```
Entry
  \{v0\_1 := \phi [v0, Entry] [v0\_2, Z]
             := \phi [v1, Entry] [v1_2, Z]
    v1 1
               := \phi [0, Entry] [i_1, Z]
    phi_mul := \phi [0, Entry] [next_mul, Z] 
    exitcond := (i == 32)
    i 1
           := i + 1
    next mul := phi mul + 0x9e3779b9
Y
               := v1 1 << 4
    tmp
    tmp1
               := tmp + k0 read
               := v1 1 >> 5
    tmp2
    tmp3
               := tmp2 + k1 read
    tmp4
               := v1 1 + next mul
    tmp5
               := tmp3 xor tmp4
    tmp6
               := tmp5 xor tmp1
    v0 2
               := tmp6 + v0 1
               = v0 2 << 4
    tmp7
    tmp8
               := tmp7 + k2 read
           = v0 2 >> 5
    tmp9
    tmp10 := tmp9 + k3 read
    tmp11 := v0 2 + next mul
    tmp12 := tmp11 xor tmp8
    tmp13
               := tmp12 + tmp10
    v1 2
               := tmp13 + v1 1
   {v0 1
X
               := \phi [v0, Entry] [v0_2, Z]
    v1 1
               := \phi [v1, Entry] [v1_2, Z]
               := \phi [0, Entry] [i_1, Z]
    phi_mul := \phi [0, Entry] [next_mul, Z] }
               := ( i == 32)
    exitcond
    i 1
               := i + 1
               := phi_mul + 0x9e3779b9
    next mul
Y
               := v1 1 << 4
    tmp
    tmp1
               := tmp + k0 read
                                                         S_{loop}
               := v1 1 >> 5
    tmp2
               := tmp2 + k1 read
    tmp3
    tmp4
               := v1 1 + next mul
Z
               := tmp3 xor tmp4
    tmp5
    tmp6
               := tmp5 xor tmp1
    v0 2
               := tmp6 + v0 1
               = v0 2 << 4
    tmp7
    tmp8
               := tmp7 + k2 read
               = v0 2 >> 5
    tmp9
    tmp10
               := tmp9 + k3 read
    tmp11
               := v0 2 + next mul
    tmp12
               := tmp11 xor tmp8
    tmp13
               := tmp12 + tmp10
    v1 2
               := tmp13 + v1 1
   {v0_1
               := \phi [v0, Entry] [v0_2, Z]
               := \phi [v1, Entry] [v1_2, Z]
               := \phi [0, Entry] [i_1, Z]
    phi mul := \phi [0, Entry] [next mul, Z] }
               := (i == 32)
    exitcond
                := i + 1
    i 1
                                                       S_{preExit}
```

Exit