HELSINKI UNIVERSITY OF TECHNOLOGY
Networking Laboratory

Qlib – Traffic Theory Library

Document modified: 18.08.02
Qlib homepages: http://www.tct.hut.fi/tutkimus/com2/Qlib/

Preface

The Qlib program library has resulted from the work conducted over several years jointly by

> **Networking Laboratory of Helsinki University of Technology (HUT)** and **VTT Information Technology, Telecommunications**

The following individuals have contributed to the library:

> *Samuli Aalto, Esa Hyytiä, Jani Lakkakorpi, Ilkka Norros, Anssi Pirhonen, Vesa Timonen and Jorma Virtamo.*

The programs can be used and modified freely.

No claims are made about the correctness of the programs and no liability is taken for any damage caused by the use of the programs in the library.

Reports of bugs in the program can be sent to qlib@tct.hut.fi

Contents

1 The Qlib library

The Qlib library implements a number of functions for solving several queuing problems encountered in the performance analysis of modern broadband communications networks. The library contains three different implementations of the functions:

- Mathematica implementation contains a package of Mathematica functions that can be used in Mathematica program.
- C function library includes standard C implementations of the functions.
- MathLink compatible version includes special support for using the C functions through Mathematica's MathLink functionality from Mathematica program.

Both the C function library and MathLink compatible program are implemented for Unix and Windows alike.

2 Mathematica package Qlib`

Mathematica versions of functions of the traffic theory library are located in a file called *qlib.m*. Note, that these versions are usually much slower than the C-functions implemented in the external MathLink version of qlib.

2.1 Installation and use

Save the file *qlib.m* to your computer e.g. into path `/full_path/`. The package Qlib` is taken in use with following command in Mathematica:

```
Get[“/full_path/qlib.m”]
```

or equivalently

```
<</full_path/qlib.m
```

If the use of the package will be frequent, it should be placed to one of the default paths of Mathematica. Mathematica’s default paths are shown with following command in Mathematica:

```
$Path
```

E.g. if the file *qlib.m* is set to default path

```
C:\Program Files\Wolfram
Research\Mathematica\4.1\AddOns\ExtraPackages\
```

the package Qlib` is taken in use with one of the following commands:

```
Get["Qlib`”]
Get[“qlib.m”]
```

or

```
<<qlib.m
```

The path info is past demanded.

When the package is enabled, a short description of its contents can be seen with a following command:

```
?qlib
```

It shows the functions that the package contains divided into groups. More detailed info about a specific function, its usage and application with command `?FunctionName`, e.g.

```
?MVA
```

See also Chapter 7: Function List.

2.2 Modifying and maintenance

File *qlib.m* is a text file that can be edited with any basic text editor e.g. *emacs* or *notepad*. Functions can be added to the package by appending the functions Mathematica code to the *qlib.m* file.

3 Mathematica packages QlibLnk` and QlibC`

Both the Unix and the Windows specific qlib versions contain Mathematica packages called *Qliblnk`* and *qlibC`*. These both are optional, and their function is to make the user interface in Mathematica as simple as possible. (?)

Note, that these packages necessitate the package *Qlib`* and the external MathLink version of qlib. Also all these files have to be placed into Mathematica's default paths.

Installation of these packages is explained in Sections 4 and 5 for Unix and Windows, respectively.

3.1 Package QlibLnk`

File *qliblnk.m* contains the package *QlibLnk`*. Reading in this package causes the enabling of the external MathLink version of the qlib. In other words, using package *QlibLnk`* is equivalent to take the external MathLink version of qlib into use by using *Install* command.

Usage as follows:

```
<<qliblnk.m
```

or equivalently

```
Get[“qliblnk.m”]
```

3.2 Wrapper QlibC`

File *qlibC.m* contains a wrapper *QlibC`* for packages *Qlib`* and *QlibLnk`*. Reading in *qlibC.m* replaces functions of *Qlib`* with corresponding faster C-versions of the external MathLink version of qlib (if the correspondence exists).

If the package *Qlib`* and/or the external MathLink version of the qlib are not enabled, then use of the *qlibC.m* causes reading in the package *Qlib`* and/or starts the external MathLink version of the qlib.

Usage as follows:

```
<<qlibC.m
```

or equivalently

```
Get[“qlibC.m”]
```

4 Qlib in Unix

The Unix specific qlib version contains following main parts:

- the Mathematica packages *qlib.m*, *qlibC.m* and *qliblnk.m*
- the C function library *qlib.a*
- the external program *qlib* to be used via MathLink in Mathematica
- manual files of functions

Mathematica packages are the same as described in chapters 2 and 3. External MathLink program *qlib* implements the function library *qlib.a* functions and creates an interface for a Mathematica user.

All needed files for installing and using the qlib library in Unix are included to the packed file *qlib.tar.gz*.

4.1 Compiling and installation

Traffic theory library is included in file *qlib.tar.gz*. To extract execute the following commands:

```
gunzip qlib.tar.gz
tar xvf qlib.tar
```

After extracting directory named `Qlib/` contains following structure:

`./link/`	directory the *qlib.a* is linked with Mathlink
`./traffic/`	contains C-functions of traffic library (*qlib.a*)
`./traffic/src/`	source code
`./traffic/doc/`	ps versions of man pages
`./traffic/man/`	man files of the routines
`./traffic/pdf/`	pdf versions of man pages
`./mathematica/`	Mathematica packages
`./example/`	a simple example how to link C code with Mathlink

Change the working directory `./` to `Qlib/` with command `cd Qlib`. The C function library *qlib.a* can be found in the directory `./traffic/src/` and MathLink program *qlib* in the directory `./link/`.

The Installation can be carried through with *Makefiles* and the *make* command. First some configuration has to be done in file *Makefile.rules* (found in the working directory `./`). File *Makefile.rules* contains a following row:

```
QLIBDIR=$(HOME)/.Mathematica/4.1/AddOns/Applications
```

It defines the path into which the Mathematica packages and the external MathLink program *qlib* are placed. This path should be one of the Mathematica's default paths (see Chapter 2.1). Also the used *Gnu make* –program and path for the *Gnu install* –program are defined:

```
MAKE=gmake
INSTALL=/usr/sbin/install
```

Modify those three default settings, if necessary.

