BluEJAMM: A Bluespec Embedded Java Architecture with Memory Management

Flavius Gruian[1] Mark Westmijze[2]

[1]Lund University, Sweden
flavius.gruian@cs.lth.se

[2]University of Twente, The Netherlands
m.westmijze@student.utwente.nl

The 1st International Workshop on
Real-Time and Embedded Systems
in conjunction with SYNASC'07

Outline

What are our goals?

1. An embedded Java architecture, as a test platform
2. Evaluate BSV as a design language

BlueSpec System Verilog (BSV)

Rule based, strongly-typed, declarative hardware specification language, making use of Term Rewriting Systems to describe computations as atomic state changes.

3. Outperform other existing Java solutions in terms of
   - design time
   - flexibility
   - execution speed
   - device area

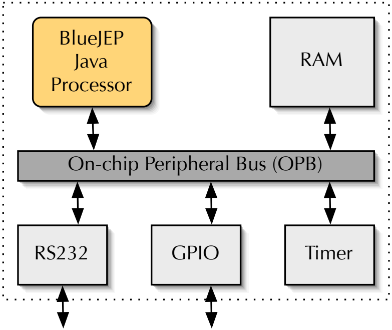
BluEJAMM

BlueSpec **E**mbedded **J**ava **A**rchitecture with **M**emory **M**anagement

Constraints and Features

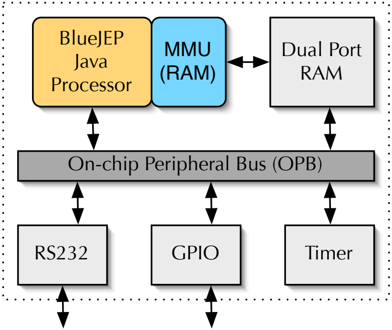
- For the BlueJEP Java processor (based on JOP):
    - micro-programmed, stack machine core
    - predictable rather than high-performance (RT systems)
    - given instruction set (bytecodes)
    - preset micro-instruction set (for ease of programming)
    - given executable image (loaded classes)
    - preset back-end (synthesis) tools
    - preset implementation platform (FPGA)
- Memory management
    - given object structure
    - both software and hardware (MMU) solutions

Architecture and Configurations Overview



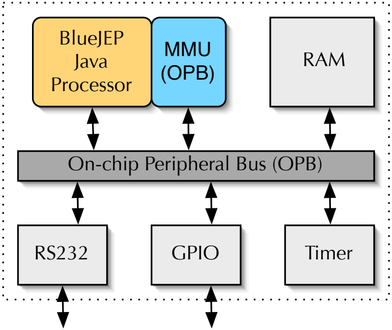
1. Software memory management

Architecture and Configurations Overview



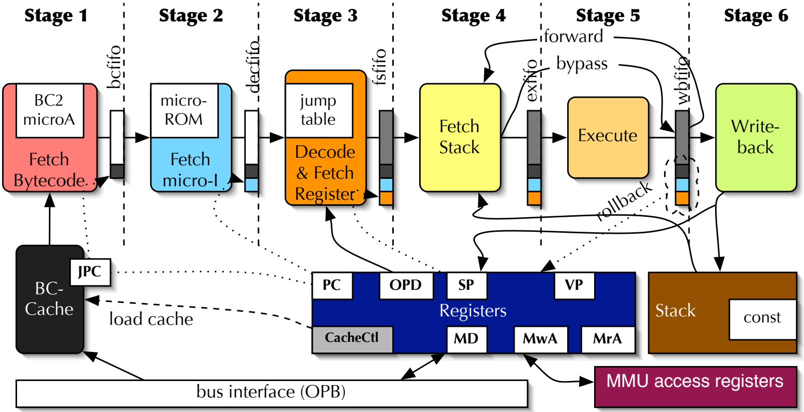
1. Software memory management
2. Hardware MMU using a dual-port RAM

