В заключение проводится упорядочивание элементов при $R = 1$, что полностью эквивалентно обычной сортировке вставками. Выбор значений расстояния R между сравниваемыми элементами может осуществляться по-разному. Для корректной работы метода необходимым условием является лишь то, что последний шаг должен равняться единице. Программирование алгоритма выполняется с использованием двух циклов. Внешний цикл проходит по всем заданным элементам, а внутренний включает два условия проверки: для упорядочения элементов и для предотвращения выхода за пределы массива. Если при каждом проходе используется небольшое число элементов или они находятся в относительном порядке, данный способ сортировки будет сравнительно эффективен.

Наиболее эффективным из алгоритмов обменной сортировки массивов является быстрая сортировка, в основу которой положен принцип разбиения. Это распространенный способ сортировки общего назначения, который хорошо работает во многих ситуациях и использует при этом меньше ресурсов, чем другие алгоритмы. Суть метода заключается в выборе для сравнения одного элемента X и разбиении массива входных данных на две части. Одну часть составляют все элементы, равные или большие X , а в другую часть входят все элементы меньшего значения. Этот процесс необходимо рекурсивно продолжать для оставшихся частей до тех пор, пока весь массив не будет отсортирован. Существуют различные способы выбора значения разбиения X : случайный выбор, выбор среднего значения, выбор медианы и т.д. В целях достижения наименьших временных затрат требуется выбрать значение, которое будет находиться на центральной позиции в массиве.

Довольно часто возникают и задачи аппаратной сортировки. Так, например, в системах радиолокации и робототехнике при поиске двух ближайших точек на плоскости требуется сортировка координат объектов. Используется сортировка и в целом ряде других приложений. Аппаратные решения, реализующие алгоритмы сортировки, зачастую близки к программным методам, хотя и имеют гораздо большее быстродействие, нежели программные, но все же достаточно специфичны и редко описываются в литературе.

Наиболее очевидными схемотехническими решениями³ можно считать устройства, принцип работы которых описан ниже.

На рис. 1 представлено устройство для выбора максимального числа. Оно включает в себя три компаратора, логический блок и мультиплексор. Устройство позволяет выбирать максимальное из трех чисел a , b и c , т.е. упорядочивает множество $\{a, b, c\}$, реализуя функцию $f : \max\{a, b, c\}$.

В зависимости от соотношения чисел a , b и c на входы логического блока поступают сигналы d , m , n , которые формируются в соответствии с таблицей (см. табл.).

Сигналы u , v , w , формируемые логическим блоком, открывают соответствующие группы ключей с третьим состоянием, в результате на выходы устройства передается максимальное число.

Аналогично строится схема быстрой сортировки чисел. В ней вместо компараторов используются арифметико-логические устройства (ALU), выходы «Перенос» (C) которых соединены с адресными входами (AI) постоянного запоминающего устройства (ROM), которое выполняет функции логического блока. Выводы данных (DO) ПЗУ соединены с адресными входами (AI) четырех мультиплексоров (MUX). Несомненным достоинством этой

схемы является возможность параллельной сортировки чисел, так как на выходах мультиплексоров они появятся одновременно, упорядоченные по убыванию или возрастанию. К недостаткам схемы можно отнести большие аппаратные затраты.

