

UNIVERZITET U BEOGRADU
ELEKTROTEHNICKI FAKULTET

Katedra za elektroniku

Predmet: Racunarska elektronika



PROJEKAT 2015/2016

Predmetni profesor: **Dr Milan Prokin**

Predmetni asistent: **Aleksandra Lekic**

Projekat radili:

Ime	Prezime	broj indeksa
Aleksandra	Skrbic	0163/2012
Selena	Colovic	0084/2012

Datum: 15.06.2016. (last revision)

SADRZAJ

1. Postavka zadatka	3
2. Koncept realizacije programa	4
3. Realizacija (izvorni kod)	6
• AddTwo.asm (ostavljen je ovaj naziv).....	6
4. Zakljucak	12
5. Literatura	13

1. POSTAVKA ZADATAKA

Projektni zadatak koji nam je bio dodeljen, je pod rednim brojem 15 na listi projekata. Trebalo je uraditi sledece: dozvoliti korisniku da unese ime fajla sa ekstenzijom **.pls** u prozoru konzole, a potom je bilo neophodno izvršiti izvesnu obradu tj. promenu formata fajla i cuvanje rezultata obrade u izlazni fajla sa ekstenzijom **.m3u**. Na sajtu je dat primer izgleda obe liste. I konkretno u projektu koristila sam bas te liste za validaciju funkcionalnosti koda. U nastavku ce biti dati primer prikaza **.pls** ulazne liste i **.m3u** izlazne liste u **Notepad++**.

Primer input fajla:

[playlist]

NumberOfEntries=50

File1=nesto-nesto.mp3

Title1=nesto- nesto.mp3

Length1=1

File2=E:\Muzika\Dream Theater\Falling Into Infinity\Dream Theater_Falling Into Infinity_01_New Millennium.mp3

Title2=Dream Theater - New Millennium

Length2=500

File3=E:\Muzika\Dream Theater\Falling Into Infinity\Dream Theater_Falling Into Infinity_02_You Not Me.mp3

Title3=Dream Theater - You Not Me

Length3=298

Primer output fajla:

#EXTM3U

#EXTINF:1,nesto- nesto.mp3

nesto-nesto.mp3

#EXTINF:500,Dream Theater - New Millennium

E:\Muzika\Dream Theater\Falling Into Infinity\Dream Theater_Falling Into Infinity_01_New Millennium.mp3

#EXTINF:298,Dream Theater - You Not Me

E:\Muzika\Dream Theater\Falling Into Infinity\Dream Theater_Falling Into Infinity_02_You Not Me.mp3

2. KONCEPT REALIZACIJE PROGRAMA

Dakle, u okviru ovog poglavlja objasnicu ideju za realizaciju samog programa za zahtevanu funkcionalnost.

Najpre je bilo potrebno dozvoliti korisniku da unese ime ulaznog fajla u prozoru konzole (**.pls** fajl). To omogućavamo pozivom fje **ReadString** za niz unet od strane korisnika. Ime koje korisnik unosi cuvacemo u **srcFilename** stringu i to cemo kasnije obraditi, da bismo dobili isto ime za izlazni fajl, samo sa drugom extenzijom (**.m3u**). Zatim, pozivom fje **OpenInputFile**, otvaramo fajl ukoliko on postoji. Sada, sledi provera validnosti unetog imena tj. postojanja ulaznog fajla: to se odnosi na **fileHandle**, odnosno mogucnost ispravnog otvaranja fajla (fajl postoji i nije prazan). Kada je ta provera prosla, ucitavamo ulazni fajl u bafer koji smo inicijalizovali u **.data** segmentu koda. Pozivamo fju **ReadFromFile**. Odrediste ce biti pomenuti bafer, a sa fajlom komuniciramo preko **fileHandle**-a. Takodje, vrsimo proveru velicine bafera za ulazni fajl i ispis odgovarajuce poruke ukoliko greska prilikom izbora velicine postoji. Ukoliko je velicina bafera odgovarajuca, prikazujemo je u **GUI** prozoru. Potom sam izvorsila ispis vrednosti smestenih u baferu takodje, radi provere validnog citanja svih podataka iz ulaznog fajla. Za kraj zatvaramo ulazni fajl, jer nam vise ne treba; ucitan je u bafer.

Sada, sledi kompleksniji deo projekta, a to je obrada sadrzaja pomenutog bafera i cuvanje rezultata u izlazni bafer. Najpre je uz pomoc instrukcije bez operanada **cld** definisano da ce se stringu pristupati u inkrementirajucem poretku.

