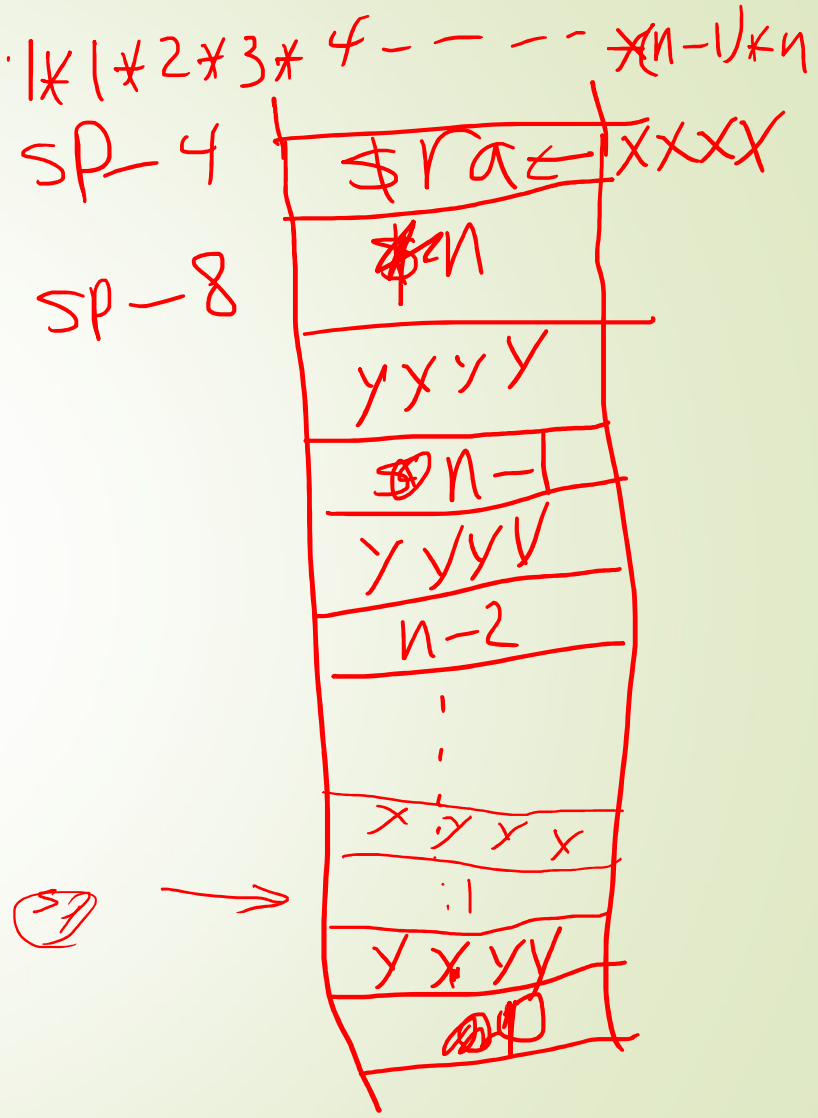
EGC442
Class Notes
2/21/2023

Baback Izadi
Division of Engineering Programs
bai@engr.newpaltz.edu

