・実験の目的

　符号なし、符号つきの２進数の様々な処理について考え、実際にプログラムを組んで実行してみることで、コンピュータの内部で処理がどのように進んでいるかについて理解を深める。

レポート課題1

　X(:xn-1, xn-2, xn-3,…x1x0)を符号付２進数として解釈すると、どのように表現することが出来るでしょうか？符号なしの場合と同じように、Σを用いて示しましょう。

・目的

　符号付２進数X(:xn-1, xn-2, xn-3,…x1,x0)の値を数式で表現する。

・道のり

　符号付２進数X(:xn-1, xn-2, xn-3,…x1,x0)はxn-1が0のとき非負、1のとき負の数であるから、xn-1によって場合分けをする。

　・xn-1が0のとき

　Xは非負でその絶対値は符号なし二進数(xn-2, xn-3,…x1,x0)であらわされるから、



　・xn-1が1のとき

　Xは負の数でその絶対値は二進数(xn-1, xn-2, xn-3,…x1,x0)の二の補数

(-xn-1+1,-xn-2+1,-xn-3+1,...,-x1+1,-x0+1)+1

であらわされるから、



・結果

xn-1が0のとき



xn-1が1のとき



・考察

　符号付２進数の値を数式であらわすときには2の補数の考え方を使うことが分かった。

レポート課題2

　Nビットの２進数データXが表現し得る整数の範囲はどうなるでしょう？符号付2進数として扱う場合と、符号なし２進数として扱う場合を、それぞれ答えてください。

・目的

Nビットの符号付二進数と符号なし二進数の範囲をNの数式であらわす。

・道のり

符号なし二進数の場合、最小値は0、最大値は１を足すと2Nになるので、

0<=X<2N

符号なし二進数の場合、最小値は-2N-1、最大値は1を足すと2N-1になるので、

-2 N-1<=X<2N-1

・結果

符号なし二進数の場合

0<=X<2N

符号つき二進数の場合

-2 N-1<=X<2N-1

・考察

　結果から、符号なし二進数と符号つき二進数の表現可能な範囲にはずれがあるが、表現し得る値の個数は同じであることが分かった。

レポート課題3

　符号付２進数の２の補数を求める方法を説明してください。

・回答

全てのビットを反転させて1を足すと符号付二進数の2の補数が得られる。

レポート課題4

　符号付２進数を「符号と絶対値」の形ではなく、２の補数の形で表現することには、どのような利点があるか説明してください。(２つ以上指摘できれば、応分の加点をします。)

・回答

「符号と絶対値」の形では、「000...」と「1000...」がどちらも0を意味するが、2の補数の形で表現することで、表現し得るすべてのビット列がすべて異なる値を表現でき、表現し得る値の範囲を最大限にすることが出来る。

レポート課題5

　計算機を構成している論理回路は、素量レベルでは１ビットずつの比較しかできません。従って、複数ビットの比較を行うには、１ビットの比較素量を組み合わせた、複数ビットの比較回路を用います。

　まず、符号なし２進数の比較について考えましょう。符号なし２進数X(:xn-1, xn-2, xn-3,…x1x0)とY(:yn-1, yn-2, yn-3,…y1y0)を比較するには、xn-1とyn-1から下の桁に向かって、１ビットずつ、値が異なるところまで比較していきます。

　値の異なる最初(最左)の桁がiとすると、xi=1かつyi=0ならばX>Yであり、 xi=0かつyi=1ならばX<Yということになります。結果的にすべてのビットが等しいならば、X=Yということになります。

　それでは、符号付２進数どうしの比較は、どのようにしたらよいでしょう？(やはり一ビットずつしか比較できないとします。)

・回答

　二つの符号付２進数

X(:xn-1, xn-2, xn-3,…x1x0)

Y(:yn-1, yn-2, yn-3,…y1y0)

