Compiling DOMLIB and other C programs

Api, then Cosi, then R2D2 and Compsys*
Revision of April 4, 2010

January 2009

Abstract

This document explains how the C files for DOMLIB, POLYLIB etc. are organized and compiled in the MMALPHA distribution.

To do List

2/1/2009: seems that POLYLIB-LIB is useless. Check it...

2008/10/04: Cannot compile stuff on Linux, on Irisa machine, on Mathematica Version 4.0.1, since I cannot run mprep... Why?

2008/10/04: Cannot compile Domlib properly for version `5.2`, but it works for version 6.0.

X Clean the mess in this file

? Add a DBG and a if in Makefile.darwin in domlib

X Add varlist option to check the value of all variables

1 Introduction

This document[1] is intended for MMALPHA developers who need to recompile or modify the DOMLIB.

*Api, Cosi and R2D2 are the names of the research groups that successively hosted research related on ALPHA and MMALPHA at Irisa, Rennes, France. Since 2001, Compsys in ENS Lyon also participates in the development of MMALPHA

[1]The source of this document is in:
`$MMALPHA/Mathematica/doc/sources/Compiling-Domlib/Compiling-Domlib.tex`
and this file should be in
`$MMALPHA/Mathematica/doc/developer/Compiling-Domlib.pdf`

2 About PolyLib

Almost all C programs of Mathematica depend on the polyhedral library POLYLIB, that was developped initally at IRISA and is now further developped and maintained in Strasburg. For convenience, the Mathematica distribution provides a version of POLYLIB. Before compiling any program, you should make sure that it is properly installed.

To be more explicit, the `POLYLIB` variable of file `Makefile.config` should contain the path to the directory where the POLYLIB is.

3 Where are the executable files?

They are located in `$MMALPHA/bin.$OSTYPE`, where `$OSTYPE` is the Operating System type. Currently, this type can be `cygwin`, `cygwin32`, `linux`.

Typically, one finds the following executables: `domlib`, `pip`, `code_gen`, `read_alpha`, `write_alpha` and `write_c`.

[2]

4 Environment variables

Two environment variables have to be set: `$MMALPHA`, `$OSTYPE`.

`$MMALPHA` the directory where MMALPHA is installed. It should have been defined for your installation to run (see the installation procedure).

`$OSTYPE` is the type of your operating system, which should be set when starting a shell. (I do not know what it should be for `cygwin`.)

5 Where are the source files?

The source files are in `$MMALPHA/sources`. There is one directory for each executable.

6 Compiling C files for MMAlpha

6.1 To Compile a New Version

Compiling a new version is a somewhat tricky (and risky) operation. Tricky, since you have to make sure that every variable, option etc. is set. Risky as you may erase some stuff that works pretty well in an old version.

So first, make sure that you have a copy of the binaries and of the libraries which work...

Then, good luck!

[2] In `bin.linux`, it seems that these executable files have a different name... This information has to be checked.

First, a description of the various directories. These depend on several parameters: your OS type (`darwin`, `cygwin`, `linux`, `unix`, `solaris`, etc.) and the Mathematica version. Your OS type should be set in an environment variable called OSTYPE, usually defined by the system (this is true on Unix-like systems, not necessarily on Windows...).

Compiling all programs The reference directory is `$MMALPHA/sources`. It contains `Makefile` which is used to compile whathever program you need. To compile, run

```
make all
```

Compiling one program To compile one particular program `pg`, run

```
make pg
```

You may even make several programs at once (`make pg1 pg2`).

Cleaning To clean before compiling, run

```
make clean
```

Seems however, not to clean everything.

Checking variables To check the variables, run

```
make checkvars
```

Testing stuff To test (no guarantee about what this tests), run

```
make tests
```

6.2 Source directories

The second important directory is ?$MMALPHA/sources/program where your program is (say for example Domlib). There, you will find source files for Domlib, a make file (`Makefile`) that will be called by the main make file, and other special make files depending on your OS type: `Makefile.darwin` or `Makefile.linux` etc...

6.3 Objects directories

There is also a directory `Obj.ostype` for each system type, which should be created by `Makefile` (not sure it is), and where the objects file are put as well as the final binary.

Finally, the binaries are all copied in the directory `$MMALPHA/bin.ostype`.

7 Versions...

The compilation process depends both on your operating system type and of your Mathematica version. I try to provide as many versions as possible for Mathematica version 6, 5 and 4[3]. However, prefered operating systems are Mac OSX and Linux.

