Computer architecure

6502 assembly

Examples

- http://www.6502.org/source/

```
define TOPNT $10
CLRMEM:
  LDA #$00        ;Set up zero value
  TAY             ;Initialize index pointer
CLRM1:
  STA (TOPNT),Y   ;Clear memory location
  INY             ;Advance index pointer
  DEX             ;Decrement counter
  BNE CLRM1       ;Not zero, continue  checking
  RTS             ;Return
```

Stack

- Used as a temporary storage
- Directly or indirectly used

- PHA – PusH Accumulator
- PLA – Pull Accumulator

Stack behaviour

- No auto context switching
- JSR pushes address to the next instruction after the command
- DEMO (easy6502)

Context

- A, SP, P, X, Y

- Can be saved
- No other place than the stack

Subroutines

- PRO
- Way of creating functions
- Creates cleaner reuseable code

- CON
- No variables
- Only stack
- No context switch

- LDA #$00
- JSR routine
- JSR ….
- JSR ….
- JMP end

- routine:
- end:

- LDA #$01
- STA $0200
- LDA #$05
- STA $0201
- LDA #$08
- STA $0202

- PHA
- JSR add
- STA $230
- PLA

- LDA #$05
- STA $0200
- LDA #$06
- STA $0201
- LDA #$07
- STA $0202

- JSR add
- STA $232

- JMP end

- add:
- CLC
- LDA $200
- ADC $201
- ADC $202
- RTS

- end:

Interrupts

- Same idea as subroutine
- Triggered externally
- Triggered internally

Interacting with the rest of the world

- No waiting

Tasks:

- Go through chapters in easy6502 from „The stack” until „creating a game”

- Create a simple program with two subroutines: add and subtract. Add will add values from X and Y and subtract will subtract X-Y. Use carry (borrow)

- Create a simple program that will reverse a string of bytes stored from $20 to $2F into locations $30 to $3F (values from $20 should end up in $3F and so on)

Thank you for your attention!