

AIS CUBE

OLED



[THE BLAZINGCORE SERIES]

















WITH SUPERIOR NUMBER CRUNCHING ABILITIES AND PERIPHERAL HANDLING ON OUR CUSTOM EMBEDDED OS,
RAPID PROTOTYPING IS NOW EASY... AND BLAZING FAST.

OLED COMMAND REFERENCE

The following pages list the members in the library available to interface with the 2.83" OLED Display with Touch Screen Board. Ready methods like print text to screen, drawing primitives or even placing bitmaps to the screen are available for immediate usage. Examples of use are also included.

OLED

Part of the Sonata IDE library for the BlazingCore OS consists of optimized methods to control the OLED natively to ensure efficiency and speed of data transfer to the OLED Driver.

Method	Description
 Init ()	Does the necessary one-time initialization of the OLED with a specified colour
 Foreground	Set Foreground colour
 Background	Set Background colour
 Clear ()	Clears the screen
 Orientation	Set the orientation of the screen to portrait or landscape
 Cursor	Get or Set the Cursor Position
 FontType	Set the Font Size
 Print	Print text to OLED
 Draw	Graphic library for drawing Primitives and Bitmaps to Screen <i>*refer to subtopic for details.</i>
 UI	User Interface Library consisting of UI Components like an Onscreen-Keyboard
 Touch	Provides access to reading values from the touch screen
 Accel	Provides Access to reading values from the onboard accelerometer
 LDR	Provides access to reading values from the onboard Light Dependent Resistor (LDR)
 GRAM	Provides Read-Only Access to the Graphics RAM
 ReadRegister	Provides read abilities to the OLED driver (S6E63D6) registers ¹
 WriteRegister	Provides write abilities to the OLED driver (S6E63D6) registers ¹

¹ Please refer to S6E63D6 Driver IC Datasheet for full register listing.

.INIT()

Does the necessary one-time initialising required for the OLED to be controlled thereafter. Parameter accepts colour for the OLED screen to be initialised with.

***Note:** Colour listing available. Please refer to the page on using colour constants.

```
OLED.Init(colour)
```

Code:

```
'CODE1
Public Sub Main()
'INIT THE OLED AND SET THE BACKGROUND COLOUR TO BLACK
OLED.Init(Const.Colour.Black)
End Sub
```

OLED COMMAND REFERENCE

.FOREGROUND

Get/Set the foreground colour of the OLED.

OLED.Foreground.Colour

Code:

```
'CODE1

Public Sub Main()
'INIT THE OLED AND SET THE BACKGROUND COLOUR TO BLACK
OLED.Init(Const.Colour.Black)
OLED.Foreground.Colour = Const.Colour.Fuschia
OLED.Print "Hello World!"
End Sub
```



.BACKGROUND

Get/Set the background colour of the OLED.

OLED.Background.COLOUR

Code:

```
'CODE1

Public Sub Main()
'INIT THE OLED AND SET THE BACKGROUND COLOUR TO BLACK
OLED.Init( Const.Colour.Black )
'CHANGE THE BACKGROUND COLOUR TO FUSCHIA
OLED.Background.Colour = Const.Colour.Fuschia
End Sub
```

.CLEAR()

Clears the screen.

OLED.Clear()



.ORIENTATION

Get/Set the orientation of the OLED screen to portrait or landscape

OLED.Orientation

Parameter/s:

- Orientation;
 - 0: Landscape (Default)
 - 1: Portrait

Code:

```
'CODE1

Public Sub Main()
OLED.Orientation = Const.OLED.Landscape
End Sub
```

.CURSOR

Get or Set the Cursor Position

Member	Description
X	Set X coordinate of cursor
Y	Set Y coordinate of cursor
Home	Sets the cursor to (0,0) position

To set the cursor position;

```
OLED.Cursor.X = value  
OLED.Cursor.Y = value  
OLED.Cursor.Home()
```


Value: Any literal, constant, or expression

To get the cursor position;

```
value = OLED.Cursor.X  
value = OLED.Cursor.Y
```

.FONTTYPE

Sets the font size of text printed to the screen.

