Architektury systemów komputerowych

Lista zadań nr 5

Na zajęcia 9 kwietnia 2020

UWAGA! Rozwiązania muszą się stosować do wytycznych podanych w nagłówku poprzedniej listy zadań! Graf przepływu sterowania należy opisać przy pomocy języka Graphviz[1], prosty przykład podano tutaj[2].

Zadanie 1. Zaimplementuj funkcję zdefiniowaną poniżej w asemblerze x86-64. Taka procedura w języku C miałaby sygnaturę «`long cmp(uint64_t x, uint64_t y)`».

$$cmp(x,y) = \begin{cases} -1 & \text{gdy } x < y \\ 1 & \text{gdy } x > y \\ 0 & \text{gdy } x = y \end{cases}$$

Wskazówka: Rozwiązanie wzorcowe ma cztery wiersze (bez `ret`). Użyj instrukcji `adc`, `sbb` i `neg`.

Zadanie 2. Poniżej zamieszczono kod procedury o sygnaturze «`long puzzle2(char *s, char *d)`». Wyznacz **bloki podstawowe** oraz narysuj **graf przepływu sterowania**. Przetłumacz tę procedurę na język C, a następnie jednym zdaniem powiedz co ona robi.

```
1  puzzle2:
2        movq  %rdi, %rax
3  .L3:  movb  (%rax), %r9b
4        leaq  1(%rax), %r8
5        movq  %rsi, %rdx
6  .L2:  movb  (%rdx), %cl
7        incq  %rdx
8        testb %cl, %cl
9        je    .L4
10       cmpb  %cl, %r9b
11       jne   .L2
12       movq  %r8, %rax
13       jmp   .L3
14 .L4:  subq  %rdi, %rax
15       ret
```

Zadanie 3. Poniżej widnieje kod funkcji o sygnaturze «`uint32_t puzzle3(uint32_t n, uint32_t d)`». Wyznacz bloki podstawowe oraz narysuj graf przepływu sterowania, po czym przetłumacz tę funkcję na język C. Na podstawie ustępu „Mixing C and Assembly Language” strony GNU Assembler Examples[3] napisz program, który pomoże Ci powiedzieć co ta funkcja robi.

```
1  puzzle3:
2        movl  %edi, %edi
3        salq  $32, %rsi
4        movl  $32, %edx
5        movl  $0x80000000, %ecx
6        xorl  %eax, %eax
7  .L3:  addq  %rdi, %rdi
8        movq  %rdi, %r8
9        subq  %rsi, %r8
10       js    .L2
11       orl   %ecx, %eax
12       movq  %r8, %rdi
13 .L2:  shrl  %ecx
14       decl  %edx
15       jne   .L3
16       ret
```

[1] `https://hackmd.io/s/features#Graphviz`
[2] `http://www.tonyballantyne.com/graphs.html#orgheadline10`
[3] `http://cs.lmu.edu/~ray/notes/gasexamples/`

Zadanie 4. Poniżej zamieszczono kod rekurencyjnej procedury o sygnaturze «`int puzzle4(long *a, long v, uint64_t s, uint64_t e)`». Wyznacz bloki podstawowe oraz narysuj graf przepływu sterowania. Przetłumacz tę procedurę na język C, a następnie jednym zdaniem powiedz co ona robi.

```
1  puzzle4:
2        movq  %rcx, %rax
3        subq  %rdx, %rax
4        shrq  %rax
5        addq  %rdx, %rax
6        cmpq  %rdx, %rcx
7        jb    .L5
8        movq  (%rdi,%rax,8), %r8
9        cmpq  %rsi, %r8
10       je    .L10
11       cmpq  %rsi, %r8
12       jg    .L11
13       leaq  1(%rax), %rdx
14       call  puzzle4
15 .L10: ret
16 .L11: leaq  -1(%rax), %rcx
17       call  puzzle4
18       ret
19 .L5:  movl  $-1, %eax
20       ret
```

Wskazówka: Z reguły procedurę «`puzzle4`» woła się następująco: «`i = puzzle4(a, v, 0, n - 1)`».

Zadanie 5. Poniższy kod w asemblerze otrzymano w wyniku deasemblacji funkcji zadeklarowanej jako «`long switch_prob(long x, long n)`». Zapisz w języku C kod odpowiadający tej funkcji.

```
1  400590 <switch_prob>:
2  400590: 48 83                    subq  $0x3c,%rsi
3  400594: 48 83 fe 05              cmpq  $0x5,%rsi
4  400598: 77 29                    ja    *0x4005c3
5  40059a: ff 24 f5 f8 06 40 00     jmpq  *0x4006f8(,%rsi,8)
6  4005a1: 48 8d 04 fd 00 00 00 00  lea   0x0(,%rdi,8),%rax
7  4005a9: c3                       retq
8  4005aa: 48 89 f8                 movq  %rdi,%rax
9  4005ad: 48 c1 f8 03              sarq  $0x3,%rax
10 4005b1: c3                       retq
11 4005b2: 48 89 f8                 movq  %rdi,%rax
12 4005b5: 48 c1 e0 04              shlq  $0x4,%rax
13 4005b9: 48 29 f8                 subq  %rdi,%rax
14 4005bc: 48 89 c7                 movq  %rax,%rdi
15 4005bf: 48 0f af ff              imulq %rdi,%rdi
16 4005c3: 48 8d 47 4b              leaq  0x4b(%rdi),%rax
17 4005c7: c3                       retq
```

Zrzut pamięci przechowującej tablicę skoków:

```
18 (gdb) x/6gx 0x4006f8
19 0x4006f8: 0x4005a1
20 0x400700: 0x4005a1
21 0x400708: 0x4005b2
22 0x400710: 0x4005c3
23 0x400718: 0x4005aa
24 0x400720: 0x4005bf
```

Zadanie 6. Poniżej zamieszczono kod procedury o sygnaturze «`struct T puzzle8(long *a, long n)`». Na jego podstawie podaj definicję typu «`struct T`». Przetłumacz tę procedurę na język C, po czym jednym zdaniem powiedz co ona robi. Gdyby sygnatura procedury nie była wcześniej znana to jaką należałoby wywnioskować z poniższego kodu?

```
1  puzzle8:
2        movq  %rdx, %r11
3        xorl  %r10d, %r10d
4        xorl  %eax, %eax
5        movq  $LONG_MIN, %r8
6        movq  $LONG_MAX, %r9
7  .L2:  cmpq  %r11, %r10
8        jge .L5
9        movq  (%rsi,%r10,8), %rcx
10       cmpq  %rcx, %r9
11       cmovg %rcx, %r9
12       cmpq  %rcx, %r8
13       cmovl %rcx, %r8
14       addq  %rcx, %rax
15       incq  %r10
16       jmp .L2
17 .L5:  cqto
18       movq  %r9, (%rdi)
19       idivq %r11
20       movq  %r8, 8(%rdi)
21       movq  %rax, 16(%rdi)
22       movq  %rdi, %rax
23       ret
```

Wskazówka: Zauważ, że wynik procedury nie mieści się w rejestrach `%rax` i `%rdx`, zatem zostanie umieszczony w pamięci.