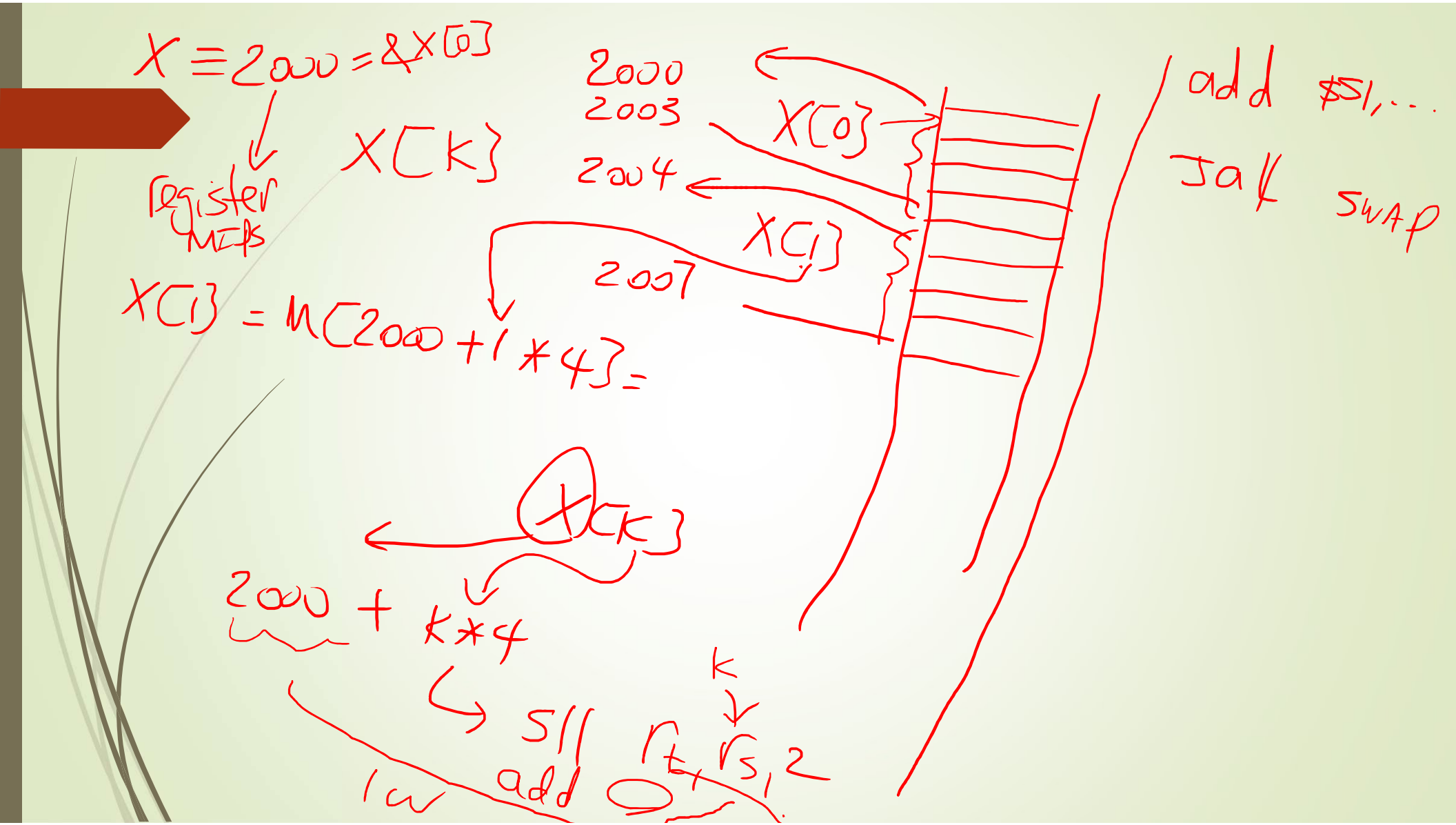
```
int fact (int n)
{
  if (n < 1) return 1;
  else return n * fact(n - 1);
}
```

