MPCA LAB 6

Name - Nandan N

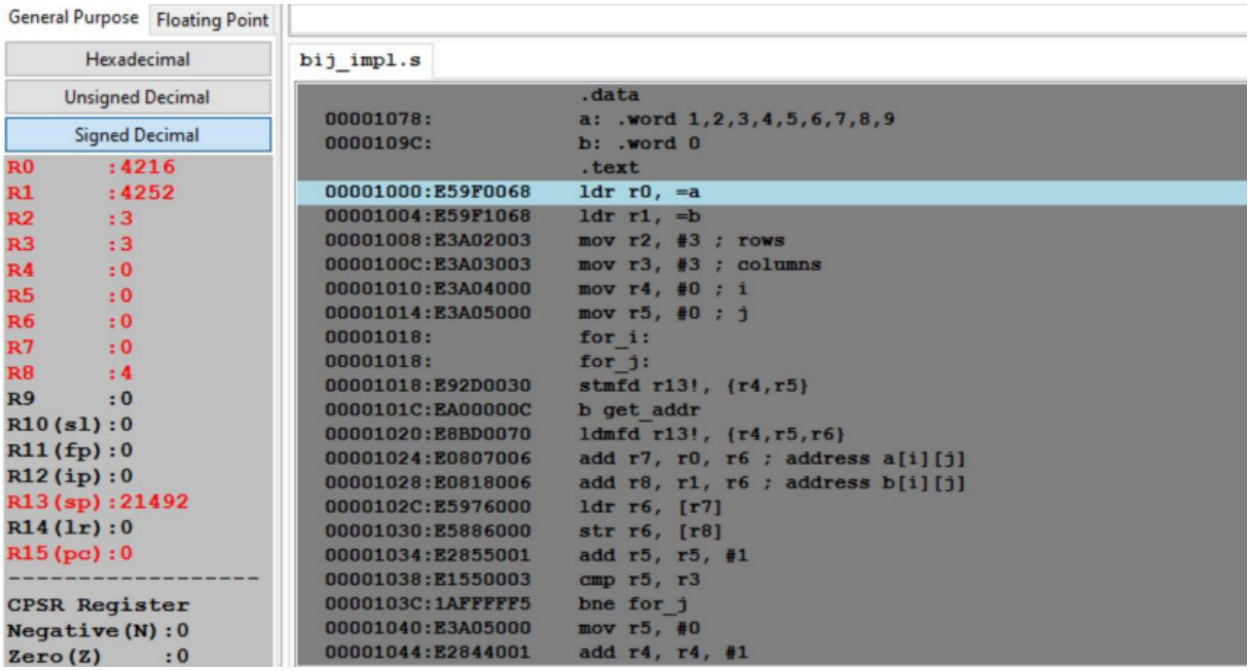
SRN - PES1UG21CS361

a.Write an ALP to implement B=a[i][j]

CODE

```
.data
a: .word 1,2,3,4,5,6,7,8,9
b: .word 0
.text
ldr r0, =a
ldr r1, =b
mov r2, #3 ; rows
mov r3, #3 ; columns
mov r4, #0 ; i
mov r5, #0 ; j
for_i:
for_j:
stmfd r13!, {r4,r5}
b get_addr
ldmfd r13!, {r4,r5,r6}
add r7, r0, r6 ; address a[i][j]
add r8, r1, r6 ; address b[i][j]
ldr r6, [r7]
str r6, [r8]
add r5, r5, #1
cmp r5, r3
bne for_j
mov r5, #0
add r4, r4, #1
cmp r4, r2
beq exit
b for_i
get_addr:
ldmfd r13!, {r4, r5}
mla r7, r3, r4, r5
mov r8, #4
mul r6, r7, r8
stmfd r13!, {r4, r5, r6}
bx lr
exit: SWI 0x011
.end
```