To compile and create or update the C function library *qlib.a* and the program *qlib* type following command:

```
make all
```

Now directories `./traffic/src/` and `./link/` should contain the C function library *qlib.a* and the program *qlib*, respectively.

To install all the Mathematica packages and the program *qlib* into the default path defined in the file *Makefile.rules*, type following command:

```
make install
```

It is also possible to compile and create only the *qlib.a* library or the *qlib* program. To compile the *qlib.a* library change the working directory to `./traffic/src/` and type command *make all*. Respectively for the *qlib* program, change the working directory to `./link/` and type command *make all*.

4.2 Use in Mathematica

When the files are placed into default paths, the enabling is carried out as follows: to read in the package *Qlib`* (see Chapter 2) type in Mathematica:

```
<<qlib.m
```

To start the external MathLink program *qlib* type in Mathematica:

```
Install["qlib"]
```

or use the package *QlibLnk`*:

```
<<qliblnk.m
```

To get both the package *Qlib`* and the program *qlib.exe* into use the wrapper *QlibC`* can be used; type in Mathematica:

```
<<qlibC.m
```

Note, that use of packages *Qliblnk`* and *qlibC`* necessitates that the Mathematica packages and the program *qlib* are in default path described above. Use of the packages *Qliblnk`* and *qlibC`* is optional, program *qlib* can be used alone as well, and it is not obligatory to use a default path. E.g. if program *qlib* is placed into arbitrary path `/full_path/`, the program can be started as follows:

- Run the *qlib* in Unix with following command:

```
/full_path/qlib
```

- That should start *qlib* program and print text *Create link:*.
- Give the link name e.g. *12345*.
- In Mathematica, type

```
Install["12345", InstallMode->Connect]
```

Alternatively, in Mathematica type command

```
Install["/full_path/qlib"]
```

After enabling the external MathLink program, the following command in Mathematica prints a list of available functions that *qlib* offers:

```
?Lnk*
```

All function names in *qlib* start with *Lnk*-prefix. More detailed info about a specific function, its usage and application with command `?FunctionName`, e.g.

```
?LnkQmd1
```

See also Chapter 7: Function List.

To uninstall the qlib program, give following command in Mathematica:

```
Uninstall[“qlib”]
```

or if the qlib program was started with package *Qliblnk`* or *qlibC`*, alternatively uninstalling can be carried through with command

```
QlibUninstall[]
```

4.3 Maintenance

Update the information in file *Makefile.rules* in the working directory `./` as described before, if necessary.

- Add wanted C code files and header files (*.h*) into the directory `./traffic/src/` or modify existing ones.
- If adding files (e.g. C code file *func.c* with two header files *func1.h* and *func2.h*), modify the *Makefile* in directory `./traffic/src/` in following way:
  Add the name of *o* –file(s) to end of row *OBJS* (`func.o`) and
  Add the file dependencies to the end of the file (`func.c: func1.h func2.h`)
- Modify files *interface.tm*, *qlib.c* and/or *qlib.h* in directory `./link/`, if necessary.
- Type following command:

```
make all
```

Now directories `./traffic/src/` and `./link/` should contain the C function library *qlib.a* and the program *qlib*, respectively.

To install the updated or rebuilt program *qlib* into the default path defined in the file *Makefile.rules*, type command:

```
make install
```

5 Qlib in PC

The Windows specific qlib version contains following main parts:

- the Mathematica packages *qlib.m*, *qliblnk.m* and *qlibC.m*
- the external program *qlib.exe* to be used via MathLink in Mathematica.
- documentation of the functions
- the C function library *qlib.lib*

Mathematica packages are the same as described in chapters 2 and 3. External MathLink program *qlib.exe* implements the function library *qlib.lib* functions and creates an interface for a Mathematica user.

All needed files for modifying and rebuilding the qlib library in Windows are included to the packed file *qlib.zip*.

5.1 Installation and enabling

Download the files *qlib.m*, *qliblnk.m*, *qlibC.m* and *qlib.exe* (can also be found in the packed file *qlib.zip*). To make use easy all the Mathematica packages (the *m*-files) should be placed into one of Mathematica's default paths (see Chapter 2.1).

Place the external MathLink program *qlib.exe* into same path with the *Mathematica.exe* and *MathKernel.exe*; path is probably something like

```
C:\Program Files\Wolfram Research\Mathematica\4.1\
```

When the files are placed into default paths, the enabling is carried out as follows: to read in the package *Qlib`* (see Chapter 2) type in Mathematica:

```
<<qlib.m
```

To start the external MathLink program *qlib.exe* type in Mathematica:

```
Install["qlib"]
```

or use the package *QlibLnk`*:

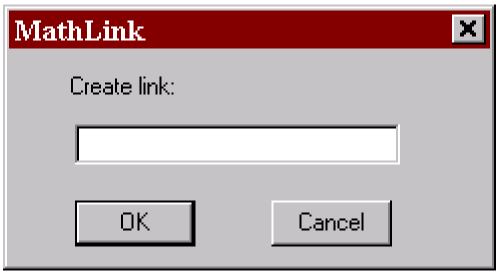
```
<<qliblnk.m
```

To get both the package *Qlib`* and the program *qlib.exe* into use the wrapper *QlibC`* can be used; type in Mathematica:

```
<<qlibC.m
```

Note, that use of packages *Qliblnk`* and *qlibC`* necessitates that the Mathematica packages are in default path and the program *qlib.exe* in the path described above. Use of the packages *Qliblnk`* and *qlibC`* is optional, program *qlib.exe* can be used alone as well, and it is not obligatory to use the path recommended above. E.g. if *qlib.exe* is placed into arbitrary path `C:\full_path\`, the program can be started as follows:

- Run the *qlib.exe*. That should open a box as shown below:



- Type the name for the MathLink program link, e.g. `qliblink` and click *OK*
- In Mathematica, type

```
Install[“qliblink”, InstallMode->Connect]
```

Alternatively, in Mathematica type command

```
Install[“qlib”]
```

Now, if Mathematica does not find the *qlib.exe* in the default path, it opens a dialog box via which the path `C:\full_path\` can be found and *qlib.exe* started.

5.2 Program qlib.exe in Mathematica

After enabling the external MathLink program, the following command in Mathematica prints a list of available functions that *qlib.exe* offers:

```
?Lnk*
```

All function names in *qlib.exe* start with *Lnk*-prefix. More detailed info about a specific function, its usage and application with command `?FunctionName`, e.g.

```
?LnkQmd1
```

See also Chapter 7: Function List.

