XPOS



XP-3615, XP-3685W USER MANUAL

Copyright

This document is copyrighted, © 2018. All rights are reserved. No part of this document may be reproduced, copied, translated, or transmitted in any form or by any means without the prior written permission from Firich Enterprise Co., Ltd. Information provided in this manual is intended to be accurate and reliable. However, Firich Enterprise Co., Ltd. assumes no responsibility for its use, nor for any infringements upon the rights of third parties, which may result from its use.

The material in this document is for product information only and is subject to change without notice. While reasonable efforts have been made in the preparation of this document to assure its accuracy, Firich Enterprise Co., Ltd., assumes no liabilities resulting from errors or omissions in this document, or from the use of the information contained herein.

VERSION CONTROL

Date	Version	Change Description	PM
11/27	1.0	Initial Version	Yuting Kao
01/22	1.1	Added 15.6 specifications; Other minor modifications	Yuting Kao

TABLE OF CONTENTS

1. INTRODUCTION

	XP-3685 Basic Introduction	005 - 009
	Dimensions	010 - 010
	Specification	011 - 011
	Packing List	012 - 012
2. S	OFTWARE INSTALLATION AND SETUP	
	Installation Order	012 - 012
	Chipset	013 - 014
	Audio	014 - 014
	Graphics	015 - 015
	Intel ME	016 - 016
		017 - 017
	USB LAN Windows Driver	010 - 010
	Set COM	020 020
	Sensor	020 - 020
	Intel® Rapid Storage Technology	022 - 023
	ADB Android Debug Bridge	023 - 023
3. Ha	ardware	
	Access Storage Device	023 - 024
	Adding or Removing M 2 PCBA from M 2 Tray	025 - 025
	Access the Motherboard	026 - 027
	Install or Remove Memory	027 - 028
4. IO	Board SDK Instruction	
	Overview	028 - 028
	System Requirements & Installation	029 - 030
	DLL Interface for FEC Io Board SDK	030 - 032
	Example of DLL Interface	033 - 033
	EXE Interface for FEC Cash Drawer SDK	033 - 034
5. Sc	oftware	
	Developers Guide for Serial Communications	034 - 041
	Ambient Light and Proximity Sensor	041 - 043
	Control the Sensor Timing	044 - 044
	Cash Drawer Command	045 - 045

XP-3685 Basic Introduction

XPOS XP-3685 is a fifteen inch all in one fanless point of sales system powered by Intel® 7th generation Celeron, i3 and i5 processors. By utilizing slim key components and enclosing them in an aluminum die-casting chassis, XP-3685 is able to achieve both a slim and strong design. The IO interface is connected with an USB cable and installed inside the stand base.

The dual hinge stand provides the user the ability to adjust the stand and display angles making it optimal for all environments.

On the bottom of the display is a standard 2in1 sensor. The proximity sensor can detect a presence infront of the sensor and wake the system up from an S1 and S3 sleep state allowing users to save on power consumption when the store traffic is low.

- Slim Panel PC 2.4cm
- Fanless Design
- 2in1 Sensor: Ambient Light Sensor & Proximity Sensor
- Dual Hinge Stand: High Profile and Low Profile
- Integrated and Extended 2nd Display Options













- Power Button with LED:
 - Red = Power adaptor connected but system is off
 - Blue = System is on
- Storage LED
 - Orange = Flashing during storage activity





Galaxy Gray with Black Silver with Black



Dimensions



Display Dimension

		XP-3685	XP-3685W
Α	Display Height	273.3mm	235.7mm
В	Display Length	342.5mm	368.2mm



Side Profile Dimension

		XP-3685	XP-3685W
Α	Panel PC Thickness	24mm	24mm
В	System	332mm	293mm
С	Base Depth	231.75mm	231.75mm

SPECIFICATION

Panel PC

Description	XP-3685				
Processor	Intel [®] Celeron [®] Processor 3965U, 2M Cache, 2.20 GHz				
	Intel® Core™ i3-7100U Proces	ssor, 3M Cache, 2.40 GHz			
	Intel® Core™ i5-7300U Processo	r, 3M Cache, up to 3.50 GHz			
System Memory	4GB Standard, Maximum 32	2GB (2 x 260-pin DDR4)			
Storage Device	Celeron: 1 x M.2 (B+M Key) SATA III ar	nd PCIE, 1 x M.2 (BM Key) SATA III			
	i3 & i5: 2 x M.2 (B+M Ke	y) SATA III and PCIE			
Speaker	2 x 2W Interna	al Speaker			
Construction	Aluminum Die-casting + Plastic + Glass				
Housing Color	Galaxy Gray + Black				
	Silver + E	Silver + Black			
Touch LCD Display					
Size / Resolution	15″ TFT-LCD / 1024 x 768	15.6" TFT-LCD / 1920 x 1080			
Brightness / Backlight	400nits (LED) PCAP	400nits (LED) PCAP			
Panel PC IO					
USB Port	1 x USB 1.0 Type A, 1 x USB 2.0 Type A				
	1 x USB DP (Reserved for 2nd Display)				
FEC DP Port	1 x FEC DP Port (Customer Displa	ay or Integrated 2 nd Display)			

IO Options



IO1	
System on Module (SOM)	ARM Cortex A7 Quad Core
Optional	1G DDR3 On Board
	8GB eMMC 5.0 On Board
SD Card Slot	1 x SD Card Slot (Must Have SOM)
Video Port	1 x Video Port (Must Have SOM)
USB DP	2 x USB DP Port (1Reserved to Connect to Panel PC)
USB Port	4 x USB 2.0 Type A
LAN Port	1 x LAN (Green Light Mega LAN, Orange Light Giga LAN)
Serial Port	2 x RJ45 (RS232)
DC-in	1 x DC-in for 90W (20V/4.5A) Adaptor

Packing List

Standard	Optional
XP-3685 x 1	WiFi Module x 1
M.2 Tray x 1	M.2 Tray x 1
COM RJ45 to DB9 Cable x 2	Add-on Device x1
90W Adaptor x 1	Customer Display x 1
Power Cord x 1	2 nd Display x 1
Driver and Manual CD x 1	

Software Installation and Setup: Motherboard

Follow the below order if installation for the Motherboard:

(1) Chipset (2) Audio (3) Graphics (4) Intel® ME

(5) Setup Serial IO (6) USB LAN (7) Windows Drivers (8) Set COM

(9) Intel® Rapid Storage Technology (RST) (Applicable to i3 & i5) (10) ADB Driver (Applicable to SOM)

📙 🛃 📕 🖛 WIN10				- 0	×		
File Home Share View							
← → × ↑ 📙 > This PC > Local Disk (C:) > New folder > 主機 > WIN10 🗸 ひ Search WIN10							
Date modified Type							
🖈 Quick access		Audio 9425	4/25/2018 4:02 AM	Compressed (zipp	205.99		
E Desktop	*	Chinset 10.1.1.45	4/25/2018 3:59 AM	Compressed (zipp	3 71		
🖶 Downloads	*	CP210x Windows Drivers.rar	10/30/2018 8:45 PM	RAR File	6 56		
🔮 Documents	*	ME 11.8.50.3425 SOL	5/29/2018 8:51 PM	Compressed (zipp	104.65		
Pictures	*	 RST_16.5.0.1021	5/27/2018 8:49 PM	Compressed (zipp	17,80		
Con Drive		SerialIO_30.100.1746.4	4/25/2018 4:02 AM	Compressed (zipp	2,52		
Chebrive		USB_LAN_Win10_10023_03012018	4/25/2018 4:02 AM	Compressed (zipp	9,46		
💻 This PC		VGA_4973	4/25/2018 4:04 AM	Compressed (zipp	338,66		
Desktop							
🔮 Documents							
👆 Downloads							
h Music							
Pictures							
Videos							
Local Disk (C:)							
i Network							
	<				>		
8 items							

• Un-compress files

1. Chipset



You must restart this computer for the changes to take effect.