OUTPUT

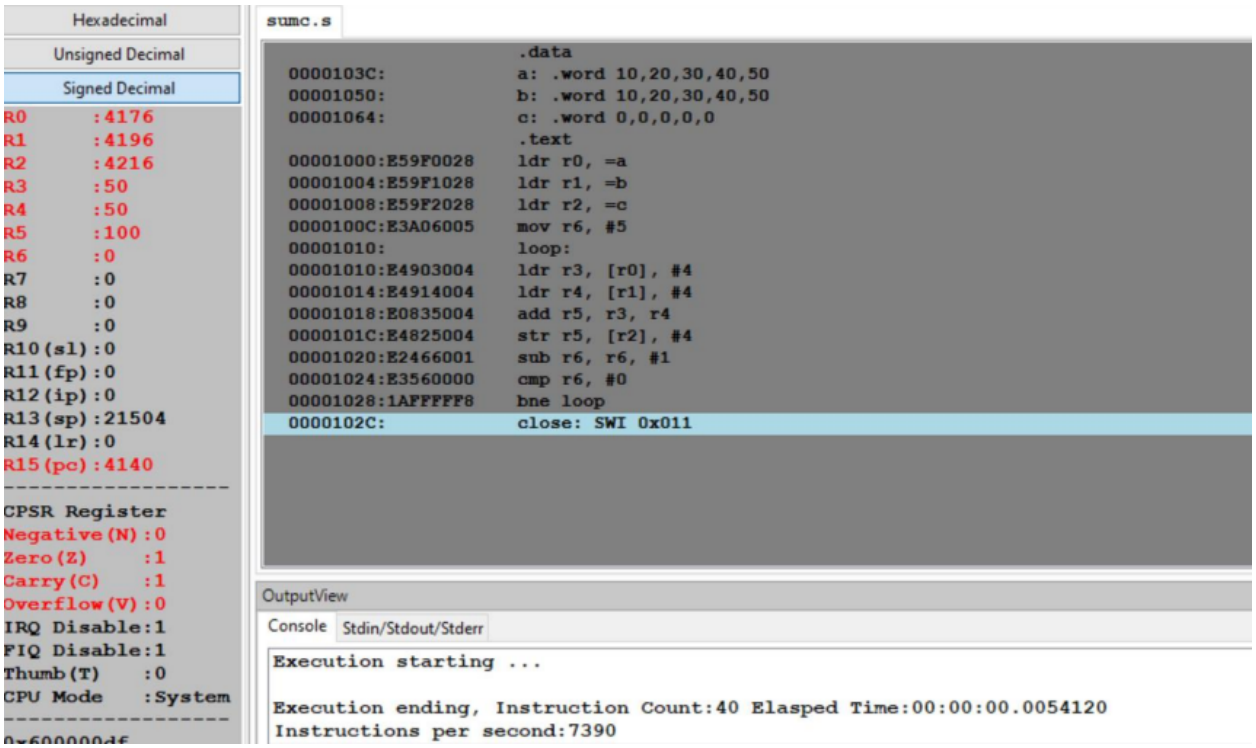


b. Write an ALP to implement C[k]=a[i]+b[j]

CODE

```
.data
a: .word 10,20,30,40,50
b: .word 10,20,30,40,50
c: .word 0,0,0,0,0
.text
ldr r0, =a
ldr r1, =b
ldr r2, =c
mov r6, #5
loop:
ldr r3, [r0], #4
ldr r4, [r1], #4
add r5, r3, r4
str r5, [r2], #4
sub r6, r6, #1
cmp r6, #0
bne loop
close: SWI 0x011
```

