



WIBU-SYSTEMS AG

WIBU-KEY User's Guide

WIBU-BOX ExtMem

Marcellus Buchheit

Version 4.10 of 13. May 2004
Copyright © 2004 by WIBU-SYSTEMS AG

© Copyright 1989-2004 by WIBU-SYSTEMS AG,
Rueppurrer Strasse 52-54, D-76137 Karlsruhe, Federal Republic of Germany

Printed in Germany

All rights reserved. No part of this documentation, accompanying software or other components of the described product may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying and recording, for any purpose other than the personal use of the purchaser without the express written permission of WIBU-SYSTEMS.

This documentation, the hardware (WIBU-BOX, CM-Stick, power supply etc.) and accompanying software were produced with great care, yet errors are possible. WIBU-SYSTEMS assumes no responsibility for errors within the documentation, the hardware or the software.

WIBU-SYSTEMS reserves the right to change programs or the documentation from time to time without informing the user; errors and omissions excepted.

Trademarks

WIBU® is a registered trademark of WIBU-SYSTEMS AG.

CodeMeter™ and SmartShelter™ are trademarks of WIBU-SYSTEMS AG.

Microsoft®, Windows®, Win32®, MS-DOS® and Visual Basic® are registered trademarks of Microsoft Corporation.

IBM®, IBM-PC®, and OS/2® are registered trademarks of International Business Machines Corporation.

Intel® is a registered trademark of Intel-Corporation.

Contents

1	Introduction	4
2	ExtMem Basics	4
2.1	Addressing ExtMem Pages via the classic WIBU-KEY API.....	4
2.2	Addressing ExtMem Pages via the WIBU-KEY COM API.....	5
2.3	Accessing Protected Type ExtMem.....	5
2.4	Raw Mode and Formatted Mode ExtMem Organization.....	7
2.5	Addressing Raw Data in Formatted Mode.....	9
2.6	Addressing Licensor Data in Formatted Mode.....	9
3	Managing ExtMem with WkCrypt	10
3.1	Formatting and Un-Formatting ExtMem.....	10
3.2	Writing ExtMem.....	11
3.3	Reading and Listing ExtMem.....	12
3.4	Calculating ExtMem Access Signatures.....	13
4	ExtMem Limitations of WIBU-KEY 4.10	14
5	Appendix	14
5.1	State of Current Version.....	14
5.2	Contact Address.....	14

Pages in Document: 14

List of Figures

Error! No table of figures entries found.

List of Tables

Error! No table of figures entries found.

1 Introduction

This chapter introduces into the Extended Memory functionality of WIBU-KEY. Extended Memory, abbreviated as **ExtMem**, is supported by all WIBU-BOXes with Version 6 or higher. In the moment (mid of 2004) this is the WIBU-BOX/P+, WIBU-BOX/RP+, WIBU-BOX/U+ and WIBU-BOX/RU+.

2 ExtMem Basics

Like the 1 res. 10 Firm Code /User Code entries of the classic WIBU-BOX, ExtMem is non-volatile memory which can be programmed by software and keep its information permanently in the WIBU-BOX.

But in contrast to the classic entries, ExtMem has some significant differences:

- ExtMem is much larger than the traditional WIBU-BOX storage. All WIBU-BOX version 6 res. 7 of 2004 have for example 16 Kbytes of available ExtMem.
- ExtMem is primarily designed for storing data, not to storing keys. ExtMem cannot be used to initialize any encryption or other security operation in the WIBU-BOX.
- ExtMem is organized in pages of 32 bytes each.
- The half of the ExtMem is available as unprotected memory, named **User Type**. These pages can be read and written without restrictions; no password or access key is required. The 8 KB of this type are organized as 256 32-byte pages, addressed with an index from 0 to 255.
- The other of the ExtMem is organized as protected memory, named **Protected Type**. Again there are 8 KB of this type, organized as 256 32- byte pages, addressed with an index from 0 to 255. The access control for reading and writing is controlled by the contents of the first WIBU-BOX entry: To read or write such a page, an 8-byte **Access Signature** is required, which depends on the stored Firm Code, User Code and Added Data Entry. Details are described in chapter 2.3.

2.1 Addressing ExtMem Pages via the classic WIBU-KEY API

A single ExtMem page is completely described by the **WKBEXTMEM** structure:

```
typedef struct {
    USHORT fsCtrl;          /* contains WKB XM xxx */
    USHORT iPage;          /* contains index of the ExtMem page */
    BYTE  abSignature[8]; /* contains access signature for WKB_XM_PROT */
    BYTE  abData[32];     /* receives or contains data of ExtMem page */
} WKBEXTMEM; /* prefix "wkbxm" */
```

The first parameter may be **WKB_XM_USER** (0) to address User Type ExtMem, or **WKB_XM_PROT** (1) to address Protected Type ExtMem. The bits 1 to 15 of this field are reserved for future use and must be set to 0. The page index is specified in `iPage`; the first page has index 0, the next 1 etc.

