BLG212E -Microprocessor Systems

Tacettin Ayar
(ayart@itu.edu.tr)

Number Formats (2010 Midterm-I) (1)

Perform the following operations by using the given numbers A and B (A=53 and B=29).

a) Calculate the B-A operation in compliance with two‘s complement arithmetics.

b) Calculate the A+B operation in binary-coded decimal (BCD) format.

Number Formats (2010 Midterm-I) (2)

a) Calculate the B-A operation in compliance with two‘s complement arithmetics.

A = 53 = %00110101   and   B = 29 = %00011101

The two‘s complement of A = 11001011 = -A

B – A = B + (-A)

  00011101

+ 11001011

  11101000 : S = B - A

Number Formats (2010 Midterm-I) (3)

b) Calculate A+B in binary-coded decimal (BCD) format.

A = !53 = 0101 0011 and B = !29 = 0010 1001

5 3 2 9

0101 0011 10 = %1010 => -10 = %0110

+ 0010 1001 =====> 0111 1100 ==> first sum

0111 1100 + 1 0110 ==> **Correction**

7 12 1000 10010 ==> !82

12: Not a single digit: half-carry occurs. **Correct** it.

Substract 10 from 12 (leave the remainder 2 here) and add 1 to 7.

Memory Design (2010 Midterm-I) (1)

Design a 16K*8 memory by using R/W type 2K*8 memory chips.

The start address of the memory will be $0000.

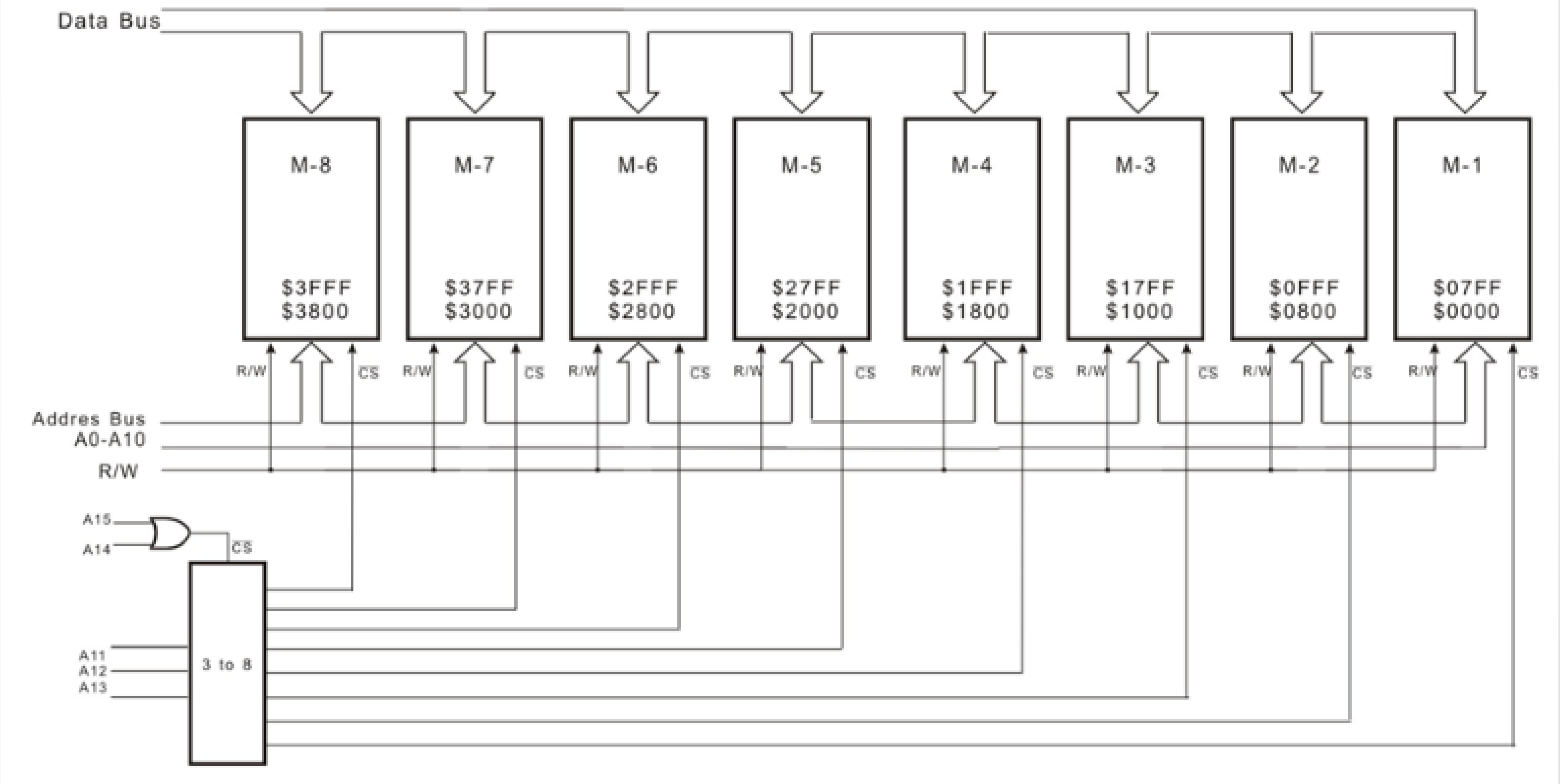
The address bus of the CPU is 16-bits and the data bus is 8-bits.

