Experiences with a Quite Big Event-B Model

Steve Wright

Bristol University

University of BRISTOL

Department of COMPUTER SCIENCE

Steve Wright

- Chartered Software Engineer.
- 8 years at Rolls-Royce.
- 3 years at STMicroelectronics.
- 8 years at Airbus.
- Now completing PhD at Bristol.

Talk Summary

- Introduce “MIDAS” project
- Discuss as example of a “quite big” Event-B model

Objective

Formally construct all possible operations and error conditions for a microprocessor Instruction Set Archicture (ISA).

ISA Common Properties

- Binary image in contiguous memory.
- Instructions selected by “Program Counter”.
- All operations as bit-manipulations between registers.
- Register data transferable to/from memory system.

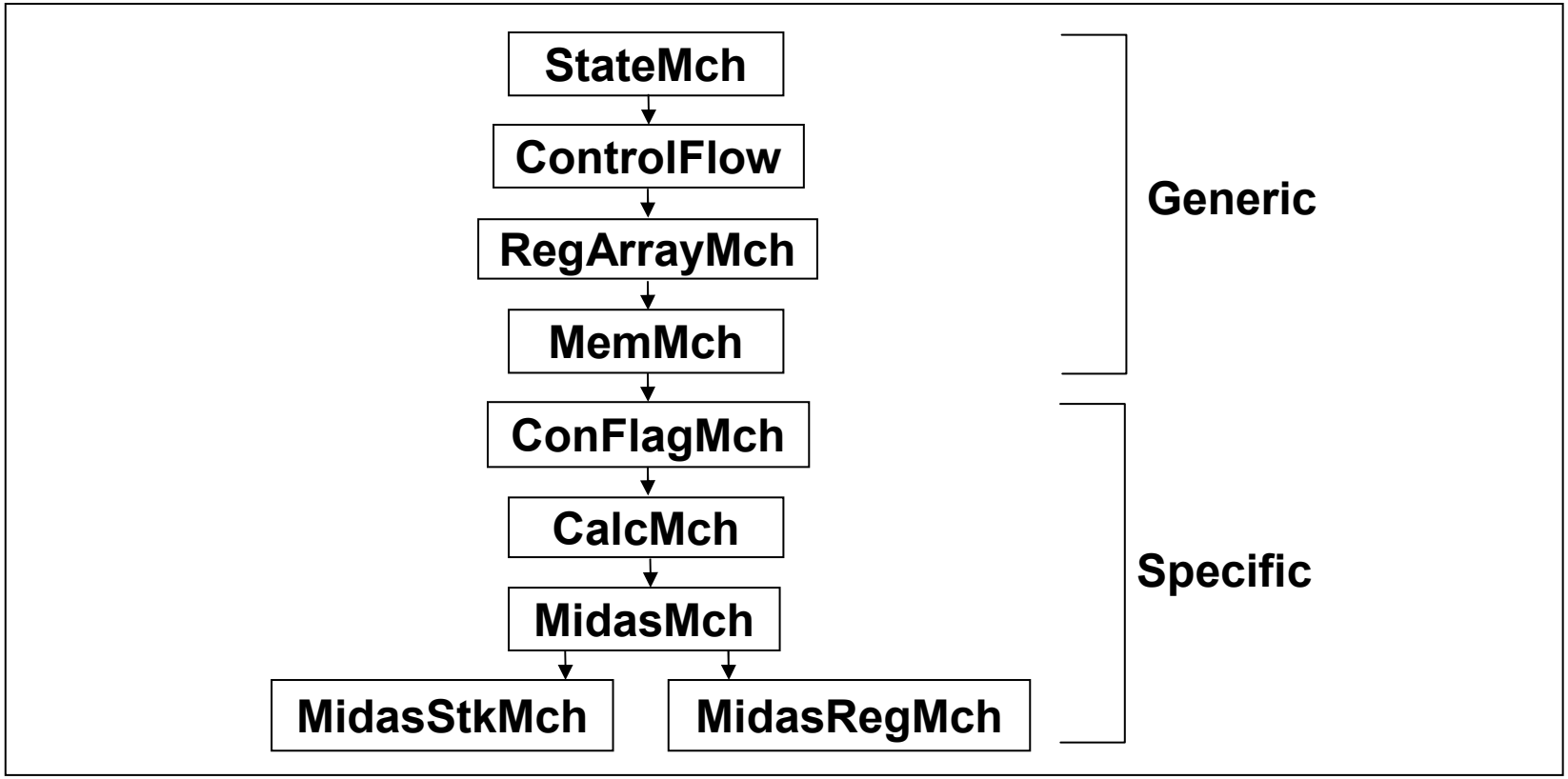
Project Objectives

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

“MIDAS” Example Machine

- 34 instructions.
- Stack and Register Variants.
- Harvard memory architecture.