The `abSignature` field contains the Access Signature. This field is not used and must be filled with eight 0-bytes if User Type ExtMem is addressed. Protected Type ExtMem requires a valid Access Signature; it can be calculated easily by **WKCRYPT** (see chapter 3.4).

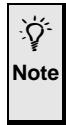
Last but not least `abData` contains the page data to be written or is filled with the read page data.

The reading of ExtMem pages is supported by the **WkbListBox2** function, using the **WKB_LIST_EXTMEM** constant in the `flCtrl` parameter and specifying the address of one **WKBEXTMEM** structure in the `pvDest` parameter. The `cbDest` parameter must be set to `sizeof(WKBEXTMEM)`.

The writing of ExtMem pages is supported by the **WkbProgram2** function, using the **WKB_PGM_EXTMEM** constant in the `flCtrl` parameter and specifying the address of one **WKBEXTMEM** structure in the `pvPgmData` parameter.

The number of available ExtMem pages in a specific WIBU-BOX can be determined by the `WkbListBox2` function using the `WKB_LIST_CONFIG` parameter. The returning `WKBBOXCONFIG` structure contains the number of all ExtMem pages in the `usExtMemSize` field. This is the sum of User Type and Protected Type Pages. The value for the current WIBU-BOX/+ is 512 (16 Kbytes).

All these operations require a WIBU-KEY API of version 4.0 or higher.



ExtMem is fully supported on Win32 starting with Windows 95 res. WIBU-KEY 4.0, on Macintosh OS X or later and on Linux. There is no support of ExtMem on 16-Bit-Windows, DOS or OS/2.

All supporting platforms permit also the unlimited access of ExtMem via WkLAN.

More details how ExtMem can be read and written are found in the WIBU-KEY User's Guide and the examples of the WIBU-KEY Development Kit.

2.2 Addressing ExtMem Pages via the WIBU-KEY COM API

This method is described in the WIBU-KEY User's Guide, chapter [TBA].

2.3 Accessing Protected Type ExtMem

Reading or Writing a Protected Type ExtMem page required always an 8-byte **Access Signature**. The value of this signature may be different for reading and writing. For reading of an ExtMem page always the *same* Access Signature is used; in dependence of the used security mode, for writing the Access Signature may be *different* for every page. Here are the precise rules:

- Read-Access and Write-Access Signatures depend on the Firm Code and User Code of the *first* WIBU-BOX entry.
- The Read-Access Signature is influenced by the content of *Added Data* or an *Expiration Date* in the first entry if the *Data Vary Flag* is set too. "Content" includes all 5 bytes and the *Vary Flag* setting. If the *Data Vary Flag* is not set, the Read-Access Signature depends only on the stored Firm and User Code.
- The Read-Access Signature is always independent from the Serial Number or the Global Programming Counter – this means it is identical for all WIBU-BOXes with identical contents in the first entry.
- The Read-Access Signature and Write-Access Signature are identical if the first WIBU-BOX entry does not contain an Added Data Entry or an Expiration Date.
- The Write-Access Signature depends on the *Serial Number*, if the first entry contains Added Data or an Expiration Date and the *Serial Vary Flag* is set. As result, the Write-Access Signature is different for each WIBU-BOX.
- The Write-Access Signature depends on the *Global Programming Counter*, if the first entry contains Added Data or an Expiration Date and the *Count Vary Flag* is set. As result, the Write-Access Signature changes after each protected WIBU-BOX programming operation.
- The Write-Access Signature depends on the *ExtMem Page Index*, if the first entry contains Added Data or an Expiration Date and the *Data Vary Flag* is set. As result, the Access Signature is different for each page.



If the first WIBU-BOX entry is *empty* or contains *User Data*, the Access Signature cannot be determined and Protected Type ExtMem pages cannot be addressed.

Other WIBU-BOX entries do *not* influence the address of ExtMem.

If the first entry contains an Expiration Date, then this is interpreted as Added Data. Changing this Expiration Date modifies always the Read-Access Signature if the Data Vary flag is set.

