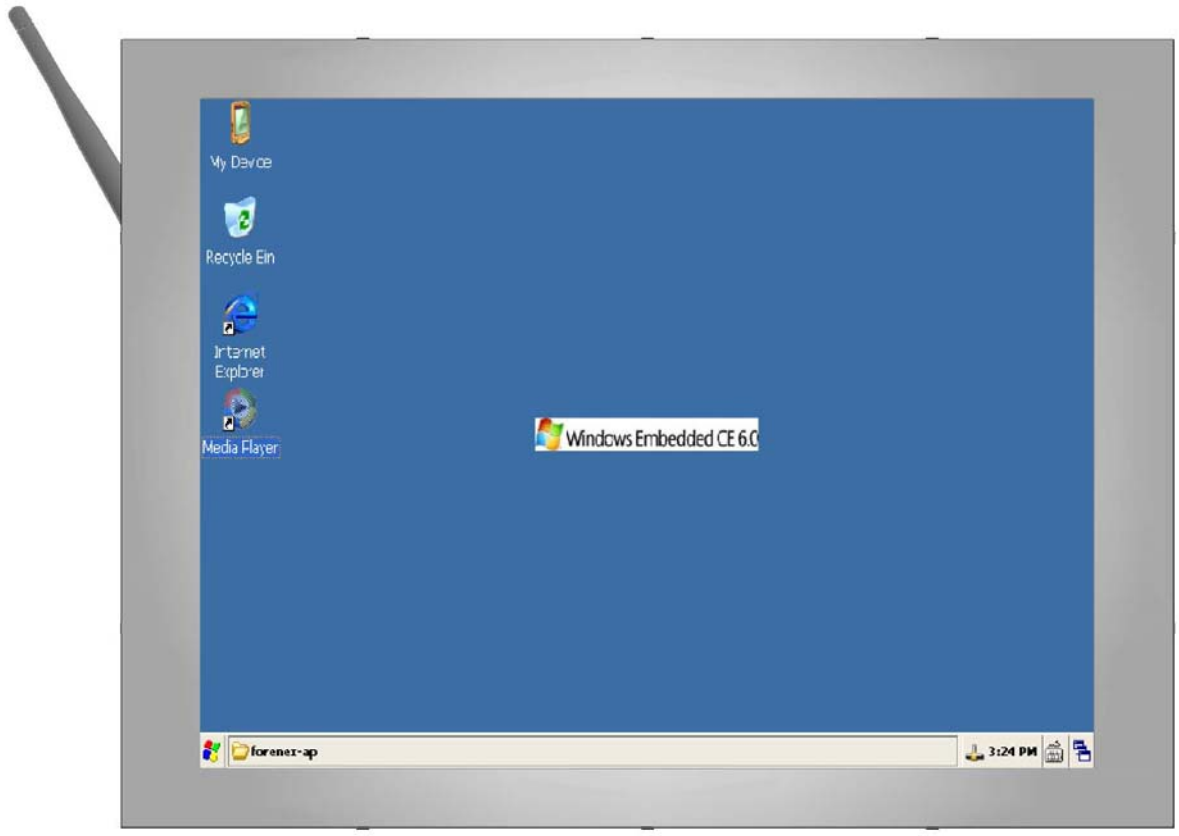
FES91W Series

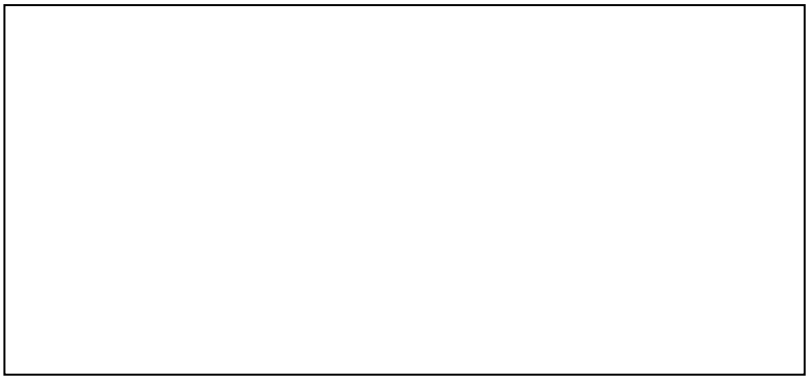
User's Manual (Preliminary)



Customer: _____

Customer Approved By: _____

Comment:



Module No:

Document No:_____

Version: **V2.3** _____

	Benson Wang	**Arthur Sung** **Hank Kuo**
Approved By	**Checked By**	**Prepared By**

Record of version

FES91W User's Manual

Version	Revise Date	Page	Description
V1.0	2011-03-24		First issue
V2.0	2011-07-30		1.Dimension changed more 1mm in (D) side.(page 9) 2.Add VESA -75 hole in rear side.(page 9) 3.Add reset hole in rear side.(page 22) 4.Speaker connector change location(page 29). 5.Add SPI description.(page 48)
V2.1	2011-08-22		1.Add Section 2-4.2 Serial port Rs485 test 2.Add Section 4-5.3 Example code
V2.2	2013-01-13		
V2.3	2013-06-04		1.Add PCL Printer Driver 2.Add SPI Write and Read of the lastTransfer parameter (page 42-43). **3. Re-define the pin assignment of J7 & J9**

The content of this document is subject to be change without notice

General Description (FES91W series)

•Order information

- OS type ID: (FES91 〝**W**〞 xx-xxx).
- FES91**W**xxx-xxx : Embedded WinCE 6.0.

Standard	Processor type	Memory size	Function option	Panel type
FES91Wxx-xxx	**FES91Wxx-xxx**	**FES91WXx-xxx**	**FES91WxX-xxx**	**FES91Wxx-XXX**
CPU:AT91SAM9M10	FES91Wxx-xxx			
64MB		FES91W**0**x-xxx		
128MB		FES91W**1**x-xxx		
WLAN			FES91Wx**A**-xxx	
CAN BUS			FES91Wx**B**-xxx	
WLAN & CAN BUS			FES91Wx**C**-xxx	
800x480				FES91Wxx-**070**
800x600				FES91Wxx-**080**
1024x768				FES91Wxx-**104**

•Packing List

•FES91W series device with metal housing.

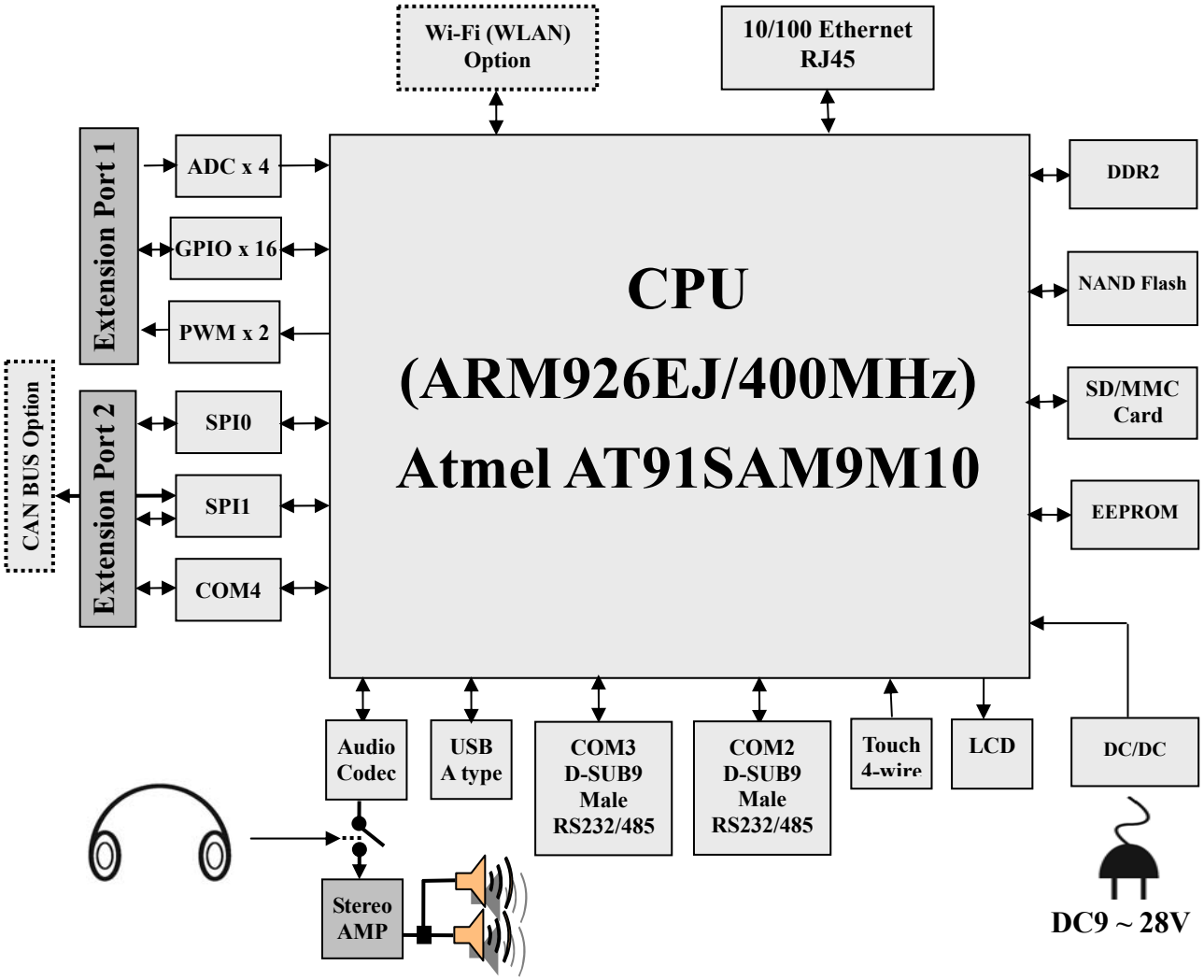
•Download Cable: USB-A male to mini-5p USB with 70cm length.
Numbering: WYUSBMINI500900

•Debug cable: header housing (2x5pin) to D_sub9 female connector.
Numbering: WH810B090173

•Utility software CD (including example program code).

1.Specifications (FES91W series)

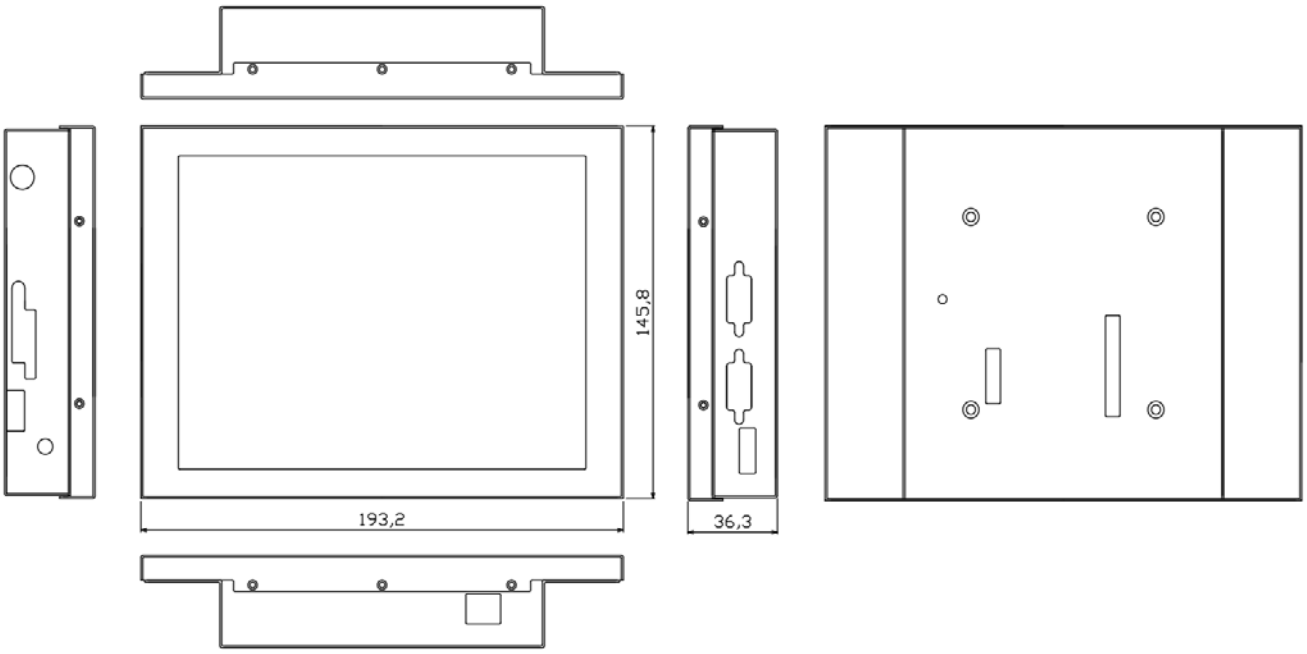
1.1 Function Block Diagram



1.2 System specifications

Parameter	Specification
CPU	■ Atmel AT91SAM9M10 @ 400MHz ■ 32 bit RISC architecture ARM926EJ core
System Memory	64MB or 128MB, DDR2 memory.
Storage Memory	2Gbit NAND Flash memory.
OS	■ Embedded WinCE 6.0.
LCD Panel	■ -070: 7" TFT LCD, 800x480, 400 cd/㎡, CR:500, 16M color. ■ -080: 8" TFT LCD, 800x600, 250 cd/㎡, CR:500, 16M color. ■ -104: 10.4" TFT LCD, 1024x768, 250 cd/㎡, CR:500, 16M color.
Video Hardware Decoder	Support MPEG1/2/4, H.264,VC-1 Hardware decoder. For FES91 series only
Touch	4-wired resistive type.
LAN	High performance RMII 10/100Mbps Ethernet controller.
Audio	■ AC97 Codec, AMP inside for Ext. Stereo speaker 8Ohm/1W. ■ Stereo Earphone Jack.
SD Slot	SD/MMC card.
Serial Port	■ 5 wire RS232 x 1 (COM2 connector sharing with RS485/422). ■ 3 wire RS232 x 1 (COM3 connector sharing with RS485/422). ■ 3 wire RS232 x 1 (COM4 located in Extension Port).
USB	■ USB 2.0 device x 1. ■ USB 2.0 Host x 1.
Extension Port1	■ Including programmable IO x 16 pin. ■ Including 10 bit high speed A/D converter channel x 4. ■ Including PWM channel x 2.
Extension Port2	■ Including SPI port x 2 (slave or master mode). ■ COM4 (3 wire RS232 only).
WiFi option	IEEE802.11 b/g/n, WiFi compliant (under development with an external dongle)
CAN BUS option	(under development with an external module)
Power Supply	DC9V ~ 28V single power input.
Operating Temperature	-20 ℃ ~ 80℃

1.3 Mechanical Dimension

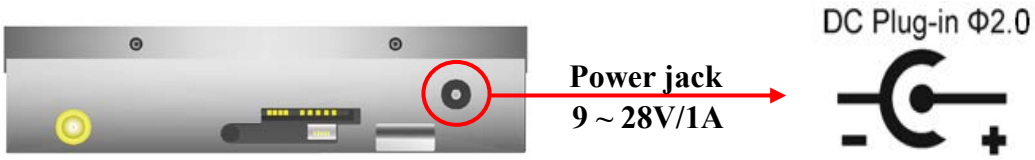


***Dimension Notice**

- **FES91xxx_080 with metal housing : 193.2(mm) x 145.8(mm) x 36.3(mm).**
- **FES91xxx_070 with metal housing : 178(mm) x 108(mm) x 34.4(mm).**

2. System installation (FES91W series)

2.1 Power Plugging



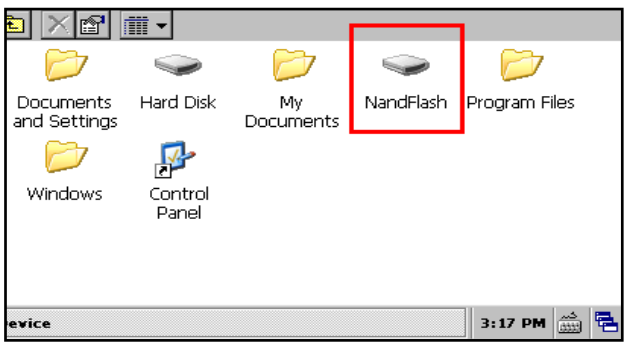
Or

2.2 Touch installation

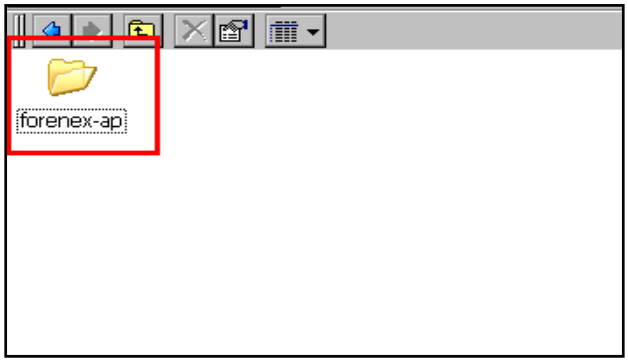
The following steps show how to use **Coordinate calibration.**

STEP 1. On Windows CE desktop, click on "**My Device**".

