INDIAN INSTITUTE OF TECHNOLOGY KHARAGPUR

Mid-Autumn Semester Examination 2022-23

Date of Examination: ___________ Session: (FN/AN) Duration: **2 hrs** Full Marks: **40**

Subject No. **CS31007** Subject: **Computer Organization and Architecture**

Department/Center/School: **Computer Science and Engineering**

Specific charts, graph paper, log book etc., required **NONE**

Special Instructions (if any): **No question/clarification will be entertained during the examination.**

CLEARLY STATE ANY VALID ASSUMPTION YOU MAKE IN ANY QUESTION

1. How many logic gates are required to build the complete Carry-Lookahead Adder shown in **Figure A**? You must briefly show the calculations. Maximum fan-in for any gate is 5. **[4]**
2. Suppose we want to multiply two 6-bit 2's complement numbers where the multiplicand A=101011 and multiplier B = 110001. Show the steps of multiplication using (a) Normal Booth algorithm, (b) using Bit-pair recoding method after Booth encoding. **[5+5=10]**
3. Suppose we want to multiply two 6-bit unsigned binary numbers where multiplicand A=101101 and multiplier B=110101 using 3-2 reducer Carry Save Addition. Explain how the multiplication will be done clearly showing the intermediate step results. **[6]**
4. We want to divide the 6-bit unsigned binary number Q=110101 (dividend) by the 3-bit unsigned binary number M=110 (divisor) using non-restoring division method using a circuit as shown in **Figure B**. Show the various steps and the contents of different registers at each step from start to finish. **[5]**
5. Describe with the help of a diagram how 8M × 16 memory can be realized using a number of 512K × 8 memory chips. You need to show the address, data and relevant control lines. **[4]**
6. With the help of a timing diagram show how a block of 8 consecutive bytes can be read from a Double Data Rate SDRAM (DDR SDRAM). You need to show the clock, Read/Write signal, RAS & CAS, initial Row & Column addresses, and Data. Assume a delay of two clock cycles between RAS and CAS, and a delay of one clock cycle from the first column address to the first data output. **[5]**
7. Consider the recursive procedure *fact* for computing factorial of a number as shown in **Figure C**. The procedure takes one input parameter *n* in $a0 and returns its result through $v0. If the procedure is initially called with a value of *n*=5, show the trace of execution of the procedure till it returns to the calling procedure. You have to clearly identify the line numbers getting executed along with the contents of the registers ($a0, $v0, $sp, $ra, $t0) and the content of the stack at the end of execution of that line. Assume the Program Counter had a value of 80000 and the Stack Pointer had a value of 200000 when the procedure was initially called. The procedure code is loaded in memory from location 40000 onwards as shown in the figure. Clearly state any other reasonable assumption if required to answer the question. **[6]**

P.T.O.

Shamik Sural

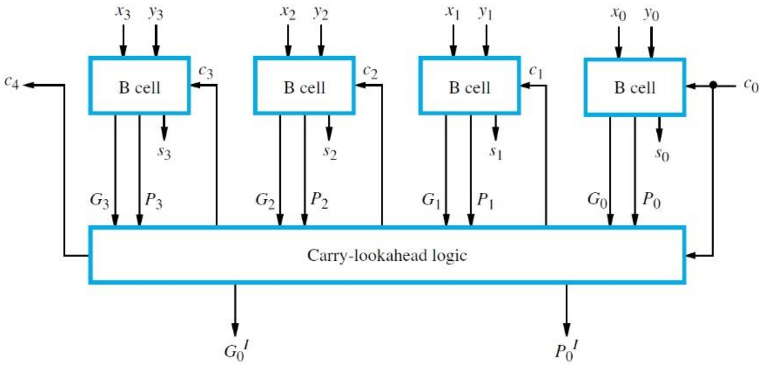


Figure A (Courtesy: Computer Organization and Embedded Systems by Hamacher, Vranesic, Zaky and Manjikian)

