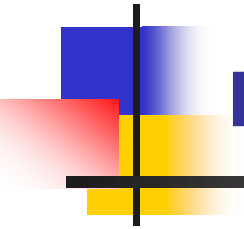


Microprocessor Systems

ASSEMBLY PROGRAMMING

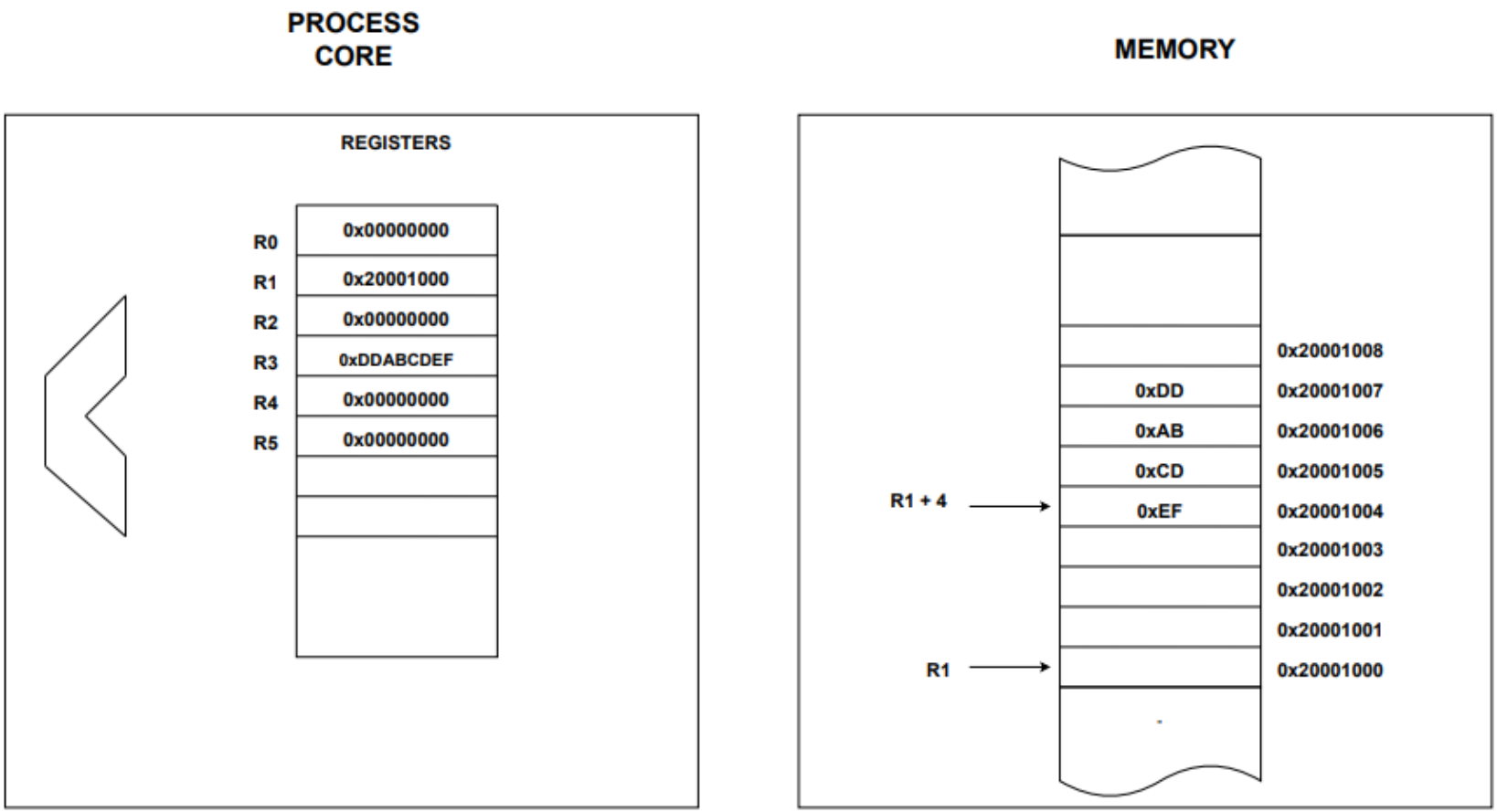
Question 1

- Assume that **R3=0xddabcdef** and **R1=0x20001000**, and all other registers and memory are initialized to zero. After running the following code, what are the values of registers R1, R3, and R5?

```
STR R3, [R1, #4]
LDRB R5, [R1, #5]
MOVS R7, #0x8F
ORRS R5, R5, R7
STM R1!, {R3}
LDRH R3, [R1]
```