Available font sizes are 8x8, 10x12, 16x16 pixels 

`OLED.FontType`

Assignment

- `OLED.FontType = font size;`
 - `0 = 8x8 pixel;`
 - `1 = 10x12 pixel;`
 - `2 = 16x16 pixel`

A BCore Constant that stores the font size values is available to make the code more readable.

The following shows a code comparison on how to set the font size using direct assignment of values and using the BCore Constants. Both codes achieve the same effect.

Code:

```
OLED.FontType = 0  
OLED.FontType = 1  
OLED.FontType = 2
```

Code:

```
OLED.FontType = Const.Font.Small  
OLED.FontType = Const.Font.Medium  
OLED.FontType = Const.Font.Large
```

.PRINT

Prints text to screen using specified foreground colour. Works just like `Debug.Print`.

***Note:** Default text size is 8x8pixels.

`OLED.Print "[string]"` | String Variable | String Const










*Full set of 256 ASCII characters supported.

.DRAW

Uses inbuilt native libraries in BCore to draw primitives and bitmaps on to the screen.

All primitives are drawn using the specified foreground colour.

Note: All drawing methods conform to OLED's current orientation setting.

Method	Description
 Point	Plots a pixel at a specified point on the OLED Screen
 Points	Plots an array of points.
 Line	Draws a Line.
 Lines	Draws an array of Lines.
 Rectangle	Draws a Rectangle of specified sizing at a specified position.
 FillRectangle	Draws a Filled Rectangle of specified sizing at a specified position.
 Circle	Draws a Circle of specified sizing at a specified position.
 BitmapFromDFlash	Draws a bitmap file from the data flash memory chip.
 BitmapFromMCard	Draws a bitmap from External Memory Card (MMC) to the Screen

OLED COMMAND REFERENCE

.POINT

Plots a single pixel at a specified point on the OLED Screen.

```
OLED.Draw.Point (point)
```

.POINTS

Plots an array of pixels.

Parameters:

- *Points()* = point array;
- *Zero-offset start* = zero-offset start index of point array to start plotting from;
- *Number of points* = number of points to plot.

```
OLED.Draw.Points (points(), zero-offset start, number of points)
```

.LINE

Draws a Line from *point1* to *point2* on the OLED Screen.

```
OLED.Draw.Line (point1, point2)
```

.LINES

Draws an array of points as a continuous connected line.

Parameters:

- *Points()* = point array;
- *Zero-offset start* = zero-offset start index of point array to start plotting from;
- *Number of points* = number of points to plot.

```
OLED.Draw.Lines (points(), zero-offset start, number of points)
```

.RECTANGLE

Draws a Rectangle using 2 points.

```
OLED.Draw.Rectangle (point1, point2)
```

.FILLRECTANGLE

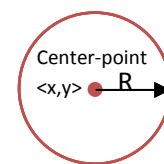
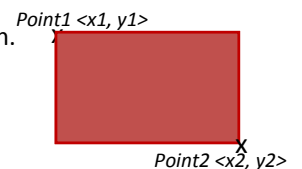
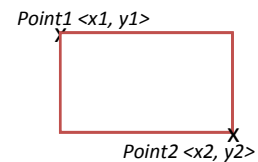
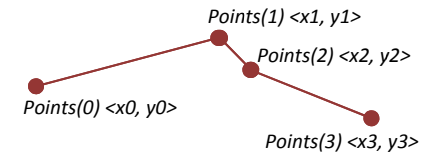
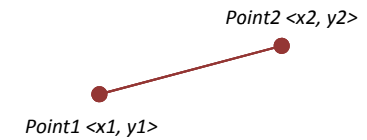
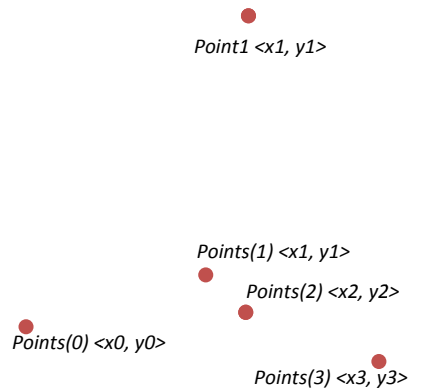
Draws a solid/filled rectangle of foreground colour from *point1* to *point2* on the OLED Screen.

```
OLED.Draw.FillRectangle (point1, point2)
```

.CIRCLE

Draws a circle. Parameters accept center point position and radius of circle.

```
OLED.Draw.Circle (center-point, radius)
```



OLED COMMAND REFERENCE

DRAWING BITMAPS TO SCREEN

Storage and Rendering of Bitmap files are supported by the BCore OS.

