Univerza v Mariboru,
Fakulteta za elektrotehniko, računalništvo in informatiko

MIKROPROCESORSKI SISTEMI
2. letnik VS

POROČILO

Maribor, 2013

Univerza v Mariboru,
Fakulteta za elektrotehniko, računalništvo in informatiko

MIKROPROCESOSRKI SISTEMI
2. letnik VS

Priprave

POROČILO 1. VAJE

Avtor:
Gregor Nikolić
E1054204

Maribor, 2013

KAZALO

1. UVOD **4**
2. VAJA 1 ..... **5**
2.1 Besedilo naloge **5**
2.2 Besedni opis delovanja programa **5**
3. DIAGRAM POTEKA **6**
4. PROGRAM **7**

1. UVOD

Poročilo obsega programsko kodo ter diagram poteka za prvo vajo pri predmetu Mikroprocesorski sistemi, smer študija Elektrotehnika VS na Univerza v Mariboru, Fakulteta za Elektrotehniko Računalništvo in informatiko.

2. VAJA 1

2.1 Besedilo naloge

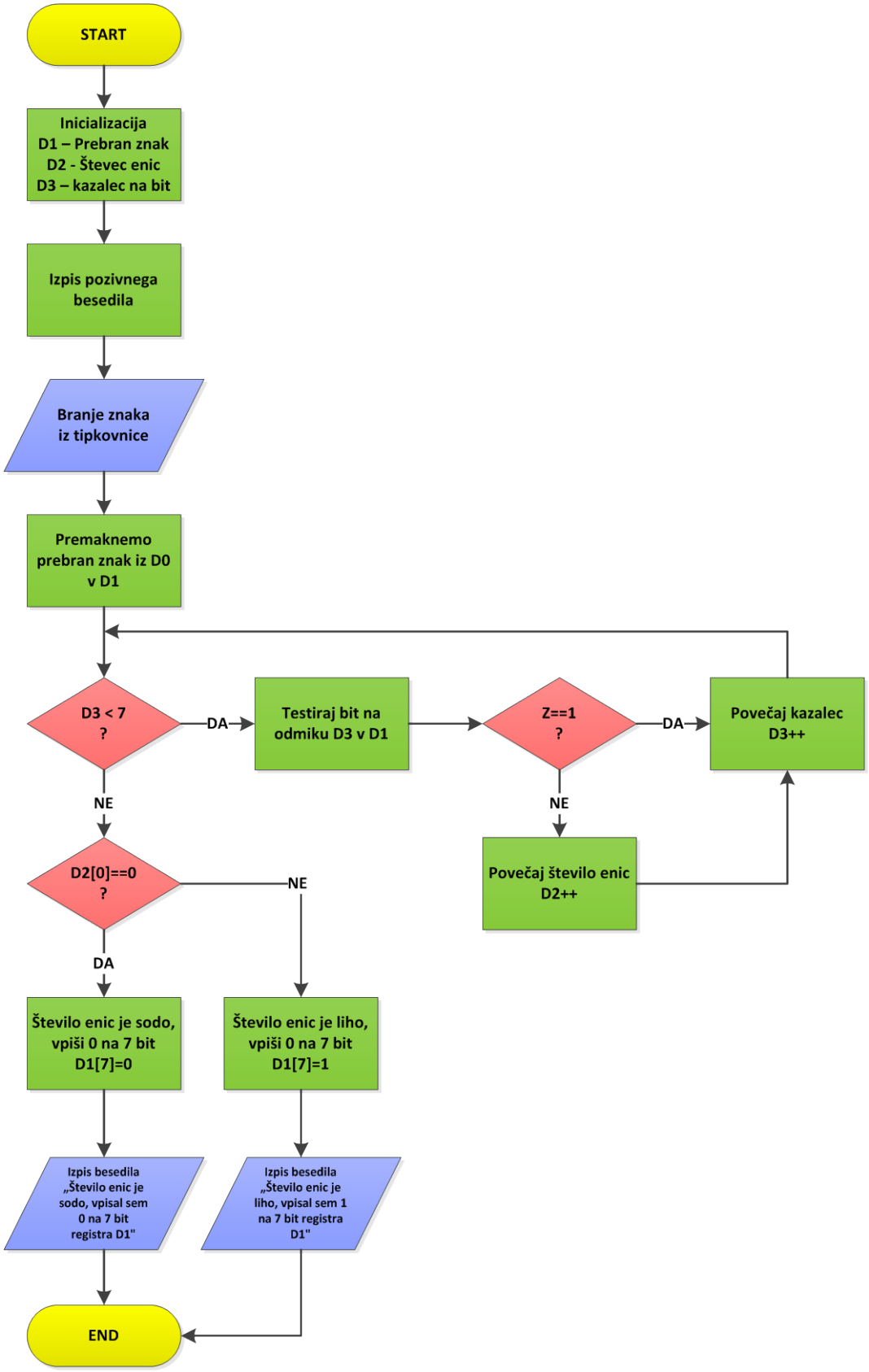
V tej vaji smo napisali program, ki je preveril število enic v ASCII znaku, katerega smo vpisali preko tastature. V primeru, da je znak vseboval sodo število enic smo na partitetni bit D1 vpisali »0« v primeru lihega števila enic pa »1«.

