

# INSTRUCTION MANUAL



## **BMP5 Direct SDK**

Revision: 8/06

Copyright © 2004-2006  
Campbell Scientific, Inc.



# ***License For Use***

---

This BMP5 Direct Software Development Kit software, hereinafter referred to as the BMP5 Direct SDK, is protected by both United States copyright law and international copyright treaty provisions. The installation and use of this software constitutes an agreement to abide by the provisions of this license agreement. The term "developer" herein refers to anyone using this BMP5 Direct SDK.

The core operational files included with this BMP5 Direct SDK (hereinafter referred to as "BMP5 Direct Binaries") include the files: SimplePB.DLL and coralib3d.dll. Developer may distribute or sell their software including the BMP5 Direct Binaries subject to the terms hereafter set forth.

## ***RELATIONSHIP***

Campbell Scientific, Inc. hereby grants a license to use BMP5 Direct Binaries in accordance with the license statement above. No ownership in Campbell Scientific, Inc. patents, copyrights, trade secrets, trademarks, or trade names is transferred by this Agreement. Developer may use these BMP5 Direct Binaries to create as many applications as desired and freely distribute them. Campbell Scientific, Inc. expects no royalties or any other compensation. Developer is responsible for supporting applications created using the BMP5 Direct Binaries.

## ***RESPONSIBILITIES OF DEVELOPER***

The Developer agrees:

- To provide a competent programmer familiar with Campbell Scientific, Inc. datalogger programming to write the applications.
- Not to sell or distribute documentation on use of the BMP5 Direct Binaries.
- Not to sell or distribute the applications that are provided as examples in the BMP5 Direct SDK.
- Developers may copy and paste portions of the code into their own applications, but their applications are expected to be unique creations.
- This Agreement does not give Developer the right to sell or distribute any other Campbell Scientific, Inc. Software (e.g., PC200W, VisualWeather, LoggerNet or any of their components, files, documentation, etc.) as part of Developer's application. Distribution of any other Campbell Scientific, Inc. software requires a separate distribution agreement.
- Not to develop applications that compete directly with any application developed by Campbell Scientific, Inc. or its affiliates.
- To assure that each application developed with BMP5 Direct Binaries clearly states the name of the person or entity that developed the application. This information should appear on the first window the user will see.

### ***WARRANTY***

There is no written or implied warranty provided with the BMP5 Direct SDK software other than as stated herein. Developer agrees to bear all warranty responsibility of any derivative products distributed by Developer.

### ***TERMINATION***

Any license violation or breach of Agreement will result in immediate termination of the developer's rights herein and the return of all BMP5 Direct SDK materials to Campbell Scientific, Inc.

### ***MISCELLANEOUS***

Notices required hereunder shall be in writing and shall be given by certified or registered mail, return receipt requested. Such notice shall be deemed given in the case of certified or registered mail on the date of receipt. This Agreement shall be governed and construed in accordance with the laws of the State of Utah, USA. Any dispute resulting from this Agreement will be settled in arbitration.

This Agreement sets forth the entire understanding of the parties and supersedes all prior agreements, arrangements and communications, whether oral or written pertaining to the subject matter hereof. This Agreement shall not be modified or amended except by the mutual written agreement of the parties. The failure of either party to enforce any of the provisions of this Agreement shall not be construed as a waiver of such provisions or of the right of such party thereafter to enforce each and every provision contained herein. If any term, clause, or provision contained in this Agreement is declared or held invalid by a court of competent jurisdiction, such declaration or holding shall not affect the validity of any other term, clause, or provision herein contained. Neither the rights nor the obligations arising under this Agreement are assignable or transferable.

If within 30 days of receiving the BMP5 Direct SDK product developer does not agree to the terms of license, developer shall return all materials without retaining any copies of the product and shall remove any use of the BMP5 Direct Binaries in any applications developed or distributed by Developer. In the absence of such return, CSI shall consider Developer in agreement with the herein, stated license terms and conditions.

# ***Limited Warranty***

---

CAMPBELL SCIENTIFIC, INC. warrants that the installation media on which the accompanying computer software is recorded and the documentation provided with it are free from physical defects in materials and workmanship under normal use. CAMPBELL SCIENTIFIC, INC. warrants that the computer software itself will perform substantially in accordance with the specifications set forth in the instruction manual published by CAMPBELL SCIENTIFIC, INC.

