

**UNIVERSITETI I PRISHTINËS “HASAN PRISHTINA”
FAKULTETI I INXHINIERISË ELEKTRIKE DHE KOMPJUTERIKE**



**RAPORTI I PROJEKTIT
LËNDA MIKROPROCESOR&MIKROKONTROLLER**

**VOLTMETËR DIGITAL ME ZGJEDHJE AUTOMATIKE/MANUALE TË
BREZIT MATËS**

Profesor,

Lavdim Kurtaj

Student,

Denis Osmani

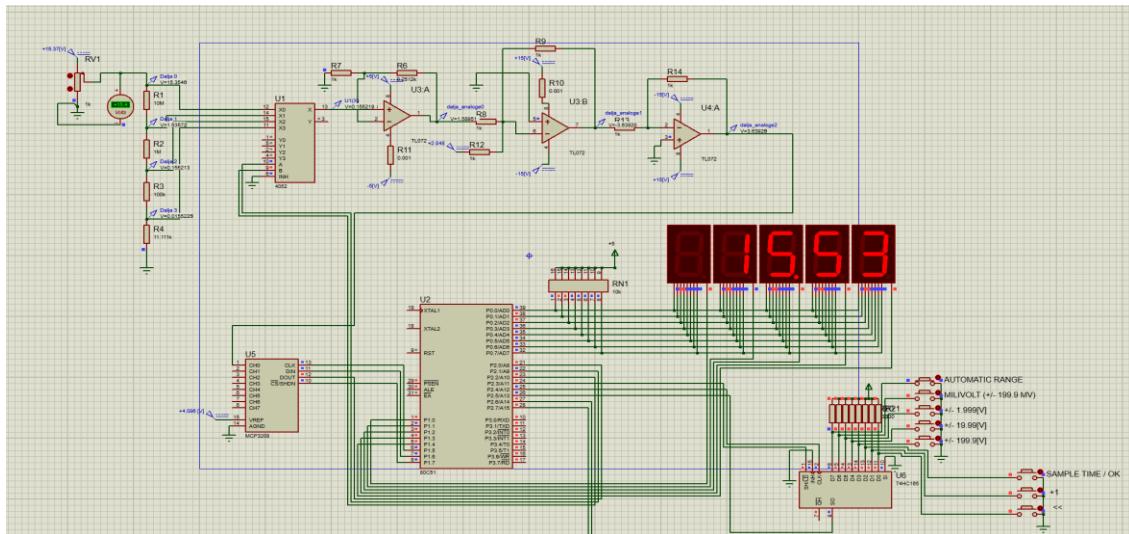
Shkurt, 2023

Përbajtja

Hyrje	2
Programet e përdorura	2
Pajisjet e përdorura.....	2
Ndarësi potenciometrik.....	3
Multiplekseri analog	4
Shndërruesi i brezit	5
Konverteri A/D	7
Përpunimi i të dhënave të konverterit A/D	11
Ekstraktimi i shifrave	14
Displeji i multipleksuar.....	16
Zgjedhja automatike e brezit.....	19
Tastaura me shiftregjistër.....	22
Zgjedhja manuale e brezit	23
Organizimi i programit në main.....	25
Koha e mostrimit.....	27
Kodi në assembly	31

Hyrje

Në këtë projekt realizohet voltmetri digital, me brez të matjes +/- 200[V], i bazuar në mikrokontrollerin 8051.



Programet e përdorura

MCU 8051 IDE

Proteus v8.11

Pajisjet e përdorura

80C51

7SEG-CC

74HC165 – shiftregjistër

4052 – multiplekser analog

TL072 – amplifikator operacional

MCP3208 – konverter A/D

Rezistorë

Potenciometër

FM24C256 – EEPROM/I2C

Ndarësi potenciometrik

Vlerat e tensionit mund të jenë të larta kështuqë nevojitet një qark dobësues i tensionit si stadi hyrës i pajisjes në mënyre që vlerat e tensionit të jenë të përshtatshme për përpunim nga pjesa digitale. Për këtë qëllim përdoret ndarësi i tensionit, apo ndarësi potenciometrik.

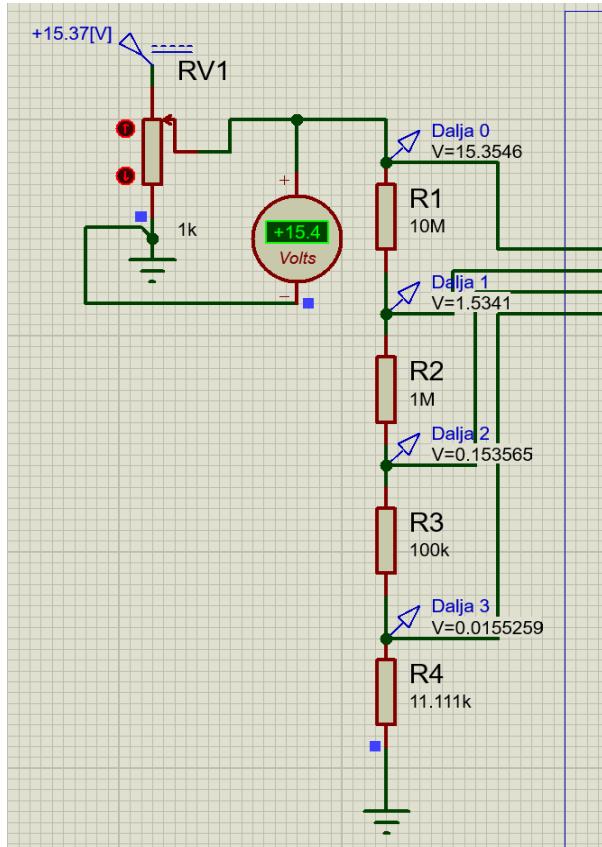


Fig. 1 – Ndarësi potenciometrik

Vlerat e tensionit në dalje varen nga tensioni në hyrje dhe nga vlerat e rezistorëve të vendosur. Llogaritja e vlerave të tensionit mund të bëhet duke e zgjidhur qarkun me metoda klasike, duke e përdorur ligjin e parë dhe të dytë të Kirchoffit. Në rastin më të thjeshtë me dy resistor, përcaktohet sipas relacionit më poshtë,

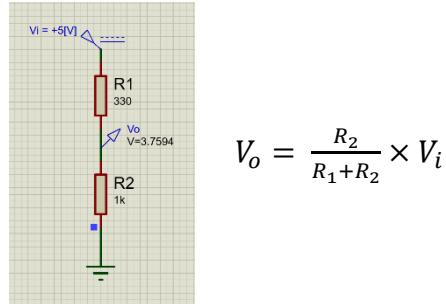


Fig. 2 – Ndarësi i tensionit me 2 rezistorë

Për rastin me n-rezistorë $V_x = \frac{R_x}{R_1+R_2 + \dots + R_n} \times V_i$. Në rastin konkret numri i rezistorëve është përzgjedhur në atë mënyrë që t'i kemi 4 dalje, ndërsa vlerat e rezistorëve janë përzgjedhur në atë mënyrë që ndarja e tensionit të jetë $1V_i$, $\frac{1}{10}V_i$, $\frac{1}{100}V_i$, $\frac{1}{1000}V_i$, në daljet 0,1,2,3 përkatësisht, sikurse shihet edhe në figurën 1.

Multiplekseri analog

Multiplekseri është një pajisje që mundëson zgjedhjen e një prej shumë hyrjeve dhe e përcjellë në një dalje.

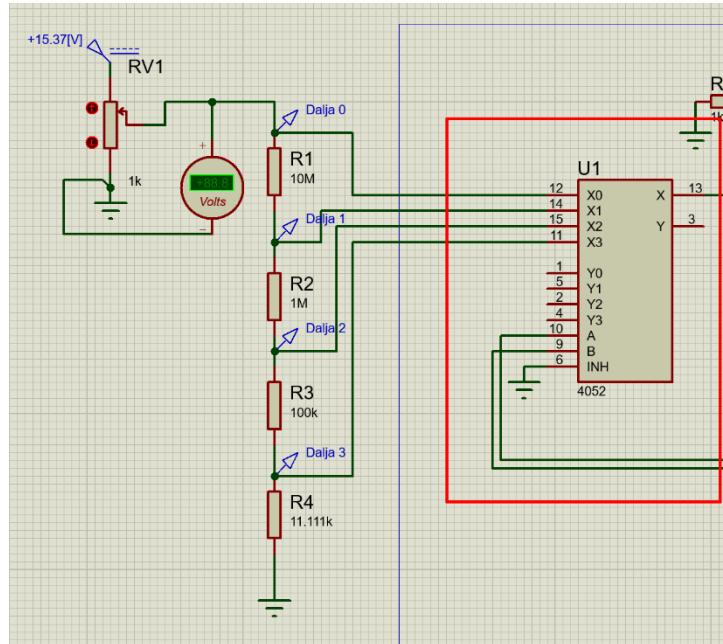


Fig. 3 – Multiplekseri Analog

Në rastin konkret, multiplekseri i ka 4 hyrje X0, X1, X2 dhe X3, ndërsa sinjali i cilës hyrje do të përcjellët në daljen X, varet nga vlera e logjike që dërgohen në pinat B dhe A të multiplekserit nga mikrokontrolleri 8051.

```

8;-----ANALOG MULTIPLEXER-----
9 OUTSA EQU 0A1H ; A pin of analog mux, 0A1H = P2.1
10 OUTSB EQU 0A2H ; B pin of analog mux 0A2H = P2.2
11;

```

Fig. 4 – Pinat e 8051 që lidhen me multiplekser

Nëse BA = 00, selektohet Dalja 0, nëse BA = 01, selektohet Dalja 1, nëse BA = 10, selektohet Dalja 2, nëse BA = 11, selektohet Dalja 3. Kur INH është 0, lejohet përcjellja e njërsës hyrje në dalje, kur INH = 1, dalja X kalon në gjendje të impedancës së lartë dhe më nuk kanë kontakt elektrik me hyrjet, rrjedhimisht pajisja nuk funksionon. Ngjashëm vlen edhe për hyrjet Yn dhe daljen Y.

TRUTH TABLE			
Control Inputs		ON Switches	
Inhibit	Select		
	B	A	
0	0	0	Y0 X0
0	0	1	Y1 X1
0	1	0	Y2 X2
0	1	1	Y3 X3
1	X	X	None

* X=Don't Care

Fig. 5 - Tabela e vërtetësisë për 4052

Shndërruesi i brezit

Secila prej daljeve të ndarësit potenciometrik përdoret për një brez të caktuar të vlerave të tensionit. Brezet matëse janë +/- 0.1999[V], +/- 1.999[V], +/- 19.99[V], +/- 199.9[V], psh për përcaktim të brezit automatik, për secilin brez përdoret dalja “adekuate” 0, 1, 2, 3 përkatësisht, mirëpo mund të përdorën edhe daljet e tjera të lejuara kur brezi zgjedhet manualisht.

Në figurën 1, shihet se tensioni i aplikuar në hyrje të voltmetrit është +15.37[V], kjo vlerë i takon brezit +/- 19.99[V], supozojmë se dalja që zgjedhet nga multiplekseri analog është adekuate në këtë rast dalja 2, në të cilën vlera e tensionit është 0.1535[V], nësë zgjedhet dalja 3 potenciali në atë nyje do të jetë 0.0153[V], nëse zgjedhet dalja 1 vlera e tensionit në atë nyje ështe 1.535[V].

Nëse aplikohet tension në hyrje më vlerë +1.537[V], atëherë kjo vlerë i takon brezit +/-1.999[V], supozojme se dalja që zgjedhet nga multiplekseri analog është adekuate, në këtë rast dalja 1, potenciali tek dalja 1 do të ishte 0.1535[V], njejtë sikurse me rastin paraprak, ndërsa nëse zgjedhet dalja 2, potenciali në atë pikë do të jetë 0.0153[V], nëse zgjedhet dalja 0 vlera e tensionit në atë pike do të ishte 1.537[V].

Nga këta dy shembuj shihet se nëse zgjedhet dalja “adekuate” vlera e tensionit në atë nyje do të jetë në brezin +/-0.1999[V]. Kështuqë nevojitet që këtë brez ta shëndrrojmë në një brez të

përshtatshëm të tensionit për konverterin Analog/Digjital MCP3208. Konverteri A/D që përdoret në këtë rast do ta përdorë brezin $0[V] - 4.096[V]$ si hyrje.

Dalja e ndarësit potenciometrik që zgjedhet nga multiplekseri analog përcjellet në dalje të multiplekserit dhe vazhdon në hyrje të shndërruesit të brezit.

Shëndrruesi i brezit përdoret më qëllim që ta shndërrojë tensionin nga brezi {nga $-0.1999[V]$ deri $+0.1999[V]$ } në brezin {nga $0[V]$ deri $4.096[V]$ }. Për ta realizuar këtë shëndërrim, fillimisht e shumëzojmë brezin me 10.24, pastaj ia shtojmë 2.048.

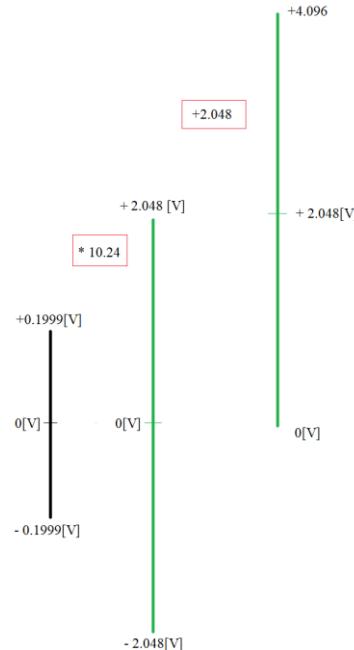


Fig. 6 - Shëndrrimi i brezit

Praktikisht mund të arrihet duke i përdorur skemat e amplifikatorëve operacional, si joinvertues, mbledhës dhe invertues.

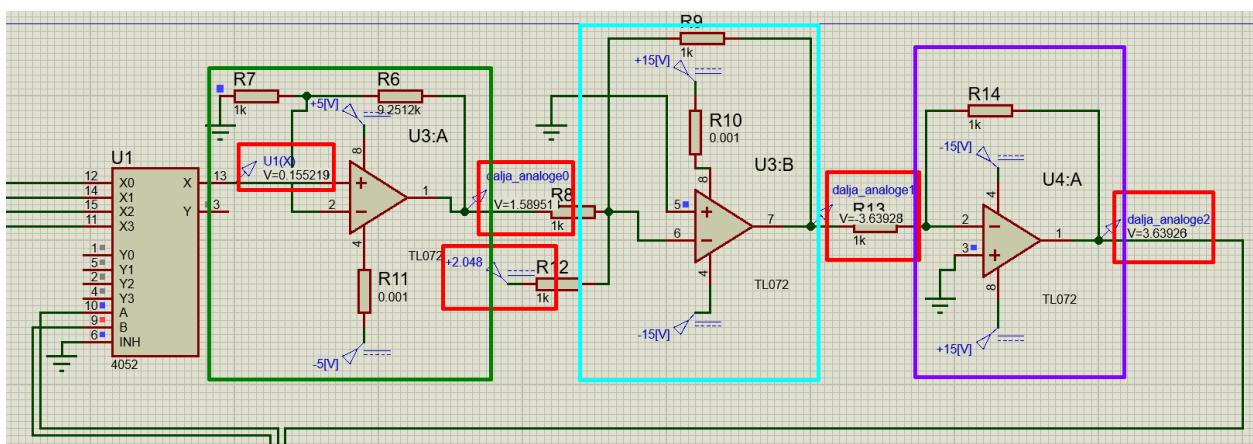


Fig. 7 - Shëndërrimi i brezit me AO

Në kuadrantin e gjelbërt është një AO joinvertues cili e kryen veprimin matematikor të shumëzimit me 10.24. Sipas relacionit:

$$V_{dalja_analoge0} = \frac{R_6+R_7}{R_7} \times U1 = 1.589 \text{ [V]}, \text{ ku } \frac{R_6+R_7}{R_7} = 10.24$$

Pastaj kësaj vlerë i shtohet 2.048 duke përdorur një AO mbledhës që është në kuadratin e kaltërt. Sipas relacionit:

$$V_{dalja_analoge1} = -\left(\frac{R_9}{R_8} V_{dalja_analoge0} + \frac{R_9}{R_{12}} \times 2.048\right) \text{ [V]}$$

$$V_{dalja_analoge1} = -(1.589 + 2.048) \text{ [V]} = -3.639 \text{ [V]}$$

Nga rezultati shihet se mbledhësi e invertion vlerën e tensionit, pra brezin e konverton në {nga -4.096 [V] deri 0 [V]}, për këtë arsyesh vendoset një amplifikator operacional invertues në kuadrantin vjollcë, që rezultatin e shumëzon me $A_v = -1$ dhe brezin e konverton në {nga 0 [V] deri 4.096 [V]}. Sipas relacionit:

$$V_{dalja_analoge2} = (-1)V_{dalja_analoge1} = 3.639 \text{ [V]}$$

Shprehjet për llogaritje të tensionit në dalje të secilit AO mund të nxjerren më metoda klasike të analizës së qarqeve.