Currently, the *mathematica version* is wired in the file `Makefile.config` of the directory `$MMALPHA/sources/MakeIncludes`. A variable called `MMAVERSION` containts its number (for example, `6.0`, or `4.0.1`). The value of this variable is very important. Indeed, the version number and the OS type define the suffixes of some make files. For example, if you want to compile for Mathematica version 6.0 on Linux, then `Makefile.config` will call `Makefile.linux`, which in turn will call `Makefile.linux.6.0` in the `MakeIncludes` directory. This allows some variables to be set properly depending on the OS and the Mathematica version. To understand that it is tricky, keep in mind that:

- The location of the preparation program (`mprep`) and the compiling add-ons of Mathematica necessary to prepare and compile `Domlib` depends strongly on the version.
- After version 5, `Domlib` had to be changed, as the API of the `Mathlink` library were changed...
- Some `lex` or `yacc` files had to be changed also for new versions of some OS...

Therefore, I suggest the following procedure.

1. Check that `$MMALPHA` and `$OSTYPE` are properly set.
2. Look for your Mathematica version. To be sure, start Mathematica and evaluate `$VersionNumber`.
3. In file `$MMALPHA/sources/MakeIncludes/Makefile.config`, check that the variable `MMAVERSION` is set to your version number.
4. Check that there exist a file `Makefile.$OSTYPE` in the directory `$MMALPHA/sources/MakeIncludes`. If it does not exist, create it, on the model of `Makefile.darwin` for example. You may have to modify the last line of this file:

   ```
   include ../MakeIncludes/Makefile.darwin.$(MMAVERSION)
   ```

   where you may replace `darwin` by the name of your operating system[4].

[3] I may soon give up for version 4, since version 7 is already here... PQ. Jan 2009.

[4] I could have replaced here the `darwin` name by variable `$OSTYPE`, but... depending on your configuration, this environment variable may be defined in various ways.

5. Check that there exist a file `Makefile.$OSTYPE.MMAVERSION` in the directory `$MMALPHA/sources/MakeIncludes`. If it does not exist, create it, for example, by a modification of `Makefile.darwin.6.0`. In this file, you have to set the `MATHLINK` variable in the second line (the remaining should be left unchanged.) This variable says where the compiler additions of your Mathematica distribution are located. This depends on your configuration and on your Mathematica version. On my own machine, for version 6, they are

   ```
   MATHLINK = /Applications/Mathematica.app/SystemFiles/Links/MathLink/\
       DeveloperKit/CompilerAdditions
   ```

   and for version 5.2:

   ```
   MATHLINK = /Applications/Mathematica\ 5.2.app/AddOns/MathLink/\
        DeveloperKit/MacOSX-x86/CompilerAdditions
   ```

If you have a problem, run a checkvars, and ... check the values of you variables.

8 How things are

The `sources` directory contains one directory per C software: `Code_Gen`, `Domlib`, `Pip`, `Poly`, `Poly-darwin`, `Polylib`, `Pretty`, `Read_Alpha`, `Write_Alpha`.

The `sources` directory contains one directory per C software: `Code_Gen`, `Domlib`, Pip, Poly, Poly-darwin, Polylib, Pretty, Read_Alpha, Write_Alpha.

8.1 The `sources/Makefile`

There is one Makefile in this directory. It contains a variable called DIR which contains the list of programs to compile. There is one `Makefile` in the source directory. It contains a variable called `DIR` which contains programs to compile. For each program `pg` to compile, `Makefile` calls `pg/Makefile`. For example, to compile `Domlib`, it goes in directory `Domlib` and calls `make`.

8.2 Where the binaries are

All binaries produced by compilation are in file `../bin.OSTYPE`, where `OSTYPE` is defined as an environment variable (this may cause a problem, as the `OSTYPE` is probably not sufficient to decide which binary to use). Another flag, `MACHTYPE` is probably also needed... at least for `MaCOS`.

8.3 Where the libraries are

The libraries produced by compilation are in file `../lib.OSTYPE`.

In each one of these directories, there is a makefile, which starts with an include of another file `../MakeIncludes/Makefile.config`.

8.4 The directory `sources/MakeIncludes`

In this directory, there are a set of make files that are common to most of the programs to be compiled. These files are :

- `Makefile.config`. A configuration file.
- `Makefile.OSTYPE`, for all OS types. In this file, which are called by `Makefile.config`, there are configurations commands which depend on the OS type.
- `Makefile.OSTYPE.MMAVERSION`. These files contain commands that are specific to a Mathematica version, for a particular OS type. They are called by `Makefile.OSTYPE`.
- A file `Makefile.checkvars` which is called when the option `checkvars` is set. This option does not work totally properly.