View Log Files

Locate chipset folder and double click on [SetupChipset] •



2. Audio

- Double click [Setup] • I 🛛 🗌 🖬 Audio_8425 > Audio ~ 主楼 Name -=1 0x0412 Desktop Downloads 0x0413 0x0414 6041 0x0416 Pictures 0x0419 0-042 ConeDrive This PC This PC
 Desktop
 Documents
 Documents
 Downloads
 Music
 Pictures
 Pictures
 Videos
 Local Disk (C:) Network setup.inx
 setup.isn
 setup.iss
 USetup.iss ISS File < 46 items
 - Click [Next]



User Account Control × Do you want to allow this app to make changes to your device? * InstallScript Setup Launcher Unicode * Verified publisher: Realtek Semiconductor Corp. File origin: Hard drive on this computer * Show more details * Yes No

• Click [Yes, I want to restart my computer now], [Finish]



• Click [Yes]

3. Graphics







4. Intel® ME

• Double Click [SetupME]

- I 🖓 📴 🖛 I	_	Compressed Folder Tools ME_11.8.50.34	25_SOL	- 0	×
File Home Share	View	Extract			~ 📀
← → ~ ↑ 📙 « WIN10	> M	_11.8.50.3425_SOL > ME_11.8.50.3425_SOL	~ Ö	Search ME_11.8.50.3425_SOL	P
		Name ^	Туре	Compressed size	Passw
📌 Quick access		IntelMEEW/er dil	Application extens	ion 13 KB	No
Desktop	Å		VMI December	3 40	
Downloads	Å	E SetupME	Application	104,643 KB	N
Documents	R				- 1
E Pictures	1				
2Audio					
3Graphics					
New folder					
windows10					
a OneDrive					
This PC					
Cesktop					
Documents					
Downloads					
Music					
Pictures					
😸 Videos					
Local Disk (C:)					
i Network					
		¢		_	>
3 items				B	

• Accept Terms then click [Next]



• Click [Yes]



• Click [Next]



• Click [Next]



• Click [Finish]



5. Serial IO

• Double Click [SetupSerialIO]



• Accept terms then click [Next]



• Click [Next]

Setup			×
Intel® Serial IO Confirmation		(inte	D
You are about to install the following components: - Intel® Serial IO GPIO Driver - Intel® Serial IO I2C Driver			
Intel Corporation	< Back	Next >	Cancel

• Click [Next]



• Click [Next]



 Select Yes, I want to restart this computer now the click [Finish]



6. USB LAN

• Double Click [Setup]

🖸 📙 🖛	Application Tools New folder		- 0	×
File Home Share Vie	v Manage			~ 6
⊢ → × ↑ 📙 > This PC >	Local Disk (C:) > New folder > New folder	✓ [™] Searce	h New folder	P
	Name ^	Date modified	Type	Size '
🖈 Quick access	街」 0x0414	11/6/2017 4:59 PM	Configuration sett	
Desktop	r 🔝 0x0415	11/6/2017 4:59 PM	Configuration sett	
Downloads	 6 0x0416 	11/6/2017 4:59 PM	Configuration sett	
Documents	0x0418	11/6/2017 4:59 PM	Configuration sett	
E Dictures	6 0x0419	11/6/2017 4:59 PM	Configuration sett	
A ANDES	0x0421	11/6/2017 4:59 PM	Configuration sett	
ZAUDIO	6 0x0424	11/6/2017 4:59 PM	Configuration sett	
3Graphics	Ox0804	11/6/2017 4:59 PM	Configuration sett	
4ME	0x0816	11/6/2017 4:59 PM	Configuration sett	
New folder	🔮 data1	11/6/2017 4:59 PM	Cabinet File	6,
 Occ004 	data1.hdr	11/6/2017 4:59 PM	HDR File	
OneDrive	🔮 data2	11/6/2017 4:59 PM	Cabinet File	
💻 This PC	EngLangID	11/6/2017 4:59 PM	Text Document	
Desktop	instCtrl	11/6/2017 4:59 PM	Text Document	
Documents	instOpt	11/6/2017 4:59 PM	Text Document	
Developede	ISSetup.dll	11/6/2017 4:59 PM	Application extens	
- Downloads	Buyeritte	10/0/2017 102 011	001100	
Music	🗹 🛂 setup	11/6/2017 4:59 PM	Application	1,
Pictures	(g) setup	T1/6/2017 4:39 PW	Configuration sett	
🚼 Videos	i setup.inx	11/6/2017 4:59 PM	INX File	
Local Disk (C:)	setup.isn	11/6/2017 4:59 PM	ISN File	- 1
A 11-1-1-1	i setup.iss	11/6/2017 4:59 PM	ISS File	
Intervork	setupctrl	11/6/2017 4:59 PM	Text Document	
	٢			>





7. Windows Driver

• Double Click [CP210XVCPInstaller_x64]



• Click [Yes]



8. Set COM and COM Address

1. Double click [SetXPOSCOM]

• You will see the COM is set in Device Manager

- 🛃 - =		Application Tools	20180910_SetXPC	SCOM v1.0.1		- 0	×
File Home Sh	are Vi	ew Manage					~ 🕐
\leftarrow \rightarrow \checkmark \land \square \diamond	This PC	Desktop > 20180910_9	SetXPOSCOM v1.0.1	~	ප Search 20180910	_SetXPOSCO	9
		Name ^		Date modified	Туре	Size	
Quick access		🗉 devcon_64		12/18/2015 1:44 AN	1 Application	80 KB	
Desktop		😽 SetXPOSCOM		9/11/2018 6:21 AM	Application	15 KB	
Downloads	* [SetXPOSCOM.exe.conf	fig	8/8/2018 4:38 PM	CONFIG File	1 KB	
Documents	* [SetXPOSCOM.pdb		9/11/2018 6:21 AM	PDB File	20 KB	
Pictures	* [😺 SetXPOSCOM.vshost		9/12/2018 1:17 AM	Application	23 KB	
sleeper	[SetXPOSCOM.vshost.e	xe.config	8/8/2018 4:38 PM	CONFIG File	1 KB	
🝊 OneDrive	[SetXPOSCOM.vshost.e	xe.manifest	8/10/2018 4:29 AM	MANIFEST File	4 KB	

COM Address

You can check Ports (COM & LPT) in the Device Manager



After the SET COM you should see the below items in the Device Manager:

- Communications Port (COM1): FEC DP interface reserved for customer display
- Communications Port (COM2): Reserved on motherboard
- Silicon Labs Quad CP2108 USB to UARD Bridge: Interface 0 (COM98): GPIO Control for IO1
- Silicon Labs Quad CP2108 USB to UARD Bridge: Interface 1 (COM3): RJ45 Interface on IO for Devices
- Silicon Labs Quad CP2108 USB to UARD Bridge: Interface 1 (COM4): RJ45 Interface on IO for Devices
- Silicon Labs Quad CP2108 USB to UARD Bridge: Interface 1 (COM3): RJ45 Interface on IO for Devices
- Intel iAMT: Available on i3 and i5 models

9. Sensor

•	Double Click	install]	Click [Yes]	
 ▶ ○ ■ = Vishayli 文件 主页 共享 ← → < ↑ ▶ □ 	stall 査看 印題 > DATA (G:) > Vishayinstall 〜 こ	ー ロ × 、 愛家"Vishayinstall" タ	User Account Control X Do you want to allow this app to make	
 助电脑 3D 对象 下载 文档 国片 资质 资析 資面 	◆ 名称	修改日期 2018/11/26 下午 0 2018/11/26 下午 0 2018/11/26 下午 0 2018/11/2 下午 0	Windows Command Processor Verified publisher: Microsoft Windows Show more details	
 OS (C:) 新加卷 (D:) Acer (E:) 	v c		Yes No	

Press any key to continue ...

C\Windows\System32\cmd.exe		-	
logfile = C:\Windows\Hi	crosoft.NET\Framework64\v4.0.30319\VishaySensorService.InstallLog		
The transacted install has [SC] ChangeServiceConfig S	completed. UCCESS		
SERVICE_NAME: SensrSvc TVPE STATE WIM32_EXIT_CODE SERVICE_EXIT_CODE CHECKPOINT WAIT_HINT PID FLAGS [SC] ChangeServiceConfig S	: 20 MIN32_SHARE_PROCESS : 2 START_PENDING (NOT_START_PENDING : 0 (OX : 0 (OX : 0 (OX : 0 (OX : 0 (OX : 0 (OX : 5222 : 5222 : 5222 : CCESS		
SERVICE_NAME: VishaySensor	Service		
TYPE STATE	: 10 WIN32_OWN_PROCESS : 2 START_PENDING (NOT STOPPABLE, NOT PAUSABLE, IGNORES_SHUTDOWN)		
WIN32_EXIT_CODE SERVICE_EXIT_CODE CHECKPOINT WAIT_HINT PID FLAGS	: 0 (0x0) : 0x0 : 0x0 : 0x0 : 4x2 : 4428 : -		
Press any key to continue			

9. Intel[®] Rapid Storage Technology (RST)

Double Click [SetupRST]

٠

For additional information about Intel RST: <u>https://downloadcenter.intel.com/product/55005/Intel-</u> <u>Rapid-Storage-Technology-Intel-RST-</u>

🛃 📮 🛛 RST_16.5.0.1	21				-		\times
【件 主页 共享 计	iff						~ 6
🛧 📜 « Produ	ts > XP-3685 > User	Manual > RST_16.5.0.1021	~ U	搜索*RST	16.5.0.1021	•	,p
User_Manual		^	修改日期		类型		
windows10	🖹 mup		2018/4/27 下	午 10:	XML文档	_	
Creative Cloud Files	🛃 SetupRST		2018/4/27下	午 10:	应用程序		
0.00	WixLicense	e	2018/4/27 下	ዋ 10:	文本文档		
5 3D 対象							
ト 102 下 102							
10 ×柏							
10 0000							
- 虞而							
- OS (C:)							
- 新加卷 (D:)							
◆新加卷 (D:) >> pm (\\192.168.0.5) (Z:							



• Click [Yes]

• Accept terms then click [Next]



Choose save location and click [Next]



Click [Install]

回 Windows 安全		
你想安装这个设备软件吗?		
名称: Intel Corporation IDE ATA/ATAPI 控制器 发布者: Intel(R) Rapid Storage Technology		
☑ 始终信任来自 *Intel(R) Rapid Storage	Install (I)	不安装(N)
Technology 的软件(A)。		





• Click [Next]

商认		intel
您即将安装以下组件:		action and
- 英特尔 ⑧快速存储技术		
Create an icon on Desktop (D)		
	the second se	and the second sec

Choose to restart then click [Finish]



10. ADB (Android Debug Bridge)

This driver is only applicable the system on module (SOM) is installed

• Double Click on [DriverInstall]

📜 🔄 📜 🖬 DriverAssitant	14.5		- 0	×
文件 主页 共享 查看	ł			~ 0
← → 👻 🖡 « User_Ma	nual > Links > windows10 > DriverAssitant_v4.5 >	~ ひ 搜索	DriverAssitant_v4.5*	P
	^ 名称 [^]	修改日期	类型	*
📙 3D 对象	ADBDriver	2018/11/25 下午 1	文件夹	
≽ 下载	📜 bin	2018/11/25 下午 1	文件夹	
D 文档	Driver	2018/11/25 下午 1	文件夹	
▶ 图片	📜 Log	2018/11/25 下午 1	文件夹	
■ 视频	sil comg	2010/10/27 11+1	IGROOM	
▶ 音乐	DriverInstall	2016/10/27 上午 1	应用程序	
늘 卓面		COTO/TO/CI at 1	- A4XII	
🥾 OS (C:)				
🥧 新加卷 (D:)				
② DVD RW 驱动器 (E:) Disc				
🦡 pm (\\192.168.0.5) (Z:)				
② DVD RW 驱动器 (E:) Disc				
IO Board				
主機				
7 个项目	v <			>

Click [Install Driver]
 RK Driver Assitant v4.5

• Click **[OK]**

DriverIn	stall	×
Install	driver ok.	

Hardware

Access Storage Device

- 1. Make sure the system is turned off (If using RAID system can be on)
- 2. Flip open the cover which is held together by magnets



3. Remove the screw(s)



If you are using RAID, follow the below step. If not, skip this step. Make sure to flip the small switch to the right ► to turn off the power to the storage device. After the M.2 is inserted, flip the switch back to on <.



5. Flip open the handle and slowly pull out the M.2 Tray



Adding or removing M.2 from tray

Note: This image is taped on the M.2 tray for users to identify which direction to place the M.2 module as well as how to install the M.2.



1. Remove Screw



2. Slowly slide out the M.2 module



Accessing the Motherboard

1. Remove the 3 screws as shown below



2. Remove the bottom plastic away from the aluminum chassis.



3. Slide the display module ~1cm as shown below



4. Lift the display up. There will be cables between the panel and motherboard. Reach your hand in and disconnect the cables on the motherboard.