Konverteri A/D

Konverteri A/D është një IC që e bën konvertimin e një sinjali analog në sinjal digital. Në rastin konkret përdoret MCP3208, është A/D 12-bit, që nënkupton se një mostër të sinjalit analog mund ta reprezentoj më njërin prej 4096 nivelle diskrete dhe digitale.

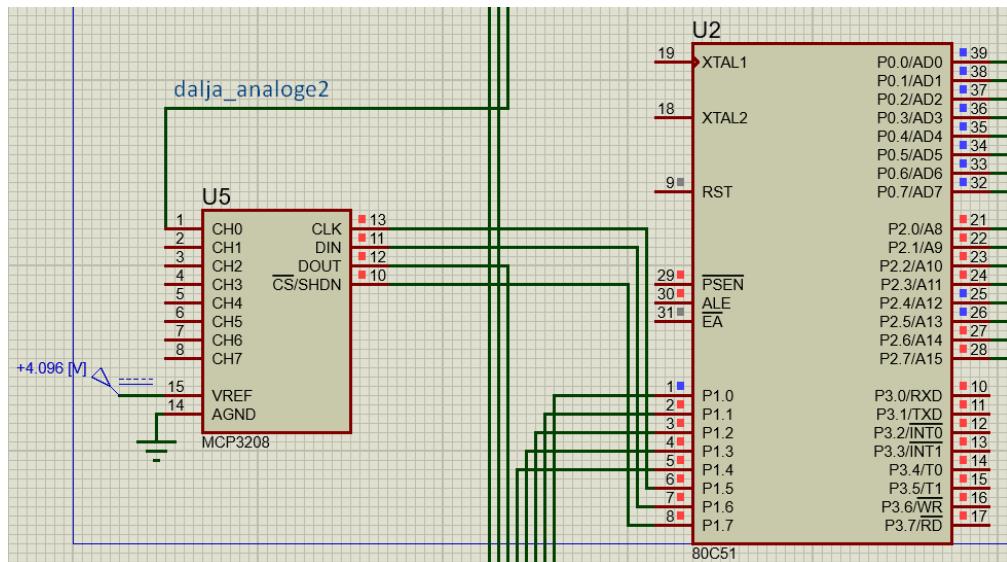


Fig.8 - Konverteri A/D MCP3208

Tensioni i aplikuar në pinin V_{ref} , e përcakton brezin e tensionit më të cilin punon konverteri, gjithashtu e përcakton edhe diferencën mes niveleve diskrete fqinje apo sensitivitetin e konverterit ndaj ndryshimeve të sinjalit analog në hyrje. Në rastin konkret, aplikohet tension prej 4.096V në V_{ref} që nënkupton se shkalla e sensitivitetit ndaj ndryshimeve të sinjalit në hyrje është 0.001V apo 1mV. Për cdo vlerë analoge nga 0V deri në 4.096V, konverteri e jep një reprezentim digital me 12 bita, cili përcaktohet sipas relacionit:

$$\text{Digital Output Code} = \frac{4096 \times V_{in}}{V_{ref}}$$

Ku V_{in} është potenciali që aplikohet në hyrje të A/D në pinin CH0, i cili vjen nga dalja e shndërruesit të brezit, saktësisht nga terminali dalja_analoge2. Për vlerën e tensionit të aplikuar në figurën 1 fitohet dalja_analoge2 si në figurën 6, pra 3.639[V], reprezentimi digital me 12 bita i vlerës analoge është 3639D = 1110 0011 0111B. Për një vlerë 1.058[V] do të fitohet 1058D etj. Komunikimi i mikrokontrollerit 8051 me konverterin A/D realizohet sipas protokolit SPI (Serial Peripheral Interface).

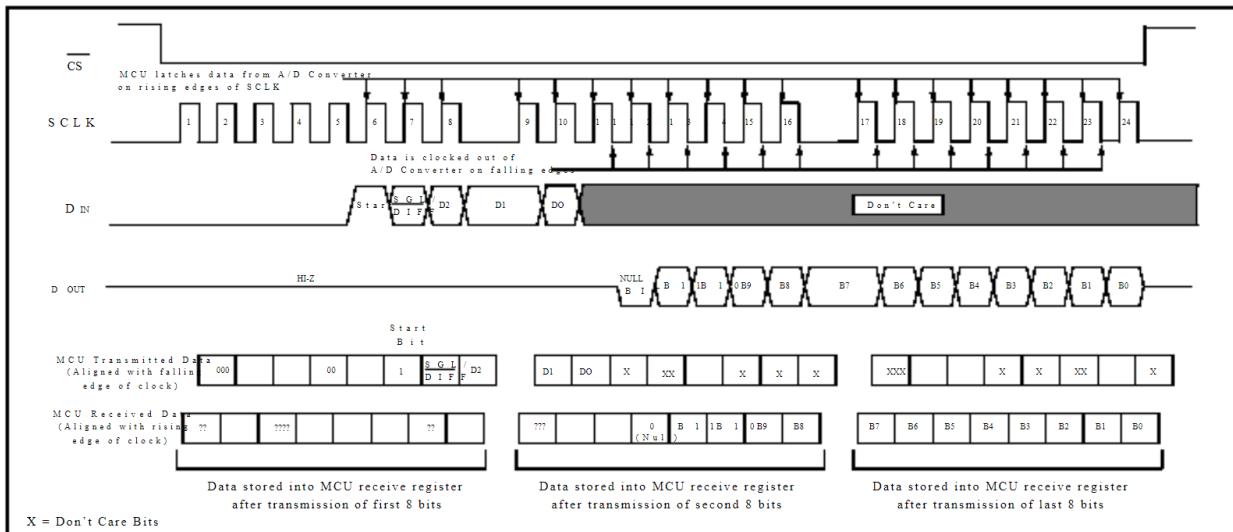


Fig. 9 - Komunikimi SPI, Mode (0,0 SCLK idles low)

```

1;----- SPI MCP32 -----
2SCLK EQU 95H ; slaveClock = 95H = P1.5
3MOSI EQU 96H ; Din = 96h = P1.6
4SS EQU 97H ; slaveSelect = 97H = P1.7
5MISO EQU 0A0H; Dout = A0 = P2.0
6;

```

Fig. 10 - Pinat e 8051 që lidhen me MCP3208

Që të filloj komunikimi nga mikrokontrolleri dërgohet ‘0’ në pinin \overline{CS} nga pini i 8051 i emëruar si SS (Slave Select). Pastaj vendoset ‘1’ në pinin MOSI (Master Out Slave In) që lidhet me Din të konverterit dhe start-bitit dërgohet në MCP3208 nga 8051 në teuhun rënës të clockut, clocku i MCP3208 kontrollohet nga pini SCLK (Slave Clock) i 8051 i lidhur me CLK.

```

133 CLR SS; mcp3208 is selected
134 SETB MOSI ; START bit
135 SETB SCLK
136 NOP
137 CLR SCLK ;send START bit on falling edge

```

Fig. 11 - Dërgimi i start-bitit

Për t'u konfiguruar konverteri duhet të dërgohen Start-Bit, SGL/DIF, D2, D1, D0 sikurse shihet në figurën 8, në mënyrë të ngjashme dërgohen edhe bitat tjera:

```

124;=====
125; FILLON - KOMUNIKIMI ME MCP3208
126;=====
127 SAMPLE: ;pas caktimit te brezit adekuat ne menyre automatike, e masim edhe njehere vleren
128
129 SETB SS ;mcp3208 nuk eshte e selektuar
130 CLR SCLK ;clk idles 'low' state, CPOL = 0
131 CLR MOSI
132
133 CLR SS; mcp3208 is selected
134 SETB MOSI ; START bit
135 SETB SCLK
136 NOP
137 CLR SCLK ;send START bit on falling edge
138 NOP
139 SETB SCLK
140 NOP
141 CLR SCLK ;send SGL bit MOSI = '1' , on falling edge
142
143 CLR MOSI
144 SETB SCLK
145 NOP
146 CLR SCLK ; send D2 = '0', on falling edge
147 NOP
148 SETB SCLK
149 NOP
150 CLR SCLK ;send D1 = '0', on falling edge
151 NOP
152 SETB SCLK
153
154 NOP
155 CLR SCLK ; send D0 = '0', on falling edge
156 NOP
157
158 SETB SCLK
159 NOP
160 CLR SCLK ;11th falling edge

```

Fig. 12 - Dërgimi i të gjithë bitave konfigurues në Din te MCP3208

C O N T R O L B I T S E L E C T I O N S				I N P U T C O N F I G U R A T I O N	C H A N N E L S E L E C T I O N
S I N G L E / D I F F	D2	D1	D0		
1	0	0	0	single ended	C H 0

Fig. 13 - Konfigurimi i MCP3208 për zgjedhje të CH0

Pastaj sikurse shihet në figurën 8, ndodh edhe një tranzicion i clock as nuk dërgohet bit dhe as nuk pranohet bit (te perioda 11), ndërsa pas këtij clocku në cdo tranzicion tjetër të clockut Digital Output Code prej (null bit)+12 bita fillojnë të dalin në Dout dhe lexohen nga pini MISO (Master In Slave Out).

```

161
162MOV R4,#5D ;get 4 bits + nullBit
163MOV A,#00H ;clr acc
164GET_DATA_MSB:
165NOP
166SETB P2.0 ;P2.0 input pin
167SETB SCLK ;(i.e 12th rising edge, null-bit is out)
168NOP
169MOV C,P2.0
170RLC A
171CLR SCLK
172NOP ; falling edge
173DJNZ R4, GET_DATA_MSB
174MOV 30H,A
175
176
177MOV R4,#8D ;get 8 bits
178MOV A,#00H ;clr acc
179GET_DATA_LSB:
180NOP
181SETB P2.0 ;P2.0 input pin
182SETB SCLK ; (i.e 17th rising edge, null-bit is out)
183NOP
184MOV C,P2.0
185RLC A
186CLR SCLK
187NOP ; falling edge
188DJNZ R4, GET_DATA_LSB
189MOV 31H,A
190;=====
191;           PERFUNDON - KOMUNIKIMI ME MCP3208
192;=====
```

Fig. 14 - Leximi i (null bit) + 12 bit data

Nëse e ndjekim linjën e shembullit të mëhershëm, i bie që 12-bitat që do të lexohen nga mikrokontrolleri prej konverterit do të janë $3639D = 1110\ 0011\ 0111B$. Fillimisht lexohen nullbit + 4 bitat MSB dhe ruhen në lokacionin 30H, pastaj lexohen 8 bitat LSB dhe ruhen në lokacionin 31H. Cdoherë kur përseritet rutina në fillim çselektohet (SETB SS) pajisja pastaj selektohet (CLR SS) prap që të jetë i mundur konverzioni tjetër.

Përpunimi i të dhënave të konverterit A/D

Të seksioni i shndërruesit të brezit mund të kuptohet se vlerat e tensionit në dalja_analoge2 që janë mes 2.048[V] dhe 4.096[V] janë vlera pozitive në brezin +/- 0.1999[V], pra janë vlera nga 0V deri në 0.1999[V], ndërsa vlerat e tensionit në dalja_analoge2 që janë mes 0[V] dhe 2.048[V] janë vlera negative në brezin +/- 0.1999[V], pra janë vlera nga -0.1999[V] deri në 0[V]. Cdo vlerë negative e brezit +/- 0.1999[V] mund të pasqyrohet në vlerë pozitive dhe pastaj të trajtohet si e tillë, pastaj në displej vetëm i vendoset parashenja minus që të diferencohet. Në figurën 1, paraqitet rasti kur aplikohet tension 15.37[V] dhe kemi supozuar se është zgjedhur Dalja2 nga multiplekseri analog, pra vlera 0.1535[V] pasi të kalon nëpër shndërrime bëhet dalja_analoge2 = 3.639[V] dhe në mikrokontroller lexohet si 3639 në numra binar 12-bit. Nëse aplikohet i njejtë tension por me vlerë negative -15.37[V], atëherë potenciali në dalja2 do të jetë -0.1535[V] dhe pas shndërrimeve bëhet dalja_analoge2 = 0.4761[V]. Pasqyrimi i vlerës negative në pozitive bëhet $4.096[V] - 0.4761[V] = 3.639[V]$. Kjo është vlera e njejtë numerike por vetem më parashenjë të kundërt. Ky pasqyrimi bëhet vetëm për ta thjeshtuar përpunimin digjital të të dhënave që lexohen nga konverteri. Pra cdo numër që lexohet nga konverteri A/D fillimisht krahasohet me 2048, nëse është më i madh vazhdohet me llogaritje tjera, nëse është më i vogël fillimisht pasqyrohet në vlerë pozitive dhe ruhet një flag që numri ka qenë negative, pastaj vazhdohet me llogaritje tjera.

```
195;=====
196;          FILLON - ZBRITJA, SHUMEZIMI ME 1000 DHE PJEZIMI ME 1024
197;=====
198MOV 54H,#12D ; <--index for blank 1st 7seg
199;vjen vlera nga mcp3208 dhe ruhet 30H(MSB) dhe 31H(LSB)
200;krahasojme numrin a eshte me i madh se 2048 apo me i vogel
201MOV A,30H
202CJNE A,#08H,CHECK_CARRY_01      ;if (A)<(data), C=1
203MOV A,31H
204CJNE A,#00H,CHECK_CARRY_01      ;if(A) < (data), C=1
205LJMP GREATER_THAN
206
207CHECK_CARRY_01:
208JC LESS_THAN
209LJMP GREATER_THAN
210
211;nese numri eshte me i vogel se 2048
212;gjejmë 2 komplementin e numrit
213LESS_THAN:
214DEC 54H ; store the dptr for minus sign, index for minus sign 1st 7seg
215MOV R7,#17D
216;nje komplementi
217ONE_COMPLEMENT:
218MOV A,30H
219RLC A
220CPL C
231
232ADD_HIGH: ;nese ka carry shto ne 30H
233MOV A,30H
234ADDC A,#00H
235MOV 30H,A
236;SJMP SUBTRACT_0
237
238SUBTRACT_0:
239MOV A,#10H ; 4096 D = 1000 H
240ADD A, 30H ; fshijme 4 bitat msb
241MOV 2FH,A
242CLR 7CH
243CLR 7DH
244CLR 7EH
245CLR 7FH
246MOV 30H,2FH
247
248
```