a) Draw the required hardware design.

b) Using the same designed unit, design a chip selector for a second 16K*8 memory.

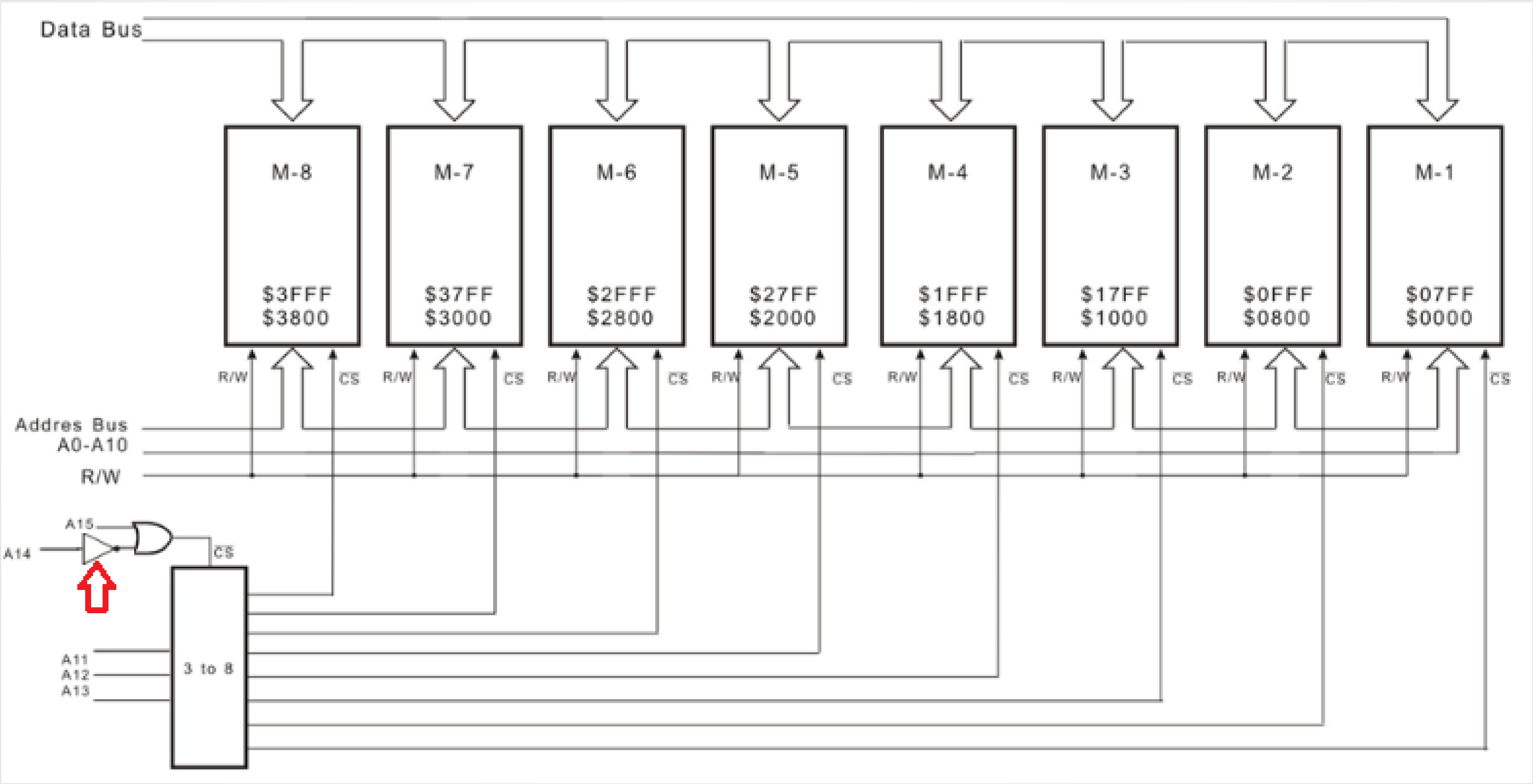
Memory Design (2010 Midterm-I) (2)

