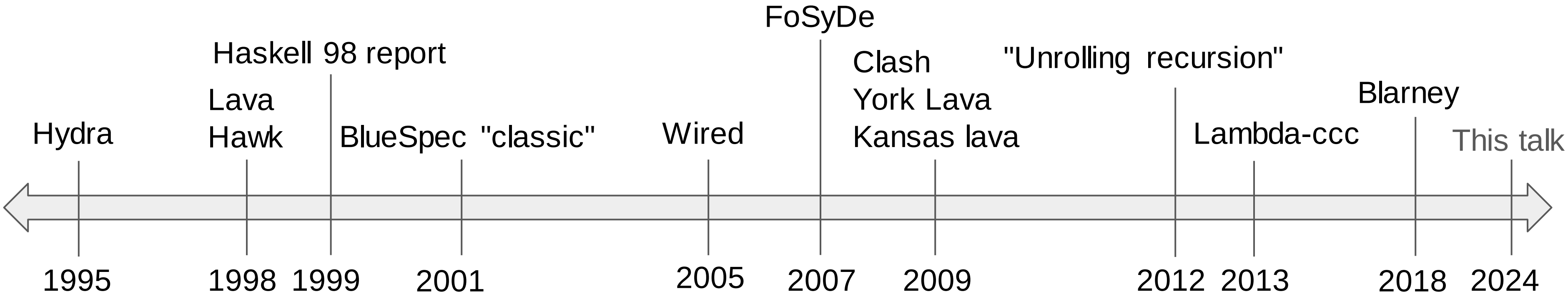
Reflections on compiling Haskell to Hardware

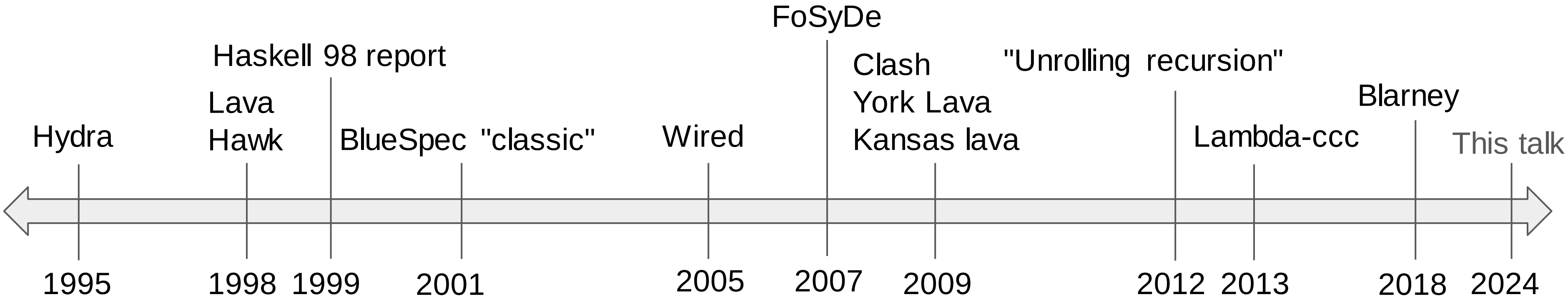
Christiaan Baaij - QBayLogic

“Clash is a functional hardware description language that borrows both its syntax and semantics from the functional programming language Haskell.”

https://clash-lang.org/

History of Haskell & Circuits





Hydra
1995

Lava
Hawk
1998

Haskell 98 report
1999

BlueSpec "classic"
2001

Wired
2005

FoSyDe
2007

Clash
York Lava
Kansas lava
2009

"Unrolling recursion"
2012

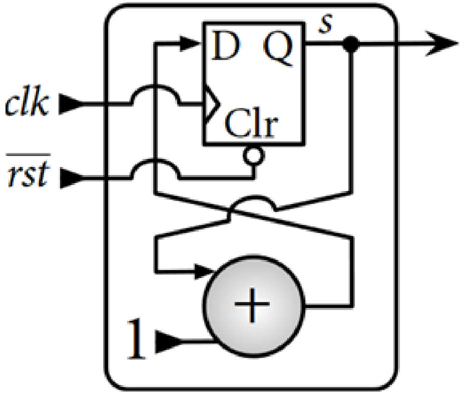
Lambda-ccc
2013

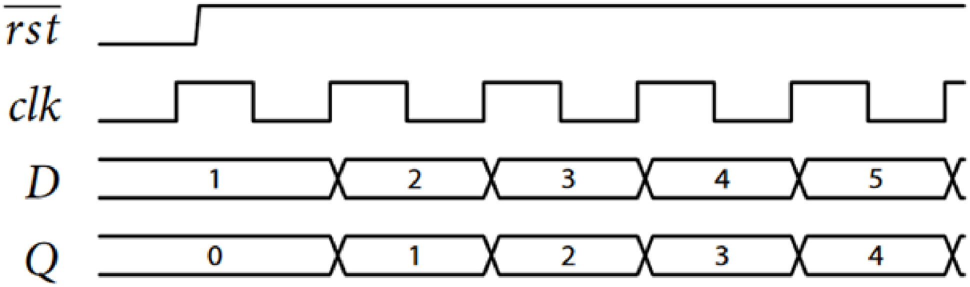
Blarney
2018

This talk
2024

	Compiled	**(Deeply) embedded**
Functions over streams	Clash ForSyDe	Hydra, Lava (all versions), Hawk, Wired
Actions	BlueSpec "classic"	Blarney
Other	"Unrolling recursion" Lamba-ccc	