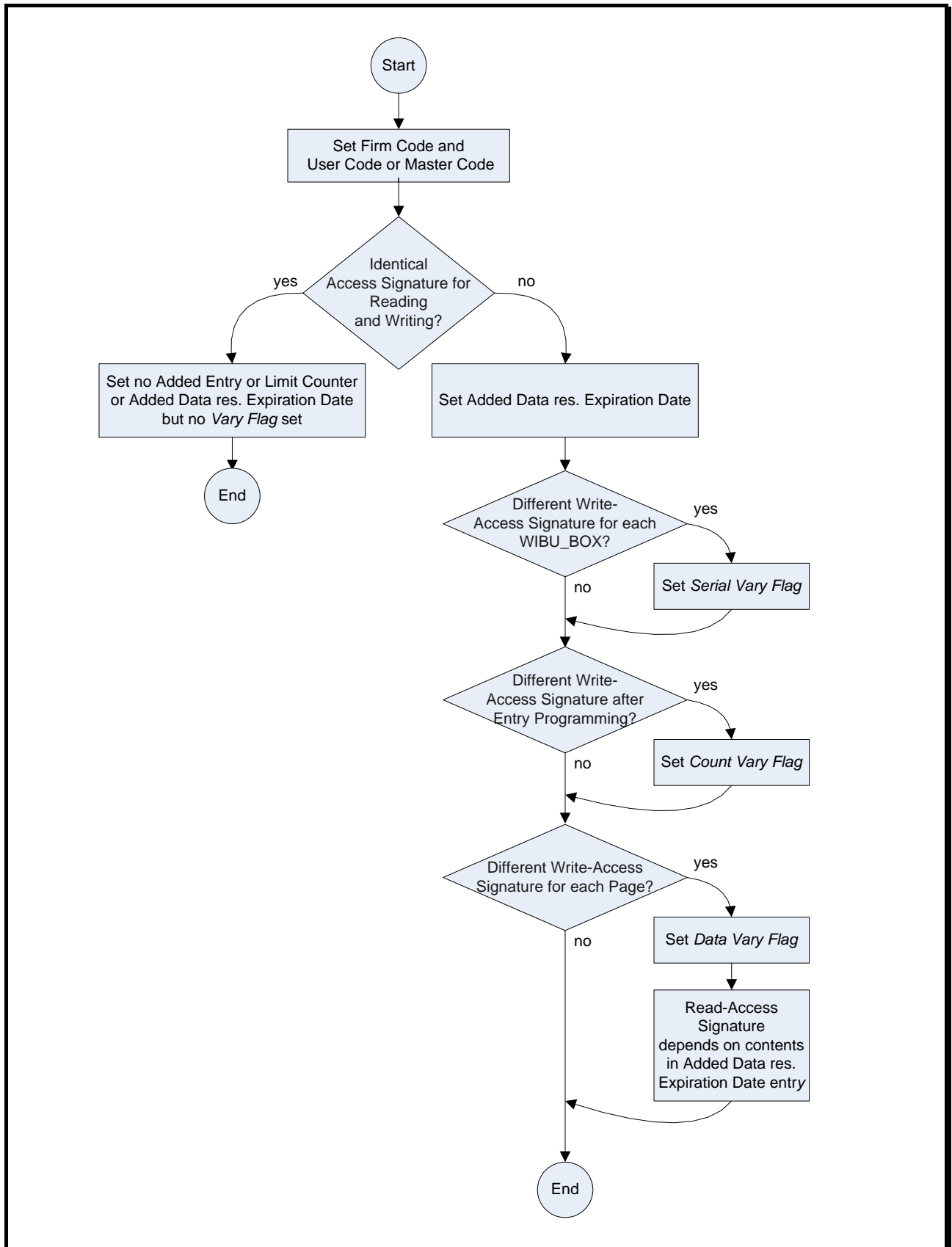
The data stored in the Added Data or Expiration Data influences the Read-Access Signature but not the Write-Access Signature.

The following table explains this behavior in dependency of the content of the first WIBU-BOX entry:

Scenario	Content of first WIBU-BOX entry	Read-Access Signature	Write-Access Signature
1	Empty or User Data	Undefined	Undefined
2	Firm Code and User Code (or Master Code) with no added entry of Limit Counter	Depends on Firm Code and User Code (fixed)	Identical with Read-Access Signature
3	Firm Code and User Code (or Master Code) with Added Data Entry or Expiration Date, no <i>Vary Flags</i> set.	Same as 2	Same as 2
4	Same as 3 but <i>Serial Vary Flag</i> set	Same as 2	Same as 2 and depends on Serial Number
5	Same as 3 but <i>Count Vary Flag</i> set	Same as 2	Same as 2 and depends on Global Programming Counter
6	Same as 3 but <i>Data Vary Flag</i> set	Same as 2 and depends on content of Added Data or Expiration Date (variable)	Same as 2 and depends the index of the accessed ExtMem page.

Any combination of Scenario 4 to 6 is permitted and combines the dependencies for the Write-Access Signature.

The following picture helps to decide which of the scenarios above the best for a specific application is.



2.4 Raw Mode and Formatted Mode ExtMem Organization

After the decision how the ExtMem pages are accessed during the Read and Write Operations, the organization of ExtMem must be determined:

- By default, the ExtMem is organized in the **Raw Mode**. Here all pages are unstructured and can be freely read and written by software, which uses the WIBU-BOX.
- If the ExtMem should be used by specific programs of WIBU-SYSTEMS, special WIBU-SYSTEMS partners, by the licensor and by the user as well, an organization of the shared memory is required to avoid that these 4 instances are overwriting the memory vice-versa. This is solved by formatting the ExtMem and using the ExtMem in **Formatted Mode**.

