

NAME- JAIMIN PANCHAL

ENROLLMENT NO.- 200280111079

BATCH- A3, 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 interface. The main window shows the assembly code for a program named 'EXP31.s'. The code includes an 'AREA PROG, CODE, READONLY' directive, an 'ENTRY' label, and a series of 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'. The 'Registers' window on the left shows the current state of the registers, with R0, R1, R2, R3, R4, R5, R6, R7, R8, and R9 all containing the value 0x00000000. The 'Disassembly' window on the right shows the assembly code with comments and addresses. The 'Output' window at the bottom shows the status of the simulation, including the address 0x40000000 and the status 'Ready'.

Registers:

Register	Value
R0	0x00004545
R1	0x00005454
R2	0x00000000
R3	0x00004500
R4	0x00000000
R5	0x00000045
R6	0x00004545
R7	0x00000045
R8	0x00004500
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x00000000
R14 (LR)	0x00000004
R15 (PC)	0x0000000c
CPSR	0x00000047
SPSR	0x00000043

Assembly Code:

```
01 AREA PROG, CODE, READONLY
02 ENTRY
03 LDR R0, #0X4545
04 LDR R1, #0X5454
05 LDR R2, #0XFF00
06 AND R3, R2, R0
07 AND R4, R2, R2
08 AND R5, R0, #0XFF
09 ORR R6, R0, #0X0F
10 ORR R7, R2, R1
11 EOR R8, R0, #0X45
12 EOR R9, R0, R0
13 END
```

Disassembly:

Address	Instruction	Comment
0x00000000	ANDEQ R0, R0, R0	
0x00000010	E59F0020 LDR R0, [PC, #0x00	3: LDR R0, #0X4545
0x00000014	E59F1020 LDR R1, [PC, #0x00	4: LDR R1, #0X5454
0x00000018	E3A02CFF MOV R2, #0x0000FF	5: LDR R2, #0XFF00
0x0000001C	E0023000 AND R3, R2, R0	6: AND R3, R2, R0
0x00000020	E0024002 AND R4, R2, R2	7: AND R4, R2, R2
0x00000024	E20050FF AND R5, R0, #0x00	8: AND R5, R0, #0XFF
0x00000028	E380600F ORR R6, R0, #0x00	9: ORR R6, R0, #0X0F
0x0000002C	E1827001 ORR R7, R2, R1	10: ORR R7, R2, R1
0x00000030	E2208045 EOR R8, R0, #0x00	11: EOR R8, R0, #0X45
0x00000034	E0209000 EOR R9, R0, R0	12: EOR R9, R0, R0
0x00000038	00004545 DD 0x00004545	
0x0000003C	00005454 DD 0x00005454	
0x00000040	00000000 ANDEQ R0, R0, R0	
0x00000044	00000000 ANDEQ R0, R0, R0	

Output:

MISSING DEVICE (R003: SECURITY KEY NOT FOUND)

Address: 0x40000000

0x40000000: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

0x40000011: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

Simulation: 0.01092292 sec, L3 C:1, CAP, 8:22 AM, 3/21/2022

QUESTION 2

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

CODE:

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

```
ENTRY
```

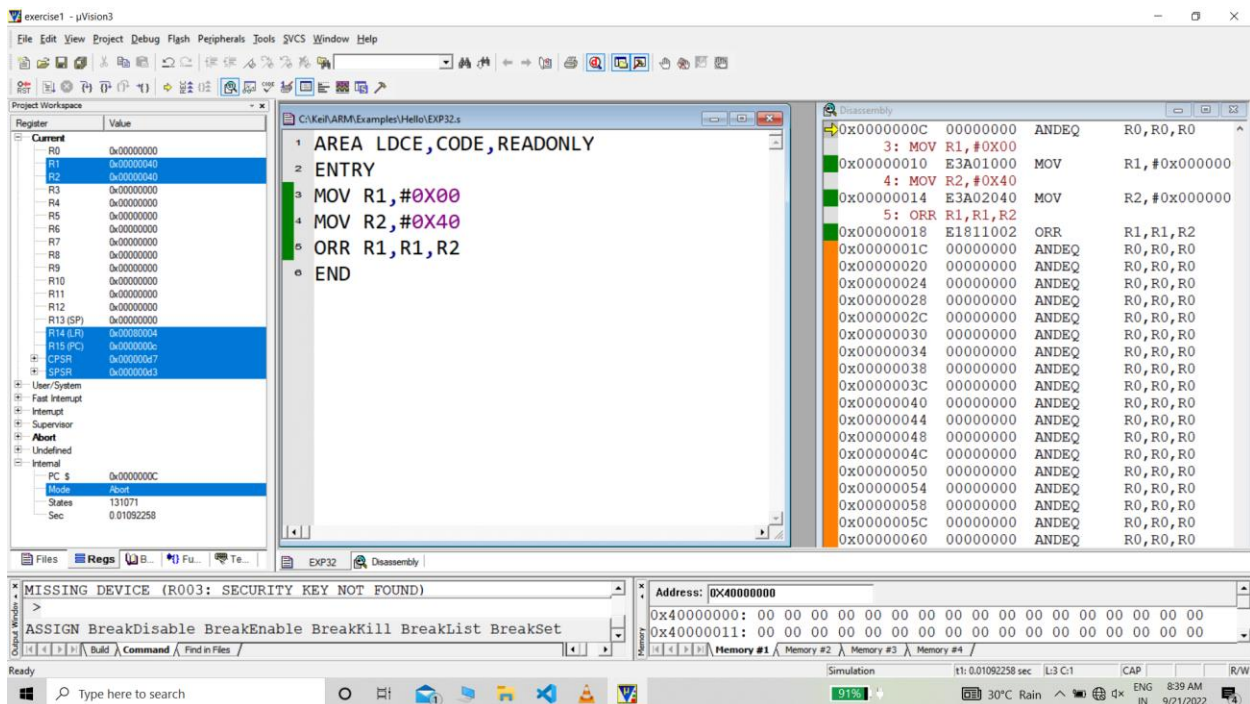
```
MOV R1, #0X00
```

```
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
```

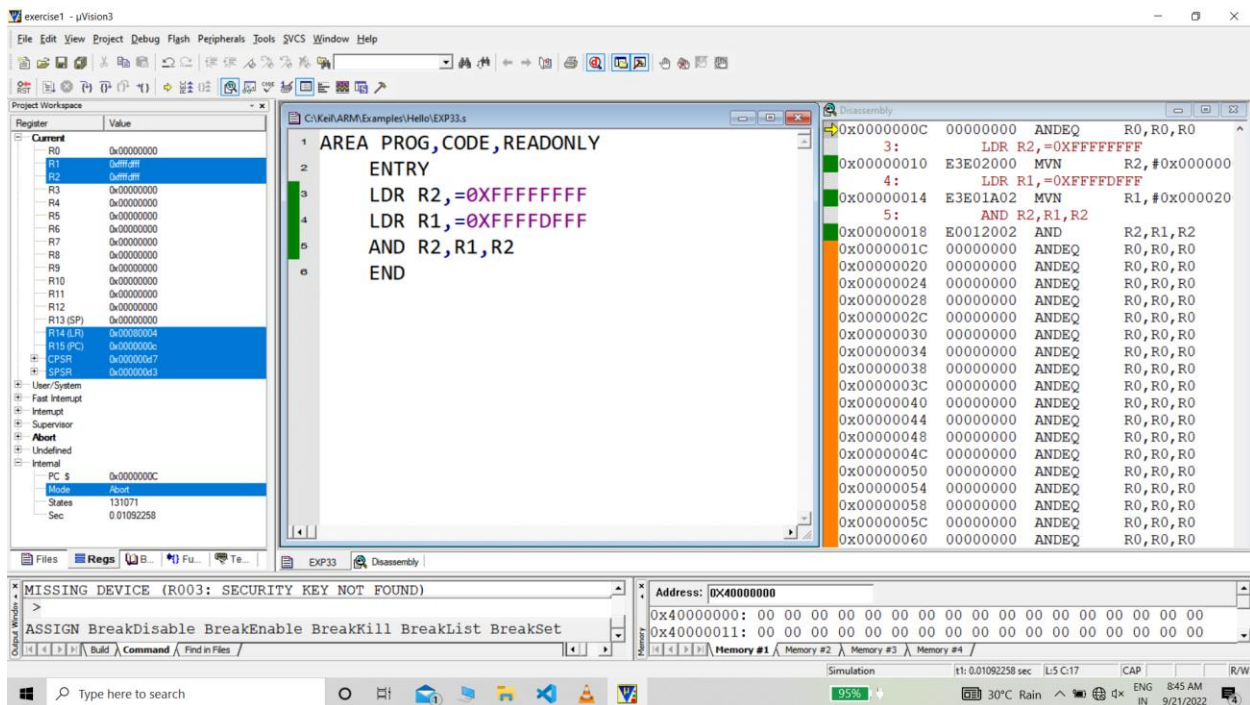
```
LDR R2, =0xFFFFFFFF
```

```
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 for the EXP34 target. The main window shows the assembly code for the program, which includes labels for registers and memory addresses. The Disassembly window on the right shows the corresponding machine code instructions. The status bar at the bottom indicates the simulation status and the target device.

Register Window:

Register	Value
R0	0x00000000
R1	0x00000578
R2	0x00000aaa
R3	0x00000000
R4	0x12345678
R5	0x00000000
R6	0x00000000
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x00000000
R14 (LR)	0x00000000
R15 (PC)	0x00000000
CPSR	0x00000000
SPSR	0x00000000

Source Window:

```
01 AREA PROG, CODE, READONLY
02 ENTRY
03 LDR R1,=0X5578
04 LDR R2,=0Xaaa
05 LDR R3,=0Xaabb987f
06 LDR R4,=0X12345678
07 LDR R10,=0X00
08 MUL R5,R1,R2
09 UMULL R6,R7,R1,R3
10 UMULL R8,R9,R3,R4
11 UMLAL R3,R10,R1,R3
12 END
```

Disassembly Window:

```
0x00000000: 00000000 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: AAB987F DD 0xAAB987F
0x00000040: 12345678 DD 0x12345678
0x00000044: 00000000 ANDEQ R0,R0,R0
```

Status Bar:

MISSING DEVICE (R003: SECURITY KEY NOT FOUND)

Address: 0X40000000

Simulation: 100% (0.00000358 sec)

ENG 9:04 AM 9/21/2022

QUESTION 5

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

CODE:

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

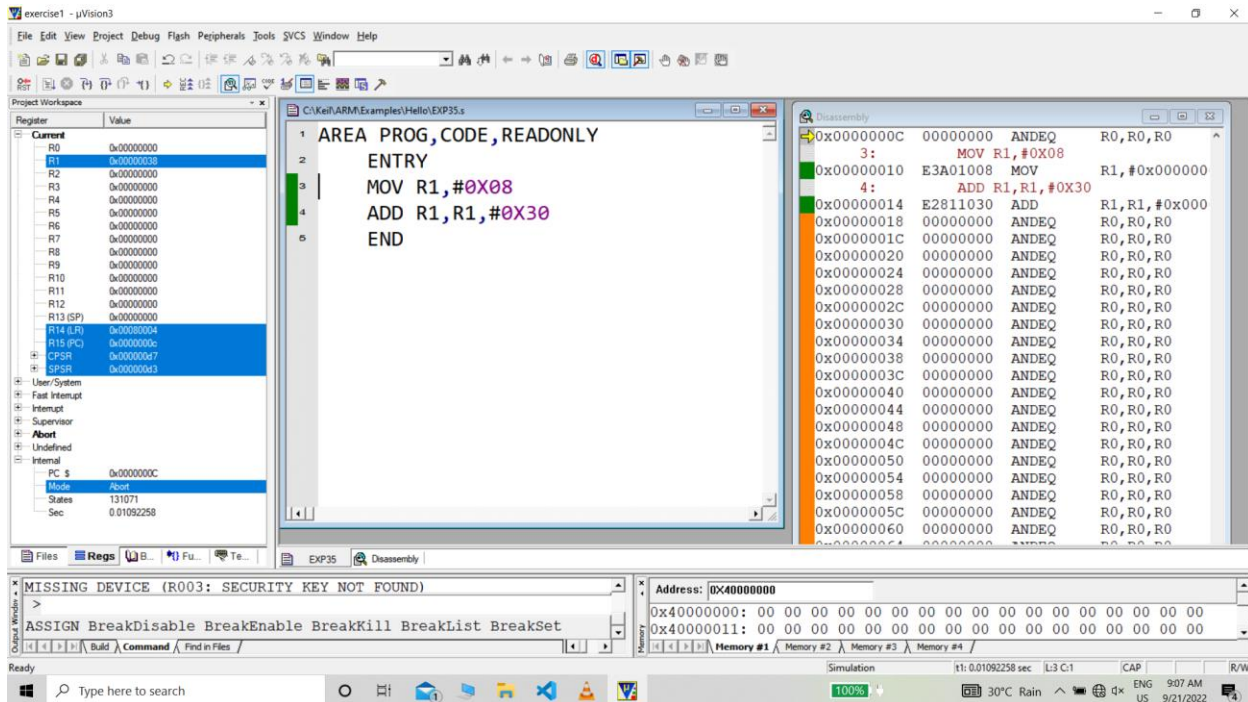
```
ENTRY
```