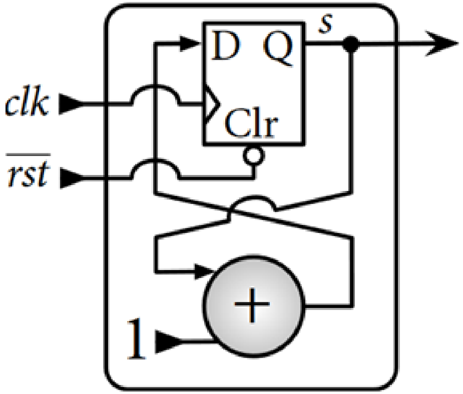
Functions over streams

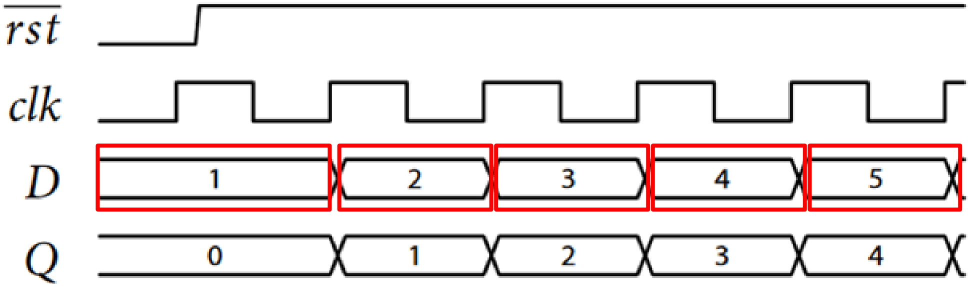




Functions over streams

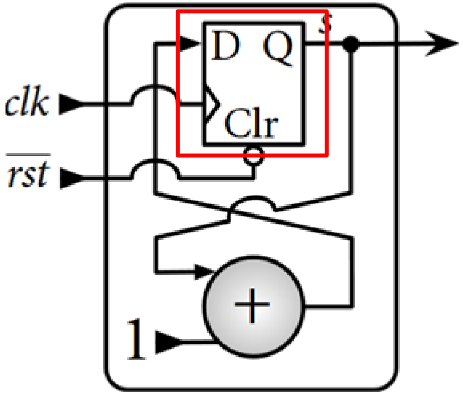
```
data Signal a = a :- Signal a
```

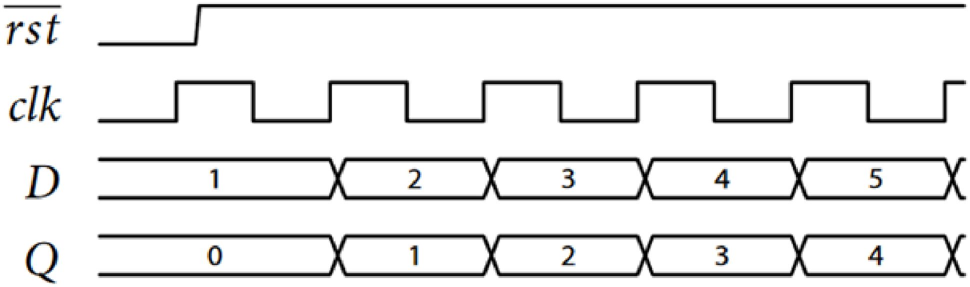




Functions over streams

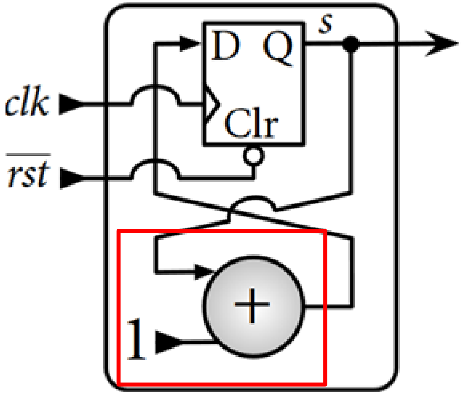
```haskell
data Signal a = a :- Signal a


register :: a -> Signal a -> Signal a
register i s = i :- s
```