Ideja je bila sledeca: citamo u baferu podatke sve dok ne dodjemo do **"=**". Potom preskacemo taj red (taj red nam za obradu ne treba), i dolazimo do reda koji sadrzi, za nasu obradu, bitne podatke. Taj red citamo do kraja. Ovo radimo koristeći registre za *Source Index* (**ESI**) on je pointer na elemente bafera sa ulaznim fajlom. U **ACC** (low part, **AL**) stavljamo stringove koji su nam od znacaja: **"=**" i **0dh** sto je simbol za prelazak u novi red. Kada ocitamo red koji nam je neophodan za dalju obradu, **ESI** vrednost cuvamo da znamo gde smo stali u obradi ulaznog fajla, a **ACC** koristimo kao registar koji nam moze pomoci da se obracamo podacima koji su nam u ovom redu znacajni, i potom ih stavljamo na stek (**ESP** - stek pointer). Kada se pogleda trazeni format liste na izlazu, kao i same kke steka, tj. kako raste i na sta ukazuje, citacemo karaktere unazad sve do **"=**". Sada smo na stek stavili string koji predstavlja putanju do date numere playlist-e. Potom nastavljamo da citamo tamo gde smo stali (**ESI**). Citamo sve do kraja sledeceg reda. Ponovo idemo sa **ACC** registrom kao pointerom unazad, sve dok ne dodjemo do **"=**". Do tog trenutka smestamo karaktere na stek, tako da sada kada zavrsi ovu petlju, program je na stek smestio putanju, ali i naziv numere u listi. Potom nastavljamo sa daljom obradom i citamo ceo sledeci red (tj. do kraja sledeceg stringa) u okviru koga je sakriven podatak o trajanju numere. Ovde ce postojati i specijalan slucaj kada necemo ocitavati znak za novi red (**0dh**) nego **0h** (**EOF**). U tom slucaju, nakon upisa podatka na stek, samo je potrebno dekrementirati **ESI** i to ne zapisujemo na stek vec samo prelazimo na ispis u izlazni bafer. U

protivnom, ukoliko je klasicko procitan novi red, takodje smestimo podatak o vremenu na stek citajuci unazad do "=" i nakon toga prelazimo na ispis u izlazni bafer.

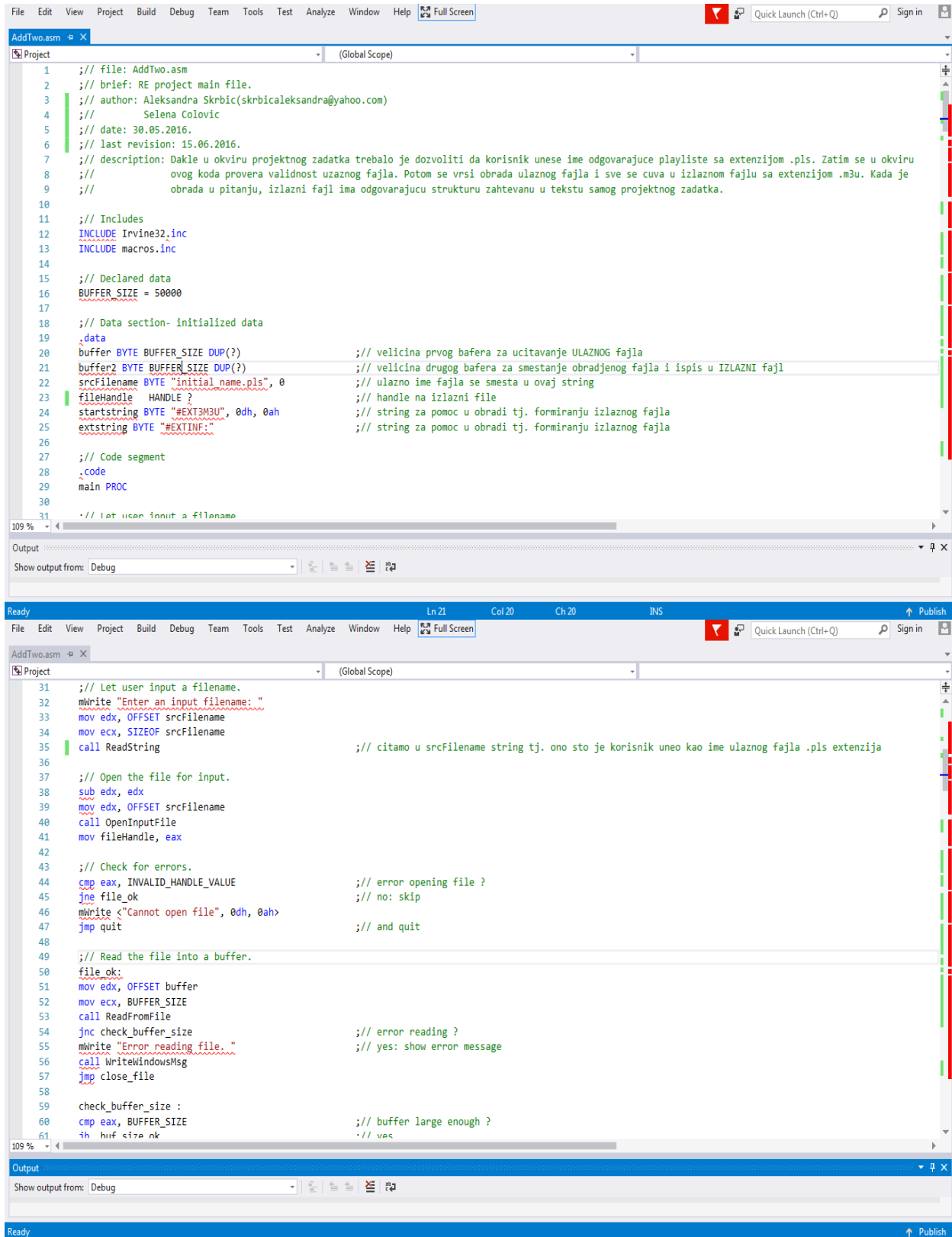
Dakle svaki put nakon smestanja vrednosti u odgovarajucoj formi za svaku numeru liste: **File**, **Title** i **Length**, mi cemo odmah sa steka sve upisivati u izlazni bafer. Ukoliko je prva iteracija (pocetak izlaznog fajla), a za to cemo imati pomocnu promenljivu tj. brojac (**BL**) da nam javi da je pitanju prvi upis, imacemo upis kljucne reci, smestene **startstring**, koju cemo ispisivati u izlaznom baferu. Potom ce slediti standardni upis za svaku numeru liste. Najpre ispisujemo string **extstring** na pocetk reda, zatim sa steka skidamo podatak o poslednjoj upisanoj vrednosti za **Length** (preskacemo ono 0dh sa kojim se string završavao), i smestamo ga u izlazni bafer. Potom upisujemo ",". Nakon toga na slican nacin skidamo sa steka vrednost za **Title** numere. Sada i nju smestamo u izlazni bafer. Potom prelazimo na novi red i tu upisujemo novu vrednost koju skidamo sa steka, a to je preostala **File** vrednost za numeru. Sada smo kompletno izvršili obradu ulaznih vrednosti na odgovarajuci nacin za jednu numeru u listi. Sada samo inkrementiramo brojac **BL**, da oznacimo da nema vise potrebe za **startstring**-om. Potom proveravamo da li je **EOF**. Ukoliko nije ovu obradu vrsimo za svaku narednu numeru u playlist-i. Ukoliko smo dosli do kraja fajla, potrebno je da kreiramo izlazni fajl u koji cemo smestiti sve rezultate obrade iz bafera2.

