# C# Software Development Kit

## How to Use StarIO in C#

## Thermal Line Mode Printing

This SDK contains a C# Visual Studio 2008 project for use on Microsoft Windows 7, 8, 8.1 & 10. Modern UI and Windows RT are not supported.

Works with these Printer Model Series:

| | |
|---|---|
| FVP10 | (Ver.1.0 or later) |
| HSP7000 | (Ver.1.0 or later) |
| TSP650 | (Ver.2.0 or later) |
| TSP650II | (Ver.1.0 or later) |
| TSP700II | (Ver.2.0 or later) |
| TSP800II | (Ver.1.0 or later) |
| TUP500 | (Ver.1.0 or later) |
| TUP900 | (Ver.1.2 or later) |
| SP700 | (Ver.1.0 or later) |
| POP10 | (Ver1.0 or later) |

Works with these DK-AirCash Model Series:

| | |
|---|---|
| SAC10 | (Ver.1.0 or later) |

Supported Interfaces:

- Serial
- Parallel
- USB
- Ethernet
- Bluetooth

Functions Include:

- Print Sample Receipt (EN and JP)
- 1D Barcodes, Code93 & I2of5
- 2D Barcodes, QR Code & PDF417
- Drawer Kick
- Check Block (ETB)
- Text Formatting
- Getting Status
- DK-AirCash
- Read Barcodes

Requirements: Visual Studio 2008 or later and .NET Framework 2.0 or later.
NOTE:

- This sample project contains StarIO components from StarIO Version 2.2.1.0.
  Details of StarIO(Restrictions, Precautions) are found in the manuals located here:
  English : <..\StarIO_help\en\StarIO\index.htm>
  Japanese : <..\StarIO_help\ja\StarIO\index.htm>

- This sample project provides source code which tells how to use StarIO components. Executable for 32/64-bit can be built by this project.

- Executable files for 64-bit cannot be executed in a 32-bit environment.

- When you open this project, execute Visual Studio as an administrator before opening the sln file. You can achieve this by right-clicking on the Visual Studio 2008 icon and clicking “Run as administrator” in the menu displayed.

# Table of Contents

# About this Manual

This manual is designed to help you understand StarIO and how to build a C# application to interact with Star Micronics Thermal Line Mode Printers. It is important to understand the basics of the C# language and the .NET framework. Although this SDK is for the programming language C#, there are other SDKs available at our website in the Developers section. Check the Developers section of our site for the newest SDKs, technical documentation, FAQs, and much more additional resources.

**Key Legend:**

| | | |
|---|---|---|
| *Warning* | ⚠ | Explains potential issues |
| *Avoid Doing This* | 🚫 | Explains things not to do |
| *Note* | 💭 | Provides important information and tips |

CAUTION:

- The information in this manual is subject to change without notice.
- STAR MICRONICS CO., LTD. has taken every measure to provide accurate information, but assumes no liability for errors or omissions.
- STAR MICRONICS CO., LTD. is not liable for any damages resulting from the use of information contained in this manual.
- Reproduction in whole or in part is prohibited.

# Star Printer Compatibility Chart

| Star Printer | SampleReceipt | SampleReceipt (Error Recovery) | Check Block - ETB | Monitoring Status | Open Cash Drawer | 1D Barcodes | 2D Barcodes | Code Page | Font | Feed | Text Formatting | DK-AirCash | Read Barcodes |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| FVP10 | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | | |
| TSP650 | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | | ✓ | ✓ | ✓ | ✓ | | |
| TSP650II | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | | |
| TSP700II | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | | |
| TSP800II | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | | |
| TUP500 | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | | |
| TUP900 | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ *1 | ✓ | | ✓ | ✓ | | |
| SP700 | ✓ | ✓ | ✓ | ✓ | ✓ | | | ✓ | *2 | ✓ | ✓ | | |
| HSP7000 | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | | |
| SAC10 | | | | ✓ | ✓ | | | | | | | ✓ | |
| POP10 | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | | ✓ |

*1 QR Code is not supported. PDF417 is available from F/W ver.3.1 or later.

*2 Sample code of this SDK is for Thermal Printer command.
When use SP700, please refer "Star Impact Printer Command Specifications".

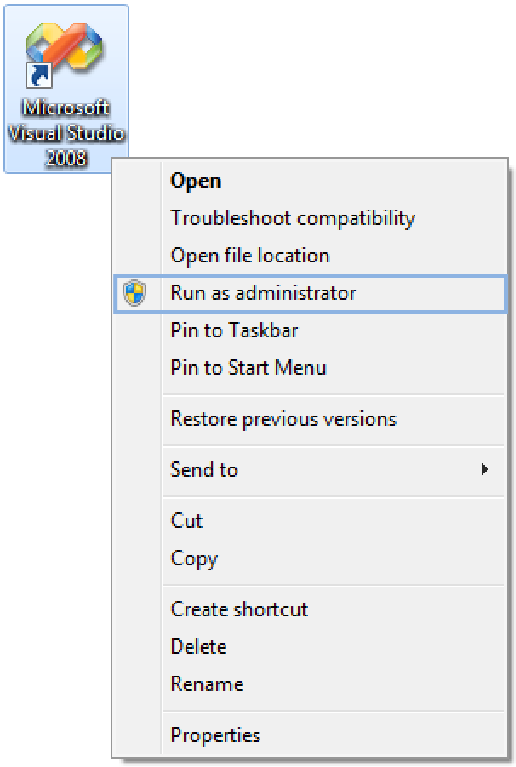
# How to compile and run the C# SDK

**This section will explain:**

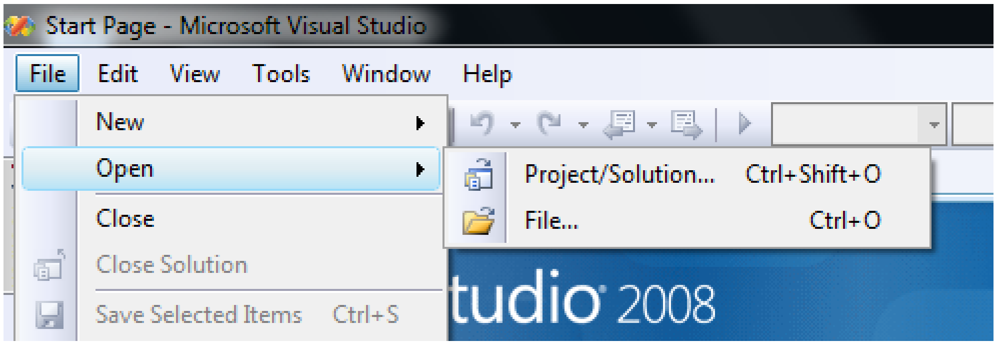
1. How to open the Visual Studio 2008 C# SDK project.
2. Compiling the project.
3. Running the project.

**How to open the Visual Studio 2008 C# SDK project:**

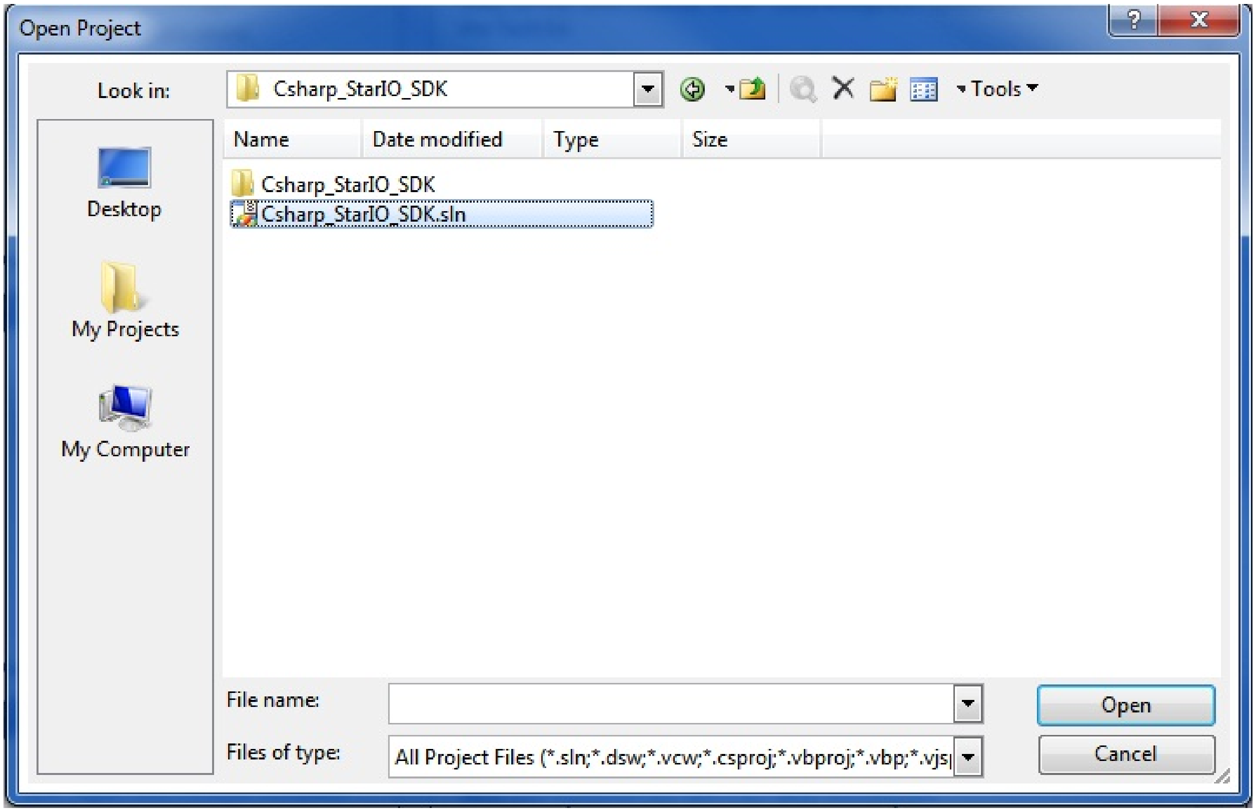
Right click on Visual Studio 2008 icon and click “Run as administrator”.



Once Visual Studio is running, click on File->Open->Project/Solution…


Start Page - Microsoft Visual Studio
File
Edit
View
Tools
Window
Help
New
Open
Close
Close Solution
Save Selected Items Ctrl+S
Project/Solution... Ctrl+Shift+O
File... Ctrl+O
tudio 2008

Navigate to the C# SDK folder titled “Csharp_StarIO_SDK” and click on the “sln” file titled "Csharp_StarIO_SDK.sln” to open the SDK project.

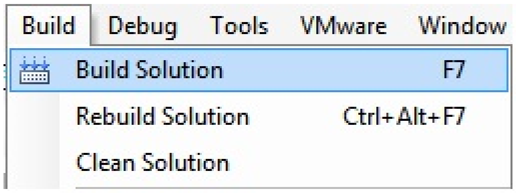


Look in: Csharp_StarIO_SDK

Files of type: All Project Files (*.sln;*.dsw;*.vcw;*.csproj;*.vbproj;*.vbp;*.vjs|

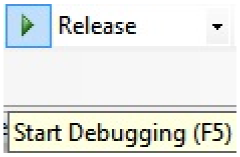
**Compiling the project:**

Click on the menu item “Build” and then click “Build Solution” or hit F7



**Running the project:**

Click on the green arrow to “Start Debugging” or hit F5



# Using the SDK with Star Micronics Printers

Please make sure you have a compatible Star Micronics thermal line mode printer model.