To uninstall the qlib program, give following command in Mathematica:

```
Uninstall[“qlib”]
```

or if the qlib program was started with package *Qliblnk`* or *qlibC`*, alternatively uninstalling can be carried through with command

```
QlibUninstall[]
```

5.3 Maintenance and modifying

Here the rebuilding and modifying the qlib library is explained as it can be done in *Windows NT* with *Borland C++ 5.0* compiler.

First the packed file *qlib.zip* has to be extracted. The working directory contains following structure after extracting:

`.\dos\`	source code of each function, traffic library *qlib.lib* and the Borland project file for it
`.\dos\pdf\`	pdf versions of manual pages
`.\mldos\`	files for the MathLink program *qlib.exe*, the program *qlib.exe* and the Borland project file for it
`.\mldos\mldev32\`	MathLink files: *mprep.exe*, *mathlink.h* and *lib*-files
`.\mathematica\`	Mathematica packages

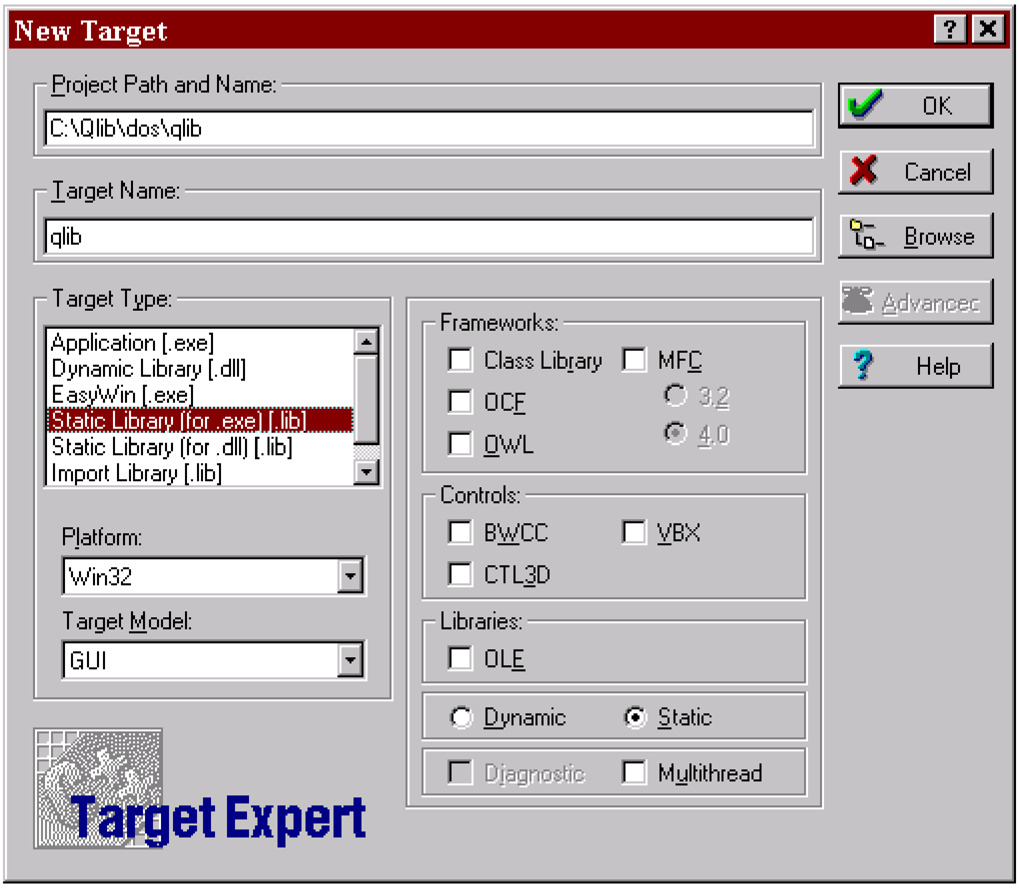
In following descriptions it is assumed that Borland compiler is located into path `C:\bc5\` and qlib files are located into path `C:\Qlib\`.

5.3.1 C-function library qlib.lib

Files for updating and/or rebuilding of the C-function library *qlib.lib* are in the directory `dos\`.

Start from scratch

- Start Borland C++ 5.0
- Choose from the menu *File* ▶ *New* ▶ *Project*
- In the *New Target* dialog box set
  Project Path and Name as `C:\Qlib\dos\qlib`,
  Target Name as `qlib`,
  Target Type as *Static Library (for .exe) [.lib]*
  Platform as *Win32*,
  Target Model as *GUI* and
  Select library type as *Static*.
  No flags should be set in *Frameworks*, *Controls* or *Libraries*.
  Click *OK*.



New Target

Project Path and Name: C:\Qlib\dos\qlib

Target Name: qlib

Target Type:
Application [.exe]
Dynamic Library [.dll]
EasyWin [.exe]
Static Library [for .exe] [.lib]
Static Library [for .dll] [.lib]
Import Library [.lib]