Fig. 15 - Krahasimi me 2048 dhe pasqyrimi i vlerës

Zbritja e 4096 me vlerën “negative” që u përmend në shembullin më lartë, në kod bëhet me mbledhje me 2-komplement. Nëse vlera ka nevojë të pasqyrohet nënkupton se është negative prandaj e dekrementojm vlerën e lokacionit DEC 54H, kështu vlera 11D në këtë lokacion do të pointon tek bajti që kur dërgohet në display shfaqet shenja minus, nëse numri është pozitiv dhe nuk ka nevoj të pasqyrohet atëherë vlera në lokacionin 54H mbetet 12D dhe pointon të bajti që kur dërgohet në display nuk shfaq asgjë. Cdoherë fjala është për 7 segmentshin e parë (nga ana e majtë). Pastaj nga kjo vlerë që gjendet në brezin {nga 0[V] deri 4.096[V]} për të kaluar në brezin {nga -2.048[V] deri 2.048[V]} në mikrokontroller vetëm e zbresim me 2048. Nga shembulli paraprak vlera që vjen nga konverteri është 3639D, kjo vlerë nuk ka nevojë të pasqyrohet sepse është pozitive, prandaj vetëm zbritet me 2048, kështu duhet fituar vlerën 1591D. Sigurisht që cdo veprim matematikor në mikrokontroller kryhet në sistemin e numrave binar.

```

247 GREAFTER_THAN:
248 DEC 54H
249 ;nese numri eshte me i madh se 2048
250 ;per ta kthyer vleren ne brez, fillimisht e zbresim me 2048
251 ;ruajme 2 komplementin e 2048 ne (MSB)4EH-4FH, 2 komplementi = 2048 poashtu
252 MOV 4EH,#08H
253 MOV 4FH,#00H
254
255 MOV A,30H
256 ADD A,4EH
257 MOV 2FH,A ;bit adressable
258
259 CLR 7CH ;e fshijm carry pas mbledhjes me dy komplement te 2048
260 CLR 7DH
261 CLR 7EH
262 CLR 7FH
263 MOV 30H,2FH

```

Fig. 16 – Zbritja me 2048

Për të kaluar nga brezi {nga -2.048[V] deri 2.048[V]} në brezin {nga -0.1999[V] deri 0.1999[V]} duhet të pjestojmë me 10.24, ndërsa në mikrokontroller pjestojmë me 1024. Pjestimi do të realizohet me shiftim. Dihet se shiftimi në të djathtë i një numri binar e pjeston atë me 2. Supozojmë se e kemi numrin 10D = 1010B, e shiftojmë djathatas për 1, bëhet 5D = 0101B, por gjatë pjestimit nuk ruhen vlerat pas presjës, nëse e shiftojmë edhe një herë 2D = 0010B, ndërsa $5/2 = 2.5$, prandaj për të mos i humbur shifrat pas presjës mund ta shumëzojmë me numrin fillestar 10 që ishte 10D, që i bie se bëhet 100D = 0110 0100B, nëse e shiftojmë djathatas bëhet 50D = 0011 0010B, pastaj nëse e shiftojmë edhe njëherë bëhet 25D = 0001 1001B, në këtë mënyre arrijmë ta ruajmë edhe numrin pas presjës që ishte 5. Nëse një numër e shiftojmë 1-herë djathatas pjestohet me 2, nëse e shiftojmë n-herë pjestohet me 2^n , prandaj për të pjestuar me 1024 mund ta shiftojmë një numër binar 10 herë djathatas, sepse $2^{10} = 1024$.

Nëse e ndjekim linjën e shembullit më herët, për të kaluar nga brezi {nga -2.048[V] deri 2.048[V]} që në këtë brez numri ishte 1591D, në brezin {nga -0.1999[V] deri 0.1999[V]}, pjestojmë me 1024 duke shiftuar. Numri 1591D = 0110 0011 0111B nëse shiftohet në të djathë 10 herë fitohet numri 1. Ndërsa $1591D \div 1024 = 1.553$, për të mos i humbur shifrat pas presjes në mikrokontroller, fillimisht numrin 1591 e shumëzojmë më 1000 pastaj e 1591000 e pjesojmë me 1024 duke e shiftuar 10-herë djathtas dhe fitohet numri 1553, që janë shifrat përafersisht të njëjtë më tensionin e aplikuar në figurën 1 (15.37[V]), vlera dallon për shkak të rrymave të offsetit të amplifikatorve operacional TL072, përndryshe në kushte ideale vlera do të ishte 1537. Shumëzimi i numrave dy bajtësh bëhet në mikrokontroller realizohet si shumëzim i numrave heskadecimal:

$$\begin{array}{r}
 \begin{array}{c} x \\ \hline \end{array} \quad \begin{array}{c} FF\ FF \\ FF\ FF \\ \hline FE\ 01 \end{array} \\
 + \quad \begin{array}{c} FE\ 01 \\ FE\ 01 \\ \hline FE\ 01 \end{array} \\
 \hline \quad \begin{array}{c} FF\ FE\ 00\ 01 \end{array}
 \end{array}$$

Fig. 17 – Shumëzimi i numrave 2 bajtësh

Ndërsa në mikrokontroller realizohet në këtë mënyrë:

```

267;pastaj e shumezojme me 1000
268;shumezuesi 1000 ruhet ne 33H(MSB) dhe 34H(LSB)
269MOV 33H,#03H
270MOV 34H,#0E8H
271
272;kodi me poshtë e shumezon numrin 12-bitesh ((MSB)30H-31H) qe vjen nga MCP3208 me 1000((MSB)33H-34H) dhe e ruan ne (MSB)30H-31H-32H
273;regjistrat e prishur: 30H - 3AH
274;shumezimi me 1000 behet me qellim qe te mos humbin numrat pas presjes kur pjestojme me shiftim
275MOV A,31H
276MOV B,34H
277MUL AB
278MOV 32H,A
279MOV 35H,B
280
281MOV A,30H
282MOV B,34H
283MUL AB
284MOV 36H,A
285MOV 37H,B
286
287MOV A,33H
288MOV B,31H
289MUL AB
290MOV 38H,A
291MOV 39H,B
292
293MOV A,33H
294MOV B,30H
295MUL AB
296.....
297
298.....
299
300.....
301
302MOV A,37H
303ADDC A,39H
304MOV 39H,A
305
306MOV A,36H
307ADD A,38H
308MOV 31H,A
309
310MOV A,39H
311ADDC A,3AH
312MOV 30H,A
313;rezultati i shumezimit del max 3 bytes, ruhet 30H-31H-32H
314

```

Fig. 18 – Shumëzimi i numrit me 1000

Ndërsa pjestimi i numrit me 1024 në mikrokontroller realizohet me shiftim si më poshtë:

```
317;per ta pjestuar numrin e ruajtur ne 30h-31h-32h me 1024, e shiftojme 10here ne te djathte
318;rezultati do te jete 2 bytes ne rastin me te keq dhe ruhet ne (MSB) 31H-32H
319MOV R2,#10D
320HERE:
321
322MOV A,30H
323RRC A
324MOV 30H,A
325
326MOV A,31H
327RRC A
328MOV 31H,A
329
330MOV A,32H
331RRC A
332MOV 32H,A
333
334CLR C
335DJNZ R2, HERE
336
337;=====
338;           PER FUNDON - ZBRITJA, SHUMEZIMI ME 1000 DHE PDESTIMI ME 1024
339=====
```

Fig. 19 – Pjestimi me 1024

Ekstraktimi i shifrave

Numri 1553 i shembullit më lartë ruhet si numër binar në regjistra (MSB) 31H dhe 32H, si $1553D = 0000\ 0110\ 0001\ 0001B$, ky numër që të shfaqet në display me 7 segmentësh duhet që njëherë të ndahet në 4 shifra dhe pastaj të shfaqet. Ndarja e shifrave mund të arrihet duke e pjestuar numrin 1553D katër herë me 10 dhe duke e ekstraktuar mbetjën në një regjistër. Pasiqë $1553 \div 10 = 155$ dhe mbetja 3 e ruajmë në regjistrin 50H, pastaj $155 \div 10 = 15$ dhe mbetja 5 e ruajmë në regjistrin 51H, $15 \div 10 = 1$ dhe mbetja 5 e ruajmë në regjistrin 52H, $1 \div 5 = 0$ dhe mbetja 1 e ruajmë në regjistrin 53H. Kështu pra kemi arritur t'i ruajmë 4 shifrat në regjistra të ndryshëm: 53H = (1D), 52H = (5D), 51H = (5D) dhe 50H = (3D), ndërsa parashenja nëse është minus apo jo ruhet në regjistrin 54H dhe përcaktohet në mënyrën që shpjegohet në seksionet me lartë. Në kodin më poshtë realizohet operacioni i përshkruar më lartë vetëm se në sistemin e numrave binar, konkretisht algoritmi i pjestimit me 10D (1010B) 4-herë dhe ekstraktimi i mbetjes në secilën prej herave.

```

342; =====
343; FILLON - EKSTRAKTIMI I SHIFRAVE
344; =====
345
346;kodi me poshte pjeston me 10 numrin qe gjendet ne (MSB)31H-32H
347;dhe e ekstraktion mbetjen, keshu ndodh 4 here per 4 shifra
348MOV R0,#4FH ; ketu ruaj mbetjen
349MOV 49H,#0AH ; store 10
350
351SHIFT AGAIN:
352
353CJNE R2,#10H,DONT_STORE_0 ; if (R2) < (data), CY=1
354LCALL STORE_REMAINDER_0
355DONT_STORE_0: ;store remainder
356CLR C
357
358;perfundo programin ketu
359CJNE R0,#53H,DONT_END
360LJMP END_DIVISION
361DONT_END:
362CLR C
363
364
365LCALL SHIFT_LEFT
366CHECK:
367MOV A,49H ; A = 10
368CJNE A, 41H, CHECK_CARRY ; if (A)<(data), CY=1
369SJMP EQUAL
370CHECK_CARRY:
371JNB C, SHIFT AGAIN
372
373CLR C ;C fshihet qe te mos e ndikoj zbritjen
374EQUAL:
375MOV A,41H
376SUBB A,49H
377MOV 41H,A
378
379CJNE R2,#10H,DONT_STORE_1 ; if (R2) < (data), CY=1
380LCALL STORE_REMAINDER_1
381DONT_STORE_1:
382
383SETB C
384LCALL SHIFT_LEFT
385CLR C
386SJMP CHECK
387
388
389STORE_REMAINDER_0:
390INC R0
391MOV @R0,41H
392MOV 41H,#00H
393LCALL SHIFT_LEFT
394MOV R2,#00H
395RET
396
397
398STORE_REMAINDER_1:
399INC R0
400MOV @R0,41H
401MOV 41H,#00H
402MOV R2,#0FFH
403RET
404
405
406SHIFT LEFT:
407INC R2
408
409MOV A,32H
410RLC A
411MOV 32H,A
412
413MOV A,31H
414RLC A
415MOV 31H,A
416
417MOV A,41H
418RLC A
419MOV 41H,A
420
421MOV A,40H
422RLC A
423MOV 40H,A
424
425RET
426
427END_DIVISION:
428CLR C
429
430;shifrat i ruan ne 53H, 52H, 51H,50H

```

Fig. 20 – Pjestimi me 10 katër herë dhe ekstraktimi i mbetjes

Mbetjet ruhen duke filluar nga 50H, ku si pointer shfrytëzohet R0. Pastaj këto shifra kthehen në kodin përkatës të numrit që do të shfaqet në 7 segmentësh dhe prap ruhen në të njejtat lokacione.

```

432;;kodi me poshte i zëvendson shifrat decimale me kodin e 7seg perkates, ne regjistrat perkates
433MOV A,53H
434MOV 59H,53H ;--> Ruhet shifra e pare
435MOVC A,@A+DPTR
436MOV 53H,A
437
438MOV A,52H
439MOV 5AH,52H ;--> Ruhet shifra e dyte
440MOVC A,@A+DPTR
441MOV 52H,A
442
443MOV A,51H
444MOV 5BH,51H ;--> Ruhet shifra e parafundit
445MOVC A,@A+DPTR
446MOV 51H,A
447
448MOV A,50H
449MOV 5CH,50H ;--> Ruhet shifra e fundit
450MOVC A,@A+DPTR
451MOV 50H,A
452
453MOV A,54H
454;;MOV 58H,54H ;--> Ruhet parashenja #00H ose #40H
455MOVC A,@A+DPTR
456MOV 54H,A
457MOV 58H,54H ;--> Ruhet parashenja #00H ose #40H
458
459RET ;return from sample subroutine
460=====PERFUNDON - EKSTRAKTIMI I SHIFRAVE=====
461;
462=====

```

Fig. 21 – Zëvendësimi i shifrave decimale me kodin për shfaqje në 7 segment

Nga fillimi i komunikimit me MCP3208 e deri te përfundimi i ekstraktimit të shifrave dhe zëvendësimit me kode për shfaqje në display kodi gjendet brenda një rutine që vetëm e mostron vlerën dhe i dërgon 4 shifrat e para dhe parashenjën në regjistrat e përmendor më lartë dhe kjo rutine në kod është emëruar SAMPLE.

Displeji i multipleksuar

Displeji përbëhet nga 5 shtatë-segmentësh me katodë të përbashkët të lidhur në portin 0, ndërsa katodat e tyre kontrollohet nga disa pinë të portit 1.

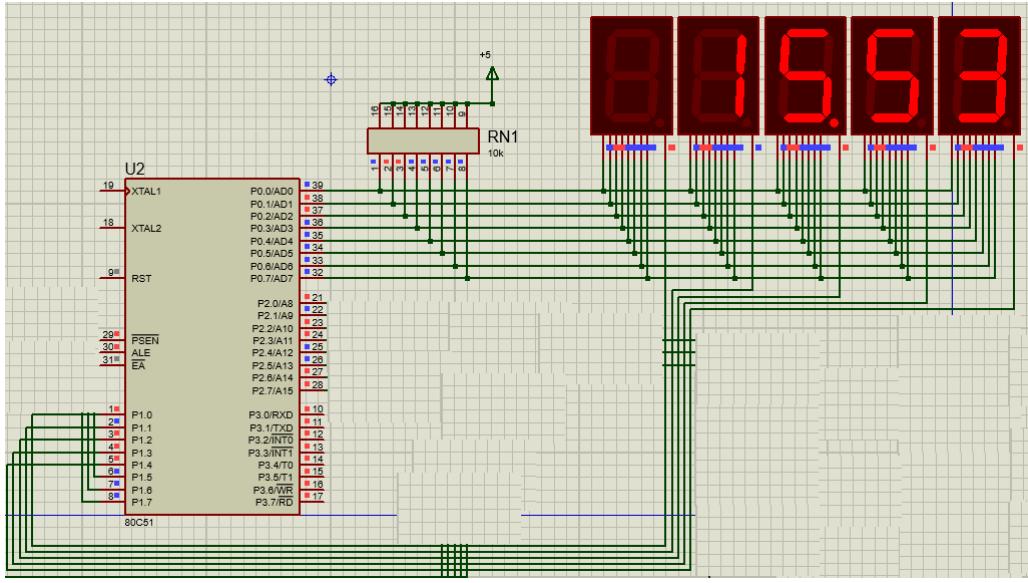


Fig. 22 – Displeji me 5 shtatë-segmentësh të multipleksuar

Që të përdoret porti 0 duhet të vendosen pull-up rezistorët eksternal pasiqë nuk i ka pull-ups internal sikurse portet tjera. Programi në main cdo periodë kohe të caktuar e lëshon 7seg e fundit e vendos kodin për numrin 3 në P0 nga regjistri 50H, dhe i ndal të tjerët, pastaj e lëshon 7seg e parafundit e vendos kodin për numrin 5 në P0 nga regjistri 51H dhe i ndal të gjithë të tjerët, pastaj e lëshon 7seg e 3-të e vendos kodin për numrin 5 në P0 nga regjistri 52H dhe i ndal të gjithë të tjerët, pastaj e lëshon 7seg e dytë e vendos numrin 1 në P0 nga regjistri 53H dhe i ndal të gjithë të tjerët, pastaj e lëshon 7seg e parë e vendos parashenjen përkatëse nga regjistri 54H në P0 dhe i ndal të gjithë të tjerët. Kështu në mënyrë periodike i shfaq të gjithë numrat e ndryshëm nga i nejti port. Pikëpamja subjektive është se të gjithë numrat shfaqen njëkohësisht për arsyen se frekuanca e iterimit nëpër displej është shumë e madhe.

```

941;=====
942;                               FILLON - SHFAQJA E SHIFRAVE
943;=====
944DISPLAY_RESULT:
945SETB P1.4
946SETB P1.3
947SETB P1.2
948SETB P1.1
949SETB P1.0
950
951SETB RS1
952SETB R50 ;Selektohet BANK 3
953
954MOV P0,@R0 ;ne fillim (R0) = 50H ndersa (R1) = 00H
955
956CJNE R1,#00H,CHECK_1
957CLR P1.4
958SJMP END_DISPP
959CHECK_1:
960CJNE R1,#01H,CHECK_2
961CLR P1.3
962SJMP END_DISPP
963CHECK_2:
964CJNE R1,#02H,CHECK_3
965CLR P1.2
966SJMP END_DISPP
967CHECK_3:
968CJNE R1,#03H,CHECK_4
969CLR P1.1
970SJMP END_DISPP
971CHECK_4:
972CJNE R1,#04H,CHECK_5
973CLR P1.0
974SJMP END_DISPP
975CHECK_5:
976
977END_DISPP:
978INC R0
979INC R1
980
981CJNE R0,#55H,CHECK_6
982MOV R0,#50H
983MOV R1,#00H
984CHECK_6:
985
986CLR RS1
987CLR RS1
988
989RET ; perfundon rutina e display
990
991DIGITS: DB 3FH, 06H, 5BH, 4FH, 66H, 6DH, 7DH, 07H, 7FH, 6FH, 40H, 00H
992;minusi-->^      ^<---blank
993
994;=====
995;                               PERFUNDON - SHFAQJA E SHIFRAVE
996;=====

```

Fig. 22 – Kodi për displej

Shikohet sa herë është thirrur rutina dhe secilën herë ndërrohet 7seg në mënyrë iterative, ndërrimi bëhet duke i kontrolluar katodat e 7seg. Në DB DIGITS ruhen kodet e numrave nga 0 (3FH) deri në 9 (6FH), 40H është kodi për parashenjën minus, ndërsa, 00H nuk shfaq asgjë në 7seg. Ndërsa në DPTR ruhet lokacioni i DIGITS, sikurse shihet në figurën 21, kodet qasen me instruksionin MOVC A,@A+D PTR, ku A shërben si indeks për qasje. Ndërsa si pointer për vendosjen e vlerave nga regjistrat 50H-54H në P0, përdoret regjistri R0 i bankës 3.

