Bit Tricks

TB Schardl

May 11, 2010

Outline

Swapping Two Values

Problem: Swap two binary values without using space for a temporary variable.

Swapping Two Values

Problem: Swap two binary values without using space for a temporary variable.
Solution:

1. $X = X \oplus Y$
2. $Y = X \oplus Y$
3. $X = X \oplus Y$

Swapping Two Values

Problem: Swap two binary values without using space for a temporary variable.

Solution:

1. $X = X \oplus Y$
2. $Y = X \oplus Y$
3. $X = X \oplus Y$

Explanation: XOR is its own inverse.

1. $X' = X \oplus Y$
2. $Y' = X' \oplus Y = X \oplus Y \oplus Y = X$
3. $X'' = X' \oplus Y' = X \oplus Y \oplus X = Y$

Finding the Minimum

Problem: Find the minimum of two values, X and Y, without an explicit branch.

Finding the Minimum

Problem: Find the minimum of two values, X and Y, without an explicit branch.
Solution: $Y \oplus ((X \oplus Y) \& (-(X < Y)))$

Finding the Minimum

Problem: Find the minimum of two values, X and Y, without an explicit branch.
Solution: $Y \oplus ((X \oplus Y) \& (-(X < Y)))$
Explanation:

- If $X \geq Y$ then $-(X < Y) = 0$. Therefore we have $Y \oplus 0 = Y$.
- If $X < Y$ then $-(X < Y) = -1$ which is all 1's. Therefore we have $Y \oplus (X \oplus Y) = X$.

Reversing an N-bit Value

Problem: Reverse the bits in an N-bit value.

Reversing an N-bit Value

Mini-Example: $X = 00101100$

- $X = 00101100$

Reversing an N-bit Value

Mini-Example: $X = 00101100$

- $X = 00101100$
- $(X << 4)|(X >> 4) = 11000000|0010$

Reversing an N-bit Value

Mini-Example: $X = 00101100$

- $X = 00101100$
- $(X << 4)|(X >> 4) = 11000000|0010$
- $X = 11000010$

Reversing an N-bit Value

Mini-Example: $X = 00101100$

- $X = 00101100$
- $(X << 4)|(X >> 4) = 11000000|0010$
- $X = 11000010$
- $X = 11000010$

Reversing an N-bit Value

Mini-Example: $X = 00101100$

- $X = 00101100$
- $(X << 4)|(X >> 4) = 11000000|0010$
- $X = 11000010$
- $X = 11000010$
- $((X\&0x33) << 2)|((X >> 2)\&0x33) = 00001000|110000$

Reversing an N-bit Value

Mini-Example: $X = 00101100$

- $X = 00101100$
- $(X << 4)|(X >> 4) = 11000000|0010$
- $X = 11000010$
- $X = 11000010$
- $((X\&0x33) << 2)|((X >> 2)\&0x33) = 00001000|110000$
- $X = 00111000$

Reversing an N-bit Value

Mini-Example: $X = 00101100$

- $X = 00101100$
- $(X << 4)|(X >> 4) = 11000000|0010$
- $X = 11000010$
- $X = 11000010$
- $((X\&0x33) << 2)|((X >> 2)\&0x33) = 00001000|110000$
- $X = 00111000$
- $X = 00111000$

Reversing an N-bit Value

Mini-Example: $X = 00101100$

- $X = 00101100$
- $(X << 4)|(X >> 4) = 11000000|0010$
- $X = 11000010$
- $X = 11000010$
- $((X\&0x33) << 2)|((X >> 2)\&0x33) = 00001000|110000$
- $X = 00111000$
- $X = 00111000$
- $(X\&0x55) << 1)|((X >> 1)\&0x55) = 00100000|0010100$

Reversing an N-bit Value

Mini-Example: $X = 00101100$

- $X = 00101100$
- $(X << 4)|(X >> 4) = 11000000|0010$
- $X = 11000010$
- $X = 11000010$
- $((X\&0x33) << 2)|((X >> 2)\&0x33) = 00001000|110000$
- $X = 00111000$
- $X = 00111000$
- $(X\&0x55) << 1)|((X >> 1)\&0x55) = 00100000|0010100$
- $X = 00110100$

