

# Megawin

# USB EasyCOM

# User Manual

## Index

1.	Introduction.....	3
2.	Advantage .....	3
3.	Use Megawin USB EasyCOM.....	4
3.1.	System Handle Block .....	4
3.2.	Hardware Installation .....	4
3.3.	Software Developing Resource .....	5
3.4.	Firmware Develop Requirements .....	5
3.5.	Firmware Library Install .....	5
3.6.	Firmware Control Flow .....	8
3.7.	Used MCU Resource.....	9
3.8.	Using Device Firmware Upgrade.....	10
3.9.	Limitation .....	10
4.	Example Application.....	11
5.	Obtaining The Parts .....	12
6.	Revision History .....	12

## 1. Introduction

In the PC world, the UART has been the industrial standard port for connection throughout decades. Nowadays, a lot of different connection protocols are getting stronger supports because the performance of UART is not catching up with current requirement.

Despite this, many applications still use COM port as a typical way to communicate with a peripheral device. A challenge remains at hand to boost the performance of this port without sacrificing huge efforts.

The USB (Universal Serial Bus) due to its speed and performance, has become the better choice in comparison with the standard COM port. So, Megawin provides the USB EasyCOM solution for users to transfer their system easily from UART to USB.

This document will show the user how to implement a “**Virtual COM**” solution. **It retains the current setting of COM, but gives more high performance.** In this solution, Megawin USB EasyCOM supports full COM port function including Read/Write data, Line Control/Status and Modem Control/Status.

In addition, MG84FL54B not only makes the device simple and easy to use, but also integrates some powerful peripherals into the same chip, such as: SPI, TWI, and General IO. It helps the user to easily control any device on the system.

## 2. Advantage

USB is on every new computer.

No modification on the PC application.

More flexibility on the USB: data buffering, no data lost, etc.

USB provides the power supply for the application

Window Build-In driver on Win2K, WinXP, Vista32, Vista64.

### 3. Use Megawin USB EasyCOM

#### 3.1. System Handle Block

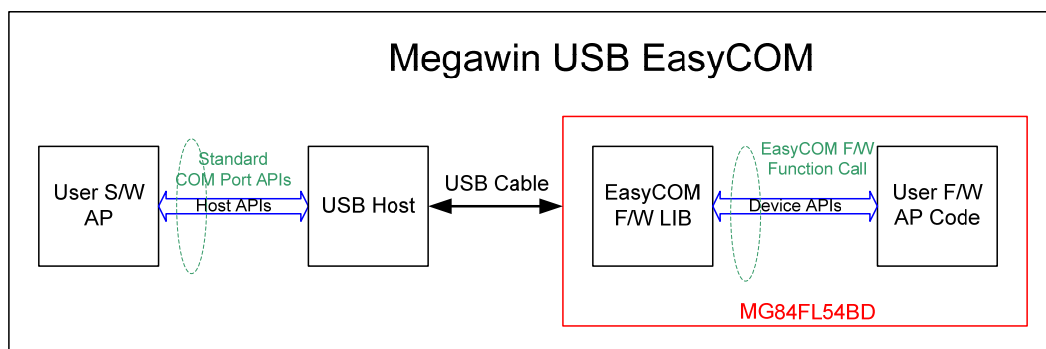


Fig-1

#### 3.2. Hardware Installation

Plug the "**MG84FL54BD Evaluation Stick**" (Please reference the " ~Megawin Easy USB\MG84FL54B\Hardware\Evaluation Stick\MG84FL54B Evaluation Stick, User Manual v1.01.pdf " for more details to confirm the EasyCOM sample code stored in MG84FL54BD Evaluation stick) into a PC's USB port, and instruct Device Wizard where the installation file "**0E6A0316\_USBEasyCOM\_v1.04.inf**" is located. After the driver is successfully installed, user will see the following page in the "**System\Hardware\Device Manager**", and see a new entry (COM<sub>n</sub>) added to the list of available COM ports. (Fig-2)