Zgjedhja automatike e brezit

Nëse i referohemi figurës 1, kur kemi supozuar se është zgjedhur Dalja2 nga multiplekseri analog, në të cilën vlera e potencialit është 0.1535[V], shifrat që shfaqen në display sikurse shihen në figurën 22, pas të gjitha përpunimeve janë 1553, edhe pse në kushte ideale do të ishte 1535, ngjashëm si vlera në dalje. Pra cdoherë shifrat që do të ruhen në regjistrat 53H-50H do të janë 4 shifrat e para pas presjës të vlerës se potencialit në daljen e caktuar të ndarësit potenciometrik.

Nëse do të ishtë selektuar Dalja3 në të cilën potenciali është 0.0155[V], atëherë shifrat që do të shfaqeshin në displej janë 0155. Ndërsa nuk bën të selektohet Dalja1 ku potenciali është 1.534[V], as Dalja0 ku potenciali është 15.35[V], sepse këto vlera janë jashtë brezit +/-0.1999[V], sipas të cilit e kemi dizajnuar shëndrruesin e brezit për të ia përshtatur vlerat konverterit A/D. Nëse tensioni në dalje është mjaftueshëm i lartë mund ta dëmtoj pjesën tjetër të qarkut nëse nuk është e mbrojur. Dalja adekuate është 2, sepse e ka formatin 0.1xxx.

Megjithatë, nëse aplikohet në hyrje tension psh 25.68[V], atëherë daljet e ndarësit potenciometrik do t'i kishin këto vlera: Dalja0 = 25.68[V], Dalja1 = 2.568[V], Dalja2 = 0.2568[V], Dalja3 = 0.0256[V], në këtë rast nuk bën të zgjedhet as Dalja2 sikur në rastin e vlerës 15.37[V] të shembullit më lartë, sepse në këtë rast edhe Dalja2 është jashtë brezit +/- 0.1999[V]. Nuk është dalje adekuate sepse nuk e ka formatin 0.1xxx.

Nëse në hyrje aplikohet tensioni $0.1678[V] = 167.8[mV]$, atëherë vlerat e potencialit në dalje të ndarësit potenciometrik do të ishin: Dalja0 = 0.1678[V], Dalja1 = 0.0167[V], Dalja2 = 0.0016[V], Dalja3 = 0.0001[V]. Në këtë rast dalja adekuate është 0, sepse e ka formatin 0.1xxx. Në rastin e vlerës millivolt, kur zgjedhet dalja0, në display shfaqen numrat 1678, ndërsa pika vendoset pas numrit 7, pra edhe në displej shfaqet si milivolt.

Pra kriteret për ta zgjedhur daljen adekuate në mënyre automatike janë: që vlera e potencialit të jetë në brezin +/- 0.1999[V] dhe që numrat të fillojnë mënjerë pas presjes 0.xxx, pra të mos jetë vlera 0.0xxx, 0.00xx, 0.000x. Implementimi i këtyre kritereve në mikrokontroller bëhet si në vazhdim.

```
702;-----  
703;----- FILLON - AUTOMATIC RANGE  
704;-----  
705AUTOMATIC_RANGE:  
706SETB 50H ;-- Setohet biti 0 i reg. ZAH nese brezi automatic  
707;  
708SETB 67H ; <- Setohet flagu qe tregon se matja eshte realizuar dhe mund te dergohen bytet ne EEPROM  
709;  
710;by default eshte i zgjedhur brezi 4 (+/- 199.9 V)  
711;shikojm sa zero permban numri 4-shifror  
712;  
713SETB OUTSA ;zgjedhet range 199.9V  
714SETB OUTSB  
715;  
716LCALL SAMPLE  
717;  
718MOV A,53H  
719;  
720CJNE A,#3FH,NOT_NEXT0 ; check if zero, 3FH eshte vlera e 7seg per nr. 0  
721SJMP NEXT0  
722NOT_NEXT0:  
723;  
724MOV A,51H ;vendos pikën per range 199.9  
725SETB ACC,7  
726MOV 51H,A  
727;  
728SJMP DALJA4_0 ;nenkupton qe brezi eshte adekuat (+/- 199.9 V)  
729;  
730DALJA4_0:  
731LJMP DALJA4 ;error 8-bit relative  
732;
```

Fig. 22 – Caktimi automatik i brezit për daljen 3

Biti 0 i regjistrat 2AH e ruan gjendjen automatike/manual. Nëse thirret rutina AUTOMATIC_RANGE kur aplikohet tension 175.3[V] fillimisht zgjedhet Dalja3, në të cilën potenciali është 0.1753[V] dhe thirret rutina SAMPLE, në regjistrat 53H-50H ruhen vlerat 1753 pastaj a ka zero para shifrave pas presjës, në këtë rast nuk ka zero, atëherë dalja adekuate është Dalja3 dhe pika vendoset të shifra e tretë, duke e setuar bitin e 7 të atij numri, pasi biti i 7 i P0 është i lidhur më pikën në 7seg dhë në display shfaqet numri 175.3, megjithatë ne i ndryshojmë vetëm regjistrat 53H-50H, ngase nga këtu i merr vlerat P0 për ti shfaqur. (Për Fig. 22)

```

732;-----
733NEXT0:
734MOV A,52H
735
736CJNE A,#3FH,NOT_NEXT1 ;check if zero
737SJMP NEXT1
738NOT_NEXT1:
739;nese display i dyte nuk eshte zero - duhet shikohet nese eshte 1 i takon brezit, nese eshte i ndryshem nuk i takon
740MOV A,52H
741CJNE A,#06H,NOT_ONE_0
742SETB OUTSB ;zgjedhet range 19.99
743CLR OUTSA
744
745LCALL SAMPLE
746
747MOV A,52H ;vendoset pika per range 19.99
748SETB ACC.7
749MOV 52H,A
750
751SJMP DALJA_TJETER
752NOT_ONE_0:
753SETB OUTSA ;zgjedhet range 199.9
754SETB OUTSB
755
756LCALL SAMPLE
757
758MOV A,51H ;vendoset pika per range 19.99
759SETB ACC.7
760MOV 51H,A
761
762
763SJMP DALJA_TJETER ;nenkupton qe vetem display i pare i shifrave eshte zero zgjedhet brezi (+/- 19.99 V)
764;-----

```

Fig. 23 – Caktimi automatik i brezit për daljen 2

Nëse thirret rutina AUTOMATIC_RANGE kur aplikohet tension 18.62[V], (shiko Fig.22) fillimisht zgjedhet Dalja3, në të cilën potenciali është 0.0186[V] dhe thirret rutina SAMPLE, në regjistrat 53H-50H ruhen vlerat 0186, shikohet a ka zero para numrave pas presjës, pra shikohet regjistri 53H, në këtë rast është zero, kështuqe kodi vazhdohet në figurën 23, shikohet regjistri 52H se a është zero, në këtë rast nuk është zero, pastaj shikohet numri a është 1 apo më i madh, nëse është 1 mund të zgjedhet Dalja2 ku potenciali është 0.1862[V], thirret rutina SAMPLE prap në regjistrat 53H-50H ruhet vlera 1862 dhe pika vendoset të shifra e dytë në displej, pra 18.62. Nëse numri në regjistrin 52H do të kishte qenë 2, psh të numri 0286, atëherë nuk do të ishte zgjedhur Dalja2 sepse nuk plotësohet kriteri i brezit që u përmend më lartë edhe pse plotësohet kriteri për zero. (Për Fig. 23)

Ngjashëm shqyrtohen edhe për dy daljet tjera. Pra cdoherë në filim zgjedhet Dalja3, shikohet sa zero janë, dhe shikohet numri i parë pas zerove se a është 1 apo më i madh, pastaj në bazë të dy kritereve përcaktohet cila dalje duhet të zgjedhet të ndarësi potenciometrik dhe ku do të vendoset pika në displej.

```

764;-----
765NEXT1:
766MOV A,51H
767
768CJNE A,#3FH,NOT_NEXT2 ;check if zero
769SJMP NEXT2
770NOT_NEXT2:
771;nëse display i trete nuk eshte zero - duhet te shikohet nese eshte 1 i takon brezit, nese eshte i ndryshem nuk i takon
772MOV A,51H
773CJNE A,#06H,NOT_ONE_1
774
775CLR OUTSB ;zgjedhet range 1.999
776SETB OUTSA
777
778LCALL SAMPLE
779
780MOV A,53H ;vendoset pika per range 1.999
781SETB ACC.7
782MOV 53H,A
783
784SJMP DALJA_TJETER
785NOT_ONE_1: ;nuk eshte 1, nuk i takon brezit
786
787SETB OUTSB ;zgjedhet range 19.99
788CLR OUTSA
789
790LCALL SAMPLE
791
792MOV A,52H ;vendoset pika per range 19.99
793SETB ACC.7
794MOV 52H,A
795
796SJMP DALJA_TJETER ;nenkupton qe displayt 1 dhe 2 te shifrave jane zero zgjedhet brezi (+/- 1.999 V)
797;-----
798NEXT2:
799MOV A,50H
800
801CJNE A,#3FH,NOT_NEXT3 ;check if zero
802SJMP NEXT3
803NOT_NEXT3:
804;nëse display i fundit nuk eshte zero - duhet shikohet nese eshte 1 ose jo, por fillimi i ndrrojm brezin per rritje te saktesise (per raste kufitarë)
805te zgjedhim brezin 19.99, per dallim nga rastet me Larte
806SETB OUTSB
807CLR OUTSA
808LCALL SAMPLE
809toni nuk e shikojmë displayn e fundit (50H) por displayn e parafundit
810MOV A,51H
811CJNE A,#06H,NOT_ONE_2
812;nëse eshte 1, zgjedhet brezi 0.1999
813
814CLR OUTSB ;zgjedhet range 0.1999
815CLR OUTSA
816
817LCALL SAMPLE
818
819MOV A,51H ;vendoset pika per milivolt (199.9 mV)
820SETB ACC.7
821MOV 51H,A
822SJMP DALJA_TJETER
823
824NOT_ONE_2: ;nëse nuk eshte 1, zgjedhet brezi 1.999V
825CLR OUTSB
826SETB OUTSA
827
828LCALL SAMPLE
829
830MOV A,53H ;vendoset pika per range 1.999
831SETB ACC.7
832MOV 53H,A
833
834SJMP DALJA_TJETER ;nenkupton qe displayt 1,2,3 te shifrave jane zero zgjedhet brezi (+/- 0.1999 V)
835;-----

```

Fig. 24 – Caktimi i brezit për daljen 1 dhe 0

```

852MOV C,OUTSA ;ruajme brezin e automatik per mostren ne bitat e regjistrat 2EH (bit-addressable)
853MOV 70H,C
854MOV C,OUTSB
855MOV 71H,C
856
857RET
858;=====
859;                      PERFUNDON - AUTOMATIC RANGE
860;=====
```

Fig. 25 – Ruajtja e brezit

Në fund brezi i zgjdhur ruhet në bitat (OUTSA) = 71H dhe (OUTSB) = 70H të regjistrat 2EH.

Tastaura me shiftregjistër

Tastatura përbëhet prej shiftregjistërit dhe butonave si në figurë.

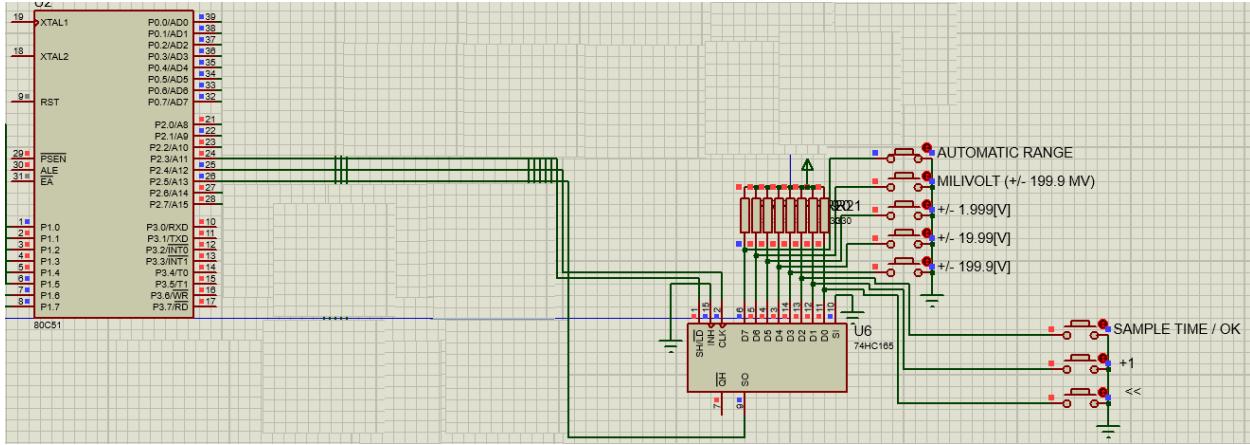


Fig. 26 – Tastatura

```

13;----- Keyboard -----
14 SHLD EQU 0A3H ;P2.3
15 SHCLK EQU 0A4H ;P2.4
16 SO EQU 0A5H ;P2.5
17;

```

Fig. 27 – Pinat e 8051 për tastatur

Shiftregjistéri 74HC165 është një qark i integruar që i pranon 8 bita në mënyrë paralele përmes pinave D7 – D0 të cilët pastaj mund të shiftohen në mënyrë serike sipas renditjes përmes pinit SO të shiftregjistrit për në mikrokontroller. Kur butonat nuk shtypen, pinat e shiftregjistërit janë të lidhur më pull-up resistor dhe gjenden në nivel logjik ‘H’, prandaj cdo lexim i bitave të shiftregjistrit në mënyrë serike nga mikrokontrolleri do të ishte 1111 1111B. Butoni i parë AUTOMATIC RANGE është i lidhur në pinin D7 të shiftregjistrit dhe cdo buton më poshtë është i lidhur në pinin D më të vogël se butoni më lartë. Nëse shtypet një buton, atëherë krijohet lidhje e shkurtë e tokëzimit të përbashkët të butonave dhe pinit të shiftregjistrit të i cili është shtypur butoni, rrjedhimisht pini do të jetë në nivel logjik ‘L’ sikurse shihet në figurën 26, dhe leximi bitave të shiftregjistrit nga mikrokontrolleri në këtë rast do të ishte 0111 1111B. Pra nëse butoni shtypet në mikrokontroller dërgohet 0 për butonin e shtypur në pozitën përkatëse.

```

465;=====
466;                               FILLON - KOMUNIKIMI ME TASTATURE
467;=====
468
469GET_UPDATED_BUTTONS:
470
471CLR SHLD ; Load-mode of shiftregister with button-pressed values
472CLR SHCLK ; clock of shiftregister idles in low-state
473
474MOV R6,#8D ;8 bits of shiftregister
475
476SETB SHLD ;shift-mode of shiftregister
477
478NEXT_BUTTON:
479
480SETB S0 ;serial input pin of mcu is set '1'
481
482MOV C,S0 ;the value of shiftregister is moved to C
483RLC A ;rotate-left Acc
484
485SETB SHCLK ;Low to high transition of shiftregister clock
486CLR SHCLK ;high to low transition of shiftregister clock
487DJNZ R6,NEXT_BUTTON ;repeat to get all 8 bits from shiftregister
488
489MOV 2DH,A ;send 8 bits to 2DH byte, -bitAddressable
490
491RET
492
493;=====
494;                               PERFUNDON - KOMUNIKIMI ME TASTATURE
495;=====

```

Fig. 28 – Komunikimi me tastaturë

Komunikimi me tastaturë realizohet në rutinën GET_UPDATED_BUTTONS dhe 8 bitat e lexuar ruhen në lokacionin 2DH i cili është bit addressable, pra në bitat 6FH – 68H. Pastaj këta bita shfrytëzohen për ta zgjedhur një rutinë të caktuar në bazë të butonit të shtypur, psh nëse shtypet butoni AUTOMATIC RANGE si në figurën 26, pinin D7 kalon në nivel ‘L’ logjik dhe kur dërgohen bitat, biti 6FH do ta ketë vlerën zero.