**Port Name and Interface Relation:**

StarIO uses specific port names to identify what port will be used. These are very important to understand because not following the naming convention correctly will fail to communicate with the printer.

| Interface | Port Name | Port Settings |
|---|---|---|
| **Serial** | COM*n* | 9600,n,8,1,h |
| **Parallel** | LPT*n* | N/A |
| **USB (Vendor Class)** | usbven: | N/A |
| **USB (Printer Class)** | usbprn:”Queue Name” | N/A |
| **Ethernet (TCP/IP)** | tcp:”IP Address” | N/A |
| **Bluetooth** | BT: COM*n* | N/A |

---

If using ***USB*** interface and the printer is in ***printer class mode***, once you install the printer drivers, you will have a queue name for the printer.

If Printer Queue Name = Star TSP700II (TSP743II)

Then we would put ***"usbprn:Star TSP700II (TSP743II)"*** as the Port Name.

If using ***USB*** interface and the printer is in ***vendor class mode***,
a Port number is not required. Just put “usbven:” as the Port Name.

"LPTn" n = your port number (1, 2, 3, 4 etc)

"COMn" n = your port number (1, 2, 3, 4 etc)

"tcp:192.168.222.244" Enter TCP IP Address of the Ethernet device.

"BT:COMn" n = your Bluetooth virtual serial port number (1, 2, 3, 4 etc)

---

# Overview of how the C# SDK is designed

This overview will touch briefly on key components of the SDK and how to find them.

Focus on the file “SMForm.cs” which contains all the business logic and StarIO commands.

The project has a DLL file called StarIOPort.dll which is a library for StarIO commands and communication with the printer that can be used with any C# application. Include this file into your application in order to expose StarIO and its methods to your program.

If you wish to find out what command is being issued when clicking on a function button of the SDK, simply double click the file in the file named SMForm.cs in the Solution Explorer. This will show you the general user interface of the C# SDK application. You can then double click on any button to find out what function is attached to that button. Once you double click it, Visual Studio 2008 will automatically jump to the function that is called when clicking that particular button.

Look through the code for comments and you will see how easily it is broken down step by step for you. Almost all functions in this SDK have comments above to explain what the function and code is doing.

The file “codePage.cs” is a Code Page Class that holds an array of strings which contain code pages compatible with Star Micronics Printers. The main form “SMForm” uses this class to determine the code pages on your computer are compatible with code pages available on the printer. If code pages do not exist on your computer that the printer supports, it will not be added to the code page combo box. Currently only SBCS Code Page is covered in this SDK.

There is a function called printToPrinter in “SMForm.cs” which is a great example on how to print to Star Micronics Printers. Start here if you just want to know how to open the port, write to the port, and close the port.

A sample receipt is also included in the file “SMForm.cs”. A sample receipt is also included in the file “SMForm.cs”, “SampleReceipt_Thermal_Form.cs” and “SampleReceipt_DotMatrix_Form.cs”. This is a great start to learning the basic StarIO commands, text formatting, and more.

The function timerGetStatus_Tick contains sample code on how to read status from a Star Micronics Printer. Review this code block to see how to get status through StarIO.

The file “lang.cs” is for future translations of this SDK, disregard this file.

# StarIO – (StarIOPort.dll & StarIO.dll)

## How to include StarIO into your project:

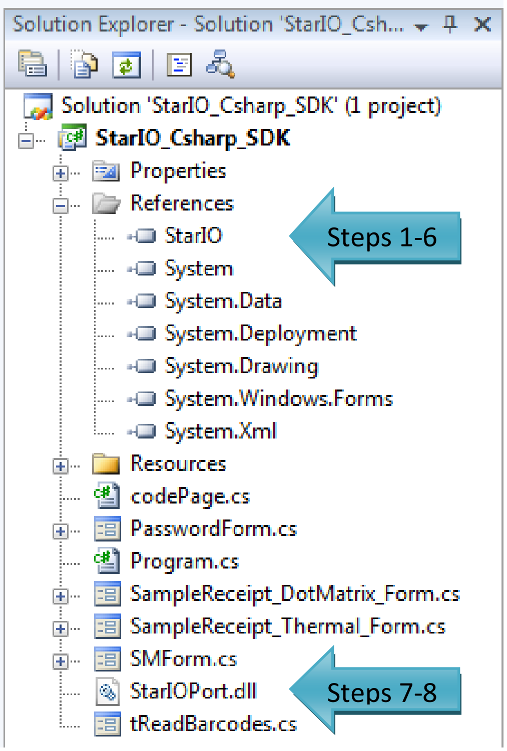
The file StarIOPort.dll is a dynamically linking library that you can include into your C# projects to expose StarIO methods. The file StarIO.dll is a .NET wrapper for StarIOPort.dll.

To include this DLL into your project:

1. Right-Click on the Project in Solution Explorer
2. Choose Add Reference
3. Click the “Browse” tab
4. Click the “Dependencies” folder attached with this SDK and select the folder “x86” or “x64” based on your target platform.
5. Click add existing item, then “StarIO.dll”
6. In the Solution Explorer, select StarIO under references. Set the Copy to Output - Property of StarIO.dll to Copy always.
7. StarIOPort.dll should reside in the same folder as StarIO.dll to work. To make sure your project copies the DLLs to your output debug folder, right click on the project name again and select “Add” -> “Existing Item…” then select show all files and select StarIOPort.dll
8. To ensure output debug directory is coping the DLL when you build, click on the item for StarIOPort.dll and under properties, select “Copy to Output Directory” and set to copy always.
9. To expose StarIO, add

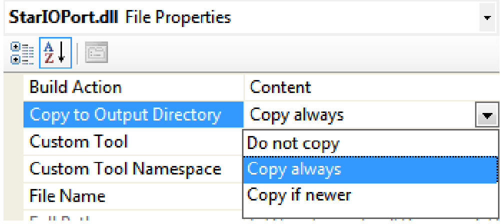
   “using StarMicronics.StarIO;”

   at the top of your main code.
10. Now you can access all of StarIO’s methods!


Solution Explorer - Solution 'StarIO_Csh...

Solution 'StarIO_Csharp_SDK' (1 project)
- **StarIO_Csharp_SDK**
  - Properties
  - References
    - StarIO
    - System
    - System.Data
    - System.Deployment
    - System.Drawing
    - System.Windows.Forms
    - System.Xml
  - Resources
  - codePage.cs
  - PasswordForm.cs
  - Program.cs
  - SampleReceipt_DotMatrix_Form.cs
  - SampleReceipt_Thermal_Form.cs
  - SMForm.cs
  - StarIOPort.dll
  - tReadBarcodes.cs

*StarIO being used as a Reference*


**StarIOPort.dll** File Properties

| | |
|---|---|
| Build Action | Content |
| Copy to Output Directory | Copy always |
| Custom Tool | Do not copy |
| Custom Tool Namespace | Copy always |
| File Name | Copy if newer |

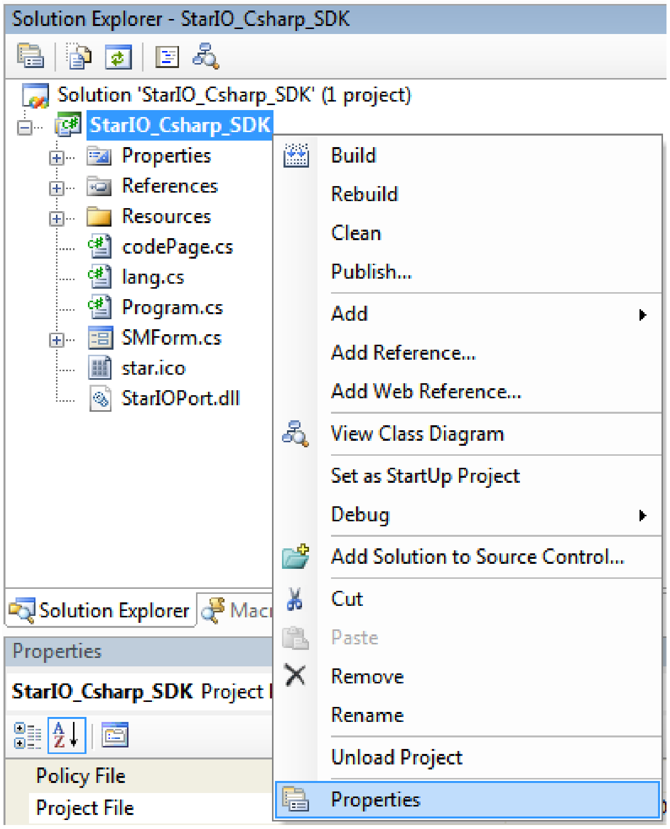
*Set StarIOPort.dll to always copy*

**WARNING:** Make sure StarIO.dll and StarIOPort.dll are in the same directory as each other. StarIO.dll links itself to StarIOPort.dll by looking in the same folder that the StarIO.dll is in.

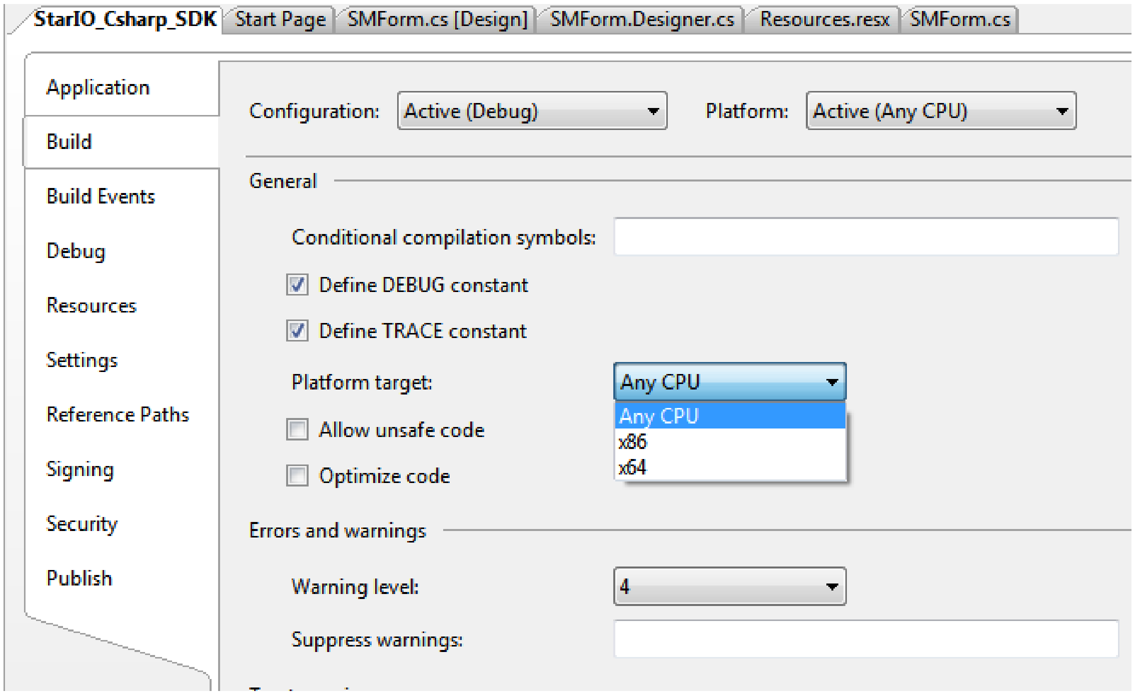
## Configuring your project to x64 or x86 with StarIO:

Compiling your project with the correct StarIOPort.DLL is very important to get the maximum speed from your CPU. Your main two choices are 32-bit and 64-bit Operating systems. C# uses Microsoft's CLR to translate your code into a functioning program and this comes in two flavors which is 32-bit & 64-bit. WoW64, which means **W**indows 32-bit **O**n **W**indows 64-bit, allows you to make a 32-bit application with 32-bit binaries to run on either 32-bit or 64-bit. ***We recommend setting your project to x86 to run on both 32 and 64 OS***. If you wish to make a 100% 64-bit project then use StarIO dlls under the folder "x64" for 64-bit libraries of StarIO.

