Oml MultiOberon/LLVM Quick-Start

Oml is the instance of MultiOberon compiler with LLVM backend. Used prepared library with LLVM 5.0.
Version 0.9 12-Nov-2019

Installation.

On Windows (this color - for Windows):
For BlackBox 1.6
```
win_toinstall.vbs 16 <path-to-blackbox>
```
For BlackBox 1.7
```
win_toinstall.vbs 17 <path-to-blackbox>
```

On Linux (this color – for Linux):
glibc 2.15 or STT_GNU_IFUNC support is needed, tinfo package required
Download and install https://blackbox.obertone.ru/download
```
tclsh lin_toinstall.tcl 17 <path-to-blackbox>
```

How to Start from Black Box

1 Installation

1.1. Preconditions.
Oml uses LLVM 5.0 Services in LLVMT.dll.
2. Oml/Docu/Quick-Start.odc

3 Compile LLVM Services:
```
^Q DevCompiler.CompileThis LlvmC LlvmForAArch64 LlvmForAMDGPU LlvmForARM
LlvmForBPF LlvmForHexagon LlvmForLanai LlvmForMips LlvmForMSP430 LlvmForNVPTX
LlvmForPowerPC LlvmForSparc LlvmForSystemZ LlvmForX86 LlvmForXCore LlvmNative
LlvmRefs
```

3 Compile the following modules:
```
^Q DevCompiler.CompileThis HostApi HostConLog HostTimes OmcCfgfile OmcTarget
OmcCRuntime OmcHooks OmcDialog OmcOPM OmcOPT OmcOPU OmcOPB OmcOPS OmcOPP OmcDump
OmcParams OmcOdcSource OmcTxtSource OmcRuntimeStd OmcDialogStd OmcDialogConsole
OmcCompiler OmcConsole OmlOPG OmlOPL OmlOPF OmlOPC OmlOPV OmlBackEnd OmlCompiler
```

2 Compiling Examples

2.1. Compiling examples for 32-bit:
```
^Q OmlCompiler.CompileThis +HostConLog OmtestHelloWorld: OmtestFormats:
OmtestDateTime: OmtestMkTraps: OmtestHeap:
```
Expected result in ~/Omtest/Clwe/ directory: OmtestHelloWorld.ll OmtestHelloWorld.bc OmtestFormats.ll OmtestFormats.bc OmtestDateTime.ll OmtestDateTime.bc OmtestMkTraps.ll OmtestMkTraps.bc OmtestHeap.ll OmtestHeap.bc

2.2. Compiling examples for 64-bit:
```
^Q OmlCompiler.CompileThis -64 +HostConLog OmtestHelloWorld: OmtestFormats:
OmtestDateTime: OmtestMkTraps: OmtestHeap:
```
Expected result in ~/Omtest/Clwr/ directory: OmtestHelloWorld.ll OmtestHelloWorld.bc OmtestFormats.ll OmtestFormats.bc OmtestDateTime.ll OmtestDateTime.bc OmtestMkTraps.ll OmtestMkTraps.bc OmtestHeap.ll OmtestHeap.bc

3 Self-Compiling Shell

3.1 Self-Compile 32-bit LLVM Services.

```
^Q OmlCompiler.CompileThis LlvmC LlvmForAArch64 LlvmForAMDGPU LlvmForARM
LlvmForBPF LlvmForHexagon LlvmForLanai LlvmForMips LlvmForMSP430 LlvmForNVPTX
LlvmForPowerPC LlvmForSparc LlvmForSystemZ LlvmForX86 LlvmForXCore LlvmNative
LlvmRefs
```

3.2 Self-Compile 32-bit console Oberon Shell.

```
^Q OmlCompiler.CompileThis SystemApi SystemKernel SystemMath SystemOStrings
SystemOLog SystemRunner SystemTimes SystemFiles HostApi HostConLog HostTimes
HostFiles
^Q OmlCompiler.CompileThis -options lb LlvmNative OmcCfgfile OmcTarget
OmcCRuntime OmcDialog OmcHooks OmcTxtSource OmcDialogConsole OmcRuntimeStd
OmcOPM OmcOPT OmcOPB OmcOPU OmcOPS OmcOPP OmcParams OmcConsole OmcDump OmcShell
OmlOPG OmlOPL OmlOPF OmlOPC OmlOPV OmlBackEnd OmlLoader OmlSh:
```

