

AI5 CUBE

MCARD

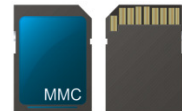
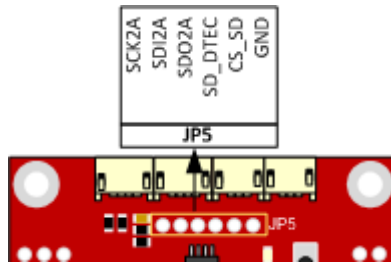


[THE BLAZINGCORE SERIES]

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

EXTERNAL MEMORY CARD

BCore provides support for accessing files located in the external memory card (MMC/SD) through JP5. MCards are accessible through the MMC Socket onboard our OLED Interface Board, or by using an MMC/SD Card Module wired to JP5.



M CARD

Provides access to methods for handling an external memory card(MMC) and file operations

| Members | Description |
|---------|---|
| Init | Handles the necessary one-time initialisation of the memory card before use |
| Dir | Provides methods to access the directories within the external memory card |
| File | Provides methods to access files and carry out file stream operations |

INIT

Handles the necessary one-time initialisation of the memory card before it can be used

Variable = `MCard.Init()` As Boolean

Returns;

- **True:** External Memory Card Successfully Initialised
- **False:** Error

Code:

```
Public Sub Main()
    I = MCard.Init()
    Debug.Print "INIT ";CSTR(I) 'I = Success/Error
End Sub
```

DIR

Provides methods to access the directories within the external memory card

| Methods | Description |
|-----------|---|
| Change | Change current directory within the external memory card |
| FindFirst | Returns first occurrence of a specified string in names of all the files in current directory |
| FindNext | Returns next occurrence of a specified string in names of all the files in current directory |
| Get | Get path of current directory |

.CHANGE

Change current directory to a specified one within the external memory card

Variable = `MCard.Dir.Change`(*STRING* PATH) As Boolean

Parameter/s:

- Path of desired directory in *String*

Returns;

- **True:** Operation Successful
- **False:** Error

Code:

```
Public Sub Main()

Dim Str as String
Dim I as Integer
STR = "\SDIR_2" 'FOLDER WITHIN MCard CALLED SDIR_2
I = MCard.Dir.Change(STR) 'GO INTO FOLDER
Debug.Print "CHANGE DIRECTORY STATUS: ";CSTR(I)

End Sub
```

.FINDFIRST

Returns first occurrence of a specified string in names of all the files in current directory

Variable = `MCard.Dir.FindFirst`(ByRef STR1 As String, ByRef STR2 As String) As Boolean

Parameter/s:

- **STR1** = Specified *string* to search for
- **STR2** = *String Variable* for Command to return *File Name* in which STR1 was found.

Returns;

- **True** : File Found
- **False**: No matches.

META CHARACTERS

The code below is an example of how to retrieve the first found file with the specified string out of all the files available within the current directory of the external memory card.

Use of * is a meta character representing any single character or group of characters.

By using *.* , the * stands in for any file name, as well as for any file extension.

Code:

```
Public Sub Main()

Dim Str1, Str2 as String
Dim I as Integer
STR1 = "*.*" '* = Wild card /meta character
I = MCard.Dir.FindFirst(STR1, STR2)
Debug.Print "FIND FIRST ";CSTR(I);" ";STR2

End Sub
```

.FINDNEXT

Returns next occurrence of a specified string in names of all the files in current directory; Continuing from where the previous file was found.

IMPORTANT: Use of `MCard.Dir.FindFirst` must precede this command

Variable = `MCard.Dir.FindNext` (ByRef STR2 As String) As Boolean

Parameter/s:

- **STR2** = String Variable for Command to return File Name in which specified string to search for (stated in the use of command (`MCard.Dir.FindFirst`), was found.

Returns;

- **True:** File Found
- **False:** No matches.

The code below is an example of how to retrieve the first 2 files with residing within the current directory of the external memory card.