CAMPBELL SCIENTIFIC, INC. will either replace or correct any software that does not perform substantially according to the specifications set forth in the instruction manual with a corrected copy of the software or corrective code. In the case of significant error in the installation media or documentation,

CAMPBELL SCIENTIFIC, INC. will correct errors without charge by providing new media, addenda or substitute pages. If CAMPBELL SCIENTIFIC, INC. is unable to replace defective media or documentation, or if CAMPBELL SCIENTIFIC, INC. is unable to provide corrected software or corrected documentation within a reasonable time, CAMPBELL SCIENTIFIC, INC. will either replace the software with a functionally similar program or refund the purchase price paid for the software.

The above warranties are made for ninety (90) days from the date of original shipment.

CAMPBELL SCIENTIFIC, INC. does not warrant that the software will meet licensee's requirements or that the software or documentation are error free or that the operation of the software will be uninterrupted. The warranty does not cover any media or documentation that has been damaged or abused. The software warranty does not cover any software that has been altered or changed in any way by anyone other than CAMPBELL SCIENTIFIC, INC. CAMPBELL SCIENTIFIC, INC. is not responsible for problems caused by computer hardware, computer operating systems or the use of CAMPBELL SCIENTIFIC, INC.'s software with non-CAMPBELL SCIENTIFIC, INC. software.

ALL WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED AND EXCLUDED. CAMPBELL SCIENTIFIC, INC. SHALL NOT IN ANY CASE BE LIABLE FOR SPECIAL, INCIDENTAL, CONSEQUENTIAL, INDIRECT, OR OTHER SIMILAR DAMAGES EVEN IF CAMPBELL SCIENTIFIC HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

CAMPBELL SCIENTIFIC, INC. IS NOT RESPONSIBLE FOR ANY COSTS INCURRED AS A RESULT OF LOST PROFITS OR REVENUE, LOSS OF USE OF THE SOFTWARE, LOSS OF DATA, COST OF RE-CREATING LOST DATA, THE COST OF ANY SUBSTITUTE PROGRAM, CLAIMS BY ANY PARTY OTHER THAN LICENSEE, OR FOR OTHER SIMILAR COSTS.

LICENSEE'S SOLE AND EXCLUSIVE REMEDY IS SET FORTH IN THIS LIMITED WARRANTY. CAMPBELL SCIENTIFIC, INC.'S AGGREGATE LIABILITY ARISING FROM OR RELATING TO THIS AGREEMENT OR THE SOFTWARE OR DOCUMENTATION (REGARDLESS OF THE FORM OF ACTION; E.G., CONTRACT, TORT, COMPUTER MALPRACTICE, FRAUD AND/OR OTHERWISE) IS LIMITED TO THE PURCHASE PRICE PAID BY THE LICENSEE.

# ***BMP5 Direct SDK Table of Contents***

---

*PDF viewers note: These page numbers refer to the printed version of this document. Use the Adobe Acrobat® bookmarks tab for links to specific sections.*

<b>1. BMP5 Direct SDK Overview .....</b>	<b>1-1</b>
1.1 General Notes on BMP5 Direct SDK Usage .....	1-1
1.2 Datalogger Program Table Structure .....	1-1
<b>2. SimplePB.DLL Reference.....</b>	<b>2-1</b>
2.1 OpenPort() .....	2-1
2.2 ClosePort().....	2-1
2.3 OpenIPPort().....	2-2
2.4 CloseIPPort() .....	2-2
2.5 GetClock() .....	2-2
2.6 SetClock().....	2-3
2.7 GetValue() .....	2-4
2.8 SetValue().....	2-5
2.9 GetData() .....	2-6
2.10 GetDataHeader().....	2-7
2.11 GetCommaData().....	2-8
2.12 File_Send() .....	2-9
2.13 GetAddress().....	2-10
2.14 GetStatus() .....	2-11
2.15 GetTableNames().....	2-12
2.16 GetDLLVersion().....	2-13
2.17 GetLastResults() .....	2-13

## ***Appendix***

<b>A. Sample Program Table Structure .....</b>	<b>A-1</b>
--	------------



# ***Section 1. BMP5 Direct SDK Overview***

---

The BMP5 Direct Software Development Kit (SDK) is a programming interface that facilitates simple and direct communication with a single datalogger containing a PakBus operating system. This SDK uses a simple call-level API (SimplePB.DLL) that does not need to be registered on the PC. However, the SimplePB.DLL wrapper accesses and therefore requires the included communications engine, CORALIB3D.DLL, to be installed in the same folder.

The BMP5 Direct SDK allows the creation of basic BMP5 application packets that are sent to PakBus dataloggers over the PakBus network. Examples and controls are installed by default in C:\Campbellsci\BMP5DirectSDK.

