ASSEMBLER <http://www.6502.org/tutorials/6502opcodes.html>



**Big vs Little endian**

• 08CE • 0000 1000 1100 1110

• Big endian zapis: • 08CE

• Little endian zapis: • CE08

***Big endian*** je način zapisa podatka u memoriji tako da je na nižoj adresi viši bajt memorijske riječi – normalan zapis

***Little endian*** je način zapisa podatka u memoriji tako da je na nižoj adresi niži bajt memorijske riječi naopačke, jednostavnije za računanje -kod malih ili starijih procesora, mikrokontroleri

**6502**

• Stari procesor • Nastao 1975 • Koristi 5V napajanje • 0-0.8V – logička 0

• 2-5V – logičko 1 • 8 bita • 16 bit adresna sabirnica (65536 lokacija)

**Registri • A, X, Y, P, S, PC**

• A jedini za aritmetiku

• X, Y mogu se koristiti kao indeksi za memoriju

**Zastavice**

P registar sadrži procesorske zastavice. Svaki bit u ovom registru ima jedinstvenu svrhu, osim bita označenog 1. 1 bit je neiskorišten i može se zanemariti. Svaki od preostalih bitova u ovom registru naziva se zastavicom i označava određeni uvjet koji se dogodio ili predstavlja postavku konfiguracije

• Bit 1 se ne koristi

• N – negativna vrijednost (ako je operacija postavila bit 7 rezultata)

• V – Overflow - ako operacija izađe iz raspona -128 do 127 – broj je pozitivan ali je krivo zapisan, staje u 8 bitova ali nije onaj broj koji smo zapisali jer je rezultat negativna vrijednost 127+15=-14

• B – Break – izvršena je operacija BRK. Služi razlikovanju softverskog i hardverskog prekida

• D – Decimal mode – označava da procesor koristi BCD zapis brojeva

• I – Interrupt disable – procesor neće obrađivati prekide (osim NMI)

• Z – Zero – Rezultat operacije je 0 - za grananje

• C – Carry – operacija je uzrokovala carry

**Stack pointer**

• Registar S • Odnosi se na memoriju $100 - $1FF • Govorimo o 256 bajtova

• Postavlja se na vrh • Svako spremanje na vrh smanjuje S • Ofset je fiksnih $100

• LIFO • Korisnik mora paziti

**Pravila pisanja**

• 24 znači 24 decimalno

• $24 znači da koristimo heksadecimalni zapis (36 decimalno)

• #24 znači izravna vrijednost 24

• #$24 znači izravno 36 u decimalnom zapisu

• LDA – Load A with value

• ADC – ADD using carry

• SBC – Subtract using Carry

• SEC – Set carry directly

• CLC – Clear carry directly

**Carry**

• Mora biti podešen prije operacije

• ADC će ga koristiti i dodati ukupnoj vrijednosti

• SBC će ga koristiti kako bi ga dodao prije oduzimanja

**Branching**

• Grananje je jedan od osnovnih načina kontrole rada

• JMP – skače na neku lokaciju

• Direktno se upisuje u JMP

• BCC i BCS – skaču ako je Carry Clear ili Set

• BNE BEQ ako je Z clear ili set

• BPL BMI Ako je N clear ili set

• BVC BVS ako je V clear ili set

• Koristi se relativno adresiranje -128 do 127

|  |  |
| --- | --- |
| Sta, x | 5 |
| Inx, dexTax, txaTay, tyaDey iny | 2 |
| lda | 2 |
| bne | 3 |
| pha | 3 |
| pla | 4 |
| Stx , ySty, x | 4 |
| loop | 256 |