OUTPUT

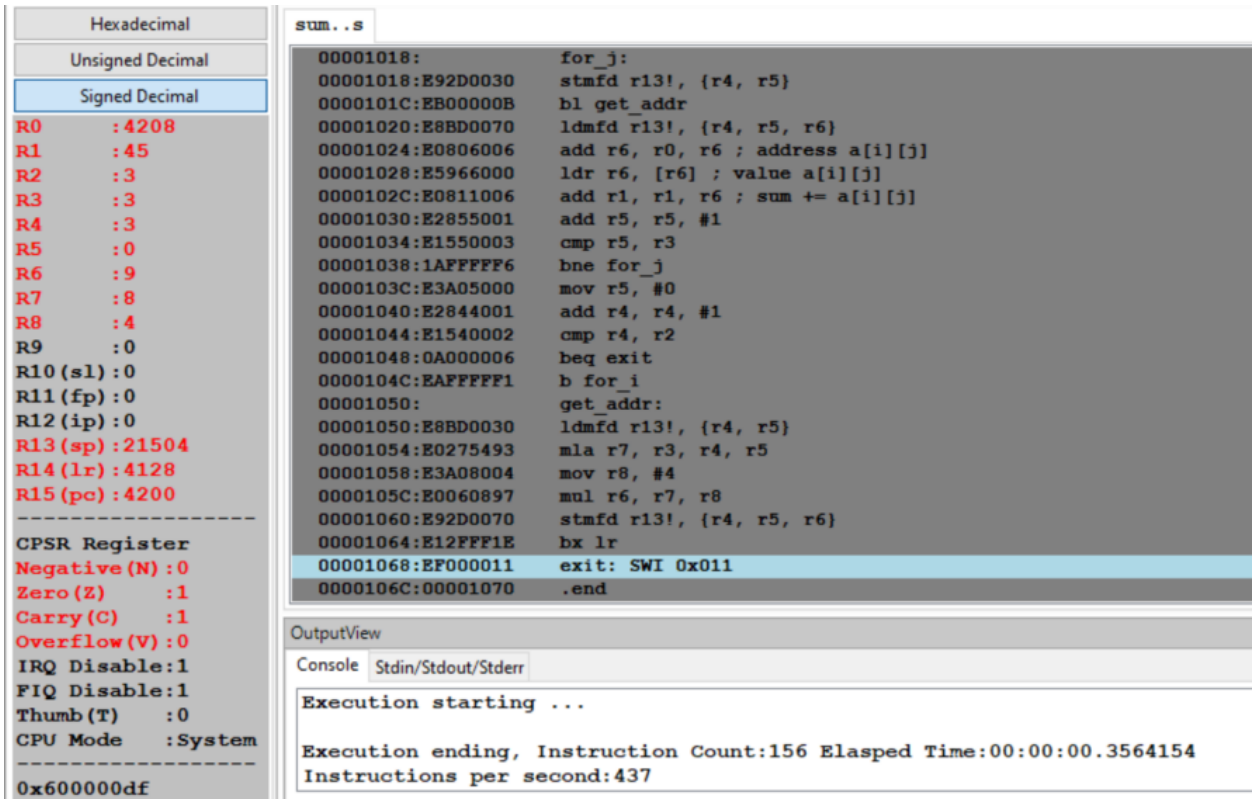


c.Write an ALP to implement Sum[i]+=a[i][j]

CODE

```
.data
a: .word 1,2,3,4,5,6,7,8,9
.text
ldr r0, =a
mov r1, #0 ; =Sum
mov r2, #3 ; =rows
mov r3, #3 ; =columns
mov r4, #0 ; =i
mov r5, #0 ; =j
for_i:
for_j:
stmfd r13!, {r4, r5}
bl get_addr
ldmfd r13!, {r4, r5, r6}
add r6, r0, r6 ; address a[i][j]
ldr r6, [r6] ; value a[i][j]
add r1, r1, r6 ; sum += a[i][j]
add r5, r5, #1
cmp r5, r3
bne for_j
mov r5, #0
add r4, r4, #1
cmp r4, r2
beq exit
b for_i
get_addr:
ldmfd r13!, {r4, r5}
mla r7, r3, r4, r5
mov r8, #4
mul r6, r7, r8
stmfd r13!, {r4, r5, r6}
bx lr
exit: SWI 0x011
```

