The Guppy Language: An Update

CPA-2013 Fringe

Fred Barnes
School of Computing, University of Kent, Canterbury

F.R.M.Barnes@kent.ac.uk
http://www.cs.kent.ac.uk/~frmb/
... at CPA-2011.

- I talked about a possible successor language to occam-pi: Guppy.
- we’re still trying to think up a better name...!

We’re still using occam-pi, of course.

- adding new things and fixing bugs as we go.

Why..?

- occam-pi is a bolt-on (kind of) to occam: and built into the existing occam compiler (circa 1990s).
- hard to add new things.
- perception issues with the name, too. :-(

Previously

Last Time ...
... at **CPA-2011**.

- I talked about a possible successor language to occam-pi: **Guppy**.
- we’re still trying to think up a better name...!

- We’re still using occam-pi, of course.
  - adding **new things** and **fixing bugs** as we go.

- **Why..?**
  - occam-pi is a bolt-on (kind of) to occam: and built into the **existing occam compiler** (circa 1990s).
  - hard to add **new** things.
  - perception issues with the name, too.  :-( 
Previously

... at **CPA-2011**.

- I talked about a possible successor language to occam-pi: **Guppy**.
- we’re still trying to think up a better name...!

We’re still using occam-pi, of course.

- adding **new things** and **fixing bugs** as we go.

Why..?

- occam-pi is a bolt-on (kind of) to occam: and built into the **existing occam compiler** (circa 1990s).
- hard to add **new** things.
- perception issues with the name, too. :-(
What We Need ... (last time)

- Preserving the **useful features** of occam/occam-pi:
  - embodiment of CSP based concurrency (though may not restrict to that alone) in the language itself.
  - strict parallel usage checks: zero aliasing.

- Preserving the **fast execution** of the resulting code:
  - no heavy run-time checks (e.g. expensive run-time typing, complex garbage collection).
  - using existing CCSP.

- Targetable at just about **any architecture** in existence:
  - by compiling (ultimately) to LLVM (low-level virtual machine).
What We Would Like … (last time)

- A language that other people would be **happy to** (and may even **want to**) use:
  - successes of Python and Go suggest indentation-based layout and concurrency are not distasteful.

- **Rapid development** – nothing overly cumbersome to program with respect to other languages:
  - need some genericity/flexibility in the type system
  - automatic ‘SEQ’ behaviour (static checks can spot likely errors)
  - may need to sacrifice some of the purity of occam to make this work..

- **Automatic mobility** (largely a compiler thing), with a couple of language hints thrown in to help the compiler when automatic static analysis gets too complex (or wrong).

- A proper ‘string’ type with Unicode support.
Current State

- Have implemented **some** of the language.
  - in the **NOCC** compiler framework (which also grew an AVR assembler recently).
- Currently generating **C code** from Guppy sources:
  - a known quantity when it comes to debugging, etc.
  - interfaces with the existing run-time system (CCSP [1]) using CIF [2].
- Recently, managed to compile and run the **commstime** benchmark!
  - ... insert live demonstration ...
Current State

- Have implemented **some** of the language.
  - in the **NOCC** compiler framework (which also grew an AVR assembler recently).
- Currently generating **C code** from Guppy sources:
  - a known quantity when it comes to debugging, etc.
  - interfaces with the existing run-time system (CCSP [1]) using CIF [2].
- Recently, managed to compile and run the **commstime** benchmark!
  - ... insert live demonstration ...
Current State

- Have implemented some of the language.
  - in the NOCC compiler framework (which also grew an AVR assembler recently).
- Currently generating C code from Guppy sources:
  - a known quantity when it comes to debugging, etc.
  - interfaces with the existing run-time system (CCSP [1]) using CIF [2].
- Recently, managed to compile and run the commstime benchmark!
  - ... insert live demonstration ...
Comparison with occam-pi

- Not as efficient, but close.
  - run-time kernel calls impose some overhead: optimised for occam-pi.
  - more memory required, e.g. commstime: 132 words for occam-pi, 434 for Guppy.
  - commstime is perhaps not a good benchmark, but not got enough compiler support for hard-core computational code yet!
- Because we go via CIF into the run-time, can (in principle) co-exist with occam-pi processes.
  - useful in various ways.
- Get it here:
  
  http://github.com/concurrency/kroc
  http://github.com/concurrency/nocc
  
  (and then you have to figure out how to make it fly, ...)
Questions?
References