Memory

1. Slightly pull to the memory socket to the left and right. The memory will pop up





2. Remove the memory



3. IO Board SDK Instruction

1. Overview

This document describes about how to use the FEC IO Board SDK to control IO functions on FEC IO Board via serial interface. The FEC IO Board SDK support the DLL and EXE interface controlling IO on windows application

The IO functions are:

- COM A: RS232 Port A Enable/Disable, 5V/12V Setting
- COM B: RS232 Port B Enable/Disable, 5V/12V Setting
- Cash Drawer: Open, Get status
- USB Smart COM: Enable/Disable
- Reset SOM (Android System)
- Reset CCG4 USB Type-C Control IC

2. System Requirement & Installation

Supported Operating System(OS) Microsoft[®] Windows 10 IOT Enterprise LTSB

Installation

• Click [FECIOBoardSDKSETUP.exe]

名稱 ^	修改日期	類型	大小
💼 FEC IOBoard SDK Integration Guide V0	2018/10/26 下午	Microsoft Word	43 KB
FECIOBoardSDKSETUP.exe	2018/10/26 下午	應用程式	296 KB

• Once completed, click [Close]



FEC IO Board SDK will installed in
 C:\Program Files(x86)\FEC\IOBoardSDK



FEC IOBoard SDK Setup	-	
Installation Complete		100
Setup was completed successfully.		3
Completed		
Show details		
ullsoft Install System v2.46		

* 本枝	奥姬崃 (C:) > Program Files (x86) > FEC >	IOBoardSDK v (· 授聘 IOBoards	SDK
^	名稱	修改日期	類型	大小
	FEC_IOBoard_Utility.exe	2018/11/7 下午 0	應用程式	16 KB
	fec_xpos_ioboard_dll.dll	2018/9/25 下午 0	應用程式擴充	12 KB
	FEC_XPOS_IOBoard_Tester.exe	2018/9/25 下午 0	應用程式	15 KB
	🛞 Uninstall.exe	2018/11/19 下午	應用程式	160 KB

3. DLL Interface for FEC IO Board SDK

FEC IO Board SDK provide DLL interface to control IO Board, the DLL name is fec_xpos_ioboard_dll.dll.

API Functions & definitions		
#define CTL_COM_MODE_RI	0x00	
#define CTL_COM_MODE_DC	0x01	
#define CTL_COM_PWR_5V	0x02	
#define CTL_COM_PWR_12V	0x03	
#define CTL_CASH_OUT_LOW	0x04	
#define CTL_CASH_OUT_HIGH	0x05	
#define CTL_CASH_OUT_HIGH_200MS	0x06	
#define CTL_CASH_PWRSEL_12V	0x07	
#define CTL_CASH_PWRSEL_24V	0x08	
#define CTL_SOM_RST_LOW	0x09	
#define CTL_SOM_RST_HIGH	0x10	
#define CTL_TYPEC_FW_RS_LOW	0x11	
#define CTL_TYPEC_FW_RS_HIGH	0x12	
#define CTL_SMART_COM_BYPASS	0x13	
#define CTL_SMART_COM_SMART_COM	0x14	
	• • • • •	

- DLLExport int SetComAMode(int mode)
- This function enable/disable the COM Port A power supply

Parameter: int mode:

CTL_COM_MODE_RI (0x00): Disable COM A Power supply CTL_COM_MODE_DC (0x01): Enable COM A Power supply

Return Value:

Fail: 1

Success: 0

DLLExport int SetComAPwr(int pwrmode)

This function set the COM Port A power level to 5V or 12V

Parameter: int pwrmode: CTL_COM_PWR_5V (0x02): Set to 5V CTL_COM_PWR_12V (0x03) : Set to 12V Return Value: Fail: 1

Success: 0

DLLExport int SetComBMode(int mode)

This function enable/disable the COM Port B power supply

Parameter:

int mode: CTL_COM_MODE_RI (0x00): Disable COM B Power supply CTL_COM_MODE_DC (0x01): Enable COM B Power supply Return Value:

Fail: 1

Success: 0

• DLLExport int SetComBPwr(int pwrmode)

This function set the COM Port B power level to 5V or 12V

Parameter:

int pwrmode:

CTL_COM_PWR_5V (0x02): Set to 5V CTL_COM_PWR_12V (0x03) : Set to 12V

Return Value:

Fail: 1

Success: 0

• DLLExport int SetCashDrawer1(int mode)

This function opens the Cash Drawer GPIO 1

Parameter:

int mode: CTL_CASH_OUT_HIGH_200MS (0x06): Open the cash drawer

Return Value:

Fail: 1

Success: 0

- DLLExport int SetCashDrawer2(int mode)
- This function opens the Cash Drawer GPIO 2

Parameter:

int mode:

CTL_CASH_OUT_HIGH_200MS (0x06): Open the cash drawer

Return Value:

Fail: 1

Success: 0

• DLLExport int SetCashDrawerPwrSel(int pwrmode)

This function set the cash drawer power level

Parameter:

int pwrmode:

CTL_CASH_PWRSEL_12V (0x07): Set to 12V

CTL_CASH_PWRSEL_24V (0x08) : Set to 24V

Return Value:

Fail: 1

Success: 0

• DLLExport int GetCashDrawerStatus(BYTE *byStatus)

This function gets the cash drawer status (Open or Close) Parameter:

BYTE *byStatus: *byStatus = 0: Close

*byStatus = 1: Open
Return Value:
Fail: 1
Success: 0
 DLLExport int SetSomReset(int mode)
This function reset the SOM Android system
Parameter:
int mode: Don't care (Set to 0~255)
Return Value:
Fail: 1
Success: 0
 DLLExport int SetSmartCom (int mode)
This function set the Smart COM enable or bypass
Parameter:
int mode:
CTL_SMART_COM_BYPASS (0x13): Set Smart COM to bypass
CTL_SMART_COM_SMART_COM (0x14): Set Smart COM enable
Return Value:
Fail: 1
Success: 0
 DLLExport int GetAllStatus(BYTE *byStatus)
This function gets all the GPIO pin status on IO Board
Parameter:
BYTE *byStatus:
0 = low, 1 = high
byStatus [0], bit0> COMA_MODE
byStatus [0], bit1> COMA_PWR
bystatus [0], bit2> COMB_MODE
byStatus [0], bit3> COMB_PWR byStatus [0], bit4 -> Cash Drawer CDIO0
bystatus [0], bit4> CashDrawer_GPIO0 bystatus [0], bit5 -> CashDrawer GPIO1
byStatus [0], bits> CashDiawei_GFIOT
byStatus [0], bito> CASH_FWIGEL
byStatus [0], bit -> SOM RST
byStatus [1], bit1> TYPEC FW RS
byStatus [1], bit2> SMART COM
Return Value:
Fail: 1
Success: 0
 DLLExport int GetFwVersion(BYTE * byVersion)
This function gets the FW version for IO Board FW.
byStatus [0] \rightarrow High byte of the FW version number
byStatus [1] \rightarrow Low byte of the FW version number
Return Value:
Fail: 1
Success: 0

4. Example for DLL Interface

FEC_IOBoard_Utility: Please refer the sample workspace: "FEC_IOBoard_Utility" create by Visual Studio 2015 for the sample code.