Platform: Win32

Target Model: GUI

Frameworks:
□ Class Library □ MFC
□ OCF ○ 3.2
□ OWL ● 4.0

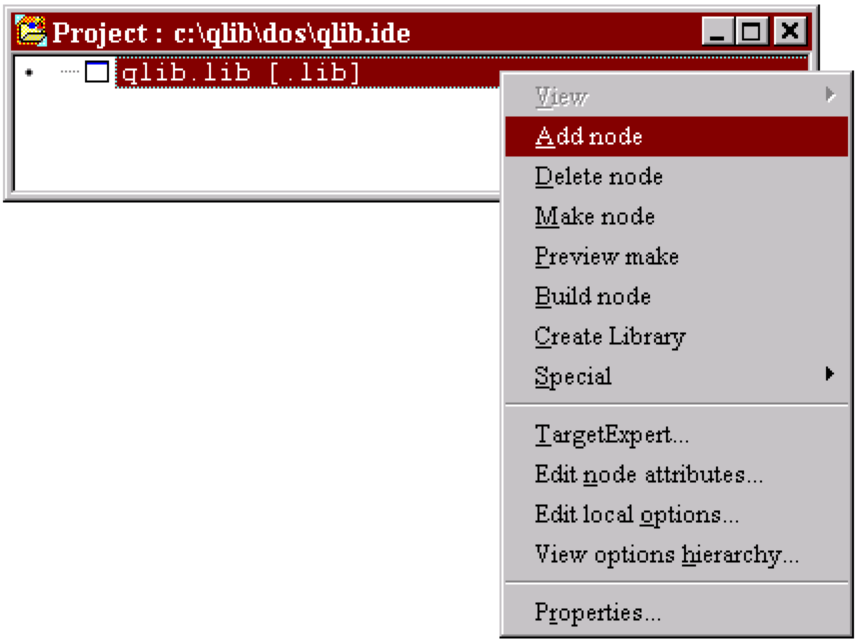
Controls:
□ BWCC □ VBX
□ CTL3D

Libraries:
□ OLE

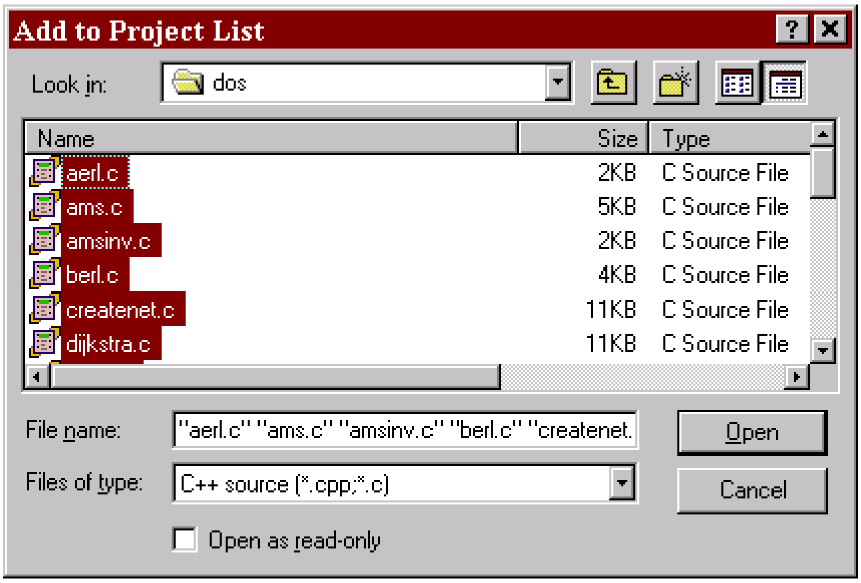
○ Dynamic ● Static

□ Diagnostic □ Multithread

- The project tree contains now one node, *qlib.lib*. Press the right mouse button on it and select *Add node*.



- In the *Add to Project List* dialog bog,
  Set the directory to `C:\Qlib\dos\`,
  Select all the C-files to be included and
  Click *Open*.

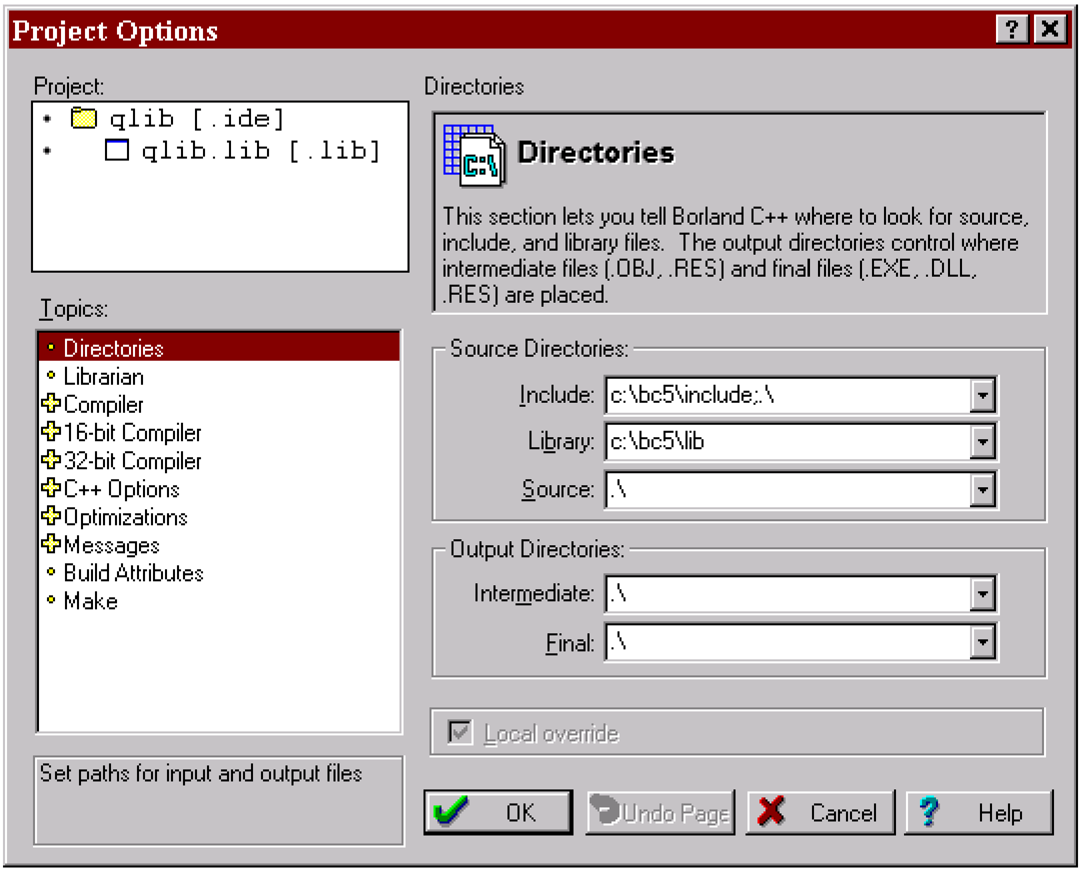


Name	Size	Type
aerl.c	2KB	C Source File
ams.c	5KB	C Source File
amsinv.c	2KB	C Source File
berl.c	4KB	C Source File
createnet.c	11KB	C Source File
dijkstra.c	11KB	C Source File

File name: "aerl.c" "ams.c" "amsinv.c" "berl.c" "createnet.
Files of type: C++ source (*.cpp;*.c)
□ Open as read-only

- On the project tree, press the right mouse button on node *qlib.lib* and select *Edit local options*.

