

```

, *****
; BASIC .ASM template file for AVR
;El programa tiene como entrada:
;Teclado matricial, para ingresar el modelo de habitacion y darle enter
;Como salida el visualizador LCD y también los pines para la conmutacion ON/OFF de los MOSFET
, *****

.include "C:\VMLAB\include\m8def.inc"

.EQU DDR_LCD=DDRB
.EQU PORT_LCD=PORTB
.EQU PIN_LCD=PINB

.EQU LCD_RS=0
.EQU LCD_RW=2
.EQU LCD_E =3

.DEF ARGUMENT_LCD=R19
.DEF RETURN_LCD=R21
.def columna = r20
.def indice = r22
.dseg
.org $060
fila: .byte 1
bloque: .byte 1
tecla: .byte 1
MARIANAX: .byte 1
INTE: .byte 1
PRESENCIA: .byte 1
CUENTA: .byte 1

.cseg

.org $000
RJMP INICIO

.ORG $06
rjmp INT_CTC ; TIMER PARA CUANDO SE TERMINA EL TIEMPO

////////////////////////////////////
////////////////////////////////////
//////
;/// INICIO DEL PROGRAMA
////////////////////////////////////
////////////////////////////////////

INICIO:

```

```
LDI R16, HIGH(RAMEND)      ; configura el puntero de pila
OUT SPH, R16                ; al final de la RAM
LDI R16, LOW(RAMEND)
OUT SPL, R16
```

```
RCALL CONFIGURA_IO
RCALL CONFIG_LCD
RCALL CONFIGURA_TMR1
```

```
SEI
```

```
CBI PORTC, 2
CBI PORTC, 3
```

```
CLR R16
CLR R18
STS TECLA, R16
STS MARIANAX, R16
STS INTE, R16
STS CUENTA, R16
STS PRESENCIA, R16
```

```
MENSAJE_INICIAL:
    LDI ZH, HIGH(TABLA0*2)
    LDI ZL, LOW(TABLA0*2)
    RCALL LCD_PRINT_MSJE

    LDI ARGUMENT_LCD, $C0
    RCALL LCD_COMMAND

    LDI ZH, HIGH(TABLA00*2)
    LDI ZL, LOW(TABLA00*2)
    RCALL LCD_PRINT_MSJE
```

```
xxxx:
```

```
    coll1:
    ldi columna, 0b11100000
    ldi indice, 0
    rcall ExploraTeclado
```

```
    coll2:
    ldi columna, 0b11010000
    ldi indice, 1
    rcall ExploraTeclado
```

```
    coll3:
    ldi columna, 0b10110000
    ldi indice, 2
    rcall ExploraTeclado
```

```
col4:
ldi columna,0b01110000
ldi indice,3
rcall ExploraTeclado
```

ESPERANDO_BOTON:

```
LDS R16, tecla
cpi r16, 5
BREQ INICIO_1
CPI R16, 1
BREQ INICIO_1
CPI R16, 2
BREQ INICIO_1
CPI R16, 3
BREQ INICIO_1
CPI R16, 4
BREQ INICIO_1
CPI R16, 8
BREQ INICIO_1
CPI R16, 9
BREQ INICIO_1
RJMP xxxx
```

INICIO_1:

```
clr r16
sts tecla, r16
```

```
LDI ARGUMENT_LCD, $80
RCALL LCD_COMMAND
```

```
LDI ZH, HIGH(TABLA*2)
LDI ZL, LOW(TABLA*2)
RCALL LCD_PRINT_MSJE
```

```
LDI ARGUMENT_LCD, $C0
RCALL LCD_COMMAND
```

```
LDI ZH, HIGH(TABLA2*2)
LDI ZL, LOW(TABLA2*2)
RCALL LCD_PRINT_MSJE
```

ESPERANDO_BOTON2:

```
col1:
ldi columna, 0b11100000
```

ldi indice, 0
rcall ExploraTeclado

col2:
ldi columna, 0b11010000
ldi indice,1
rcall ExploraTeclado

col3:
ldi columna, 0b10110000
ldi indice,2
rcall ExploraTeclado

col4:
ldi columna,0b01110000
ldi indice,3
rcall ExploraTeclado

LDS R16, tecla
CPI R16, 1
BREQ INICIO_2
CPI R16, 2
BREQ INICIO_2
CPI R16, 3
BREQ INICIO_2
CPI R16, 4
BREQ INICIO_2
RJMP esperando_boton2

