OpenMP* support in Clang/LLVM

Jim Cownie

OpenMP* Support in LLVM: A Brief History

Terminology: Typical OpenMP code generation requires "outlining", the conversion of a inline parallel region (or task) to an outlined function.

2H 2012: Several proposals with late outlining

- All of them involve changes to LLVM IR and thus, modifications of LLVM phases
- None of them received enough community support to make it into the trunk

October 2012: OpenMP in Clang project

- Started by AMD*, continued by Intel
- Early outlining
- OpenMP RTL calls generated in Clang

No Changes to LLVM IR

* Other brands and names are the property of their respective owners.

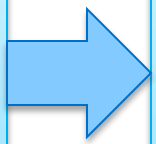
intel Software

Early vs Late Outlining

Parallel regions are "outlined" into separate routines

- To be executed in separate threads
- This can be done either in the front-end or the back-end

```
float a,x,y,z;
#pragma omp parallel for
for (i = 0; i < N; i++) {
  a[i] = x * y * z;
  ... // rest of loop
}
```



```
omp_parallel_for(0, N,
N/omp_get_num_threads(), forb)
...
void forb(int L, int U, R *r) {
  for (i = L; i < U; i++) {
    r->a[i] = r->x * r->y * r->z;
    ... // rest of loop
  }
}
```


* Other brands and names are the property of their respective owners.

Comparison Early vs Late

	Early	Late
LLVM IR unchanged	Yes	No
Common language independent parallel optimisation	No	Yes
Preserves other optimizations (constant propagation, …)	No	Yes
Affects later compilation phases	No	Yes

Late outlining viewed as too intrusive by LLVM architects
Early outlining (in clang frontend) has been implemented

Tiny Example

```
void vzero(float *a, int n)
{
#pragma omp parallel for
    for (int i = 0; i < n; ++i)
        a[i] = 0;
}
```

$ clang -cc1 -ast-dump -fopenmp test.c

```
|-FunctionDecl 0x7fbdc2026930 <test.c:1:1, line:6:1> vzero 'void (float *, int)'
| |-ParmVarDecl 0x7fbdc20267f0 <line:1:12, col:19> a 'float *'
| |-ParmVarDecl 0x7fbdc2026860 <col:22, col:26> n 'int'
| `-CompoundStmt 0x7fbdc2070c18 <line:2:1, line:6:1>
|   `-OMPParallelDirective 0x7fbdc2070be8 <line:3:9, col:25>
|     `-CapturedStmt 0x7fbdc2070b90 <col:9, col:25>
|       |-Capture byref ParmVar 0x7fbdc2026860 'n' 'int'
|       |-Capture byref ParmVar 0x7fbdc20267f0 'a' 'float *'
|       |-DeclRefExpr 0x7fbdc20702a8 <line:4:25> 'int' lvalue ParmVar 0x7fbdc2026860 'n' 'int'
|       `-DeclRefExpr 0x7fbdc20704b8 <line:5:9> 'float *' lvalue ParmVar 0x7fbdc20267f0 'a' 'float *'
```

OpenMP* Runtime

The OpenMP runtime is responsible for executing the compiler-generated code in parallel

The existing GCC implementation uses libgomp, which is licensed under version(s) of the GPL

Clang / LLVM uses UoI / NCSA* Open Source License

- Permissive (aka BSD-style) free software license

Intel's Runtime now Open Source, with LLVM compatible license

- BSD
- Patent grant

* Other brands and names are the property of their respective owners.

intel Software

Additional Advantages of Open Source runtime

Ability for tools providers to instrument the runtime

- E.g. HPCToolkit (John will say more on this next)

Prototyping of new language features

Port to other architectures

Detailed instrumentation to support hardware simulation

Current Status

OpenMP* 3.1 in Clang patch is available

- Hosted on clang-omp.github.com
- Targets Intel OpenMP runtime ABI
- Runs SPEC OMP 2012, passes internal Intel tests

Upstreaming to Clang trunk is under way

- Approved by Chris Lattner and other Clang architects
- Code reviewers throughput is limiting factor

Intel OpenMP RTL available

- Hosted at openmp.llvm.org and openmprtl.org
- LLVM compatible licenses

Other Issues

Intel libiomp5 is interoperable with gcc OpenMP code generation so clang, gcc and icc compiled OpenMP code can be linked into the same image

There is no direct Dragon Egg (gfortran) solution for LLVM because outlining is done in the clang frontend

- Gfortran-generated objects continue to work because of library interoperability.

Intel compiler suite fully interoperable as always

- If libiomp5.so is the OpenMP runtime library

Conclusions

OpenMP support is available in clang now from clang-omp.github.com

Promotion to the clang mainline in process

Runtime available from openmp.llvm.org

- Usable for other purposes

Code is available
Contributions are welcome
Feedback please

Legal Disclaimer & Optimization Notice

INFORMATION IN THIS DOCUMENT IS PROVIDED "AS IS". NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. INTEL ASSUMES NO LIABILITY WHATSOEVER AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO THIS INFORMATION INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products.

Copyright © , Intel Corporation. All rights reserved. Intel, the Intel logo, Xeon, Xeon Phi, Core, VTune, and Cilk are trademarks of Intel Corporation in the U.S. and other countries.

Optimization Notice

Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice.

Notice revision #20110804