## **1.1 General Notes on BMP5 Direct SDK Usage**

In order to communicate with the datalogger, a direct connection must be created using the SimplePB.DLL. The BMP5 Direct SDK allows a connection to a single datalogger through either an IP port or a COM port using the OpenIPPort() or OpenPort() command.

Opening the COM or IP port starts the CORALIB3D.DLL. Therefore, the COM port or IP port opened by the application must be closed before exiting the application or the CORALIB3D.DLL will crash. Use the functions CloseIPPort() or ClosePort() before the application exits to stop the CORALIB3D.DLL and to properly close the connection. Starting the CORALIB3D.DLL creates a working directory including log files by default in C:\Campbellsci\SimplePB.

Once a connection is established, additional commands can be used to access and administer the datalogger. A datalogger program can be sent, the clock can be checked or set, values can be set, and data can be collected. Use a combination of the commands found in the SimplePB.DLL to accomplish the desired task.

## **1.2 Datalogger Program Table Structure**

One default table in a PakBus datalogger is the Status table. The datalogger may also include either a Public table or an Inlocs table. The Status table contains values describing the datalogger and datalogger program and the Public or Inlocs table stores all public variables or input locations. The user defines any additional tables in the program sent to the datalogger.

When using the SimplePB.DLL to access specific tables and fields in a datalogger program, the table number and the field number are implied by their respective positions. For example, table number one is the first table in the datalogger program and field number one is the first field following the mandatory record number and timestamp fields within the table. The user must understand the table structure of the program running in the datalogger because table and field names and numbers are used to identify and access specific tables and fields in several of the SimplePB.DLL commands.



## ***Section 2. SimplePB.DLL Reference***

---

The SimplePB.DLL is a call-level API that does not need to be registered on the PC. However, the SimplePB.DLL wrapper accesses and therefore requires the included communications engine, CORALIB3D.DLL, to be installed in the same folder. The SimplePB wrapper provides an easy interface through the CORALIB3D.DLL communications engine to access a single datalogger. The following commands are available in the SimplePB.DLL:

### **2.1 OpenPort()**

Opens a COM port to allow a direct connection to the datalogger.

#### **Syntax**

OpenPort(com\_port\_no, baud)

#### **Parameters**

com\_port\_no: Integer – The COM port to open.

baud: Integer – The baud rate used by the COM port.

#### **Return Codes**

0: Integer – Successful.

-1: Integer – COM port failed to open or is already open.

### **2.2 ClosePort()**

Closes the previously opened COM port.

#### **Syntax**

ClosePort()

#### **Return Codes**

0 = Successful.

-1 = COM port failed to close or was not open.

## 2.3 OpenIPPort()

Opens an IP port to allow a connection to a datalogger.

### Syntax

OpenIPPort(ip\_address, tcp\_port)

### Parameters

ip\_address: String – A valid IP address for the datalogger.

tcp\_port: Integer – The port that will be used when communicating with the datalogger.

### Return Codes

0 = Successful.

-1 = IP port failed to open or is already open.

## 2.4 CloseIPPort()

Closes the previously opened IP port.

### Syntax

CloseIPPort()

### Return Codes

0 = Successful.

-1 = IP port failed to close or was not open.

## 2.5 GetClock()

Query the datalogger for its current date and time.

### Syntax

GetClock(pakbus\_address, device\_type, return\_data, return\_data\_len)

### Parameters

pakbus\_address: Integer – The PakBus address of the datalogger.

device\_type: Integer – The type of datalogger:

- 1=CR200
- 2=CR10XPB, CR23XPB, CR510PB
- 3=CR1000
- 4=CR3000
- 5=CR800, CR850

return\_data: Char – The location in memory where the data returned from the datalogger exists.

return\_data\_len: Integer – Number of bytes in the data returned from the datalogger.

## Return Codes

0 = Successful.  
 -1 = Communication timed out.  
 -2 = Port is not open.

## Example of data returned by function call

14:12:35 04/16/2004

## 2.6 SetClock()

Sets the date and time of the datalogger to match the PC clock.

### Syntax

SetClock(pakbus\_address, device\_type, return\_data, return\_data\_len)

### Parameters

pakbus\_address: Integer – The PakBus address of the datalogger.

device\_type: Integer – The type of datalogger:

- 1=CR200
- 2=CR10XPB, CR23XPB, CR510PB
- 3=CR1000
- 4=CR3000
- 5=CR800, CR850

return\_data: Char – The location in memory where the data returned from the datalogger exists.

return\_data\_len: Integer – Number of bytes in the data returned from the datalogger.