Reversing an N-bit Value

Problem: Reverse the bits in an N-bit value.
Solution: For 32-bit value X:

1. $X = (X << 16)|(X >> 16)$
2. $X = ((X\&0x00FF00FF) << 8)|((X >> 8)\&0x00FF00FF)$
3. $X = ((X\&0x0F0F0F0F) << 4)|((X >> 4)\&0x0F0F0F0F)$
4. $X = ((X\&0x33333333) << 2)|((X >> 2)\&0x33333333)$
5. $X = ((X\&0x55555555) << 1)|((X >> 1)\&0x55555555)$

Explanation:

- First line swaps 2-byte chunks of X.
- Second line swaps bytes.
- Third line swaps nibbles...
- Pattern repeats until all bits are reversed.

Rounding to a Power of 2

Problem: Round a value X up to the next highest power of 2.

Rounding to a Power of 2

Problem: Round a value X up to the next highest power of 2.
Note: The next highest power of 2 of 00111111 is
$01000000 = 00111111 + 1$.

Rounding to a Power of 2

Problem: Round a value X up to the next highest power of 2.
Note: The next highest power of 2 of 00111111 is $01000000 = 00111111 + 1$.
Idea: Use shifts and ORs to fill LOBs of X with 1s, then increment to get next power of 2.

Rounding to a Power of 2

Mini-Example: To round up $X = 0010000000000111$:

Rounding to a Power of 2

Mini-Example: To round up $X = 0010000000000111$:

1. $X = 0001000000000011|X$
   $X = 0011000000000111$

Rounding to a Power of 2

Mini-Example: To round up $X = 0010000000000111$:

1. $X = 0001000000000011|X$
   $X = 0011000000000111$
2. $X = 0000110000000001|X$
   $X = 0011110000000111$

Rounding to a Power of 2

Mini-Example: To round up $X = 0010000000000111$:

1. $X = 0001000000000011 | X$
   $X = 0011000000000111$
2. $X = 0000110000000001 | X$
   $X = 0011110000000111$
3. $X = 0000001111000000 | X$
   $X = 0011111111000111$

Rounding to a Power of 2

Mini-Example: To round up $X = 0010000000000111$:

1. $X = 0001000000000011 | X$
   $X = 0011000000000111$
2. $X = 0000110000000001 | X$
   $X = 0011110000000111$
3. $X = 0000001111000000 | X$
   $X = 0011111111000111$
4. $X = 0000000000111111 | X$
   $X = 0011111111111111$

Rounding to a Power of 2

Mini-Example: To round up $X = 0010000000000111$:

1. $X = 0001000000000011|X$
   $X = 0011000000000111$
2. $X = 0000110000000001|X$
   $X = 0011110000000111$
3. $X = 0000001111000000|X$
   $X = 0011111111000111$
4. $X = 0000000000111111|X$
   $X = 0011111111111111$
5. $X = X + 1 = 0100000000000000$

Rounding to a Power of 2

Problem: Round a value up to the next highest power of 2.
Solution: To round up X

1. $X = X - 1$
2. $X = X|(X >> 1)$
3. $X = X|(X >> 2)$
4. $X = X|(X >> 4)$
5. $X = X|(X >> 8)$
6. $X = X|(X >> 16)$
7. $X = X|(X >> 32)$
8. $X = X + 1$

Indexing a 1

Problem: Find the position of the least-significant set bit in X.

Indexing a 1

Problem: Find the position of the least-significant set bit in X.
deBruijn Sequence: A sequence of n 0's and 1's such that every 0-1 sequence of length $\lg n$ occurs exactly once as a contiguous substring.
Example deBruijn Sequence: For length 8, one deBruijn sequence is 00011101. This value contains the sequences 000, 001, 011, 111, 110, 101, 010 (wrapping), and 100 (wrapping).

