

## ACS. Homework 5.

Горбачева Маргарита Валерьевна | БПИ-245

**О самом ТЗ:** написать программу, которая будет выводить на экран бегущие цифры от 0 до 9 и бегущие буквы (привет шестнадцатеричной системе счисления) от A до F.

**Работа программы:** у меня очень долго не получалось написать программу, которая будет выводить цифры и буквы на экран, либо программа ломалась на цифре 0, либо вообще не работала. Поэтому после многих попыток написать что-то работающее, я создала программу, которая запускает цифру или букву, которую мы можем указать в окошке Digital Sim Lab.

Вот так выглядела эта программа:

```
riscv2.asm
1  .text
2  .globl main
3
4  main:
5      # Базовый адрес MMIO для Digital Lab Sim
6      lui a0, 0xffff0
7
8      # Тест: выводим разные цифры на оба индикатора
9      li t0, 0x3F # Цифра 0
10     sb t0, 0x10(a0) # Правый индикатор
11
12     li t0, 0x06 # Цифра 1
13     sb t0, 0x11(a0) # Левый индикатор
14
15     # Бесконечный цикл чтобы программа не завершилась
16 loop:
17     j loop
```

После у меня получилось

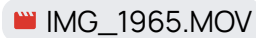
собрать такую программу,

которая в памяти хранит список, где указывается, какие индикаторы для какой цифры или буквы должны гореть. Например, для цифры "1" код 0x06 означает правые два сегмента. В программу записывается адрес индикатора, далее мы инициализируем счётчик, сначала он, логично, равен 0. Запускаем наш бесконечный цикл, идея такова: смотрим в секцию даты, посылаем оттуда код на индикатор, тогда на часах появляются новые сегменты, далее увеличивается счетчик и все происходит заново.

```
riscv1.asm
1  .data
2  digit_codes:
3      .byte 0x3F # 0 начиная отсюда и вниз, это все сегменты, которые отвечают за циферки на экране
4      .byte 0x06 # 1
5      .byte 0x5B # 2
6      .byte 0x4F # 3
7      .byte 0x66 # 4
8      .byte 0x6D # 5
9      .byte 0x7D # 6
10     .byte 0x07 # 7
11     .byte 0x7F # 8
12     .byte 0x6F # 9
13     .byte 0x77 # A
14     .byte 0x7C # b
15     .byte 0x39 # c
16     .byte 0x5E # d
17     .byte 0x79 # E
18     .byte 0x71 # F
19
20     .text #секция кода
21     .globl main #глобальная метка
22
23     main:
24         li s0, 0xffff0010 #загружаем в регистр s0 адрес индикатора
25         #Byte value at address 0xffff0010 : command right seven segment display
26         #s0 теперь содержит адрес устройства вывода
```

