# WarpRace

Sam Jordan

**COLLABORATORS**

| | *TITLE* :<br><br>WarpRace | |
|---|---|---|

| *ACTION* | *NAME* | *DATE* | *SIGNATURE* |
|---|---|---|---|
| WRITTEN BY | Sam Jordan | August 7, 2022 | |

**REVISION HISTORY**

| NUMBER | DATE | DESCRIPTION | NAME |
|---|---|---|---|
| | | | |

# Contents

# Chapter 1

# WarpRace

## 1.1 WarpRace

WarpRace

1997 by Sam Jordan
© HAAGE & PARTNER Computer GmbH

The modular performance measurement program



## 1.2 Introduction

Warprace is a modular program for performance measurement. It supports both 68K and PPC processors and is thus perfectly suitable to compare the performance of different processors. Warprace is also suitable to test memory performance as well as overhead times at context switches.

The modular concept of WarpRace allows every programmer to develop new modules which are executed by WarpRace. This document explains detailed how to develop new modules.

Warprace requires at least a 68020 processor and OS2.0. To execute PPC modules you need any PPC processor and any version of the powerpc.library.

## 1.3 Installation

The installation of WarpRace is very easy. Just copy the 'WarpRace' directory whereever you want. It's possible not to copy the directory, but if WarpRace is located on a CD or any other write-protected device it's recommended to copy the directory to allow adding new modules.

## 1.4 Usage

WarpRace is a program which looks for modules in a certain directory and which executes them. Modules are standard executable programs which get special input parameters and which return special output parameters. Every module has the suffix '.wrm' (WarpRace module). The module must be located in the directory 'Modules'. However it's possible to create sub-directories to put modules into groups or to assemble a test serie.

WarpRace is controlled from the CLI. With 'WarpRace ?' a list of all CLI parameters is printed out. It follows a description of all CLI parameters:

```
MODULES/M       - All modules to be executed are specified here. It must
      not be specified a path because all modules are searched
      in the directory 'Modules' and in further sub-directories.
      The modules can be specified also without the suffix '.wrm'.
      It's also possible to specify a directory (again without
      path!), then all modules which are located in this
      directory, are executed.

M=M68K/S        - Usually, WarpRace executes only modules which support the
      PPC processor. If the switch M68K is specified, WarpRace
      executes all modules which support the 68K processor.
      A modul can also support both processors. It gets from
      WarpRace the information, which test is desired, so
      it can branch appropriately.

A=ALL/S         - If this switch is specified, WarpRace executes all modules
      which support the current processor and which are located
      in the 'Modules' directory.

F=FULL/S        - If this option is specified, a more detailed desription
      is printed out for every module executed.

S=STATS         - This parameter requires additionally a file name (no path).
      WarpRace then creates a special formatted output with the
      results of all tests. These results are written under the
      file name specified, into the 'Stats' directory.
```

Some examples of the usage of WarpRace:

```
warprace        ALL                     ;executes all PPC modules
warprace        ALL M68K                ;executes all 68K modules
warprace        LongRead LongWrite      ;executes the modules 'LongRead'
          ;and 'LongWrite' on the PPC
warprace        Memory M68K             ;executes all modules in the directory
```

```
          ;'Memory' on the 68K
warprace        Copy STATS test.txt     ;executes all modules in the directory
          ;'Copy' on the 68K and creates a
          ;statistical output to 'Stats/test.txt'
```

## 1.5 Output format

```
WarpRace prints out a title text after startup. After that two further
information are printed out:

CPU:                         - The processor to be tested.
Version of powerpc.library - The version of the powerpc.library


The output of every module is standardized. It follows an explanation
of the output information.

Module:          - The name of the module and its version number
Author:          - The author of the module
Short:           - A short description of the module
Description:     - A more detailed description of the module. This text
      only appears, if the CLI parameter FULL was specified.
Result:          - The result of the test. The module is free to choose
      the output format. For example it can output a time
      value of a memory performance value.
```

## 1.6 Statistics

```
WarpRace creates a special formatted output with the test results, if
it is desired. In this case the CLI parameter STATS must be specified
with a file name.

The purpose of this output is to allow to make a presentation of the
results (for example graphical diagram). The special formatting should
ease the evaluation of the results.

WarpRace doesn't contain such an evaluation program at the moment. So
everyone can write such a program which should make the results much
more readable.

In the following the formatting of the output file is explained. Every
information has the following syntax:

KEYWORD=Information

There are keywords with global character and there are others which
are module specific.


Global keywords (can appear everywhere in the file):

CPU=<CPUString>           - The processor type as string (i.e. PPC604E)
```

```
LIBVER=<Versionnumber>   - The version of the powerpc.library (i.e. 12.0)


Module specific keywords:

NAME=<Modulname>         - The name of the module (i.E. TurboCopy). This keyword
         is ALWAYS the beginning of a new module.
VERSION=<Versionsnumber> - The version of the module (i.e. 1.0)
RESTYPE=<Result type>    - A numeric value which describes the type of the
         result. The possible values are specified in the
         include file 'warprace.i'.

       Possible values for the result type:

       0     - The result type is unknown and it's not
         possible to do comparisons.
       1     - The result type is unknown, but the result
         is proportional to the power (i.e.
         memory performance values) so it's possible
         to do comparisons.
       2     - The result type is unknown, but the result
         is invers proportional to ther power
         (i.e. time values) so it's possible to do
         comparisons.
       3     - The result has the type 'Number of microseconds'
         (and is therefore invers proportional to ther
         power). The result is a numeric value.
       4     - The result has the type 'Bytes per second'
         (memory performance) and is therefore
         proportional to the power. The result is a
         numeric value.

  New values are eventually added in future if this is desired.

RESULT=<Result>          - The result which can be interpreted with the result
         type.
```

## 1.7 Developer infos

```
In the following it is explained in detail how to create a WarpRace module.
A WarpRace module is a standard executable program which gets and
returns other parameters as usually.

If a module is written in a high level language, only the prototype
of the 'main' function has to be adapted. Additionally the program
has to be linked without startup code. A module should also not
print out information directly to the CLI. Error messages should be
printed out using the parameters explained below.

In the directory 'ModSrc' there are located some examples of modules
which were written in assembler.

IMPORTANT: The 'main' function must exist in 68K code even if the
module only supports the PPC processor. Then the 'main' function
has to call the PPC part of the module.
```

The structures and definitions necessary are located in the include files 'warprace.h' (C) and 'warprace.i' (assembler).

