**Εργαστηριακή Άσκηση 1 (PART\_1 & PART\_2 are separated)**

.CSEG

.org 0x0000

.def ZH=R31

.def ZL=R30

**/\*\*\* PART\_1 \*\*\*/**

RJMP RESET

RESET:

LDI R18, 0x00

OUT 0x11, R18 ; Port D (SWITCHES) becomes an input port [0x11 is equivalent to DDRD]

LDI R18, 0xFF

OUT 0x17, R18 ; Port B (LEDS) becomes an output port [0x17 is equivalent to DDRB]

CLT

AEMs\_COMP:

LDI ZL, LOW(2\*AEMs\_table)

LDI ZH, HIGH(2\*AEMs\_table)

COMP:

LPM R16, Z

SUBI ZL, -4

LPM R17, Z

SUBI ZL, 3

CP R16, R17

BRLO PART\_1\_3 ; If First\_AEM<Second\_AEM the programme moves on to the 3rd part of the exercise

BREQ COMP ; If the present digits of AEMs are equal we load the next two digits and compare them

; If First\_AEM>Second\_AEM the appropriate LEDS turn on

; thereafter, we form the number that consists of the last two digits of the First\_AEM

LDI ZL, LOW((2\*AEMs\_table)+2)

LDI ZH, HIGH((2\*AEMs\_table)+2)

LPM R16, Z+ ; we load to R16 the 3rd digit of First\_AEM from PM

SUBI R16, 0x30

SWAP R16

LPM R17, Z ; we load to R17 the 4th digit of First\_AEM from PM

SUBI R17, 0x30

ADD R16, R17

COM R16

OUT 0x18, R16 ; We copy the 3rd and 4th digits of First\_AEM (if it's greater than Second\_AEM) to Port B (LEDS)

; [0x18 is equivalent to PORTB]

SET ; If T=1 then First\_AEM>Second\_AEM, else if T=0 then First\_AEM<Second\_AEM

; delay subroutine

.def inner\_count\_L=R24

.def inner\_count\_H=R25

.def outer\_count\_L=R26 ; (XL)

.def outer\_count\_H=R27 ; (XH)

;----------

TIME\_DEL\_A1:

ldi outer\_count\_L, 0x8E ; 1 cycle

ldi outer\_count\_H, 0x08 ; 1 cycle

outer\_loop1:

ldi inner\_count\_L, 0x8E ; 1 cycle

ldi inner\_count\_H, 0x08 ; 1 cycle

inner\_loop1:

nop ; 1 cycle, possible use of more "nop" instructions for longer delays

sbiw inner\_count\_L, 1 ; 2 cycles

brne inner\_loop1 ; 2 cycles if true (Ni-1 times), 1 cycle if false

sbiw outer\_count\_L, 1 ; 2 cycles

brne outer\_loop1 ; 2 cycles if true (No-1 times), 1 cycle if false

ret ; 4 cycles (plus 3 cycles for rcall)

;----- End of subroutine TIME\_DEL\_A -----

.undef inner\_count\_L

.undef inner\_count\_H

.undef outer\_count\_L

.undef outer\_count\_H

; end of delay subroutine

PART\_1\_3:

LDI R18, 0xFF

OUT 0x18, R18 ; we turn off all the LEDS in output

BRTS First\_AEM\_Greater

Second\_AEM\_Greater:

LDI ZL, LOW((2\*AEMs\_table)+7)

LDI ZH, HIGH((2\*AEMs\_table)+7)

LPM R17, Z

SUBI ZL, 4

LPM R16, Z

RJMP ODD\_OR\_EVEN

First\_AEM\_Greater:

LDI ZL, LOW((2\*AEMs\_table)+3)

LDI ZH, HIGH((2\*AEMs\_table)+3)

LPM R17, Z

SUBI ZL, -4

LPM R16, Z

RJMP ODD\_OR\_EVEN

ODD\_OR\_EVEN:

SBRC R17, 0

CBI 0x18, 1 ; we clear bit 1 of POTRTB if the greater AEM is odd

SBRC R16, 0

CBI 0x18, 0 ; we clear bit 0 of POTRTB if the smaller AEM is odd

.UNDEF ZH

.UNDEF ZL

FAR\_END:

RJMP FAR\_END

AEMs\_table:

.DB 0x37, 0x35, 0x30, 0x37 ; First\_AEM (7507)

.DB 0x37, 0x35, 0x34, 0x39 ; Second\_AEM (7549)

.CSEG

.org 0x0000

.def ZH=R31

.def ZL=R30

**/\*\*\* PART\_2 \*\*\*/**

RJMP RESET

RESET:

LDI R18, 0x00

OUT 0x11, R18 ; Port D (SWITCHES) becomes an input port [0x11 is equivalent to DDRD]

LDI R18, 0xFF

OUT 0x17, R18 ; Port B (LEDS) becomes an output port [0x17 is equivalent to DDRB]

CLT

AEMs\_COMP:

LDI ZL, LOW(2\*AEMs\_table)

LDI ZH, HIGH(2\*AEMs\_table)

COMP:

LPM R16, Z

SUBI ZL, -4

LPM R17, Z

SUBI ZL, 3

CP R16, R17

BRLO PART\_1\_3 ; If First\_AEM<Second\_AEM the programme moves on to the 3rd part of the exercise

BREQ COMP ; If the present digits of AEMs are equal we load the next two digits and compare them

SET ; If T=1 then First\_AEM>Second\_AEM, else if T=0 then First\_AEM<Second\_AEM

PART\_1\_3:

LDI R18, 0xFF

OUT 0x18, R18 ; we turn off all the LEDS in output

BRTS First\_AEM\_Greater

Second\_AEM\_Greater:

LDI ZL, LOW((2\*AEMs\_table)+7)

LDI ZH, HIGH((2\*AEMs\_table)+7)

LPM R17, Z

SUBI ZL, 4

LPM R16, Z

RJMP ODD\_OR\_EVEN

First\_AEM\_Greater:

LDI ZL, LOW((2\*AEMs\_table)+3)

LDI ZH, HIGH((2\*AEMs\_table)+3)

LPM R17, Z

SUBI ZL, -4

LPM R16, Z

RJMP ODD\_OR\_EVEN

ODD\_OR\_EVEN:

LDI R21, 0xFF

SBRC R17, 0

SUBI R21, 2

SBRC R16, 0

SUBI R21, 1

LDI R16, 0xFF

OUT 0x10, R16

LOOP:

IN R16, 0x10 ; [0x10 is equivalent to PIND]

CPI R16, 0xFF ; none of the SWITCHES is pressed

BREQ LOOP

IN R16, 0x10

CPI R16, 0xEF ; SWITCH\_4 is pressed

BREQ LOOP

IN R16, 0x10

CPI R16, 0xDF ; SWITCH\_5 is pressed

BREQ LOOP

IN R16, 0x10

CPI R16, 0xBF ; SWITCH\_6 is pressed

BREQ LOOP

IN R16, 0x10 ; [0x10 is equivalent to PIND]

CPI R16, 0xFF ; none of the SWITCHES is pressed

BREQ LOOP

LOOP2:

IN R18, 0x10

CPI R18, 0xFF

BRNE LOOP2

SWITCH\_PRESSED:

CPI R16, 0xFE

BREQ SW0\_PRESSED

CPI R16, 0xFD

BREQ SW1\_PRESSED

CPI R16, 0xFB

BREQ SW2\_PRESSED

CPI R16, 0xF7

BREQ SW3\_PRESSED

CPI R16, 0x7F

BREQ SW7\_PRESSED

SW7\_PRESSED:

OUT 0x18, R21

RJMP FAR\_END

SW0\_PRESSED:

BRTS First\_AEM\_bigger\_SW0

Second\_AEM\_bigger\_SW0:

LDI ZL, LOW((2\*AEMs\_table)+4)

LDI ZH, HIGH((2\*AEMs\_table)+4)

LPM R19, Z+

SUBI R19, 0x30

SWAP R19

LPM R20, Z

SUBI R20, 0x30

ADD R19, R20

RJMP END

First\_AEM\_bigger\_SW0:

LDI ZL, LOW(2\*AEMs\_Table)

LDI ZH, HIGH(2\*AEMs\_Table)

LPM R19, Z+

SUBI R19, 0x30

SWAP R19

LPM R20, Z

SUBI R20, 0x30

ADD R19, R20

RJMP END

SW1\_PRESSED:

BRTS First\_AEM\_bigger\_SW1

Second\_AEM\_bigger\_SW1:

LDI ZL, LOW((2\*AEMs\_table)+6)

LDI ZH, HIGH((2\*AEMs\_table)+6)

LPM R19, Z+

SUBI R19, 0x30

SWAP R19

LPM R20, Z

SUBI R20, 0x30

ADD R19, R20

RJMP END

First\_AEM\_bigger\_SW1:

LDI ZL, LOW((2\*AEMs\_table)+2)

LDI ZH, HIGH((2\*AEMs\_table)+2)

LPM R19, Z+

SUBI R19, 0x30

SWAP R19

LPM R20, Z

SUBI R20, 0x30

ADD R19, R20

RJMP END

SW2\_PRESSED:

BRTC First\_AEM\_smaller\_SW2

Second\_AEM\_smaller\_SW2:

LDI ZL, LOW((2\*AEMs\_table)+4)

LDI ZH, HIGH((2\*AEMs\_table)+4)

LPM R19, Z+

SUBI R19, 0x30

SWAP R19

LPM R20, Z

SUBI R20, 0x30

ADD R19, R20

RJMP END

First\_AEM\_smaller\_SW2:

LDI ZL, LOW(2\*AEMs\_Table)

LDI ZH, HIGH(2\*AEMs\_Table)

LPM R19, Z+

SUBI R19, 0x30

SWAP R19

LPM R20, Z

SUBI R20, 0x30

ADD R19, R20

RJMP END

SW3\_PRESSED:

BRTC First\_AEM\_smaller\_SW3

Second\_AEM\_smaller\_SW3:

LDI ZL, LOW((2\*AEMs\_table)+6)

LDI ZH, HIGH((2\*AEMs\_table)+6)

LPM R19, Z+

SUBI R19, 0x30

SWAP R19

LPM R20, Z

SUBI R20, 0x30

ADD R19, R20

RJMP END

First\_AEM\_smaller\_SW3:

LDI ZL, LOW((2\*AEMs\_table)+2)

LDI ZH, HIGH((2\*AEMs\_table)+2)

LPM R19, Z+

SUBI R19, 0x30

SWAP R19

LPM R20, Z

SUBI R20, 0x30

ADD R19, R20

RJMP END

END:

COM R19

OUT 0x18, R19

; delay subroutine

.def inner\_count\_L=R24

.def inner\_count\_H=R25

.def outer\_count\_L=R26 ; (XL)

.def outer\_count\_H=R27 ; (XH)

;----------

TIME\_DEL\_A:

ldi outer\_count\_L, 0x8E ; 1 cycle

ldi outer\_count\_H, 0x08 ; 1 cycle

outer\_loop:

ldi inner\_count\_L, 0x8E ; 1 cycle

ldi inner\_count\_H, 0x08 ; 1 cycle

inner\_loop:

nop ; 1 cycle, possible use of more "nop" instructions for longer delays

sbiw inner\_count\_L, 1 ; 2 cycles

brne inner\_loop ; 2 cycles if true (Ni-1 times), 1 cycle if false

sbiw outer\_count\_L, 1 ; 2 cycles

brne outer\_loop ; 2 cycles if true (No-1 times), 1 cycle if false

ret ; 4 cycles (plus 3 cycles for rcall)

;----- End of subroutine TIME\_DEL\_A -----

.undef inner\_count\_L

.undef inner\_count\_H

.undef outer\_count\_L

.undef outer\_count\_H

; end of delay subroutine

LDI R21, 0xFF

OUT 0x18, R21

.UNDEF ZH

.UNDEF ZL

RJMP FAR\_END

FAR\_END:

RJMP FAR\_END

AEMs\_table:

.DB 0x37, 0x35, 0x30, 0x37 ; First\_AEM (7507)

.DB 0x37, 0x35, 0x34, 0x39 ; Second\_AEM (7549)

**Εργαστηριακή Άσκηση 2 (all parts together)**

.include "m16def.inc"

RJMP RESET

.CSEG

.DEF YL=R28

.DEF YH=R29

RESET:

LDI R18, LOW(RAMEND) ; Stack pointer points at the end of SRAM

OUT SPL, R18

LDI R18, HIGH(RAMEND)

