Джозефу У. Гауду (Joseph W. Gaud),
моему школьному учителю алгебры,
который зажег во мне пламя
восхищения математикой

ПРЕДИСЛОВИЕ

Около 30 лет назад, во время первой летней практики, мне посчастливилось участвовать в проекте MAC в Массачусетском технологическом институте. Тогда я впервые смог поработать на компьютере DEC PDP-10, который в то время предоставлял более широкие возможности при программировании на ассемблере, чем любой другой компьютер. Этот процессор имел большой набор команд для выполнения операций с отдельными битами, их тестирования, маскирования, для работы с полями и целыми числами. Несмотря на то что PDP-10 давно снят с производства, до сих пор есть энтузиасты, работающие на этих машинах либо на их программных эмуляторах. Некоторые из них даже пишут новые приложения; по крайней мере в настоящее время есть, как минимум, один Web-узел, поддерживаемый эмулированным PDP-10. (Не смейтесь — поддержка такого узла ни в чем не уступает поддержке антикварного автомобиля “на ходу”.)

Тем же летом 1972 года я увлекся чтением серии отчетов по новым исследованиям, публиковавшихся в институте под общим названием HAKMEM — довольно странный и эклектичный сборник разнообразных технических мелочей[1]. Тематика статей могла быть любой: от электронных схем до теории чисел; но больше всего меня заинтриговал небольшой раздел, в котором публиковались программистские хитрости. Практически в каждой статье этого раздела содержалось описание некоторой (зачастую необычной) операции над целыми числами или битовыми строками (например, подсчет единичных битов в слове), которую можно было легко реализовать в виде длинной последовательности машинных команд либо цикла, а затем обсуждалось, как можно сделать это же, используя только четыре, три или две тщательно подобранные команды, взаимодействие между которыми становилось очевидным только после соответствующих объяснений или самостоятельных исследований. Я изучал эти маленькие программные самородки с тем же наслаждением, с которым другие пьют пиво или едят сладости. Я просто не мог остановиться. Каждая такая программа оказывалась для меня находкой, в каждой из них была интеллектуальная глубина, элегантность и даже своеобразная поэзия.

“Наверняка, — думал я тогда, — таких программ должно быть намного больше. Должна же где-то быть книга, посвященная таким штукам”.

Книга, которую вы держите в руках, потрясла меня. Автор систематически, на протяжении многих лет собирал такие программные перлы, а теперь свел их воедино, тематически организовал и снабдил каждый четким и подробным описанием. Многие из них можно записать в машинных командах, но книга полезна не только для тех, кто пишет программы на ассемблере. Главная тема книги — рассмотреть базовые структурные отношения среди целых чисел и битовых строк и эффективные приемы реализации операций над ними. Рассмотренные в книге методы столь же полезны и практичны при программировании на языках высокого уровня, например C или Java, как и на языке ассемблера.

Во многих книгах по алгоритмам и структурам данных описаны сложные методы сортировки и поиска, поддержания хэш-таблиц и двоичных деревьев, работы с записями и указателями. Здесь же рассматривается, что можно сделать с очень крошечной частью

[1] Почему HAKMEM? Сокращение от hacks memo; одно 36-битовое PDP-10 слово может содержать шесть 6-битовых символов, поэтому многие имена PDP-10, с которыми тогда работали программисты, были ограничены шестью символами. Аббревиатуры из шести символов широко использовались для сокращенных названий, иногда ими пользовались просто для удобства. Поэтому название сборника HAKMEM имело ясный смысл для специалистов, по крайней мере в то время.

данных — битами и массивами битов. Поразительно, сколько всего можно сделать, используя только операции двоичного сложения и вычитания вместе с некоторыми поразрядными операциями. Операции с переносом позволяют одному биту воздействовать на все биты, находящиеся слева от него, что делает сложение особенно мощной операцией при работе с данными способами, не получившими широкого распространения.

Сейчас у вас в руках книга именно о таких методах. Если вы работаете над оптимизирующим компилятором или просто создаете высокопроизводительный код, вам просто необходимо прочитать эту книгу. Возможно, в своей повседневной работе вы не часто будете использовать изложенные здесь приемы, но, если вам потребуется организовать цикл по всем битам слова, ускорить работу с отдельными битами во внутреннем цикле или вы просто столкнетесь с ситуацией, когда код получается слишком сложным, загляните в эту книгу — я уверен, вы найдете в ней помощь в решении стоящей перед вами проблемы. В любом случае чтение этой книги доставит вам огромное удовольствие.

Гай Л. Стил, мл. (Guy L. Steele, Jr.)
Берлингтон, Массачусетс
Апрель 2002

ВСТУПЛЕНИЕ

Caveat emptor[2]*: стоимость сопровождения программного обеспечения пропорциональна квадрату творческих способностей программиста.*

(Первый закон творческого
программирования,
Роберт Д. Блис (Robert D. Bliss), 1992)

Перед вами сборник программных приемов, которые я собирал много лет. Большинство из них работают только на компьютерах, в которых целые числа представлены в дополнительном коде. Хотя в данной книге речь идет о 32-битовых машинах с соответствующей длиной регистра, бо́льшую часть представленных здесь алгоритмов легко перенести на машины с другими размерами регистров.

