

It's just an enlarged version of the 16-bit division. There's not enough unused page-zero locations on a C64 to have all the variables in page-zero. The remainder variable is in page-zero because is the most used.

```

; Executes an unsigned integer division of a 24-bit dividend by a 24-bit
divisor
; the result goes to dividend and remainder variables
;
; Verz!!! 18-mar-2017

div24   lda #0                ;preset remainder to 0
        sta remainder
        sta remainder+1
        sta remainder+2
        ldx #24               ;repeat for each bit: ...

divloop asl dividend          ;dividend lb & hb*2, msb -> Carry
        rol dividend+1
        rol dividend+2
        rol remainder         ;remainder lb & hb * 2 + msb from carry
        rol remainder+1
        rol remainder+2
        lda remainder
        sec
        sbc divisor          ;subtract divisor to see if it fits in
        tay                  ;lb result -> Y, for we may need it later
        lda remainder+1
        sbc divisor+1
        sta pztemp
        lda remainder+2
        sbc divisor+2
        bcc skip             ;if carry=0 then divisor didn't fit in yet

        sta remainder+2      ;else save subtraction result as new remainder,
        lda pztemp
        sta remainder+1
        sty remainder
        inc dividend         ;and INCrement result cause divisor fit in 1 times

skip    dex
        bne divloop
        rts

dividend .ds 3
divisor  .ds 3
remainder .equ $fb          ; remainder is in zero-page to gain some cycle/byte
($fb-$fd)
pztemp   .equ $fe

```

From:
<https://codebase64.org/> - **Codebase 64 wiki**

Permanent link:
https://codebase64.org/doku.php?id=base:24bit_division_24-bit_result

Last update: **2017-03-18 10:41**

