Exercise 1:

State consists of registers `X`, `A` (`HI`), `LOW` plus `C` and `Z` bits, plus two inputs `F1` and `F2`. A reasonable representation for State the following tuple:

```
   type State = ([8] // F1
                ,[8] // F2
                ,[8] // A
                ,[8] // X
                ,[8] // LOW
                ,Bit // C (Carry)
                ,Bit // Z (Zero)
                )
```

Exercise 2:

Since `step04` follows from `step03` and the only change is to the Carry bit `C`, `step03` can be written as the following:

```
  step03 : State - > State
  step03 (f1,f2,a,x,l,_,z) = step04 (f1,f2,a,x,l,False,z)
```

Exercise 3:

```
  step04: State -> State
  step04 (f1,f2,a,x,l,c,z) = step05 (f1',f2,a,x,l,b0,z)
    where
      [b7,b6,b5,b4,b3,b2,b1,b0] = f1 // LSB is on the right
      f1' = [c,b7,b6,b5,b4,b3,b2,b1] // ROR – right rotation through carry
```

Exercise 4:

```
  step05 : State -> State
  step05 (f1,f2,a,x,l,c,z) = if (c == 0) then step08 (f1,f2,a,x,l,c,z)
                                         else step06 (f1,f2,a,x,l,c,z)
```

Exercise 5:

```
  step06 : State -> State
  step06 (f1,f2,a,x,l,_,z) = step07 (f1,f2,a,x,l,False,z)
```

Exercise 6:

```
step07 : State -> State
step07 (f1, f2, a, x, l, c, z) = step08 (f1, f2, a', x, l, c', z')
  where
    a' = a + f2 + (if c then (1:[8]) else (0:[8]))
    a'Large : [9]
    a'Large = (zero # a) + (zero # f2) + (if c then (1:[9]) else (0:[9]))
    c' = a'Large > (255:[9])
    z' = a' == 0
```

Galois, Inc. © 2023 | All rights reserved

Exercise 7:

```
step08 : State -> State
step08 (f1,f2,a,x,l,c,z) = step09 (f1,f2,a',x,l,a0,z)
  where
    [a7,a6,a5,a4,a3,a2,a1,a0] = a // LSB is on the right
    a' = [c,a7,a6,a5,a4,a3,a2,a1] // ROR - result is the right rotation

step09 : State -> State
step09 (f1,f2,a,x,l,c,z) = step10 (f1,f2,a,x,l',l0,z)
  where
    [l7,l6,l5,l4,l3,l2,l1,l0] = l // LSB is on the right
    l' = [c,l7,l6,l5,l4,l3,l2,l1] // ROR - result is the right rotation
```

Exercise 8:

```
step10 : State -> State
step10 (f1,f2,a,x,l,c,z) = step11 (f1,f2,a,x',l,c,x'==0)
  where x' = x-1  // need a ‘where’ since x’ is used twice above
```

Exercise 9:

```
step11 : State -> State
step11 (f1,f2,a,x,l,c,z) = if (z == 0) then step04 (f1,f2,a,x,l,c,z)
                                       else (f1,f2,a,x,l,c,z)
```

Exercise 10:

```
legato : ([8],[8],State) -> ([8],[8])
legato (f1,f2,st) = (hi,lo)
  where
    // get the relevant parts to construct the starting state
    (_,_,A,X,LOW,C,Z) = st
    // Run legato multiplier; final A is hi; and final LOW is lo
    (_,_,hi,_,lo,_,_) = step01 (f1,f2,A,X,LOW,C,Z)
```

Galois, Inc. © 2023 | All rights reserved