Thus we have the following chain of makes:

1. `Makefile` (in `sources`)
2. Calls a make file `Makefile` located in some directory of DIR (say `read_alpha`).
   (a) This `Makefile` includes `Makefile.config`, which is common to all programs, and is in directory `sources/MakeIncludes`.
   (b) `Makefile.config` includes itself a special configuration file customized for a system, named `Makefile.OSTYPE` and located in the same directory, which may redefine some variables that were already defined in `Makefile.config`.
   (c) `Makefile.OSTYPE` includes another file, named

   `Makefile.OSTYPE.version`

   located in the same directory, where `version` is the version number of Mathematica you are using (say 5.0, 5.2, 6.0). In this file, the set-up of variables related to Mathlink is done, as these may depend on the version of Mathematica.
3. After including, `Makefile` includes also `Makefile.rules` also located in `sources/MakeIncludes`, which contains rules. `Makefile.OSTYPE` in directory `read_alpha` where there are special rules for compiling this program on operating system `OSTYPE`.

9 The configuration files, details

9.1 `Makefile.config`

Defines `SHELL`, `MMAVERSION`, `LDLIB`, `CC`, `CFLAGS`, `CXX`, `YACC`, `LEX`, `LFLAGS`, `STRIP`, `MAKEFILES`, `BINDIR`, `LIBDIR`, `OBJDIR`, `NEEDBINDIR`, `NEEDLIBDIR`, `NEEDOBJDIR`, `SOURCES`, `POLYDIR`, `POLYINCLUDE`, `POLYOBJS`, `CODEGENDIR`, `CODEGENOBJS`, `PRETTYDIR`, `PRETTYOBJS`, `LPSOLVEDIR`, `OMEGALIB`, `OMEGABASIC`.

9.2 The file Makefile.cygwin

This is a configuration file where special variables are set. Defines SHELL, MMALPHA, MMAVERSION, SUFFIX, DEFINES, LD, PRECISIONFLAG, LDFLAGS, MATHLINK, OMEGADIR, MATHPREP, MATHPREPCOR, MATHINCLUDE, MATHLIB, MATHLOADLIB, MATHEXTRALIBS and MATHEXTRAOBJS.

All variables related to Mathlink are set in a file `Makefile.darwin.mmaversion` where `mmaversion` is the number of the Mathematica version for which you compile. There exists only a file for version 4.2 of Mathematica.

Notes: the Mathlink stuff is actually copied in the subdirectory $MMALPHA/sources/Mathlink.

9.3 The file Makefile.linux

Defines DEFINES, LD, PRECISIONFLAG, LDFLAGS, MATHLINK, OMEGADIR, MATHPREP, MATHPREPCOR, MATHINCLUDE, MATHLIB, MATHLOADLIB, MATHEXTRALIBS and MATHEXTRAOBJS. There is a file Makefile.linux-new. I do not know the differences.

In the `linux` file, there is the definition of `DEFINES`, `LD` (`gcc`, `PRECISIONFLAGS`, `LDFLAGS`. `OMEGADIR`. Then this file calls in the same directory another configutation file related to Mathematica (see 9.6).

All variables related to Mathlink are set in a file `Makefile.linux.mmaversion` where `mmaversion` is the number of the Mathematica version for which you compile.

MATHLINK, OMEGADIR, MATHPREP, MATHPREPCOR, MATHINCLUDE, MATHLIB, MATHLOADLIB, MATHEXTRALIBS and MATHEXTRAOBJS.

Notes: the MMAVERSION variable is currently wired in the file `makefile.linux` as 4.2, thus making a call to `makefile.linux.4.2`. There exists another such file, for version 3.0.1, referring to the location of Mathematica at Irisa, but I think that it is obsolete.

In the `cygwin` file, there are special definitions (this file is not up-to-date).

In the `darwin` file, there is the definition of `DEFINES`, `LD` (`gcc`, `LDFLAGS`. Then this file calls in the same directory another configutation file related to Mathematica (see 9.6).

9.4 The file Makefile.darwin

Defines DEFINES, LD, PRECISIONFLAG, LDFLAGS, OMEGADIR.

All variables related to Mathlink are set in a file `Makefile.darwin.mmaversion` where `mmaversion` is the number of the Mathematica version for which you compile.

MATHLINK, OMEGADIR, MATHPREP, MATHPREPCOR, MATHINCLUDE, MATHLIB, MATHLOADLIB, MATHEXTRALIBS and MATHEXTRAOBJS.

9.5 The Makefile.rules file

This file contains general rules to compile programs. These rules are explained in the file itself.

9.5.1 List of variables, and where they are defined

In Makefile.config, there is an inclusion of the definitions in MakeIncludes/Makefile.$(OSTYPE) which are peculiar to each OS type.

BINDIR : in Makefile.config. $(MMALPHA)/bin.$(OSTYPE). Where the binaries will be put.

CC : in Makefile.config. I guess it is the compiler...