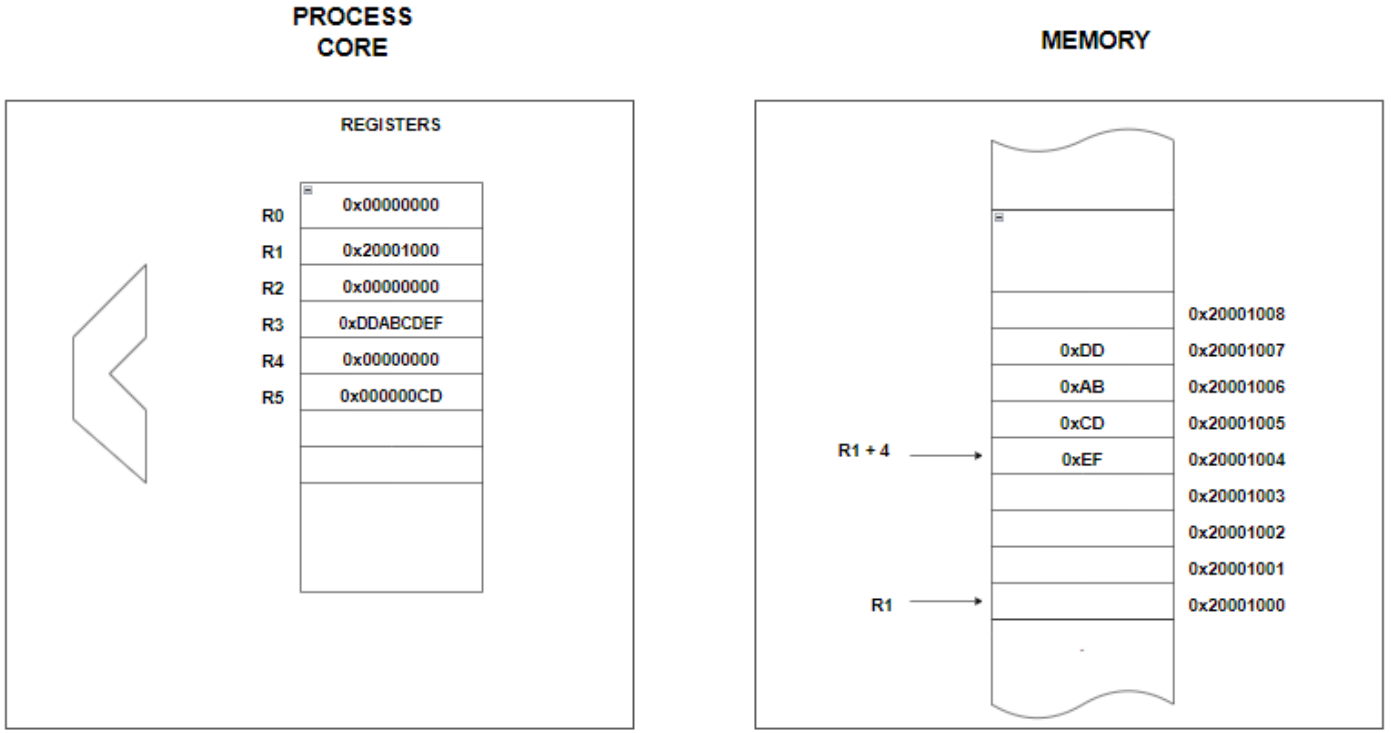
Question 1

```
STR R3, [R1, #4]
```



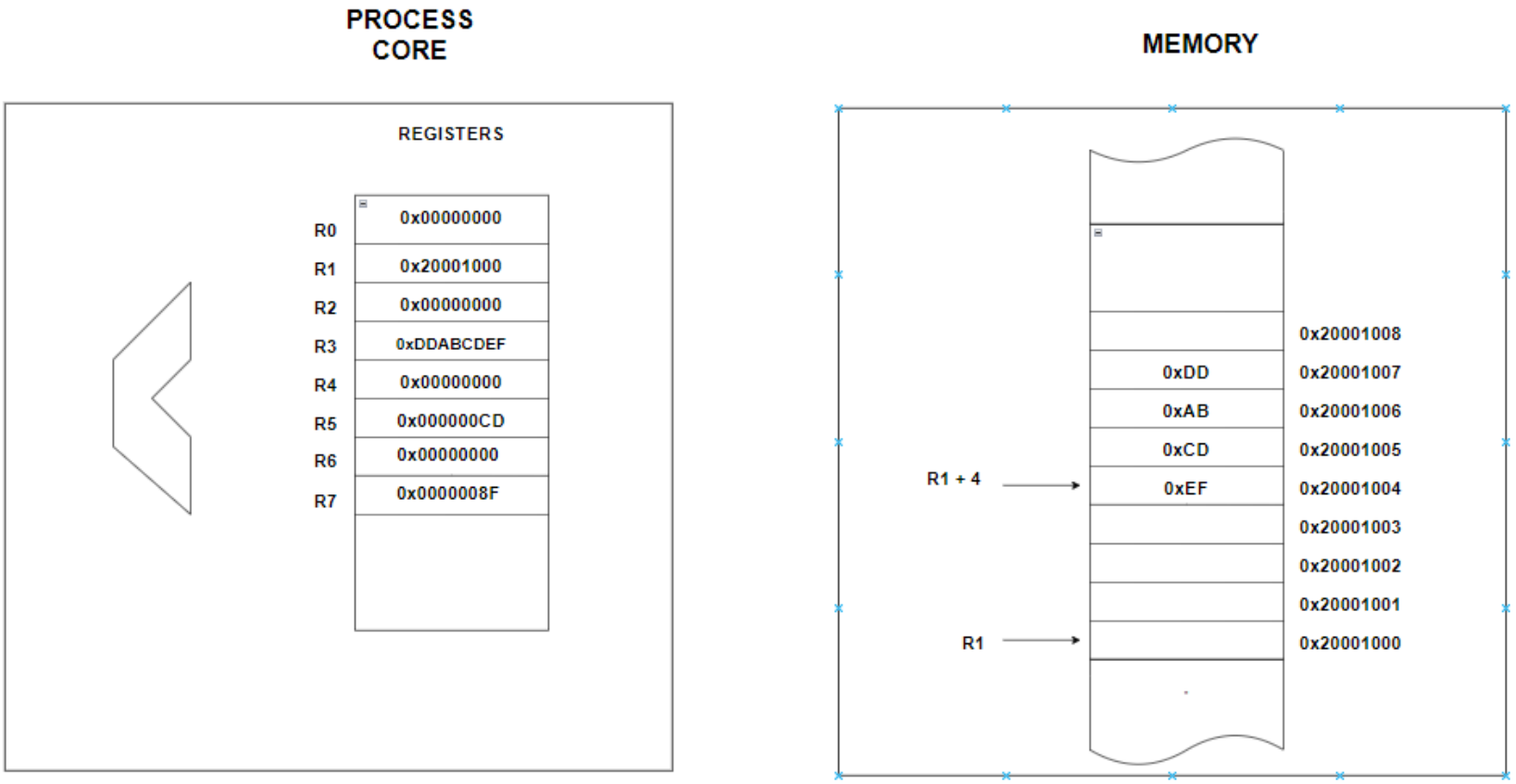
Question 1

```
LDRB R5, [R1, #5]
```



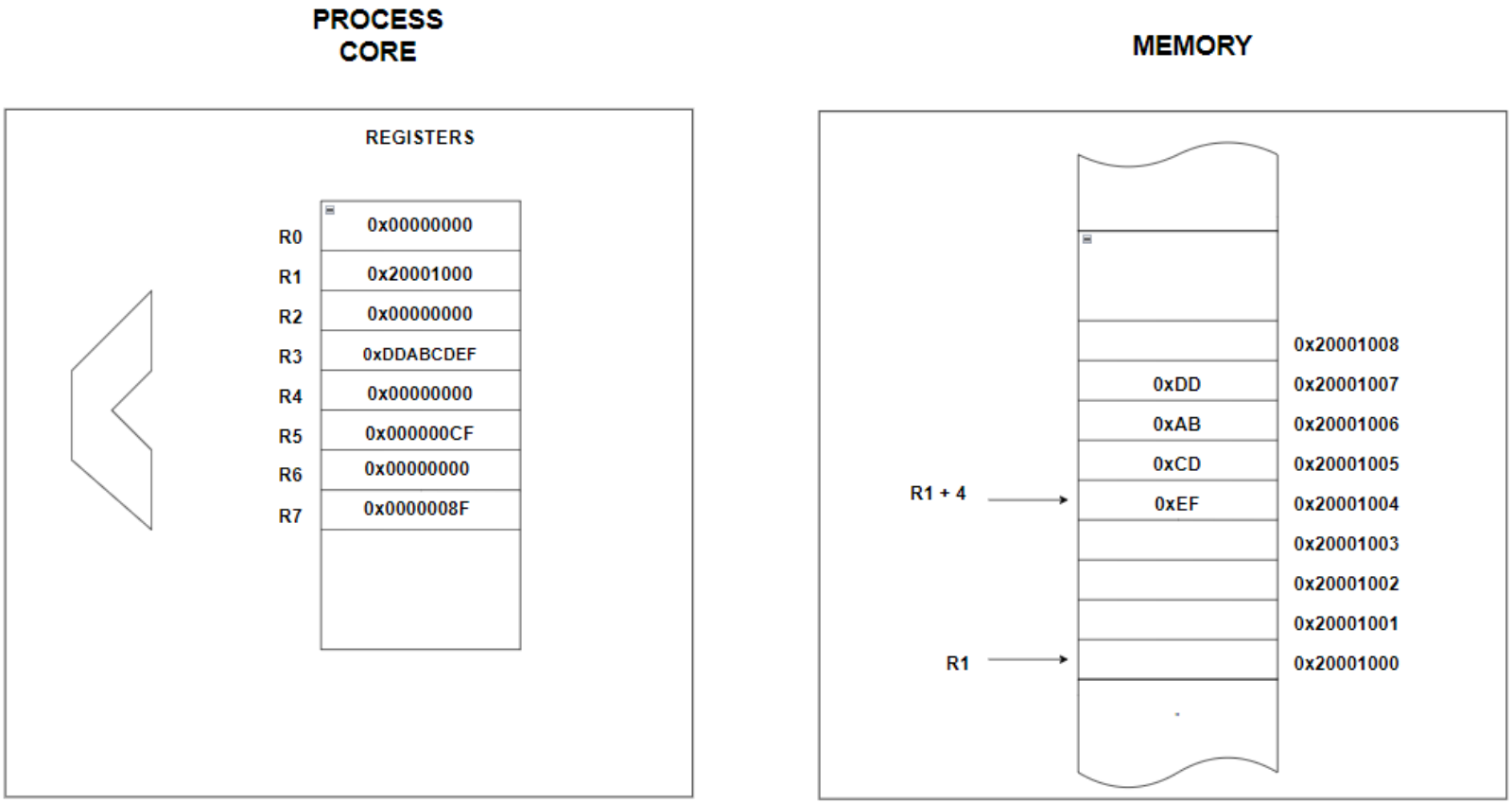
Question 1

```
MOVS R7, #0x8F
```



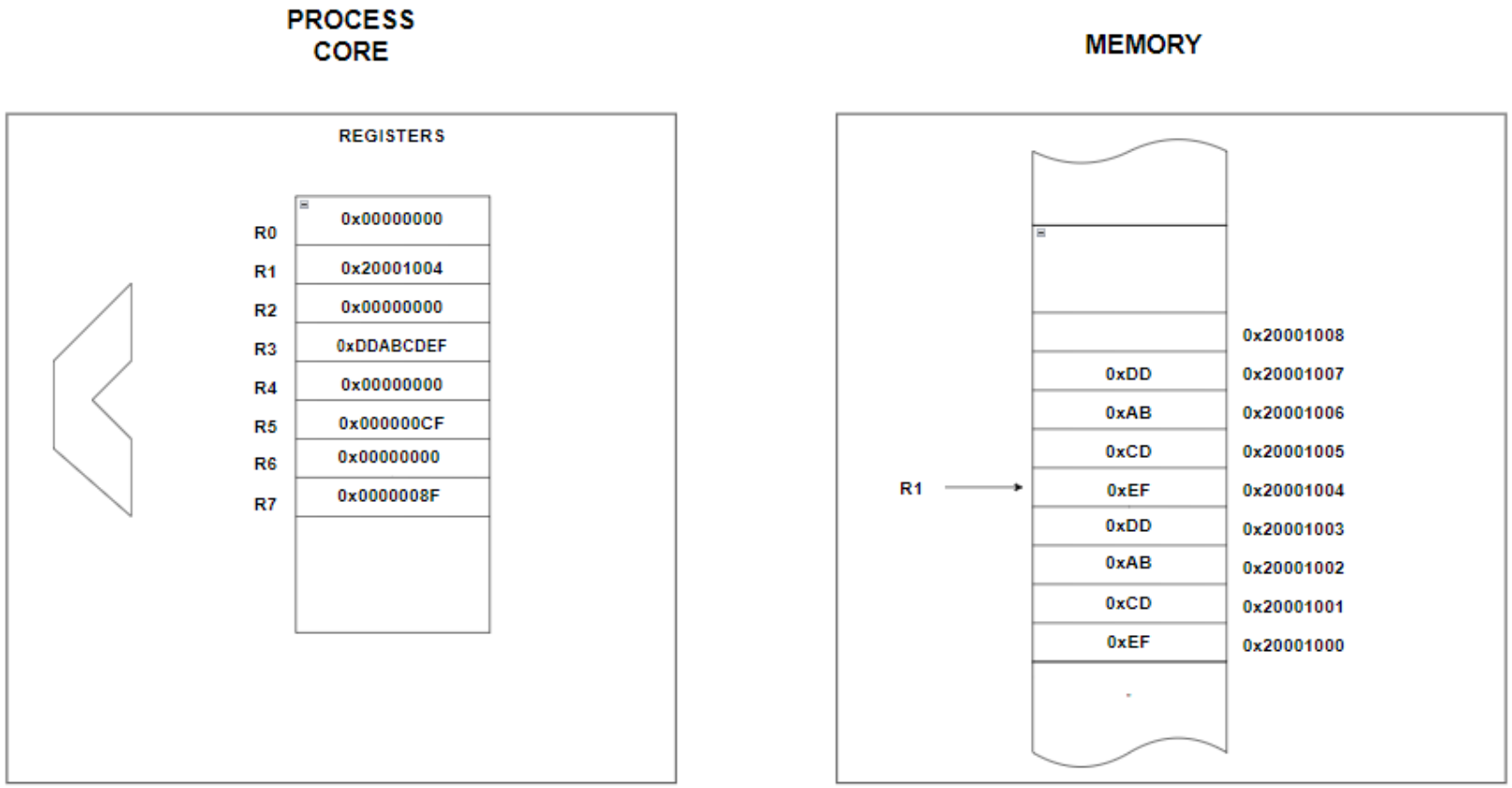