The screenshot for FEC_IOBoard_Utility:



C#:

[DllImport("fec_xpos_ioboard_dll", CharSet = CharSet.Unicode)]
public static extern int SetComAMode(int mode);
const int CTL_COM_MODE_RI = 0x00;
const int CTL_COM_MODE_DC = 0x01;
SetComAMode(CTL_COM_MODE_RI);
SetComAMode(CTL_COM_MODE_DC);

5. EXE Interface for FEC Cash Drawer SDK

Enable COM Port A Power

Set the COM Port A Power Enable with the parameter 1: "SetComAMode" & parameter 2: "enable" or "disable"

> FEC_XPOS_IOBoard_Tester.exe SetComAMode enable

> FEC_XPOS_IOBoard_Tester.exe SetComAMode disable

Enable / Disable the COM Port B Power

Set the COM Port B Power Enable / Disable with the parameter 1: "SetComBMode" & parameter 2: "enable" or "disable"

> FEC_XPOS_IOBoard_Tester.exe SetComBMode enable

> FEC_XPOS_IOBoard_Tester.exe SetComBMode disable

Set the COM Port A Power level

Set the COM Port A Power to 5v / 12v with the parameter 1: "SetComAPwr" & parameter 2: "5v" or "12v"

> FEC_XPOS_IOBoard_Tester.exe SetComAPwr 5v

> FEC_XPOS_IOBoard_Tester.exe SetComAPwr 12v

Open the Cash Drawer 1

Open the Cash Drawer 1 with the parameter 1: "SetCashDrawer1" & parameter 2: "activate"

> FEC_XPOS_IOBoard_Tester.exe SetCashDrawer1 activate

Open the Cash Drawer 2 Open the Cash Drawer 2 with the parameter 1: "SetCashDrawer2" & parameter 2: "activate" > FEC_XPOS_IOBoard_Tester.exe SetCashDrawer2 activate Set the Cash Drawer Power Level Set the Cash Drawer Power Level to 12v/24v with the parameter 1: "SetCashDrawerPwrSel" & parameter 2: "12v" or "24v" > FEC_XPOS_IOBoard_Tester.exe SetCashDrawerPwrSel 12v > FEC_XPOS_IOBoard_Tester.exe SetCashDrawerPwrSel 12v > FEC_XPOS_IOBoard_Tester.exe SetCashDrawerPwrSel 24v Get the Cash Drawer Status Get the Cash Drawer Status with the parameter : "GetCashDrawerStatus" > FEC_XPOS_IOBoard_Tester.exe GetCashDrawerStatus CashDrawer Status = 0 means close CashDrawer Status = 1 means open

Reset the SOM Reset the SOM Android system with the parameter : "SetSomReset" > FEC_XPOS_IOBoard_Tester.exe SetSomReset

Set the Smart COM enable/bypass

Set the Smart COM enable/bypass with the parameter 1: "SetSmartCom" & parameter 2: "enable" or "disable"

> FEC_XPOS_IOBoard_Tester.exe SetSmartCom enable> FEC_XPOS_IOBoard_Tester.exe SetSmartCom disable

4. Software

Developers Guide for Serial Communications

This document is intended for developers creating products based on the CP210x USB to UART Bridge Controller. It provides information about serial communications and how to obtain the port number for a specific CP210x device. Code samples are provided for opening, closing, configuring, reading, and writing to a COM port. Also included is a Get PortNum function that can be copied and used to determine the port number on a CP210x device by using its Vendor ID (VID), Product ID (PID), and serial number.

Opening a COM Port

Before configuring and using a COM port to send and receive data, it must first be opened. When a COM port is opened, a handle is returned by the CreateFile() function that is used from then on for all communication. Here is example code that opens COM3:

```
HANDLE hMasterCOM = CreateFile("\\\\\\COM3",
GENERIC_READ | GENERIC_WRITE,
0,
0,
0,
OPEN_EXISTING,
FILE_ATTRIBUTE_NORMAL | FILE_FLAG_OVERLAPPED,
0);
```

The first parameter in the **CreateFile()** function is a string that contains the COM port number to use. This string will always be of the form **\\\\.\COMX** where 'X' is the COM port number to use. The second parameter contains flags describing access, which will be **GENERIC_READ** and **GENERIC_WRITE** for the example in this document, and allows both read and write access. Parameters three and four must always be 0, and the flag in parameter five must always be **OPEN_EXISTING** when using **CreateFile()** for COM applications. The sixth parameter should always contain the **FILE_ATTRIBUTE_NORMAL** flag. In addition, the **FILE_FLAG_OVERLAPPED** is an optional flag that is used when working with asynchronous transfers (this option is used for the example in this document). If overlapped mode is used, functions that read and write to the COM port must specify an OVERLAPPED structure identifying the file pointer, which is demonstrated in the sections **Purging the COM Port** and **Saving the COM Port's Original State** (more information on overlapped I/O is located at <u>https://msdn.microsoft.com/en-us/library/windows/desktop/ms686358(v=vs.85).aspx</u>). The seventh, and last, parameter must always be 0.

If this function returns successfully, then a handle to the COM port will be assigned to the HANDLE variable. If the function fails, **then INVALID_HANDLE_VALUE** will be returned. Upon return, check the handle and if it is valid, then prepare the COM port for data transmission.

Preparing an Open COM Port for Data Transmission

Once a handle is successfully assigned to a COM port, several steps must be taken to set it up. The COM port must first be purged and its initial state should be retrieved. Then the COM port's new settings can be assigned and set up by a device control block (DCB) structure (more information is provided on the DCB structure in the section **Setting up a DCB Structure to Set the New COM State** and at <u>https://msdn.microsoft.com/en-us/library/windows/desktop/aa363214(v=vs.85).aspx</u>).

Purging the COM Port

First, the COM port should be purged to clear any existing data going to or from the COM port using the **PurgeComm()** function:

PurgeComm(hMasterCOM, PURGE_TXABORT | PURGE_RXABORT | PURGE_TXCLEAR | PURGE_RXCLEAR);

The first parameter in the **PurgeComm()** function is a handle to the open COM port that will be purged. The second parameter contains flags that further describe what actions should be taken. All

four flags, **PURGE_TXABORT, PURGE_RXABORT, PURGE_TXCLEAR**, and **PURGE_RXCLEAR** should always be used. The first two flags terminate overlapped write and read operations, and the last two flags clear the output and input buffers.

If this function returns successfully then a non-zero value is returned. If the function fails, then it returns 0. Upon return, check the return value; if it is non-zero, continue to set up the COM port (more information on the PurgeComm() function is located at <u>https://msdn.microsoft.com/en-us/library/windows/desktop/aa363428(v=vs.85).aspx</u>).

Saving the COM Port's Original State

Since the COM port settings can be modified to meet different needs, it is good practice to obtain the COM port's current state and

store it so that when the COM port is closed, the COM port can be restored back to its original state. This can be done using the **GetCommState()** function:

DCB dcbMasterInitState; GetCommState(hMasterCOM, &dcbMasterInitState);

The first parameter in the **GetCommState()** function is a handle to the open COM port to obtain settings from. The second parameter is an address to a DCB structure to store the COM port's settings. This DCB structure should also be used as the initial state when specifying new settings for the COM port (see section **Setting up a DCB Structure to Set the New COM State**). If this function returns successfully then a non-zero value is returned. If the function fails, then it returns 0. Upon return, check the return value; if it is non-zero, continue to set up the COM port (more information on the **GetCommState()** function is located at <u>https://msdn.microsoft.com/enus/library/windows/desktop/aa363260(v=vs.85).aspx</u>).