## Return Codes

0 = Successful.  
-1 = Communication timed out.  
-2 = Port is not open.

## Example of data returned by function call

14:22:51 04/16/2004 (Old Time Old Date)  
14:22:27 04/16/2004 (New Time New Date)

## 2.7 GetValue()

Query the datalogger for a value or group of values.

### Syntax

GetValue(pakbus\_address, device\_type, swath, table\_name, field\_name, return\_data, return\_data\_len)

### Parameters

pakbus\_address: Integer – The PakBus address of the datalogger.

device\_type: Integer – The type of datalogger:

- 1=CR200
- 2=CR10XPB, CR23XPB, CR510PB
- 3=CR1000
- 4=CR3000
- 5=CR800, CR850

swath: Integer – The range within an indexed variable array to collect starting at the location described in the field\_name parameter. The requested swath must be within the bounds of the indexed array or an error will occur.

table\_name: String – The name of the table in the datalogger to collect.

field\_name: String – The name of the field in the table where collection of values should start.

return\_data: Char – The location in memory where the data returned from the datalogger exists.

return\_data\_len: Integer – Number of bytes in the data returned from the datalogger.

## Return Codes

0 = Successful.  
-1 = Communication timed out.  
-2 = Port is not open.

## Example of data returned by function call

12.753 111.9 1.239 (Swath of 3 values from fields)

## 2.8 SetValue()

Set a value in the datalogger.

### Syntax

SetValue(pakbus\_address, device\_type, table\_name, field\_name, value)

### Parameters

pakbus\_address: Integer – The PakBus address of the datalogger.

device\_type: Integer – The type of datalogger:

- 1=CR200
- 2=CR10XPB, CR23XPB, CR510PB
- 3=CR1000
- 4=CR3000
- 5=CR800, CR850

table\_name: String – The name of the table in which the field will be set.

field\_name: String – The field that will be set with the new value.

value: String – The value used to set the field.

## Return Codes

0 = Successful.  
-1 = Communication timed out.  
-2 = Port is not open.

## 2.9 GetData()

Query the datalogger for column names and data from one of its tables.

### Syntax

GetData(pakbus\_address, device\_type, table\_no, record\_no, return\_data, return\_data\_len)

### Parameters

pakbus\_address: Integer – The PakBus address of the datalogger.

device\_type: Integer – The type of datalogger:

- 1=CR200
- 2=CR10XPB, CR23XPB, CR510PB
- 3=CR1000
- 4=CR3000
- 5=CR800, CR850

table\_no: Integer – The number of the table from which to collect data.

record\_no: Integer – The record number where data collection will start. All records following this record number will be included in the collection. Therefore, if the record number is set to 0, all records in the table will be collected. In addition, if the record number specified does not exist in the datalogger, all existing records from the oldest to the newest will be returned. However, if the record number is set to a negative number, only the most recent record in the table will be collected. There is not a way to specify and collect a single record from a table using this command unless that record is the most recent record in the table.

return\_data: Char – The location in memory where the data returned from the datalogger exists.

return\_data\_len: Integer – Number of bytes in the data returned from the datalogger.

### Return Codes

- 0 = Successful.
- 1 = Success but more data to collect.
- 1 = Communication timed out.
- 2 = Port is not open.

## Example of data returned by function call

```
"2004-04-16 14:18:03",1 (Time stamp, Record number)
1,OSversion,v03A (Field number, Field name, Field value)
2,OSDate,06-Jan-04
3,ProgName,BATT.CR2
4,ProgSig,54451
5,CalOffset,2.625
6,PakBusAddress,1
7,RfInstalled,424
8,RfNetAddr,0
9,RfAddress,0
10,RfHopSeq,0
11,RfPwrMode,RF1_Sec
12,Rf_ForceOn,0
13,RfSignalLevel,0
14,RfRxPakBusCnt,0
15,VarOutOfBounds,0
16,SkipScan,0
17,TrapCode,0
18,WatchDogCnt,0
19,ResetTables,0
20,BattVoltage,12.3943
```

## 2.10 GetDataHeader()

Query the datalogger for only the header information from one of its tables.

### Syntax

```
GetData(pakbus_address, device_type, table_no, return_data, return_data_len)
```

### Parameters

**pakbus\_address:** Integer – The PakBus address of the datalogger.

**device\_type:** Integer – The type of datalogger:

- 1=CR200
- 2=CR10XPB, CR23XPB, CR510PB
- 3=CR1000
- 4=CR3000
- 5=CR800, CR850

**table\_no:** Integer – The number of the table from which to collect data.

**return\_data:** Char – The location in memory where the data returned from the datalogger exists.

return\_data\_len: Integer – Number of bytes in the data returned from the datalogger.

