

With all the network cards for the C64 these days, a user-friendly IP address input/parsing routine might be handy. Here's one I wrote for Artillery Duel - could be optimized, but hopefully easy to follow.

Use in conjunction with the [robust_string_input](#) code. DASM.

```
; =====
; IP Address Input routine - Schema/AIC
; =====

;Converts e.g.
;192.168.1.64
;to
;ABC ABC ABC ABC
;192.168.001.064
;and then to 4 bytes at gotip

include "robust_string_input.asm"

IPADDRESS_FILTER
    dc.b "1234567890.",0

getip
    lda #$00
    sta $cc          ; Force cursor to flash
    lda #>IPADDRESS_FILTER
    ldx #<IPADDRESS_FILTER
    ldy #15
    jsr FILTERED_INPUT
    jsr CONVERTIP
    lda #$01
    sta $cc          ; Force cursor to not flash
    lda #$20
    jsr $ffd2        ;overwrite cursor detritus
    rts

gotip
    dc.b $00,$00,$00,$00

;=====

DOTS
    dc.b #$00

STRINDEX
    dc.b #$00

IPINDEX
    dc.b $00

CONVERTIP
```

```
;First, check that there are three periods '.' in the string.
ldx #$00
sta DOTS
sta IPINDEX
sta STRINDEX
COUNTLOOP
lda GOTINPUT,x
beq DONECOUNT ; end of string (0)
cmp #'.'
bne COUNTNEXT
inc DOTS
COUNTNEXT
inx
jmp COUNTLOOP

DONECOUNT
lda DOTS
cmp #$03
beq DOTSOK
jmp INVALIDIP

; OK, now pad the values if needed for the conversion, one octet at a time
DOTSOK

;Init A,B,C with 0
NEXTBYTE
lda #$0
sta A
sta B
sta C

BYTELOOP
ldx STRINDEX
lda GOTINPUT,x
beq ENDOFBYTE ;end of string (0)
cmp #'.'
beq ENDOFBYTE
;Shift ABC over
ldy B
sty A ;old A is lost
ldy C
sty B
sbc #$30 ;Convert from PETSCII to equivalent value
sta C ;insert new digit

NEXTDIGIT
inc STRINDEX
jmp BYTELOOP

ENDOFBYTE
```

```

jsr dec2hex ;Returns byte in accumulator, carry set if >255
bcs INVALIDIP
;Save the returned byte
ldy IPINDEX
sta gotip,y
inc IPINDEX
lda IPINDEX
cmp #$04 ;Done
beq IP_DONE
inc STRINDEX ;Skip over the '.'
jmp NEXTBYTE

```

```

IP_DONE
rts

```

```

INVALIDIP
; ERROR MESSAGE HERE!
inc $d020
jmp INVALIDIP
rts

```

```

;=====
; Convert three-digit decimal (ABC) into a byte
;=====

```

```

dec2hex
lda A
jsr MULT10
jsr MULT10 ;x100
sta TMP0
lda B
jsr MULT10 ;x10
sta TMP1
lda C
sta TMP2 ;x1
clc
lda #$00
adc TMP0
adc TMP1
adc TMP2
rts ;Carry will be set if result was > 255

```

```

; =====

```

```

TMP0
dc.b #$00

```

```

TMP1
dc.b #$00

```

```
TMP2
  dc.b #$00

A          ;First digit
  dc.b #$00

B          ;Second digit
  dc.b #$00

C          ;Third digit
  dc.b #$00

; =====
; Multiply by 10 - from http://www.6502.org/source/integers/fastx10.htm

MULT10  ASL          ;multiply by 2
        STA TEMP10  ;temp store in TEMP
        ASL          ;again multiply by 2 (*4)
        ASL          ;again multiply by 2 (*8)
        CLC
        ADC TEMP10  ;as result, A = x*8 + x*2
        RTS

TEMP10   .byte 0
```

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

Permanent link:
https://codebase64.org/doku.php?id=base:ip_address_input

Last update: **2015-04-17 04:32**

