

Format CALL COLLIDE(#sprite-number,#sprite-number,
 tolerance,dot-row,dot-column[,...])

CALL COLLIDE(#sprite-number,dot-row,
dot-column,tolerance,dot-row,
dot-column[,...])

Description

See EXTENDED BASIC MANUAL PAGE 64 has COINC. The XB COINC never tells you the location of a sprite and absolutely limits the types of way sprites could be used.

If sprites COLLIDE where did this happen?

COLLIDE tells you exactly where they did collide and the location of how close to the hit box you wanted be informed. Tolerance could be up to 256 pixels which could always be a collide result or 0 for exactly on pixel of top left corner of the sprite. I recommend a setting of 6 for best results. COLLIDE runs from ROM.

Programs

Clear screen	>100 CALL CLEAR ! SPRITES
Set up 3 sprites to be on screen	>110 CALL SPRITE(#1,65,2,9,99, 20,22,#2,66,2,64,99,X,30,25, #3,67,2,9,99,-20,-35)
COLLIDE scans 3 sprites for sprite hits on #1,#2,#3 sprite	>120 CALL COLLIDE(#1,#2,8,R1, C1,#1,#3,8,R2,C2,#2,#3,8,R3, C3)
Check for non zero?	>130 IF R1+C1+R2+C2+R3+C3
If zero loop forever	THEN 140 ELSE 120
Show hits or non hits	>140 PRINT "#1";R1;C1;"#2";R2; ;C2;"#3";R3;C3
Zero out variables	>150 R1,C1,R2,C2,R3,C3=0
Loop forever	>160 GOTO 120

Clear screen	>100 CALL CLEAR ! ROW: COLUMN
Set up 3 sprites to be on screen	>110 CALL SPRITE(#1,65,2,9,99, 20,22,#2,66,2,64,99,30,25,#3, 67,2,9,99,-20,-20)
COLLIDE for DOT ROW DOT COLUMN at row 99 and column 99 for sprites #1,#2,#3 hit?	>120 COLLIDE(#1,99,99,8,R1,C1, #2,99,99,8,R2,C2,#3,99,99,8, R3,C3)
Check for non zero?	>130 IF R1+C1+R2+C2+R3+C3
If zero loop forever	THEN 140 ELSE 120
Zero out variables	>150 R1,C1,R2,C2,R3,C3=0
Loop forever	>160 GOTO 120