1. Right Click on your C# project name and click Properties.



2. Click on the "Build" tab on the left and find "Target Platform".



Configuration: Active (Debug)
Platform: Active (Any CPU)

General

Conditional compilation symbols:

☑ Define DEBUG constant

☑ Define TRACE constant

Platform target: Any CPU
- Any CPU
- x86
- x64

☐ Allow unsafe code

☐ Optimize code

Errors and warnings

Warning level: 4

Suppress warnings:

3. Select the platform you wish to compile for (Any CPU, **x86**_(recommended)_, x64).
4. Now that you have selected your platform, you must add 32-bit or 64-bit StarIO DLL.

**WARNING**: If you set your project to "Any Cpu" and use 32-bit StarIO libraries, you will find that Windows x64 will not be able to run it. To fix this, setup your project as x86 and use only x86 StarIO which will run on both systems. If you are developing purely for the x64 environment, then set to x64 with x64 StarIO.

5. Once you have selected your target platform and saved your changes, follow the directions in "How to include StarIO to your project" on how to add a reference to StarIO 32-bit or 64-bit DLL binaries into your project with the corresponding library.

---

**StarIO Methods Overview:**

**Class Variables** include portName (string), portSettings (string), and Timeout (int).

These 3 variables will be "read only" if accessed directly. To assign them use GetPort(portName,portSettings,Timeout); which will allow you to pass in variables to this methods which then assigns the 3 class variables with values.

**portName** is what you will be using to specify the port of communication to the printer.

*Ex. "usbven:" "usbprn:TSP650" "tcp:192.168.1.2" "COM4" "LPT1" "BT:COM3"*

**portSettings** is what you will use for configuring Serial connections correctly.

*Ex. "9600,n,8,1,h"*

The following are the acceptable inputs from left to right:

baud: 38400, 19200, 9600, 4800, 2400
parity: n, e, o
data-bits: 8, 7
stop-bits: 1
flow-ctrl: n, h

**Timeout** is a millisecond timeout controlled internally and is used for communication in the APIs (this parameter guarantees that all of the below APIs will complete in a bounded amount of time, but does NOT guarantee the exact timeout length)

**GetPort**

```
public static StarIOPort GetPort(String portName, String portSettings, Int TimeoutMillis)
                                                                throws StarIOPortException
```

GetPort is what you will be using to "open" the port to the device. Using one of the valid inputs for portName and portSettings as mentioned previously before this, you can pass your connection string into the StarIO class so that it will correctly set its private variables.

The following would be an actual usage of GetPort in C#:

```
IPort port = null;

    Try
    {
        port = StarMicronics.StarIO.Factory.I.GetPort(portName, portSettings, 10000);
    }
    Catch(PortException)
    {
        //There was an error opening the port
    }
```

`IPort` is a part of StarIO and this will allow you to create a "port" handle. The above example shows the port being created and set to null then being assigned the actual port hook on the following line that contains GetPort.

Always use a Try, Catch when using **GetPort**. If the port cannot be opened because of connection problems, your program will crash unless you use a Try, Catch like the above example.

### ReadPort

```
public int ReadPort(Byte[] readBuffer, Int offset, Int size)
                                                throws StarIOPortException
```

This method reads data from the device. Only use this if you really need to read raw bytes from the device.

**Do not use this method to try and read raw status.**
Use GetOnlineStatus or GetParsedStatus for getting status.

**Parameters:**

`readbuffer` – A Byte Array buffer into which data is read.

`offset` - specifies where to begin writing data into the readBuffer[]

`size` – Total number of bytes to read.

**Returns:**

The number of bytes that were actually read. Under some interface types, this function will succeed even when no data was read in. Your application should call this function a limited number of times until the expected data has been read in or until an application determined retry threshold has been reached.

**Throws:**

`StarIOPortException` - when a communication failure occurs

### ReleasePort

```
public static void ReleasePort(StarIOPort port)
```

This function closes a connection to the port specified.

**Parameters:**

port - StarIOPort type representing a previously initialized port.

Always release (close) ports that you get (open).
Leaving a port open will cause future calls to open the port to fail.

**WritePort**

```
public int WritePort(Byte[] writeBuffer, Int offset, Int size)
                                                        throws StarIOPortException
```

This method writes data to the device. Use this to print to the printer, send commands to the DK-AirCash, etc. The following is an example of how to use this method:

*Please keep in mind this is the simplest way to send data to the device.*
*The C# SDK has code in printToPrinter that is more complex than this but that code block will show you how to verify data transmission to the device whereas this code is just dumping it:*

```
//Set a byte array to send to the device
//command = { A, B, C, D, Feed 3mm, Full Cut}
Byte[] command = {0x41, 0x42, 0x43, 0x44, 0x1B, 0x7A, 0x00, 0x1B, 0x64, 0x02};

Uint bytesWritten = 0;

Try
{
    While(command.Length > bytesWritten)
    {
      bytesWritten += port.WritePort(command, bytesWritten,(unit)command.Length - bytesWritten);
    }
}
Catch(PortException)
{
  //There was an error writing to the port
}
```

Remember to use a Try, Catch for safe programming practices.

**Parameters:**

`writeBuffer` - Contains the output data in a byte array.

`offset` - Specifies where to begin pulling data from writeBuffer .

`size` - Number of bytes to write.

**Returns:**

The number of bytes that were actually written. Under some interface types, this function will succeed even when no data was written out. Your application should call this function a limited number of times until all the data has been written out or until an application determined retry threshold has been reached.

**Throws:**

`StarIOPortException` - when a communication failure occurs

### ETB

### BeginCheckedBlock

public StarPrinterStatus **BeginCheckedBlock**()
throws StarIOPortException

This method initiates a checked block printing operation and returns the device's detailed status.

**Returns:**

StarPrinterStatus structure giving the current device status - don't bother printing if the printer is offline

**Throws:**

`StarIOPortException` - when a communication failure occurs

### EndCheckedBlock

public StarPrinterStatus **EndCheckedBlock**()
throws StarIOPortException

This method ends a checked block printing operation and returns the device's detailed status. This function does not return until either the printer has successfully printed all data or has gone offline in error. If the StarPrinterStatus structure indicates that the printer is online upon return than all data was successfully printed.

**Returns:**

StarPrinterStatus structure giving the current device status - if it's offline then printing failed

**Throws:**

`StarIOPortException` - when a communication failure occurs

Here is an **example** usage of **BeginCheckedBlock** and **EndCheckedBlock** methods:

```
Iport port;
StarPrinterStatus printerStatus;

// Port Open
Try
{
    port = Factory.I.GetPort(portName, portSettings, 10000);
}
Catch (PortException)
{
    //Fail: There was an error in opening the port
    return;
}

//Begin checked block
Try
{
    printerStatus = port.BeginCheckedBlock();
}
Catch (PortException)
{
    //Fail: Cannot begin checked block!
    Factory.I.ReleasePort(port);
    return;
}

//Check status
If (printerStatus.Offline == true)
{
    //Fail: The device is offline!
    Factory.I.ReleasePort(port);
    return;
}

Byte[] command = {0x41, 0x42, 0x43, 0x44, 0x0a};
Uint bytesWritten = 0;

//Send data
Try
{
     While(command.Length > bytesWritten)
     {
       bytesWritten += port.WritePort(command, bytesWritten,(unit)command.Length - bytesWritten);
     }
}
Catch(PortException){
     //Fail: There was an error writing to the port
}

// End checked block
Try
{
    printerStatus = port.EndCheckedBlock();
}
Catch (PortException)
{
    //Fail: Cannot end checked block!
    Factory.I.ReleasePort(port);
    return;
}

If (this.sPrinterStatus.Offline == true)
{
    //Fail: The device is offline!
    Factory.I.ReleasePort(port);
    return;
}
//Success! The print data was successfully sent.

//Regardless, release the port if it is still open.
If (port != null)
    Factory.I.ReleasePort(port);
}
```

### ResetDevice

public void **ResetDevice**()
throws StarIOPortException

This method resets the device at the hardware level.

**Throws:**

`StarIOPortException` - when a communication failure occurs

### GetOnlineStatus

```
public boolean GetOnlineStatus()
                                                     throws StarIOPortException
```

This method returns a Boolean value if the device is online or offline.

**Returns:**

Boolean value: true = device is online false = device is offline

**Throws:**

`StarIOPortException` - when a communication failure occurs

An example of its usage:

```
Boolean onlineStatus = port.GetOnlineStatus();

If(onlineStatus){
  //Success! The device is online.
}
Else{
  //Fail! The device is offline.
}
```

### GetParsedStatus

```
public StarPrinterStatus GetParsedStatus ()
                                                     throws StarIOPortException
```

This method retrieves detailed status form the device with StarIO.

**Returns:**

StarPrinterStatus structure giving the current device status

**Throws:**

`StarIOPortException` - when a communication failure occurs

This method uses a class structure that is included with StarIO called StarPrinterStatus

This structure gives the printer's status in both boolean and binary form.

Create the StarPrinterStatus object in your project by doing the following:

```
StarPrinterStatus printerStatus = port.GetParsedStatus();

If(printerStatus.Offline == false)
{
     If(printerStatus.BlackMarkError == true){
          //There was a black mark error
     }

     If(printerStatus.CompulsionSwitch == true){
          //Cash drawer is open
     }
     Else{
          //Cash drawer is closed
     }

}
Else{
     //If True, then the printer is offline.
}
```

There are lots of different statuses that are pulled when you initialize **StarPrinterStatus**.

**This is a list of statuses that are in the class structure StarPrinterStatus:**

CoverOpen returns a Boolean.

Offline returns a Boolean.

CompulsionSwitch returns a Boolean.

OverTemp returns a Boolean.

UnrecoverableError returns a Boolean.

CutterError returns a Boolean.

MechanicalError returns a Boolean.

HeadThermistorError returns a Boolean.

ReceiveBufferOverflow returns a Boolean.

PageModeCommadError returns a Boolean.

BlackMarkError returns a Boolean.

PresenterPaperJamError returns a Boolean.

HeadUpError returns a Boolean.

VoltageError returns a Boolean.

ReceiptBlackMarkDetection returns a Boolean.

ReceiptPaperEmpty returns a Boolean.

ReceiptPaperNearEmptyInner returns a Boolean.

ReceiptPaperNearEmptyOuter returns a Boolean.

PresenterPaperPresent returns a Boolean.

PeelerPaperPresent returns a Boolean.

StackerFull returns a Boolean.

slipTOF returns a Boolean.

slipCOF returns a Boolean.

slipBOF returns a Boolean.

validationPaperPresent returns a Boolean.

slipPaperPresent returns a Boolean.

ETBAvailable returns a Boolean.

ETBCounter returns a Byte.

PresenterState returns a Byte.

RawStatus returns a Byte[] array.