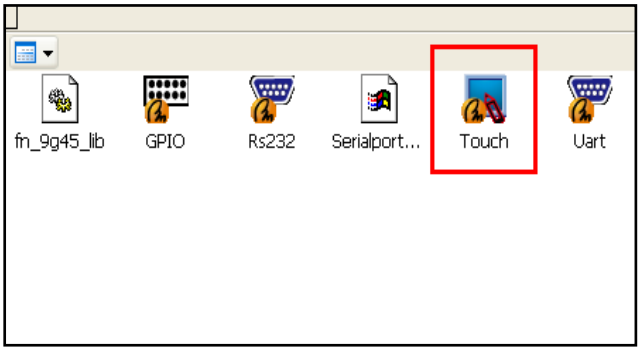
STEP 2. Select "**NandFlash**" folder.



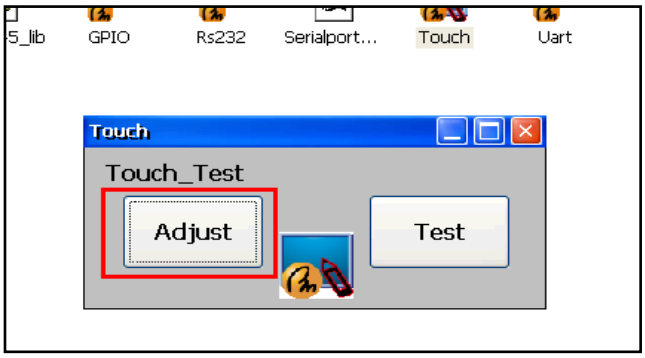
STEP 3. Select the "**forenex-ap**" folder.



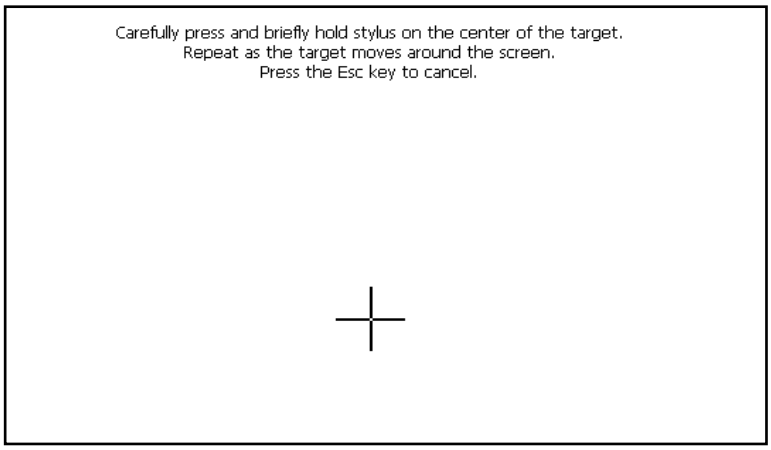
STEP 4. Click Touch icon, then Touch AP will display.



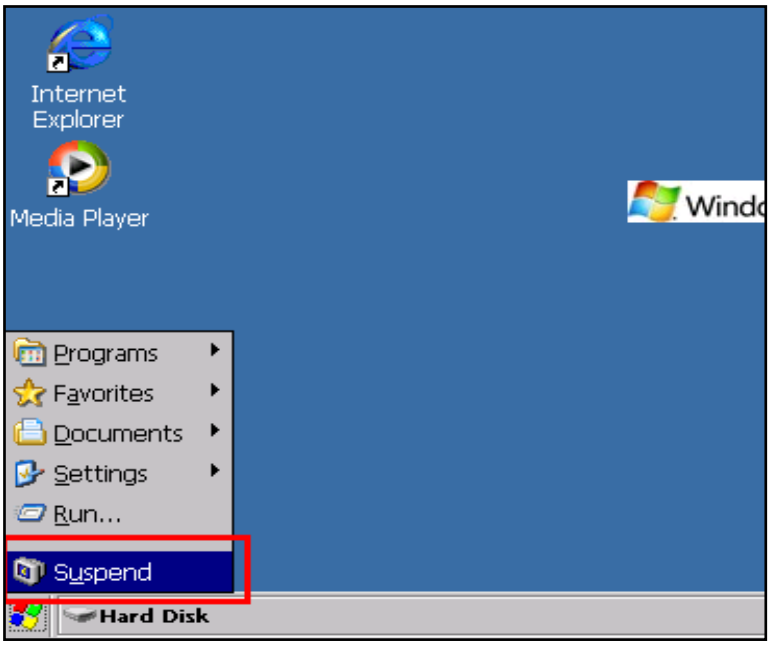
STEP 5. Click **"Adjust"** button.



This is WinCE Touch Calibration running in screen.

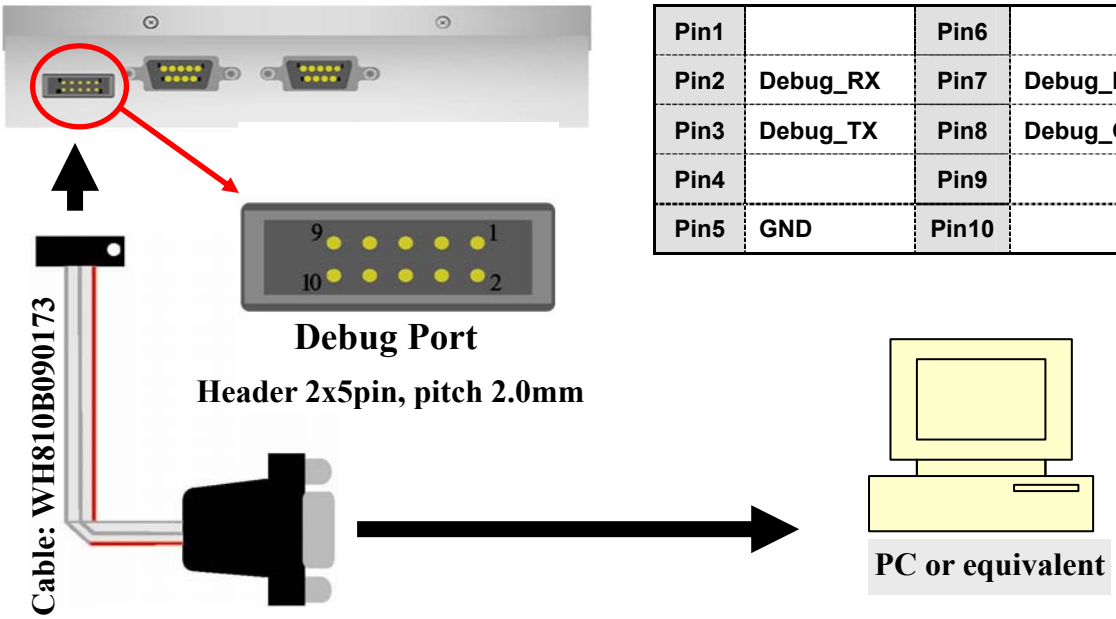


After using "Touch Calibration", system need reboot over suspend to store new touch parameter.



2.3 Debug Port installation

Debug port is only available for internal use. User's application program is unavailable.



Pin1		Pin6	
Pin2	Debug_RX	Pin7	Debug_RTS
Pin3	Debug_TX	Pin8	Debug_CTS
Pin4		Pin9	
Pin5	GND	Pin10	

Setup HyperTerminal

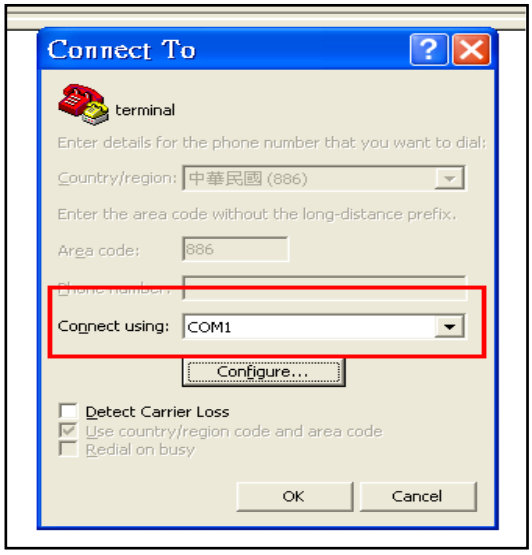
Using **HyperTerminal** on PC end can read information from FES91W or input command to FES91W.

●It was necessary that PC connect with FES91W**.**

The following steps show how to setup **HyperTerminal** on PC end.

STEP 1. Run the **HyperTerminal** on PC.

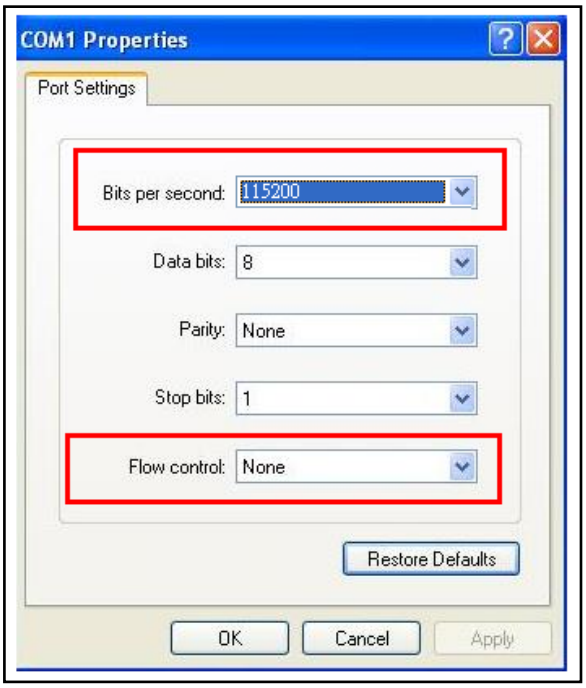
STEP 2. Select the applicable COM port, click OK.



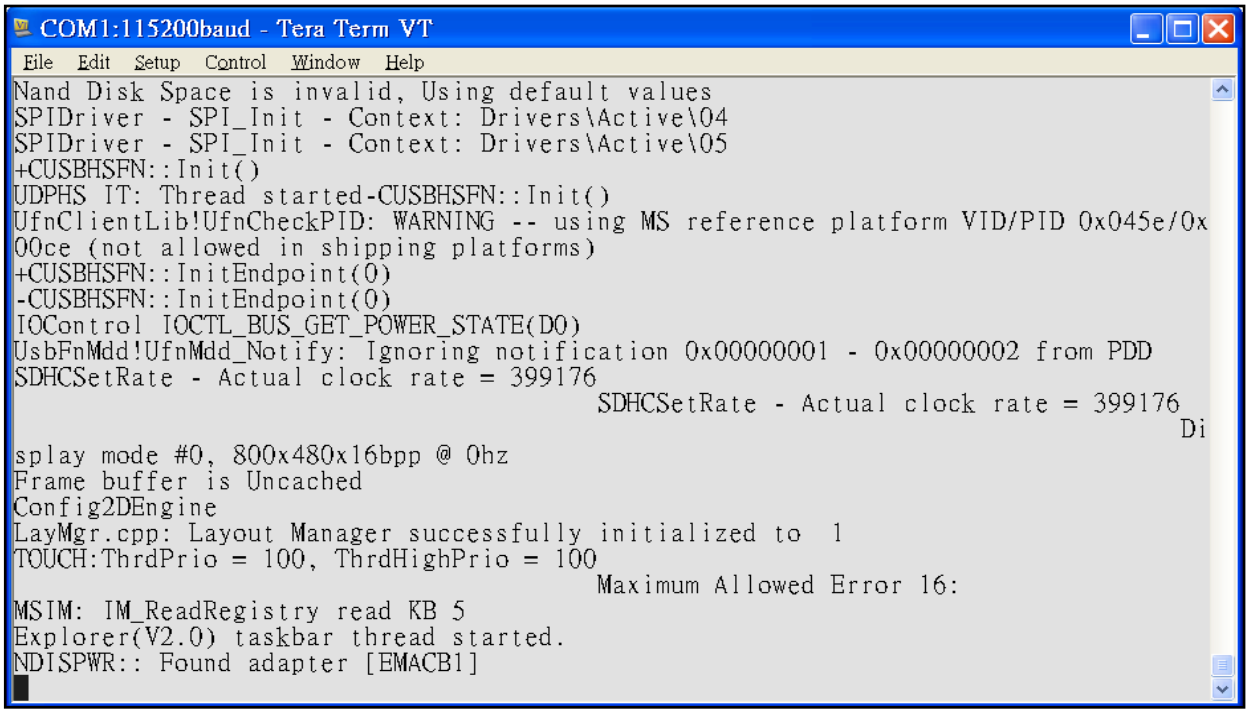
●To find applicable COM port, click **Control panel –> system –> hardware –> device manager –> COM&LPT ports.**

STEP 3. Port Settings

Bits per second (or **Baud rate**): 115200, **Data bits**: 8, **Parity**: None, **Stop bits**: 1, **Flow control**: None. Click Apply, and then OK.

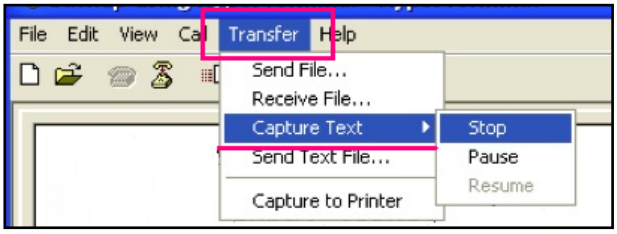


When PC connects to FES91W with cable, reboot FES91W, **PC-HyperTerminal** will display information from FES91W.



STEP 4. Save information from FES91W

On **HyperTerminal**, click Transfer -> Capture Text.



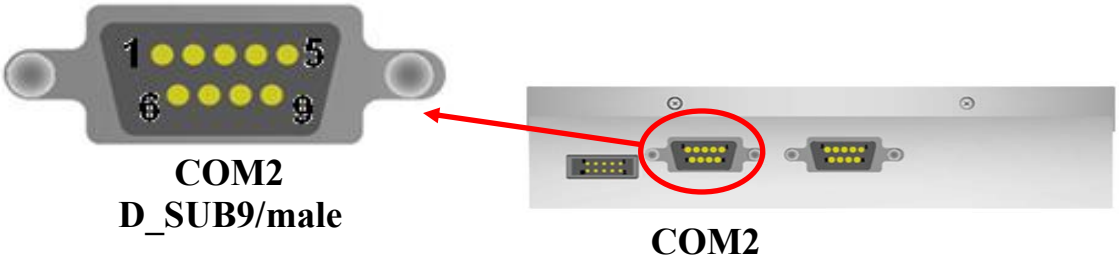
Then select where to save the information and name it and click Start.

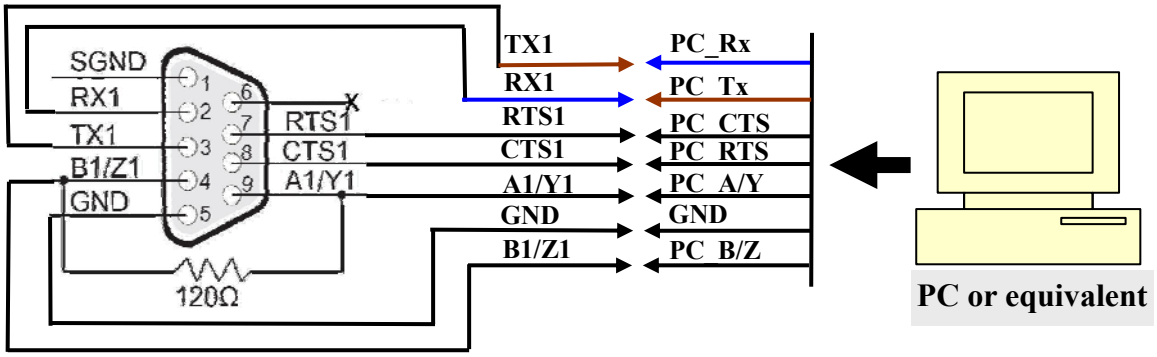
2.4 Serial port installation

FES91W supports three sets of RS232 or two sets of RS485 (option) or two sets of RS422 (option).

