MicrolifeDevice SDK (WBP_O3 2G - Windows)

Table of Contents

Revise history

Date	Document Version	Description
2023/08/04	1.0	First release.
2023/08/08	1.1	Add Chapter 6

Chapter 1 Development Environment

This user manual serves as a quick guide to MicroLifeDeviceSDK / APIs and shows how to integrate into a Windows C# Demo App. Development Environment in the following

Compatible Development Tools	Microsoft Visual Studio 2019 (recommended)
Programming language:	C#
Target framework:	.NET Standard 2.0

Importing steps are described below.

1.1. First of all, add WBP_O3, Connection.dll into a development project.

1.2. Import class as bellows.

```
using WBP_O3.Class;
using Connection.Class;
```

Chapter 2 Connection Sequence of WBP_O3 2G

The WBP_O3 object is applied managing the USB/Bluetooth communication.

2.1 Initiate WBP_O3 Object with API key and set connected/disconnected delegation for WBP_O3.

```
o3 = new WBP_O3.WBP_O3("");
o3.OnConnected += O3_OnConnected;
o3.OnDisConnected += O3_OnDisConnected;
```

2.2 Call InitDeviceWatcher to initialize WBP_O3 monitoring for USB/Bluetooth status.

```
o3.InitDeviceWatcher();
```

2.3 Connection

2.3-1 For USB, the OnConnected delegate will be called automatically when the device is connected to the computer via USB.

2.3-2 For BLE, Call ConnectWithBLE API with device's BLE id, the OnConnected delegate will be called automatically when the device is connected successfully.

2.4 When the data is transferred via USB/Bluetooth. The parsed data will be returned through the command object, which will be explained in section 3-4.

```
var callback = o3.SendMessage(WBP_O3.SendMessage.ToCommand.timeRead());
if (callback.Command.CMD == (byte)WBP_O3.Enum.Command.NACK)
{
    Console.WriteLine("send command failed");
    return;
}
else if (!callback.Success)
{
    Console.WriteLine("No call back,Please try again");
    return;
}
DateTime dateTime = (DateTime)callback.Command.Data;
```

2.5 The OnDisConnected delegate will be called when the device is disconnected.

The following is the sample code.

```
private static WBP_O3.WBP_O3 o3;
0 references
public MainWindow() {
    InitializeComponent();
    o3 = new WBP_O3.WBP_O3(key);
    o3.OnConnected += O3_OnConnected;
    o3.OnDisConnected += O3_OnDisConnected;
    o3.InitDeviceWatcher();
}
4 references
private void O3_OnConnected()
{
}
6 references
private void O3_OnDisConnected()
{
}
```

Chapter 3 APIs of WBP_O3 2G

This chapter will explain the application of each API and the meaning of its parameters.

3-1.WBP_O3 2G Initialize

API	WBP_O3(string key);
Function	Initialize WBP O3 object
Return object	WBP_O3**:** After the initialization is successful, a WBP_O3 object will be created.
Parameter	**key**: API Key is required to use subsequent APIs, if you do not have it, please contact Microlife **** If the API key is incorrect, you will receive a "Key Fail!" exception..**

3-2 Initiate and wait for device to connect

API	void InitDeviceWatcher()
Function	Initiate and wait for device to connect
Return object	None
Parameter	None

3-3 Connect with device by BLE

API	Task<ConnectionResult> ConnectWithBLE(string deviceID, bool TryToPair = false);
Function	Use this API to connect with device by BLE..
Return object	**ConnectionResult:** A class containing connection result, described in 4-1-10
Parameter	**DevuceID**: Bluetooth ID of the device to be connected. **TryToPair**: Whether the device needs to attempt pairing**.**

3-4.Send Command to device

API	(bool Success, Command Command) **SendMessage**(Message message, Func<Command, bool> predicate = null, int retry = 3, bool ResetCommand = true, int timeout = 0);
Function	Transmit message to the Device.
Return object	**Success:** Indicates whether the command was successfully written to the device. **Command**: A class containing parsed data and message which sent to the device, described in 4-1-1. **** If the Success is true, and the Command.CMD is not NACK, the device has successfully received this command.**
Parameter	**Message**: A class containing message which sent to the device. **Predicate**: If the command requires additional data, it will be provided here. **Retry**: Retry times

	ResetCommand: Whether to clear the last command **timeout**: timeout in milliseconds **** The contents of the above parameters are generated by the following API.**

3-4-1. Read all history data from BPM

Command	**dataReadAll**
API	SendMessage(WBP_O3.SendMessage.ToCommand.**dataReadAll**());
Return Command Data Type.	The object type of Command.data is MData class, which is a class containing parsed BP data, described in 4-1-2.

3-4-2. Read central BP memory data by index from BPM

Command	**dataRead**
API	SendMessage(WBP_O3.SendMessage.ToCommand.**dataRead**(int index, DataFormat format););
Parameter	**Index**:Data index. **format**: A Enum of format, described in 4-2-2.
received from **OnReceived** delegate.	**command.CMD**: dataRead **command.Data**: The object type of Command.data is Data class, which is a class containing CBPdata & CalCBP data, described in 4-1-3.

3-4-3. Clear the all history data of the BPM

Command	**dataClear**
API	SendMessage(WBP_O3.SendMessage.ToCommand.**dataClear**());
Return Command Data Type.	None. **** If the Success is true, and the Command.CMD is not NACK, the device has successfully received this command.**

3-4-4. Read user ID and version data from BPM.

Command	**userInfoRead**
API	SendMessage(WBP_O3.SendMessage.ToCommand.**userInfoRead**());
Return Command Data Type.	The object type of Command.data is UserInfo class, which is a class containing parsed user's information data, described in 4-1-4. **** Because the machine has two versions of two schedules and five schedules, if the Protocol ID of Userinfo class is less than V1.05, it is represented as a two schedule machine.**

3-4-5. Write a new user ID to BPM.

Command	**userIDWrite**
API	SendMessage(WBP_O3.SendMessage.ToCommand.**userIDWrite**(string ID));
Return Command Data Type.	None. **** If the Success is true, and the Command.CMD is not NACK, the device has successfully received this command.**
Parameter	**ID**: User ID string to be written, maximum 30 characters

3-4-6. Read Two Schedule Setting.

**** Only works if protocol id is less than v.1.05**

Command	**settingRead_ABPM**
API	SendMessage(WBP_O3.SendMessage.ToCommand.**settingRead_ABPM** ());
Return Command Data Type.	The object type of Command.data is Setting_ABPM class, which is a class containing parsed ABPM setting data, described in 4-1-5.

3-4-7. Write Two Schedule Setting.

**** Only works if protocol id is less than v.1.05**

Command	**settingWrite_BPM**
API	SendMessage(WBP_O3.SendMessage.ToCommand.**settingWrite_ABPM** (Setting_ABPM setting));
Return Command Data Type.	None. **** If the Success is true, and the Command.CMD is not NACK, the device has successfully received this command.**
Parameter	**setting**: BPM Setting to be written, described in 4-1-5.

3-4-8. Read device ID and info from BPM.

Command	**deviceIDRead**
API	SendMessage(WBP_O3.SendMessage.ToCommand.**deviceIDRead**());
Return Command Data Type.	The object type of Command.data is DeviceID_and_Info class, which is a class containing parsed Device ID & info data, described in 4-1-6.

3-4-9. Read device Time from BPM.

Command	**timeRead**
API	SendMessage(WBP_O3.SendMessage.ToCommand.**timeRead** ());
Return Command	The object type of Command.data is DateTime.

Data Type.	

3-4-10. Write device Time to BPM.

Command	**timeWrite**
API	SendMessage(WBP_O3.SendMessage.ToCommand.**timeWrite**(DateTime dateTime));
Return Command Data Type.	None. **** If the Success is true, and the Command.CMD is not NACK, the device has successfully received this command.**
Parameter	**dateTime**: DateTime to be written.

3-4-11.Read BPM function setting value from BPM.

Command	**settingRead_BPM_Function**
API	SendMessage(WBP_O3.SendMessage.ToCommand.**settingRead_BPM_Function**());
Return Command Data Type.	The object type of Command.data is Setting_BPM_Function class, which is a class containing parsed BPM function data. These values are set at the factory and are only available for reading and cannot be rewrite, described in 4-1-7.

3-4-12. Read BTmodule name from BPM.

Command	**bluetoothnameRead**
API	SendMessage(WBP_O3.SendMessage.ToCommand.**bluetoothnameRead**());
Return Command Data Type.	The object type of Command.data is string.

3-4-13. Read SN.

Command	**deviceSNRead**
API	SendMessage(WBP_O3.SendMessage.ToCommand.**deviceSNRead**());
Return Command Data Type.	The object type of Command.data is string.

