**Design Assignment on**

**Microwave Oven**

*For the partial fulfillment of the course*

*CS F241 - Microprocessor Programming & Interfacing BITS Pilani, Pilani Campus*

**Designed By:**

AYUSH JAIN 2017A7PS0093P

KUNDA SATYANARAYANA 2017A7PS0095P

SAHIL DUBEY 2017A7PS0096P

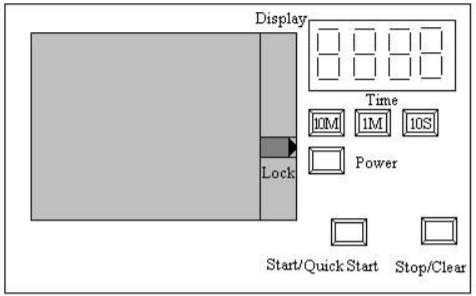
BURHAN BOXWALLA 2017A7PS0097P

# Problem Statement

***System to be Designed : Microwave Oven (P5)***

**Description:** A Simple Microwave Oven without grill.

**User Interface:** Is shown in the following Figure



* User can cook at 3 different Power levels: 90%, 60%, 30%
* Press of a Power Button decrements the power level by 30 %
* 1 Press - 90%; 2 Presses – 60% ; 3 Presses – 30%; 4 Presses – 10 %;
* 4 Presses – Brings the power level back to 100 %
* The Default power level is 90%
* Power Level is varied by controlling the amount of time for which the microwave is turned on.
* Time of cooking is broken up into 10 sec slots, if power is 60% then for 6 seconds the microwave is on and rest of the 4 seconds the microwave is off.
* Time is set as multiples of 10 Mins, 1 Min, 10 Secs. For e.g. if the cooking time is 12 Minutes and 40 secs- the 10 Minutes button has to be pressed once, 1 Minute Button has to be pressed Twice and 10 seconds button has to be pressed four times.
* Once Time has been set Power cannot be modified.
* When user is setting power level or Time, the value being pressed should be displayed, and when user presses the Start button, the cooking process begins and the time left for cooking to complete is displayed.
* Once the cooking begins the door gets locked and should open only when cooking process is terminated.
* User can terminate cooking anytime by pressing the STOP button. When Stop button is pressed once cooking is aborted, timer is stopped, not cleared; cooking can be resumed by pressing Start.
* When stop is pressed twice, cooking is aborted and timer is also cleared.
* When cooking time elapses, a buzzer is sounded; pressing the Stop Button stops the buzzer.
* A Quick Start mode is available where timer or power need not be set, just Start button needs to be pressed, the default power value is taken and time is set as 30 secs, for every press of the start button time is incremented by 30 seconds.

# Assumptions

* There is mechanism already in place whereby door will get locked if PC7 of 8255A is high and unlocked if PC7 is low.
* The heating element of microwave oven is already available which amplifies the current sent to it by 8253.
* The time required for loading the latched values into counters of 8253 after giving the gate trigger has been taken as negligible in comparison to total time.
* The fourth press of the power button sets the power back to 90%
* To enhance user experience, the LEDs display total time of cooking instead of the value of the button pressed
* Maximum time for cooking user can set is 9999 seconds.
* Code is stored - 0000h
* Time Display format - SSSS
* Power Display format - PPPP
* The door will automatically get locked once the user presses start/quick start and will open when the process gets completed.
* Multiple keys cannot be pressed simultaneously
* A clock frequency of 1Hz is available to be given to TIMER 2.

# List of Important Components Used

|  |  |  |  |
| --- | --- | --- | --- |
| Chip Number | Chip | No. of Chips used | Use |
| 8086 | Microprocessor | 1 | CPU |
| 2732 | ROM | 2 | Read Only Memory |
| 74LS373 | 8 - BIT latch | 3 | To latch Address Bus |
| 74541 | 8 - BIT Buffer | 1 | Unidirectional Buffer |
| 8255 | Programmable Peripheral Interface | 2 | Connected to various I/O devices |
| 8253 | Clock timer | 1 | To produce a 1 second delay |
| 74HC138 | 3:8 Decoder | 1 | For selecting between the various components like ROM, RAM, TIMER1 |
| 6116 | RAM - 2K | 2 | Random Access Memory |
| 74LS245 | Octal bidirectional buffer | 2 | To be filled!!!! |

Other Components Used

1. **An LED is used as a Buzzer** - To Indicate the end of cooking

## Resistors ·

1. **7 Segment Display(DL707)** - To Display Time and Power(5)(active low)
2. **OR** Gates(4071)
3. **Push** Buttons - To input the power, time, start and stop signals from the user.