## Return Codes

0 = Successful.  
1 = Success but more data to collect.  
-1 = Communication timed out.  
-2 = Port is not open.

## Example of data returned by function call

"TIMESTAMP", "RECORD", OSVersion, OSDate, OSSignature

## 2.11 GetCommaData()

Query the datalogger for only the values of a record and display those values in a comma-separated format.

### Syntax

GetData(pakbus\_address, device\_type, table\_no, record\_no, return\_data, return\_data\_len)

### Parameters

pakbus\_address: Integer – The PakBus address of the datalogger.

device\_type: Integer – The type of datalogger:

- 1=CR200
- 2=CR10XPB, CR23XPB, CR510PB
- 3=CR1000
- 4=CR3000
- 5=CR800, CR850

table\_no: Integer – The number for the table from which to collect data.

record\_no: Integer – The record number where data collection will start. All records following this record number will be included in the collection. Therefore, if the record number is set to 0, all records in the table will be collected. In addition, if the record number specified does not exist in the datalogger, all existing records from the oldest to the newest will be returned. However, if the record number is set to a negative number, only the most recent record in the table will be collected. There is not a way to specify and collect a single record from a table using this command unless that record is the most recent record in the table.

return\_data: Char – The location in memory where the data returned from the datalogger exists.

return\_data\_len: Integer – Number of bytes in the data returned from the datalogger.

## Return Codes

0 = Successful.  
 1 = Success but more data to collect.  
 -1 = Communications timed out.  
 -2 = Port is not open.

## Example of data returned by function call

"2005-09-08 14:13:47",1,"CR1000.Std.05","050624",47178

## 2.12 File\_Send()

Send a program to the datalogger.

### Syntax

File\_Send(pakbus\_address, device\_type, file\_name, return\_data, return\_data\_len)

### Parameters

pakbus\_address: Integer – The Pakbus address of the datalogger.

device\_type: Integer – The type of datalogger:

- 1=CR200
- 2=CR10XPB, CR23XPB, CR510PB
- 3=CR1000
- 4=CR3000
- 5=CR800, CR850

file\_name: String – The path and file name of the program to be sent.

return\_data: Char – The location in memory where the data returned from the datalogger exists.

return\_data\_len: Integer – Number of bytes in the data returned from the datalogger.

## Return Codes

0 = Successful.  
1 = Success but more data to transfer.  
-1 = Communication timed out.  
-2 = Port is not open.  
-3 = Cannot open source file.  
-4 = File name is too long.  
-5 = Datalogger timed out.  
-6 = File offset does not match.  
-7 = Datalogger reported an error.  
-8 = File control error.  
-9 = Cannot get program status.

## Example of data returned from a CR1000

OS Version : CR1000.Std.05  
OS Signature : 19128  
Serial Number : 1031  
PowerUp Progr : CPU:Program.cr1  
Compile Status: Datalogger Program Running  
Program Name : CPU:Program.cr1  
Program Sig. : 32083  
Compile Result: Compiled in SequentialMode.

## 2.13 GetAddress()

Query a datalogger for its PakBus address.

### Syntax

GetAddress(device\_type, return\_data, return\_data\_len)

### Parameters

Device: Integer –Type of datalogger:

- 1=CR200
- 2=CR10XPB, CR23XPB, CR510PB
- 3=CR1000
- 4=CR3000
- 5=CR800, CR850

return\_data: Char – The location in memory where the data returned from the datalogger exists.

return\_data\_len: Integer – Number of bytes in the data returned from the datalogger.

## Return Codes

0 = Successful.  
 -1 = Communication timed out.  
 -2 = Port is not open.

## Example of data returned by function call

PakBusAddress=1;

## 2.14 GetStatus()

Query the datalogger for its current status.

### Syntax

GetStatus(pakbus\_address, device\_type, return\_data, return\_data\_len)

### Parameters

pakbus\_address: Integer – The PakBus address of the datalogger.

device\_type: Integer – The type of datalogger:

- 1=CR200
- 2=CR10XPB, CR23XPB, CR510PB
- 3=CR1000
- 4=CR3000
- 5=CR800, CR850

return\_data: Char – The location in memory where the data returned from the datalogger exists.

return\_data\_len: Integer – Number of bytes in the data returned from the datalogger.