COM2

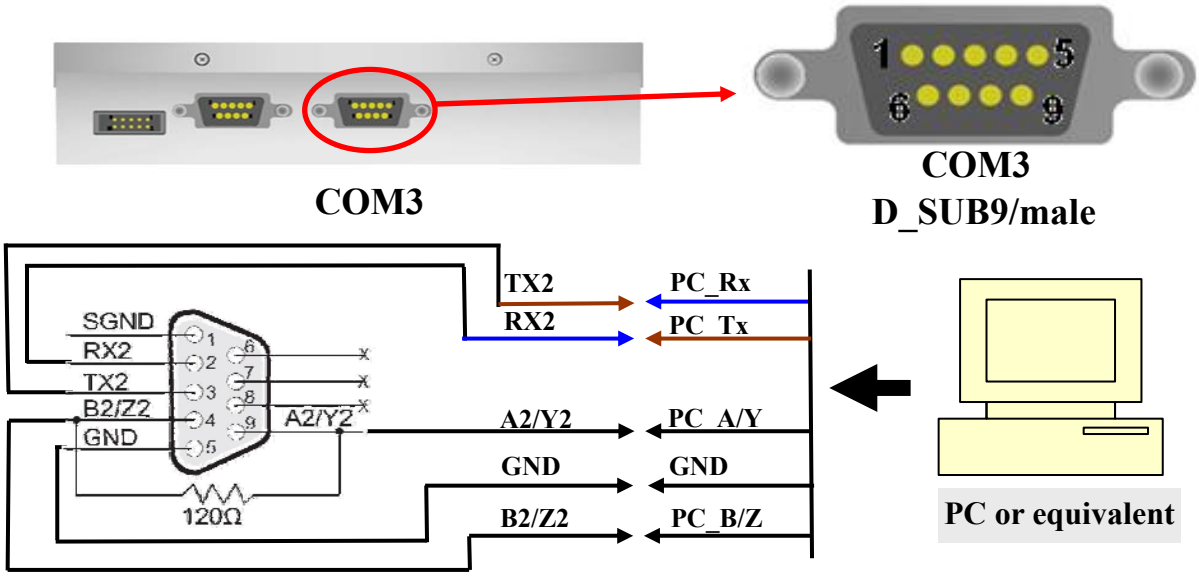
- **The port (COM2) provided 5-wire RS232 on D_SUB9-male connector.**
- **The connector D_SUB9 is shared with the signals of RS485/422.**
- **Due to a terminating resistor 120 Ohm has been installed inside. COM2 has to enable as master or the last slave device while COM2 be dedicated to work with mode RS485/422.**
- **Both working mode of RS232 and RS485/422 can not be enabled in simultaneously.**





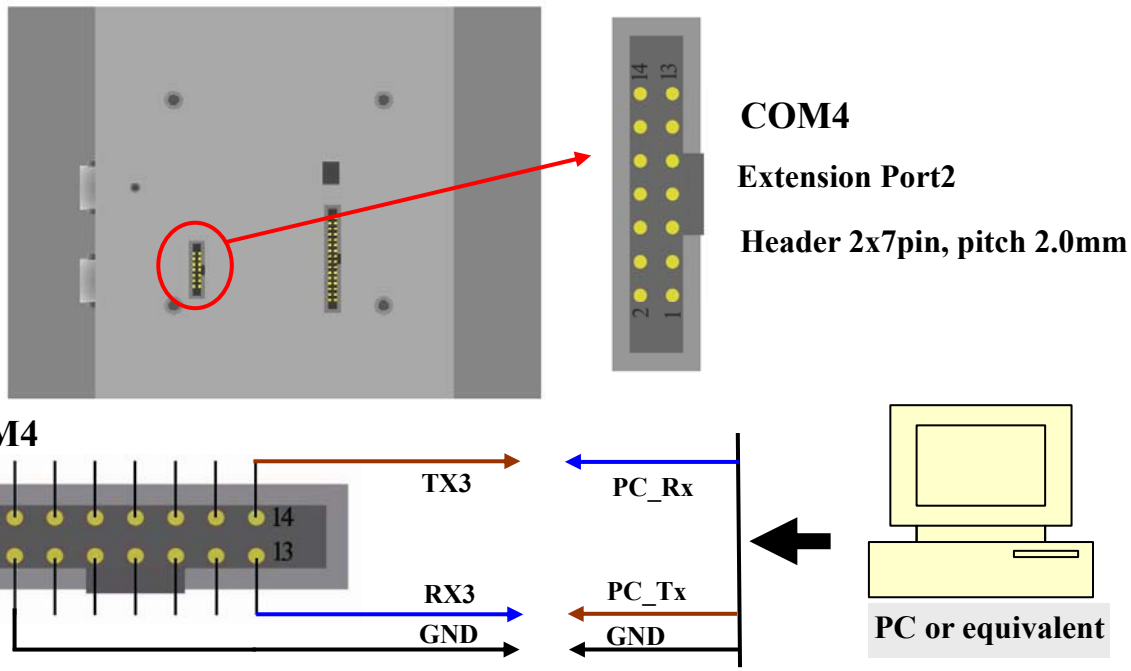
COM3

- The port (COM3) provided 3-wire RS232 on D_SUB9-male connector.
- The connector D_SUB9 is shared with RS485/422 format.
- Due to a terminating resistor 120 Ohm has been installed inside. COM3 has to enable as master or the last slave device while COM3 be dedicated to work in mode RS485/422.
- Working mode RS232 and RS485/422 can not be enable in simultaneously.



COM4

- The port (COM4) provided 3-wire RS232 on two pin headers of extension port2.

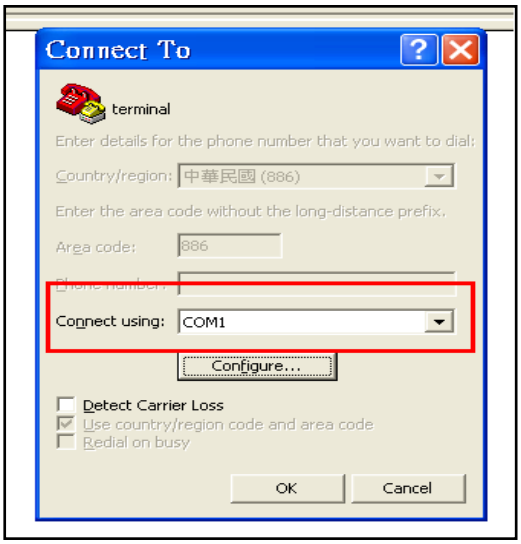


2-4.1> Serial port Rs232 test

The following steps show how to set HyperTerminal at PC end.

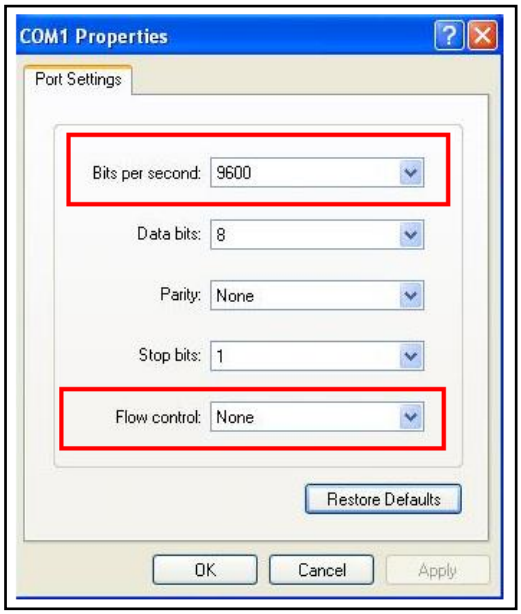
STEP 1. Run the hyper-terminal on PC.

STEP 2. Select the applicable COM port, click OK.



●To find applicable COM port, click Control panel –> system –> hardware –> device manager –> COM&LPT ports

STEP 3. Make sure the “**Bits per second**” (or Baud rate) ranges from 9600 to 115200 and ”**Flow control**” settings “**None**”. Click on apply, then OK.

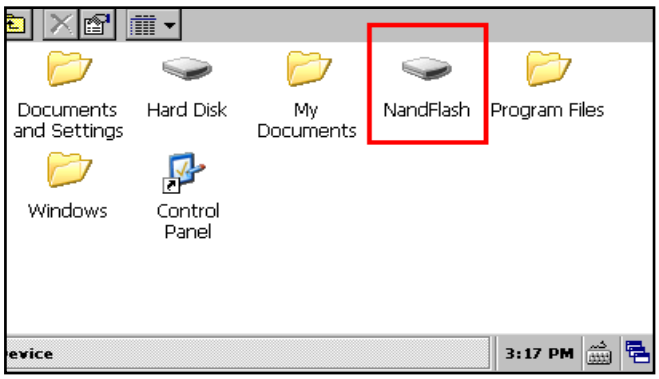


●**Steps of PC end is complete.**

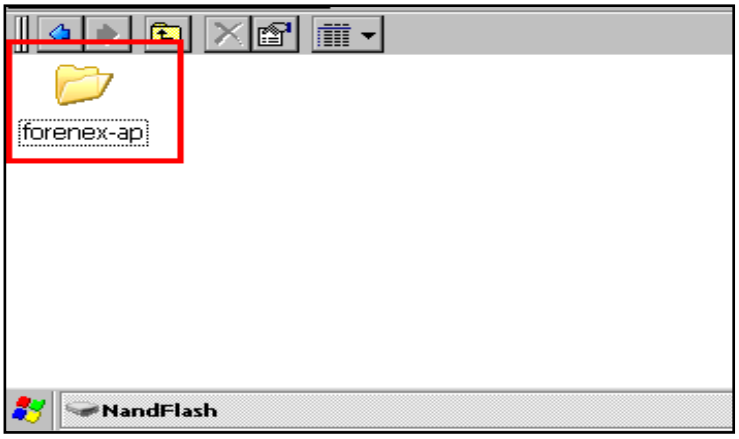
Below steps guide is use Serialport-ap at Windows CE.

STEP 1. On Windows CE desktop, click on “**My Device**”.

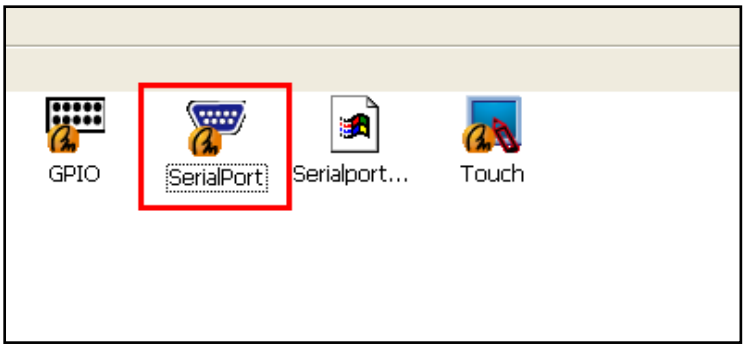
STEP 2. Select “**NandFlash**” folder.



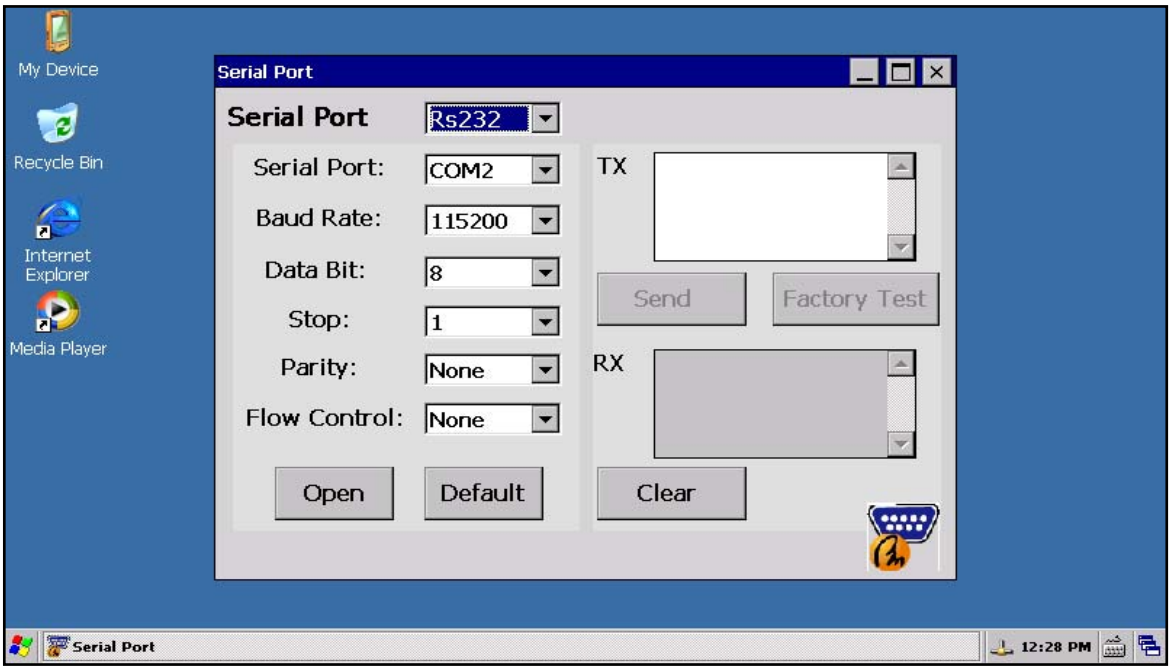
STEP 3. Select "**Forenex-ap**" folder.



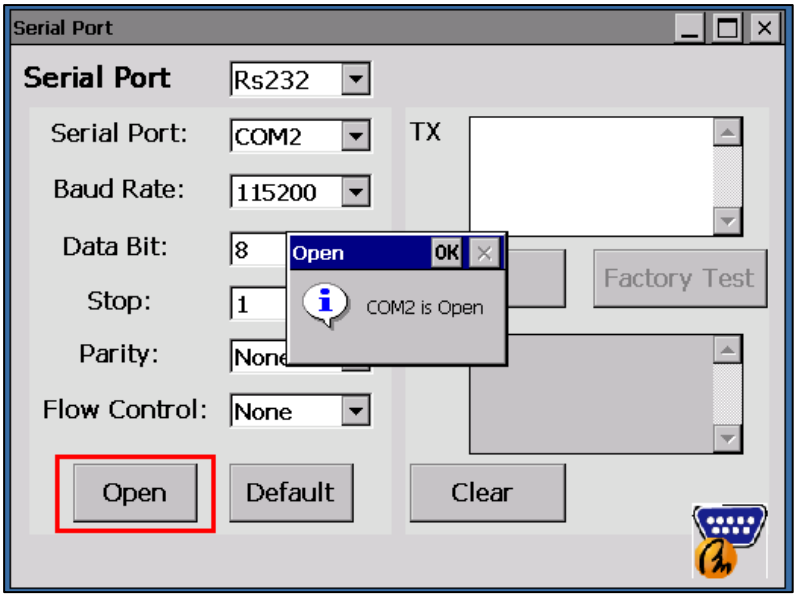
STEP 4. Click **SerialPort** icon



This is SerialPort-ap running On Windows CE desktop.



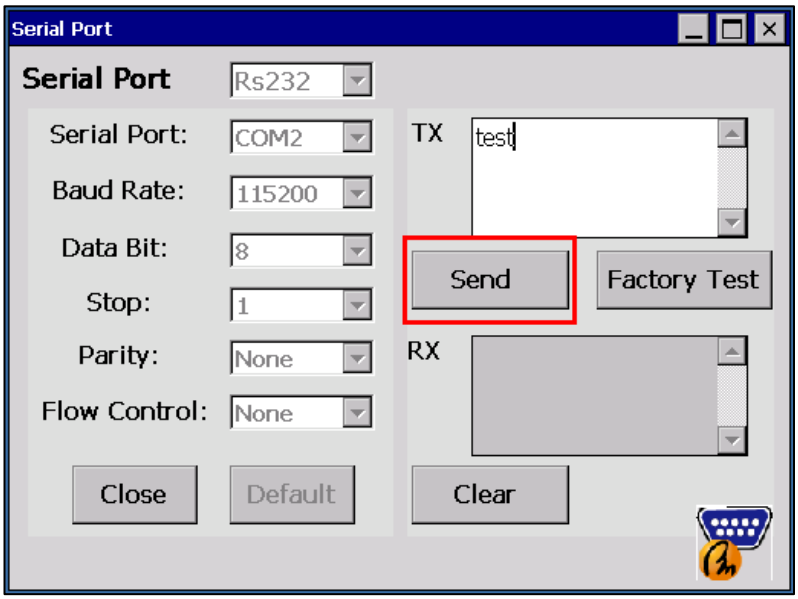
STEP 5. Please confirm parameters ("**Baud rate**", "**Data bit**", "**Flow Control**", "**Parity**", "**Stop**") setting as same as **HyperTerminal** and then click "**Open**" button to open serial port.