CFLAGS : in Makefile.config. -O3 -g -I$(POLYINCLUDE) $(PRECISIONFLAG). I guess that these are the flags of the compiler.

CODEGENDIR : in Makefile.config. $(SOURCES)/Code_Gen. Where Code_Gen is

CODEGENOBJS : in Makefile.config. $(OBJDIR)/gen.o and nodeprocs.o. The objects of Code_gen.

CXX : in Makefile.config g++

DIR : in Makefile. The list of directories where there are files to be compiled.

LDLIB : in Makefile.config. ld -r. It is the command for the linker.

LEX : in Makefile.config. flex

LFLAGS : in Makegile.config. Flags for flex

LIBDIR : in Makefile.config. $MMALPHA/lib.$(OSTYPE). Where the libraries are put.

LPSOLVEDIR : in Makefile.config. Where LP solve is, unused currently.

OMEGALIB : in Makefile.config. Where the OMEGA library is. unused currently.

MMALPHA : environment variable. May also be wired by a definition in Makefile.config or in a system config file.

MMAVERSION : environment variable. May also be wired by a definition in Makefile.config or in a Makefile.ostype.version file. Defines the version number of Mathematica.

MAKEFILES : in Makefile.config. Contains the list of makefiles, i.e. Makefile itself (in the sources directory), Makefile.rules, Makefile.config, and Makefile.$(OSTYPE) located in the directory MakeIncludes.

NEEBINDIR : in Makefile.config. $(BINDIR)/.make. This variable is checked by a rule in Makefile.rules.

NEEDLIBDIR : in Makefile.config. $(LIBDIR)/.make. Use?

NEEDOBJDIR : in Makefile.config. $(OBJDIR)/.make. Use?

OBJDIR : in Makefile.config. Obj.$(OSTYPE). Defines where the object files will be put. This is relative to the directory of the application.

OSTYPE : environment variable

POLYDIR : in Makefile.config. $(SOURCES)/Polylib. Where the source files of the polylib are.

POLYOBJS : in Makefile.config. The list of object files of the polylib. These files are all in $(OBJDIR). I changed this line in July 2007, I do not know why... Maybe to add object files that were needed...

POLYXOBJ : in Makefile.config. Another set of object files

POLYMSG : in Makefile.config. $(OBJDIR)/errormsg.o Defines an object file where the error messages are defines for Polylib.

PRETTYOBJS

PRETTYDIR : in Makefile.config. $(SOURCES)/Pretty. Where Pretty is.

PRETTYOBJS : in Makefile.config. ($PRETTYDIR$)/(OBJDIR)//itemprocs.o and writeitem.o. The objects of Pretty.

SHELL : in Makefile.config. /bin/sh

SOURCES : in Makefile.config. $(MMALPHA)/sources. The source files of MMAlpha.

STRIP : in Makegile.config. strips. Meaning?

YACC : in Makefile.config. bison

YFLAGS in Makefile.config. Flags of Yacc

9.5.2 ReadAlpha

First line includes Makefile.config

CPPFLAGS = -I$(POLYDIR) -g (flags for cc)

LOADLIBES = empt

DUMMY = list of files to be destroyed when cleaning

NAME = the name of the program, read_alpha

OBJS = the list of object files, $(OBJDIR)/read_alpha.o, node.o, $(POLYOBJS)

LIBS = libraries to link again

Then we have a command of the form $(NAME) : $(BINDIR)/$(NAME) @# bla
Then special rules or dependencies for .o files Seems that the object files and the temporary binary are in Obj.darwin.

9.5.3 WriteAlpha

First line includes Makefile.config

CPPFLAGS = -I$(POLYDIR)

LDFLAGS = -L$(LIBDIR) # -mwindows

LOADLIBES = empty

DUMMY = list of files to be destroyed when cleaning

NAME = the name of the program, read_alpha

OBJS = the list of object files, $(OBJDIR)/yacc.o, nodeprocs.o, node2item.o, $(POLYOBJS), $(POLYMSG) $(PRETTYOBJS)

LIBS = libraries to link again, i.e, $(LIBDIR)/libpoly.a, libpretty.a and libpolymsg.a

Then we have a command of the form $(NAME) : $(BINDIR)/$(NAME) @# bla
Then special rules or dependencies for .o files Seems that the object files and the temporary binary are in Obj.darwin.

9.6 The files `sources/MakeIncludes/Makefiles.OSTYPE.Number`

There is one such file for each OS and each version of Mathematica which is supported. Currently:

- For `darwin`: versions 5.2 and 6.0
- For `linux`: versions 3.0.1 and 4.2.

