MEF UNIVERSITY

YOUR FREEDOM IN LEARNING

EE306 - Microprocessors

Term Project
Sliding Text

June 8, 2020

Ali Bahadır Bulut
#041602017

Alp Gokcek
#041701014

Erdal Sidal Dogan
#041702023

1 Project Description

In this project, the text has been shown by scrolling them on a 7-segment display. Which text to print on 7-segment displays has been chosen by using switches and has been adjusted the scroll speed of texts with the buttons. The main goal has been that this project was done by using the peripherals on the board. However, the project has been done via the internet browser using the CPUlator ARMv7 System Simulator. The advantage of using this simulation site was to use eight 7-segment displays. ASCII codes have been used to describe letters and numbers in this project. ASCII equivalent of the characters has been converted to hexadecimal and implemented in the project. Additionally, A9 private timer has been used as a timer in the project.

Our design supports the characters provided in Figure 1.

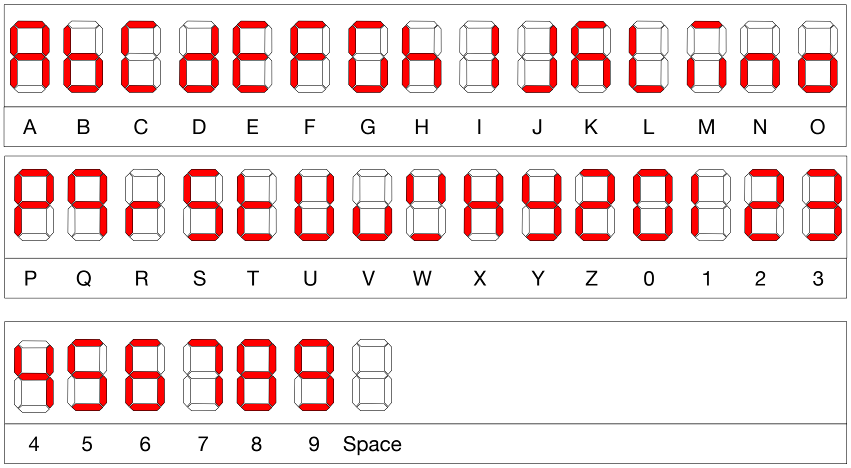


Figure 1: Sample output on 7-segment displays for input ”EE306 Fun”

2 Sample Output

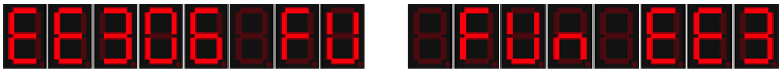


Figure 2: Sample output on 7-segment displays for input ”EE306 Fun”

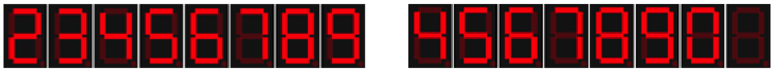


Figure 3: Sample output on 7-segment displays for input ”1234567890”

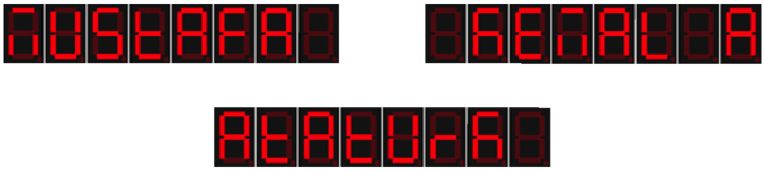


Figure 4: Sample output on 7-segment displays for input ”Mustafa Kemal Ataturk”

3 Flowcharts

3.1 MAIN

After the initialization, the program starts executing the *Main* subroutine. It consists of chaining other subroutines sequentially such as *SET_WORD*, *CHECK_PAUSE*, *SET_SPEED*, *PRINT_STRING*.

