Alternative ACTION! Runtime Source#

first published in german magazine "Computer Kontakt (CK)", written by Peter Finzel (see ACTION noch schneller)

additional Runtime parts by Carsten Strotmann

```
MODULE ; for user

;**********************************
;  ACTION!-Runtime Package
;
;        Filename: RUNTIME.ACT
;
;PETER FINZEL                   1986
;CARSTEN STROTMANN              1990
;**********************************

;*****  Multiplication *****
PROC RUDIV2=*()
[ $85 $86 $86 $87 $38 $A9 $00 $E5 $86
$A8 $A9 $00 $E5 $87 $AA $98 $60 ]

PROC Rumlt0=*()
[ $F0 $1B $CA $86 $C1 $AA $F0 $15
$86 $C0 $A9 $00 $A2 $08 $0A $06 $C0
$90 $02 $65 $C1 $CA $D0 $F6 $18
$65 $87 $85 $87 $A5 $86 $A6 $87 $60 ]

PROC Rumlt1=*()
[ $86 $C2 $E0 $00 $10 $03 $20 RUDIV2
$85 $82 $86 $83 $A5 $85 $10 $0E
$AA $45 $C2 $85 $C2 $A5 $84 $20
RUDIV2 $85 $84 $86 $85 $A9 $00 $85
$87 $60 ]

PROC Mult=*()
[ $20 RUMLT1 $A6 $82 $F0 $1B $86 $C0
$A6 $84 $F0 $15 $CA $86 $C1 $A2
$08 $0A $26 $87 $06 $C0 $90 $06
$65 $C1 $90 $02 $E6 $87 $CA $D0 $F0
$85 $86 $A5 $82 $A6 $85 $20 RUMLT0
$A5 $83 $A6 $84 $20 RUMLT0 ]

;***** Division *****
PROC RUDIV=*()
[ $A4 $C2 $10 $03 $4C RUDIV2 $60 ]

PROC Div=*()
[ $20 RUMLT1 $A5 $85 $F0 $27 $A2 $08
$26 $82 $26 $83 $26 $87 $38 $A5
$83 $E5 $84 $A8 $A5 $87 $E5 $85
$90 $04 $85 $87 $84 $83 $CA $D0 $E7
$A5 $82 $2A $A2 $00 $A4 $83 $84 $86
$18 $90 $1D $A2 $10 $26 $82 $26
$83 $2A $B0 $04 $C5 $84 $90 $03
$E5 $84 $38 $CA $D0 $EF $26 $82 $26
$83 $85 $86 $A5 $82 $A6 $83 $A4 $C2
$10 $10 $85 $84 $86 $85 $38 $A9
```

```
$00 $E5 $84 $A8 $A9 $00 $E5 $85
$AA $98 $60 ]

;***** Modulo *****
PROC Modulo=*()
[ $20 DIV $A5 $86 $A6 $87 $60 ]

;***** Left- and Rightshift *****
PROC Rrsh=*()
[ $A4 $84 $F0 $0A $86 $85 $46 $85
$6A $88 $D0 $FA $A6 $85 $60 ]

PROC Rlsh=*()
[ $A4 $84 $F0 $0A $86 $85 $0A $26
$85 $88 $D0 $FA $A6 $85 $60 ]

;***** Parameter-Routine *****
PROC Par=*()
[ $85 $A0 $86 $A1 $84 $A2 $18 $68 $85
$84 $69 $03 $A8 $68 $85 $85 $69
$00 $48 $98 $48 $A0 $01 $B1 $84 $85
$82 $C8 $B1 $84 $85 $83 $C8 $B1
$84 $A8 $B9 $A0 $00 $91 $82 $88
$10 $F8 $A5 $11 $D0 $05 $E6 $11
$6C $0A $00 $60 ]

SET $4E4=Rlsh
SET $4E6=Rrsh
SET $4E8=Mult
SET $4EA=Div
SET $4EC=Modulo
SET $4EE=Par

;Global-Variable for Error
;================================

MODULE
BYTE ARRAY EOF(7)=$5C0
BYTE color=$2FB,device=[0],ioerr,trace,list

;Help-Funktions for IO