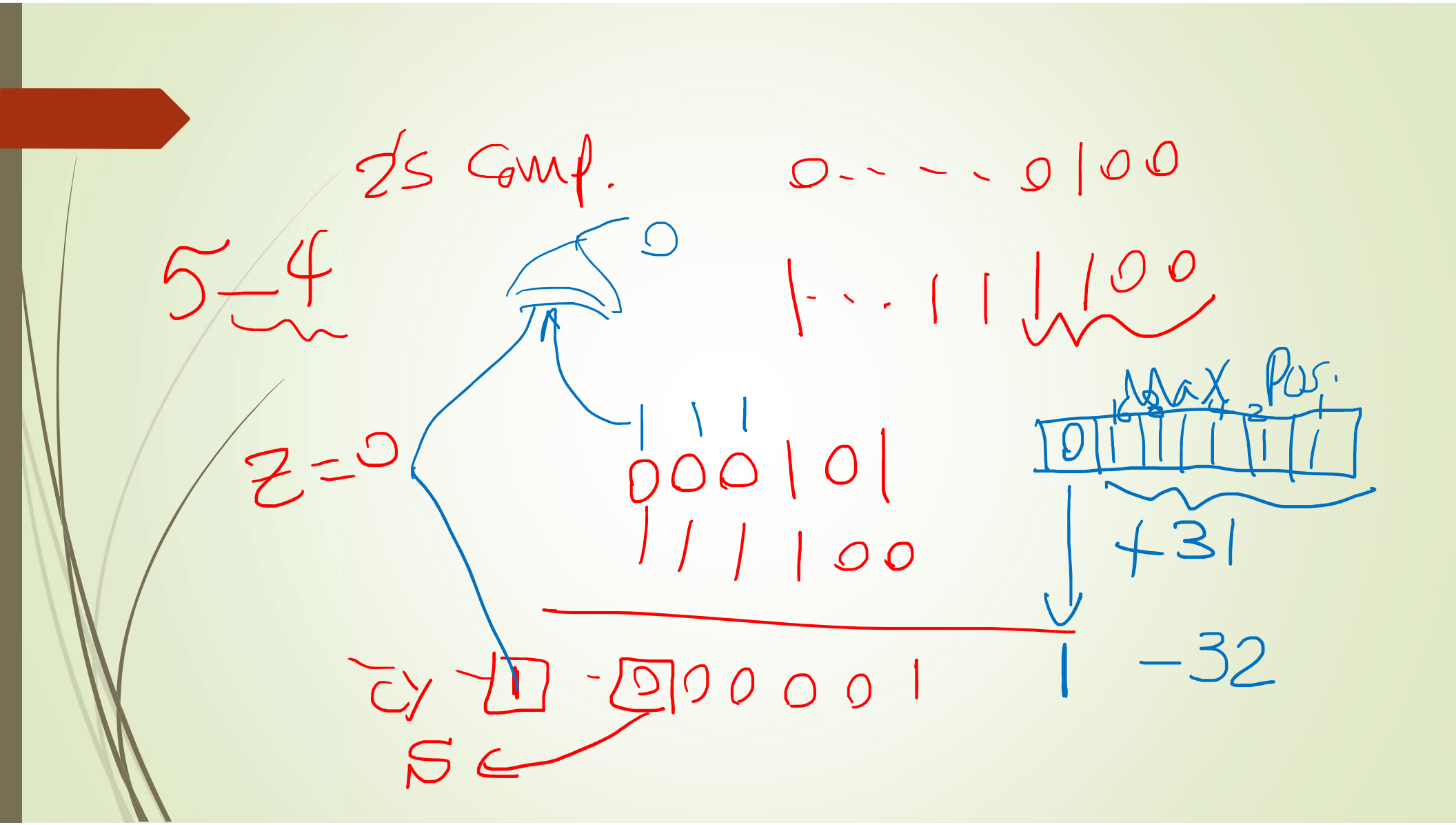
Argument n in $a0
Result in $v0



Non-Leaf Procedure Example

```
fact:
     addi    $sp, $sp, -8     # adjust stack for 2 items
     sw      $ra, 4($sp)      # save return address
     sw      $a0, 0($sp)      # save argument
     slti    $t0, $a0, 1      # test for n < 1
     beq     $t0, $zero, L1
     addi    $v0, $zero, 1    # if so, result is 1
     addi    $sp, $sp, 8      #   pop 2 items from stack
     jr      $ra              #   and return
L1:  addi    $a0, $a0, -1     # else decrement n
     jal     fact             # recursive call
     lw      $a0, 0($sp)      # restore original n
     lw      $ra, 4($sp)      #   and return address
     addi    $sp, $sp, 8      # pop 2 items from stack
     mul     $v0, $a0, $v0    # multiply to get result
     jr      $ra              # and return
```