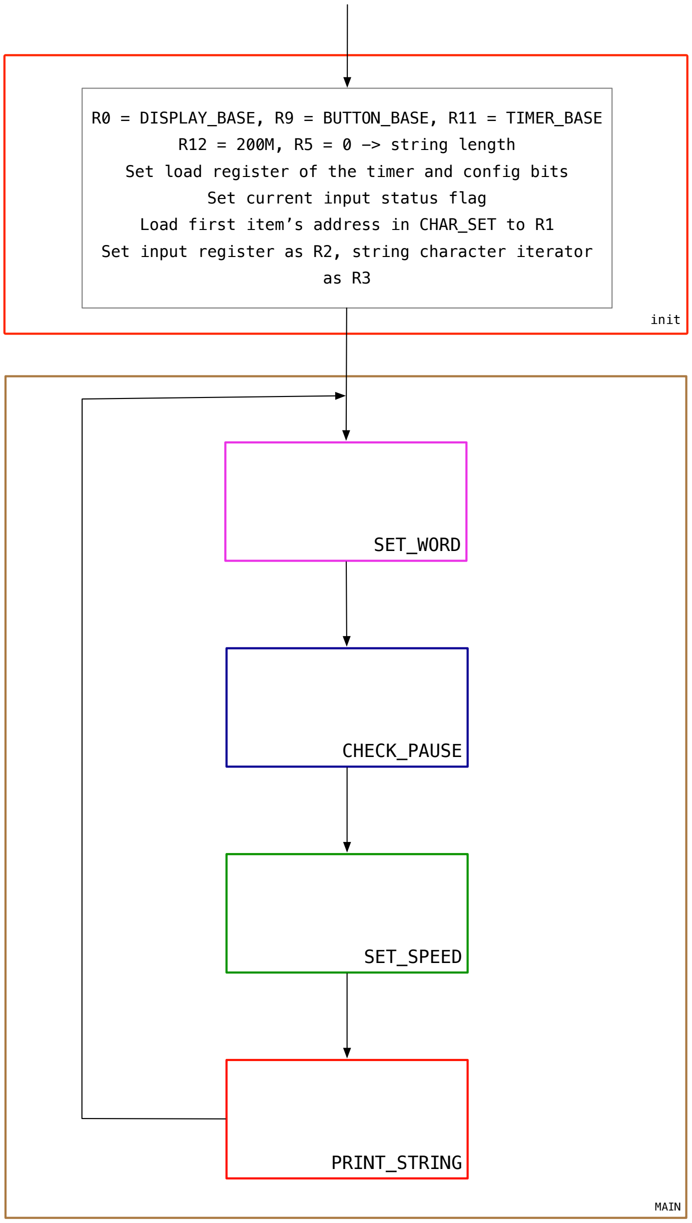
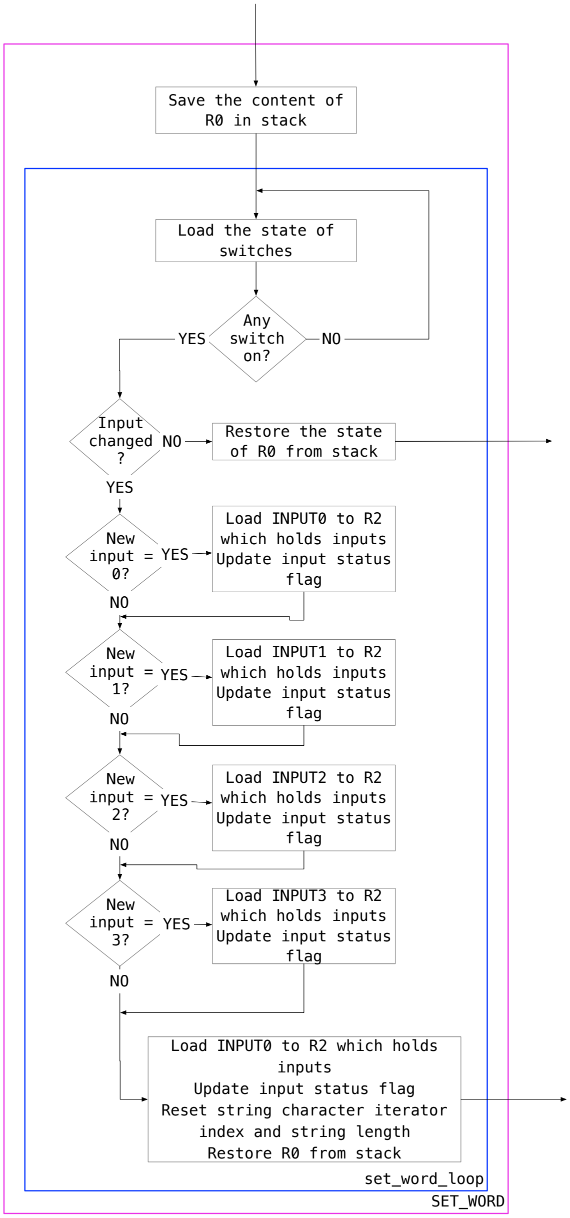

