Developing Bare-Metal GPGPU Drivers From Scratch

What prevents scientists from developing own GPGPU drivers?

Marcel Lütke Dreimann Daniel Kessener

Institut für Informatik
Universität Osnabrück

12. März 2021

Motivation

- Trend: immer mehr heterogene Hardware
  - leistungsstarke GPUs
- Forschungsbetriebsysteme wollen Hardware nutzen (s. MxKernel)
- Bare-metal Ausführung benötigt Treiber
- Treiber portieren?
  - viel Code
  - Abhängigkeiten (z.B. DRM)
  - weniger Flexibilität

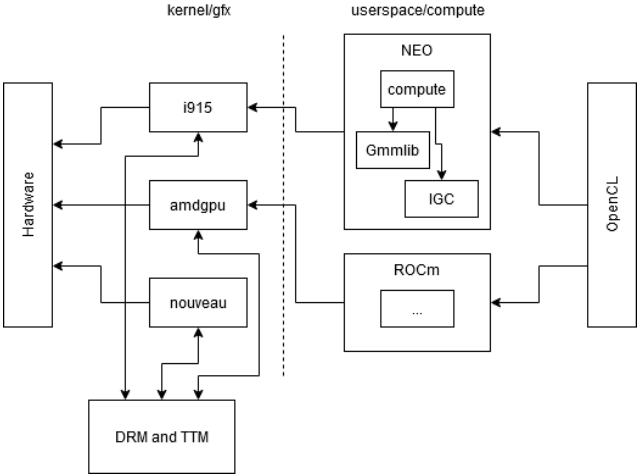
Verwandte Arbeiten

- kein Forschungsbetriebssystem mit GPGPU Treiber bekannt
- GPU Virtualisierung [7] [3]
- GPUTask Integration in Linux [6] [4]
- Treiber Erweiterungen [2]

