Problem Solving Microprocessor Systems

PROBLEM 1

- There is an array (ARY1) consisting unsigned, 8-bit integers in the memory that starts from memory address $0100. The size of ARY1 is stored in $000A
- Another array (ARY2) starting from $0200, should be created using the elements of ARY1, which are lower than the number in the memory address $000F.
- The size of ARY2 is stored in $000B.
- Write this program using the instruction set of Motorola 6800. Write brief comments next to your code.

SOLUTION 1 –INITIAL VALUES

Values in ARY1 is set. Size of array is stored.

```
        .ORG $0000
        LDAA #4
        STAA $000A
        LDAA #8
        STAA $000F
        LDAA #0
        STAA $000B
        .ORG $0100
ARY1    .byte 3,9,2,4
        .end
```

Memory Address	Content (Hexadecimal)
$000A	$04
$000F	$08
$000B	$00
$0100 (start of ARY1)	$03
$0101	$09
$0102	$02
$0103 (end of ARY1)	$04

MAIN PROGRAM		A	B	IX	SP	$000B
START	LDX #$0100	00	00	0100	F000	00
	CLRB	00	00	0100	F000	00
BACK	LDAA ,x	03	00	0100	F000	00
	CMPA $000F	03	00	0100	F000	00
	BHI FORWARD	03	00	0100	F000	00
	PSHA	03	00	0100	EFFF	00
	INC $000B	03	00	0100	EFFF	01
FORWARD	INCB	03	01	0100	EFFF	01
	CMPB $000A	03	01	0100	EFFF	01
	BEQ STOP	03	01	0100	EFFF	01
	INX	03	01	0101	EFFF	01
	BRA BACK	03	01	0101	EFFF	01

Memory Address	Content (hexadecimal)
$000A	$04
$000F	$08
$F000 (STACK)	$03

MAIN PROGRAM		*A*	*B*	*IX*	*SP*	*$000B*
START	LDX #$0100					
	CLRB					
BACK	LDAA ,x	09	01	0101	EFFF	01
	CMPA $000F	09	01	0101	EFFF	01
	BHI FORWARD	09	01	0101	EFFF	01
	PSHA					
	INC $000B					
FORWARD	INCB	09	02	0101	EFFF	01
	CMPB $000A	09	02	0101	EFFF	01
	BEQ STOP	09	02	0101	EFFF	01
	INX	09	02	0102	EFFF	01
	BRA BACK	09	02	0102	EFFF	01

Memory Address	Content (hexadecimal)
$000A	$04
$000F	$08
$F000 (STACK)	$03

MAIN PROGRAM		*A*	*B*	*IX*	*SP*	*$000B*
START	LDX #$1000					
	CLRB					
BACK	LDAA ,x	02	02	0102	EFFF	01
	CMPA $000F	02	02	0102	EFFF	01
	BHI FORWARD	02	02	0102	EFFF	01
	PSHA	02	02	0102	EFFE	01
	INC $000B	02	02	0102	EFFE	02
FORWARD	INCB	02	03	0102	EFFE	02
	CMPB $000A	02	03	0102	EFFE	02
	BEQ STOP	02	03	0102	EFFE	02
	INX	02	03	0102	EFFE	02
	BRA BACK	02	03	0102	EFFE	02

Memory Address	Content (hexadecimal)
$000A	$04
$000F	$08
$EFFF(STACK)	$02
$F000 (STACK)	$03

MAIN PROGRAM		*A*	*B*	*IX*	*SP*	*$000B*
START	LDX #$1000					
	CLRB					
BACK	LDAA ,x					
	CMPA $000F					
	BHI FORWARD					
	PSHA					
	INC $000B					
FORWARD	INCB					
	CMPB $000A					
	BEQ STOP					
	INX					
	BRA BACK					
WRITE	CLRB	00	00	0103	EFFD	03
	LDX #$0200	00	00	0200	EFFD	03
BACK2	PULA	04	00	0200	EFFE	03
	INCB	04	01	0200	EFFE	03
	STAA ,x	04	01	0200	EFFE	03
	INX	04	01	0201	EFFE	03
	CMPB $000B	04	01	0201	EFFE	03
	BNE WRITE	04	01	0201	EFFE	03
	.end					

