Duo Blast

You will need to use sprite pad to draw 3 sprites. Actually, just draw two triangles. One that points up and another that points down. Finally draw a small square or a dot then save the lot. Now try this Acme Cross Assembler routine.

source

Creating your first game

```
;This is an example tutorial for you to
;learn how to create your own 2 player
;Shot 'em Up, which includes joystick
;control, a score system, etc.

;Let's setup the perameters for our
;game. These are the player positions.

plr1_x   = $0340 ;Player 1 x-position
plr1_y   = $0341 ;Player 1 y-position
plr2_x   = $0342 ;Player 2 x-position
plr2_y   = $0343 ;Player 2 y-position

;Our parameters for the player bullets

plr1b_x  = $0344 ;Player 1 bullet-xpos
plr1b_y  = $0345 ;Player 1 bullet-ypos
plr2b_x  = $0346 ;Player 2 bullet-xpos
plr2b_y  = $0347 ;Player 2 bullet-ypos

;Collisions parameters

plr1col  = $0350 ;Value storage for p1
plr2col  = $0354 ;Value storage for p2

;bullet lockup routines

plr1lockup = $0360 ;Lockup for players
plr2lockup = $0361 ;shooting

;We need to create a jump start for this
;example, so we will create our own jump
;address where it does not overlap data
;which had been loaded.

         !to "duoblast.prg",cbm
         *= $1000-2
         !binary "music.dat"
         *= $2000-2
```

```
         !binary "sprites.dat"
         *= $2400

start      sei ;Set irq flag

;Clear the screen without JSR $E544

         ldx #$00    ;Calls a routine
clear    lda #$20    ;to fill the whole
         sta $0400,x ;screen with #$20,
         sta $0500,x ;which is the
         sta $0600,x ;blank space
         sta $06e8,x ;routine.
         inx         ;
         bne clear   ;

;You should be familiar with the next
;example code. If not then look at the
;earlier chapters of A$$EMBLE IT!

         lda #$00
         sta $d020
         sta $d021

         lda #$16
         sta $d018

         lda #$1b
         sta $d011

         lda #$ff
         sta $d015

;Now here we create the sprite objects

         lda #$81
         sta $07f8 ;Player 1 ship
         lda #$82
         sta $07f9 ;Player 1 bullet
         lda #$80
         sta $07fa ;Player 2 ship
         lda #$82
         sta $07fb ;Player 2 bullet

;This is where we setup the colours of
;the two players and bullets.

         lda #$02  ;Colour red
         sta $d027 ;Player 1
         sta $d028 ;Player 1 bullet
```

```
         lda #$07  ;Colour yellow
         sta $d029 ;Player 2
         sta $d02a ;Player 2 bullet

;This is a different routine, as now the
;default sprite positions are copied to
;the declared parameters.

         lda $d000   ;This is a method
         sta plr1_x  ;of copying and
         lda $d001   ;pasting the sprite
         sta plr1_y  ;positions so that
         lda $d002   ;you can use this
         sta plr1b_x ;to program the
         lda $d003   ;sprites positions
         sta plr1b_y ;later on in the
         lda $d004   ;routines, for a
         sta plr2_x  ;much faster
         lda $d005   ;and decent
         sta plr2_y  ;movement for all
         lda $d006   ;the sprites in
         sta plr2b_x ;this game.
         lda $d007
         sta plr2b_y

;Now we reposition the two players and
;put the bullets into zero

         lda #$42    ;All this is the
         sta plr1_y  ;repositioning the
         lda #$18    ;two player ships,
         sta plr1_x  ;by using the 'x'
         lda #$e0    ;positions and the
         sta plr2_y  ;'y' positions,
         lda #$98    ;as simple as that
         sta plr2_x  ;:)

         lda #$00    ;All bullets are
         sta plr1b_x ;repositioned to
         sta plr1b_y ;the zero value
         sta plr2b_x ;yet again 'x' and
         sta plr2b_y ;'y' positions

;Setup the scoreboard

         lda #$30  ;We put zero on:
         sta $0400 ;first line
         lda #$02  ;paint first line
         sta $d800 ;red
         lda #$30  ;zero put on
         sta $0427 ;first line as last
```

```
         lda #$07  ;paint character
         sta $d827 ;yellow

;Now for the main body of this program
;the Interrupt flag, but we wont use
;JMP $EA81 or JMP $EA31, as no keyboard
;control will be required

         lda #<int ; Call INT values
         ldx #>int ; into an IRQ raster
         ldy #$00  ; interrupt value and
         sta $0314 ; zero the rasterline
         stx $0315 ;
         sty $d012
         lda #$7f  ; Keep the screen on
         ldx #$1b  ; and continue the
         sta $dc0d ; main interrupt
         stx $d011 ; read

         lda #$00  ;Initialise music
         tax       ;according to tune
         tay       ;number
         jsr $1000 ;

         lda #$01
         sta $d019 ;IRQ is turned on
         sta $d01a ;----------------

         lda $dc0d ;Copy $DC0D to $DD0D
         sta $dd0d ;to have the IRQ
                   ;working properly.