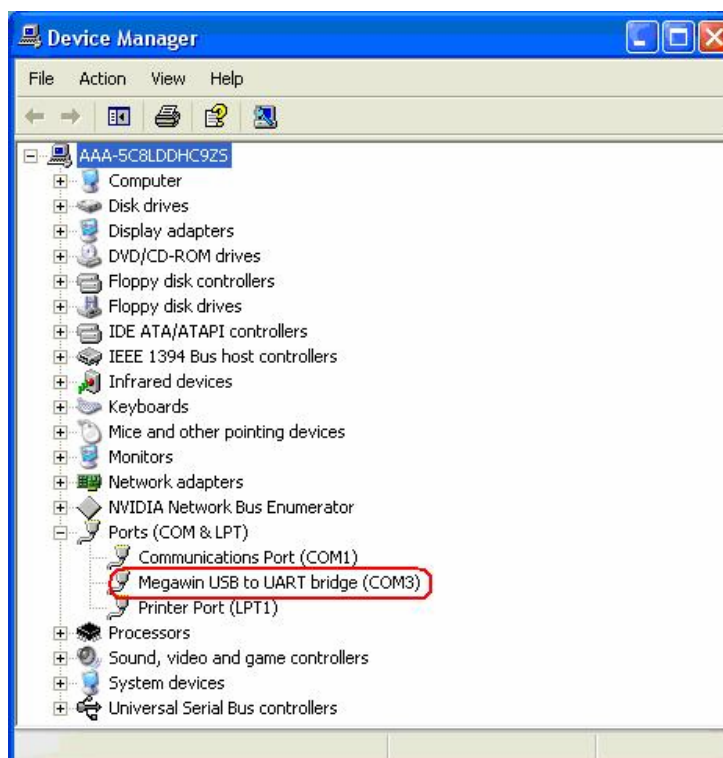


Fig-2

### 3.3. Software Developing Resource

**3.3.1.** INF: 0E6A0316\_USBEasyCOM\_v1.04.inf ( Driver Installation file )

**3.3.2.** API: Used standard COM port APIs in OS defined.

### 3.4. Firmware Develop Requirements

**3.4.1.** EasyCOM.LIB ( Firmware Library File )

**3.4.2.** EasyCOM.H ( Firmware Header File )

**3.4.3.** DFU.EXE ( Device Firmware Upgrade Software )

### 3.5. Firmware Library Install

**3.5.1.** Add the “ EasyCOM.LIB “into your own project. (Fig-3)

3.5.1.1. **InFlag** , in “ Extern.h” (This flag indicates the data has been received in InBuffer)

3.5.1.2. **InLen** , in “ Extern.h” (Indicate the data size in InBuffer)

3.5.1.3. **InBuffer[64]** , in “ Extern.h” (Data Buffer)

3.5.1.4. **Initial();**, in “ Extern.h”(This function enables USB and will be called in user project)

3.5.1.5. **USB\_Read\_Data\_Complete();** , in “ Extern.h” (This function will release buffer for next data transfer from PC)

3.5.1.6. **USB\_Send\_Data\_To\_PC( Len , Buffer );** , in “ Extern.h” (This function will send “Len” Data to PC)

3.5.1.7. **USB\_Event();** , in “ Extern.h “ (Notice the USB power event from Host)

3.5.1.7.1. **Suspend** : Suspend event from the Host

3.5.1.7.2. **Wakeup** : Wakeup event from the Host

3.5.1.7.3. **Reset** : Reset event from the Host

3.5.1.7.4. **EmuOK**: USB enumeration OK

3.5.1.8. **UART\_Event();** , in “ Extern.h “ (Notice the UART Line/Control state from Host ,and return UART state to Host)

3.5.1.8.1. **LC.Flag** : Setting Line coding properties event from Host

- **Com.LC.BaudRate.DW** : Data terminal rate  
(BaudRate), in bits per second. Please refer sample code to get more clear definition easily.

- **Com.LC.StopBit** : 0 for 1 Stop bit, 1 for 1.5 Stop bits, 2 for 2 Stop bits

- **Com.LC.ParityChk** : 0 for None, 1 for Odd, 2 for Even,

3 for Mark, 4 for Space

- **Com.LC.DataBit** : 5, 6, 7, 8, 16. It defines the data bit count in one packet transfer.