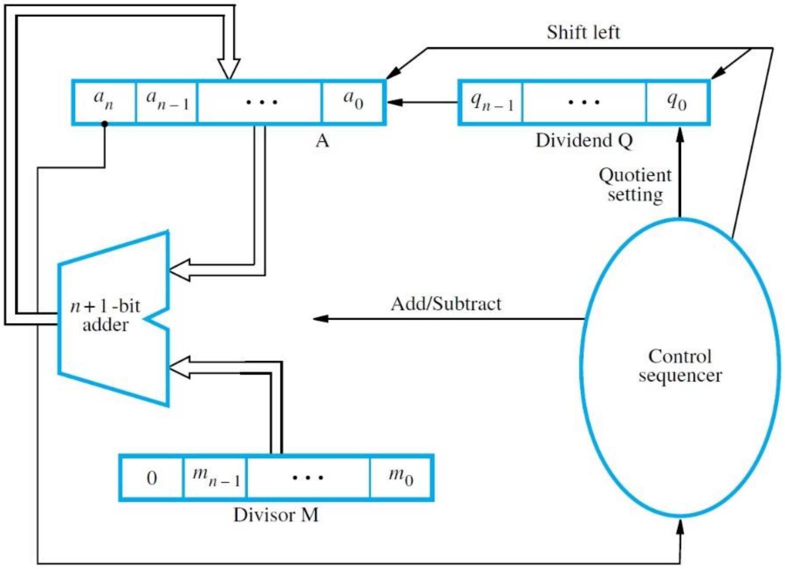


Figure B (Courtesy: Computer Organization and Embedded Systems by Hamacher, Vranesic, Zaky and Manjikian)

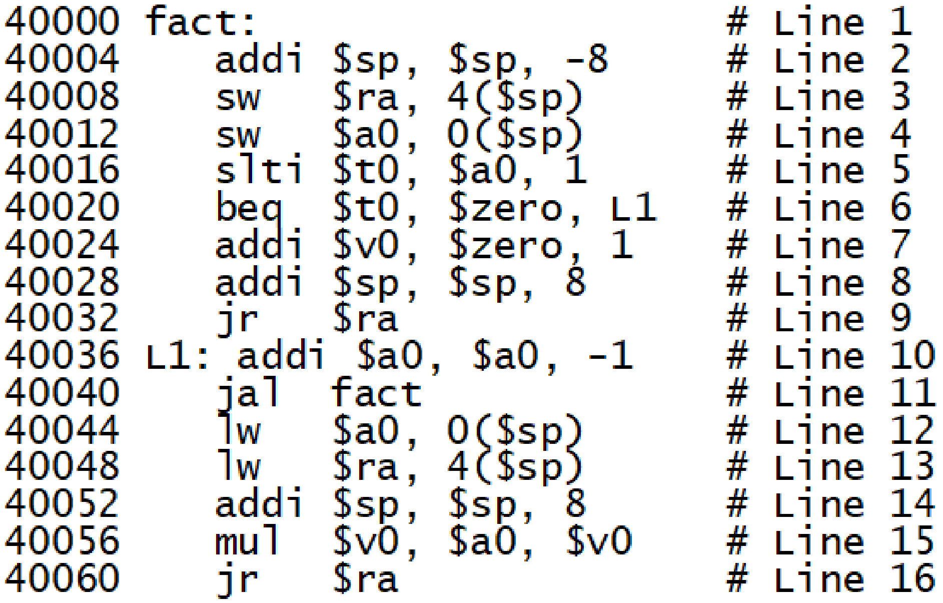
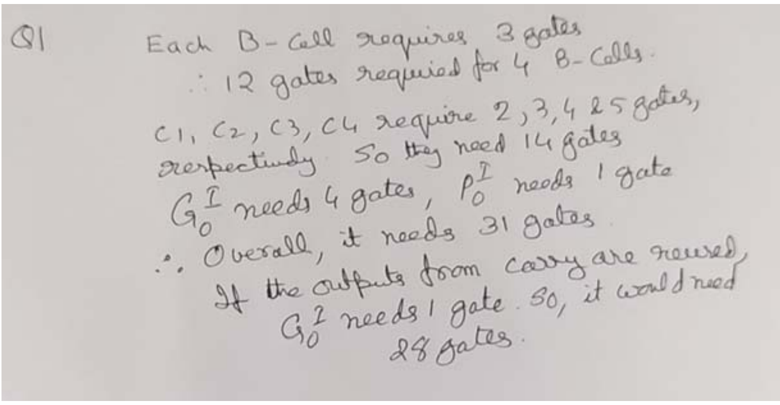

```
40000 fact:                        # Line 1
40004    addi $sp, $sp, -8         # Line 2
40008    sw   $ra, 4($sp)          # Line 3
40012    sw   $a0, 0($sp)          # Line 4
40016    slti $t0, $a0, 1          # Line 5
40020    beq  $t0, $zero, L1       # Line 6
40024    addi $v0, $zero, 1        # Line 7
40028    addi $sp, $sp, 8          # Line 8
40032    jr   $ra                  # Line 9
40036 L1: addi $a0, $a0, -1        # Line 10
40040    jal  fact                 # Line 11
40044    lw   $a0, 0($sp)          # Line 12
40048    lw   $ra, 4($sp)          # Line 13
40052    addi $sp, $sp, 8          # Line 14
40056    mul  $v0, $a0, $v0        # Line 15
40060    jr   $ra                  # Line 16
```

