Представление целых чисел в памяти ЭВМ

Представление целых чисел в памяти ЭВМ не самая простая задача как может показаться. Память электронных вычислительных машин можно представить в виде конечной последовательности нулей и единиц, но так как целые числа бывают положительные и отрицательные, возникает вопрос представления отрицательных чисел. Ведь, если положительные числа можно хранить в двоичной системе счисления без изменений, отрицательные числа нужно каким-то образом отметить таким образом, чтобы положительные и отрицательные числа можно было однозначно идентифицировать и декодировать. При этом способ кодирования должен желательно удовлетворять следующим требованиям:

- способ кодирования не должен усложнять архитектуру вычислительного устройства;
- не усложнял арифметические действия;
- в фиксированном количестве бит хранил бы одинаковое количество положительных и отрицательных чисел.

Все примеры кодов в лекции подразумевают, что числа представлены в восьмибитном типе данных (т.е. все числа имеют восемь разрядов вне зависимости от того, сколько бит информации им фактически требуется для записи).

Первый способ кодирования, который мы рассмотрим — ***прямой код***. В прямом коде положительные числа не кодируются:

$85_{10} \rightarrow 01010101_2$

$71_{10} \rightarrow 01000111_2$

$62_{10} \rightarrow 00111110_2$

Последовательности нулей и единиц, полученные переводом положительных чисел из десятичной системы в двоичную, являются прямым кодом этих чисел.

В случае, если число отрицательное, модуль числа переводится в двоичную систему счисления, а в старший разряд записывается, единица:

$-58_{10} \rightarrow 10111010_2$

$-88_{10} \rightarrow 11011000_2$

$-100_{10} \rightarrow 11100100_2$

Полученная двоичная последовательность является прямым кодом числа.

Т.е. получается, что старший разряд битовой последовательности кодирует знак числа (0 если число положительное, 1 если число отрицательное).

Процесс декодирования чисел в прямом коде очень прост. Нужно посмотреть на старший разряд, что даст информацию о знаке числа, затем оставшиеся n-1 разряды перевести в нужную систему счисления (n - количество бит в типе данных):

$10010000_2 \rightarrow -16_{10}$

$00010111_2 \rightarrow 23_{10}$

$10100111_2 \rightarrow -39_{10}$

Таким образом в n-битовом типе данных при помощи прямого кода можно представить числа в диапазоне $[-2^{n-1}+1; 2^{n-1}-1]$.

Главным недостатком прямого кода является необходимость усложнения архитектуры вычислителя для выполнения операций с отрицательными числами.

Второй способ представления целых чисел – ***код со сдвигом***. При использовании кода со сдвигом можно кодировать числа в диапазоне $[-2^{n-1}; 2^{n-1}-1]$.

Идея кода со сдвигом заключается в сдвиге двоичных представлений чисел от нуля к началу диапазона. Т.е. самое маленькое число кодируется последовательностью нулей, следующее число единицей, следующее за ним двойкой в двоичной системе счисления и т.д.:

$-2^{n-1} \rightarrow 000...00$

$-2^{n-1}+1 \rightarrow 000...01$

$-2^{n-1}+2 \rightarrow 000...10$

…

$2^{n-1}-1 \rightarrow 111...11$

Для того чтобы закодировать число в коде со сдвигом необязательно выписывать коды для всех чисел начиная с самого маленького. Достаточно прибавить к кодируемому числу сдвиг:

$$N = P + E,$$

где P – кодируемое число; E – сдвиг; N – код числа P.

Двоичное представление N – представление числа P в памяти ЭВМ, закодированного кодом со сдвигом.

Сдвиг вычисляется по формуле:

$$E = 2^{n-1},$$

где n – количество бит в типе данных.

Например:

$-47_{10} \rightarrow -47 + 2^{8-1} = -47 + 128 = 81 \rightarrow 01010001_2$

$111_{10} \rightarrow 111 + 2^{8-1} = 111 + 128 = 239 \rightarrow 11101111_2$

$88_{10} \rightarrow 88 + 2^{8-1} = 88 + 128 = 216 \rightarrow 11011000_2$

Нужно обратить внимание на то, что в отличие от прямого кода положительные числа в дополнительном коде тоже кодируются.

Для декодирования чисел из кода со сдвигом нужно вычесть из кода числа сдвиг:

$$P = N - E,$$

где P – закодированное число; E – сдвиг; N – код числа P.

$11010101 \rightarrow 213 - 2^{8-1} = 213 - 128 = 85$

$01100110 \rightarrow 102 - 2^{8-1} = 102 - 128 = 26$

Основной недостаток кода со сдвигом в том, что при выполнении арифметических операций приходится учитывать сдвиг, что обязывает выполнять дополнительные операции для получения корректного результата.

Следующий способ кодирования – ***обратный код***. Как и в случае с прямым кодом положительные числа в обратном коде никак не модифицируются. Если число отрицательное нужно перевести в двоичную систему счисления его модуль и инвертировать все биты. При этом старший бит полученной последовательности будет показывать положительное или отрицательное было закодировано число (1 - число отрицательное):

62 → 00111110

51 → 00110011

82 → 01010010

-80 → |-80| → 01010000 → 10101111

-60 → |-60| → 00111100 → 11000011

Для декодирования числа нужно посмотреть на старший бит и понять положительное или отрицательное было закодировано число. Если в старшем разряде стоит единица, закодировано отрицательное число, и для получения модуля числа нужно инвертировать все разряды последовательности:

11001100 → -00110011 → -51

Если в старшем разряде последовательности стоит ноль – закодировано положительное число и достаточно перевести последовательность в нужную систему счисления.

00110111 → 55

Диапазон чисел который можно закодировать в n-битном типе данных при помощи обратного кода – $[-2^{n-1}+1; 2^{n-1}-1]$.

Главный недостаток обратного кода в том, что для выполнения операций над отрицательными числами требуется усложнить архитектуру вычислительного устройства.

Последний способ кодирования – ***дополнительный код***. Дополнительный код очень похож на обратный код. Положительные числа в этом способе представления не кодируются. Чтобы получить дополнительный код отрицательного числа требуется получить обратный код и к нему прибавить единицу:

62 → 00111110

51 → 00110011

82 → 01010010

-80 → |-80| → 01010000 → 10101111 → 10101111 + 1 → 10110000

-60 → |-60| → 00111100 → 11000011 → 11000011 + 1 → 11000100

Для декодирования числа нужно посмотреть на старший бит и понять положительное или отрицательное было закодировано число. Если в старшем разряде стоит единица, закодировано отрицательное число, и для получения модуля числа нужно вычесть из последовательности единицу, а затем инвертировать все разряды последовательности:

11001101 → 11001101 - 1 → 11001100 → -00110011 → -51

Если в старшем разряде последовательности стоит ноль – закодировано положительное число и достаточно перевести последовательность в нужную систему счисления.

00110111 → 55