Zgjedhja manuale e brezit

Zgjedhja manuale e brezit nënkupton zgjedhjen e njërsës prej daljeve të ndarësit potenciometrik dhe vendosjen e pikës në displej në pozitën e duhur. Kjo pjesë në kod realizohet brenda rutinës MANUAL_RANGE.

```

498;=====
499;                               FILLON - CAKTIMI MANUAL I RANGE
500;=====
501;Nese dergohet zero ne lokacionin perkates, butoni eshte shtypur
502MANUAL_RANGE:
503;duhet te ruhet koha relative ne regjistra tjere, matet ne MSB-45H dhe LSB-46H
504;dergohet ne MSB-5EH dhe LSB-5FH
505MOV 5EH,45H
506MOV 5FH,46H
507
508
509;LCALL AUTOMATIC_RANGE ; per ta caktuar brezin adekuat, sepse duhet per krahasim, brezi ruhet ne reg. 2EH
510;
511;LCALL GET_UPDATED_BUTTONS
512;
513JNB 6FH,ZGJEDH_AUTOMATIKE ;bitat e regjistrat 2DH
514JNB 6EH,ZGJEDH_MILIVOLT
515JNB 6DH,ZGJEDH_1_999
516JNB 6CH,ZGJEDH_19_99
517JNB 6BH,ZGJEDH_199_9
518JNB 6AH,ZGJEDH_MANUAL_SAMPLE_TIME
519
520LJMP END_MANUAL_RANGE
521

```

Fig. 30 – Zgjedhja e daljes përmes butonave

Në fillim të rutinës MANUAL_RANGE shikohet se cili bit i regjistrat 2DH është zero, pra cili buton është shtypur. Nëse shtypet butoni AUTOMATIC_RANGE, atëherë biti 6FH bëhet zero, dhe kur ky bit është zero programi degëzohet të etiketa ZGJEDH_AUTOMATIKE, që nënkupton se brezi për këtë mostër do të caktohet në mënyre automatike, brezi në fund ruhet dhe dërgohet në bitat e regjistrat 29H.

Nëse shtypet butoni MILIVOLT (+/- 199.9 mV), kodi kalon në procedurën e njejtë dhe degëzohet në etiketën ZGJEDH_MILIVOLT.

```

542;
543 ZGJEDH_MILIVOLT:
544 LCALL AUTOMATIC_RANGE ; per ta caktuar brezin adekuat, sepse duhet per krahasim, brezi ruhet ne reg. 2EH
545
546 CLR OUTSB
547 CLR OUTSA
548
549 MOV A,#00H ;fshihet A, ruhet acc.1=0, acc.0=0
550
551 CJNE A,2EH, CHECK_RANGE_02 ;krahasojme brezin e zgjedhur me brezin automatik
552 SJMP BREZI_NJEJTE_02 ;nese brezi njejte = OK
553 CHECK_RANGE_02:
554 JB C,BREZI_VOGEL_02 ;brezi me i vogel se automatik, shfaq [-1], brezi me i madh = OK, llogarit dhe shfaq sipas brezit
555
556 BREZI_NJEJTE_02:
557
558 LCALL SAMPLE ; je ri llogarit vleren dhe per brezin e zgjedhur (rezultati del mir nese vlera i takon brezit perkates, ose nese brezi zgjedhet me i madh);
559
560 MOV A,51H ;vendoset pika per milivolt (199.9 mV)
561 SETB ACC.7
562 MOV 51H,A
563 LJMP END_MANUAL_RANGE
564
565 BREZI_VOGEL_02:
566 MOV 54H,#40H
567 MOV 53H,#06H
568 MOV 52H,#00H
569 MOV 51H,#00H
570 MOV 50H,#00H
571
572 CLR 50H ;---fshihet flagu i reg. 2A sepse brezi manual
573
574 MOV C,OUTSA ;ruajme brezin e automatik per mostren ne bitat e regjistrat 29H, per EEPROM
575 MOV 49H,C
576 MOV C,OUTSB
577 MOV 48H,C
578
579 LJMP END_MANUAL_RANGE
580;

```

Fig. 31 – Zgjedh manualisht brezin milivolt (Dalja0)

Para se të zgjedhet Dalja0, fillimisht thirret rutina AUTOMATIC_RANGE, prej të cilës caktohet dalja adekuate dhe ruhet në regjistrin 2EH. Pastaj në akumulator ruhet Dalja0, mëgenëse jemi në brezin milivolt. Pastaj krahasohen këto dy breze dhe nëse brezi i zgjedhur më buton është i njejtë apo më i madh se sa brezi adekuat i caktuar nga rutina AUTOMATIC_RANGE, atëherë ky brez mund të zgjedhet, thirret edhe njëherë rutina SAMPLE për ta mostrar vlerën nga ky brez i zgjedhur me buton dhe pastaj caktohet pika të shifra e tretë sepse vlera matet në daljen zero dhe shfaqet në displej në milivolt. Mirëpo nëse kur është matur vlera nga AUTOMATIC_RANGE brezi adekuat ka dalë te një dalje më e madhe se Dalja0 e brezit milivolt psh ka dalë Dalja1, Dalja2 apo Dalja3, atëherë nuk bën të selektohet Dalja0 manualisht nga mikrokontrolleri për arsyet se mund të dëmtohet pjesa tjeter e qarkut nga tensioni i lartë në hyrje. Prandaj në këtë rast vetëm vendoset vlera [-1] në displej që tregon se ky brez nuk bën të selektohet për arsyet se nuk i përmbrush dy kriteret përmendur te seksioni i AUTOMATIC_RANGE.

Në mënyrë plotësisht analoge analizohen edhe rastet e brezeve tjera të zgjedhura manualisht nga tastatura.

Organizimi i programit në main

Programi në main i ka dy pjesë kryesore, pjesa e parë ku bëhet inicializimi i parametrave të cilët i shfrytëzon cdo rutinë më poshtë në kod, dhe pjesa e dytë shqyrtimi i flags për ta ekzekutuar një process.

```
29;-----MAIN-----  
30MAIN:  
31SETB SCL  
32SETB SDA  
33  
34SETB OUTSA  
35SETB OUTSB ; zgjedhet dalja e 4-te default ( range: +/- 199.9 V )  
36  
37MOV D PTR,#DIGITS ;D PTR = Lokacioni i numrave per 7seg  
38  
39MOV SP,#60H ; <- STACK  
40  
41SETB RS1  
42SETB RS0 ;Selektohet BANK 3  
43  
44MOV R1,#00H  
45MOV R0,#50H ;parametrat per rutinen e display  
46  
47CLR RS1 ;Selektrohet BANK 0  
48CLR RS0  
49  
50MOV IE,#82H ;mundeso interruptin per timer0  
51MOV TMOD,#02H ;timer0 8-bit auto-reload  
52MOV TL0,#6D ;250 us = 4000Hz  
53MOV TH0,#6D  
54SETB TR0 ;fillon numerimi i timer0  
55  
56MOV 42H,#4D ;4*250us = 1ms / frequency divider  
57  
58MOV 43H,#00H  
59MOV 44H,#100D ;100ms msb(43h) dhe lsb(44h)  
60  
61MOV 56H,#00H  
62MOV 57H,#100D ;100ms msb(56h) dhe lsb(57h)  
63  
64MOV 26H,#06H  
65MOV 27H,#3FH  
66MOV 28H,#3FH ;vendoset vlera 100 ne display klikon MANUAL_SAMPLE_TIME, default  
67;-----MAIN-----
```

Fig. 32 – Inicializimi i parametrave

Zgjedhet Dalja4 në fillim prej multiplekserit analog. Regjistri D PTR mban adresën e kodeve të numrave që shfaqen nga displej. Pointeri i stackut vendoset të lokacioni 60H. Selektobet banka 3. Regjistrat R1 dhe R0 mbushen më parametrat e nevojshëm për displej, pastaj selektobet banka 0 regjistrat e të cilës përdoren për llogaritje. Aktivizohet interrupti i timer 0 në modin 8-bit auto-reload, që mundëson futjen në interrupt cdo 250us. Në regjistrat 43H dhe 44H ruhet vlera 100D, ngjashëm në regjistrat 56H dhe 57H të cilët përdoren për ndryshim të vlerës së kohës se mostrimit nga tastatura.

Ndërsa pjesa e dytë është pjesa ku shqyrtohen flags e proceseve.

```

70;-----FLAGS-----
71CHECK_FLAG_EXECUTE:
72
73JNB 66H,CHECK_FLAG_EXECUTE
74CLR 66H
75
76LCALL MANUAL_RANGE
77
78JNB 67H,CHECK_FLAG_EXECUTE ;nese nuk behet asnje matje nuk dergohet asnje byte
79CLR 67H
80
81LCALL SEND_ALL_BYTES
82
83SJMP CHECK_FLAG_EXECUTE
84;-----FLAGS-----

```

Fig. 33 – Flags e proceseve

Nëse setohet një flag i një procesi, atëherë ai proces apo rutinë do të thirret në main dhe do të ekzekutohet në main në kohën kur interrupti nuk është duke ndodhur.

```

86;-----TIMER_0_ISR-----
87T0_ISR:
88MOV 78H,A
89MOV 58H,C
90DJNZ 42H,END_T0ISR ;4*250us = 1ms
91
92LCALL DISPLAY_RESULT ;shfaq rezultatin
93LCALL GET_UPDATED_BUTTONS ;checks buttons
94
95MOV 42H,#4D ;4*250us = 1ms ;Cdo 1ms mostrohet state-diagram / frequency divider
96LCALL RELATIVE_TIME_1MS ;E inkrementron counterin e kohes relative, me rezolucion 1ms
97
98;
99LCALL DECREMENT_16BIT ;Mostrohet n * 1ms, ku "n" = nr. 16 bitesh
100MOV A,44H
101JZ LSB_00
102SJMP END_T0ISR
103LSB_00:
104MOV A,43H
105JZ PERFUNDOI_KOHA_MOSTRIMIT
106SJMP END_T0ISR ;nese nuk eshte zero
107PERFUNDOI_KOHA_MOSTRIMIT:
108;rimbushet me vleren e dhene nga perdoruesi
109MOV 43H,56H ; MSB (56H) DHE LSB (57H)
110MOV 44H,57H ;100ms msb(43h) dhe lsb(44h)
111;
112
113SETB 66H ;flagu per mostrim
114
115END_T0ISR:
116MOV A,78H
117MOV C,58H
118
119RETI
120;-----TIMER_0_ISR-----

```

Fig. 34 – Interrapti i timer 0

Saherë futemi në interrupt, cdo 250us i ruajmë regjistrat dhe bitat (ACC dhe CY) që do të përdorën në interrupt dhe që eventualisht mund të prishen gjatë përdorimit nga rutinat në main. Brenda interruptit është regjistëri 42H që duhet të dekrementohet 4 herë në mënyrë që pastaj të thirret rutina për displej dhe për komunikim me tastaturë, pasi që regjistri bëhet 0 dhe vazhdohet në

interrupt regjistri prap rimbushet më vlerën 4D dhe kjo procedurë përseritet në mënyre periodike cdo 1ms. Cdoherë kur kalon 1ms, apo cdo 4 herë kur futemi në interrupt thirret rutina DECREMENT_16BIT ku dekrementohet për 1 numri 16bitësh që ruhet ne 43H dhe 44H që në main inicializohet si 100D dhe paraqet numrin e milisekondave të kohës së mostrimit.

```

999;=====
1000;                               FILLON - DECREMENT-16bit nr.
1001;=====
1002;rutina i merr parametrat nga (MSB)43H dhe 44H (LSB)
1003;numri 16-bit ne decimal eshte "n", (n*1ms) = koha e mostrimit
1004DECREMENT_16BIT:
1005DECREMENT_UNTIL_0:
1006DEC 44H
1007MOV A,44H
1008JZ IT_IS_0_NOW
1009SJMP DECREMENTED_ONCE
1010
1011IT_IS_0_NOW:
1012MOV A,43H
1013JZ DECREMENTED_ONCE
1014DEC 43H
1015MOV 44H,#0FFH
1016SJMP DECREMENT_UNTIL_0
1017
1018DECREMENTED_ONCE:
1019RET
1020;=====
1021;                               PERFUNDON - DECREMENT-16bit nr.
1022;=====
```

Fig. 35 – Dekrementimi i numrit 16-bit

Në regjistrat (MSB) 43H dhe 44H është numri i milisekondave të kohës së mostrimit. Derisa këta dy regjistra të janë zero pas dekrementimit cdo 1ms, pra derisa të kalon koha e mostrimit nuk setohet flagu i procesit për mostrim, apo flagu 66H, pasi të bëhet koha dhe para se të setohet flagu për mostrim rimbushen regjistrat për numërim të kohës se mostrimit nga regjistrat 56H dhe 57H të cilët përdoren për ndryshim të kohës së mostrimit nga tastatura.

Koha e mostrimit

Koha e mostrimit është koha në milisekonda për të cilën në mënyre periodike merret një mostër nga terminalet e voltmetrit. Në fillim të programit kjo kohë vendoset default me vlerë 100ms, pastaj mund të ndryshohet manualisht përmes tastaturës.

