

Presented By:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Name** | **Contact #** | **D1** | **D2** | **D3** | **D4** |
| 19105477 | Nada Khaled Ahmed Mazen | 01121399923 |  |  |  |  |
| 19104707 | Mariam Ashraf Ahmed | 01113230131 |  |  |  |  |
| 20106344 | Zeyad Ahmed Samir | 01115995531 |  |  |  |  |

Memory Game

Department : Computer Engineering

Course Code : CC421

Course Name : Microprocessor Systems

Lecturer : Prof. Dr Ahmed Fahmy

Teaching Asst. : Eng. Muhammad Sami

# Summary:

The project is a Leveled Memory Game Based on LEDs and buttons. The microcontroller is connected to the buttons in one port and the other port is connected to the LEDs. The microcontroller will view to the user the level and the score on the LCD then LED pattern will be viewed after that immediately the timer will start to decrement from 9-0 waiting for the user to enter the viewed pattern if the player passed within less than 9 seconds the reminder of the time is added to score. If the player’s current score is 12 the player can not lose the game the game can only return to a specified level.

The expected input from the user and the equivalent output:

* The user has 9 seconds to play every level, if the user couldn’t finish the pattern within the given time the microcontroller if the user fails the LCD will show Game Over!
* If the user succeeds, then lcd will show passed until the user passes level 7 which is the final level then the LCD will show YOU’VE WON
* In each level the user has 2 LEDs red and Blue there purpose is to check one by one which means the 2 LEDs checks if the 1st button matches the 1st LED in the pattern then the 2nd until the level is finished

# Individual Tasks/ Responsibilities

|  |  |
| --- | --- |
| **Name** | Mariam Ashraf Ahmed |
| **ID** | 19104707 |
| **Tasks** | * The Hardware Circuit * The Core of the Project Software |

|  |  |
| --- | --- |
| **Name** | Nada Khaled Ahmed |
| **ID** | 19105477 |
| **Tasks** | * The Hardware Circuit * The Core of the software Project |

|  |  |
| --- | --- |
| **Name** | Zeyad Ahmed |
| **ID** | 20106344 |
| **Tasks** | * Timer * Score |

# Introduction

* Memory Game based on LEDs and button across each other. The Game plays a pattern with the LEDs then the player must insert the same pattern on the buttons during 9 second in each level.
* The Game includes 8 buttons and 8 LEDs
* Each led has a specific button across it in the circuit
* Each level has only 9 seconds displayed on the 7-segment
* At the begging of each level lcd display the number of the level
* At the end of each level lcd displays whether the player passed or lost
* The player losses when either the timer hits 0 or when inserting the wrong pattern, then the LCD displays GAME OVER!
* If the player manages to pass all 7 levels the LCD displays YOU’VE WON!
* At passing each level the LCD display Passed

# Methodology

* First, The Game displays what level the player is at
* Then, the LEDs displays a pattern according to which level the player plays
* Finally, the LCD shows whether the player has passed the level, or the game is over
* The input of the Game is the Buttons and the output shown on the Lcd and the LEDs
* Whichever the button the player strikes the LED across it turns on
* if the player passed within less than 9 seconds the reminder of the time is added to score. If the player’s current score is 12 the player cannot lose the game the game can only return to a specified level.

# Schematic:

Chart

Description automatically generated with medium confidence

# Components and Tools:

-SEVEN SEGMENT

-LCD

-PUSH BUTTONS

-LEDS

# Code Illustration:

Code Text:

RED\_LED EQU P1.7

BLUE\_LED EQU P1.6

LED\_PORT EQU P2

RS BIT P1.0

E BIT P1.1

SEVEN\_SEGMENT EQU P3

BUTTON\_PORT EQU P0

;MAIN

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

ORG 00H

MOV R2,#0

JMP MAIN

ORG 000BH

JMP ISRT0

ORG 0100H

MAIN:

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

MOV LED\_PORT , #00H

LEVEL\_1:

MOV LED\_PORT,#0H

MOV DPTR,#LCD\_L1

MOV R6 ,#0

ACALL LCD

MOV DPTR ,#LEVEL1

MOV R3 ,#04H

MOV R5 , #9H

ACALL VIEW\_LED

MOV R7 ,#04H

MOV DPTR ,#LEVEL1

ACALL GAME

MOV DPTR,#PASS

ACALL LCD

JMP LEVEL\_2

LEVEL\_2:

MOV LED\_PORT,#0H

ACALL RESET\_TIMER

MOV DPTR,#LCD\_L2

ACALL LCD

MOV DPTR ,#LEVEL2

MOV R3 ,#05H

MOV R5 , #8H

ACALL VIEW\_LED

MOV R7 ,#05H

MOV DPTR ,#LEVEL2

ACALL GAME

MOV DPTR,#PASS

ACALL LCD

JMP LEVEL\_3

LEVEL\_3:

MOV LED\_PORT,#0H

ACALL RESET\_TIMER

MOV DPTR,#LCD\_L3

ACALL LCD

MOV DPTR ,#LEVEL3

MOV R3 ,#05H

MOV R5 , #7H

ACALL VIEW\_LED

MOV R7 ,#05H

MOV DPTR ,#LEVEL3

ACALL GAME

MOV DPTR,#PASS

ACALL LCD

JMP LEVEL\_4

LEVEL\_4:

MOV LED\_PORT,#0H

ACALL RESET\_TIMER

MOV DPTR,#LCD\_L4

ACALL LCD

MOV DPTR ,#LEVEL4

MOV R3 ,#06H

MOV R5 , #6H

ACALL VIEW\_LED

MOV R7 ,#06H

MOV DPTR ,#LEVEL4

ACALL GAME

MOV DPTR,#PASS

ACALL LCD

JMP LEVEL\_5

LEVEL\_5:

MOV LED\_PORT,#0H

ACALL RESET\_TIMER

MOV DPTR,#LCD\_L5

ACALL LCD

MOV DPTR ,#LEVEL5

MOV R3 ,#05H

MOV R5 , #4H

ACALL VIEW\_LED

MOV R7 ,#06H

MOV DPTR ,#LEVEL5

ACALL GAME

MOV DPTR,#PASS

ACALL LCD

JMP LEVEL\_6

LEVEL\_6:

MOV LED\_PORT,#0H

ACALL RESET\_TIMER

MOV DPTR,#LCD\_L6

ACALL LCD

MOV DPTR ,#LEVEL6

MOV R3 ,#06H

MOV R5 , #8H

ACALL VIEW\_LED

MOV R7 ,#06H

MOV DPTR ,#LEVEL6

ACALL GAME

MOV DPTR,#PASS

ACALL LCD

JMP LEVEL\_7

LEVEL\_7:

MOV LED\_PORT,#0H

ACALL RESET\_TIMER

MOV DPTR,#LCD\_L7

ACALL LCD

MOV DPTR ,#LEVEL7

MOV R3 ,#06H

MOV R5 , #8H

ACALL VIEW\_LED

MOV R7 ,#06H

MOV DPTR ,#LEVEL7

ACALL GAME

MOV DPTR,#WON

ACALL LCD

JMP EXIT

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

GAME:

ACALL START\_TIMER ; START THE TIMER AT THE BEGINNING OF EVERY LEVEL

IS\_PRESSED: ;CHECK THE TIMER WHILE WAITING FOR THE USER TO PRESS A BUTTON

CJNE R4,#0,NATURAL\_FLOW ;IF NOT EQUAL ZERO ACCEPT THE PRESSED BUTTON ELSE

JMP NOT\_EQ ;JUMP TO NOT EQUAL

NATURAL\_FLOW:

MOV P0, #0FFH

JNB P0.0 , B1

JNB P0.1 , B2

JNB P0.2 , B3

JNB P0.3 , B4

JNB P0.4 , B5

JNB P0.5 , B6

JNB P0.6 , B7

JNB P0.7 , B8

SJMP IS\_PRESSED

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

B1:

MOV R3,#01H

JMP COMPARE

B2:

MOV R3,#02H

JMP COMPARE

B3:

MOV R3,#04H

JMP COMPARE

B4:

MOV R3,#08H

JMP COMPARE

B5:

MOV R3,#10H

JMP COMPARE

B6:

MOV R3,#20H

JMP COMPARE

B7:

MOV R3,#40H

JMP COMPARE

B8:

MOV R3,#80H

JMP COMPARE

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

COMPARE:

MOV A , BUTTON\_PORT

CPL A

MOV LED\_PORT,A

CLR A

MOVC A ,@A+DPTR

CPL A

ANL A,R3

CJNE A,#0,NOT\_EQ

CLR RED\_LED ;IF EQUAL, RED LED OFF

SETB BLUE\_LED ;BLUE LED ON

MOV B, #9 ;COUNTER B INITIAL VALUE = 9 FOR DELAY

LOOP2:

ACALL DELAY ;APPLY SOME DELAY

DJNZ B, LOOP2 ;DECREMENT COUNTER B, JUMP IF NOT 0

CLR BLUE\_LED ;BLUE LED OFF

INC DPTR

DJNZ R7, IS\_PRESSED

;PASSED CASE MEANS THAT USER SUCCEEDED IN PATTERN WE WILL UPDATE SCORE AND RETURN

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

MOV B,R6 ; CURRENT VALUE OF SCORE

MOV A,R4 ; MOV THE REMAING TIME THAT IS STORED IN R4 TO THE ACCUMLATOR

ADD A ,B ; ADD THE DIFFERENCE IN TIME TO THE CURRENT VALUE OF SCORE

MOV R6,A ; RESTORE THE NEW SCORE TO THE R6 REGISTER

RET

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

NOT\_EQ: ACALL RESET\_TIMER

SETB RED\_LED ;RED LED ON

MOV B, #9 ;COUNTER B INITIAL VALUE = 9

LOOP3:

ACALL DELAY

DJNZ B , LOOP3 ;APPLY SOME DELAY

CLR RED\_LED

;HERE WE SHOULD CHECK THE SCORE

ACALL CHECK\_SCORE\_AFTER\_LOSS

MOV DPTR,#LOST

ACALL LCD

JMP $

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

VIEW\_LED:

CLR A

MOVC A ,@A+DPTR

;CPL A

MOV LED\_PORT ,A

MOV B , R5

LOOP0:

ACALL DELAY

DJNZ B, LOOP0

INC DPTR

DJNZ R3,VIEW\_LED

MOV LED\_PORT,#0

RET

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

DELAY:

MOV A, #0BBH

LOOP4:

MOV R1, #0BBH

LOOP5:

DJNZ R1, LOOP5

DEC A

CJNE A,0,LOOP4

RET

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

EXIT:

JMP $

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

LCD:

MOV A, #02H

CALL CMDWRITE

MOV A, #28H

CALL CMDWRITE

MOV A, #0CH

CALL CMDWRITE

MOV A, #01H

CALL CMDWRITE

;BEGIN AT 5TH COL, 1ST LINE

MOV A, #84H

CALL CMDWRITE

S1: CLR A

MOVC A, @A+DPTR

JZ NEXT

ACALL DATAWRITE

;ACALL DELAY

INC DPTR

SJMP S1

NEXT:

;==========

MOV A,#0C0H ;MAKES NEW LINE

ACALL CMDWRITE

MOV DPTR,#SCORE\_MESSAGE

SCORE\_VIEW:

CLR A

MOVC A,@A+DPTR

JZ SCORE\_VALUE\_VIEW

ACALL DATAWRITE

INC DPTR

SJMP SCORE\_VIEW

SCORE\_VALUE\_VIEW:

CLR A ; CLEAR THE A

MOV A,R6 ; MOV THE SCORE TO THE ACUMLATOR

MOV B,#10;

DIV AB ;

ADD A,#30H ; CONVERT THE VALUE TO A CHARCTER

ACALL DATAWRITE;SEND THE TENS

MOV A,B

ADD A,#30H

ACALL DATAWRITE;SEND THE UNITS

; ADD A,#30H

;ADD A,R6

;ACALL DATAWRITE

;==========

EE:

RET

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

; SUBROUTINES TO ADJUST THE NIBBLES TO MATCH THE LCD DATA BUS

HIGH\_NIBBLE\_ADJUST:

ANL A, #11110000B ;MASK HIGHER NIBBLE

RR A

RR A

RET

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

LOW\_NIBBLE\_ADJUST:

RL A

RL A

RL A

RL A

ANL A, #11110000B ;MASK HIGHER NIBBLE

RR A

RR A

RET

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

CMDWRITE:

; THE DATA NEEDS TO BE ADJUSTED SO THE NIBBLE CAN BE PLACED IN THE 5, 4, 3, 2 OUT OF (7 ~ 0) BITS

MOV R0, A ;MAKE A COPY OF THE COMMAND

CALL HIGH\_NIBBLE\_ADJUST

MOV P1, A

CLR RS

SETB E

CALL DELAY

CLR E

;PREPARE THE LOWER NIBBLE BY SHIFTING

MOV A, R0

CALL LOW\_NIBBLE\_ADJUST

MOV P1, A

CLR RS

SETB E

CALL DELAY

CLR E

;CALL DELAY

RET

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

;;;; DATA WRITE SUBROUTINE FOR 4-BIT MODE;;;;

DATAWRITE:

MOV R0, A ;MAKE A COPY OF THE DATA

CALL HIGH\_NIBBLE\_ADJUST

MOV P1, A

SETB RS

SETB E

CALL DELAY

CLR E

; PREPARE THE LOWER NIBBLE BY SHIFTING

MOV A, R0

CALL LOW\_NIBBLE\_ADJUST

MOV P1, A

SETB RS

SETB E

CALL DELAY

CLR E

;CALL DELAY

RET

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

START\_TIMER:

MOV IE, #10000010B ;ENABLE EXTERNAL INTERRUPTS ET0, EX0

SETB IT0 ; NEGATIVE EDGE TRIGGER FOR INTERRUPT 0

MOV R2,#0 ; NUMBER OF OVERFLOWS

MOV R4,#9

MOV SEVEN\_SEGMENT,R4

MOV TMOD, #00000001B ; TIMER 0 AS 16-BIT TIMERS

;TIMER CLK = 11.0592/12\*1 = 0.9216 MHZ (1 CYCLE = 1.085.... µS)

