# Computer architecure

Recap 6502

# Examples-.-.-----

# Fill 512 bytes of memory from location $200 with value $1

```
LDA #$01
LDX #$00
loop:
STA $200, X
STA $300, X
INX
BNE loop
```

- Moving the other way

```
LDA #$01
LDX #$00

start:
STA $200, X
STA $300, X
INX
BNE start
```

Fill half of the memory with one and other half with another colour

```
LDX #$00
loop:
LDA #$01
STA $200, X
STA $300, X
LDA #$02
STA $400, X
STA $500, X
INX
BNE loop
```

```
LDA #$05
LDX #$00

start:
STA $200, X
STA $300, X
INX
BNE start

LDA #$06
LDX #$00

start2:
STA $400, X
STA $500, X
INX
BNE start2
```

Store one 32 bit number in memory starting at $200, another at $210. Choose how to store the number yourselve. Add those numbers and store result in $220.

```
LDA #$ff
STA $200
STA $201
STA $202
STA $203
STA $204
STA $205
STA $206
STA $207
LDA #$ff
STA $210
STA $211
STA $212
STA $213
STA $214
STA $215
STA $216
STA $217
LDX #15
LDY #0
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
```

```
;prvi broj
LDA #$14
STA $0200
LDA #$56
STA $0201
LDA #$ae
STA $0202
LDA #$e8
STA $0203

;drugi broj
LDA #$54
STA $0210
LDA #$56
STA $0211
LDA #$ae
STA $0212
LDA #$e8
STA $0213

;zbroj
LDA $0203
ADC $0213
STA $0223

LDA $0202
ADC $0212
STA $0222

LDA $0201
ADC $0211
STA $0221

LDA $0200
ADC $0210
STA $0220
```

```
LDA #$01
STA $200
STA $201
STA $202
STA $203

LDA #$01
STA $210
STA $211
STA $212
STA $213
```

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

LDA $200
ADC $210
STA $220
```

Create two dots in the middle of the „video memory”. Animate them moving in oposite directions, first on X then on Y axis. Repeat until reset.

```
function:
LDX #$0
LDY #$10
LDA #$01
STA $410
loop:
LDA #$00
STA $410, X
STA $400, Y
INX
DEY
CPX #$10
BEQ quit
LDA #$01
STA $410, X
STA $400, Y
JMP loop
quit:
JMP function
```

```
function:
LDX #$00
LDY #$ff
LDA #$01
STA $410
LDA #$00
STA $410
CLC
loop:
LDA #$00
STA $411, X
STA $311, Y
TXA
ADC #$1f
TAX
TYA
SBC #$1f
TAY
LDA #$01
STA $411, X
STA $311, Y
TXA
CMP #$ff
SEC
BNE loop
LDA #$00
STA $411, X
LDX #$00
LDY #$ff
CLC
loop2:
LDA #$00
STA $511, X
STA $211, Y
TXA
ADC #$1f
TAX
TYA
SBC #$1f
TAY
LDA #$01
STA $511, X
STA $211, Y
TXA
CMP #$df
SEC
BNE loop2
LDA #$00
STA $511, X
STA $211, Y
JMP function
```

```
jsr MainLoop

DoAnimationOnX:
  jsr InitX
  jsr LoopX
  rts

InitX:
  LDA #$03 ; A = #$03
  LDX #$00 ; X = #$10 ; X = right dot
  LDY #$10 ; Y = #$0F ; Y = left dot
  rts

LoopX:
  ; print both dot
  LDA #$03
  STA $3F0,X
  STA $3FF,Y

  ; clear both dot
  LDA #$00
  STA $3F0,X
  STA $3FF,Y

  INX
  DEY

  BNE LoopX
  rts

; ==== Y animation ==== ;
define dotUpL $10
define dotUpH $11

define dotDownL $20
define dotDownH $21

DoAnimationOnY:
  JSR InitY
  JSR LoopY
  rts

InitY:
  ; initialise
  LDA #$F0
  STA dotUpL
  LDA #$03
  STA dotUpH

  LDA #$0F
  STA dotDownL
  LDA #$04
  STA dotDownH
  rts

print:
  CLC
  LDX #0
  LDA #$03
  STA (dotUpL,x) ; print up dot
  STA (dotDownL,x) ; print down dot

  LDA #$00
  STA (dotUpL,x) ; clear up dot
  STA (dotDownL,x) ; clear down dot
  rts

DecreaseDownDot:
  CLC
  LDA dotDownL
  adc #$20
  sta dotDownL
  BCS IncreaseDownDotH
  rts

IncreaseDownDotH:
  CLC
  LDA dotDownH
  adc #$01
  cmp #$06 ; Outside of screen
  beq MainLoop

  sta dotDownH
  rts

DecreaseUpDot:
  SEC
  LDA dotUpL
  SBC #$20
  sta dotUpL
  BCC DecreaseDownDotH
  rts

DecreaseDownDotH:
  SEC
  LDA dotUpH
  SBC #$01
  cmp #$01 ; Outside of screen leave the
game
  beq MainLoop
  sta dotUpH
  rts

LoopY:
  jsr print
  jsr DecreaseDownDot
  jsr DecreaseUpDot

  jsr LoopY

MainLoop:
  jsr DoAnimationOnX
  jsr DoAnimationOnY
  jsr MainLoop
```

```
LDA #$01
LDY #$00
loop1:
    LDA #$00
    STA $42E, X
    STA $430, Y
    LDA #$01
    INX
    DEY
    STA $42E, X
    STA $430, Y
    BNE loop1
```