Ponovo sam izvršila pozive odgovarajucih instrukcija za settovanje rada sa stringom. Zatim je usledilo formiranje imena za izlazni fajl, tako sto sam citala string formiran na osnovu unosa korisnika u prozoru, **srcFilename** i to sve dok nisam dosla do ".". Zatim je samo promenjena extenzija ulaznog fajla prepisivanjem na mesto "**pls**", "**m3u**". Na ovaj nacin imali smo validno ime fajla koje je zahtevano u textu projekta.

Za sam kraj, prosledjivanjem odgovarajcih parametara odnosno smestanjem odg. vrednosti u registre i pozivom fja **CreateOutputFile**, kao i **WriteToFile** krairan je output fajl odgovarajuceg imena i extenzije, i u njega je upisan sadrzaj izlaznog bafera (buffer2).

Time je trazena funkcionalnost u potpunosti zadovoljena. Kod je testiran na ulaznoj **playlist.pls**, a rezultat je bila izlazna lista **playlist.m3u**, koja je u potpunosti identicna zahtevanoj formi u textu projektnog zadatka. Obe su prilozene u **.zip** datoteci.

3. REALIZACIJA (IZVORNI KOD)



```
1  ;// file: AddTwo.asm
2  ;// brief: RE project main file.
3  ;// author: Aleksandra Skrbic(skrbica@sandra@yahoo.com)
4  ;//
5  ;// date: 30.05.2016.
6  ;// last revision: 15.06.2016.
7  ;// description: Dakle u okviru projektnog zadatka trebalo je dozvoliti da korisnik unese ime odgovarajuće playliste sa ekstenzijom .pls. Zatim se u okviru
8  ;// ovog koda provjera validnost uzasnog fajla. Potom se vrši obrada ulaznog fajla i sve se čuva u izlaznom fajlu sa ekstenzijom .m3u. Kada je
9  ;// obrada u pitanju, izlazni fajl ima odgovarajuću strukturu zahtevanu u tekstu samog projektnog zadatka.
10
11  ;// Includes
12  INCLUDE Irvine32.inc
13  INCLUDE macros.inc
14
15  ;// Declared data
16  BUFFER_SIZE = 50000
17
18  ;// Data section- initialized data
19  .data
20  buffer BYTE BUFFER_SIZE DUP(?) ;// velicina prvog bafera za učitavanje ULAZNOG fajla
21  buffer2 BYTE BUFFER_SIZE DUP(?) ;// velicina drugog bafera za smestanje obradjenog fajla i ispis u IZLAZNI fajl
22  srcFilename BYTE "initial name.pls", 0 ;// ulazno ime fajla se smesta u ovaj string
23  fileHandle HANDLE ? ;// handle na izlazni file
24  startString BYTE "#EXTM3U", 0dh, 0ah ;// string za pomoc u obradi tj. formiranju izlaznog fajla
25  extString BYTE "#EXTINF:" ;// string za pomoc u obradi tj. formiranju izlaznog fajla
26
27  ;// Code segment
28  .code
29  main PROC
30
31  ;// let user input a filename
32  ;//
33  ;//
34  ;//
35  ;//
36  ;//
37  ;//
38  ;//
39  ;//
40  ;//
41  ;//
42  ;//
43  ;//
44  ;//
45  ;//
46  ;//
47  ;//
48  ;//
49  ;//
50  ;//
51  ;//
52  ;//
53  ;//
54  ;//
55  ;//
56  ;//
57  ;//
58  ;//
59  ;//
60  ;//
61  ;//
```