OUTPUT

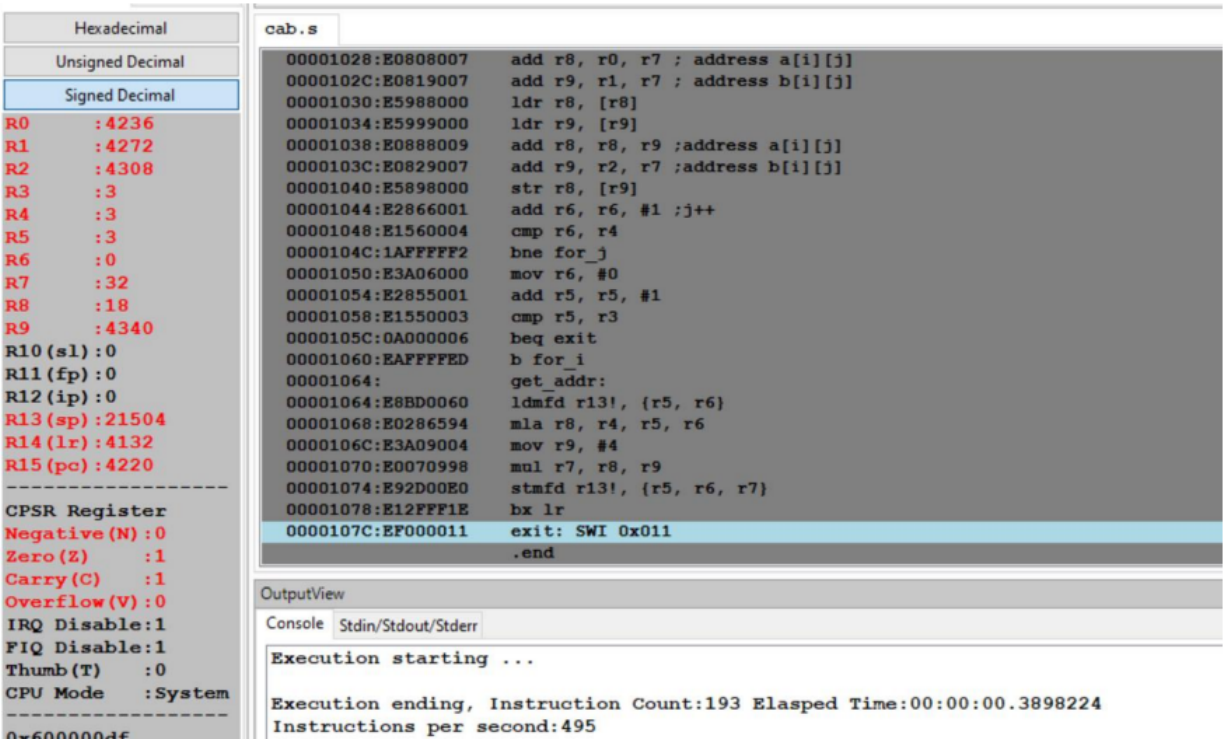

Hexadecimal
Unsigned Decimal
Signed Decimal
R0 :4208
R1 :45
R2 :3
R3 :3
R4 :3
R5 :0
R6 :9
R7 :8
R8 :4
R9 :0
R10(sl):0
R11(fp):0
R12(ip):0
R13(sp):21504
R14(lr):4128
R15(pc):4200
CPSR Register
Negative(N):0
Zero(Z) :1
Carry(C) :1
Overflow(V):0
IRQ Disable:1
FIQ Disable:1
Thumb(T) :0
CPU Mode :System
0x600000df
sum..s
00001018: for_j:
00001018:E92D0030 stmfd r13!, {r4, r5}
0000101C:EB00000B bl get_addr
00001020:E8BD0070 ldmfd r13!, {r4, r5, r6}
00001024:E0806006 add r6, r0, r6 ; address a[i][j]
00001028:E5966000 ldr r6, [r6] ; value a[i][j]
0000102C:E0811006 add r1, r1, r6 ; sum += a[i][j]
00001030:E2855001 add r5, r5, #1
00001034:E1550003 cmp r5, r3
00001038:1AFFFFF6 bne for_j
0000103C:E3A05000 mov r5, #0
00001040:E2844001 add r4, r4, #1
00001044:E1540002 cmp r4, r2
00001048:0A000006 beq exit
0000104C:EAFFFFF1 b for_i
00001050: get_addr:
00001050:E8BD0030 ldmfd r13!, {r4, r5}
00001054:E0275493 mla r7, r3, r4, r5
00001058:E3A08004 mov r8, #4
0000105C:E0060897 mul r6, r7, r8
00001060:E92D0070 stmfd r13!, {r4, r5, r6}
00001064:E12FFF1E bx lr
00001068:EF000011 exit: SWI 0x011
0000106C:00001070 .end
OutputView
Console
Stdin/Stdout/Stderr
Execution starting ...
Execution ending, Instruction Count:156 Elasped Time:00:00:00.3564154
Instructions per second:437