## VCC, Ground, LED’s

Hardware Specifications

|  |  |  |  |
| --- | --- | --- | --- |
| **Ports of 8255A** | **Address** | **Mode** | **Function of port** |
| **Port A** | 5000h | - | - |
| **Port B** | 5002h | - | - |
| **Port C** | 5004h | Mode 0 - Output | Signals sent to buzzer, lock, select lines of 7 segment display |
| **Control Register** | 5006h | - | Programming of PPI |

|  |  |  |  |
| --- | --- | --- | --- |
| **Ports of 8255B** | **Address** | **Mode** | **Function of Port** |
| **Port A** | 6000h | Mode 0 - Output | Output for 7 segment display |
| **Port B** | 6002h | Mode 1 - Input | Takes input from the 6 buttons and the timer |
| **Port C** | 6004h | - | - |
| **Control Register** | 6006h | - | Programming of PPI |

|  |  |  |  |
| --- | --- | --- | --- |
| **8253 - TIMER 2** | **Address** | **Mode** | **Count Loaded/Function** |
| **Counter A** | 3000h | Mode 0 | 1 second |
| **Counter B** | 3002h | - | - |
| **Counter C** | 3004h | - | - |
| **Control Register** | 3006h | - | Used for Programming Timer |

MEMORY MAPPING

**RAM : 01000H - 01FFFH**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **RAM**  **(even)** | **A19** | **A18** | **A17** | **A16** | **A15** | **A14** | **A13** | **A12** | **A11** | **A10** | **A9** | **A8** | **A7** | **A6** | **A5** | **A4** | **A3** | **A2** | **A1** | **A0** |
| **From 01000h** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **To 01FFEh** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **RAM**  **(odd)** | **A19** | **A18** | **A17** | **A16** | **A15** | **A14** | **A13** | **A12** | **A11** | **A10** | **A9** | **A8** | **A7** | **A6** | **A5** | **A4** | **A3** | **A2** | **A1** | **A0** |
| **From 01001h** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| **To 01FFFh** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

**ROM: 00000H - 00FFFH**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ROM**  **(even)** | **A19** | **A18** | **A17** | **A16** | **A15** | **A14** | **A13** | **A12** | **A11** | **A10** | **A9** | **A8** | **A7** | **A6** | **A5** | **A4** | **A3** | **A2** | **A1** | **A0** |
| **From 00000h** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **To 00FFEh** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ROM**  **(odd)** | **A19** | **A18** | **A17** | **A16** | **A15** | **A14** | **A13** | **A12** | **A11** | **A10** | **A9** | **A8** | **A7** | **A6** | **A5** | **A4** | **A3** | **A2** | **A1** | **A0** |
| **From 00001h** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| **To 00FFFh** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

**Flowchart**

Insrt flowchart

Insert picof design

9

## CODE

.model tiny

.data

sec dw 0 ;Stores the total time left for the countdown

power db 0 ;Stores the number of power presses

start db 0 ;Stores the number of start button presses

stop db 0 ;Stores the number of stop button presses

disp equ 1015h ;Next 4 bytes will be stored for 4-digit BCD count

display\_table equ 1019h ;Display table for displaying digits on the 7-segement display

inp equ 1029h ;Stores the input from the port - to check which button is pressed

time\_loop equ 1030h

align 2

print dw 0

start\_flag db 0 ; Gives status of start button

stop\_flag db 0 ;Gives status of stop button

time\_flag db 0 ;Gives status of time button

flag\_10m db 0 ; Gives status of 10 min button

flag\_1m db 0 ; Gives status of 1 min button

flag\_10s db 0 ; Gives status of 10 sec button

power\_flag db 0 ; Gives status of power button

button\_flag db 0 ; Gives status whether any button is pressed or not

power\_value dw 0 ; Gives value of power to be dsplayed

temp db 0

temp2 db 0 ; Checks no. of times power button pressed

quick\_flag db 0 ; Checks for quickstart mode

out\_flag db 0

ms\_status db 0 ; Time for which power led must be on while cooking

ms\_flag db 0

;8255(A) port addresses

A\_a equ 5000h

A\_b equ 5002h

A\_c equ 5004h

A\_creg equ 5006h

;8255(B) port addresses

B\_a equ 6000h

B\_b equ 6002h

B\_c equ 6004h

B\_creg equ 6006h

.code

.startup

;Display Table Initialization for 7 segment display

mov si,display\_table

mov byte ptr [si],3fh

inc si

mov byte ptr [si],06h

inc si

mov byte ptr [si],5bh

inc si

mov byte ptr [si],4fh

inc si

mov byte ptr [si],66h

inc si

mov byte ptr [si],6dh

inc si

mov byte ptr [si],7dh

inc si

mov byte ptr [si],07h

inc si

mov byte ptr [si],7fh

inc si

mov byte ptr [si],67h

inc si

;Initializing 8255(A)

