EE 306 - MICROPROCESSORS

TERM PROJECT REPORT

CYCLONE GAME

Instructor: Tuba Ayhan
Institution: MEF University
Date: 08.06.2020

Berfin Elif Erdoğan 041701005
Ezgi Nihal Subaşı 041701022
Mert Kömürcüoğlu 041402020

Explanation Of The Game

Our project is called Cyclone Game which can be played by two players. There are 10 LEDs and only one LED is on at any time. In the beginning of the game, players determine the LEDs shifting speed by pressing a button between 0 and 3. Then the game starts with setting the LED0 on and it shifts through right to left continuously. In order to gain score players have to press their corresponding push buttons when LED9 is on. If one player or both players scored 9 times, a 7-segment display shows the winner/s and total score in a half second loop.

Initially address of location of LEDs loads to the R0, timer loads to the R1, push buttons load to the R3, and seven-segment display loads to the R5 registers. The “SPEED0-3” message is shown in 7-segment display by loading the values that are read from speed_Check to indicate that users are able to choose the shifting speed of LEDs and to determine the level of the game. The timer is set as 1, 0.5, 0.25, or 0.125 seconds according to the input received from push buttons. After setting the timer properly, segmentCode loads into R10 for number representations of 7-segment display. R8 and R9 registers hold the score of the players while R11 and R12 registers point the R10 separately.

The game officially starts with displaying scores of the players and light on the LEDs in order. DISPLAY subroutine first clears the 4 left digits, then loads the scores of the players. Finally, it adds the registers by left shifting one of them in order to display scores side by side. Maintaining the lighting order and checking the keys provided by Mainloop. Innerloop controls the status register of the timer until it's equal to 1. When it's 1, the program resets the status register then left shifts the R2 by one to shift the LEDs. When 8th led is on, the value in the R2 becomes 512 then the program resets the push button’s edge-capture register to get the input when 9th LED is on. When R2's value becomes 1024, led is on 9th led, therefore, KEY_CHECK checks if any players pushed their own button.

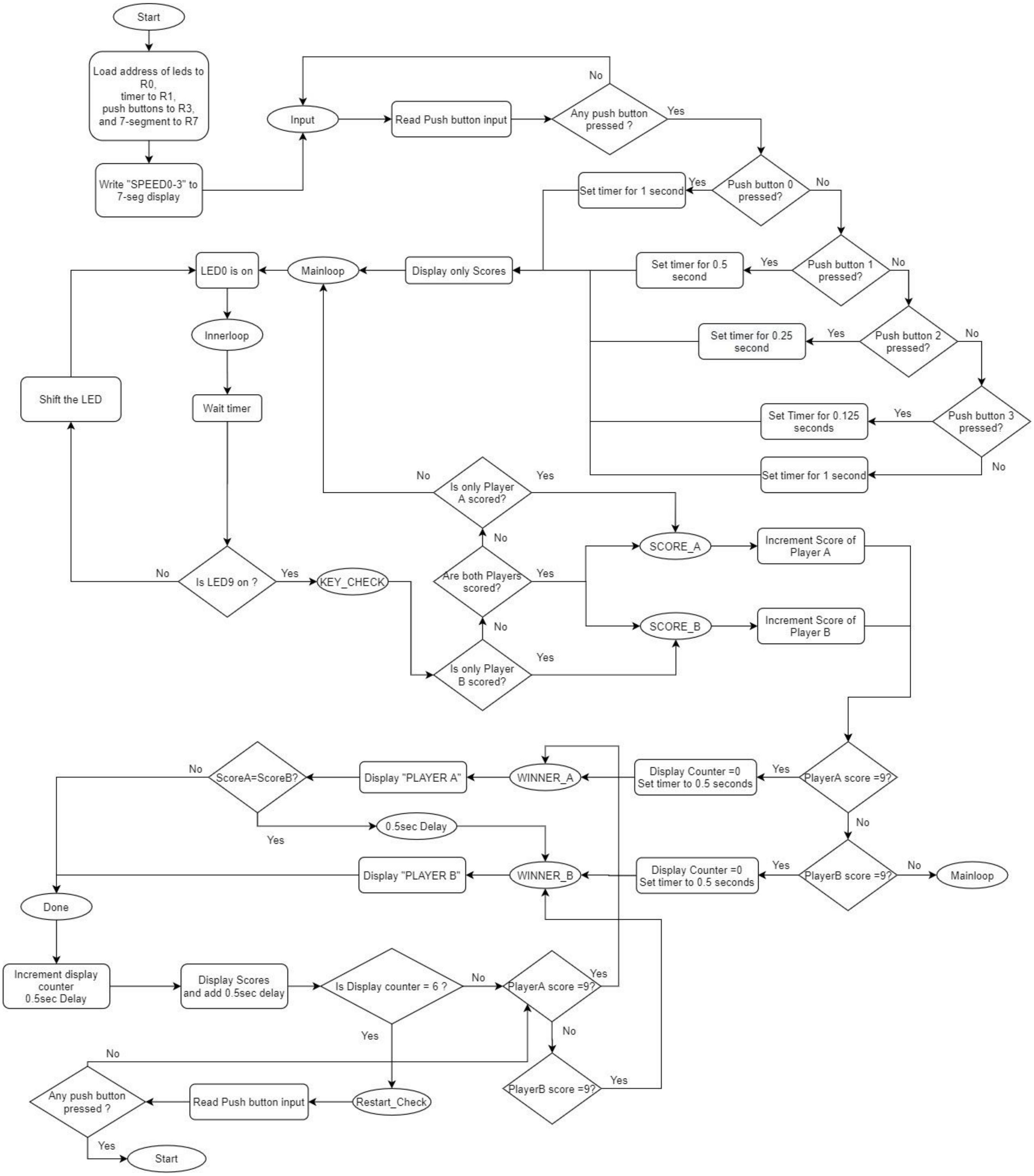
KEY_CHECK subroutine checks the value of the push button’s edge-capture register in order to determine if any button is pushed or not. Which player clicks the button, the score of that player increases by SCORE_A/B subroutines. After displaying the score, KEY_CHECK also checks for if any player wins by comparing R8 and R9 with #9. In the winner case, if at least one of the players won, the program moves 0 to R6 for displaying the winner in 8 cycles. Also calling the Set_Timer subroutine to set the timer for 0.5 seconds which will be used to the “PLAYER A/B” “SCORE” loop.