3-4-14. Read Five Schedule Setting.

**** Only works if protocol id is bigger than or equal to v.1.05**

Command	**settingRead_FiveSchedule**
API	SendMessage(WBP_O3.SendMessage.ToCommand. **settingRead_FiveSchedule**());

Return Command Data Type.	The object type of Command.data is ABPM_5ScheduleObj class, which is a class containing parsed ABPM_5Schedule setting data, described in 4-1-8.

3-4-15. Write Five Schedule Setting.

**** Only works if protocol id is bigger than or equal to v.1.05**

Command	**settingWrite_FiveSchedule**
API	SendMessage(WBP_O3.SendMessage.ToCommand**.settingWrite_FiveSchedule** (ABPM_5ScheduleObj setting));
Return Command Data Type.	None. **** If the Success is true, and the Command.CMD is not NACK, the device has successfully received this command.**
Parameter	**setting**: ABPM Setting to be written, described in 4-1-8.

3-4-16. Disconnect the bluetooth.

Command	**bluetoothDisconnect**
API	SendMessage(WBP_O3.SendMessage.ToCommand**.bluetoothDisconnect**);
Return Command Data Type.	None. **** Because after the command is sender, the Bluetooth will be disconnected immediately, and it can no longer be sent back through Bluetooth, so there will be no return value.**

3-5 Connect with USB/BLE event

API	event ConnectedDelegate OnConnected
Function	When the USB or Bluetooth connection is successful, this event will be triggered
Return object	event

3-6 Disconnect with USB/BLE event

API	event DisConnectedDelegate OnDisConnected
Function	When the USB or Bluetooth becomes disconnected, this event will be triggered
Return object	None

Chapter 4 Class & Object of WBP_O3 2G

4-1 Class

4-1-1.Command Class

Name:	**Command**
Definition	A class containing parsed data and message sent to the device.
members	**byte CMD:** Record the current command, if it is NACK, it means the transmission failed. **byte Device:** Record the device. **object Data:** parsed data

4-1-2.MData Class

Name:	**MData**
Definition	A class containing parsed BP data.
members	**int Index:** Data index **Condition Condition:** A Enum of Condition, described in 4-2-1. **int Systole:**The value of systole ** If the Condition is EmptyBattery or BPError, these three values are 0. ** If Heart rate>=240 or <30, Hr will be 0, and note HI(Heart rate>=240), LO(Heart rate <30) in Code. **int DiaStole:**The value of diastole **int Hr:** heart rate **DateTimeOffset Time:** Record time **ABPM ABPM: Deprecated** **bool LowBattery:** true：low battery detected(battery contains < 4.75V) **bool Anti_Artifact:** true：anti-artifact detected **bool Start_of_a_manual_measurement:** true：start of a manual measurement detected **bool Afib:** true：afib detected **int MAP:**The value of MAP. **int Mean_CBP_data:**The value of mean CBP data. **int PVR_length:**The value of PVR length. **bool CBP_Error:** Record if this is a CBP error. **int CSBP:** The value of CSBP **int CDIA:** The value of CDIA. **int CPP:**The value of CPP. **bool WithPVPWave:** true：PVP Wave included in measurement. **List<string> Code:** List of error codes or special cases, described in 4-3-1

4-1-3. Data Class

Name:	**Data**
Definition	A class containing parsed CBP and CalCBP data.
members	**DataFormat Format:** A Enum of format, described in 4-2-2 **List<int> CBPData:** A list of CBP Data. **List<int> CalCBP:** A list of CalCBP Data.

4-1-4. UserInfo Class

Name:	**UserInfo**
Definition	A class containing parsed user's information data.
members	**string UserID:** User ID, maximum 30 characters **string ProtocolID:** Protocol ID, if the Protocol is less than V1.05, it is represented as a two schedule machine. **int MaxMemory:** Maximum of memory data can be saved for every user. **bool IHB:** true:IHB Enbale **string FMVersion:** FW version in BPM, send the ASCII code **DateTimeOffset FMDate:** The release date of firmware. **bool CBP:** true:CBP Enbale. **float BatteryVoltage:** Voltage of the device battery. **bool Afib:** true:AFIB Enbale **arrhythmiaDisplay arrhythmiaDisplay:** Display arrhythmia name in software. It may be one of NONE, IHB, PAD described in 4-2-3

4-1-5. Setting_ABPM

Name:	**Setting_ABPM**
Definition	A class containing parsed ABPM function data.
members	**Int ABPMStart_1:** The starting time of the first measurement time zone (valid range: 0~23) **Int ABPMEnd_1:** The end time of the first measurement time zone (valid range: 0~23) **Int ABPMInt_1:** The interval of the first measurement time zone (valid range: 5, 10, 15, 20, 30, and 60) **Int ABPMInt_2:** The interval of the second measurement time zone (valid range: 5, 10, 15, 20, 30, and 60) **int HI_infPressure:** Highest inflation pressure Valid parameter: 0(not setting),140, 160, 180, 200, 220, 240, 260, 280 **bool CBP_zone1_meas_off:** t true: the first time zone of CBP measurement disabled

	false: the first time zone of CBP measurement enabled **bool CBP_zone2_meas_off:** t true: the second time zone of CBP measurement disabled false: the second time zone of CBP measurement enabled **bool SW_SEL_silent:** true: Beeper is disabled. false: Beeper is enabled **bool SW_checkhide:** true: Don't show readings after measurement. false: Show readings after measurement **Int CBPInt_1:** The interval of the CBP first measurement time zone (valid range: 5, 10, 15, 20, 30, and 60) **** CBPInt_1 should multiple times than ABPMInt_1** **Int CBPInt_2:** The interval of the CBP second measurement time zone (valid range: 5, 10, 15, 20, 30, and 60) **** CBPInt_2 should multiple times than ABPMInt_2**

4-1-6. DeviceID_and_Info Class

Name:	**DeviceID_and_Info**
Definition	A class containing parsed Device ID and information data.
members	**string ID:** Device ID string **ConnectType ConnectType:** A Enum of ConnectType which would be one of the following, described in 4-2-4 BothOfUSBAndBT: Device support USB & BT. USBOnly: Device only support USB. **int Mea_times:** Total number of measurements **int[] ErrorCount:** The array of the total number of errors is recorded in the order of error1,error2,error3,error5,errorF. **In some devices, the length or ErrorCoun will be 6, the value by the order will become error1, error2, error3, error5, errorF**, **error 6 (error 5 in ABPM mode.)** Device's error 5 count times = error5 + **error 6.**

4-1-7. Setting_BPM_Function Class

Name:	**Setting_BPM_Function**
Definition	A class containing parsed Device ID and information data.
members	**bool Fun_SEL_AFib:** true: AFIB algorithm ON. false: AFIB algorithm OFF.

	bool Fun_SEL_CBP_algo: true: CBP algorithm ON. false: CBP algorithm OFF. **AMPM Fun_SEL_AMPM:** A Enum of AMPM setting, described in 4-2-5. o24: only 24hr o12: select 24hr or 12hr by UI **bool Fun_SEL_CBP_CL:** true: Enable CBP clinical data transmission. false: Disable CBP clinical data transmission. **bool Fun_SEL_AutoTest:** true: AutoTest ON. false: AutoTest OFF. **bool Fun_SEL_Bluetooth:** true: Bluetooth function ON. false: Bluetooth function OFF. **bool Fun_SEL_UnitKpa:** true: select Kpa or mmHg by UI. false: only mmHg. **bool Fun_SEL_RS232:** true: LabView data transmission ON. false: LabView data transmission OFF.

4-1-8. ABPM_5ScheduleObj Class

Name:	**ABPM_5ScheduleObj**
Definition	A class containing parsed ABPM_5Schedule setting data.
members	**int HI_infPressure:** Highest inflation pressure Valid parameter: 0(not setting),140, 160, 180, 200, 220, 240, 260, 280 **bool SW_checkhide:** true: Don't show readings after measurement. false: Show readings after measurement **ABPMObj ABPM_period_1~ ABPM_period_5:** A class containing parsed ABPM_setting data for each period, described in 4-1-9.

