Mapping CSP Models to Hardware using CLaSH

Frist Kuipers, Rinse Wester, Jan Kuper & Jan Broenink
University of Twente, Enschede

2016, August 23

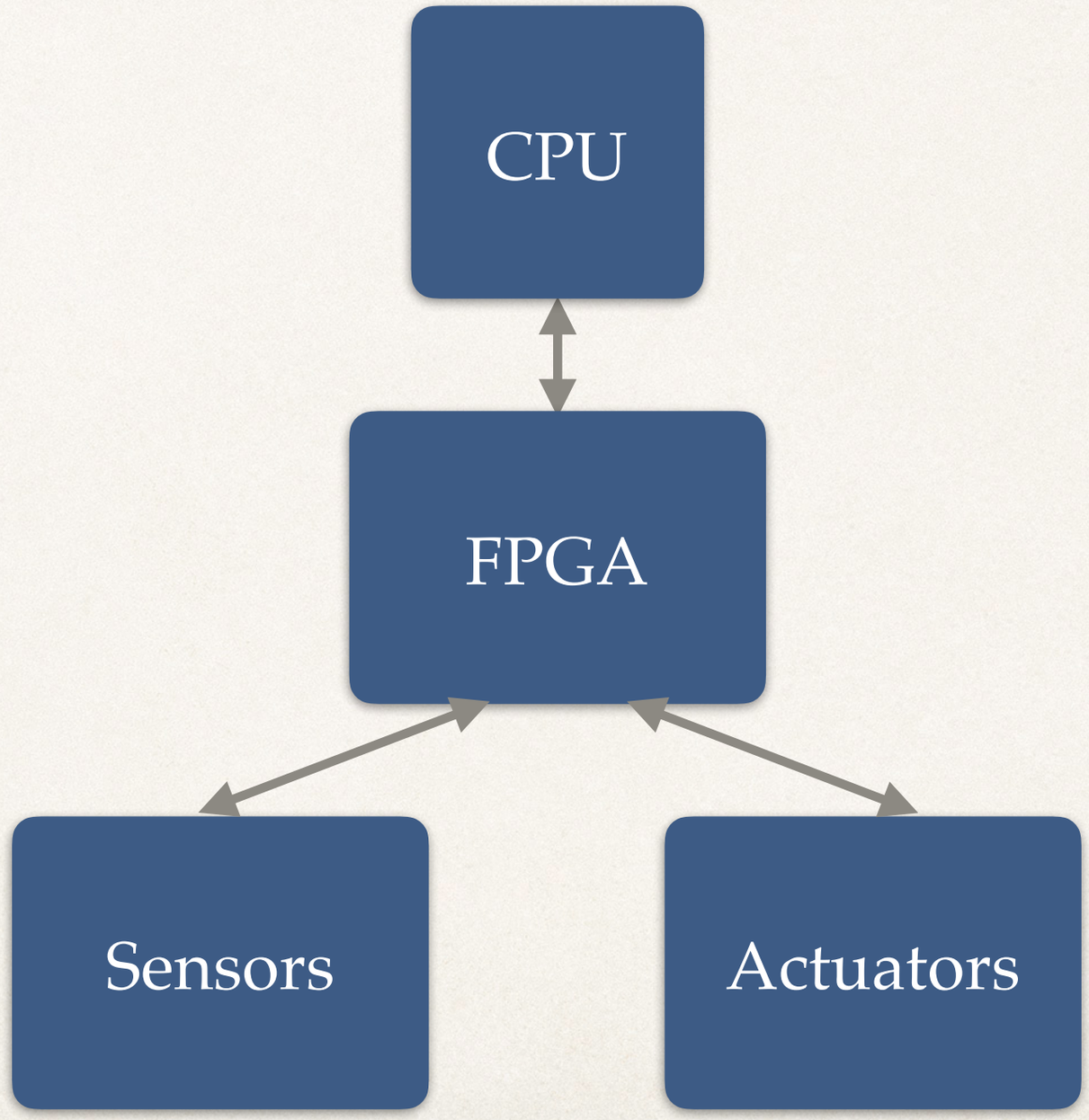
Contents

- Introduction
- CLaSH
- CSP constructs
- Results
- Conclusions

Introduction

- Embedded system design more complicated
- Increase in number of requirements
- Model-Driven Design (MDD)