R0 = DISPLAY_BASE, R9 = BUTTON_BASE, R11 = TIMER_BASE
R12 = 200M, R5 = 0 -> string length
Set load register of the timer and config bits
Set current input status flag
Load first item's address in CHAR_SET to R1
Set input register as R2, string character iterator as R3
init
SET_WORD
CHECK_PAUSE
SET_SPEED
PRINT_STRING
MAIN

Figure 5: Flowchart of Main Loop

3.2 SET_WORD

The program supports multiple words. Selection is being made by the 'Switch' input. The subroutine *SET_WORD* is responsible for choosing the correct word according to the input.



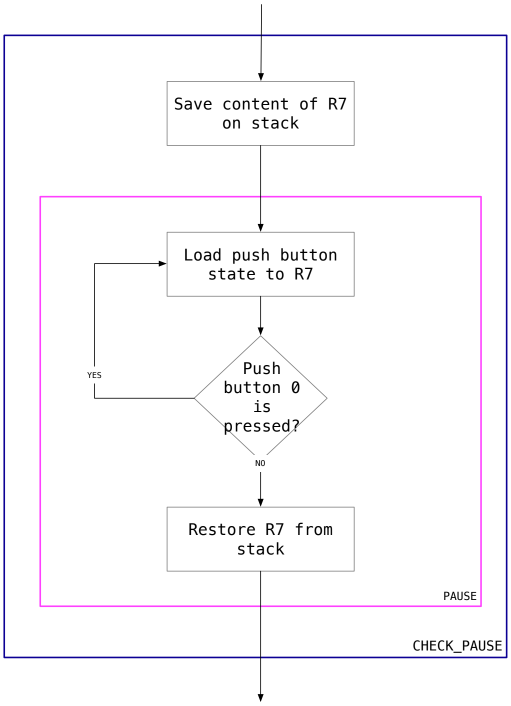
Save the content of R0 in stack
Load the state of switches
Any switch on?
YES
NO
Input changed ?
NO
Restore the state of R0 from stack
YES
New input = 0?
YES
Load INPUT0 to R2 which holds inputs Update input status flag
NO
New input = 1?
YES
Load INPUT1 to R2 which holds inputs Update input status flag
NO
New input = 2?
YES
Load INPUT2 to R2 which holds inputs Update input status flag
NO
New input = 3?
YES
Load INPUT3 to R2 which holds inputs Update input status flag
NO
Load INPUT0 to R2 which holds inputs Update input status flag Reset string character iterator index and string length Restore R0 from stack
set_word_loop
SET_WORD

