Q1: Apply a suitable integer binary arithmetic algorithm to solve A x B for A = 15 and B = 7 where each number is represented using 6 bits

ans:

A x B   A = 15   B = 7   15 x 7

6 001111   6 000111

Iteration	steps	multiplier(B)	multiplicand(A)	product(P)
0	initial value	000111	000000 001111	000000 000000
1	P = P + A	000111	000000 001111	000000 001111
	Shift left A	000111	000000 011110	000 000 001111
	Shift right B	000011	000000 011110	000 000 001111
2	P = P + A	000011	000000 011110	000000 101101
	SLA	000011	000000 111100	000000 101101
	SRB	000001	000000 111100	000000 101101
3	P = P + A	000001	000000 111100	000001 101001
	SLA	000001	000001 111000	000001 101001
	SRB	000000	000001 111000	000001 101001
4	SLA	000000	000011 110000	000001 101001
	SRB	000000		
5	SLA	000000	000111 100000	000001 101001
	SRB	000000	000111 100000	000001 101001
6	SLA	000000	001111 000000	000001 101001
	SRB	000000	001111 000000	000001 101001

1: 000000 000000
000000 001111
000000 001111

2: 000000 001111
0 00000 011110
000000 101101

3: 000000 101101
000000 111100
000001 101001

A×B = 000001 101001

A×-B → so we take 2s complement

$\boxed{A\times -B = 111110\ 010111}$ answer

Q2. For the given C code produce the equivalent MIPS assembly code

```
              $s0  $t1        $t2
while ( s*x[i] != z[j] )
   $t0     x[i+1] = y[j] + z[j];
           i += 1;    $s1    $s2
           j += 1;
```

ans:

```
Loop:   sll  $t3, $t1, 2        # offset i
        sll  $t4, $t2, 2        # offset j
        addi $t5, $t1, 1        # i+1
        sll  $t6, $t5, 2        # offset i+1

        add  $t7, $t3, $s0      # effective address x[i]
        add  $t8, $t4, $s2      # effective address z[j]
        lw   $s3, 0($t7)        # value of x[i]
        lw   $s4, 0($t8)        # value of z[j]
        mul  $s5, $t0, $s3      # s * x[i]

        Beq  $s5, $s4, Exit     # if s*x[i] = z[j] then exit

        add  $t9, $s1, $t4      # effective address y[j]
        add  $s6, $t6, $s0      # effective address x[i+1]

        lw   $t3, 0($t9)        # value of y[j], overwrite value of t3
        add  $t4, $t3, $s4      # y[j] + z[j] , overwrite value of t4
        sw   $s6, 0($t4)        # x[i+1]=
```

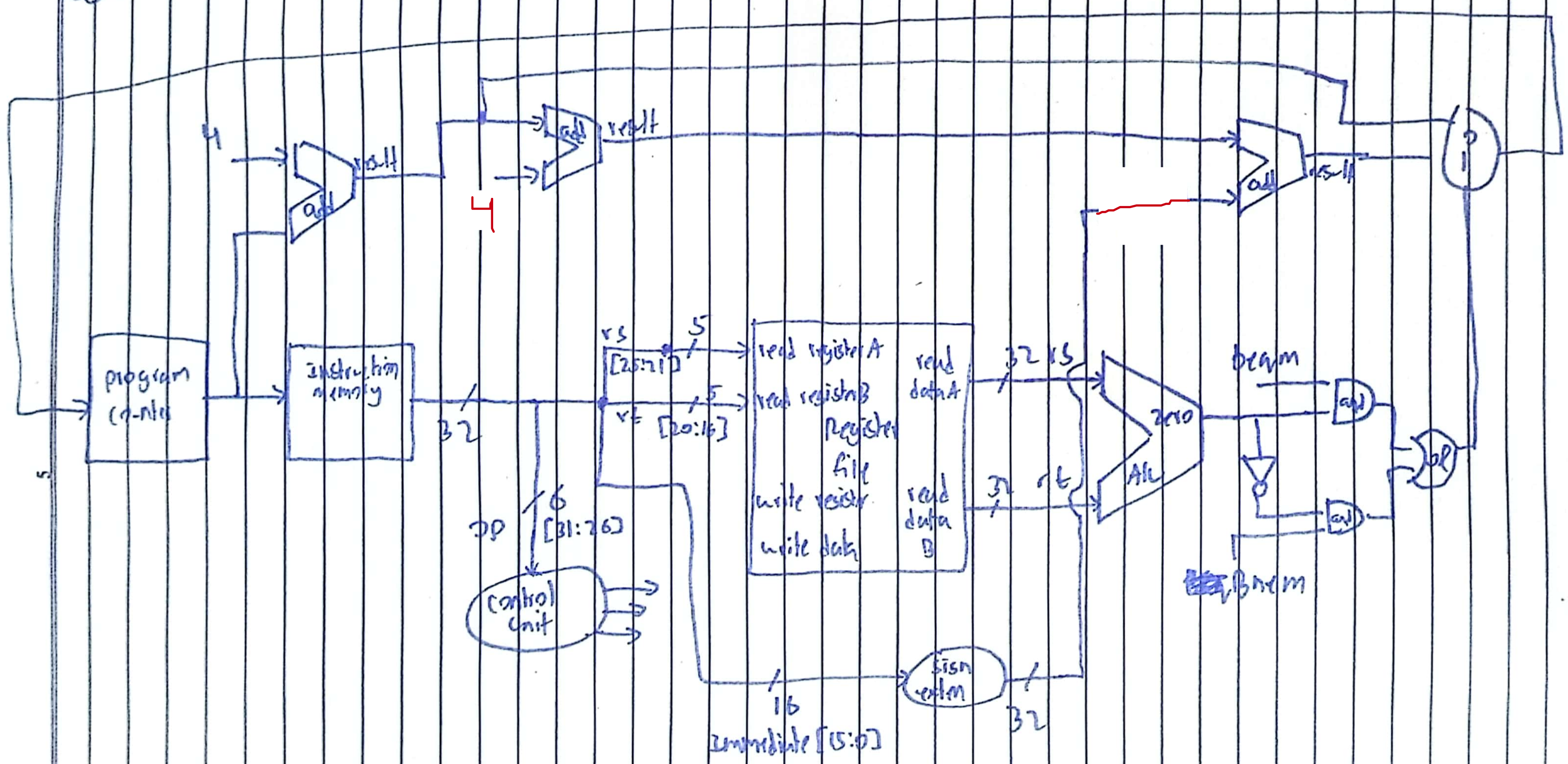
```
addi $t1, $t1, 1        # i=i+1
addi $t2, $t2, 1        # j=j+1

j loop                  # jump to loop

Exit:
```

001397

Q3.



program counter
instruction memory
4
add
result
32
rs [25:21]
rt [20:16]
5
op [31:26]
6
control unit
read register A
read register B
Register file
write register
write data
read data A
read data B
32 rs
32 rt
ALU
zero
16
immediate [15:0]
sign extend
32
beqm
bnem
and
or