Практическое занятие № 1.2

ПРЕДСТАВЛЕНИЕ ЧИСЕЛ В КОМПЬЮТЕРЕ. МОДЕЛЬ ПАМЯТИ

1. ЦЕЛИ РАБОТЫ

- Изучение алгоритма представления целых чисел со знаком и без знака.
- Построение формальной модели оперативной памяти.

2. ТЕОРЕТИЧЕСКИЙ БЛОК

Системы счисления

Логические схемы компьютера условно могут различать два состояния: включено (логическая единица) и выключено (логический нуль). Поэтому любая информация в компьютере представляется в форме двоичного кода (т. е. чисел), а вычислительные операции строятся на основе двоичной алгебры.

Двоичная система счисления является естественной для компьютера, но малоудобной для программиста. Основные операции мы осуществляем через десятичную систему счисления. Однако она не позволяет анализировать представление информации естественным образом; требуется сначала перевести число в двоичную форму.

Таким образом, двоичная форма числа естественна, однако запись даже небольших чисел в ней получается весьма громоздкой. Поэтому гораздо чаще операции производят в шестнадцатеричной системе счисления.

Вне зависимости от используемой системы счисления ассемблер переводит число в двоичную форму.

Перевод целых неотрицательных чисел

Ассемблер способен работать с двоичной, восьмеричной, десятичной и шестнадцатеричной системами счисления. Чаще всего в ассемблерных программах используется шестнадцатеричная форма числа, поскольку с ней удобно оперировать; отладчики выдают информацию в этой системе счисления.

Перевод из десятичной в двоичную систему

Для перевода в двоичную систему счисления требуется делить число на 2, пока неполное частное не меньше двух. Остатки от деления в обратном порядке задают цифры двоичного числа.

Например, разложение для 2015:

Неполное частное	2015	1007	503	251	125	62	32	15	7	3	1
Остаток	1	1	1	1	1	0	1	1	1	1	

Таким образом, $2015_{10} = 11111011111_2$.

Перевод из десятичной в шестнадцатеричную систему

Для перевода в шестнадцатеричную систему счисления требуется делить число на 16, пока неполное частное не меньше 16. Остатки от деления в обратном порядке задают цифры шестнадцатеричного числа.

Например, разложение для 2015:

Неполное частное	2015	125	7
Остаток	15 (F)	13 (D)	

Таким образом, $2015_{10} = 7\mathrm{DF}_{16}$.

Перевод из двоичной системы в шестнадцатеричную и обратно

Для перевода двоичного числа в шестнадцатеричное его разбивают на секции по четыре разряда, начиная от младших разрядов. Каждая двоичная секция соответствует одной шестнадцатеричной цифре.

Например:

$$11111011111_2 = 111.1101.1111_2 = 7\mathrm{DF}_{16}.$$

Для перевода шестнадцатеричного числа в двоичное каждую цифру числа заменяют четырьмя двоичными цифрами.

Например:

$$7\mathrm{DF}_{16} = 0111.1101.1111_2 = 11111011111_2.$$

Перевод в десятичную систему

Для перевода двоичного числа в десятичное требуется разложить его в линейную комбинацию по степеням двойки.

Например:

$$11111011111_2 = 1 \cdot 2^{10} + 1 \cdot 2^9 + 1 \cdot 2^8 + 1 \cdot 2^7 + 1 \cdot 2^6 + \\ +0 \cdot 2^5 + 1 \cdot 2^4 + 1 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 = 2015_{10}.$$

Для перевода шестнадцатеричного числа в десятичное требуется разложить его в линейную комбинацию по степеням числа 16.

Например:

$$7\mathrm{DF}_{16} = 7 \cdot 16^2 + 13 \cdot 16^1 + 15 \cdot 16^0 = 2015_{10}.$$

Формальная модель оперативной памяти

Несмотря на историческую преемственность архитектуры процессоров Intel, модель адресации памяти претерпела серьезные изменения. Однако это не мешает нам построить формальную модель, которая в целом будет корректно описывать реально существующие.

Измерение количества информации

Любой разряд двоичного числа может принимать значение 0 либо 1. Ячейка в один разряд называется битом и является наименьшей единицей измерения количества информации. 1 бит способен закодировать два числа, два бита – 4 числа, три бита – 8 чисел и т. д.