Output

Show output from: Debug

```
1  ;// file: AddTwo.asm
2  ;// brief: RE project main file.
3  ;// author: Aleksandra Skrbic(skrbica@sandra@yahoo.com)
4  ;//
5  ;// date: 30.05.2016.
6  ;// last revision: 15.06.2016.
7  ;// description: Dakle u okviru projektnog zadatka trebalo je dozvoliti da korisnik unese ime odgovarajuće playliste sa ekstenzijom .pls. Zatim se u okviru
8  ;// ovog koda provjera validnost uzasnog fajla. Potom se vrši obrada ulaznog fajla i sve se čuva u izlaznom fajlu sa ekstenzijom .m3u. Kada je
9  ;// obrada u pitanju, izlazni fajl ima odgovarajuću strukturu zahtevanu u tekstu samog projektnog zadatka.
10
11  ;// Includes
12  INCLUDE Irvine32.inc
13  INCLUDE macros.inc
14
15  ;// Declared data
16  BUFFER_SIZE = 50000
17
18  ;// Data section- initialized data
19  .data
20  buffer BYTE BUFFER_SIZE DUP(?) ;// velicina prvog bafera za učitavanje ULAZNOG fajla
21  buffer2 BYTE BUFFER_SIZE DUP(?) ;// velicina drugog bafera za smestanje obradjenog fajla i ispis u IZLAZNI fajl
22  srcFilename BYTE "initial name.pls", 0 ;// ulazno ime fajla se smesta u ovaj string
23  fileHandle HANDLE ? ;// handle na izlazni file
24  startString BYTE "#EXTM3U", 0dh, 0ah ;// string za pomoc u obradi tj. formiranju izlaznog fajla
25  extString BYTE "#EXTINF:" ;// string za pomoc u obradi tj. formiranju izlaznog fajla
26
27  ;// Code segment
28  .code
29  main PROC
30
31  ;// let user input a filename
32  ;//
33  ;//
34  ;//
35  ;//
36  ;//
37  ;//
38  ;//
39  ;//
40  ;//
41  ;//
42  ;//
43  ;//
44  ;//
45  ;//
46  ;//
47  ;//
48  ;//
49  ;//
50  ;//
51  ;//
52  ;//
53  ;//
54  ;//
55  ;//
56  ;//
57  ;//
58  ;//
59  ;//
60  ;//
61  ;//
```

Output

Show output from: Debug

File Edit View Project Build Debug Team Tools Test Analyze Window Help Full Screen Quick Launch (Ctrl+Q) Sign in

AddTwo.asm* X

Project (Global Scope)

```
58
59 check_buffer_size :
60 cmp eax, BUFFER_SIZE           ;// buffer large enough ?
61 jb buf_size_ok                 ;// yes
62 mwrite <"Error: Buffer too small for the file", 0dh, 0ah>
63 jmp quit                       ;// and quit
64
65 ;// Display the buffer size.
66 buf_size_ok:
67 mov buffer[ecx], 0             ;// insert null terminator
68 mwrite "File size: "
69 call WriteDec                 ;// display file size
70 call Crlf
71
72 ;// Display the buffer.
73 mwrite <"Buffer:", 0dh, 0ah, 0dh, 0ah>
74 mov ecx, OFFSET buffer        ;// display the buffer
75 call WriteString
76 call Crlf
77
78 close_file :
79 mov eax, fileHandle
80 call CloseFile
81
82 cld                           ;// CLEAR DIRECTION, brise flag smeru => elementima stringa se pristupa u rastucem poretku
83 mov esi, OFFSET buffer        ;// SOURCE INDEX REG. ukazuje na prvi element buffera (adresa prvog elementa u baferu) => ulaz
84 mov edi, OFFSET buffer2       ;// DESTINATION INDEX REG. ukazuje na prvi element buffera2 (adresa prvog elementa u baferu2) => izlaz
85 mov bl, 1                     ;// '1' => samo da znamo kada je prvi upis zbog pocetnog stringa za otput fajl (samo na pocetku fajla se javlja)
86
87 ;// idem redom u ulaznom fajlu .pls sve dok ne dodjem do "=", a onda ode u novi red => to je uloga petlji preskoci i preskoci_two
88 nasknri
```

109 %

Output

Show output from: Debug

Ready Publish

File Edit View Project Build Debug Team Tools Test Analyze Window Help Full Screen Quick Launch (Ctrl+Q) Sign in

AddTwo.asm* X

Project (Global Scope)

```
87 ;// idem redom u ulaznom fajlu .pls sve dok ne dodjem do "=", a onda ode u novi red => to je uloga petlji preskoci i preskoci_two
88 preskoci :
89 mov ecx, 100                  ;// broj iteracija po svakoj instrukciji za rad sa stringovima; ide pointer dok ne nadje "="
90 mov al, "="
91 cmp [esi], al                 ;// ako u baferu dodje do "=" to je onaj prvi red
92 je preskoci_two
93 inc esi
94 jmp preskoci
95
96 preskoci_two :
97 mov al, 0dh                  ;// sad idemo od "=" u redu koji nam ne treba sve do pocetka sledeceg reda
98 cmp [esi], al
99 je loop1pre
100 inc esi
101 jmp preskoci_two
102
103 ;// ovo je nas red u kome se nalazi podatak o adresi
104 loop1pre :
105 add esi, 2                    ;// sada samo pomerimo da pokazje na "F", to je red u kome je podatak koji nam treba!
106 loop1 :
107 mov al, 0dh                  ;// citaj do pocetka sledeceg reda
108 cmp [esi], al
109 je firstpush                  ;// dodjemo do kraja stringa reda File_i tj. do pocetka Title_i
110 inc esi
111 jmp loop1
112
113 ;// upisivanje odg. dela File_i podatka na STEK
114 firstpush :
115 mov eax, esi                  ;// ACC ce nam pomoci da se obratimo clanovima u nizu koji obradjujemo (trenutno 0dh), a u ESI pamtimmo gde smo stali
116 add esi, 2                    ;// ESI uvekamo da bi pokazivao na 'T' u Title_i redu to ce biti naredna obrada
117 firstpush subh
```