OUT SPH, R18

LDI R18, 0x00

OUT 0x11, R18 ; Port D (SWITCHES) becomes an input port [0x11 is equivalent to DDRD]

LDI R18, 0xFF

OUT 0x17, R18 ; Port B (LEDS) becomes an output port [0x17 is equivalent to DDRB

LDI R18, 0xFF

OUT 0x18, R18

CLR R0

CLR R1

CLR R2

CLR R3

CLR R4

LDI YL, 0x00

LDI YH, 0x01

LDI R16, 0x07 ; Insert both AEMs (BCD coded) in SRAM [SRAM begins from address 0x0100]

ST Y+, R16

LDI R16, 0x05

ST Y+, R16

LDI R16, 0x00

ST Y+, R16

LDI R16, 0x07

ST Y+, R16

LDI R16, 0x07

ST Y+, R16

LDI R16, 0x05

ST Y+, R16

LDI R16, 0x04

ST Y+, R16

LDI R16, 0x09

ST Y, R16

CLR R15 ; R15 is used to keep the carry

CALL ADD\_AEMs

MOV R4, R16

CALL ADD\_AEMs

MOV R3, R16

CALL ADD\_AEMs

MOV R2, R16

CALL ADD\_AEMs

MOV R1, R16

MOV R0, R15

RJMP Cont2

ADD\_AEMs:

LD R16, Y

SUBI YL, 4

LD R17, Y

ADD R16, R17

ADD R16, R15

SBRC R15, 0

DEC R15

CPI R16, 0x0A

BRLO Cont

SUBI R16, 0x0A

INC R15

Cont:

SUBI YL, -3

RET

Cont2:

LDI YL, 0x08

LDI YH, 0x01

ST Y+, R0

ST Y+, R1

ST Y+, R2

ST Y+, R3

ST Y, R4

/\*\* PART 2 \*\*/

LDI R21, 0xFF

OUT 0x10, R21

LOOP:

IN R16, 0x10 ; [0x10 is equivalent to PIND]

CPI R16, 0xFF ; none of the SWITCHES is pressed

BREQ LOOP

IN R16, 0x10

CPI R16, 0xFE ; SWITCH\_0 is pressed

BREQ LOOP

IN R16, 0x10

CPI R16, 0xFB ; SWITCH\_2 is pressed

BREQ LOOP

IN R16, 0x10

CPI R16, 0xF7 ; SWITCH\_3 is pressed

BREQ LOOP

IN R16, 0x10

CPI R16, 0xEF ; SWITCH\_4 is pressed

BREQ LOOP

IN R16, 0x10

CPI R16, 0xDF ; SWITCH\_5 is pressed

BREQ LOOP

IN R16, 0x10

CPI R16, 0xBF ; SWITCH\_6 is pressed

BREQ LOOP

IN R16, 0x10

CPI R16, 0x7F ; SWITCH\_7 is pressed

BREQ LOOP

IN R16, 0x10 ; [0x10 is equivalent to PIND]

CPI R16, 0xFF ; none of the SWITCHES is pressed

BREQ LOOP

LOOP2:

IN R18, 0x10

CPI R18, 0xFF

BRNE LOOP2

SWITCH1\_PRESSED:

LDI YL, 0x04

LDI R22, 4

Display\_next\_digit1:

LD R20, -Y

SWAP R20

COM R20

OUT 0x18, R20

RCALL TIME\_DEL\_A

DEC R22

BRNE Display\_next\_digit1

LDI R18, 0xFF

OUT 0x18, R18

LOOP3:

IN R16, 0x10 ; [0x10 is equivalent to PIND]

CPI R16, 0xFF ; none of the SWITCHES is pressed

BREQ LOOP3

IN R16, 0x10

CPI R16, 0xFE ; SWITCH\_0 is pressed

BREQ LOOP3

IN R16, 0x10

CPI R16, 0xFD ; SWITCH\_1 is pressed

BREQ LOOP3

IN R16, 0x10

CPI R16, 0xF7 ; SWITCH\_3 is pressed

BREQ LOOP3

IN R16, 0x10

CPI R16, 0xEF ; SWITCH\_4 is pressed

BREQ LOOP3

IN R16, 0x10

CPI R16, 0xDF ; SWITCH\_5 is pressed

BREQ LOOP3

IN R16, 0x10

CPI R16, 0xBF ; SWITCH\_6 is pressed

BREQ LOOP3

IN R16, 0x10

CPI R16, 0x7F ; SWITCH\_7 is pressed

BREQ LOOP3

IN R16, 0x10 ; [0x10 is equivalent to PIND]