を比較する。

・xn-1=0かつyn-1=0のとき

２つの符号なし２進数

X'(: xn-2, xn-3,…x1x0)

Y'(: yn-2, yn-3,…y1y0)

X'>Y'ならばX>Y

X'=Y'ならばX=Y

X'<Y'ならばX<Y

・xn-1=0かつyn-1=1のときX>Y

・xn-1=1かつyn-1=0のときX<Y

・xn-1=1かつyn-1=1のとき

２つの符号なし２進数

X'(: xn-2, xn-3,…x1x0)

Y'(: yn-2, yn-3,…y1y0)

X'>Y'ならばX<Y

X'=Y'ならばX=Y

X'<Y'ならばX>Y

選択問題６

　シフト演算には、左シフトと右シフト演算があります。

　左シフト演算は、対象データの各ビットの値をひとつ上位のビットに「同時に」移動する演算であり、符号の有無によらず、シフトの方法は同じです。１ビットの左シフトによって、対象データの２倍の値を得ることが出来ます。Nビットの左シフトは2n倍の乗算に相当するので、しばしば乗算の代わりに用いられます。

　右シフト演算は、対象データの各ビットの値をひとつ下位のビットに「同時に」移動する演算です。つまり１ビットの右シフトによって、約半分の値を求めることが出来ます。

　右へのシフトは、対象データを符号なし２進数として扱う場合と、符号付２進数として扱う場合とで、演算の内容が異なります。

　右シフトの対象データを、符号なし２進数とみなす場合(srl)には、最上位ビットに0を挿入します。

　一方、符号付とみなす場合(sra)は最上位ビットの値をそのまま残します。符号なし右シフトは論理右シフト(srl:Shift Right Logical)、符号付右シフトは算術右シフト(sra:Shift Right Arithmetic)とも呼ばれます。

　32ビットの2進数xを符号つきとみなして、１ビットの右シフトを31回繰り返す(31ビット右にシフトする)と、x<0の場合は-1になり、x>=0の場合は0になります。一方で、xを符号なしとみなして、31ビット右にシフトすると、x<0の場合は1になり、x>=0の場合は0になります。これを理解したうえで、次の問題を考えましょう。

　シフト演算のこの性質を、整数の非負の判定に応用すると、プログラムの実行時間を短縮できることがあります。これを確かめるために、図1.3のようなプログラムのif文を、シフト演算を用いて書き直し、両者(if文版とシフト演算版)の実行時間を比較してください。以下のプログラム:

<http://www.cis.shimane-u.ac.jp/~suzuki/cee/count.c>

を基に、if文やシフト演算を書き直して、実行時間の秒計測ができる時計で測りましょう。さらに、両者の実行時間がなぜ違うのか、理由を説明できれば加点します。

　なお、C言語において、符号付の整数(int)xを、符号なしとみなして、nビット右にシフトするには、「(unsigne)x >> n」と記述します。符号なし(unsigned)へのキャストを行わないと、符号「あり」右シフトを行ってしまうので注意しましょう。

|  |
| --- |
| int buf[] = {-1, 0, 1, -1, -1, -1, 0, 0, 1,…};  int count = 0;  …  if(buf[j] < 0)count++; |

図1.3:if文を使う場合

・目的

　符号の判定にシフト演算を使うことによって処理速度が上がることを確かめる。

・道のり

<http://www.cis.shimane-u.ac.jp/~suzuki/cee/count.c>

をダウンロードしてquestion6count.cとして保存し、

question6count.cのmain関数

int main() {

int count = 0;

int i, j;

for (i = 0; i < 1024 \* 128; i ++)

for (j = 0; j < 1024 \* 3; j ++)

if (buf[j] < 0) count ++;

printf("count: %d\n", count);

return 0;

}

に変更を加えて、

int main() {

int count = 0;

int i, j;

for (i = 0; i < 1024 \* 128; i ++)

for (j = 0; j < 1024 \* 3; j ++)

