17.Write a program to perform Booth’s multiplication of two signed numbers using any high level language

.

from bitwise import \*

def main():

print("This program excecutes Booth's multiplication algorithm.\n")

print("The formula it's going to calculate is: M \* R = ?")

print("Input the bit length of first variable M: ", end="")

mlen = int(input())

print("Input the bit length of second variable R: ", end="")

rlen = int(input())

print("Input the number of first variable M: ", end="")

m = int(input())

if m < 0:

m = TwoComp( ("{0:0%db}" % mlen).format(m) ) #Calculate the two's complement number of m

else:

m = ("{0:0%db}" % mlen).format(m) #Convert to bits and assign directly

print("Input the number of second variable R: ", end="")

r = int(input())

if r < 0:

r = TwoComp( ("{0:0%db}" % rlen).format(r) )

else:

r = ("{0:0%db}" % rlen).format(r)

ilen = mlen + rlen + 1 #The common length of internal variables

a = m + GenZeroStr(rlen + 1) #A: place M in leftmost position. Fill the left bits with 0.

s = TwoComp(m) + GenZeroStr(rlen + 1) #S: place negative M in leftmost position.

p = GenZeroStr(mlen) + r + "0" #P: place R by rightmost 0.

print("Internal variables:")

print("M = %s" % m)

print("R = %s" % r)

print("A = %s" % a)

print("S = %s" % s)

print("P = %s\n" % p)

for i in range(rlen): #Do operation rlen times

print("Step %d:" % (i+1))

op = p[-2:]

print(" " + "The last 2 bits of p are: %s" % "".join(op))

if op == "10":

print(" " + "P = (P+S) >> 1")

p = BitAdd(p, s, len(p))

elif op == "01":

print(" " + "P = (P+A) >> 1")

p = BitAdd(p, a, len(p))

elif op == "00":

print(" " + "P = P >> 1")

elif op == "11":

print(" " + "P = P >> 1")

p = BitShift(p, 1)

print(" " + "P = %s\n" % p)

p = p[:-1]

print("The answer is: %s" % p)

if \_\_name\_\_ == "\_\_main\_\_":

main()