The Formatted Mode is explained in the next chapter in greater detail. The formatting is done with WkCrypt (see chapter 3.1).

If the Formatted Mode is active, it is always used for the *User Type* **and** the *Protected Type* pages. For example, it is not possible to address User Type pages in Raw Mode and Protected Type pages in Protected Type or vice-versa.


The Formatted Mode is activated by writing a special and fix signature (a GUID) into the first 16 bytes of the last User Mode **and** Protected Mode page. This GUID has the following form:

```
{00090001-0000-1200-8002-0000C06B5161}
```

This represents the following byte sequence:

```
01 00 09 00.00 00 00 12:80 02 00 00.C0 6B 51 61
```

Whenever tools of WIBU-SYSTEMS detect this sequence in the first 16 bytes of the last page of User Type or Protected Type page, the Formatted Mode is assumed.

 To avoid that ExtMem in Raw Mode is erroneously interpreted as Formatted Mode by WIBU-SYSTEMS tools, these 16 bytes should **never** be stored in Raw Format into the last page of User Type or Protected Type ExtMem.

If the ExtMem is properly formatted, the last page of the User Type and Protected Type is always **identical**. In both pages, the bytes 16 to 32 define how the pages with lower index are used. Two adjacent bytes define a little-endian 16-bit value, which specifies the length of a specific **Format Block** in pages:

- Byte 16/17: Raw Block User Type length, **RawUserSize**
- Byte 18/19: Licensor Block User Type, **LicensorUserSize**
- Byte 20/21: OEM Block User Type, **OemUserSize**
- Byte 22/23: WIBU Block User Type, **WibuUserSize**
- Byte 24/25: Raw Block Protected Type, **RawProtectedSize**
- Byte 26/27: Licensor Block Protected Type, **LicensorProtectedSize**
- Byte 28/29: OEM Block Protected Type, **OemProtectedSize**
- Byte 30/31: WIBU Block Protected Type, **WibuProtectedSize**

The sum of the values in byte 16 to 23 must be identical with the sum of values in byte 24 to 31 and the sum of both must be identical with the number of available pages in the ExtMem - 2.

The next table describes the type of the 4 different format blocks:

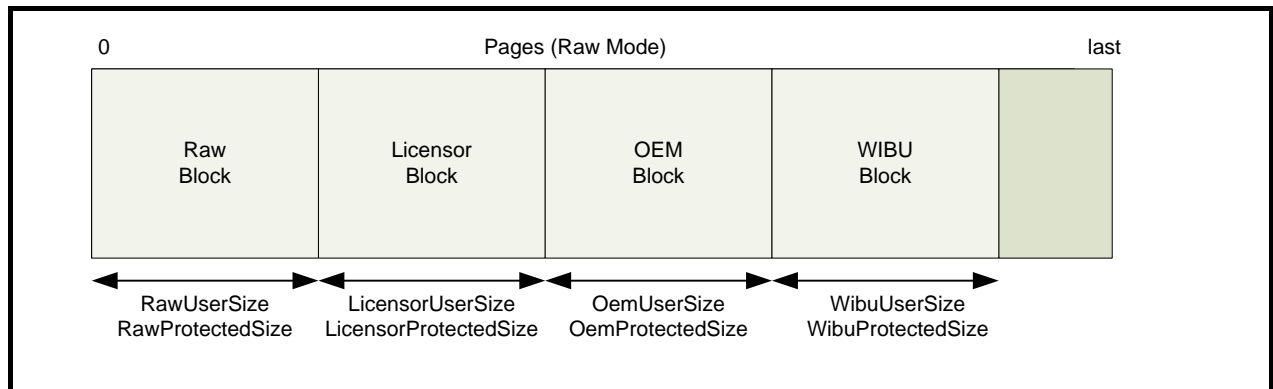
Name	Description	Access by Licensor Application
Raw	Freely accessible by interactive tools like WIBUKEY Control Panel. Can also be written by WkCrypt and WkList32.	Yes (see chapter 2.5)
Licensor	Accessible only by applications of a WIBU-KEY Licensor, can also be written by WkCrypt and WkList32	Yes (see chapter 2.6)
OEM	Reserved for special partners of WIBU-SYSTEMS which share their products with Licensors of WIBU-KEY.	no
WIBU	Reserved for WIBU-SYSTEMS.	no

The OEM and WIBU Format Block cannot be directly addressed by page reading or writing operations in WkCrypt and WkList32. But they can be written or read with special operations in such tools.



The WIBU-KEY API does not distinguish between Raw Mode and Formatted Mode. It sees ExtMem memory always in Raw Mode. The user of this API is responsible that no Formatted Mode data are unexpectedly destroyed by wrong Raw Mode write operations.