They define: `MATHLINK`, `MATHPREP`, `MATHINCLUDE`, `MATHLIB`, `MATHLOADLIB`, `MATHEXTRALIBS`, `MATHEXTRAOBJS`.

The `linux` version is set for IRISA and should be checked.

Notes: the MMAVERSION variable is currently wired in the file `makefile.linux` as 4.2, thus making a call to `makefile.linux.4.2`. There exists another such file, for version 3.0.1, referring to the location of Mathematica at Irisa, but I think that it is obsolete.

9.7 The `MakeIncludes/Makefile.rules` file

This file is called somewhere (I do not remember where) and sets common dependence rules. There are rules for:

- `BINDIR`, `OBJDIR` (binaries, objects), `NEEDBINDIR` (to create BINDIR, create the directory etc.) `NEEDLIBDIR`, `NEEDOBJDIR`, `LIBDIR` (for each library).

10 Organization for `Domlib`

The `Domlib/Makefile` calls another `Makefile.OSTYPE`, except for cygwin.

For each OS, there is a `Makefile.OSTYPE`. This Makefile calls first `MakeIncludes/Makefile.config` (see 9.1).

It calls then `MakeIncludes/Makefile.OSTYPE.MMAVERSION` (see 9.6) (check that this file is not already called by the config file). It defines `POLYLIB-LIB`, `DBG`, `EXTRA_CFLAGS`. It contains dependence rules for `BINDIR`, `OBJDIR`. Seems that there is no call to Makefile.rules.

Notes: the Mathlink stuff is actually copied in the subdirectory $MMALPHA/sources/Mathlink.

This directory conaints a `MATHLINK` directory where all the stuff needed for the Mathematica interface is put. The `Obj.OSTYPE` directory is where the object files will be put.

11 `ReadAlpha`

First line includes Makefile.config

`CPPFLAGS` = -I$(POLYDIR) -g (flags for cc)

LOADLIBES = empty

DUMMY = list of files to be destroyed when cleaning

NAME = the name of the program, read_alpha

OBJS = the list of object files, $(OBJDIR)/read_alpha.o, node.o, $(POLYOBJS)

LIBS = libraries to link again

These definitions are followed by special dependence rules, and also include the general rules in MakeIncludes/Makefile.rules. Finally, there is a file `Makefile.checkvars` which contains the rules for checking variables. All variables are not checked.

11.0.1 WriteAlpha

First line includes Makefile.config

`CPPFLAGS` = -I$(POLYDIR)

`LDFLAGS` = -L$(LIBDIR) # -mwindows

`LOADLIBES` = empty

`DUMMY` = list of files to be destroyed when cleaning

`NAME` = the name of the program, read_alpha

`OBJS` = the list of object files, $(OBJDIR)/yacc.o, nodeprocs.o, node2item.o, $(POLYOBJS), $(POLYMSG) $(PRETTYOBJS)

`LIBS` = libraries to link again, i.e, $(LIBDIR)/libpoly.a, libpretty.a and libpolymsg.a

These definitions are followed by special dependence rules, and also include the general rules in MakeIncludes/Makefile.rules. Finally, there is a file `Makefile.checkvars` which contains the rules for checking variables. All variables are not checked.

11.0.2 WriteC

First line includes Makefile.config

`CPPFLAGS` = -I$(POLYDIR)

`LDFLAGS` = -L$(LIBDIR) # -mwindows

`LOADLIBES` = empty

`DUMMY` = list of files to be destroyed when cleaning

`NAME` = the name of the program, read_alpha

`OBJS` = the list of object files, $(OBJDIR)/yacc.o, nodeprocs.o, node2item.o, $(POLYOBJS), $(POLYMSG) $(PRETTYOBJS)

`LIBS` = libraries to link again, i.e, $(LIBDIR)/libpoly.a, libpretty.a and libpolymsg.a

These definitions are followed by special dependence rules, and also include the general rules in MakeIncludes/Makefile.rules. Finally, there is a file `Makefile.checkvars` which contains the rules for checking variables. All variables are not checked.

12 List of variables, and where they are defined

In Makefile.config, there is an inclusion of the definitions in MakeIncludes/Makefile.$(OSTYPE) which are peculiar to each OS type.

`BINDIR` : in Makefile.config. $(MMALPHA)/bin.$(OSTYPE). Where the binaries will be put.

`CC` : in Makefile.config. I guess it is the compiler...

`CFLAGS` : in Makefile.config. -O3 -g -I$(POLYINCLUDE) $(PRECISIONFLAG). I guess that these are the flags of the compiler.

`CODEGENDIR` : in Makefile.config. $(SOURCES)/Code_Gen. Where Code_Gen is

`CODEGENOBJS` : in Makefile.config. $(OBJDIR)/gen.o and nodeprocs.o. The objects of Code_gen.