//if (buf[j] < 0) count ++;

count -= (buf[j] >> (sizeof(int) \* CHAR\_BIT - 1));

printf("count: %d\n", count);

return 0;

}

としたものをquestion6countshift.cとして保存した。

その後、

cc question6count.c -o question6count

cc question6countshift.c -o question6countshift

(time ./question6count) 2> question6counttime.txt

(time ./question6countshift) 2> question6countshifttime.txt

を実行して時間を測定した結果、

question6counttime.txt

real 0m1.681s

user 0m1.392s

sys 0m0.000s

question6countshifttime.txt

real 0m1.497s

user 0m1.168s

sys 0m0.000s

が得られた。

・結果

　実際にシフト演算を使った方が処理速度が上がることが確認できた。

レポート課題7

計算機が演算の溢れを検出する方法を調べよ。

　nビットの整数X(:xn-1, xn-2, xn-3,…x1x0)とY(:yn-1, yn-2, yn-3,…y1y0)の加算や減算の結果を、n+1ビットのZ(:zn, zn-1, zn-2,…z1z0)に格納することを考えましょう。この時、溢れを検出するには、Zの値をどのように利用すればいいでしょう？XやYを符号なしの数とみなす場合と、符号付の数とみなす場合を区別して考察しましょう。

　なお、その方法が正しいことを確かめるために、検出結果が表1.2と一致することを確認してください。(確認内容もレポートに明記してください。)

(ヒント:nビットの数同士の加減算の結果を溢れなく記録するのに必要なビット数を考えたうえで、演算結果のznとzn-1の値に注目しましょう。)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | A | B | C | A+B | A-C | C-A |
| 符号なしの場合 | 128 | 128 | 127 | 溢れ | 正常 | 溢れ |
| 符号つきの場合 | -128 | -128 | 127 | 溢れ | 溢れ | 溢れ |

表1.2:演算と溢れの例

・回答

・符号なし２進数の場合、実際の足し算ではznに格納される情報は失われるため、zn=1のときにあふれを起こしていることになる。

・符号つき２進数の場合、符号の異なるもの同士の足し算では絶対値が表現可能な範囲を超えることはできないので、溢れが起きるのは正の数同士、または負の数同士の足し算の場合に限る。正の数同士の足し算の場合は符号なし２進数の足し算と同じなので、zn=1のときにあふれを起こしていることになる。負の数同士の足し算の場合、Zの2の補数が2n-1以下であればよい。そしてZの2の補数が2n-1以下であることとzn=zn-1は同値であるから、zn!=zn-1のときにあふれを起こしていることになる。

上の表に基づいて符号なしの場合にはzn を、符号付の場合にはznとzn-1を並べて書くと、

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | A | B | C | A+B | A-C | C-A |
| 符号なしの場合 | 128 | 128 | 127 | 1 | 0 | 1 |
| 符号つきの場合 | -128 | -128 | 127 | 11 | 11 | 11 |

となり、表1.2と一致している。

レポート課題9

図1.4のような16進数の「足し算表」と「掛け算表」を作成せよ。手計算でも、プログラムによる表示結果でも、どちらでも構いません。

|  |
| --- |
| 1 2 3 4 5 6 7 8 9 a b c d e f  1 2 3 4 5 6 7 8 9 a b c d e f 10  ・・・  ・・・途中略・・・  ・・・  f 10 11 12 13 14 15 16 17 18 19 1a 1b 1c 1d 1e |

図1.4:16進数の足し算

・回答

足し算表を求めるプログラムquestion9add.c

#include<stdio.h>

#include<stdlib.h>

int main(void)

{

unsigned int a,b;

printf(" ");

for(b = 0x1; b < 0x10; b++)printf("%3x", b);

printf("\n");

for(a = 0x1; a < 0x10; a++)

{

printf("%1x ", a);

for(b = 0x1; b < 0x10; b++)printf("%3x", a+b);

printf("\n");