Figure C (Courtesy: Adapted from Computer Organization and Design - The Hardware/Software Interface by Patterson and Hennessy)

Shamik Sural

Q1



Q1 Each B-Cell requires 3 gates

∴ 12 gates requied for 4 B-Cells.

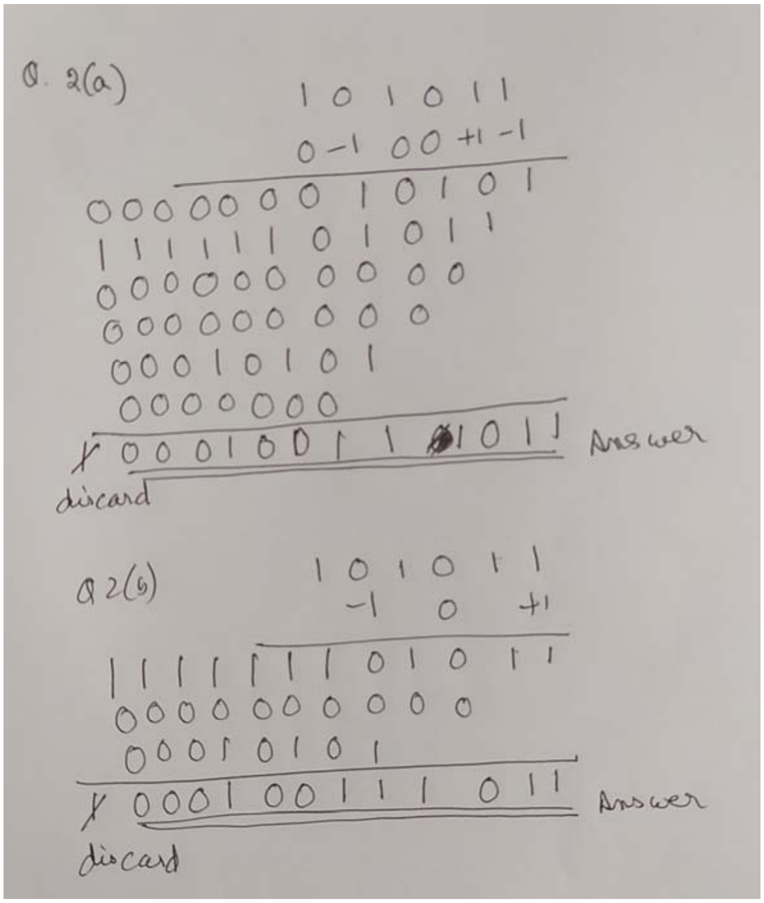
C1, C2, C3, C4 require 2, 3, 4 & 5 gates, respectively. So they need 14 gates

G_0^I needs 4 gates, P_0^I needs 1 gate

∴ Overall, it needs 31 gates.