`CXX` : in Makefile.config g++

`DEFINES` : in Makefile.OSTYPE.

`DIR` : in Makefile. The list of directories where there are files to be compiled.

`LDLIB` : in Makefile.config. ld -r. It is the command for the linker.

`LEX` : in Makefile.config. flex

`LFLAGS` : in Makegile.config. Flags for flex

`LDFLAGS` : in Makegile.OSTYPE. Flags for the loader.

`LIBDIR` : in Makefile.config. $MMALPHA/lib.$(OSTYPE). Where the libraries are put.

`LPSOLVEDIR` : in Makefile.config. Where LP solve is, unused currently.

`OMEGALIB` : in Makefile.config. Where the OMEGA library is. unused currently.

`MMALPHA` : environment variable. May also be wired by a definition in Makefile.config or in a system config file.

`MMAVERSION` : environment variable. May also be wired by a definition in Makefile.config or in a Makefile.ostype.version file. Defines the version number of Mathematica.

`MAKEFILES` : in Makefile.config. Contains the list of makefiles, i.e. Makefile itself (in the sources directory), Makefile.rules, Makefile.config, and Makefile.$(OSTYPE) located in the directory MakeIncludes.

`NEEBINDIR` : in Makefile.config. $(BINDIR)/.make. This variable is checked by a rule in Makefile.rules.

`NEEDLIBDIR` : in Makefile.config. $(LIBDIR)/.make. Use?

`NEEDOBJDIR` : in Makefile.config. $(OBJDIR)/.make. Use?

`OBJDIR` : in Makefile.config. Obj.$(OSTYPE). Defines where the object files will be put. This is relative to the directory of the application.

`OMEGADIR` : in Makefile.linux. Directory of the OMEGA software, not used currently.

`OSTYPE` : environment variable

`POLYDIR` : in Makefile.config. $(SOURCES)/Polylib. Where the source files of the polylib are.

`POLYOBJS` : in Makefile.config. The list of object files of the polylib. These files are all in $(OBJDIR). I changed this line in July 2007, I do not know why... Maybe to add object files that were needed...

`POLYXOBJ` : in Makefile.config. Another set of object files

`POLYMSG` : in Makefile.config. $(OBJDIR)/errormsg.o Defines an object file where the error messages are defines for Polylib.

`PRECISIONFLAGS:` in Makefile.linux. Check this.

`PRETTYDIR` : in Makefile.config. $(SOURCES)/Pretty. Where Pretty is.

`PRETTYOBJS` : in Makefile.config. $(PRETTYDIR)$/(OBJDIR)//itemprocs.o and writeitem.o. The objects of Pretty.

`SHELL` : in Makefile.config. /bin/sh

`SOURCES` : in Makefile.config. $(MMALPHA)/sources. The source files of MMAlpha.

`STRIP` : in Makegile.config. strips. Meaning?

`YACC` : in Makefile.config. bison

`YFLAGS` : in Makefile.config. Flags of Yacc

`MATHLINK` : in MakeIncludes/Makefile.OSTYPE.number.

`MATHPREP` : in MakeIncludes/Makefile.OSTYPE.number.

`MATHINCLUDE` : in MakeIncludes/Makefile.OSTYPE.number.

`MATHLIB` : in MakeIncludes/Makefile.OSTYPE.number.

`MATHLOADLIB` : in MakeIncludes/Makefile.OSTYPE.number.

`MATHEXTRALIBS` : in MakeIncludes/Makefile.OSTYPE.number.

`MATHEXTRAOBJS` : in MakeIncludes/Makefile.OSTYPE.number.

`EXTRA_CFLAGS` : in Domlib/Makefiles.darwin.

`CPPFLAGS` : in ReadAlpha, WriteAlpha, WriteC

`LOADLIBES` : in ReadAlpha, WriteAlpha, WriteC

`DUMMY` : in ReadAlpha, WriteAlpha, WriteC

`NAME` : in ReadAlpha, WriteAlpha, WriteC

`OBJS` : in ReadAlpha, WriteAlpha, WriteC

`LIBS` : in ReadAlpha, WriteAlpha, WriteC

A Installing POLYLIB

Look in MMALPHA/doc/Polylib/note.pdf for more details. I describe here the full procedure, but in the current distribution, POLYLIB version 5.22.3 is already present in the distribution: you may then skip steps 1 and 2.

1. Get polylib from Strasburg site (http://icps.u-strasbg.fr/polylib/)
2. Copy it in `$MMALPHA/sources`
3. Gunzip it and untar it (double click, or gunzip)
4. The name of this polylib should be something like `polylib-5.22.3`. Go in this directory (in a shell).
5. Run the following commands:

   ```
   ./configure --prefix="$MMALPHA/sources/Polylib"
   make
   make install
   ```

6. Run `make test` to check that everything is OK

During step 5, the value of the environment variable `$MMALPHA` should be the path of your Mathematica installation.