Общий закон, очевидно, задается по правилу[1]

$$I = 2^i,$$

где I – количество двоичных чисел, которые можно закодировать; i – количество бит.

Бит – очень маленькая ячейка памяти. Поэтому операции осуществляются над ячейками, кратными 8 битам, именуемыми байтами. Большие объемы памяти обычно переводятся в более высокие порядки единиц измерения.

Единица измерения	Обозначение	Степень двойки	Число байт
Килобайт	Кб	10	1024
Мегабайт	Мб	20	1024^2
Гигабайт	Гб	30	1024^3
Терабайт	Тб	40	1024^4

Адресация оперативной памяти

Для обращения к ячейке оперативной памяти процессор должен знать ее адрес. Схематично устройство оперативной памяти можно представить в виде ленты подряд идущих байтов.

---

[1] Читателю этот закон должен быть знаком по формуле Хартли.

Адрес	. . .	2012	2013	2014	2015	2016	. . .
Значение	. . .	56	2F	00	FE	A8	. . .

Однако для однозначных операций с ячейками оперативной памяти кроме адреса необходимо знать и размер ячейки – количество однобайтовых ячеек, которое занимает значение. Если речь идет о переменных, то размер определяется типом переменной.

Перевод целых чисел со знаком

Пусть формально имеется ячейка размером в k бит, тогда можно составить 2^k различных чисел. Если числа неотрицательные, то они входят в диапазон $0..2^k-1$.

Для возможности работы с отрицательными числами половину диапазона чисел отводят на отрицательные, а другую на неотрицательные, т. е. в общем случае число со знаком лежит в диапазоне $-2^{k-1}..2^{k-1}-1$. Отрицательные числа представляются в дополнительном коде.

Дополнительный код числа можно задать следующей функцией:

$$\text{доп}(x)=\begin{cases} x, \text{ если } x \ge 0; \\ 2^k-|x|, \text{ если } x<0. \end{cases}$$

Ограничимся ячейкой размером в байт ($k=8$):

$\text{доп}(1)=1=00000001$	$\text{доп}(-1)=256-1=11111111$
$\text{доп}(2)=2=00000010$	$\text{доп}(-2)=256-2=11111110$
$\text{доп}(3)=3=00000011$	$\text{доп}(-3)=256-3=11111011$
...	...
$\text{доп}(126)=126=01111110$	$\text{доп}(-126)=256-126=10000010$
$\text{доп}(127)=127=01111111$	$\text{доп}(-127)=256-127=10000001$
$\text{доп}(0)=0=00000000$	$\text{доп}(-128)=256-128=10000000$

Очевидно, что старший бит числа можно рассматривать как флажок знака: у отрицательных чисел он равен 1.

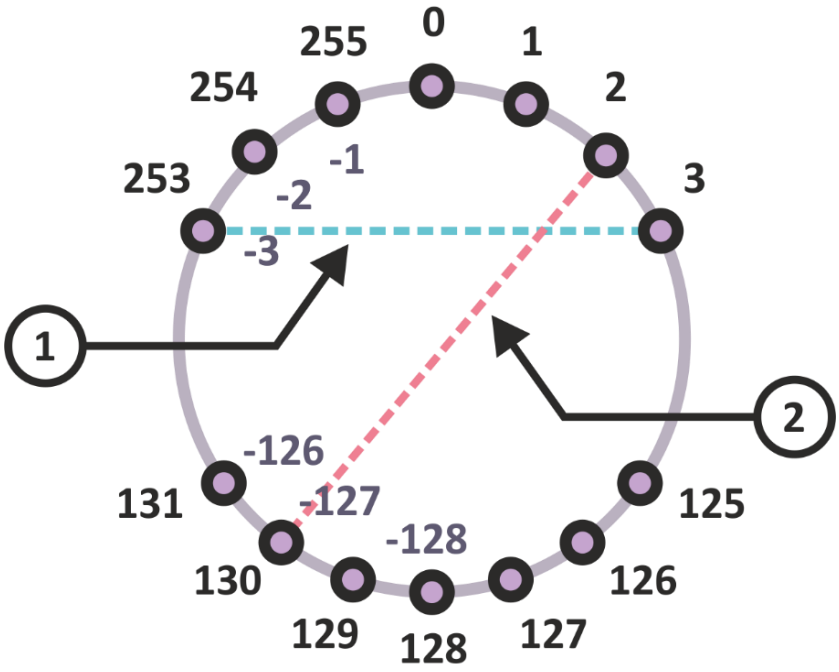
Дополнительный код упрощает разработку логических схем. Так, операция разности замещается сложением. В частности, сумма двух противоположных чисел должна быть равной нулю.