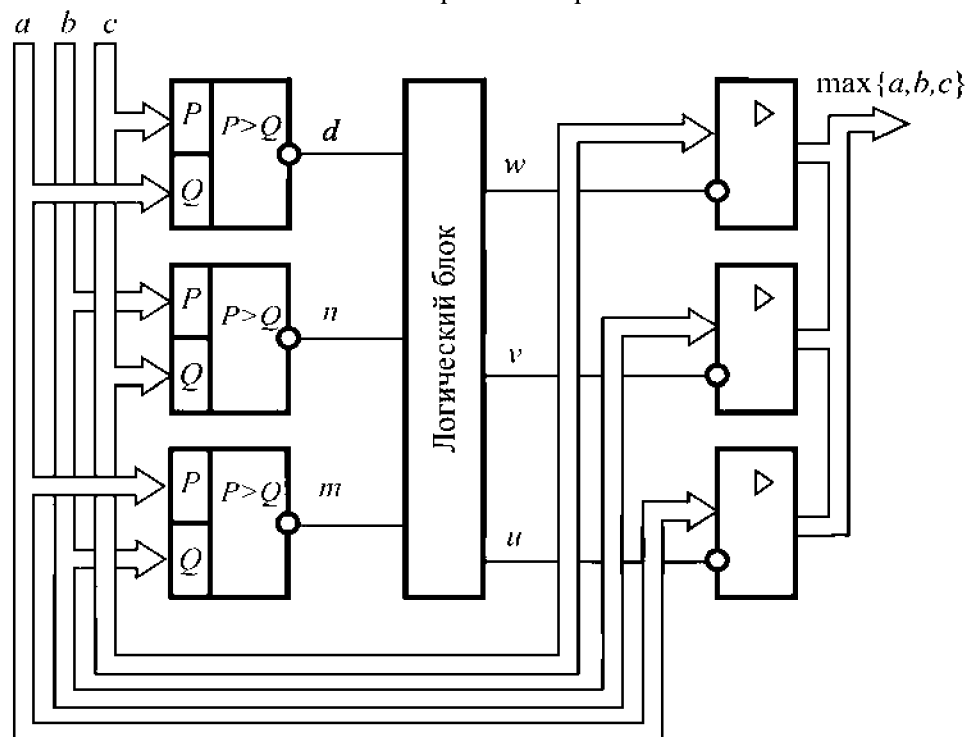


Рис. 1. Устройство для выбора максимального из трех чисел

Таблица

Сигналы во внутренних точках устройства							
Соотношение чисел a, b, c	Логические уровни						f
	n	m	d	u	v	w	
$a > b > c$	0	0	1	0	1	1	$f = a$
$a > c > b$ $a > c = b$ $a = c > b$	1	0	1	0	1	1	
$a = b = c$	1	1	1	0	1	1	
$b > c > a$	0	1	0	1	0	1	
$b > a = c$ $b > a > c$ $b = a > c$	0	1	1	1	0	1	$f = b$
$c > a > b$	1	0	0	1	1	0	$f = c$
$c > b > a$ $c > b = a$ $c = b > a$	1	1	0	1	1	0	

Так, например, количество элементов сравнения (АЛУ) определяется как количество сочетаний из n чисел по два:

$$C(n, 2) = \frac{n!}{2 \cdot (n-2)!} \quad (1)$$

Поэтому при возрастании количества сортируемых чисел резко возрастает количество компараторов (или АЛУ) и разрядность ПЗУ.