;==============================

PROC CIOV=$E456 (BYTE areg,xreg)

PROC CIOL=*(BYTE chn,cmd,
CARD Buffer,Length)
[ $85 $A0 $86 $A1 $0A $0A $0A $0A $AA
$A5 $A1 $9D $42 $03 $98 $9D $44
$03 $A5 $A3 $9D $45 $03 $A5 $A4
$9D $48 $03 $A5 $A5 $9D $49 $03 ]

PROC CIO=*()
[ $20 $56 $E4 $A6 $A0 $85 $A0 $C0 $88
$D0 $09 $A9 $01 $9D $C0 $05 $8D
ioerr $60 $A9 $00 $9D $C0 $05
$8C $FF $06 $60 ]

BYTE FUNC CIOS=*(BYTE chn,cmd,data)
```

```
[ $85 $A0 $86 $A1 $0A $0A $0A $0A $AA
$A5 $A1 $9D $42 $03 $A9 $00 $9D
$48 $03 $9D $49 $03 $98 $4C CIO ]

PROC SETAUX=*(BYTE chn,aux1,aux2)
[ $86 $A1 $84 $A2 $0A $0A $0A $0A $AA
$A5 $A1 $9D $4A $03 $A5 $A2 $9D
$4B $03 $60 ]

PROC AFP=$D800 ()
PROC FASC=$D8E6  ()
PROC IFP=$D9AA ()
PROC FPI=$D9D2 ()

;** OPEN- and CLOSE-Command **

PROC Open(BYTE chn,
BYTE POINTER fname,
BYTE aux1,aux2)
BYTE ARRAY fstr(17)
BYTE POINTER bptr
BYTE z

bptr=fname+1
FOR z=0 TO fname^-1 DO
fstr(z)=bptr^
bptr==+1
OD
fstr(z)=$9B
SETAUX(chn,aux1,aux2)
CIOL(chn,3,fstr,0)
RETURN

PROC Close(BYTE chn)
CIOS(chn,$0C,0)
RETURN

;GET- and PUT-Command
;====================

BYTE FUNC GetD(BYTE chn)
RETURN (CIOS(chn,7,0))

PROC Put(Byte chr)
CIOS(device,$0B,chr)
RETURN

PROC PutE()
CIOS(device,$0B,$9B)
RETURN

PROC PutD(BYTE chn,chr)
CIOS(chn,$0B,chr)
RETURN

PROC PutDE(BYTE chn)
CIOS(chn,$0B,$9B)
RETURN
```

```
;PRINT-Command for Strings
;=========================

PROC PrintD(BYTE chn,
BYTE POINTER buffer)
CIOL(chn,$0B,buffer+1,buffer^)
RETURN

PROC PrintDE(BYTE chn,
BYTE POINTER buffer)
PrintD(chn,buffer)
PutDE(chn)
RETURN

PROC Print(BYTE POINTER buffer)
PrintD(device,buffer)
RETURN

PROC PrintE(BYTE POINTER buffer)
PrintDE(device,buffer)
RETURN

;GRAPHICS-Commands
;===============

PROC Graphics(BYTE Gr)
Close(6)
Open(6,"S:",(Gr&$F0)!$1C,Gr)
RETURN

PROC Xio (BYTE chn,xx,cmd,aux1,aux2,BYTE POINTER fname)

BYTE ARRAY fstr(17)
BYTE POINTER bptr
BYTE z

bptr=fname+1
FOR z=0 TO fname^-1 DO
fstr(z)=bptr^
bptr==+1
OD
fstr(z)=$9B
SETAUX(chn,aux1,aux2)
CIOL(chn,cmd,fstr,0)
RETURN

PROC Position (CARD x,BYTE y)

BYTE py=$54
CARD px=$55

px=x
py=y

RETURN

PROC SetColor (BYTe reg,hue,lum)

BYTE ARRAY col(0)=$2C4
```

```
BYTE colw

colw=hue*16+lum

col(reg)=colw

RETURN

BYTE FUNC Stick (BYTE num)

BYTE ARRAY st(0)=$278

RETURN (st(num))

BYTE FUNC STrig (BYTE num)

BYTE ARRAY st(0)=$284

RETURN (st(num))

