

# **RevUp Documentation**

Boris Folgmann

<b>COLLABORATORS</b>
----------------------

	<i>TITLE :</i> RevUp Documentation		
<i>ACTION</i>	<i>NAME</i>	<i>DATE</i>	<i>SIGNATURE</i>
WRITTEN BY	Boris Folgmann	December 8, 2024	

<b>REVISION HISTORY</b>
-------------------------

NUMBER	DATE	DESCRIPTION	NAME

# Contents

<b>1</b>	<b>RevUp Documentation</b>	<b>1</b>
1.1	RevUp . . . . .	1
1.2	copyrights . . . . .	1
1.3	description . . . . .	2
1.4	requirements . . . . .	2
1.5	contents . . . . .	2
1.6	installation . . . . .	3
1.7	usage . . . . .	3
1.8	header . . . . .	4
1.9	include . . . . .	4
1.10	headerbeta . . . . .	5
1.11	includebeta . . . . .	5
1.12	version . . . . .	6
1.13	projectname . . . . .	6
1.14	dependencies . . . . .	7
1.15	revision . . . . .	7
1.16	beta . . . . .	7
1.17	noc . . . . .	8
1.18	noasm . . . . .	8
1.19	extra . . . . .	8
1.20	tiny . . . . .	8
1.21	history . . . . .	9

## Chapter 1

# RevUp Documentation

### 1.1 RevUp

RevUp - Amiga Revision Update System

Freeware © 1994 by PROXITY SOFTWARES

Development by Boris Folgmann

RevUp 1.2 (4.9.94) User Manual

Copyrights	Copyright information.
Disclaimer	Legal stuff.
Description	What is it for?
System requirements	What is needed?
Contents	Archive contents.
Installation	How to install.
Usage	How is it used?
History	What's new?
Support	How to contact us.
Update	Where to get new releases.
Credits	Thanks to some persons.

### 1.2 copyrights

#### COPYRIGHTS

Unless otherwise noted, all files are  
Freeware © 1994 by PROXITY SOFTWARES  
All Rights Reserved.

MagicWB © 1994 Martin Huttenloher

---

Kickstart and Workbench are Copyright © 1985-1994  
Commodore-Amiga, Inc.

## 1.3 description

### DESCRIPTION

- o RevUp generates and maintains include and header files with revision information.
- o RevUp is compatible to the CBM developer tool BumpRev, but offers a lot of new features.
- o Multiple dependency files are supported.
- o Beta count management for beta versions included.

## 1.4 requirements

### SYSTEM REQUIREMENTS

Kickstart 2.04  
Workbench 2.0

Workbench 2.1 for localized DOS error messages.

## 1.5 contents

### CONTENTS

This software package consists of the following files:

RevUp

The executable shell command.

RevUp.guide

This AmigaGuide document for Multiview.

PSI

Proximity Softworks information text.

All icons are part of MagicWB and included with permission of the author.

---

## 1.6 installation

### INSTALLATION

Simply copy RevUp to a directory in your path.

For example type:

```
copy RevUp to C:
```

## 1.7 usage

### USAGE

Usage: RevUp <version> <projectname> {<dependency>} [REV <revision>]  
[BETA] [NOC] [NOASM] [EXTRA] [TINY]

Template:

VERSION/N	Version number.
PROJECTNAME/A	Name of the project.
DEPENDENCIES/M	Files to check the date.
REV/K/N	Explicit revision number.
BETA/S	Generate beta information.
NOC/S	Don't generate C Headerfile.
NOASM/S	Don't generate ASM Includefile.
EXTRA/S	Generate extra information.
TINY/S	Don't print credits.

RevUp uses a revision storage file which contains the current revision number. Therefore it's possible to increment the revision number on every invocation of the program.

Look at the example smakefile for SuperTool to get an impression how RevUp is used.

```
SuperTool: SuperTool.c SCOPTIONS SMAKEFILE
  @RevUp 1 SuperTool SuperTool.c EXTRA BETA
  sc SuperTool.c@
```

```
RELEASE:
  @RevUp 1 SuperTool SuperTool.c EXTRA
  sc SuperTool.c NODEBUG OPT
```

Now have a look at the generated files SuperTool\_rev.h and SuperTool\_rev.i.

Call smake RELEASE to get the files without beta information.  
SuperTool\_rev.h, SuperTool\_rev.i.

Note that all definitions from TIME to HOST are only added if EXTRA is specified.

---

`BASENAME` is the project name in capital letters, e.g. `'MYTOOL'`. Useful for `ARexx` port or public screen naming. Note that language-sensitive functions are used to convert to upper case.

`USER` and `HOST` are imported from the environment variables `USERNAME` and `HOSTNAME` which are mentioned in the Amiga User Interface Style Guide.

To enable Commodore's `VERSION` command to display version information about your program you have to put `VERSTAG` in your code.

