Code Together Podcast #28: *ZIB's oneAPI Story: Discovering Portability across Multi-vendor Hardware* - June 2021
Host: Brenda Christoffer, Intel
Guests: Dr. Thomas Steinke & Dr. Steffen Christgau, Zuse Institute Berlin (ZIB), & Klaus-Dieter Oertel of Intel

Brenda Christoffer (00:04): Welcome to Code Together. An interview series, exploring the possibilities of cross-architecture development with those at the forefront. I'm your host Brenda Christoffer.

Brenda (00:16): Heterogeneous programming has been around for quite a while. It's challenging for developers because it's been difficult to code or forces compromises in performance, and that's no fun. oneAPI is working to break that mold. It's open, unified and standard-based approach is different. Developers can use a single code base across multiple types of architectures and vendors, and oneAPI also carries a promise of both performance and productivity. Many HPC research centers, enterprises and developers are now experimenting and adopting oneAPI. Today, we turn to a few experts that have dived in to share their experience. Joining us is Dr. Thomas Steinke, head of the super computing department at Zuse Institute Berlin, which we'll refer to as ZIB. Thomas has been at ZIB for 29 years and is a well-known veteran across the HPC industry. He is the author of many papers covering heterogeneous computing, Fortran, memory, oneAPI, openMP and more. Welcome Thomas.

Thomas Steinke (01:27): Hello, this is Thomas.

Brenda (01:29): And also from ZIB, we have Dr. Steffen Christgau. He is an expert in parallel and distributed computing, super computing and algorithms for innovative architectures. Thanks for joining us, Steffen.

Steffen Christgau (01:43): Yeah. Hi Brenda.

Brenda (01:45): And last, we have Klaus-Dieter Oertel, an HPC senior technical consultant from Intel. He belongs to the first generation of parallel programming experts in Germany and has worked on all kinds of super computers. Klaus-Dieter collaborates with ZIB on oneAPI, along with many other European customers. Great to have you with us.

Klaus-Dieter Oertel (02:07): Hello Brenda.

Brenda (02:08): Well, let's get started. So, Thomas, could you share how you got started with oneAPI for heterogeneous programming?

Thomas (02:16): Yeah. Thanks for this question. We, as an HPC center, watched of course technology trends over the last decade and we saw GPUs becoming more and more important, popular in the HPC scene. We saw also that the type of programming models wasn't what we are looking for. So we want to have most code portability across different processor architectures, and we don't want to be bound to specific architectures and software programming models as devices. Nowadays, we consider along with the CPUs also GPUs of course, an FPGA, but there might be other processing devices on the horizon in the artificial intelligence space and beyond. The challenge here is that we have a high innovation rate and the hardware, so we buy every three to five years and new big system, but the software needs to be run over decades. So we have software codes running for 30 years and so on and still needs to be maintained.

Thomas (03:26): So deciding then for an attractive hardware isn't like chicken-egg problems. So one may buy hardware, which gives a highest theoretical performance, but maybe no software's running on it or vice versa that there's a bunch of running software, but they support only specific hardware at our site. We want to remove these when mounting to specific technologies and here comes oneAPI into place. As one goal of oneAPI is to support approaches like that one source code can be able to run on different vendor architectures, not even different processor, incarnations like CPUs, GPUs and so on, but also running on different vendors on this is much important for us as this brings us much more productivity and less maintenance for the code developers.

Klaus-Dieter (04:25): Thank you Thomas for sharing your motivation to get involved with oneAPI. So what was then the first use case you started with at ZIB and why did you choose it?

Steffen (04:37): Yeah, maybe I can jump into this question because I was one that actually did a lot of work with oneAPI here at ZIB. And the very first workload that we started with was not that much more than the usual vector addition example that you find in ZB codes and so on. On usual tutorials, we had a very simple standard application with really one single kernel that we migrated where they have all the Compatibility Tool to oneAPI, but then we started looking at something more serious and in my old working group at the University of Potsdam, there was a collaboration between the university there and the German research center for geo-sciences, and they are working on a code named *easyWave*.