d . Write an ALP to implement C[i][j]=a[i][j]+b[i][j]

CODE

```
.data
a: .word 1,2,3,4,5,6,7,8,9
b: .word 1,2,3,4,5,6,7,8,9
c: .word 0
.text
ldr r0, =a
ldr r1, =b
ldr r2, =c
mov r3, #3 ; rows
mov r4, #3 ; columns
mov r5, #0 ; i
mov r6, #0 ; j
for_i:
for_j:
stmfd r13!, {r5,r6}
bl get_addr
ldmfd r13!, {r5,r6,r7}
add r8, r0, r7 ; address a[i][j]
add r9, r1, r7 ; address b[i][j]
ldr r8, [r8]
ldr r9, [r9]
add r8, r8, r9 ;address a[i][j]
add r9, r2, r7 ;address b[i][j]
str r8, [r9]
add r6, r6, #1 ;j++
cmp r6, r4
bne for_j
mov r6, #0
add r5, r5, #1
cmp r5, r3
beq exit
b for_i
get_addr:
ldmfd r13!, {r5, r6}
mla r8, r4, r5, r6
mov r9, #4
mul r7, r8, r9
stmfd r13!, {r5, r6, r7}
bx lr
exit: SWI 0x011
.end
```

OUTPUT


Hexadecimal
Unsigned Decimal
Signed Decimal
R0 :4236
R1 :4272
R2 :4308
R3 :3
R4 :3
R5 :3
R6 :0
R7 :32
R8 :18
R9 :4340
R10(sl):0
R11(fp):0
R12(ip):0
R13(sp):21504
R14(lr):4132
R15(pc):4220
CPSR Register
Negative(N):0
Zero(Z) :1
Carry(C) :1
Overflow(V):0
IRQ Disable:1
FIQ Disable:1
Thumb(T) :0
CPU Mode :System
cab.s
00001028:E0808007 add r8, r0, r7 ; address a[i][j]
0000102C:E0819007 add r9, r1, r7 ; address b[i][j]
00001030:E5988000 ldr r8, [r8]
00001034:E5999000 ldr r9, [r9]
00001038:E0888009 add r8, r8, r9 ;address a[i][j]
0000103C:E0829007 add r9, r2, r7 ;address b[i][j]
00001040:E5898000 str r8, [r9]
00001044:E2866001 add r6, r6, #1 ;j++
00001048:E1560004 cmp r6, r4
0000104C:1AFFFFF2 bne for_j
00001050:E3A06000 mov r6, #0
00001054:E2855001 add r5, r5, #1
00001058:E1550003 cmp r5, r3
0000105C:0A000006 beq exit
00001060:EAFFFFED b for_i
00001064: get_addr:
00001064:E8BD0060 ldmfd r13!, {r5, r6}
00001068:E0286594 mla r8, r4, r5, r6
0000106C:E3A09004 mov r9, #4
00001070:E0070998 mul r7, r8, r9
00001074:E92D00E0 stmfd r13!, {r5, r6, r7}
00001078:E12FFF1E bx lr
0000107C:EF000011 exit: SWI 0x011
.end
OutputView
Console
Stdin/Stdout/Stderr
Execution starting ...
Execution ending, Instruction Count:193 Elasped Time:00:00:00.3898224
Instructions per second:495