## Return Codes

0 = Successful  
 -1 = Communication timed out  
 -2 = Port is not open

## Example of data returned from a CR200

OS Version: v03A  
 OS Signature: 43529  
 Serial Number:  
 PowerUp Progr:

Compile Status: Datalogger Program Running  
Program Name: BATT.CR2  
Program Sig.: 54451  
Compile Result: Program Running  
Batt=12.38V

## 2.15 GetTableNames()

Query the datalogger for its table names and numbers.

### Syntax

GetTableNames (pakbus\_address, device\_type, return\_data, return\_data\_len)

### Parameters

pakbus\_address: Integer – The PakBus address of the datalogger.

device\_type: Integer – The type of datalogger:

- 1=CR200
- 2=CR10XPB, CR23XPB, CR510PB
- 3=CR1000
- 4=CR3000
- 5=CR800, CR850

return\_data: Char – The location in memory where the data returned from the datalogger exists.

return\_data\_len: Integer – Number of bytes in the data returned from the datalogger.

### Return Codes

- 0 = Successful.
- 1 = Communication timed out.
- 2 = Cannot read table definitions from the datalogger.

### Example of data returned by function call

1 Status  
2 DataTable1  
3 DataTable2  
4 Public

## 2.16 GetDLLVersion()

Get the version of the SimplePB.dll being used.

### Syntax

```
GetDLLVersion(return_data, return_data_len)
```

### Parameters

return\_data: Char – The location in memory where the data returned from the datalogger exists.

return\_data\_len: Integer – Length of the data returned from the DLL.

### Return Codes

0 = Successful

### Example of data returned by function call

SimplePB.DLL Version 2.0 / 2,2,3,0

## 2.17 GetLastResults()

Retrieves the return\_data results from memory for the previous function as a String. This function is useful for developers that don't want to manage memory pointers.

### Syntax

```
GetLastResults()
```

### Return Codes

0 = Successful



# Appendix A. Sample Program Table Structure

---

The table structure of a PakBus datalogger is given in the example below. This example shows a datalogger with two user defined tables plus the Status table and Public or Inlocs table. The second table in the example below contains three records and the third table contains four records. Both the Status table and Public or Inlocs table will always return the most recent records and will not contain any historical data records.

The first table is the Status table, which shows the status of the datalogger. The Public or Inlocs table contains all public variables or input locations. All other tables found in the datalogger are created and defined by the user in the datalogger program. The tables in a PakBus datalogger will always contain a record number and timestamp followed by the data fields.

**Table 1 – Status**

Record No	Time Stamp	Data Field 1	Data Field 2	Data Field 3-19	Data Field 20
-----------	------------	--------------	--------------	-----------------	---------------

**Table 2 – User Defined**

RN 0	Time Stamp	Data Field 1	Data Field 2
RN 1	Time Stamp	Data Field 1	Data Field 2
RN 2	Time Stamp	Data Field 1	Data Field 2

**Table 3 – User Defined**

RN 0	Time Stamp	Data Field 1	Data Field 2	Data Field 3	Data Field 4	Data Field 5
RN 1	Time Stamp	Data Field 1	Data Field 2	Data Field 3	Data Field 4	Data Field 5
RN 2	Time Stamp	Data Field 1	Data Field 2	Data Field 3	Data Field 4	Data Field 5
RN 3	Time Stamp	Data Field 1	Data Field 2	Data Field 3	Data Field 4	Data Field 5

**Table 4 – Public or Inlocs**

Record No	Time Stamp	Data 1	Data 2	Data 3	Data 4	Data 5	Data 6
-----------	------------	--------	--------	--------	--------	--------	--------

## CR200 Datalogger Program Tables

The following tables show the table structure from a program installed in a CR200 datalogger. This program measures and stores the minimum battery voltage and the minimum and maximum temperature over a 60-minute interval. When communicating with a datalogger using the BMP5 Direct SDK, knowing the table structure of the running program is necessary for some commands.

**Table Number 1 – Status**

Field Number	Field Name	Units	Notes:
Field 1	OSVersion		Operating System Version
Field 2	OSDate		Date of Operating System
Field 3	ProgName		The name of the program running in the datalogger
Field 4	ProgSig		The signature of the running program
Field 5	CalOffset		
Field 6	PakBusAddress		The PakBus address of the datalogger (1-4094)
Field 7	RfInstalled		Radio Detected
Field 8	RfNetAddr		Valid Addresses are 0-63
Field 9	RfAddress		Valid Addresses are 0-1023
Field 10	RfHopSeq		Valid numbers are 0-6
Field 11	RfPwrMode		RF1_Sec
Field 12	Rf_ForceOn		
Field 13	RfSignalLevel		RF signal strength should be above 40
Field 14	RfRxPakBusCnt		
Field 15	VarOutOfBounds		
Field 16	SkipScan		Program didn't complete before the next execution interval
Field 17	TrapCode		
Field 18	WatchDogCnt		Number of Watchdog Errors
Field 19	ResefTables		Clears All Stored Data
Field 20	BattVoltage	Volts	Current Battery Voltage

**Table Number 2 – Hourly:** The Hourly table contains the minimum battery voltage and the minimum and maximum temperature over a 60-minute interval.