Architecture and Configurations Overview

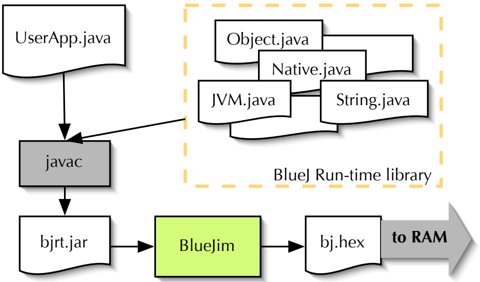


1. Software memory management
2. Hardware MMU using a dual-port RAM
3. Hardware MMU using the system bus

Six Stages Pipeline, Stack Machine



Run-Time Environment



BlueJim image generator

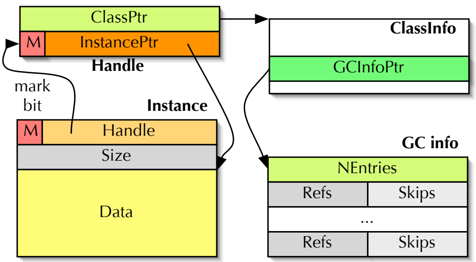
- offline class loading and linking
- replaces native calls with custom bytecodes
- throws away unused methods and fields
- adds GC information

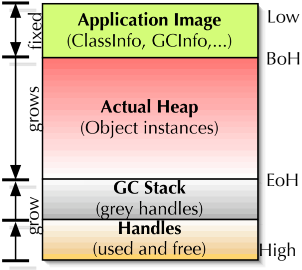
JVM.java Java implemented bytecodes.

Native.java Java-hardware interface.

***.java** Reduced JRE library.

Object structure, address space and garbage collection





- GC Algorithm: Mark-Compact
- Hardware (MMU):
  - handles all memory management functions
  - tight integration with the processor core (scans the stack for references)
  - stop-the-world GC for now, concurrent later on

Synthesis input, tools, and results

Input: BSV code, 1300 lines (BlueJEP) + 600 lines (MMU)

Tools:

- BSV compiler 2006.11, *BSV* → *Verilog*
- Xilinx EDK 9.1i, *Verilog* + *IPs* → *System*
- Xilinx ISE 9.1i, *System* → *FPGA*
- Chipscope, to monitor and debug

Target: FPGA, Xilinx Virtex-II (XC2V1000, fg456-4)

Area: (no optimization efforts)

- BlueJEP = 3460 slices (68%)
- BlueJEP + MMU = 4340 slices (85%)

Clock speed: (few optimization efforts)

- BlueJEP = 85 MHz
- BlueJEP + MMU = 64 MHz

Bytecode and application execution speed

- Execution time in clock cycles for several bytecodes:

Bytecode(s)	JOP	BlueJEP
iload iadd	2	3
iinc	11	13
ldc	9	12
if_icmplt taken	6	23
... n/taken	6	8
getfield	23	38
getstatic	15	18
iaload	29	45
invoke	126	166
invoke static	100	111

Profile for a simple application

Profile	SoftGC	MMU	ratio
bytecodes	24810	10304	42%
cc/byte	6	7	117%
cache fills	1601	675	42%
mem accesses	9063	3139	34%
GC clocks	49214	2626	5%
total cc	168977	73981	44%

- Performance similar to JOP, taking into account the faster clock
- Faster with MMU, even with the reported clock speed degradation

To conclude...

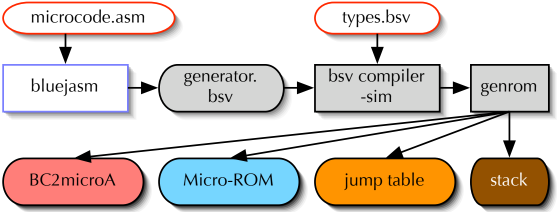
Summary: we introduced **BluEJAMM**, which:

- includes a native Java processor
- includes a hardware MMU
- is specified in BlueSpec System Verilog
- proves that BSV is perfect for fast prototyping

Extensions:

- Multi-block, multi-method caching [completed]
- Micro-instruction folding [under evaluation]
- Concurrent MMU [under development]

Micro-code Generation



- The encoding of the micro-instructions does not affect the assembler (bluejasm)!
- The actual encoding is interesting for optimization purposes only.