В этой книге не рассматриваются сложные вопросы наподобие методов сортировки или оптимизации компилируемого кода. Основное внимание уделено приемам работы с отдельными машинными словами или командами, например подсчету количества единичных битов в заданном слове. В подобных приемах часто используется смесь арифметических и логических команд.

Предполагается, что прерывания, связанные с переполнением целых чисел, замаскированы и произойти не могут. Программы на C, Fortran и даже Java работают в таком окружении, но программистам на Pascal и ADA следует быть осторожными!

Представление материала в книге неформальное. Доказательства приводятся только в том случае, если алгоритм неочевиден, а иногда не приводятся вообще. Все методы используют компьютерную арифметику, базовые функции, комбинации арифметических и логических операций и др., а доказательства теорем в этой предметной области часто сложны и громоздки.

Чтобы свести к минимуму количество типографских ошибок и опечаток, многие алгоритмы реализованы на языке программирования высокого уровня, в качестве которого используется C. Это обусловлено его распространенностью и тем, что он позволяет непосредственно комбинировать операции с целыми числами и битовыми строками; кроме того, компилятор языка C генерирует объектный код высокого качества.

Ряд алгоритмов написан на машинном языке. В книге применяется трехадресный формат команд, главным образом для повышения удобочитаемости. Использован язык ассемблера для некой абстрактной машины, которая является представителем современных RISC-компьютеров.

Отсутствие ветвлений в программе всячески приветствуется. Это связано с тем, что на многих машинах наличие ветвлений замедляет выборку команд и блокирует их параллельное выполнение. Кроме того, наличие ветвлений может препятствовать выполнению оптимизации компилятором. Оптимизирующий компилятор более эффективно работает с несколькими большими блоками кода, чем с множеством небольших.

Крайне желательно использовать малые величины при непосредственном задании операнда, сравнение с 0 (а не с другими числами) и параллелизм на уровне команд. Хотя программу

[2] Caveat emptor (лат.) – да будет осмотрителен покупатель.

часто можно значительно сократить за счет использования поиска в таблице, этот метод не слишком распространен. Дело в том, что по сравнению с выполнением арифметических команд загрузка данных из памяти занимает намного больше времени, а методы поиска в таблице зачастую не представляют большого интереса (хотя и весьма практичны).

Напоследок мне хотелось бы напомнить исходное значение слова хакер[3]. Хакер — это страстный любитель компьютеров, он создает что-то новое, переделывает или совершенствует то, что уже есть. Хакер очень хорошо разбирается в том, что делает, хотя часто не является профессиональным программистом или разработчиком. Обычно хакер пишет программы не ради выгоды, а ради собственного удовольствия. Такая программа может оказаться полезной, а может остаться всего лишь игрой интеллекта. Например, хакер может написать программу, которая во время выполнения выводит точную копию себя самой[4]. Таких людей называют хакерами, и эта книга написана именно для них, а не для тех, кто хочет получить совет о том, как взломать что-либо в компьютере.

Благодарности

Прежде всего я хочу поблагодарить Брюса Шрайвера (Bruce Shriver) и Денниса Аллисона (Dennis Allison), оказавших мне помощь в опубликовании книги. Я признателен коллегам из IBM, многие из которых упомянуты в библиографии. Особой благодарности достоин Мартин Хопкинс (Martin E. Hopkins) из IBM, которого по праву можно назвать “мистер Компьютер”. Этот человек неудержим в своем стремлении подсчитать в программе каждый такт, и многие его идеи нашли отражение в этой книге. Благодарю также обозревателей из Addison-Wesley, которые значительно улучшили мою книгу. Со многими из них я не знаком, но не могу не упомянуть выдающийся 50-страничный обзор одного из них, Гая Л. Стила, мл. (Guy L. Steele, Jr.), где, в частности, затронуты вопросы перемешивания битов, обобщенного упорядочения и многие другие, которые обязательно войдут во второе издание книги (☺). Ряд предложенных им алгоритмов использован в данной книге. Помогла мне и его пунктуальность. Например, я ошибочно написал, что шестнадцатеричное число 0xAAAAAAAA можно разложить на множители 2·3·17·257·65537; Гай указал, что 3 необходимо заменить на 5. Он не упустил ни одной из подобных мелочей и намного улучшил стиль изложения материала. Кроме того, весь материал, связанный с методом “параллельного префикса”, появился в книге исключительно благодаря Гаю.

Г.С. Уоррен мл. (H.S. Warren, Jr.)
Йорктаун, Нью-Йорк
Февраль 2002

[3] В последнее время “хакерами” часто называют тех, кто получает несанкционированный доступ к банковским системам, взламывает Web-узлы и ведет прочую разрушительную деятельность либо ради получения денег, либо для демонстрации всем своей “крутости”. К настоящим хакерам таковые не имеют никакого отношения. — *Прим. перев.*

[4] Самая короткая такая программа на C, известная автору, написана Владом Таировым и Рашидом Фахреевым и содержит всего 64 символа:

```
main(a){printf(a,34,a="main(a){printf(a,34,a=%c%s%c,34);}",34);}
```