Figure 6: Flowchart of Set Word Subroutine

3.3 SET_SPEED & CHECK_PAUSE

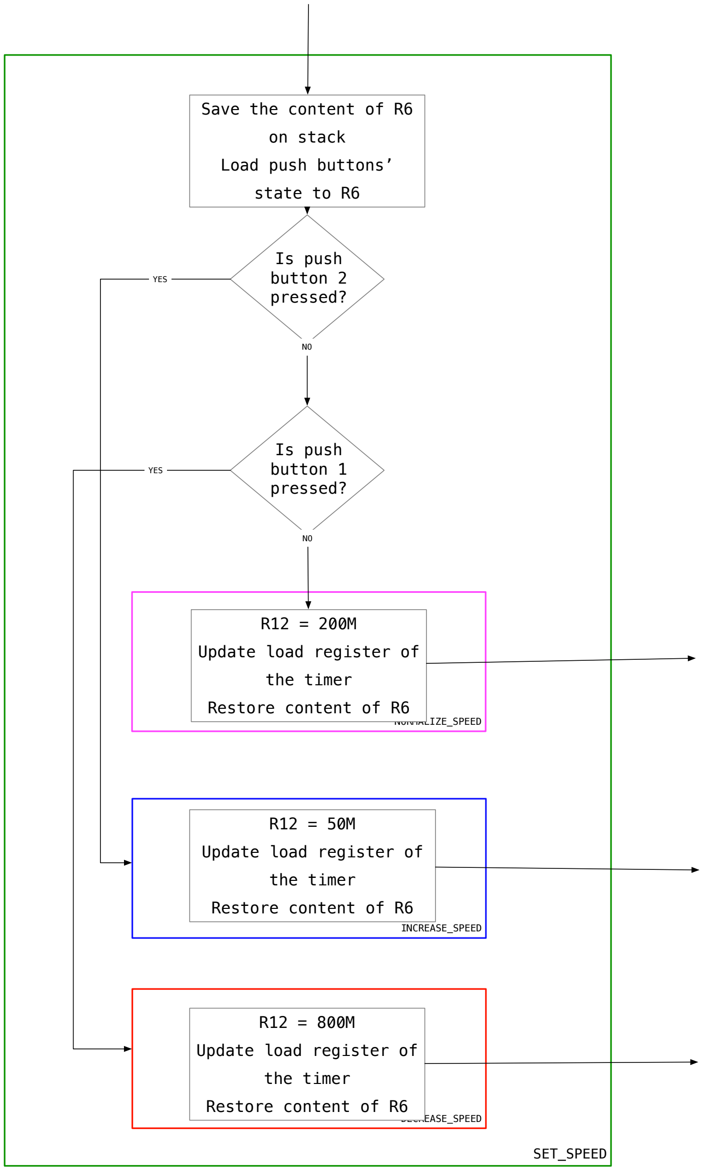
As the names suggests, *SET_SPEED* and *CHECK_PAUSE* subroutines are pausing the slider and sets its speed. Both of these functions are triggered by certain combinations of push buttons. They check the latest push button configuration pause or increase/decrease the speed accordingly. There are 3 different predefined levels of speed;

- Normal: 1 sec
- Fast: .25 sec
- Slow: 4 sec



Save content of R7 on stack
Load push button state to R7
Push button 0 is pressed?
YES
NO
Restore R7 from stack
PAUSE
CHECK_PAUSE

Figure 7: Flowchart of Check Pause Subroutine



Save the content of R6 on stack
Load push buttons' state to R6
Is push button 2 pressed?
YES
NO
Is push button 1 pressed?
YES
NO
R12 = 200M
Update load register of the timer
Restore content of R6
LIZE_SPEED
R12 = 50M
Update load register of the timer
Restore content of R6
INCREASE_SPEED
R12 = 800M
Update load register of the timer
Restore content of R6
EASE_SPEED
SET_SPEED

Figure 8: Flowchart of Set Speed Subroutine

3.4 PRINT_STRING & DELAY & FIND_LOC

3.4.1 DELAY

The A9 Private Timer is being used for delays. This subroutine checks the timer configuration continuously until it finishes its countdown, after it finished this subroutine resets it and start over again when called next time.