When player A wins, “PLAYER A” message is displayed via 7-segment display to indicate that Player A won by WINNER_A. Since R8 and R9 hold the scores of players, checking if these two registers holds the same value in case of a tie situation. If there isn’t a tie, the program goes to the “Done” function. In the case of a tie, calling DELAY subroutine to have a 0.5 seconds delay before going to WINNER_B. In WINNER_B “PLAYER B” is displayed via 7-segment display to indicate Player B won the game. Then the program goes to the “Done” function.

In Done, the display counter is incremented by 1 and DELAY subroutine is called to have 0.5 seconds delay before calling DISPLAY subroutine to display the scores. Adding another DELAY subroutine to have a minimal delay before calling Restart_Check subroutine. When the program is back from Restart_Check, compares R6 with 8 before it goes to proper WINNER_A/B functions. If R6 is equal to 8, the program clears the push button’s edge-capture register to be able to read new inputs. If R6 is greater than 8 in each cycle calling the Restart_Check subroutine.Then the program goes back to proper WINNER_A/B, if there aren't any new push button inputs.

In Restart_Check reading the edge-capture register of the push button to see if there is any new input after the reset. If there is a new input, the game restarts.

Flowchart


Start
Load address of leds to R0, timer to R1, push buttons to R3, and 7-segment to R7
Write "SPEED0-3" to 7-seg display
Input
Read Push button input
Any push button pressed ?
No
Yes
Push button 0 pressed?
Yes
No
Set timer for 1 second
Push button 1 pressed?
Set timer for 0.5 second
Push button 2 pressed?
Set timer for 0.25 second
Push button 3 pressed?
Set Timer for 0.125 seconds
Set timer for 1 second
Display only Scores
Mainloop
LED0 is on
Innerloop
Wait timer
Shift the LED
Is LED9 on ?
KEY_CHECK
Is only Player B scored?
Are both Players scored?
Is only Player A scored?
SCORE_A
SCORE_B
Increment Score of Player A
Increment Score of Player B
PlayerA score =9?
PlayerB score =9?
Display Counter =0 Set timer to 0.5 seconds
Mainloop
WINNER_A
WINNER_B
Display "PLAYER A"
Display "PLAYER B"
ScoreA=ScoreB?
0.5sec Delay
Done
Increment display counter 0.5sec Delay
Display Scores and add 0.5sec delay
Is Display counter = 6 ?
Restart_Check
Read Push button input
Any push button pressed ?
Start