CPI R16, 0xFF ; none of the SWITCHES is pressed

BREQ LOOP3

LOOP4:

IN R18, 0x10

CPI R18, 0xFF

BRNE LOOP4

SWITCH2\_PRESSED:

LDI YL, 0x08

LDI R22, 4

Display\_next\_digit2:

LD R20, -Y

SWAP R20

COM R20

OUT 0x18, R20

RCALL TIME\_DEL\_A

DEC R22

BRNE Display\_next\_digit2

LDI R18, 0xFF

OUT 0x18, R18

LDI R22, 5

LDI YL, 0x0D

LOOP5:

IN R16, 0x10 ; [0x10 is equivalent to PIND]

CPI R16, 0xFF ; none of the SWITCHES is pressed

BREQ LOOP5

IN R16, 0x10

CPI R16, 0xF7 ; SWITCH\_3 is pressed

BREQ LOOP6

IN R16, 0x10

CPI R16, 0xFE ; SWITCH\_0 is pressed

BREQ LOOP5

IN R16, 0x10

CPI R16, 0xFD ; SWITCH\_1 is pressed

BREQ LOOP5

IN R16, 0x10

CPI R16, 0xFB ; SWITCH\_2 is pressed

BREQ LOOP5

IN R16, 0x10

CPI R16, 0xEF ; SWITCH\_4 is pressed

BREQ LOOP5

IN R16, 0x10

CPI R16, 0xDF ; SWITCH\_5 is pressed

BREQ LOOP5

IN R16, 0x10

CPI R16, 0xBF ; SWITCH\_6 is pressed

BREQ LOOP5

IN R16, 0x10

CPI R16, 0x7F ; SWITCH\_7 is pressed

BREQ LOOP5

IN R16, 0x10 ; [0x10 is equivalent to PIND]

CPI R16, 0xFF ; none of the SWITCHES is pressed

BREQ LOOP5

LOOP6:

IN R18, 0x10

CPI R18, 0xFF

BRNE LOOP6

SWITCH3\_PRESSED:

LD R20, -Y

SWAP R20

COM R20

OUT 0x18, R20

RCALL TIME\_DEL\_B

DEC R22

BRNE LOOP5

RJMP FAR\_END

; delay subroutine

TIME\_DEL\_A:

.def inner\_count\_L=R24

.def inner\_count\_H=R25

.def outer\_count\_L=R26 ; (XL)

.def outer\_count\_H=R27 ; (XH)

;----------

ldi outer\_count\_L, 0xFC ;1 cycle

ldi outer\_count\_H, 0x06; 1 cycle

outer\_loop:

ldi inner\_count\_L, 0xFC ;1 cycle

ldi inner\_count\_H, 0x06; 1 cycle

inner\_loop:

nop ; 1 cycle, possible use of more "nop" instructions for longer delays

sbiw inner\_count\_L, 1 ; 2 cycles

brne inner\_loop ; 2 cycles if true (Ni-1 times), 1 cycle if false

sbiw outer\_count\_L, 1 ; 2 cycles

brne outer\_loop ; 2 cycles if true (No-1 times), 1 cycle if false

;----- End of subroutine TIME\_DEL\_A -----

.undef inner\_count\_L

.undef inner\_count\_H

.undef outer\_count\_L

.undef outer\_count\_H

RET

; end of delay subroutine

; delay subroutine

TIME\_DEL\_B:

.def inner\_count\_L=R24

.def inner\_count\_H=R25

.def outer\_count\_L=R26 ; (XL)

.def outer\_count\_H=R27 ; (XH)