# Functionality

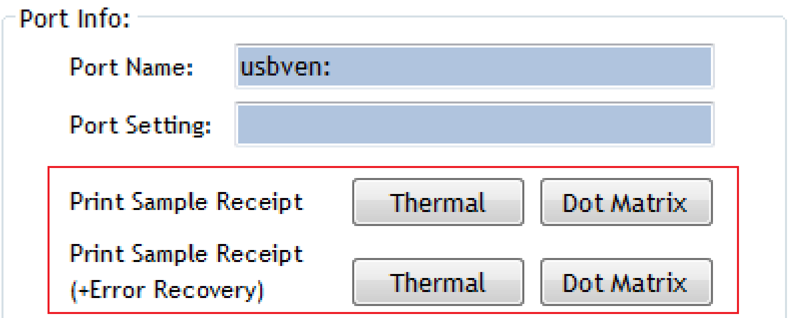
## StarIO Printer Commands

**All of these commands can be found in the [Star Thermal Line Mode Command Manual , Star Impact Printer Command Manual and StarPRNT Command Manual.]**

The C# SDK also has page and section references to this document for more information so please download that manual and study it if you need more detail on a specific command.

---

### Sample Receipts



Port Info:

Port Name: usbven:

Port Setting:

### Print Sample Receipt

Prints an example in English or Japanese. Reference the Text Formatting section of this document for instructions on modifying text.

### Print Sample Receipt (+ Error Recovery)

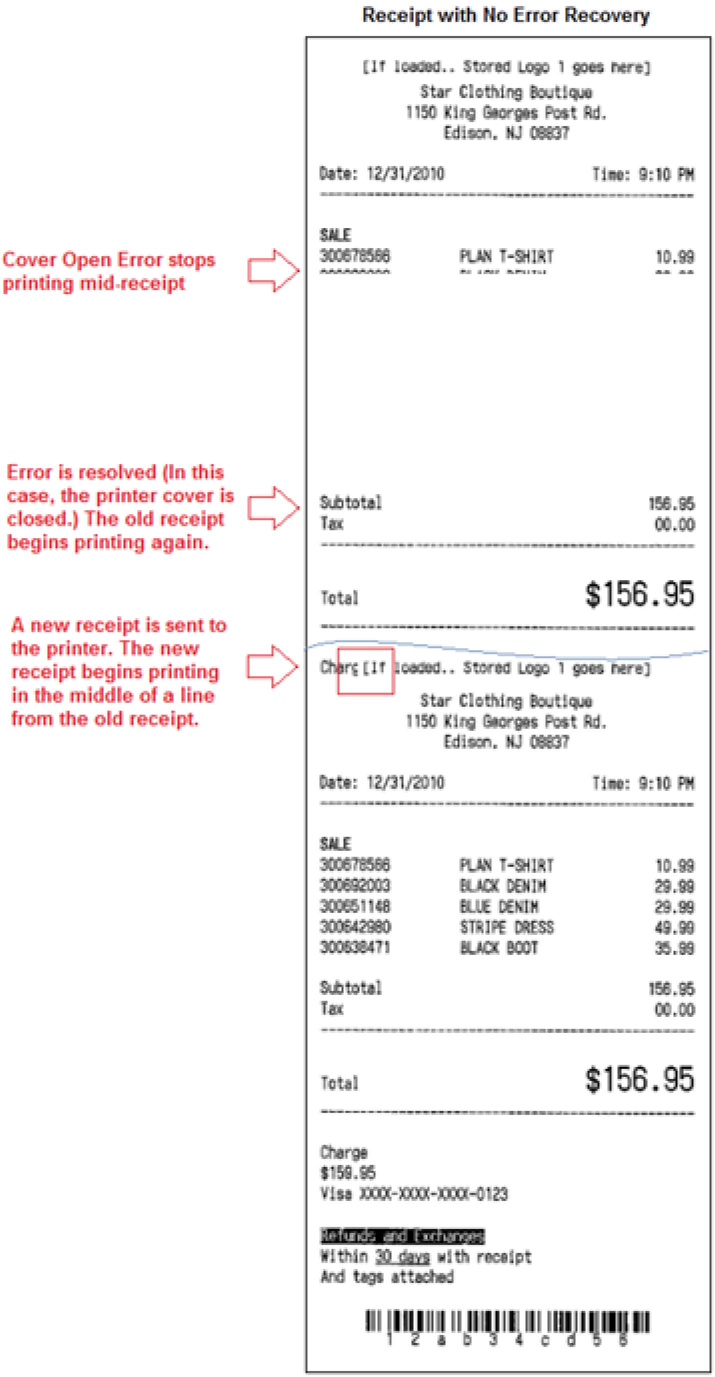
Enables the printer to completely recover from a failed print job due to an error occurring in the middle of printing. For example, the paper runs out in the middle of printing a receipt. The cashier loads another roll of paper and reprints it. Depending on where the old print job was cut short due to paper out, the new receipt may begin in the middle of the old one. Star provides two commands, one to be inserted before the print job and one after, to avoid this issue. A graphic is included on the following page to illustrate this functionality.

Command A: (Add to the head of the receipt data)

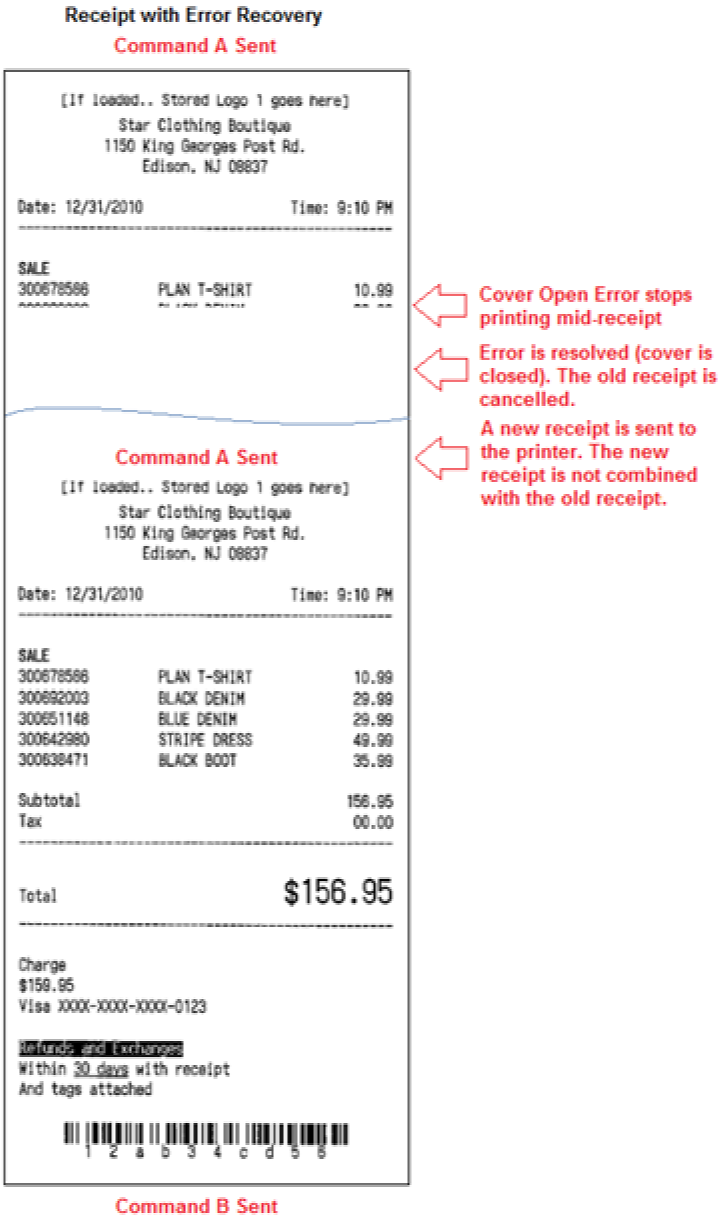
ESC GS ETX EOT NUL NUL ESC GS ETX ETX NUL NUL

Command B: (Add to the bottom of the receipt data)

ESC GS ETX EOT NUL NUL


Receipt with No Error Recovery
Cover Open Error stops printing mid-receipt
Error is resolved (In this case, the printer cover is closed.) The old receipt begins printing again.
A new receipt is sent to the printer. The new receipt begins printing in the middle of a line from the old receipt.

```
[If loaded.. Stored Logo 1 goes here]
Star Clothing Boutique
1150 King Georges Post Rd.
Edison, NJ 08837

Date: 12/31/2010                 Time: 9:10 PM
----------------------------------------------

SALE
300678586        PLAN T-SHIRT           10.99

Subtotal                               156.95
Tax                                     00.00
----------------------------------------------

Total                               $156.95
----------------------------------------------
Charg [If loaded.. Stored Logo 1 goes here]
Star Clothing Boutique
1150 King Georges Post Rd.
Edison, NJ 08837

Date: 12/31/2010                 Time: 9:10 PM
----------------------------------------------

SALE
300678586        PLAN T-SHIRT           10.99
300692003        BLACK DENIM            29.99
300651148        BLUE DENIM             29.99
300642980        STRIPE DRESS           49.99
300638471        BLACK BOOT             35.99

Subtotal                               156.95
Tax                                     00.00
----------------------------------------------

Total                               $156.95
----------------------------------------------

Charge
$159.95
Visa XXXX-XXXX-XXXX-0123

Refunds and Exchanges
Within 30 days with receipt
And tags attached

1 2 a b 3 4 c d 5 6
```


Receipt with Error Recovery
Command A Sent
Cover Open Error stops printing mid-receipt
Error is resolved (cover is closed). The old receipt is cancelled.
A new receipt is sent to the printer. The new receipt is not combined with the old receipt.
Command A Sent
Command B Sent

```
[If loaded.. Stored Logo 1 goes here]
Star Clothing Boutique
1150 King Georges Post Rd.
Edison, NJ 08837

Date: 12/31/2010                 Time: 9:10 PM
----------------------------------------------

SALE
300678586        PLAN T-SHIRT           10.99

[If loaded.. Stored Logo 1 goes here]
Star Clothing Boutique
1150 King Georges Post Rd.
Edison, NJ 08837

Date: 12/31/2010                 Time: 9:10 PM
----------------------------------------------

SALE
300678586        PLAN T-SHIRT           10.99
300692003        BLACK DENIM            29.99
300651148        BLUE DENIM             29.99
300642980        STRIPE DRESS           49.99
300638471        BLACK BOOT             35.99

Subtotal                               156.95
Tax                                     00.00
----------------------------------------------

Total                               $156.95
----------------------------------------------

Charge
$159.95
Visa XXXX-XXXX-XXXX-0123

Refunds and Exchanges
Within 30 days with receipt
And tags attached

1 2 a b 3 4 c d 5 6
```

**Important Notes about This Function:**

1. Text Formatting

   Command A will release the effect of text formatting commands (ex: bold, underline). Text formatting commands must be added between Command A and Command B for all print data.