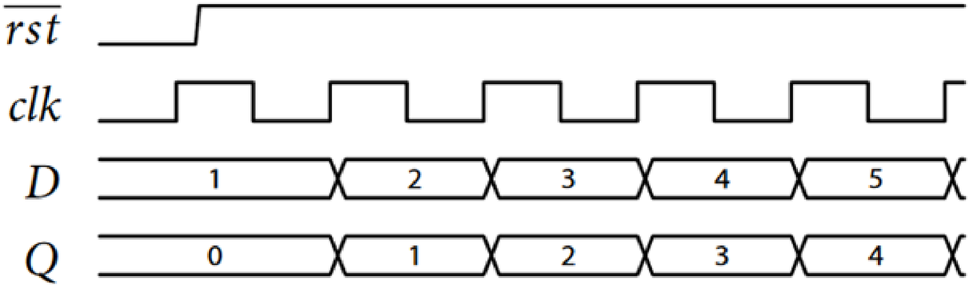




Functions over streams

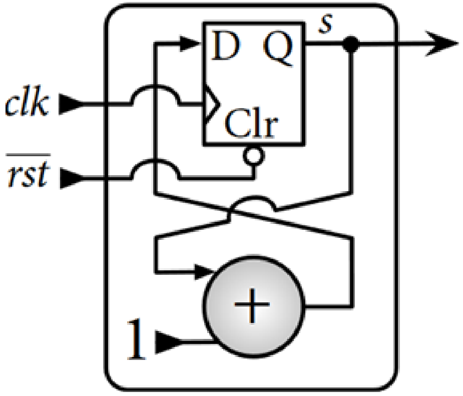
```
data Signal a = a :- Signal a


register :: a -> Signal a -> Signal a
register i s = i :- s


instance Num a => Num (Signal a) where
  (a :- as) + (b :- bs) =
    a + b :- as + bs
  fromInteger i = i :- fromInteger i
```



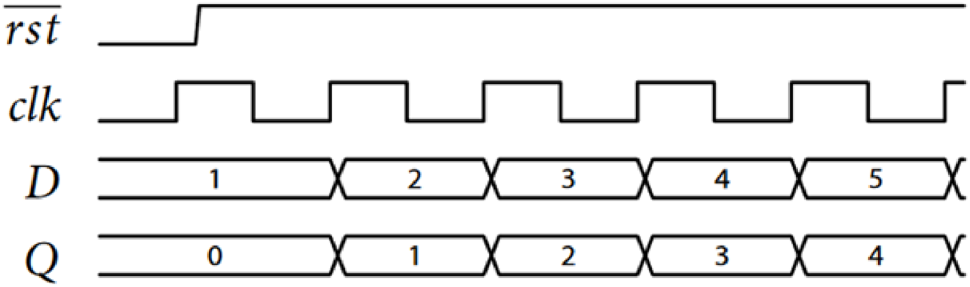


Functions over streams

```
data Signal a = a :- Signal a


register :: a -> Signal a -> Signal a
register i s = i :- s


instance Num a => Num (Signal a) where
  (a :- as) + (b :- bs) =
    a + b :- as + bs
  fromInteger i = i :- fromInteger i
```

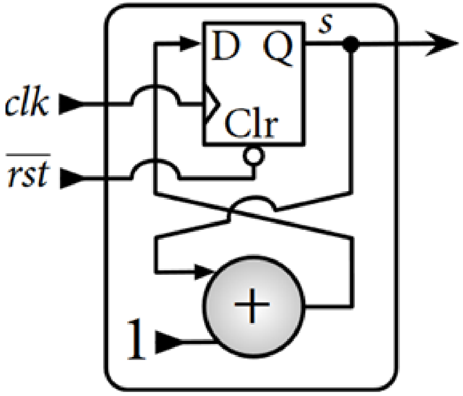


```
counter :: Signal Int
counter = s
  where
    s = register 0 (s + 1)
```