The following picture explains how the 4 Format Blocks are located in the ExtMem area. This model is identical for User Type and Protected Type:



2.5 Addressing Raw Data in Formatted Mode

The addressing of Raw Data in Formatted Mode is identical with that in Raw Mode with one important exception: The maximum index of the Raw Block is no longer limited by the ExtMem itself but by the Raw Block. This size value is determined by following operations, which are identical for User Type and Protected Type:

- Determine the index of the last ExtMem page.
- Read this page and check the GUID (see chapter 2.4). If it is not found, the ExtMem is used in Raw Mode and all pages can be addressed without restrictions.
- Otherwise read the value at 16/17 for User Type res. 24/25 for Protected Type. The data is stored in little endian, so the complete number is: byte 16 + 256 * byte 17 res. byte 24 + 256 * 25. This value is **RawUserSize** res. **RawProtectedSize**.
- If this value is 0, no Raw Block exists. Abort the operation then.
- The Raw Block can now have any index between 0 and (**RawUserSize-1**) res. (**RawProtectedSize-1**).

Typically the first page of the block is used to describe the structure of the Raw Block itself. This may be freely defined by the Licensor (or user). To define a proper start, the Format operation writes always 32 0 bytes in the first Raw Block page (User Type and Protected Type).

2.6 Addressing Licensor Data in Formatted Mode

The addressing of Licensor Data in Formatted Mode is a little more complicated than that in Raw Mode because the first page has no longer index 0. The first index and the last index are determined by following operations, which are identical for User Type and Protected Type:

- Determine the index of the last ExtMem page.
- Read this page and check the GUID (see chapter 2.4). If it is not found, the ExtMem is used in Raw Mode and no Licensor Block exists. Abort the operation then.
- Otherwise read the value at 16/17 for User Type res. 24/25 for Protected Type. The data is stored in little endian, so the complete number is: byte 16 + 256 * byte 17 res. byte 24 + 256 * 25. This value is **RawUserSize** res. **RawProtectedSize** and describes the *first* index of the Licensor Block.
- Read the value at 18/19 for User Type res. 26/27 for Protected Type. Both are stored in the same endian format. This value is **LicensorUserSize** res. **LicensorProtectedSize** and describes the number of available blocks in the Licensor Block.
- If this value is 0, no Raw Block exists. Abort the operation then.
- The Raw Block can now have any index between **RawUserSize** and (**RawUserSize+LicensorUserSize-1**) res. **RawProtectedSize** and (**RawProtectedSize+LicensorProtectedSize-1**)

Typically the first page of the block is used to describe the structure of the Licensor Block itself. This may be freely defined by the Licensor. To define a proper start, the Format operation writes always 32 0 bytes in the first Licensor Block page (User Type and Protected Type).

3 Managing ExtMem with WkCrypt

WkCrypt Version 4.10 supplies some new command line options to format and un-format ExtMem as well as to read ("list") and write ExtMem in Raw Mode and Formatted Mode.

To determine ExtMem size of ExtMem altogether and determine the block sizes in Formatted Mode, the /L option can be used. At LPT1 there is a WIBU-BOX in Raw Mode:

```
C>wkcrypt /PAL1 /L
wkcrypt - WIBU-KEY Encryption and Programming Tool.
Version 4.10 of 2004-May-12 (Build 23) for Win32.
Copyright (C) 1989-2004 by WIBU-SYSTEMS AG. All rights reserved.

### WIBU-BOX no. 1 at LPT1 [0x03BC] (Version 6):
```

1:	10:130671	2:	3:
4:		5:	6:
7:		8:	9:
10:		Serial-No 008-0000100002 <04253> BD C4	

* ASIC Build Number: 818, ExtMem: 16 KBytes (not formatted).

If the ExtMem is in Formatted Mode, also the 8 block sizes are returned, for example at USB:

```
C>wkcrypt /PAU /L
wkcrypt - WIBU-KEY Encryption and Programming Tool.
Version 4.10 of 2004-May-12 (Build 23) for Win32.
Copyright (C) 1989-2004 by WIBU-SYSTEMS AG. All rights reserved.

### WIBU-BOX no. 1 at USB (Version 6):
```

1:	10:13	2:	10:123	3:
└─# 0-2	┆SD	└─		
4:		5:		
7:		8:		9:
10:		Serial-No 008-0000012345 <00030>		

* ASIC Build Number: 818, ExtMem: 16 KBytes formatted, page structure:
 * User: Raw = 128, Licensor = 0, OEM = 0, WIBU = 127.
 * Prot: Raw = 255, Licensor = 0, OEM = 0, WIBU = 0.

This shows that 128 pages are available in the Raw Block and 127 pages in the WIBU Block of User Type and that all pages (but not the last) are available for the Raw Block with Protection Type.