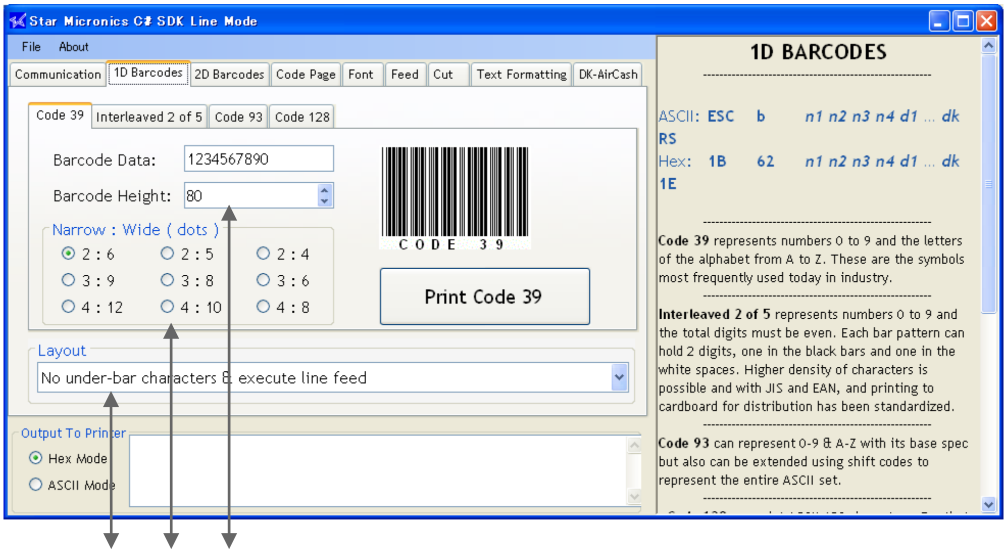
2. In the following conditions, the printer will discard all data if Command A is not sent:
   - i. Error occurs in sending print data with Command A
   - ii. Within ten seconds after an error occurs

(continued from Print Sample Receipt + Error Recovery Notes section)

3. Supported printers and minimum firmware version required:

| | |
|---|---|
| FVP10 | Ver 1.2 or later |
| TSP650 | Ver 3.0 or later |
| TSP650II | Ver 1.0 or later |
| TSP700II | Ver 3.0 or later |
| TSP800II | Ver 1.0 or later |
| TUP500 | Ver 3.0 or later |
| SP700 | Ver 3.0 or later |

## 1D Barcodes



Barcode Data: 1234567890

Barcode Height: 80

Narrow : Wide ( dots )
● 2 : 6 ○ 2 : 5 ○ 2 : 4
○ 3 : 9 ○ 3 : 8 ○ 3 : 6
○ 4 : 12 ○ 4 : 10 ○ 4 : 8

Layout: No under-bar characters & execute line feed

1D BARCODES

ASCII: ESC b n1 n2 n3 n4 d1 ... dk RS
Hex: 1B 62 n1 n2 n3 n4 d1 ... dk 1E

**Code 39** represents numbers 0 to 9 and the letters of the alphabet from A to Z. These are the symbols most frequently used today in industry.

**Interleaved 2 of 5** represents numbers 0 to 9 and the total digits must be even. Each bar pattern can hold 2 digits, one in the black bars and one in the white spaces. Higher density of characters is possible and with JIS and EAN, and printing to cardboard for distribution has been standardized.

**Code 93** can represent 0-9 & A-Z with its base spec but also can be extended using shift codes to represent the entire ASCII set.

ESC b n1 n2 n3 n4 d1 ... dk RS

n1 = Barcode Type
- 0 = UPC-E
- 1 = UPC-A
- 2 = JAN/EAN8
- 3 = JAN/EAN13
- 4 = Code39
- 5 = ITF
- 6 = Code128
- 7 = Code93
- 8 = NW-7

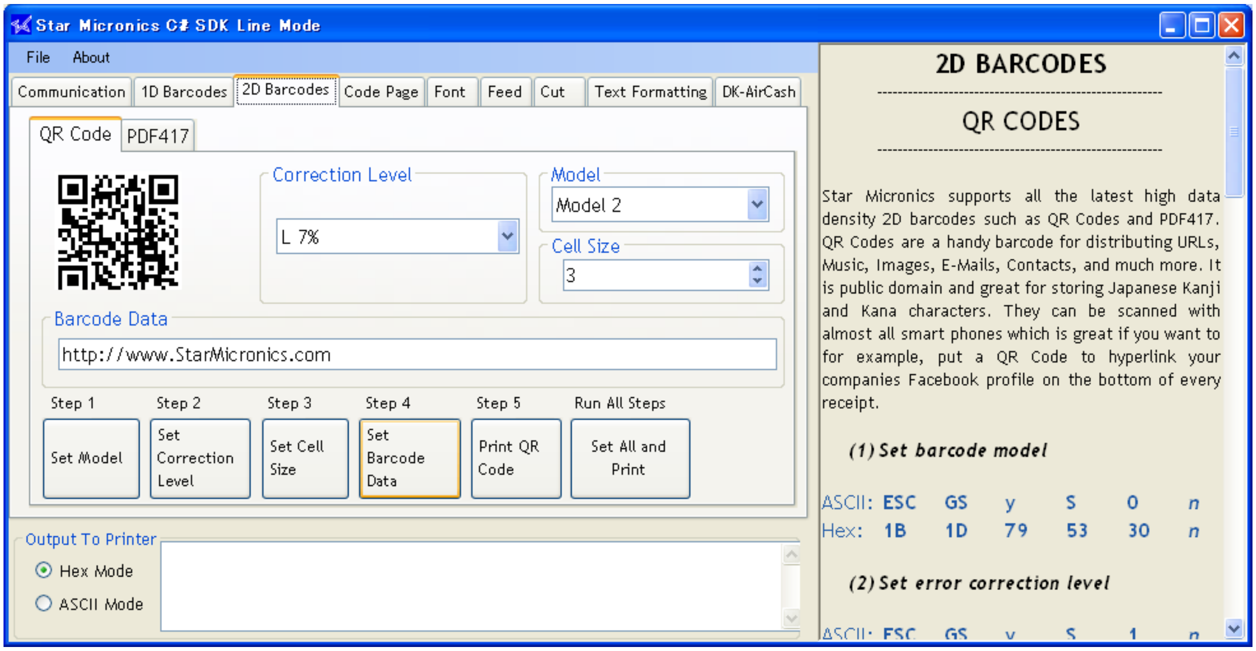
n2 = Under-bar character selection and added line feed selection
- 1 = No added under-bar characters & Executes line feed after printing barcode
- 2 = Adds under-bar characters & Executes line feed after printing barcode
- 3 = No added under-bar characters & doesn’t line feed after printing barcode
- 4 = Adds under-bar characters & doesn’t line feed after printing barcode

n3 = Barcode mode selection specifies the size of the narrow and wide barcode lines

n4 = Barcode height (dot count)

## 2D Barcodes

### QR Codes



Correction Level: L 7%

Model: Model 2

Cell Size: 3

Barcode Data: http://www.StarMicronics.com

**2D BARCODES**

QR CODES

Star Micronics supports all the latest high data density 2D barcodes such as QR Codes and PDF417. QR Codes are a handy barcode for distributing URLs, Music, Images, E-Mails, Contacts, and much more. It is public domain and great for storing Japanese Kanji and Kana characters. They can be scanned with almost all smart phones which is great if you want to for example, put a QR Code to hyperlink your companies Facebook profile on the bottom of every receipt.

***(1) Set barcode model***

| ASCII: | **ESC** | **GS** | **y** | **S** | **0** | *n* |
|---|---|---|---|---|---|---|
| Hex: | **1B** | **1D** | **79** | **53** | **30** | *n* |

***(2) Set error correction level***

ASCII: ESC GS y S 1 n

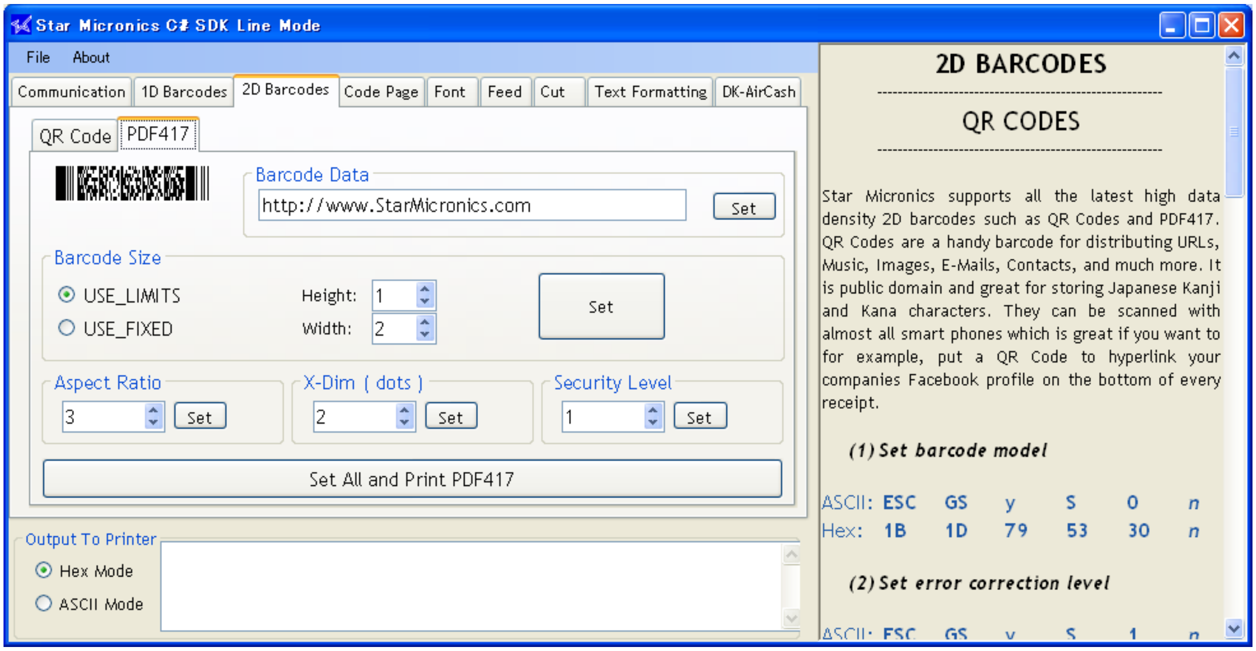
There are 5 commands below that are very important to printing a good QR code.

| | |
|---|---|
| (1) Set QR Code Model # | ESC GS y S 0 n |
| (2) Set QR Code Correction Level | ESC GS y S 1 n |
| (3) Set QR Code Cell Size | ESC GS y S 2 n |
| (4) Set QR Code Data | ESC GS y D 1 NUL nL nH d1…dk |
| (5) Print the QR Code | ESC GS y P |

Here is the order in which commands need to be sent to the printer for it to print the QR code.

QR model + QR Correction Level + QR Cell Size + QR Data + Print QR Code

## PDF417



Barcode Data: http://www.StarMicronics.com

Barcode Size: ● USE_LIMITS ○ USE_FIXED Height: 1 Width: 2

Aspect Ratio: 3

X-Dim ( dots ): 2

Security Level: 1

Output To Printer: ● Hex Mode ○ ASCII Mode

2D BARCODES

QR CODES

Star Micronics supports all the latest high data density 2D barcodes such as QR Codes and PDF417. QR Codes are a handy barcode for distributing URLs, Music, Images, E-Mails, Contacts, and much more. It is public domain and great for storing Japanese Kanji and Kana characters. They can be scanned with almost all smart phones which is great if you want to for example, put a QR Code to hyperlink your companies Facebook profile on the bottom of every receipt.

