Arm SVE Hackathon
February 2021
Contact: john.linford@arm.com
Welcome!

Goals and Objectives

- Introduce SVE as a tool for enhancing scientific application codes
- Equip prospective SVE programmers with performance engineering tools for SVE
- Found positive working relationships between application developers and SVE experts
- Have fun!

Content Structure

| SVE Foundations          | • VLA Programming  
                           | • SVE in Silicon   |
|---------------------------|---------------------|
| Compilers, Libraries, and Tools | • Vendor Toolchains  
                             | • Open Source       |
| Hand-tuning               | • ACLE for SVE      
                           | • Inline Assembly   
                           | • VLA vs VLS         |
| Future Tech: SME and SVE2 | • Arm Instruction Emulator |
Virtual Event 101

• **This event is being recorded.** Please be careful while sharing your screen

• Please be VOCAL! If you have a technical issue or start to fall behind **let us know!**

• Please speak up! And please mute when not speaking

  Click on this => ![Participants Icon]

  Use these =>

  ![Zoom Control Panel]

• **Upgrade zoom!** Version 5.4.0 or later is strongly recommended
## Structured Content Schedule

<table>
<thead>
<tr>
<th>Minutes</th>
<th>Topic</th>
<th>Materials</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>Welcome, Cluster First-touch, and Intro to Fujitsu A64FX</td>
<td>06_A64FX</td>
</tr>
<tr>
<td>10</td>
<td>Break</td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>The Arm HPC Ecosystem</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>Introduction to the Scalable Vector Extension (SVE)</td>
<td>05_Apps</td>
</tr>
<tr>
<td>10</td>
<td>Q&amp;A / Prep for tomorrow’s hands-on</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>Open Source SVE Compilers: GNU and LLVM</td>
<td>01_Compiler</td>
</tr>
<tr>
<td>15</td>
<td>Arm Compiler for Linux and ArmPL</td>
<td>01_Compiler</td>
</tr>
<tr>
<td>15</td>
<td>Fujitsu Compiler and Fujitsu SSL II</td>
<td>01_Compiler</td>
</tr>
<tr>
<td>10</td>
<td>Cray Compiler and Cray LibSci</td>
<td>01_Compiler</td>
</tr>
<tr>
<td>5</td>
<td>Break</td>
<td></td>
</tr>
<tr>
<td>60</td>
<td>Hands-on</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>45</td>
<td>SVE Intrinsics and Advanced Features</td>
<td>02_ACLE, 03_SVE</td>
</tr>
<tr>
<td>15</td>
<td>Arm Instruction Emulator</td>
<td>04_ArmIE</td>
</tr>
<tr>
<td>5</td>
<td>Break</td>
<td></td>
</tr>
<tr>
<td>60</td>
<td>Hands-on</td>
<td></td>
</tr>
</tbody>
</table>
Hands-on Materials
https://gitlab.com/arm-hpc/training/arm-sve-tools

Content Structure

- **01_Compiler**: Compare autovec compilers
- **02_ACLE**: SVE Intrinsics
- **03_SVE**: Low-level SVE examples
  - See PDF documentation in this directory
- **04_ArmIE**: Arm Instruction Emulator (SVE2)
- **05_Apps**: HPC application examples
- **06_A64FX**: Demonstrate Fujitsu A64FX Features
- **Slides**: These slides

Tips and Suggestions

- Examples may be taken in any order. The numbering is a suggested order.
- Many examples support multiple compilers. Type `make COMPILER=help` to see options.
- Some examples use optimized math libraries. Type `make LIBRARY=help` to see options. If no library is specified, a library will be chosen based on the selected compiler.
- Each example includes a detailed README.md that can be easily read in your web browser or terminal.
Let’s Get Started!
### Day 1

- HW Register Renaming?
  - Yes, 64+96+32 renaming registers. See [uArch manual, Page 16](#).
- Paper on A64FX perf modeling with ECM?
- Energy consumption?
  - Yes, see [PMU manual](#).
  - Events: 01e0, 03e0, 03e8

### Day 2

- More power/energy details?
  - Many thanks to Yuetsu Kodama!
  - [06_A64FX/03_energy](#)
- SVE128 vs. NEON?
  - Slide 15, *Intro to SVE*
- How to find slow code?
  - Demo: MAP with NPB
- Why won't GCC vectorize HACC with SVE?
  - GCC11 -mcpu=a64fx

### Day 3

- Send links to recordings
- Send link to Forge client