MAIN PROGRAM		*A*	*B*	*IX*	*SP*	*$000B*
START	LDX #$1000					
	CLRB					
BACK	LDAA ,x					
	CMPA $000F					
	BHI FORWARD					
	PSHA					
	INC $000B					
FORWARD	INCB					
	CMPB $000A					
	BEQ STOP					
	INX					
	BRA BACK					
STOP	CLRB					
	LDX #$0200					
BACK2	PULA	02	01	0201	EFFF	03
	INCB	02	02	0201	EFFF	03
	STAA ,x	02	02	0201	EFFF	03
	INX	02	02	0202	EFFF	03
	CMPB $000B	02	02	0202	EFFF	03
	BNE BACK2	02	02	0202	EFFF	03
	.end					

```
            .ORG $0010
START       LDX #$0100   ;starting address of ARY1
            CLRB
BACK        LDAA ,x      ; load elements of ARY1 to A
            CMPA $000F   ; compare with the value in $000F
            BHI FORWARD  ;if higher branch
            PSHA         ; for adding ARY2, push STACK
            INC $000B    ;increase size of ARY2
FORWARD     INCB         ; end of ARY1 or not?
            CMPB $000A
            BEQ WRITE
            INX
            BRA BACK
WRITE       CLRB
            LDX #$0200   ;starting address of ARY2
BACK2       PULA         ; pull elements of ARY2
            INCB
            STAA ,x      ; store elements of ARY2
            INX
            CMPB $000B   ; end of ARY2 or not?
            BNE BACK2
```

Memory Address	Contents
$EFFE	$04
$EFFF	$02
$F000	$03

PROBLEM 2

- There is an array (ARY1) consisting of unsigned, 8-bit integers in the memory. Its starting address is the content of $000A-$000B and its size is the content of $000C memory addresses. There are same number of even integers and odd integers in the ARY1
- A new array (ARY2) is created according to

→ First odd integer in ARY1 is the nTH member of ARY2. First event integer in ARY1 is the (n-1)th member of ARY2

→Second odd integer in ARY1 is (n-2)thmember of ARY2. Second event integer in ARY1 is the (n-3)th member of ARY2

→ So on..

- **STACK**

ARY1
E1
O1
O2
O3
E2
O4
E3
E4

ARY2
E4
O4
E3
O3
E2
O2
E1
O1

SOLUTION- Initial values

```
BASE   .equ $000A   ; Starting address of ARY 1
NEW    .equ $000D   ; Starting address of ARY2
LENGTH .equ $000C   ; Length of ARY1
ODD    .equ $0006   ; Keep IX for odd integers
EVEN   .equ $0008   ;Keep IX for even integers
TEMP   .rmb 1       ; Temporary variable
```

Memory address	Contents
$000A	$12
$000B	$00
$000C	$13
$000D	$00

ARY1	Content
$1200	$32
$1201	$33
$1202	$22
$1203	$87

ARY2	Content
$1300	$22
$1301	$87
$1302	$32
$1303	$33

Main Program		A	B	IX	SP	ODD	EVEN
START	LDX BASE	00	00	1200	F000	0000	0000
	STX ODD	00	00	1200	F000	1200	0000
	STX EVEN	00	00	1200	F000	1200	1200
	CLRB	00	00	1200	F000	1200	1200
BACK	LDAA ,x	32	00	1200	F000	1200	1200
	STAA TEMP	32	00	1200	F000	1200	1200
	ANDA #$01	00	00	1200	F000	1200	1200
	BNE ODDBRA	00	00	1200	F000	1200	1200
	INX	00	00	1201	F000	1200	1200
	BRA BACK	00	00	1201	F000	1200	1200
BACK2	LDAA ,x						
	STAA TEMP						
	ANDA #$01						
	BEQ EVENBRA						
	INX						
	BRA BACK2						