Steffen (05:22):That is a quite simple written application. That's why it's named *easyWave* and it's easy in that sense that it's compute tsunamis really, really fast because it is used in early warning centers. So in case you have an earthquake inside an ocean or so, you may be interested as a government, is my coastline affected and are my people affected or will they be affected by the tsunami that might develop out of this earthquake and therefore you need a fast computation that's why they developed *easyWave*.

Steffen (05:55):At the time they were developing the application they also addressed GPUs because they were well suited for this particular problem and they wrote that application at that time some years ago in CUDA. But the promise now was from the Intel folks that there exists some tool that would help us to migrate this CUDA code base into DPC++ into a single base. And that's where our journey actually started to become more and more interesting for us. That's where we actually get started with oneAPI and I think that was kind of a good collaboration between us at ZIB was this application and with the folks at Intel concerning this *easyWave* code in the compatibility tool.

Klaus-Dieter (06:39): Okay, thanks. Going now into some more technical details. Could you elaborate on the challenges of programing with oneAPI? So did you face any surprises, what are the lessons learned.

Steffen (06:52): Well actually one surprise for us was this compatibility tool that it actually works as promised so to say. So if you first get confronted with a tool that promises you there is some kind of automatic support for migrating a code base from, let's say, one programming model to another one, you raise your eyebrows and think, oh, that will really work, but essentially it did work and this was really, really surprising for us and positive surprise. And for me personally, the challenge was learning a new programming model, so I had some CUDA background and usually I'm C programmer.

Steffen (07:33): Now I was confronted with SYCL or DPC++, and at the first glance, I was kind of skeptic because I have all these abstractions and C++ going on, with the buffers and accessors for accessing the data on your GPU, which well, hides a lot of details from you, but also you have the feeling personally, for me, as a C programmer that you lose some kind of control over the application. Meanwhile, SYCL now provide you with different levels of abstractions that you can actually use. So you can actually use these higher-level constructs with C++ abstractions hiding a lot of details for you, which might be convenient.

Steffen (08:12): But on the other hand, you also have the control over things like memory management with the help of unified shared memory. That's good to have this choice because in the end, I think it is often personal preference, which way you want to go. Do you want clean code that abstracts a lot of things, or do you want to write more low-level oriented code to have more control over where does the memory comes from and where does it go to and which layout would it have maybe.

Steffen (08:40): So this large space of personal preferences and how to write code and the nice thing about DPC++ and SYCL oneAPI. So in general is that you can choose from all of it. You can even mix your code and have different flavors inside your code. So this is really something that is good from our perspective and what is also good is that these things are not only bound to a particular vendor, which would be Intel in that case, but Intel is also interested to push the things that are beneficial also from our perspective into the standard to have other people implementing

the standards and by doing so, providing other users with these great features as well, so that they can benefit from that.

Steffen (09:23): And yeah, as I said, I think we provided a lot of feedback to Intel on that side, when it comes to usability from the tool on its own, how it is working, how we can interact with that, but also on aspects of the language and how you can use actually SYCL and DPC++ in an efficient way.

Klaus-Dieter (09:41): Yes. Thank you. And I thought I really have to thank you here for all the collaboration, because that at least started at a very early stage of oneAPI so you were really risk-taking and your feedback on our compiler and tools such as the DPC++ compatible tool. It was really essential for the rapid development from our beta product through the quality software that oneAPI is today. So this was really great to have your feedback and as an outcome, for example, many other customers now have successfully used DPCT tool to migrate their CUDA software to DPC++ code. So our collaboration was really fruitful here. So what were then the benefits of oneAPI for you, I think in particular with respect to the 4Ps of portability, programming, performance and productivity.

Thomas (10:34): Yes Klaus-Dieter, let me start maybe with one of the 4Ps that is Portability, as I said in the introduction, portability is very important for us. And through the work we have done plying compilers and tools from the oneAPI framework, we were able to demonstrate that it is possible to have a functional source code written in DPC++, and running that on CPUs, GPUs and even FPGAs, I come to that later on. The other aspect, programming stuff might dive into that later, but I want to continue with the portability aspect why it is so important, why we are so satisfied with oneAPI at this point.