|  |  |
| --- | --- |
| A picture containing text, stationary, writing implement  Description automatically generated**Prugice**LDX #$00loop:TXASTA $200, XINXBNE loop// sta $300, x za drugi dio ekrana itd. | **Crno bijele prugice** **– ispiši svakog drugog - neparni**LDX #$FFLDA #$01loop:STA $0200, XINXINXBNE loop |
|  |  |
| **Fill 512 bytes of memory from location $200****with value $1**LDA #$01LDX #$00loop:STA $200, XSTA $300, XINXBNE loop | **Drugi način za 512**LDA #$08LDX #$00start:STA $200, XSTA $300, XINXBNE start |
| **Ispisati pola ekrana s jednom pola s drugom bojom**LDX #$00 ; 2loop:LDA #$08 ;2STA $200,X ;5STA $300,X ;5LDA #$02 ;2STA $400,X ;5STA $500,X ;5INX ;2 BNE loop ;3  | Ili ----LDX #$00LDA #$03 loop:STA $200,X ;5STA $300,X ;5INX ;2BNE loop ;3 -- 15 x 256LDA #$02 ;2loop2:STA $400,X ;5STA $500,X ;5INX ;2BNE loop2 ;3 -- 15 x 256 |
| **Svaki treći**LDX #$00loop:LDA #$01STA $200,XINXTXACMP #$0BEQ endINXINX JMP loopend: | **Svaki peti**LDX #$00loop:LDA #$01STA $200,XINXTXACMP #$0BEQ endINXINX inxinxJMP loopend: |
| **Obojati 1 / 4 ekrana jednom bojom**LDA #$01LDX #$00 ; krećemo od 00 da pokrijemo rubni slučajloop:STA $200, XDEXBNE loopTrik – iscrtavamo prije nego smanjimo vrijednost – staviti komentar -ispravnije je staviti INX umjesto dex | **Drugi način**LDA #$08LDX #$00start:STA $200, XSTA $300, XINXBNE start |
| **Povukli smo crtu u memoriji**LDA #$01 LDX #$05start:STA $200, XDEXBNE start | **Spremanje na stack**LDA #$01PHALDA #$02PHALDA #$03PHALDA #$FFPLAPLAPLA |
| **; Add four bytes together using absolute** **addressing mode**LDA $0203CLCADC $0202ADC $0201ADC $0200 | Drugi način:LDX #$03CLCLDA $0200, XDEXADC $0200, XDEXADC $0200, XDEXADC $0200, X |
| **Subrutina**loop: LDA #$01STA $300, XJSR drugaINXBNE loopJMP enddruga:LDA #$5STA $400,XRTSend: | **Subrutina i stack -isto samo stack**LDA #$06loop:STA $300, XJSR drugaINXBNE loopJMP enddruga:PHALDA #$5STA $400,XPLARTSend: |

|  |  |
| --- | --- |
| **Pola ekrana jedna pola druga boja**LDX #$00 loop: LDA #$08 STA $200,X STA $300,X LDA #$02 STA $400,X  STA $500,X INX     BNE loop   | **Isto to samo sa subrutinom**LDA #$08loop:STA $200, XSTA $300, XJSR drugaINXBNE loopJMP enddruga:PHALDA #$02STA $400,XSTA $500,XPLARTSend: |
| **Subrutina četvrtine ekrana** LDA #$06LDX #$00loop:STA $200,XJSR subINXBNE loopJMP endsub:TAYLDA #$05STA $300,XLDA #$1TYARTSend: | **Subrutina četvrtine ekrana sa stackom**LDA #$06LDX #$00loop:STA $200,XJSR subINXBNE loopJMP endsub:PHALDA #$05STA $300,XLDA #$1PLARTSend: |

-----------------------------------------------

LDX #$03

CLC

LDA $0200, X

DEX

ADC $0200, X

DEX

ADC $0200, X

DEX

ADC $0200, X

Looping it

LDX #$03

LDA $0200, X

DEX

CLC

ADD\_LOOP:

ADC $0200, X

DEX

BPL ADD\_LOOP

Indirect indexed addressing

LDA #$00

STA $10

LDA #$02

STA $11

LDY #$03

LDA ($10), Y

DEY

CLC

ADD\_LOOP:

ADC ($10), Y

DEY

BPL ADD\_LOOP

------------------------------------------------------

LDA #$01 - spremi vrijednost 1 u registar a, ako je #broj to je decimalni broj, ako je #$ heksadecimalni broj

LDA $200 – odi u lokaciju 200 i uzmi vrijednost i spremi u akumulator

LDA $200, X – odi u lokaciju 200 i na njezinu vrijednost dodaj vrijednost x

Ako stisnemo disasemble on će ih pretvoriti u neke heksadecimalne kodove – opcode

---------------------