3.1 Formatting and Un-Formatting ExtMem

To format the ExtMem of a WIBU-BOX, the WkCrypt option /PXM is used at command line or in a WKC file. To address a specific WIBU-BOX, the /PA option (port), /PB (a specific index) and /PQ (number of WIBU-BOXes to be operated at same time) can be used. The command expects the sizes of the Licensor, OEM and WIBU blocks for User Type and Protected Type. If less than 6 size specifications are done, the size of the missing block is set to 0. An error occurs if the sum of all pages exceeds the number of available pages and formatting is aborted. If the number of pages is less than are available, the rest is used as Raw Block.

The size values are specified after a colon, separated by comma (but no space). W describes the WIBU-BOX, O the OEM-Block and L the Licensor-Block, a preceding P defines Protected Type instead User Type (default).

The next command defines following blocks (and this is checked by an additional list command):

- User Type: 64 WIBU, 28 OEM, 100 Licensor, 63 Raw
- Protected Type: 0 WIBU, 0 OEM, 200 Licensor, 55 Raw

```
C>wkcrypt /PA1 /PXMW:W64,O28,L100,PL200 /L
wkcrypt - WIBU-KEY Encryption and Programming Tool.
Version 4.10 of 2004-May-12 (Build 23) for Win32.
Copyright (C) 1989-2004 by WIBU-SYSTEMS AG. All rights reserved.
```

```
WIBU-BOX 1: ExtMem area (512 pages) formatted.
### WIBU-BOX no. 1 at LPT1 [0x03BC] (Version 6):
```

1:	10:130671	2:	3:
4:		5:	6:
7:		8:	9:
10:		Serial-No 008-0000100002 <04253> BD C4	

```
* ASIC Build Number: 818, ExtMem: 16 KBytes formatted, page structure:
* User: Raw = 63, Licensor = 100, OEM = 28, WIBU = 64.
* Prot: Raw = 55, Licensor = 200, OEM = 0, WIBU = 0.
```

This example will be used in the next chapters.

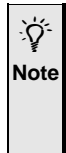
To un-format the ExtMem, the `/PXMW-` command is used; again a trailing `/L` command shows the result:

```
C>wkcrypt /PA1 /PXMW- /L
wkcrypt - WIBU-KEY Encryption and Programming Tool.
Version 4.10Gamma (Level 13) of 2004-May-12 (Build 23) for Win32.
Copyright (C) 1989-2004 by WIBU-SYSTEMS AG. All rights reserved.
```

```
WIBU-BOX 1: ExtMem area (512 pages) un-formatted.
### WIBU-BOX no. 1 at LPT1 [0x03BC] (Version 6):
```

1:	10:130671	2:	3:
4:		5:	6:
7:		8:	9:
10:		Serial-No 008-0000100002 <04253> BD C4	

```
* ASIC Build Number: 818, ExtMem: 16 KBytes (not formatted).
```



Formatting and Un-Formatting destroys all stored data when block sizes are changed. Otherwise at least the first page of each block is filled with 32 0-bytes by Format as initialization.

On the other hand, the Format or Un-Format command is not a “Wipe” command: Possibly secret data are not overwritten by this operation.

3.2 Writing ExtMem

To write one or more ExtMem pages with specific data sequences, the WkCrypt option `/PXMW` is used at command line or in a WKC file. To address a specific WIBU-BOX, the `/PA` option (port), `/PB` (a specific index) and `/PQ` (number of WIBU-BOXes to be operated at same time) can be used.

WkCrypt supports only the writing of pages in the Raw Block and Licensor Block in User Type or Protected Type. If the ExtMem is in the Raw Mode it is completed assumed as Raw Block.

The `/PXMW` option has following syntax

- An optional **P** describes Protected Type instead User Type.
- A required **R** or **L** describes the Raw Block or the Licensor Block.
- The index of the first listed page is followed.
- Optional with a colon separated the number of pages to read follows. The default value is 1
- The data are specified as a WkCrypt aggregate, separated by a comma. If the aggregate contains less data bytes than the written pages, the aggregate value is continuously concatenated.

The following example writes the sequence 5,6,7,8 repeatedly into the Protected Type Raw Block page 10 and the character sequence "ABCBBC" into User Type Licensor Block page 5-7.

```
C:>wkcrypt /PA1 /PXMWPR10,{5,6,7,8} /PXMWL5:3,{"ABC","B":2, "C":1}
wkcrypt - WIBU-KEY Encryption and Programming Tool.
Version 4.10 of 2004-May-12 (Build 23) for Win32.
Copyright (C) 1989-2004 by WIBU-SYSTEMS AG. All rights reserved.

WIBU-BOX 1: Start Writing ExtMem Protected Raw page 10.
WIBU-BOX 1: Writing Protected ExtMem Raw page 10.
WIBU-BOX 1: Start Writing ExtMem User Licensor page 5 to 7.
WIBU-BOX 1: Writing User ExtMem Raw page 68.
WIBU-BOX 1: Writing User ExtMem Raw page 69.
WIBU-BOX 1: Writing User ExtMem Raw page 70.
```