Thomas (11:23): Portability sounds like a trade-off between function and performance and the big challenge is that one doesn't want to give up performance, particularly in the HPC scene. So of course, we also looking at performance. Steffen will talk to it later because he had this nice experiences with that. But what also to take into account that tuning code to getting the last bit of performance requires often manual labor work. And so we are looking a reasonable balance between approaching a high-sustained application performance verses maintaining the code and this is part of the portability aspect and oneAPI supports us along this way that we have functional correct code running on different devices. And we then can make decisions whether to take more thought in getting performance on the specific device.

Steffen (12:27): Yeah, so concerning the programming point, I mean, Thomas already extended a little bit so if you want to support multiple devices from different vendors, we don't want to have different code bases for all these hardware classes and hardware generations and so on. So what we were able to do with oneAPI or in particular was a use case *easyWave* is to write code that runs both on Intel CPU's as well as GPU's, but also on an Nvidia GPU, so we were both covering different GPUs from different vendors and we were also covering CPUs and GPUs, so different device classes with a single code base. That doesn't mean that you have to do particular things or to do further optimizations for the specific device classes, but in principle it's a single programming model and oneAPI provides you here with a good portability using the same source code and using the same programming language.

Klaus-Dieter (13:27): Thanks. So now I'm really keen to hear about the holy grail in HPC which is performance. So what can you tell us about the performance of the oneAPI code?.

Steffen (13:38): Yeah, in particular for this *easyWave* code, this was a really good result that we achieved. So as I said, we started with the original code base that was written in CUDA or had support for CUDA, then we use the compatibility tool to migrate the code and then we were able to run it on, as I said, Intel CPU, Intel GPUs and Nvidia GPUs. So for the Nvidia GPU, we already had a baseline, which was your original program version and then we had this migrated code base and surprisingly, we had only very little performance overhead. So when it comes to extra numbers, we can say that they were very close to the original Nvidia code or CUDA code.

Steffen (14:20): The maximum performance loss that we observed was 4% of the original run time. So 1.04 speed up or slow down as you want to call it, depending on how you look at it and this was enabled by the contribution to the open oneAPI ecosystem from Codeplay. So they provided their implementation of the runtime and compiler, which is based on LLVM. This was actually the technical enabler to run the migrated code on Nvidia hardware back again. So when we use the same device with different codes, but it's always the same problem. Optimistically speaking, we didn't lose any performance in that particular use case and this was really surprising and good news for us.

Klaus-Dieter (15:03): Thanks for providing these insights and the outcome for you. So my final question that for ZIB, what is the future going on to bring in HPC?

Thomas (15:13): First, I'd like to take these upcoming challenge in the HPC industry that is providing systems at exascale performance and I want to position our HPC center in that area. The Zuse Institute is become recently a member of the tier two HPC Alliance in German. So this gives us the opportunity mission to support the scientists coming from their laptop and then the tier-three university data center up to the exascale at the end. We see one of our university here to support the scientist, helping them to prepare their codes for exascale. So bridging the gap from resources, which are available at the three and up to the tier one, and of course in between our resources and services. And this is important because younger scientists need to be introduced with the technologies and challenges of HPC to solve fascinating scientific questions and areas like health care and our G-conversion, or basic science to give you a few examples on what our system has performed.

Thomas (16:37): We are active, involved in projects where the COVID-19 virus spread is analyzed and predictions are made, which are immediately used by politicians to make decisions so we support decision-makers in that area. We also are running workloads to applying model-driven and data-driven approaches in the drug-design process and we are successful on that as well. And other examples are for instance, helping scientists and companies about decisions about the location of theme parks and energy conversion projects. Or even something like very basic science questions in the area of astrophysics requires large HPC resources, which can be given by tier three, tier two and tier one centers in a pyramid shape. They, in that case, for instance, to simulate the radiator transfer and stellar and planetary atmospheres, geometric observations that came by astrophysics people. So if you're asking about the future, I may base two aspects.