Main Program		A	B	IX	SP	ODD	EVEN
START	LDX BASE						
	STX ODD						
	STX EVEN						
	CLRB						
	LDX ODD						
BACK	LDAA ,x	33	00	1201	F000	1200	1200
	STAA TEMP	33	00	1201	F000	1200	1200
	ANDA #$01	01	00	1201	F000	1200	1200
	BNE ODDBRA	01	00	1201	F000	1200	1200
	INX						
	BRA BACK						
	LDX EVEN						
BACK2	LDAA ,x						
	STAA TEMP						
	ANDA #$01						
	BEQ EVENBRA						
	INX						
	BRA BACK2						

Main Program		A	B	IX	SP	ODD	EVEN
ODDBRA	LDAA TEMP	33	00	1201	F000	1200	1200
	PSHA	33	00	1201	EFFF	1200	1200
	INX	33	00	1202	EFFF	1200	1200
	STX ODD	33	00	1202	EFFF	1202	1200
	INCB	33	01	1202	EFFF	1202	1200
	CMPB LENGTH	33	01	1202	EFFF	1202	1200
	BEQ STOP	33	01	1202	EFFF	1202	1200
	LDX EVEN	33	01	1200	EFFF	1202	1200
	BRA BACK2	33	01	1200	EFFF	1202	1200
EVENBRA	LDAA TEMP						
	PSHA						
	INX						
	STX EVEN						
	INCB						
	CMPB LENGTH						
	BEQ STOP						
	LDX ODD						
	BRA BACK						

Memory Address	Content
$F000	33

Main Program	A	B	IX	SP	ODD	EVEN
START LDX BASE						
STX ODD						
STX EVEN						
CLRB						
LDX ODD						
BACK LDAA ,x						
STAA TEMP						
ANDA #$01						
BNE ODDBRA						
INX						
BRA BACK						
LDX EVEN						
BACK2 LDAA ,x	32	01	1200	EFFF	1202	1200
STAA TEMP	32	01	1200	EFFF	1202	1200
ANDA #$01	00	01	1200	EFFF	1202	1200
BEQ EVENBRA	00	01	1200	EFFF	1202	1200
INX						
BRA BACK2						

Main Program		A	B	IX	SP	ODD	EVEN
ODDBRA	LDAA TEMP						
	PSHA						
	INX						
	STX ODD						
	INCB						
	CMPB LENGTH						
	BEQ STOP						
	LDX EVEN						
	BRA BACK2						
EVENBRA	LDAA TEMP	32	01	1200	EFFF	1202	1200
	PSHA	32	01	1200	EFFE	1202	1200
	INX	32	01	1201	EFFE	1202	1200
	STX EVEN	32	01	1201	EFFE	1202	1201
	INCB	32	01	1201	EFFE	1202	1201
	CMPB LENGTH	32	01	1201	EFFE	1202	1201
	BEQ STOP	32	01	1201	EFFE	1202	1201
	LDX ODD	32	01	1202	EFFE	1202	1201
	BRA BACK	32	01	1201	EFFE	1202	1201

Memory Address	Content
$EFFF	32
$F000	33

Main Program		A	B	IX	SP	ODD	EVEN
STOP	CLRB	00	04	1203	EFFC	1204	1203
	LDX NEW	00	00	1300	EFFC	1204	1203
BACK3	PULA	22	00	1300	EFFD	1204	1203
	STAA ,x	22	00	1300	EFFD	1204	1203
	INX	22	00	1301	EFFD	1204	1203
	INCB	22	01	1301	EFFD	1204	1203
	CMPB LENGTH	22	01	1301	EFFD	1204	1203
	BNE BACK3	22	01	1301	EFFD	1204	1203
	.end						