Using the /V option displays the written data in detail:

```
C:>wkcrypt /V /PA1 /PXMWPR10,{5,6,7,8} /PXMWL5:3,{"ABC","B":2, "C":1}
wkcrypt - WIBU-KEY Encryption and Programming Tool.
Version 4.10 of 2004-May-12 (Build 23) for Win32.
Copyright (C) 1989-2004 by WIBU-SYSTEMS AG. All rights reserved.

Used Format for Remote Programming: WIBU-SYSTEMS Control (WBC).
Port LPT1 used for programming WIBU-BOXes.
Preparing of Programming WIBU-BOX no. 1 (serial 8-100002) at LPT1:
  Start Writing ExtMem Protected Raw page 10.
  Writing ExtMem Protected Raw page 10 (hexadecimal data):
  05 06 07 08.05 06 07 08:05 06 07 08.05 06 07 08
  05 06 07 08.05 06 07 08:05 06 07 08.05 06 07 08
  Start Writing ExtMem User Licensor page 5 to 7.
  Writing ExtMem User Raw page 68 (hexadecimal data):
  41 42 43 42.42 43 41 42:43 42 42 43.41 42 43 42
  42 43 41 42.43 42 42 43:41 42 43 42.42 43 41 42
  Writing ExtMem User Raw page 69 (hexadecimal data):
  43 42 42 43.41 42 43 42:42 43 41 42.43 42 42 43
  41 42 43 42.42 43 41 42:43 42 42 43.41 42 43 42
  Writing ExtMem User Raw page 70 (hexadecimal data):
  42 43 41 42.43 42 42 43:41 42 43 42.42 43 41 42
  43 42 42 43.41 42 43 42:42 43 41 42.43 42 42 43
Completing of Programming WIBU-BOX no. 1 (serial 8-100002) at LPT1:
  4 programming commands executed.
```

3.3 Reading and Listing ExtMem

To read one or more ExtMem pages and list them at console, the WkCrypt option /PXML is used at command line or in a WKC file. To address a specific WIBU-BOX, the /PA option (port), /PB (a specific index) and /PQ (number of WIBU-BOXes to be operated at same time) can be used.

WkCrypt supports only the reading of pages in the Raw Block and Licensor Block in User Type or Protected Type. If the ExtMem is in the Raw Mode it is completed assumed as Raw Block.

The /PXML option has following syntax

- An optional **P** describes Protected Type instead User Type.
- A required **R** or **L** describes the Raw Block or the Licensor Block.
- The index of the first listed page is followed.
- Optional with a colon separated the number of pages to read follows. The default value is 1

The following example reads Protected Type Raw Block page 10 and User Type Licensor Block page 5-7 and lists the contents at console:

```
C:>wkcrypt /PA1 /PXMLPR10 /PXMLL5:3
wkcrypt - WIBU-KEY Encryption and Programming Tool.
Version 4.10 of 2004-May-12 (Build 23) for Win32.
Copyright (C) 1989-2004 by WIBU-SYSTEMS AG. All rights reserved.

WIBU-BOX no. 1 (serial 8-100002) at LPT1,
ExtMem Protected Raw page 10:
0000: 05 06 07 08.05 06 07 08:05 06 07 08.05 06 07 08 .....
0010: 05 06 07 08.05 06 07 08:05 06 07 08.05 06 07 08 .....
WIBU-BOX no. 1 (serial 8-100002) at LPT1,
```

```
ExtMem User Licensor page 5 to 7 (Raw 68 to 70):
0000: 41 42 43 42.42 43 41 42:43 42 42 43.41 42 43 42 ABCBCCABCBBCABCB
0010: 42 43 41 42.43 42 42 43:41 42 43 42.42 43 41 42 BCABCBBCABCBBCAB
0020: 43 42 42 43.41 42 43 42:42 43 41 42.43 42 42 43 CBBCABCBBCABCBBC
0030: 41 42 43 42.42 43 41 42:43 42 42 43.41 42 43 42 ABCBCCABCBBCABCB
0040: 42 43 41 42.43 42 42 43:41 42 43 42.42 43 41 42 BCABCBBCABCBBCAB
0050: 43 42 42 43.41 42 43 42:42 43 41 42.43 42 42 43 CBBCABCBBCABCBBC
```

