

NAME- GHANSHYAM THAKKAR

ENROLLMENT NO.- 200280111051

BATCH- A2, 5TH SEM

EXERCISE 3

QUESTION 1

1. Give the register value for each of the following instructions after it is executed. Assume R0=0x4545, R1=0x5454 and R2=0xFF00.

- a) AND R3, R2, R0
- b) AND R3, R2, R2
- c) AND R3, R0, #0xFF
- d) ORR R3, R0, #0x0F
- e) ORR R3, R2, R1
- f) EOR R0, R0, #0x45
- g) EOR R0, R0, R0

CODE:

```
AREA PROG, CODE, READONLY
```

```
ENTRY
```

```
LDR R0, =0X4545
```

```
LDR R1, =0X5454
```

```
LDR R2, =0XFF00
```

```
AND R3, R2, R0
```

```
AND R4, R2, R2
```

```
AND R5, R0, #0XFF
```

```
ORR R6, R0, #0X0F
```

```
ORR R7, R2, R1
```

```
EOR R8, R0, #0X45
```

```
EOR R9, R0, R0
```

```
END
```

OUTPUT:

The screenshot displays the Keil uVision3 IDE with the following components:

- Project Workspace:** Shows the project structure with files like `EXP31.s`.
- Register Window:** Lists registers R0 through R15, SP, PC, CPSR, and SPSR with their current values. For example, R0 is 0x00004545.
- Source Window:** Displays the assembly code for `EXP31.s`. The code includes an `AREA` directive, an `ENTRY` point, and several instructions: `LDR R0,=0X4545`, `LDR R1,=0X5454`, `LDR R2,=0XFF00`, `AND R3,R2,R0`, `AND R4,R2,R2`, `AND R5,R0,#0XFF`, `ORR R6,R0,#0X0F`, `ORR R7,R2,R1`, `EOR R8,R0,#0X45`, `EOR R9,R0,R0`, and `END`.
- Disassembly Window:** Shows the disassembled instructions with their addresses and opcodes. For example, `0x0000000C 00000000 ANDEQ R0,R0,R0`.
- Status Bar:** Displays simulation status, including time (0:01:09.2292 sec), CPU usage (78%), and temperature (30°C).

QUESTION 2

Write an instruction that sets bit 6 of R1 without affecting other bits.

CODE:

```
AREA LDCE, CODE, READONLY
```

```
ENTRY
```

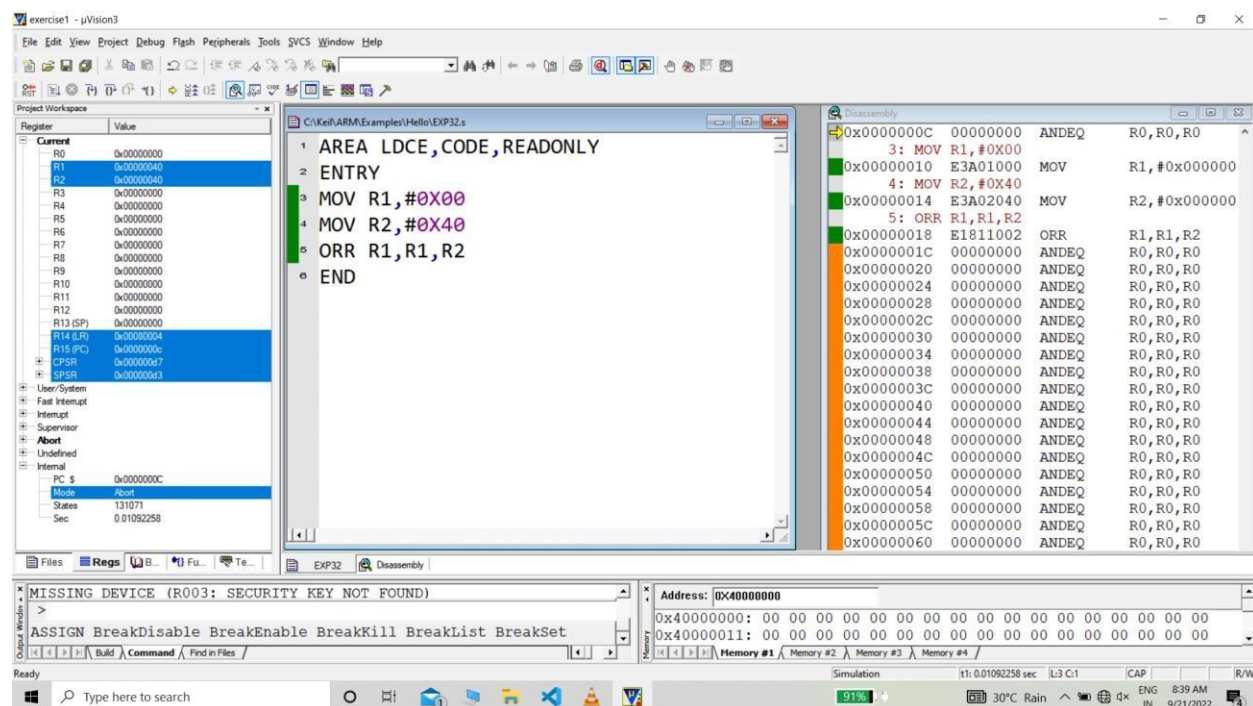
```
MOV R1, #0X00
```

```
MOV R2, #0X40
```

```
ORR R1, R1, R2
```

```
END
```