If the outputs from carry are reused, G_0^I needs 1 gate. So, it would need 28 gates.

Q2



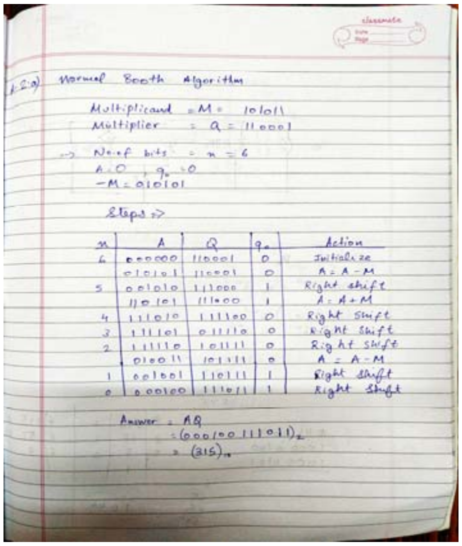
Q. 2(a)

```
                1 0 1 0 1 1
                0 -1 0 0 +1 -1
        ------------------------
  0 0 0 0 0 0 0 1 0 1 0 1
  1 1 1 1 1 1 0 1 0 1 1
  0 0 0 0 0 0 0 0 0 0
  0 0 0 0 0 0 0 0 0
  0 0 0 1 0 1 0 1
  0 0 0 0 0 0 0
  ------------------------
X 0 0 0 1 0 0 1 1 1 0 1 1   Answer
discard
```

Q 2(b)

```
                1 0 1 0 1 1
                  -1  0  +1
        ------------------------
  1 1 1 1 1 1 1 0 1 0 1 1
  0 0 0 0 0 0 0 0 0 0
  0 0 0 1 0 1 0 1
  ------------------------
X 0 0 0 1 0 0 1 1 1 0 1 1   Answer
discard
```

Q2(a) Alternative solution



A-2(a) Normal Booth Algorithm

Multiplicand = M = 101011
Multiplier = Q = 110001

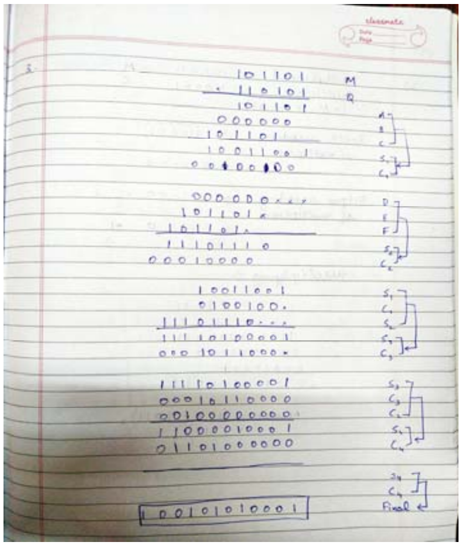
→ No. of bits = n = 6
A = 0 , q_0 = 0
−M = 010101

Steps ⇒

n	A	Q	q_0	Action
6	000000	110001	0	Initialize
	010101	110001	0	A = A − M
5	001010	111000	1	Right shift
	110101	111000	1	A = A + M
4	111010	111100	0	Right Shift
3	111101	011110	0	Right Shift
2	111110	101111	0	Right Shift
	010011	101111	0	A = A − M
1	001001	110111	1	Right Shift
0	000100	111011	1	Right Shift