Model Structure



Layer Summary

Name	Description	Steps
StateMch	Load/Run/Halt states	5
ControlFlowMch	PC behaviour	5
RegMch	Register interactions	5
MemMch	Data memory storage	9
ConFlagMch	Conditional branch	4
CalcMch	Register operations	2
MidasMch	Opcodes/sizes	8/9

Top-level Abstraction

RUNNING/FAILED/HALTED state and instruction:

```
Iterate =
BEGIN

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

END
```

Final Refinement

> 100 events, ready 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
```

VM Auto-generation

```
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
```

> B2C >

```
/* Event5 [NopOk] */
BOOL NopOk(void)
{
/* Local variable declarations */
DataLargeNat nextInstPtr;
DataSmall op;
DataSmallNat opVal;

/* Guard 1 */
op = mem[instPtr];
DataSmall2Nat(op,&opVal);
if(opVal!=16) return BFALSE;

/* Guard 2 */
if(instPtr>99994) return BFALSE;

/* Guard 3 */
if(statusCode!=2) return BFALSE;

/* Local assignments in actions */
nextInstPtr = (instPtr+1);

/* Actions */
instPtr = nextInstPtr;

/* Report hit */
ReportEventbEvent("NopOk",5);
return BTRUE;
}
```

MIDAS Publications

- “MIDAS Machine Specification”, 2008
- “Using EventB to Create a Virtual Machine Instruction Set Architecture”, 2008
- “Automatic Generation of C from Event-B”, IM-FMT, 2009
- Doctoral Thesis, 2009
- “Using Event-B to Create Instruction Set Architectures”, [to appear in] FACJ, 2010

MIDAS Download

- Event-B model
- C-coded prototype
- GCC compilers
- B2C translator source code
- Execution environment
- Install/build instructions

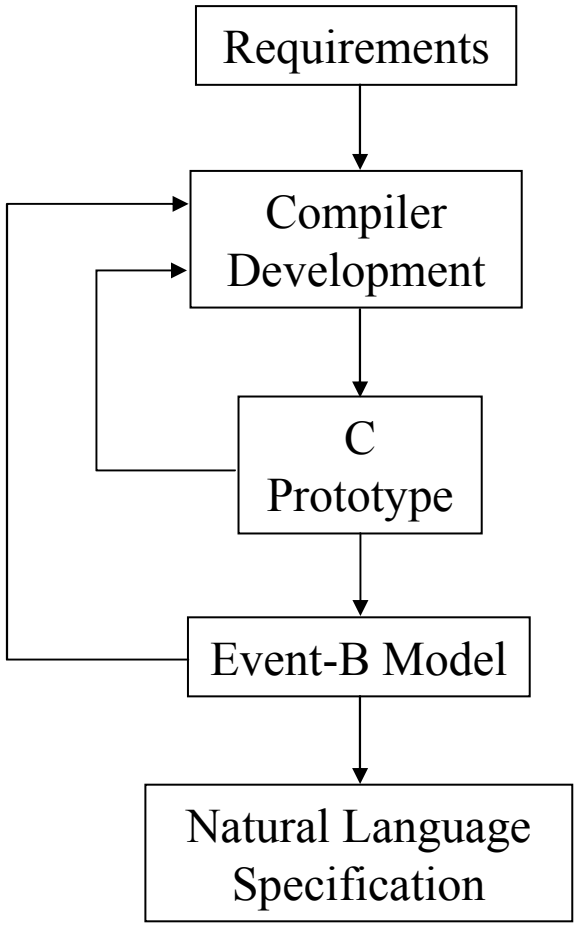
all at:

http://deploy-eprints.ecs.soton.ac.uk/84/