INICIO_2:

clr r16
sts tecla, r16

LDI ARGUMENT_LCD, \$80
RCALL LCD_COMMAND

LDI ZH, HIGH(TABLA5*2)
LDI ZL, LOW(TABLA5*2)
RCALL LCD_PRINT_MSJE

LDI ARGUMENT_LCD, \$C0
RCALL LCD_COMMAND

LDI ZH, HIGH(TABLA6*2)
LDI ZL, LOW(TABLA6*2)
RCALL LCD_PRINT_MSJE

INICIO_3:

Ccol1:

ldi columna, 0b11100000

ldi indice, 0

rcall ExploraTeclado

Ccol2:

ldi columna, 0b11010000

ldi indice,1

rcall ExploraTeclado

Ccol3:

ldi columna, 0b10110000

ldi indice,2

rcall ExploraTeclado

Ccol4:

ldi columna,0b01110000

ldi indice,3

rcall ExploraTeclado

LDS R16, tecla

CPI R16, 8

BREQ INICIO_ENTER

CPI R16, 9

BRNE INICIO_3

RJMP INICIO_1

INICIO_ENTER: ; SE ACTIVAN LAS SALIDAS PARA LOS MOSFET, UNA DE CADA SENSOR

SBI PORTC, 2

SBI PORTC, 3

LDI ARGUMENT_LCD, \$80

RCALL LCD_COMMAND

LDI ZH, HIGH(TABLA7*2)

LDI ZL, LOW(TABLA7*2)

RCALL LCD_PRINT_MSJE

LDI ARGUMENT_LCD, \$C0

RCALL LCD_COMMAND

LDI ZH, HIGH(TABLA8*2)

LDI ZL, LOW(TABLA8*2)

RCALL LCD_PRINT_MSJE

FINAL_FINAL:

LDS R16, CUENTA
CPI R16, 10
BRNE FINAL_FINAL

CBI PORTC, 2
CBI PORTC, 3

LDI ARGUMENT_LCD, \$80
RCALL LCD_COMMAND
 LDI ZH, HIGH(TABLA9*2)
 LDI ZL, LOW(TABLA9*2)
 RCALL LCD_PRINT_MSJE

LDI ARGUMENT_LCD, \$C0
RCALL LCD_COMMAND

LDI ZH, HIGH(TABLA10*2)
LDI ZL, LOW(TABLA10*2)
RCALL LCD_PRINT_MSJE

TERMINO:
RJMP TERMINO

ExploraTeclado:
out portd, column
nop
in r16, pind
andi r16, \$0f
sts fila, r16
cpi r16, 0b00001111
brne PrimeraFila
rjmp finteclado

PrimeraFila:
cpi r16, 0b00001110
brne SegundaFila
uno:
ldi r16, \$0f
out portd, r16
in r16, pind
andi r16, \$0f
cpi r16, 0b00001111
brne uno

ldi r17,0

```
sts bloque, r17
rjmp lecturaTablaTecla
```

SegundaFila:

```
cpi r16, 0b00001101
brne TerceraFila
```

dos:

```
ldi r16, $0f
out portd, r16
in r16, pind
andi r16, $0f
cpi r16, 0b00001111
brne dos
ldi r17, 1
sts bloque, r17
rjmp lecturaTablaTecla
```

TerceraFila:

```
cpi r16, 0b00001011
brne CuartaFila
```

tres:

```
ldi r16, $0f
out portd, r16
in r16, pind
andi r16, $0f
cpi r16, 0b00001111
brne tres
ldi r17, 2
sts bloque, r17
rjmp lecturaTablaTecla
```

CuartaFila:

```
cuatro:
ldi r16, $0f
out portd, r16
in r16, pind
andi r16, $0f
cpi r16, 0b00001111
brne cuatro
ldi r17, 3
sts bloque, r17
rjmp lecturaTablaTecla
```

lecturaTablaTecla:

```
lds r16, bloque
add r16, r16
add r16, r16
```

```
add r16, indice
```

```
; Se lee la tabla
```

```
ldi zh, high(tablatecla*2)
```

```
ldi zl, low(tablatecla*2)
```

```
add zl, r16
```

```
clr r16
```

```
adc zh, r16
```

```
lpm r16, z
```

```
sts tecla, r16 ; TECLA TIENE EL VALOR DE LO PRESIONADO
```

```
finteclado:
```

```
RET
```

