Modelling of the XCore Microprocessor with Rodin

Steve Wright

Bristol University

University of BRISTOL

Department of COMPUTER SCIENCE

Or….

Life after the PhD

A word from our sponsors

- Funded by an EPSRC Knowledge Transfer Secondment (Grant EP/H500316/1)

- Hosted by XMOS Ltd

Project Timescale

October 2010 - October 2011

PhD Thesis

"Formal Construction of Instruction Set Architectures"

University of Bristol Library, 2009

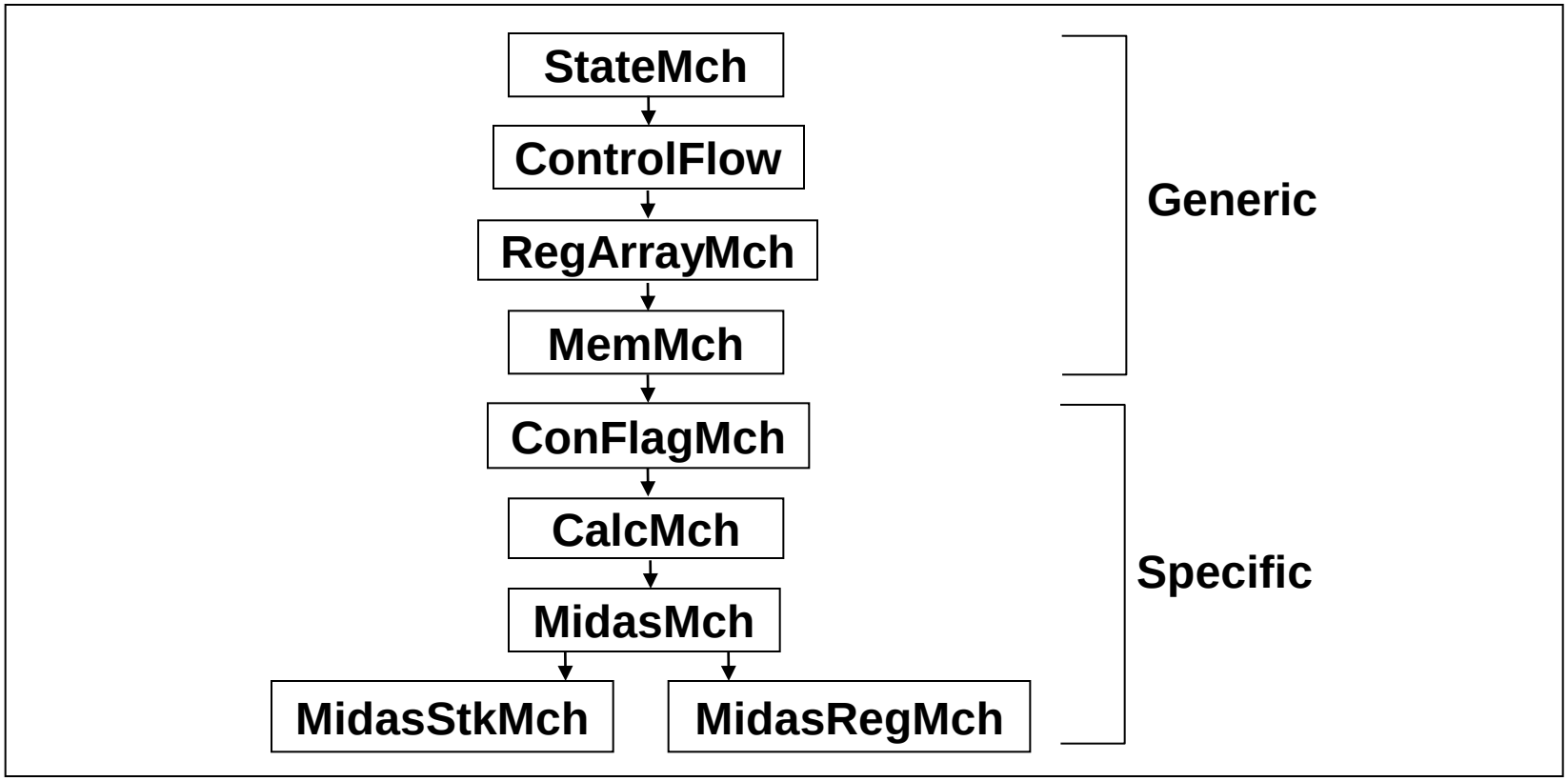
Instruction Set Architecture?

The instructions, registers and memory seen by an executing program.

Event-B Process

- Capture ISA common properties in generic model.
- Refine to multiple ISAs.
- Refine to example capable of auto-generation to C executable.

Model Structure



Top-level Abstraction

RUNNING/FAILED/HALTED state and instruction:

```
Iterate =
BEGIN

 act1: inst :∈ INST
 act2: status :∈ STATE

END
```

Final Refinement

122 events, reduced for C translation :

```
NopOk
REFINES NopOk
ANY
  op
  opVal
  nextInstPtr
WHERE
  grd6: op :  DataSmall
  grd7: op =  mem(instPtr)
  grd5: opVal :  DataSmallNat
  grd2: opVal =  DataSmall2Nat(op)
  grd1: opVal = 16
  grd3: instPtr <=  99994
  grd4: statusCode =  2
  grd8: nextInstPtr :  DataLargeNat
  grd9: nextInstPtr =  instPtr + 1
THEN
  act1: instPtr :=  nextInstPtr
END
```

MIDAS

- 34 instructions
- 30/33 refinement steps
- 109/113 final events
- 4916 PO’s
- 4092/4444 lines generated C

MIDAS Conclusions

- MIDAS was at upper size limit for Rodin modelling.
- Main issues: editing and proving.
- Memory allocation bottlenecks close.