Introduction

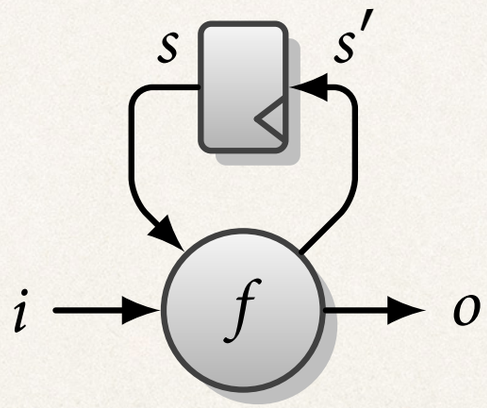


CLaSH

- Functional Hardware Description Language (Haskell)
- Structural description of hardware
- Components based on Mealy-machine

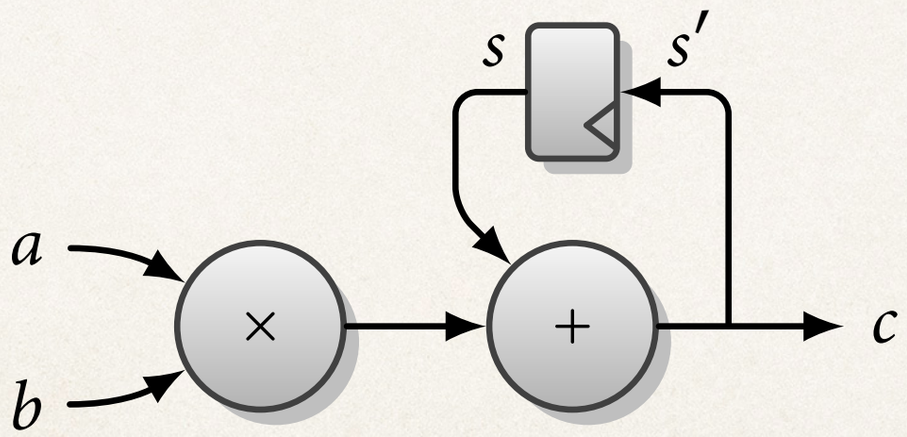
CLaSH

```
mealy s i = (s', o)
   where
      (o, s') = f s i
```



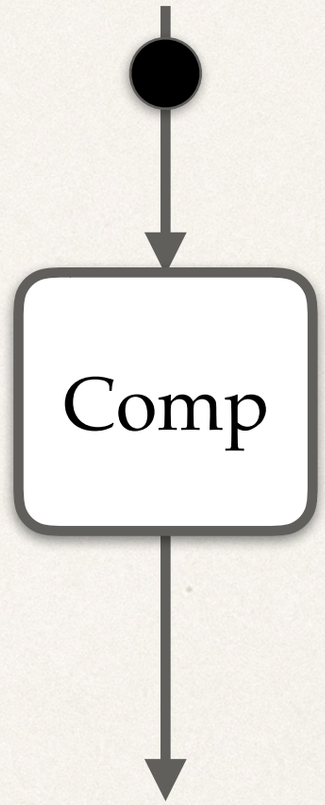
CLaSH

```
mac s (a, b) = (s', out)
   where
      output = s'
      s'     = s + a * b
```