When POLYLIBis installed, make sure that the value of `POLYDIR` in file `Makefile.config` is that of the path to the POLYLIB (as the source files of POLYLIB are needed to compile the C programs of Mathematica), and that

B Introduction

This document explains how the DOMLIB library is installed, in this directory. It was written while porting the DOMLIB on Mac OS X.

C To be checked

Is your Polylib up-to-date? Look in sources, where you must find both Polylib and Poly (where the files produced by the make command of Polylib are).

D Content of the directory

The directory contains the source, `domlib.c`, a make file, `Makefile`, the MATHLINK template file `domlib.tm`. The make creates an object file `domlib.o`, a binary file `domlib`. On the fly, it translates the template file into a source file `domlibtm.c`.

The file `Makefile` calls another make file, located in:

```
$MMALPHA/sources/MakeIncludes/Makefile.$OSTYPE
```

In this make file, the location of various directories or programs are set. In particular, that of MLINKDIR.

To run properly, the Polylib must have been compiled previously. See documentation on Polylib, but in a few words, just go in PolyLib and compile it by specifying that the directory is Poly. This location is defined in Makefile.darwin, and it may be changed. The commands to be done, once you have copied the gziped filed obtained from strasburg are:

```
./configure --prefix="$MMALPHA/sources/Poly"
make all
make install
```

E The rational of the installation

To be compiled, DOMLIB needs a MATHLINK preprocessor called MPREP, and the POLYLIB library. MATHLINKcompiler additions are also needed.

On this machine, Mathematica is located in[5]

```
/Applications/Mathematica 5.2.app/
```

The variable `MLINKDIR` contains the location

```
/Applications/Mathematica 5.0.app/AddOns/Mathlink/DeveloperKit/MacOSX-x86
```

[5]This depends on the current version of Mathematica that you have.

On other systems, this location may change. If you do not have an Intel Processor, you may have to switch `MLINKDIR` to either

`/Applications/Mathematica 5.0.app/AddOns/Mathlink/DeveloperKit/Darwin`

or

`/Applications/Mathematica 5.0.app/AddOns/Mathlink/DeveloperKit/Darwin-PowerPC64`

In the `Makefile` , variables contain the system name, the place of the POLYLIB header files, the place of the POLYLIB library. The compiled additions are then located. Variable `POLYLIB_INCLUDE` contains the directory where the include files of the POLYLIB are. Similarly variable `POLYLIB_LIB` contains the directory where the libraries of the POLYLIB are.

Variable `CADDSDIR` gives the place where the compiler additions of Mathematica are.

Variable `EXTRA_CFLAGS` is set to

`-DLINEAR_VALUE_IS_LONGLONG -DUNIX`

Variable `INCDIR` is set to CADDSDIR. Variable `INCPOLYDIR` is set to POLYLIB_INCLUDE. Same for `LIBDIR` and `LIBPOLYLIB`.

Variable `MPREP` is set to where `mprep`is.

The `domlibtm.c` file is obtained by running MPREP (provided by Mathematica) on `domlib.tm`.

The `domlib` binary file is obtained by first running `gcc` on both `domlib.c` and `domlibtm.c` then linking the object files with the appropriate POLYLIB library. This library is itself installed *separately*.[6] In the current version, we used the 64 bit version, but other versions are available.

The form of the gcc command is

```
gcc -DLINEAR_VALUE_IS_LONGLONG -DUNIX domlibtm.o domlib.o
  -L${LIBDIR} -L${CADDSDIR}
  -L${POLYLIB} -lpolylib64 -lML -o $@ -v
```

F Running the DOMLIB from Mathematica

To run this library from Mathematica, the following conditions must be satisfied.

1. The DOMLIB binary file must be located in a directory whose address appears on the `$Path` variable of Mathematica.

2. To load the library, run

   ```
   Install[ "domlib" ]
   ```

   in Mathematica. The list of available functions in the library are obtained by evaluating

   ```
   ?Dom*
   ```

In fact, this library is usually run within the MMALPHA software, using the `Domlib` package.

[6] See companion POLYLIB documentation.

G Perspectives

A better way to organize the library would be to make sure that it is portable accross platforms. To do so, we plan to build a version using the `autoconf` and `automake` software, and more generally, the `gnu` building tools.

H Additional remarks

In this directory, the installation is done using a make file. To compile for Mac OS X, the UNIX macro should be defined, and it is by means of the EXTRA_CFLAGS option -DUNIX.