●If COM port opened success, it will display an **open message**.

●If not, it will display an **error message**.

STEP6. Typing some strings on the **TX block**, and click **Send** button for testing.

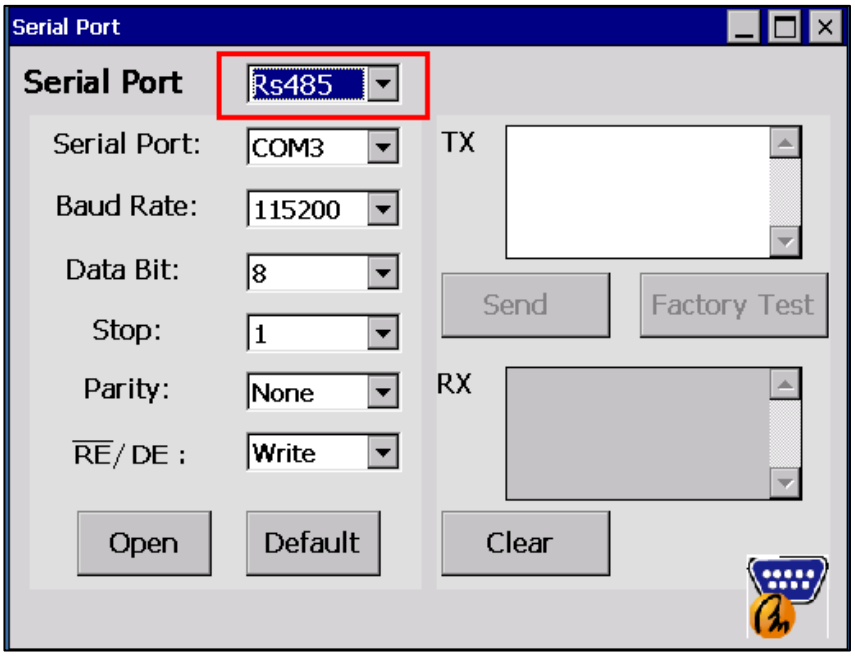


If everything goes fine, user will see the same strings on the **HyperTerminal.**

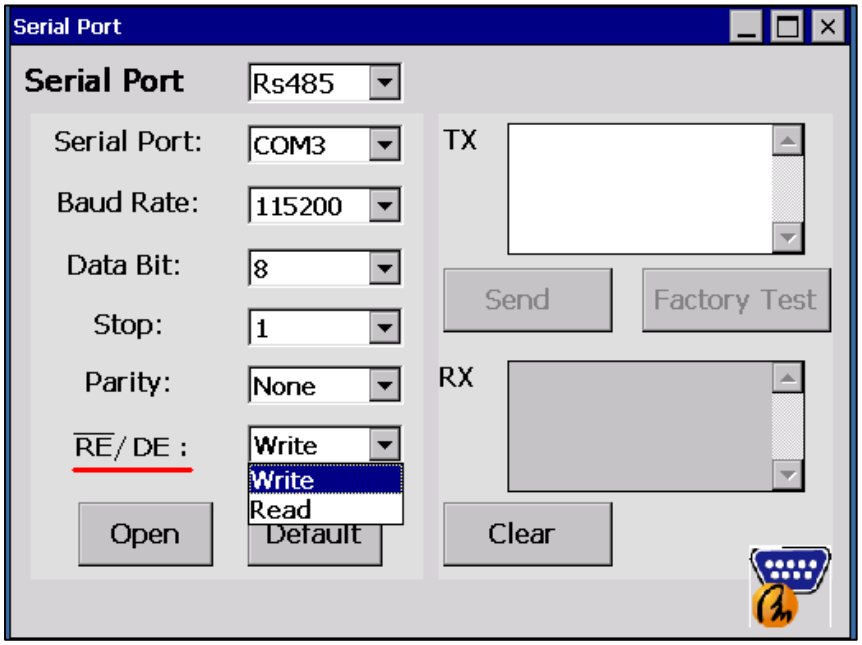
2-4.2> Serial port Rs485 test

The following steps show how to use Rs485 on Serialport-ap.

STEP 1. Select **Rs485** and confirm **parameters** setting as same as **HyperTerminal**



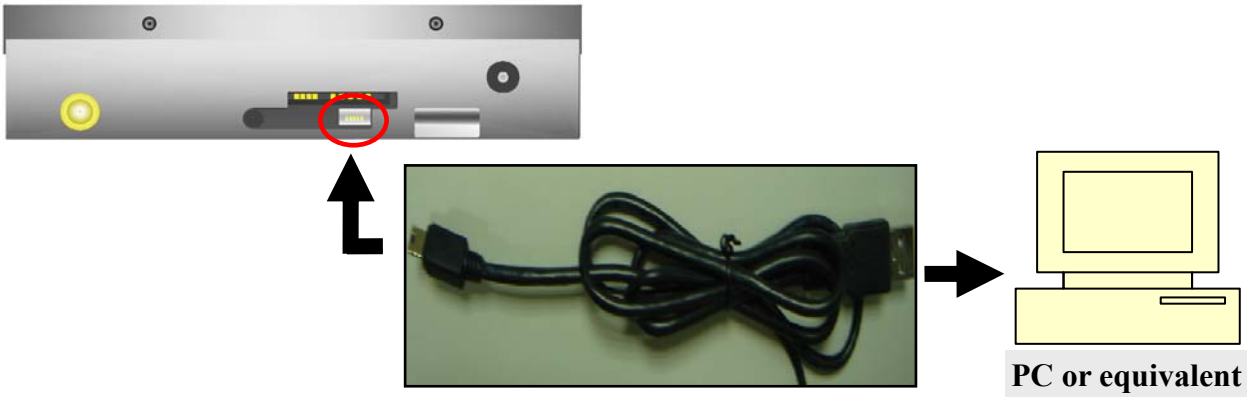
STEP 2. RE/DE Select and then click "**Open**" to open serial port.



●Rs485 can't write and read at same time.

2-5 Device USB installation

●JACK Mini-B On board: Support a dedicated Device port with USB2.0 compliant.
Provide for download user's AP from PC or equivalent.

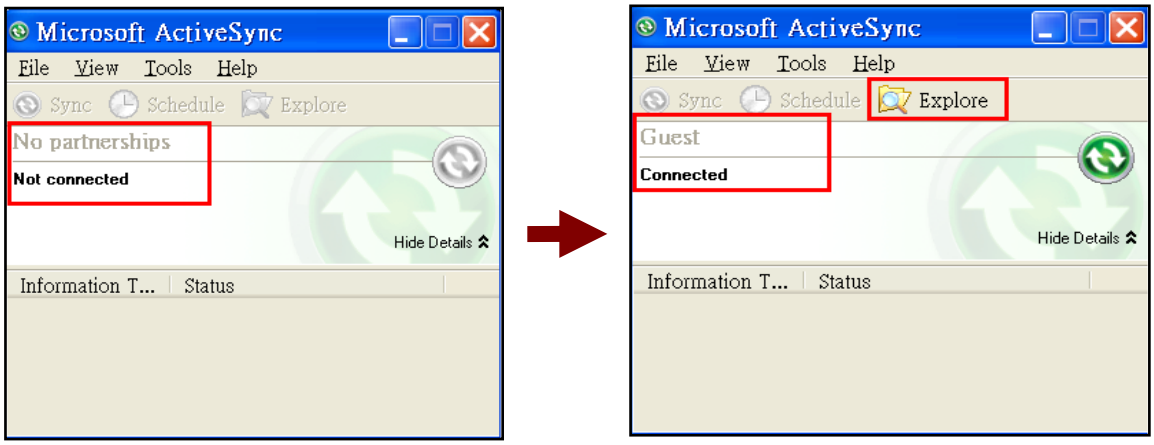


Cable: WYUSBMINI500900

2-5.1> USB connect test

It is necessary to download free software" Microsoft ActiveSync".
The download website is
"http://www.microsoft.com/downloads/en/details.aspx?FamilyID=9e641c34-6f7f-404d-a04b-dc09f8141141 "

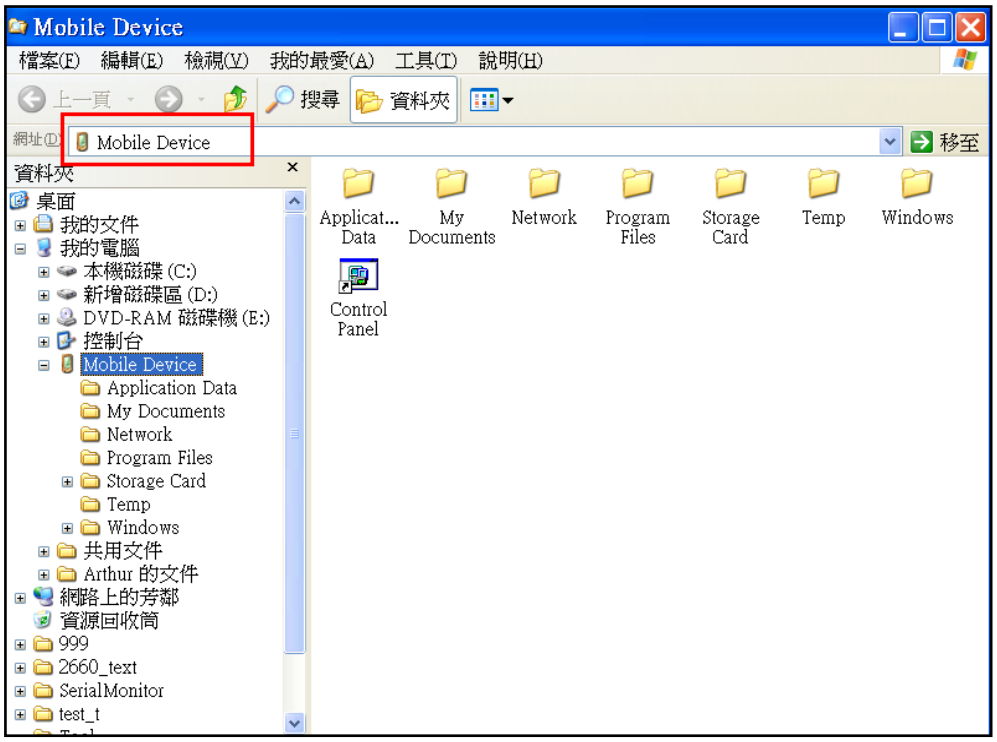
The following pictures are ActiveSync. After Install ActiveSync, when connect FES91W and computer with USB cable, the ActiveSync will display Connected and display green cycle.



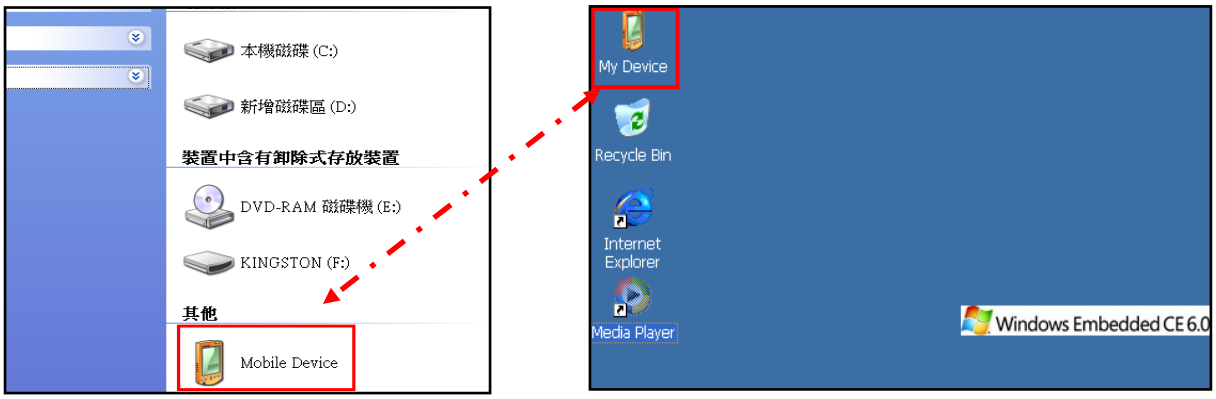
Not connect **Connect**

After executing “Explore” program. The PC will display a folder (Mobile Device), it has the same content as the folder (My Device) of FES91W.

User can transfer files between PC and FES91W over USB cable. When user transfer or delete data to folder (Mobile Device), FES91W folder (My Device) will synchronize.



●The folder (Mobile Device) is equivalent to the “My Device” folder of FES91W.

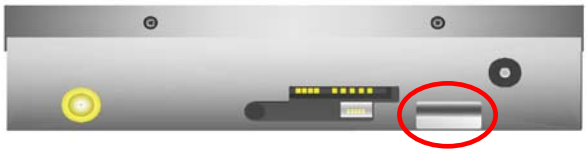


PC end

Windows CE6.0 end

2.6 Host USB installation

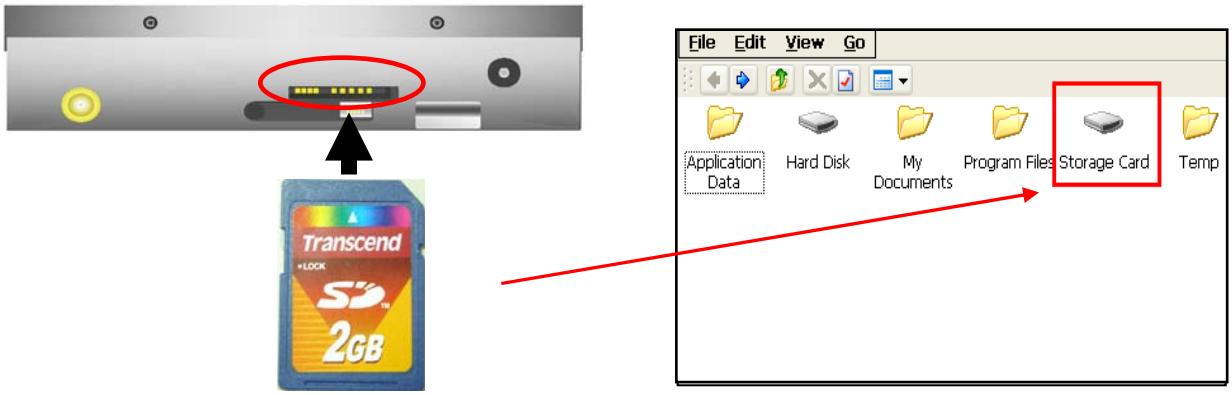
●JACK A-Type On board: Support a Host/Device port with USB2.0 compliant.
Provide for upload data stream from external USB device.



2.7 SD/MMC card installation

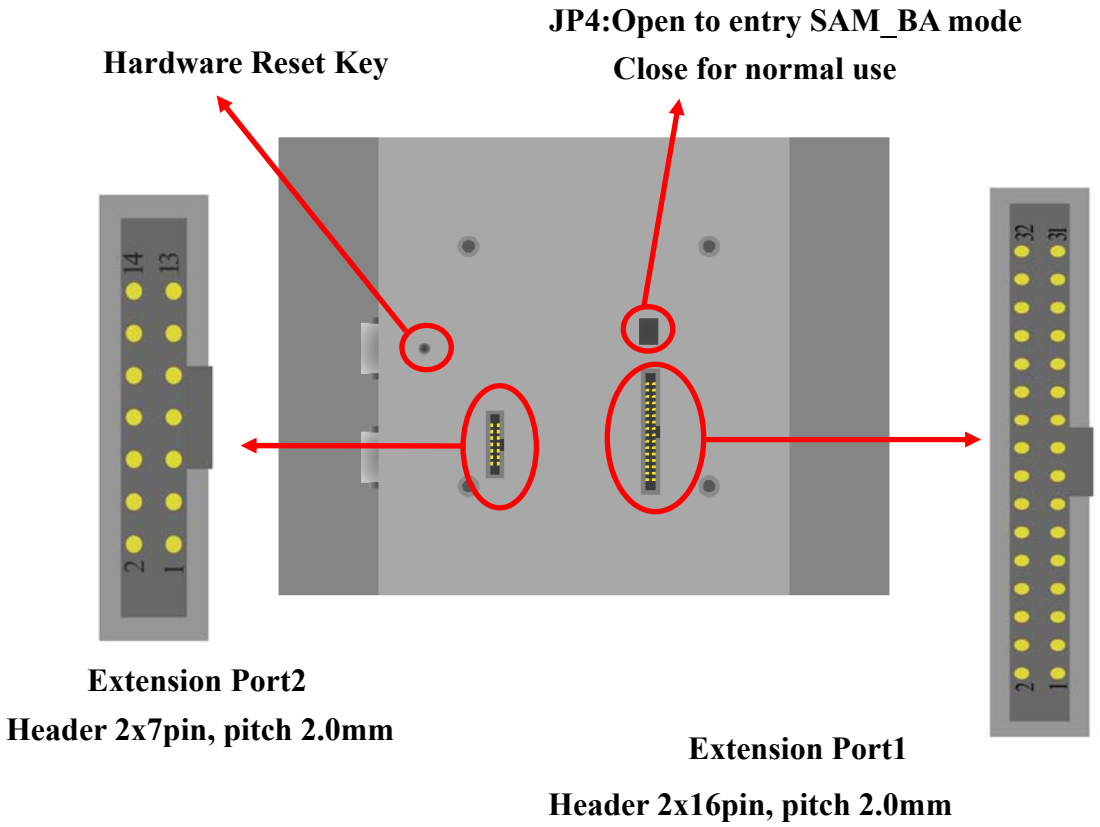
●Support 1/4 bit data transmission

●Supported card SD/MMC



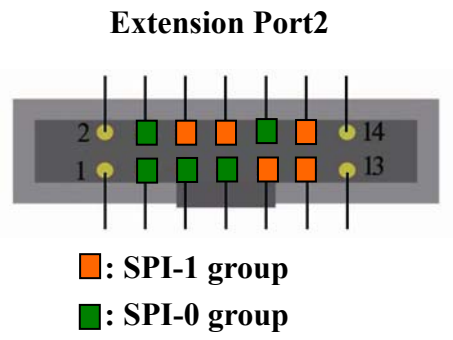
●When user insert SD Card, FES91W will display a “Storage Card” folder.