BYTE FUNC Paddle (BYTE num)

BYTE ARRAY pd(0)=$270

RETURN (pd(num))

BYTE FUNC PTrig (BYTE num)

BYTE ARRAY pd(0)=$27C

RETURN (pd(num))

BYTE FUNC Peek (CARD adr)

BYTE ret
BYTE POINTER addr

addr=adr
ret=addr^

RETURN (ret)

CARD FUNC PeekC (CARD adr)

CARD ret
BYTE POINTER addr

addr=adr+1
ret=addr^
ret==*$FF

addr==-1
ret==+addr^

RETURN (ret)

PROC Poke (CARD adr,BYTE value)

BYTE POINTER addr
```

```
addr=adr
addr^=value

RETURN

PROC PokeC (CARD adr,CARD value)

BYTE POINTER addr

addr=adr
addr^=value
addr==+1
addr^=value/$FF

RETURN

PROC Fill (CARD x,BYTE y)

BYTE col=$2FD

col=color
Position (x,y)
Xio (6,0,18,0,0,"S:")

RETURN

PROC DrawTo (CARD x,BYTE y)

BYTE iocmd=$3A2,
ioaux1=$3AA,
ioaux2=$3AB,
atachr=$2FB

atachr=color

iocmd=$11
ioaux1=$0C
ioaux2=0

Position (x,y)
CIOV (0,$60)

RETURN

PROC Plot (CARD x,BYTE y)

BYTE iocmd=$3A2,
icbll=$3A8,
icblh=$3A9

iocmd=$0B
icbll=0
icblh=0

Position (x,y)
CIOV (color,$60)

RETURN
```

```
BYTE FUNC Locate (CARD x,BYTE y)

BYTE ret

Position (x,y)
ret=GetD (device)

RETURN (ret)

PROC Sound (BYTE voic,freq,zerr,val)

BYTE ARRAY audfc(0)=$D200

voic==*2
audfc(voic)=freq
audfc(voic+1)=zerr*16+val

RETURN

PROC SndRst ()

BYTE u

FOR u=0 TO 3
DO
Sound (u,0,0,0)
OD

RETURN

BYTE FUNC Rand (BYTE rang)

BYTE random=$D20A,rand

DO
rand=random
UNTIL rand<=rang
OD

RETURN (rand)

PROC PrintCD (BYTE chan,CARD value)


BYTE f
BYTE POINTER adr

CARD inbuff=$F3,
fr0=$D4


fr0=value

IFP ()
FASC ()

adr=inbuff
f=1
```

```
DO
fr0=adr^
IF fr0>$80 THEN
f=0
fr0==-$80
FI
PutD (chan,fr0)
adr==+1
UNTIL f=0
OD
RETURN

PROC PrintC (CARD value)

PrintCD (device,value)

RETURN

PROC PrintCE (CARD value)

PrintCD (device,value)
PutD (0,$9B)

RETURN

PROC PrintCDE (BYTE chan,CARD value)

PrintCD (chan,value)
PutD (device,$9B)

RETURN

PROC PrintB (BYTE value)

PrintC (value)

RETURN

PROC PrintBD (BYTE chan,BYTE value)

PrintCD (chan,value)

RETURN

PROC PrintBE (BYTE value)

PrintCE (value)

RETURN

PROC PrintBDE (BYTE chan,BYTE value)

PrintCDE (chan,value)

RETURN

PROC PrintID (BYTE chan,CARD value)
```

```
IF value>32767 THEN
PutD (chan,$2D)
value==!$FFFF
value==+1
FI

PrintCD (chan,value)

RETURN

PROC PrintI (CARD value)

PrintID (device,value)

RETURN

PROC PrintIE (CARD value)

PrintID (device,value)
PutD (device,$9B)

RETURN

PROC PrintIDE (BYTE chan,CARD value)