Në figurën 26 shihet se është një buton i dedikuar për këtë qëllim i emërtuar me etiketën SAMPLE_TIME, i cili nëse shtypet dërgon zero në bitin 6AH të regjistrit 2DH. Në figurën 30 ku bëhet shqyrtimi i butonave shihet se nëse shtypet butoni SAMPLE_TIME do të vahzdohet në rutinën ZGJEDH_MANUAL_SAMPLE_TIME.

```

863;=====
864;                               FILLON - MANUAL SAMPLE TIME
865;=====
866
867MANUAL_SAMPLE_TIME:
868JNB 6AH,MANUAL_SAMPLE_TIME ;Nuk vazhdohet ne subroutine deri sa OK button is unpressed
869
870MOV 54H,#00H
871MOV 53H,#00H
872MOV 52H,26H ;<- disp_52
873MOV 51H,27H ;<- disp_51
874MOV 50H,28H ;<- disp_50
875
876MOV R0,#50H ;regjistri per pointim te 7seg
877
878MOV R1,#0FFH ;ruan vleren per disp
879
880CHECK_BUTTONS_FOR_NEXT_PRESS:
881JNB 6AH, PERFUNDO_MANUAL_SAMPLE_TIME
882JNB 69H, INCREMENT_DISP_01
883JNB 68H, CHANGE_DISP_01
884SJMP CHECK_BUTTONS_FOR_NEXT_PRESS
885
886INCREMENT_DISP_01:
887LCALL INCREMENT_DISP
888
889NOT_UNPRESSED_00:
890JNB 69H, NOT_UNPRESSED_00 ;Nuk vazhdohet tutje, deri sa "+1" button is unpressed
891LJMP CHECK_BUTTONS_FOR_NEXT_PRESS
892
893CHANGE_DISP_01:
894MOV R1,#0FFH
895INC R0
896CJNE R0,#53H,CAN_INCREMENT_11
897MOV R0,#50H
898CAN_INCREMENT_11:
899
900NOT_UNPRESSED_01:
901JNB 68H,NOT_UNPRESSED_01 ; Nuk vazhdohet tutje, deri sa "<<" button is unpressed
902LJMP CHECK_BUTTONS_FOR_NEXT_PRESS
903
904PERFUNDO_MANUAL_SAMPLE_TIME:
905
906OK_UNPRESSED_11:
907JNB 6AH,OK_UNPRESSED_11
908
909;kur shtypet buttoni OK, ruhet vlera qe te shfaqet pastaj heren tjeter ne display kur do te caktohet nje vlere tjeter
910MOV 28H,50H
911MOV 27H,51H
912MOV 26H,52H
913
914;Kadi me poshte i ruan vlerat BCD ne reg
915LCALL SEVESEG_TO_BCD_TO_HEX ;vleren e ruan ne msb-56H dhe 57H
916
917LCALL AUTOMATIC_RANGE; <- qe te shfaqet vlera e matur
918
919RET ;perfundo caktimin e manual sample time
920
921;
922INCREMENT_DISP: ;Rutina per rritje +1 te displayt
923INC R1
924
925CJNE R1,#10D, CAN_INCREMENT ;Nese vlera me e madhe se 9 => kthehu ne 0
926MOV R1,#00H
927CAN_INCREMENT:
928MOV A,R1
929MOV @R0,A
930
931MOV A,@R0
932MOVC A,@A+DPTR
933MOV @R0,A
934RET ; perfundo rritjen e displayt
935;
936;=====
937;                               PERFUNDON - MANUAL SAMPLE TIME
938;=====

```

Fig. 36 - Rutina për caktim manual të kohës së mostrimit

Pasi të futemi në rutinën për caktim manual të kohës së mostrimit, në tre 7seg e fundit shfaqet vlera në milisekonda që ka qenë paraprakisht, e cila pastaj mund të ndryshohet me butonat “+1”

dhe “<<” që shihen në figurën 26. Kur futemi në rutinë i selektuar është 7seg i fundit, vlerën e të cilit mund ta rrisim duke shtypur butonin “+1” e cila në kod realizohet sipas rutinës INCREMENT_DISP, për ta ndërruar 7seg e shtypim butonin “<<” e cila në kod realizohet sipas rutinës CHANGE_DISP_01. Pasiqë e kemi vendosur vlerën e dëshiruar në displej dhe më të vendosim ta konfigurojmë kohën e mostrimit në mikrokontroller e shtypim butonin OK, i cili është butoni i njejtë me SAMPLE_TIME, pra përdoret i njëjtë buton për t'u future në rutinë dhe për të dalë prej saj. Para se të perfundohet rutina MANUAL_SAMPLE_TIME, vlerat e kodeve të numrave të tre 7 segmentve të fundit ruhen në regjistrat 26H nga 52H (vlera e qindshëve), 27H nga 51H (vlera e dhjetshëve) dhe 28H nga 50H (vlera e njësheve). Pastaj këto kode të vlerave duhet njëherë të kthehen në BCD pastaj në vlerë HEX që të dërgohen në regjistrat 56H dhe 57H prej të cilëve rimbushet vlera e kohës së mostrimit brenda interruptit. Për këtë qëllim thirret rutina SEVENSEG_TO_BCD_TO_HEX.

```

1050;=====
1051; FILLON - Konvertimi 7seg to BCD to HEX
1052;=====
1053;kodi me poshte i merr vlerat e dhena nga perdoruesi ne 52H, 51H, 50H (tre displayt e fundit)
1054;dhe i konverton ne BCD pastaj ne HEX
1055;vlerat i ruan MSB-56H dhe LSB-57H
1056
1057;SEVENSEG_TO_BCD_TO_HEX:
1058MOV A,#00H
1059TRY_NEW_VALUE00x:
1060MOV 47H,A ;ne 47H ruhet vlera BCD e njësheve
1061MOVC A,@+DPTR
1062CJNE A,50H,VALUE_ISNT_EQUAL00
1063SJMP CONVERT_SECOND_DIGIT
1064VALUE_ISNT_EQUAL00x:
1065MOV A,47H
1066INC A
1067SJMP TRY_NEW_VALUE00x
1068
1069CONVERT_SECOND_DIGIT:
1070MOV A,#00H
1071TRY_NEW_VALUE00x:
1072MOV 55H,A ;ne 55H ruhet vlera BCD e dhjetshëve
1073MOVC A,@+DPTR
1074CJNE A,51H,VALUE_ISNT_EQUAL00
1075SJMP CONVERT_THIRD_DIGIT
1076VALUE_ISNT_EQUAL00x:
1077MOV A,55H
1078INC A
1079SJMP TRY_NEW_VALUE00x
1080
1081CONVERT_THIRD_DIGIT:
1082MOV A,#00H
1083TRY_NEW_VALUE00x:
1084MOV 56H,A ;ne 55H ruhet vlera BCD e dhjetshëve
1085MOVC A,@+DPTR
1086CJNE A,52H,VALUE_ISNT_EQUAL00
1087;keto kalo te shumezimi te dhe mbledhja
1088SJMP SHUMEZO_VLERAT
1089VALUE_ISNT_EQUALx00:
1090MOV A,56H
1091INC A
1092SJMP TRY_NEW_VALUE00x
1093
1094SHUMEZO_VLERAT:
1095MOV A,55H
1096MOV B,#0AH ;*10
1097MUL AB
1098MOV 55H,A
1099
1100MOV A,56H
1101MOV B,#100D ;*100
1102MUL AB
1103MOV 56H,B
1104ADD A,55H
1105MOV 57H,A
1106MOV A,56H
1107ADDC A,#00H
1108MOV A,47H
1109ADD A,57H
1110MOV 57H,A
1111MOV A,56H
1112ADDC A,#00H
1113MOV 56H,A
1114RET ;RETURN FROM SEVEN SEG TO BCD
1115;=====
1116;           PERFUNDON - Konvertimi 7seg to BCD to HEX
1117;=====

```

Fig. 37 – Konvertimi i numrit prej display

Fillimisht kodet e 7seg që ruhen në 26H, 27H dhe 28H psh nëse janë ruajtur vlerat 4FH që është numri 3, 66H që është numri 4 dhe 6DH që është numri 5 (mund të shihen të DB DIGITS), kthehen në numra decimal në regjistrat përkatës, pastaj shifra e dytë pra në 27H shumëzohet më 10 dhe shifra e parë shumëzohet me 100, pastaj mbledhen tre regjistrat dhe fitohet numri 2 bajtësh (345D) që ruhet në 56H dhe 57H regjistrat që shfrytëzohën për rimbushje të regjistrave të kohës së mostrimit që dekrementohen pra 43H dhe 44H. Kjo vlerë nënkupton që cdo 345ms do të merret një mostër e re nga terminalet e voltmetrit.

Kodi në assembly

```
;----- SPI MCP32 -----
SCLK EQU 95H ; slaveClock = 95H = P1.5
MOSI EQU 96H ; Din = 96h = P1.6
SS  EQU 97H ; slaveSelect = 97H = P1.7
MISO EQU 0A0H; Dout = A0 = P2.0
;-----
```

```
;-----ANALOG MULTIPLEXER-----
OUTSA EQU 0A1H ; A pin of analog mux, 0A1H = P2.1
OUTSB EQU 0A2H ; B pin of analog mux 0A2H = P2.2
;-----
```

```
;----- Keyboard -----
SHLD EQU 0A3H ;P2.3
SHCLK EQU 0A4H ;P2.4
SO EQU 0A5H ;P2.5
;-----
```

```
;-----EEPROM-----
SCL EQU P2.6
SDA EQU P2.7
;-----
```

```
ORG 0000H
LJMP MAIN
ORG 000BH
```

LJMP T0_ISR

;-----MAIN-----

MAIN:

SETB SCL

SETB SDA

SETB OUTSA

SETB OUTSB ; zgjedhet dalja e 4-te default (range: +/- 199.9 V)

MOV DPTR,#DIGITS ;DPTR = lokacioni i numrave per 7seg

MOV SP,#60H ;<- STACK

SETB RS1

SETB RS0 ;Selektohet BANK 3

MOV R1,#00H

MOV R0,#50H ;parametrat per rutinen e display

CLR RS1 ;Selektohet BANK 0

CLR RS0

MOV IE,#82H ;mundeso interruptin per timer0

MOV TMOD,#02H ;timer0 8-bit auto-reload

MOV TL0,#6D ;250 us = 4000Hz

MOV TH0,#6D

SETB TR0 ;fillon numerimi i timer0

MOV 42H,#4D ;4*250us = 1ms / frequency divider

MOV 43H,#00H

MOV 44H,#100D ;100ms msb(43h) dhe lsb(44h)

MOV 56H,#00H

MOV 57H,#100D ;100ms msb(56h) dhe lsb(57h)

MOV 26H,#06H

MOV 27H,#3FH

MOV 28H,#3FH ;vendoset vlera 100 ne display klikon MANUAL_SAMPLE_TIME, default
;-----MAIN-----

;-----FLAGS-----

CHECK_FLAG_EXECUTE:

JNB 66H,CHECK_FLAG_EXECUTE

CLR 66H

LCALL MANUAL_RANGE

JNB 67H,CHECK_FLAG_EXECUTE ;nese nuk behet asnje matje nuk dergohet asnje byte
CLR 67H

LCALL SEND_ALL_BYTES

```

SJMP CHECK_FLAG_EXECUTE

;-----FLAGS-----

;-----TIMER_0_ISR-----

T0_ISR:
MOV 78H,A
MOV 58H,C
DJNZ 42H,END_T0ISR ;4*250us = 1ms

LCALL DISPLAY_RESULT ;shfaq rezultatin
LCALL GET_UPDATED_BUTTONS ;checks buttons

MOV 42H,#4D ;4*250us = 1ms ;Cdo 1ms mostrohet state-diagram / frequency divider
LCALL RELATIVE_TIME_1MS ;E inkrementron counterin e kohes relative, me rezolucion 1ms

;-----

LCALL DECREMENT_16BIT ;Mostrohet n * 1ms, ku "n" = nr. 16 bitesh
MOV A,44H
JZ LSB_00
SJMP END_T0ISR

LSB_00:
MOV A,43H
JZ PERFUNDOI_KOHA_MOSTRIMIT
SJMP END_T0ISR ;nese nuk eshte zero
PERFUNDOI_KOHA_MOSTRIMIT:
;rimbushet me vleren e dhene nga perdoruesi
MOV 43H,56H ; MSB (56H) DHE LSB (57H)
MOV 44H,57H ;100ms msb(43h) dhe lsb(44h)

```

;-----

SETB 66H ;flagu per mostrim

END_T0ISR:

MOV A,78H

MOV C,58H

RETI

;-----TIMER_0_ISR-----

;=====

; FILLON - KOMUNIKIMI ME MCP3208

;=====

SAMPLE: ;pas caktimit te brezit adekuat ne menyre automatike, e masim edhe njehere vleren

SETB SS ;mcp3208 nuk eshte e selektuar

CLR SCLK ;clk idles 'low' state, CPOL = 0

CLR MOSI

CLR SS; mcp3208 is selected

SETB MOSI ; START bit

SETB SCLK

NOP

CLR SCLK ;send START bit on falling edge

NOP

SETB SCLK

NOP

CLR SCLK ;send SGL bit MOSI = '1' , on falling edge

CLR MOSI

SETB SCLK

NOP

CLR SCLK ; send D2 = '0', on falling edge

NOP

SETB SCLK

NOP

CLR SCLK ;send D1 = '0', on falling edge

NOP

SETB SCLK

NOP

CLR SCLK ; send D0 = '0', on falling edge

NOP

SETB SCLK

NOP

CLR SCLK ;11th falling edge

MOV R4,#5D ;get 4 bits + nullBit

MOV A,#00H ;clr acc

GET_DATA_MSB:

NOP

SETB P2.0 ;P2.0 input pin

SETB SCLK ;(i.e 12th rising edge, null-bit is out)

NOP

```
MOV C,P2.0
RLC A
CLR SCLK
NOP ; falling edge
DJNZ R4, GET_DATA_MSB
MOV 30H,A
```

```
MOV R4,#8D ;get 8 bits
MOV A,#00H ;clr acc
GET_DATA_LSB:
NOP
SETB P2.0 ;P2.0 input pin
SETB SCLK ; (i.e 17th rising edge, null-bit is out)
NOP
MOV C,P2.0
RLC A
CLR SCLK
NOP ; falling edge
DJNZ R4, GET_DATA_LSB
MOV 31H,A
;=====
;           PERFUNDON - KOMUNIKIMI ME MCP3208
;=====

;=====;
;           FILLON - ZBRITJA, SHUMEZIMI ME 1000 DHE PDESTIMI ME 1024
;=====;
```

```
MOV 54H,#12D ; <--index for blank 1st 7seg  
;vjen vlera nga mcp3208 dhe ruhet 30H(MSB) dhe 31H(LSB)  
;krahasojme numrin a eshte me i madh se 2048 apo me i vogel  
MOV A,30H  
CJNE A,#08H,CHECK_CARRY_01 ;if (A)<(data), C=1  
MOV A,31H  
CJNE A,#00H,CHECK_CARRY_01 ;if(A) < (data), C=1  
LJMP GREATER_THAN
```

CHECK_CARRY_01:

```
JC LESS_THAN  
LJMP GREATER_THAN
```

```
;nese numri eshte me i vogel se 2048  
;gjejme 2 komplementin e numrit
```

LESS_THAN:

```
DEC 54H ; store the dptr for minus sign, index for minus sign 1st 7seg
```

```
MOV R7,#17D
```

```
;nje komplementi
```

ONE_COMPLEMENT:

```
MOV A,30H
```

```
RLC A
```

```
CPL C
```

```
MOV 30H,A
```

```
MOV A,31H
```

```
RLC A
```

```
MOV 31H,A
```

```
DJNZ R7,ONE_COMPLEMENT
```

INC 31H ; add +1 for 2's complement

JC ADD_HIGH ;check if carry again

SJMP SUBTRACT_0

ADD_HIGH: ;nese ka carry shto ne 30H

MOV A,30H

ADDC A,#00H

MOV 30H,A

;SJMP SUBTRACT_0

SUBTRACT_0:

MOV A,#10H ; 4096 D = 1000 H

ADD A, 30H ; fshijme 4 bitat msb

MOV 2FH,A

CLR 7CH

CLR 7DH

CLR 7EH

CLR 7FH

MOV 30H,2FH

GREATER_THAN:

DEC 54H

;nese numri eshte me i madh se 2048

;per ta kthyer vleren ne brez, fillimisht e zbresim me 2048

;ruajme 2 komplementin e 2048 ne (MSB)4EH-4FH, 2 komplementi = 2048 poashtu

MOV 4EH,#08H

MOV 4FH,#00H

MOV A,30H

ADD A,4EH

MOV 2FH,A ;bit adressable

CLR 7CH ;e fshijm carry pas mbledhjes me dy komplement te 2048

CLR 7DH

CLR 7EH

CLR 7FH

MOV 30H,2FH

;pastaj e shumezojme me 1000

;shumezuesi 1000 ruhet ne 33H(MSB) dhe 34H(LSB)