4-1-9. ABPMObj Class

Name:	**ABPMObj**
Definition	A class containing parsed ABPM_setting data for each period.
members	**int start:** The starting hour of the period valid range: 0~23 **int end:** The end hour of the period

	valid range: 0~23 **int interval:** The interval of the period valid range: 5,10,15,20,30,60 **bool silent:** true: reminder disabled false: reminder enabled ****If Protocol ID<107, device will enable beeper before reminder.**** **bool BP_meas_off:** true: Disable this period BP measurement. false: enable this period BP measurement **bool CBP_meas_off:** true: Disable this period CBP measurement. false: enable this period CBP measurement **bool disable:** true: Disable this period. false: Enable this period. **Period rule: 1. start and end value should not equal in the same period. (exclude 255) 2. Period should not cross the other period: If "PX End Hour" > "PX Start Hour" AND "PX Start Hour" < "the other period Start Hour" < "PX End Hour" fail If 'PX End Hour" > "PX Start Hour" AND "PX Start Hour" < "the other period End Hour" < "PX End Hour" fail If "PX End Hour" < "PX Start Hour" AND ("the other period Start Hour" ≧ "PX Start Hour" 0R "the other period Start Hour" < "PX End Hour") fail If "PX End Hour" < "PX Start Hour" AND ("the other period End Hour" > "PX Start Hour" 0R "the other period End Hour" < "PX End Hour") fail

4-1-10. ConnectionResult Class

Name:	**ConnectionResult**
Definition	A class containing Bluetooth connection status
members	**string DeviceId:** Device ID string

	string Name: Device's BLE module name **bool IsConnected:** true: Device is connected. **bool HasError:** true: Device has error. **string ErrorMessage:** Error message string

4-2 Enum

4-2-1 Condition

Condition	
0	BP
1	BP + CBP
2	Empty battery
3	Error code for original BP
4	stop measurement by manual
5	Pill memo
6	device switch off
7	re-start during a 24h-profile
10	Retry 1st -BP
11	Retry 1st -BP+CBP
12	Retry 1st -Empty battery
13	Retry 1st -Error code for original BP
14	Retry 1st -stop measurement by manual
20	Retry 2nd (BP)
21	Retry 2nd (BP+CBP)
22	Retry 2nd -Empty battery
23	Retry 2nd - Error code for original BP
24	Retry 2nd - stop measurement by mamual

4-2-2 DataFormat

DataFormat		Note
0	No_CBP_Raw_Data	No CBP raw data
1	Low_resolution_CBP_data	low resolution CBP data (sampling rate =16Hz)
3	Full_CBP_raw_data	full CBP raw data (sampling rate=256Hz)

4-2-3 arrhythmiaDisplay

arrhythmiaDisplay	
0	NONE
1	IHB
2	PAD

4-2-4 ConnectType

ConnectType	
0x55	USBOnly
0x42	BothOfUSBAndBT

4-2-5 AMPM

AMPM	
0	o24
1	o12

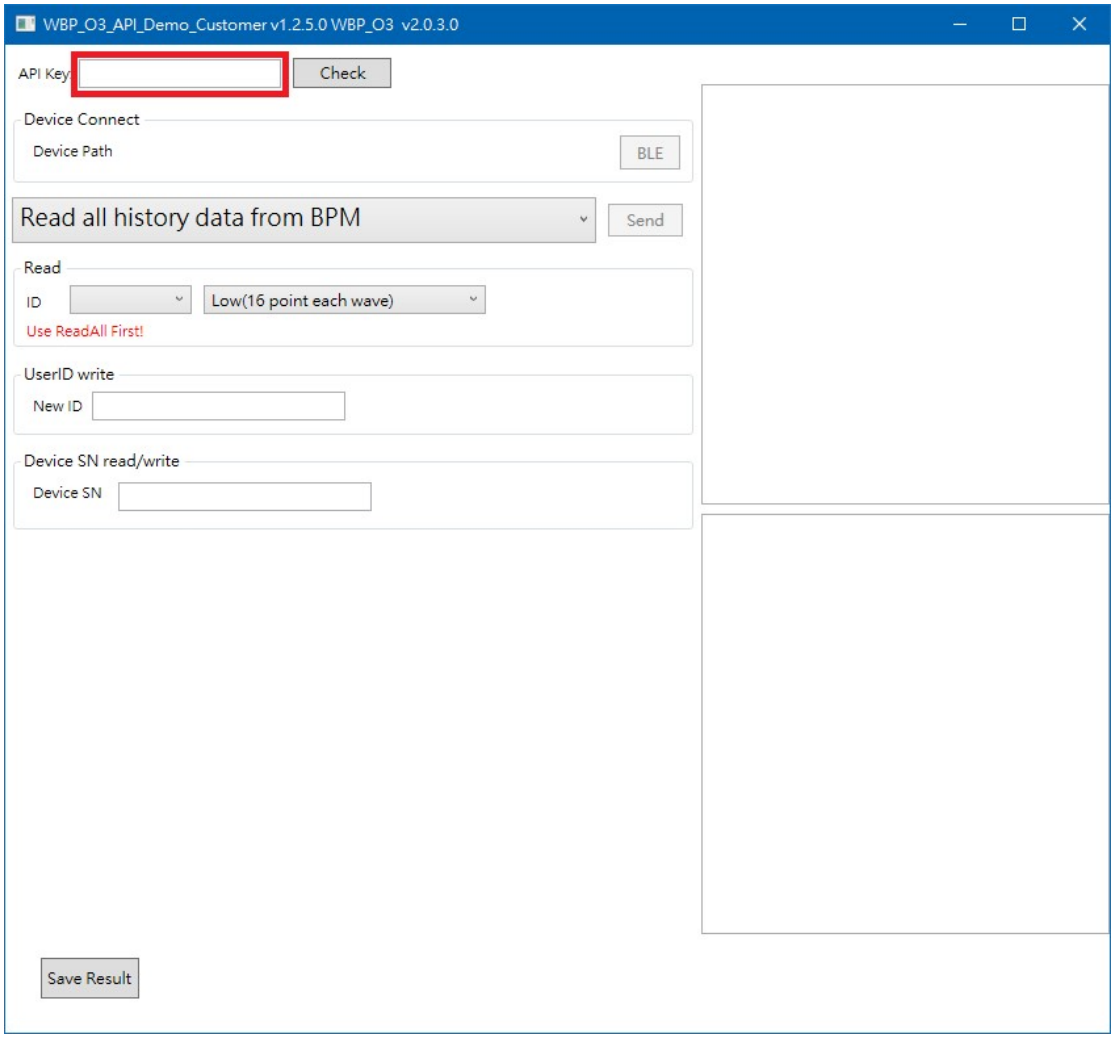
4-3 list

4-3-1 Code

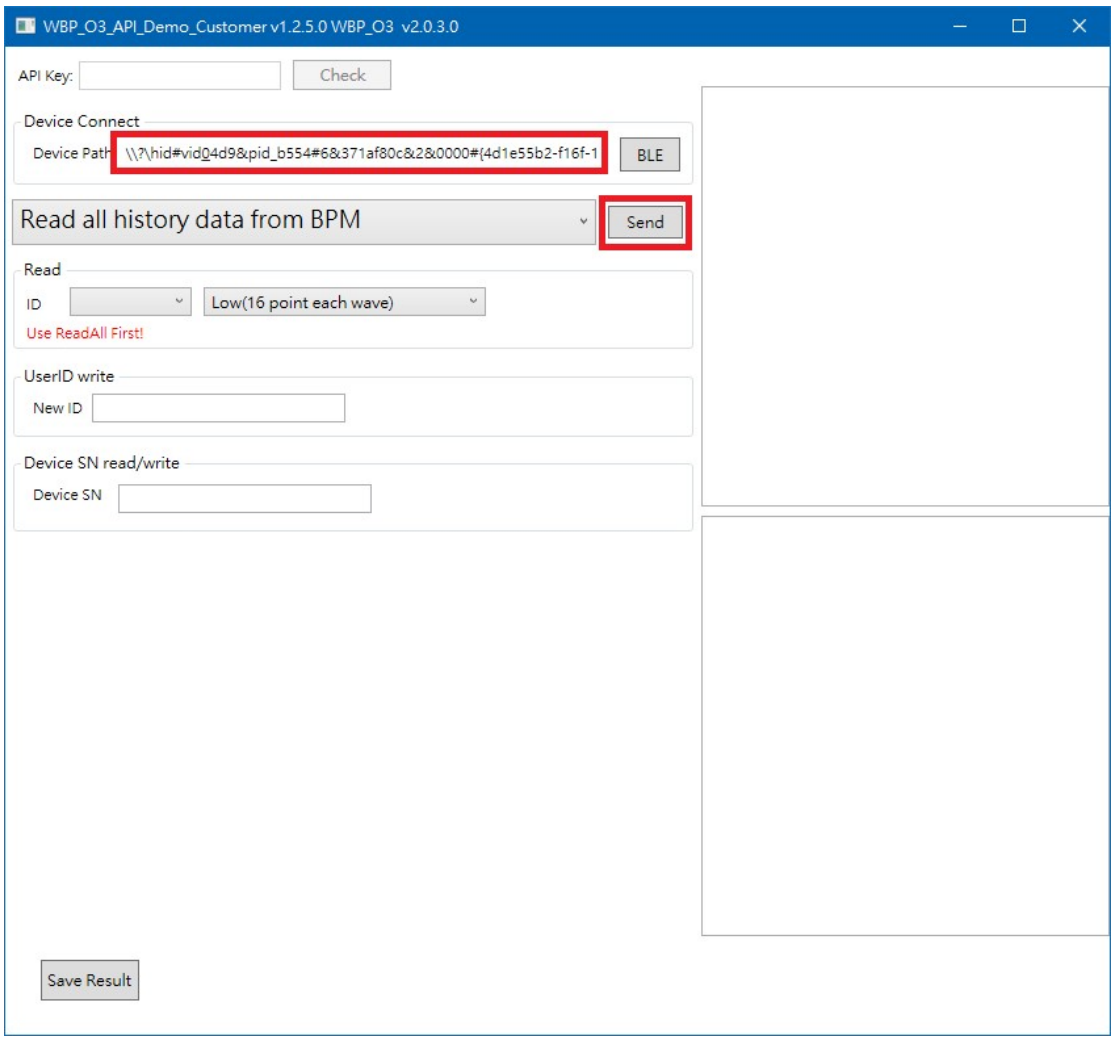
Value	Description
1	Pill memo
2	re-start during a 24h-profile
3	device switch off
4	Low Battery, the battery contains < 4.75V
5	Empty Battery, the battery contains < 4.5V
6	Retry 1st -BP
7	stop measurement by manual
8	start of a manual measurement
LO	Hr<30
HI	Hr>=240
ER 1	Signals is too weak
ER 2	Error signal
ER 3	No pressure in the cuff
ER 5	Abnormal result
ER A	Flash error
ER F	reach of maximum 30 min of measurements
ER 11	Signal too weak during central blood pressure measurement
ER 12	Error signal during central blood pressure measurement
ER 13	Cuff pressure errors during central blood pressure measurement
ER 15	Abnormal result of central blood pressure reading
ER T	Transmission error