- In the *Project Options* dialog bog,
  Set *Include* directories to `C:\bc5\include` and `.\` (separaded with ';'.
  Directory `.\` is the current working directory, here `C:\Qlib\dos\`)
  Set *Library* directory to `C:\bc5\lib`
  Set *Source* directory to `.\`
  Set *Intermediate* directory to `.\`
  Set *Final* directory to `.\` and
  Click *OK*.



Project Options

Project:

- qlib [.ide]
  - qlib.lib [.lib]

Directories

Directories

This section lets you tell Borland C++ where to look for source, include, and library files. The output directories control where intermediate files (.OBJ, .RES) and final files (.EXE, .DLL, .RES) are placed.

Source Directories:

Include: c:\bc5\include;.\

Library: c:\bc5\lib

Source: .\

Output Directories:

Intermediate: .\

Final: .\

☑ Local override

Set paths for input and output files

- Choose from the menu *Project* ▶ *Build all*
- Choose from the menu *Project* ▶ *Close project*

Now directory `C:\Qlib\dos\` should contain Borland project file *qlib.ide*, precompiled *obj*-files and the C-function library *qlib.lib*.

Modifying, adding and rebuilding:

- Add wanted C code files and header files (*.h*) into the directory `dos\` or modify existing ones.
- Start Borland C++ 5.0
- Choose from the menu *Project* ▶ *Open project*
- In the *Open Project File* dialog bog,
  Set the directory to `C:\Qlib\dos\`,
  Select the project *qlib.ide* and
  Click *Open*.
- If adding new files to the library *qlib.lib*, press the right mouse button on the project tree on node *qlib.lib* and select *Add node* and in the *Add to Project List* dialog bog,
  Set the directory to `C:\Qlib\dos\`,
  Select all the C-files to be added and
  Click *Open*.
- Choose from the menu *Project* ▶ *Build all*
- Choose from the menu *Project* ▶ *Close project*

Now directory `C:\Qlib\dos\` should contain Borland project file *qlib.ide*, precompiled *obj*-files and the updated C-function library *qlib.lib*.

5.3.2 MathLink program qlib.exe

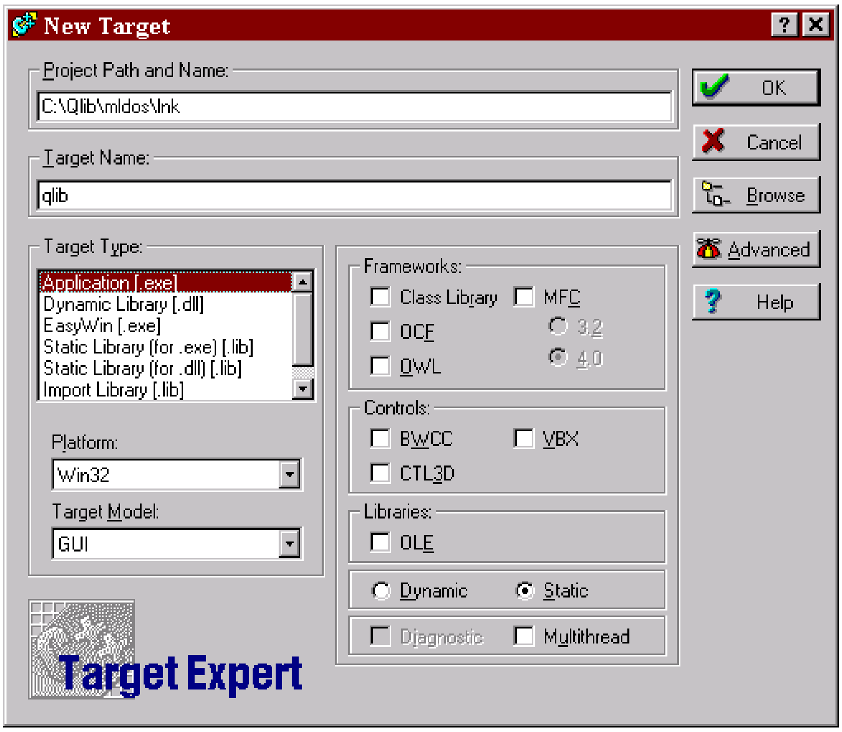
Files for updating and/or rebuilding of the program *qlib.exe* are in the directories `dos\` and `mldos\`.

Start from scratch

- Start *MSDos Command Prompt*
- Change the working directory to `mldos\` and preprocess file *interface.tm* into *interfacetm.c* with the *mprep* program by typing the following command at the *Command Prompt*:

```
.\mldev32\bin\mprep interface.tm -o interfacetm.c
```

- Start Borland C++ 5.0
- Choose from the menu *File* ► *New* ► *Project*
- In the *New Target* dialog box set
  Project path and Name as `C:\Qlib\mldos\lnk`,
  Target Name as `qlib`,
  Target Type as *Application [.exe]*
  Platform as *Win32*,
  Target Model as *GUI* and
  Select library type as *Static*.
  No flags should be set in *Frameworks*, *Controls* or *Libraries*.
  Click *Advanced* and in the *Advanced Options* dialog box
  Set *No Source Node* and
  No flags set for *.rc* or *.def*.
  Click *OK* and click *OK* again.