Thomas (17:54): One is very much important for us. That is the aspect of power consumption or energy efficiency giving our end users, the next performance steps. We have to carefully select technologies, which are power-efficient. So if you're looking very much forward to mobile developments, technology implementations, and the other aspect is immediately connected why we are so interested using oneAPI that is the diversity of processing such devices, the contention so we see not only the CPUs, and GPUs in the talk here in the last minutes briefly, but I also mentioned FPGAs which we are actively working with in particular to be getting more insight about competent performance pivot.

Thomas (18:51): And for that oneAPI that's another interesting usage model and of course there might be other accelerators in the artificial intelligence space, or maybe even quantum computing and that's the processor side. I want also to remark that there are fascinating technology developments on the memory side, but only HPM and non-volatile RAM, but there are improvements only horizontal as well. One of these challenges related to memory technologies, for instance, bringing down the memory, latencies is one challenging aspect in the future for HPC.

Steffen (19:33): Thomas you mentioned a very important point. I think though, you mentioned that there will be more heterogeneous systems and you will not only have from my perspective, a particular type of accelerator in your supercomputer, but you will have from my perspective, different types. I mean, there are already data centers out there, which not only use GPUs as accelerators, but they also have FPGAs or other components attached to the supercomputer to do computations on them. And if you want to program them, I think you don't want to learn new

programming models over and over again, and tune your particular code to these different architectures. And that's where oneAPI has an important point with that programming modernized ecosystem. You can potentially write code. I think not for all of these hardware platforms since they're somehow different yet, but you can easily write code for the same hardware class and similar hardware architectures with the same code base, which increases your productivity.

Steffen (20:46): So the domain researcher that has written successfully is SYCL code or DPC++ code is now able to run this particular code, not on only let's say on CPU, but also on different platforms without major rewrite efforts, but maybe was only little effort. So the productivity for the researcher increases drastically I think.

Brenda (21:08): Thomas and Steffen, it's been really exciting to hear about the success you're having with oneAPI and having it be both performant and bringing in productivity. So you can code once rather than having to maintain your code across all the different architectures with different codes. Thomas, are there any places you want to share with people and where they can learn more?

Thomas (21:34): Yeah, so our design so far studies and investigations, we put it in papers which are then public accessible. I point people who are interested to learn what we did in the past to the web site of our institute, that is www.zib.de and then from there, you can search in the menu for our names at this Thomas Steinke or Steffen Christgau, and you will find the publications there.

Steffen (22:12): What you may also find interesting is the oneAPI workshops that we had at ZIB. There's a lot of insights for starters, I think as well. So what does a programming model look like, we also provide lots more details on the migration process together with Intel. And that might be a good starting point as well.

Brenda (22:32): And Klaus-Dieter, where could people learn more about oneAPI and from Intel?.

Klaus-Dieter (22:38): Yeah, this is very easy to get more information on this because we have a nice entry portal to oneAPI. This was a webpage is intel.com/oneAPI and on this entry portal you will find more information, more links, for example, how to get access to the oneAPI software. Also, there's a link to trainings and webinars and similar to what Steffen mentioned already. There's also a link to a training on how to migrate existing conduct code to oneAPI. And in particular, what should not be forgotten is we also offer the DevCloud so this bunch of systems and it's very easy to get an account on our DevCloud and on the DevCloud, you can then try out the pre-installed oneAPI software on a real accelerator.

Brenda (23:29): Super. Well, thanks again, Thomas. Thank you so much for joining us.

Thomas (23:33): Yeah, thank you. It was a pleasure to talk to you.

Brenda (23:36): Stephen. Thank you, it's been great to hear your insights on the programming and the performance side.

Steffen (23:42): Yeah, thank you. It was a pleasure for me as well

Brenda (23:45): And Klaus-Dieter, thanks so much for joining.

Klaus-Dieter (23:48): Thank you Brenda.

Brenda (23:49): And thank you to our listeners today for joining us. Let's continue the conversation @oneapi.com