109 %

Output

Show output from: Debug

File Edit View Project Build Debug Team Tools Test Analyze Window Help Full Screen Quick Launch (Ctrl+Q) Sign in

AddTwo.asm* X

Project (Global Scope)

```
113 ;// upisivanje odg. dela File_i podatka na STEK
114 firstpush :
115 mov eax, esi ;// ACC ce nam pomoci da se obratimo clanovima u nizu koji obradjujemo (trentno 0dh), a u ESI pamtimo gde smo stali
116 add esi, 2 ;// ESI uvecamo da bi pokazivao na 'T' u Title_i redu to ce biti naredna obrada
117 firstpush_sub:
118 mov cl, '=' ;// vracaj se unazad od kraja (0dh) sve dok ne detektujes "=" i to ce biti adresa
119 cmp [eax], cl
120 je loop2 ;// sada je podatak o adresi na steku
121 mov dl, [eax]
122 mov [esp], dl ;// stek raste ka nizim mem. lokacijama, ukazuje na poslednju zauzetu
123 dec esp ;// stek pointer pomeramo po svakom upisu na odg. nacin
124 dec eax ;// idemo nazad u stringu koji obradjujemo sve dok ne ocita "="
125 jmp firstpush_sub
126
127 loop2 :
128 mov al, 0dh ;// idemo sada od 'T' (tu smo stali) tj. citamo sad do kraja Title_i reda za ime
129 cmp [esi], al
130 je secondpush
131 inc esi
132 jmp loop2
133
134 ;// upisivanje imena => Title_i na stek
135 secondpush :
136 mov eax, esi ;// ACC ce nam pomoci da se obratimo clanovima u nizu koji obradjujemo (trentno 0dh), a u ESI pamtimo gde smo stali
137 add esi, 2 ;// ESI uvecamo da bi pokazivao na 'L' u Length_i redu to ce biti naredna obrada
138 mov cl, '=' ;// idi unazad sve dok ne ocitas "=", tada ces imati celo ime
139 secondpush_sub :
140 cmp [eax], cl ;// vracaj se unazad od kraja (0dh) sve dok ne detektujes "=" i to ce biti ime
141 je loop3
142 mov dl, [eax]
143 mov [esp], dl
```

109 %

Output

Show output from: Debug

Ready Publish

File Edit View Project Build Debug Team Tools Test Analyze Window Help Full Screen Quick Launch (Ctrl+Q) Sign in

AddTwo.asm* X

Project (Global Scope)