Stack	Content
$EFFD	$22
$EFFE	$87
$EFFF	$32
$F000	$33

Start of ARY2	Content
$1300	$22
$1301	$87
$1302	$32
$1303	$33

PROBLEM 3

- VAR 1is in $0000-$0001 memory addresses. VAR2 is in $0002-$0003 memory addresses

- VAR 1 and VAR2 is 16 bit integers that is in two complements arithmetic

- If the content in $0004 is higher than $A4, sum of VAR1 and VAR 2 is stored in *$000B-$000D* . Otherwise VAR2 is subtracted from VAR1 and result is stored in *$000B-$000D*

PROBLEM 3

- Main program checks the content of $0004 and branch the SUM subprogram or SUB subprogram
- Main program sends VAR1, VAR2 and the starting address that contains result (*$000B*) to STACK.
- Two subprograms

SUM: Sum up two 16 bit integers.

SUB: Substract VAR2 from VAR 1.

If carry flag set after operations, $000D (result address) must updated.

```
        .ORG $0000
VALUE .byte $83,$14,$84,$12,$0B,$00
CHECK .byte $A6
        .ORG $0010
        LDX #VALUE
        LDAA 0,x
        LDAB 1,x
        PSHA
        PSHB
        LDAA 2,x
        LDAB 3,x
        PSHA
        PSHB
        LDAA 4,x
        LDAB 5,x
        PSHA
        PSHB
        LDAA CHECK
        CMPA #$A4
        BLT SUBST
        BSR SUM
SUBST   BSR SUB
```

Memory Address	Content
$0000 (VAR1 high)	$83
$0001 (VAR1 low)	$14
$0002 (VAR2 high)	$84
$0003 (VAR2 low)	$12
$0004 (Result high)	$00
$0005 (Result low)	$0B

Main program	*A*	*B*	*SP*
.ORG $0010			
LDX #VALUE	00	00	F000
LDAA 0,x	83	00	F000
LDAB 1,x	83	14	F000
PSHA	83	14	EFFF
PSHB	83	14	EFFE
LDAA 2,x	84	14	EFFE
LDAB 3,x	84	12	EFFE
PSHA	84	12	EFFD
PSHB	84	12	EFFC
LDAA 4,x	00	12	EFFC
LDAB 5,x	00	0B	EFFC
PSHA	00	0B	EFFB
PSHB	00	0B	EFFA
LDAA CHECK	A6	0B	EFFA
CMPA #$A4	A6	0B	EFFA
BLT SUBST	A6	0B	EFFA
BSR SUM	A6	0B	EFFA

SUBST BSR SUB

SP	Memory contents
EFFB	00
EFFC	0B
EFFD	12
EFFE	84
EFFF	14
F000	83

```
SUM     CLC
        TSX
        LDAA 5,x
        LDAB 7,x
        STAA $4000
        ADCB $4000
        BCS SETC
BACK    STAB 0,x
        TSX
        LDAA 4,x
        LDAB 6,x
        ABA
        LDX 2,x
        STAA 1,x
        RTS
SETC    LDAA #$01
        LDX 2,x
        STAA 2,x
        BRA BACK
```

Memory Address	Content	Explain
EFF9	00	Sub Prog. return address high
EFFA	2D	Sub Prog. return address low
EFFB	00	Result high
EFFC	0B	Result low
EFFD	12	VAR2 low
EFFE	84	VAR2 high
EFFE	14	VAR1 low
F000	83	VAR1 high

```
SUM     CLC
        TSX
        LDAA 5,x
        LDAB 7,x
        STAA $4000
        ADCB $4000
        BCS SETC
BACK    STAB 0,x
        TSX
        LDAA 4,x
        LDAB 6,x
        ABA
        LDX 2,x
        STAA 1,x
        RTS
SETC    LDAA #$01
        LDX 2,x
        STAA 2,x
        BRA BACK
```