Действительно, сложим, например, 3 и –3:

$3+(-3)=00000011+1111101=1.00000000$ (занимает 9 бит).

Поскольку мы ограничены ячейкой размером в байт, 1 из старшего разряда игнорируется, а в качестве результата берется 0.

Логику построения дополнительного кода хорошо иллюстрирует следующая модель. Представим, что числа от 0 до 255 расположены по окружности в порядке возрастания. Если в противоположную сторону отметить отрицательные числа, то заметим, что соответствующие им числа – дополнительный код:



Очевидно следующее:

- сумма противоположных чисел всегда равна 256, что с учетом «заворота» дает нуль (указатель 1);
- диаметрально противоположные числа в двоичном представлении отличаются лишь знаковым битом (указатель 2).

Дополнительный код можно получить другим способом, если число уже в двоичной или шестнадцатеричной форме.

Алгоритм получения дополнительного кода с помощью инверсии

Пусть дано отрицательное число x.

1. Переведем модуль x в двоичную форму.

2. Инвертируем биты числа (нули заменяем единицами, единицы – нулями).

3. К полученному числу добавим 1.

Например:

1. $-10 \rightarrow 10 = 00001010.$
2. $00001010 \rightarrow 11110101.$
3. $11110101 \rightarrow 11110101 + 1 = 11110110.$

Пусть дано отрицательное число x.

1. Переведем модуль x в шестнадцатиричную форму.
2. Инвертируем цифры числа (каждую цифру вычитаем из 15).
3. К полученному числу добавим 1.

Например:

1. $-125 \rightarrow 125 = 7\text{D}.$
2. $7D \rightarrow [15 - 7][15 - 13] = 82.$
3. $82 \rightarrow 82 + 1 = 83.$

Различие знаковых и беззнаковых чисел

Предположим, в некотором байте памяти хранится число FE. Как понять, что это число 254 либо –2?

Для случайно взятой ячейки памяти нельзя определить однозначно, какое из чисел в ней записано.

Однозначность возможна только в случае, если нам известно, что заданный участок памяти является переменной (данными) определенного типа.

3. ВОПРОСЫ ДЛЯ САМОПРОВЕРКИ

1. Какие системы счисления используются в ассемблерном коде?
2. Сколько целых неотрицательных чисел можно закодировать в ячейку размерами в 2 байта, 32 бита? В каком диапазоне можно закодировать числа со знаком, занимающие в памяти 3 байта?
3. Каким образом осуществляется перевод числа из двоичной системы счисления в шестнадцатеричную и обратно?
4. Опишите формальную модель устройства оперативной памяти.
5. Что такое дополнительный код? Почему говорят, что дополнительный код обладает свойством инверсии относительно сложения?

4. САМОСТОЯТЕЛЬНАЯ РАБОТА

1. Переведите числа в указанную систему счисления:
   - $509_{10} \rightarrow{***}_2$ и обратно;
   - $187_{10} \rightarrow{***}_{16}$ и обратно;

- $10111010_2 \rightarrow ***_{16}$ и обратно;
- $9\text{FA}_{16} \rightarrow ***_2$ и обратно.

2. Укажите минимальное число бит и байт, требуемое для хранения указанных чисел:
   - 34; 65; 309;
   - 101_2; 1100001_2; 1110001111010011_2;
   - 0_{16}; 56_{16}; $1\text{F}67\text{A}56_{16}$.
3. Получите дополнительный код указанных чисел (в двоичной и шестнадцатеричной формах), если операции замкнуты относительно двухбайтовой ячейки:
   - –6; -1023.
   - -101001_2; -101111100111_2;
   - -6E_{16}; $1\text{F}09_{16}$.
4. Покажите, что дополнительный код нуля является нулем.
5. Найдите дополнительный код числа -128 (в рамках одного байта). Какой вывод можно сделать?
6. Продемонстрируйте корректность сложения любых чисел на основе дополнительного кода:
   - двух положительных;
   - положительного и отрицательного (отрицательное меньше по модулю);
   - положительного и отрицательного (отрицательное больше по модулю);
   - двух отрицательных.