2.8 Extension Port installation



2.8-1> Extension Port2

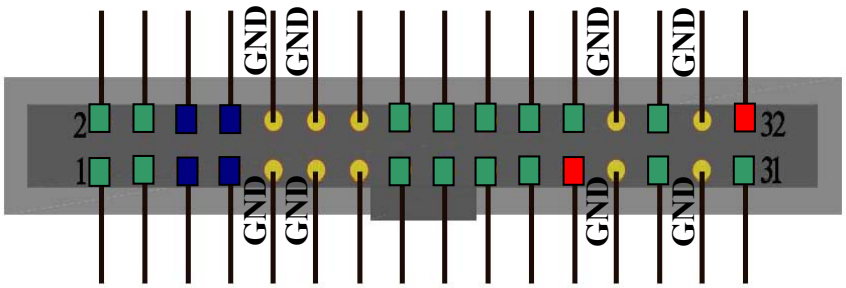
Extension Port 2 (Header 2x7_2.0mm_straight)			
Pin1	GND	Pin2	5V / 100mA out
Pin3	SPI0_MOSI	Pin4	SPI0_IRQ
Pin5	SPI0_MISO	Pin6	SPI1_MISO
Pin7	SPI0_NPCS0	Pin8	SPI1_MOSI
Pin9	SPI1_NPCS0	Pin10	SPI0_SPCK
Pin11	SPI1_IRQ	Pin12	SPI1_SPCK
Pin13	ComRX3	Pin14	ComTX3



- FES91W supports two sets of SPI, SPI 0 and SPI 1.
- Each SPI can supported Master or Slave mode.

***If CAN BUS installed, SPI_1 is responsible for CANBUS write /read. ***

2.8-2> Extension Port2



Extension Port1

Extension Port 1 (Header 2x16_2.0mm_straight)			
Pin1	GPIO14	Pin2	GPIO15
Pin3	GPIO12	Pin4	GPIO13
Pin5	AD7	Pin6	AD6
Pin7	AD4	Pin8	AD5
Pin9	GND	Pin10	GND
Pin11	GND	Pin12	GND
Pin13	3.3V/150mA out	Pin14	5V/150mA out
Pin15	GPIO11	Pin16	GPIO9
Pin17	GPIO6	Pin18	GPIO10
Pin19	GPIO7	Pin20	GPIO5
Pin21	GPIO8	Pin22	GPIO4
Pin23	PWM2	Pin24	GPIO3
Pin25	GND	Pin26	GND
Pin27	GPIO1	Pin28	GPIO2
Pin29	GND	Pin30	GND
Pin31	GPIO0	Pin32	PWM1

: GPIO Pins	Min	Typ	Max
Input Low-level	0.3V		0.8V
Input High-level	2V		3.6V
Output Low-level			0.4V
Output High-level	2.8V		3.3V

: A/D Pins	Min	Typ	Max
Input Voltage level			3.3V
Resolution	10 bit		
Throughput Rate	Max. 440KSPS		

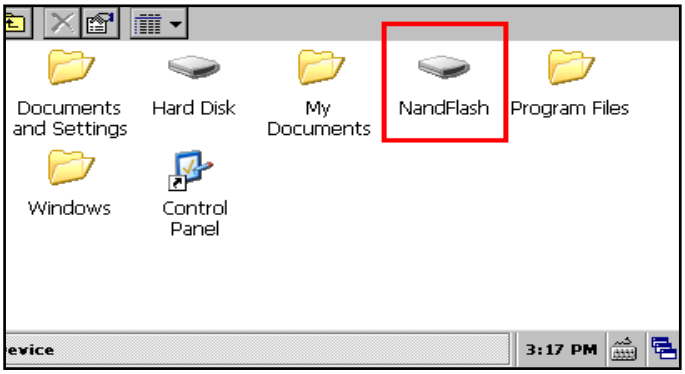
: PWM Pins	Min	Max
Output Low-level		0.4V
Output High-level	2.8V	3.3V
Frequency Range	15Hz	999KHz

2.8-3> GPIO test

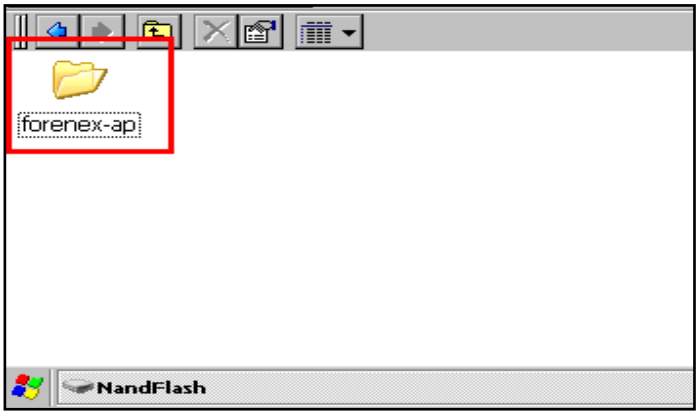
The following steps show how to use GPIO-ap on Windows CE.

STEP 1. On Windows CE desktop, click on "**My Device**".

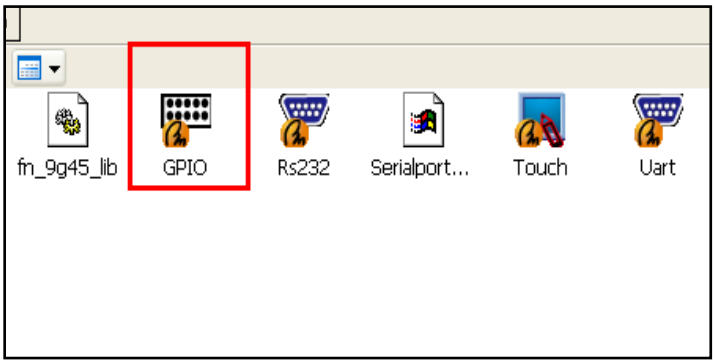
STEP 2. Select "**NandFlash**" folder.



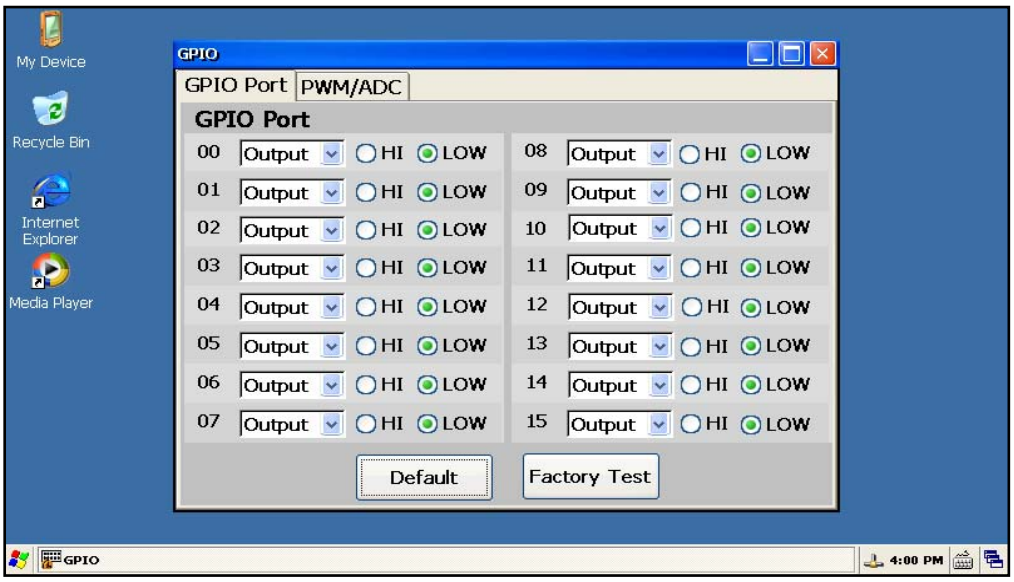
STEP 3. Select "**Forenex-ap**" folder.



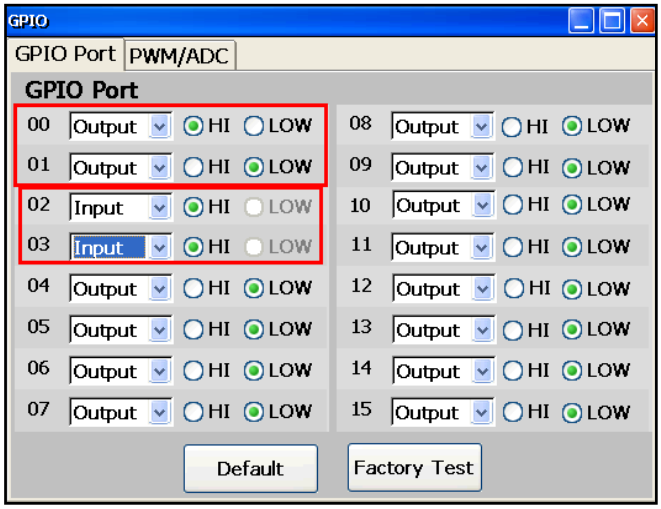
STEP 4. Click **GPIO** icon



This is GPIO-ap running On Windows CE desktop.

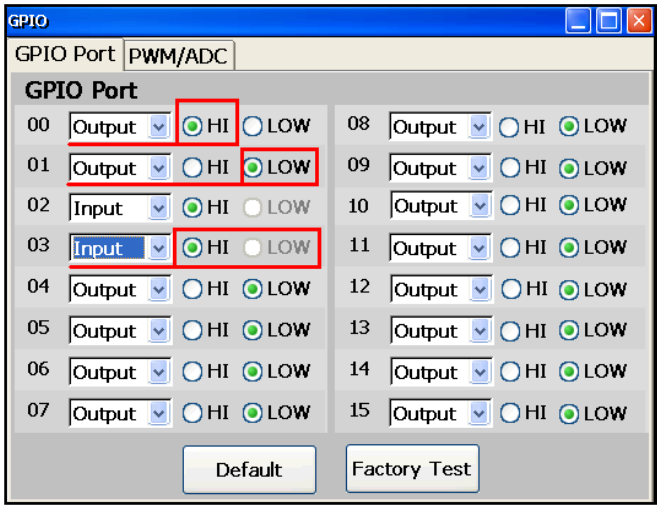


The GPIO pin can be set as **Output mode** or **Input mode.**



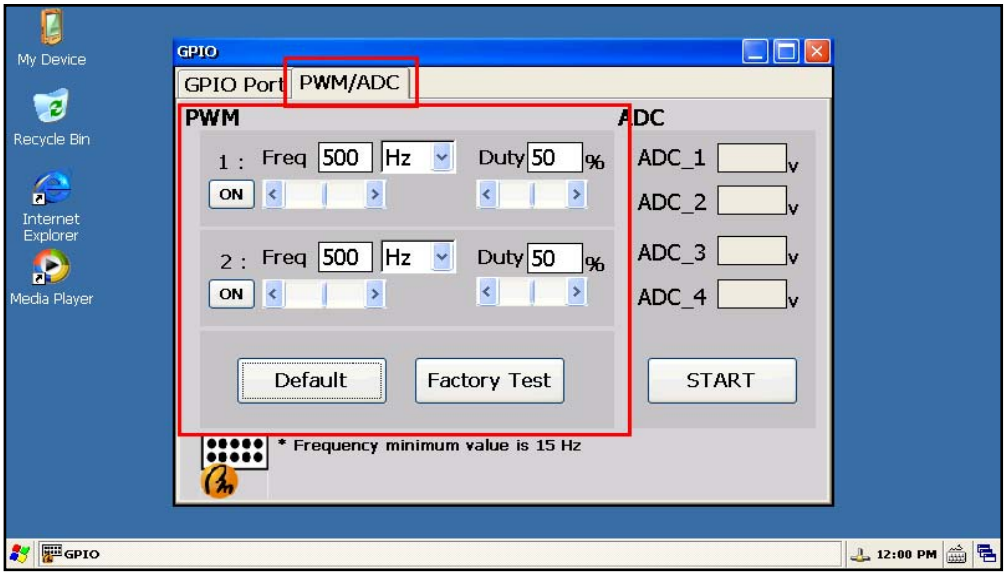
As **Output mode,** each voltage level of GPIO port can be set to **1** or **0** by selecting **"Hi"** or **"Low"**.

As **Input mode,** each voltage level of GPIO port can be read **"Hi=1"** or **"Low=0"** by system automatically.



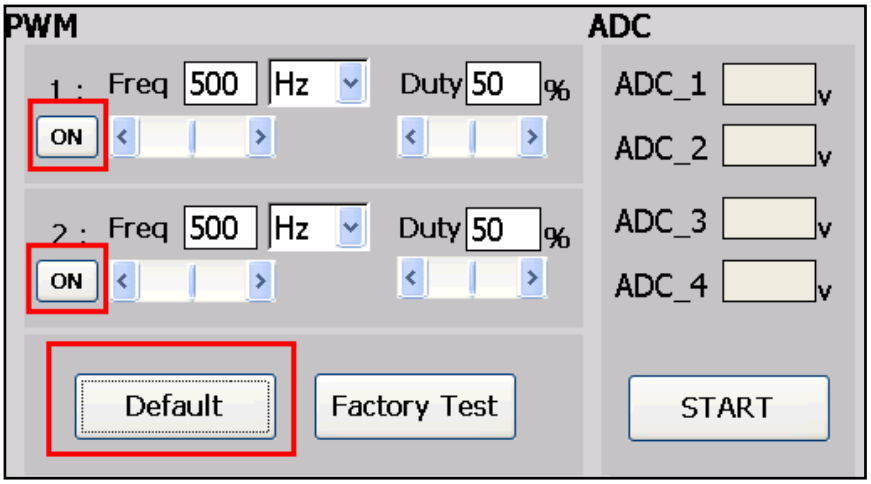
2.8-4> PWM test

STEP 1. Select GPIO-ap, and choose **PWM/ADC** page.

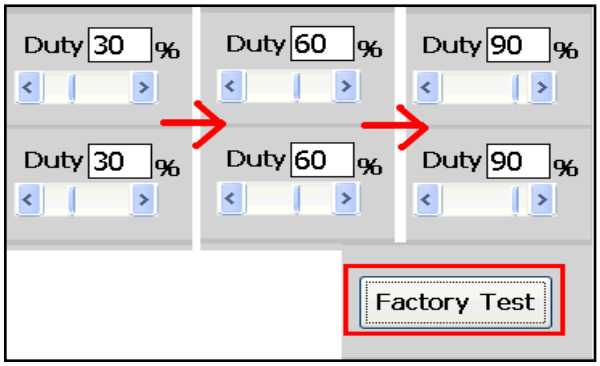


●Button **"ON/OFF"**; enable/disable PWM channel individually.

●Button **"Default"** recover the Frequency value back to 500 Hz and duty become 50.