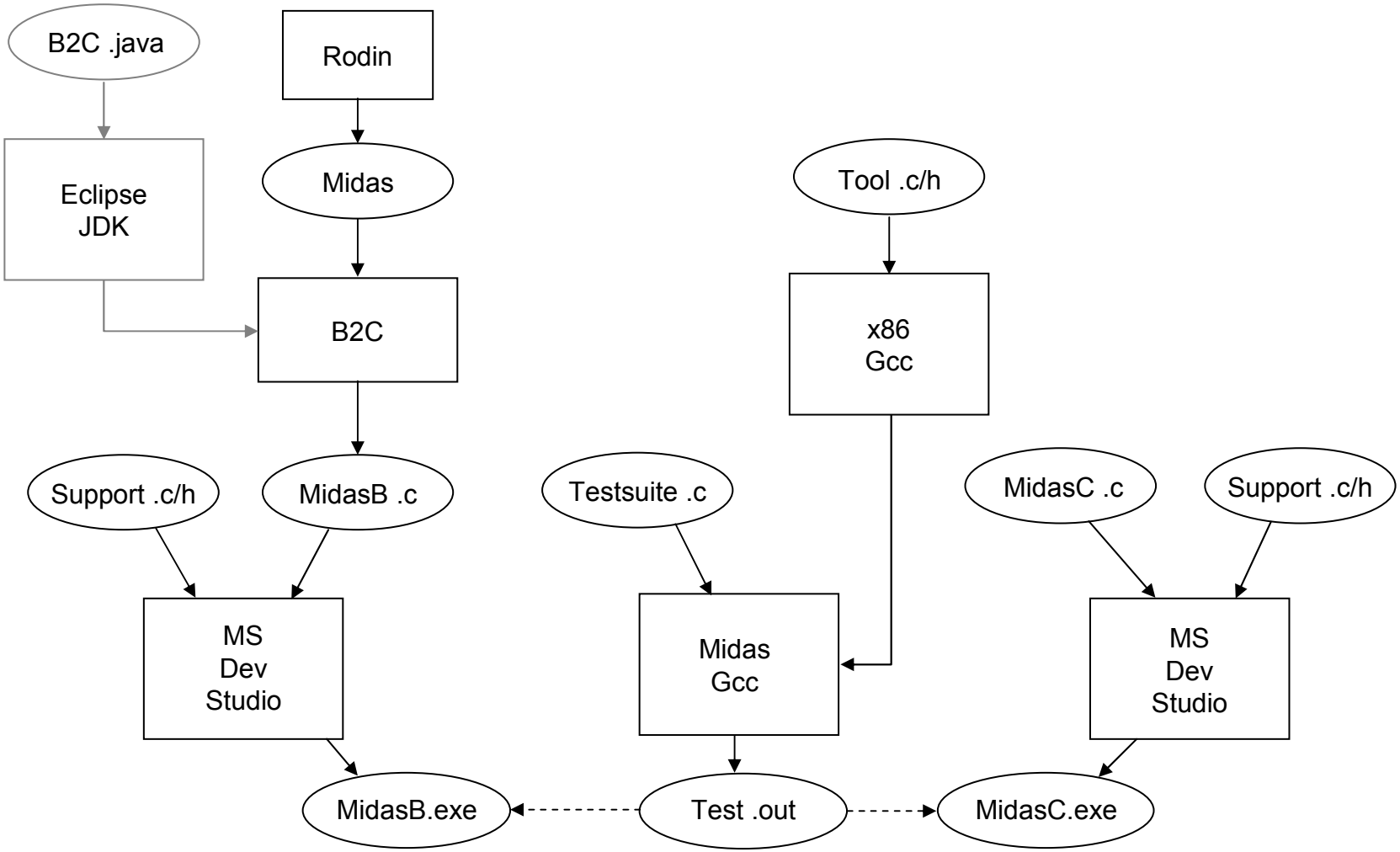
“Quite Big”

- 34 instructions
- 32/33 refinement steps
- 109/113 final events
- 4916 PO’s
- 4092/4444 lines generated C

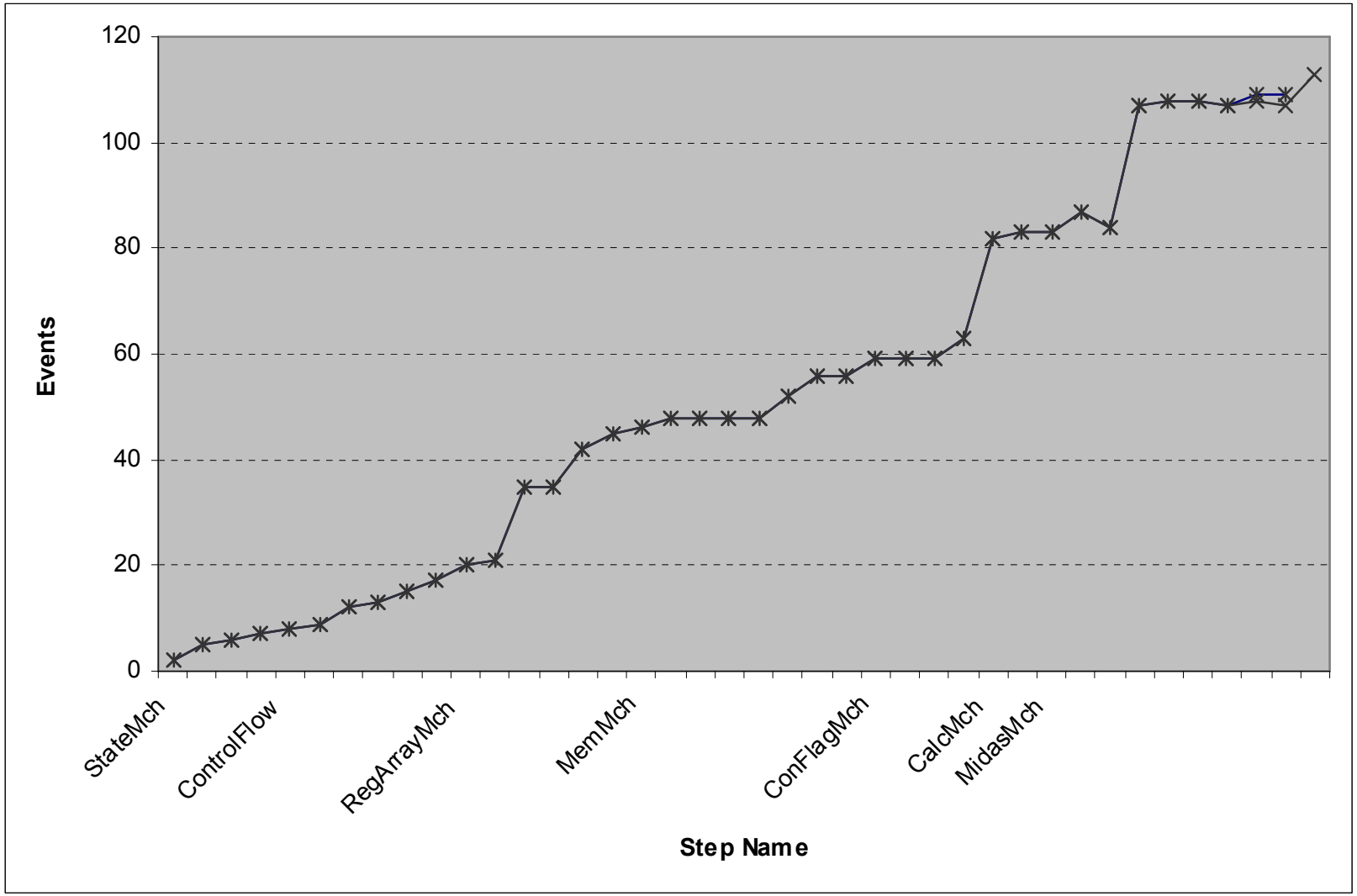
Development Flow



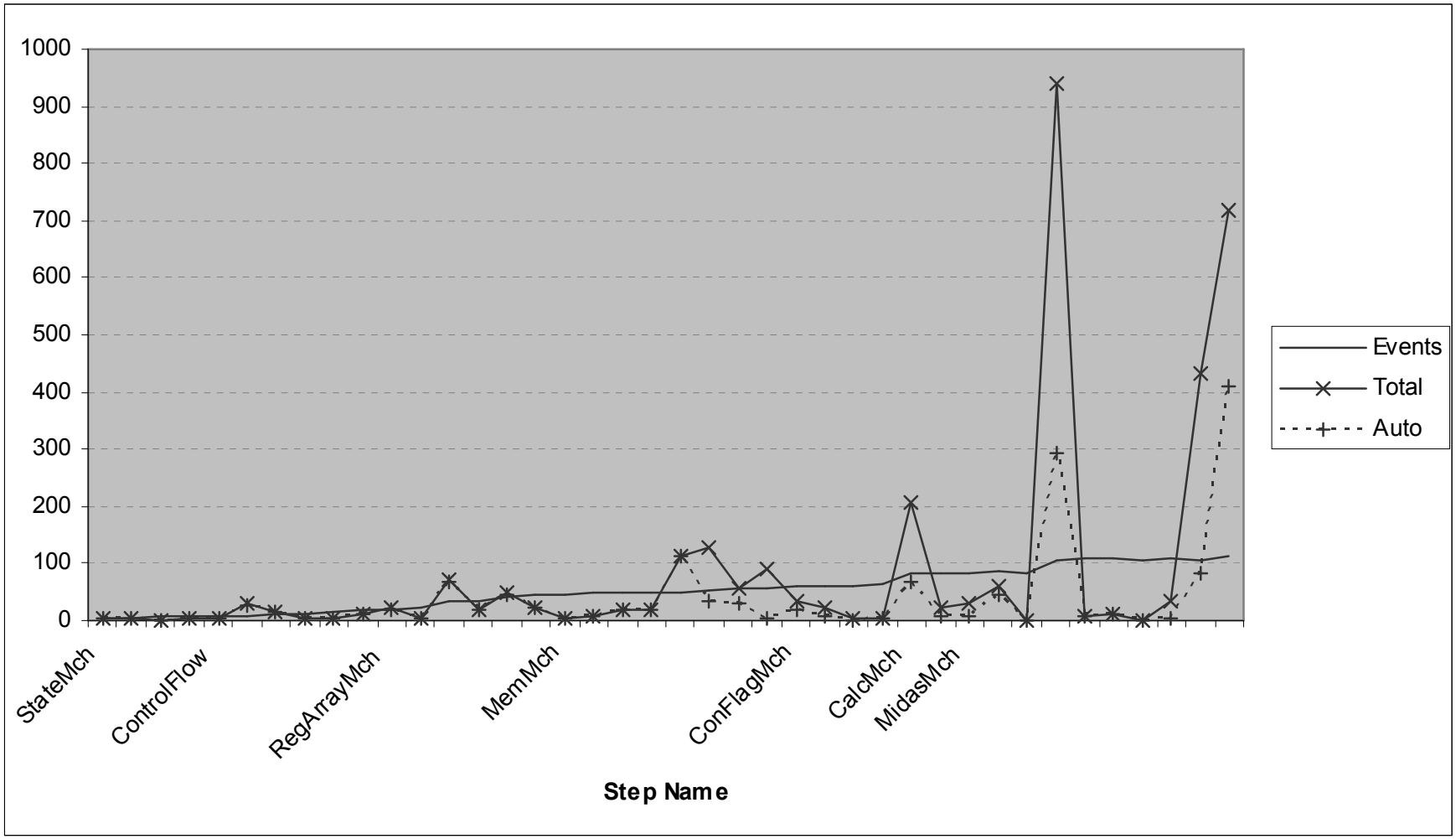
Tool Flow


B2C .java
Rodin
Eclipse
JDK
Midas
Tool .c/h
B2C
x86
Gcc
Support .c/h
MidasB .c
Testsuite .c
MidasC .c
Support .c/h
MS
Dev
Studio
Midas
Gcc
MS
Dev
Studio
MidasB.exe
Test .out
MidasC.exe

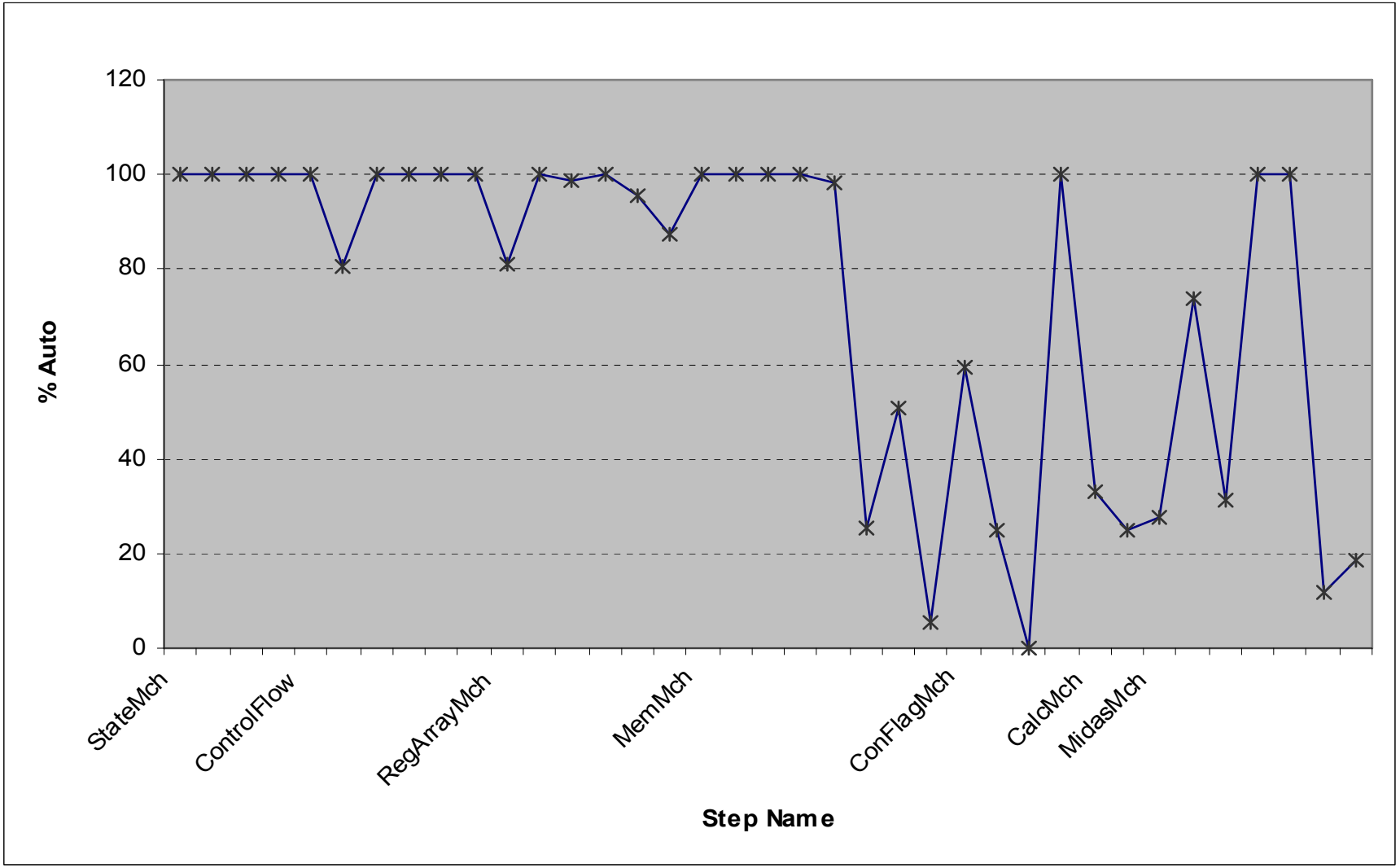
Event Buildup



PO Buildup


1000
900
800
700
600
500
400
300
200
100
0
StateMch
ControlFlow
RegArrayMch
MemMch
ConFlagMch
CalcMch
MidasMch
Step Name
Events
Total
Auto

Auto-Proving Breakdown



Editing

Proving - Midas/MidasStkMchB2C.bum - Rodin Platform

File Edit Navigate Search Project B2C Run Refactor Event-B Window Help