Setting up a DCB Structure to Set the New COM State

All of a COM port's settings are stored in a DCB structure. In section Saving the COM Port's Original State a DCB structure was retrieved that contained the initial settings of the COM port by using the **GetCommState()** function. To change a COM port's settings, a DCB structure must be created and filled out with the desired settings. Then the **SetCommState()** function can be used to activate those settings:

DCB dcbMaster = dcbMasterInitState;

dcbMaster.BaudRate = 57600; dcbMaster.Parity = NOPARITY; dcbMaster.ByteSize = 8; dcbMaster.StopBits = ONESTOPBIT;

SetCommState(hMasterCOM, &dcbMaster);

Delay(60);

Here a new DCB structure dcbMaster has been initialized to **dcbMasterInitState**, which are the current settings of the COM port. After it has been initialized to the current settings, new settings can be assigned.

Baud Rate

The baud rate property is set to 57600 bps, but can be set to any of the baud rates supported by the CP210x. (See the current datasheet for the list of supported baud rates for the CP210x.)

Parity

The parity is set to NOPARITY, however it can also be set to ODDPARITY, EVENPARITY, SPACEPARITY, and MARKPARITY if supported by the CP210x. (See the current data sheet for the list of supported parities for the CP210x.)

Byte Size

The byte size is set to 8, so there are 8 data bits in every byte of data sent. This can also be set to 5, 6, or 7 if supported by the

CP210x. (see the data sheet for the list of supported byte sizes for the CP210x.)

Stop Bits

The stop bits are set to ONESTOPBIT, but could also be set to TWOSTOPBITS or ONE5STOPBITS (1.5). (See the current data sheet for the list of supported stop bits for the CP210x.) All combinations of data and stop bits can be used except for the combination of 5 data bits with 2 stop bits and the combination of 6, 7, or 8 data bits with 1.5 stop bits. After each of these settings is set to the desired value, the **SetCommState()** function can be called to set up the COM port. The first parameter in the **SetCommState()** function is a handle to the open COM port to change the settings on. The second parameter is an address to a DCB structure containing the COM port's new settings (more information on serial settings using DCB structures is located at https://msdn.microsoft.com/en-us/library/windows/desktop/aa363214(v=vs.85).aspx

If this function returns successfully, a non-zero value is returned. If the function fails, it returns 0. Upon return, check the return value; if it is non-zero, delay for 60 ms to allow time for the settings to change and then continue to set up the COM port. This delay is not required; however, a conservative time of 60 ms is good practice to ensure that the settings are changed before any other operations take place.

Transmitting Data Across the COM Port

Once the COM port is successfully opened and configured, data can be written or read.

Writing Data

There are several things that need to happen in a write, so it is a good idea to create a function for the writes to be called whenever a write must occur. Here is an example of a write function:

```
bool WriteData(HANDLE handle, BYTE* data, DWORD length, DWORD* dwWritten)
        bool success = false;
        OVERLAPPED o = \{0\};
        o.hEvent = CreateEvent(NULL, FALSE, FALSE, NULL);
        if (!WriteFile(handle, (LPCVOID)data, length, dwWritten, &o))
                if (GetLastError() = = ERROR IO PENDING)
                        if (WaitForSingleObject(o.hEvent, INFINITE) == WAIT_OBJECT_0)
                                if (GetOverlappedResult(handle, &o, dwWritten, FALSE))
                                success = true;
        }
        Else
               success = true;
        if (*dwWritten != length)
                success = false;
        CloseHandle(o.hEvent);
        return success;
```

The parameters passed in to this function are the handle to an open COM port, a pointer to an array of bytes that will be written, the number if bytes that are in the array, and a pointer to a variable to store and return the number of bytes written. Two local variables are declared at the beginning of the function: a bool named success that will store the success of the write (this is initialized to false, and only set true when the write succeeds) and an overlapped object o which is passed to the **WriteFile()** function and alerts if the transfer is complete or not (this is always initialized to {0} before the **hEvent** is assigned). Creating an event with the **CreateEvent (NULL, FALSE, FALSE, NULL)** function sets the **hEvent** property of o to prepare it to be passed to the **WriteFile()** function (more information on CreateEvent() is located at <u>https://msdn.microsoft.com/en-us/library/windows/desktop/ms682396(v=vs.85).aspx</u>).

Next, the **WriteFile()** function is called with the handle, data, length of the data, and variable to store the amount of data that was written (more information on **WriteFile()** is located at https://msdn.microsoft.com/en-us/library/windows/desktop/aa365747(v=vs.85).aspx). If this function returns successfully, a non-zero value is returned. If the function fails, it returns 0. The if statement will determine if the write succeeded and if it did not, the last error is retrieved to see if there really was an error or the write just wasn't finished. If **ERROR_IO_PENDING** is returned then object o is then waited on until either the write finishes or fails (if something other than **ERROR_IO_PENDING** is returned by the **GetLastError()** function, then there is the possibility of surprise removal; see section

Application Design Notes for comments on surprise removal). When the wait is over, the result is obtained so that the amount of bytes written is updated. The success variable will then be assigned with the appropriate value, and the handle of **o.hEvent** is closed. Then the amount of bytes written is checked, and finally the function returns the success of the write, which will be true if the write successfully completed.

Reading Data

There are several things that need to happen in a read, so it is a good idea to create a function for the reads to be called whenever a read must occur. Here is an example of a read function: bool ReadData(HANDLE handle, BYTE* data, DWORD length, DWORD* dwRead, UINT timeout)

```
bool success = false;
OVERLAPPED o = {0};
o.hEvent = CreateEvent(NULL, FALSE, FALSE, NULL);
if (!ReadFile(handle, data, length, dwRead, &o))
{
    if (GetLastError() == ERROR_IO_PENDING)
        if (WaitForSingleObject(o.hEvent, timeout) == WAIT_OBJECT_0)
            success = true;
        GetOverlappedResult(handle, &o, dwRead, FALSE);
}
else
    success = true;
CloseHandle(o.hEvent);
return success;
```

The parameters passed in to this function are the handle to an open COM port, a pointer to an array of bytes that will be read, the number if bytes that are in the array, a pointer to a variable to store and return the number of bytes read, and a timeout value. Two local variables are declared at the beginning of the function: a bool named success that will store the success of the read (this is initialized to false, and only set true when the read succeeds), and an overlapped object o which is passed to the **ReadFile()** function and alerts if the transfer is complete or not (this is always initialized to {0} before the **hEvent** is assigned). Creating an event with the **CreateEvent(NULL, FALSE, FALSE, NULL)** function sets the **hEvent** property of o to prepare it to be passed to the ReadFile() function (more information on **CreateEvent()** is located at

https://msdn.microsoft.com/enus/library/windows/desktop/ms682396(v=vs.85).aspx).

Next, the **ReadFile()** function is called with the handle, data, length of the data, and variable to store the amount of data that was written (more information on the **ReadFile()** function is located at <u>https://msdn.microsoft.com/en-us/library/windows/desktop/aa365467(v=vs.85).aspx</u>). If this function returns successfully then a non-zero value is returned. If the function fails, then it will return 0.