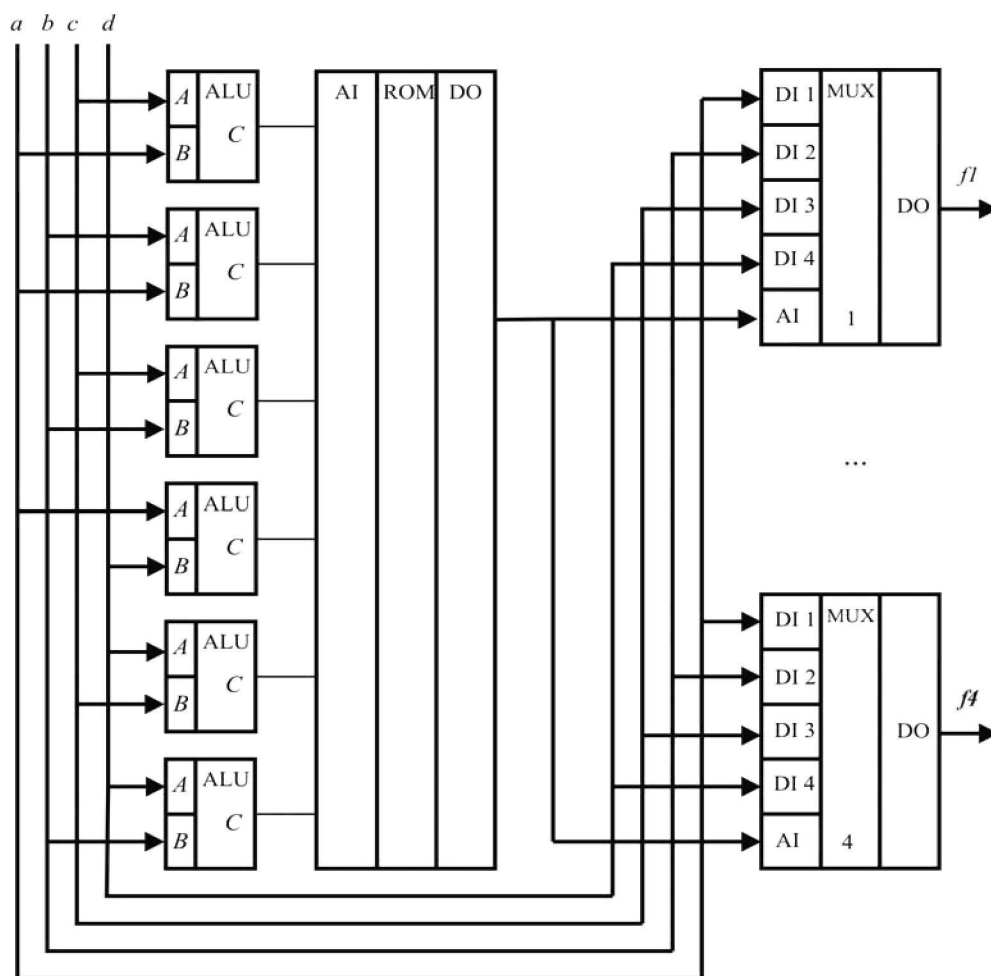


Рис. 2. Устройство для быстрой сортировки чисел

Число компараторов в схеме, представленной на рис. 2, можно сократить, если перестроить схему в виде пирамиды (см. рис. 3). Однако параллельность процессов сравнения чисел сохранится лишь частично. В случае использования этой схемы сортировки будет необходимо реализовывать *n* тактов работы схемы, где *n* – количество чисел для сортировки.

Схема пирамидальной сортировки (рис. 3) состоит из блоков выбора наибольшего/наименьшего числа (рис. 4).

Каждый блок выбора (рис. 4) в зависимости от режима сортировки выбирает одно из двух чисел – наибольшее или наименьшее. Он состоит из схемы сравнения, логического блока и буферов с третьим состоянием, включение/отключение которых производится сигналом (*CS*). Режим сортировки определяется сигналом (*max/min*).