Indexing a 1

Problem: Find the position of the least-significant set bit in X.
deBruijn Sequence: A sequence of n 0's and 1's such that every 0-1 sequence of length $\lg n$ occurs exactly once as a contiguous substring.
Example deBruijn Sequence: For length 8, one deBruijn sequence is 00011101. This value contains the sequences 000, 001, 011, 111, 110, 101, 010 (wrapping), and 100 (wrapping).
Idea: Multiply LSB of X by a deBruijn sequence to index into a lookup table.

Indexing a 1

Solution: For an n-bit number X:

1. Multiply a length n deBruijn sequence by the lowest order 1 in X.
   - This is equivalent to shifting the deBruijn sequence left by some amount.
2. Examine the $\lg n$ subsequence of the deBruijn sequence at the top of the low half of the product bits.
3. Use that subsequence to index into a lookup table to determine the index of the lowest order 1.

For 32-bit value X:

1. $Table[] = \{0, 1, 28, 2, 29, 14, 24, 3, 30, 22, 20, 15, 25, 17, 4, 8, 31, 27, 13, 23, 21, 19, 16, 7, 26, 12, 18, 6, 11, 5, 10, 9\}$
2. Return $Table[((X \& (-X)) * 0x077CB531) >> 27]$

Finding the $\log_2$ of an N-bit Integer

Problem: Quickly find the $\log_2$ of an N-bit integer.

Finding the $\log_2$ of an N-bit Integer

Problem: Quickly find the $\log_2$ of an N-bit integer.
Solution: Round down to a power of 2, then determine the index of the 1 in the result.
For 32-bit value X:

1. $Table[] = \{0, 1, 28, 2, 29, 14, 24, 3, 30, 22, 20, 15, 25, 17, 4, 8, 31, 27, 13, 23, 21, 19, 16, 7, 26, 12, 18, 6, 11, 5, 10, 9\}$
2. $X = X|(X >> 1)$
3. $X = X|(X >> 2)$
4. $X = X|(X >> 4)$
5. $X = X|(X >> 8)$
6. $X = X|(X >> 16)$
7. $X = (X >> 1) + 1$
8. Return $Table[(X * 0x077CB531) >> 27]$

Counting Bits Set

Problem: Count the number of bits set in a given value.

Counting Bits Set

Problem: Count the number of bits set in a given value.
Simple Solution: For given value X:

1. Set $c = 0$.
2. While $X \neq 0$, increment c and set $X = X\&(X - 1)$.

Counting Bits Set

Problem: Count the number of bits set in a given value.
Simple Solution: For given value X:

1. Set $c = 0$.
2. While $X \neq 0$, increment c and set $X = X \& (X - 1)$.

Explanation:

- $X \& (X - 1)$ clears the lowest bit set in X.
- c counts the number of clear-lowest-set-bit operations needed to make $X = 0$.

Counting Bits Set

Solution: For 32-bit value X:

1. $X = X - ((X >> 1)\&0x55555555)$
2. $X = (X\&0x33333333) + ((X >> 2)\&0x33333333)$
3. $c = ((X + (X >> 4)\&0x0F0F0F0F) * 0x01010101) >> 24$

Counting Bits Set

Mini-example: $X = 0110110001011110$
First, we compute $X = X - ((X >> 1) \& 0x5555)$:

Counting Bits Set

Mini-example: $X = 0110110001011110$

First, we compute $X = X - ((X >> 1) \& 0x5555)$:

- $X = 01\ 10\ 11\ 00\ 01\ 01\ 11\ 10$

Counting Bits Set

Mini-example: $X = 0110110001011110$

First, we compute $X = X - ((X >> 1)\&0x5555)$:

- $X = 01\ 10\ 11\ 00\ 01\ 01\ 11\ 10$
- $(X >> 1)\&0x5555 =$

$$\begin{array}{ll} & 00\ 11\ 01\ 10\ 00\ 10\ 11\ 11 \\ \& & 01\ 01\ 01\ 01\ 01\ 01\ 01\ 01 \\ = & 00\ 01\ 01\ 00\ 00\ 00\ 01\ 01 \end{array}$$