***(1) Set barcode model***

ASCII: ESC GS y S 0 *n*
Hex: 1B 1D 79 53 30 *n*

***(2) Set error correction level***

ASCII: ESC GS y S 1 *n*

Please visit page 3-120 in the Line Mode Spec Manual for more details on PDF417

(6) Set PDF417 barcode size ESC GS x S 0 n p1 p2
(7) Set PDF417 ECC (Security Level) ESC GS x S 1 n
(8) Set PDF417 module X direction size ESC GS x S 2 n
(9) Set PDF417 module aspect ratio ESC GS x S 3 n
(10) Set PDF417 barcode data ESC GS x D nL nH d1 d2 … dk
(11) Print PDF417 barcode ESC GS x P

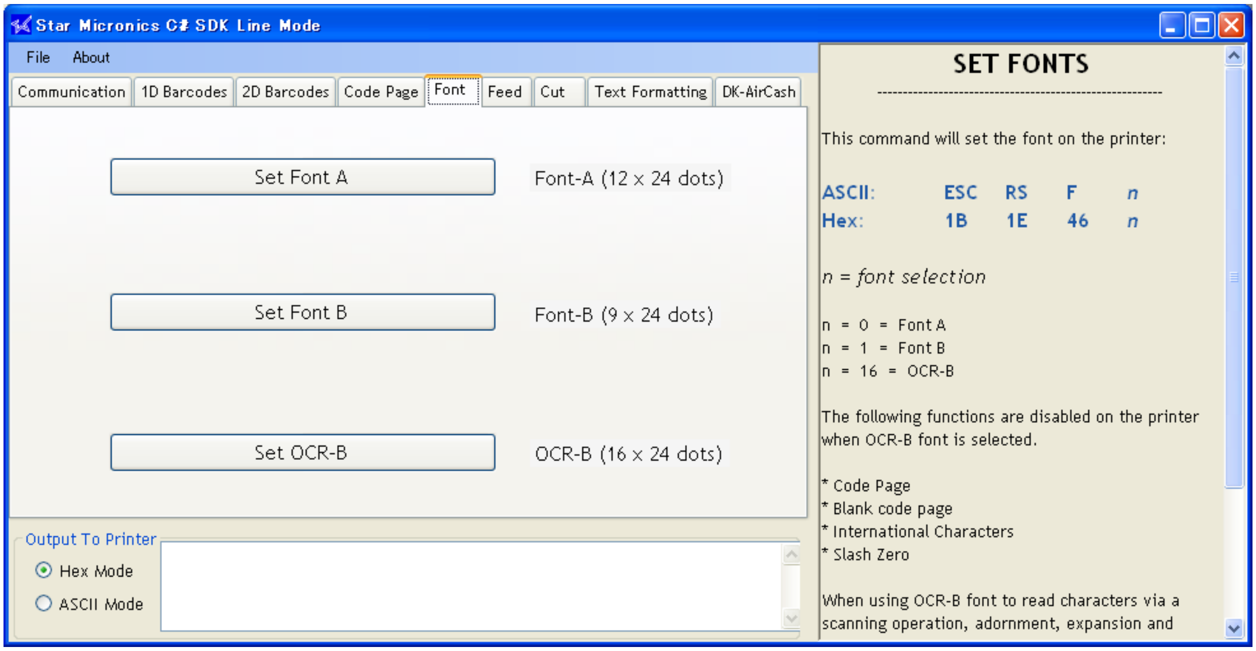
Here is the order in which commands need to be sent to the printer for it to print the PDF417.

PDF417 Size + PDF417 ECC + PDF417 X-dim + PDF417 Ratio + PDF417 Data + Print PDF417

## Change Font

Changing the font on the printer can be done with the following commands.

ESC RS F n n = 0 for A, 1 for B, 10 for OCR-B



**SET FONTS**

---

This command will set the font on the printer:

**ASCII: ESC RS F *n***
**Hex: 1B 1E 46 *n***

*n = font selection*

n = 0 = Font A
n = 1 = Font B
n = 16 = OCR-B

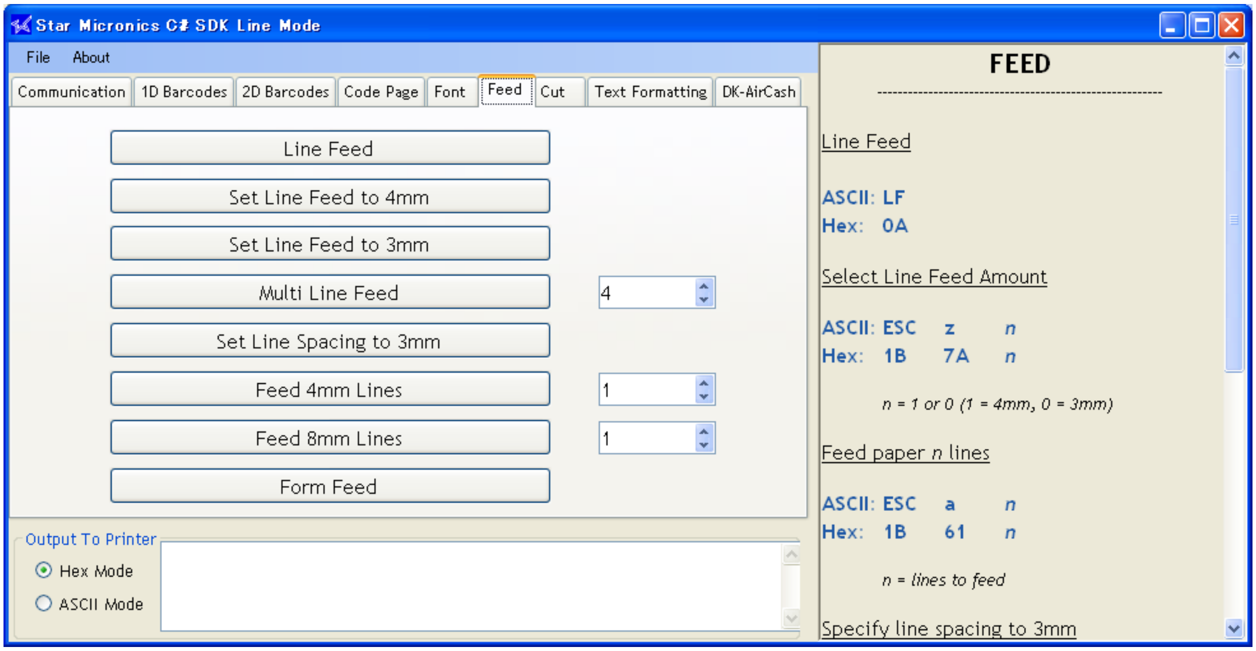
The following functions are disabled on the printer when OCR-B font is selected.

* Code Page
* Blank code page
* International Characters
* Slash Zero

When using OCR-B font to read characters via a scanning operation, adornment, expansion and

## Feed

The feed commands are very straight forward. Use LF for best results.



**FEED**

---

Line Feed

**ASCII: LF**
**Hex: 0A**

Select Line Feed Amount

**ASCII: ESC z *n***
**Hex: 1B 7A *n***

*n = 1 or 0 (1 = 4mm, 0 = 3mm)*

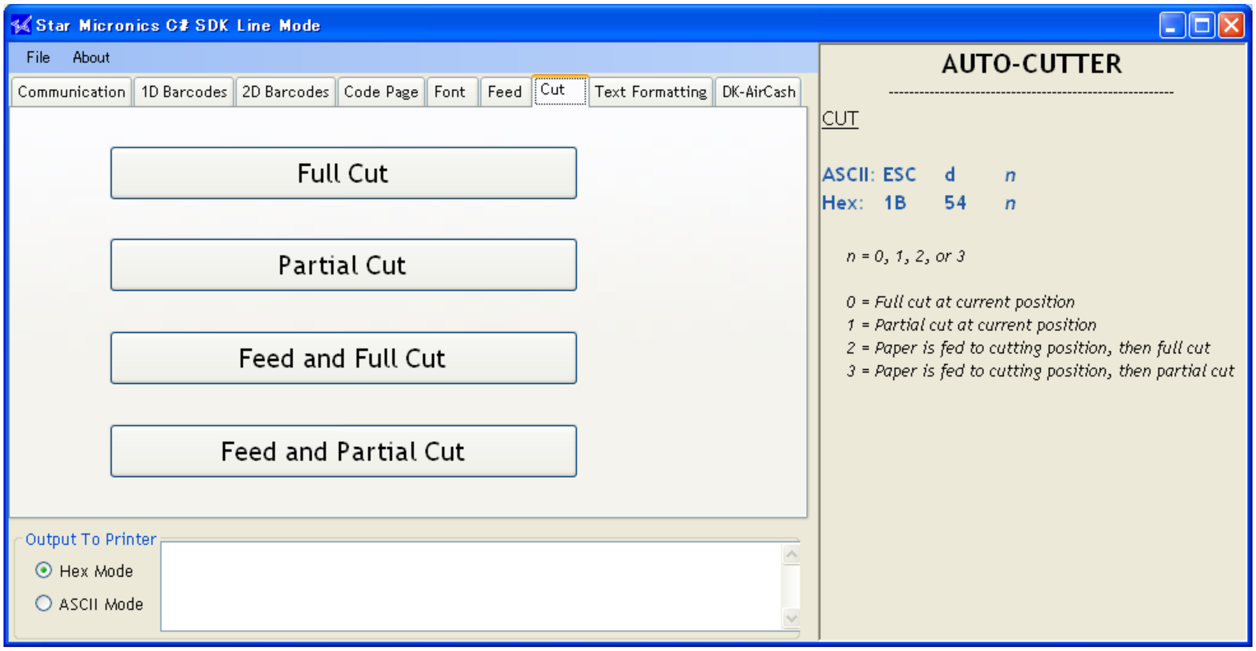
Feed paper *n* lines

**ASCII: ESC a *n***
**Hex: 1B 61 *n***

*n = lines to feed*

Specify line spacing to 3mm

## Cut



AUTO-CUTTER

------------------------------------------------------------

CUT

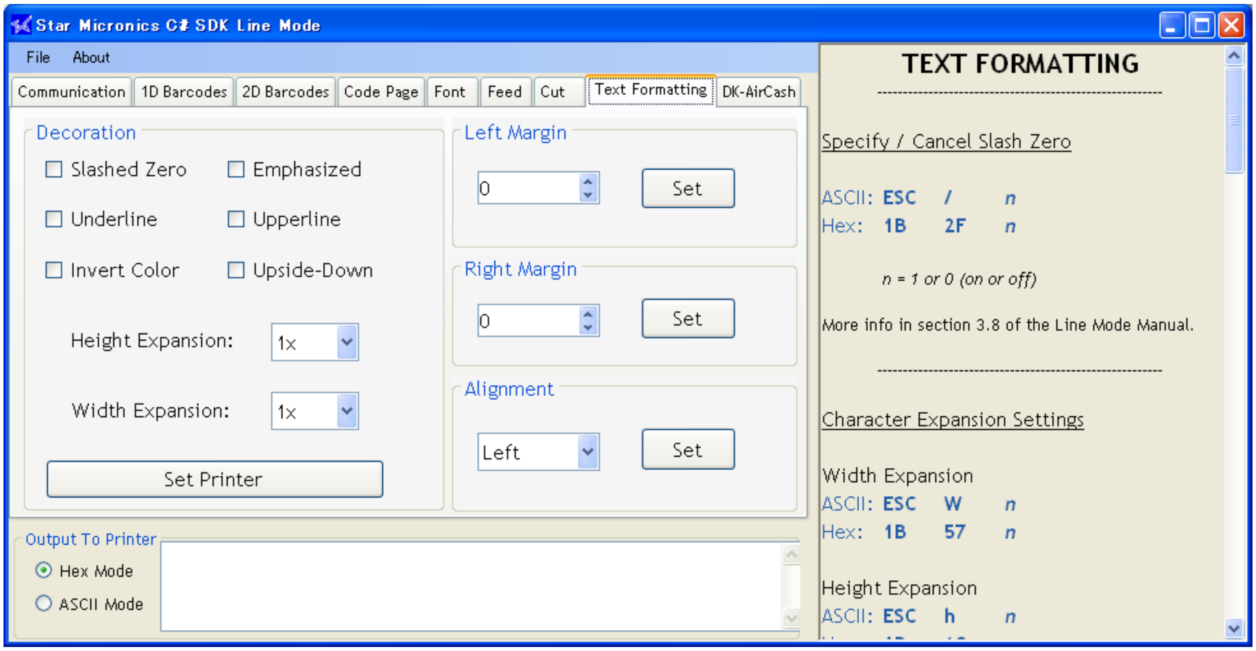
ASCII: ESC d *n*
Hex: 1B 54 *n*

*n = 0, 1, 2, or 3*

*0 = Full cut at current position*
*1 = Partial cut at current position*
*2 = Paper is fed to cutting position, then full cut*
*3 = Paper is fed to cutting position, then partial cut*

| | |
|---|---|
| Partial Cut | ESC d 1 or 3 |
| Full Cut | ESC d 0 or 2 |

## Text Formatting



TEXT FORMATTING

------------------------------------------------------------

Specify / Cancel Slash Zero

ASCII: ESC / *n*
Hex: 1B 2F *n*

*n = 1 or 0 (on or off)*

More info in section 3.8 of the Line Mode Manual.