3.4 Calculating ExtMem Access Signatures

When Access Signatures are required to access ExtMem Protected Type data, they can be calculated by specifying the `/PO` option, additionally to the `/PXML` and `/PXMW` command. Then no list or write operations are executed but the signatures listed at console. Here is an example of returning the Read-Access Signatures: To write the Raw Block page 10, a single 8-byte signature is returned; to write Licensor Block pages 5-7 three 8-bytes signatures are displayed:

```
C:>wkcrypt /L /PO /PXMWPR10,{0} /PXMWPL5:3,{0}
wkcrypt - WIBU-KEY Encryption and Programming Tool.
Version 4.10 of 2004-May-12 (Build 23) for Win32.
Copyright (C) 1989-2004 by WIBU-SYSTEMS AG. All rights reserved.
```

```
### WIBU-BOX no. 1 at LPT1 [0x03BC] (Version 6):
```

1: 10:13	2:	3:
# 1-5000 †SD		
4:	5:	9:
7:	8:	
10:	Serial-No 008-0000100002 <04255> BD C4	

```
* ASIC Build Number: 818, ExtMem: 16 KBytes formatted, page structure:
* User: Raw = 63, Licensor = 100, OEM = 28, WIBU = 64.
* Prot: Raw = 55, Licensor = 200, OEM = 0, WIBU = 0.
WIBU-BOX signature output redirection activated, physical programming stopped.
WIBU-BOX 1: Start Writing ExtMem Protected Raw page 10.
WIBU-BOX 1: Writing Protected ExtMem Raw page 10.
Used WIBU-BOX ExtMem write Access Signature: 124 91 89 4 39 32 140 13.
WIBU-BOX 1: Start Writing ExtMem Protected Licensor page 5 to 7.
WIBU-BOX 1: Writing Protected ExtMem Raw page 60.
Used WIBU-BOX ExtMem write Access Signature: 7 187 185 55 25 33 57 201.
WIBU-BOX 1: Writing Protected ExtMem Raw page 61.
Used WIBU-BOX ExtMem write Access Signature: 20 156 162 213 89 235 103 55.
WIBU-BOX 1: Writing Protected ExtMem Raw page 62.
Used WIBU-BOX ExtMem write Access Signature: 141 47 160 203 84 199 183 145.
```

The signature values are different because the WIBU-BOX entry 1 has set the *Data Vary Flag* (see `/L` command).

If the calculating of the Read-Access Signatures is done for the same pages in the following example, then the signatures are always identical, so normally the signature must be calculated only one for any page:

```
C:>wkcrypt /L /PO /PXMLPR10 /PXMLPL5:3
wkcrypt - WIBU-KEY Encryption and Programming Tool.
Version 4.10 of 2004-May-12 (Build 23) for Win32.
Copyright (C) 1989-2004 by WIBU-SYSTEMS AG. All rights reserved.
```

```
### WIBU-BOX no. 1 at LPT1 [0x03BC] (Version 6):
```

1: 10:13	2:	3:
# 1-5000 †SD		
4:	5:	9:
7:	8:	
10:	Serial-No 008-0000100002 <04255> BD C4	

```
* ASIC Build Number: 818, ExtMem: 16 KBytes formatted, page structure:
* User: Raw = 63, Licensor = 100, OEM = 28, WIBU = 64.
* Prot: Raw = 55, Licensor = 200, OEM = 0, WIBU = 0.
```

```
WIBU-BOX signature output redirection activated, physical programming stopped.
WIBU-BOX no. 1 (serial 8-100002) at LPT1,
ExtMem Protected Raw page 10:
Used WIBU-BOX ExtMem read Access Signature: 64 12 82 244 191 81 39 129.
WIBU-BOX no. 1 (serial 8-100002) at LPT1,
ExtMem Protected Licensor page 5 to 7 (Raw 60 to 62):
Used WIBU-BOX ExtMem read Access Signature: 64 12 82 244 191 81 39 129.
Used WIBU-BOX ExtMem read Access Signature: 64 12 82 244 191 81 39 129.
Used WIBU-BOX ExtMem read Access Signature: 64 12 82 244 191 81 39 129.
```

4 ExtMem Limitations of WIBU-KEY 4.10

Following features are current restrictions of WIBU-KEY 4.10. WIBU-SYSTEMS plan to fix them in future versions of WIBU-KEY:

- WkList32 and the WIBU-KEY control panel applet access the ExtMem always in Raw Mode even if the ExtMem is available in Formatted Mode. This possibly destroys the formatted memory.
- The WIBU-KEY COM API always addresses ExtMem in Raw Mode and supports only User Type reading and writing.
- Remote Programming is not supported to read, write, format or un-format ExtMem in Raw Mode or Formatted Mode.

5 Appendix

5.1 State of Current Version

This document describes the ExtMem management for Licensors of WIBU-SYSTEMS. It reflects the Version 4.10 of WIBU-KEY.

5.2 Contact Address

If you have further questions or comments, please contact:

WIBU-SYSTEMS AG
Website <http://www.wibu.com>
E-Mail info@wibu.com
Fax +49-721-93172-22
Phone +49-721-93172-0