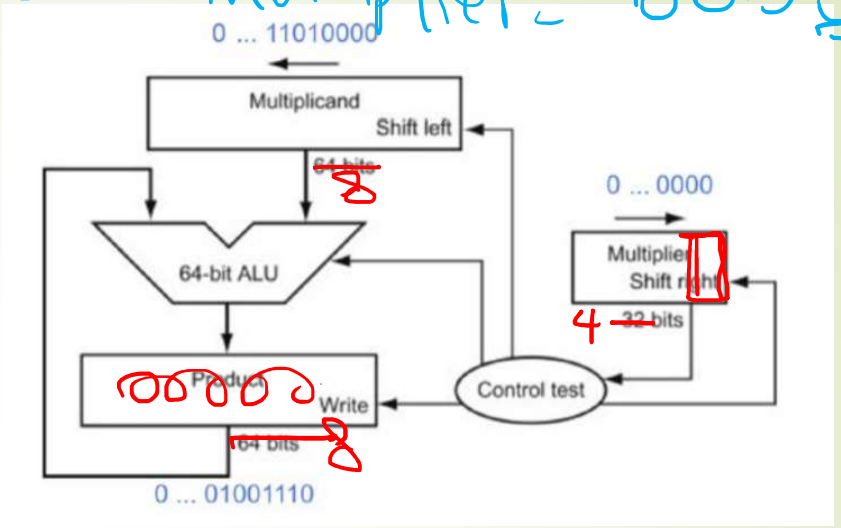

X = 2000 = &X[0]
X[k]
register MIPS
X[1] = M[2000 + 1 * 4] =
2000
2003
2004
2007
X[0]
X[1]
add $s1, ...
jalr
swap
X[k]
2000 + k * 4
k
sll
add
lw


2s Comp.
0----0100
5-4
0
1...1100
111
000101
111100
Z=0
MAX POS.
+31
-32
Cy
1
0
000001
S

5) Multiplicand = 1101000 Multiplier = 0010

8421
1 1 0 1 (Multiplicand)
x 0 1 1 0 (Multiplier)