Obligation Explor | Proof Tree | Search Hypothe

MemMchR1(19|19,0|0,0)
MemMchR2(22|22,0|0,0)
MemMchR3(130|42,88|0,0)
MemMchR4(99|56,43|0,0)
MemMchR5(89|5,84|0,0)
MemMchR6(185|25,160|0,0)
MidasMch(29|8,21|0,0)
MidasMchR1(65|38,27|0,0)
MidasMchR2(0|0,0|0,0)
MidasMchR3(940|289,651|0,0)
MidasMchR4(8|0,8|0,0)
MidasMchR5(13|0,13|0,0)
MidasMchR6(0|0,0|0,0)
MidasRegMch(517|142,375|0,0)
MidasRegMchB2C(1166|588,578|0,0)
MidasStkMch(24|0,24|0,0)
MidasStkMchB2C(715|374,341|0,0)
MidasStkMchR01(11|0,11|0,0)
MidasStkMchR1(436|75,361|0,0)
MidasStkMchR2(0|0,0|0,0)
RegArrayMch(16|13,3|0,0)
RegArrayMchR1(4|4,0|0,0)
RegArrayMchR2(70|69,1|0,0)
RegArrayMchR3(25|25,0|0,0)
RegArrayMchR4(54|47,7|0,0)
StateMch(4|4,0|0,0)
StateMchR1(2|2,0|0,0)
StateMchR2(1|1,0|0,0)
StateMchR3(5|5,0|0,0)
StateMchR4(2|2,0|0,0)
Toy

MidasStkMchB2C

var grd

Element	Content
INITIALISATION	
Loading	
LoadComplete	
UnknownOpCode0	
NopOk	
NopBadPc	
NopBadQualifier	
FetchNoQualifier	
FetchInstPtrOk	
FetchInstPtrNoDest	
FetchInstPtrBadPc	
FetchImmByteOk	
FetchImmByteBadPc	
FetchImmLongOk	
FetchImmLongBadPc	
FetchImmNoDest	
FetchDirOk	
FetchDirNoStk	
FetchDirLoSrcOffset	
FetchDirHiSrcOffset	
FetchDirBadPc	
FetchIndirByteOk	
FetchIndirShortOk	
FetchIndirLongOk	
FetchIndirNoStk	
FetchIndirLoPtr	
FetchIndirHiPtr	
FetchIndirBadPc	
FetchBadQualifier	
StoreNoQualifier	

Add Event | Add Var. | Add Guard | Add Action | Up | Down

Pretty Print | Edit | Synthesis | Dependencies | Events | Invariants | Variables | Theorems

32M of 57M

Editing!

Microsoft Developer Studio - [C:\...\MidasStkMchB2C.bum]

File Edit View Insert Project Build Tools Window Help

grd6