------------------------------------------------------------

Character Expansion Settings

Width Expansion
ASCII: ESC W *n*
Hex: 1B 57 *n*

Height Expansion
ASCII: ESC h *n*

The following are all Text Decoration or formatting related.

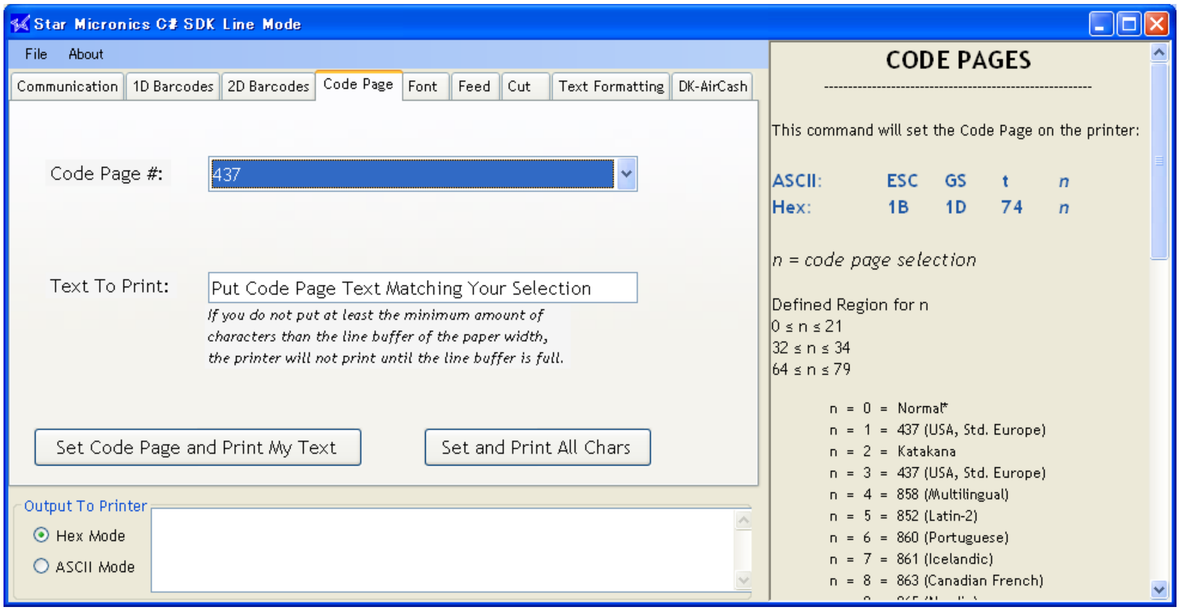
| | | |
|---|---|---|
| Slashed Zero | ESC / n | |
| Underline | ESC – n | |
| Upperline | ESC _ n | |
| Invert Color (B/W) | ESC 4 | |
| Emphasized (Bold) | ESC E = on | ESC F = off |
| Upside-Down | SI = Start | DC2 = off |
| Character expansion | | |
| Width | ESC W n | 0 ≤ n ≤ 5 |
| Height | ESC h n | 0 ≤ n ≤ 5 |
| Set Left Margin | ESC l n | 0 ≤ n ≤ 255 |
| Set Right Margin | ESC Q n | 0 ≤ n ≤ 255 |
| Alignment | | |
| Left | ESC GS a 0 | |
| Center | ESC GS a 1 | |
| Right | ESC GS a 2 | |

## Code Pages

The code pages supported by Star Printers can be found in the codePage.cs class but please be aware that not all code pages on the printer will be on your PC.

To set a code page on the printer:

ESC GS t n
n = The Code Page Selection Index



Code Page #: 437

Text To Print: Put Code Page Text Matching Your Selection

*If you do not put at least the minimum amount of characters than the line buffer of the paper width, the printer will not print until the line buffer is full.*

● Hex Mode
○ ASCII Mode

**CODE PAGES**

------------------------------------------------------------

This command will set the Code Page on the printer:

| | | | | |
|---|---|---|---|---|
| ASCII: | ESC | GS | t | *n* |
| Hex: | 1B | 1D | 74 | *n* |

*n = code page selection*

Defined Region for n
0 ≤ n ≤ 21
32 ≤ n ≤ 34
64 ≤ n ≤ 79

n = 0 = Normal*
n = 1 = 437 (USA, Std. Europe)
n = 2 = Katakana
n = 3 = 437 (USA, Std. Europe)
n = 4 = 858 (Multilingual)
n = 5 = 852 (Latin-2)
n = 6 = 860 (Portuguese)
n = 7 = 861 (Icelandic)
n = 8 = 863 (Canadian French)

## Stored Logo Printing

Stored logo printing is done in the sample receipt. Please review that and the Thermal Line Mode Manual for more information.

ESC FS p 1 0

## Getting Online Status of the Printer

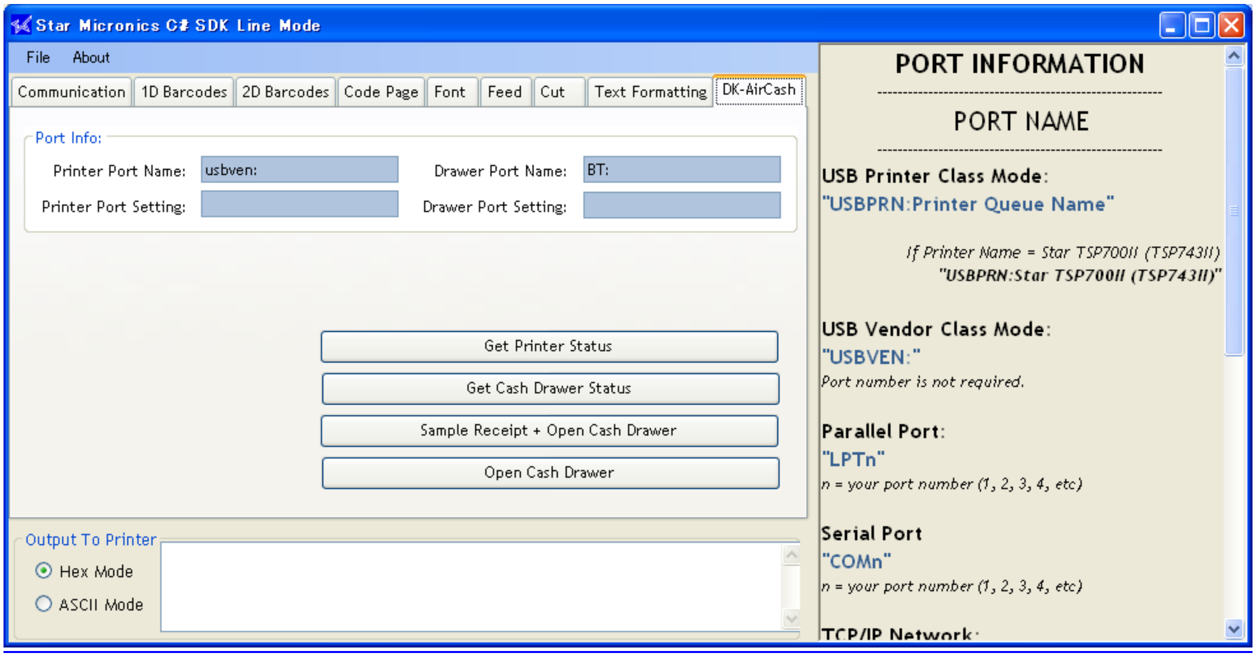
Visit the function code block called **printToPrinter** and there you will see the StarIO method **GetOnlineStatus** being used to retrive a boolean value for online status.

True = Online
False = Offline

## Getting Parsed Status of the Printer

Review the function code block called **timerGetStatus_Tick** and there you will see the StarIO method **GetParsedStatus** being used to pull a struct down of all the potential status flags the printer can throw. Click here for the full list of statuses.

## DK-AirCash



Printer Port Name: usbven:

Printer Port Setting:

Drawer Port Name: BT:

Drawer Port Setting:

● Hex Mode

○ ASCII Mode

PORT INFORMATION

PORT NAME

USB Printer Class Mode:
"USBPRN:Printer Queue Name"

*If Printer Name = Star TSP700II (TSP743II)*
*"USBPRN:Star TSP700II (TSP743II)"*

USB Vendor Class Mode:
"USBVEN:"
*Port number is not required.*

Parallel Port:
"LPTn"
*n = your port number (1, 2, 3, 4, etc)*

Serial Port
"COMn"
*n = your port number (1, 2, 3, 4, etc)*

TCP/IP Network:

### Get Printer Status /Get Cash Drawer Status

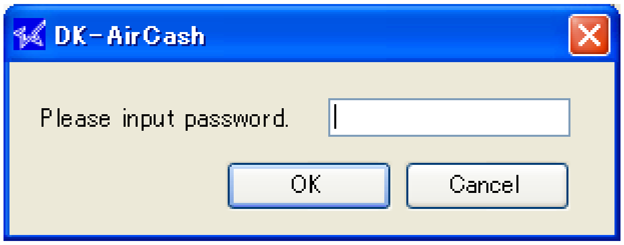
The status of the connected printer or cash drawer is displayed.

### Sample Receipt + Open Cash Drawer

When a printer is connected, it prints a sample receipt and a cash drawer is opened if connected. A password is required to open the cash drawer. The default password is "1234"

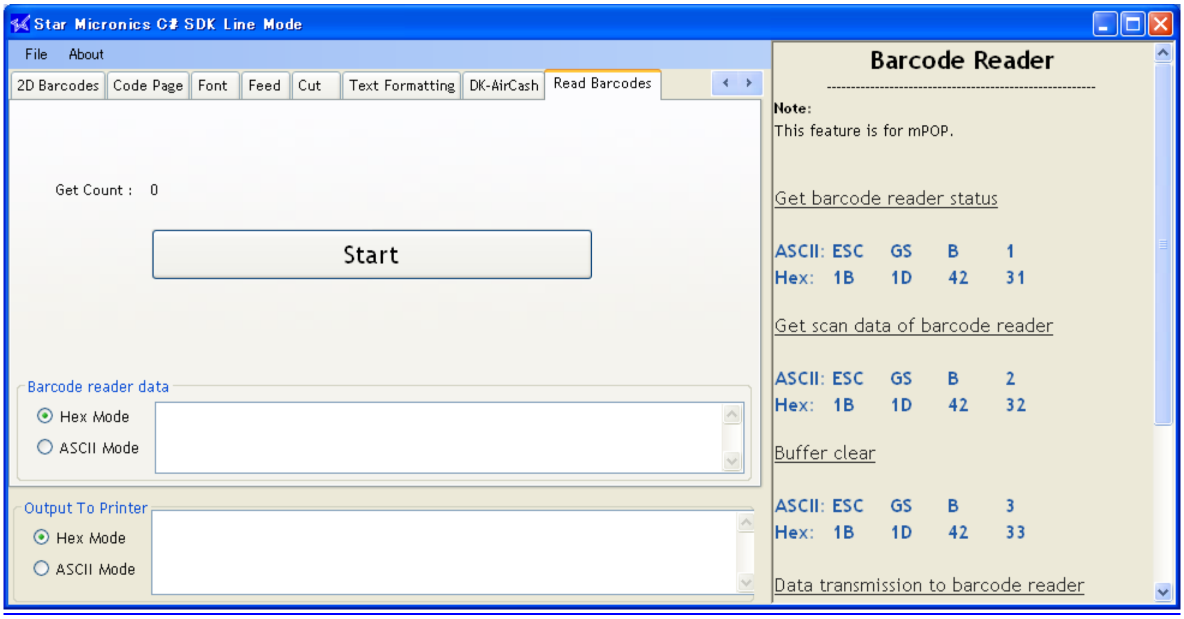
### Open Cash Drawer

A cash drawer is opened if connected. A password is required to open the cash drawer. The default password is "1234".