}

return EXIT\_SUCCESS;

}

掛け算表を求めるプログラムquestion9multi.c

#include<stdio.h>

#include<stdlib.h>

int main(void)

{

unsigned int a,b;

printf(" ");

for(b = 0x1; b < 0x10; b++)printf("%3x", b);

printf("\n");

for(a = 0x1; a < 0x10; a++)

{

printf("%1x ", a);

for(b = 0x1; b < 0x10; b++)printf("%3x", a\*b);

printf("\n");

}

return EXIT\_SUCCESS;

}

を作成し、

cc question9add.c -o question9add

cc question9multi.c -o question9multi

question9add > question9add.txt

question9multi > question9multi.txt

を実行した結果、

question9add.txt

1 2 3 4 5 6 7 8 9 a b c d e f

1 2 3 4 5 6 7 8 9 a b c d e f 10

2 3 4 5 6 7 8 9 a b c d e f 10 11

3 4 5 6 7 8 9 a b c d e f 10 11 12

4 5 6 7 8 9 a b c d e f 10 11 12 13

5 6 7 8 9 a b c d e f 10 11 12 13 14

6 7 8 9 a b c d e f 10 11 12 13 14 15

7 8 9 a b c d e f 10 11 12 13 14 15 16

8 9 a b c d e f 10 11 12 13 14 15 16 17

9 a b c d e f 10 11 12 13 14 15 16 17 18

a b c d e f 10 11 12 13 14 15 16 17 18 19

b c d e f 10 11 12 13 14 15 16 17 18 19 1a

c d e f 10 11 12 13 14 15 16 17 18 19 1a 1b

d e f 10 11 12 13 14 15 16 17 18 19 1a 1b 1c

e f 10 11 12 13 14 15 16 17 18 19 1a 1b 1c 1d

f 10 11 12 13 14 15 16 17 18 19 1a 1b 1c 1d 1e

question9multi.txt

1 2 3 4 5 6 7 8 9 a b c d e f

1 1 2 3 4 5 6 7 8 9 a b c d e f

2 2 4 6 8 a c e 10 12 14 16 18 1a 1c 1e

3 3 6 9 c f 12 15 18 1b 1e 21 24 27 2a 2d

4 4 8 c 10 14 18 1c 20 24 28 2c 30 34 38 3c

5 5 a f 14 19 1e 23 28 2d 32 37 3c 41 46 4b

6 6 c 12 18 1e 24 2a 30 36 3c 42 48 4e 54 5a

7 7 e 15 1c 23 2a 31 38 3f 46 4d 54 5b 62 69

8 8 10 18 20 28 30 38 40 48 50 58 60 68 70 78

9 9 12 1b 24 2d 36 3f 48 51 5a 63 6c 75 7e 87

a a 14 1e 28 32 3c 46 50 5a 64 6e 78 82 8c 96

b b 16 21 2c 37 42 4d 58 63 6e 79 84 8f 9a a5

c c 18 24 30 3c 48 54 60 6c 78 84 90 9c a8 b4

d d 1a 27 34 41 4e 5b 68 75 82 8f 9c a9 b6 c3

e e 1c 2a 38 46 54 62 70 7e 8c 9a a8 b6 c4 d2

f f 1e 2d 3c 4b 5a 69 78 87 96 a5 b4 c3 d2 e1

が得られた。

レポート課題10

次の3つの計算を筆算で行いなさい。いずれも符号なしと仮定してください。

・11001010111111102×10111010101111102

・cafe16×babe16

・cafebabe16×babecafe16

３つのプログラム

question10-1.c

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

const char \* const stra = "1100101011111110";

const char \* const strb = "1011101010111110";

int main(void)

{

unsigned int a,b;

unsigned int i,j;

a = 0;

b = 0;

for(i = 0; i < strlen(stra); i++)

{

a <<= 1;

if(stra[i] == '1')a++;