;----------

ldi outer\_count\_L, 0xAC ;1 cycle

ldi outer\_count\_H, 0x00; 1 cycle

outer\_loop\_b:

ldi inner\_count\_L, 0xAC ;1 cycle

ldi inner\_count\_H, 0x00; 1 cycle

inner\_loop\_b:

nop ; 1 cycle, possible use of more "nop" instructions for longer delays

sbiw inner\_count\_L, 1 ; 2 cycles

brne inner\_loop\_b ; 2 cycles if true (Ni-1 times), 1 cycle if false

sbiw outer\_count\_L, 1 ; 2 cycles

brne outer\_loop\_b ; 2 cycles if true (No-1 times), 1 cycle if false

;----- End of subroutine TIME\_DEL\_A -----

.undef inner\_count\_L

.undef inner\_count\_H

.undef outer\_count\_L

.undef outer\_count\_H

RET

; end of delay subroutine

FAR\_END:

RJMP FAR\_END

**Εργαστηριακή Άσκηση 3**

.include "m16def.inc"

RJMP RESET

.CSEG

.DEF YL=R28

.DEF YH=R29

.DEF MASK=R21

.DEF COUNTER=R22

RESET:

LDI R18, LOW(RAMEND) ; Stack pointer points at the end of SRAM

OUT SPL, R18

LDI R18, HIGH(RAMEND)

OUT SPH, R18

LDI R18, 0x00

OUT 0x11, R18 ; Port D (SWITCHES) becomes an input port [0x11 is equivalent to DDRD]

LDI R18, 0xFF

OUT 0x17, R18 ; Port B (LEDS) becomes an output port [0x17 is equivalent to DDRB]

OUT 0x10, R18

LDI R18, 0x7E

OUT 0x18, R18

LDI R20, 0xFF

LOOP:

IN R16, 0x10 ; [0x10 is equivalent to PIND]

CPI R16, 0xFF ; none of the SWITCHES is pressed

BREQ LOOP

IN R16, 0x10

CPI R16, 0xFE ; SWITCH\_0 is pressed

BREQ LOOP

IN R16, 0x10

CPI R16, 0xFD ; SWITCH\_1 is pressed

BREQ LOOP

IN R16, 0x10

CPI R16, 0xFB ; SWITCH\_2 is pressed

BREQ LOOP2

IN R16, 0x10

CPI R16, 0xF7 ; SWITCH\_3 is pressed

BREQ LOOP2

IN R16, 0x10

CPI R16, 0xEF ; SWITCH\_4 is pressed

BREQ LOOP2

IN R16, 0x10

CPI R16, 0xDF ; SWITCH\_5 is pressed

BREQ LOOP2

IN R16, 0x10

CPI R16, 0xBF ; SWITCH\_6 is pressed

BREQ LOOP2

IN R16, 0x10

CPI R16, 0x7F ; SWITCH\_7 is pressed

BREQ LOOP

IN R16, 0x10 ; [0x10 is equivalent to PIND]

CPI R16, 0xFF ; none of the SWITCHES is pressed

BREQ LOOP

LOOP2:

IN R18, 0x10

CPI R18, 0xFF

BRNE LOOP2

SBRS R16, 2

AND R20, R16

SBRS R16, 3

AND R20, R16

SBRS R16, 4

AND R20, R16

SBRS R16, 5

AND R20, R16

SBRS R16, 6

RJMP CONT\_1

RJMP LOOP

CONT\_1:

SBRC R20, 2

RJMP CONT\_1\_1

LDI R18, 0x78

OUT 0x18, R18

LDI COUNTER, 0x04

CONT\_1\_0:

RCALL DELAY\_1sec

DEC COUNTER

BRNE CONT\_1\_0

CONT\_1\_1:

LDI MASK, 0x18

AND MASK, R20

CPI MASK, 0x00

BREQ SECS8

CPI MASK, 0x08

BREQ SECS12

CPI MASK, 0x10

BREQ SECS24

CPI MASK, 0x18

BREQ SECS30

SECS8:

LDI COUNTER, 0x08

RJMP CONT\_2

SECS12:

LDI COUNTER, 0x0C

RJMP CONT\_2

SECS24:

LDI COUNTER, 0x18

RJMP CONT\_2

SECS30:

LDI COUNTER, 0x1E

RJMP CONT\_2

CONT\_2:

LDI R18, 0x74

OUT 0x18, R18

RCALL DELAY\_1sec

DEC COUNTER

BRNE CONT\_2

LDI R18, 0x6C

OUT 0x18, R18

RCALL DELAY\_1sec

SBRC R20, 5

RJMP CONT\_3

LDI R18, 0x5C

OUT 0x18, R18

LDI COUNTER, 0x02

CONT\_3\_0:

RCALL DELAY\_1sec

DEC COUNTER

BRNE CONT\_3\_0

CONT\_3:

LDI R18, 0x7E

OUT 0x18, R18

RJMP FAR\_END

DOOR\_OPEN:

LDI R19, 0x7E

OUT 0x18, R19

RCALL DELAY\_1sec\_2

IN R16, 0x10

SBRS R16, 0

RJMP LOOP3

LDI R19, 0x7F

OUT 0x18, R19

RCALL DELAY\_1sec\_2

IN R16, 0x10

SBRS R16, 0

RJMP LOOP3

RJMP DOOR\_OPEN

OVERLOAD:

LDI R19, 0x7C

OUT 0x18, R19

RCALL DELAY\_1sec\_2

IN R16, 0x10

SBRS R16, 1

RJMP LOOP3

LDI R19, 0x7E

OUT 0x18, R19

RCALL DELAY\_1sec\_2

IN R16, 0x10

SBRS R16, 1

RJMP LOOP3

RJMP OVERLOAD

NO\_WATER:

LDI R19, 0x3C

OUT 0x18, R19

RCALL DELAY\_1sec\_2

IN R16, 0x10

SBRS R16, 7

RJMP LOOP3

LDI R19, 0x3E

OUT 0x18, R19

RCALL DELAY\_1sec\_2

IN R16, 0x10

SBRS R16, 7

RJMP LOOP3

RJMP NO\_WATER

LOOP3:

IN R18, 0x10

CPI R18, 0xFF

BRNE LOOP3

OUT 0x18, R18

RET

DELAY\_1sec:

.def inner\_count\_L=R24

.def inner\_count\_H=R25

.def outer\_count\_L=R26 ; (XL)

.def outer\_count\_H=R27 ; (XH)

;----------

ldi outer\_count\_L, 0x7E ;1 cycle

ldi outer\_count\_H, 0x03; 1 cycle

outer\_loop:

ldi inner\_count\_L, 0x7E ;1 cycle

ldi inner\_count\_H, 0x03; 1 cycle

IN R16, 0x10

SBRS R16, 0

RCALL DOOR\_OPEN

IN R16, 0x10

SBRS R16, 1

RCALL OVERLOAD

IN R16, 0x10

SBRS R16, 7

RCALL NO\_WATER

inner\_loop:

nop ; 1 cycle, possible use of more "nop" instructions for longer delays

sbiw inner\_count\_L, 1 ; 2 cycles

brne inner\_loop ; 2 cycles if true (Ni-1 times), 1 cycle if false

sbiw outer\_count\_L, 1 ; 2 cycles

brne outer\_loop ; 2 cycles if true (No-1 times), 1 cycle if false

;----- End of subroutine DELAY\_1sec -----

.undef inner\_count\_L

.undef inner\_count\_H

.undef outer\_count\_L

.undef outer\_count\_H

RET

; end of delay subroutine

DELAY\_1sec\_2:

.def inner\_count\_L=R24

.def inner\_count\_H=R25

.def outer\_count\_L=R26 ; (XL)

.def outer\_count\_H=R27 ; (XH)

;----------

ldi outer\_count\_L, 0x7E ;1 cycle

ldi outer\_count\_H, 0x03; 1 cycle

outer\_loop2:

ldi inner\_count\_L, 0x7E ;1 cycle

ldi inner\_count\_H, 0x03; 1 cycle

inner\_loop2:

nop ; 1 cycle, possible use of more "nop" instructions for longer delays

sbiw inner\_count\_L, 1 ; 2 cycles

brne inner\_loop2 ; 2 cycles if true (Ni-1 times), 1 cycle if false

sbiw outer\_count\_L, 1 ; 2 cycles

brne outer\_loop2 ; 2 cycles if true (No-1 times), 1 cycle if false

;----- End of subroutine DELAY\_1sec -----

.undef inner\_count\_L

.undef inner\_count\_H

.undef outer\_count\_L

.undef outer\_count\_H

RET

; end of delay subroutine

FAR\_END:

RJMP FAR\_END