3.3 Self-Compile 64-bit LLVM Services.

```
^Q OmlCompiler.CompileThis -64 LlvmC LlvmForAArch64 LlvmForAMDGPU LlvmForARM
LlvmForBPF LlvmForHexagon LlvmForLanai LlvmForMips LlvmForMSP430 LlvmForNVPTX
LlvmForPowerPC LlvmForSparc LlvmForSystemZ LlvmForX86 LlvmForXCore LlvmNative
LlvmRefs
```

3.4 Self-Compile 64-bit console Oberon Shell.

```
^Q OmlCompiler.CompileThis SystemApi SystemKernel SystemMath SystemOStrings
SystemOLog SystemRunner SystemTimes SystemFiles HostApi HostConLog HostTimes
HostFiles
^Q OmlCompiler.CompileThis -options lb -64 LlvmNative OmcCfgfile OmcTarget
OmcCRuntime OmcDialog OmcHooks OmcTxtSource OmcDialogConsole OmcRuntimeStd
OmcOPM OmcOPT OmcOPB OmcOPU OmcOPS OmcOPP OmcParams OmcConsole OmcDump OmcShell
OmlOPG OmlOPL OmlOPF OmlOPC OmlOPV OmlBackEnd OmlLoader OmlSh:
```

4 Unloading Oml Compiler

```
^Q DevDebug.UnloadThis OmlCompiler OmlBackEnd OmlOPV OmlOPC OmlOPF OmlOPL OmlOPG
OmcCompiler OmcDialogStd OmcRuntimeStd OmcOdcSource OmcParams OmcDump OmcOPP
OmcOPS OmcOPU OmcOPB OmcOPT OmcOPM OmcDialog OmcHooks OmcCRuntime OmcTarget
OmcCfgfile Runner
```

<u>How to Start from Command Line</u>.

1 Installation

1. Preconditions.
Oml uses LLVM 5.0 Services in LLVMT.dll.. Process all the commands below from the Mob-master root dir.

2 Compiling examples

```
Blwe\omlsh co OmtestHelloWorld
Blue/omlsh co OmtestHelloWorld
```
A new symbol file is created first, then OmtestHelloWorld.mod is compiled to 32-bit Omtest/Clwe/HelloWorld.bc. A list of files can be compiled.

```
Blwe\omlsh ru OmtestHelloWorld
Blue/omlsh ru OmtestHelloWorld
```
Run 32-bit OmtestHelloWorld.bc with Oml Shell as dynamically loaded module.

```
Blwe\omlsh ex OmtestHelloWorld
Blue/omlsh ex OmtestHelloWorld
```
Execute means both 32-bit compile and run OmtestHelloWorld.mod with Oml Shell.

```
Blwe\omlsh co OmtestHelloWorld: OmtestFormats: OmtestDateTime: OmtestMkTraps:
OmtestHeap:
Blue/omlsh co OmtestHelloWorld: OmtestFormats: OmtestDateTime: OmtestMkTraps:
OmtestHeap:
```
The command above compiles all the examples listed for 32-bit.

```
Blwr\omlsh co OmtestHelloWorld
Blur/omlsh co OmtestHelloWorld
```
A new symbol file is created first, then OmtestHelloWorld.mod is compiled to 64-bit Omtest/Clwr/HelloWorld.bc. A list of files can be compiled.

```
Blwr\omlsh ru OmtestHelloWorld
Blur/omlsh ru OmtestHelloWorld
```
Run 64-bit OmtestHelloWorld.bc with Oml Shell as dynamically loaded module.

```
Blwr\omlsh ex OmtestHelloWorld
Blur/omlsh ex OmtestHelloWorld
```
Execute means both 64-bit compile and run OmtestHelloWorld.mod with Oml Shell.

```
Blwr\omlsh co +HostConLog OmtestHelloWorld: OmtestFormats: OmtestDateTime:
OmtestMkTraps: OmtestHeap:
Blur/omlsh co +HostConLog OmtestHelloWorld: OmtestFormats: OmtestDateTime:
OmtestMkTraps: OmtestHeap:
```
The command above compiles all the examples listed for 64-bit.