Field Number	Field Name	Units	Notes:
Field 1	Battery_Min	Volts	
Field 2	Battery_Time	Time	
Field 2	Temp_Min	Deg C	
Field 3	Temp_Max	Deg C	

**Table Number 3 – Public:** The Public table contains only the most recent “real-time” record for the variable described in the datalogger program.

Field Number	Field Name	Units	Notes:
Field 1	Batt_Volt	Volts	
Field 2	Temp	Deg C	

## CR200 Datalogger Program

```
'CR200 Series
'Declare Variables and Units
Public Batt_Volt, Temp

Units Batt_Volt=Volts
Units Temp=Deg C

'Define Data Tables
DataTable(Hourly,True,-1)
  DataInterval(0,60,Min)
  Minimum(1,Batt_Volt,False,True)
FieldNames("Battery_MIN,Battery_Time")
  Maximum(1,Temp,False,False)
  Minimum(1,Temp,False,False)
EndTable
'Main Program
BeginProg
  Scan(10,Sec)
    'Default Datalogger Battery Voltage measurement Batt_Volt:
    Battery(Batt_Volt)
    '109 Temperature Probe measurement Temp:
    Therm109(Temp,1,1,1,1.0,0.0)
    'Call Data Tables and Store Data
    CallTable(Hourly)
  NextScan
EndProg
```

## WeatherHawk Weather Station Tables

The following tables show the table structure from a default WeatherHawk weather station program installed in a CR200 datalogger. When communicating with a datalogger using the BMP5 Direct SDK, knowing the table structure of the running program is necessary for some commands

**Table Number 1 – Status**

Field Number	Field Name	Units	Notes:
Field 1	OSVersion		Operating System Version
Field 2	OSDate		Date of Operating System
Field 3	ProgName		The name of the program running in the datalogger
Field 4	ProgSig		The signature of the running program
Field 5	CalOffset		
Field 6	PakBusAddress		The PakBus address of the datalogger (1-4094)
Field 7	RfInstalled		Radio Detected
Field 8	RfNetAddr		Valid Addresses are 0-63
Field 9	RfAddress		Valid Addresses are 0-1023
Field 10	RfHopSeq		Valid numbers are 0-6
Field 11	RfPwrMode		RF1_Sec
Field 12	Rf_ForceOn		
Field 13	RfSignalLevel		RF signal strength should be above 40
Field 14	RfRxPakBusCnt		
Field 15	VarOutOfBounds		
Field 16	SkipScan		Program didn't complete before the next execution interval
Field 17	TrapCode		
Field 18	WatchDogCnt		Number of Watchdog Errors
Field 19	ResetTables		Clears All Stored Data
Field 20	BattVoltage	Volts	Current Battery Voltage

**Table Number 2 – SiteVal:** The SiteVal table contains values that are stored for calculations by the WeatherHawk program. Data is only stored when field “SaveSite” in the Public table is set to one.

Field Number	Field Name	Units	Notes:
Field 1	Altitude_m	Meter	
Field 2	Latitude	Degrees	
Field 3	Longitude	Degrees	
Field 4	BPOffset_KPa	KPa	
Field 5	Int_Timer	Minutes	

**Table Number 3 – Data1:** This table contains data output at the Int\_timer rate from the Public table. For example, if Int\_timer = 15 min, this table contains 15 min data.