OUTPUT:



QUESTION 3

Write an instruction that clear bit 13 of R2 without affecting other bits.

CODE:

```
AREA PROG, CODE, READONLY
```

```
ENTRY
```

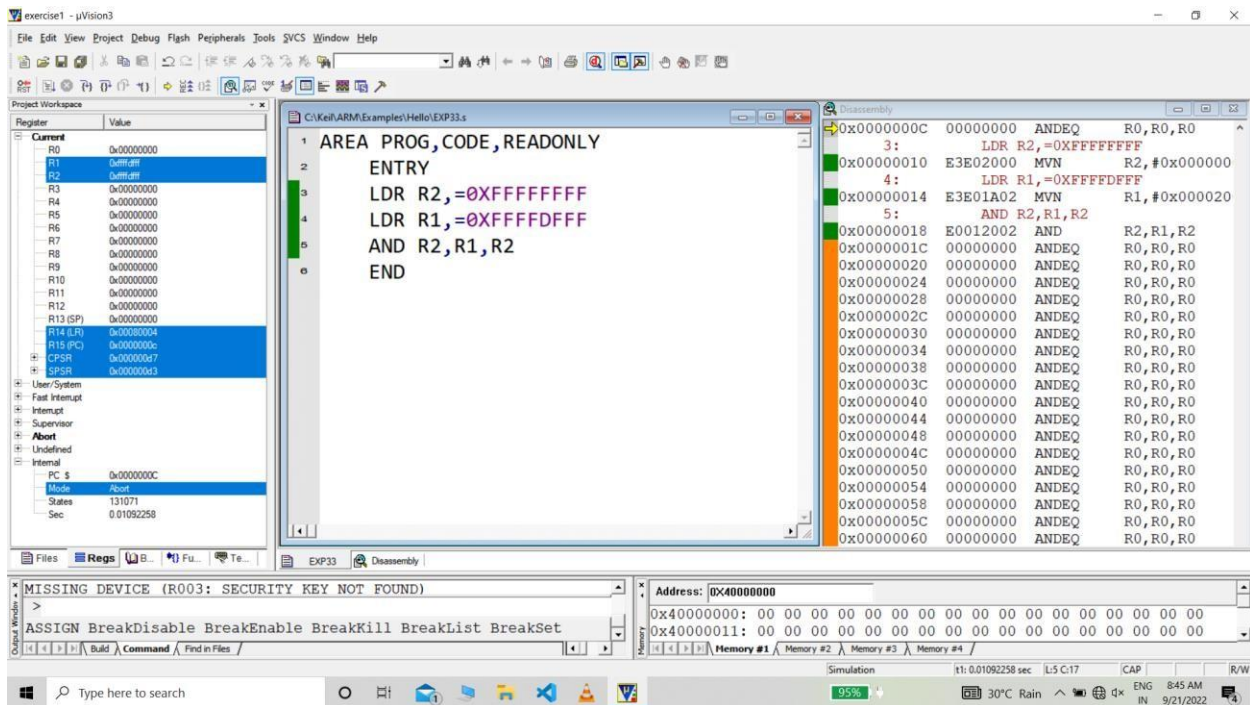
```
LDR R2, =0xFFFFFFFF
```

```
LDR R1, =0xFFFFDFFF
```

```
AND R2, R1, R2
```

```
END
```

OUTPUT:



QUESTION 4

Write a program to multiply the following values. (R1=0x5578, R2=0xaaa, R3=0xaabb987f, R4=0x12345678).