MOV 33H,#03H

MOV 34H,#0E8H

;kodi me poshte e shumezon numrin 12-bitesh ((MSB)30H-31H) qe vjen nga MCP3208 me 1000((MSB)33H-34H) dhe e ruan ne (MSB)30H-31H-32H

;regjistrat e prishur: 30H - 3AH

;shumezimi me 1000 behet me qellim qe te mos humbin numrat pas presjes kur pjestojme me shiftim

MOV A,31H

MOV B,34H

MUL AB

MOV 32H,A

MOV 35H,B

MOV A,30H

MOV B,34H

MUL AB

MOV 36H,A

MOV 37H,B

MOV A,33H

MOV B,31H

MUL AB

MOV 38H,A

MOV 39H,B

MOV A,33H

MOV B,30H

MUL AB

MOV 3AH,A

MOV A,35H

ADD A,36H

MOV 36H,A

MOV A,37H

ADDC A,39H

MOV 39H,A

MOV A,36H

ADD A,38H

MOV 31H,A

MOV A,39H

ADDC A,3AH

MOV 30H,A

;rezultati i shumezimit del max 3 bytes, ruhet 30H-31H-32H

;per ta pjestuar numrin e ruajtur ne 30h-31h-32h me 1024, e shiftojme 10here ne te djathte

;rezultati do te jete 2 bytes ne rastin me te keq dhe ruhet ne (MSB) 31H-32H

MOV R2,#10D

HERE:

MOV A,30H

RRC A

MOV 30H,A

MOV A,31H

RRC A

MOV 31H,A

MOV A,32H

RRC A

MOV 32H,A

CLR C

DJNZ R2, HERE

;=====

; PERFUNDON - ZBRITJA, SHUMEZIMI ME 1000 DHE Pjestimi me 1024

;=====

;=====

; FILLON - EKSTRAKTIMI I SHIFRAVE

;=====

:kodi me poshte pjeston me 10 numrin qe gjendet ne (MSB)31H-32H

;dhe e ekstrakton mbetjen, keshtu ndodh 4 here per 4 shifra

MOV R0,#4FH ; ketu ruaj mbetjen

MOV 49H,#0AH ; store 10

SHIFT AGAIN:

CJNE R2,#10H,DONT_STORE_0 ; if (R2) < (data), CY=1

LCALL STORE_REMAINDER_0

DONT_STORE_0: ;store remainder

CLR C

;perfundo programin ketu

CJNE R0,#53H,DONT_END

LJMP END_DIVISION

DONT_END:

CLR C

LCALL SHIFT_LEFT

CHECK:

MOV A,49H ; A = 10

CJNE A, 41H, CHECK_CARRY ; if (A)<(data), CY=1

SJMP EQUAL

CHECK_CARRY:

JNB C, SHIFT AGAIN

CLR C ;C fshihet qe te mos e ndikoj zbritjen

EQUAL:

MOV A,41H

SUBB A,49H

MOV 41H,A

CJNE R2,#10H,DONT_STORE_1 ; if (R2) < (data), CY=1

LCALL STORE_REMAINDER_1

DONT_STORE_1:

SETB C

LCALL SHIFT_LEFT

CLR C

SJMP CHECK

STORE_REMAINDER_0:

INC R0

MOV @R0,41H

MOV 41H,#00H

LCALL SHIFT_LEFT

MOV R2,#00H

RET

STORE_REMAINDER_1:

INC R0

MOV @R0,41H

MOV 41H,#00H

MOV R2,#0FFH

RET

SHIFT_LEFT:

INC R2

MOV A,32H

RLC A

MOV 32H,A

MOV A,31H

RLC A

MOV 31H,A

MOV A,41H

RLC A

MOV 41H,A

MOV A,40H

RLC A

MOV 40H,A

RET

END_DIVISION:

CLR C

;shifrat i ruan ne 53H, 52H, 51H,50H

;kodi me poshte i zevendson shifrat decimale me kodin e 7seg perkates, ne regjistrat perkates

MOV A,53H

MOV 59H,53H ;<-- Ruhet shifra e pare

MOVC A,@A+DPTR

MOV 53H,A

MOV A,52H

MOV 5AH,52H ;<-- Ruhet shifra e dyte

MOVC A,@A+DPTR

MOV 52H,A

MOV A,51H

MOV 5BH,51H ;<-- Ruhet shifra e parafundit

MOVC A,@A+DPTR

MOV 51H,A

```
MOV A,50H  
MOV 5CH,50H ;<--Ruhet shifra e fundit  
MOVC A,@A+DPTR  
MOV 50H,A
```

```
MOV A,54H  
;MOV 58H,54H ;<-- Ruhet parashenja #00H ose #40H  
MOVC A,@A+DPTR  
MOV 54H,A  
MOV 58H,54H ;<-- Ruhet parashenja #00H ose #40H
```

```
RET ;return from sample subroutine
```

```
;=====  
;           PERFUNDON - EKSTRAKTIMI I SHIFRAVE  
;=====  
;  
;           FILLON - KOMUNIKIMI ME TASTATURE  
;=====
```

```
GET_UPDATED_BUTTONS:
```

```
CLR SHLD ; load-mode of shiftregister with button-pressed values
```

```
CLR SHCLK ; clock of shiftregister idles in low-state
```

```
MOV R6,#8D ;8 bits of shiftregister
```

SETB SHLD ;shift-mode of shiftregister

NEXT_BUTTON:

SETB SO ;serial input pin of mcu is set '1'

MOV C,SO ;the value of shiftregister is moved to C

RLC A ;rotate-left Acc

SETB SHCLK ;low to high transition of shiftregister clock

CLR SHCLK ;high to low transition of shiftregister clock

DJNZ R6,NEXT_BUTTON ;repeat to get all 8 bits from shiftregister

MOV 2DH,A ;send 8 bits to 2DH byte, -bitAddressable

RET

;=====

; PERFUNDON - KOMUNIKIMI ME TASTATURE

;=====

;=====

; FILLON - CAKTIMI MANUAL I RANGE

;=====

;Nese dergohet zero ne lokacionin perkates, butoni eshte shtypur

MANUAL_RANGE:

;duhet te ruhet koha relative ne regjistra tjere, matet ne MSB-45H dhe LSB-46H

;dergohet ne MSB-5EH dhe LSB-5FH

MOV 5EH,45H

MOV 5FH,46H

;LCALL AUTOMATIC_RANGE ; per ta caktuar brezin adekuat, sepse duhet per krasim, brezi ruhet ne reg. 2EH

;LCALL GET_UPDATED_BUTTONS

JNB 6FH,ZGJEDH_AUTOMATIKE ;bitat e regjistrat 2DH

JNB 6EH,ZGJEDH_MILIVOLT

JNB 6DH,ZGJEDH_1_999

JNB 6CH,ZGJEDH_19_99

JNB 6BH,ZGJEDH_199_9

JNB 6AH,ZGJEDH_MANUAL_SAMPLE_TIME

LJMP END_MANUAL_RANGE

;nese asnjë button nuk shtypet, shfaqet: [- - - -]

;MOV 54H,#40H

;MOV 53H,#40H

;MOV 52H,#40H

;MOV 51H,#40H

;MOV 50H,#40H

;LJMP END_MANUAL_RANGE

;

ZGJEDH_MANUAL_SAMPLE_TIME:

LCALL MANUAL_SAMPLE_TIME ; thirret rutina e caktimit te kohes se mostrimit

LJMP END_MANUAL_RANGE ;dil nga manual_range routine

ZGJEDH_AUTOMATIKE:

LJMP ZGJEDH_AUTOMATIKE_0 ;percjell kercimin, error 8-bit relative

ZGJEDH_199_9:

LJMP ZGJEDH_199_9_0 ;percjell kercimin, error 8-bit relative

ZGJEDH_19_99:

LJMP ZGJEDH_19_99_0 ;percjell kercimin, error 8-bit relative

ZGJEDH_MILIVOLT:

LCALL AUTOMATIC_RANGE ; per ta caktuar brezin adekuat, sepse duhet per krahasim, brezi ruhet ne reg. 2EH

CLR OUTSB

CLR OUTSA

MOV A,#00H ;fshihet A, ruhet acc.1=0, acc.0=0

CJNE A,2EH, CHECK_RANGE_02 ;krahasojme brezin e zgjedhur me brezin automatik

SJMP BREZI_NJEJTE_02 ;nese brezi njejte = OK

CHECK_RANGE_02:

JB C,BREZI_VOGEL_02 ; brezi me i vogel se automatik, shfaq [-1], brezi me i madh = OK, llogarit dhe shfaq sipas brezit

BREZI_NJEJTE_02:

LCALL SAMPLE ;e ri llogarit vleren dhe per brezin e zgjedhur (rezultati del mir nese vlera i takon brezit perkates, ose nese brezi zgjedhet me i madh);

MOV A,51H ;vendoset pikë per milivolt (199.9 mV)

SETB ACC.7

MOV 51H,A

LJMP END_MANUAL_RANGE

BREZI_VOGEL_02:

MOV 54H,#40H

MOV 53H,#06H

MOV 52H,#00H

MOV 51H,#00H

MOV 50H,#00H

CLR 50H ;<--fshihet flagu i reg. 2A sepse brezi manual

MOV C,OUTSA ;ruajme brezin e automatik per mostren ne bitat e regjistrat 29H, per EEPROM

MOV 49H,C

MOV C,OUTSB

MOV 48H,C

LJMP END_MANUAL_RANGE

ZGJEDH_1_999:

LCALL AUTOMATIC_RANGE ; per ta caktuar brezin adekuat, sepse duhet per krasim, brezi ruhet ne reg. 2EH

CLR OUTSB

SETB OUTSA

MOV A,#00H ;fshihet A

MOV C,OUTSA ;ruhet brezi ne A

MOV ACC.0,C

MOV C,OUTSB

MOV ACC.1,C

CJNE A,2EH, CHECK_RANGE_00 ;krahasojme brezin e zgjedhur me brezin automatik

SJMP BREZI_NJEJTE_00 ;nese brezi njejte = OK

CHECK_RANGE_00:

JB C,BREZI_VOGEL_00 ; brezi me i vogel se automatik, shfaq [-1], brezi me i madh = OK,
llogarit dhe shfaq sipas brezit

BREZI_NJEJTE_00:

LCALL SAMPLE

MOV A,53H ;vendoset pika per milivolt (1.999 V)

SETB ACC.7

MOV 53H,A

LJMP END_MANUAL RANGE

BREZI_VOGEL_00:

MOV 54H,#40H ;-

MOV 53H,#06H ;1

MOV 52H,#00H

MOV 51H,#00H

MOV 50H,#00H

CLR 50H ;<--fshihet flagu i reg. 2A sepse brezi manual

MOV C,OUTSA ;ruajme brezin e automatik per mostren ne bitat e regjistrit 29H, per EEPROM

MOV 49H,C

MOV C,OUTSB

MOV 48H,C

SJMP END_MANUAL_RANGE

ZGJEDH_19_99_0:

LCALL AUTOMATIC_RANGE ; per ta caktuar brezin adekuat, sepse duhet per krahasim, brezi ruhet ne reg. 2EH

SETB OUTSB

CLR OUTSA

MOV A,#00H ;fshihet A

MOV C,OUTSA ;ruhet brezi ne A

MOV ACC.0,C

MOV C,OUTSB

MOV ACC.1,C

CJNE A,2EH, CHECK_RANGE_01 ;krahasojme brezin e zgjedhur me brezin automatik

SJMP BREZI_NJEJTE_01 ;nese brezi njejte = OK

CHECK_RANGE_01:

JB C,BREZI_VOGEL_01 ; brezi me i vogel se automatik, shfaq [-1], brezi me i madh = OK, llogarit dhe shfaq sipas brezit

BREZI_NJEJTE_01:

LCALL SAMPLE

MOV A,52H ;vendoset pika per milivolt (19.99 V)

SETB ACC.7

MOV 52H,A

SJMP END_MANUAL_RANGE

BREZI_VOGEL_01:

MOV 54H,#40H ;-

MOV 53H,#06H ;1

MOV 52H,#00H

MOV 51H,#00H

MOV 50H,#00H

CLR 50H ;<--fshihet flagu i reg. 2A sepse brezi manual

MOV C,OUTSA ;ruajme brezin e automatik per mostren ne bitat e regjistrit 29H, per EEPROM

MOV 49H,C

MOV C,OUTSB

MOV 48H,C

SJMP END_MANUAL_RANGE

ZGJEDH_199_9_0:

LCALL AUTOMATIC_RANGE ; per ta caktuar brezin adekuat, sepse duhet per krahasim, brezi ruhet ne reg. 2EH

SETB OUTSB

SETB OUTSA

LCALL SAMPLE

MOV A,51H ;vendoset pika per milivolt (199.9 V)

SETB ACC.7

MOV 51H,A

CLR 50H ;<--fshihet flagu i reg. 2A sepse brezi manual

MOV C,OUTSA ;ruajme brezin e automatik per mostren ne bitat e regjistrit 29H, per EEPROM

MOV 49H,C

MOV C,OUTSB

MOV 48H,C

SJMP END_MANUAL_RANGE

ZGJEDH_AUTOMATIKE_0:

LCALL AUTOMATIC_RANGE

MOV C,OUTSA ;ruajme brezin e automatik per mostren ne bitat e regjistrit 29H, per EEPROM

MOV 49H,C

MOV C,OUTSB

MOV 48H,C

END_MANUAL_RANGE:

RET ;return from manual range

;=====

; MBARON - CAKTIMI MANUAL I RANGE

;=====

;=====

; FILLON - AUTOMATIC RANGE

;=====

AUTOMATIC_RANGE:

SETB 50H ;<-- Setohet biti 0 i reg. 2AH nese brezi automatik

SETB 67H ; <-- Setohet flagu qe tregon se matja eshte realizuar dhe mund te dergohen bytet ne EEPROM

;by default eshte i zgjedhur brezi 4 (+/- 199.9 V)

;shikojm sa zero permban numri 4-shifror

SETB OUTSA ;zgjedhet range 199.9V

SETB OUTSB

LCALL SAMPLE

;-----

MOV A,53H

CJNE A,#3FH,NOT_NEXT0 ; check if zero, 3FH eshte vlera e 7seg per nr. 0

SJMP NEXT0

NOT_NEXT0:

```
MOV A,51H ;vendos piken per range 199.9  
SETB ACC.7  
MOV 51H,A
```

SJMP DALJA4_0 ;nenkupton qe brezi eshte adekuat (+/- 199.9 V)

DALJA4_0:

```
LJMP DALJA4 ;error 8-bit relative  
;-----
```

NEXT0:

```
MOV A,52H
```

CJNE A,#3FH,NOT_NEXT1 ;check if zero

```
SJMP NEXT1
```

NOT_NEXT1:

;nese display i dyte nuk eshte zero - duhet shikohet nese eshte 1 i takon brezit, nese eshte i ndryshem nuk i takon

```
MOV A,52H
```

```
CJNE A,#06H,NOT_ONE_0
```

```
SETB OUTSB ;zgjedhet range 19.99
```

```
CLR OUTSA
```

LCALL SAMPLE

```
MOV A,52H ;vendoset pika per range 19.99  
SETB ACC.7  
MOV 52H,A
```

SJMP DALJA_TJETER

NOT_ONE_0:

SETB OUTSA ;zgjedhet range 199.9

SETB OUTSB

LCALL SAMPLE

MOV A,51H ;vendoset pika per range 19.99

SETB ACC.7

MOV 51H,A

SJMP DALJA_TJETER ;nenkupton qe vetem display i pare i shifrave eshte zero zgjedhet brezi
(+/- 19.99 V)

NEXT1:

MOV A,51H

CJNE A,#3FH,NOT_NEXT2 ;check if zero

SJMP NEXT2

NOT_NEXT2:

;nese display i trete nuk eshte zero - duhet te shikohet nese eshte 1 i takon brezit, nese eshte i ndryshem nuk i takon