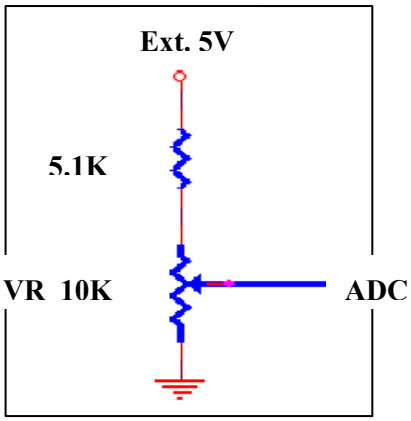
"Factory Test" is set Duty value change from 30 to 60 to 90.



2.8-5> ADC test

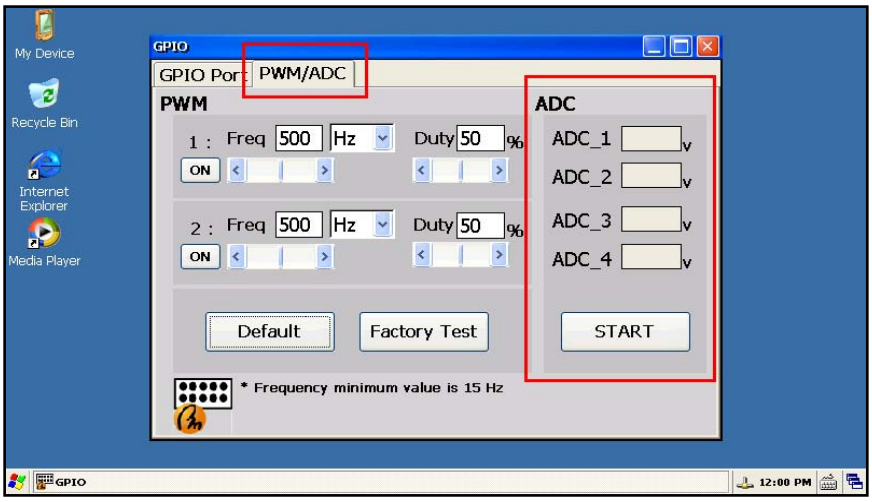
Below chart is an ADC test circuit for reference and use external power 5V.

●**Attention:** The external GND must be connected to GND of FES91W.

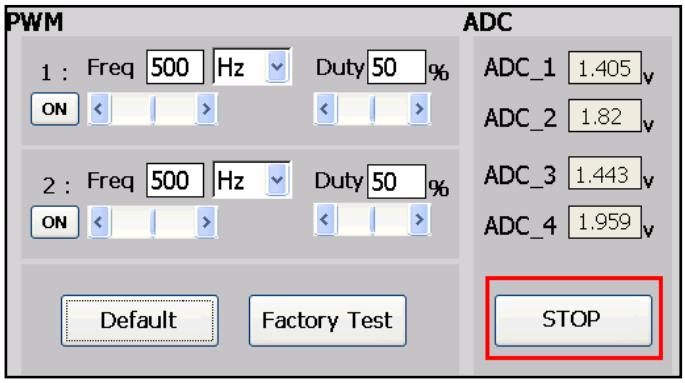


●**ADC voltage level have to be kept between 0~3.3V**

STEP 1. Select GPIO-ap, and then choose **PWM/ADC** page**,** ADC block is on the right side.

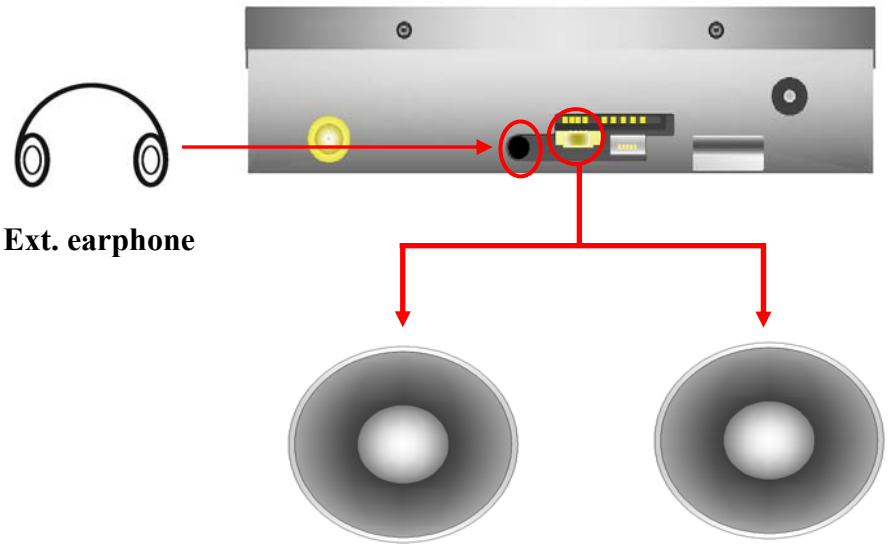


●Click **"START"** button, the value of each channel of ADC1 ~ 4 will be updated once in 100ms until click **"STOP"** button.



2.9 External stereo earphone installation

- ●Inside speaker would be muted while earphone has had plugged.
- ●AC97 Codec inside.



Ext. earphone

Ext. speaker with 8Ohm/1W

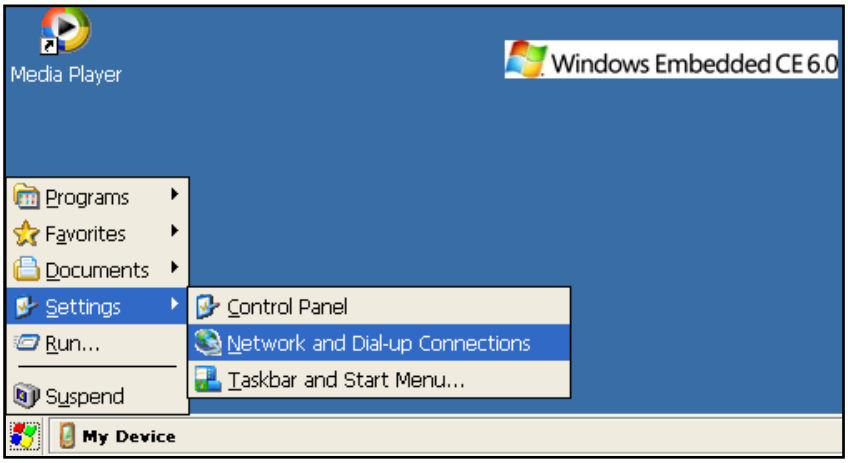
2.10 Internet installation

- **10/100 Mbps Ethernet controller full compliant with IEEE802.3u standards.**
- **Working mode RMII.**
- **RJ45 connector with LED indicator x 2 for active & link status.**

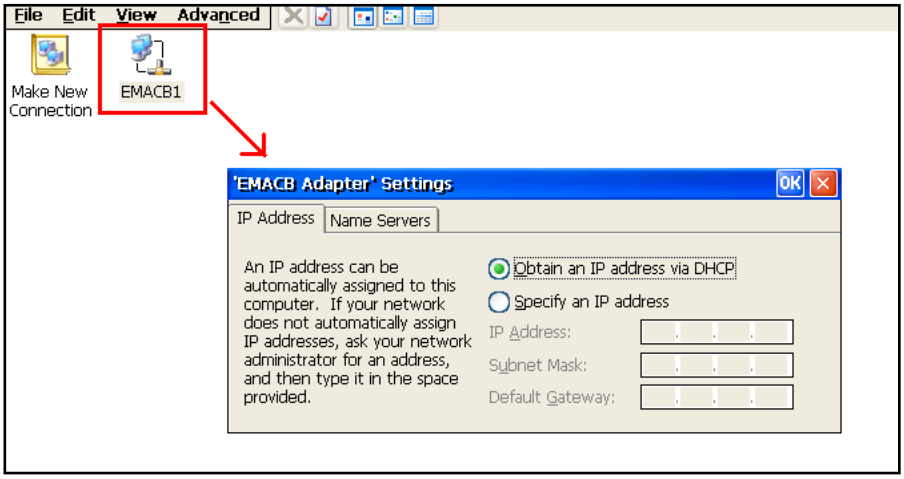
2-10.1> Internet setting

STEP1. Insert the internet line into the LAN port.

STEP2. Click Star-> Setting-> Network and Dail-up connections.



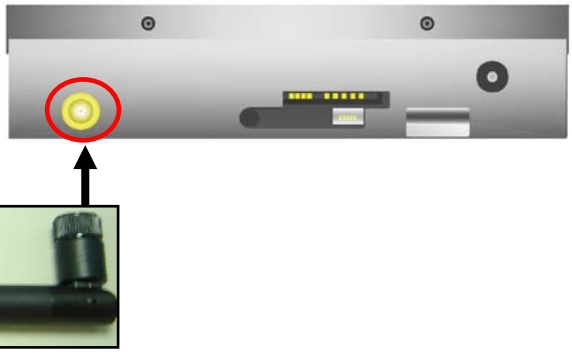
STEP3. Click "**EMACB1**" and Set specify an IP address or obtain an IP address via DHCP.



- **Check the network connection status is normal or not.**

2.12 Wi-Fi installation (with an external dongle)

- **Optional WLAN Module with SDIO Interface.**
- **Support WLAN module compliant with IEEE802.11b/g/n.**
- **Operating on 2.4GHz ISM radio band.**
- **WiFi compliant.**

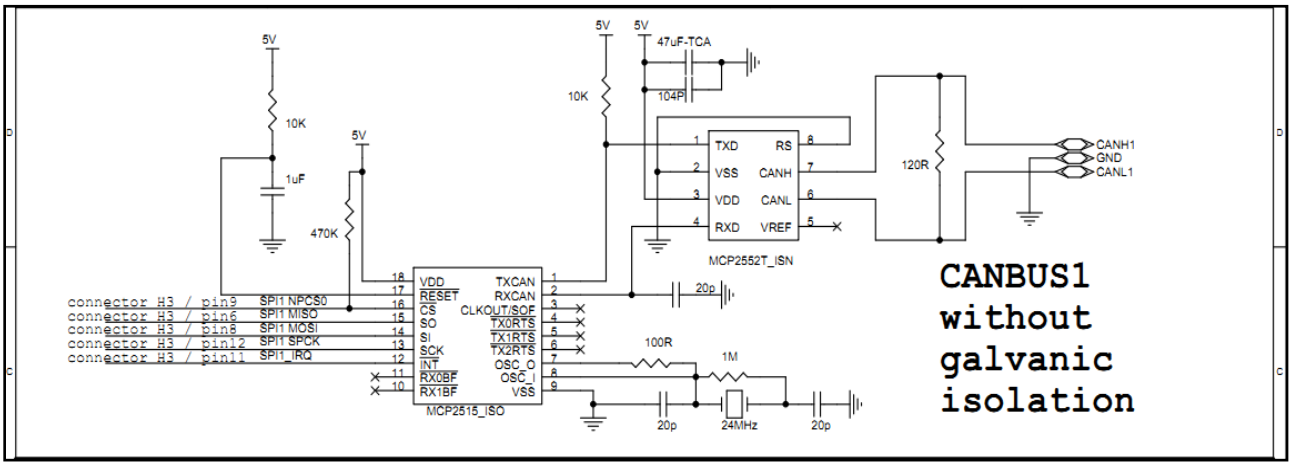


2.13 CAN BUS installation (with an external module)

The two CAN signals can **NOT be** connected directly to the CAN bus. User needs special interface logic to doing this. Take a look at the following to examples to get an idea how to connect the CAN bus.

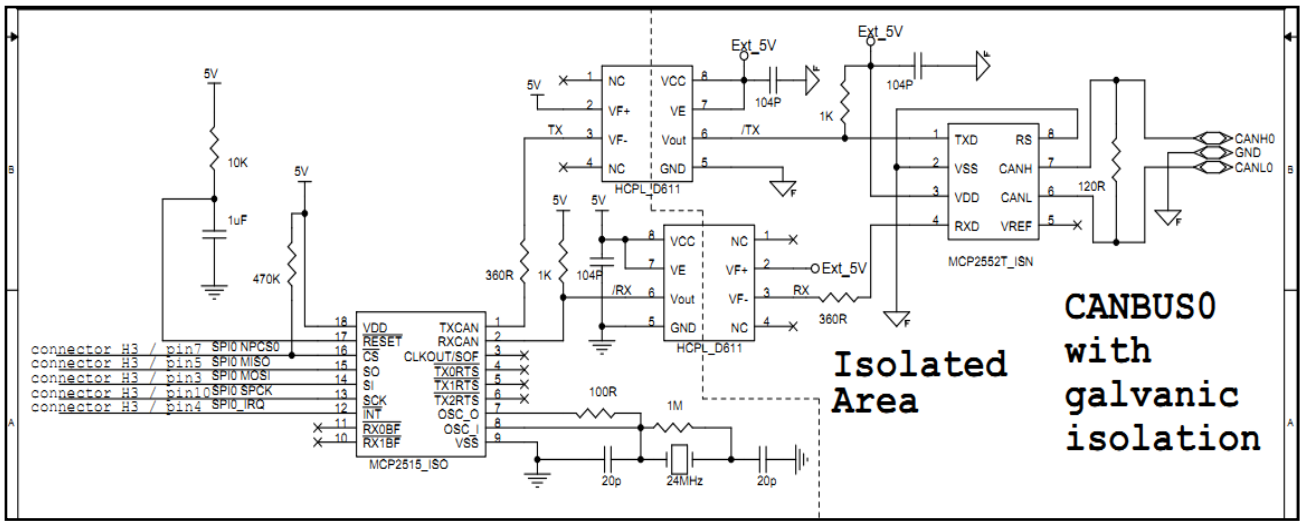
Example CAN-Interface type 1

The following schematic shows the needed interface between FES91X and CAN bus. It **has no** galvanic isolation.



Example CAN-Interface type 2

The following schematic shows the needed interface between FES91X and CAN bus. It **has** galvanic isolation.



3. FES91W series Programming Guide

FES91W provide DLL file (**fn_9g45_lib.dll**), it include some function call for control FES91W IO pins. User can refer below function table list to understand how to use these function.

3-1 Function table

Item	Function name
1	gpio_output
2	gpio_input
3	pwm_config
4	pwm_status
5	adc_config
6	touch
7	spi_config
8	spi_write
9	spi_read
10	rs485_status

- The function can call by "**fn_9g45_lib.dll**", fn_9g45_lib.dll located the folder of **"CD\FES91Wxx_xxx \FES91W-AP source code\"**

3-2 Function member description

Function	gpio_output
Prototype	**void gpio_output(int pin_num, int hi_low)**
Parameters	1. pin_num (integer) 2. hi_low (integer)
Return value	None
Description	**pin_num**(0 ~15) sets GPIO pin(0 ~15) as Output mode. **hi_low** sets GPIO pins voltage level: 0 means Low, 1 means Hight.
Example	**gpio_output(1, 1)** // Set GPIO pin 1 as output and voltage level is Hight

Function	gpio_input
Prototype	**int gpio_input(int pin_num)**
Parameters	1. pin_num (integer)
Return value	0 or 1
Description	**pin_num**(0 ~15) sets GPIO pin(0 ~15) as Input mode. **Return value** is 1 means the voltage level is Hight, 0 means Low. ●It only reads one time in this function.
Example	**input_status =gpio_input(0)** // Set GPIO pin 0 as input mode. **gpio_input( )** will read the voltage level from GPIO pin 0, and return the value to input_status.

<table>
<tr><th>Function</th><th>pwm_config</th></tr>
<tr><td>Prototype</td><td>void pwm_config(int pwm_num, int frequency, int duty_cycle)</td></tr>
<tr><td>Parameters</td><td>1. pwm_num (integer)
2. frequency (integer)
3. duty_cycle (integer)</td></tr>
<tr><td>Return value</td><td>None</td></tr>
<tr><td>Description</td><td>pwn_num(1 ~2) sets PWM (1 ~2).
frequency and duty_cycle are the parameters of PWM.
<table>
<tr><th>Parameters</th><th>frequency</th><th>duty_cycle</th></tr>
<tr><td>Minimum</td><td>15(Hz)</td><td>0(%)</td></tr>
<tr><td>Maximum</td><td>999000(Hz)</td><td>100(%)</td></tr>
</table>
●The minimum of frequency cannot smaller than 15Hz; otherwise the screen will be flashing
●When duty_cycle value is 0, PWM would turn Off.