- a) Multiply R1 and R2
- b) Multiply R1 and R3
- c) Multiply R3 and R4
- d) Perform (R1 * R3 + R3)

CODE:

```
AREA PROG, CODE, READONLY
```

```
ENTRY
```

```
LDR R1,=0X5578
```

```
LDR R2,=0Xaaa
```

```
LDR R3,=0Xaabb987f
```

```
LDR R4,=0X12345678
```

```
LDR R10,=0X00
```

```
MUL R5,R1,R2
```

```
UMULL R6,R7,R1,R3
```

```
UMULL R8,R9,R3,R4
```

```
UMLAL R3,R10,R1,R3
```

```
END
```

OUTPUT:

The screenshot displays the Keil uVision3 IDE interface. The main window shows the assembly source code for a program named `EXP34.s`. The code is as follows:

```
AREA PROG, CODE, READONLY
ENTRY
LDR R1, =0X5578
LDR R2, =0Xaaa
LDR R3, =0Xaabb987f
LDR R4, =0X12345678
LDR R10, =0X00
MUL R5, R1, R2
UMULL R6, R7, R1, R3
UMULL R8, R9, R3, R4
UMLAL R3, R10, R1, R3
END
```

The left pane shows the Register window with the following values:

Register	Value
R0	0x00000000
R1	0x00005578
R2	0x0000aaaa
R3	0xf04d307
R4	0x12345678
R5	0x000078d0
R6	0x5191e588
R7	0x00003900
R8	0xf462583
R9	0x2e2d1878
R10	0x00003900
R11	0x00000000
R12	0x00000000
R13 (SP)	0x00000000
R14 (FP)	0x00000000
R15 (PC)	0x00000000
CPSR	0x00000007
SPSR	0x00000003

The right pane shows the Disassembly window with the following instructions:

Address	Instruction	Comment
0x00000000	ANDEQ R0, R0, R0	
3:	LDR R1, =0X5578	
0x00000010	E59F101C LDR R1, [PC, #0x00	
4:	LDR R2, =0Xaaa	
0x00000014	E59F201C LDR R2, [PC, #0x00	
5:	LDR R3, =0Xaabb987f	
0x00000018	E59F301C LDR R3, [PC, #0x00	
6:	LDR R4, =0X12345678	
0x0000001C	E59F401C LDR R4, [PC, #0x00	
7:	LDR R10, =0X00	
0x00000020	E3A0A000 MOV R10, #0x000000	
8:	MUL R5, R1, R2	
0x00000024	E0050291 MUL R5, R1, R2	
9:	UMULL R6, R7, R1, R3	
0x00000028	E0876391 UMULL R6, R7, R1, R3	
10:	UMULL R8, R9, R3, R4	
0x0000002C	E0898493 UMULL R8, R9, R3, R4	
11:	UMLAL R3, R10, R1, R3	
0x00000030	E0AA3391 UMLAL R3, R10, R1, R3	
0x00000034	00005578 DD 0x00005578	
0x00000038	00000AAA DD 0x00000AAA	
0x0000003C	AABB987F DD 0xAABB987F	
0x00000040	12345678 DD 0x12345678	
0x00000044	00000000 ANDEQ R0, R0, R0	

The bottom status bar shows the simulation is running at 100% speed, with a clock of 0:00:00.0358 sec. The system is in User mode, and the CPU is at 30°C. The date and time are 9:04 AM on 9/21/2022.

QUESTION 5

Write a program to convert unpacked BCD number into ASCII number.

CODE:

```
AREA PROG, CODE, READONLY
```

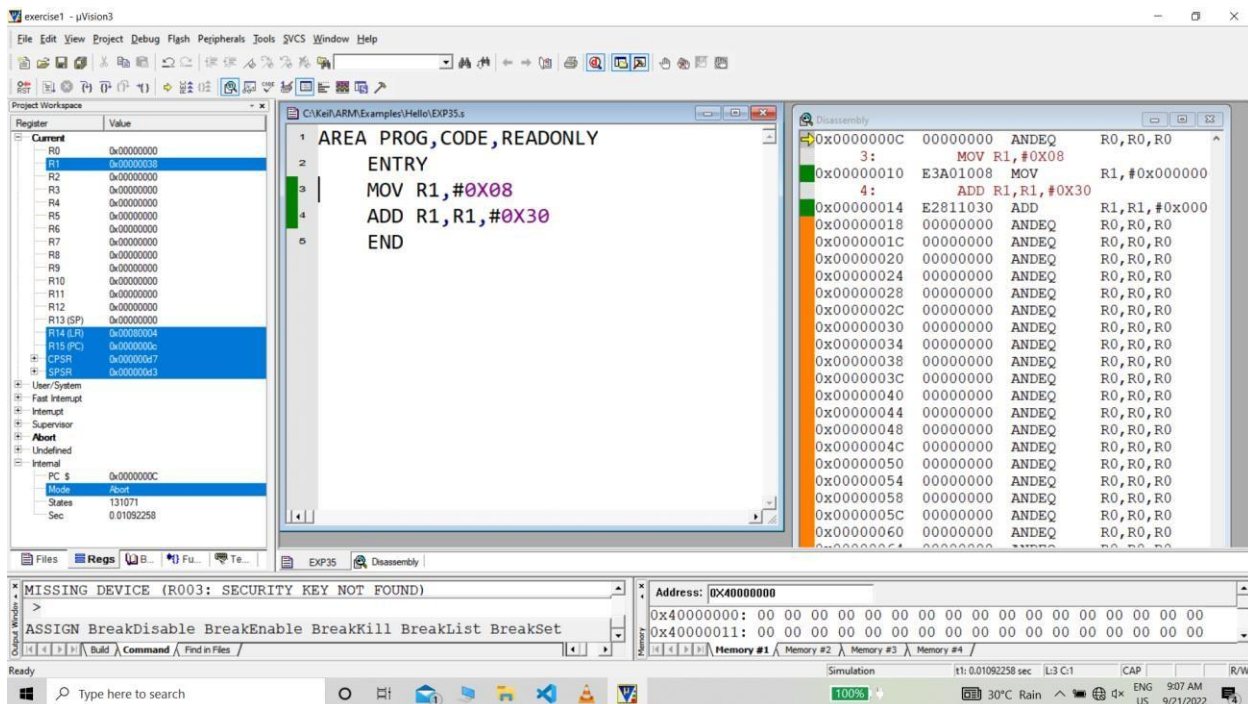
```
ENTRY
```

```
MOV R1, #0X08
```

```
ADD R1, R1, #0X30
```

```
END
```