Question 1

```
ORRS R5, R5, R7
```



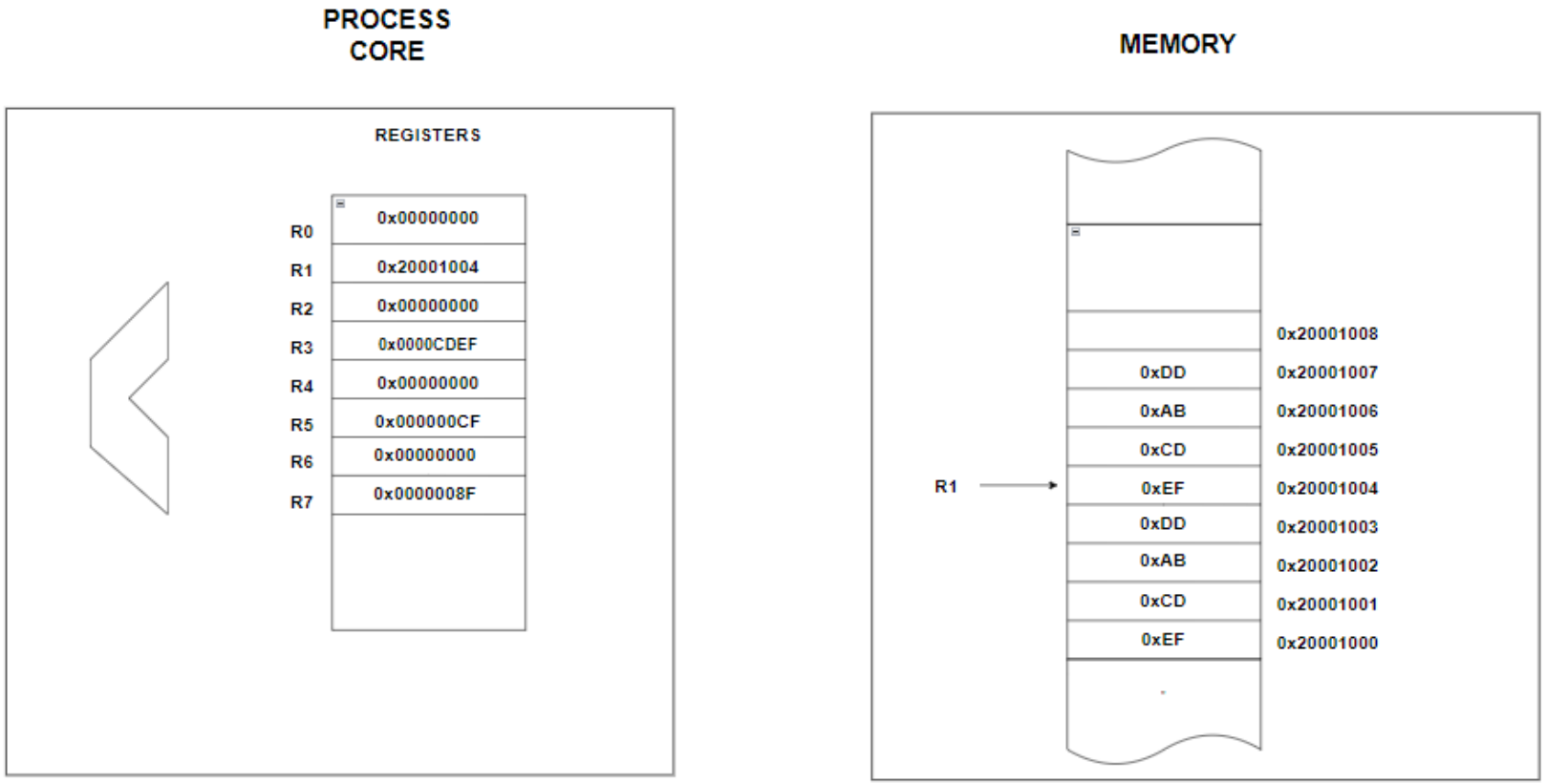
Question 1

```
STM R1!, {R3}
```



Question 1

```
LDRH R3, [R1]
```



Question 2

- For the following program segment, assume you start with all memory locations in question equal to zero. Indicate the values found in these memory locations when the programs end. Write all answers in hex.

```
LDR R1, =0x20000100
MOVS R2, #80
STM R1!, {R2}
STR R2, {R1, #4}
STRH R2, {R1, #6}
```

Question 2

```
LDR R1, =0x20000100
MOVS R2, #80
STM R1!, {R2}
STR R2, {R1, #4}
STRH R2, {R1, #6}
```

Memory Address	Value
0x20000100	0x50
0x20000101	0x00
0x20000102	0x00
0x20000103	0x00
0x20000104	0x00
0x20000105	0x00
0x20000106	0x00
0x20000107	0x00
0x20000108	0x50
0x20000109	0x00
0x2000010A	0x50
0x2000010B	0x00

Question 3 – Array Copy

- Write a program using the instruction set for the Arm Cortex-M0 that copies an array to other one.

```c
#define SIZE 10

int main(){
    int x[SIZE]={12,20,25,60,15,53,17,65,22,1};
    int y[SIZE];

    for(int i=0;i<SIZE;i++){
        y[i]=x[i];
    }

    return 0;
}
```