13
× 6
78

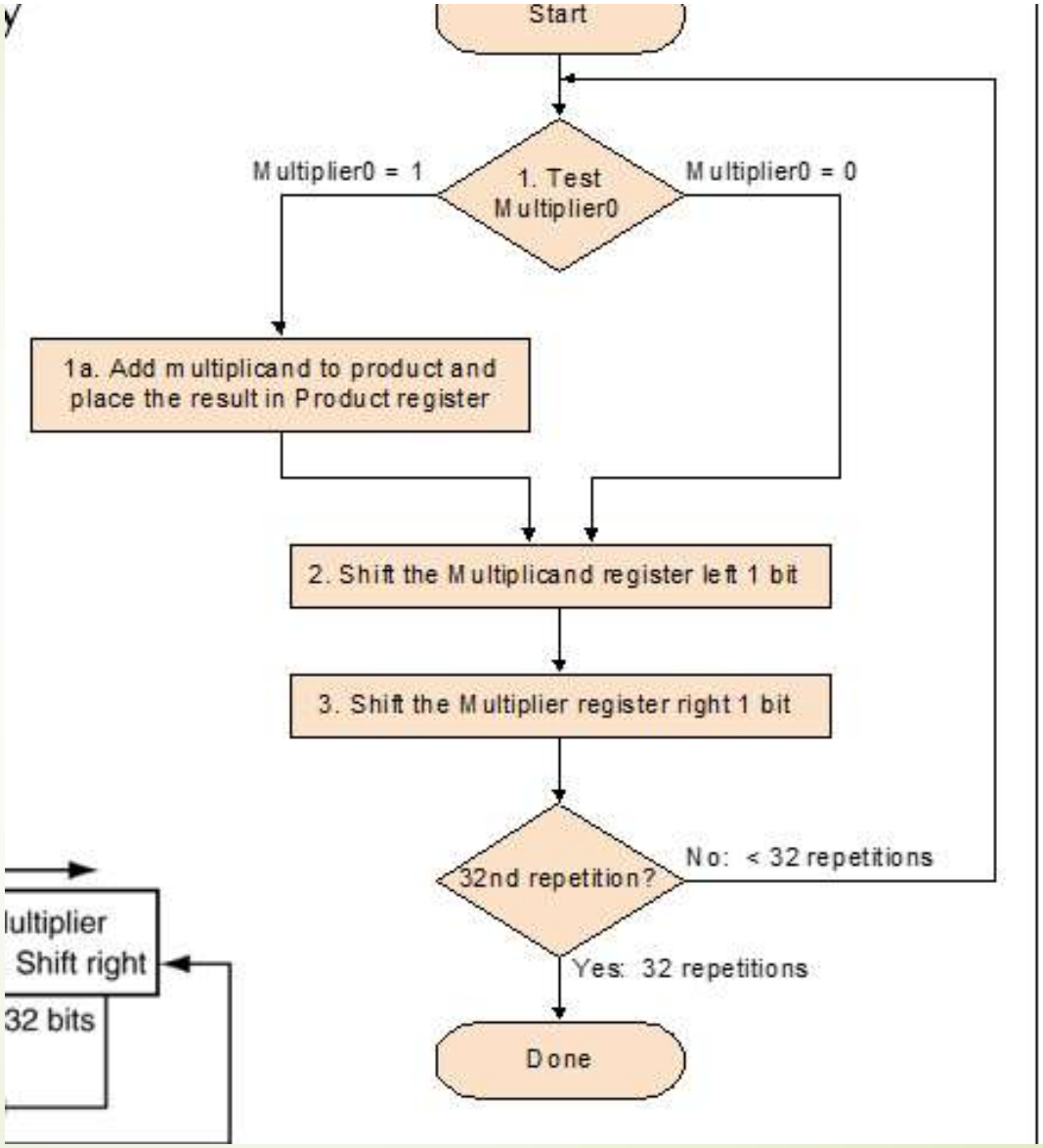


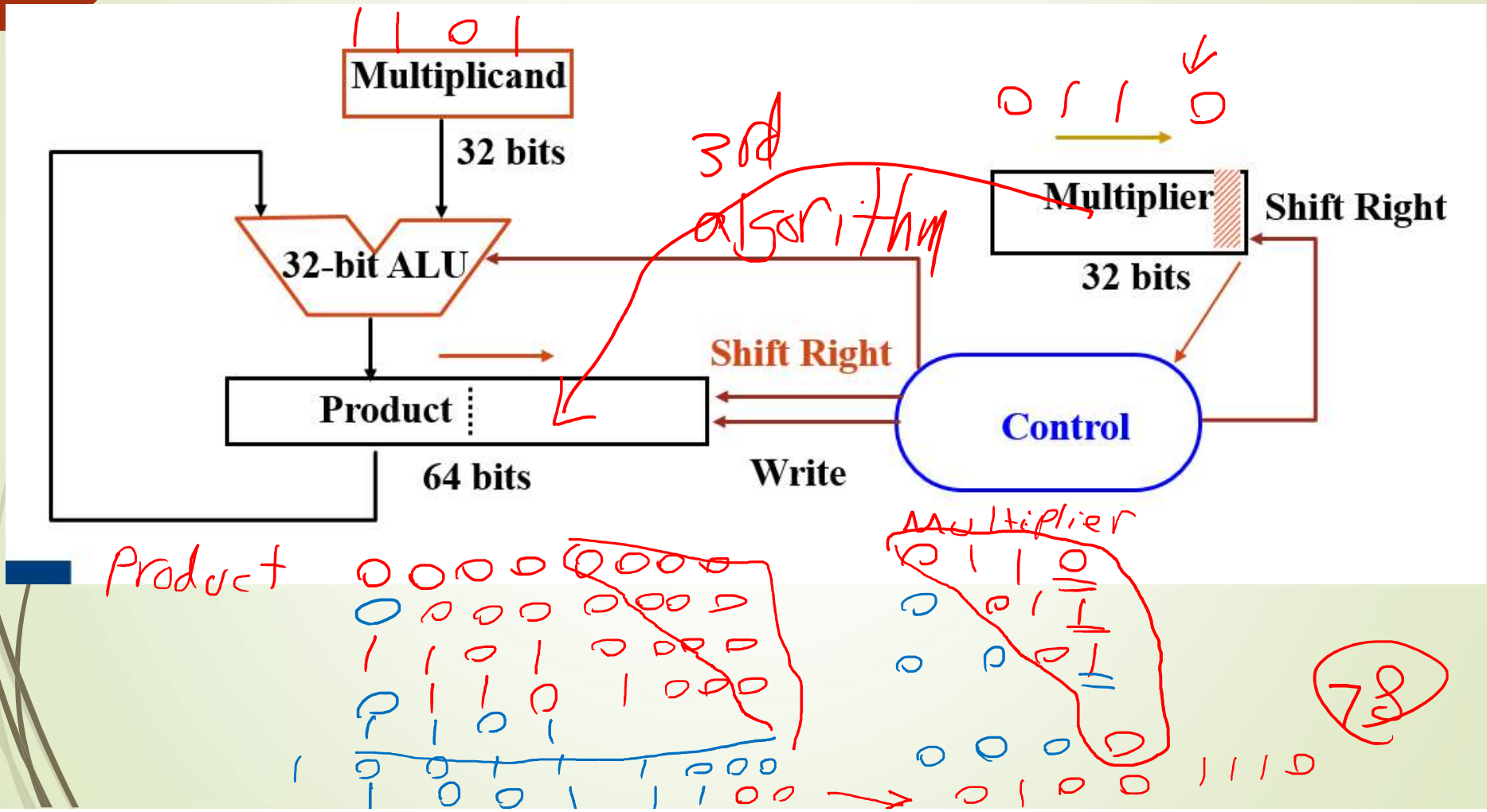
1) Product 0000 0000

2) Multiplier 0110 no add
product 0000 0000

3) Multiplicand = 1101,0 multiplier 0011 — add
product 0001 1010

4) Multiplicand = 1101,00 multiplier = 0001 ← add
01101 0
01001110 → product → 78


Start
1. Test Multiplier0
Multiplier0 = 1
Multiplier0 = 0
1a. Add multiplicand to product and place the result in Product register
2. Shift the Multiplicand register left 1 bit
3. Shift the Multiplier register right 1 bit
32nd repetition?
No: < 32 repetitions
Yes: 32 repetitions
Done
ultiplier
Shift right
32 bits


Multiplicand
32 bits
32-bit ALU
Product
64 bits
Shift Right
Write
Multiplier
32 bits
Shift Right
Control
3rd algorithm
Product
Multiplier

Consider the subtraction of base ten numbers 5 - 4 using 32-bit binary numbers, and achieved by adding 5 with the two's complement of 4:

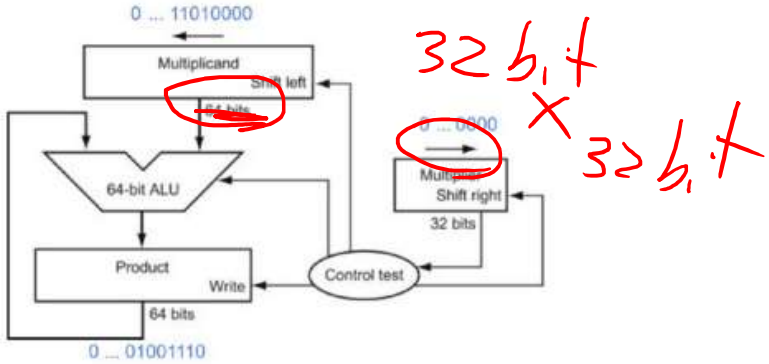
```
 0 0 ... 0 1 0 1     0 0 ... 0 1 0 1
-0 0 ... 0 1 0 0   + 1 1 ... d c b a
-----------------   -----------------
                    s 0 0 ... z y x w
```

3) dcba
- 1011
