Tags: #cec32x, #hw, cec32x_HW_40–HW10_structured_programming, 134 points

HW 10. Structured Programming

Prob 1. (20 points) Consider the problem on page 3 of the class notes for Module 10. Write the assembly code if we want to perform ‘’A = A + B * (C + D)”. Note that you need to use as few registers as possible.

Prob 2. We can see from page 9 of the class notes for Module 10 that the algorithms of C and assembly can be very different since C does not have statements that are corresponding to some of the efficient instructions of assembly. Here, we will see and understand the details by stepping through the code. Specifically, we use a number x = 0x9D000000 as a test case.

Prob 2.1 (63 points, 3 points each) You need to determine the values of count, x, y, z, and u, as shown on page 9 of notes for Module 10:

- Initialization: count = ?
- In the first pass of the while loop:

```
* x = ? (after executing line 70)
* y = ? (after executing line 71)
* z = ? (after executing line 72)
* u (= x<<1) = ? (on line 73)
* count = ? (after running the if-else statement)
```

- In the second pass of the while loop:

```
* x = ? (after executing line 70)
* y = ? (after executing line 71)
* z = ? (after executing line 72)
* u (= x<<1) = ? (on line 73)
* count = ? (after running the if-else statement)
```

- In the third pass of the while loop:

```
* x = ? (after executing line 70)
* y = ? (after executing line 71)
* z = ? (after executing line 72)
* u (= x<<1) = ? (on line 73)
* count = ? (after running the if-else statement)
```

- In the fourth pass of the while loop:

```
* x = ? (after executing line 70)
* y = ? (after executing line 71)
* z = ? (after executing line 72)
* u (= x<<1) = ? (on line 73)
* count = ? (after running the if-else statement)
```

Prob 2.2 (51 points, 3 points each) You also need to determine the values of r1, r0, and C shown on page 10 of the notes:

- Initialization: r1 = ? (after executing line 125)
- In the first pass of the while loop:

```
* r0 = ? (while executing line 126)
* r0 = ? (after executing line 127)
* C flag = ? (after executing line 127)
* r1 = ? (after executing line 128)
```

- In the second pass of the while loop:

```
* r0 = ? (while executing line 126)
* r0 = ? (after executing line 127)
* C flag = ? (after executing line 127)
* r1 = ? (after executing line 128)
```

- In the third pass of the while loop:

```
* r0 = ? (while executing line 126)
* r0 = ? (after executing line 127)
* C flag = ? (after executing line 127)
* r1 = ? (after executing line 128)
```

- In the fourth pass of the while loop:

```
* r0 = ? (while executing line 126)
* r0 = ? (after executing line 127)
* C flag = ? (after executing line 127)
* r1 = ? (after executing line 128)
```