Code:

```
Public Sub Main()  
  
Dim Str1, Str2 as String  
Dim I as Integer  
STR1 = "*" '* = Wild card/ meta character  
I = MCard.Dir.FindFirst (STR1, STR2)  
Debug.Print " FIND FIRST ";CSTR(I);" ";STR2  
I = MCard.Dir.FindNext (STR2)  
Debug.Print " FIND NEXT ";CSTR(I);" ";STR2  
End Sub
```

.GET

Get path of current directory

Variable = `Mcard.Dir.Get` (ByRef PATH As String) As Boolean

Parameter/s:

- **PATH;** String variable that contains path of directory that it is currently in.

Returns;

- **True:** File Operation Successful
- **False:** Error

Code:

```
Public Sub Main()  
  
Dim Str1, Str2 as String  
Dim I as Integer  
STR1 = "*" '* = Wild card/ meta character  
I = MCARD.Dir.Change (STR1)  
Debug.Print " CHANGE DIR:";CSTR(I) 'I = Success/Error  
I = Mcard.Dir.Get (STR2)  
Debug.Print CSTR(I);" DIR : [";STR2;"]"  
End Sub
```

FILE

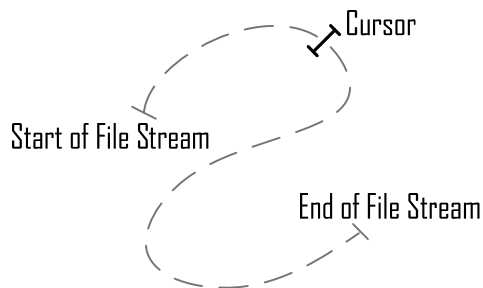
The BCore OS uses streams to support reading from and writing to files located in the external memory card (MMC).

| Methods | Description |
|---------|--|
| Open | Open a specified file |
| Close | Closes a currently open file stream |
| Read | Start reading a specified number of bytes of data from the file stream |
| Write | Perform write operations to specified file stream |
| Seek | Sets cursor position to a specified index within the file stream |
| GetPos | Returns current position of data being read within an open file |
| IsEOF | Checks if end of file stream has been reached |
| Rewind | Sets cursor position back to the beginning of the file stream |

FILE STREAMS

When the BCore OS accesses the files located in the external memory card (MMC), it performs file operations through the use of streams. A file stream is essentially a single dimensional set of contiguous data obtained from the file. A file stream has a start and an end, and a cursor that represents the current position of data being read from the file.

A visual representation of a file stream can be seen in the image below.



.OPEN

Open a specified file located in the external memory card (MMC).

```
Variable = MCard.File(FileNum).Open(FileName, R/W) As Boolean
```

Parameter/s:

- **FileNum;** Assign an available number to associate the file with
- **FileName;** String variable containing name of file to open
- **R/W;**
 - **0**, to open a file for **read** operations,
 - **1**, to open the file for **write** operations
 - **2**, to open the file for **read and write** operations

Returns:

- **True:** File successfully opened
- **False:** Error

Code:

```
Dim Str1, Str2 as String
Dim I as Integer

I = MCard.Init()
Debug.Print "INIT ";CSTR(I)'I = Success/Error
STR2 = "FILE_01.TXT"
'Assign number 1 to FILE_01.TXT and open as read-only
I = MCard.File(1).Open(STR2, 0)
Debug.Print "FILE OPEN STATUS: ";CSTR(I)'I = Success/Error
```

.CLOSE

Closes a currently open file stream

```
Variable = MCard.File(fileNum).Close() As Boolean
```

Parameter/s:

- **fileNum**; File Stream Number to close.

Returns;

- **True**: File Operation Successful
- **False**: Error

The following code is an example of how to initialise the external memory card, open a file and assign a number to it, and closing the file using the previously assigned number.