- 1100

4) zyxw
- 0000
- 0001

5) If a 33rd sum bit, s, existed on the left, what value would that bit get?
- 0
- 1

For the first multiplication algorithm, as depicted the following diagram



15) Each step of the multiplication algorithm shifts the Multiplier register 1 bit to the _____.

- right
- left

16) The Multiplier register is _____-bits wide.

- 32
- 64

17) Each step of the multiplication algorithm shifts the Multiplicand register 1 bit to the _____.

- right
- left

18) The Multiplicand register is _____-bits wide.

- 32
- 64

19) The Product register is _____-bits wide.

- 32
- 64
- 128

Consider $13_{ten} \times 6_{ten}$, or $1101_{two} \times 0110_{two}$. Fill in the missing values.

```
      1 1 0 1  (Multiplicand)
x     0 1 1 0  (Multiplier)
---------------
      ? ? ? ?  (Partial product 1)
    ? ? ? ?    (Partial product 2)
  ? ? ? ?      (Partial product 3)
+ ? ? ? ?      (Partial product 4)
---------------
? ? ? ? ? ? ?  (Product)
```

9) Partial product 1
- 0000
- 1101

10) Partial product 2
- 0000
- 1101

11) Partial product 3
- 0000
- 1101

12) Partial product 4
- 0000
- 1101