

**Calculator using PIC16F887 MCU**

Computers and computerized systems in automatics

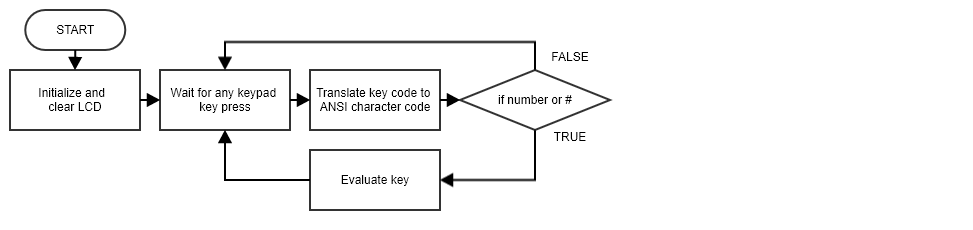
Task 2

Done by: EKSfmu-16 gr. st. Arūnas Butkus

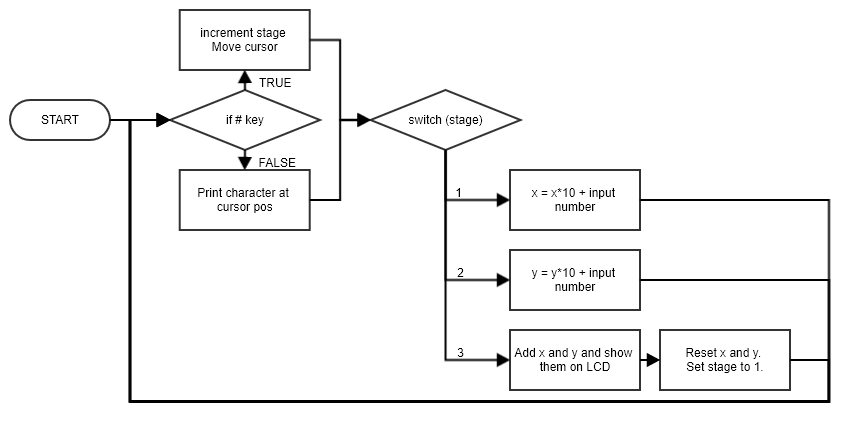
Checked by: prof. Algirdas Baškys

**Work goal:** create a calculator program for PIC16F887 using C language which would be able to add two numbers and show them and the result on LCD screen.

Algorithms used in tasks.



**Fig. 1.** Full program algorithm loop.



**Fig. 2.** Key evaluation algorithm.

Results analysis.

Building upon things achieved in Task 1 main issue here is design of effective algorithm for number input and display. I came up with a system using stages. First stage is variable x input. Second stage is variable y input. Third stage adds them, displays them and resets system to stage 1. I will not go into details here much because I find the algorithm (Fig. 2) quite self‑explanatory. And source code is easily readable too.

As far as result images are concerned, I forgot to take any, but row one of LCD shows labels and values for x and y; row two – shows label and value for sum of said values.

Conclusions.

Not all input cases are accounted for and erroneous results/display could be noticed in cases, but knowing how to use this program and adhering to the limitation set in task, program works.

Source code.

001 /\*

002 Sudarykite kalkuliatoriaus programa C kalba mikrovaldikliui

003 PIC16F887, kuri sumuotu du skaicius ir sumuojamus skaicius

004 bei atsakyma atvaizduotu LCD ekrane. Maksimalus sumuojamas

005 skaicius 255. Skaiciu ivedimui panaudokite EasyPIC 6

006 stende esancia klaviatura 4x4.