I Mathematica at Irisa

At Irisa, Mathematica is located in directory

`/soft/Mathematica`

Several versions are available. The most recent one is `V4.0.2`. It is located in

`/soft/Mathematica/V4.0.2`

Three version are available : for `Linux`, `SGI` and `Solaris`. Inside the directory, there are subdirectories `CompilerAdditions` where one can find the `mprep` pre-processor.

The transit.irisa.fr machine is a Linux machine. I could not compile on this architecture, for unknown reasons. (mprep does not work).

The europe machine is a solaris machine. I was able to compile. I had to use in Makefile.solaris the same as in Linux. In particular, the MATHEXTRALIB has to be -lnsl -l socket I had to add the `$Path` definition, which is not standard (init file?).

J The Make file for the MMAlpha distribution

In sources, one find Makefile. This file defines LDLIB as ld -r, CC as gcc, CFLAGS as -03 -g -I$(POLYINCLUDE) $(PRECISIONFLAG). It defines some variables for Yacc and Lex.

It includes the makefile.system which is in the same directory MakeIncludes, here Makefile.darwin. In Makefile.darwin, LD is set to gcc, PRECISIONFLAGS, LDFLAGS, MATHLINK, MATHPREP, MATHINCLUDE, MATHLIB, MATHLOADLIB (-lML -lm) and MATHEXTRALIBS is set to empty.

It then defines MAKEFILES, BINDIR, LIBDIR, OBJDIR, NEEDBINDIR, NEEDLIBDIR, NEEDOBJDIR, SOURCES,

For all binary files in DIR, goes in directory, then runs make.

In the Domlib directory, one finds the Makefile.darwin file. It first calls the Makefile.config in the directory MakeIncludes.

K A new version of DOMLIB (May 6, 2008)

It appears that the MathLink functions were modified between Mathematica version 4 and version 5. I thus had to modify Domlib accordingly. The current version of Domlib, compiled with the tools fo version 6, is correct for version 5 and 6. I did not try yet to recompile it for version 5, to see it there are some differences. Actually, I guess that there are such differences.

The Mathlink documentation of version 6.0 is quite useful. Actually, the main differences is in the functions `MLGetFunction` and `MLCheckFunction`, which do not support the same arguments... Too bad !

I added 3 functions to DOMLIB: `mirrorDomain`, `mirrorMatrix` and `mirrorZDomain`. These function may be useful to see if there is something wrong.

The old version of domlib is kept in the directory `sources/Domlib-Version4`. The source files are those which were compiled for Powermac on version 5.

L Compiling DOMLIB

In the ZDomlib file, one finds the `domlib.c` file, which is the source file of DOMLIB, the `domlib.tm` file which is the patterns of the DOMLIB functions for `mathlink`, and finally, several make files.

I guess that compiling DOMLIB is done by simply typing `make` in the ZDomlib directory. But the best approach is to run the make file that is contained in `$MMALPHA/sources`.

A `README` file gives some explanations regarding DOMLIB. Beware that only the first few lines seem to be uptodate.

L.0.3 The `Makefile` file

The file `Makefile` contains a switch to make files that depend on the OS type (this one should be, I guess, in the environment variable `$OSTYPE`). For example, for `cygwin`, the corresponding make file is `Makefile.Visual`.

L.0.4 The `Makefile.Visual` file

The first thing that this make file does is to include the configuration make file, the name of which is `Makefile.config` and which is located in the `$MMALPHAMakeIncludes` directory.

Therefore, you will have to look at the documentation of this make file given in **??** if you want to understand everyhing.

In addition, the configuration make file calls other make files situated in the same directory...

Coming back to `Makefile.Visual`, it defines the following variables:

- `CC`: the compiler, here `cl`;
- `LD`: the linker, `link`;

- `POLYLIB`: where the POLYLIB is (it seems to me that this variable is already set);
- `CFLAGS`: redefines the compilation flags;
- `LFLAGS`: refedines the link flags;
- `BIN`: gives the name of the binary file for domlib;
- `OBJ`: where are the object codes. Here, they are in a directory `Obj.cygwin` located in the same directory.
- `DEPOBJ`: a list of object files which are going to be generated by the compiling.

Then one finds some rules to create DOMLIB. The `all` rule is for creating binary files. I do not understand totally the other rules...

One particular rule allows the `mlink.c` file to be created using MMALPHA's tools from `domlib.tm`.

L.0.5 The `Makefile.linux` file

I am not sure that this file works properly. It defines `CPPFLAGS`, `CLDFLAGS`, `LOADLIBS`, `DUMMY`, `NAME`, `OBJS`, `LIBS`. It uses `POLYDIR`, `MATHINCLUDE`, `LIBDIR`

M Conclusion

Contents