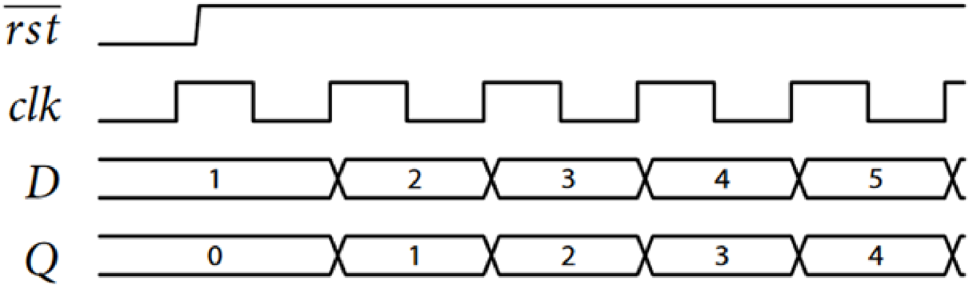


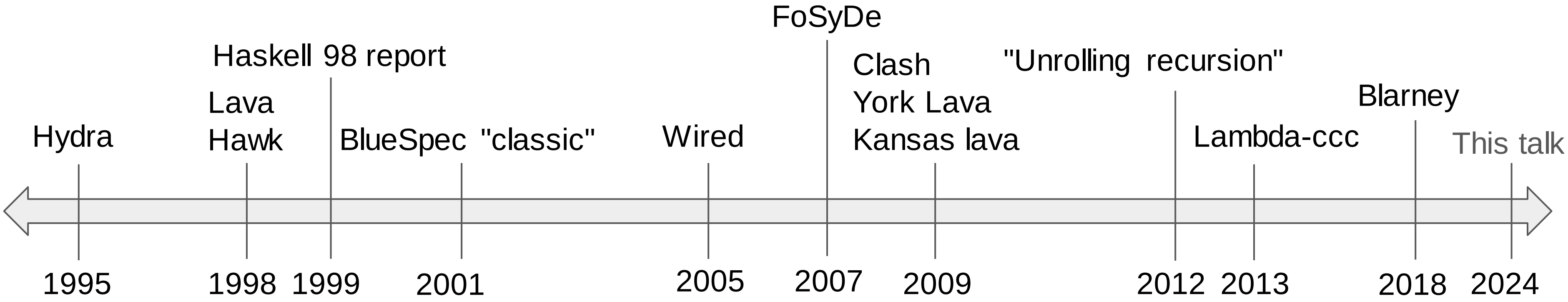
Actions

```
makeReg :: Bits a => a -> Module (Reg a)
makeReg init = ...


(<==) :: Bits a => v a -> a -> Action ()


instance Num (Bits n) where
  (+) = (.+.)
  fromInteger = constant
```



```
counter :: Module (Bits 4)
counter = do
  s <- makeReg 0

  always do
    s <== s.val + 1

  return (s.val)
```





1995 Hydra

1998 Lava
Hawk

1999 Haskell 98 report

2001 BlueSpec "classic"

2005 Wired

2007 FoSyDe

2009 Clash
York Lava
Kansas lava

2012 "Unrolling recursion"

2013 Lambda-ccc

2018 Blarney

2024 This talk

	Compiled	**(Deeply) embedded**
Functions over streams	Clash ForSyDe	Hydra, Lava (all versions), Hawk, Wired
Actions	BlueSpec "classic"	Blarney
Other	"Unrolling recursion" Lamba-ccc	

Compiling vs deep embedding

- Compiling:
  - Parse source to AST
  - Optimize, transform to intermediate form, optimize, etc
  - Pretty-print AST of final intermediate form as netlist in target HDL (Verilog/VHDL)
- Deep embedding:
  - Graph data-type to represent the netlist
  - Circuit description is an "elaborate way" to construct this data
  - Running the Haskell program builds the graph
  - Optional: optimize graph
  - Pretty-print graph as netlist in target HDL