Answer = AQ
= $(000100111011)_2$
= $(315)_{10}$

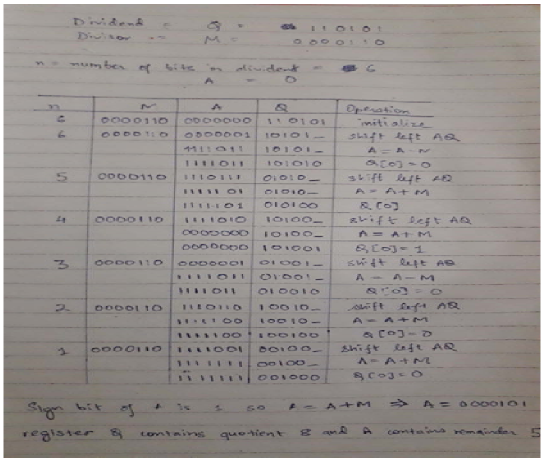
Q3



3.

```
          101101      M
        × 110101      Q
          101101          A
         000000           B
        101101            C
        10011001          S1
        00100100          C1

       000000
      101101
     101101
     11101110             S2
    00010000              C2

        10011001          S1
        0100100           C1
     1110111              S2
     1110100001           S3
     00010110000          C3

     1110100001           S3
     00010110000          C3
     00100000000          C2
     11000010001          S4
     01101000000          C4

                          S4
                          C4
   100101010001           Final
```

Q4



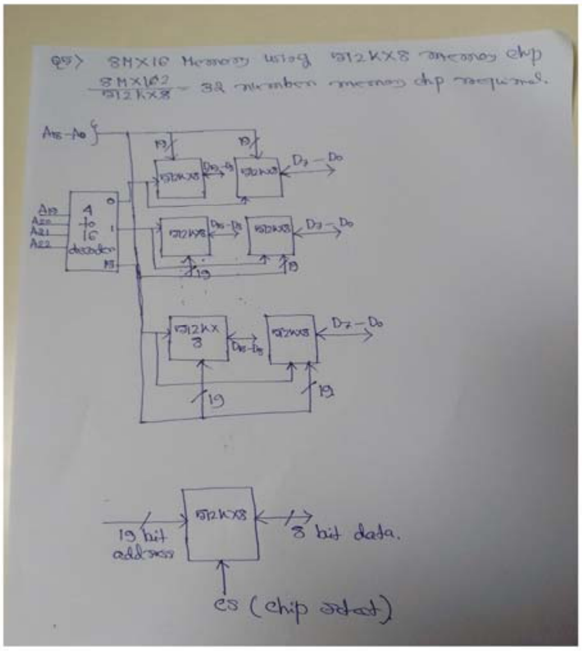
Dividend = Q = 110101
Divisor = M = 0000110

n = number of bits in dividend = 6
A = 0

n	M	A	Q	Operation
6	0000110	0000000	110101	initialize
6	0000110	0000001	10101_	shift left AQ
		1111011	10101_	A = A − M
		1111011	101010	Q[0] = 0
5	0000110	1110111	01010_	shift left AQ
		1111101	01010_	A = A + M
		1111101	010100	Q[0]
4	0000110	1111010	10100_	shift left AQ
		0000000	10100_	A = A + M
		0000000	101001	Q[0] = 1
3	0000110	0000001	01001_	shift left AQ
		1111011	01001_	A = A − M
		1111011	010010	Q[0] = 0
2	0000110	1110110	10010_	shift left AQ
		1111100	10010_	A = A + M
		1111100	100100	Q[0] = 0
1	0000110	1111001	00100_	shift left AQ
		1111111	00100_	A = A + M
		1111111	001000	Q[0] = 0

Sign bit of A is 1 so A = A + M ⇒ A = 0000101
register Q contains quotient 8 and A contains remainder 5

