Original citation:
Truby, David, Bertolli, Carlo, Wright, Steven A., Bercea, Gheorghe-Teodor, O'Brien, Kevin and Jarvis, Stephen A. (2018) Implicit mapping of pointers inside C++ Lambda closure objects in OpenMP target offload regions. In: UK OpenMP Users' Conference 2018, St Catherine's College, Oxford, 21-22 May 2018

Permanent WRAP URL:
http://wrap.warwick.ac.uk/99430

Copyright and reuse:
The Warwick Research Archive Portal (WRAP) makes this work by researchers of the University of Warwick available open access under the following conditions. Copyright © and all moral rights to the version of the paper presented here belong to the individual author(s) and/or other copyright owners. To the extent reasonable and practicable the material made available in WRAP has been checked for eligibility before being made available.

Copies of full items can be used for personal research or study, educational, or not-for-profit purposes without prior permission or charge. Provided that the authors, title and full bibliographic details are credited, a hyperlink and/or URL is given for the original metadata page and the content is not changed in any way.

A note on versions:
The version presented here may differ from the published version or, version of record, if you wish to cite this item you are advised to consult the publisher's version. Please see the 'permanent WRAP URL' above for details on accessing the published version and note that access may require a subscription.

For more information, please contact the WRAP Team at: wrap@warwick.ac.uk

warwick.ac.uk/lib-publications

Implicit Mapping of Pointers Inside C++ Lambda Closure Objects in OpenMP Target Offload Regions

David Truby[1], Carlo Bertolli[2], Steven Wright[1], Gheorghe-Teodor Bercea[2], Kevin O'Brien[2], and Stephen Jarvis[1]

[1]University of Warwick
[2]IBM Research

February 28, 2018

Abstract

With the diversification of HPC architectures beyond traditional CPU-based clusters, a number of new frameworks for performance portability across architectures have arisen. One way of implementing such frameworks is to use C++ templates and lambda expressions to design loop-like functions. However, lower level programming APIs that these implementations must use are often designed with C in mind and do not specify how they interact with C++ features such as lambda expressions.

This paper proposes a change to the behavior of the OpenMP specification with respect to lambda expressions such that when functions generated by lambda expressions are called inside GPU regions, any pointers used in the lambda expression correctly refer to device pointers. This change has been implemented in a branch of the Clang C++ compiler and demonstrated with two representative codes. Our results show that the implicit mapping of lambda expressions always exhibits identical performance to an explicit mapping but without breaking the abstraction provided by the high level frameworks, and therefore also reduces the burden on the application developer.