24 bit multiplication (signed or unsigned)

```
    ; Signed or unsigned 24-bit multiply (24-bit product)
    ; by Neils
    ; DASM format

    PROCESSOR 6502
factor1 EQU $61
factor2 EQU $64
product EQU $67

    MAC twoscomplement
    lda {1}+2
    eor #$ff
    sta {1}+2
    lda {1}+1
    eor #$ff
    sta {1}+1
    lda {1}
    eor #$ff
    clc
    adc #$01
    sta {1}
    ENDM

    ; An example program that multiplies -981 by 1340
    ; Benchmark: 812 cycles (with this input)

    SEG PROGRAM
    ORG $C000
    ; factor1 := -981
    lda #$2b
    ldx #$fc
    ldy #$ff
    sta factor1
    stx factor1+1
    sty factor1+2
    ; factor2 := 1340
    lda #$3c
    ldx #$05
    ldy #$00
    sta factor2
    stx factor2+1
    sty factor2+2
    ; result := factor1 * factor2
    jsr signed_mul24
    rts

    ; Signed 24-bit multiply routine
```

```
    ; Clobbers a, x, factor1, factor2

signed_mul24    SUBROUTINE
    ldx #$00            ; .x will hold the sign of product
    lda factor1+2
    bpl .skip            ; if factor1 is negative
    twoscomplement factor1        ; then factor1 := -factor1
    inx                ; and switch sign
.skip
    lda factor2+2
    bpl .skip2            ; if factor2 is negative
    twoscomplement factor2        ; then factor2 := -factor2
    inx                ; and switch sign
.skip2
    jsr unsigned_mul24        ; do unsigned multiplication
    txa
    and #$01            ; if .x is odd
    beq .q
    twoscomplement product        ; then product := -product
.q  rts

    ; Unsigned 24-bit multiply routine
    ; Clobbers a, factor1, factor2

unsigned_mul24  SUBROUTINE
    lda #$00            ; set product to zero
    sta product
    sta product+1
    sta product+2

.loop
    lda factor2            ; while factor2 != 0
    bne .nz
    lda factor2+1
    bne .nz
    lda factor2+2
    bne .nz
    rts
.nz
    lda factor2            ; if factor2 is odd
    and #$01
    beq .skip
    lda factor1            ; product += factor1
    clc
    adc product
    sta product
    lda factor1+1
    adc product+1
    sta product+1
    lda factor1+2
```

```
    adc product+2
    sta product+2           ; end if

.skip
    asl factor1           ; << factor1
    rol factor1+1
    rol factor1+2
    lsr factor2+2           ; >> factor2
    ror factor2+1
    ror factor2

    jmp .loop           ; end while
```