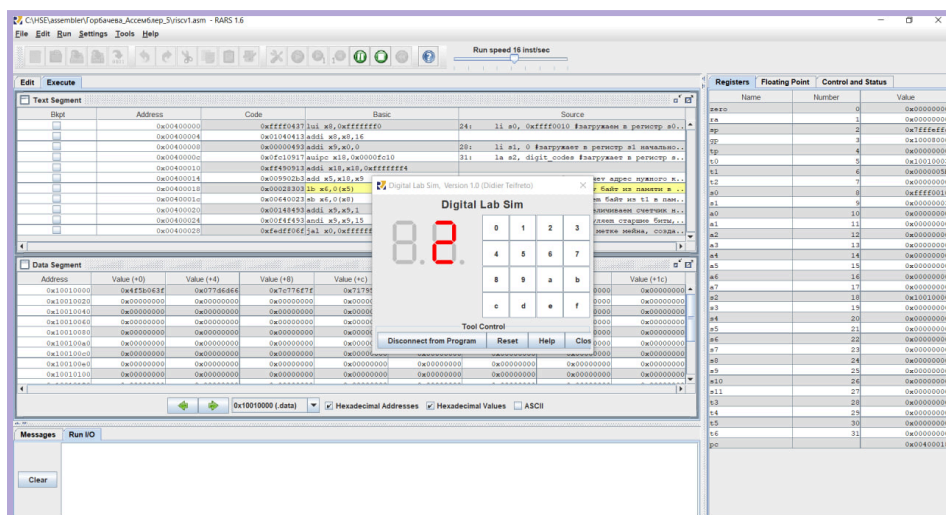
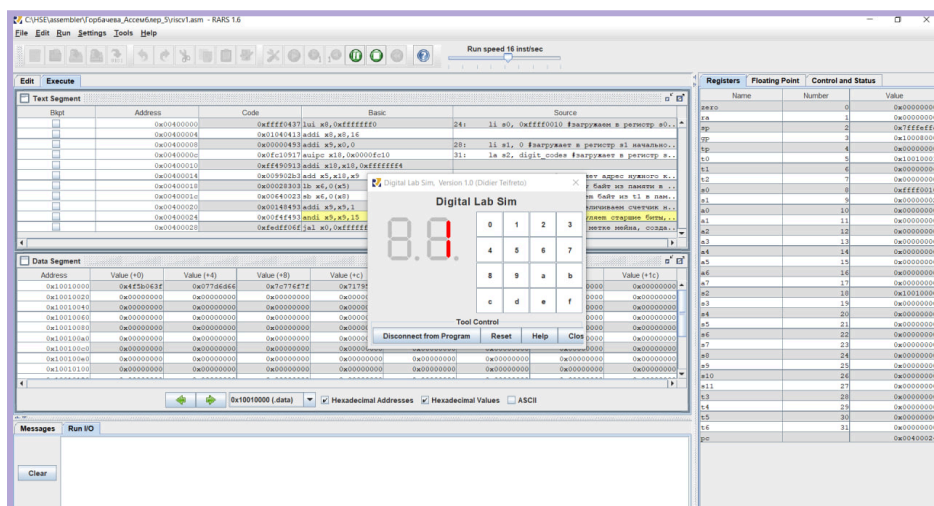
**Ссылка на видео-обзор программы:**

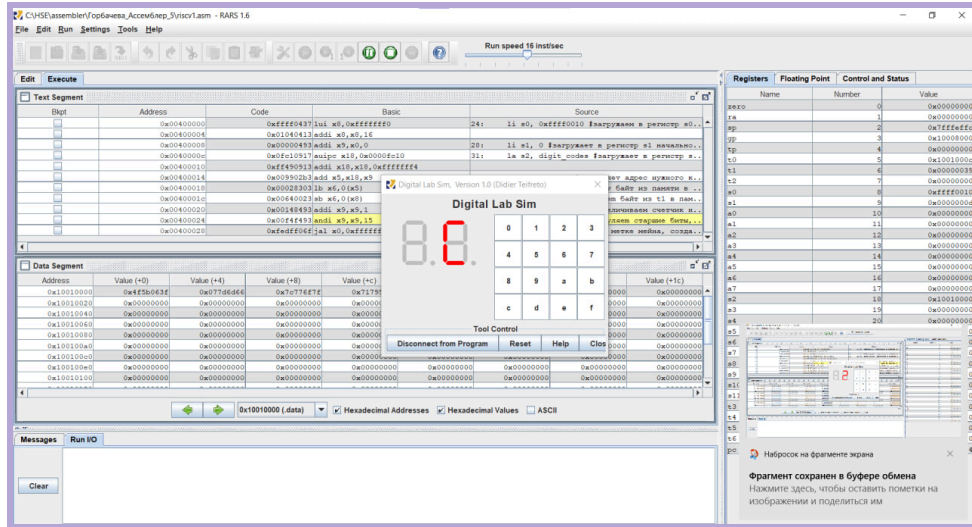
Для наглядности работы программы было записано видео, правда не записью экрана, а записью на телефон, тк Xbox на Windows 10 решил меня подставить...



[https://drive.google.com/file/d/1Tn4ZdHdGtKp6x77OzqHrBolE\\_pQoRenq/view?usp=drive\\_li nk](https://drive.google.com/file/d/1Tn4ZdHdGtKp6x77OzqHrBolE_pQoRenq/view?usp=drive_li nk) Вот ссылка на данное видео, но если по ссылке оно не откроется, в папке .zip оно спокойно лежит. Видео хорошо и наглядно демонстрирует, как по экрану бегают циферки.

## Скриншоты программы (на всякий случай)





**САМОЕ ГЛАВНОЕ - СНИЗИТЬ СКОРОСТЬ ВЫПОЛНЕНИЯ ПРОГРАММЫ!!!** Несколько раз я об этом забывала и думала, что программа ломалась, однако цифры просто бежали слишком быстро для человеческих глаз.