         cli       ;Clear IRQ flag
loop     jmp loop  ;Jump to the loop

;Our main interrupt

int      asl $d019 ;Keep $D019 running

;Call routine to expand and reconvert
;the sprite positions

         jsr expand

;Call routine to read joystick port 2
;for player 1 and player 2

         jsr read1up
         jsr read2up

;Call routine for bullet movements
```

```
         jsr bullmove

;Call routine for collision detection
;and player 1 and player 2 bullet to
;player collision.

         jsr detect
         jsr p1col
         jsr p2col

;And finally play the music

         jsr $1003 ;Play music

         pla ; An IRQ loop routine.
         tay ;
         pla ; This will keep all the
         tax ; jsr routines playing
         pla ; without using JMP $EA81
         rti ; or JMP $EA31

;Expand and reconver the sprite position

expand   lda plr1_y   ;Copy player y
         sta $d001    ;to exact position
         lda plr1b_y  ;Copy bullet 1 y
         sta $d003    ;to exact position
         lda plr2_y   ;The same goes
         sta $d005    ;with this routine
         lda plr2b_y  ;but instead it
         sta $d007    ;works with p2.

         lda plr1_x   ;Copy player x
         asl a        ;calculate 64
         ror $d010    ;Expand x pos.
         sta $d000    ;put at exact xpos
         lda plr1b_x  ;Copy bullet
         asl a        ;and do the same
         ror $d010    ;as with the
         sta $d002    ;player.
         lda plr2_x   ;
         asl a        ;All this is the
         ror $d010    ;same except that
         sta $d004    ;it will work with
         lda plr2b_x  ;Player 2 and the
         asl a        ;Player 2 bullet
         ror $d010    ;instead.
         sta $d006    ;
         rts          ;

;Read joystick control for player1
```

```
read1up  lda $dc00 ;Read port 2
         lsr a ;Joystick up
         lsr a ;Joystick down
left1    lsr a ;Joystick left
         bcs right1
         ldx plr1_x ;Move player 1
         dex        ;left across screen
         dex        ;using 2 for speed
         cpx #$0e   ;does the player
         bcs set1   ;move further than
         ldx #$0e   ;$0e, if so then
set1     stx plr1_x ;stop. Else continue
right1   lsr a      ;Joystick right
         bcs fire1
         ldx plr1_x ;Move player 1
         inx        ;right across screen
         inx        ;using the same
         cpx #$9a   ;speed. If player
         bcc set2   ;exceeds $9a, then
         ldx #$9a   ;make the player
set2     stx plr1_x ;stop.

fire1    lsr a ;Firebutton
         bcs nojoy1
         lda plr1lockup ;Is player fire
         cmp #$00       ;control
         bne nojoy1     ;unlocked. If so
         lda #$01       ;then lock fire
         sta plr1lockup ;and position
         ldx plr1_x     ;the bullet x
         stx plr1b_x    ;and y positions
         ldx plr1_y     ;exactly at the
         stx plr1b_y    ;same areas as
                        ;the player.
nojoy1   rts

;Read joystick control for Player 2
;(joystick port 1)

read2up  lda $dc01    ;Read JOY Port 1
         lsr a ;up
         lsr a ;down  ;Please read the
left2    lsr a ;left  ;player 1
         bcs right2   ;joystick control
         ldx plr2_x   ;as this routine
         dex          ;is exactly the
         dex          ;same as the
         cpx #$0e     ;player 1 controls
         bcs set1_1   ;but works for
         ldx #$0e     ;player 2.
set1_1   stx plr2_x
```

```
right2   lsr a
         bcs fire2
         ldx plr2_x
         inx
         inx
         cpx #$9a
         bcc set1_2
         ldx #$9a
set1_2   stx plr2_x
fire2    lsr a
         bcs nojoy2
         lda plr2lockup
         cmp #$00
         bne nojoy2
         lda #$01
         sta plr2lockup
         ldx plr2_x
         stx plr2b_x
         ldx plr2_y
         stx plr2b_y
nojoy2   rts

;call bullet routines

bullmove ldx plr1b_y  ;The bullet moves
         inx          ;but if it hits
         inx          ;$F6+ it will stop
         inx          ;off screen else
         inx          ;if below that
         inx          ;limit, the bullet
         inx          ;/ moves on, else
         cpx #$f6     ;bullet off screen
         bcc repsbul1
         lda #$00       ;Turn off the
         sta plr1lockup ;fire lockup
         ldx #$f6
repsbul1 stx plr1b_y

         ldx plr2b_y  ;The bullet moves
         dex          ;up but if it hits
         dex          ;$0C- it will stop
         dex          ;off screen, else
         dex          ;if below that
         dex          ;limit, the bullet
         dex          ;/ moves on. else
         cpx #$06     ;bullet off screen
         bcs repsbul2
         lda #$00       ;Turn off the
         sta plr2lockup ;fire lockup
         ldx #$06
repsbul2 stx plr2b_y
```

```
         rts

;Setup collision detection for player 1
;and player 2

detect   lda plr1_x       ;Calculate
         sec              ;the storage
         sbc #$06         ;values for
         sta plr1col+$00  ;the exact
         clc              ;positions of
         adc #$0c         ;player 1, so
         sta plr1col+$01  ;that when the
         lda plr1_y       ;opponents'
         sec              ;bullet hits
         sbc #$0c         ;the player,
         sta plr1col+$02  ;it has to be
         clc              ;in the exact
         adc #$18         ;position of
         sta plr1col+$03  ;player 1.

