

Rechnerstrukturen: Übungsblatt 5

Florian Ludewig (185722)

15. Juni 2020

Aufgabe 1

Aufgabe 1 (Ich nehme an wir können die Zahlen in Bitlänge 8 umrechnen)

$$B_0 = 1$$

$$\cdot 30 = 30$$

$$B_1 = -\frac{1}{2} \left[\left(\begin{smallmatrix} 2 \\ 0 \end{smallmatrix} \right) \cdot 1 \right] = -\frac{1}{2}$$

$$\cdot 30 = -15$$

$$B_2 = -\frac{1}{3} \left[\left(\begin{smallmatrix} 3 \\ 0 \end{smallmatrix} \right) \cdot 1 + \left(\begin{smallmatrix} 3 \\ 1 \end{smallmatrix} \right) \cdot B_1 \right] = -\frac{1}{3} \cdot \left(-\frac{1}{2} \right) = \frac{1}{6}$$

$$\cdot 30 = 5$$

$$B_3 = -\frac{1}{4} \left[\left(\begin{smallmatrix} 4 \\ 0 \end{smallmatrix} \right) \cdot 1 + \left(\begin{smallmatrix} 4 \\ 1 \end{smallmatrix} \right) \cdot \left(-\frac{1}{2} \right) + \left(\begin{smallmatrix} 4 \\ 2 \end{smallmatrix} \right) \cdot \frac{1}{6} \right] = 0$$

$$\cdot 30 = 0$$

$$B_4 = -\frac{1}{5} \left[\left(\begin{smallmatrix} 5 \\ 0 \end{smallmatrix} \right) \cdot 1 + \left(\begin{smallmatrix} 5 \\ 1 \end{smallmatrix} \right) \cdot \left(-\frac{1}{2} \right) + \left(\begin{smallmatrix} 5 \\ 2 \end{smallmatrix} \right) \cdot \frac{1}{6} + \left(\begin{smallmatrix} 5 \\ 3 \end{smallmatrix} \right) \cdot 0 \right] = -\frac{1}{30}$$

$$\cdot 30 = -1$$

a) $30: 00011110_2$

$$\begin{array}{r} 11100001 \\ + 00000001 \\ \hline 11100010 \end{array} \text{ EK}$$

$-15: 00001111_2$

$$\begin{array}{r} 00001111 \\ + 01110000 \\ \hline 00000001 \\ + 00000001 \\ \hline 01110000 \end{array} \text{ EK}$$

$5: 00000101_2$

$$\begin{array}{r} 11111010 \\ + 00000001 \\ \hline 11111011 \end{array} \text{ EK}$$

$0: 00000000_2$

$$\begin{array}{r} 11111111 \\ + 00000001 \\ \hline 11111111 \end{array} \text{ EK}$$

$-1: 10000001_2$

$$\begin{array}{r} 01111100 \\ + 00000001 \\ \hline 01111111 \end{array} \text{ EK}$$

... Overflow

b) $30: 30/8 = 3 \text{ R } 6 \uparrow$

$$3/8 = 0 \text{ R } 3$$

$$036_8$$

$-15: 11110001_2 \text{ EK (15)}$

$$= 241_{10}$$

$5: 5/8 = 0 \text{ R } 5$

$$005_8$$

$$361_8$$

$0: 0/8 = 0 \text{ R } 0$

$$000_8$$

$-1: 01111111_2 \text{ EK (1)}$

$$= 255_{10}$$

$$\begin{array}{r} 255/8 = 31 \text{ R } 7 \\ 31/8 = 3 \text{ R } 7 \\ 3/8 = 0 \text{ R } 3 \end{array}$$

$$377_8$$

c) $30: 30/16 = 1 \text{ R } 14 \uparrow$

$$1/16 = 0 \text{ R } 1$$

$$1E_{16}$$

$-15: 11110001_2 \text{ EK (15)}$

$$= 241_{10}$$

$$\begin{array}{r} 241/16 = 15 \text{ R } 1 \uparrow \\ 15/16 = 0 \text{ R } 15 \end{array}$$

