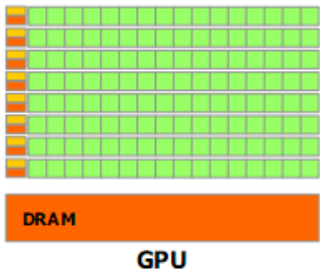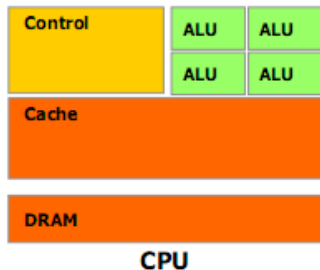# F# OpenCL C Type Provider

Kirill Smirenko, **Semyon Grigorev**

JetBrains Research, Programming Languages and Tools Lab
Saint Petersburg University

September 27, 2018

GPGPU

General purpose computations on grphical processor units

- (Almost) SIMD architecture
- Huge amount of "simple" ALUs on single chip
- May be a good choice for huge data processing

General purpose applications of GPGPU

- Initially is sientific computations
  - Phis
  - Math
  - Chem
- But more amd more general application
  - Finance/Banking
  - Bioinformatics
  - Data Analytics and Data Science (Hadoop, Spark ...)
  - Security analytics (log processing)

Problem: GPGPU <-> High level programming

- .NET, JVM, etc
- Interaction is a problem!

Existing solutions and problems

- Generative approach (haskell, Alea, etc): good byt what about code reusing
- Simple drivers (textual, for example)

Type providers

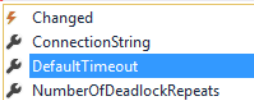our focus is .NET, so it is the way to solve our problem

Type providers

more inside, exaples of R, COM, INI providers

- Config

```
type Config = YamlConfig<"Config.yaml">
let config = Config()

config.DB.
    Changed
    ConnectionString
    DefaultTimeout          property Config.DB_Type.DefaultTimeout: System.TimeSpan
    NumberOfDeadlockRepeats
```

Brahma.FSharp

- F# quotations to OpenCL C translator
- Runtime
  - Comand queue
  - Context management
  - Memory management
  - F# aliases for OpenCL-specifc functions

OpenCL C type provider

- Imorove OpenCL C lexer, parser and translator
- Unify kernels on client side
- Improve user exp

Architecture

Diagram

Limitations

- Only (small) subset of OpenCL C
  - ▶ h files
  - ▶ preprocessor
  - ▶ subset of sintax
  - ▶ !!!
- Very simple type mapping
- !!!
- !!!

Examples

Screens

Future work

- Imorove OpenCL C lexer, parser and translator
- Unify kernels on client side
- Improve user exp

Summary

- F# OpenCL C type provider
- Prototype
- Type-safe using of existing OpenCL C code in F# applications

Contact Information

- Semyon Grigorev: s.v.grigoriev@spbu.ru
- Kirill Smirenko: k.smirenko@gmail.com

- Brahma.FSharp:
  https://github.com/YaccConstructor/Brahma.FSharp

Thakns!