; 50000 US = (65536 - COUNT) \* (1 / 0.9216). COUNT = 19456 => 4C00H

;OR 50000 US / (1 / 0.9216) US = 19456 (65536 - 19456 = 19456 => 4C00H)

MOV TH0,#4CH

MOV TL0,#00H

SETB TR0

RET

ISRT0:

; 1 SEC REQUIRES 20 OVERFLOW

;OVERFLOW TIMER INTERRUPT

CLR TR0

;RELOAD TIMERS

MOV TH0, #4CH

MOV TL0, #00H

;CLR TF0

CJNE R2 ,#20,NOT\_END\_YET

MOV R2,#0; RESTART OVERFLOW COUNTER

DEC R4

MOV SEVEN\_SEGMENT,R4

CJNE R4,#0,TIMER\_NOT\_FINISH

JMP END\_TIMER

TIMER\_NOT\_FINISH:

JMP RESTART\_TIMER

NOT\_END\_YET:

INC R2

;RUN AGAIN

RESTART\_TIMER: SETB TR0

END\_TIMER:

RETI

RESET\_TIMER:

CLR TR0

MOV R4,#9

MOV SEVEN\_SEGMENT,#9

RET

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

CHECK\_SCORE\_AFTER\_LOSS:

;WE ARE CHECKING ON R6

MOV A,R6

;{CJNE SRC,DEST} : (FIRST ADD SECOND) IF (FIRST > = SECOND) => [CARRYFLAG = 0] ELSE [CARRYFLAG = 1]

CJNE A,#12,CHECK1

ACALL TRY\_AGAIN\_PRINT

JMP LEVEL\_2 ;IF SCORE = 12

CHECK1: JC GOTO\_LEVEL\_1 ; A : SCORE # A < 12 CARRY FLAG WILL IS 1 # GO TO LEVEL 1

CJNE A,#20 ,CHECK2 ; IS SCORE LESS THAN 20 JMP TO LEVEL 2

ACALL TRY\_AGAIN\_PRINT

JMP LEVEL\_3 ; ELSE GO TO LEVEL 3

CHECK2: JC GOTO\_LEVEL\_2

CJNE A,#35,CHECK3

ACALL TRY\_AGAIN\_PRINT

JMP LEVEL\_4

CHECK3: JC GOTO\_LEVEL\_4

JMP LEVEL\_3 ;IF SCORE <25 BUT GREATER THAN 20

GOTO\_LEVEL\_1: RET

GOTO\_LEVEL\_2 : ACALL TRY\_AGAIN\_PRINT

JMP LEVEL\_2

GOTO\_LEVEL\_3 :ACALL TRY\_AGAIN\_PRINT

JMP LEVEL\_3

GOTO\_LEVEL\_4 : ACALL TRY\_AGAIN\_PRINT

JMP LEVEL\_4

RET

TRY\_AGAIN\_PRINT:

MOV A, #01H

ACALL CMDWRITE

;BEGIN AT 5TH COL, 1ST LINE

MOV A, #84H

ACALL CMDWRITE

MOV DPTR,#TRY\_AGAIN\_MESSAGE

M: CLR A

MOVC A,@A+DPTR

JZ Z

INC DPTR

ACALL DATAWRITE

SJMP M

Z:

RET

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

ORG 400H

LEVEL1: DB 01H, 02H, 04H, 08H

LEVEL2: DB 08H, 02H, 04H, 02H, 10H

LEVEL3: DB 04H, 08H, 80H, 20H, 01H

LEVEL4: DB 02H, 80H, 02H, 40H, 01H, 10H

LEVEL5: DB 10H, 08H, 01H, 08H, 04H, 02H

LEVEL6: DB 04H, 20H, 10H, 02H, 80H, 04H

LEVEL7: DB 20H, 80H, 08H, 02H, 10H, 08H

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

LCD\_L1 : DB "LEVEL 1",0

LCD\_L2 : DB "LEVEL 2",0

LCD\_L3 : DB "LEVEL 3",0

LCD\_L4 : DB "LEVEL 4",0

LCD\_L5 : DB "LEVEL 5",0

LCD\_L6 : DB "LEVEL 6",0

LCD\_L7 : DB "LEVEL 7",0

PASS: DB "PASSED",0

WON: DB "YOU WON",0

LOST: DB "GAME OVER!",0

SCORE\_MESSAGE: DB "SCORE:",0

TRY\_AGAIN\_MESSAGE: DB "TRY AGAIN",0

END

References:

* LCD 4 BIT lab
* Timer Interrupt lab