mov al,10000000b ;Because all ports are input ports, we make them output for bsr

mov dx,A\_creg

out dx,al

mov al,00001010b ;Buzzer Indicator OFF

out dx,al

mov al,00001100b ;Power input can be taken

out dx,al

mov al,00001001b ;Countdown can't be started

out dx,al

;PC0 - PC3 set 1 -- all the 4 seven segement displays are disabled

mov al,00000001b

out dx,al

mov al,00000011b

out dx,al

mov al,00000101b

out dx,al

mov al,00000111b

out dx,al

;Initializing 8255(B)

mov al,10000010b ;Port B is taken as input

mov dx,B\_creg

out dx,al

;Initializing the values at the following addresses:

mov sec , 0 ;Total no. of seconds loaded initially

mov power,-1 ;No. of times Power is pressed

mov start,0 ;No. of times start is pressed

mov stop,00 ;No. of times stop is pressed

;Initializing timers

mov al,00010001b ;TIMER2 COUNTER 0 Control Word

mov dx,3006h

out dx,al

;----------------------------------------------------------------------------

button\_press:

mov quick\_flag, 0 ; reset all values

mov sec , 0

call check\_button ; check if any button pressed

cmp button\_flag, 0 ; non zero value indicates some button pressed

jz button\_press ; if no button pressed, wait foe button to be pressed

cmp start\_flag, 0 ; check if start button was pressed

jz x1 ; if not, jump to default initialization

jmp quick\_start ; if yes, go to quickstart mode

jqs: jmp button\_press

x1:

mov power\_value, 90 ; move default power value

mov ms\_status , 09 ; set time for power led to be on accordingly

cmp power\_flag, 0 ; check if power button pressed

jz x2

mov temp2 , 1

cp:

call display\_power ; display power on 7 segment display

cmp power\_flag, 0 ; check if power button pressed

jz x2 ; if not, jump

inc temp2 ; if yes, increment no. of power button presses by 1

jmp cp

;-------------------------------------

display\_power proc near

call calculate\_power ; power will be calculated

mov ax , power\_value

mov print , ax

cd:

call display ; power displayed

call check\_button

cmp button\_flag , 00 ; check if button pressed

jz cd ; if no, keep checking

ret

display\_power endp

;-----------------------------------------

calculate\_power proc near

mov cl , temp2

mov ch, 0

mov ax, cx ; move total power button presses to ax

mov bl, 3

div bl ; divide by 3

cmp ah, 0

jnz p1 ; if no. of times power button pressed is not a multiple of 3, jump

mov power\_value, 30 ; if multiple of 3, set values accordingly

mov ms\_status , 03

jmp r

p1:

cmp ah, 1

jnz p2

mov power\_value, 90 ; if no. of times power button pressed is one more than a multiple ;of 3, set values accordingly

mov ms\_status , 09

jmp r

p2:

cmp ah, 2

mov power\_value, 60 ; if no. of times power button pressed is two more than a multiple ; of 3, set values accordingly

mov ms\_status , 06

jmp r

r: ret

calculate\_power endp

;------------------------------------------

quick\_start:

mov sec,0

mov power\_value, 90d ; set default values for quickstart

mov ms\_status , 09d

mov quick\_flag, 1

call enable\_lock ; lock the door

qs:

mov ax , sec

add ax , 30

mov sec , ax ; increment time by 30 secs

mov print,ax

jmp timer\_display

;------------------------------------------

x2:

cmp stop\_flag, 0 ; check if stop button pressed

jnz button\_press ; if pressed, stop execution and wait for next sequence

cmp start\_flag, 0 ; check if start button pressed

jnz cp ; if pressed, jump

before\_start\_normal:

time\_set:

cmp flag\_10m, 0 ; check if 10 min button pressed

jz call\_1m ; if no, jump

mov ax , sec

add ax , 600d

mov sec , ax ; 600 secs added to sec

mov print , ax

cd1:

call display

call check\_button

cmp button\_flag , 00 ; check if any button pressed

jz cd1 ;display the total time

jmp time\_set ; if yes, check which button

call\_1m:

cmp flag\_1m, 0 ; check if 1 min button pressed

jz call\_10s ; if no, jump

mov ax , sec

add ax , 60d

mov sec , ax ; 60 secs added to sec

mov print , ax

jmp cd1 ; display the total time

call\_10s:

cmp flag\_10s, 0 ; check if 1 sec button pressed

jz final ; if no, jump

mov ax , sec

add ax , 10d

mov sec , ax ; 10 secs added to sec

mov print , ax

jmp cd1 ; display the time

final:

cmp start\_flag, 0 ; check if start button pressed

jnz cooking ; if yes, start cooking

cmp stop\_flag, 0 ; check if stop button pressed

jnz button\_press ; if yes, jump

;-------------------------------------------------

cooking:

call enable\_lock ; lock the door

timer\_display:

mov ax , sec

mov print , ax

call check\_ms\_status ; check if power led must be on or off

call display

call delay\_1s ; delay by 1 sec

dec sec ; decrement time count

jnz ccb ; if not zero, jump

mov print, 0

call display ; display 0 on 7 segment display

call enable\_buzzer ; buzzer led switched on

call remove\_lock ; door lock opened

call power\_off ; power turned off

kunda1: call check\_button

cmp button\_flag, 0 ; check if any button pressed

jz kunda1 ; if no button pressed, wait for button press

cmp stop\_flag, 1 ; check if stop button pressed

jnz kunda1 ; if not, wait

call delay

call reset ; if yes, reset

jmp dead

ccb: call check\_button ; check if any button pressed

cmp button\_flag, 0

jz timer\_display

cmp quick\_flag, 1 ; check for quickstart mode

jnz ayush1 ; if no, jump

cmp start\_flag, 0 ; check if start button pressed

jnz qs ; if yes, go to quickstart mode

ayush1:

cmp stop\_flag, 0 ; check if stop button pressed

jnz final\_stop ; if yes, jump

jmp timer\_display

;-----------------------------

reset proc near ; reset to initial mode

mov print, 0

call display

mov sec, 0

call stop\_buzzer

call remove\_lock

call power\_off

ret

reset endp

;---------------------------

final\_stop:

call power\_off ; switch off power led

call remove\_lock ; open lock

mov ax, sec

mov print, ax

cb: call display

call check\_button

cmp button\_flag, 0 ; check if any button pressed

jz cb

cmp stop\_flag, 0 ; check if stop button pressed

jz Burhan ; if no, jump

call reset ; if yes, reset

jmp dead ; jump to end of sequence

burhan:

cmp start\_flag, 0 ; check if start button pressed

jz final\_stop ; if no, jump

call enable\_lock ; if yes, lock the door

cmp quick\_flag, 0 ; check for quickstart mode

jnz timer\_display ; if no, display time

jmp before\_start\_normal

;----------------------------------

remove\_lock proc near

mov ax,00001110b ;pc7 resets i.e lock led turns off

mov dx, A\_creg

out dx,al

ret

remove\_lock endp

;---------------------------------

enable\_lock proc near

mov ax,00001111b ;pc7 resets i.e lock led turns on

mov dx, A\_creg

out dx,al

ret

enable\_lock endp

;---------------------------------

stop\_buzzer proc near

mov ax,00001010b ;pc5 sets i.e buzzer led turns off

mov dx,A\_creg

out dx,al

ret

stop\_buzzer endp

;-------------------------------

enable\_buzzer proc near

mov ax,00001011b ;pc5 sets i.e buzzer led glows

mov dx,A\_creg

out dx,al

ret

enable\_buzzer endp

;-------------------------------------------------

power\_on proc near

mov al, 00001101b ; set control word to light up power led

mov dx, A\_creg

out dx, al

ret

power\_on endp

;-------------------------------------------------

power\_off proc near

mov al, 00001100b ; set control word to switch off led

mov dx, A\_creg

out dx,al

ret

power\_off endp

;--------------------------------------------------

check\_ms\_status proc near

mov ax , sec

mov bl , 10d

div bl

cmp ah , ms\_status ; check if power led should remain on or off

jb l1 ; if last digit if time is less, jump

call power\_off ; switch off

jmp r

l1:

call power\_on ; switch on

r: ret

check\_ms\_status endp

;---------------------------------------------

display proc near

mov si,disp

mov di,offset print

mov ax, [di] ;Current number of seconds is stored in AX now

;mov ax , 1234d

mov bx,10

xor dx,dx

div bx

mov [si],dl ;Remainder is in DL as remainder is not greater than 9, so ;no need to considerDH - Hence we extract the last digit

xor dx,dx ;Digit at Tens place

div bx

mov [si+1],dl

xor dx,dx ; Digit at hundredth place

div bx

mov [si+2],dl

xor dx,dx ; Digit at Thousandth place

div bx

mov [si+3],dl

mov di,display\_table

;Display Last Digit

mov al,[si]