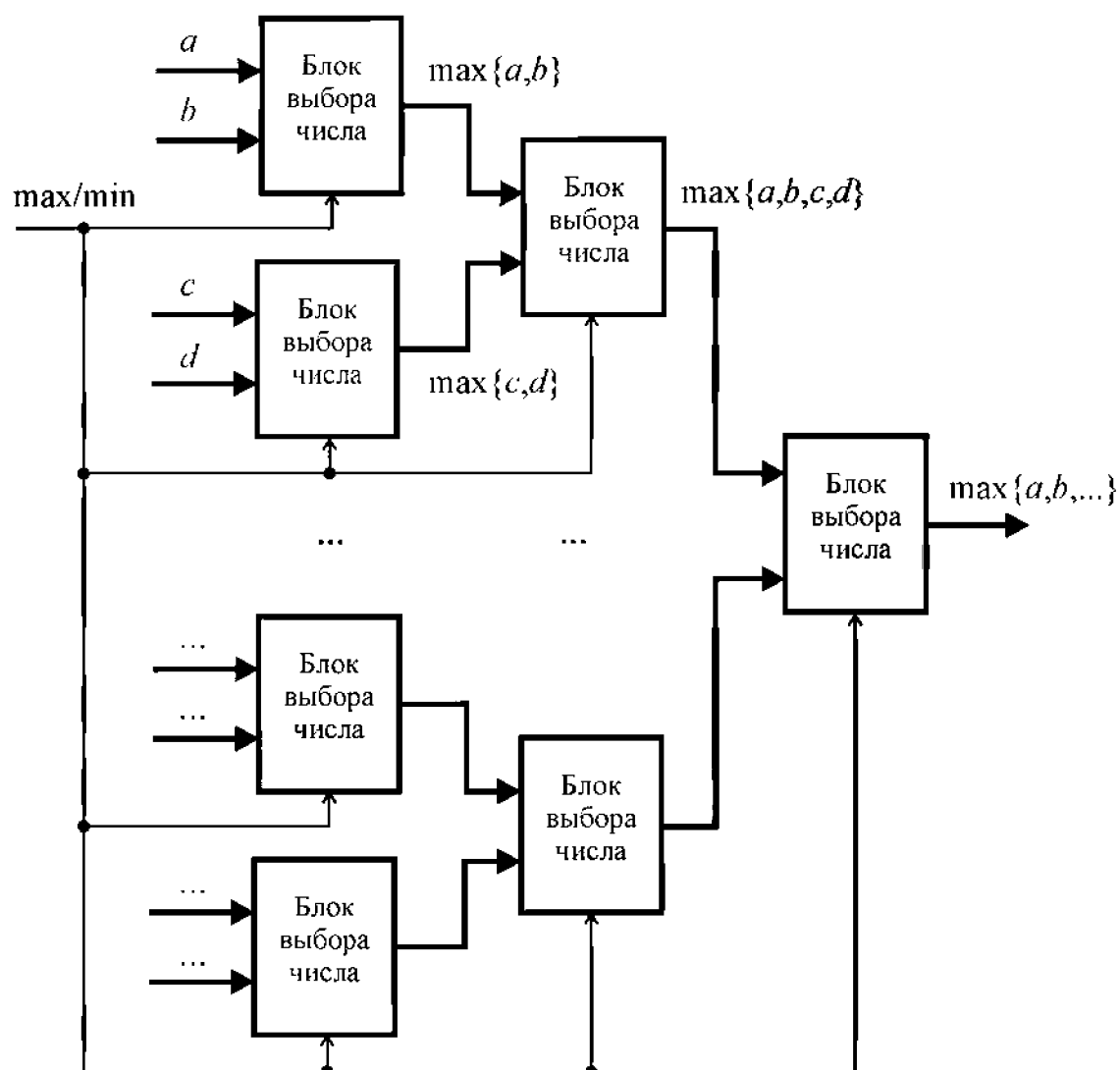


Рис. 3. Схема пирамидальной сортировки чисел

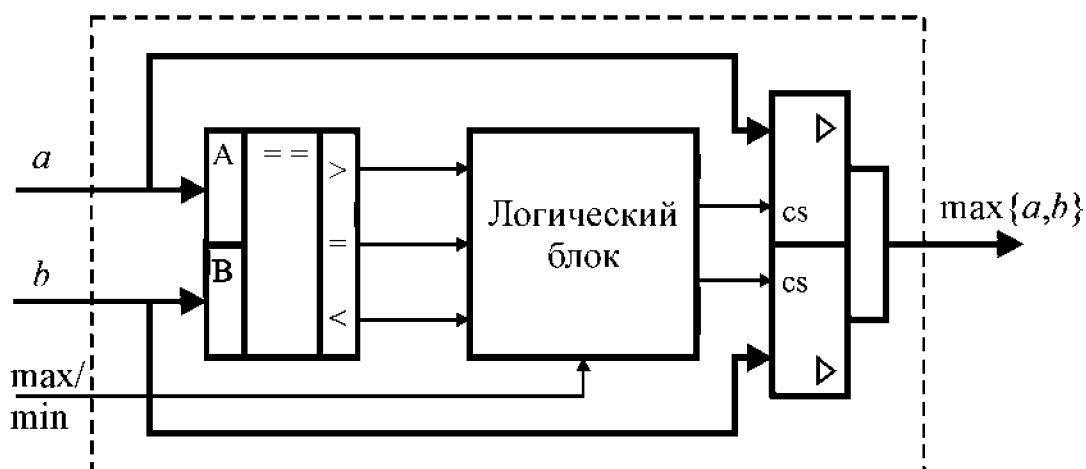


Рис. 4. Блок выбора наибольшего/наименьшего числа из двух

Другими, возможно, неочевидными способами сортировки чисел, являются схемы, представленные на рис. 5. Рассмотрим устройство⁴, изображенное на рис. 5а.