However, due to the nature of bitmap files, they tend to require a large amount of memory. As such, Bitmaps are only allowed to be stored on External Memory Cards (like MMC Cards), or the external DataFlash chip.

Depending on where you store your Bitmap Files, a single line command is able to fetch the image file and draw it onto the OLED screen, provided it is given the correct information.

Please note: The commands to draw bitmaps to screen returns a value and must be assigned to a variable.

This value will indicate if there has been any error during file fetching and/or drawing to the screen.

Value '0': Successful

Value '1': Error

Note: ¹Bitmaps must be in Windows® Bitmap .bmp format.

²Bitmaps of bit depth 16bit 565 Mode, 24bpp, 32bpp are supported.

³For applications that require the OLED to update images really quickly, we recommend that the bitmaps are first converted to the 16bit 565 RGB Format before importing it into the storage media for the BCore to access, since 24bpp and 32bpp files are converted by the chip to the 16bit 565 format before sending to the OLED Display, thereby imposing a limitation to the update rate from the BCore to the OLED Display.

.BITMAPFROMDFLASH

Draws a bitmap on the OLED screen from a .bmp file stored in the external DataFlash chip.

Code:

```
Dim P1 As Point
Public Sub Main()

    Dim I1 As Integer
    DEBUG.PRINT "OLED - BITMAP"
    DELAY(200)
    OLED.INIT(0)
    OLED.SETCOLOUR.FOREGROUND(31)
    I1 = ADDRESSOF(DATA1.B_PLAYER)
    P1.X = 50
    P1.Y = 20
    OLED.DRAW.BitmapFromDFlash(P1, I1)

End Sub
```

Note: Please refer to **APPENDIX A** for more details on transferring Bitmap Files from the PC to the BCore's External DataFlash.

.BITMAPFROMMCCARD

Draws a bitmap on the OLED screen from a .bmp file stored in the External Memory Card (MMC / SD Compatible).

```
<variable> = OLED.Draw.BitmapFromMCard(ByRef FileName As String, ByRef Position As Point)
```

Parameters:

- **FileName:** Name of Bitmap File in *String*
- **Position:** *Point* XY location to draw bitmap

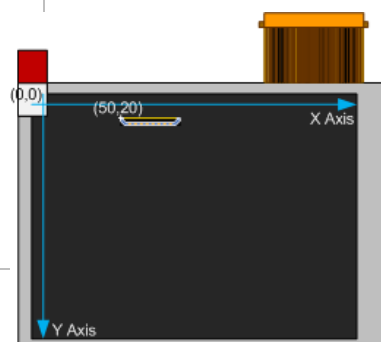
Returns: MCard File Operation Result

Code:

```
Dim P1 As Point
Public Sub Main()

    Dim I As Integer
    Dim STR1 AS STRING
    OLED.INIT(0)
    OLED.SETCOLOUR.FOREGROUND(31)
    I = OS.MEMCARD.INIT()
    STR1 = "B_PLAYER.BMP"
    P1.X = 50
    P1.Y = 20
    I = OLED.Draw.BitmapFromMCard(STR1, P1)

End Sub
```



.UI

User Interface Library consisting of UI Components like an Onscreen-Keyboard

Note: Keypad Display conforms to Orientation Setting

Member	Description
Init	Initialise User Interface with the specified Mode
Keypad	Provides access to methods for the Keypad region of the Onscreen Keyboard
Entry	Provides access to methods for the Entry region of the Onscreen Keyboard

.INIT

Initialise User Interface with the specified Mode

`OLED.UI.Init` (*ByVal Block As Integer, ByVal Mode As Integer*)

Parameters;