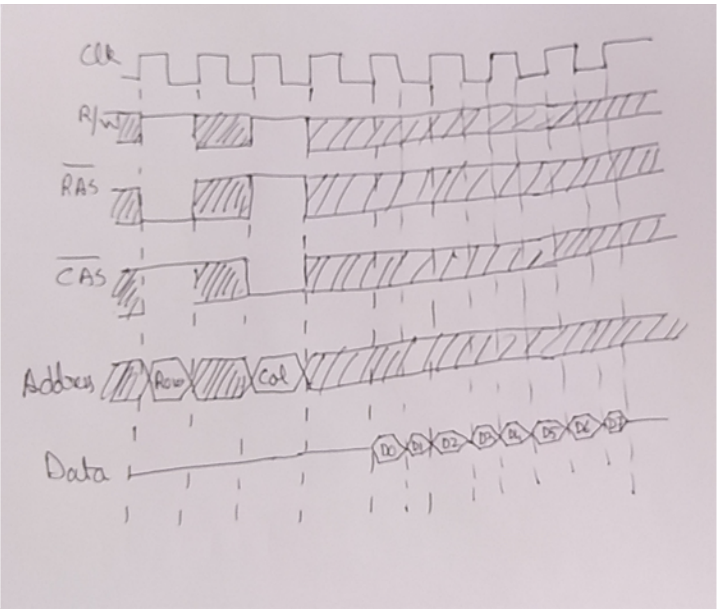
Q5



Q5> 8M×16 Memory using 512K×8 memory chip

$\frac{8M\times16}{512K\times8}$ - 32 number memory chip required.

Q6



Q7

Line no	$a0	$v0	$sp	$ra	$t0	Stack	Remarks
1	5	X	200000	80000	X	Content before the procedure call	
2	5	X	199992	80000	X	Content before the procedure call	
3	5	X	199992	80000	X	199996[80000]	
4	5	X	199992	80000	X	199996[80000], 199992[5]	
5	5	X	199992	80000	0	199996[80000], 199992[5]	
6	5	X	199992	80000	0	199996[80000], 199992[5]	Jump to Line No. 10
10	4	X	199992	80000	0	199996[80000], 199992[5]	
11	4	X	199992	40044	0	199996[80000], 199992[5]	Jump to Line No. 1
1	4	X	199992	40044	0	199996[80000], 199992[5]	
2	4	X	199984	40044	0	199996[80000], 199992[5]	
3	4	X	199984	40044	0	199996[80000], 199992[5], 199988[40044]	
4	4	X	199984	40044	0	199996[80000], 199992[5], 199988[40044], 199984[4]	
5	4	X	199984	40044	0	199996[80000], 199992[5], 199988[40044], 199984[4]	
6	4	X	199984	40044	0	199996[80000], 199992[5], 199988[40044], 199984[4]	Jump to Line No. 10
10	3	X	199984	40044	0	199996[80000], 199992[5], 199988[40044], 199984[4]	
11	3	X	199984	40044	0	199996[80000], 199992[5], 199988[40044], 199984[4]	Jump to Line No. 1
1	3	X	199984	40044	0	199996[80000], 199992[5], 199988[40044], 199984[4]	
2	3	X	199976	40044	0	199996[80000], 199992[5], 199988[40044], 199984[4]	
3	3	X	199976	40044	0	199996[80000], 199992[5], 199988[40044], 199984[4], 199980[40044]	
4	3	X	199976	40044	0	199996[80000], 199992[5], 199988[40044], 199984[4], 199980[40044], 199976[3]	
5	3	X	199976	40044	0	199996[80000], 199992[5], 199988[40044], 199984[4], 199980[40044], 199976[3]	
6	3	X	199976	40044	0	199996[80000], 199992[5], 199988[40044], 199984[4], 199980[40044], 199976[3]	Jump to Line No. 10
10	2	X	199976	40044	0	199996[80000], 199992[5], 199988[40044], 199984[4], 199980[40044], 199976[3]	
11	2	X	199976	40044	0	199996[80000], 199992[5], 199988[40044], 199984[4], 199980[40044], 199976[3]	Jump to Line No. 1
1	2	X	199976	40044	0	199996[80000], 199992[5], 199988[40044], 199984[4], 199980[40044], 199976[3]	
2	2	X	199968	40044	0	199996[80000], 199992[5], 199988[40044], 199984[4], 199980[40044], 199976[3]	