}

for(i = 0; i < strlen(strb); i++)

{

b <<= 1;

if(strb[i] == '1')b++;

}

printf(" ");

for(i = 0; i < strlen(strb); i++)printf(" ");

printf("%s\n", stra);

printf("x");

for(i = 0; i < strlen(stra); i++)printf(" ");

printf("%s\n", strb);

for(i = 0; i < 1 + strlen(stra) + strlen(strb); i++)printf("-");

printf("\n");

for(i = 0; i < strlen(strb); i++)

{

if(i == strlen(strb) - 1)printf("+");

else printf(" ");

for(j = 0; j < strlen(strb) - i; j++)printf(" ");

if(b >> i & 1)printf("%s", stra);

else for(j = 0; j < strlen(stra); j++)printf("0");

for(j = 0; j < i; j++)printf(" ");

printf("...%s x %c\n", stra, b >> i & 1 ? '1' : '0');

}

for(i = 0; i < 1 + strlen(stra) + strlen(strb); i++)printf("-");

printf("\n ");

for(i = 0; i < strlen(stra) + strlen(strb); i++)printf("%c", (a \* b) >> (strlen(stra) + strlen(strb) - i - 1) & 1 ? '1' : '0');

printf("\n");

return EXIT\_SUCCESS;

}

question10-2.c

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

#include<limits.h>

const unsigned int a = 0xcafe;

const unsigned int b = 0xbabe;

unsigned int getBit(unsigned int data, unsigned int digit);

unsigned int getDigit16(unsigned int data);

int main(void)

{

unsigned int i,j;

char stra[getDigit16(a) + 1], strb[getDigit16(b) + 1];

printf(" ");

for(i = 0; i < getDigit16(b); i++)printf(" ");

printf("%x\n", a);

printf("x");

for(i = 0; i < getDigit16(a); i++)printf(" ");

printf("%x\n", b);

for(i = 0; i < 1 + getDigit16(a) + getDigit16(b); i++)printf("-");

printf("\n");

sprintf(stra, "%x", a);

sprintf(strb, "%x", b);

for(i = 0; i < getDigit16(b); i++)

{

unsigned int bOneDigit;

char strBOneDigit[2];

strBOneDigit[1] = '\0';

strBOneDigit[0] = strb[getDigit16(b) - i - 1];

sscanf(strBOneDigit, "%x", &bOneDigit);

if(i == getDigit16(b) - 1)printf("+");

else printf(" ");

for(j = 0; j < getDigit16(a) + getDigit16(b) - getDigit16(a \* bOneDigit) - i; j++)printf(" ");

printf("%x", a \* bOneDigit);

for(j = 0; j < i; j++)printf(" ");

printf("...%s x %s\n", stra, strBOneDigit);

}

for(i = 0; i < 1 + getDigit16(a) + getDigit16(b); i++)printf("-");

printf("\n");

printf(" %x\n", a \* b);

return EXIT\_SUCCESS;

}

unsigned int getBit(unsigned int data, unsigned int digit)

{

return data >> digit & 1;

}

unsigned int getDigit16(unsigned int data)

{

unsigned int i;

for(i = sizeof(unsigned int) \* CHAR\_BIT - 1; i < sizeof(unsigned int) \* CHAR\_BIT; i--)if(getBit(data, i))break;

return (i + 3) / 4;

}

question10-3.c

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

#include<limits.h>

const unsigned long int a = 0xcafebabe;

const unsigned long int b = 0xbabecafe;

unsigned int getBit(unsigned long int data, unsigned int digit);

unsigned int getDigit16(unsigned long int data);

int main(void)