Primer ASCII znak »K« je njegova heksadecimalna vrednost 0x4B, binarno 0b01001011 v kateri se nahajajo štiri enice – sodo število enic.

2.2 Besedni opis delovanja programa

Po dogovoru so registri D1 za prebran znak, D2 za števec enic ter D3 uporabljen kot kazalec na bit.
V programu najprej preventivno počistimo vse tri registre (D1-D3), da si zagotovimo vrednost 0. Kličemo podprogram za naložitev naslova besedila ter izpišemo besedilo, ki se nahaja na tistem naslovu – pozivno besedilo po vnosu znaka. Nato kličemo podprogram za vnos znaka iz tastature, kateri čaka na vnesen znak. Takoj po vnosu prestavimo prebran znak iz registra D0 v D1.
Ker je register D3 kazalec na bit preverimo ali smo že preverili vse bite v registru D1 s pomočjo primerjave (CMP). Če ta vrednost še ni enaka moramo preverjati naprej, tako skočimo na mesto »TST_BIT«, kjer testiramo bit na odmiku D3 v D1. Če je vrednost enaka torej se je »Zero« zastavica postavila na 1, smo nasli ničlo, zato povečamo le kazalec na bit torej register D3 za ena in nadaljujemo z skokom nazaj na preverjanje kazalca (D3) če smo že preverili vse bite. V primeru, da najdemo enico pri preverjanju program ne veji le na povečanje registra D3 temveč tudi na povečanje registra D2 – število enic. Ko preverimo vse bite, program testira nulti bit v registru D2, saj je le ta v primeru sodega števila enic enak »0« v primeru lihega števila enic pa enak »1«. V primeru, da je testiran bit enak »0« bo program vejil na izpis besedila, da je število enic sodo ter postavil osmi bit na nič, v nasprotnem primeru pa bo program vejil na izpis besedila za liho število ter partitetni bit postavil na ena.

3. DIAGRAM POTEKA



Slika 1 Diagram poteka

4. PROGRAM

```
main:
        CLR.L   D1              // Preventivno pocistimo register D1. .L - 32 bitov - le tako zagotovimo, da je vse postavljeno na 0
        CLR.L   D2              // - || - D2
        CLR.L   D3              // - || - D3
        LEA     POZIV(PC),A1    // LEA - Load Effective Address - predhodna namestitev za izpis besedila
        MOVE.W  #$000D,-(A7)    // Skopiraj neposredno vrednost $D na naslov kamor kaze register A7 s predhodnim zmanjsanjem - 000D - program za izpis
        TRAP    #4              // Klic podprograma

        MOVE.W  #$0006,-(A7)    // Podprogram za branje iz tastature
        TRAP    #4              // Klic podprograma

        MOVE.B  D0,D1           // Premaknemo znak iz D0 v D1 (prebran znak je bil shranjen s podprogramom v D0)
NAZAJ:
        CMP     #7,D3           // Compare vr. 7 in vr. v D3, (ponor-izvor) (D3-7), ce je neg. st. se N postavi na 1 (N=1) ce je rezultat enak 0 se z postavi na 1 (Z=1)
        BLT     TST_BIT         // Branch if lower - Ce je vrednost manjsa pojdi testirati bit (iskanje enic)
        BTST    #0,D2           // Testiramo nulti bit v D2 - ce je vrednost 0 je stevilo sodo, ce je 1 je liho
        BEQ     SODA
        JMP     LIHA

SODA:
        BCLR    #7,D1           // Postavimo 7 bit v D1 na 0 - sodo stevilo
        LEA     IZPIS_SODO(PC),A1
        MOVE.W  #$000D,-(A7)
        TRAP    #4
        JMP     IZPIS_SODO      // Klic za izpis stavka v primeru sodega stevila
        RTS

LIHA:
        BSET    #7,D1           // Postavimo 7 bit v D1 na 1 - liho stevilo
        LEA     IZPIS_LIHO(PC),A1
        MOVE.W  #$000D,-(A7)
        TRAP    #4
        JMP     IZPIS_LIHO      // Klic za izpis stavka v primeru lihega stevila
        RTS

TST_BIT:
        BTST    D3,D1           // Testiraj bit v D1 na odmiku D3 (prvic je ta vrednost 0)
        BEQ     POVECAJ_D3      // Ce je Z=1 povecaj register D3 (nasli smo enico)
        JMP     POVECAJ_D2      // Ce ni povecaj D2 ( v podprogramu pa se tudi D3 )

POVECAJ_D3:
        ADD     #1,D3           // Povecamo stevec D3 za 1
        JMP     NAZAJ           // Skocimo nazaj preveriti ali je na naslednjem mestu enica

POVECAJ_D2:
        ADD     #1,D2           // Povecamo stevec enic D2
        JMP     POVECAJ_D3      // Skocimo na program za povecanje stevca D3


POZIV:  DC.B    32,"Vnesite poljuben znak. (^-^)"
IZPIS_SODO:     DC.B    32,"Stevilo je sodo. D7 = 0"
IZPIS_LIHO:     DC.b    32,"Stevilo je liho. D7 = 1"

        END
```

Slika 2 Program