mov bl,al

xor bh,bh

mov ax,[di+bx] ; So the displacement is equal to the digit we have

mov dx,B\_a

out dx,al

mov ax, 00000001b ; Previous display disabled

mov dx,A\_creg

out dx,al

mov ax, 00000110b ; Last digit shown

out dx,al

call delay

mov al,[si+1]

mov bl,al

mov ax,[di+bx]

mov dx,B\_a

out dx,al

mov ax, 00000111b ; Previous display disabled

mov dx,A\_creg

out dx,al

mov ax, 00000100b ; Second Last digit shown

out dx,al

call delay

mov al,[si+2]

mov bl,al

mov ax,[di+bx]

mov dx,B\_a

out dx,al

mov ax, 00000101b ; Previous display disabled

mov dx,A\_creg

out dx,al

mov ax, 00000010b ; Digit at Hundredth place shown

out dx,al

call delay

mov al,[si+3]

mov bl,al

mov ax,[di+bx]

mov dx,B\_a

out dx,al

mov ax, 00000011b ; Previous display disabled

mov dx,A\_creg

out dx,al

mov ax, 00000000b ; Digit at thousandth place shown

out dx,al

call delay

mov ax, 00000001b ; Finally clearing the previous display also

out dx,al

ret

display endp

;----------------------------------------------------------

delay proc near ; used to delay the program

mov cx, 15000d

l5: dec cx

loop l5

ret

delay endp

;----------------------------------------------------------

delay\_1s proc near

mov al, 01 ; Loading count into TIMER2 COUNTER 0

mov dx,3000h

out dx,al

sahil:

call check\_out

cmp out\_flag , 00h

jz sahil

ret

delay\_1s endp

;---------------------------------------------------------

; procedure to check flags

check\_button proc near ; Check if any button pressed

mov button\_flag , 00h

mov bh , 00

mov [temp], 00

call check\_start

mov bh, start\_flag

add [temp] , bh

call check\_stop

mov bh, stop\_flag

add [temp] , bh

call check\_power

mov bh, power\_flag

add [temp] , bh

call check\_10m

mov bh, flag\_10m

add [temp] , bh

call check\_1m

mov bh, flag\_1m

add [temp] , bh

call check\_10s

mov bh, flag\_10s

add [temp] , bh

mov al, [temp]

mov button\_flag , al

ret

check\_button endp

;-------------------------------------------------------------

check\_start proc near ; Check if start button pressed

mov start\_flag , 00h

mov dx,B\_b

in al,dx

and al , 10h

jnz c2

mov start\_flag , 01d

jmp r

c2: mov start\_flag , 00d

r:

ret

check\_start endp

;------------------------------------------------------------------

check\_stop proc near ; Check if stop button pressed

mov stop\_flag , 00h

mov dx,B\_b

in al,dx

and al , 20h

jnz c2

mov stop\_flag , 01d

jmp r

c2: mov stop\_flag , 00d

r:

ret

check\_stop endp

;-----------------------------------------------------------------

check\_power proc near ; Check if power button pressed

mov power\_flag , 00h

mov dx,B\_b

in al,dx

and al , 08h

jnz c2

mov power\_flag , 01d

jmp r

c2:

mov power\_flag , 00d

jmp r

r:

ret

check\_power endp

;----------------------------------------------------------------

check\_10m proc near ; Check if 10-minute button pressed

mov flag\_10m , 00h

mov dx,B\_b

in al,dx

and al , 01h

jnz c2

mov flag\_10m , 01d

jmp r

c2: mov flag\_10m , 00d

r:

ret

check\_10m endp

;----------------------------------------------------------------

check\_1m proc near ; Check if 1-minute button pressed

mov flag\_1m , 00h

mov dx,B\_b

in al,dx

and al , 02h

jnz c2

mov flag\_1m , 01d

jmp r

c2: mov flag\_1m , 00d

r:

ret

check\_1m endp

;----------------------------------------------------------------

check\_10s proc near ; Check if 10 sec button pressed

mov flag\_10s , 00h

mov dx,B\_b

in al,dx

and al , 04h

jnz c2

mov flag\_10s , 01d

jmp r

c2: mov flag\_10s , 00d

r:

ret

check\_10s endp

;---------------------------------------------------------------

check\_out proc near

mov out\_flag , 00h

mov dx, B\_c

in al , dx

and al , 02h

jz c2

mov out\_flag , 01d

jmp r

c2: mov out\_flag , 00d

r:

ret

check\_out endp

;--------------------------------------------------------------

dead:

jmp button\_press ; Check for next sequence of instructions

. exit

end ; End of program