3.5.1.8.2. **LS.Flag** : Setting Line state event from Host and “**Com.LS.State**” content indicates the control signal bitmap as following definition:

- Bit0 (DTR) : Indicates to DCE if DCE is present or not , 0 for not present , 1 for present
- Bit1 (RTS) : Carrier control for half duplex modems , 0 for Deactivate carrier , 1 for Activate carrier

3.5.1.8.3. **SB.Flag** : Send break event from Host and “**Com.SB.Time.W**” contains the length of time in milliseconds.

- 0xFFFF : Continuous break
- 0x0000 : Stop break

3.5.1.9. **USB\_Send\_UartState\_To\_PC( State ); , Extern.h** (This function will send Uart State to PC)

3.5.1.9.1. **Com.State** : (bitmap), set “1” for event active.

- Bit0 : “**UART\_RX\_CARRIER**” return the state of carrier detection mechanism of device (DCD)
- Bit1 : “**UART\_TX\_CARRIER**” return the state of transmission carrier (DSR)
- Bit2 : “**UART\_BREAK**” return the state of break detection mechanism of device
- Bit3 : “**UART\_RING\_SIGNAL**” return the state of ring signal detection of device (RI)
- Bit4 : “**UART\_FRAM\_ERROR**” return a framing error has occurred
- Bit5 : “**UART\_PARITY\_ERROR**” return a parity error has occurred
- Bit6 : “**UART\_OVER\_RUN**” return a received data has been discarded due to overrun in the device
- Bit7 : Reserved

**3.5.2.** Include the Header file “**EasyCOM.H**” in the source modules which will use parameter or function call (**Fig-3**) , The following items could be modified by user application.

3.5.2.1. **USB\_VID** , in “**Define.h**” (This VID, 0x0E6Ah, is registered under Megawin Technology Co., Ltd. at **USB-IF**, Any third party needs the written approval from Megawin in order to use this VID)

- 3.5.2.2. **MF\_STRING** , in “ **Define.h**” (Define for Manufacture String are supported)
- 3.5.2.3. **PD\_STRING** , in “ **Define.h**” ( Define for Product String are supported )
- 3.5.2.4. **SN\_STRING** , in “ **Define.h**” ( Define for SerialNumber String are supported )

