# Ishod 1

1. Imamo dvije hipotetske ALU, zadane tablicama stanja. Jedinica A ima 16 mogućih operacija. Jedinica B ima 8. Ako pretpostavimo da je logički dio obje jedinice identičan koja od njih će nužno imati više sklopova i osnovnih logičkih elemenata? (2)  
   Ne znam 😊  
   Pa valjda će onaj ALU sa 16 operacija imat vise sklopova jer ima vise operacija (ovo pitanje mi se čini prelagano) - M
2. Za ALU koju smo definirali slijedećom tablicomTable

   Description automatically generated

pojasnite zašto ima dva različita skupa stanja sa istim rezultatom (Prijenos F=X) (2)  
Kod prvog prijenosa gdje je Y=0 i C=0 X se prenosi dalje jer nije obavljena niti jedna aritmetička operacija na njemu, stoga ostaje isti, pa možemo reći da se prenosi dalje.  
Kod drugoga prijenosa gdje su kod Y sve jedinice i još nadodamo jedan iz Cin zbrajanjem dolazi do overflowa, stoga se naizgled čini kao da ništa nije promijenjeno. (Npr. X=0, Y=1, Cin=1 -> rezultat=10, odnosno zbog owerflova vidimo samo nulu).

1. Skicirajte poluzbrajalo i objasnite njegove prednosti i mane u usporedbi sa serijskim zbrajalom.  
   Sajd 73.  
   Prednosti- jednostavniji, lakši za izvesti, zauzima manje mjesta, nije potreban bistabil, može se izvesti pomoću osnovnih sklopova  
   Mane – ne može izvesti potpuno zbrajanje neke binarne riječi, ima samo 2 ulaza
2. Čemu služe zastavice u procesoru?  
   Odnose se na trenutno stanje u procesoru, kako bi se znao rezultat prethodno izvedenih operacija na procesoru.
3. Definirajte NILI sklop korištenjem tablice stanja.  
   Naći na internetu.
4. Što je vremenski brže - pristup cacheu ili pristup memoriji?  
   Pristup cacheu.
5. Gdje biste spremili potencijalni overflow kod neke operacije koju izvodi ALU i zašto?  
   U zastavicu V.
6. Što se događa ako kod operacija ne vodimo račun o bitu prijenosa?  
   Ako kod operacije ne vodimo računa o bitu prijenosa događa se owerflov ili underflow.

# Ishod 2

**NAPOMENA: U asembleru se komentar označava sa ;**

1. Izračunajte broj ciklusa za slijedeći primjer:

Text

Description automatically generated  
(?) 2+2+(4+4+2+2) +2+2+(4+4+2+2) =  
[2 + 2 + 256\*(5 + 5 + 2 + 2)] + [2 + 2 + 256\*(5 + 5 + 2 + 2)] = 7172

2. Izbacite nepotrebne instrukcije iz slijedećeg koda tako da on zadrži svoju osnovnu funkcionalnost:

loop:  
**;**TYA  
ADC $200, X  
ADC $210, X  
BCS set\_carry  
LDY #0  
JMP return  
set\_carry:  
LDY #1  
**;**JMP return  
return:  
STA $220, X  
DEX  
BPL loop

3. Napišite slijedeći kod tako da izbjegnete potrebu za dodanim instrukcijama nakon petlje a sačuvate funkcionalnost koda.

Izbrišu se zadnje tri linije originalnog koda

LDX #$03  
loop1:  
LDA $200, X  
ADC $210, X  
STA $220, X  
DEX  
BNE loop1

; Obrišemo linije ispod  
; LDA $200  
; ADC $210  
; STA $220

LDX #$03 ;; u x loadaj 3

loop1:

LDA $200, X ;; u a 200+3 ;; ...

ADC $210, X ;; zbroji sa 210+3 ;;...

STA $220, X ;;spremi na 220+3 ;;...

DEX ;; smanji x za 1

BNE loop1 ;; kad dode do 0 prestaje se vrtiti

LDA $200 ;;kad se petlja izvrsi loadaj u a adresu 200

ADC $210 ;; zbroji a s onim na adresi 210

STA $220 ;; spremi a na adresu 220

4. Izračunajte broj ciklusa za slijedeći primjer:

Text

Description automatically generated(4)  
2+(2+5+5+2+5+5+2+2)(?)  
2 + 256\*(2 + 5 + 5 + 2 + 5 + 5 + 2 + 2) = 7170

5. Prethodni zadatak riješite korištenjem subroutine, lokacije brojeva učitajte sa $+30 i $+40, te $+50 i $+60 lokacija.  
LDX #$00

loop:

LDA #$01

JSR ucitaj1

STA $300,X

LDA #$02

STA $400,X

STA $500,X

INX

BNE loop

ucitaj1:

STA 200,X

RTS

;...

# Ishod 3

Zašto su Spectre i Meltdown opasni?  
Jer zbog spekulativnog izvršavanja u cache memoriji procesora ostaju tragovi tajnih podataka koje maliciozni programi moguiskoristiti kako bi došli do zaključka gdje se nalaze tajni podaci kojima nebi smjeli pristupiti.

Kako radi RowHammer napad?  
Zbog latencije(koja se javlja zbog fizikalnih promjene) dolazi do „flipanja“ bitova u memoriji, i to onih bitova koji se nalaze u blizini bita kojem pristupamo(kojeg „flipamo“ brzo mu mojenjajući signal iz 1 u 0 i obratno).

Da li možemo spriječiti napade na hardverske jedinice?  
Napade na hardverske jedinice ne možemo spriječiti zbog ljudskih propusta koji su naprevljeni kroz povijest, a sada ih je teško ispraviti zbog velike rasprostranjenosti komponenti koje ne bi bilo isplativo mijenjati; ili zbog čiste fizike 😊

Kada zbrojimo dva broja, da li možemo pospremiti rezultat na proizvoljnu lokaciju i iskoristiti proizvoljni bajt za spremanje *carry* bita?  
Rezultat možemo spremiti na proizvoljnu lokaciju, a carry bit se sprema u zastavicu.

Da li se kod mikroprogramirane CU može napraviti zabrana pristupa registrima centralne procesne jedinice?  
Može se napraviti zabrana registara jer oni zapravo služe za spremanje(služe kao memorija), a glavni dio mikroprogramirane CU je upravo memorija.

Kakva je razlika između little endian i big endian zapisa?  
Big endian stavlja bitove po redu od najvećeg do najmanjeg po važnosti, dok little endian bitove raspoređuje ~~tako da je bit najveće važnosti u sredini, odnosno kao da broj počinje s drugom polovicom zapisa.~~ obrnuto od Big endiana, odnosno od najmanje do najveće važnog bita.  
Big endian zapis:  
08CE  
0000 1000 1100 1110

Little endian zapis:  
CE08  
1100 1110 0000 1000