```
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
```

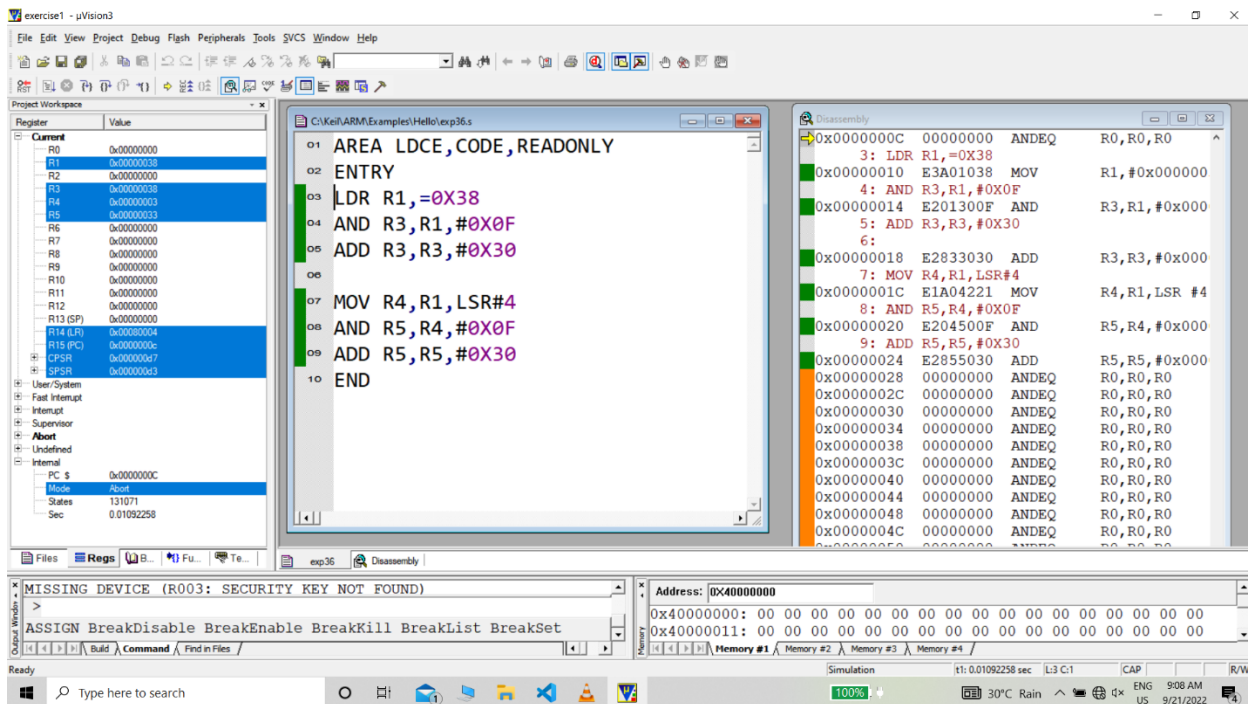
```
MOV R4, R1, LSR #4
```

```
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:

