Learning SYCL: challenges and opportunities

Roberto Di Remigio Eikås

oneAPI Developer Summit, 9 May 2022

ENCCS
EuroHPC National Competence Centre Sweden
EURO
UPPSALA UNIVERSITET
RI. SE
Swedish Research Council
VINNOVA
Sweden's Innovation Agency

EuroHPC JU Systems

LUMI, FINLAND

VEGA, SLOVENIA

KAROLINA, CZECH REPUBLIC

MELUXINA, LUXEMBOURG

DEUCALION, PORTUGAL

LEONARDO, ITALY

DISCOVERER, BULGARIA

https://eurohpc-ju.europa.eu/discover-eurohpc-ju

LUMI
Sustained perf: 375 petaflops
Peak perf: 552 petaflops
AMD EPYC CPUs, **AMD** Instinct GPUs

Leonardo
Sustained perf: 249.4 petaflops
Peak perf: 322.6 petaflops
Intel Ice-Lake (Booster), Intel Sapphire Rapids (data-centric), **NVIDIA** Ampere GPUs

Vega
6,8 petaflops
AMD EPYC CPUs, **NVIDIA** A100 GPUs

Karolina
9,13 petaflops
AMD EPYC CPUs, **NVIDIA** A100 GPUs

MeluXina:
10 petaflops HPL
AMD EPYC CPUs, **NVIDIA** A100 GPUs

Discoverer:
4,44 petaflops
AMD EPYC CPUs

Deucalion:
10 petaflops
ARM A64FX CPUs, **AMD** CPUs, **NVIDIA** GPUs

EURO

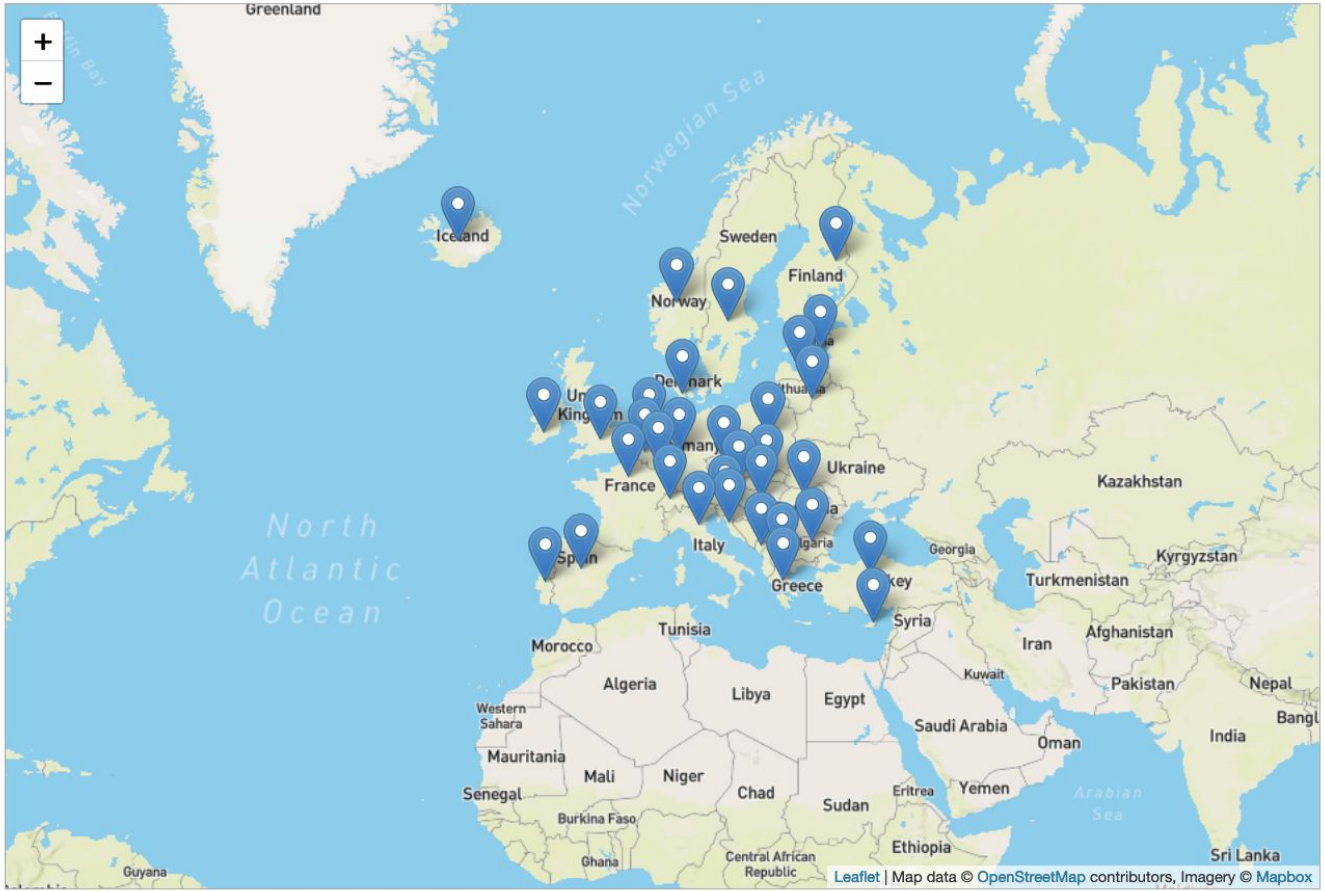
ABOUT US | OUR SERVICES | NEWS | Login

Competences

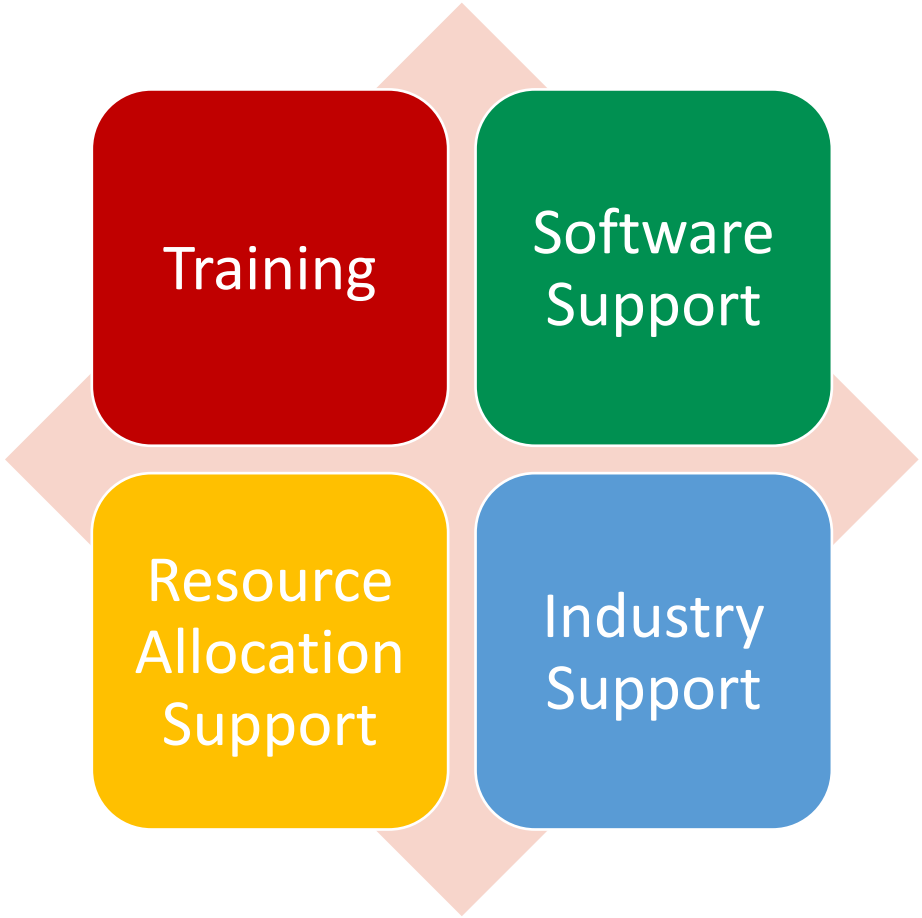
Your competence level

Please select...

- [ ] Awareness Creation
- [ ] Business & Project Consultancy
- [ ] Expert Technical Consultancy
- [ ] Mastering the European HPC Ecosystem
- [ ] Products & Services
- [ ] Technology Assessment and PoC



ENCCS


Training
Software Support
Resource Allocation Support
Industry Support


ENCCS
EuroCC National Competence Centre Sweden
Home
News
Events
Training
Industry
Supported Software
Proposal Support
About
Contact
The Swedish EuroCC Hub for High-Performance Computing
TRAINING
Providing training on GPU usage, AI and HPC optimisations as well as on usage of HPC in scientific disciplines such as Life sciences, Chemistry, Climate modelling, Engineering and more.
INDUSTRY – PUBLIC ADMINISTRATION
Assisting SMEs, large businesses and public administration in delivering competitive benefits from advanced HPC and build awareness about HPC and AI/HPDA competences and identifying strategies for technology transfer from academia.
EUROHPC JU SYSTEMS ACCESS PROPOSAL SUPPORT
Helping researchers both in industry and academia with application forms to apply for access to EuroHPC JU (pre)exa-scale system application support.
SUPPORTED SOFTWARE
Support users move to pre-exascale systems. Set up mechanisms to educate users about technical requirements for scaling and provide consulting in HPC, GPU acceleration AI/HPDA and data handling.
Want to get 100x more CPU/GPU power?
Get up to a year access to European high-performance computers to develop, test, or run your code for free!
Do you need compute capacity for your Jupyter hub solutions, or for data analysis? Perhaps you use simulations in engineering or life science or any other domain? Check your needs below!
Unsure? We can help you!
JUST CONTACT US

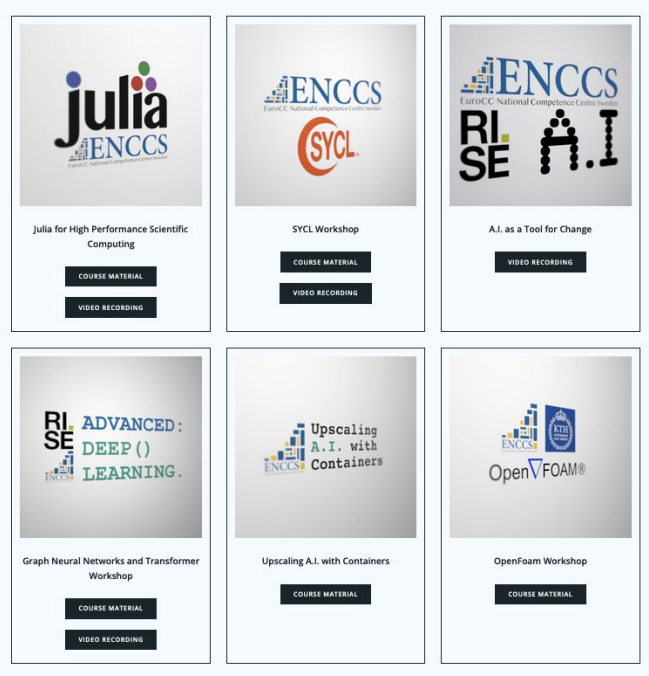
ENCCS training portfolio

Targeting **academia**, **industry**, and **public sector**:

- Programming models
- Domain-specific software
- Hackathons and bootcamps
- How to apply for EuroHPC JU resources

Training material:

- Available online https://enccs.se/training-resources/
- Creative commons: free to reuse and modify, with acknowledgement



Why learn SYCL? HPC heterogeneity

EuroHPC JU Systems and **beyond**:

- Computational power delivered by heterogeneous CPU+GPU architectures
- Heterogeneity also in CPU and GPU vendors

Performance and performance portability:

- GPU ports are costly
- Multiple ports unfeasible

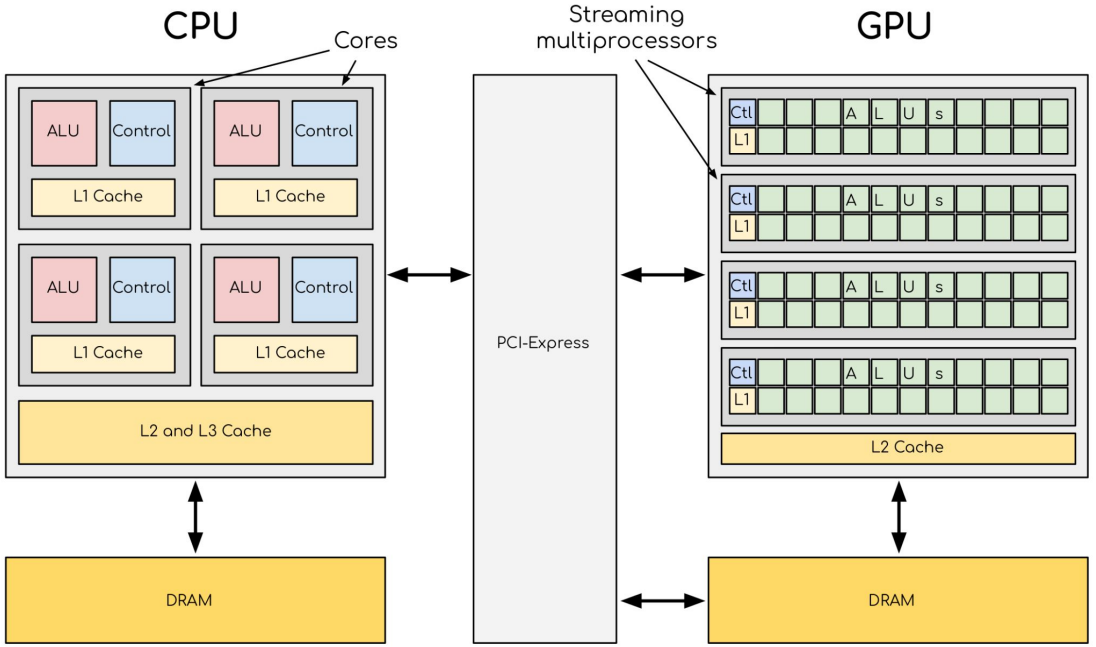
Why learn SYCL? Portability

Low-level, hardware-specific:

- Code duplication
- Time-consuming maintenance.
- Vendor lock-in

`pragma`-based:

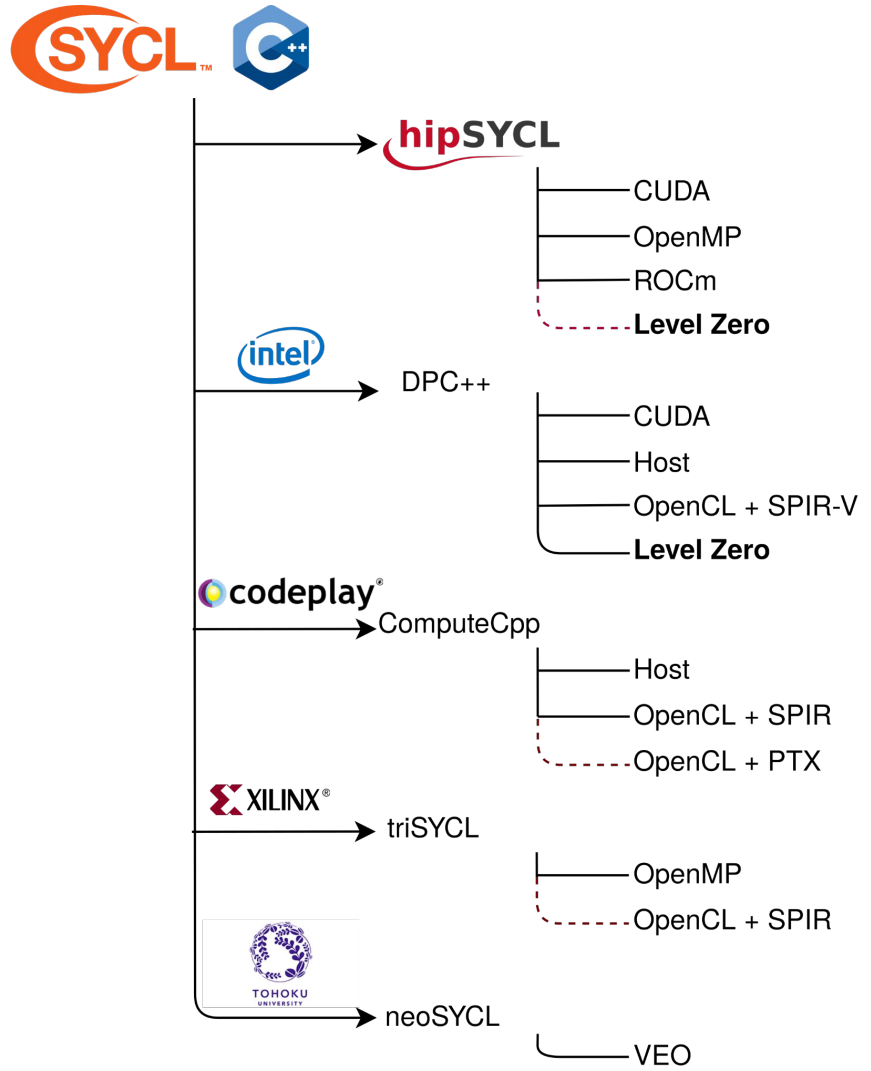
- Highly dependent on compiler support
- Readability and code divergence



Why learn SYCL? Portability

SYCL:

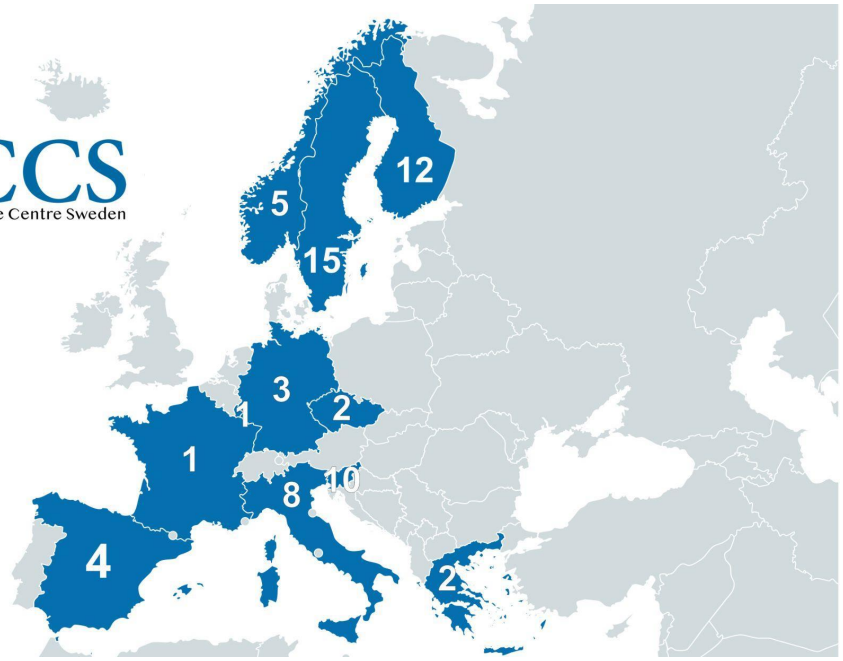
- Standard C++
- High-level abstractions
- Multiple backends