The prototype of the 'main' function of a module looks like this:

```
  struct WR_Output* main(WR_ID, struct WR_Input*)
  d0                     d0     a0
```

The input parameters are transferred in registers as well as on the stack.

The input parameters:

WR_ID : This is a constant (defined in the include file) which can be used to evaluated if the module was really started by WarpRace. If this is not the case, the program should terminated properly.

WR_Input : This structure contains some important information for the module. It has the following format:

```
      struct WR_Input {
    APTR    WRI_PowerPCBase;
    ULONG   WRI_Version;
    void    (*WRI_StartTimer_68K)(void);
    ULONG   (*WRI_StopTimer_68K)(void);
    void    (*WRI_StartTimer_PPC)(APTR);
    ULONG   (*WRI_StopTimer_PPC)(APTR);
    APTR    WRI_LinkerDB;
    BOOL    WRI_68K;
    ULONG   WRI_Flags;
    APTR    WRI_Ext;
      };
```

The elements have the following meaning:

WRI_PowerPCBase - Base address of the powerpc.library
WRI_Version - Version of the powerpc.library
WRI_StartTimer_68K - Pointer to a 68K function which starts the internal timer. This function is always called at the beginning of a time measurement.
WRI_StopTimer_68K - Pointer to a 68K function which stops the internal timer. This function returns the number of microseconds elapsed between the call of WRI_StartTimer_68K and WRI_StopTimer_68K.
WRI_StartTimer_PPC - Pointer to a PPC function which starts the internal timer. This function gets the element WRI_LinkerDB as input parameter.
WRI_StopTimer_PPC - Pointer to a PPC function which stops the internal timer. This function gets the element WRI_LinkerDB as input parameter and returns

```
         the number of microseconds elapsed between
         WRI_StartTimer_PPC and WRI_StopTimer_PPC.
      WRI_LinkerDB       - This element must be passed to the PPC timer
         functions as input parameter if these functions
         are called.
      WRI_68K            - If this switch is set, the module should execute
         on the 68K processor otherwise on the PPC. If
         the module doesn't support the desired CPU it
         should return with NULL as output parameter.
      WRI_Flags          - Not used yet.
      WRI_Ext            - Not used yet.

      The timer functions must not be used 'mixed'.


The output parameter:

WR_Output:  Pointer to a WR_Output structure or NULL, if the module can't
      be executed (i.e. if the desired CPU is not supported). If NULL
      is returned then no output is done to the CLI window. If the
      module recognizes an error it can return the WR_Output structure
      and fill the error variables appropriately. In this case the
      error message is printed out in the CLI window.

      The WR_Output structure has the following format:

      struct WR_Output {
    STRPTR  WRO_Modname;
    STRPTR  WRO_Short;
    STRPTR  WRO_Description;
    STRPTR  WRO_Author;
    ULONG   WRO_Version;
    ULONG   WRO_Revision;
    ULONG   WRO_RevSize;
    ULONG   WRO_Flags;
    STRPTR  WRO_Result;
    ULONG   WRO_ResultType;
    STRPTR  WRO_ResultString;
    APTR    WRO_ResultParams;
    ULONG   WRO_Status;
    STRPTR  WRO_ErrorString;
    APTR    WRO_ErrorParams;
    APTR    WRO_Ext;
      };

      The elements have the follwoing meaning:

      WRO_Modname        - The name of the module.
      WRO_Short          - A short description of the module (should not
         be longer than one row).
      WRO_Description    - A more detailed description of the module.
         Can comprise more than one line (but the
         last line should not contain a LineFeed).
      WRO_Author         - The name of the author.
      WRO_Version        - The version of the module.
      WRO_Revision       - The revision of the module (fractional part).
      WRO_RevSize        - Number of digits after decimal point. This
```

```
   avoids problems with version numbers of i.e.
   1.01. In this case, WRO_Revision is 1 and
   WRO_RevSize is 2.
WRO_Flags          - Not used yet.
WRO_Result         - The result of the measurement. This result
   is only used if the statistical output is
   enabled. Note that the result should be
   consistent with the result type.
WRO_ResultType     - The type of the result (see chapter @("Statistics" link ↩
    Statistics}).
   It is only used for statistical output.
WRO_ResultString   - A format string which is printe out by WarpRace
   at 'Result' using VPrintf.
WRO_ResultParams   - A pointer to the parameters for
   WRO_ResultString.
WRO_Status         - Can contain the following values:
   STATUS_SUCCESS : The module was executed
        successfully.
   STATUS_ERROR   : The module has recognized
        an error.
WRO_ErrorString    - A format string which is printed out using
   VPrintf in the case of an error.
WRO_ErrorParams    - A pointer to the parameters for WRO_ErrorString.
WRO_Ext            - Not used yet.


Please note that the structure must be filled completely and
correctly. You should also consider the case that the user has
enabled the statistical output.

Note again that the module should not return an error message if
it can't be executed because the CPU desired is not supported.
In this case it should return NULL. In this way it is avoided
that no obsolete information is printed out which could annoy
the user.
```