# 6502 Assembler

**Goals:** Run first 6502 assembler program in c64 simulator. Edit code. Assemble Code. Read Opcode table. scan c64-kernal. Code 6502 assembler with different addressing modes

**Knowledge assembled:** Addressing modes, immediate adr. mode, absolute adr. mode, absolut with index adr. mode, implicit adr. mode

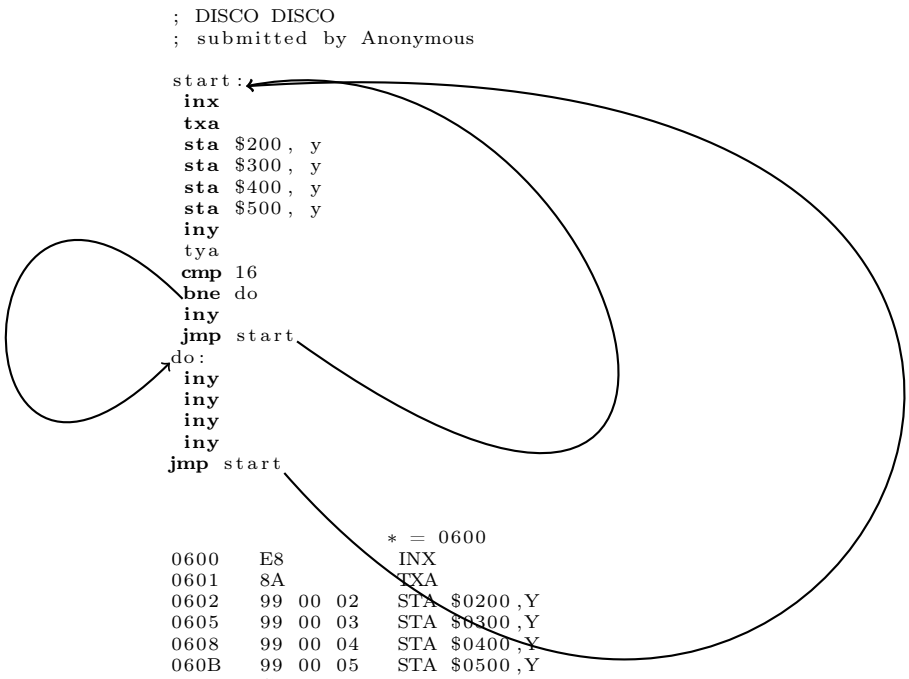
**Exercises Intro:** Work through `simple-hello-world`, `hello-world`, `count10` in this order.

1. Read, assemble, run program
2. Code assembler-code into 6502 binary code (by hand)
3. compare your code with the generated code

**Exercise First Steps:** Modify programs to use other registers, less register, less code etc.

**Exercise First Problem:** Write a program which counts the character of a string `x` and prints "`<x> has <number of x> character`".

**Exercise Addressing Modes:** Try rewriting sum.asm by replacing the instructions using absolute with index addressing modes with other addressing modes



```
; DISCO DISCO
; submitted by Anonymous

start:
 inx
 txa
 sta $200, y
 sta $300, y
 sta $400, y
 sta $500, y
 iny
 tya
 cmp 16
 bne do
 iny
 jmp start
do:
 iny
 iny
 iny
 iny
jmp start
```

```
                    * = 0600
0600    E8          INX
0601    8A          TXA
0602    99 00 02    STA $0200,Y
0605    99 00 03    STA $0300,Y
0608    99 00 04    STA $0400,Y
060B    99 00 05    STA $0500,Y
060E    C8          INY
060F    98          TYA
0610    C5 10       CMP $10
0612    D0 04       BNE $0618
0614    C8          INY
0615    4C 00 06    JMP $0600
0618    C8          INY
0619    C8          INY
061A    C8          INY
061B    C8          INY
061C    4C 00 06    JMP $0600
061F                .END
```

Figure 1: disassembler code Tabelle ab Adresse $0x0600$

| | | | |
|---|---|---|---|
| Pin 1 | GND | – | GrouND: Masse (0V) |
| Pin 2-9 | PA0-PA7 | I/O | Parallel port a signals. Bidirectional parallel port. |
| Pin 10-17 | PB0-PB7 | I/O | Parallel port b signals. Bidirectional parallel port. |
| Pin 18 | PC | O | Handshake output. A low pulse is generated after a read or write on port b. |
| Pin 19 | TOD | I | Time od day clock input. Programmable 50hz or 60hz. |
| Pin 20 | Vcc | – | Supply voltage: +5V DC |
| Pin 21 | $\overline{\text{IRQ}}$ | O | Interrupt output to microprocessor input IRQ. |
| Pin 22 | R/$\overline{\text{W}}$ | I | READ/WRITE input |
| Pin 23 | $\overline{\text{CS}}$ | I | Chip select input. A low pulse will activate CIA. |
| Pin 24 | $\overline{\text{Flag}}$ | I | Negative edge sensitive interrupt input. Can be used as a handshake line for either parallel port. |
| Pin 25 | ø2 | I | clock input connected to processor |
| Pin 26-33 | DB0-DB7 | I/O | Bidirectional data bus. Connects to processor data bus. |
| Pin 34 | $\overline{\text{RES}}$ | I | Low active reset input. Initializes CIA. |
| Pin 35-38 | RS0-RS3 | I | Register select inputs. Used to select all internal registers for communications with the parallel ports, time of day clock and serial port (SP). |
| Pin 39 | SP | I/O | Serial Port bidirectional connection. An internal shift register converts microprocessor parallel data into serial data, and vice versa. |
| Pin 40 | CNT | I | Count input. Internal timers can count pulses applied to this input. Can be used for frequency dependant operations. |

Figure 2: Pin layout des CIA (Complex Interface Adapter) Chip für C64

| | | CIA 1 Port B ($DC01) | Joy 2 | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | PB7 | PB6 | PB5 | PB4 | PB3 | PB2 | PB1 | P |
| CIA1 Port A ($DC00) | PA7 | STOP | Q | C= | SPACE | 2 | CTRL | <- | |
| PA6 | / | ^ | = | RSHIFT | HOME | ; | * | Â£ | |
| PA5 | , | @ | : | . | - | L | P | + | |
| PA4 | N | O | K | M | 0 | J | I | 9 | F |
| PA3 | V | U | H | B | 8 | G | Y | 7 | Ri |
| PA2 | X | T | F | C | 6 | D | R | 5 | L |
| PA1 | LSHIFT | E | S | Z | 4 | A | W | 3 | Do |
| PA0 | CRSR DN | F5 | F3 | F1 | F7 | CRSR RT | RETURN | DELETE | U |
| Joy 1 | | | | | Fire | Right | Left | Down | U |