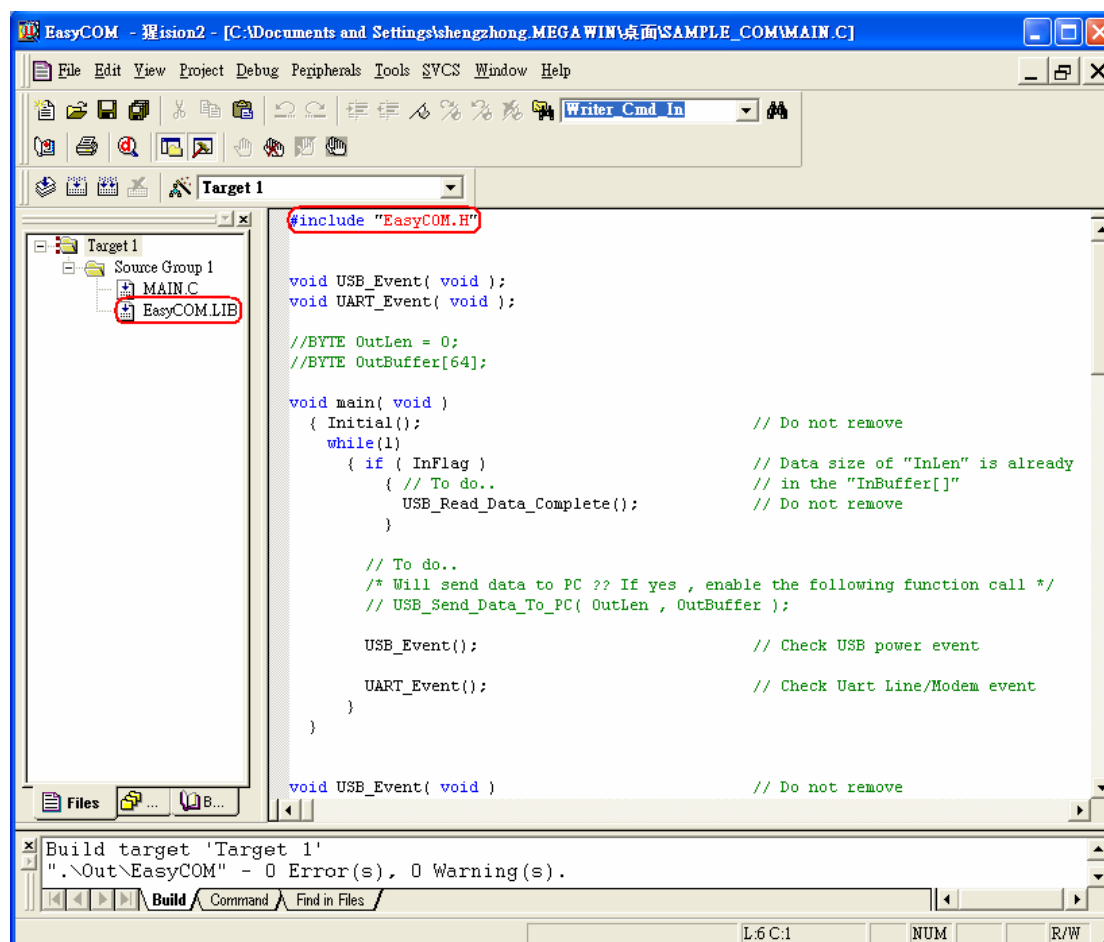
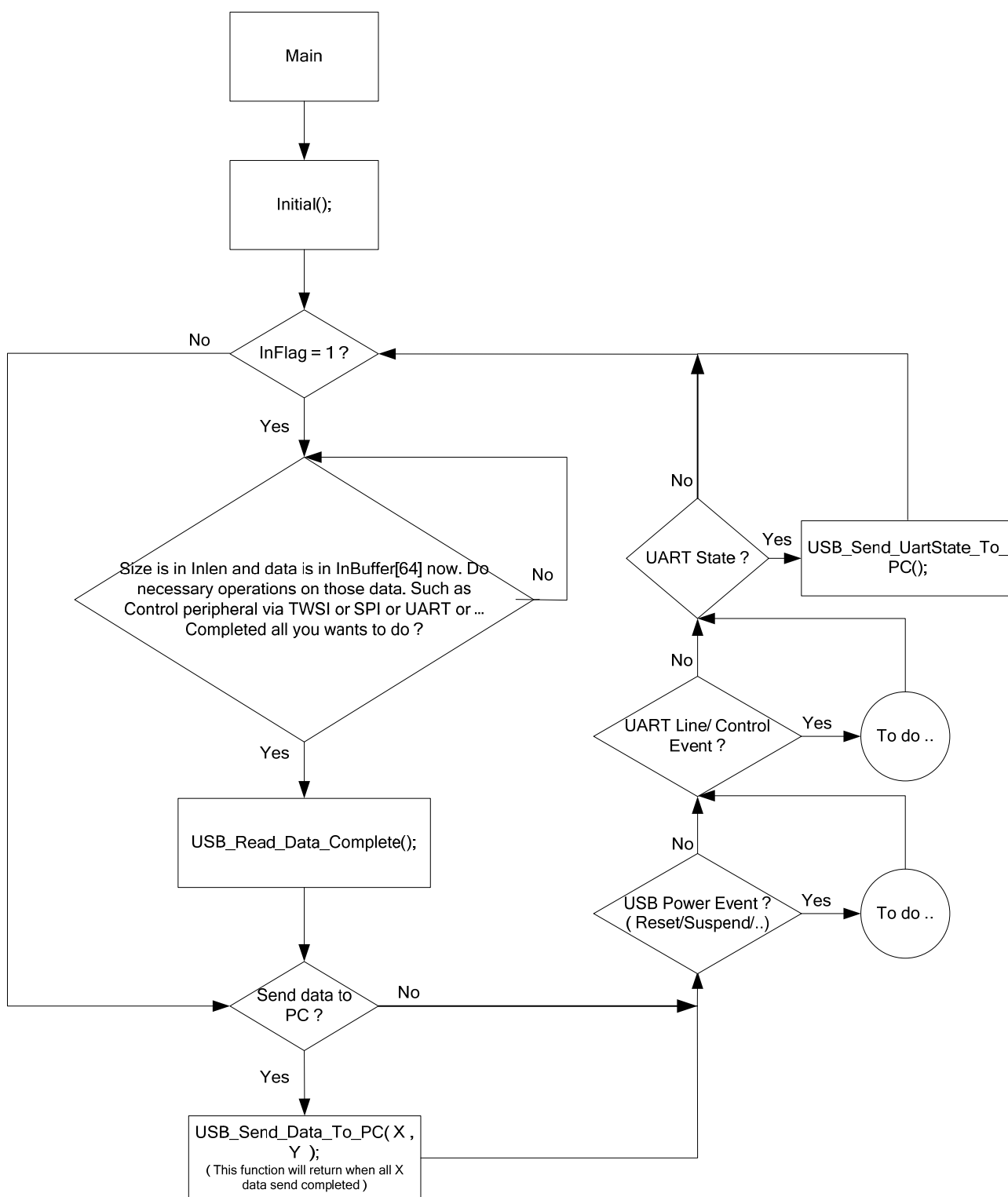