Please input password.

## Read Barcodes



Get Count : 0

**Barcode Reader**

**Note:**
This feature is for mPOP.

Get barcode reader status

ASCII: ESC GS B 1
Hex: 1B 1D 42 31

Get scan data of barcode reader

ASCII: ESC GS B 2
Hex: 1B 1D 42 32

Buffer clear

ASCII: ESC GS B 3
Hex: 1B 1D 42 33

Data transmission to barcode reader

### Start

If “Start” is pressed when a printer can communicate with a barcode reader, it will change to “Stop” and update a value of Get Count.
The result of reading by the barcode reader is displayed in the “Barcode reader data” window.

*Note
In case of Bluetooth interface, the reading response of a barcode reader is dependent on the Timeout value specified by OpenPort.
If a quick response is needed, set the timeout value low after a sufficient verification.

### Status request

ESC GS B 1

### Barcode data request

ESC GS B 2

### Buffer clear

ESC GS B 3

### Send data to a barcode reader

ESC GS B 0 n1 n2 d1 … dk
n = byte length
k = n1 + n2 * 256

# Tips for App Development when using StarIO

Star Micronics prides itself as the industry leader in great POS products and with great power comes great responsibility. Below is a tips section just to help you get on the fast track to software development with StarIO.

**TIP #1:** If you are going to be coding a large project, create a class to abstract all the printing methods into class(s) instead of having the code reside in the main code block. This will help with code reusability and will also save you time in the long run from having to find one line of code in the main code. By having StarIO only reside in the class(s), you will be fully taking advantage of object oriented programming.

**TIP #2:** Know what the differences and definitions of (ASCII & Unicode), (Hex & Decimal), and (Byte & Char) are. A byte is normally 8-bits long which would be 8 digits of binary (1s and 0s). These bytes are just 8 bits of binary data but bytes can also be int or char. The three different variable types basically hold the data in the same way but there are slight differences. Try to code with Bytes instead of Chars, ints, or strings when choosing a variable to contain your print job data. ASCII to Unicode and vice versa conversions are sometimes unsecure so make sure you know what and how the encoding class works with these. Big mistakes made in Unicode are culture-sensitive search and casing, surrogate pairs, combining characters, and normalization which are answered here.

**TIP #3:** HEX DUMP MODE! If you are debugging and your application seems to have a bug in it use hex dump mode on the printer. This is the best way to verify what is being sent out of the computer is being received correctly. To put the printer in hex dump mode, turn the printer off, open the cover to the paper, hold the feed button down, turn the printer back on, close the cover, let go of the feed button. Hex dump mode is a sure fire way to verify hex data is sent correctly. When in hex dump mode, printer functions will not work.

**TIP #4:** Do not waste time trying to reverse engineer StarIO command codes. All the available StarIO commands are available in the Thermal Line Mode Spec Manual and that is the best resource to use when researching a specific StarIO command. This SDK & Manual was built to help you (The Developer) have a very easy job ahead of you to program for Star Printers.

**TIP #5:** If there is a command that is not covered in this SDK but you wish to see a code snippet of that command in use then visit our Developers' section for a possible code block that matches your needs.

**TIP #6:** StarIO, ESC/POS, UPOS: JavaPOS, POS for .NET, & OPOS are all different ways to communicate with the printer. Visit our Developers' section for more info on these. This SDK covers StarIO only.

# Additional Resources

This section will share resources that will help you develop good software with StarIO.

Star Micronics Developers Network

Browse Star Micronics' FAQs, ask a question, look up information, etc.

The Developers Network gets you access to:

- Updated Versions of this Manual and Source Code
- Getting Started Advice and Industry Information
- Star Micronics Printer Drivers
- Technical Questions/Support

Download the Star Thermal Line Mode Command Spec Manual

Download the StarPRNT Command Specifications for mPOP

Download the Star Impact Printer Command Spec Manual

Use it as your reference for all StarIO Line Mode commands.

Character Encoding in the .NET Framework

If you don't know what ASCII and Unicode is, this is a good place to start.

Microsoft .NET Internationalization

Good resource for more detail on internationalization.

Visual C# Developer Center

Great place to learn more about the C# language.

Unicode.org

The Unicode Consortium – Good place to learn more about Unicode.

1D Barcodes

Barcode Island is a great resource for specs on 1D barcodes.

2D Barcodes

Great place for information on 2D Barcodes, QR Codes, and PDF417

Code Pages

Learn about Code Pages here.

# ASCII Table Resource

| ASCII | Hex | Symbol | ASCII | Hex | Symbol | ASCII | Hex | Symbol | ASCII | Hex | Symbol |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | NUL | 16 | 10 | DLE | 32 | 20 | (space) | 48 | 30 | 0 |
| 1 | 1 | SOH | 17 | 11 | DC1 | 33 | 21 | ! | 49 | 31 | 1 |
| 2 | 2 | STX | 18 | 12 | DC2 | 34 | 22 | " | 50 | 32 | 2 |
| 3 | 3 | ETX | 19 | 13 | DC3 | 35 | 23 | # | 51 | 33 | 3 |
| 4 | 4 | EOT | 20 | 14 | DC4 | 36 | 24 | $ | 52 | 34 | 4 |
| 5 | 5 | ENQ | 21 | 15 | NAK | 37 | 25 | % | 53 | 35 | 5 |
| 6 | 6 | ACK | 22 | 16 | SYN | 38 | 26 | & | 54 | 36 | 6 |
| 7 | 7 | BEL | 23 | 17 | ETB | 39 | 27 | ' | 55 | 37 | 7 |
| 8 | 8 | BS | 24 | 18 | CAN | 40 | 28 | ( | 56 | 38 | 8 |
| 9 | 9 | TAB | 25 | 19 | EM | 41 | 29 | ) | 57 | 39 | 9 |
| 10 | A | LF | 26 | 1A | SUB | 42 | 2A | * | 58 | 3A | : |
| 11 | B | VT | 27 | 1B | ESC | 43 | 2B | + | 59 | 3B | ; |
| 12 | C | FF | 28 | 1C | FS | 44 | 2C | , | 60 | 3C | < |
| 13 | D | CR | 29 | 1D | GS | 45 | 2D | - | 61 | 3D | = |
| 14 | E | SO | 30 | 1E | RS | 46 | 2E | . | 62 | 3E | > |
| 15 | F | SI | 31 | 1F | US | 47 | 2F | / | 63 | 3F | ? |

| ASCII | Hex | Symbol | ASCII | Hex | Symbol | ASCII | Hex | Symbol | ASCII | Hex | Symbol |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 64 | 40 | @ | 80 | 50 | P | 96 | 60 | ` | 112 | 70 | p |
| 65 | 41 | A | 81 | 51 | Q | 97 | 61 | a | 113 | 71 | q |
| 66 | 42 | B | 82 | 52 | R | 98 | 62 | b | 114 | 72 | r |
| 67 | 43 | C | 83 | 53 | S | 99 | 63 | c | 115 | 73 | s |
| 68 | 44 | D | 84 | 54 | T | 100 | 64 | d | 116 | 74 | t |
| 69 | 45 | E | 85 | 55 | U | 101 | 65 | e | 117 | 75 | u |
| 70 | 46 | F | 86 | 56 | V | 102 | 66 | f | 118 | 76 | v |
| 71 | 47 | G | 87 | 57 | W | 103 | 67 | g | 119 | 77 | w |
| 72 | 48 | H | 88 | 58 | X | 104 | 68 | h | 120 | 78 | x |
| 73 | 49 | I | 89 | 59 | Y | 105 | 69 | i | 121 | 79 | y |
| 74 | 4A | J | 90 | 5A | Z | 106 | 6A | j | 122 | 7A | z |
| 75 | 4B | K | 91 | 5B | [ | 107 | 6B | k | 123 | 7B | { |
| 76 | 4C | L | 92 | 5C | \ | 108 | 6C | l | 124 | 7C | \| |
| 77 | 4D | M | 93 | 5D | ] | 109 | 6D | m | 125 | 7D | } |
| 78 | 4E | N | 94 | 5E | ^ | 110 | 6E | n | 126 | 7E | ~ |
| 79 | 4F | O | 95 | 5F | _ | 111 | 6F | o | 127 | 7F | □ |

*Use this to compare hex values to symbol (ASCII) values.*

Star Micronics is a global leader in the manufacturing of small printers. We apply over 50 years of knowhow and innovation to provide elite printing solutions that are rich in stellar reliability and industry-respected features. Offering a diverse line of Thermal, Hybrid, Mobile, Kiosk and Impact Dot Matrix printers, we are obsessed with exceeding the demands of our valued customers every day.

We have a long history of implementations into Retail, Point of Sale, Hospitality, Restaurants and Kitchens, Kiosks and Digital Signage, Gaming and Lottery, ATMs, Ticketing, Labeling, Salons and Spas, Banking and Credit Unions, Medical, Law Enforcement, Payment Processing, and more!

High Quality POS Receipts, Interactive Coupons with Triggers, Logo Printing for Branding, Advanced Drivers for Windows, Mac and Linux, Complete SDK Packages, Android, iOS, Blackberry Printing Support, OPOS, JavaPOS, POS for .NET, Eco-Friendly Paper and Power Savings with Reporting Utility, ENERGY STAR, MSR Reading, *future*PRNT, StarPRNT… How can Star help you fulfill the needs of your application?

Don’t just settle on hardware that won’t work as hard as you do. Demand everything from your printer. Demand a Star!

| Version | Release Date |
|---|---|
| 1.0.0 | July 2011 |
| 1.1.0 | Oct. 2011 |
| 1.2.0 | Jan. 2012 |
| 2.0.0 | Mar. 2014 |
| 2.1.0 | July 2014 |
| 2.2.0 | Jan. 2016 |

## Star Micronics Worldwide

Star Micronics Co., Ltd.
536 Nanatsushinya
Shimizu-ku, Shizuoka 424-0066
Japan
+81-54-347-2163
http://www.star-m.jp/eng/index.htm

Star Micronics America, Inc.
1150 King Georges Post Road
Edison, NJ 08837
USA
1-800-782-7636
+1-732-623-5500
http://www.starmicronics.com

Star Micronics EMEA
Star House
Peregrine Business Park, Gomm Road
High Wycombe, Buckinghamshire HP13 7DL
UK
+44-(0)-1494-471111
http://www.star-emea.com

Star Micronics Southeast Asia Co., Ltd.
Room 2902C. 29th Fl. United Center Bldg.
323 Silom Road, Silom Bangrak, Bangkok 10500
Thailand
+66-2-631-1161 x 2
http://www.starmicronics.co.th/