Chapter 5 Instruction of Demo App

5-1. Input the API Key on the API Key textbox and click “Check” button to active the demo.

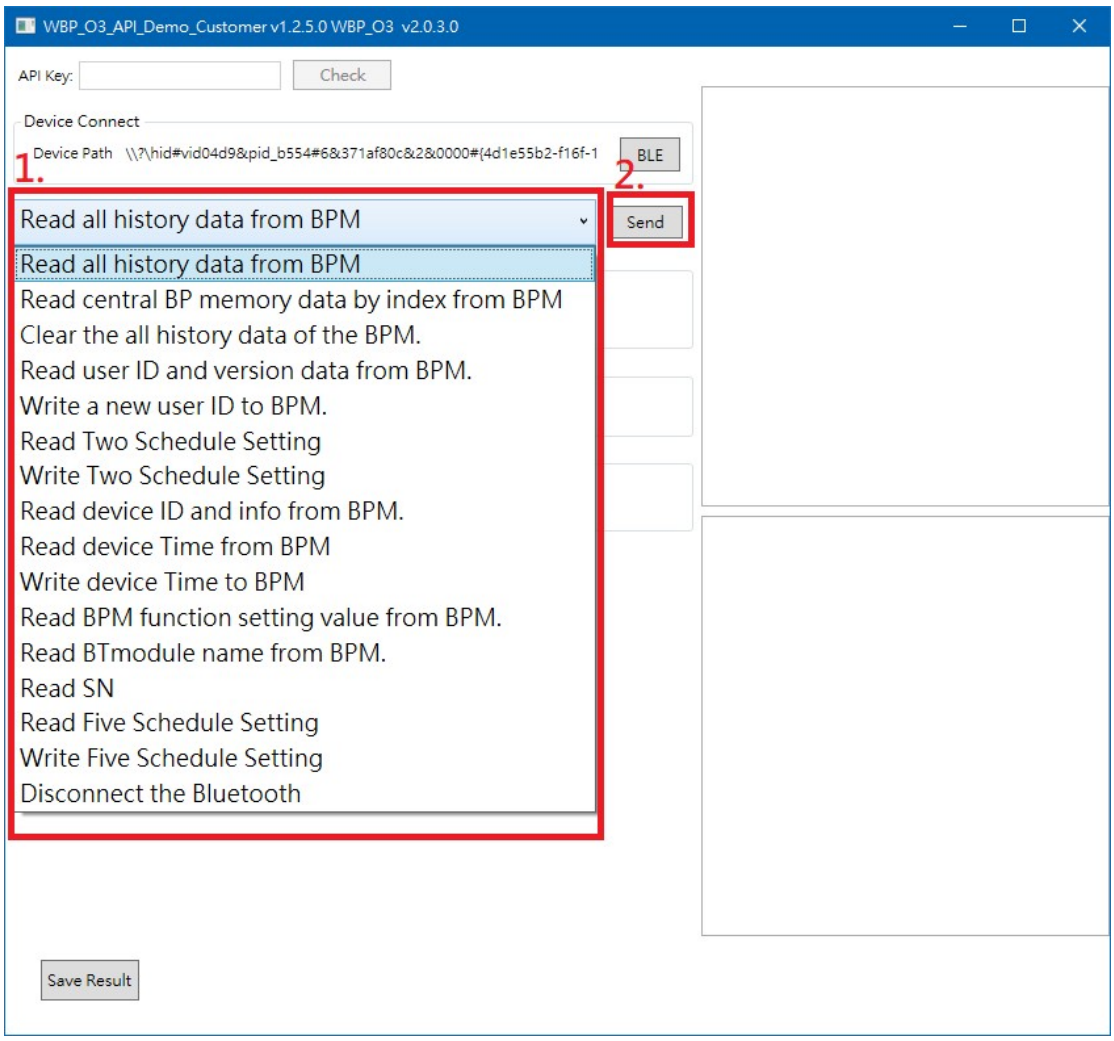


5-2. If the device is connected, the Send button will be enabled and Device Path will display the path



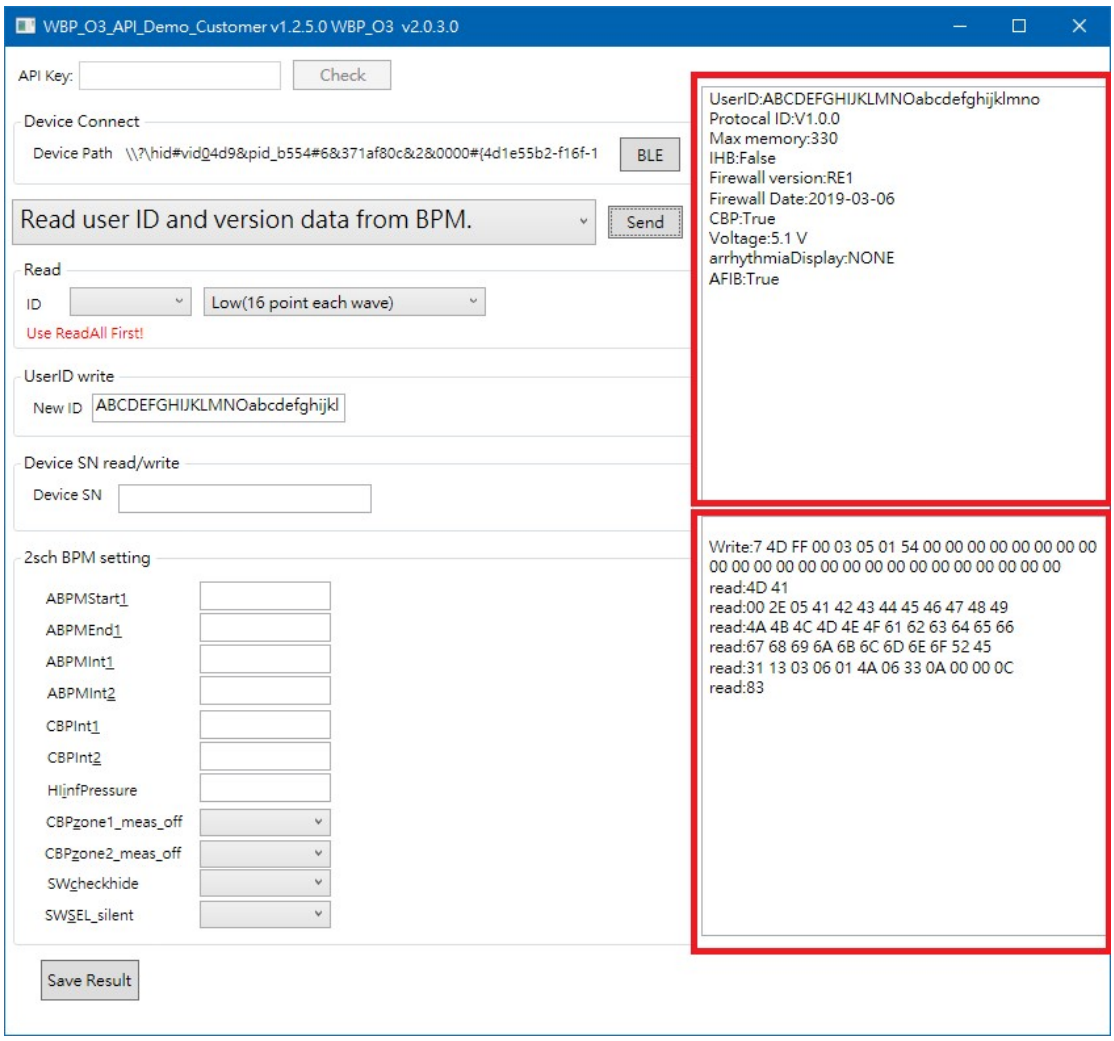
Similarly, if the device is disconnected in anytime, the Send button will be disabled.