3 Running the examples

3.1.The simplest Hello, World example
```
Blwe\omlsh ru OmtestHelloWorld
Blue/omlsh ru OmtestHelloWorld
```

Logging with char, int and real formats
```
Blwe\omlsh ru OmtestFormats
Blue/omlsh ru OmtestFormats
```

3.2 Shows date, time and delay

```
Blwe\omlsh ru OmtestDateTime
Blue/omlsh ru OmtestDateTime
```

3.3 Traps handling abilities of runtime

Simple Assert

```
Blwe\omlsh ru OmtestMkTraps -trap a
Blue/omlsh ru OmtestMkTraps -trap a
```

Simple Halt

```
Blwe\omlsh ru OmtestMkTraps -trap h
Blue/omlsh ru OmtestMkTraps -trap h
```

Zero divide

```
Blwe\omlsh ru OmtestMkTraps -trap z
Blue/omlsh ru OmtestMkTraps -trap z
```

Nil pointer dereference

```
Blwe\omlsh ru OmtestMkTraps -trap p
Blue/omlsh ru OmtestMkTraps -trap p
```

3.4 Dynamic memory and garbage collector

```
Blwe\omlsh ru OmtestHeap
Blue/omlsh ru OmtestHeap
```

4 Example set executives

In order to compile and link to binary executives the C-development environment is needed. I provide no Visual Studio or MinGW or CMake tools. Please, use external tools or modify scripts. I use the following:

- gcc, ar – for lwe,
- clang – for lwr.

```
lwe_tomake
lue_tomake.sh
```

Makes all the 32-bit executives of example set

```
lwe_toclean
lue_toclean.sh
```

Cleans all the 32-bit executives of example set

```
lwr_tomake
lur_tomake.sh
```

Makes all the 64-bit executives of example set

```
lwr_toclean
lur_toclean.sh
```

Cleans all the 64-bit executives of example set

5 Running executives

5.1.The simplest Hello, World example (64-bit)

```
Omtest\Clwr\OmtestHelloWorld.exe
Omtest/Clur/OmtestHelloWorld.exe
```

Logging with char, int and real formats

```
Omtest\Clwr\OmtestFormats.exe
Omtest/Clur/OmtestFormats.exe
```

5.2 Shows date, time and delay

```
Omtest\Clwr\OmtestDateTime
Omtest/Clur/OmtestDateTime
```

5.3 Traps handling abilities of runtime

Simple Assert

```
Omtest\Clwr\OmtestMkTraps -trap a
Omtest/Clur/OmtestMkTraps -trap a
```

Simple Halt

```
Omtest\Clwr\OmtestMkTraps -trap h
Omtest/Clur/OmtestMkTraps -trap h
```

Zero divide

```
Omtest\Clwr\OmtestMkTraps -trap z
Omtest/Clur/OmtestMkTraps -trap z
```

Nil pointer dereference

```
Omtest\Clwr\OmtestMkTraps -trap p
Omtest/Clur/OmtestMkTraps -trap p
```

5.4 Dynamic memory and garbage collector

```
Omtest\Clwr\OmtestHeap
Omtest/Clur/OmtestHeap
```

6 Making Compiled Shell Binaries

In order to compile and link to binary executives the C-development environment is needed. I provide no Visual Studio or MinGW or CMake tools. Please, use external tools or modify scripts. I use the following:

- gcc, ar – for lwe,
- clang – for lwr.

```
lwe_compiler_tomake
lue_compiler_tomake.sh
```

Makes all the 32-bit executive of omlsh

```
lwe_compiler_toclean
lue_compiler_toclean.sh
```

Cleans all the 32-bit executive of omlsh

```
lwr_compiler_tomake
lur_compiler_tomake.sh
```

Makes all the 64-bit executive of omlsh

```
lwr_compiler_toclean
lur_compiler_toclean.sh
```

Cleans all the 64-bit executive of omlsh

Change log

may 2019 original MultiOberon pre-version 0.8 released
nov 2019 MultiOberon pre-version 0.9 released

Use it and enjoy! - ЎЪsalos y disfrъtalos! - Bonne utilisation - Приятного использования - Powodzenia - Viel SpaЯ

Dmitry V. Dagaev
dvdagaev@yahoo.com