Our goals in designing a SYCL workshop

- Create a training offer of interest for all EuroHPC-JU countries
- Build up competence in cutting-edge programming frameworks
- Collaboration with other competence centres
- Show that SYCL is viable on **today's HPC machines**

Target audience



Students, researchers, engineers, and programmers with no previous experience with SYCL.

Prerequisites

- A working knowledge of recent C++ standards
- Some familiarity with C++17

No knowledge of CUDA/HIP/OpenMP/OpenACC!

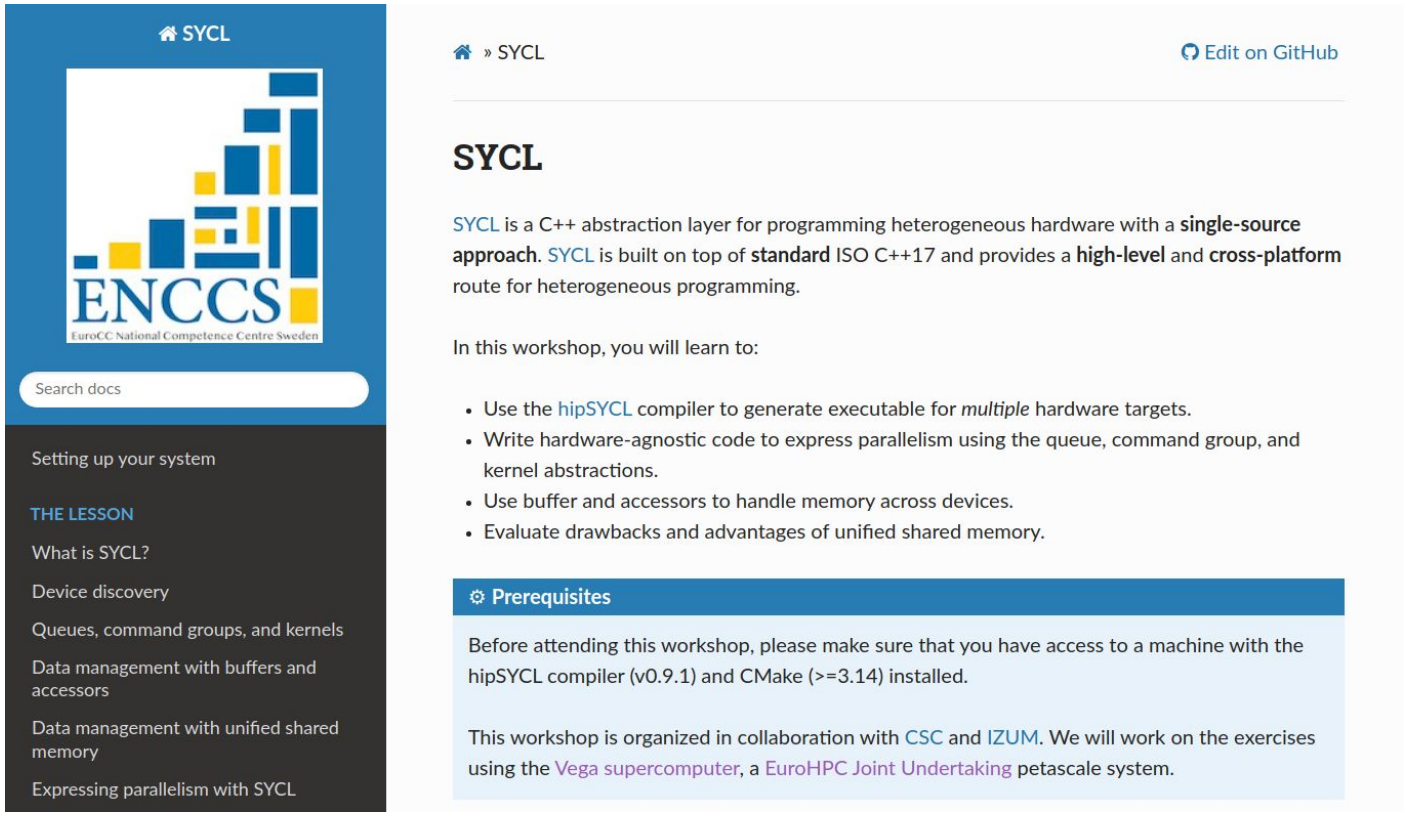
Overarching philosophy:

- We are **not** teaching the SYCL standard
- Learn by doing: plenty of hands-on sessions

Learning goals

- How to define parallel work? how to offload it to devices?
- Write hardware-agnostic code to express parallelism
- Use buffers and accessors to handle memory across devices
- Evaluate drawbacks and advantages of unified shared memory
- Run examples and exercises on **EuroHPC-JU hardware**

Workshop set up

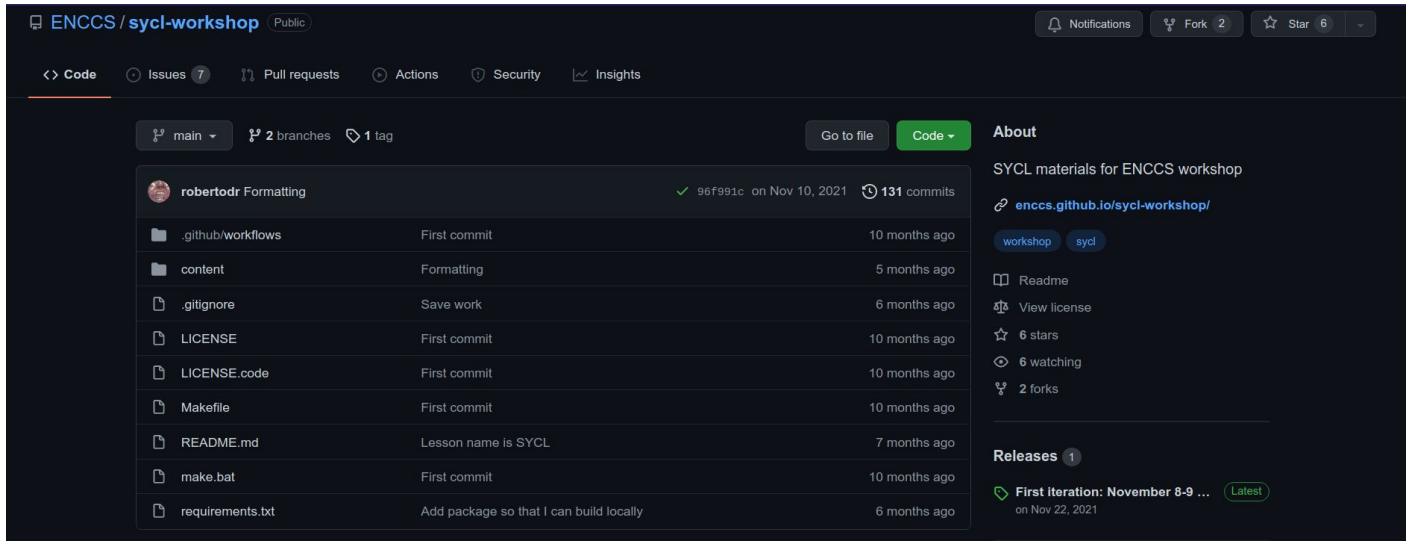


https://enccs.github.io/sycl-workshop/

Website in `sphinx-lesson` format

- No slides. All content spelled out on the webpage => Better for self-learning
- Based on Carpentries and CodeRefinery formats

Workshop set up



All content in GitHub repo

- github.com/ENCCS/sycl-workshop
- Easily accessible
- Open for contributions
- Use issues to keep track of what to improve

Lesson structure

30 min	What is SYCL?
30 min	Device discovery
30 min	Queues, command groups, and kernels
30 min	Data management with buffers and accessors
30 min	Data management with unified shared memory
30 min	Expressing parallelism with SYCL: basic data-parallel kernels
30 min	Expressing parallelism with SYCL: nd-range data-parallel kernels
40 min	The task graph: data, dependencies, synchronization
40 min	Heat equation mini-app
30 min	Using sub-groups in SYCL
30 min	Profiling SYCL applications
40 min	Buffer-accessor model vs unified shared memory

First run 8-9 November 2021.
Using **VEGA** https://en-vegadocs.vega.izum.si/
Second run 19-21 April 2022.
Using **Karolina** https://docs.it4i.cz/karolina/introduction/