5-3. Select the command on the combobox and click the “Send” button to send it.

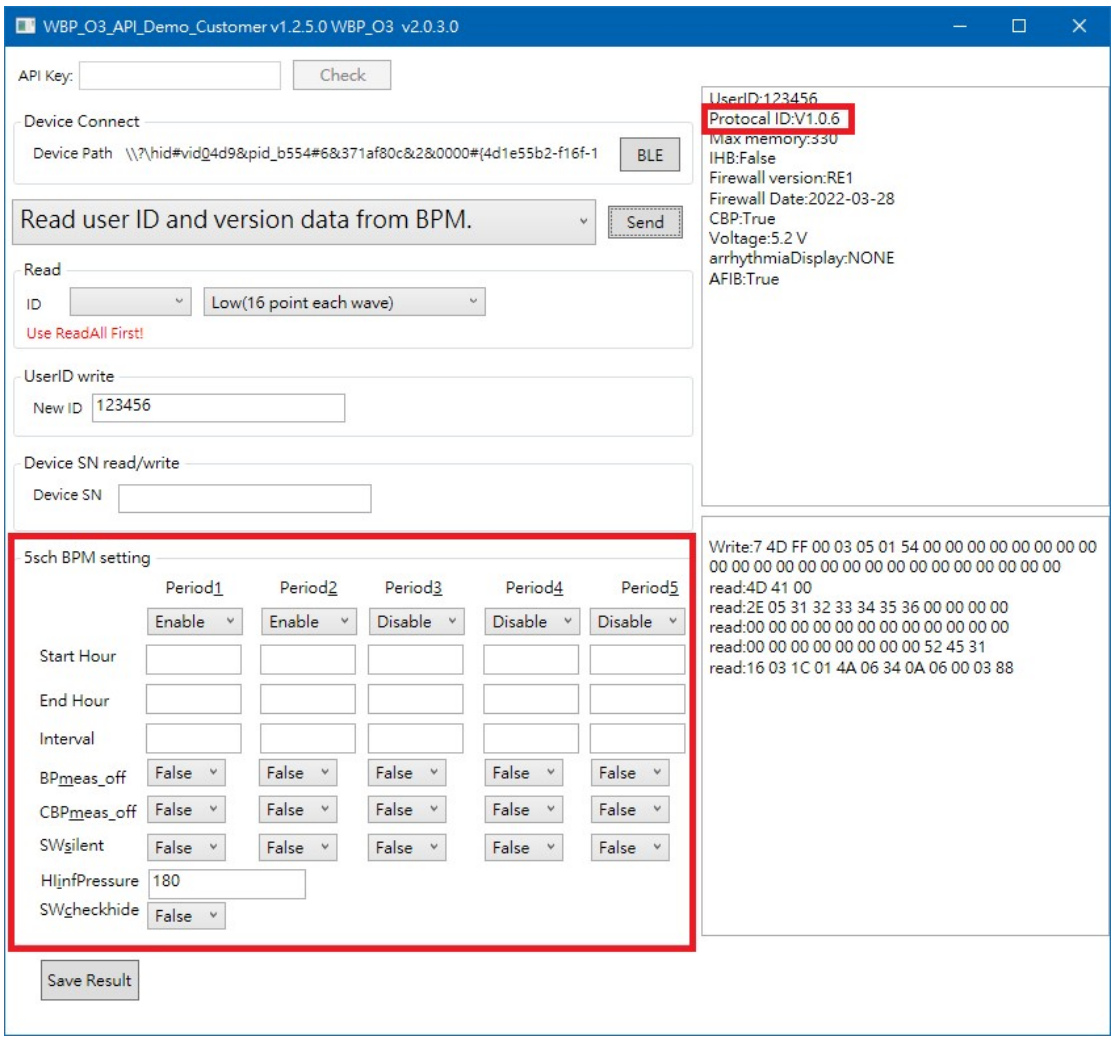


The output will be displayed in the textbox on the right

In addition to the data that has been parsed, the original raw data will be displayed in the bottom.



Because there are two models of two-schedules and five- schedules, it needs to be judged by the version number. When the version number is less than 1.05, it will be judged as two-segment. For details, please refer to the Demo sample source code and section 3-4-4.


WBP_O3_API_Demo_Customer v1.2.5.0 WBP_O3 v2.0.3.0
API Key:
Check
Device Connect
Device Path \\?\hid#vid04d9&pid_b554#6&371af80c&2&0000#{4d1e55b2-f16f-1
BLE
Read user ID and version data from BPM.
Send
Read
ID
Low(16 point each wave)
Use ReadAll First!
UserID write
New ID
123456
Device SN read/write
Device SN
5sch BPM setting
Period1
Period2
Period3
Period4
Period5
Enable
Enable
Disable
Disable
Disable
Start Hour
End Hour
Interval
BPmeas_off
CBPmeas_off
SWsilent
HIinfPressure
180
SWcheckhide
False
Save Result
UserID:123456
Protocal ID:V1.0.6
Max memory:330
IHB:False
Firewall version:RE1
Firewall Date:2022-03-28
CBP:True
Voltage:5.2 V
arrhythmiaDisplay:NONE
AFIB:True
Write:7 4D FF 00 03 05 01 54 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
read:4D 41 00
read:2E 05 31 32 33 34 35 36 00 00 00 00
read:00 00 00 00 00 00 00 00 00 00 00 00
read:00 00 00 00 00 00 00 00 52 45 31
read:16 03 1C 01 4A 06 34 0A 06 00 03 88

Chapter 6 The description for each command of Demo App

6-1. Read all history data from BPM

Index: 1

Condition: 1

Systole: 120

DiaStole: 84

Hr: 60

Time: 2017-04-20 00:00:00

ABPM: ABPM2

LowBattery: False

Anti_Artifact: False

Start_of_a_manual_measurement: False

Afib: True

MAP: 96

Mean_CBP_data: 741

PVR_length: 256

CBP_Error: False

CSBP: 109

CDIA: 81

CPP: 28

WithPVPWave: True

Code:

The description is as follows

(a)Index:

Data index

(b)Condition:

The result could be one of the following.

Condition	Description
0	BP
1	BP + CBP
2	Empty battery
3	Error code for original BP
4	stop measurement by manual
5	Pill memo
6	device switch off
7	re-start during a 24h-profile
10	Retry 1st -BP
11	Retry 1st -BP+CBP

12	Retry 1st -Empty battery
13	Retry 1st -Error code for original BP
14	Retry 1st -stop measurement by manual
20	Retry 2nd (BP)
21	Retry 2nd (BP+CBP)
22	Retry 2nd -Empty battery
23	Retry 2nd - Error code for original BP
24	Retry 2nd - stop measurement by mamual

(b) Systole, DiaStole, Hr

Blood pressure and heart rate values

** If the Condition is EmptyBattery or BPError, these three values are 0.

** If Heart rate>=240 or <30, Hr will be 0, and note HI(Heart rate>=240), LO(Heart rate <30) in Code.

(c)Time:

Record time

(d) ABPM:

Deprecated

(e) LowBattery:

true：low battery detected(battery contains < 4.75V)

(f)Anti_Artifact

true：anti-artifact detected

(g)Start_of_a_manual_measurement

true：start of a manual measurement detected

(h)Afib

true：afib detected.

(i)MAP

The value of MAP.

