PathScale ENZO

GTC12 S0631 – Programming Heterogeneous Many-Cores Using Directives

C. Bergström | May 14th, 2012

Brief Introduction to ENZO

ENZO Overview & Goals

Speed transition to GPU & many-core systems

- Simplify the task of migrating software written in C, C++ & Fortran
- Uses OpenHMPP Standard (easy migration)
- CAPS HMPP compatible

Performance & HPC focused

- Fully exploits NVIDIA GPU features
- Generates native instructions optimized for NVIDIA GPU

Project Schedule & Status

Project Schedule

- ENZO Production release June 2012
  - OpenHMPP 2.5 C, C++ and Fortran

- Next ENZO Production release October 2012
  - More tools and better support for libraries
  - x8664 OpenHMPP task parallelism (similar to OMP3 tasks)
  - More optimizations (IPA / CG2 / textures)
  - OpenHMPP 3.0
  - CUDA 4.x
  - Kepler

Project Status

- OpenHMPP 2.5
  - Running CAPS C & Fortran Labs
  - PathScale written HMPP test suite
  - Customer code
- New C++ compiler
  - Perennial C++VS and CVSA regression free
  - Corner case compile time issues
  - Corner case runtime issues
- Ongoing effort
  - Performance tuning & benchmarking
  - Compiler robustness
    - Nightly compiler builds to address issues

Performance

Performance

- NVIDIA Tesla 2050 - “Lab2” SGEMM
  - ENZO
    - /opt/enzo/bin/pathcc -hmpp sgemm_time_1K.c -o sgemm_time_1K && ./sgemm_time_1K
    - 39ms
  - Source-to-source CUDA
    - hmpp gcc -std=c99 sgemm_time_1K.c -o sgemm_time_1K_caps && ./sgemm_time_1K_caps
    - 50ms
  - Optimal hand written GPU assembly
    - ~35ms

Performance – Copies H2D ENZO vs CUDA SDK

10 runs, time in ms (min/max/avg)

size	libenvyrt	libcuda
4M	(1/1/1)	(1/1/1)
8M	(2/2/2)	(3/4/3.1)
16M	(3/4/3.9)	(5/6/5.1)
32M	(7/7/7)	(11/13/11.4)
64M	(14/15/14.5)	(22/35/27.9)
128M	(27/29/28.2)	(43/45/43.6)
256M	(51/78/56.7)	(86/130/95.2)
512M	(108/117/112.5)	(171/225/182.8)
1G	(215/234/225)	(342/450/361.2)
2G	(431/468/449.9)	(699/1042/971.7)

Performance – Copies D2H ENZO vs CUDA SDK

10 runs, time in ms (min/max/avg)

size	libenvyrt	libcuda
4M	(3/3/3)	(10/10/10)
8M	(6/6/6)	(19/19/19)
16M	(11/12/11.9)	(38/38/38)
32M	(23/24/23.6)	(74/76/74.7)
64M	(45/47/46)	(148/160/151.7)
128M	(91/93/91.9)	(296/301/298.5)
256M	(179/187/181.8)	(587/599/592.1)
512M	(359/365/362)	(1179/1204/1189)
1G	(715/730/722.5)	(2353/2420/2384.5)
2G	(1424/1457/1441.3)	(4657/4912/4746.9)

Performance – ENZO driver api

RUNTIME OVERHEAD COMPARISON (CUDA DRIVER API, DEADBEEF KERNEL)

	LIBCUDA	LIBENVYRT
cuInit	17 ms	0 ms
cuDeviceGet	0 ms	1 ms
cuCtxCreate_v2	206 ms	8 ms
cuModuleLoad	0 ms	0 ms
cuModuleGetFunction	0 ms	0 ms
cuMemAlloc (1K)	0 ms	0 ms
cuFuncSetSharedSize	0 ms	0 ms
cuFuncSetBlockShape	0 ms	0 ms
cuParamSetv	0 ms	0 ms
cuParamSetSize	0 ms	0 ms
cuLaunch	0 ms	0 ms
cuCtxSynchronize	0 ms	0 ms
cuMemcpyDtoH (1K)	0 ms	0 ms
cuMemFree	0 ms	0 ms
cuModuleUnload	0 ms	0 ms
cuCtxDestroy	187 ms	2 ms
TOTAL	412 ms	12 ms