New Target

Project Path and Name: C:\Qlib\mldos\lnk

Target Name: qlib

Target Type:
Application [.exe] (selected)
Dynamic Library [.dll]
EasyWin [.exe]
Static Library (for .exe) [.lib]
Static Library (for .dll) [.lib]
Import Library [.lib]

Platform: Win32

Target Model: GUI

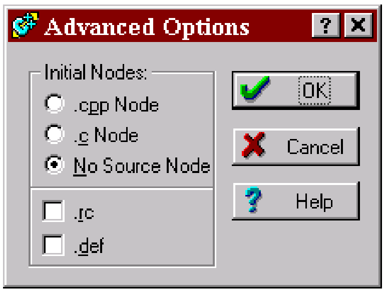
Frameworks: □ Class Library □ MFC □ OCF ○ 3.2 □ OWL ● 4.0

Controls: □ BWCC □ VBX □ CTL3D

Libraries: □ OLE

○ Dynamic ● Static

□ Diagnostic □ Multithread

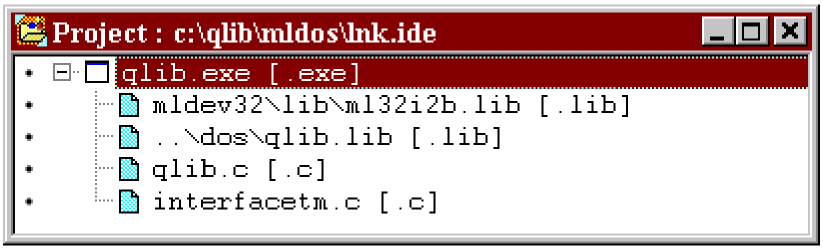


Advanced Options

Initial Nodes:
○ .cpp Node
○ .c Node
● No Source Node

□ .rc
□ .def

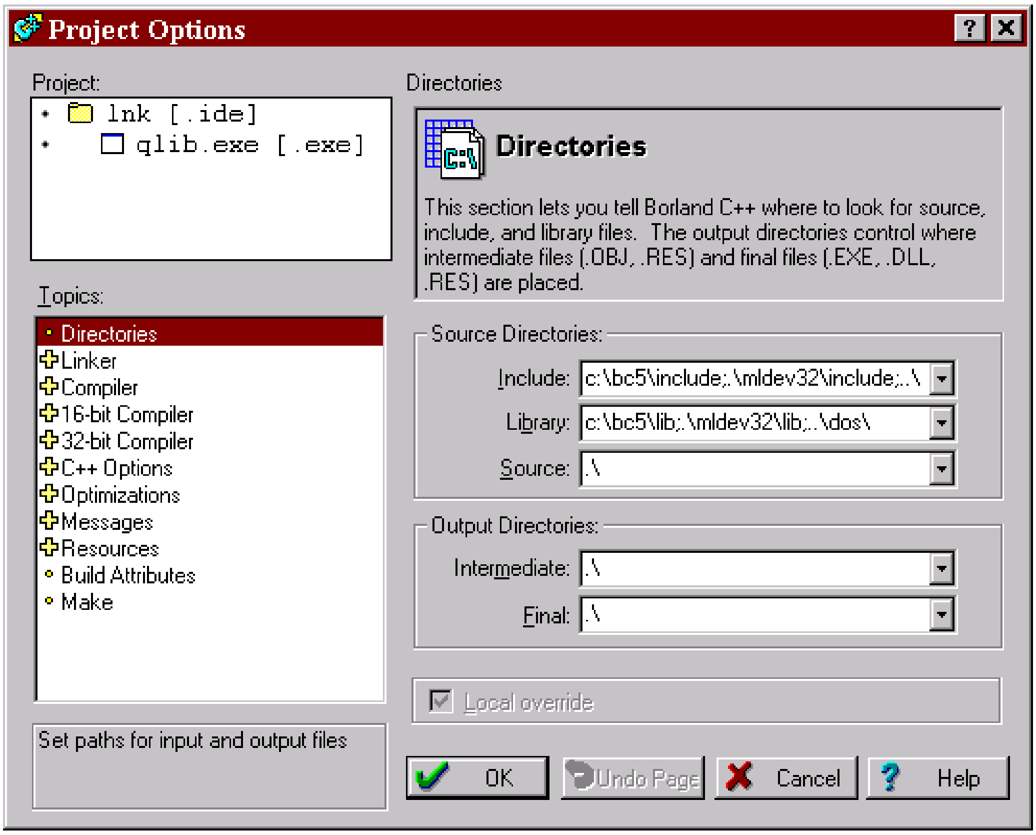
- The project tree contains now one node, *qlib.exe*. Press the right mouse button on it and select *Add node*.
- In the *Add to Project List* dialog bog,
  Set the directory to `C:\Qlib\mldos\`,
  Select files *qlib.c* and *interfacetm.c* to be included and
  Click *Open*.
- Repeat the previos procedure and add files *qlib.lib* from path `C:\Qlib\dos\` and file *ml32i2b.lib* from path `C:\Qlib\mldos\mldev32\lib\`.



Project : c:\qlib\mldos\lnk.ide

```
qlib.exe [.exe]
    mldev32\lib\ml32i2b.lib [.lib]
    ..\dos\qlib.lib [.lib]
    qlib.c [.c]
    interfacetm.c [.c]
```

- On the project tree, press the right mouse button on node *qlib.exe* and select *Edit local options*.
- In the *Project Options* dialog bog,
  Set *Include* directories to `C:\bc5\include`, `.\mldev32\include\` and `..\dos\` (separaded with ';')
  Set *Library* directories to `C:\bc5\lib`, `.\mldev32\lib\` and `..\dos\`
  Set *Source* directory to `.\`
  Set *Intermediate* directory to `.\`
  Set *Final* directory to `.\` and
  Click *OK*.



Project Options

Project:
lnk [.ide]
qlib.exe [.exe]

Directories

Directories

This section lets you tell Borland C++ where to look for source, include, and library files. The output directories control where intermediate files (.OBJ, .RES) and final files (.EXE, .DLL, .RES) are placed.

Source Directories:
Include: c:\bc5\include;.\mldev32\include;..\
Library: c:\bc5\lib;.\mldev32\lib;..\dos\
Source: .\

Output Directories:
Intermediate: .\
Final: .\

☑ Local override

Set paths for input and output files

- Choose from the menu *Project* ▶ *Build all*
- Choose from the menu *Project* ▶ *Close project*

Now directory `C:\Qlib\mldos\` should contain Borland project file *lnk.ide*, precompiled *obj*-files and the MathLink program *qlib.exe*.

Modifying and rebuilding

- Modify files *interface.tm*, *qlib.c* and *qlib.h*, if necessary.
- Start *MSDos Command Prompt*
- Change the working directory to `mldos\` and preprocess file *interface.tm* into *interfacetm.c* with the *mprep* program by typing the following command at the *Command Prompt*:

```
.\mldev32\bin\mprep interface.tm -o interfacetm.c
```

- Start Borland C++ 5.0
- Choose from the menu *Project* ▶ *Open project*
- In the *Open Project File* dialog bog,
  Set the directory to `C:\Qlib\mldos\`,
  Select the project *lnk.ide* and
  Click *Open*.
- Choose from the menu *Project* ▶ *Build all*
- Choose from the menu *Project* ▶ *Close project*

Now directory `C:\Qlib\mldos\` should contain Borland project file *lnk.ide*, precompiled *obj*-files and the updated MathLink program *qlib.exe*.

