School of Computer Science & Engineering

Trustworthy Systems Group

Pancake: A Language for Verified Systems Programming

Miki Tanaka

miki.tanaka@unsw.edu.au

Pancake: overview

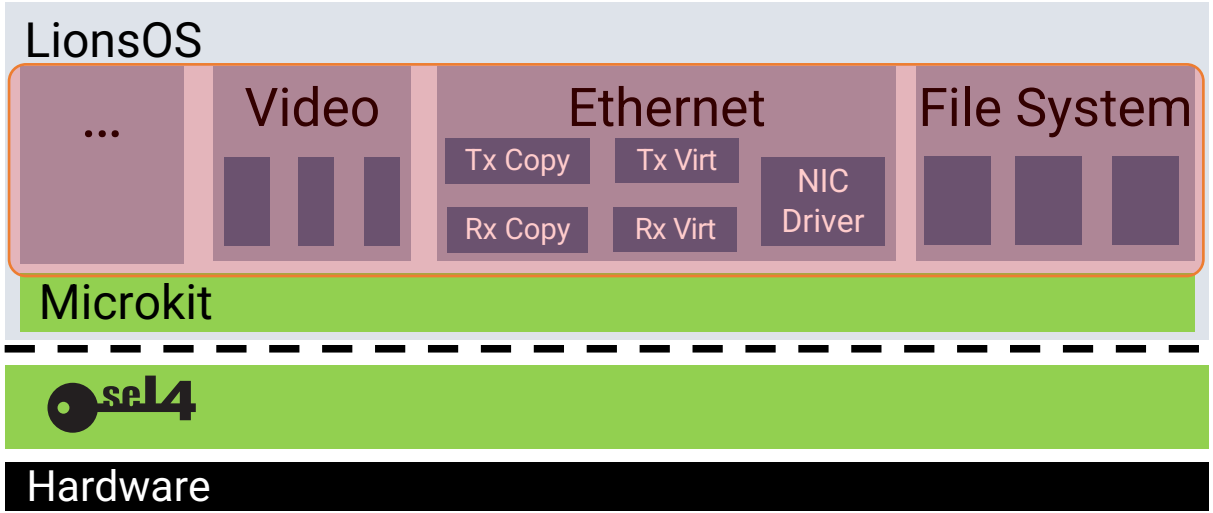
PANCAKE
A Language for
Verified Systems Programming

- **Pancake** [PLOS'23] is an in-development language for verified low-level systems programming
- With a verified compiler, developed on HOL4.
- **In particular**: it aims for verified "device drivers" and other OS components.
- **Design goals:**
  - Simple, easy to learn syntax
    -> for ease of adoption
  - Simple, easy formal semantics
    -> for ease of verification
  - Light resource footprint: no runtime, no GC
    -> for performance and verifiability

Device driver bugs are the leading cause of OS compromise, accounting for the majority of the CVEs

Pancake and seL4

Pancake aims to enhance the guarantee at this level



Pancake: contributors

PANCAKE
A Language for
Verified Systems Programming

Pancake: contributors

PANCAKE
A Language for
Verified Systems Programming

TS

Current collaborators:

Magnus O. Myreen
Thomas Qu
Halogen Truong
Joong Do Chiang
Johannes Aman Pohjola
Miki Tanaka
Michael Norrish
Tiana Tsang Ung
Charran Kethees
Adam Stucci
Ronald Chiang
Gernot Heiser
Ben Nott
Junming Zhao
Tran Dao Le
Thomas Sewell
Rob Sison
Alessandro Legnani

Previous contributors:

Hira Taqdees Syeda
Remy Seassau
Charles Lewis
Craig McLaughlin
Krishnan Winter
Ken Li
Tsun Wang Sau
Thomas Liang

Pancake : sample code

PANCAKE
A Language for
Verified Systems Programming

TS

Looks similar to C?

Pancake aims to be instantly familiar to C programmers

```
export fun handle_irq() {
  var got_char = 0;
  got_char = getchar();

  if (got_char == -1) {
    return -1;
  } else {
    while true {
        var buffer_addr = @base + 544;
        var dequeue_ret = 0;
        // 0: dequeue_avail rx
        dequeue_ret = serial_dequeue_avail(0, buffer_addr);
        if (dequeue_ret != 0) {
          // dequeue_avail ring is empty
          return -1;
        }
        var enqueue_ret = 0;
        // 0: enqueue_used rx
        enqueue_ret = serial_enqueue_used(0, got_char, buffer_addr);
        if (enqueue_ret != 0) {
          return -1;
        }
      }
  }
}
```

Pancake

PANCAKE
A Language for
Verified Systems Programming

- No types!
  - Everything is a machine word

- Simpler semantics!
  - Complete formal semantics fits in a few hundred lines of HOL4
  - Almost nothing is undefined behaviour

- With a verified compiler!
  - Semantic equality from AST to ISA levels
  - Using part of CakeML

