Models of Computation II, Exercises 2: The Universal RM

1.

$$\begin{aligned}
\ulcorner B_0 \urcorner &= \langle\!\langle 2 \times 1 + 1, \langle 1, 6 \rangle \rangle\!\rangle = \langle\!\langle 3, 2^1(2 \times 6 + 1) - 1 \rangle\!\rangle \\
&= \langle\!\langle 3, 25 \rangle\!\rangle = 2^3(2 \times 25 + 1) = 8 \times 51 = 408
\end{aligned}$$

$$\begin{aligned}
\ulcorner B_1 \urcorner &= \langle\!\langle 2 \times 2 + 1, \langle 2, 4 \rangle \rangle\!\rangle = \langle\!\langle 5, 2^2(2 \times 4 + 1) - 1 \rangle\!\rangle \\
&= \langle\!\langle 5, 35 \rangle\!\rangle = 2^5(2 \times 35 + 1) = 32 \times 71 = 2272
\end{aligned}$$

$$\ulcorner B_2 \urcorner = \langle\!\langle 2 \times 0, 3 \rangle\!\rangle = \langle\!\langle 0, 3 \rangle\!\rangle = 2^0(2 \times 3 + 1) = 7$$

$$\ulcorner B_3 \urcorner = \langle\!\langle 2 \times 3, 1 \rangle\!\rangle = \langle\!\langle 6, 1 \rangle\!\rangle = 2^6(2 \times 1 + 1) = 192$$

$$\begin{aligned}
\ulcorner B_4 \urcorner &= \langle\!\langle 2 \times 3 + 1, \langle 5, 0 \rangle \rangle\!\rangle = \langle\!\langle 7, 2^5(2 \times 0 + 1) - 1 \rangle\!\rangle \\
&= \langle\!\langle 7, 31 \rangle\!\rangle = 2^7(2 \times 31 + 1) = 8064
\end{aligned}$$

$$\ulcorner B_5 \urcorner = \langle\!\langle 2 \times 2, 4 \rangle\!\rangle = \langle\!\langle 4, 4 \rangle\!\rangle = 2^4(2 \times 4 + 1) = 144$$

$$\ulcorner B_6 \urcorner = 0$$

2. (a) To decode the program, we first decode it as a list l and then decode each individual instruction. First, observe that

$$2^{216} \times 833 = 2^{216}(2 \times 416 + 1) = \langle\!\langle 216, 416 \rangle\!\rangle$$

so $l = 216 :: l_1$ for some list l_1 with $\ulcorner l_1 \urcorner = 416$.
To decode l_1, we need to find x, y such that $2^x(2y+1) = 416$. We should therefore work out what power of 2 divides 416, which we can do by repeatedly factoring out 2:

$$416 = 2 \times 208 = 2^2 \times 104 = 2^3 \times 52 = 2^4 \times 26 = 2^5 \times 13$$

Now, 2 does not divide 13, but $2 \times 6 + 1 = 13$. Therefore $\ulcorner l_1 \urcorner = 2^5(2 \times 6 + 1) = \langle\!\langle 5, 6 \rangle\!\rangle$. Consequently, $l_1 = 5 :: l_2$ for some list l_2 with $\ulcorner l_2 \urcorner = 6$.
To decode l_2, we need to find x, y such that $2^x(2y + 1) = 6$. It is easy to see that the solution is $x = y = 1$, so $l_2 = 1 :: l_3$ for some list l_3 with $\ulcorner l_3 \urcorner = 1$.
Now $1 = 2^0(2 \times 0 + 1)$, so $l_3 = 0 :: l_4$ for some list l_4 with $\ulcorner l_4 \urcorner = 0$. It must be that $l_4 = []$.
Thus, we have

$$l = [216, 5, 1, 0]$$

Now we have to decode each instruction.

216 is non-zero, so it represents either an increment or decrement instruction. To find out which, we should decode it as $\langle\langle x, y\rangle\rangle = 2^x(2y+1)$:

$$216 = 2 \times 108 = 2^2 \times 54 = 2^3 \times 27 = 2^3(2 \times 13 + 1) = \langle\langle 3, 13\rangle\rangle$$

Now $3 = 2 \times 1 + 1$, so the instruction is a decrement to register 1. To determine which labels the instruction goes to, we need to decode 13 as $\langle j, k\rangle = 2^j(2k+1) - 1$:

$$13 = 14 - 1 = 2 \times 7 - 1 = 2^1(2 \times 3 + 1) - 1 = \langle 1, 3\rangle$$

We have determined that $216 = \langle\langle 2 \times 1 + 1, \langle 1, 3\rangle\rangle\rangle$, so we have

$$L_0 : R_1^- \to L_1, L_3$$

5 is also non-zero.

$$5 = 2^0(2 \times 2 + 1) = \langle\langle 0, 2\rangle\rangle = \langle\langle 2 \times 0, 2\rangle\rangle$$