XMOS

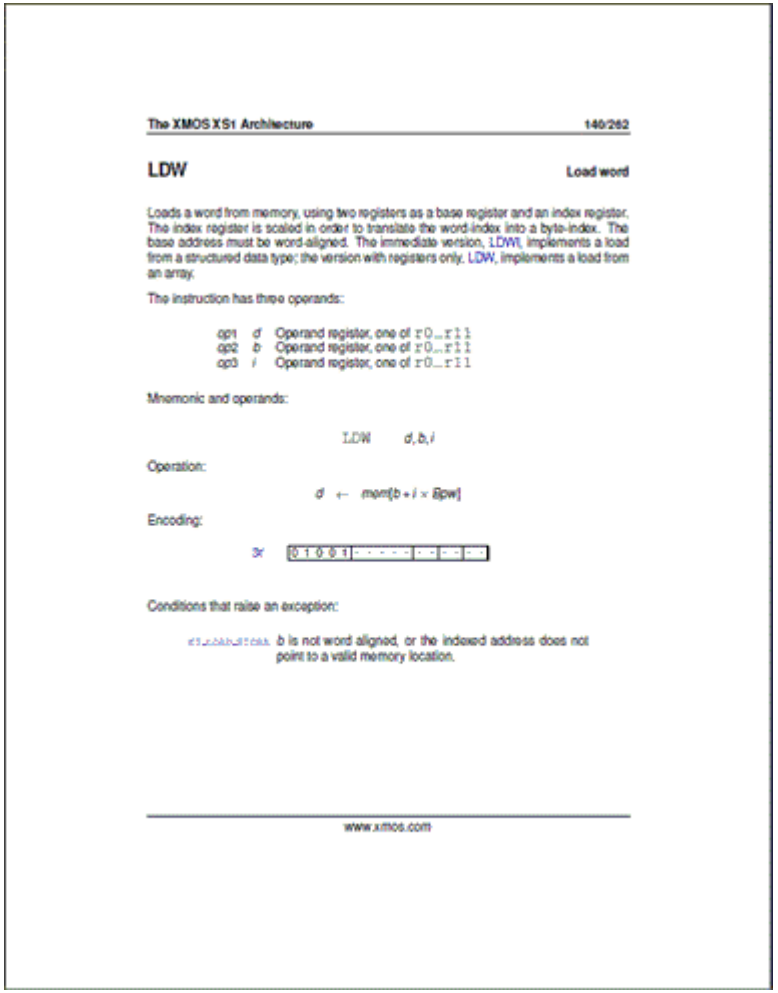
- Founded 2005
- University of Bristol spin-off
- Venture-capital funded

XCore

XCore

- **150** instructions
- Conventional ISA features
- Special instructions for thread scheduling and inter-core comms
- Well-structured ISA

XCore Specification



LDW — Load word

Loads a word from memory, using two registers as a base register and an index register. The index register is scaled in order to translate the word-index into a byte-index. The base address must be word-aligned. The immediate version, LDWI, implements a load from a structured data type; the version with registers only, LDW, implements a load from an array.

The instruction has three operands:

op1	*d*	Operand register, one of `r0...r11`
op2	*b*	Operand register, one of `r0...r11`
op3	*i*	Operand register, one of `r0...r11`

Mnemonic and operands:

```
LDW     d,b,i
```

Operation:

$$d \leftarrow mem[b + i \times B/pw]$$

Encoding:

3r | 0 1 0 0 1 | - - - - - | - - | - - | - -

Conditions that raise an exception:

ET_LOAD_STORE	*b* is not word aligned, or the indexed address does not point to a valid memory location.

Motive

- Check specification and documentation
- Check for behaviour of all software
- Test-case generation

Work Plan

Table 1. Work Plan Summary.

WP	Task	Description	Deliverable	Duration (weeks)	Quarter 1	Quarter 2	Quarter 3	Quarter 4
					M1		M2	M3
1	1	Architecture/Toolchain Familiarization	-	4				
	2	VM Development	D1	5				
	3	Rodin Review	-	1				
2	1	Rodin ISA Model Editor	D2.1	8				
	2	Rodin Proof Discharge Tool		8				
	3	Update of Generic Model		4				
3	1	Initial XCore Refinement	D3	5				
	2	B2C XCore Refinement		5				
	3	B2C Enhancement	D2.2	4				
	4	VM Generation and Test	-	4				
4	1	Formally Derived Specification	-	4				
			Total:	52				

Expected “Challenges”

- 500-1000 events in final refinement?
- 10,000-50,000 PO’s?
- Model too big for Rodin/Eclipse?

Main Activities

- Automatic Refinement Generation
- Improved automatic PO discharge
- Partitioning of model

Summary

- Boundary of academic/industrial.
- Rodin extensions needed to make project practical.
- Expected to push Event-B/Rodin scaling limits.

Questions

?