```
<org.eventb.core.guard name="internal_grd5" org.eventb.core.label="grd5" org.eventb.core.predicate="currentPtr â‖ 0â¦¥2147
<org.eventb.core.guard name="internal_grd6" org.eventb.core.label="grd6" org.eventb.core.predicate="currentPtr = loadPtr"/>
</org.eventb.core.event>
<org.eventb.core.event name="evt64" org.eventb.core.convergence="0" org.eventb.core.inherited="false" org.eventb.core.label
<org.eventb.core.refinesEvent name="internal_refinesEvent1" org.eventb.core.target="UnknownOpCode0"/>
<org.eventb.core.variable name="internal_var100" org.eventb.core.identifier="op"/>
<org.eventb.core.variable name="internal_var101" org.eventb.core.identifier="opVal"/>
<org.eventb.core.guard name="internal_grd6" org.eventb.core.label="grd6" org.eventb.core.predicate="op â‖ DataByte"/>
<org.eventb.core.guard name="internal_grd2" org.eventb.core.label="grd2" org.eventb.core.predicate="op = memByte(instPtr)"/
<org.eventb.core.guard name="internal_grd7" org.eventb.core.label="grd7" org.eventb.core.predicate="opVal â‖ 0â¦¥255"/>
<org.eventb.core.guard name="internal_grd4" org.eventb.core.label="grd4" org.eventb.core.predicate="opVal = DataByte2Int(op
<org.eventb.core.guard name="internal_grd3" org.eventb                                                                  "/>
<org.eventb.core.guard name="internal_grd5" org.eventb                                                                  5"/>
<org.eventb.core.guard name="internal_grd1" org.eventb                                                                  e = 2"/>
<org.eventb.core.action name="internal_act2" org.event                                                                  mment="Excepti
</org.eventb.core.event>
<org.eventb.core.invariant name="internal_inv1" org.ev                                                                  tatusCode â‖
<org.eventb.core.invariant name="internal_inv2" org.ev                                                                  emByte â‖ 1â¦
<org.eventb.core.invariant name="internal_inv3" org.ev                                                                  egArrayDataLon
<org.eventb.core.invariant name="internal_inv4" org.ev                                                                  nstPtr â‖ 1â¦
<org.eventb.core.invariant name="internal_inv5" org.ev                                                                  oadPtr â‖ 1â¦
<org.eventb.core.invariant name="internal_inv7" org.ev                                                                  tkSize â‖ 0â¦
<org.eventb.core.invariant name="internal_inv6" org.ev                                                                  atus = Nat2Sta
<org.eventb.core.event name="evt75" org.eventb.core.co                                                                  ntb.core.label
<org.eventb.core.refinesEvent name="internal_refinesEvent1" org.eventb.core.target="NopOk"/>
<org.eventb.core.variable name="internal_var1" org.eventb.core.identifier="op"/>
<org.eventb.core.variable name="internal_var2" org.eventb.core.identifier="opVal"/>
<org.eventb.core.variable name="internal_var3" org.eventb.core.identifier="nextInstPtr"/>
<org.eventb.core.guard name="internal_grd8" org.eventb.core.label="grd8" org.eventb.core.predicate="op â‖ DataByte"/>
<org.eventb.core.guard name="internal_grd2" org.eventb.core.label="grd2" org.eventb.core.predicate="op = memByte(instPtr)"/