OUTPUT:



QUESTION 6

Write a program to convert packed BCD number into ASCII number.

CODE:

```
AREA LDCE, CODE, READONLY
```

```
ENTRY
```

```
LDR R1, =0X38
```

```
AND R3, R1, #0X0F
```

```
ADD R3, R3, #0X30
```

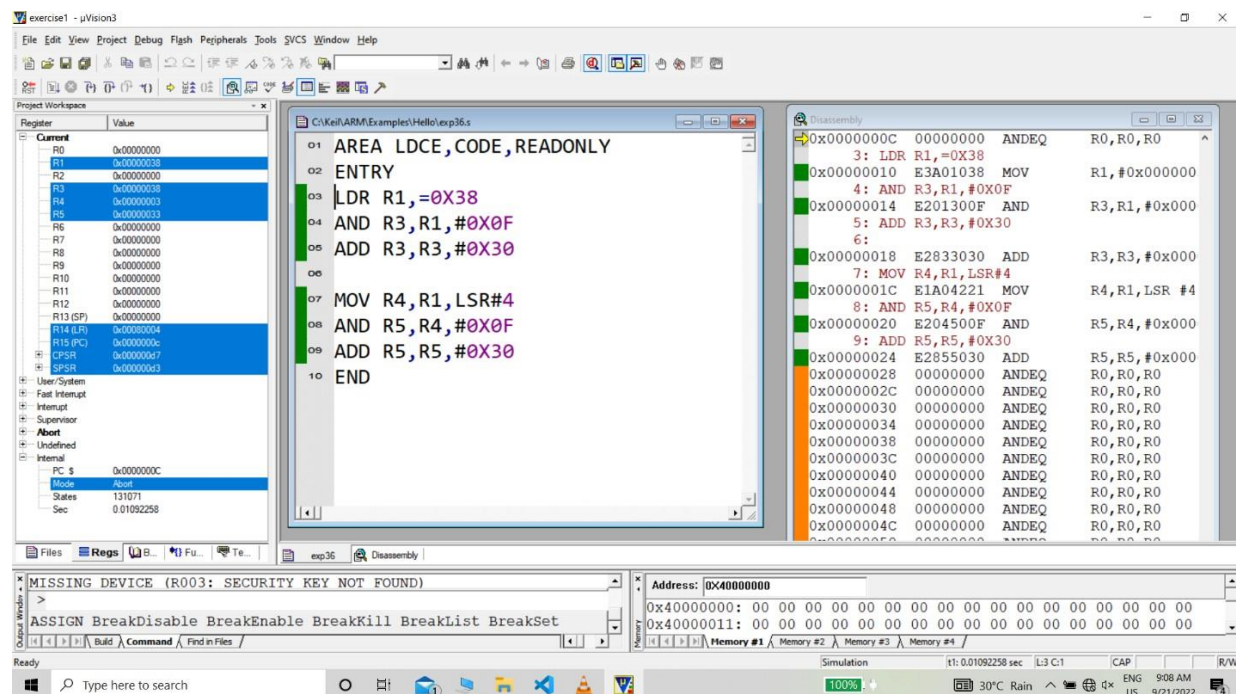
```
MOV R4, R1, LSR#4
```

```
AND R5, R4, #0X0F
```

```
ADD R5, R5, #0X30
```

```
END
```

OUTPUT:



QUESTION 7

Write a program to convert ASCII number into packed BCD number.

CODE:

```
AREA PROG,CODE,READONLY
```

```
ENTRY
```

```
MOV R1,#0X36
```

```
MOV R2,#0X38
```

```
SUB R3,R1,#0X30
```

```
SUB R4,R2,#0X30
```

```
ADD R5,R4,R3,LSL #4
```

```
END
```

OUTPUT:

