Шестнадцатеричный дамп целого числа

Задание. Для заданного короткого целого числа построить его шестнадцатеричный дамп.

Проблема. Представить описание алгоритма, позволяющего выполнить это задание.

Обсуждение хода решения:

1) ввести число;
2) сформировать символьную последовательность на основе символьных литералов от ***'0'*** до ***'f'*** или от ***'0'*** до ***'F'***,сохраняя в ней ведущие нули;
3) сформировать символьную последовательность на основе символьных литералов от ***'0'*** до ***'f'*** или от ***'0'*** до ***'F'***, “подавляя” в ней ведущие нули.

Циклическая операция маскирования позволяет последовательно выявить значения для всех тетрад (шестнадцатеричных цифр) из двоичных разрядов числа, начиная с тетрады, следующей за старшей, включающей в себя знаковый двоичный разряд. Старшую тетраду (старшую шестнадцатеричную цифру), как видим, следует рассматривать независимо от остальных. Начальное значение маски можно задать с помощью восьмеричного литерала ***07400*** или шестнадцатеричного литерала ***0xf00***. Маску, как видим, всякий раз следует сдвигать вправо на четыре двоичных разряда. Для представления коротких целых чисел будем опираться на 16-разрядную архитектуру.

Обратимся к шестнадцатеричному дампу короткого целого числа 13_{10}, строя его на основе анализа значений маскируемых младших шестнадцатеричных разрядов числа с последующим арифметическим сдвигом вправо на четыре двоичных разряда маски:

Десятичный код	Шестнадцатеричный код	Двоичный код	Шестнадцатеричный дамп
13	000D	0000000000001101	0
13	000D	0000000000001101	0
	Маска	Маска	
	0F00	0000111100000000	
13	000D	0000000000001101	0
	Маска	Маска	
	00F0	0000000011110000	
13	000D	0000000000001101	D
	Маска	Маска	
	000F	0000000000001111	

Обратимся к шестнадцатеричному дампу короткого целого числа -13_{10}, строя его на основе анализа значений маскируемых младших шестнадцатеричных разрядов числа с последующим арифметическим сдвигом вправо на четыре двоичных разряда маски:

Десятичный код	Шестнадцатеричный код	Двоичный код	Шестнадцатеричный дамп
–13	FFF3	1111111111110011	F
–13	FFF3	1111111111110011	F
	Маска	Маска	
	0F00	0000111100000000	
–13	FFF3	1111111111110011	F
	Маска	Маска	
	00F0	0000000011110000	
–13	FFF3	1111111111110011	3
	Маска	Маска	
	000F	0000000000001111	

Описания алгоритмов построения шестнадцатеричных дампов.

При разработке алгоритмов построения шестнадцатеричных дампов коротких целых чисел будем опираться на арифметический сдвиг вправо маски. Чтобы символьная последовательность при этом формировалась в прямом порядке, следует использовать циклическую операцию маскирования исходного числа, которая позволит последовательно выявить значения всех его шестнадцатеричных разрядов (тетрад), начиная с тетрады, следующей за старшей, включающей в себя знаковый двоичный разряд. Старшую тетраду (старшую шестнадцатеричную цифру) следует рассматривать независимо от остальных. Начальное значение маски следует задать с помощью восьмеричного литерала ***07400*** или шестнадцатеричного литерала ***0xf00***. Маску всякий раз следует сдвигать вправо на четыре двоичных разряда. В основу механизма преобразования тетрады в шестнадцатеричный символ кладётся функция ***char()***, аргументом которой является увеличенный двоичный код тетрады либо на 48 для цифр от ***0*** до ***9***, либо на 55 для верхнего регистра букв от ***A*** до ***F***, либо на 87 для нижнего регистра букв от ***a*** до ***f***.

Представим структурированное описание алгоритма на естественном языке:

1. Начало
2. Повторить
2.1. Вывести значение “N? ”
2.2. Ввести значение N
пока не будет (N >= –32768) и (N <= 32767)
3. Положить M = 0xf00
4. Положить D = ((N and 0xf000) asr 12) and 0xf
5. Если D > 9
то
5.1. Вывести значение char(D + 87)
иначе
5.2. Вывести значение char(D + 48)
6. От k = 1 до k = 3 повторить
6.1. Положить D = (N and M) asr 16 – 4 * k – 4
6.2. Если D > 9
то
6.2.1. Вывести значение char(D + 87)
иначе
6.2.2. Вывести значение char(D + 48)
6.3. Положить M = M asr 4
7. Конец

Теперь зададимся целью реализовать механизм подавления ведущих нулей для неотрицательных чисел. В основу этого механизма кладётся флаг – переменная, ненулевое значение которой “просигналит” о появлении первой “ведущей” цифры, отличной от нуля, в символьной последовательности. Нулевое значение флага препятствует появлению в символьной последовательности литералов ***'0'***, трактуя их как ведущие нули. Очевидно, что сам нуль станет теперь “особым” числом, которое следует рассматривать независимо от остальных неотрицательных чисел.

Представим структурированное описание алгоритма на естественном языке:

1. Начало
2. Повторить
2.1. Вывести значение “N? ”
2.2. Ввести значение N
пока не будет (N >= –32768) и (N <= 32767)
3. Положить M = 0xf00
4. Положить flag = false
5. Если N не равно 0
то
5.1. Положить D = ((N and 0xf000) asr 12) and 0xf
5.2. Если D не равно 0
то
5.2.1. Положить flag = true
5.2.2. Если D > 9
то
5.2.2.1. Вывести значение char(D + 87)
иначе
5.2.2.2. Вывести значение char(D + 48)
5.3. От k = 1 до k = 3 повторить
5.3.1. Положить D = (N and M) asr 16 – 4 * k – 4
5.3.2. Если D не равно 0
то
5.3.2.1. Положить flag = true
5.3.2.2. Если D > 9
то
5.3.2.2.1. Вывести значение char(D + 87)
иначе
5.3.2.2.2. Вывести значение char(D + 48)
иначе
5.3.2.3. Если flag = true
то
5.3.2.3.1. Вывести значение ‘0’
5.3.3. Положить M = M asr 4
иначе
5.4. Вывести значение ‘0’
6. Конец