# arm

## GPU Compiler Construction at Arm

Karl Hylén, Compiler Engineer Erik Hogeman, Compiler Engineer



- About Arm and Mali
- Graphics APIs and Rasterization
- The Compiler
- Day-to-day Challenges of a GPU Compiler Engineer
- Opportunities for Students at Arm
- Wrap up and Questions

## **About Arm and Mali**



© 2017 Arm Limited

### What do we do, in Lund in Specific?



- Part of MPG Media Processing Group
- Products:
  - Mali GPUs
  - Mali VPUs (Video Processing Units)
- SW engineering
  - Including the **compiler**
  - Also hardware modelling, video driver etc.
- HW engineering
  - System design, RTL design and verification etc.
- Test development
  - Both SW and HW

Lund

### **Arm's Partnership Model**

#### SILICON PARTNERS

#### DESIGN SUPPORT PARTNERS

Comment CMarasent Aller Partie Millegass Maraon ICCor Dates ANYKA -Torodex mctx as orezedit IBH GENGICAN LIONIC stdialog unvariant MMMM alvaview Uniquify P OWNSYS SERVER SERVER CASK Internet AROADCOM BAX STI FUJITSU oct- Getto 🔍 @Memory THISILICON SE 5 COLUMN COUNT ATES KEYRSE SONAX Freescale MARY IS I MARY IN THE VISIO WALTLA HELESHERHER Magnathp\* AMOROVAS ANTEFRICE @ ANTER SMOTH EPIXIM UMC INDERK Samble 3 **DVIDIA** Ganispan Panasonio TEACN GIGODAN Streams CALICEDA THINE CAL ZTEPR SSlabTech. STILL SHARP, STITT SMT SHARP GIRD SPANSION 1 ST LADRAG MARAGE (ITHEESING Farchip. T BORSON AN HULDED TOSHER # HAWE Silicon G + MET (Viebond BCMOGB Net ZELLES EXUNX HTSAS videaritis 1 11111 WEDIATEK Wavecow Carbons A BURFORAN FIST Constant Andrea Andrea Andrea Andrea M25EGLANS CO BARLAS 5 Helian PLEATADARES Vierein Maria Atmet Atmet Splay Splay Atmet Atmet Schos XFAB hkespec || C Joitec Sto D synery & tursens Rambus Deal Base RFE Scenar D Housett Reckchip Gro 200 troon & MOUCHP # 14 15 Here and Anna Anna Selected Aperetalism THEM SHEET ON QUILDING TO STREET CONTRICT Streetint with C formos des answer Hasee Madate TCL Strendart Mark C COROS & Hasses Hassee Medale LING C BREAK - UNKLYNN, CONTONNY KS BURNING BOBITESS GARAGESS O marrie (min 1% in out) and EVELU THE THE PARTY AND THE VELLE

IZMES firent DY Intellar AA A 13 @ IELSA ar mexelen NACES I The runster norm 117 0 WETTON VIETON Star 151 Personale BRACH Coo Cadence' miter Catter dennes (m) Correl 1 1 1 1000 B. And where URir Bacacan W. ¥2525. \* Anna Dav Ell Stran OK- I - Con Q Kill BEET MARKET ALT Systematic Mar Ana AL II Concin Samo MM antis 1 ..... was plann and of the performance Antis Other Say & Augustan manys In Conserver Bookered ALDEC. M.M. PTERTON Conterns 101 1 and 9 1 August IRTE soute- seelineeren 🍲 besst amer and The CARE HER & Married Married Com . Schoris Plaf Home GAbaint Water Tear Freise Without Change CTearlies Change Hold County House Borr BERTHER BERTHER CONTRACTOR STATES CHARGES / Derade Charder Bratters AMPAGE PRANTING SHARE BE LEVELAN CONTRACT OF CONTRACT MathWorks 🛞 n 🔤 Mutroom BOTH CATLAS Realitors provide Contraction Planates CALVETO CONTRACTOR TRAINING CONTRACTOR CONTRACTOR @tyne: .... Østeretaks Grosses Colonica & Blocwork Stan Break Do on DVVDN Hedera HEONERS TO THEMELING eterres Adventes Prostomic Panetasi Stations & Isanis Constantions of And States Big ofer Heren TOPTAN (Barrier Official Octobalogie Cont and Prove Catta and Mercel is Sendary Ayean MYR Court and and same Reason C. sold with Jacoust sufficients Press MY14 Greek

Contract Della ovice operation towards the second operation Annual Constant Annual Standard Standard Standard Constant