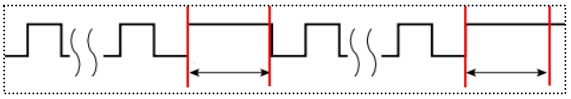
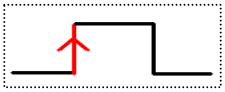
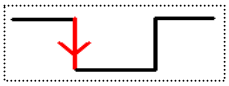
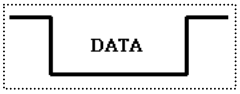
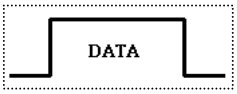
GPIOapp defines two different units of frequency: one is Hz, and another is KHz. KHz is value (1 ~999)*1000.</td></tr>
<tr><td>Example</td><td>pwm_config(1, 100, 50)
// Set parameter of pwm 1, frequency is 100 Hz and duty_cycle is 50.
pwm_config(2, 100000, 60)
// Set parameter of pwm 2, frequency is 100 KHz and duty_cycle is 60.</td></tr>
</table>

<table>
<tr><th>Function</th><th>pwm_status</th></tr>
<tr><td>Prototype</td><td>void pwm_status(int pwm_num, int status)</td></tr>
<tr><td>Parameters</td><td>1. pwm_num (integer)
2. state (integer)</td></tr>
<tr><td>Return value</td><td>None</td></tr>
<tr><td>Description</td><td>pwn_num(1 ~2) sets PWM(1 ~2).

●state : Start/Stop
1 : start,
0 : stop</td></tr>
<tr><td>Example</td><td>pwm_status(1, 0) // Set pwm 1 stop
pwm_status(2, 1) // Set pwm 2 start</td></tr>
</table>

Function	adc_config
Prototype	**int adc_config(int adc_num)**
Parameters	1. adc_num (integer)
Return value	Integer
Description	**channel**(1 ~4) sets ADC(1 ~4) and return ADC's value. ●Due to the Visual Basic cannot read a floating value (from DDL), therefore, once **"read_adc"** get a value from ADC, it will be multiplied by 10000 as an integer. ●To read the value of ADC channel by function call **"read_adc",** the result has to divide by 10000 to get back a real floating value. ●In GPIO-ap, the value, showed on screen, is divided by 10000.
Example	**adc_value = adc_config(1) / 1000** // read ADC(1) value and divide by 1000 to get real ADC(1) voltage level.

Function	touch
Prototype	**void touch()**
Parameters	None
Return value	None
Description	The function call is for calibration of touch panel on WinCE directly.
Example	**touch()** // The calibration of touch panel will starting at FES91W.

Function	spi_config
Prototype	**void spi_config(int spi_num, int baud, int bits, int delay, int mode, int phase, int polarity)**
Parameters	**1.** spi_num (integer) **2.** baud (integer) **3.** bits (integer) **4.** delay (integer) **5.** mode (integer) **6.** phase (integer) **7.** polarity (integer)
Return value	None
Description	**spi_num**(0 ~1) sets SPI(0 ~1). ●**baud :** Baud Rate of Serial Clock. Serial Clock Baud Rate = 133MHz / **baud**. **baud** value between 1 and 255. ●**bits :** Bits per transfer ; set 8 or 16. ●**delay :** Delay Between Consecutive Transfers. Delay Between Consecutive Transfers = 32 * **delay** / 133MHz. **delay** value between 0 and 255.  ●**mode :** Set 0 -> SPI is in Slave mode. Set 1 -> SPI is in Master mode. ●**phase :** Set 0 -> Data is changed on the leading edge of Serial Clock and captured on the following edge of Serial Clock.  Set 1 -> Data is captured on the leading edge of Serial Clock and changed on the following edge of Serial Clock.  ●**polarity :** Set 0 -> The inactive state value of Serial Clock is logic level zero.  Set 1 -> The inactive state value of Serial Clock is logic level one.  Initialize Parameter of SPI.
Example	**spi_config(0, 133, 8, 0, 1, 1, 0)** // Initialize parameter of SPI 0 : Serial Clock Baud Rate set 1MHz; bits set 8; No delay ; mode in master; phase set 1; polarity set 0

Function	spi_write
Prototype	**void spi_write(int spi_num, int length, int* data, int lastTransfer)**
Parameters	1. spi_num (integer) 2. length (integer) 3. data (integer) 4. lastTransfer(integer)
Return value	None
Description	**spi_num**(0 ~1) sets SPI(0 ~1). ●**length :** The amount of data to be transmitted. ●*** data :** The data to be transmitted through the SPI. (The data need in array, because data would transmit from address to address by SPI). ●**lastTransfer :** Set CS in active mode. 0 : CS keep active after the last data has been wrote-out. 1 : Disable CS after the last data has been wrote-out.

Example	**spi_write(0, 3, &Spi_Wri_Data[ ], 0)**
	// Datas in the array[0] ~ array[2] would be transmitted by SPI 0.
	// CS keep active after the last data has been wrote-out

Function	spi_read
Prototype	**void spi_read(int spi_num, int length, int* data, int lastTransfer)**
Parameters	1. spi_num (integer) 2. length (integer) 3. data (integer) 4. lastTransfer(integer)
Return value	None
Description	**spi_num**(0 ~1) sets SPI(0 ~1). ●**length :** The amount of data to be receives. ●*** data :** The data to be received through the SPI. (Need to prepare an array to receive data, because data would transmit to address by SPI). ●**lastTransfer :** Set CS in active mode. 0 : CS keep active after the last data has been read-in. 1 : Disable CS after the last data has been read-in.
Example	**spi_read(int spi_num, int length, int* data, int lastTransfer)** // Array[0] ~ array[1] will receive data from SPI 1.

	// CS will be disable after the last data has been read-in

Function	rs485_status
Prototype	**void rs485_status(int com_num, int hi_low)**
Parameters	1. com_num (integer) 2. hi_low (integer)
Return value	None
Description	This function will set Com port to be Rs485 in write or read mode . ●**com_num :** Set Com2 or Com3 to be Rs485. 1 : COM1 dedicated for Debug, not available for general use. 2 : COM2 3 : COM3 4 : COM4 ●**hi_low :** Set Com port in read or write mode. 1 : Write mode 0 : Read mode Rs485 can not be read or write at same time.
Example	**rs485_status(2,1)** // Set Com Port 2 to be Rs485 and as Write mode.

4. Programming for FES91W

4-1 Install Visual Studio 2005

In this section, we will describe how to Set up Development environment and create a new project in Visual Studio 2005.

4-1.1> Install Software Development environment

SDK for WinCE 6.0 is using **Visual Studio 2005**.

●User can go to below website to download **Visual Studio 2005**:

"**http://www.microsoft.com/windowsembedded/en-us/downloads/download-windows-embedded-ce6.aspx**"

Follow below steps to download complete Visual Studio 2005 on Windows XP.

1> Visual Studio 2005 Trial

2> Microsoft Visual Studio 2005 Team Suite Service Pack 1

3> Windows Embedded CE 6.0 Evaluation Edition

4> Windows Embedded CE 6.0 Platform Builder Service Pack 1

5> Windows Embedded CE 6.0 R2

6> Windows Embedded CE 6.0 R3

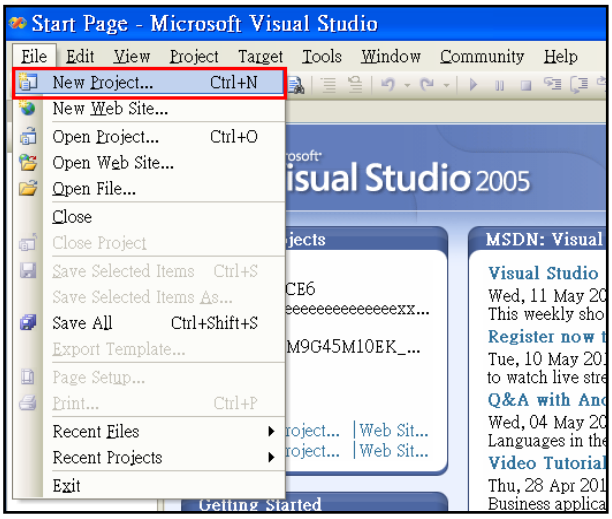
●User can get more development information for VS2005 from MSDN website.

http://msdn.microsoft.com/en-us/library/ms950416.aspx

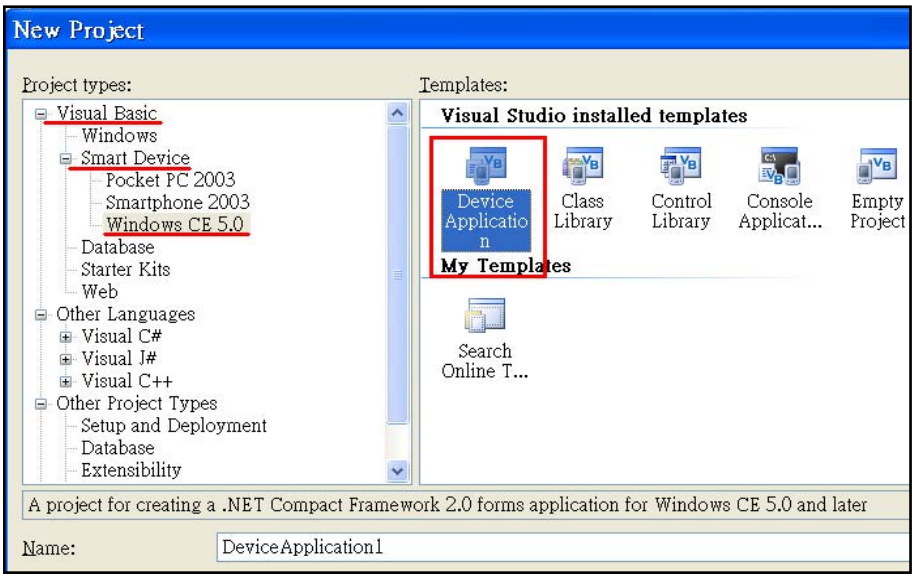
4-1.2> Create New Project

User could create a new project for your application by following steps:

STEP 1. Select File -> New Project

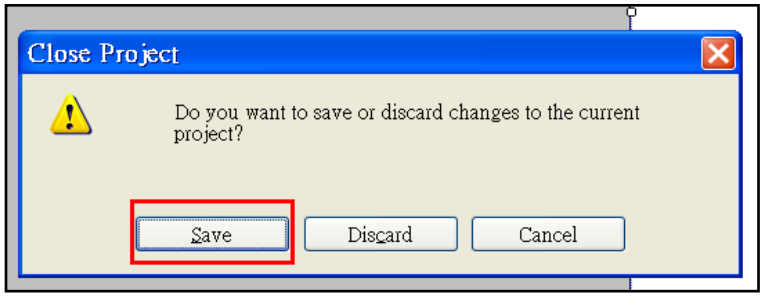


STEP 2. Select project types -> Visual Basic -> Smart Device -> Windows CE 5.0
And Templates select “Device Application”, and then OK

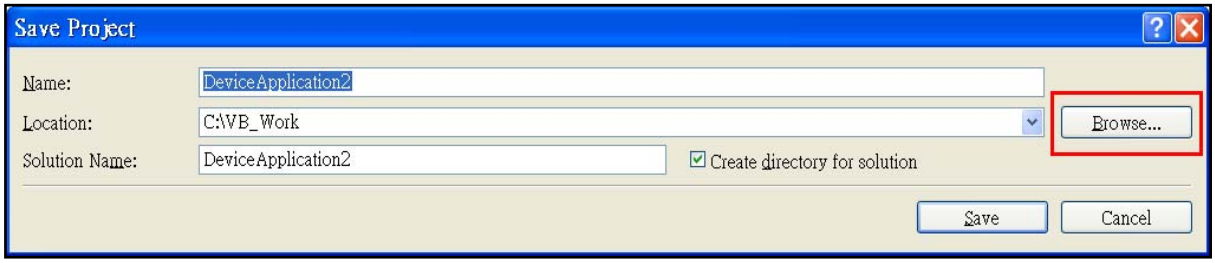


●Using Visual Basic development tool to create program which can running in Windows CE

STEP 3. The first time user want to close project, Visual Stdio 2005 will display a message to ask user want to save project or not, click “**Save**”



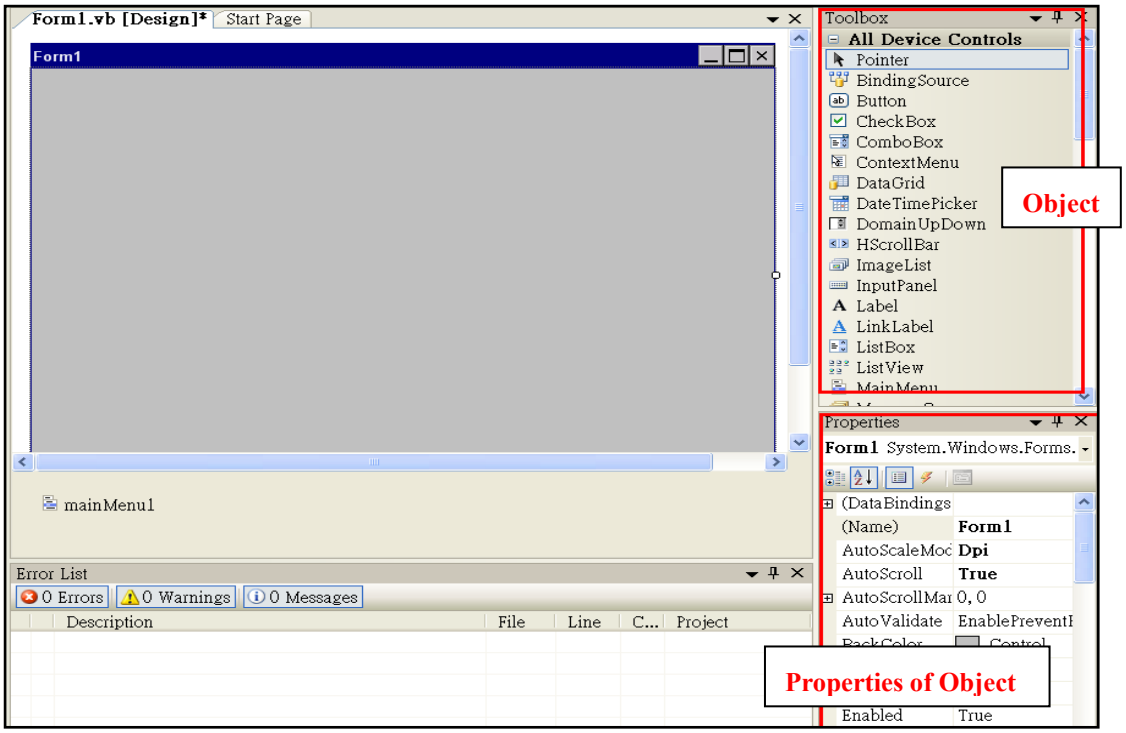
Select the location where is user want to save to.



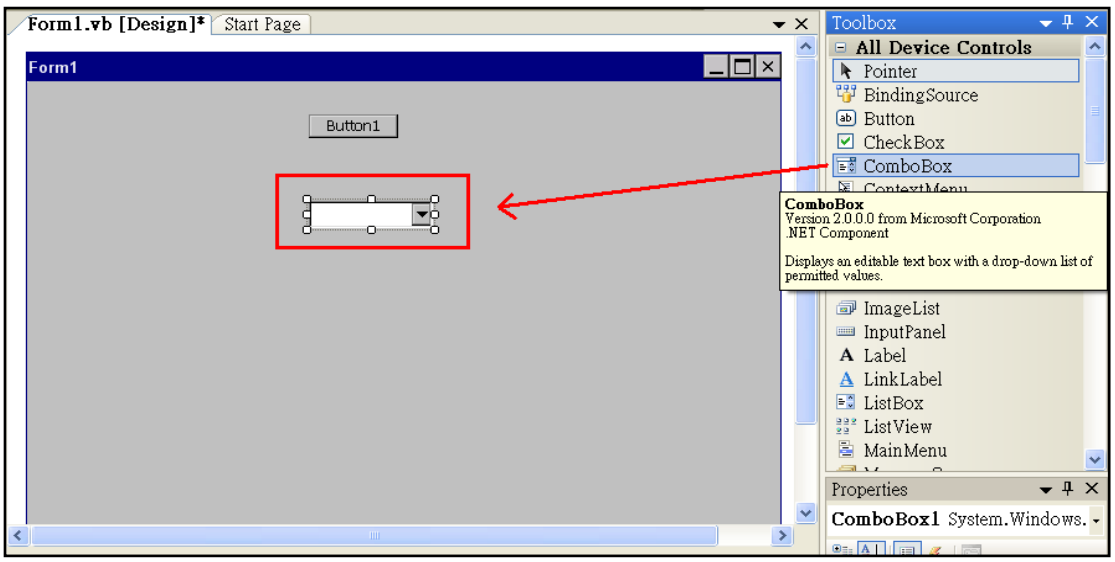
●User can get more development information from MSDN website.
http://msdn.microsoft.com/en-us/library/ms950416.aspx

4-1.3> Visual Basic simple Introduction.