Block: Block Number in the Data Flash Chip containing the files used for the UI

Mode: Colour Scheme

.KEYPAD

Provides access to methods for the Keypad region of the Onscreen Keyboard

Member	Description
Hide	Hide the Keypad region with a specified colour
Show	Show the Keypad with a specified mode

.HIDE

Hide the Keypad region with a specified colour

`OLED.UI.Keypad.Hide` (*ByVal Colour As Integer*)

Parameters;

Colour: Colour Value used to hide the Keypad region

.SHOW

Show the Keypad with a specified mode

`OLED.UI.Keypad.Show` (*ByVal Mode As Integer*)

Parameter/s;

Mode:

- '0': Capital/Uppercase Alphabetical Letters
- '1': Lowercase Alphabetical Letters
- '2': Numerical

.ENTRY

Provides access to methods for the Keypad region of the Onscreen Keyboard

Member	Description
Hide	Hide the Entry region with a specified colour
Display	Display the Entry Field with a specified String value
GetKey	Poll the Keyboard for keys pressed

.HIDE

Hide the Entry region with a specified colour

```
OLED.UI.Entry.Hide (ByVal Colour As Integer)
```

Parameters;

Colour: Colour Value used to hide the Keypad region

.DISPLAY

Display the Entry Field with a specified String value

```
OLED.UI.Entry.Display (ByVal Data As String)
```

Parameter/s;

Data: String variable of keys obtained from onscreen keyboard to display in the entry field.

.GETKEY

Poll the Keyboard for keys pressed

```
<variable> = OLED.UI.Entry.GetKey() As Integer
```

Returns: Value of Key Pressed

.TOUCH

Provides access to touch screen methods.

Member	Description
X	Returns optimized X coordinate of touch
Y	Returns optimized Y coordinate of touch
Raw	X Returns Raw Value of X
	Y Returns Raw Value of Y

X

Returns calculated X coordinate of a touch on the touch screen.

```
<variable> = OLED.Touch.X()
```

Y

Returns calculated Y coordinate of a touch on the touch screen.

```
<variable> = OLED.Touch.Y()
```

Code:

```
Public X,Y as Integer

Public Sub Main()

    Do
        X = OLED.Touch.X()
        Y = OLED.Touch.Y()
        Debug.Print Cstr(X); " ";Cstr(Y)
        Delay(200)
    Loop

End Sub
```

X.RAW

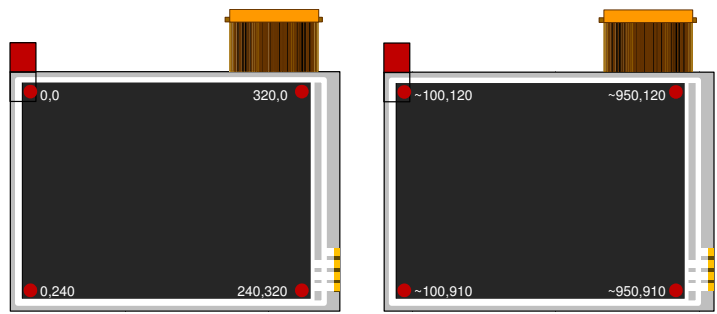
Returns raw X coordinate of a touch on the touch screen.

```
<variable> = OLED.Touch.X.Raw()
```

Y.RAW

Returns raw Y coordinate of a touch on the touch screen.

```
<variable> = OLED.Touch.Y.Raw()
```



Optimized Values

Raw Values

OLED COMMAND REFERENCE

.ACCEL

Provides Access to methods for reading values from the onboard accelerometer.

Member	Description
X	Get X axis reading
Y	Get Y axis reading
Z	Get Z axis reading

```
<variable> = OLED.Accel.X()  
<variable> = OLED.Accel.Y()  
<variable> = OLED.Accel.Z()
```

Code:

```
Public X,Y,Z as Integer  
  
Public Sub Main()  
    Do  
        X = OLED.Accel.X()  
        Y = OLED.Accel.Y()  
        Z = OLED.Accel.Z()  
        Debug.Print Cstr(X); " ";Cstr(Y); " ";Cstr(Z)  
        Delay(200)  
    Loop  
End Sub
```

.LDR

Provides access to reading values from the onboard Light Dependent Resistor (LDR)



```
<variable> = OLED.LDR.Read()
```

Code:

```
Public LDR as Integer  
Public Sub Main()  
    Do  
        LDR = OLED.LDR.READ()  
        Debug.Print Cstr(LDR)  
        Delay(200)  
    Loop  
End Sub
```

.GRAM

The pixels displayed on the OLED screen correspond to the image data stored in the Graphics RAM (GRAM).