Performance – ENZO pinned memory

RUNTIME OVERHEAD COMPARISON (CUDA DRIVER API, DEADBEEF KERNEL)

allocation 377 ms vs 404 ms
copy: 185 ms vs 213 ms
free: 91 ms vs 614 ms

Outstanding Issues

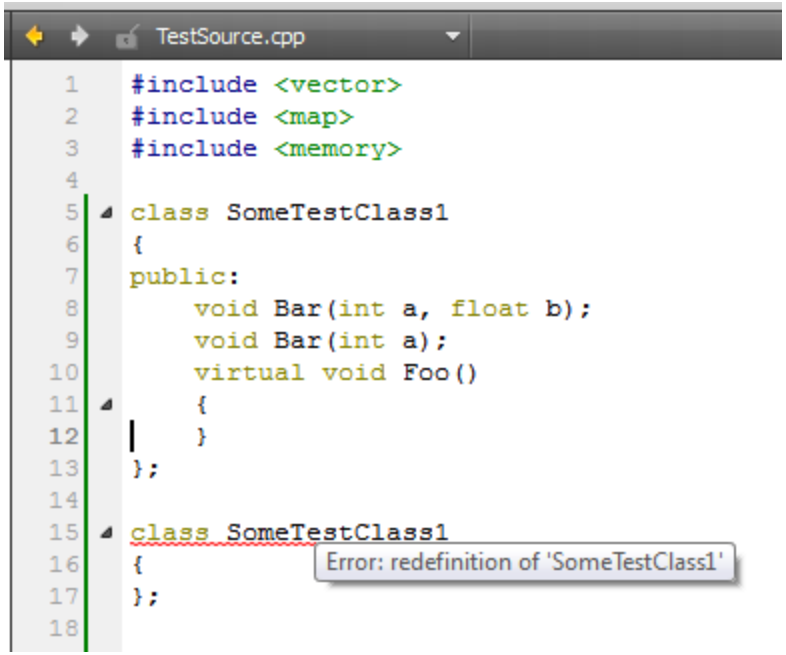
- **Very** difficult to debug
- Missing support for HMPPCG
- Missing code coverage and profiling tools
- Missing support for HMPP shared libraries

PathScale Labs

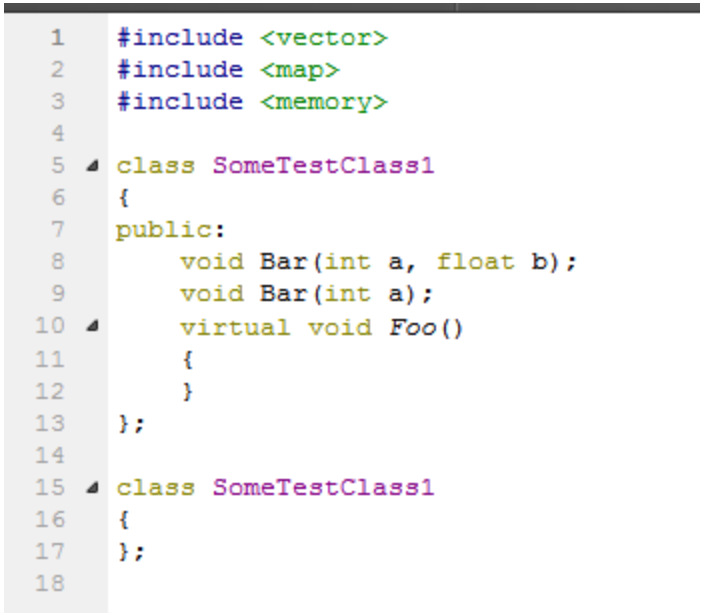
- IDE Project & Productivity tools (codename DogFood)
- OpenACC
- Distributed OpenHMPP
- ARM
- AMP++