(j) Mean_CBP_data

The value of mean CBP data.

(k)PVR_length

The value of PVR length.

(l)CBP_Error

true：Record if this is a CBP error.

(m)CSBP, CDIA, CPP

The value of CSBP, CDIA, CPP.

(n)WithPVPWave

true：PVP Wave included in measurement.

(o)Code

The results of the Code received are as follows

Code	Description	Corresponding Condition
1	Pill memo	5
2	re-start during a 24h-profile	7
3	device switch off	6
4	Low Battery, the battery contains < 4.75V	
5	Empty Battery, the battery contains < 4.5V	2,12,22
6	Retry 1st -BP	10,11,12,13,14,20,21,22,23,24
7	stop measurement by manual	4,14,,24
8	start of a manual measurement	
LO	Hr<30	
HI	Hr>=240	
ER 1	Signals is too weak	1,3,11,13,21,23
ER 2	Error signal	1,3,11,13,21,23
ER 3	No pressure in the cuff	1,3,11,13,21,23
ER 5	Abnormal result	1,3,11,13,21,23
ER A	Flash error	1,3,11,13,21,23
ER F	reach of maximum 30 min of measurements	1,3,11,13,21,23
ER 11	Signal too weak during central blood pressure measurement	1,3,11,13,21,23
ER 12	Error signal during central blood pressure measurement	1,3,11,13,21,23
ER 13	Cuff pressure errors during central blood pressure measurement	1,3,11,13,21,23
ER 15	Abnormal result of central blood pressure reading	1,3,11,13,21,23
ER T	Transmission error	1,3,11,13,21,23

6-2. Read central BP memory data by index from BPM

The results of the data format received are as follows

(a)Format:

The result could be one of the following.

No_CBP_Raw_Data:

Low_resolution_CBP_data: (sampling rate =16Hz)

Full_CBP_raw_data: (sampling rate=256Hz)

6-3. Clear the all history data of the BPM.

The result could be one of the following.

ACK

No call back,Please try again

Memory have clear

6-4. Read user ID and version data from BPM.

The results of the data received are as follows

UserID:ABCDEFGHIJKLMNOabcdefghijklmno
Protocal ID:V1.0.0
Max memory:330
IHB:False
Firewall version:RE1
Firewall Date:2019-03-06
CBP:True
Voltage:5.4 V
arrhythmiaDisplay:NONE
AFIB:True

The description is as follows

(a) UserID:

User ID , maximum 30 characters

(b) Protocal ID:

Protocol ID, if the Protocol is less than V1.05, it is represented as a two schedule machine. Others are five schedule.

(c) Max memory:

Maximum of memory data can be saved for every user.

(d) IHB:

true:IHB Enbale

(e) Firmware version:

The version of firmware.

(f) Firmware Date:

The release date of firmware.

(g) CBP:

true:CBP Enbale.

(h) Voltage:

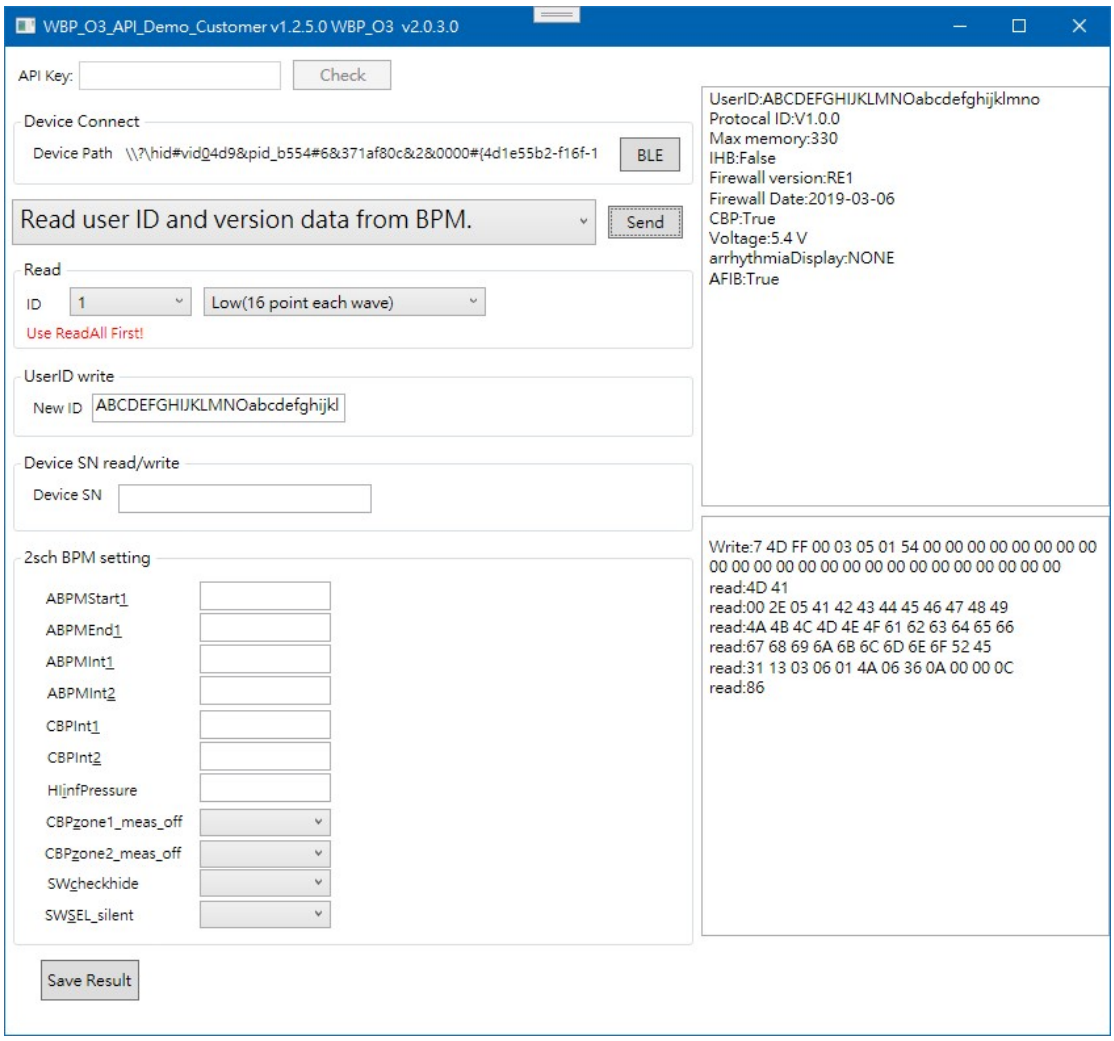
Voltage of the device battery.

(i) arrhythmiaDisplay:

Display arrhythmia name in software. It may be one of NONE, IHB, PAD.

(j) AFIB:

true:AFIB Enbale



6-5. Write a new user ID to BPM.

The result could be one of the following.

NACK

No call back,Please try again

The new ID was written successfully

6-6. Read Two Schedule Setting

Please refer to 6-4, if the Protocol ID is less than 1.0.5, it means it is a two schedule model.

The results of the data received are as follows

ABPMStart_1: 6

ABPMEnd_1: 22

ABPMInt_1: 30

ABPMInt_2: 60

HI_infPressure: 0

CBP_zone2_meas_off: False

CBP_zone1_meas_off: False

SW_SEL_silent: True
SW_checkhide: True
CBPInt_1: 30
CBPInt_2: 60

The description is as follows

(a) ABPMStart_1:

The starting time of the first measurement time zone.

(b) ABPMEnd_1:

The end time of the first measurement time zone

(c) ABPMInt_1:

The interval of the first measurement time zone

(d) ABPMInt_2:

The interval of the second measurement time zone

(e) HI_infPressure:

Highest inflation pressure

(f) CBP_zone1_meas_off:

true: the first time zone of CBP measurement disabled

false: the first time zone of CBP measurement enabled

(g) CBP_zone2_meas_off:

true: the second time zone of CBP measurement disabled

false: the second time zone of CBP measurement enabled

(h) SW_SEL_silent:

true: Beeper is disabled.

false: Beeper is enabled

(i) SW_checkhide:

true: Don't show readings after measurement.