         lda plr2_x       ;The same
         sec              ;calculations
         sbc #$06         ;for player 2
         sta plr2col+$00  ;bullet to
         clc              ;player
         adc #$0c         ;collision.
         sta plr2col+$01
         lda plr2_y
         sec
         sbc #$0c
         sta plr2col+$02
         clc
         adc #$18
         sta plr2col+$03
         rts

;Check player 1 bullet collision on
;player 2 ship

p2col    lda plr1b_x     ;Is the bullet
         cmp plr2col+$00 ;at the correct
         bcc missp2      ;position where
         cmp plr2col+$01 ;the player is?
         bcs missp2      ;
         lda plr1b_y     ;If not then
         cmp plr2col+$02 ;the bullet
         bcc missp2      ;misses the
         cmp plr2col+$03 ;player.
         bcs missp2      ;
         lda #$f6        ;Else move
         sta plr1b_y     ;bullet off
```

```
         lda #$00        ;screen, turn
         sta plr1lockup  ;off fire lock
         inc $0400       ;add 1 point
         lda $0400       ;check score
         cmp #$3a        ;is it over 9?
         bne missp2      ;if not then
                         ;miss, else
         jmp victory1    ;jump to win
missp2   rts

;Check player2 bullet on player 1 ship

p1col    lda plr2b_x     ;Does the
         cmp plr1col+$00 ;bullet hit
         bcc missp1      ;the player in
         cmp plr1col+$01 ;exact position
         bcs missp1      ;of the player
         lda plr2b_y     ;ship? If so
         cmp plr1col+$02 ;then continue
         bcc missp1      ;else jump to
         cmp plr1col+$03 ;missp1
         bcs missp1
         lda #$06
         sta plr2b_y
         lda #$00
         sta plr2lockup
         inc $0427       ;Add 1 point
         lda $0427       ;check score
         cmp #$3a        ;is it over '9'
         bne missp1      ;if so then
         jmp victory2    ;player2 wins
missp1   rts

victory1 sei             ;Stop all IRQs
         lda #$00        ;Turn off all
         sta $d015       ;the sprites

         ldx #$00        ;Call a routine
win1     lda vic1,x      ;to display
         sta $0400,x     ;player1 wins
         lda #$01        ;message and
         sta $d800,x     ;paint the
         inx             ;message white
         cpx #$28        ;and display
         bne win1        ;as 40 chars

         jmp space       ;Jump to space
                         ;prompt

victory2 sei             ;   Look at
         lda #$00        ;  'victory1'
```

```
         sta $d015       ;same function

         ldx #$00        ;This routine
win2     lda vic2,x      ;does exactly
         sta $0400,x     ;the same, but
         lda #$01        ;it reads the
         sta $d800,x     ;text from
         inx             ;vic2 so that
         cpx #$28        ;player 2 wins
         bne win2        ;the game

         jmp space       ;You know this
                         ;if not then
                         ;look at the
                         ;previous
                         ;'jmp space'
                         ;prompt

space    ldx #$00        ;Another text
setspc   lda spc,x       ;display
         sta $07c0,x     ;routine. This
         lda #$03        ;time position
         sta $dbc0,x     ;the text at
         inx             ;the bottom of
         cpx #$28        ;the screen as
         bne setspc      ;40 chars

;Lame hit space routine :)

hitspace lda #$80        ;Create a
raster   cmp $d012       ;raster control
         bne raster      ;to continue
         jsr $1003       ;playing music

         lda $dc01       ;Read SPACEBAR
         cmp #$ef        ;if not pressed
         bne raster      ;then jump to
                         ;the raster to
                         ;continue
                         ;the SPACE read
                         ;routine and
                         ;play music

         jmp start       ;Restart game

;Our text display data tables.

         ;Player 1 victory

vic1     !scr "player 1 proves that"
         !scr " player 2 is a loser"
```

```
         ;Player 2 victory

vic2     !scr "player 2 proves that"
         !scr " player 1 is a loser"

         ;Spacebar prompt message

spc      !scr "  press the spacebar"
         !scr " for another game!  "
```