DogFood - Error Diagnostics

DF

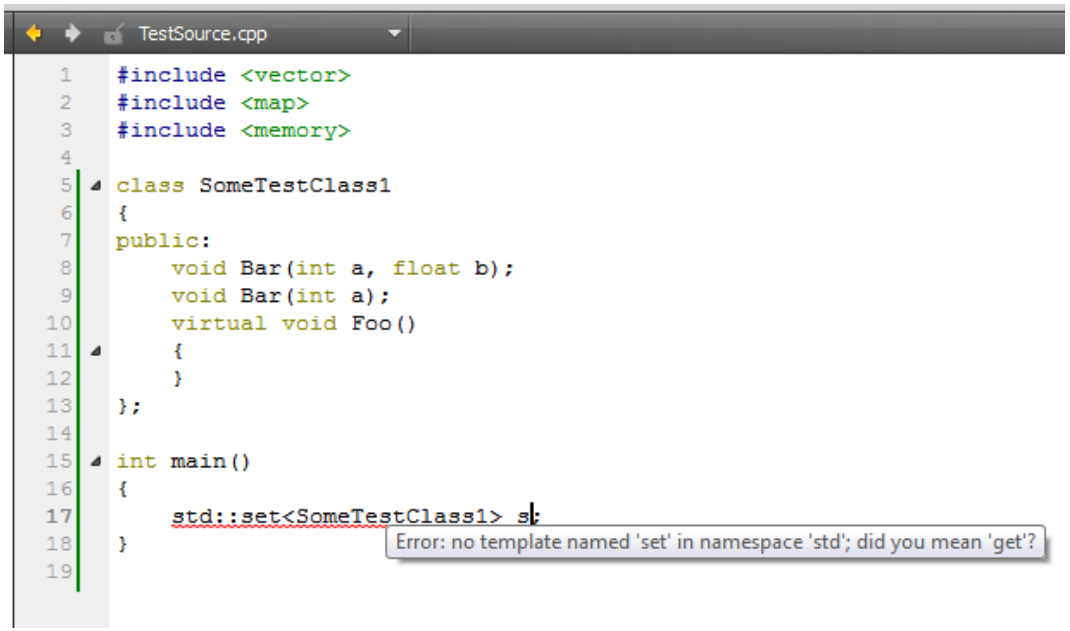


QtC

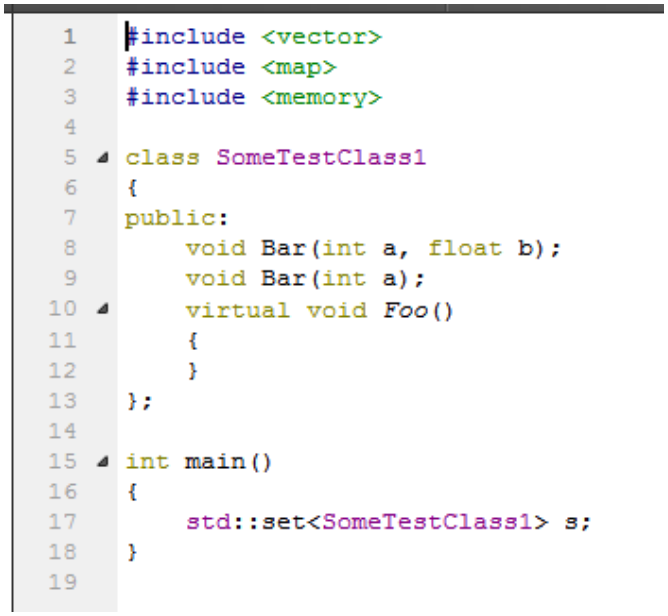


DogFood - Error Diagnostics

DF



QtC



DogFood - Warning

DF

TestSource.cpp

```
1  #include <vector>
2  #include <map>
3  #include <memory>
4
5  class SomeTestClass1
6  {
7  public:
8      void Bar(int a, float b);
9      void Bar(int a);
10     virtual void Foo()
11     {
12     }
13 };
14
15 int main()
16 {
17     std::vector<SomeTestClass1>;
18 }
19
```

