COLLSLIB

Conversion program

COLLABORATORS			
	TITLE : COLLSLIB		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY	Conversion program	October 9, 2022	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

COLLSLIB

1.1 Overview of COLLSLIB

Overview

An Acid Software Library

Converted to AmigaGuide by

Red When Excited Ltd

Used with the permission of Acid Software

Edited, fixed and cleaned by Toby Zuijdveld 26/02/1999.
mailto:hotcakes@abacus.net.au

1.2 COLLSLIB

```
Statement: SetCollOdd
------------------------------------------------------------------------------
Modes  :
Syntax : SetCollOdd

SetCollOdd is used to control the detection of sprite/bitmap
collisions. SetCollOdd will cause ONLY the collisions between sprites
and 'odd coloured' bitmap graphics to be reported. Odd coloured bitmap
graphics refers to any bitmap graphics rendered in an odd colour number
(ie: 1,3,5...). This allows you to design bitmap graphics in such a way
that even coloured areas are 'safe' (ie: they will not report a
collision) whereas odd colour areas are 'unsafe' (ie: they will report
a collision).

The DoColl and PColl commands are used to detect the actual
sprite/bitmap collisions.
```

1.3 COLLSLIB

```
Statement: SetColl
------------------------------------------------------------------------------
Modes  :
Syntax : SetColl Colour,Bitplanes[,Playfield 1=front, 2=back]
```

There are 3 different commands involved in controlling sprite/bitmap collision detection, of which SetColl is one (the other 2 being SetCollOdd and SetCollHi). All three determine what colours in a bitmap will cause a collision with sprites. This allows you to design bitmaps with 'safe' and 'unsafe' areas.

SetColl allows you to specify a single colour which, when present in a bitmap, and in contact with a sprite, will cause a collision. The Colour parameter refers to the 'collidable' colour. Bitplanes refers to the number of bitplanes (depth) of the bitmap collisions are to be tested for in.

The optional PlayField parameter is only used in a dualplayfield slice. If Playfield is 1, then Colour refers to a colour in the foreground bitmap. If Playfield is 0, then Colour refers to a colour in the background bitmap.

DoColl and PColl are the commands used for actually detecting the collisions.

1.4 COLLSLIB

```
Statement: SetCollHi
------------------------------------------------------------------------------
Modes  :
Syntax : SetCollHi Bitplanes
```

1.5 COLLSLIB

```
Function: ShapesHit
------------------------------------------------------------------------------
Modes  :
Syntax : ShapesHit (Shape#,X,Y,Shape#,X,Y)
```

The ShapesHit function will calculate whether the rectangular areas occupied by 2 shapes overlap. ShapesHit will automatically take the shape handles into account.
If the 2 shapes overlap, ShapesHit will return true (-1), otherwise ShapesHit will return false (0).

1.6 COLLSLIB

```
Function: ShapeSpriteHit
------------------------------------------------------------------------------
Modes  :
Syntax : ShapeSpriteHit (Shape#,X,Y,Sprite#,X,Y)

The ShapeSpriteHit function will calculate whether the rectangular area
occupied by a shape at one position, and the rectangular area occupied
by a sprite at another position are overlapped. If the areas do
overlap, ShapeSpriteHit will return true (-1), otherwise ShapeSpriteHit
will return false (0).

ShapeSpriteHit automatically takes the handles of both the shape and
the sprite into account.
```

1.7 COLLSLIB

```
Function: SpritesHit
------------------------------------------------------------------------------
Modes  :
Syntax : SpritesHit (Sprite#,X,Y,Sprite#,X,Y)

The SpritesHit function will calculate whether the rectangular areas
occupied by 2 sprites overlap. SpritesHit will automatically take the
sprite handles into account.

If the 2 sprites overlap, SpritesHit will return true (-1), otherwise
SpritesHit will return false (0).
```

1.8 COLLSLIB

```
Function: RectsHit
------------------------------------------------------------------------------
Modes  :
Syntax : =RectsHit (X1,Y1,Width1,Height1,X2,Y2,Width2,Height2)

The RectsHit function may be used to determine whether 2 arbitrary
rectangular areas overlap. If the specified rectangular areas overlap,
RectsHit will return true (-1), otherwise RectsHit will return false
(0).
```

1.9 COLLSLIB

```
Function: SColl
------------------------------------------------------------------------------
```

```
Modes  :
Syntax : SColl (Sprite Channel,Sprite Channel)

SColl may be used to determine whether the 2 sprites currently
displayed through the specified sprite channels have collided. If they
have, SColl will return true (-1), otherwise SColl will return false
(0).
DColl must have been executed prior to using SColl.
```

1.10 COLLSLIB

```
Function: PColl
------------------------------------------------------------------------------
Modes  :
Syntax : PColl (Sprite Channel)

The PColl function may be used to find out if a particular sprite has
collided with any bitmaps. Sprite Channel refers to the sprite channel
the sprite you wish to check is being displayed through.
If the specified sprite has collided with any bitmap graphics, PColl
will return a true (-1) value, otherwise PColl will return false
(0).

Before using PColl, a DoColl must previously have been executed. Please
refer to DoColl for more information.
```

1.11 COLLSLIB

```
Statement: DoColl
------------------------------------------------------------------------------
Modes  :
Syntax : DoColl

DoColl is used to perform sprite/bitmap collision checking. Once DoColl
is executed, the PColl and/or SColl functions may be used to check for
sprite/bitmap or sprite/sprite collisions.

Before DoColl may be used with PColl, the type of bitmap collisions to
be detected must have been specified using one of the SetColl,
SetCollOdd or SetCollHi commands.

After executing a DoColl, PColl and SColl will return the same values
until the next time DoColl is executed.
```

1.12 COLLSLIB

.---

| COLLSLIB |
`---'

Overview

Command Index

DoColl

PColl

RectsHit

SColl

SetColl

SetCollHi

SetCollOdd

ShapesHit

ShapeSpriteHit

SpritesHit