IX	SP
EFF9	EFF8

Memory Address	Content	Explain
EFF9	00	Sub Prog. return address high
EFFA	2D	Sub Prog. return address low
EFFB	00	Result high
EFFC	0B	Result low
EFFD	12	VAR2 low
EFFE	84	VAR2 high
EFFE	14	VAR1 low
F000	83	VAR1 high

```
                          A     B     IX      COMMENT
SUM     CLC                                   ;clear carry
        TSX                             EFF9  ;transfer SP to IX
        LDAA 5,x          84    00
        LDAB 7,x          84    83
        STAA $4000
        ADCB $4000                            ;high bits, add with carry
        BCS SETC
BACK    STAB 0,x                              ;stored $000B HIGH
        TSX                             EFF9   ;transfer SP to IX
        LDAA 4,x          14    83
        LDAB 6,x          14    12
        ABA                                   ; low bits, add A and B
        LDX 2,x                         000B  ;address contains result
        STAA 1,x                              ; stored $000C LOW
        RTS
SETC    LDAA #$01
        LDX 2,x                         000B  ; contents of  $EFFB-$EFFC
        STAA 2,x                              ;stored $000D,if carry set
        BRA BACK
```

```
SUM     CLC
        TSX
        LDAA 5,x
        LDAB 7,x
        STAA $4000
        ADCB $4000
        BCS SETC
BACK    STAB 0,x
        TSX
        LDAA 4,x
        LDAB 6,x
        ABA
        LDX 2,x
        STAA 1,x
        RTS
SETC    LDAA #$01
        LDX 2,x
        STAA 2,x
        BRA BACK
```

IX	SP
EFF9	EFF8

IX (EFF9)	Content	Explain
IX+0	00	Sub Prog. return address high
IX+1	2D	Sub Prog. return address low
IX+2	00	Result high
IX+3	0B	Result low
IX+4	12	VAR2 low
IX+5	84	VAR2 high
IX+6	14	VAR1 low
IX+7	83	VAR1 high

Main program	*A*	*B*	*SP*
.ORG $0010			
LDX #VALUE	00	00	F000
LDAA 0,x	83	00	F000
LDAB 1,x	83	14	F000
PSHA	83	14	EFFF
PSHB	83	14	EFFE
LDAA 2,x	84	14	EFFE
LDAB 3,x	84	12	EFFE
PSHA	84	12	EFFD
PSHB	84	12	EFFC
LDAA 4,x	00	12	EFFC
LDAB 5,x	00	04	EFFC
PSHA	00	04	EFFB
PSHB	00	04	EFFA
LDAA CHECK	A3	04	EFFA
CMPA #$A4	A3	04	EFFA
BLT SUBST	A3	04	EFFA
BSR SUM			
SUBST BSR SUB	A3	04	EFFA

SP	Memory contents
EFFB	04
EFFC	00
EFFD	12
EFFE	83
EFFF	14
F000	83

```
SUB     CLC                 ;clear carry
        TSX                 ; transfer SP to IX
        LDAA 5,x
        LDAB 7,x
        LDX 2,x             ; load $000B to IX
        STAB $4000
        SBCA $4000          ;substract with carry VAR1-VAR2
        BCS SETC2
BACK2   STAB 0,x            ;High level bits to $000B
        TSX                 ;transfer SP to IX
        LDAA 6,x
        LDAB 4,x
        SBA                 ; substract VAR1-VAR2
        LDX 2,x             ; IX has $000B
        STAA 1,x            ; Low level bits is stored in $000C
        RTS
SETC2   LDAB #$FE           ;if carry flag set BORROW
        STAB 2,x            ;set $FE to 3th part of result ($000D address)
        BRA BACK2
STOP    .end
```

QUESTIONS?