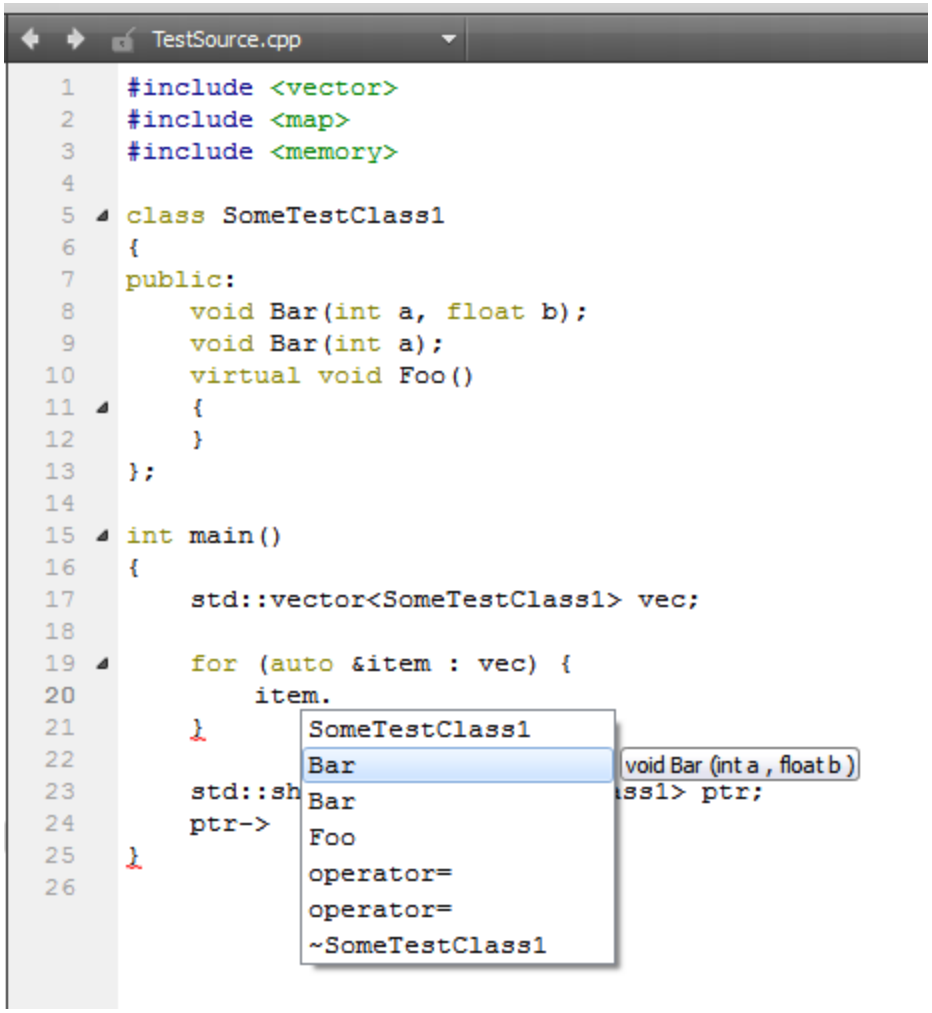
Warning: declaration does not declare anything

QtC

```
1  #include <vector>
2  #include <map>
3  #include <memory>
4
5  class SomeTestClass1
6  {
7  public:
8      void Bar(int a, float b);
9      void Bar(int a);
10     virtual void Foo()
11     {
12     }
13 };
14
15 int main()
16 {
17     std::vector<SomeTestClass1>;
18 }
19
```

