8-Bit Sequential Multiplier

Aim: The project is the design of a 8-bit sequential multiplier, with 8-bit A and B inputs and a 16-bit result. This multiplier has an 8-bit bi-directional I/O for inputting its A and B operands, and outputting its 16-bit output one byte at a time.

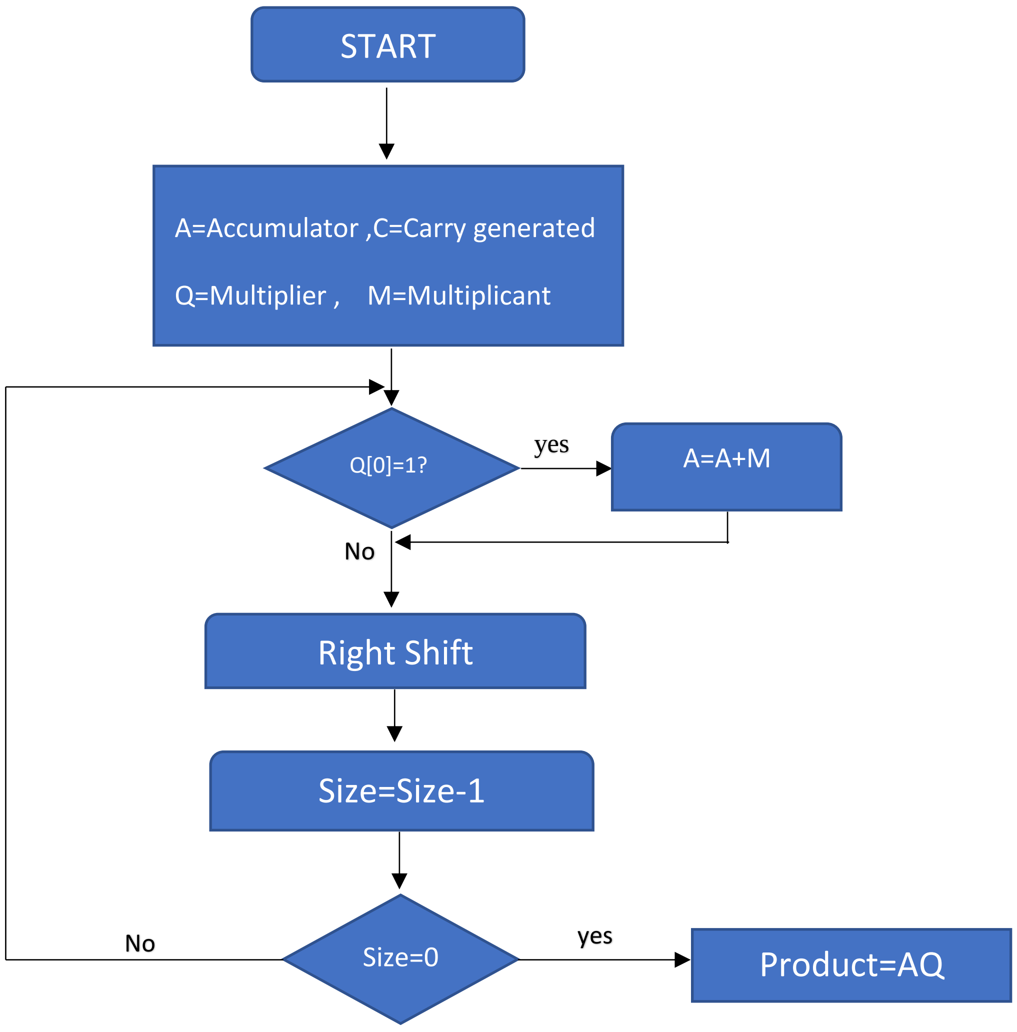
Language Used: Python

Project Description:

When designing multipliers there is always a compromise to be made between how fast the multiplication process is done and how much hardware we are using for its implementation. A simple multiplication method that is slow, but efficient in use of hardware is the shift-and-add method. In this method, depending on bit i of operand A, either operand B is added to the collected partial result and then shifted to the right (when bit i is 1), or (when bit i is 0) the collected partial result is shifted one place to the right without being added to B. This method is justified by considering how binary multiplication is done manually.

Algorithm:

Flow Chart:

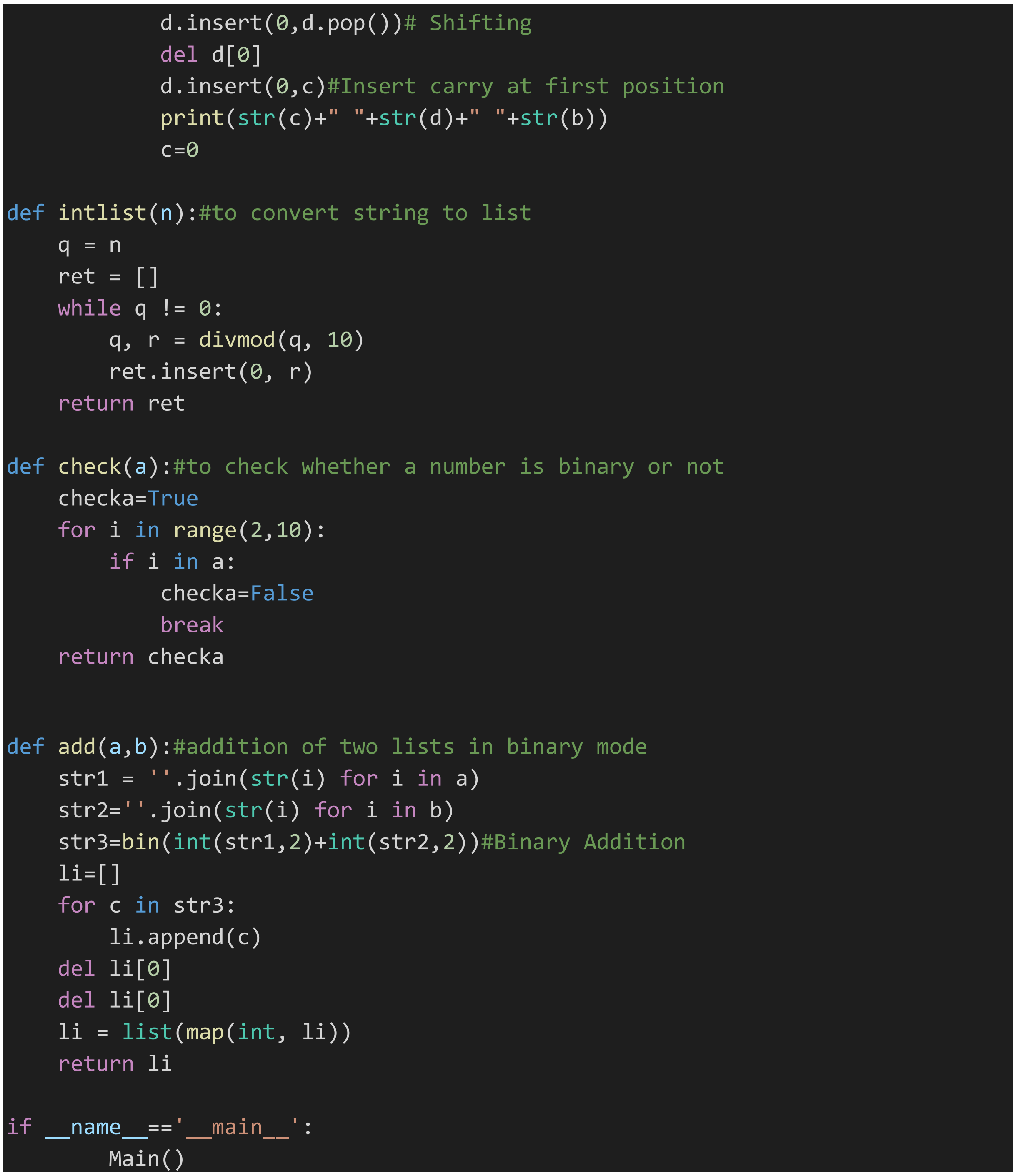


CODE:

```python
def Main():
    a=input('Enter multiplicand: ')
    b=input('Enter multiplier: ')
    a=intlist(a)
    b=intlist(b)
    if len(a)>8 or len(b)>8:# Check if numbers are of 8 bit or not
        print('Invalid')
        exit()
    checka=check(a)#Calling check function
    checkb=check(b)
    if checka==False or checkb==False:
        print("Invalid Values")
        exit()
    if len(a)<8:# put leading zeroes if number is less than 8 bits
        for _ in range(8-len(a)):
            a.insert(0, 0)
    if len(b)<8:
        for _ in range(8-len(b)):
            b.insert(0, 0)

    c=0

    print(a)
    print(b)
    d=[0,0,0,0,0,0,0,0]#Initial zero 8 bit number
    for _ in range(1,9):
        if b[-1]==1:# To check if last digit in one
            d=add(a,d)# adding multiplicand and Zero array
            if len(d)>8:# Check for carry
                c=d[0]
                del d[0]
            b.insert(0,b.pop())# Shifting
            del b[0]
            b.insert(0,d[-1])
            d.insert(0,d.pop())#Shifting
            del d[0]
            d.insert(0,c)#Insert carry at first position
            print(str(c)+" "+str(d)+" "+str(b))
            c=0
        elif b[-1]==0:# To check if last bit is zero
            if len(d)>8:# Check for carry
                c=d[0]
                del d[0]
            b.insert(0,b.pop())# Shifting
            del b[0]
            b.insert(0,d[-1])
```



```python
            d.insert(0,d.pop())# Shifting
            del d[0]
            d.insert(0,c)#Insert carry at first position
            print(str(c)+" "+str(d)+" "+str(b))
            c=0

def intlist(n):#to convert string to list
    q = n
    ret = []
    while q != 0:
        q, r = divmod(q, 10)
        ret.insert(0, r)
    return ret

def check(a):#to check whether a number is binary or not
    checka=True
    for i in range(2,10):
        if i in a:
            checka=False
            break
    return checka


def add(a,b):#addition of two lists in binary mode
    str1 = ''.join(str(i) for i in a)
    str2=''.join(str(i) for i in b)
    str3=bin(int(str1,2)+int(str2,2))#Binary Addition
    li=[]
    for c in str3:
        li.append(c)
    del li[0]
    del li[0]
    li = list(map(int, li))
    return li

if __name__=='__main__':
        Main()
```

OUTPUT: **Enter multiplicand: 11111111**

```
                Enter multiplier: 11111111
[1,1,1,1,1,1,1,1]
[1,1,1,1,1,1,1,1]
0[0,1,1,1,1,1,1,1] [1,1,1,1,1,1,1,1]
1[1,0,1,1,1,1,1,1] [0,1,1,1,1,1,1,1]
1[1,1,0,1,1,1,1,1] [0,0,1,1,1,1,1,1]
1[1,1,1,0,1,1,1,1] [0,0,0,1,1,1,1,1]
1[1,1,1,1,0,1,1,1] [0,0,0,0,1,1,1,1]
1[1,1,1,1,1,0,1,1] [0,0,0,0,0,1,1,1]
1[1,1,1,1,1,1,0,1] [0,0,0,0,0,0,1,1]
1[1,1,1,1,1,1,1,0] [0,0,0,0,0,0,0,1]   ───►  Final Answer
```