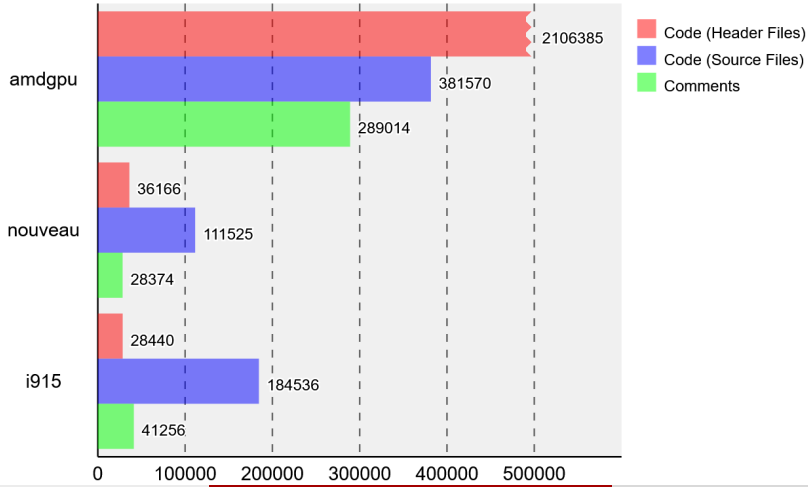
Überblick

Linux GPGPU Treiber

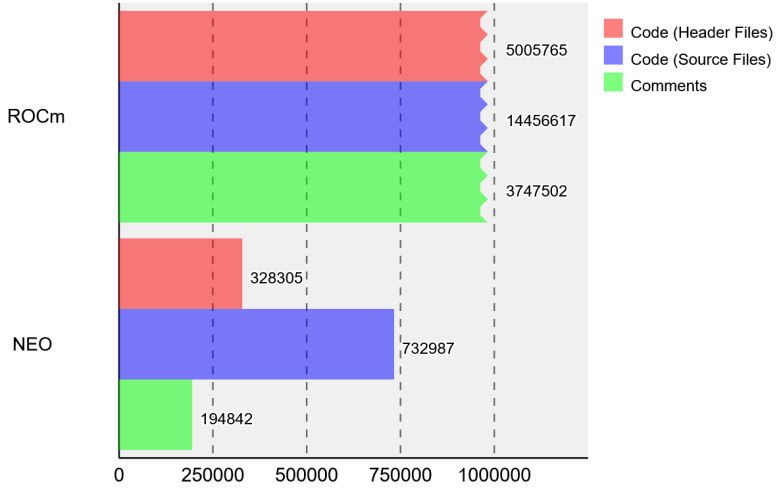
Open Source Treiber Übersicht



Kernel Module Lines of Code



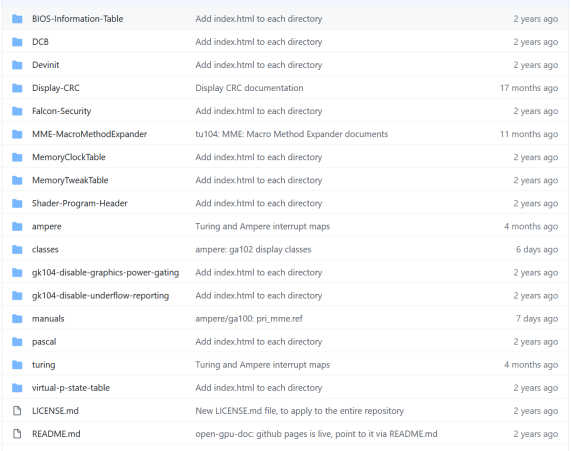
Compute Frameworks Lines of Code



Hardware Dokumentation

Nvidia

- Git Repo (https://github.com/NVIDIA/open-gpu-doc)
- txt/html/h Format mit Listen, ASCII-Zeichnungen und kleinen Erklärungen
- (Pascal), Volta, Ampere und Turing



BIOS-Information-Table	Add index.html to each directory	2 years ago
DCB	Add index.html to each directory	2 years ago
Devinit	Add index.html to each directory	2 years ago
Display-CRC	Display CRC documentation	17 months ago
Falcon-Security	Add index.html to each directory	2 years ago
MME-MacroMethodExpander	tu104: MME: Macro Method Expander documents	11 months ago
MemoryClockTable	Add index.html to each directory	2 years ago
MemoryTweakTable	Add index.html to each directory	2 years ago
Shader-Program-Header	Add index.html to each directory	2 years ago
ampere	Turing and Ampere interrupt maps	4 months ago
classes	ampere: ga102 display classes	6 days ago
gk104-disable-graphics-power-gating	Add index.html to each directory	2 years ago
gk104-disable-underflow-reporting	Add index.html to each directory	2 years ago
manuals	ampere/ga100: pri_mme.ref	7 days ago
pascal	Add index.html to each directory	2 years ago
turing	Turing and Ampere interrupt maps	4 months ago
virtual-p-state-table	Add index.html to each directory	2 years ago
LICENSE.md	New LICENSE.md file, to apply to the entire repository	2 years ago
README.md	open-gpu-doc: github pages is live, point to it via README.md	2 years ago

Nvidia

```
#define NV_PDISP_FE                                 0x00615FFF:0x00610000 /* RW--D */
#define NV_PDISP_HEADS                                                8 /*       */
#define NV_PDISP_SORS                                                 8 /*       */
#define NV_PDISP_PIORS                                                4 /*       */
#define NV_PDISP_MAX_HEAD                                             4 /*       */
#define NV_PDISP_MAX_DAC                                              0 /*       */
#define NV_PDISP_MAX_SOR                                              4 /*       */
#define NV_PDISP_MAX_PIOR                                             3 /*       */
#define NV_PDISP_CHANNELS                                            84 /*       */
#define NV_PDISP_CHN_NUM_CORE                                         0 /*       */
#define NV_PDISP_CHN_NUM_WIN(i)                                 (1+(i)) /*       */

// ...

Each define in the .ref file has a 5 field code to say what kind of define it is: i.e. /* RW--R */
The following legend shows accepted values for each of the 5 fields:
  Read, Write, Internal State, Declaration/Size, and Define Indicator.

  Read
    ' ' = Other Information
    '-' = Field is part of a write-only register
    'C' = Value read is always the same, constant value line follows (C)
    'R' = Value is read


  Write
    ' ' = Other Information
    '-' = Must not be written (D), value ignored when written (R,A,F)
    'W' = Can be written
```