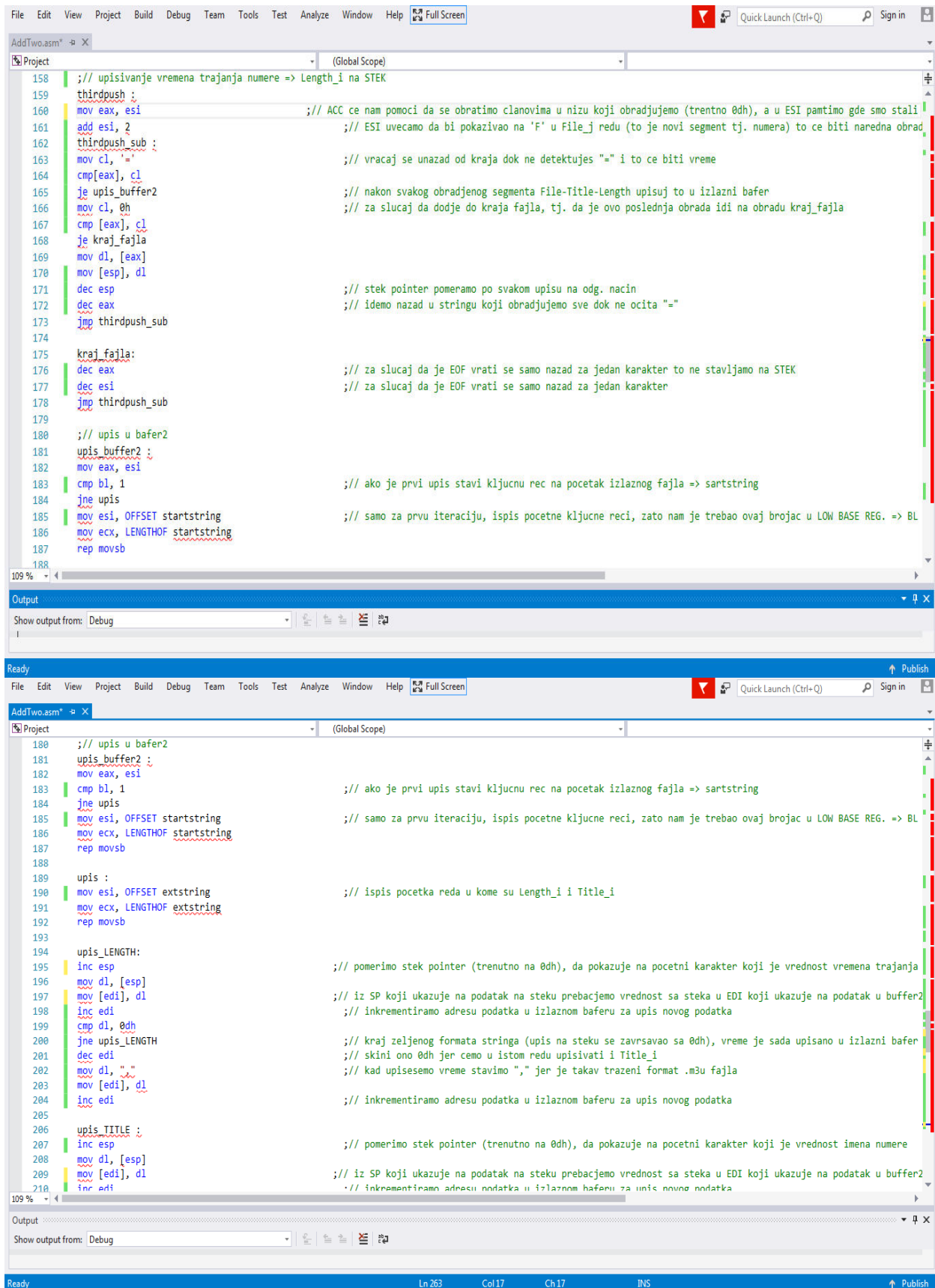
```
134 ;// upisivanje imena => Title_i na stek
135 secondpush :
136 mov eax, esi ;// ACC ce nam pomoci da se obratimo clanovima u nizu koji obradjujemo (trentno 0dh), a u ESI pamtimo gde smo stali
137 add esi, 2 ;// ESI uvecamo da bi pokazivao na 'L' u Length_i redu to ce biti naredna obrada
138 mov cl, '=' ;// idi unazad sve dok ne ocitas "=", tada ces imati celo ime
139 secondpush_sub :
140 cmp [eax], cl ;// vracaj se unazad od kraja (0dh) sve dok ne detektujes "=" i to ce biti ime
141 je loop3
142 mov dl, [eax]
143 mov [esp], dl
144 dec esp ;// stek pointer pomeramo po svakom upisu na odg. nacin
145 dec eax ;// idemo nazad u stringu koji obradjujemo sve dok ne ocita "="
146 jmp secondpush_sub
147
148 loop3 :
149 mov al, 0dh ;// idemo po sad kraja reda Length_i za vreme tj. MOZDA i do kraja fajla!
150 cmp [esi], al
151 je thirdpush
152 mov al, 0h ;// ovo je za slucaj kada ce to biti kraj fajla
153 cmp [esi], al
154 je thirdpush
155 inc esi
156 jmp loop3
157
158 ;// upisivanje vremena trajanja numere => Length_i na STEK
159 thirdpush :
160 mov eax, esi ;// ACC ce nam pomoci da se obratimo clanovima u nizu koji obradjujemo (trentno 0dh), a u ESI pamtimo gde smo stali
161 add esi, 2 ;// ESI uvecamo da bi pokazivao na 'F' u File_j redu (to je novi segment tj. numera) to ce biti naredna obrada
162 thirdpush_sub :
163 mov cl, '=' ;// vracaj se unazad od kraja dok ne detektujes "=" i to ce biti vreme
164 cmp [eax], cl
```