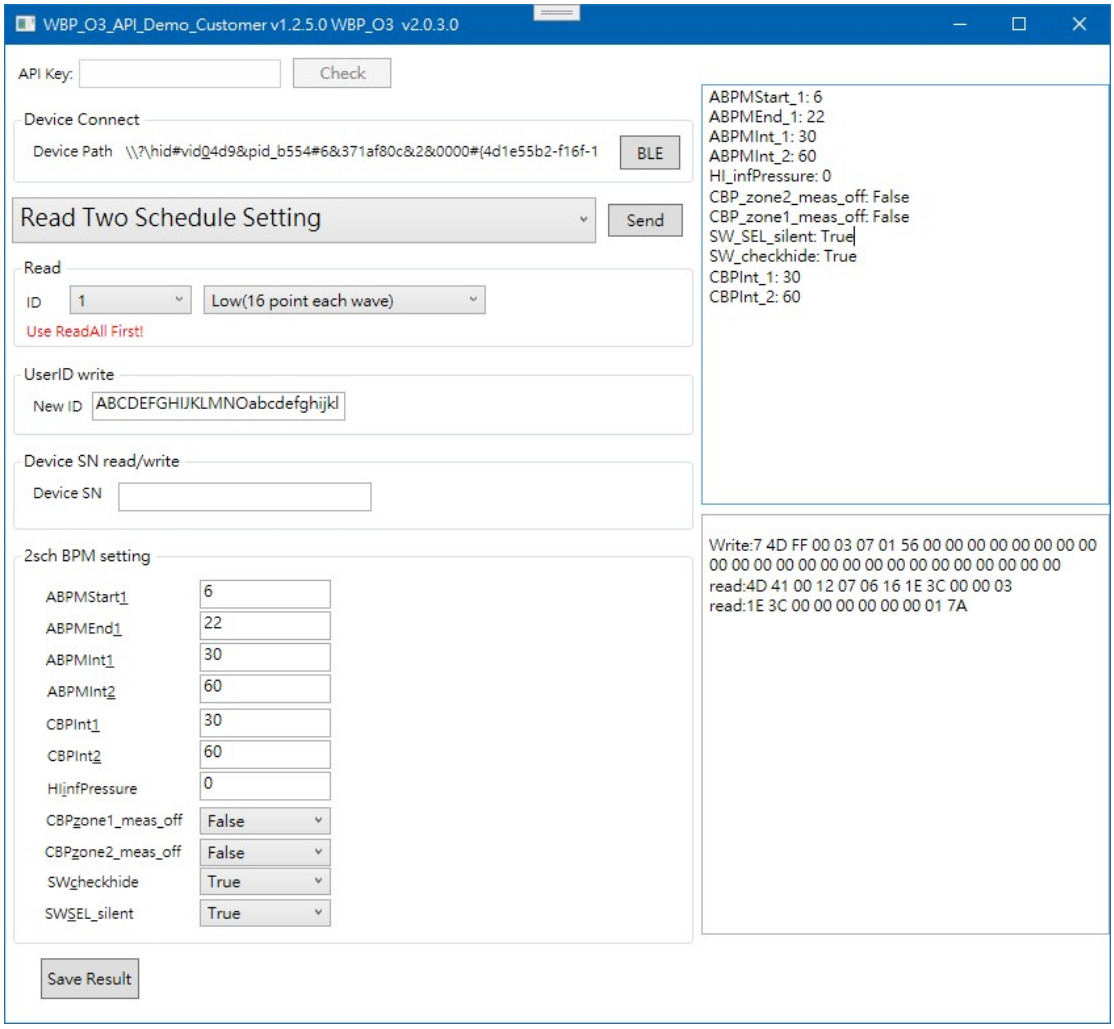
false: Show readings after measurement

(j): CBPInt_1:

The interval of the CBP first measurement time zone

(k): CBPInt_2:

The interval of the CBP second measurement time zone



6-7. Write Two Schedule Setting

The result could be one of the following.

NACK

No call back,Please try again

Write Success

6-8. Read device ID and info from BPM.

The results of the data received are as follows

ABPMStart_1: 6 ID: F0CDAC606C39

ConnectType: BothOfUSBAndBT

Mea_times: 2981

ErrorCount:

2, 0, 360, 0, 0, 0

The description is as follows

(a) ID:

Device ID string

(b) ConnectType:

BothOfUSBAndBT

USBOnly

(c) Mea_times:

measurement times.

(d) ErrorCount:

The total number of occurrences of error 1, error 2, error 3, error 5,error 6.

6-10. Read device Time from BPM

The time of device will be sent back, if no time is set, NACK will be sent back.

6-11. Write device Time to BPM

The result could be one of the following.

NACK

No call back,Please try again

Time had been sync

6-10. Read BPM function setting value from BPM.

Fun_SEL_AFib: True

Fun_SEL_CBP_algo: True

Fun_SEL_AMPM: o24

Fun_SEL_CBP_CL: False

Fun_SEL_AutoTest: False

Fun_SEL_Bluetooth: True

Fun_SEL_UnitKpa: True

Fun_SEL_RS232: False

The description is as follows

(a) Fun_SEL_AFib :

true: AFIB algorithm ON.

false: AFIB algorithm OFF.

(b) Fun_SEL_CBP_algo :

true: CBP algorithm ON.

false: CBP algorithm OFF.

(c) Fun_SEL_AMPM:

o24: only 24hr

o12: select 24hr or 12hr by UI

(d) Fun_SEL_CBP_CL:

true: Enable CBP clinical data transmission.

false: Disable CBP clinical data transmission.

(e) Fun_SEL_AutoTest:

true: AutoTest ON.

false: AutoTest OFF.

(f) Fun_SEL_Bluetooth :

true: Bluetooth function ON.

false: Bluetooth function OFF.

(g) Fun_SEL_UnitKpa :

true: select Kpa or mmHg by UI.

false: only mmHg.

(h) Fun_SEL_RS232:

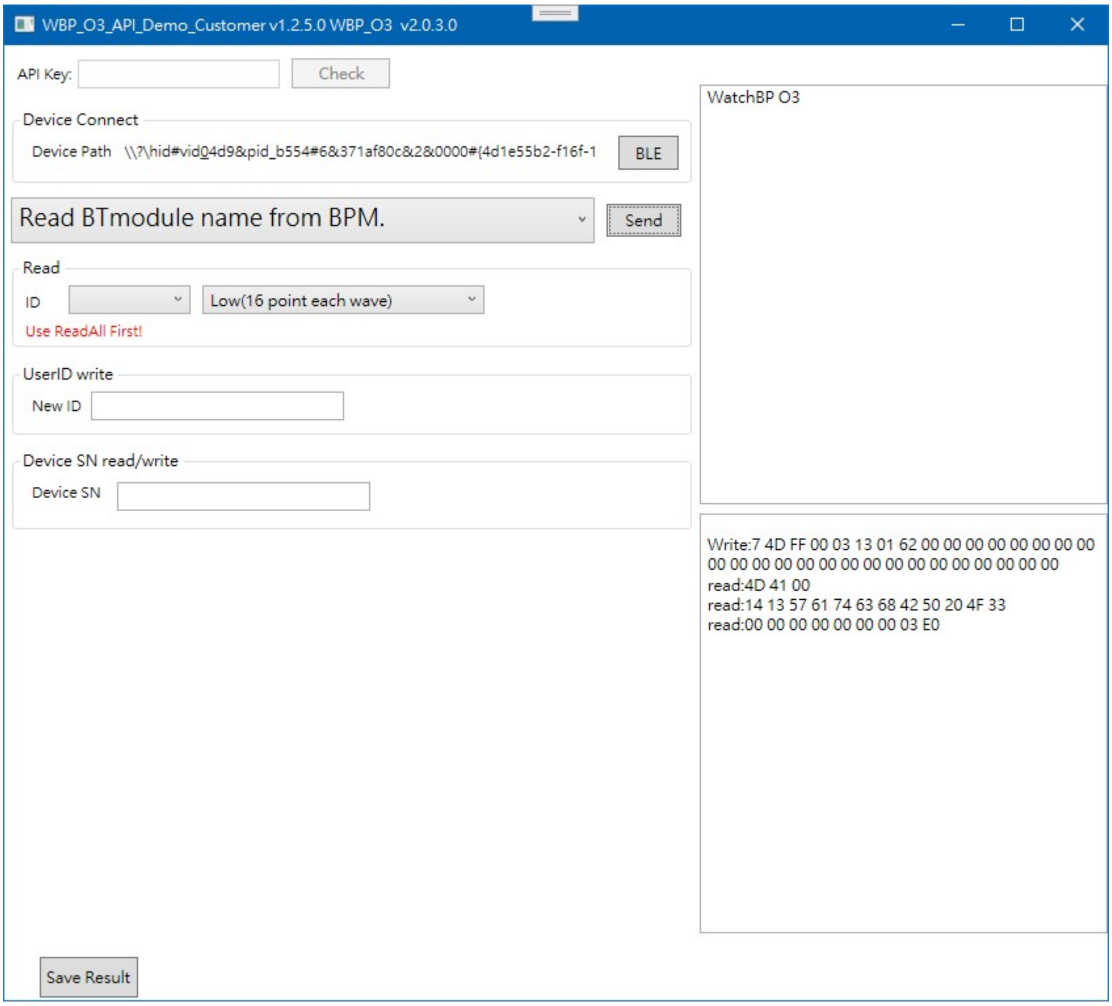
true: LabView data transmission ON.

false: LabView data transmission OFF.