The if statement will determine if the write succeeded and if it didn't, the last error is retrieved to see if there really was an error or the write just wasn't finished. If **ERROR_IO_PENDING** is returned then object o is then waited on until either the write finishes or fails (if something other than **ERROR_IO_PENDING** is returned by the **GetLastError()** function, then there is the possibility of surprise removal; see section **Application Design Notes** for comments on surprise removal). When the wait is over, the result is obtained so that the amount of bytes read is updated. The success variable will then be assigned with the appropriate value, and the handle of **o.hEvent** is closed. Finally, the function returns the success of the read, which will be true if the read successfully completed.

Closing the COM Port

After all communication is finished, then the COM port should then be closed. First, the COM port should be set back to its initial state, and then the handle to the COM port should be closed and set to an invalid handle. Example code is shown below:

SetCommState(hMasterCOM, &dcbMasterInitState);

Delay(60);

CloseHandle(hMasterCOM); hMasterCOM = INVALID_HANDLE_VALUE;

The SetCommState() function works the same as described in section **Setting up a DCB Structure to Set the New COM State**. A delay of 60 ms is used to make sure the settings have time to be set. Finally the device is closed using the **CloseHandle()** function. This function just takes in the handle of the COM port. After this function is called, it is important to set the variable to an **INVALID_HANDLE_VALUE**.

Sample Program to Demonstrate Serial Communications

Included in the AN197 software package is a directory named CP210xSerialTest which contains the source code and executables for a Visual Studio project that makes use of all the serial communication functions described in section **Preparing an Open COM Port for Data Transmission**, section **Transmitting Data Across the COM Port**, and section **Closing the COM Port**. The program is a basic dialog based application that accepts two COM port numbers, and then will send a test array of 64 bytes of data back and forth between them.

Application Design Notes

The functions used in sections Preparing an Open COM Port for Data Transmission, Transmitting Data Across the COM Port, and section Closing the COM Port are Windows COMM API functions. The examples provided are just samples of the recommended way of dealing with serial communication. For more specific information on these functions, see the MSDN website at:

https://msdn.microsoft.com/enus/library/ff802693.aspx.

It should also be noted that the **SetCommState()** function does not save the settings between opening and closing the COM port. As stated before, it is good practice to get the current settings after the COM port is opened, and then restore them before it is closed. All of the functions here will return an error code. It is a good idea to nest these functions in order to catch errors if they occur by using the **GetLastError()** function. This will also solve any surprise removal problems by allowing the discovery of an invalid handle to be found and dealt with. The example application (CP210xSerialTest) has several cases that will detect surprise removal. In this example, there are checks on every function to make sure that the return code is true. If it is not, then it will display where the error occurred in the output window. As long as correct and supported settings are passed to the functions they should execute normally. Most failures can occur from having an **INVALID_HANDLE_VALUE**, however, the handles must be set to this value after a surprise removal occurs. Because regular COM ports will always be visible, then data can always be written to them successfully, even if there is no way to read it. However, because the CP210x is a virtual COM port, if the device is removed, then the handle that it uses becomes invalid when trying to write to it. If for some reason the CP210x device is unplugged the write will fail and ERROR OPERATION ABORTED will be returned by GetLastError(). When this happens, the handle needs to be closed and then set to **INVALID HANDLE VALUE**. Alternatively, a regular COM port can always be read from, but if there is no data then it will time out. When using the CP210x as the virtual COM port and it is removed before a read occurs, then the read will fail and ERROR ACCESS DENIED will be returned by GetLastError().

Again when this happens, the handle needs to be closed and then set to **INVALID_HANDLE_VALUE**.

Enable and Disable Ambient Light and Proximity Sensor

There are three ways to enable or disable the 2in1 sensor.

1. Utility Batch File

File is located in your Driver CD



2. BIOS

- 1. Start up the system
- 2. Press [Delete] during startup to enter BIOS

3. Under Advanded > Sensor Device Configuration click Disabled to turn off, Sensor 1 to turn on proximity and anmbient light sensor. Sensor 2 has no function.

Note: Sensor Service and BIOS Sensor settings both need to be on to work



3. Windows 10 OS

1. Under Computer Management go under Services and Applications to click on [Services]

2. Click on [Sensor Service]

Computer Management							x I
ile Action View Help							
• 🔿 🙍 📆 🖕							
Computer Management (Local	O. Services	Actions					
System Tools	Sensor Service	Name	Description	Status	Startu ^	Services	-
> Vent Viewer		🧠 Retail Demo Service	The Retail D		Manu	More Actions	,
> 😥 Shared Folders	Start the service	Routing and Remote Access	Offers routi		Disabl	Sentor Sentice	
> 👰 Local Users and Groups		RPC Endpoint Mapper	Resolves RP	Running	Autor	Mars Asting	
> (N) Performance	Description:	Secondary Logon	Enables star		Manu	More Actions	,
Device Manager	different sensors' functionality.	Secure Socket Tunneling Pr	Provides su	0	Manu		
Disk Management	Manages Simple Device Orientation	Security Accounts Manager	The WSCSV	Running	Autor		
Services and Applications	(SDO) and History for sensors. Loads the SDO sensor that reports device	Sensor Data Sensice	Deliver: dat	Kunning	Manu		
G Services	orientation changes. If this service is	Sensor Monitoring Service	Monitors va		Autor		
🗃 WMI Control	stopped or disabled, the SDO sensor	Sensor Service	A service fo		Manu		
	rotation will not occur. History	🖏 Server	Supports fil	Running	Autor		
	collection from Sensors will also be	🥋 Shared PC Account Manager	Manages pr		Disabl		
	stopped.	🥋 Shell Hardware Detection	Provides no	Running	Autor		
		Smart Card	Manages ac		Disabl		
		Smart Card Device Enumera	Creates soft		Manu		
		Smart Card Removal Policy	Allows the s		Manu		
		SNMP Trap	Receives tra		Manu		
		Software Protection	Enables the		Autor		
		Spot Verifier	Verifies pote		Manu		
		SSDP Discovery	Discovers n	Running	Manu		
		Still Image Acquisition Events	Provides re	Kunning	Manu		
		Storage Service	Drovides en	Running	Manu		
		Storage Tiers Management	Ontimizes t	riannig	Manu		
		Superfetch	Maintains a	Running	Autor		
		Sync Host_2ebd6	This service	Running	Autor		
		System Event Notification S	Monitors sy	Running	Autor 🗸		
		<			>		
>	Extended Standard					1	

3. Sensor Services Properties make sure the "Startup type" is set to Automatic.

Click [Start] [Apply] [OK]

Sensor Service Pro	perties (Local Computer)	×				
General Log On	Recovery Dependencies					
Service name:	SensorService					
Display name:	Sensor Service					
Description:	A service for sensors that manages different sensors' functionality. Manages Simple Device Orientation					
Path to executabl	e:					
C:\Windows\syste	em32\svchost.exe -k LocalSystemNetworkRestricted	.				
Startup type:	Automatic ~					
Service status:	Stopped					
Start Stop Pause Resume						
You can specify the start parameters that apply when you start the service from here.						
Start parameters:						
	OK Cancel Apply					
		_				

Control the Sensor Timing

1. In the command prompt, type regedit and click enter



 Locate the following folder through this path HKEY_LOCAL_MACHINE >SOFTWARE > Microsoft > Windows NT > CurrentVersion > AdaptiveDisplayBrightness > {23B44AF2-78CE-4943-81DF-89817E8D23FD}