MOV A,51H

CJNE A,#06H,NOT_ONE_1

CLR OUTSB ;zgjedhet range 1.999

SETB OUTSA

LCALL SAMPLE

MOV A,53H ;vendoset pika per range 1.999

SETB ACC.7

MOV 53H,A

SJMP DALJA_TJETER

NOT_ONE_1: ;nuk eshte 1, nuk i takon brezit

SETB OUTSB ;zgjedhet range 19.99

CLR OUTSA

LCALL SAMPLE

MOV A,52H ;vendoset pika per range 19.99

SETB ACC.7

MOV 52H,A

SJMP DALJA_TJETER ;nenkupton qe displayt 1 dhe 2 te shifrave jane zero zgjedhet brezi (+/- 1.999 V)

NEXT2:

MOV A,50H

CJNE A,#3FH,NOT_NEXT3 ;check if zero

SJMP NEXT3

NOT_NEXT3:

;nese display i fundit nuk eshte zero - duhet shikohet nese eshte 1 ose jo, por fillimisht e ndrrojm brezin per rritje te sakesise (per raste kufitare)

;e zgjedhim brezin 19.99, per dallim nga rastet me larte

```
SETB OUTSB  
CLR OUTSA  
LCALL SAMPLE  
;tani nuk e shikojme displayn e fundit (50H) por displayn e parafundit  
MOV A,51H  
CJNE A,#06H,NOT_ONE_2  
;nese eshte 1, zgjedhet brezi 0.1999
```

CLR OUTSB ;zgjedhet range 0.1999

CLR OUTSA

LCALL SAMPLE

```
MOV A,51H ;vendoset pikë per milivolt (199.9 mV)
```

SETB ACC.7

MOV 51H,A

SJMP DALJA_TJETER

NOT_ONE_2: ;nese nuk eshte 1, zgjedhet brezi 1.999V

CLR OUTSB

SETB OUTSA

LCALL SAMPLE

```
MOV A,53H ;vendoset pikë per range 1.999
```

SETB ACC.7

MOV 53H,A

SJMP DALJA_TJETER ;nenkupton qe displayt 1,2,3 te shifrave jane zero zgjedhet brezi (+/- 0.1999 V)

;-----

NEXT3:

CLR OUTSB

CLR OUTSA ;nenkupton qe 4 displayt jane zero zgjedhet brezi (+/- 0.1999 V)

LCALL SAMPLE

MOV A,51H ;vendoset pika per milivolt (019.9 mV)

SETB ACC.7

MOV 51H,A

DALJA_TJETER:

DALJA4: ; brezi eshte adekuat, vetem vazhdon ne rastin e +/- 199.9 V

MOV C,OUTSA ;ruajme brezin e automatik per mostren ne bitat e regjistrit 2EH (bit-addressable)

MOV 70H,C

MOV C,OUTSB

MOV 71H,C

RET

;=====

; PERFUNDON - AUTOMATIC RANGE

;=====

```
;=====;  
; FILLON - MANUAL SAMPLE TIME  
=====;
```

MANUAL_SAMPLE_TIME:

JNB 6AH,MANUAL_SAMPLE_TIME ;Nuk vazhdohet ne subroutine deri sa OK button is unpressed

MOV 54H,#00H

MOV 53H,#00H

MOV 52H,26H ;<- disp_52

MOV 51H,27H ;<- disp_51

MOV 50H,28H ;<- disp_50

MOV R0,#50H ;regjistri per pointim te 7seg

MOV R1,#0FFH ;ruan vleren per disp

CHECK_BUTTONS_FOR_NEXT_PRESS:

JNB 6AH, PERFUNDO_MANUAL_SAMPLE_TIME

JNB 69H, INCREMENT_DISP_01

JNB 68H, CHANGE_DISP_01

SJMP CHECK_BUTTONS_FOR_NEXT_PRESS

INCREMENT_DISP_01:

LCALL INCREMENT_DISP

NOT_UNPRESSED_00:

JNB 69H, NOT_UNPRESSED_00 ;Nuk vazhdohet tutje, deri sa "+1" button is unpressed
LJMP CHECK_BUTTONS_FOR_NEXT_PRESS

CHANGE_DISP_01:

MOV R1,#0FFH

INC R0

CJNE R0,#53H,CAN_INCREMENT_11

MOV R0,#50H

CAN_INCREMENT_11:

NOT_UNPRESSED_01:

JNB 68H,NOT_UNPRESSED_01 ; Nuk vazhdohet tutje, deri sa "<<" button is unpressed

LJMP CHECK_BUTTONS_FOR_NEXT_PRESS

PERFUNDO_MANUAL_SAMPLE_TIME:

OK_UNPRESSED_11:

JNB 6AH,OK_UNPRESSED_11

;kur shtypet buttoni OK, ruhet vlera qe te shfaqet pastaj heren tjeter ne display kur do te caktohet nje vlere tjeter

MOV 28H,50H

MOV 27H,51H

MOV 26H,52H

;Kodi me poshte i ruan vlerat BCD ne reg

LCALL SEVESEG_TO_BCD_TO_HEX ;vleren e ruan ne msb-56H dhe 57H

LCALL AUTOMATIC_RANGE; <-- qe te shfaqet vlera e matur

RET ;perfundo caktimin e manual sample time

;-----

INCREMENT_DISP: ;Rutina per rritje +1 te displayt

INC R1

CJNE R1,#10D, CAN_INCREMENT ;Nese vlera me e madhe se 9 => kthehu ne 0

MOV R1,#00H

CAN_INCREMENT:

MOV A,R1

MOV @R0,A

MOV A,@R0

MOVC A,@A+DPTR

MOV @R0,A

RET ; perfundo rritjen e displayt

;-----

;=====

; PERFUNDON - MANUAL SAMPLE TIME

;=====

;=====

; FILLON - SHFAQJA E SHIFRAVE

;=====

DISPLAY_RESULT:

SETB P1.4

SETB P1.3

SETB P1.2

SETB P1.1

SETB P1.0

SETB RS1

SETB RS0 ;Selektohet BANK 3

MOV P0,@R0 ;ne fillim (R0) = 50H ndersa (R1) = 00H

CJNE R1,#00H,CHECK_1

CLR P1.4

SJMP END_DISPP

CHECK_1:

CJNE R1,#01H,CHECK_2

CLR P1.3

SJMP END_DISPP

CHECK_2:

CJNE R1,#02H,CHECK_3

CLR P1.2

SJMP END_DISPP

CHECK_3:

CJNE R1,#03H,CHECK_4

CLR P1.1

SJMP END_DISPP

CHECK_4:

CJNE R1,#04H,CHECK_5

CLR P1.0

SJMP END_DISPP

CHECK_5:

END_DISPP:

INC R0

INC R1

CJNE R0,#55H,CHECK_6

MOV R0,#50H

MOV R1,#00H

CHECK_6:

CLR RS1

CLR RS1

RET ; perfundon rutina e display

DIGITS: DB 3FH, 06H, 5BH, 4FH, 66H, 6DH, 7DH, 07H, 7FH, 6FH, 40H, 00H

;minusi-->^ ^<---blank

=====

; PERFUNDON - SHFAQJA E SHIFRAVE

=====

=====

; FILLON - DECREMENT-16bit nr.

=====

;rutina i merr parametrat nga (MSB)43H dhe 44H (LSB)

;numri 16-bit ne decimal eshte "n", (n*1ms) = koha e mostrimit

DECREMENT_16BIT:

DECREMENT_UNTIL_0:

DEC 44H

MOV A,44H

JZ IT_IS_0_NOW

SJMP DECREMENTED_ONCE

IT_IS_0_NOW:

MOV A,43H

JZ DECREMENTED_ONCE

DEC 43H

MOV 44H,#0FFH

SJMP DECREMENT_UNTIL_0

DECREMENTED_ONCE:

RET

=====

; PERFUNDON - DECREMENT-16bit nr.

=====

=====

; FILLON - Counter - Kohe Relative

=====

;rutina i ruan parametrat ne MSB-45H dhe LSB-46H

RELATIVE_TIME_1MS:

INC 46H

```

MOV A,46H
CJNE A,#0FFH, NOT_OVERFLOW
MOV 46H,#00H
INC 45H

NOT_OVERFLOW:
MOV A,45H
CJNE A,#0FFH,END_REL_TIME_ROUTINE
MOV A,46H
CJNE A,#0FFH,END_REL_TIME_ROUTINE
MOV 46H,#00H ;Reset counter of relative time
MOV 45H,#00H
END_REL_TIME_ROUTINE:
RET
;=====
;           PERFUNDON - Counter - Kohe Relative
;=====

;=====
;           FILLON - Konvertimi 7seg to BCD to HEX
;=====

;kodi me poshte i merr vlerat e dhena nga perdoruesi ne 52H, 51H, 50H (tre displayt e fundit)
;dhe i konverton ne BCD pastaj ne HEX
;vlerat i ruan MSB-56H dhe LSB-57H

SEVESEG_TO_BCD_TO_HEX:
MOV A,#00H

```

TRY_NEW_VALUE00x:

MOV 47H,A ;ne 47H ruhet vlera BCD e njesheve

MOVC A,@A+DPTR

CJNE A,50H,VALUE_ISNT_EQUAL00x

SJMP CONVERT_SECOND_DIGIT

VALUE_ISNT_EQUAL00x:

MOV A,47H

INC A

SJMP TRY_NEW_VALUE00x

CONVERT_SECOND_DIGIT:

MOV A,#00H

TRY_NEW_VALUE0x0:

MOV 55H,A ;ne 55H ruhet vlera BCD e dhjetsheve

MOVC A,@A+DPTR

CJNE A,51H,VALUE_ISNT_EQUAL0x0

SJMP CONVERT_THIRD_DIGIT

VALUE_ISNT_EQUAL0x0:

MOV A,55H

INC A

SJMP TRY_NEW_VALUE0x0

CONVERT_THIRD_DIGIT:

MOV A,#00H

TRY_NEW_VALUEEx00:

MOV 56H,A ;ne 55H ruhet vlera BCD e dhjetsheve

MOVC A,@A+DPTR

CJNE A,52H,VALUE_ISNT_EQUALx00

;ketu kalo te shumezimi te dhe mbledhja

SJMP SHUMEZO_VLERAT

VALUE_ISNT_EQUALx00:

MOV A,56H

INC A

SJMP TRY_NEW_VALUEx00

SHUMEZO_VLERAT:

MOV A,55H

MOV B,#0AH ;*10

MUL AB

MOV 55H,A

MOV A,56H

MOV B,#100D ;*100

MUL AB

MOV 56H,B

ADD A,55H

MOV 57H,A

MOV A,56H

ADDC A,#00H

MOV A,47H

ADD A,57H

MOV 57H,A

MOV A,56H

ADDC A,#00H

MOV 56H,A

RET ;RETURN FROM SEVEN SEG TO BCD

```

;=====;
;           PERFUNDON - Konvertimi 7seg to BCD to HEX
;=====;

;=====;
;           FILLON - Dergimi i te gjithe Bytes
;=====;

SEND_ALL_BYTES:
;5DH eshte buffer per data
MOV 5DH,5EH ;<- LOW of relative TIME
LCALL SEND_ONE_BYTE_EEPROM
LCALL COUNT_ADDRESS

MOV 5DH,5FH ;<- HIGH of relative TIME
LCALL SEND_ONE_BYTE_EEPROM
LCALL COUNT_ADDRESS

MOV 5DH,2AH ;<- MANUAL=0/AUTOMATIC=1 Range
LCALL SEND_ONE_BYTE_EEPROM
LCALL COUNT_ADDRESS

MOV 5DH,29H ;<- Dalja 0,1,2,3
LCALL SEND_ONE_BYTE_EEPROM
LCALL COUNT_ADDRESS

MOV 5DH,5CH ;shifra e fundit
LCALL SEND_ONE_BYTE_EEPROM

```

LCALL COUNT_ADDRESS

MOV 5DH,5BH ;shifra e parafundit

LCALL SEND_ONE_BYTE_EEPROM

LCALL COUNT_ADDRESS

MOV 5DH,5AH ;shifra e dyte

LCALL SEND_ONE_BYTE_EEPROM

LCALL COUNT_ADDRESS

MOV 5DH,59H ;shifra e pare

LCALL SEND_ONE_BYTE_EEPROM

LCALL COUNT_ADDRESS

MOV 5DH,58H ; parashenja #00H ose 40H

LCALL SEND_ONE_BYTE_EEPROM

LCALL COUNT_ADDRESS

RET

;=====

; PERFUNDON - Dergimi i te gjithe Bytes

;=====

;=====

; FILLON - Komunikimi me EEPROM

;=====

SEND_ONE_BYTE_EEPROM:

```
;----START----  
CLR SDA  
LCALL SHORT_DELAY  
CLR SCL  
LCALL SHORT_DELAY  
;----/START----  
  
;---SLAVE-ADDRESS---  
MOV A,#0A0H  
LCALL SHIFT_DATA_OUT  
LCALL ACK_BIT_CHECK  
  
JNC AGAIN_SLAVE_ADDRESS  
SJMP $  
AGAIN_SLAVE_ADDRESS:  
;---/SLAVE-ADDRESS---  
  
;-----WORD ADDRESS - 15 bit --  
;WORD_ADDRESS:  
LCALL SHORT_DELAY  
MOV A,3EH ; <- (vendoset msb i adreses)  
LCALL SHIFT_DATA_OUT  
LCALL ACK_BIT_CHECK  
JNC WORD_ADDRESS  
SJMP $  
WORD_ADDRESS:  
LCALL SHORT_DELAY
```

MOV A,3FH ;<-- (vendoset lsb i adreses)

LCALL SHIFT_DATA_OUT

LCALL ACK_BIT_CHECK

JNC WORD_ADDRESS_1

SJMP \$

WORD_ADDRESS_1:

;----WORD ADDRESS - 15bit --

;---DATA-----

MOV A,5DH ;<-- duhet ndryshohet (vendoset regjistri i data)

LCALL SHORT_DELAY

LCALL SHIFT_DATA_OUT

LCALL ACK_BIT_CHECK

JNC DATA_WRITE

SJMP \$

DATA_WRITE:

;---DATA-----

;---STOP-----

SETB SCL

LCALL SHORT_DELAY

SETB SDA

LCALL SHORT_DELAY

;----/STOP----

```
RET ;return from SEND_ONE_BYTE_EEPROM
```

```
SHORT_DELAY: ; delay e shkurt  
MOV R0,#4D  
DJNZ R0,$  
RET
```

```
SHIFT_DATA_OUT: ;i dergon 8 bits ne bus each at a time  
MOV R1,#8D  
SHIFT_DATA_OUT0:  
RLC A  
MOV SDA,C  
LCALL SHORT_DELAY  
SETB SCL  
LCALL SHORT_DELAY  
CLR SCL  
LCALL SHORT_DELAY  
DJNZ R1,SHIFT_DATA_OUT0  
RET
```

```
ACK_BIT_CHECK:  
SETB SDA  
LCALL SHORT_DELAY  
SETB SCL  
LCALL SHORT_DELAY  
MOV C,SDA
```

```

LCALL SHORT_DELAY

CLR SCL

LCALL SHORT_DELAY ;<--

CLR SDA

RET

=====

;           PERFUNDON - Komunikimi me EEPROM

=====

=====

;           FILLON - Counteri i Addressave per EEPROM

=====

;rutina i ruan parametrat ne MSB-3EH dhe LSB-3FH

COUNT_ADDRESS:

INC 3FH

MOV A,3FH

CJNE A, #0FFH, REG_NOT_OVERFLOW_22

MOV A,3EH

CJNE A,#07FH, REG_NOT_OVERFLOW_23

INC 3FH

MOV 3EH,#00H

SJMP END_COUNTER_ADDRESS

REG_NOT_OVERFLOW_23:

INC 3EH

SJMP END_COUNTER_ADDRESS

REG_NOT_OVERFLOW_22:

END_COUNTER_ADDRESS:

RET

```

```
;=====;  
;          PERFUNDON - Counteri i Addresave per EEPROM  
=====;  
END
```