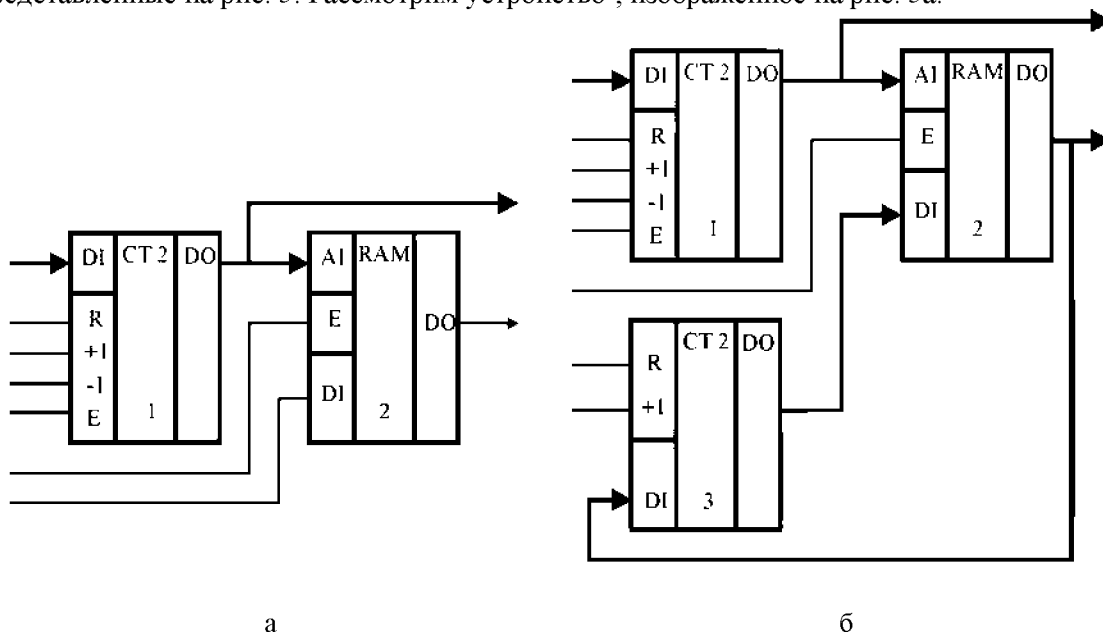


Рис. 5. Устройства для сортировки чисел

Оно состоит из оперативного запоминающего устройства (RAM) и двоичного счетчика с возможностью параллельной загрузки (СТ2). Основная идея состоит в том, что каждое число для сортировки представляется единицей, записанной по адресу, равному этому числу. Очевидно, что емкость ОЗУ в этом случае должна быть равна 2^n бит, где n – разрядность чисел.

Устройство имеет следующий алгоритм работы. Первым циклом работы является обнуление ОЗУ. Для этого счетчик переводится в режим счета, на вход данных (DI) ОЗУ подается уровень логического нуля и сигнал разрешения записи (E). На инкрементирующий вход счетчика (+1) подается импульсный сигнал, в результате чего счетчик последовательно перебирает все адресное пространство ОЗУ, заполняя его нулями.

Далее производится загрузка ОЗУ числами для сортировки. Для этого счетчик переводится в режим параллельной загрузки данных и фактически представляет собой параллельный регистр. Каждое число, предназначенное для сортировки, подается на вход данных (DI) счетчика и через него на адресные входы (AI) ОЗУ. На вход данных (DI) ОЗУ подается уровень логической единицы, а на вход разрешения записи (E) – сигнал, разрешающий загрузку данных в ОЗУ. После того как будут загружены все числа, выполняется режим сортировки. Он может выполняться по возрастанию или по убыванию. В случае сортировки по возрастанию импульсный сигнал подается на инкрементирующий вход (+1) счетчика, если выбран режим по убыванию – на декрементирующий (-1) вход. В этом режиме ОЗУ переводится в режим чтения, а счетчик последовательно пробегает все адресное пространство ОЗУ. В случае, если на выходе данных (DO) ОЗУ обнаруживается единица, это означает, что на выходах счетчика (DO) и выходной шине присутствует очередное отсортированное число.