{

unsigned int i,j;

char stra[getDigit16(a) + 1], strb[getDigit16(b) + 1];

printf(" ");

for(i = 0; i < getDigit16(b); i++)printf(" ");

printf("%lx\n", a);

printf("x");

for(i = 0; i < getDigit16(a); i++)printf(" ");

printf("%lx\n", b);

for(i = 0; i < 1 + getDigit16(a) + getDigit16(b); i++)printf("-");

printf("\n");

sprintf(stra, "%lx", a);

sprintf(strb, "%lx", b);

for(i = 0; i < getDigit16(b); i++)

{

unsigned int bOneDigit;

char strBOneDigit[2];

strBOneDigit[1] = '\0';

strBOneDigit[0] = strb[getDigit16(b) - i - 1];

sscanf(strBOneDigit, "%x", &bOneDigit);

if(i == getDigit16(b) - 1)printf("+");

else printf(" ");

for(j = 0; j < getDigit16(a) + getDigit16(b) - getDigit16(a \* bOneDigit) - i; j++)printf(" ");

printf("%lx", a \* bOneDigit);

for(j = 0; j < i; j++)printf(" ");

printf("...%s x %s\n", stra, strBOneDigit);

}

for(i = 0; i < 1 + getDigit16(a) + getDigit16(b); i++)printf("-");

printf("\n");

for(i = 0; i < 1 + getDigit16(a) + getDigit16(b) - getDigit16(a \* b); i++)printf(" ");

printf("%lx\n", a \* b);

return EXIT\_SUCCESS;

}

unsigned int getBit(unsigned long int data, unsigned int digit)

{

return data >> digit & 1;

}

unsigned int getDigit16(unsigned long int data)

{

unsigned int i;

for(i = sizeof(unsigned long int) \* CHAR\_BIT - 1; i < sizeof(unsigned long int) \* CHAR\_BIT; i--)if(getBit(data, i))break;

return (i + 3) / 4;

}

を作成し、

for i in `seq 3`

do

cc question10-${i}.c -o question10-${i}

question10-${i} > question10-${i}.txt

done

を実行した結果、

question10-1.txt

1100101011111110

x 1011101010111110

---------------------------------

0000000000000000...1100101011111110 x 0

1100101011111110 ...1100101011111110 x 1

1100101011111110 ...1100101011111110 x 1

1100101011111110 ...1100101011111110 x 1

1100101011111110 ...1100101011111110 x 1

1100101011111110 ...1100101011111110 x 1

0000000000000000 ...1100101011111110 x 0

1100101011111110 ...1100101011111110 x 1

0000000000000000 ...1100101011111110 x 0

1100101011111110 ...1100101011111110 x 1

0000000000000000 ...1100101011111110 x 0

1100101011111110 ...1100101011111110 x 1

1100101011111110 ...1100101011111110 x 1

1100101011111110 ...1100101011111110 x 1

0000000000000000 ...1100101011111110 x 0

+ 1100101011111110 ...1100101011111110 x 1

---------------------------------

10010100000100110011010010000100

question10-2.txt

cafe

x babe

---------

b19e4...cafe x e

8b8ea ...cafe x b

7edec ...cafe x a

+8b8ea ...cafe x b

---------

94133484

question10-3.txt

cafebabe

x babecafe

-----------------

b19ee3664...cafebabe x e

be4ecf122 ...cafebabe x f

7edf34b6c ...cafebabe x a

983f0c0e8 ...cafebabe x c

b19ee3664 ...cafebabe x e

8b8f2062a ...cafebabe x b

7edf34b6c ...cafebabe x a

+8b8f2062a ...cafebabe x b

-----------------

94145db30d1b3484

が得られた。

・考察

　今回の実験では符号なし、符号つきの２進数の様々な処理について考え、課題6,9,10については実際にプログラムを組んで実行してみることで、コンピュータの内部で処理がどのように進んでいるかについて理解を深めることが出来た。Cなどの高級言語では2進数を意識することはほとんどないが、選択問題6のように高級言語であっても2進数の考えに基づくことによって処理速度を速めることが可能であることがわかり、良質なプログラムを作成するためにはハードウェアよりの知識が必要であると考えた。