For Fermi through Pascal GPUs, the CRC memory layout is organized like this:

```
    struct crc {
        uint32_t status;
        uint32_t reserved0;
        struct crc_entry {
            uint32_t status
            uint32_t compositor_crc;
            uint32_t primary_output_crc;
            uint32_t secondary_output_crc;
        } entries[255];
    }
```

The status 32-bit value is organized like this:

```
     31                                                          0
    .-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-.
    |     COUNT     |0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0| | | | | | | |
    `-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-'
```

```
/**
 * This CRC notifier structure is complete, i.e. all requested CRC's have been written and the
 * CRC notifier context DMA has been changed or made NULL_HANDLE.
 */
#define NV907D_NOTIFIER_CRC_1_STATUS_0_DONE                                   0:0
#define NV907D_NOTIFIER_CRC_1_STATUS_0_DONE_FALSE                             0x00000000
#define NV907D_NOTIFIER_CRC_1_STATUS_0_DONE_TRUE                              0x00000001
```

AMD

- Reference Guides
  - enthalten Register Definitionen
  - Umfang: ca. 250 Seiten PDF
- Programming Guides
  - enthalten Informationen zur Funktionsweise
  - Umfang: 288 Seiten PDF
- veraltet (2010 - 2013)
- unvollständig: es fehlen ganze Kapitel
- Mail: gpudriverdevsupport@amd.com

AMD Revision 1.5 June 8, 2010

11. Registers

11.1 Command Processor Registers

CP:CP_CSQ2_STAT · [R] · 32 bits · Access: 8/16/32 · MMReg:0x7fc			
DESCRIPTION: *(RO) Command Stream Indirect Queue 2 Status*			
Field Name	Bits	Default	Description
CSQ_WPTR_INDIRECT	9:0	none	Current Write Pointer into the Indirect Queue. Default = 0.
CSQ_RPTR_INDIRECT2	19:10	none	Current Read Pointer into the Indirect Queue. Default = 0.
CSQ_WPTR_INDIRECT2	29:20	none	Current Write Pointer into the Indirect Queue. Default = 0.

CP:CP_CSQ_ADDR · [W] · 32 bits · Access: 8/16/32 · MMReg:0x7f0			
DESCRIPTION: *(WO) Command Stream Queue Address*			
Field Name	Bits	Default	Description
CSQ_ADDR	11:2	none	Address into the Command Stream Queue which is to be read from. Used for debug, to read the contents of the Command Stream Queue.

Intel

- Programmer's Reference Manual (PRM)
  - enthalten grundlegende Informationen
  - Funktionsweisen und Definitionen für Register, Tabellen und Strukturen
  - höhere Ebene, wenig Implementierungsdetails
  - Umfang: 7020 Seiten PDF
- erhältlich für aktuelle Generationen
- teilweise unvollständig und fehlerhaft (s. später)

Intel

RING_BUFFER_CTL - Ring Buffer Control

Register Space:	MMIO: 0/2/0
Source:	BSpec
Default Value:	0x00000000
Access:	R/W
Size (in bits):	32
Address:	0203Ch-0203Fh
Name:	Ring Buffer Control
ShortName:	RING_BUFFER_CTL_RCSUNIT