6-11. Read BTmodule name from BPM.

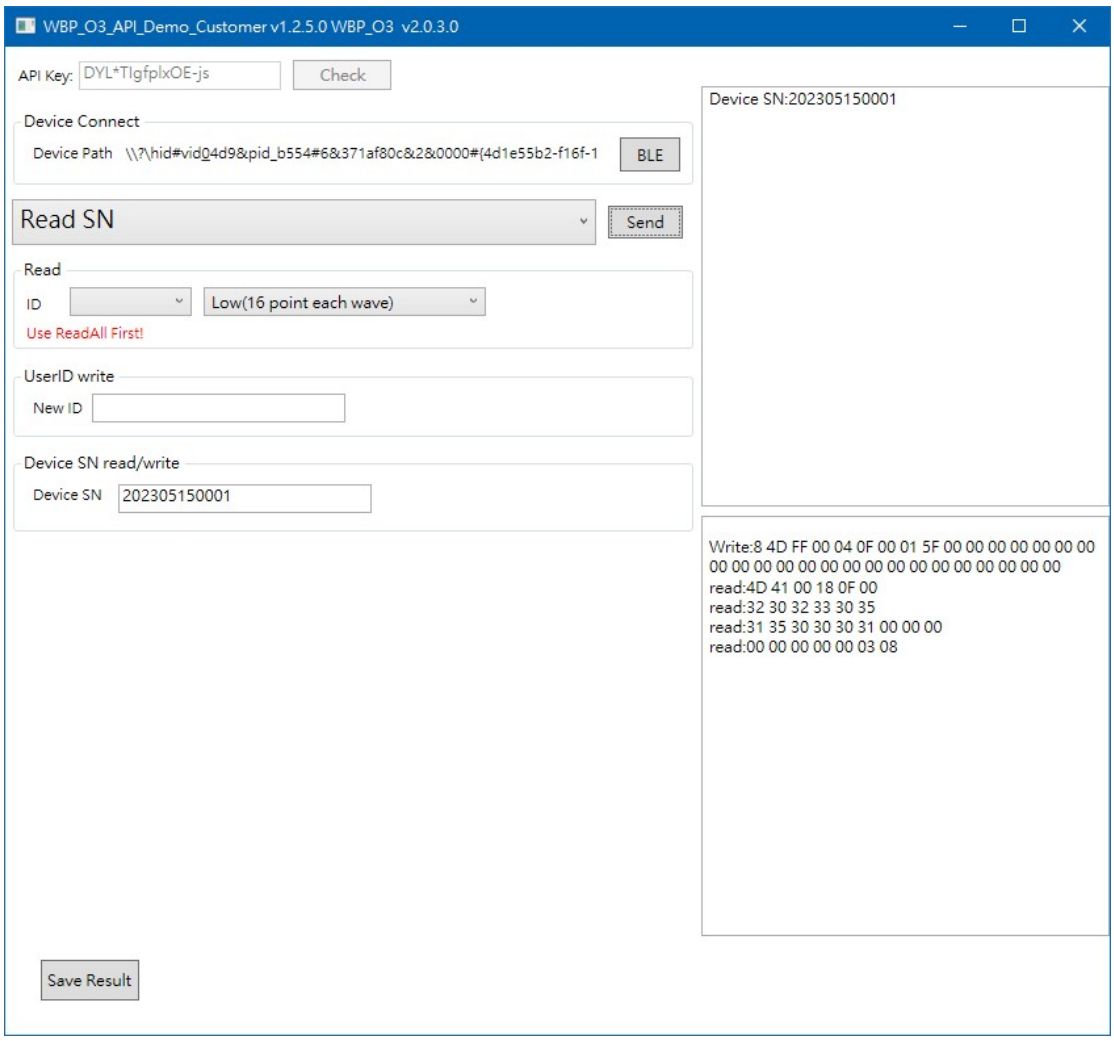
The results of the data received are as follows

The name of the BT module will be "WatchBP O3"



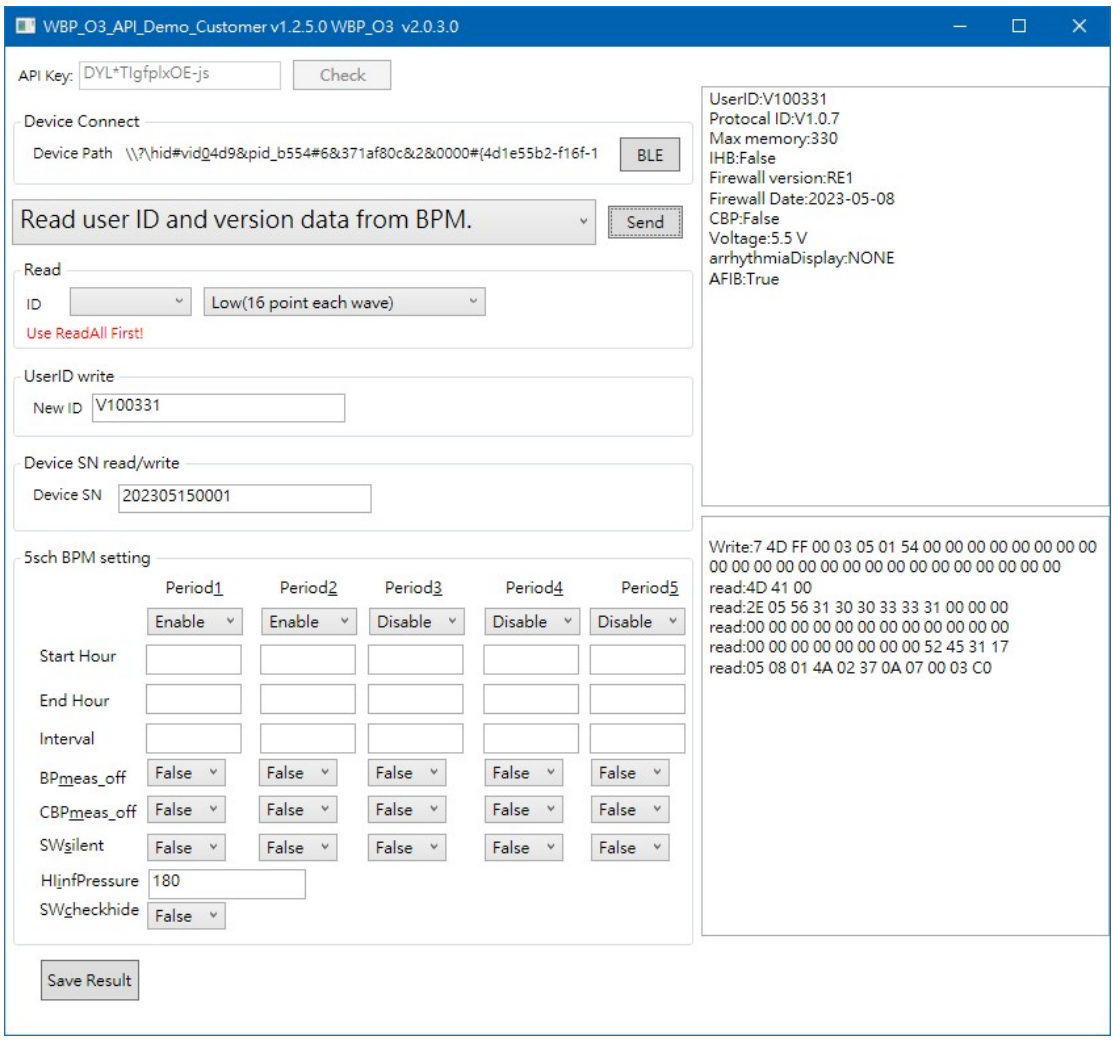
6-11. Read SN.

The results of the data received are as follows



6-12. Read Five Schedule Setting

The results of the data received are as follows. Because there are too many parameters in Five Schedule Setting, all of them are directly represented by UI. The corresponding relationship can directly refer to the source code of the demo program.



6-13. Write Five Schedule Setting

The result could be one of the following.

NACK

No call back,Please try again

The new setting was written successfully

6-14. Disconnect the Bluetooth

The result of the data received is None.

** Because after the command is sender, the Bluetooth will be disconnected immediately, and it can no longer be sent back through Bluetooth, so there will be no return value.