6 MathLink via TCP/IP –connection

With MahLink it is possible e.g. to run the Mathematica *FrontEnd* and the *Kernel* in different computers. For details, see *The Mathematica Book*, Chapter 2.12.

Following example describes how to run external MathLink program *qlib* in Unix computer e.g. *Tele* (*tele.tct.hut.fi*) and use it through FrontEnd in PC or in another Unix computer.

- In *Tele* type following command:

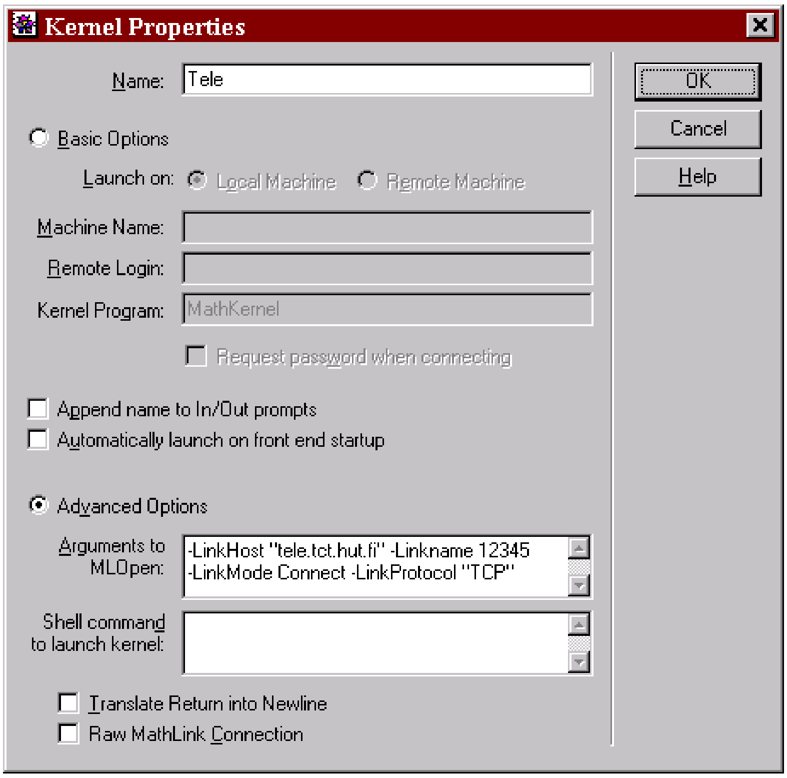
```
qlib –mathlink –linkname 12345 –linkmode listen –linkprotocol
tcp
```

- In Mathematica give the following command:

```
Install[“12345”, LinkMode->Connect, LinkProtocol->“TCP”,
LinkHost->“tele.tct.hut.fi”]
```

Following example describes how to run Kernel in Unix computer *Tele* (*tele.tct.hut.fi*) and FrontEnd in PC or in another Unix computer.

- Choose from FrontEnd’s menu *Kernel* ► *Kernel Configuration Options*
- In the dialog box click *Add*
- In the *Kernel Properties* or *MathLink Connection* dialog bog,
  Set *Name* or *Connection Name* as `Tele`
  Select *Advanced Options*
  Into field *Arguments to MLOpen* type:
  `–LinkHost “tele.tct.hut.fi” –LinkName 12345–LinkMode Connect –LinkProtocol “TCP”`
  No flags should be set
  Click *OK*.



Kernel Properties

Name: Tele

○ Basic Options

Launch on: ● Local Machine ○ Remote Machine

Machine Name:

Remote Login:

Kernel Program: MathKernel

□ Request password when connecting

□ Append name to In/Out prompts

□ Automatically launch on front end startup

● Advanced Options

Arguments to MLOpen: -LinkHost "tele.tct.hut.fi" -Linkname 12345 -LinkMode Connect -LinkProtocol "TCP"

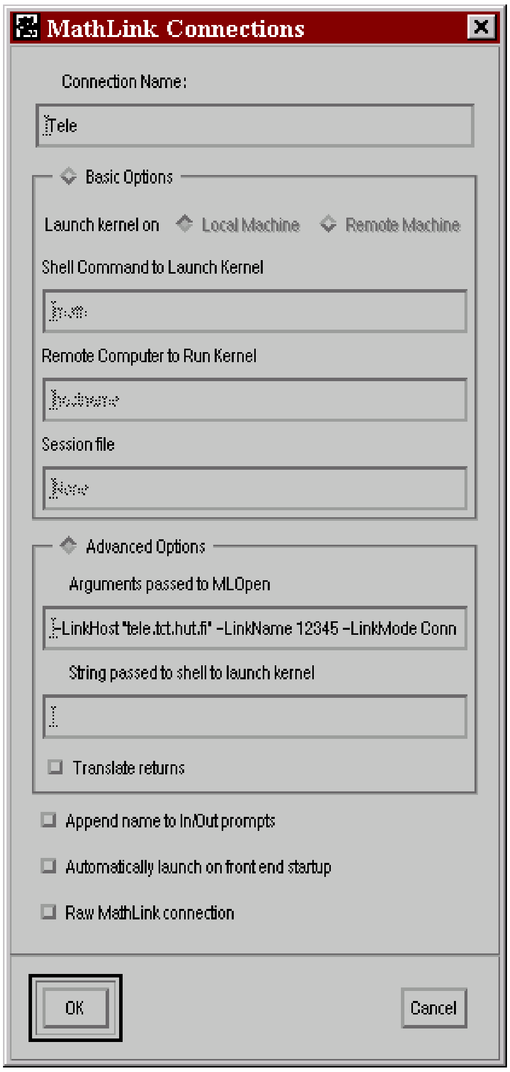
Shell command to launch kernel:

□ Translate Return into Newline

□ Raw MathLink Connection

in Windows ↑

in Unix →



MathLink Connections

Connection Name: Tele

Basic Options

Launch kernel on ◇ Local Machine ◇ Remote Machine

Shell Command to Launch Kernel: math

Remote Computer to Run Kernel: localhost

Session file: None

Advanced Options

Arguments passed to MLOpen: -LinkHost "tele.tct.hut.fi" -LinkName 12345 -LinkMode Conn

String passed to shell to launch kernel:

□ Translate returns

□ Append name to In/Out prompts

□ Automatically launch on front end startup

□ Raw MathLink connection

- Start the Kernel in *Tele* with following command:

```
math -mathlink -linkmode listen -linkprotocol tcp –linkname
12345
```

- Choose from FrontEnd’s menu *Kernel* ▶ *Notebook’s Kernel* ▶ *Tele*
- Choose from FrontEnd’s menu *Kernel* ▶ *Start Kernel* ▶ *Tele*

Note that when a program or a *Kernel* is started in the *listen* –mode, it can be contacted by knowing the link name (in previous examples *12345*). Thus, the link name should be considered as a password.

