Our instruction will have the Operand code to consist of 8 bits and the memory addresses to consist of 24 bits, equaling a total of 32 bits.

(I asked earlier on piazza if we had to do this part and Qin said just “partially” by designing an instruction format for the simulated machines, so I designed the operand code to have 8 bits and the memory addresses to have 24)

For the assembly code, mine was already in hex code (refer to the stackCode.txt), but I will translate here:

For stack:

```
.text
PUSH X
PUSH X
PUSH X
MULT
PUSH A
MULT
PUSH B
PUSH X
MULT
PUSH C
PUSH
ADD
POP
END

.data
X:3
A:7
B:5
C:4
```

I made the text segment to be from 0 -> 0x00100000, data segment to be to 0x00200000, and stack segment to be 0x00300000.

For accumulator:

```
.text
LOAD X
MUL X
MUL A
STO S1
```

```
LOAD B
MUL X
STO S2
LOAD C
ADD S1
ADD S2
END

.data
X:3
A:7
B:5
C:4
```

Text and Data segments were the same for the accumulator.