CONFIGURA_IO:

```
LDI R16 , $F0 ;configuro pines pd7, pd6, pd5, pd4 como salidas: col4, col3, col2, col1
```

```
OUT DDRD , R16 ;configuro pines pd3, pd2, pd1, pd0 como entradas: fil4, fil3, fil2, fil1
```

```
LDI R16 , $0F
```

```
OUT PORTD, R16
```

```
LDI R16 , 0b1111100 ; puerto C como salida y entrada
```

```
OUT DDRC , R16 ;PC0, PC1 entradas de los sensores. PC2 y PC3 salidas para activar MOSFETs de los sensores
```

```
RET
```

```
; Tabla que contiene al valor correspondiente a cada tecla
```

```
tablatecla:
```

```
.db 1,2,3,5,4,5,5,5,5,5,5,5,5,8,9 ; 8 ENTER Y 9 MENU
```

CONFIG_LCD:

```
LDI R16, 0B00001110
```

```
OUT DDR_LCD, R16
```

```
RCALL LCD_DELAY
```

```
LDI ARGUMENT_LCD, $20
```

```
RCALL LCD_COMMAND8
```

```
RCALL LCD_WAIT
```

```
LDI ARGUMENT_LCD, $28
```

```
RCALL LCD_COMMAND
```

```
RCALL LCD_WAIT
```

```
LDI ARGUMENT_LCD, $0F
```


RCALL LCD_COMMAND

RCALL LCD_WAIT

LDI ARGUMENT_LCD, \$01

RCALL LCD_COMMAND

RCALL LCD_WAIT

LDI ARGUMENT_LCD, \$06

RCALL LCD_COMMAND

RET

,*****
/
*

LCD_DELAY:

CLR R2

LCD_DELAY1:

CLR R3

LCD_DELAY2:

DEC R3

BRNE LCD_DELAY2

DEC R2

BRNE LCD_DELAY1

RET

,*****
/
*

LCD_COMMAND8:

LDI R16, \$FF

OUT DDR_LCD, R16

IN R16, PORT_LCD

ANDI R16, \$0F

ANDI ARGUMENT_LCD, \$F0

OR R16, ARGUMENT_LCD

OUT PORT_LCD, R16

SBI PORT_LCD, LCD_E

NOP

```
NOP
NOP
```

```
CBI PORT_LCD, LCD_E
```

```
IN R16, DDR_LCD
ANDI R16, $0F
OUT DDR_LCD, R16
```

```
RET
```

```
,*****
,*
*
```

LCD_WAIT:

```
RCALL LCD_GETADDR
ANDI RETURN_LCD, $80
BRNE LCD_WAIT
RET
```

```
,*****
,*
*
```

LCD_GETADDR:

```
IN R16, DDR_LCD
ANDI R16, $0F
OUT DDR_LCD, R16

CBI PORT_LCD, LCD_RS
SBI PORT_LCD, LCD_RW
SBI PORT_LCD, LCD_E

NOP

IN R16, PIN_LCD
ANDI R16, $F0
MOV RETURN_LCD, R16

CBI PORT_LCD, LCD_E

NOP
NOP
NOP

SBI PORT_LCD, LCD_E

NOP
```

```
IN R16, PIN_LCD
ANDI R16, $F0
SWAP R16
OR RETURN_LCD, R16
```

```
CBI PORT_LCD, LCD_E
CBI PORT_LCD, LCD_RW
```

```
RET
```

```
,*****
/*
```

LCD_COMMAND:

```
PUSH ARGUMENT_LCD
IN R16, DDR_LCD
SBR R16, $F0
OUT DDR_LCD, R16
IN R16, PORT_LCD
CBR R16, 0B11111110
CBR ARGUMENT_LCD, $0F
OR R16, ARGUMENT_LCD
```

```
OUT PORT_LCD, R16
SBI PORT_LCD, LCD_E
NOP
NOP
NOP
CBI PORT_LCD, LCD_E
POP ARGUMENT_LCD
CBR R16, $F0
SWAP ARGUMENT_LCD
CBR ARGUMENT_LCD, $0F
OR R16, ARGUMENT_LCD
OUT PORT_LCD, R16
SBI PORT_LCD, LCD_E
NOP
NOP
NOP
CBI PORT_LCD, LCD_E
IN R16, DDR_LCD
CBR R16, $F0
OUT DDR_LCD, R16
```

```
RET
```

```
,*****
/*
```

LCD_PRINT_CHAR:

```
    PUSH ARGUMENT_LCD
    IN R16, DDR_LCD
    SBR R16, $F0
    OUT DDR_LCD, R16
    IN R16, PORT_LCD
    CBR R16, 0B11111110
    CBR ARGUMENT_LCD, $0F
    OR R16, ARGUMENT_LCD

    OUT PORT_LCD, R16
    SBI PORT_LCD, LCD_RS
    SBI PORT_LCD, LCD_E
    NOP
    NOP
    NOP
    CBI PORT_LCD, LCD_E
    POP ARGUMENT_LCD
    CBR R16, $F0
    SWAP ARGUMENT_LCD
    CBR ARGUMENT_LCD, $0F
    OR R16, ARGUMENT_LCD
    OUT PORT_LCD, R16
SBI PORT_LCD, LCD_RS
    SBI PORT_LCD, LCD_E
    NOP
    NOP
    NOP
    CBI PORT_LCD, LCD_E
    CBI PORT_LCD, LCD_RS
    IN R16, DDR_LCD
    CBR R16, $F0
    OUT DDR_LCD, R16
```

RET

,*****
,*
*

LCD_GET_CHAR:

```
    IN R16, DDR_LCD
    ANDI R16, $0F
    OUT DDR_LCD, R16

    SBI PORT_LCD, LCD_RS
    SBI PORT_LCD, LCD_RW
    SBI PORT_LCD, LCD_E
```

```

NOP
nop
nop
cbi port_lcd, lcd_e

IN R16, PIN_LCD
ANDI R16, $F0
MOV RETURN_LCD, R16

CBI PORT_LCD, LCD_E

NOP
NOP
NOP

SBI PORT_LCD, LCD_E

NOP

IN R16, PIN_LCD
ANDI R16, $F0
SWAP R16
OR RETURN_LCD, R16

CBI PORT_LCD, LCD_E
CBI PORT_LCD, LCD_RW

RET
,*****
*

LCD_PRINT_MSJE:

LPM R16, Z+
CPI R16, $FF
BREQ FIN_LCD_PRINT_MSJE

MOV ARGUMENT_LCD, R16
RCALL LCD_WAIT
RCALL LCD_PRINT_CHAR
RJMP LCD_PRINT_MSJE

FIN_LCD_PRINT_MSJE:
RET
,*****
*

;DEFINICION DE TABLAS

```

,*****
,
*

TABLA0:

; 0123456789ABCDEF
.DB " BIENVENIDO ", \$FF

TABLA00:

;0123456789ABCDEF
.DB "PRESIONE UN BOTO", \$FF

TABLA:

; 0123456789ABCDEF
.DB " INGRESO MODELO ", \$FF

TABLA2:

;0123456789ABCDEF
.DB " 1,2,3 o 4 ", \$FF

TABLA3:

.DB "PRODUCCION DE O3", \$FF

TABLA4:

.DB " DETENIDA ", \$FF

TABLA5:

.DB " PRESIONE ", \$FF

TABLA6:

.DB " ENTER o MENU ", \$FF

TABLA7:

.DB " DESINFECCION ", \$FF

TABLA8:

.DB " EN PROCESO ", \$FF

TABLA9:

.DB " AMBIENTE ", \$FF

TABLA10:

.DB " DESINFECTADO ", \$FF

CONFIGURA_TMR1:

; Prescalamiento 1:1024

; Modo CTC

LDI R16, (0<<COM1A1 | 0<<COM1A0 | 0<<WGM11 | 0<<WGM10)

```

OUT TCCR1A, R16
LDI R16, (0<<WGM13 | 1<<WGM12 | 1<<CS12 | 0<<CS11 | 1<<CS10)
OUT TCCR1B, R16
LDI R17, $00 ; Valor de Registro OCR1A: 244 ($00F4)
LDI R16, $F4
OUT OCR1AH, R17 ; (1us)x(1024)x(244) = 249.8ms
OUT OCR1AL, R16
IN R16, TIMSK ; Habilitamos interrupción
ORI R16, (1<<OCIE1A) ; TIMER1 COMPA
OUT TIMSK, R16
RET

```

```

,*****
,
;***** RUTINA DE SERVICIO DE INTERRUPCION *****
;Para que se apague cuando se termina el tiempo
,*****
INT_CTC:
PUSH R16 ; Salvamos registros en pila
IN R16, SREG
PUSH R16
LDS R16, CUENTA ; Incrementamos el valor de CUENTA
INC R16
STS CUENTA, R16
SAL_INT: POP R16 ; Recuperamos registros de pila
OUT SREG, R16
POP R16
RETI

```