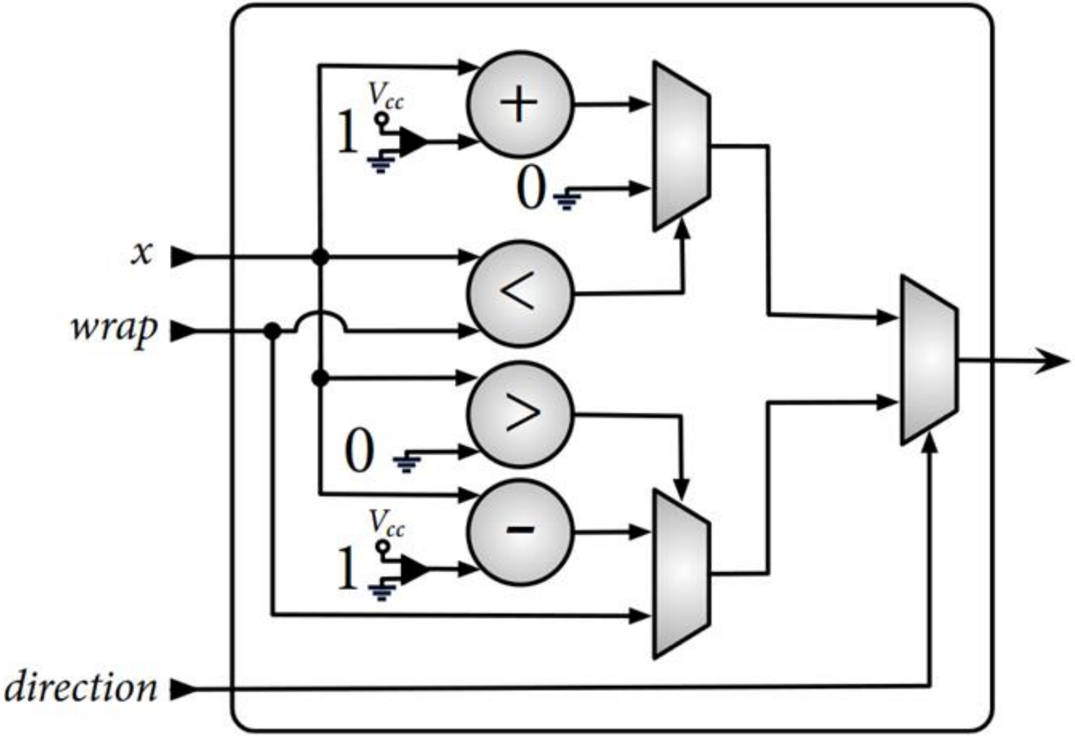
Choice

case and if-then-else expressions

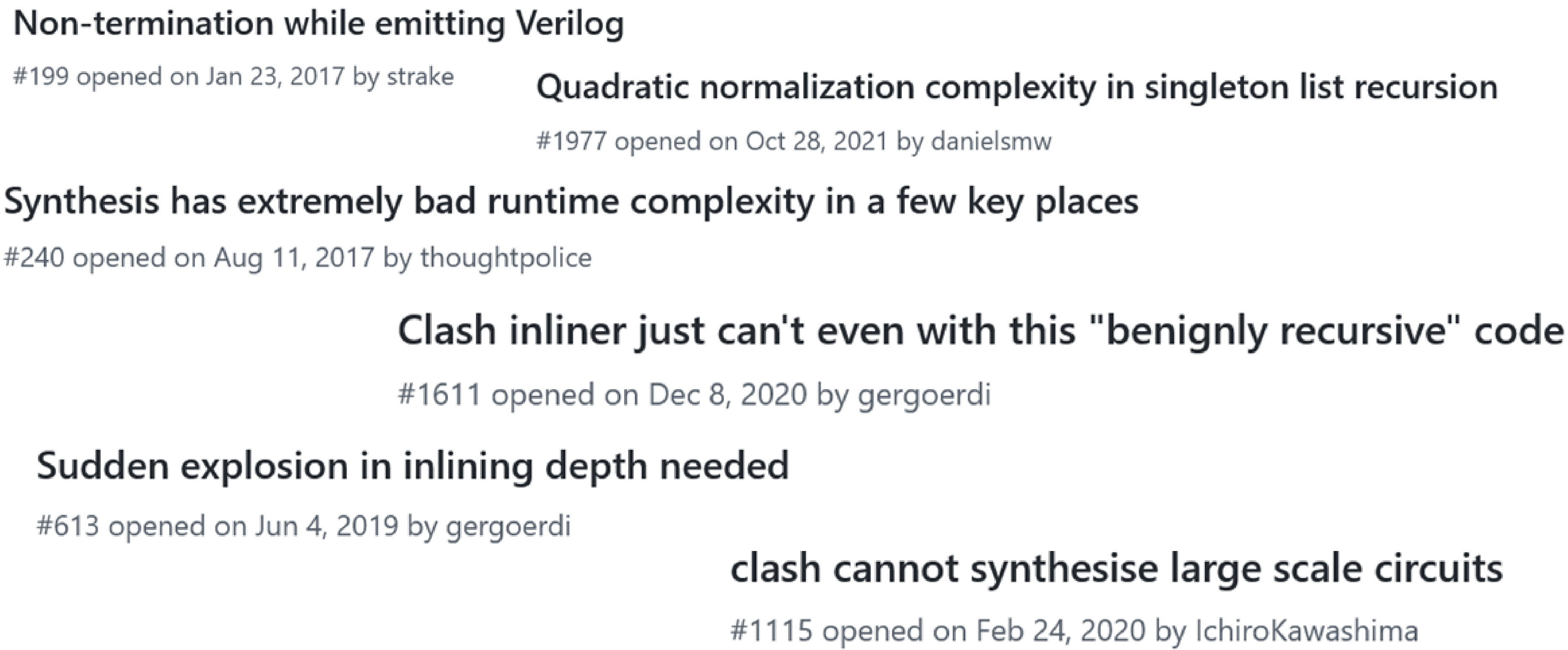
```
data Direction = Up | Down

counter wrap direction x = case of
  Up   -> if x < wrap then
             x + 1    else
             0
  Down -> if x > 0 then
             x - 1 else
             wrap
```

Pattern matching and guards

```
data Direction = Up | Down

counter wrap Up   x | x < wrap   = x + 1
                    | otherwise = 0
counter wrap Down x | x > 0      = x - 1
                    | otherwise = 0
```



Clash’s price for ADTs



Non-termination while emitting Verilog

#199 opened on Jan 23, 2017 by strake

Quadratic normalization complexity in singleton list recursion

#1977 opened on Oct 28, 2021 by danielsmw

Synthesis has extremely bad runtime complexity in a few key places

#240 opened on Aug 11, 2017 by thoughtpolice

Clash inliner just can't even with this "benignly recursive" code

#1611 opened on Dec 8, 2020 by gergoerdi

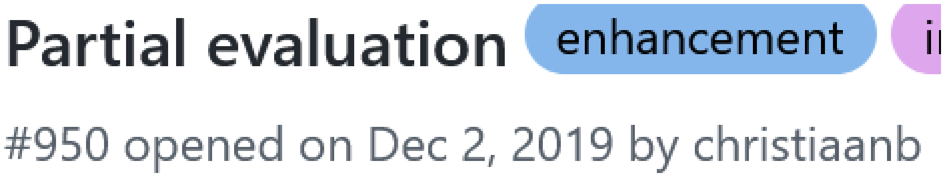
Sudden explosion in inlining depth needed

#613 opened on Jun 4, 2019 by gergoerdi

clash cannot synthesise large scale circuits

#1115 opened on Feb 24, 2020 by IchiroKawashima

Clash’s price for ADTs



Partial evaluation enhancement i

#950 opened on Dec 2, 2019 by christiaanb

https://github.com/clash-lang/clash-compiler/issues/950

Partial evaluation - what makes it interesting

- Non-strict language
- Recursive let-bindings
- Performance metric: circuit size

Relevant existing work:

- http://www.cs.ru.nl/P.Achten/IFL2013/symposium_proceedings_IFL2013/ifl2013_submission_7.pdf
- https://www.microsoft.com/en-us/research/wp-content/uploads/2016/07/supercomp-by-eval.pdf
- https://www.researchgate.net/publication/220989852_Taming_code_explosion_in_supercompilation

Meta-programming circuits

- https://github.com/SpinalHDL/VexRiscv
  There are very few fixed things. Nearly everything is plugin based. The PC manager is a plugin, the register file is a plugin, the hazard controller is a plugin, ..
- https://github.com/blarney-lang/pebbles
  Pebbles is a RISC-V processor framework supporting *plugable pipelines*.
- https://github.com/enjoy-digital/litex
  The LiteX framework provides a convenient and efficient infrastructure to create FPGA Cores/SoCs, to explore various digital design architectures and create full FPGA based systems.

Clash in the wild

Demcon FOCAL & QBayLogic

Laser-based satellite communication

- Terabit-per-second optical communication between satellites and ground station
- FPGA implementation for calculating the steering vector adjustments for an array of mirror segments

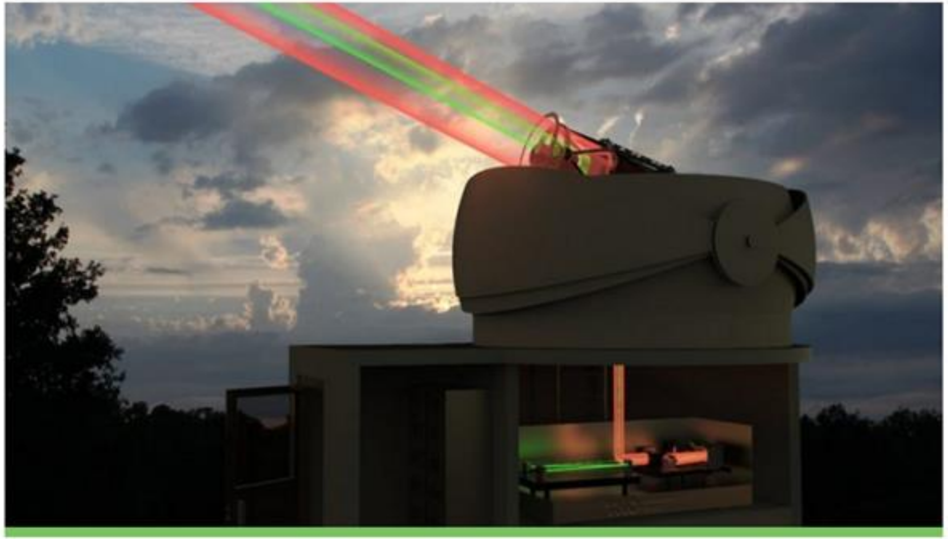


Figure 1: Within the Tomcat project, an optical ground station, including an optical ground terminal, for laser satellite communication is being developed.

Clash in the wild

Myrtle.ai
Real-Time Text-to-Speech Synthesis

- 256 simultaneous text-to-speech streams
  8x faster than state-of-the-art
- Based on WaveNet:
  Wavelets + neural networks
- First 3rd party application on Intel’s new AI-optimized FPGA

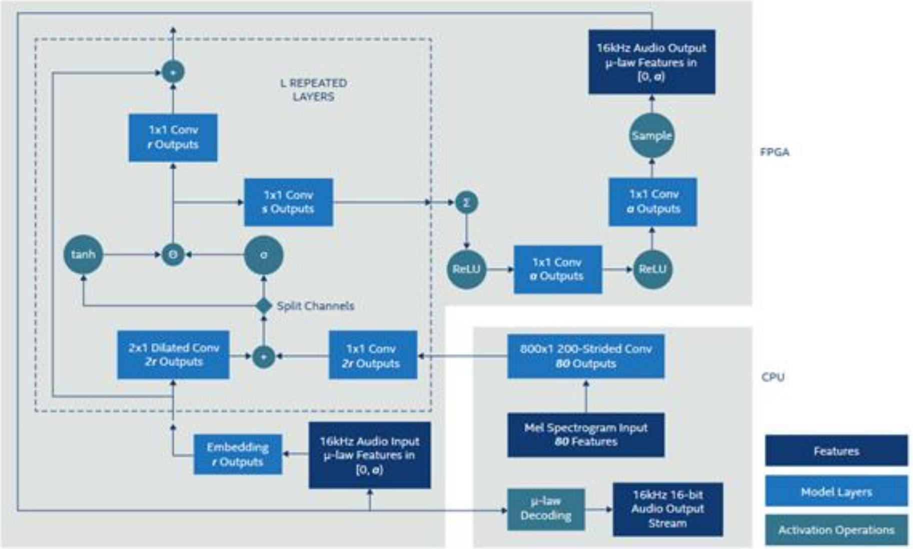


Figure 1. The WaveNet architecture where the number of skip, residual and audio channels is parameterised by **s, r, a** and the number of layers by **L.**

What would I personally chose when (in 2024)