a) Draw the required hardware design.



Memory Design (2010 Midterm-I) (3)

b) Using the same designed unit, design a chip selector for a second 16K*8 memory.



Machine Code (1)

Convert the given program into the machine codes of the example microprocessor and write the starting addresses of each instruction to the address column.

	Register
000	A
001	B
010	C
011	D
100	CCR
101	IX
110	SP

Address			Machine Code	Label	Intruction
					ORG $0100
				START	LDA B, <$000A>
					LDA IX, <$000B>
					CLR A
				LOOP	ADD A, <IX+00> +1
					DEC B
					BHI LOOP
					CMP A, **$7F**
					BGT FINISH
					STA A, $011F
				FINISH	SWI

Machine Code (2)

Convert the given program into the machine codes of the example microprocessor and write the starting addresses of each instruction to the address column.

	Register
000	A
001	B
010	C
011	D
100	CCR
101	IX
110	SP

Address			Machine Code	Label	Instruction
					ORG $0100
0100			00 21 00 0A	START	LDA B, <$000A>
0104			20 25 00 0B		LDA IX, <$000B>
0108			4B 40		CLR A
010A			03 80 00 01	LOOP	ADD A, <IX+00> +1
010E			51 41		DEC B
0110			86 F8		BHI LOOP
0112			1C 00 7F		CMP A, **$7F**
0115			83 04		BGT FINISH
0117			01 20 01 1F		STA A, $011F
011B			C3	FINISH	SWI

Machine Code (3)

Assume that the contents of the memory cells are as in the left table at the beginning of the program.
After the program is run, what are the new contents of the memory cells?

Before

000A	03
000B	00
000C	0D
000D	01
000E	02
000F	03
0010	05
0011	08
:	
011F	FF

After

000A	??
000B	??
000C	??
000D	??
000E	??
000F	??
0010	??
0011	??
:	
011F	??

Machine Code (4)

Assume that the contents of the memory cells are as in the left table at the beginning of the program.
After the program is run, what are the new contents of the memory cells?