Description	Source
These registers are used to define and operate the ring buffer mechanism which can be used to pass instructions to the command interface. The buffer itself is located in a physical memory region. The ring buffer is defined by a 4 Dword register set that includes starting address, length, head offset, tail offset, and control information. Refer to the Programming Interface chapter for a detailed description of the parameters specified in this ring buffer register set, restrictions on the placement of ring buffer memory, arbitration rules, and in how the ring buffer can be used to pass instructions.	RenderCS
Ring Buffer Head and Tail Offsets must be properly programmed before it is enabled. A Ring Buffer can be enabled when empty.	BlitterCS, VideoCS, VideoEnhancementCS
Graphics Engine doesn't go IDLE when head offset is not equal to tail offset when ring buffer is disabled.	

DWord	Bit	Description
0	31:21	**Reserved** Format: MBZ
	20:12	**Buffer Length** Format: U9-1 in 4 KB pages - 1 This field is written by SW to specify the length of the ring buffer in 4 KB Pages.Range = [0 = 1 page = 4 KB, 1FFh = 512 pages = 2 MB] Value / Name / Description: 0 / 1 page = 4 KB 1FFh / 512 pages = 2 MB
	11	**RBWait** Indicates that this ring has executed a WAIT_FOR_EVENT instruction and is currently waiting. Software can write a "1" to clear this bit, write of "0" has no effect. When the RB is waiting for an event and this bit is cleared, the wait will be terminated and the RB will be returned to arbitration.
	10	**Semaphore Wait** Description: Indicates that this ring has executed a MI_SEMAPHORE_WAIT instruction and is currently waiting for wait condition to satisfy. Software can write a "1" to clear this bit, write of "0" has no effect. Programming Notes: Writing a value of '1' will unconditionally cancel the semaphore wait on the next memory comparison. Memory comparison is triggered in signal mode on receiving a semaphore signal and in poll mode on wait timer getting expired.

Zwischenfazit

Zwischenfazit

- Dokumentation reicht oft nicht aus
  - Intel GPUs am besten dokumentiert
- intensives Befassen mit Open Source Treiber notwendig
  - mit viel Zeit verbunden, da Treiber relativ umfangreich
- zusätzliche Herausforderung: GPU Entwicklung
  - amdgpu hat ca. 670.000 loc Änderungen pro Jahr

Beispiel: Intel und AMD Treiber

Intel

- Intel UHD 6xx
- Bachelorarbeit: "Ein Treiber für die native Codeausführung auf Intel GPUs für den MxKernel"
  - ca. 4 Monate, 1 Entwickler

Intel

- Intel UHD 6xx
- Bachelorarbeit: "Ein Treiber für die native Codeausführung auf Intel GPUs für den MxKernel"
  - ca. 4 Monate, 1 Entwickler
- Problem: GPU aufwecken
  - wird nicht in Dokumentation erwähnt
  - Register in aktueller Dokumentation nicht vorhanden
  - Lösung erst duch Tracen von i915 herausgefunden

Intel: Ergebnisse

- Treiber kann GPGPU Programme ausführen
- Performance Probleme später gelöst
- Treiber ist (noch) nicht perfekt

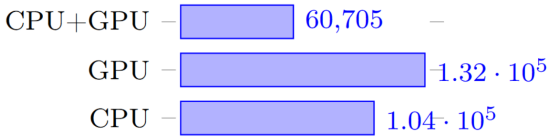


Abbildung 1: Beispiel Makespan in μs
Quelle: *[5]*

AMD

- AMD Rx570
- Teil der Projektgruppe
  - ca. 1 Jahr, 1 Entwickler + Erfahrungen von Intel

AMD

- AMD Rx570
- Teil der Projektgruppe
  - ca. 1 Jahr, 1 Entwickler + Erfahrungen von Intel
- Problem: AtomBIOS
  - Initialisierung wird teilweise von AtomBIOS übernommen
  - Ausführung benötigt eigenen Interpreter
  - erfolgreiches Ausführen schaltet VGA Modus ab