Compiled language

- (Relatively) fixed functionality
- State machines*

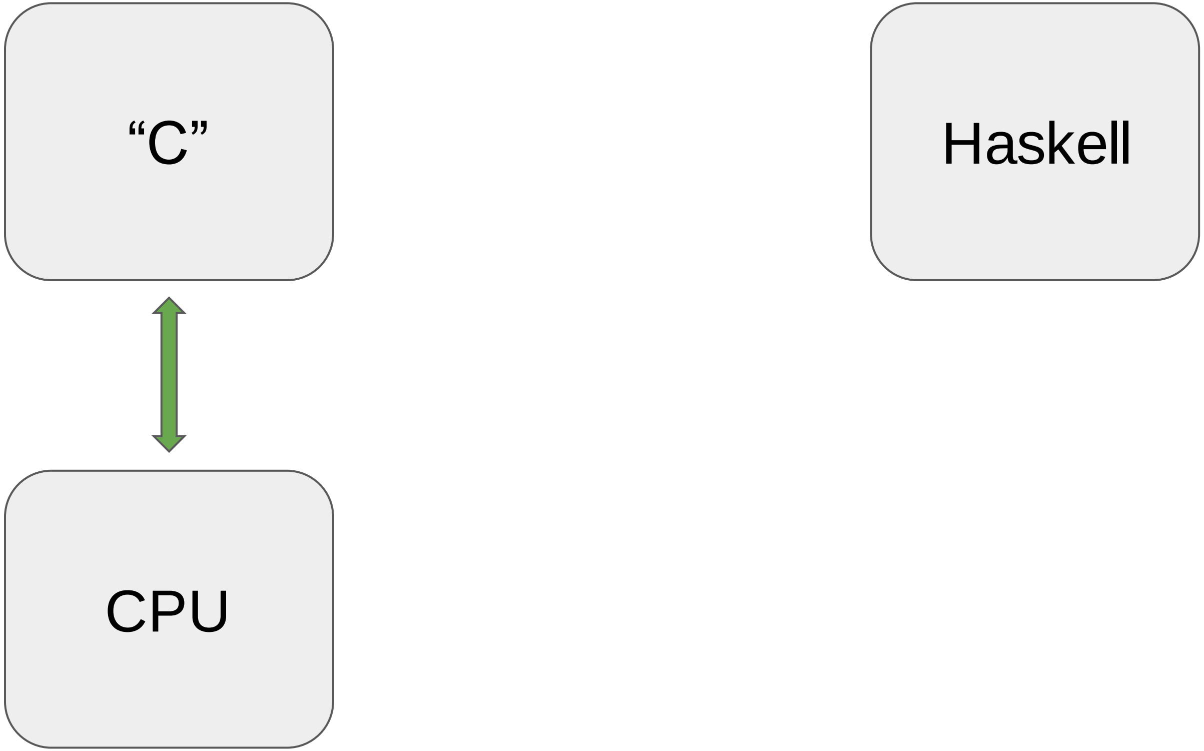
Embedded language

- Design that’s highly reconfigurable in terms of functionality

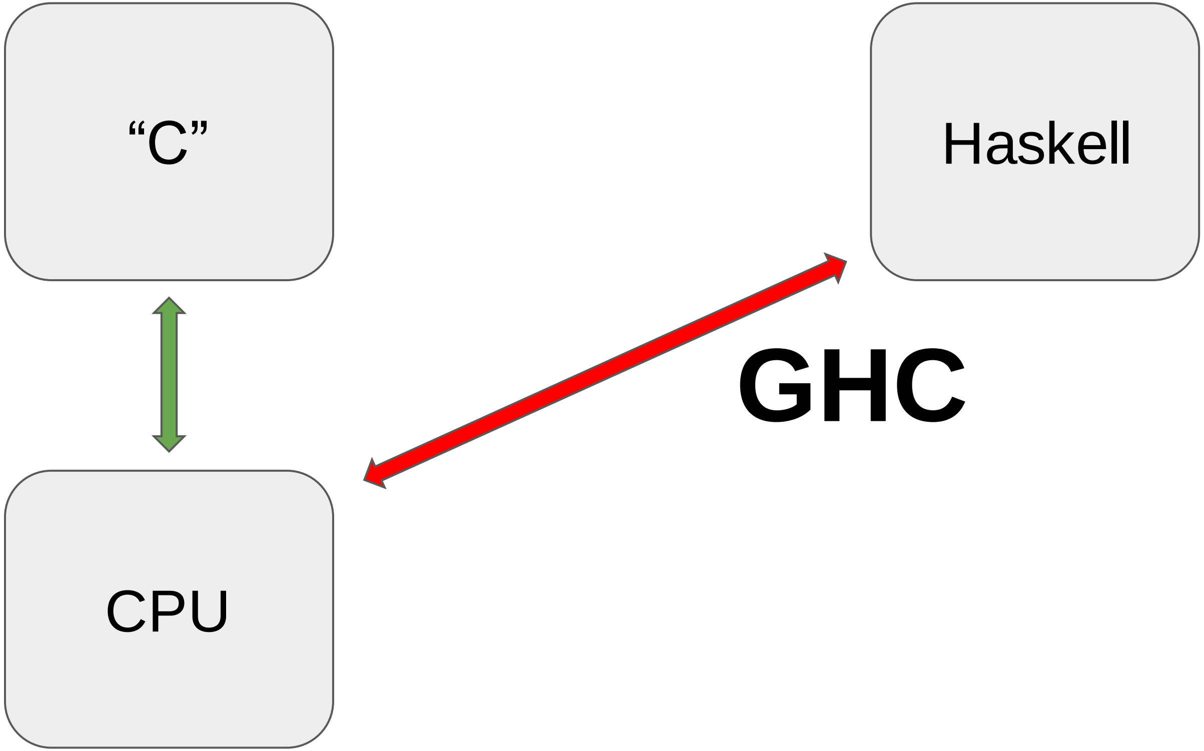
FPGAs are becoming mainstream

- PCIe accelerator cards
  - ~~Xilinx~~ AMD Alveo
  - ~~Intel~~ Altera PAC