Counting Bits Set

Mini-example: $X = 0110110001011110$

First, we compute $X = X - ((X >> 1) \& 0x5555)$:

- $X = 01\ 10\ 11\ 00\ 01\ 01\ 11\ 10$
- $(X >> 1) \& 0x5555 =$

$$\begin{array}{ll} & 00\ 11\ 01\ 10\ 00\ 10\ 11\ 11 \\ \& & 01\ 01\ 01\ 01\ 01\ 01\ 01\ 01 \\ = & 00\ 01\ 01\ 00\ 00\ 00\ 01\ 01 \end{array}$$

- $X - ((X >> 1) \& 0x5555) =$

  $01\ 01\ 10\ 00\ 01\ 01\ 10\ 01$

Counting Bits Set

Mini-example(cont'd): $X = 01\ 01\ 10\ 00\ 01\ 01\ 10\ 01$
Next we compute $X = (X\&0x3333) + ((X >> 2)\&0x3333)$:

Counting Bits Set

Mini-example(cont'd): $X = 01\ 01\ 10\ 00\ 01\ 01\ 10\ 01$
Next we compute $X = (X\&0x3333) + ((X >> 2)\&0x3333)$:

- $X\&0x3333 =$

$$
\begin{array}{ll}
 & 0101\ 1000\ 0101\ 1001 \\
\& & 0011\ 0011\ 0011\ 0011 \\
= & 0001\ 0000\ 0001\ 0001
\end{array}
$$

Counting Bits Set

Mini-example(cont'd): $X = 01\ 01\ 10\ 00\ 01\ 01\ 10\ 01$
Next we compute $X = (X\&0x3333) + ((X >> 2)\&0x3333)$:

- $X\&0x3333 =$
$$\begin{array}{ll} & 0101\ 1000\ 0101\ 1001 \\ \& & 0011\ 0011\ 0011\ 0011 \\ = & 0001\ 0000\ 0001\ 0001 \end{array}$$
- $(X >> 2)\&0x3333 =$
$$\begin{array}{ll} & 0001\ 0110\ 0001\ 0110 \\ \& & 0011\ 0011\ 0011\ 0011 \\ = & 0001\ 0010\ 0001\ 0010 \end{array}$$

Counting Bits Set

Mini-example(cont'd): $X = 01\ 01\ 10\ 00\ 01\ 01\ 10\ 01$
Next we compute $X = (X \& 0x3333) + ((X >> 2) \& 0x3333)$:

- $X \& 0x3333 =$
  $$\begin{array}{ll} & 0101\ 1000\ 0101\ 1001 \\ \& & 0011\ 0011\ 0011\ 0011 \\ = & 0001\ 0000\ 0001\ 0001 \end{array}$$
- $(X >> 2) \& 0x3333 =$
  $$\begin{array}{ll} & 0001\ 0110\ 0001\ 0110 \\ \& & 0011\ 0011\ 0011\ 0011 \\ = & 0001\ 0010\ 0001\ 0010 \end{array}$$
- $(X \& 0x3333) + ((X >> 2) \& 0x3333) =$
  $0010\ 0010\ 0010\ 0011$

Counting Bits Set

Mini-example (cont'd): $X = 0010\ 0010\ 0010\ 0011$
Finally, $c = ((X + (X >> 4)\&0x0F0F) * 0x0101) >> 8$:

Counting Bits Set

Mini-example (cont'd): $X = 0010\ 0010\ 0010\ 0011$
Finally, $c = ((X + (X >> 4) \& 0x0F0F) * 0x0101) >> 8$:

- $X + (X >> 4) =$
$$\begin{array}{rl} & 0010\ 0010\ 0010\ 0011 \\ + & 0000\ 0010\ 0010\ 0010 \\ = & 0010\ 0100\ 0100\ 0101 \end{array}$$

Counting Bits Set

Mini-example (cont'd): $X = 0010\ 0010\ 0010\ 0011$
Finally, $c = ((X + (X >> 4)\&0x0F0F) * 0x0101) >> 8$:

- $X + (X >> 4) =$

$$\begin{array}{rl} & 0010\ 0010\ 0010\ 0011 \\ + & 0000\ 0010\ 0010\ 0010 \\ = & 0010\ 0100\ 0100\ 0101 \end{array}$$

- $X\&0x0F0F = 0000\ 0100\ 0000\ 0101$

Counting Bits Set

Mini-example (cont'd): $X = 0010\ 0010\ 0010\ 0011$
Finally, $c = ((X + (X >> 4) \& 0x0F0F) * 0x0101) >> 8$:

- $X + (X >> 4) =$

$$\begin{array}{rl} & 0010\ 0010\ 0010\ 0011 \\ + & 0000\ 0010\ 0010\ 0010 \\ = & 0010\ 0100\ 0100\ 0101 \end{array}$$

- $X \& 0x0F0F = 0000\ 0100\ 0000\ 0101$
- $X * 0x0101 =$

$$\begin{array}{rl} & 0000\ 0100\ 0000\ 0101 \\ + & 0000\ 0101\ 0000\ 0000 \\ = & 0000\ 1001\ 0000\ 0101 \end{array}$$

Counting Bits Set

Mini-example (cont'd): $X = 0010\ 0010\ 0010\ 0011$
Finally, $c = ((X + (X >> 4)\&0x0F0F) * 0x0101) >> 8$:

- $X + (X >> 4) =$
  $$\begin{array}{ll} & 0010\ 0010\ 0010\ 0011 \\ + & 0000\ 0010\ 0010\ 0010 \\ = & 0010\ 0100\ 0100\ 0101 \end{array}$$
- $X\&0x0F0F = 0000\ 0100\ 0000\ 0101$
- $X * 0x0101 =$
  $$\begin{array}{ll} & 0000\ 0100\ 0000\ 0101 \\ + & 0000\ 0101\ 0000\ 0000 \\ = & 0000\ 1001\ 0000\ 0101 \end{array}$$
- $c = X >> 8 = 1001$

Counting Bits Set

Mini-example (cont'd): $X = 0010\ 0010\ 0010\ 0011$
Finally, $c = ((X + (X >> 4)\&0x0F0F) * 0x0101) >> 8$:

- $X + (X >> 4) =$
  $$\begin{array}{rl} & 0010\ 0010\ 0010\ 0011 \\ + & 0000\ 0010\ 0010\ 0010 \\ = & 0010\ 0100\ 0100\ 0101 \end{array}$$
- $X\&0x0F0F = 0000\ 0100\ 0000\ 0101$
- $X * 0x0101 =$
  $$\begin{array}{rl} & 0000\ 0100\ 0000\ 0101 \\ + & 0000\ 0101\ 0000\ 0000 \\ = & 0000\ 1001\ 0000\ 0101 \end{array}$$
- $c = X >> 8 = 1001$
- Original $X = 0110110001011110$

Counting Bits Set

Solution: For 32-bit value X:

1. $X = X - ((X >> 1)\&0x55555555)$
2. $X = (X\&0x33333333) + ((X >> 2)\&0x33333333)$
3. $c = ((X + (X >> 4)\&0x0F0F0F0F) * 0x01010101) >> 24$

Explanation:

- First step computes bit counts for each successive pair of bits.
- Second step sums bit counts for bit pairs evaluated in step 1.
- Third step sums bit counts for pairs of pairs evaluated in step 2, then totals these sums together with the multiply by 0x01010101 and subsequent shift right by 24.

References

- Sean Andreson. Bit Twiddling Hacks. Available at `http://graphics.stanford.edu/~seander/bithacks.html`
- Charles Leiserson, Harald Prokop, and Keith H. Randall. Using deBruijn Sequences to Index a 1 in a Computer Word. July 7, 1998.
- M. Beeler, R. W. Gosper, and R. Schroeppel. HAKMEM. February 29, 1972. Available at `http://home.pipeline.com/~hbaker1/hakmem/hakmem.html`