#### SOFTWARE, TRAINING AND CONSORTIA PARTNERS

Breefern State Present MARIE & August Im 🏠 🐲 1982 artenates S & O MONT CO S Burntrook AN A honort Officient PE\_\_\_\_\_ N1 TT THE MOUNT AL AND HUONE ST 000 pocsonic. BitWise the Critical Genaley Proxama there hereit and Delena Creatie Round me DER KHRONDA Microsoft Sais BPUX ubuntu" Company prodapt 2 2 margar Linaro Contratte craftwork ACMATA 10000 (Comes 2000) whith Customer (H) 😥 unices 😫 😡 🏈 🎆 🛵 💇 🎊 🖉 🙏 🕄 🌫 👯 There were the source of the service and the service the General derester HELAGL KILC Guge Total Culture Happen to house O Profyst VITINGITE ENEA Branches diame press, Orman Strong strong Mater Axeda Gladet O COM SISC Emic SPRINT O DANKS Soferie of Lance Of Meetre MRETECH & You I Greater CO Antole Antel Control & KOREC AND Lager Frank & O Corde relation Oranger avances avances of Theme Con Cox Co D To Recomposated BUR GEERS NVIA VARIATE GALINE BUR GOTTING ST LINE E. Horn NEXLA - HISTAN OCSOUND TRUSTONS ao E Categore Cris Vical Blacocross ceptoo Textia ENPLUC Character Start & Karat & Start NORM THE CLOSE OF STATES X BOARD LINED ---- -- DEVICES SCEARAN SOLACIA SpecialCorp otelo 🖉 dilate Manager 202 NDCAL BRACKING CHIERD TAVIE AVER CINER VICKIV CHARTE DESOUND

### Arm's partners shipped nearly **15 billion chips** with ARM technology in 2015.

Over 86 billion chips accumulated over 26 years

### **Arm Offices Worldwide**



## Graphics APIs and Rasterization

arm

© 2017 Arm Limited

### **GPUs and Software**

#### GPUs are built to support specific graphics (and compute) APIs

- Hardware is thus always licensed together with a software driver, implementing the interface to the hardware through these APIs
- Example: OpenGL, DirectX, Vulkan, OpenCL
- The graphics API provides a set of C function calls that implements a technique called rasterization

## The Rasterization Pipeline

- 3D models to be rendered consist of vertices and primitives (triangles)
- Vertex stage: Transform the position of each vertex depending on the camera
- Rasterizer stage: Compute covered pixels from the resulting vertex positions
- Fragment stage: compute a color for each resulting fragment (pixel)





### **The Mali Shader Compiler**

The Mali Shader Compiler transforms **ESSL** source into binary executables for the GPU

- Compiler is just one part of a larger driver
  - Development requires cooperation with other software teams
  - Compiler is shipped together with rest of driver on mobile phones and other devices



### **Compiler Performance**

- Competitors implement the same APIs, compete on performance
- The code is almost always hot
- If the compiler perfoms badly, it will reduce the frame rate



## **The Compiler**



© 2017 Arm Limited

### **The Compiler Frontends**

#### We implement several frontends with many different configurations

- OpenGL ES Shading Language
  - Defines several shader types, versions and extensions on top of the core language
- SPIR-V (intermediate representation used in Vulkan)
  - The newest graphics frontend, feature-wise similar to OpenGL ES Shading Language
- OpenCL/Vulkan Compute Kernels
  - C/C++-style compute language

### **OpenGL ES Shading Language**

Most widely used graphics programming language for mobile

- Six shader stages
- Several extensions on top of the core language



### **OpenGL ES Shading Language Features**

### Core language is similar to C, but with some differences

- Significantly simpler (no pointers or recursive function calls)
- Provides an in/out interface for each stage in the graphics pipeline
  - In/out variables to communicate with other stages
  - Read-only memory that is shared between threads
  - Global memory that can be both read and written
- The language defines a large number of built-in functions
  - Mathematical functions, such as 'cos' and 'normalize'
  - Texture lookups, to read color values from textures
  - Atomic operations
- Available features can be configured through extensions

### **Shader Examples**



16

© 2017 Arm Limited

arm

### **Vulkan and SPIR-V**

Vulkan is the newest graphics API, and provides a slightly different approach to programmable pipeline stages

- Uses the "Standard Portable Intermediate Representation", or SPIR-V, as shader language
- An intermediate representation is typically used in compilers to represent code closer to assembly than the original source, but still high level enough to not be too HW specific
- Feature-wise very similar to OpenGL ES Shading Language
  - Same shader types
  - Supports similar built-in operations
  - Also supports extensions for added features in the future

### **SPIR-V Intermediate Language**

#### Vulkan shaders are supplied directly in IR form

- Easier to parse than OpenGL ES Shaders, but still contains many high-level operations
- Optimally code generation could be done directly on SPIR-V IR, but not possible in practice
  - No guarantee input is optimized for our HW
  - Several operations cannot necessarily be natively transformed into GPU HW instructions
- Then what are the benefits?
  - Tools in the graphics ecosystem targeting SPIR-V can be shared between different vendors.

