```
;-------------------------------------------------
; Testbench aller zur Verfügung stehenden Befehle
;-------------------------------------------------

.org 0x000
.start

LDIH  r0, 0x00   ; => 00000000
LDIL  r0, 0x00   ; => 0000000000000000
LDIH  r1, 0x00   ; => 00000000
LDIL  r1, 0x00   ; => 0000000000000000
LDIH  r3, 0x00   ; => 00000000
LDIL  r3, 0x00   ; => 0000000000000000

XOR   r0, r0, r0 ; Nur '0' in r0.  ///
LDIL  r0, 8      ; 1. Wert in r0
XOR   r1, r1, r1
LDIL  r1, 4

; ----------
; Arithmetic
; ----------

XOR   r3, r3, r3
ADD   r3, r0, r1 ; Addition: r3 = r0 + r1     => 12

XOR   r3, r3, r3
SUB   r3, r0, r1 ; Subtraktion: r3 = r0 - r1  => 4

XOR   r3, r3, r3
SAL   r3, r0     ; Links-Shift:               => 16

XOR   r3, r3, r3
SAR   r3, r0     ; Rechts-Shift:              => 4


; -----
; Logic
; -----

XOR   r3, r3, r3
LDIH  r3, 0xFF   ; => 1111111100000000
LDIL  r3, 0xFF   ; => 1111111111111111
AND   r3, r0, r1 ; => 0

XOR   r3, r3, r3
OR    r3, r0, r1 ; => 12

XOR   r3, r3, r3
XOR   r3, r0, r1 ; => 12

XOR   r3, r3, r3
NOT   r3, r0     ; => 1111111111110111
```

```
; -------
; Control
; -------

;LD
LDIL r1, 0x00
LDIH r1, 0x01
LD  r3, [r1]     ; => 0x0100


;ST
XOR   r3, r3, r3
LDIL  r3, result & 255
LDIH  r3, result >> 8
XOR   r1, r1, r1
XOR   r1, r1, r1
LDIH  r1, 0xFF
LDIL  r1, 0xFF   ; => 1111111111111111
ST  [r3], r1


;JMP
XOR   r3, r3, r3
XOR   r1, r1, r1
LDIL r1, loopjmp
LDIH r1, loopjmp>>8
JMP r1
LDIL r3, 0xFF
loopjmp: ; kein neuer Wert in r3!


;JNZ
XOR   r3, r3, r3
XOR   r1, r1, r1
XOR   r2, r2, r2
LDIL r2, 0x01
LDIH r2, 0x00
LDIL r1, loopjnz
LDIH r1, loopjnz>>8
JNZ r2, r1
LDIL r3, 0xFF
loopjnz: ; kein neuer Wert in r3!

;JZ
XOR   r3, r3, r3
XOR   r1, r1, r1
XOR   r2, r2, r1
LDIL r2, 0x00
LDIH r2, 0x00
LDIL r1, loopjz
LDIH r1, loopjz>>8
JZ r2, r1
LDIL r3, 0xFF
```

```
loopjz: ; kein neuer Wert in r3!


XOR   r1, r1, r1


HALT

.org 0x0100
result: .res 8
        .data 42

.end
```