7 Function list

List of functions that the current Qlib library contains. Information below is from the Qlib homepages: http://www.tct.hut.fi/tutkimus/com2/Qlib/. For most resent information check the site.

In the table below is a list of all the functions implemented in the library and their calling patterns.

Function name and call pattern for Mathlink functions (1)	**Function name for Mathematica functions**	**Short description of the function**
Markov Processes		
-	CtMarkovChain[Matrix] (MMA)	Gives the stationary probabilities of a continuous time Markov chain with the transition rate matrix Q.
-	DtMarkovChain[Matrix] (MMA)	Gives the stationary probabilities of a discrete time Markov chain with the transition probability matrix P.
Loss Systems		
-	MVA[Matrix, Vector, Vector] (MMA)	Algorithm for the Mean Value Analysis of a closed Jackson network. It returns the average queue lengths and the average sojourn times in the queues. The branching ratios are given by the matrix R; the vector mu specifies the service rates of the queues; and K is the number of customers in the network.
LnkBerli[Integer, Real] (2)	Berli[Integer, Real] (MMA)	Erlang loss probability.
LnkBerld[Real, Real] (2)	Berld[Real, Real] (MMA)	Erlang loss probability.
LnkXerl[Real, Real]	Xerl[Real, Real] (MMA)	Inverse Erlang function.
LnkAerl[Real, Real]	Aerl[Real, Real] (MMA)	Inverse Erlang function.
LnkBkaufman[Integer, Integer, Integer, IntegerList, RealList]	-	Erlang blocking probability for multiple traffic classes.
LnkBmitra[Integer, Integer, Real, IntegerList, RealList]	-	Erlang blocking probability for multiple traffic classes.
GAMS (3)	-	Anick-Mitra-Sondhi handling function.
EvAMS (3)	-	Anick-Mitra-Sondhi handling function.
initAMS (3)	-	Anick-Mitra-Sondhi handling function.
freeAMS (3)	-	Anick-Mitra-Sondhi handling function.
M/D/1 etc Queuing Systems		
LnkQmd1[Real, Real]	Qmd1[Real, Real] (MMA)	Virtual waiting time distribution for the M/D/1 queue.
LnkQndd1[Real, Integer, Real]	Qndd1[Real, Integer, Real] (MMA)	Virtual waiting time distribution for the N*D/D/1 queue.

LnkQsdd1[Real, RealList]	Qsdd1[Real, RealList] (MMA)	Virtual waiting time distribution for the $\sum D_i/D/1$ queue.
LnkMg1[Integer, Real, Integer]	Mg1[Integer, Real, Function] (MMA) (4)	Queue length probability function for the M/G/1 queue.
LnkQmxd[Real, Real, RealList]	Qmxd[Real, Real, RealList] (MMA)	Unfinished work tail distribution function for the $M^x/D/1$ queue.
LnkFmd1[Real, Real]	Fmd1[Real, Real] (MMA)	Virtual waiting time distribution for the M/D/1 queue.
LnkIntFmd1[Integer, Real]	IntFmd1[Integer, Real] (MMA)	Virtual waiting time distribution for integral values of the amount of unfinished work in the system.
LnkSumMd1[Real, Real]	SumMd1[Real, Real] (MMA)	Calculates state probabilities of the M/D/1 queue.
LnkRecMd1[Integer, Real]	RecMd1[Integer, Real] (MMA)	Calculates state probabilities of the queue with a recursive algorithm.
LnkFmdn[Real, Real, Integer]	Fmdn[Real, Real, Integer] (MMA)	Virtual waiting time distribution of the M/D/n queue.
LnkIntFmdn[Integer, Real, Integer]	IntFmdn[Integer, Real, Integer] (MMA)	Virtual waiting time distribution for integral values of the amount of unfinished work in the system.
LnkMdn[Real, Real, Integer]	Mdn[Real, Real, Integer] (MMA)	Calculates state probabilities of the M/D/n queue.
LnkFekdn[Real, Real, Integer, Integer]	Fekdn[Real, Real, Integer, Integer] (MMA)	Virtual waiting time distribution for the $E_k/D/n$ queue.
LnkFend1[Real, Integer, Real]	Fend1[Real, Integer, Real]	Virtual waiting time distribution for the $E_n/D/n$ queue.
Network generation etc		
LnkGabrielNet[Integer] (5)	GabrielNet[Integer] (5)	Generates a random network based on Gabriel criterion.
LnkDijkstra[DistanceMatrix, Integer, Integer] (5)	Dijkstra[DistanceMatrix, Integer, Integer] (5)	Finds out the shortest path from the given root node to the given destination node.
-	ShowNetwork[Output_GabrielNet, Output_Dijkstra] (5)	Creates graphs of the network and the route(s).

(1) These are the MathLink function names. The "original" C-functions lack the Lnk- prefix and their call pattern is different. Click on the link of each function for further information.

(2) The C-functions are called *Berl_i* and *Berl_d*.

(3) Anick-Mitra-Sondhi handling functions are not included in the qlib -file and cannot be used via MathLink. No Mathematica -versions of these functions are either available.

(4) Here is an example: first we define function *F*: `F[t_] := If[t < 1, 0, 1]` in Mathematica, and then we use *Mg1*: `Mg1[3, 0.4, F]`.

(5) More specific and detailed descriptions can be found in the qlib.m file or after installing the Mathematica-package typing eg. `?Dijkstra` in the Mathematica.