Max 4 hours per day

- Reduce learning overload
- Allow new knowledge to sediment

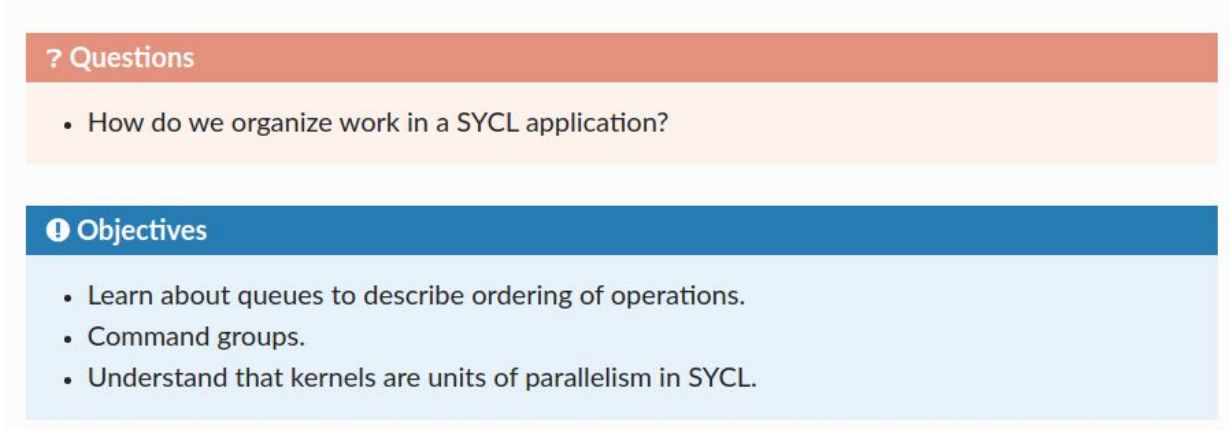
Episodes

- Maximum 40 minutes
- At least 50% time for hands-on exercises
- One episode per main concept

Episodes: *keep participants engaged in active learning*

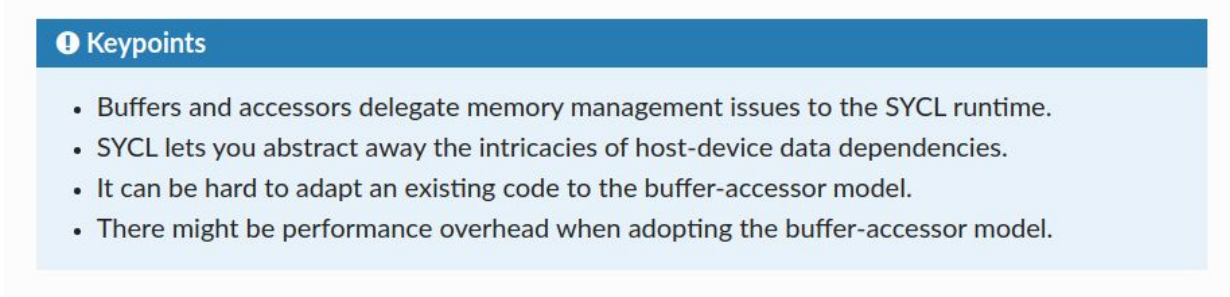
Starts with:

- Questions: *What? How? Why?*
- Objectives: learning goals at a glance



Ends with:

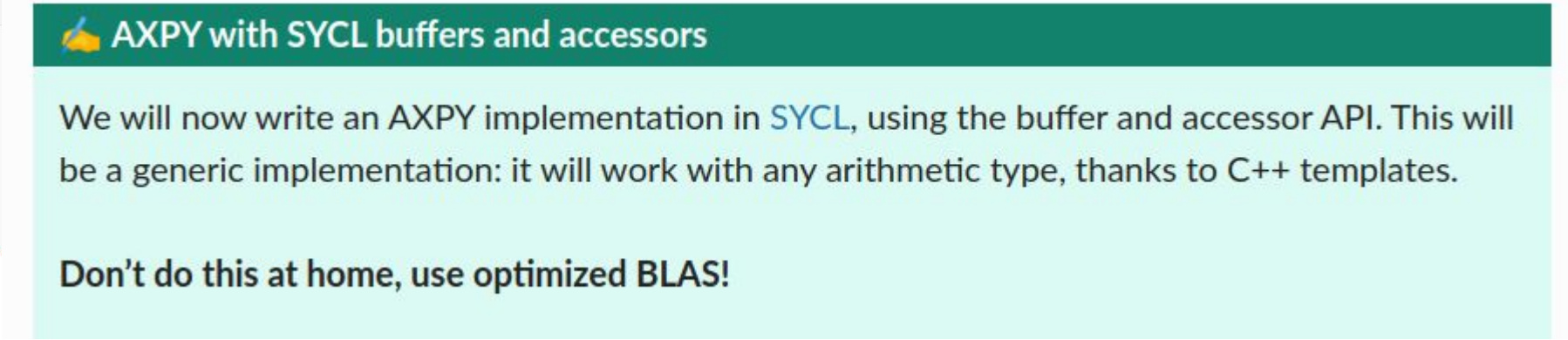
- Keypoints: take-home messages



Episodes: *keep participants engaged in active learning*

Visual cues

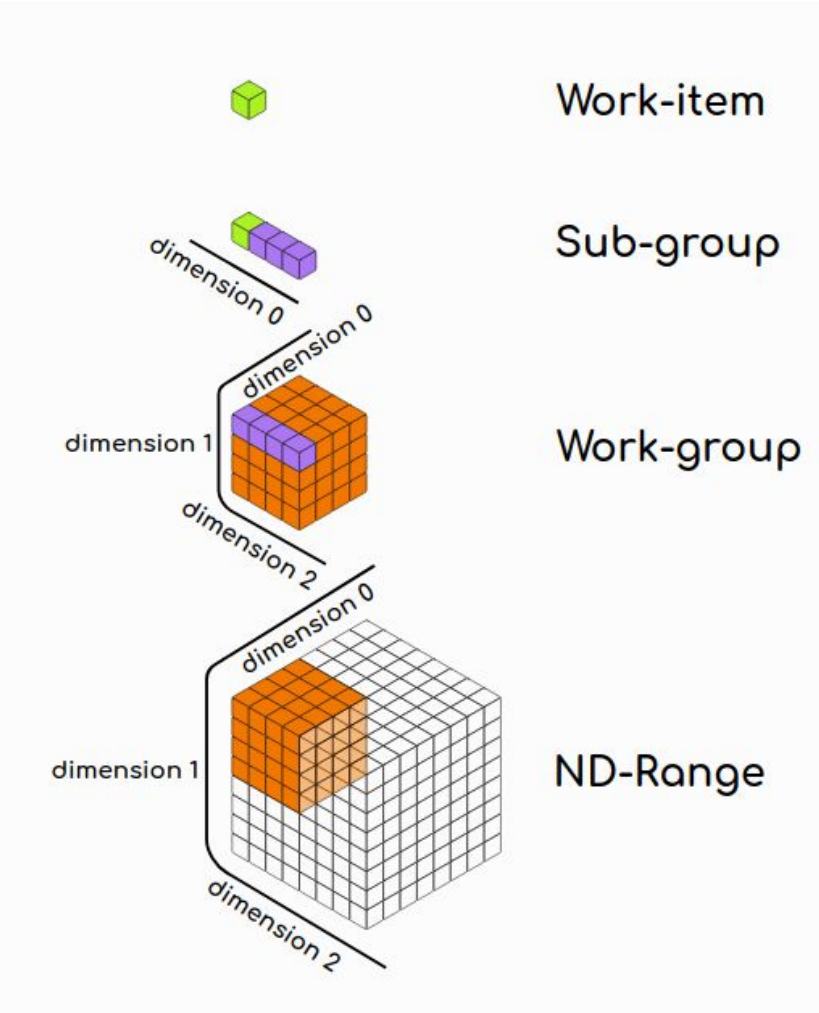
- Signposting of activities during the lesson: function signatures, typealongs, hands-on exercises



Episodes: *keep participants engaged in active learning*

“Lecture”

- Use clear prose, in active voice
  - Not a university lecture
  - Not a manual
- Tables, figures, schematics
  - Distill information
  - Easier to recall concepts
- Exercises must be self-explanatory
  - Solution available within the repo
  - Correctness can be self-checked



Lesson delivery

- 1 “**lecturer**” + 4-6 **helpers**
  - “Lecturer” goes through concepts
  - “Lecturer” introduces hands-on exercises
  - Helpers lead hands-on exercises in **breakout rooms**

- **Fully remote** using Zoom
  - Low barrier for international audience
  - Hands-on in breakout rooms: 1 helper + 6-8 participants.
  - Main room recorded and uploaded to YouTube

- Use **HackMD collaborative scratchpad** for Q&A
  - All questions are in public record
  - Anyone can answer
  - No risk of talking over one another

Outlook, Challenges, Opportunities

- **First** SYCL training offered by a EuroCC competence centre
- Well-attended: interest in vendor-agnostic, standard-based frameworks!
- Great catalyst for European collaborations

Tricky concepts to teach effectively

- Host-device synchronization
- Potential for performance portability difficult to fit in a 2-3 day event

Call to action

More free, online training resources for self-learning are needed:

- Focus on performance analysis of SYCL applications
- Analysis of performance portability aspects
- Debugging SYCL applications

Thank you!

enccs.se