Before

000A	03
000B	00
000C	0D
000D	01
000E	02
000F	03
0010	05
0011	08
:	
011F	FF

After

000A	03
000B	00
000C	0D
000D	01
000E	02
000F	03
0010	05
0011	08
:	
011F	06

Address			Machine Code	Label	Instruction
					ORG $0100
0100			00 21 00 0A	START	LDA B, <$000A>
0104			20 25 00 0B		LDA IX, <$000B>
0108			4B 40		CLR A
010A			03 80 00 01	LOOP	ADD A, <IX+00> +1
010E			51 41		DEC B
0110			86 F8		BHI LOOP
0112			1C 00 7F		CMP A, **$7F**
0115			83 04		BGT FINISH
0117			01 20 01 1F		STA A, $011F
011B			C3	FINISH	SWI

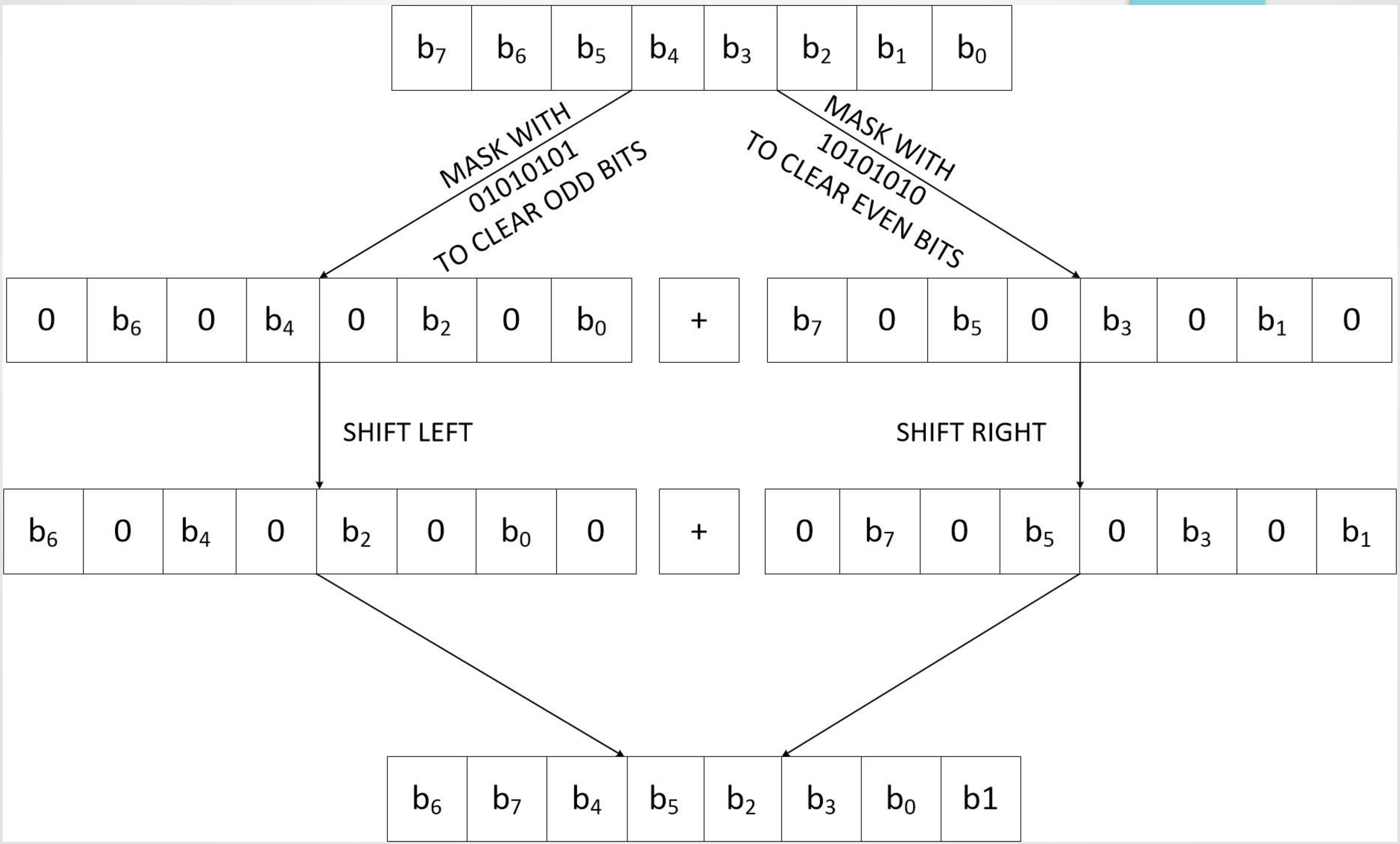
Example Program: Swap Bits (1)