Field Number	Field Name	Units	Notes:
Field 1	BatVolt_V	Volts	
Field 2	BatVolt_V_Min	Volts	
Field 3	AirTemp_C_Avg	Celsius	
Field 4	RH_Avg	Percent	
Field 5	WindSpeed_ms_Avg	m/s	
Field 6	Solar_Avg	W/m <sup>2</sup>	
Field 7	ETo	mm	
Field 8	AirTemp_C_Min	Celsius	
Field 9	AirTemp_C_TMn	Time	Example: 2004-01-25 13:49:50
Field 10	Max_AirTemp	Celsius	
Field 11	AirTemp_C_C_Tmx	Time	Example: 2004-01-25 13:49:50
Field 12	WindSpeed_ms_WVc(1)	m/s	Average Wind Speed
Field 13	WindSpeed_ms_WVc(2)	Degrees	Unit Vector Wind Direction
Field 14	WindSpeed_ms_Max	m/s	
Field 15	Baromete_KPa	KPa	
Field 16	RainYearly_mm	mm	

**Table Number 4 – Data2:** This table contain daily data values.

Field Number	Field Name	Units	Notes:
Field 1	BatVolt_V_Min	Volts	
Field 2	AirTemp_C_Max	Celsius	
Field 3	AirTemp_C_Min	Celsius	
Field 4	WindSpeed_ms_Max	m/s	
Field 5	RainYearly_mm	mm	
Field 6	DailyETo_mm	mm	

**Table Number 5 – Public:** The public table contains only the most recent “real-time” record.

Field Number	Field Name	Units	Notes:
Field 1	SaveSite		Set to one to save values to SiteVal table
Field 2	Latitude	Degrees	Decimal format: 41 deg 45 min = 14.75
Field 3	Longitude	Degrees	Decimal format: 41 deg 45 min = 14.75
Field 4	Altitude_m	Meter	
Field 5	Bpoffset_KPa	KPa	
Field 6	Int_timer	Minutes	
Field 7	RainReset		Set to 1 to reset RainYearly_mm variable
Field 8	BatVolt_V	Volts	
Field 9	Solar	W/m <sup>2</sup>	
Field 10	AirTemp_C	Celsius	
Field 11	RH	Percent	
Field 12	Barometer_KPa	KPa	Sea level adjustment barometric pressure
Field 13	WindSpeed_ms	m/s	
Field 14	WindDirect_deg	Degrees	
Field 15	RainYearly_mm	mm	Running Sum of Rainfall
Field 16	DailyETo_mm	mm	Running Sum of Eto (Resets at midnight)



## **Campbell Scientific Companies**

---

### **Campbell Scientific, Inc. (CSI)**

815 West 1800 North  
Logan, Utah 84321  
UNITED STATES  
[www.campbellsci.com](http://www.campbellsci.com)  
[info@campbellsci.com](mailto:info@campbellsci.com)

### **Campbell Scientific Africa Pty. Ltd. (CSAf)**

PO Box 2450  
Somerset West 7129  
SOUTH AFRICA  
[www.csafrica.co.za](http://www.csafrica.co.za)  
[cleroux@csafrica.co.za](mailto:cleroux@csafrica.co.za)

### **Campbell Scientific Australia Pty. Ltd. (CSA)**

PO Box 444  
Thuringowa Central  
QLD 4812 AUSTRALIA  
[www.campbellsci.com.au](http://www.campbellsci.com.au)  
[info@campbellsci.com.au](mailto:info@campbellsci.com.au)

### **Campbell Scientific do Brazil Ltda. (CSB)**

Rua Luisa Crapsi Orsi, 15 Butantã  
CEP: 005543-000 São Paulo SP BRAZIL  
[www.campbellsci.com.br](http://www.campbellsci.com.br)  
[suporte@campbellsci.com.br](mailto:suporte@campbellsci.com.br)

### **Campbell Scientific Canada Corp. (CSC)**

11564 - 149th Street NW  
Edmonton, Alberta T5M 1W7  
CANADA  
[www.campbellsci.ca](http://www.campbellsci.ca)  
[dataloggers@campbellsci.ca](mailto:dataloggers@campbellsci.ca)

### **Campbell Scientific Ltd. (CSL)**

Campbell Park  
80 Hathern Road  
Shepshed, Loughborough LE12 9GX  
UNITED KINGDOM  
[www.campbellsci.co.uk](http://www.campbellsci.co.uk)  
[sales@campbellsci.co.uk](mailto:sales@campbellsci.co.uk)

### **Campbell Scientific Ltd. (France)**

Miniparc du Verger - Bat. H  
1, rue de Terre Neuve - Les Ulis  
91967 COURTABOEUF CEDEX  
FRANCE  
[www.campbellsci.fr](http://www.campbellsci.fr)  
[campbell.scientific@wanadoo.fr](mailto:campbell.scientific@wanadoo.fr)

### **Campbell Scientific Spain, S. L.**

Psg. Font 14, local 8  
08013 Barcelona  
SPAIN  
[www.campbellsci.es](http://www.campbellsci.es)  
[info@campbellsci.es](mailto:info@campbellsci.es)