Assembly: Tracing; conditionals intro

COSC 208, Introduction to Computer Systems, 2021-10-13

Announcements

• Project 2 Part 1 due next Thursday, at 11pm

Outline

- Warm-up
- Tracing assembly code
- Conditionals

Warm-up

• Q1: Write the C code equivalent for each line of assembly, treating registers as if they were variable names. For example, the C code equivalent for sub sp, sp, #0x20 is sp = sp - 0x20

```
0000000000400544 <sum>:
   400544: d10083ff sub sp, sp, #0x20
   400548: b9001fe0 str w0, [sp, #28]
   40054c: f9000be1
                      str x1, [sp, #16]
                       ldr x8, [sp, #16]
   400550: f9400be8
                       ldr w9, [x8]
   400554: b9400109
   400558: b9000fe9
                       str w9, [sp, #12]
                       ldr w9, [sp, #28]
   40055c: b9401fe9
   400560: b9400fea
                       ldr w10, [sp, #12]
   400564: 0b0a0129
                       add w9, w9, w10
   400568: b9000be9
                       str w9, [sp, #8]
   40056c: b9400be0
                       ldr w0, [sp, #8]
   400570: 910083ff
                       add sp, sp, #0x20
```

Tracing assembly code

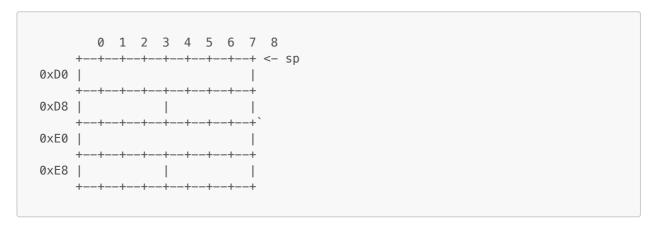
• Q2: The assembly corresponds to the following C code. Label each line of assembly code with the line number of the line of C code from which the assembly instruction was derived.

```
1 int sum(int a, int *b) {
2   int c = *b;
3   int d = a + c;
4   return d;
5 }
```

```
0000000000400544 <sum>:

400544: d10083ff sub sp, sp, #0x20
400548: b9001fe0 str w0, [sp, #28]
400550: f9400be1 str x1, [sp, #16]
400554: b9400109 ldr w9, [x8]
400558: b9000fe9 str w9, [sp, #12]
40055c: b9401fe9 ldr w9, [sp, #28]
400560: b9400fea ldr w10, [sp, #12]
400564: 0b0a0129 add w9, w9, w10
400568: b9000be9 str w9, [sp, #8]
400560: b9400be0 ldr w0, [sp, #8]
400570: 910083ff add sp, sp, #0x20
```

Q3: Place in the stack below the parameters a, b and local variables c and d (before executing last assembly instruction; and assuming sp = 0xF0 initially)



Conditionals

• Q4: The following C code was compiled into assembly. Label each line of assembly code with the line number of the line of C code from which the assembly instruction was derived.

```
int divide(int numerator, int denominator) {
   int result = -1;
   result = numerator / denominator;
   return result;
}
```

• Q5: Why is #0xffffffff being stored in w8?

• Q6: When might this function cause an error?

• The following code prevents this error.

```
int divide_safe(int numerator, int denominator) {
  int result = -1;
  if (denominator != 0) {
    result = numerator / denominator;
  }
  return result;
}
```

Conditional assembly code

• Q7: Its compiled assembly include a branch. Label each line of assembly code with the line number of the line of C code from which the assembly instruction was derived

• Q8: What does the cbz instruction do?

Q9: Why does the assembly use cbz when the C code contains != ∅?