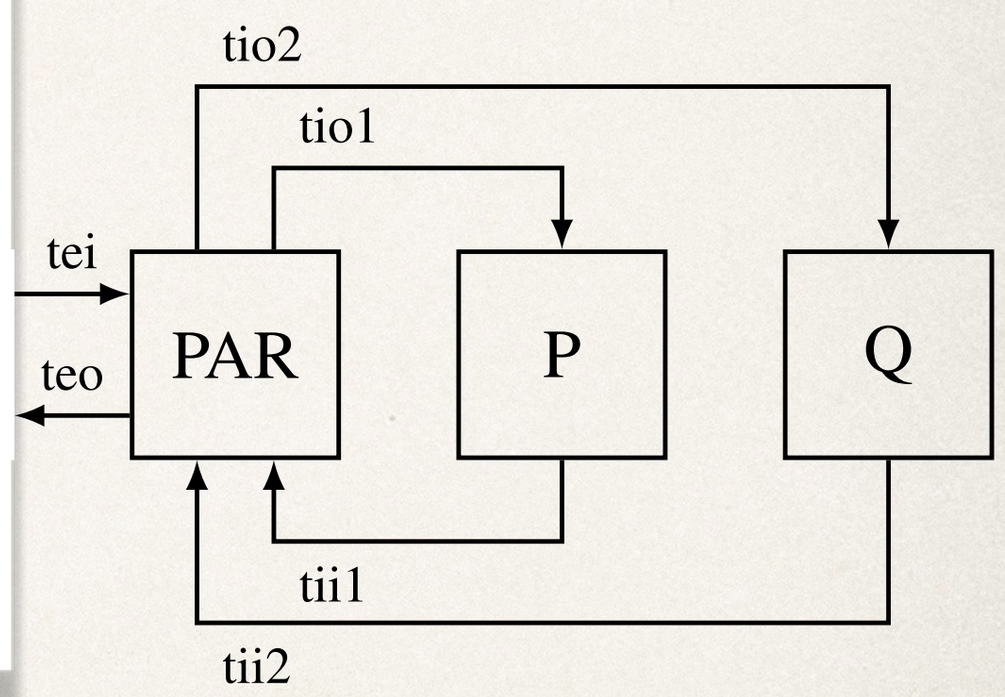
CSP constructs in CLaSH

- Components with trigger tokens
- Three basic CSP constructs
  - Parallel
  - Sequential
  - Channels



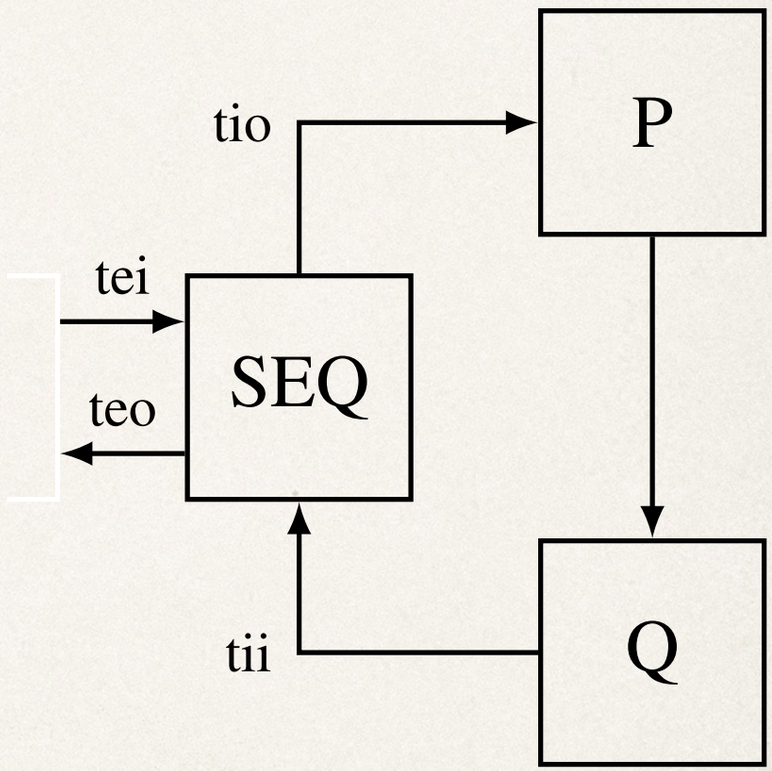
Parallel construct

```
parallel' (te, ti1, ti2) (tei, tii1, tii2) =
((tei, ti1r, ti2r), (teo, tio1, tio2))
  where
    -- Return token when both are received
     teo = ti1 && ti2

    -- Only consume token one when both received
    ti1r = ti1 && ti2
    ti2r = ti1 && ti2

   -- Return token to both structures in parallel
   tio1 = te
   tio2 = te
```