#### %1 = OpLabel

- %2 = OpLoad %11 %samp
- %3 = OpLoad %v2f32type %out\_texcoord0
- %4 = OpImageSampleImplicitLod %v4f32type %2 %3
- %5 = OpLoad %v2f32type %out\_texcoord0
- %6 = OpVectorShuffle %v4f32type %5 %5 0 0 1 1
- %7 = OpFAdd %v4f32type %4 %6
  - OpStore %color %7
  - OpReturn
  - OpFunctionEnd

### **OpenCL**

#### General purpose compute on GPUs.

- API + programming languages for general purpose compute on GPUs
- Programming languages are based on C99 and C++
- Same API as on desktop, but different set of extensions
- SPIR-V support
- Coming up: Compute on Vulkan?
- A common application on mobile devices is image filtering

### The OpenCL C++ Kernel Language

- New in OpenCL 2.1, C++14 subset
- No virtual functions (override), no function pointers unless compile-time constant expression, no recursive calls
  - Means everything can be inlined into every entrypoint (kernel)
- Templates, lambda expressions and function overloading are available
  - Enables generic and meta programming
- Built-in vector data types similar to GLES

```
template <typename T, size_t Rows, size_t Columns>
class Matrix {
. . .
};
template <typename T, size_t Rows, size_t Columns>
Matrix<T, Rows, Columns> operator*(
    const Matrix<T, Rows, Columns>& x,
    const Matrix<T, Rows, Columns>& y) {
. . .
kernel void matrix mult(float4 *A vec,
                        float4 *B vec,
                        float4 *Res) {
  size_t ID = get_global_id(0);
  Matrix<float, 2, ,2> A = to_matrix(A_vec[ID]);
  Matrix<float, 2, ,2> B = to_matrix(B_vec[ID]);
 Res[ID] = to vector(A * B);
```

### **OpenCL Support in the Mali Compiler**

- Leverages the Clang frontend
  - Our colleague is Code Owner for OpenCL in Clang
- Mostly supported in Cambridge

## Handling Multiple APIs and Targets

Handle several APIs using IR and middle end







## Day-to-day Challenges of a GPU Compiler Engineer

arm

© 2017 Arm Limited

### **Implementing New Graphics Features**

- New features are sometimes added, for example an update to the API, new extensions or even completely new APIs.
- This requires analyzing and understanding the features, and to create a plan on how the implementation should be done.



### **Implementing New Graphics Features**

- Typically new built-in constructs in the language
  - New built-in variables or functions, texture formats etc.
- Correctness of course important, but also performance
  - Are the new features optimized for the important use cases?
  - What are these important use cases?

### **Performance Work**

- One of the biggest selling points, so quite important part of our work
- Includes both analysis and implementation work
- Implementation can be general compiler improvements or more graphics specific optimizations



arm

### Performance Work cont.

- Performance on benchmarks and apps are typically measured in FPS
- Content can include hundreds of shaders, understanding which ones to focus on is important
- Understanding general flow and bottlenecks of content is also important



27

### **Supporting New Hardware**





### **Supporting New Hardware**

#### Common changes include:

- Changes to the ISA (new instructions, slight changes to instruction, removed instructions)
  - The ISA can change even between GPUs of the same architecture
- A new hardware feature that the driver needs information from the compiler to utilize
- A new architecture with an entirely new ISA
- Collaboration with the hardware team

### **Compilation Time**

#### Since the compiler is running on user devices, reducing compilation time is important

- Normally would be running as a part of the driver, but profiling there is hard
  - Usually use a stand alone compiler program for profiling
- The open source projects we're using don't always fit our use-case
  - Sometimes we need creative solutions for how to adapt them
- Otherwise optimize like any other program
  - Profile using some tools to find problem areas
  - Read the code, see if something can be improved a lot
  - Implement and measure the result

### Working Downstream with Upstream Code

- Problem: OS projects we're using don't always fit our needs
  - Solution: Add downstream changes (not upstreamed)
- When updating to newer versions such changes can break...
  - ...but we want the new features!
- Often when working with a problem we need to consider...
  - ...maintainability
  - ...performance
  - ...compilation time
  - ...upstream friendliness



## **Opportunities for Students at Arm**



© 2017 Arm Limited

### **Opportunities**

- Internships
  - Part time during a semester or full time during summer
- Thesis
- Graduate positions
- Want to know more?
  - 10 Oct in the entrance of E-huset
  - ARKAD
  - Teknikfokus
  - Email us: karl.hylen@arm.com, erik.hogeman@arm.com

## **Questions?**

The Arm trademarks featured in this presentation are registered trademarks or trademarks of Arm Limited (or its subsidiaries) in the US and/or elsewhere. All rights reserved. All other marks featured may be trademarks of their respective owners.

www.arm.com/company/policies/trademarks

# arm