This therfore represents the instruction that increments R_0 and jumps to L_2:

$$L_1 : R_0^+ \to L_2$$

1 is also non-zero.

$$1 = 2^0(2 \times 0 + 1) = \langle\langle 0, 0\rangle\rangle = \langle\langle 2 \times 0, 0\rangle\rangle$$

This therefore represents the instruction that increments R_0 and jumps to R_0:

$$L_2 : R_0^+ \to L_0$$

0 represents the halting instruction:

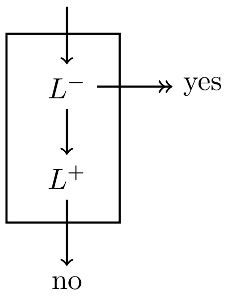
$$L_3 : HALT$$

Consequently, the complete decoded program is:

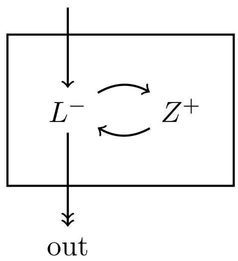
$$\begin{array}{l} L_0 : R_1^- \to L_1, L_3 \\ L_1 : R_0^+ \to L_2 \\ L_2 : R_0^+ \to L_0 \\ L_3 : HALT \end{array}$$

(b) The program increments R_0 by twice the initial value of R_1. If f is the function of one argument computed by the register machine with this program, $f(x)$ is the final value of R_0 when the machine is run from the intial state with $R_0 = 0$ and $R_1 = x$. Therefore $f(x) = 2x$ — the machine computes the doubling function.

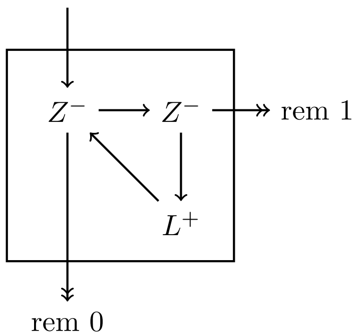
3. (a) "test $L = 0$":



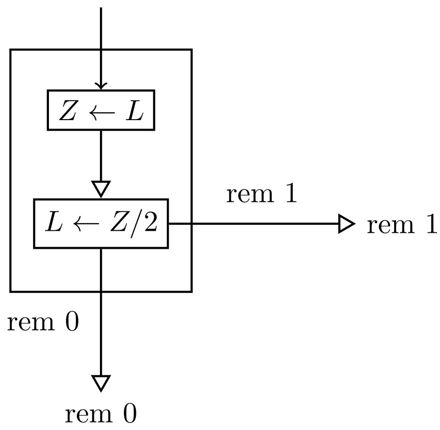
(b) "$Z \leftarrow L$":



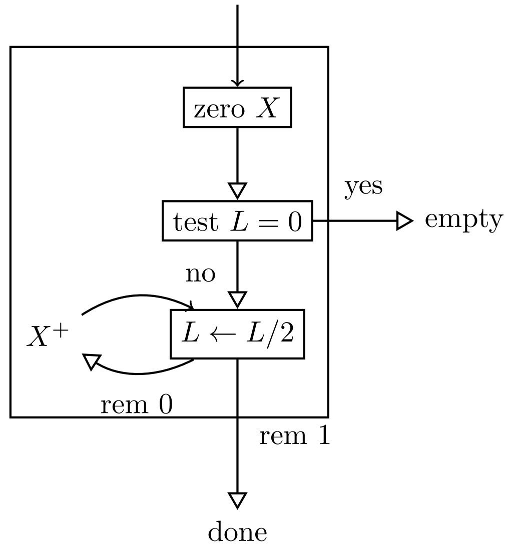
(c) "$L \leftarrow Z/2$":



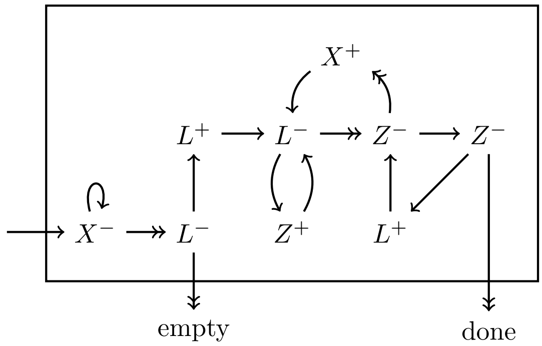
(d) "$L ::= L/2$":



(e) "$\langle\!\langle X, L \rangle\!\rangle ::= L$":



(f) If you have made the same design choices as here, you should get something like:



This should be familiar as the implementation of the "pop L to X" graph component from the lectures.

4. No solution, but a hint: it is possible to simulate three registers using two registers by representing the values x, y, z of the three registers as $2^x 3^y 5^z$.