Line no	$a0	$v0	$sp	$ra	$t0	Stack	Remarks
3	2	X	199968	40044	0	199996[80000], 199992[5], 199988[40044], 199984[4], 199980[40044], 199976[3], 199972[40044]	
4	2	X	199968	40044	0	199996[80000], 199992[5], 199988[40044], 199984[4], 199980[40044], 199976[3], 199972[40044], 199968[2]	
5	2	X	199968	40044	0	199996[80000], 199992[5], 199988[40044], 199984[4], 199980[40044], 199976[3], 199972[40044], 199968[2]	
6	2	X	199968	40044	0	199996[80000], 199992[5], 199988[40044], 199984[4], 199980[40044], 199976[3], 199972[40044], 199968[2]	Jump to Line No. 10
10	1	X	199968	40044	0	199996[80000], 199992[5], 199988[40044], 199984[4], 199980[40044], 199976[3], 199972[40044], 199968[2]	
11	1	X	199968	40044	0	199996[80000], 199992[5], 199988[40044], 199984[4], 199980[40044], 199976[3], 199972[40044], 199968[2]	Jump to Line No. 1
1	1	X	199968	40044	0	199996[80000], 199992[5], 199988[40044], 199984[4], 199980[40044], 199976[3], 199972[40044], 199968[2]	
2	1	X	199960	40044	0	199996[80000], 199992[5], 199988[40044], 199984[4], 199980[40044], 199976[3], 199972[40044], 199968[2]	
3	1	X	199960	40044	0	199996[80000], 199992[5], 199988[40044], 199984[4], 199980[40044], 199976[3], 199972[40044], 199968[2], 199964[40044]	
4	1	X	199960	40044	0	199996[80000], 199992[5], 199988[40044], 199984[4], 199980[40044], 199976[3], 199972[40044], 199968[2], 199964[40044], 199960[1]	
5	1	X	199960	40044	0	199996[80000], 199992[5], 199988[40044], 199984[4], 199980[40044], 199976[3], 199972[40044], 199968[2], 199964[40044], 199960[1]	
6	1	X	199960	40044	0	199996[80000], 199992[5], 199988[40044], 199984[4], 199980[40044], 199976[3], 199972[40044], 199968[2], 199964[40044], 199960[1]	Jump to Line No. 10
10	0	X	199960	40044	0	199996[80000], 199992[5], 199988[40044], 199984[4], 199980[40044], 199976[3], 199972[40044], 199968[2], 199964[40044], 199960[1]	
11	0	X	199960	40044	0	199996[80000], 199992[5], 199988[40044], 199984[4], 199980[40044], 199976[3], 199972[40044], 199968[2], 199964[40044], 199960[1]	Jump to Line No. 1
1	0	X	199960	40044	0	199996[80000], 199992[5], 199988[40044], 199984[4], 199980[40044], 199976[3], 199972[40044], 199968[2], 199964[40044], 199960[1]	
2	0	X	199952	40044	0	199996[80000], 199992[5], 199988[40044], 199984[4], 199980[40044], 199976[3], 199972[40044], 199968[2], 199964[40044], 199960[1]	
3	0	X	199952	40044	0	199996[80000], 199992[5], 199988[40044], 199984[4], 199980[40044], 199976[3], 199972[40044], 199968[2], 199964[40044], 199960[1], 199956[40044]	