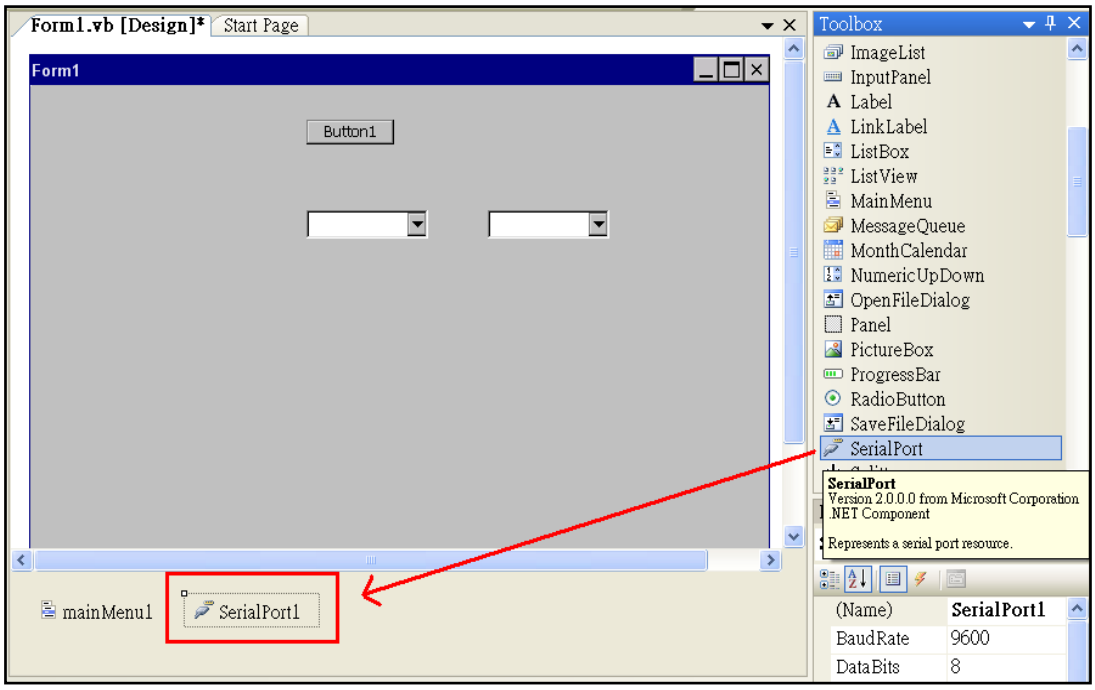
Introduction Interface



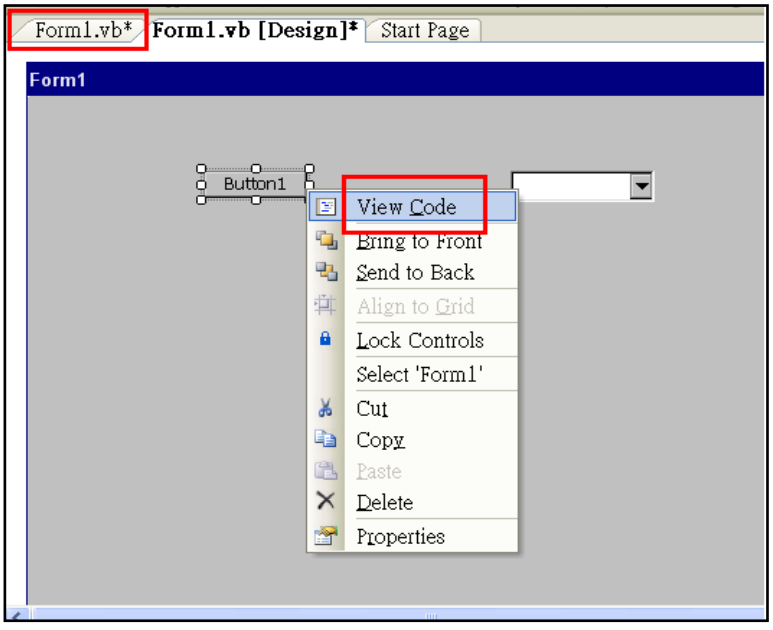
User can double click object or drag object from toolbox to Form, and From will display the choice object.



Some object wont display on the Form, it will display on below block, for example: Serial port.



Select object and click the mouse right key; it can select view code, and will display program code. It also can double click object, program code will display too.



●Program code is in “Form1.vb” page.

4-2 GPIO Control

4-2.1> How to control GPIO for FES91W

In FES91W, there are 16 definable GPIO. Depends on setting of each GPIO, user could read or write the level status of each GPIO of FES91W.

FES91W provide **control functions for GPIO**; user can use it to initial the General Purpose GPIO as input or output. (Refer to section 3-2 to know how)

Below procedures are to merge **GPIO control functions** into user's program:

STEP1. Add "**fn_9g45_lib.dll**" into project folder.

STEP2. Load "**fn_9g45_lib.dll**" in public declare area of user's program.

STEP3. Use the function "**gpio_output**" & "**gpio_input**" refer to below 4-2.2.

●The source code located the folder of

"CD\FES91Wxx_xxx \FES91W-AP source code\GPIO"

4-2.2> Example code

Below is a simple example code to use the GPIO controlling functions in Visual Studio 2005.

'Declaration loaded GPIO control function from DLL file

```
<DllImport("fn_9g45_lib.dll")> _
 Public Shared Function gpio_output(ByVal pin_num As Integer, ByVal hi_low As Integer) As
 Integer
 End Function
<DllImport("fn_9g45_lib.dll")> _
 Public Shared Function gpio_input(ByVal pin_num As Integer) As Integer
 End Function
```

'The following program is how to execute GPIO function on RadioButton

```
 Private Sub RadioButton1_CheckedChanged(ByVal sender As System.Object, ByVal e As
 System.EventArgs) Handles RadioButton1.CheckedChanged
     If ComboBox1.SelectedIndex = 0 Then    ' Select GPIO become Output mode
         If RadioButton1.Checked = True Then    ' Hight
             gpio_output(0, 1) 'Set GPIO(0) to be output mode and set voltage is HI
         End If
     End If
     If ComboBox1.SelectedIndex = 1 Then    ' Select GPIO become Input mode
         If RadioButton1.Checked = True Then    'LOW
         End If
     End If
 End Sub
```

'The following program is how to execute GPIO function on RadioButton

```
Private Sub RadioButton2_CheckedChanged(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles RadioButton17.CheckedChanged
    If ComboBox1.SelectedIndex = 0 Then    ' Select GPIO become Output mode
        If RadioButton2.Checked = True Then     'LOW
            gpio_output(0, 0) 'Set GPIO(0) to be output mode and set voltage is LOW
        End If
    End If
    If ComboBox1.SelectedIndex = 1 Then    ' Select GPIO become Input mode
        If RadioButton17.Checked = True Then    'LOW
        End If
    End If
End Sub
```

'The following program is how to execute GPIO function on ComboBox

```
Private Sub ComboBox1_SelectedIndexChanged(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles ComboBox1.SelectedIndexChanged
    Dim input_status As Integer
    If ComboBox1.SelectedIndex = 0 Then    ' Select GPIO become Output mode
        RadioButton1.Enabled = True         'Hight
        RadioButton17.Enabled = True        'Low
    Else     ' Select GPIO become Input mode
        input_status = gpio_input(0)    'Set GPIO(0) as input mode and read intput voltage
        If input_status = 1 Then  'If read voltage is HI
            RadioButton1.Checked = True
            RadioButton1_CheckedChanged(sender, e)
        ElseIf input_status = 0 Then    ' If read voltage is Low
            RadioButton2.Checked = True
            RadioButton2_CheckedChanged(sender, e)
        End If
     End If
End Sub
```

●When GPIO as Output mode, user can select voltage level Hi or Low by RadioButton.

4-3 PWM/Backlight Control

4-3.1> How to control PWM/Backlight for FES91W

In FES91W, there are 3 definable Pulse Width Modulation. User could get current brightness value or change the brightness of backlight by control functions.

FES91W provide **control functions for PWM**; user can use it to initial the General Purpose PWM/Backlight signal. (Refer to section 3-2 to know how)

Below procedures are to merge **PWM/Backlight control functions** into user's program:

STEP1. Add "**fn_9g45_lib.dll**" into project folder.

STEP2. Include "**fn_9g45_lib.dll**" in public declare area of user's program.

STEP3. Use the function "**pwm_config**" & "**pwm_status**" refer to below 4-3.2.

●The source code located the folder of

"CD\FES91Wxx_xxx \FES91W-AP source code\GPIO"

4-3.2> Example code

Below is a simple example code to use the PWM controlling functions in Visual Studio 2005.

```
' Declaration loaded PWM control function from DLL file
    <DllImport("fn_9g45_lib.dll")> _
    Public Shared Function pwm_config(ByVal pwm_num As Integer, ByVal frequency As Integer,
    ByVal duty_cycle As Integer) As Integer
    End Function

    <DllImport("fn_9g45_lib.dll")> _
    Public Shared Function pwm_status(ByVal pwm_num As Integer, ByVal status As Integer) As
    Integer
    End Function
```

```
'The following program is how to execute PWM function
    pwm_status(1, 1) 'Set PWM(1) Star
    frequency = HScrollBar2.Value
    duty_cycle = HScrollBar1.Value
    'Set frequency and duty_cycle of PWM1
    pwm_config(1, frequency, duty_cycle)
```

●**frequency and duty_cycle value will follow the HScrollBar to change.**

4-4 ADC Control

4-4.1> How to control ADC for FES91W

In FES91W, there are 4 definable A/D converters. FES91W provide **control functions for ADC**; user can use it to read the level status from ADC channel. (Refer to section 3-2 to know how)

Below procedures are to merge **ADC control functions** into user's program:

STEP1. Add "**fn_9g45_lib.dll**" into project folder.

STEP2. Include "**fn_9g45_lib.dll**" in public declare area of user's program.

STEP3. Use the function "**adc_config**" refer to below 4-4.2.

●The source code located the folder of

"CD\FES91Wxx_xxx \FES91W-AP source code\GPIO"

4-4.2> Example code

Below is a simple example code to use the ADC controlling functions in Visual Studio 2005.

```
' Declaration loaded ADC control function from DLL file
    <DllImport("fn_9g45_lib.dll")> _
     Public Shared Function adc_config(ByVal adc_num As Integer) As Integer
     End Function

'The following program is how to execute ADC function
     Dim adc1_val As Single
     'Read ADC1 value, and divide by 1000
     adc1_val = adc_config(1) / 1000
```

4-5 Serial port Control

4-5.1> Overview

In FES91W, there are 4 definable serial ports. Below table lists the function of each serial port.

Name	Function	Comment
Debug port	Internal used	Can't open by application program.
COM2	RS232 or RS485	Option (4 wire)
COM3	RS232 or RS485	Option (2 wire)
COM4	RS232	2 wire RS232

4-5.2> Member function of class SerialPort

Microsoft Visual studio has provide Class of Serial port in website MSDN. (Serialport Class), user can use these class in Visual studio 2005 to control Serial port.

Serialport Property

Name	Description
PortName	Gets or sets the port for communications, including but not limited to all available COM ports.
BaudRate	Gets or sets the serial baud rate.
DataBits	Gets or sets the sFtandard length of data bits per byte.
Parity	Gets or sets the parity-checking protocol.
StopBits	Gets or sets the standard number of stop bits per byte.

●**More Properties details in website MSDN.**

http://msdn.microsoft.com/en-us/library/system.io.ports.serialport_properties%28v=VS.85%29.aspx

●The source code located the folder of

"CD\FES91Wxx_xxx \FES91W-AP source code\Serialport"

4-5.3> Example code

Below is a simple example code to use serial port (**Rs232**) functions in Visual studio 2005.

```
'The following program is simple code for setting Serial port
    Private Sub InitSerialPort()
        SerialPort1.PortName = "COM2"          'Set the port for communications
        SerialPort1.BaudRate = 115200          'Baudrate
        SerialPort1.DataBits = 8               'Data Bits
        SerialPort1.Parity = 0                 'Parity
        SerialPort1.StopBits = 1               'Stop bit
        SerialPort1.Open()                     'Set Serialport open
    End Sub
```

```
'The following program is how to execute serial port send data
        SerialPort1.Write(TextBox1.Text)    'Send textbox.text by serialport
        Timer1.enable = true                'Setup timer to receive data
```

```
'The following program is how to execute serial port receive data
    Private Sub Timer1_Tick(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Timer1.Tick
        Dim TmpStr As String = ""
        Dim Bytearray() As Byte     ' Declare a Byte array to receive data
        Dim rcvBytes As Integer     ' Record the number of bytes received
        Do
            'Get the number of Byte from serial port buffer
            rcvBytes = SerialPort1.BytesToRead
            'Set size of Byte array
            ReDim Bytearray (rcvBytes - 1)
            'Write Byte to Bytearray() array
            Me.SerialPort1.Read(Bytearray, 0, rcvBytes)
            'Convert Byte array to a string
            TmpStr = Encoding.Default.GetString(Bytearray, 0, rcvBytes)
            'Display on textbox
            TxtTerminal.Text &= TmpStr
        Loop While rcvBytes <> 0
    End Sub
```

- **More serial port information in website MSDN.**

http://msdn.microsoft.com/en-us/library/system.io.ports.serialport.portname%28v=VS.80%29.aspx

Below is a simple example code to use serial port (**Rs485**) functions in Visual studio 2005

Need to add this control function to control Rs485.

```
' Declaration loaded Rs485 control function from DLL file
    <DllImport("fn_9g45_lib.dll")> _
    Public Shared Function rs485_status(ByVal com_num As Integer, ByVal hi_low As Integer) As
    Integer
    End Function
```

```
'The following program is how to control Com2 in Rs485 to write or read
    Private Sub ComboBox8_SelectedIndexChanged(ByVal sender As System.Object, ByVal e As
    System.EventArgs) Handles ComboBox8.SelectedIndexChanged
        'com2 control
        Select Case ComboBox8.Text
            Case "Write"
                rs485_status(2, 1)
            Case "Read"
                rs485_status(2, 0)
        End Select
         End Sub
```

4-6 SPI Control

4-6.1> How to control SPI for FES91W

In FES91W, there are 2 definable SPI. . FES91W provide **control functions for SPI**; user can use it to read the level status from ADC channel. (Refer to section 3-2 to know how)

Below procedures are to merge **ADC control functions** into user's program:

STEP1. Add "**fn_9g45_lib.dll**" into project folder.

STEP2. Include "**fn_9g45_lib.dll**" in public declare area of user's program.

STEP3. Call "**spi_config**" Function to initialize SPI's parameter.

STEP3. Use the function "**spi_write**" or **"spi_read"** refer to below 4-6.2.

4-6.2> Example code

Below is a simple example code to use the SPI controlling functions in Visual Studio 2005.

' Declaration loaded SPI control function from DLL file.

```
<DllImport("fn_9g45_lib.dll")> _
  Public Shared Function spi_config(ByVal spi_num As Integer, ByVal baud As Integer, ByVal
  bits As Integer, ByVal delay As Integer, ByVal mode As Integer, ByVal phase As Integer,
  ByVal polarity As Integer) As Integer
  End Function

 <DllImport("fn_9g45_lib.dll")> _
  Public Shared Function spi_read(ByVal spi_num As Integer, ByVal length As Integer, ByRef
  read_data As Integer) As Integer
  End Function

<DllImport("fn_9g45_lib.dll")> _
  Public Shared Function spi_write(ByVal spi_num As Integer, ByVal length As Integer, ByRef
  write_data As Integer) As Integer
  End Function
```

'The following program is initialize SPI's parameter.

```
spi_config(0, 133, 8, 0, 1, 1, 0)
```

●Initialize SPI 0

'The following program is use SPI function.

```
Dim SPI_NUM As Integer
Dim Spi_Wri_Data(3) As Integer
Dim Spi_Wri_Len As Integer = 3

Dim Spi_Rea_Data(3) As Integer
Dim Spi_Rea_Len As Integer = 3

Spi_Wri_Data(0) =1
Spi_Wri_Data(1) =2
Spi_Wri_Data(2) =3
'spi_write(SPI_NUM , Length, Spi_Wri_Data)
spi_write(0, Spi_Wri_Len, Spi_Wri_Data(0))

'spi_read(SPI_NUM , Length, Spi_Rea_Data)
spi_read(0, Spi_Rea_Len, Spi_Rea_Data(0))
```

●SPI 0 was initialized, so 3 data(1, 2, 3) can be transmitted by SPI 0, and receive data from SPI 0.