Достоинствами данной схемы являются независимость времени работы от количества чисел для сортировки, небольшие аппаратные затраты при малой разрядности чисел, возможность сортировать числа в заданном диапазоне. К недостаткам следует отнести не очень высокое быстродействие, а также его резкое снижение и рост аппаратных затрат при наращивании разрядности чисел для сортировки. Другой недостаток заключается в том, что устройство принимает два и более одинаковых числа за одно. Однако он может быть преодолен, если заменить ОЗУ на n -разрядное и включить в схему еще один счетчик (см. рис. 5б).

Теперь при загрузке чисел в ОЗУ устройство вначале считывает из него данные, затем добавляет к ним единицу и вновь записывает их в ОЗУ. Эти данные соответствуют количеству одинаковых чисел, предназначенных для сортировки.

В данной работе изложены лишь некоторые, наиболее популярные решения программной и аппаратной сортировки чисел, которые показывают направления решения этой, можно сказать, классической задачи.

¹ *Зубов В.С.* Справочник программиста. Базовые методы решения графовых задач и сортировки. М.: Филинь, 1999. 256 с.

² *Ахо, Альфред В., Хопкрфт, Джон, Ульман, Джефффри, Д.* Структуры данных и алгоритмы: Пер. с англ. М.: Вильямс, 2003. 384 с.

³ *Шевкопляс Б.В.* Микропроцессорные структуры. Инженерные решения: Справочник. 2-е изд., перераб. и доп. М.: Радио и связь, 1990. 512 с.

⁴ Пат. 60746 Российская Федерация, МПК⁷ G06F7/06, G06F7/08. Устройство для сортировки чисел / Кутузов Д.В., Осовский А.В.; заявитель и патентообладатель ГОУ ВПО «Астраханский государственный университет». № 2006125360; заявл. 13.07.06; опубл. 27.01.07

УДК 621.382

СХЕМОТЕХНИКА АНАЛОГОВЫХ ИНТЕГРАТОРОВ С ИЗМЕНЯЕМОЙ ПОСТОЯННОЙ ВРЕМЕНИ

Д.М. Сурков

В статье рассматриваются интеграторы на операционных усилителях с изменяемой постоянной времени, рассмотрены достоинства и недостатки интеграторов на переключаемых конденсаторах. Предлагается интегратор с изменяемой в широком диапазоне постоянной времени, в котором в качестве управляемого напряжением резистора используется полевой транзистор с управляющим р-п-переходом. Рассмотрено схемотехническое решение, позволяющее существенно уменьшить нелинейность характеристик полевого транзистора.

В импульсной и вычислительной технике для преобразования сигналов применяются аналоговые интеграторы, часто при этом необходимо изменять постоянную времени интегратора в зависимости от характеристик обрабатываемого сигнала¹. В аналоговом интеграторе на операционном усилителе (рис. 1а) для изменения постоянной времени могут применяться подключаемые с помощью коммутатора к R1 дополнительные резисторы². Диапазон изменения постоянной времени при постоянной емкости конденсатора C2 определяется общим сопротивлением подключаемых резисторов. Для изменения постоянной времени в широких пределах необходимо увеличивать количество этих резисторов, а это приведет, в свою очередь, к использованию коммутаторов с большим количеством выходов и усложнению схемы управления коммутацией (за счет увеличения числа адресных входов коммутаторов).

Применение интеграторов с коммутируемыми конденсаторами (рис. 1б) вместо обычных интеграторов дает два преимущества: 1) коэффициент передачи интегратора зависит только от отношения емкостей конденсаторов C1 и C2, а не от их абсолютных значений; 2) возможность изменения постоянной времени интегратора с помощью изменения только тактовой частоты.