Subject: Stack tracing
Posted by Zbych on Thu, 24 Jan 2013 12:49:00 GMT
View Forum Message <> Reply to Message

Hi,

Since assertion messages aren't very helpful in hunting for bugs, I decided to add some extra help to them (at least for gcc users). Modified Core/Utils.ccp/AssertFailed displays call stack similar to this one:

Stack trace:
Upp::AddStackTrace(char*, int)
Upp::AssertFailed(char const*, int, char const*)
Upp::AString<Upp::String0>::operator[](int) const
Upp::CCTalk::GetAsciiData(int, int, Upp::String&)
Upp::CCTalk::EquipmentID(int, Upp::String&)
Upp::CCTalk::FindDevs()
Upp::CCTalk::WorkingThread()
Upp::CallbackMethodAction<Upp::CCTalk, void (Upp::CCTalk::*)()>::Execute()
Upp::Callback::Execute() const
Upp::Callback::operator()() const

clone

All you need (beside modified version of Core/Utils.cpp) is '-rdynamic' added to linker options in your application.
Disadvantages? Bigger executable, much longer linking time and of course visible function names.
Debug symbols are not necessary, all function names are kept in .dynsym section.

Without -rdynamic, call stack looks like this one:

Stack trace:
() [0x82bf968]
() [0x82bfc18]
() [0x8053fbe]
() [0x8052948]
() [0x8053c51]
() [0x80522b0]
() [0x8053009]
() [0x8054d26]
() [0x825cb0a]
() [0x8050fd3]
() [0x8241889]
(

(clone

Edit:
New version of Util.cpp is uploaded.

File Attachments

1) Util.cpp, downloaded 378 times

Subject: Re: Stack tracing
Posted by mirek on Thu, 24 Jan 2013 18:32:09 GMT
View Forum Message <> Reply to Message

Nice. I will merge it into the trunk as option...

Mirek

Subject: Re: Stack tracing
Posted by Zbych on Thu, 24 Jan 2013 18:52:36 GMT
View Forum Message <> Reply to Message

OK. I have just spotted that mingw doesn't support backtrace and all conditional blocks should be:

#if defined(PLATFORM_POSIX) && defined(COMPILER_GCC)

Subject: Re: Stack tracing
Posted by Zbych on Fri, 03 May 2013 15:13:44 GMT
View Forum Message <> Reply to Message

Mirek,

When can you add this patch to the trunk?

Subject: Re: Stack tracing
Posted by Zbych on Tue, 18 Mar 2014 10:17:27 GMT
View Forum Message <> Reply to Message

mirek wrote on Thu, 24 January 2013 19:32Nice. I will merge it into the trunk as option...
Mirek

I noticed that stack trace requires flagSTACKTRACE since revision 7000. Maybe you could also add new link option to the core package:
When GCC POSIX STACKTRACE: -rdynamic
It will add function names to a binary file when flag STACKTRACE is present.

Subject: Re: Stack tracing
Posted by mirek on Tue, 18 Mar 2014 10:33:36 GMT
View Forum Message <> Reply to Message

Now thinking about it, we could also activate with "SO"?

Subject: Re: Stack tracing
Posted by Zbych on Wed, 19 Mar 2014 10:19:26 GMT
View Forum Message <> Reply to Message

mirek wrote on Tue, 18 March 2014 11:33we could also activate with "SO"?

Some people say that instead of adding -rdynamic it is simpler to execute:
dlopen(NULL, RTLD_NOW | RTLD_GLOBAL);
to load main binary symbol table, but I never tried it.

Subject: Re: Stack tracing
Posted by mirek on Wed, 19 Mar 2014 18:31:52 GMT
View Forum Message <> Reply to Message

Zbych wrote on Wed, 19 March 2014 06:19mirek wrote on Tue, 18 March 2014 11:33we could also activate with "SO"?

Some people say that instead of adding -rdynamic it is simpler to execute:
dlopen(NULL, RTLD_NOW | RTLD_GLOBAL);
to load main binary symbol table, but I never tried it.

Does not seem to work..

Subject: Re: Stack tracing
Posted by Zbych on Fri, 05 Dec 2014 11:43:44 GMT
View Forum Message <> Reply to Message

Hi Mirek,

Could you add link option "GCC POSIX STACKTRACE: -rdynamic" to the core package?

Subject: Re: Stack tracing
Posted by mirek on Fri, 05 Dec 2014 15:50:13 GMT
View Forum Message <> Reply to Message

Done