- Write a program using the instruction set for the educational CPU that swaps the even and odd bits of an 8 bit integer:
  - Swap bit 0 with bit 1, bit 2 with bit 3, ...
- Integer is in $1000
- New integer should be in $1001

%0**11**0**1**1**01** → %**1**0**0**1**1**1**1**0

$8D → $9E

Example Program: Swap Bits (2)



Example Program: Swap Bits (3)

```
        ORG $1000       The Byte Is At $1000
        DAT $A5

        ORG $1002       Place Machine Code Starting From $1002

START   LDA A, <$1000>  Load The Byte Into ACC-A
        AND A, $55      Mask With 0101 0101 To Clear ODD Bits
        LSL A           Shift The Result To The Left (EVEN Bits Become ODD Bits)
        STA A, <$1001>  Store The Partial Result in $1001 (Memory Location of The Result)

        LDA A, <$1000>  Load The Byte Into ACC-A
        AND A, $AA      Mask With 1010 1010 To Clear EVEN Bits
        LSR A           Shift The Result To The Right (ODD Bits Become EVEN Bits)
        ADD A, <$1001>  ADD ONLY-EVEN Bits and ONLY-ODD Bits

        STA A, <$1001>  Store The Final Result in $1001 (Memory Location of The Result)
        SWI             End Of The Program
```

Example Program: Create Sub-Array (Midterm 2008) (1)

- There is an array in memory that contains at most 255 unsigned integers. Create an array that contains integers from this array that are smaller than a threshold and calcutate its size.
- Array starts from $1000 and its size is in $0FFF
- Threshold value is in $0FFE
- Sub-array should start from $1100 and its size should be in $10FF

Example Program: Create Sub-Array (Midterm 2008) (2)

- We need to process all the array elements:
  - IX can be used to index the array elements
- We need to loop until array size is reached
  - Use a counter in ACC-B to stop looping
- We need to know where the array data that are smaller than the threshold must be stored
  - Sub-array should start from $1100. Use CD registers to keep that address
  - After an array element is stored, increase D to point to the next-store address which is also sub-array size

Example Program: Create Sub-Array (Midterm 2008) (3)

```
        ORG $0FFE                               Threshold value is in $0FFE
        DAT $A                                  Threshold Value is $A
        DAT $7                                  Array Size = 7 is in $0FFF
        DAT $7, $8, $A, $15, $2E, $1, $5        Array data starts From $1000


        ORG $1100 + $5          Indeed, Must Be $1100 + $FF Since Max Array-Size is 255
*                               $5 Is Enough For This Example

START   LDA IX, $1000           LOAD Start Address of The Array Into IX
        LDA CD, $1100           LOAD Next-Slot Address of The Sub-Array Into CD
        CLR B                   ACC-B Keeps The Number of Processed Elements
LOOP    CMP B, <$0FFF>          Is B Equal To Array Size?
        BEQ FINISH              Finish The Program: Reached To The End Of The Array
        INC B                   Increment Number Of Processed Elements
        LDA A, <IX+00>          Load Current Array Element Into ACC-A    | OR IN A SINGLE INSTRUCTION |
        INC IX                  Set Current Element As The Next Element |      LDA A, <IX+00> + 1     |
        CMP A, <$0FFE>          Compare ACC-A With The Threshold
        BGE LOOP                If Value >= Threshold, Process The Next Array Element
        STA A, <CD>             Else Store ACC-A As The Next Element Of Sub-Array
        INC D                   Increment Next-Slot Address of The Sub-Array
        JMP LOOP                Repeat The Loop
FINISH  STA D, <$10FF>          D Includes Number Of Stored Elements Into The Sub-Array
        SWI                     End The Program
```