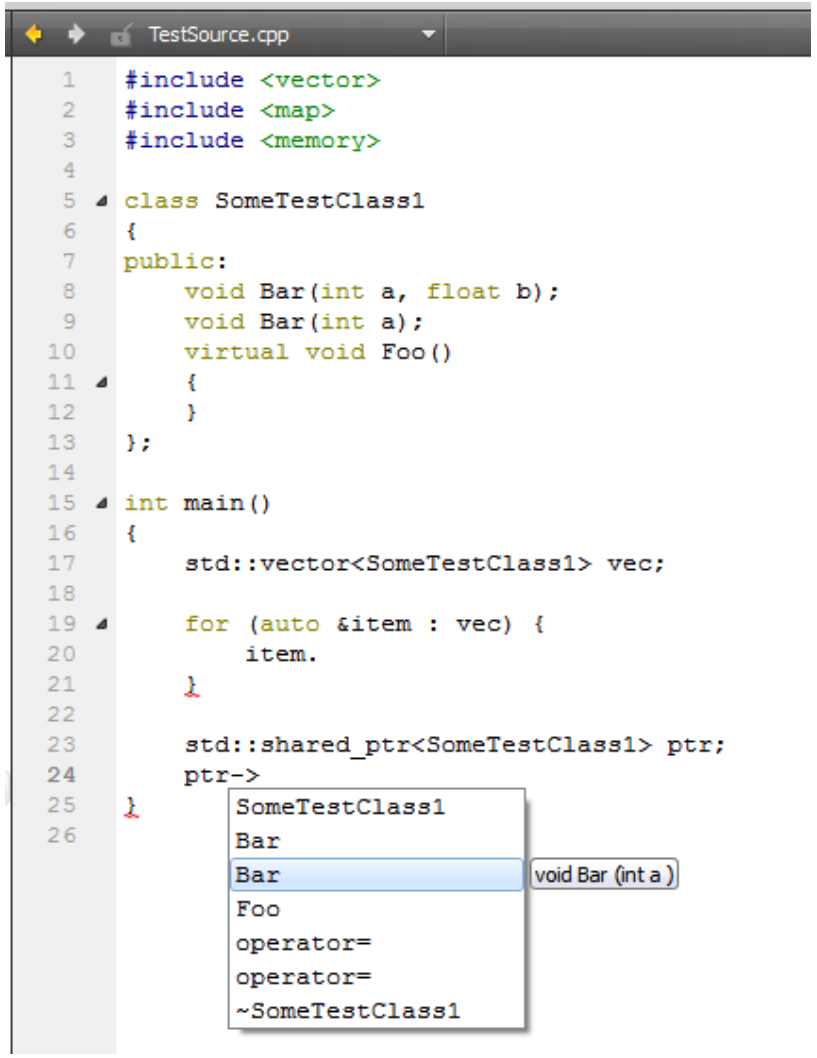
DogFood – Code completion

Code completion for
range-based 'for' loop variable



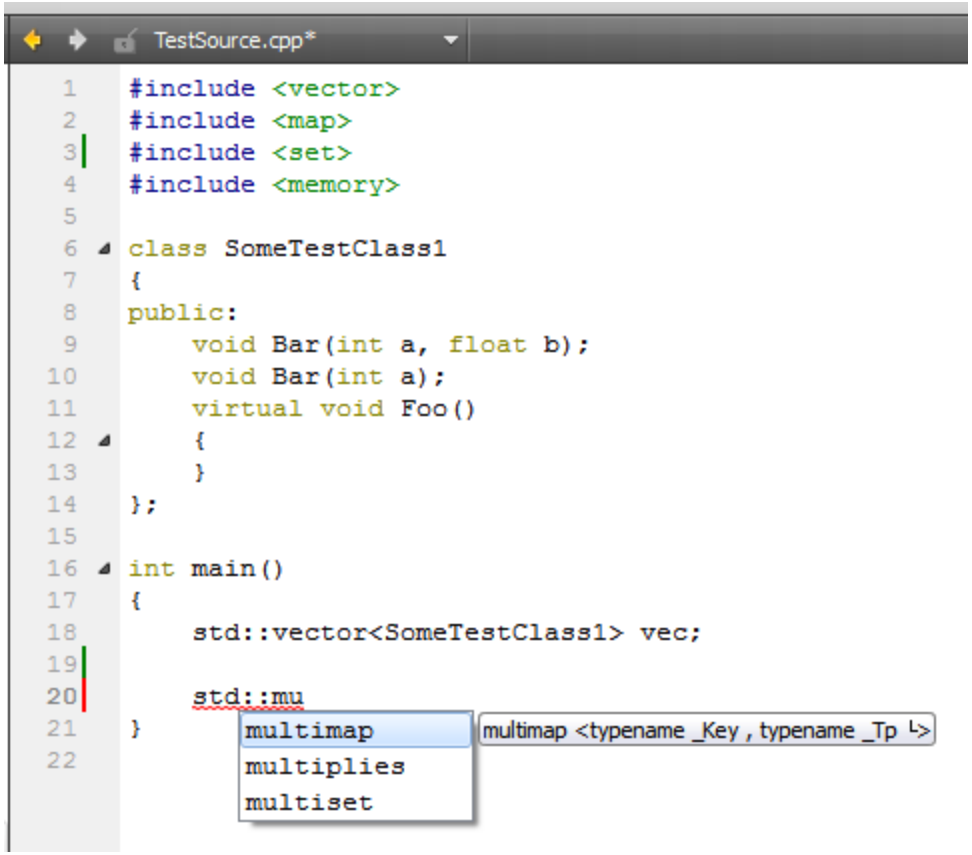
DogFood – Code completion

Code completion for
smart pointer inner type



DogFood – Code completion

Code completion for qualified names



Questions?

Thanks

- @CTOPathScale
- cbergstrom@pathscale.com
- http://www.pathscale.com/ENZO

Trademark Attribution

PathScale, the PathScale logo and combinations thereof are trademarks of PathScale Inc. in the United States and/or other jurisdictions. Other names used in this presentation are for identification purposes only and may be trademarks of their respective owners.