Example for `SuperTool`:

```
#include "SuperTool_rev.h"

const static char VersTag[] = VERSTAG;
```

## 1.8 header

```
/* C Headerfile generated by RevUp 1.2 */

#define VERSION 1
#define REVISION 1
#define DATE "17.8.94"
#define VERS "SuperTool 1.1"
#define VSTRING "SuperTool 1.1 (17.8.94)\r\n"
#define VERSTAG "\0$VER: SuperTool 1.1 (17.8.94)"
#define TIME "18:34:23"
#define PRGNAME "SuperTool"
#define VSTR "SuperTool 1.1 (17.8.94)"
#define BASENAME "SUPERTOOL"
#define USER "Boris Folgmann"
#define HOST "prox"
```

## 1.9 include

```
; * ASM Includefile generated by RevUp 1.2 *

VERSION EQU 1
REVISION EQU 1
DATE MACRO
    dc.b '17.8.94'
ENDM
VERS MACRO
    dc.b 'SuperTool 1.1'
ENDM
VSTRING MACRO
    dc.b 'SuperTool 1.1 (17.8.94)',13,10,0
ENDM
VERSTAG MACRO
    dc.b 0,'$VER: SuperTool 1.1 (17.8.94)',0
```

---

```
ENDM
TIME  MACRO
    dc.b  '18:34:23'
ENDM
PRGNAME MACRO
    dc.b  'SuperTool'
ENDM
VSTR  MACRO
    dc.b  'SuperTool 1.1 (17.8.94)'
ENDM
BASENAME  MACRO
    dc.b  'SUPERTOOL'
ENDM
USER  MACRO
    dc.b  'Boris Folgmann'
ENDM
HOST  MACRO
    dc.b  'prox'
ENDM
```

## 1.10 headerbeta

```
/* C Headerfile generated by RevUp 1.2 */

#define VERSION  1
#define REVISION 2
#define BETA     1
#define DATE     "3.9.94"
#define VERS     "SuperTool 1.2 BETA 1"
#define VSTRING  "SuperTool 1.2 BETA 1 (3.9.94)\r\n"
#define VERSTAG  "\0$VER: SuperTool 1.2 BETA 1 (3.9.94)"
#define TIME     "21:56:02"
#define PRGNAME  "SuperTool"
#define VSTR     "SuperTool 1.2 BETA 1(3.9.94)"
#define BASENAME "SUPERTOOL"
#define USER    "Boris Folgmann"
#define HOST     "prox"
```

## 1.11 includebeta

```
/* ASM Includefile generated by RevUp 1.2 */

VERSION  EQU 1
REVISION EQU 2
BETA     EQU 1
DATE  MACRO
    dc.b  '3.9.94'
ENDM
VERS  MACRO
    dc.b  'SuperTool 1.2 BETA 1'
ENDM
VSTRING MACRO
```



```
    dc.b  'SuperTool 1.2 BETA 1 (3.9.94)',13,10,0
ENDM
VERSTAG MACRO
    dc.b  0,'$VER: SuperTool 1.2 BETA 1 (3.9.94)',0
ENDM
TIME MACRO
    dc.b  '21:56:02'
ENDM
PRGNAME MACRO
    dc.b  'SuperTool'
ENDM
VSTR MACRO
    dc.b  'SuperTool 1.2 BETA 1(3.9.94)'
ENDM
BASENAME MACRO
    dc.b  'SUPERTOOL'
ENDM
USER MACRO
    dc.b  'Boris Folgmann'
ENDM
HOST MACRO
    dc.b  'prox'
ENDM
```

## 1.12 version

VERSION/N

This needed argument specifies the version number (the number before the dot).

Use 0 for beta versions and 1 for the first release version.

You should increase the version number only when major changes and improvements are done.

## 1.13 projectname

PROJECTNAME/A

This needed argument should be the name of your project. It will be used for the naming of the RevUp revision storage file ('projectname\_rev.rev'), the C header file ('projectname\_rev.h') and the assembly include file ('projectname\_rev.i').

Is is also used for building the version strings found in the header/include files, so keep care of the right capitalisation.

---

## 1.14 dependencies

DEPENDENCIES/M

RevUp checks the dates of the dependency files against the revision storage file. Therefore it's possible to update the revision number only if certain parts of the project are changed.

## 1.15 revision

REV/K/N

Explicitly set the revision to a specific value. Normally RevUp gets the old revision from `projectname_rev.rev` and increments it by 1.

## 1.16 beta

BETA/S

Enables the beta version count. If BETA is given the first time the revision is incremented as on a normal call. Additionally a beta count storage file is created. The beta count starts at 1 as the revision does.

Assume that version is 1 and revision is 1, with BETA specified the header/include file will look like this:

```
SuperTool_rev.h, SuperTool_rev.i
```

On every following call with BETA, only the beta count will be incremented, the revision will stay at 2.

If the beta stage of the program is finished call RevUp without BETA. The beta count storage file will be deleted and VERS will now look like this:

```
#define VERS "SuperTool 1.2"
```

That means all beta information removed.

NOTE

You should not publicly distribute versions of your program containing a VERSTAG with beta information because this string does not follow the CBM version string standard. Although all your beta testers will be able to correctly display the version information with the current version command (40.1).

HINT

---

Using the beta switch avoids to increment the revision on each compilation, therefore giving shorter revision numbers for your programs public releases.

BETA is only defined in the header file if specified in the commandline. So you could easily include additional debugging code like that:

```
#ifdef BETA
    kprintf("Variable x contains: %ld", x);
#endif
```

## 1.17 noc

NOC/S

Prevents RevUp from generating a header file for C language. Useful if version information is only used in assembly source codes.

## 1.18 noasm

NOASM/S

Prevents RevUp from generating an include file for 68k assembly language. Useful if version information is only used in C source codes.

## 1.19 extra

EXTRA/S

Adds some extra information to the header/include files which are not generated by CBMs BumpRev!

## 1.20 tiny

TINY/S

Supresses printing of the copyright message, therefore giving less output.

---

## 1.21 history

### HISTORY

1.1 (18.8.94) Release

First public release.

1.2 (4.9.94) Release

NEW: BETA switch enables beta version count.