$$F1_{16}$$

$0: 0/16 = 0 \text{ R } 0$

$$00_{16}$$

$-1: 01111111_2 \text{ EK (1)}$

$$= 255_{10}$$

$$\begin{array}{r} 255/16 = 15 \text{ R } 15 \uparrow \\ 15/16 = 0 \text{ R } 15 \end{array}$$

$$2FF_{16}$$

Aufgabe 2

```
1 #include <stdio.h>
2 #include <limits.h>
3 #include <time.h>
4 #include <stdlib.h>
5
6 unsigned W = sizeof(unsigned) * 8;
7 unsigned x, y;
8
9 int arithmeticRightShift(int x, int n) {
10     if (x < 0 && n > 0)
11         return x >> n | ~(~0U >> n);
12     else
13         return x >> n;
14 }
15
16 unsigned a() {
17     return ~(arithmeticRightShift((x|(~x+1)), W-1))&1;
18 }
19 unsigned b() {
20     return ~(arithmeticRightShift(x, W-1) << 1);
21 }
22 unsigned c() {
23     return ~(~x|(~y^(INT_MIN+INT_MAX)));
24 }
25 unsigned d() {
26     return x^(INT_MIN+INT_MAX);
27 }
28 unsigned e() {
29     return ((x^y)&~y)|(~(x^y)&y);
30 }
31 unsigned f() {
32     return arithmeticRightShift(((x<0) ? (x+3) : x), 2);
33 }
34
35 void test_all(unsigned expected_value) {
36     printf("a: %0x == %0x (match: %d)\n", expected_value, a(), expected_value
37             == a());
38     printf("b: %0x == %0x (match: %d)\n", expected_value, b(), expected_value
39             == b());
40     printf("c: %0x == %0x (match: %d)\n", expected_value, c(), expected_value
41             == c());
42     printf("d: %0x == %0x (match: %d)\n", expected_value, d(), expected_value
43             == d());
44     printf("e: %0x == %0x (match: %d)\n", expected_value, e(), expected_value
45             == e());
46     printf("f: %0x == %0x (match: %d)\n", expected_value, f(), expected_value
47             == f());
48 }
49
50 int main(void) {
51     srand(time(NULL));
52     x = rand();
53     y = rand();
54
55     printf("\n1:\n");
56     test_all(x);
57
58     printf("\n2:\n");
59     test_all(x&y);
60
61     printf("\n3 positive x:\n");
62     test_all((x<0?1:-1));
63     x = (~-1 * x)+1;
```

```

58     printf("\n3 negative x:\n");
59     test_all((x<0?1:-1));
60
61     return 0;
62 }
```

Die Ausgabe des obigen C-Programms ist:

```

1 1:
2 a: 47ad6e0f ?== 0 (match: 0)
3 b: 47ad6e0f ?== fffffff (match: 0)
4 c: 47ad6e0f ?== 43800800 (match: 0)
5 d: 47ad6e0f ?== b85291f0 (match: 0)
6 e: 47ad6e0f ?== 47ad6e0f (match: 1)
7 f: 47ad6e0f ?== 11eb5b83 (match: 0)
8
9 2:
10 a: 43800800 ?== 0 (match: 0)
11 b: 43800800 ?== fffffff (match: 0)
12 c: 43800800 ?== 43800800 (match: 1)
13 d: 43800800 ?== b85291f0 (match: 0)
14 e: 43800800 ?== 47ad6e0f (match: 0)
15 f: 43800800 ?== 11eb5b83 (match: 0)
16
17 3 positive x:
18 a: ffffffff ?== 0 (match: 0)
19 b: ffffffff ?== ffffffff (match: 1)
20 c: ffffffff ?== 43800800 (match: 0)
21 d: ffffffff ?== b85291f0 (match: 0)
22 e: ffffffff ?== 47ad6e0f (match: 0)
23 f: ffffffff ?== 11eb5b83 (match: 0)
24
25 3 negative x:
26 a: ffffffff ?== 0 (match: 0)
27 b: ffffffff ?== ffffffff (match: 1)
28 c: ffffffff ?== 0 (match: 0)
29 d: ffffffff ?== ffffffe (match: 0)
30 e: ffffffff ?== 1 (match: 0)
31 f: ffffffff ?== 0 (match: 0)
```

Wodurch sich folgende Lösungen ergeben:

- 1) e
- 2) c
- 3) b