3.4.2 PRINT_STRING

This subroutine determines which letter to print relying on the input and which 7segment to print it.

3.4.3 FIND_LOC

The representation of possible inputs, which are capital and lowercase letters along with numbers, are stored sequentially in the memory. This subroutine is used for locating the correct 7-segment representation in the memory.

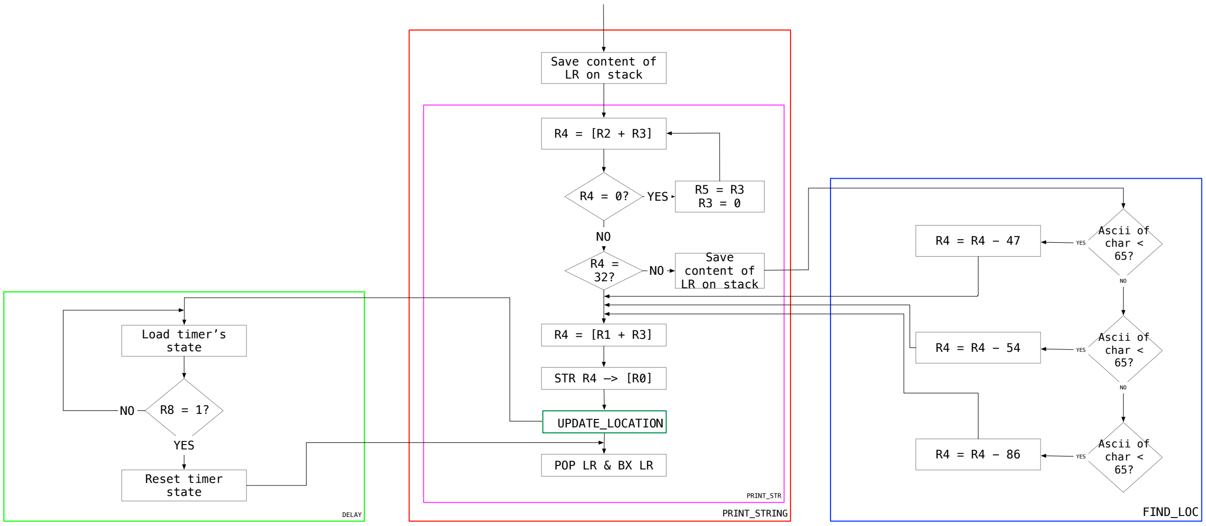


Figure 9: Flowchart of Print String Subroutine

3.4.4 UPDATE_LOCATION

This subroutine determines which 7-segment to write next character. Given that there are jumps between memory locations of 7-segments and circular loop, it has been decided that its best to handle these operations in a different subroutine. First we set it to be the 7-segment that is right next to the previous one. Then we check if has exceeded any limits, if it did, reset the position accordingly.

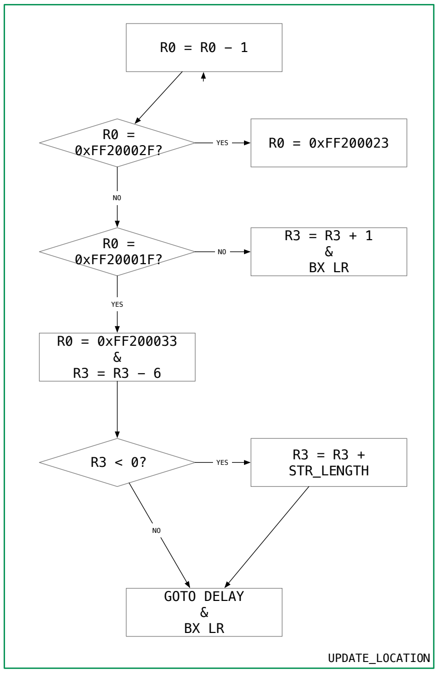


Figure 10: Flowchart of Update Location Subroutine