AMD

- AMD Rx570
- Teil der Projektgruppe
  - ca. 1 Jahr, 1 Entwickler + Erfahrungen von Intel
- Problem: AtomBIOS
  - Initialisierung wird teilweise von AtomBIOS übernommen
  - Ausführung benötigt eigenen Interpreter
  - erfolgreiches Ausführen schaltet VGA Modus ab
- Problem: Firmware laden
  - Benutzung von GPU Komponenten benötigt Firmware
  - binärer Blob muss in den GPU Speicher und per Register geladen werden
  - Register antwortet nicht mit ACK

AMD: Ergebnisse

- kein fertiger GPGPU Treiber
- Teile der GPU lassen sich verwenden
- Erkenntnis: Treiber für AMD GPUs schwer umsetzbar

Abschluss und Ausblick

Zusammenfassung

- Hauptproblem: mangelhafte Dokumentation
- schnelle GPU Entwicklung
- Intel GPU Treiber möglich
  - gute Dokumentation
  - einfachere Hardware
  - langsamere Entwicklung

Ausblick

- eigenen Intel Treiber Open Source veröffentlichen
  - einfach benutzbar für andere Betriebssysteme
  - https://ess.cs.uos.de/git/software/uos-intel-gpgpu
- dedizierte Intel GPUs
- Open Source GPU (ähnlich wie RISC-V Prozessor)
  - RV64X
  - MIAOW: Open Source GPGPU [1]

Quellen I

[1] Raghuraman Balasubramanian u. a. „Enabling GPGPU Low-Level Hardware Explorations with MIAOW: An Open-Source RTL Implementation of a GPGPU“. In: *ACM Trans. Archit. Code Optim.* 12.2 (Juni 2015). ISSN: 1544-3566. DOI: 10.1145/2764908. URL: https://doi.org/10.1145/2764908.

[2] Feras Daoud, Amir Watad und Mark Silberstein. „GPUrdma: GPU-Side Library for High Performance Networking from GPU Kernels“. In: *Proceedings of the 6th International Workshop on Runtime and Operating Systems for Supercomputers*. ROSS '16. Kyoto, Japan: Association for Computing Machinery, 2016. ISBN: 9781450343879. DOI: 10.1145/2931088.2931091. URL: https://doi.org/10.1145/2931088.2931091.

Quellen II

[3] José Duato u. a. „An Efficient Implementation of GPU Virtualization in High Performance Clusters“. In: *Euro-Par 2009 – Parallel Processing Workshops*. Hrsg. von Hai-Xiang Lin u. a. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, S. 385–394. ISBN: 978-3-642-14122-5.

[4] Shinpei Kato u. a. „Gdev: First-Class GPU Resource Management in the Operating System“. In: *2012 USENIX Annual Technical Conference (USENIX ATC 12)*. Boston, MA: USENIX Association, Juni 2012, S. 401–412. ISBN: 978-931971-93-5. URL: https://www.usenix.org/conference/atc12/technical-sessions/presentation/kato.

[5] Michael Müller u. a. „He..ro DB: A Concept for Parallel Data Processing on Heterogeneous Hardware“. In: 2020, S. 82–96.

Quellen III

[6] Christopher J. Rossbach u. a. „PTask: Operating System Abstractions to Manage GPUs as Compute Devices“. In: *Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles*. SOSP '11. Cascais, Portugal: Association for Computing Machinery, 2011, S. 233–248. ISBN: 9781450309776. DOI: 10.1145/2043556.2043579. URL: https://doi.org/10.1145/2043556.2043579.

[7] A. J. Younge u. a. „Evaluating GPU Passthrough in Xen for High Performance Cloud Computing“. In: *2014 IEEE International Parallel Distributed Processing Symposium Workshops*. 2014, S. 852–859. DOI: 10.1109/IPDPSW.2014.97.