109 %

Output

Show output from: Debug

Ready Publish

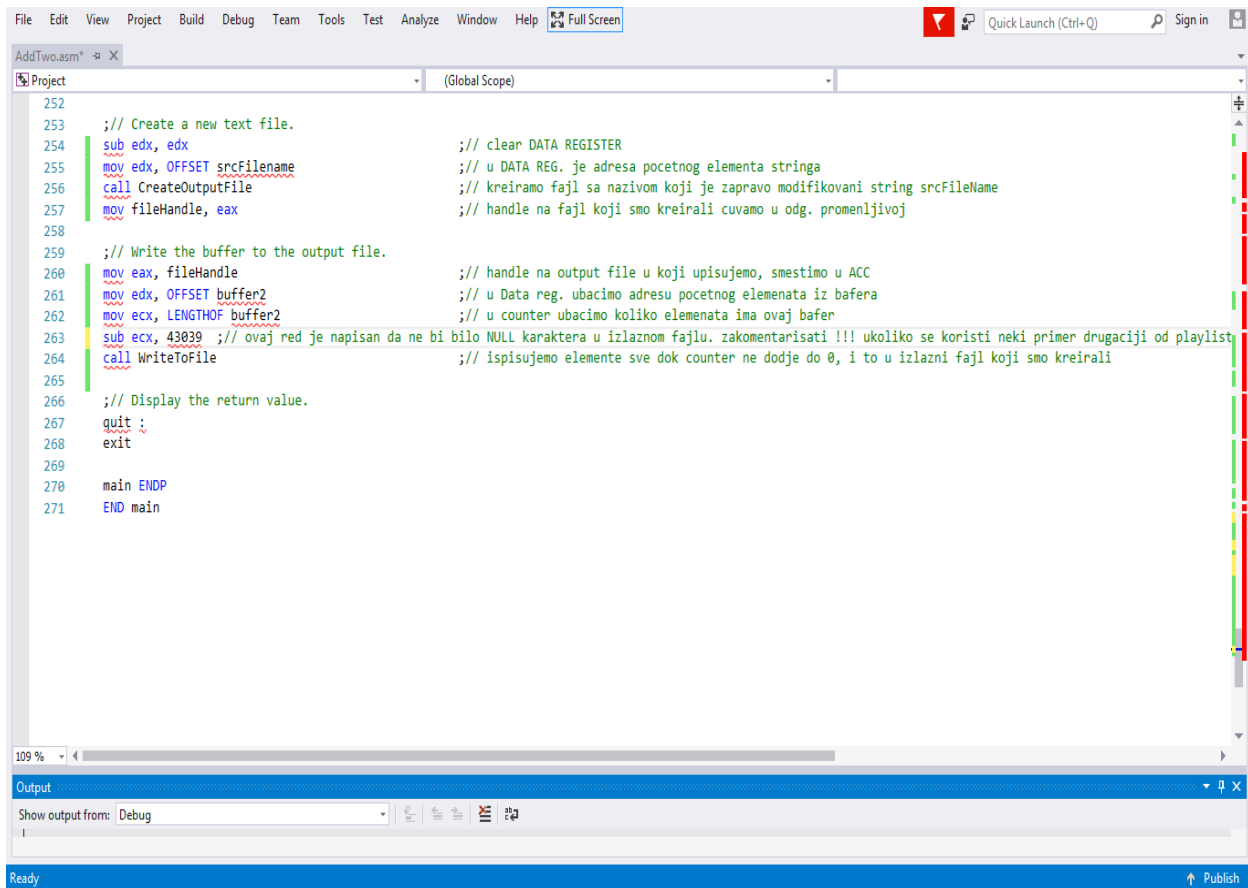


```
File Edit View Project Build Debug Team Tools Test Analyze Window Help Full Screen
Quick Launch (Ctrl+Q) Sign in

AddTwo.asm* X
Project (Global Scope)
205
206 upis_TITLE :
207 inc esp ;// pomerimo stek pointer (trenutno na 0dh), da pokazuje na pocetni karakter koji je vrednost imena numere
208 mov dl, [esp]
209 mov [edi], dl ;// iz SP koji ukazuje na podatak na steku prebacjemo vrednost sa steka u EDI koji ukazuje na podatak u buffer2
210 inc edi ;// inkrementiramo adresu podatka u izlaznom baferu za upis novog podatka
211 cmp dl, 0dh ;// kraj zeljenog formata stringa (upis na steku se završavao sa 0dh), ime numere je sada upisano u izlazni bafer
212 jne upis_TITLE
213
214 upis_FILE :
215 inc esp ;// pomerimo stek pointer (trenutno na 0dh), da pokazuje na pocetni karakter koji je vrednost adrese (putanje)
216 mov dl, [esp]
217 mov [edi], dl ;// iz SP koji ukazuje na podatak na steku prebacjemo vrednost sa steka u EDI koji ukazuje na podatak u buffer2
218 inc edi ;// inkrementiramo adresu podatka u izlaznom baferu za upis novog podatka
219 cmp dl, 0dh ;// kraj zeljenog formata stringa (upis na steku se završavao sa 0dh), putanja do numere je sada upisana u izlazni bafer
220 jne upis_FILE
221 inc bl ;// kada završimo prvi upis inkrementiramo brojac da ne bi stalno ispisivao START string u outpt file-u
222 mov esi, eax
223 mov cl, 0h
224 cmp [esi], cl ;// kada dođje do EOF izadji jer je upis završen, u protivnom se vrtimo u petlji sve dok ne obradimo sve pakete
225 jne loop1
226
227 ;// ovaj deo se odnosi na formiranje imena izlaznog fajla, na osnovu srcFilename koje korisnik unosi
228 cld ;// CLEAR DIRECTION, brise flag smera => elementima stringa se pristupa u rastucem poretку
229 mov ecx, 1 ;// broj iteracija po svakoj instrukciji za rad sa stringovima
230 sub edi, edi ;// DESTINATION INDEX REGISTER = 0, ima ulogu pokazivaca tj. tu je adresa pocetnog elementa stringa
231 mov edi, OFFSET srcFilename ;// prosledimo mu offset adrese na string srcFilename, sad ukazuje na njegov pocetak
232
233 ;// sad formiramo deo koji nam treba od ulaznog imena
234 izlaz_ime :
235 mov al, "" ;// u donji bajt akumulatora ubacimo string ""
109 %
```

```
Ready Publish
File Edit View Project Build Debug Team Tools Test Analyze Window Help Full Screen
Quick Launch (Ctrl+Q) Sign in

AddTwo.asm* X
Project (Global Scope)
232
233 ;// sad formiramo deo koji nam treba od ulaznog imena
234 izlaz_ime :
235 mov al, "." ;// u donji bajt akumulatora ubacimo string "."
236 cmp [edi], al ;// poredimo trenutni element u stringu sa "."
237 je extenzija ;// kada stigne do tacke u ulaznom imenu tu stajemo => to nam treba, sad ostaje samo da promenimo extenziju
238 inc edi ;// ukoliko jos nije stigao do tacke inkrementiraj pointer
239 jne izlaz_ime ;// vratiti se da proveris novi element
240
241 ;// ovaj deo sada služi da dodamo odgovarajucu extenziju za formiranje imena izlaznog fajla
242 extenzija :
243 inc edi ;// posto edi u tom trenutku ukazuje na "."
244 mov dl, "m" ;// u donji bajt DATA REGISTER-a stavimo string "m" od extenzije M3U
245 mov [edi], dl ;// posle tacke u do sada obradjenom stringu najpre dodamo "m"
246 inc edi ;// dodajemo dalje
247 mov dl, "3" ;// u donji bajt DATA REGISTER-a stavimo string "3" od extenzije M3U
248 mov [edi], dl ;// posle "m" u do sada obradjenom stringu dodamo "3"
249 inc edi ;// dodajemo dalje
250 mov dl, "u" ;// u donji bajt DATA REGISTER-a stavimo string "u" od extenzije M3U
251 mov [edi], dl ;// posle "3" u do sada obradjenom stringu dodamo za kraj "u"
252
253 ;// Create a new text file.
254 sub edx, edx ;// clear DATA REGISTER
255 mov edx, OFFSET srcFilename ;// u DATA REG. je adresa pocetnog elementa stringa
256 call CreateOutputFile ;// kreiramo fajl sa nazivom koji je zapravo modifikovani string srcFileName
257 mov fileHandle, eax ;// handle na fajl koji smo kreirali cuvamo u odg. promenljivoj
258
259 ;// Write the buffer to the output file.
260 mov eax, fileHandle ;// handle na output file u koji upisujemo, smestimo u ACC
261 mov edx, OFFSET buffer2 ;// u Data reg. ubacimo adresu pocetnog elementa iz bafera
262 mov ecx, LENGTHOF buffer2 ;// u counter ubacimo koliko elemenata ima ovaj bafer
109 %
```



```
252
253 // Create a new text file.
254 sub edx, edx // clear DATA REGISTER
255 mov edx, OFFSET srcFilename // u DATA REG. je adresa pocetnog elementa stringa
256 call CreateOutputFile // kreiramo fajl sa nazivom koji je zapravo modifikovani string srcFileName
257 mov fileHandle, eax // handle na fajl koji smo kreirali cuvamo u odg. promenljivoj
258
259 // Write the buffer to the output file.
260 mov eax, fileHandle // handle na output file u koji upisujemo, smestimo u ACC
261 mov edx, OFFSET buffer2 // u Data reg. ubacimo adresu pocetnog elemenata iz bafera
262 mov ecx, LENGTHOF buffer2 // u counter ubacimo koliko elemenata ima ovaj bafar
263 sub ecx, 43039 // ovaj red je napisan da ne bi bilo NULL karaktera u izlaznom fajlu. zakomentarisati !!! ukoliko se koristi neki primer drugaciji od playlist
264 call WriteToFile // ispisujemo elemente sve dok counter ne dodje do 0, i to u izlazni fajl koji smo kreirali
265
266 // Display the return value.
267 quit :
268 exit
269
270 main ENDP
271 END main
```