C Code

Question 3 – Array Copy

C Code

```c
#define SIZE 10

int main(){
    int x[SIZE]={12,20,25,60,15,53,17,65,22,1};
    int y[SIZE];

    for(int i=0;i<SIZE;i++){
        y[i]=x[i];
    }

    return 0;
}
```

Assembly Structure

```
#define SIZE 10

;allocate 10 unit area from memory for y

int main(){
    int i =0
    while(i<SIZE){
        int temp=x[i];
        y[i]=temp;
        i++;
    }
}

; write data to code area as x
```

Question 3 – Array Copy - Solution

```
ArraySize   EQU 0x28                              ;Array size = 40

        AREA     My_Array, DATA, READWRITE        ;Define this part will write in data area
        ALIGN
y_array SPACE    ArraySize                        ;Allocate space from memory for y
y_end

        AREA copy_array, code, readonly           ;Define this part will write as code
        ENTRY
        THUMB
        ALIGN
__main  FUNCTION
        EXPORT __main


        MOVS     r3,#ArraySize                    ;Load array size
        MOVS     r0,#0                            ;i=0 as index value
        LDR      r1,=y_array                      ;Load start address of the allocated space for y
        LDR      r2,=x_array                      ;Load start address of x
Copy    CMP      r0,r3                            ;Check i<array_size
        BGE      stop                             ;if not finish loop
        LDR      r5,[r2,r0]                       ;temp = x[i]
        STR      r5,[r1,r0]                       ;x[i] = temp
        adds     r0,r0,#4                         ;i=i+4 for word.
        B        Copy                             ;End of the loop, jump start point
stop    B stop                                    ;Infinite loop
        ALIGN
        ENDFUNC

x_array DCD      12,20,25,60,15,53,17,65,22,1     ;write x array to code memory
x_end
        END
```

Question 4 – Array Sort

- Write a program block using the instruction set for the Arm Cortex-M0 that sorts an array using bubble sort algorithm.

```
for(int i=0; i<SIZE-1;i++){
    for(int j=0;j<SIZE-1-i;j++){
        if(y[j]>y[j+1]){
            temp=y[j];
            y[j]=y[j+1];
            y[j+1]=temp;
        }
    }
}
```

C Code

Question 4 – Array Sort

```c
for(int i=0; i<SIZE-1;i++){
    for(int j=0;j<SIZE-1-i;j++){
        if(y[j]>y[j+1]){
            temp=y[j];
            y[j]=y[j+1];
            y[j+1]=temp;
        }
    }
}
```

C Code

```
int i = 0;
int size=SIZE-1;
while(i<size){
    int j=0;
    int cond=size-i;
    while(j<cond){
        int firstval=y[j];
        j++;
        int secondval=y[j];
        if(firstval>secondval){
            y[j]=firstval;
            j--;
            y[j]=secondval;
            j++;
        }
    }
    i++;
}
```

Assembly Structure

Question 4 – Array Sort - Solution

```
endcopy MOVS    r0,#0                   ;i=0 as index value
        MOVS    r3,#ArraySize           ;Load array size
        SUBS    r3,r3,#4                ;size=SIZE-1
        LDR     r2,=y_array             ;Load start address of the allocated space for y
L1      CMP     r0,r3                   ;Check i<array_size
        BGE     stop                    ;if not finish loop
        MOVS    r1,#0                   ;j=1 as second index
        MOVS    r4,r3                   ;cond=size
        SUBS    r4,r4,r0                ;cond = size-1
L2      CMP     r1,r4                   ;check j< cond
        BGE     EndL2                   ;if j >= cond, finish inner loop
        LDR     r5,[r2,r1]              ;firstval = y[j]
        ADDS    r1,r1,#4                ;j=j+4
        LDR     r6,[r2,r1]              ;secondval = j[i]
        CMP     r5,r6                   ;check firstval > secondval
        BLE     L2                      ;if firstval<=second then jump L1
        STR     r5,[r2,r1]              ;y[j]=firstval
        SUBS    r1,r1,#4                ;j=j-4
        STR     r6,[r2,r1]              ;y[j]=firstval
        ADDS    r1,r1,#4                ;j=j+4
        B       L2                      ;Go to L2
EndL2   ADDS    r0,r0,#4                ;i=i+4 for word.
        B       L1                      ;Go to L1

stop    B stop                          ;Infinite loop
```