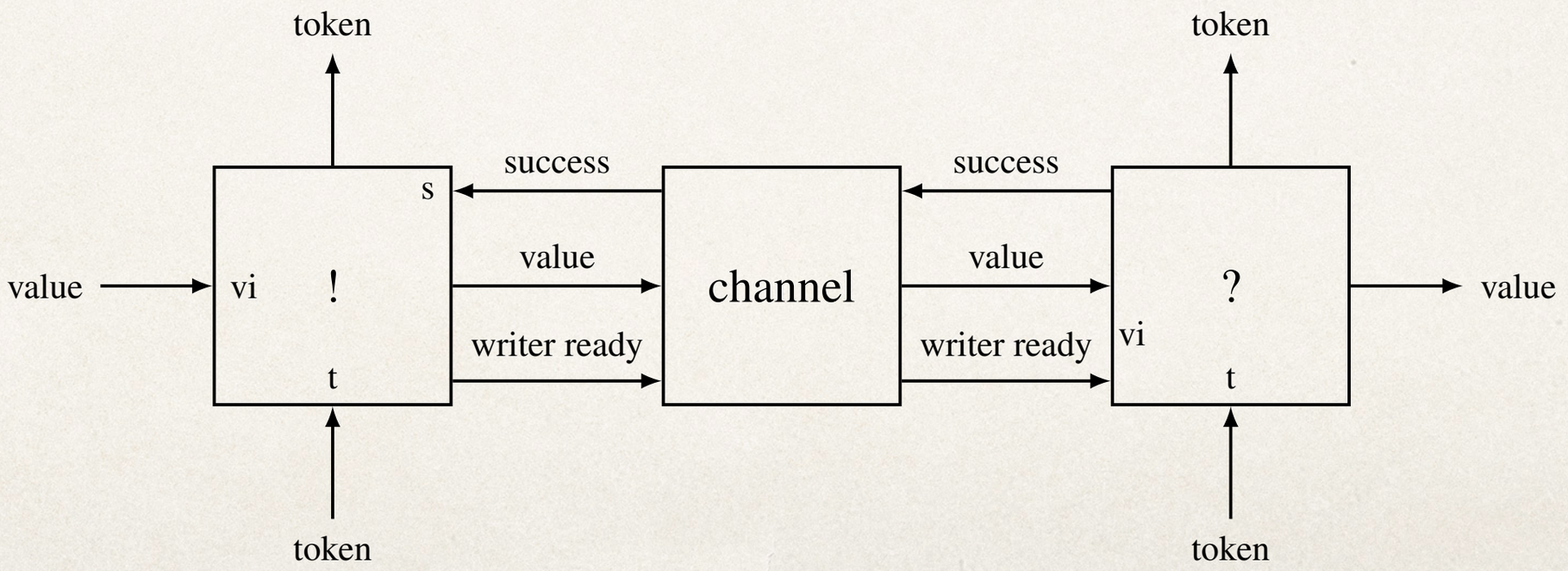
Sequential construct

```
sequential tei tii = (teo, tio)
  where
    teo = register False tii
    tio = register False tei
```