Fig-3 ~\Megawin Easy USB\EasyCOM\SampleCode\MAIN.C

### 3.6. Firmware Control Flow



**Fig-4**



### 3.7. Used MCU Resource

**3.7.1.** Direct Data Memory : 14 bytes

**3.7.2.** Indirect Data Memory : 33 bytes

**3.7.3.** eXternal Data Memory : 64 bytes

**3.7.4.** Sample Code size : 2884 bytes

**3.7.5.** USB ISR use "REG BANK 1"

TYPE	BASE	LENGTH	RELOCATION	SEGMENT NAME
-----				
* * * * * D A T A M E M O R Y * * * * *				
REG	0000H	0008H	ABSOLUTE	"REG BANK 0"
REG	0008H	0008H	ABSOLUTE	"REG BANK 1"
DATA	0010H	0005H	UNIT	?DT?_USB_SEND_DATA_TO_PC?USB
DATA	0015H	0005H	UNIT	_DATA_GROUP_
DATA	001AH	0002H	UNIT	?DT?USB
	001CH	0004H		*** GAP ***
BIT	0020H.0	0000H.1	UNIT	?BI?USB
	0020H.1	0000H.7		*** GAP ***
IDATA	0021H	0013H	UNIT	?ID?USB
IDATA	0034H	000EH	UNIT	?ID?COM
IDATA	0042H	0001H	UNIT	?STACK
* * * * * X D A T A M E M O R Y * * * * *				
	0000H	0200H		*** GAP ***
XDATA	0200H	0040H	ABSOLUTE	