Code:

```
Public Sub Main()  
  
Dim Str1, Str2 as String  
Dim I as Integer  
I = MCard.Init()  
Debug.Print "INIT "; Cstr(I)  
STR2 = "FILE_01.TXT"  
I = MCard.File(1).Open(STR2, 0)  
Debug.Print "OPEN "; Cstr(I)  
I = MCard.File(1).Close()  
Debug.Print "CLOSE "; Cstr(I)  
  
End Sub
```

.READ

Start reading a specified number of bytes of data from the file stream

```
Variable = MCard.File(fileNum).Read(ByRef StrVar As String, ByVal BytesToRead As Integer) As Boolean
```

Parameter/s:

- **fileNum**; File Stream Number to read from.
- **StrVar**; String variable to read the bytes into
- **BytesToRead**; Specified number of bytes to read into StrVar

Returns;

- **True**: Read Successful
- **False**: Error

Code:

```
Public Sub Main()  
Dim Str1, Str2 as String  
Dim I as Integer  
  
I = MCard.Init()  
Debug.Print "INIT "; CSTR(I)  
STR1 = "FILE_01.TXT"  
I = MCard.File(1).Open(STR1, 0)  
Debug.Print "OPEN "; CSTR(I)  
I = MCard.File(1).Read(STR2, 10)  
Debug.Print "String Read: "; CSTR(I); " <"; STR2; ">"  
  
End Sub
```

.WRITE

Perform write operations to specified file stream

```
Variable = MCard.File(fileName).Write(ByRef StrVar As String, ByVal BytesToWrite As Integer) As Boolean
```

Parameter/s:

- **fileNum**; File Stream Number to Write to.
- **StrVar**; String variable containing characters to write to the file.
- **BytesToWrite**; Number of bytes from StrVar to write to the file.

Returns:

- **True**: Write Successful
- **False**: Error

Code:

```
Public Sub Main()  
  
Dim Str1, Str2 As String  
Dim I As Integer  
I = MCard.Init()  
Debug.Print "INIT ";CSTR(I)  
STR1 = "FILE_NEW.TXT"  
I = MCard.File(1).Open(STR1, 2) 'Open as read & write  
Debug.Print "OPEN ";CSTR(I) 'I = Success/Error  
STR2 = "12345"  
I = MCard.File(1).Write (STR2, 3) 'write "123" to file  
Debug.Print "WRITE ";CSTR(I) 'I = Success/Error  
  
End Sub
```

.SEEK

Sets cursor position to a specified index within the file stream

```
Variable = MCard.File(fileName).Seek(Index) As Boolean
```

Parameter/s:

- **fileNum**; File Stream Number for Seek Operation
- **Index**: Position within the file stream to set the cursor

Returns:

- **True**: Successful
- **False**: Error

Code:

```
Dim I As Integer  
Dim STR1 As String  
  
I = MCard.Init()  
STR1 = "FILE_01.TXT"  
I = MCard.File(1).Open(STR1, 0)  
I = MCard.File(1).Seek(5) 'Set Cursor Position to 5  
  
Debug.Print "SEEK 5 ";CSTR(I)  
I = MCard.File(1).GetPos()  
Debug.Print "GETPOS ";CSTR(I) 'I = 5
```

.GETPOS

Returns current position of cursor in a specified file stream

Variable = **MCard.File**(fileNum).**GetPos**() *As Boolean*

Parameter/s:

- **fileNum**; File Stream Number in which to get the cursor position.

Returns;

- **Cursor Position**

Code:

```
Public Sub Main()  
  
Dim Str1, Str2 as String  
Dim I as Integer  
  
I = MCard.Init()  
Debug.Print "INIT ";Cstr(I)  
STR2 = "FILE_01.TXT"  
I = MCard.File(1).Open(STR2, 0)  
Debug.Print "OPEN ";Cstr(I)  
I = MCard.File(1).GetPos()  
Debug.Print " GETPOS ";Cstr(I)  
  
End Sub
```

.ISEOF

Checks if cursor has reached the end of the specified file stream

Variable = **MCard.File**(fileNum).**IsEOF**() *As Boolean*

Parameter/s:

- **fileNum**; File Stream Number in which to check the cursor position.

Returns:

- **True** if cursor is at the end of the file stream;
- **False** if cursor is not at the end of the file stream.

.REWIND

Sets cursor position back to the beginning of the file stream

Variable = **MCard.File**(fileNum).**Rewind**() *As Boolean*

Parameter/s:

- **fileNum**; File Stream Number in which to set the cursor position back to the beginning of the file stream.

Returns:

- **True**: Successful
- **False**: Error

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.