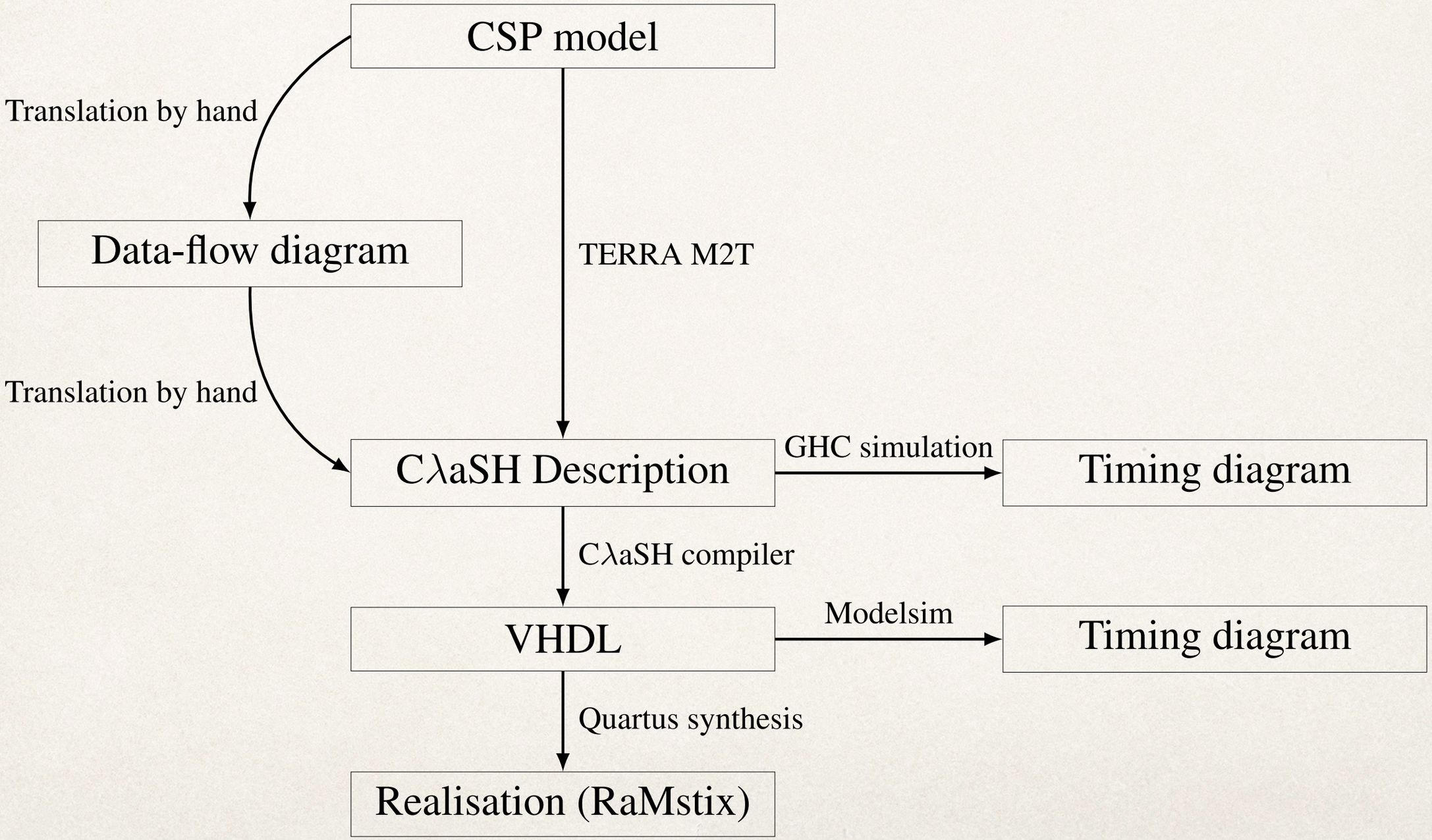


Channels

```
circuit (val_in, tkn_in_writer, tkn_in_reader) = (val_out, tkn_out_writer, tkn_out_reader)
  where
    (value, writer_ready, tkn_out_)   = writer (val_in, success, tkn_in_writer)
    (success, value, writer_ready)    = channel (success, value, writer_ready)
    (val_out, tkn_out_reader, succes) = reader (value, writer_ready, tkn_in_reader)
```



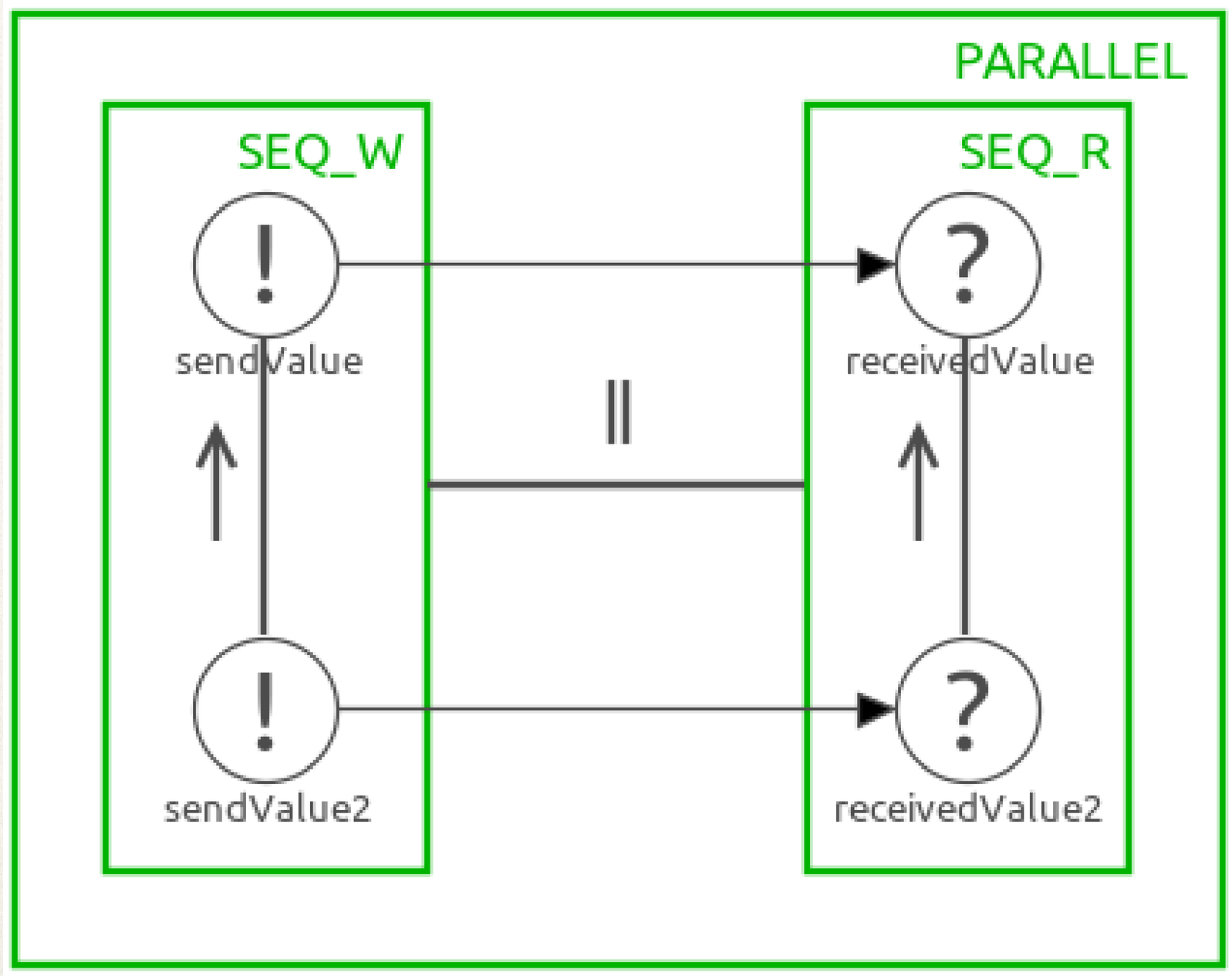
Work flow


CSP model
Translation by hand
Data-flow diagram
TERRA M2T
Translation by hand
CλaSH Description
GHC simulation
Timing diagram
CλaSH compiler
VHDL
Modelsim
Timing diagram
Quartus synthesis
Realisation (RaMstix)