No

 \times

3. Click on CRI then the radio head decimal and change the number to the desired timing (ex 3000 is approximately 3 seconds, 10000 is around 10 seconds)

) Value 🛛 🗙
Base
Decimal
OK Cancel

4. The sensor needs to be deactivated then reactivated under Computer Management > Sensor Services

Computer Management File Action Very Help Computer Management File Action File Action File File Action File Action File Action File File Action File Act									
File Action Second	Computer Management						- 0	×	Sensor Service Properties (Local Computer)
Compared Management Locs Service Analyses Compared Management Locs Service Service Se	File Action View Help								
 Compared Musagement Loc V System Todes Service Service Service Service Service Service Service Service Service Service Service Service Service Service Service Service Service Service Service S	🔶 🏟 🙇 📷 🖾 🗟	2 📷 🕨 🗉 H 🕪							General Log On Recovery Dependencies
Construction C	🌆 Computer Management (Local	O. Services					Actions		
Berry Viewer Breit Steern's een Berry Viewer Breit Steern's een Berry Viewer Berry	System Tools Task Scheduler	Sensor Service	Name	Description	Status	Startu ^	Services	•	Service name: SensorService
Bardel older Construction C	> 🛃 Event Viewer	Start the service	Retail Demo Service	The Retail D		Manu	More Actions		Display name: Sensor Service
Berchanger Bordenmarker Bor	Shared Folders		RPC Endpoint Mapper	Resolves RP	Running	Autor	Sensor Service	•	
Consistent and a second state of the seco	> (S) Performance	Description: A service for sensors that manages	Secondary Logon Secure Socket Tunneling Pr	Enables star Provides su		Manu Manu	More Actions	•	Description: A service for sensors that manages different sensors' functionality. Manages Simple Device Orientation
Construction of the second secon	🗸 🔮 Storage	different sensors' functionality. Manages Simple Device Orientation	Security Accounts Manager	The startup	Running	Autor		_	
Sources and Applications He 5D cances the targets device Sources and Applications He 5D cances the targets device Sources and Applications He 5D cances the targets device Sources and Applications He 5D cances the targets device Sources and Applications He 5D cances the targets device Sources and Applications Sources and Applications He 5D cances the targets device Sources and Applications He 5D cances the targets device Sources and Applications He 5D cances the targets device Sources and Applications He 5D cances the targets device Sources and Applications Monu Sources and Applications Sources and Applications Monu	📅 Disk Management	(SDO) and History for sensors. Loads	Security Center	The WSCSV	Running	Autor			Path to executable:
 tropped or disability the SD service show will also be loaded and support of the show of	 Services and Applications Services 	the SDO sensor that reports device orientation changes. If this service is	Sensor Data Service	Delivers dat		Manu		_	C:\Windows\system32\sychost exe +k LocalSystemNetwork Restricted
Startup type: Automatic Super Service Super Servi	WMI Control	stopped or disabled, the SDO sensor	Sensor Monitoring Service	A senice fo		Manu			
Collection from Sensors will also be Source PC Account Manager PL Acc		will not be loaded and so auto-	Server	Supports fil	Running	Autor			Startup type: Automatic ~
 stepped. Shell Hardware Detection Provide no. Running Autor Smatt Carl Device frammase. Create solt. Manu Smatt Carl Device frammase. Create solt. Manu Shaft Tap Receives ta. Manu Shaft Tap Receives ta. Manu Shaft Tap State Repository Strike State Resume <		collection from Sensors will also be	Shared PC Account Manager	Manages pr		Disabl			
Smart Card Manages ac. Diabl Smart Card Smart		stopped.	🍓 Shell Hardware Detection	Provides no	Running	Autor			
Smart Card Device Funnema. Create soft. Manu Sing Card Removed Poly Share Protection Share Protection Share Protection Share Provider a. Running Manu Share Repository Strike Stare Repository Stare Re			🧠 Smart Card	Manages ac		Disabl			
Sinart Card Removal Policy Allows the s Manu Sinart Card Removal Policy Allows the s Manu Sinart Portection Source status: Stopped Start Stop Pause Resume Start Stop Pause Resume Stopped Start Stop Start Stop Start Stop Start Stop Start Stop Start Stop Start			Smart Card Device Enumera.	. Creates soft		Manu			
Shafe Trap G_Shafe Trap G_S			Smart Card Removal Policy	Allows the s		Manu			Service status: Stopped
Start Stop Pause Resume Start Stop Pause Resume You can specify the start parameters that apply when you start the service from here. Start Stop Start Stop Pause Resume You can specify the start parameters that apply when you start the service from here. Start Start Stop Pause Resume You can specify the start parameters that apply when you start the service from here. Start Start Start Stop Pause Resume You can specify the start parameters that apply when you start the service from here. Start Start Start Start Stop Pause Resume You can specify the start parameters that apply when you start the service from here. Start St			SNMP Trap	Receives tra		Manu			
Control C			Software Protection	Enables the		Autor			Start Stop Pause Resume
Store Duckovery Duckovery Duckovery New Provides re-, Ruming Manu State Store Duckovery Society and Socie			Spot Ventier	Verifies pote		Manu			Start Stop Flabac Flabac
State Apploating Service Provide Learning Autor State parameters that apply when you state the service from here. State parameters that apply when you state the service from here. State parameters that apply when you state the service from here. State parameters that apply when you state the service from here. State parameters that apply when you state the service from here. State parameters that apply when you state the service from here. State parameters that apply when you state the service from here. State parameters State parameters State parameters State parameters State parameters State parameters			Chate Beneritary Contine	Discovers n	Running	Manu			Yes and the state of the state
An Integra Agendance Versite Landing Manu Storage Tirres Management Optimizes La. Maning Manu Storage Tirres Management Optimizes La. Maning Mater Storage Tirres Management Optimizes Landing Auter Synce Host, Zebedis Start parameters: OK Cancel Apply OK Cancel Apply			Chill Image A socialities Event	Provides re	Kunning	Manu			rou can specify the start parameters that apply when you start the service
Start parameters: OK Cancel Apply			Storage Service	Provides en	Running	Manu			from here.
Superfect: Maintains a. Running Autor Synce Host, Zebdő This service a. Running Autor System Event Netification S Monitors syn. Superfect: Maintains a Running Autor Start parameters: OK Cancel Apply			Storage Tiers Management	Ontimizes t	nanning	Manu			
< Konchot.2bdb This service - Running Autor G System Event Nethfication S Monitors sys. Running Autor C Koncel OK Cancel Apply			Superfetch	Maintains a	Running	Autor			Start parameters:
< Autor Konded / Standard / OK Cancel Apply			Svnc Host 2ebd6	This service	Running	Autor			
< >> Etended (Standard / OK Cancel Apply			System Event Notification S	Monitors sy	Running	Autor 🗸			
Cancel Apply			<			>			
	< >	Extended Standard					1	_	OK Cancel Apply

Cash Drawer Command

Note: It is recommended that developers use the chapter 3: IO Board SDK Instruction

Command:

CashDrawer output 1: A5 01 05 02 6B Return A5 01 08 00 5B is Open A5 01 08 01 5B is Closed

CashDrawer output 2: A5 01 06 02 6B Return A5 01 08 00 6B is Open A5 01 08 01 6B is Closed