* Dodala sam slike za izvorni kod, iz razloga sto se po mom misljenju to cini preglednijim od jednostavnog kopiranja celog koda. Ovde se na slikama moze videti i kako je zakomentarisana svaka linija koda, tj. tu se veoma detaljno objasnjava sta svaki red koda predstavlja. Ukratko, sve je objasnjeno u prethodnom poglavlju izvestaja.

4. ZAKLJUCAK

Ono sto sam samo zeleda da napomenem je da je celokupni dizajn koda u skladu sa zahtevima u projektu zadatku, i da je funkcionalnost u potpunosti ispunjena. Medjutim, jedini problem koji eventualno moze da se uoci je sledeci: NIJE MOGUCE TACNO PODESITI VELICINU BAFERA (izlaznog). Ulazni i izlazni bafer ne mogu se staviti na istu velicinu, kada je smestanje podataka u pitanju, iz prostog razloga sto su formati ulaznog i izlaznog fajla drugaciji. Zato ce se javiti jedan problem koji sam primetila na samom ispisu u izlazni fajl. Za sva neiskoriscena mesta prilikom obrade i upisa podataka iz ulaznog fajla, pisace se u izlaznom fajlu simbol **NULL**! Zato u kodu postoji jedan red koji umanjuje velicinu bafera za odredjeni broj, da bi velicina bila potpuno odgovarajuca za ulaznu playlist-u datu kao primer za ovaj projektni zadatak. Tako da posto je ovaj program na njoj testiran, on u potpunosti i podesava velicinu izlaznog bafera specijalno za tu listu. Ukoliko zelite da isprobate na nekoj drugoj listi tipa **.pls**, taj red, kao sto i pise u samom kodu **zakomentarisite**. To je jedini problem koji sam uocila u realizaciji resenja. Probala sam da ga resim upisom **EOF** simbola (**0h**) na kraju, posle upisa poslednjeg karaktera iz bafera, ali to nije uspelo. Pokusala sam i brojac da kreiram, pa da velicinu buffera2 cuvam u nekoj promeljivoj, pa da kad se ona "prevrti" prekinem upis, ali ni to nije uspelo. Tako da, to bi to bila eventualna mala mana, jer bi izlazni fajl imao te **NULL** karaktere (simbole) u poslednjem redu, ali sve u svemu **trazena funkcionalnost je u potpunosti zadovoljena**.

5. LITERATURA

- [1] Materijal sa vezbi - Aleksandra Lekic
- [2] Assembly Language for x86 Processors, 7th edition by Kip Irvine, Florida International University (chapters I found on the Internet)

KRAJ :)