007 \*/

008

009 // LCD connection

010 sbit LCD\_RS at RB4\_bit**;**

011 sbit LCD\_EN at RB5\_bit**;**

012 sbit LCD\_D4 at RB0\_bit**;**

013 sbit LCD\_D5 at RB1\_bit**;**

014 sbit LCD\_D6 at RB2\_bit**;**

015 sbit LCD\_D7 at RB3\_bit**;**

016

017 sbit LCD\_RS\_Direction at TRISB4\_bit**;**

018 sbit LCD\_EN\_Direction at TRISB5\_bit**;**

019 sbit LCD\_D4\_Direction at TRISB0\_bit**;**

020 sbit LCD\_D5\_Direction at TRISB1\_bit**;**

021 sbit LCD\_D6\_Direction at TRISB2\_bit**;**

022 sbit LCD\_D7\_Direction at TRISB3\_bit**;**

023

024 // keypad connection

025 char keypadPort at PORTD**;**

026

027 unsigned short

028 kp**,**

029 wasStage **=** 1**,**

030 stage **=** 1**,**

031 x **=** 0**,**

032 y **=** 0**,**

033 sum **=** 0**,**

034 val**;**

035 char txt**[**3**];**

036

037 // function prototypes

038 void LCD\_move\_cursor**();**

039 void LCD\_inputs\_reset**();**

040 void LCD\_sum\_reset**();**

041

042 void main**()** **{**

043 Keypad\_Init**();** // keypad init

044 ANSEL **=** 0**;** // make I/O digital

045 ANSELH **=** 0**;**

046

047 // LCD setup

048 Lcd\_Init**();**

049 Lcd\_Cmd**(**\_LCD\_CLEAR**);**

050 Lcd\_Out**(**1**,**9**,**"y = "**);**

051 Lcd\_Out**(**2**,**1**,**"x+y = "**);**

052 Lcd\_Out**(**1**,**1**,**"x = "**);**

053

054 **while** **(**1**)** **{**

055 kp **=** 0**;** // reset key press variable

056

057 // wait for keypress

058 **while** **(!**kp**)** **{**

059 kp **=** Keypad\_Key\_Click**();**

060 **}**

061

062 // interpret the keypress

063 **switch** **(**kp**)** **{**

064 **case** 1**:** kp **=** 49**;** val **=** 1**;** **break;** // 1

065 **case** 2**:** kp **=** 50**;** val **=** 2**;** **break;** // 2

066 **case** 3**:** kp **=** 51**;** val **=** 3**;** **break;** // 3

067 **case** 4**:** kp **=** 65**;** **continue;** **break;** // A

068 **case** 5**:** kp **=** 52**;** val **=** 4**;** **break;** // 4

069 **case** 6**:** kp **=** 53**;** val **=** 5**;** **break;** // 5

070 **case** 7**:** kp **=** 54**;** val **=** 6**;** **break;** // 6

071 **case** 8**:** kp **=** 66**;** **continue;** **break;** // B

072 **case** 9**:** kp **=** 55**;** val **=** 7**;** **break;** // 7

073 **case** 10**:** kp **=** 56**;** val **=** 8**;** **break;** // 8

074 **case** 11**:** kp **=** 57**;** val **=** 9**;** **break;** // 9

075 **case** 12**:** kp **=** 67**;** **continue;** **break;** // C

076 **case** 13**:** kp **=** 42**;** **continue;** **break;** // \*

077 **case** 14**:** kp **=** 48**;** val **=** 0**;** **break;** // 0

078 **case** 15**:** kp **=** 35**;** val **=** 0**;** **break;** // #

079 **case** 16**:** kp **=** 68**;** **continue;** **break;** // D

080 **}**

081

082 **if** **(**kp **==** 35**)** **{** // if # then next stage

083 stage**++;**

084 LCD\_move\_cursor**();**

085 **}** **else** **{**

086 **if** **(**kp **>** 47 **&&** kp **<** 58**)** **{** // if number pressed then write it down

087 Lcd\_Chr\_CP**(**kp**);**

088 **}**

089 **}**

090

091 // update values

092 **switch** **(**stage**)** **{**

093 **case** 1**:** // x input stage

094 x **=** x **\*** 10 **+** val**;**

095 **break;**

096 **case** 2**:** // y input stage

097 y **=** y **\*** 10 **+** val**;**

098 **break;**

099 **case** 3**:** // calculation stage

100 LCD\_sum\_reset**();**

101 LCD\_move\_cursor**();**

102 val **=** x**+**y**;**

103 WordToStr**(**val**,** txt**);**

104 Lcd\_Out\_CP**(**txt**);**

105 x **=** 0**;**

106 y **=** 0**;**

107 stage **=** 1**;**

108 LCD\_inputs\_reset**();**

109 LCD\_move\_cursor**();**

110 **break;**

111 **}**

112 **}**

113 **}**

114

115 void LCD\_move\_cursor**(){**

116 **switch** **(**stage**)** **{**

117 **case** 1**:** Lcd\_Chr**(**1**,**4**,**32**);** **break;**

118 **case** 2**:** Lcd\_Chr**(**1**,**12**,**32**);** **break;**

119 **case** 3**:** Lcd\_Chr**(**2**,**6**,**32**);** **break;**

120 **}**

121 **}**

122

123 void LCD\_inputs\_reset**(){**

124 Lcd\_Out**(**1**,**1**,**"x = "**);**

125 Lcd\_Out**(**1**,**9**,**"y = "**);**

126 **}**

127

128 void LCD\_sum\_reset**(){**

129 Lcd\_Out**(**2**,**1**,**"x+y = "**);**

130 **}**

Electrical Schemes