Line no	**$a0**	**$v0**	**$sp**	**$ra**	**$t0**	**Stack**	**Remarks**
4	0	X	199952	40044	0	199996[80000], 199992[5], 199988[40044], 199984[4], 199980[40044], 199976[3], 199972[40044], 199968[2], 199964[40044], 199960[1], 199956[40044], 199952[0]	
5	0	X	199952	40044	1	199996[80000], 199992[5], 199988[40044], 199984[4], 199980[40044], 199976[3], 199972[40044], 199968[2], 199964[40044], 199960[1], 199956[40044], 199952[0]	
6	0	X	199952	40044	1	199996[80000], 199992[5], 199988[40044], 199984[4], 199980[40044], 199976[3], 199972[40044], 199968[2], 199964[40044], 199960[1], 199956[40044], 199952[0]	No Jump
7	0	1	199952	40044	1	199996[80000], 199992[5], 199988[40044], 199984[4], 199980[40044], 199976[3], 199972[40044], 199968[2], 199964[40044], 199960[1], 199956[40044], 199952[0]	
8	0	1	199960	40044	1	199996[80000], 199992[5], 199988[40044], 199984[4], 199980[40044], 199976[3], 199972[40044], 199968[2], 199964[40044], 199960[1]	
9	0	1	199960	40044	1	199996[80000], 199992[5], 199988[40044], 199984[4], 199980[40044], 199976[3], 199972[40044], 199968[2], 199964[40044], 199960[1]	Jump to Line 12 (Location 40044)
12	1	1	199960	40044	1	199996[80000], 199992[5], 199988[40044], 199984[4], 199980[40044], 199976[3], 199972[40044], 199968[2], 199964[40044], 199960[1]	
13	1	1	199960	40044	1	199996[80000], 199992[5], 199988[40044], 199984[4], 199980[40044], 199976[3], 199972[40044], 199968[2], 199964[40044], 199960[1]	
14	1	1	199968	40044	1	199996[80000], 199992[5], 199988[40044], 199984[4], 199980[40044], 199976[3], 199972[40044], 199968[2]	
15	1	1	199968	40044	1	199996[80000], 199992[5], 199988[40044], 199984[4], 199980[40044], 199976[3], 199972[40044], 199968[2]	
16	1	1	199968	40044	1	199996[80000], 199992[5], 199988[40044], 199984[4], 199980[40044], 199976[3], 199972[40044], 199968[2]	Jump to Line 12 (Location 40044)
12	2	1	199968	40044	1	199996[80000], 199992[5], 199988[40044], 199984[4], 199980[40044], 199976[3], 199972[40044], 199968[2]	
13	2	1	199968	40044	1	199996[80000], 199992[5], 199988[40044], 199984[4], 199980[40044], 199976[3], 199972[40044], 199968[2]	
14	2	1	199976	40044	1	199996[80000], 199992[5], 199988[40044], 199984[4], 199980[40044], 199976[3]	
15	2	2	199976	40044	1	199996[80000], 199992[5], 199988[40044], 199984[4], 199980[40044], 199976[3]	
16	2	2	199976	40044	1	199996[80000], 199992[5], 199988[40044], 199984[4], 199980[40044], 199976[3]	Jump to Line 12 (Location 40044)

Line no	**$a0**	**$v0**	**$sp**	**$ra**	**$t0**	**Stack**	**Remarks**
12	3	2	199976	40044	1	199996[80000], 199992[5], 199988[40044], 199984[4], 199980[40044], 199976[3]	
13	3	2	199976	40044	1	199996[80000], 199992[5], 199988[40044], 199984[4], 199980[40044], 199976[3]	
14	3	2	199984	40044	1	199996[80000], 199992[5], 199988[40044], 199984[4]	
15	3	6	199984	40044	1	199996[80000], 199992[5], 199988[40044], 199984[4]	
16	3	6	199984	40044	1	199996[80000], 199992[5], 199988[40044], 199984[4]	Jump to Line 12 (Location 40044)
12	4	6	199984	40044	1	199996[80000], 199992[5], 199988[40044], 199984[4]	
13	4	6	199984	40044	1	199996[80000], 199992[5], 199988[40044], 199984[4]	
14	4	6	199992	40044	1	199996[80000], 199992[5]	
15	4	24	199992	40044	1	199996[80000], 199992[5]	
16	4	24	199992	40044	1	199996[80000], 199992[5]	Jump to Line 12 (Location 40044)
12	5	24	199992	40044	1	199996[80000], 199992[5]	
13	5	24	199992	80000	1	199996[80000], 199992[5]	
14	5	24	200000	80000	1	Content before the procedure call	
15	5	120	200000	80000	1	Content before the procedure call	
16	5	120	200000	80000	1	Content before the procedure call	Jump to Calling Procedure (PC = $ra = 80000)