PrintID (chan,value)
PutD (chan,$9B)

RETURN

CARD FUNC InputCD (BYTE chan)

BYTE cix=$F2
BYTE ARRAY lbuff(0)=$580
CARD inbuff=$F3,fr0=$D4

CIOL (chan,5,lbuff,39)

inbuff=$580
cix=0

AFP ()
FPI ()

RETURN (fr0)

CARD FUNC InputC ()
;
RETURN (InputCD (device))

BYTE FUNC InputBD (BYTE chan)
;
RETURN (InputCD (chan))

BYTE FUNC InputB ()
;
RETURN (InputCD (device))

CARD FUNC InputID (BYTE chan)
```

```
BYTE cix=$F2,flag
BYTE ARRAY lbuff(0)=$580
CARD inbuff=$F3,fr0=$D4

CIOL (chan,5,lbuff,39)

inbuff=$580
cix=0
flag=0

IF lbuff(0)='- THEN
cix=1
flag=1
FI

AFP ()
FPI ()

IF flag=1 THEN
fr0=-fr0
FI

RETURN (fr0)

PROC InputI ()

InputID (device)

RETURN

PROC InputMD (BYTE chan,BYTE ARRAY adr,BYTE max)

BYTE u
CARD ARRAY icbl(0)=$348

CIOL (chan,5,adr+1,max+1)

adr(0)=0
u=0
DO
adr(0)==+1
u==+1
UNTIL adr(u)=$9B
OD
adr (u)=$20
RETURN

PROC InputSD (BYTE chan,BYTE ARRAY adr)

InputMD (chan,adr,120)

RETURN

PROC InputS (BYTE ARRAY adr)

InputMD (device,adr,120)

RETURN
```

```
PROC SCopy (BYTE ARRAY dest,source)

BYTE u

FOR u=0 to source(0)
DO
dest(u+1)=source(u)
OD

RETURN

PROC SCopyS (BYTE ARRAY dest,source,BYTE start,stop)

BYTE u,st

st=stop-(start)

FOR u=0 to st
DO
dest(u+1)=source(u+start)
OD

RETURN

PROC SAssign (BYTE ARRAY dest,source,BYTE start,stop)

BYTE u,st

st=stop-start

FOR u=1 to st
DO
dest (u+start)=source(u)
OD

RETURN

PROC StrC (CARD num,BYTE ARRAY dest)
;
RETURN

PROC StrB (BYTE num,BYTE ARRAY dest)
;
RETURN

PROC StrI (INT num,BYTE ARRAY dest)
;
RETURN

BYTE FUNC ValB (BYTE ARRAY source)
;
RETURN (0)

CARD FUNC ValC (BYTE ARRAY source)
;
RETURN (0)

INT FUNC ValI (BYTE ARRAY source)
```

```
;
RETURN (0)

PROC Error (BYTE errcode)
;
RETURN

PROC SetBlock (BYTE POINTER adr,CARD size,BYTE value)

CARD u

FOR u=0 TO size-1
DO
adr^=value
adr==+1
OD

RETURN

PROC Zero (BYTE POINTER adr,CARD size)

SetBlock (adr,size,0)

RETURN

PROC MoveBlock (BYTE POINTER dest,source,CARD size)

CARD u

FOR u=0 to size-1
DO
dest^=source^
dest==+1
source==+1
OD

RETURN

PROC Point (BYTE chan,CARD sec,BYTE byt)

BYTE ARRAY iocb(0)=$340

iocb (chan*$10+2)=$25
iocb (chan*$10+$C)=sec
iocb (chan*$10+$E)=byt

CIOV (0,chan)

RETURN

PROC Note (BYTE chan,CARD sec,BYTE byt)

BYTE ARRAY iocb(0)=$340

iocb (chan*$10+2)=$26

CIOV (0,chan)

sec=iocb (chan*$10+$C)
```

```
byt=iocb (chan*$10+$E)

RETURN
```