**Fig-5**

### 3.8. Using Device Firmware Upgrade

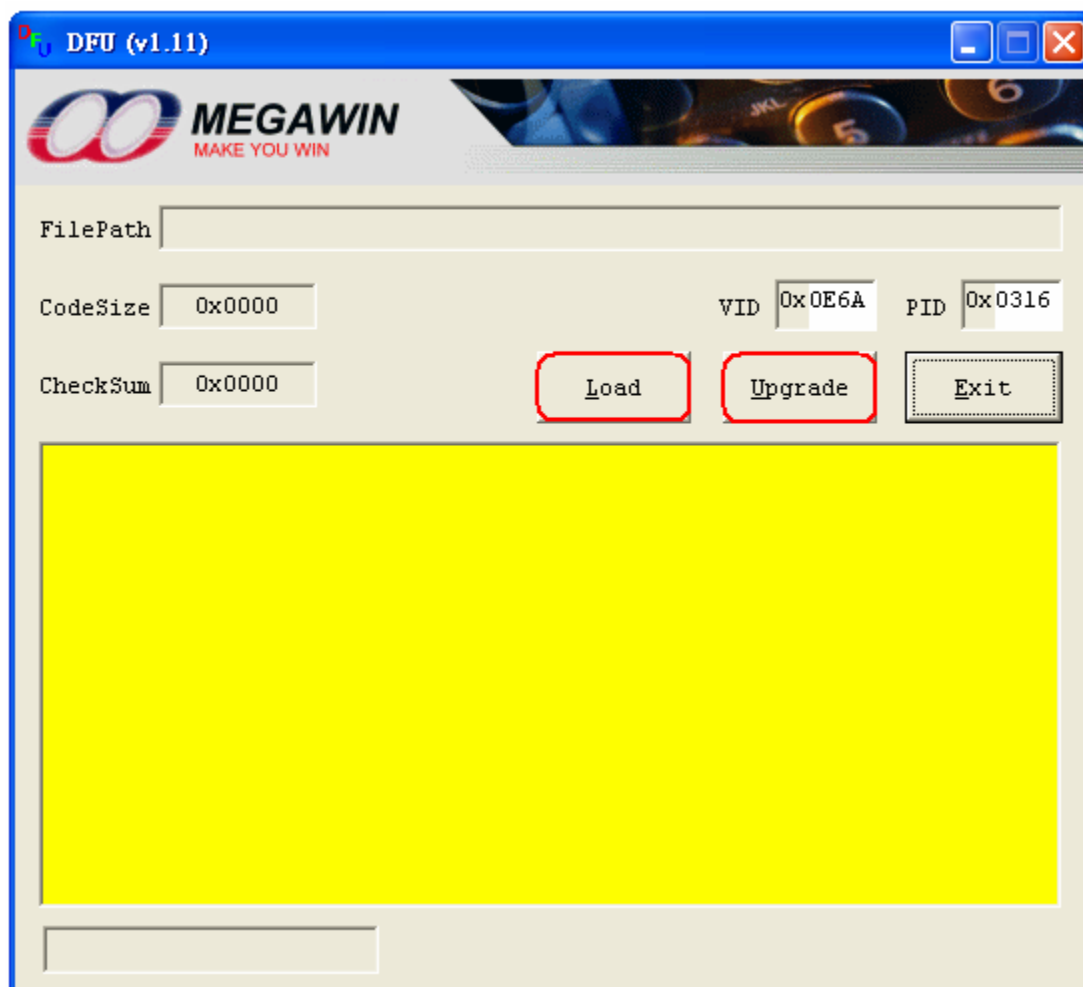
After firmware development, user could update the device firmware through this software tool by following procedures.

