LLVM (Clang/Flang) OpenMP Implementation Status

Bullet Points Only

Ask Questions Any Time!

OpenMP Support in LLVM/Clang

OpenMP 5.0 feature status currently available under:

https://clang.llvm.org/docs/OpenMPSupport.html

Category	Feature	Status	Reviews
loop extension	support != in the canonical loop form	done	D54441
loop extension	#pragma omp loop (directive)	unclaimed	
loop extension	collapse imperfectly nested loop	done	
loop extension	collapse non-rectangular nested loop	done	
loop extension	C++ range-base for loop	done	
loop extension	clause: nosimd for SIMD directives	unclaimed	
loop extension	inclusive scan extension (matching C++17 PSTL)	unclaimed	
memory mangagement	memory allocators	done	r341687,r357929
memory mangagement	allocate directive and allocate clause	done	r355614,r335952
OMPD	OMPD interfaces	not upstream	**https://github.com/ /LLVM-openmp/tree/**
OMPT	OMPT interfaces	mostly done	
thread affinity extension	thread affinity extension	done	
task extension	taskloop reduction	done	
task extension	task affinity	not upstream	
task extension	clause: depend on the taskwait construct	worked on	
task extension	depend objects and detachable tasks	worked on	
task extension	mutexinoutset dependence-type for tasks	done	D53380,D57576
task extension	combined taskloop constructs	unclaimed	
task extension	master taskloop	done	
task extension	parallel master taskloop	done	
task extension	master taskloop simd	done	
task extension	parallel master taskloop simd	done	
SIMD extension	atomic and critical constructs inside SIMD code	unclaimed	
SIMD extension	SIMD nontemporal	unclaimed	
device extension	infer target functions from initializers	worked on	
device extension	infer target variables from initializers	worked on	
device extension	OMP_TARGET_OFFLOAD environment variable	done	D50522
device extension	support full ‘defaultmap’ functionality	worked on	
device extension	device specific functions	unclaimed	
device extension	clause: device_type	done	
device extension	clause: in_reduction	unclaimed	r308768
device extension	omp_get_device_num()	worked on	D54342
device extension	structure mapping of references	unclaimed	
device extension	nested target declare	done	D51378
device extension	implicitly map ‘this’ (this[:1])	done	D55982
device extension	allow access to the reference count (omp_target_is_present)	worked on	
device extension	requires directive (unified shared memory)	worked on	
device extension	clause: unified_address, unified_shared_memory	worked on	D52625,D52359
device extension	clause: reverse_offload	unclaimed parts	D52780
device extension	clause: atomic_default_mem_order	unclaimed parts	D53513
device extension	clause: dynamic_allocators	unclaimed parts	D53079
device extension	user-defined mappers	worked on	D56326,D58638,D58523,D58074,D60972,D59474
device extension	mapping lambda expression	done	D51107
device extension	clause: use_device_addr for target data	done	
device extension	map(replicate) or map(local) when requires unified_shared_me	worked on	D55719,D55892
device extension	teams construct on the host device	worked on	Clang part is done, r371553.
atomic extension	hints for the atomic construct	worked on	D51233
base language	C11 support	unclaimed	
base language	C++11/14/17 support	unclaimed	
base language	lambda support	done	
misc extension	array shaping	unclaimed	
misc extension	library shutdown (omp_pause_resource[_all])	unclaimed parts	D55078
misc extension	metadirectives	unclaimed	
misc extension	conditional modifier for lastprivate clause	unclaimed	
misc extension	user-defined function variants	worked on	D67294, D64095
misc extensions	pointer/reference to pointer based array reductions	unclaimed	
misc extensions	prevent new type definitions in clauses	unclaimed	

OpenMP Testing

- LLVM-Test Suite needs OpenMP support (parallelism in tests)
- LLVM-Test Suite support for the OpenMP V&V suite (ECP)
- Offload buildbots are under active planning
- “Host” backend for the device runtime planned (sanitizer support!)

OpenMP Optimizations

- “Scalar” Optimizations mostly merged
  - Enable with “-mllvm -attributor-disable=false”
  - Constant propagation, alias information transfer, ...
- OpenMP “aware” optimizations under review
  - Parallel region expansion (merging)
  - Runtime call deduplication

OpenMP Target Offloading

- Offloading to the host, Xeon PHI, and NVIDIA devices upstream
  - AMD GPU this year
- Some support for `math.h`/`cmath`* functions in target regions (Clang 9.0+)
  - “Native” language supported in target offloading
- (Static) linking still problematic expected to work before Clang 10.0

* To support native implementations of math functions (sin/cos/…) as well as other native intrinsics (basically everything Clang in CUDA-mode offers), we implemented a short-term workaround that can also cause problems. We are working on a stable solution for Clang 10.0 using TR8 features.

OpenMP in F18 (=new Flang)

- Good support for OpenMP parsing
- Some OpenMP semantic analysis already (want to help? Let me know!)
- Code generation of Clang is reused

OpenMP-IR-Builder

- Code generation parts (taken from Clang)
- Under active development, first patches accepted
- Help needed!

OpenMP Device Runtime “Redesign”

- Split common and target related functionality
- Simplify new offload targets (AMD first)
- Minimal code duplication
- New API layers (e.g., TRegions)