Read access to the OLED GRAM is provided to the user.

Read the colour data direct from GRAM, by providing the position to read.

Read command returns the colour in 16bit 565 RGB Integer format.

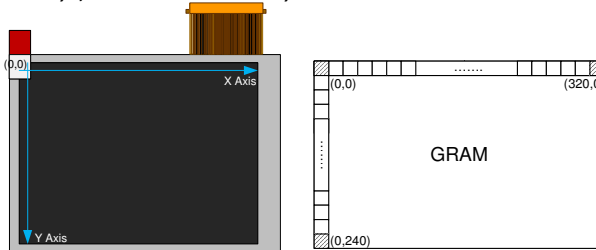
Member	Submember	Description
Read	Point	Returns the colour value from a single point in the GRAM
	Rectangle	Returns the colour values from a specified rectangular area in the GRAM into a specified user declared integer array.

```
<variable> = OLED.GRAM.Read.Point(point)
```

```
<variable> = OLED.GRAM.Read.Rectangle(memoryArr(), point1, point2)
```

Parameter/s:

- `memoryArr();`
 - User declared single or 2 dimensional integer array to store the colour values read from the GRAM
 - Users are expected to ensure that the memory allocation of the Integer Array is sufficient to store the total number of points being read to it
 - Data from specified area will be stored in a continuous stream inside the integer array. Any row alignments (applicable only to 2 dimensional arrays) will be automatically done.
- `point1`
 - Starting point of Rectangle
- `point2`
 - Ending point of Rectangle



.READREGISTER

Returns the contents of the OLED driver (S6E63D6) register at a specified address.

```
<variable> = OLED.ReadRegister(Address)
```

.WRITEREGISTER

Provides write abilities to the OLED driver (S6E63D6) registers

¹ Please refer to S6E63D6 Driver IC Datasheet for full register listing.

```
OLED.WriteRegister(Address, Value)
```

LATEST DOCUMENTATION

All of our documentations are constantly updated to provide accurate and/or new information that we feel would help you with developing with our products.

The latest documentation may be obtained from our website: <http://www.aiscube.com/main/downloads.html>

HOW YOU CAN HELP

You can help us to improve our documentations by emailing to us or posting a thread in our forum, reporting any mistakes/typos or errata that you might spot while reading our documentation.

Email: TechSupport@aiscube.com

Forum: <http://forum.aiscube.com/index.php>

DISCLAIMER

All information in this documentation is provided 'as-is' without any warranty of any kind.

The products produced by AIS Cube are meant for rapid prototyping and experimental usage; they are not intended nor designed for implementation in environments that constitute high risk activities.

AIS Cube shall assume no responsibility or liability for any indirect, specific, incidental or consequential damages arising out of the use of this documentation or product.

COPYRIGHT© 2009 - 2011 AIS CUBE. ALL RIGHTS RESERVED.

ALL PRODUCT AND CORPORATE NAMES APPEARING IN THIS DOCUMENTATION MAY OR MAY NOT BE REGISTERED TRADEMARKS OR COPYRIGHTS OF THEIR RESPECTIVE COMPANIES. AND ARE ONLY USED FOR IDENTIFICATION OR EXPLANATION FOR THE OWNER'S BENEFIT. WITH NO INTENT TO INFRINGE.

SONATA IDE AND BLAZINGCORE(BCORE) ARE TRADEMARKS OF AIS CUBE IN SINGAPORE AND/OR OTHER COUNTRIES. ALL IMAGES DEPICTING THE BLAZINGCORE OR ANY PART OF IT IS COPYRIGHTED.

ALL OTHER TRADEMARKS OR REGISTERED TRADEMARKS ARE THE PROPERTY OF THEIR RESPECTIVE OWNERS.