**3.8.1.** Press the **DFU ( Reset )** button on “**MG84FL54BD Evaluation Stick**”.

**3.8.2.** Run “ **DFU.EXE (Fig-5)**”

**3.8.3.** “ **Load** ” file which you want to upgrade

**3.8.4.** “ **Upgrade** ” to process upgrade procedure



**Fig-6**

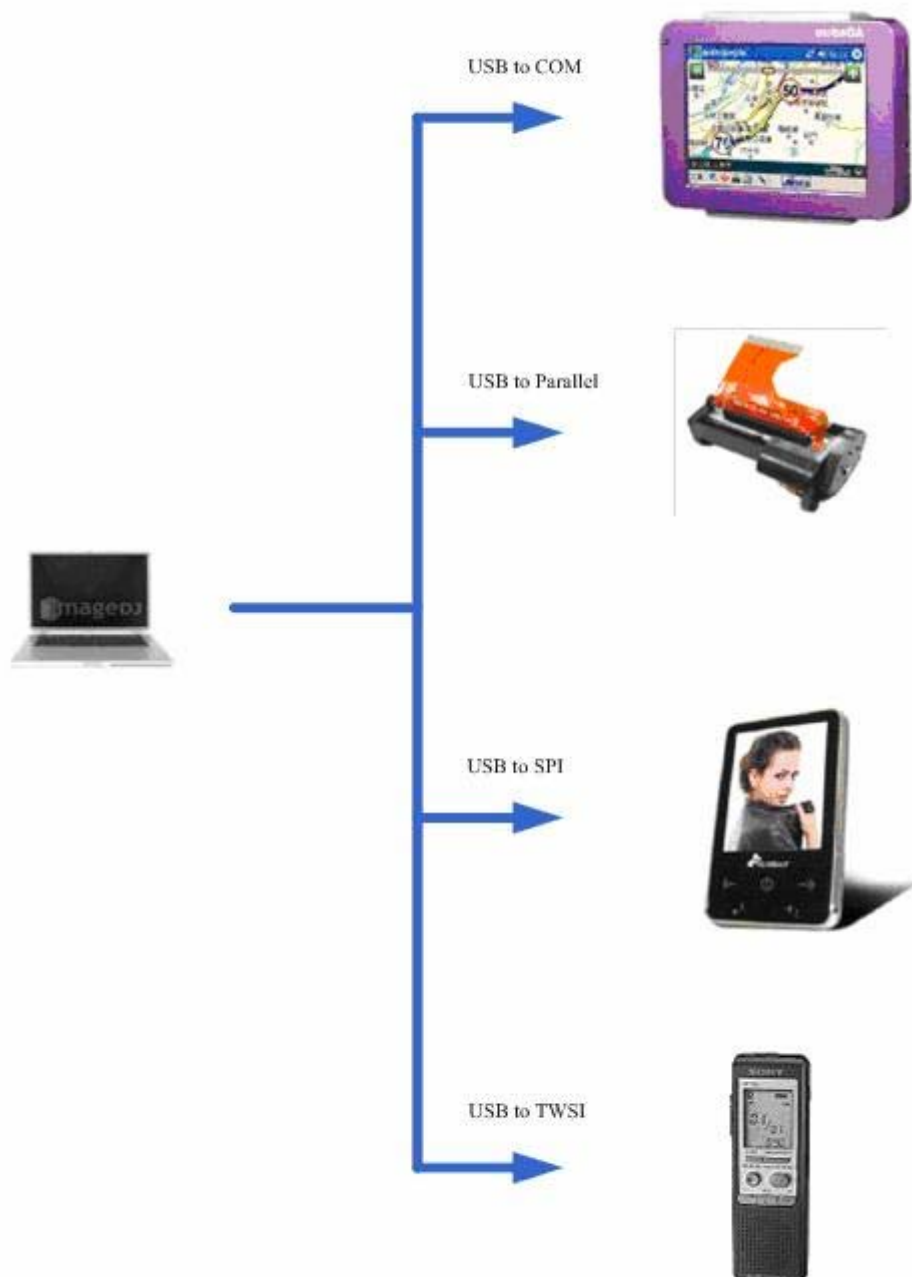
### 3.9. Limitation

**3.9.1.** The application doesn't work with windows98/ME and Linux OS.

**3.9.2.** Crystal input should be force at **12Mhz**. If user wants to use other frequency, Please contact Megawin technical support team.

#### 4. Example Application

- 4.1. USB to MCU GPIO
- 4.2. USB to UART
- 4.3. USB to Parallel Port
- 4.4. USB to SPI
- 4.5. USB to TWSI
- 4.6. USB to user's interface



**Fig-7**

## 5. Obtaining The Parts

The **MG84FL54BD** device comes in a 48 pin surface mount LQFP-48 package. For this solution, it's only requires a **12 MHz** crystal, some passives, and the USB socket. Of course, customers could obtain it from Megawin in Taiwan. Please, visit the Megawin's website at <http://www.megawin.com.tw> for the latest details on pricing and availability

## 6. Revision History

Revision	Description	Date
v1.00	Initial version	2007/09/27
v1.01	Modify " Control Flow "	2007/10/08
v1.02	1. Modify " Control Flow " 2. Add user define " USB_VID/USB_PID/USB/DID "	2007/11/16
v1.03	Modify " Application Circuit " and " Pin assignment "	2007/11/30
v1.04	1. Change driver name ( 0E6A0316_USBEasyCOM_v1.04.inf ) 2. Remove " Application Circuit " and " Pin Assignments"	2008/01/30
v1.05	Notice the USB power event and UART Line/Control event that issued by Host and return the UART state to Host	2008/04/10
v1.06	Notice the USB enumeration OK flag	2008/06/30
v1.07	Update the EasyCOM.LIB in FW_SampleCode folder	2008/12/29