If you are verifying your code anyway, types don't seem to help much actually

Simplifying design decisions:
- No dynamic allocation
- Single-threaded only
- No pointers to stack variables
- No side effects in expressions

CakeML

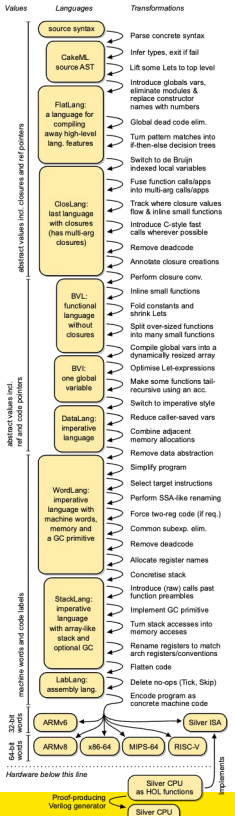
PANCAKE
A Language for
Verified Systems Programming

Pancake is an offspring of CakeML.

CakeML is an impure functional programming language similar to Standard ML

```
fun exists f xs =
  case xs of [] => False
| x::xs => f x orelse exists f xs
```

The machine code generated at the bottom has the same observable events as the source AST



...with

- optimising compiler & bootstrapped binary
- verified from AST down to the ISA level
- developed on theorem prover HOL4

CAKEML
A Verified Implementation of ML

UNSW SYDNEY

Compiling into CakeML

PANCAKE
A Language for
Verified Systems Programming

This is used for bootstrapping the binary compiler

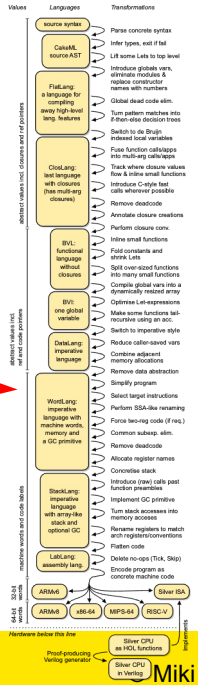
HOL (synthesis) [ICFP'12]

PureCake [PLDI'23]

Kalas [ITP'22]

Pancake [PLOS'23]

Pancake compiles into wordLang, halfway through the compiler (CakeML is too high level)



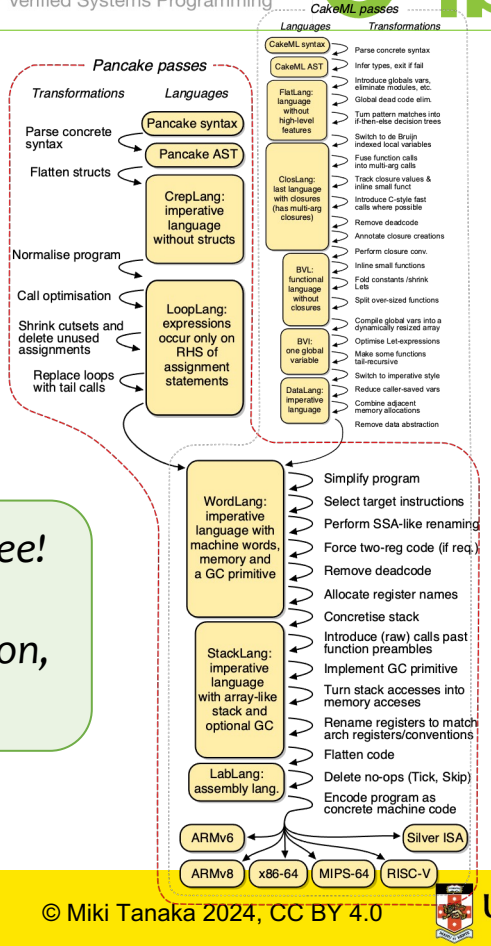
UNSW SYDNEY

Compiling into CakeML

PANCAKE
A Language for
Verified Systems Programming

- End-to-end correctness is proved for Pancake compiler
- wordLang:
  after GC is introduced, where everything's a word,
  and almost everything is unmanaged except the stack
- No GC for Pancake
- We extended Pancake with
  shared memory feature
  (but not for CakeML)

Half a verified compiler for free!
(no need to verify register allocation, instruction selection, etc.)