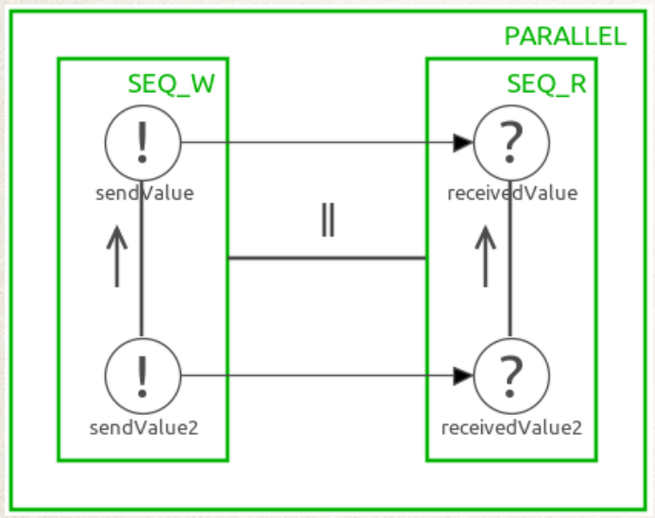
Results

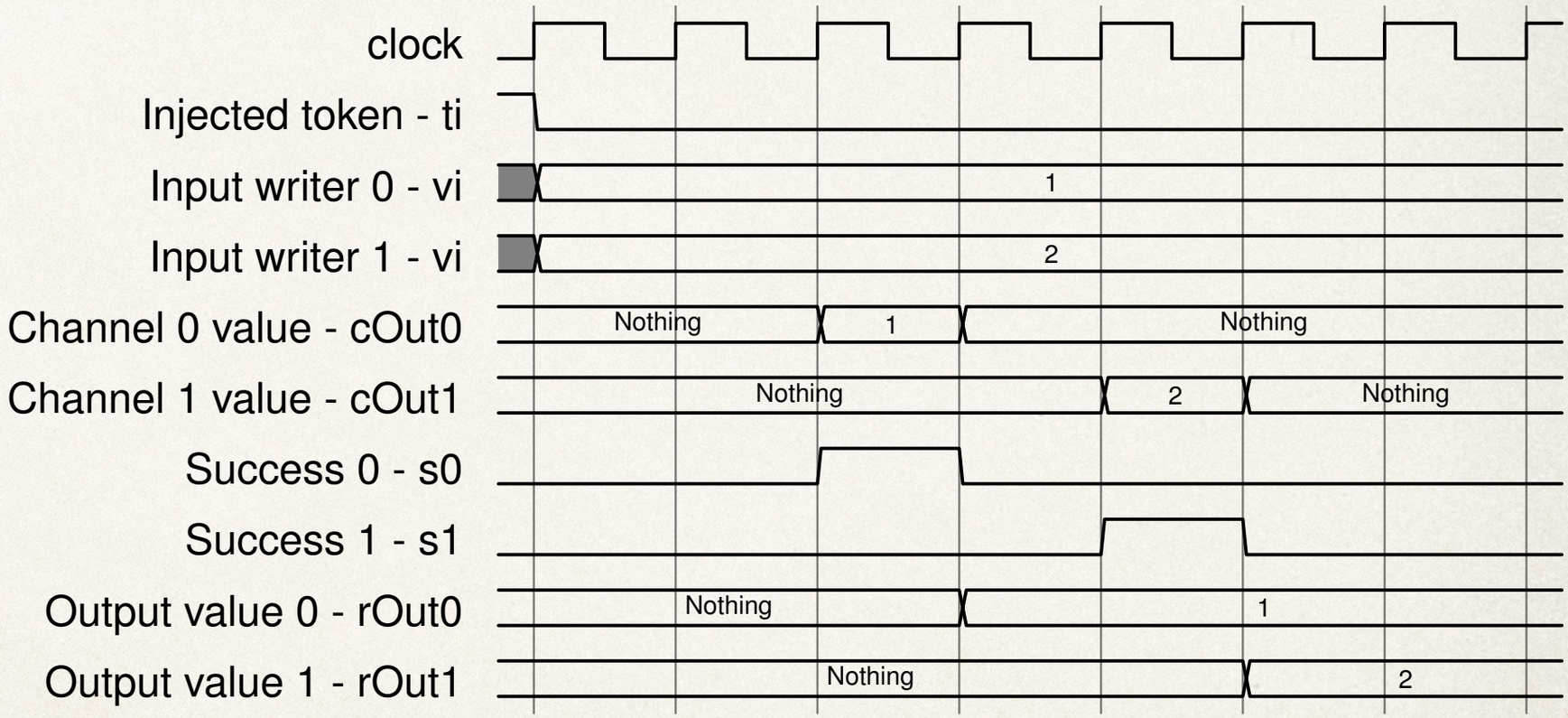
- Two examples implemented
  - Single reader and writer
  - Double reader and writer

Double reader/writer



Double reader/writer





Hardware results

Example	Logic Elements
Producer consumer	23
Double producer consumer	37

Conclusion

- Mapping for CSP to FPGA developed
- Feasibility illustrated using examples

Future work

- Integration in TERRA tool
- Support for alt construct

Questions?