<org.eventb.core.guard name="internal_grd9" org.eventb.core.label="grd9" org.eventb.core.predicate="opVal â‖ 0â¦¥255"/>
<org.eventb.core.guard name="internal_grd5" org.eventb.core.label="grd5" org.eventb.core.predicate="opVal = DataByte2Int(op
<org.eventb.core.guard name="internal_grd1" org.eventb.core.label="grd1" org.eventb.core.predicate="opVal = 16"/>
<org.eventb.core.guard name="internal_grd7" org.eventb.core.label="grd7" org.eventb.core.predicate="nextInstPtr â‖ â¦´2147
<org.eventb.core.guard name="internal_grd6" org.eventb.core.label="grd6" org.eventb.core.predicate="nextInstPtr = instPtr +
<org.eventb.core.guard name="internal_grd10" org.eventb.core.label="grd10" org.eventb.core.predicate="nextInstPtr â‖¤ 99992
<org.eventb.core.guard name="internal_grd4" org.eventb.core.label="grd4" org.eventb.core.predicate="statusCode = 2"/>
<org.eventb.core.action name="internal_act1" org.eventb.core.assignment="instPtr â‖ nextInstPtr" org.eventb.core.label="ac
</org.eventb.core.event>
<org.eventb.core.event name="evt82" org.eventb.core.convergence="0" org.eventb.core.inherited="false" org.eventb.core.label
<org.eventb.core.refinesEvent name="internal_refinesEvent1" org.eventb.core.target="NopBadPc"/>
<org.eventb.core.variable name="internal_var1" org.eventb.core.identifier="op"/>
<org.eventb.core.variable name="internal_var2" org.eventb.core.identifier="opVal"/>
<org.eventb.core.guard name="internal_grd7" org.eventb.core.label="grd7" org.eventb.core.predicate="op â‖ DataByte"/>
<org.eventb.core.guard name="internal_grd2" org.eventb.core.label="grd2" org.eventb.core.predicate="op = memByte(instPtr)"/
```

Replace

Find what: grd6

Replace with: grd100

Find Next | Replace | Replace All | Cancel

Match whole word only

Match case

Regular expression

Replace in: Selection / Whole file

Ready | Ln 57, Col 69 | REC COL OVR READ

start | Inbox - Wind... | Inbox for ste... | Rodin Works... | Midas | Microsoft Po... | Microsoft De... | 11:07

“Quite Big”

“When auto-proving breaks down and editing becomes unwieldy”

i.e. More time spent editing and trivial proving than thinking about model and “real” proofs.

Workaround Tricks

- Use of local variables to avoid INVs.
- Repeated local variable names to allow proof C’n’P.
- Instantiation of AXIOMs in machine-scoped THEOREMs.

Eclipse Limits

- Eclipse resources are non infinite!
- JVM heap limits.
- Performance limits.

Contraversial Statement

Formal development takes >10 times more effort than conventional methods

Why?

- Superior product being developed

and

- Tools less mature

but

- Both can be addressed by Rodin enhancements

Wish List: Model Construction

- Automated refinement
- Take “extends” further
  - Syntactic sugar
- Parallel event refinement

Wish List: Model Interfacing

- Text editor
- Hierarchy browsers
- Graphical front ends (e.g. UML-B)

Wish List: Proving

- Programmable prover tactics?
- Automatic tactic generation?
- Prover hints in model (explicit guard refinement)?

Good Coding Practice

- Plan refinement structure
- Group related events on page
- Good comments
- Use meaningful tags

Summary

- MIDAS at current upper size limit for Rodin.
- Main issues: editing and proving.
- Problems solvable with Rodin.

Questions

?