- Cloud instances
  - AWS F1
  - Microsoft Azure NP
  - Nimbix
- Embedded in SSDs
  - Samsung SmartSSD

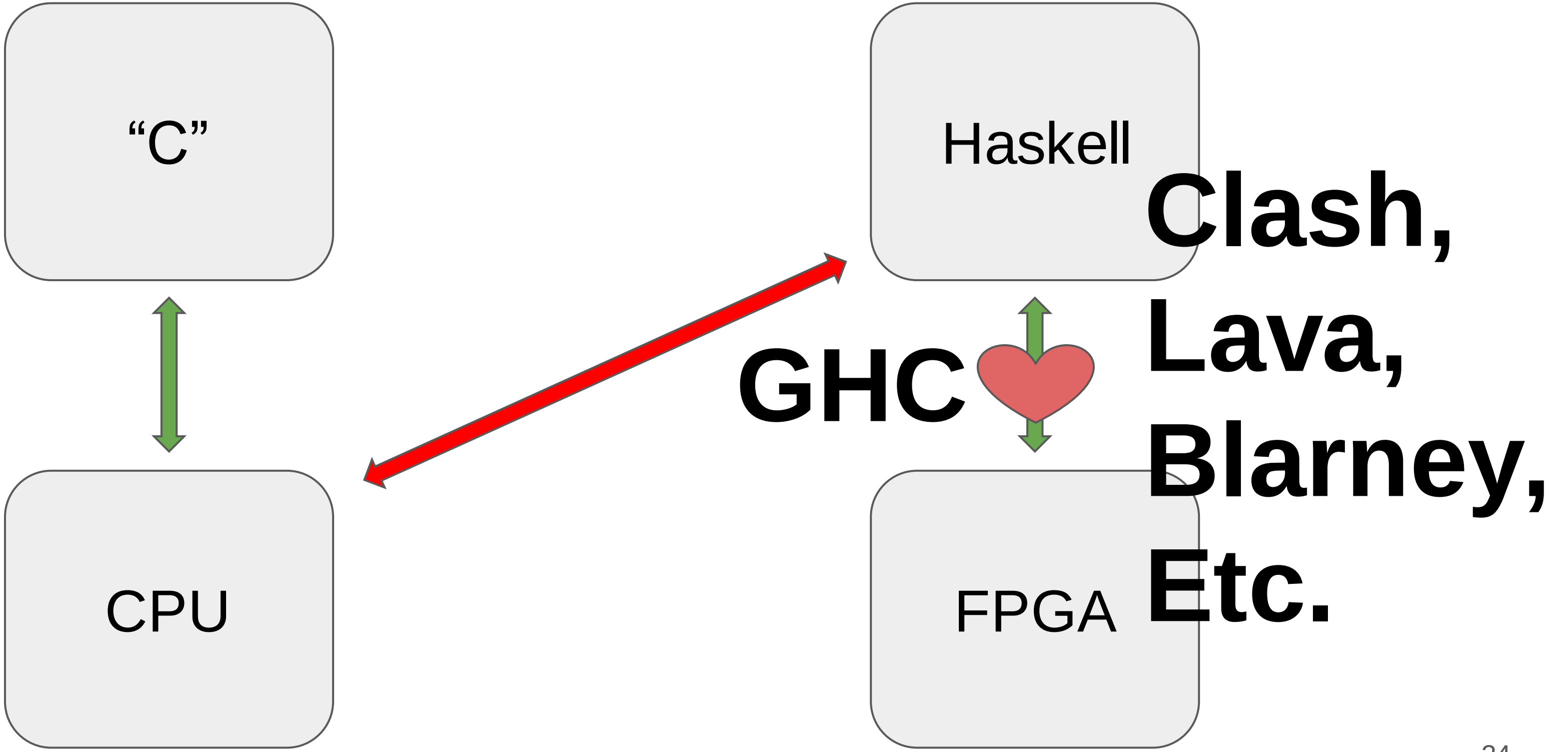
“High performance code is imperative code”



The Herculean effort



Functional high-performance Haskell



Clash,
Lava,
Blarney,
Etc.

https://clash-lang.org/install

https://retrocla.sh/