UNSW SYDNEY

Compiling into CakeML

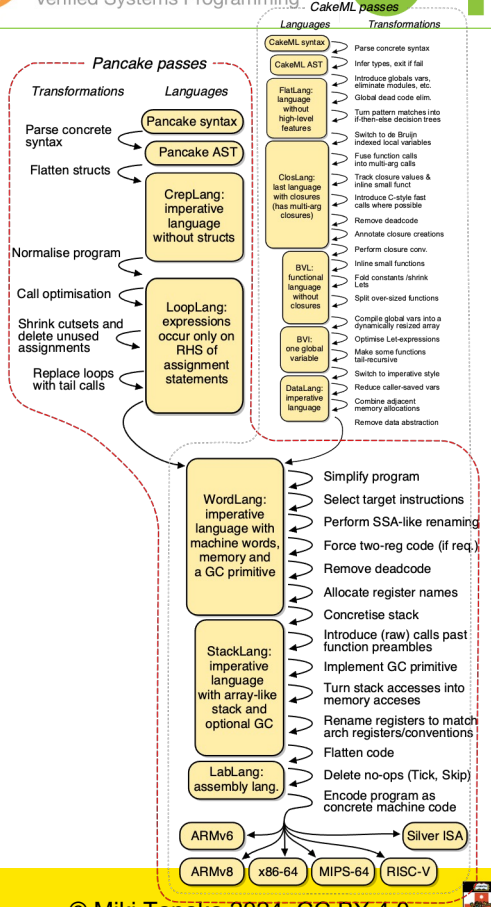
PANCAKE
A Language for
Verified Systems Programming

Bootstrapped binary for Pancake is available (as part of CakeML binary)

```
./cake --pancake
```

Pancake compiler is defined as a HOL function. Pancake binary is obtained via CakeML by the synthesis tool and in-logic compilation.

https://cakeml.org/



UNSW SYDNEY

Pancake : performance

PANCAKE
A Language for
Verified Systems Programming

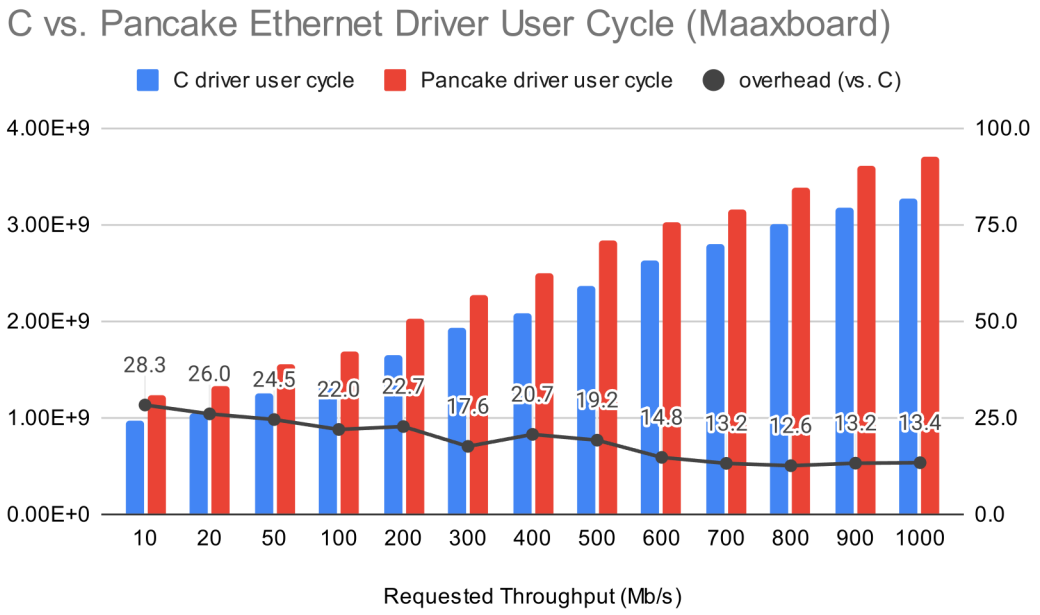
TS

Good progress in relatively short time

But still noticeable overhead.

- Reducing ffi calls
- Better code generation

Further improvement expected



UNSW SYDNEY

Pancake : semantics

PANCAKE
A Language for
Verified Systems Programming

Functional big-step semantics [ESOP'16]

- Used for compiler correctness proof (similar to CakeML)
- Deterministic
  - Not great for non-determinism
- Finite structure
  - Not great for divergence

The "state" carries a clock and an oracle:
- Divergence = as limit of (clock -> ∞)
- Evaluation uses the oracle to model ffi

Interaction tree semantics [Xia et al, POPL'20]

- Coinductive, captures interactions and continuations
  - Direct representation of divergence
  - No clock or oracle in the state*
- Semantic itree interacts with any oracle
- Soundness proved

Pancake: driver verification

PANCAKE
A Language for
Verified Systems Programming

TS

SMT:
Translation
from Pancake to Viper

PANCAKE
A Language for
Verified Systems Programming

Target:
virtualiser
serial driver
ethernet driver
....

HOL4:
Functional big-step semantics
Interaction tree semantics

x

direct verification
Hoare logic

HOL4:
Decompilation
from Pancake to HOL

Pancake

PANCAKE
A Language for
Verified Systems Programming

Thank you!

(Almost) all this stuff is open source and freely available here:
https://code.cakeml.org

Read more at:
https://cakeml.org/pancake

Hang out with us on Discord:
https://discord.gg/a8UUs6Ce6m