Why SpecInt95 Should Not Be Used to Benchmark Embedded Systems Tools

Jakob Engblom
Industrial PhD Student
Uppsala U / IT / Dept. of Computer Systems & IAR Systems AB
jakob@iarc.uu.se

Background
- The MARE Project:
  - Measurements of Actual Real-Time and Embedded Systems
- SpecInt95 used to benchmark everything
  - Including embedded tools
- Not appropriate for embedded systems
- This talk: demonstrate the difference!

Background (2)
- Basic Question: How do embedded programmers code?

Methodology
- Modified IAR C compiler
- Replace code generator
- Mimic other compilers:
  - Size of int
  - Keywords
  - Intrinsics
- Measuring optimized intermediate code

Studied Programs
- Embedded programs:
  - 13 applications, 337 kloc
  - Various industrial applications:
    - Telecomm, Vehicles, Consumer Products, ...
  - Medium-capacity 8- and 16-bit CPUs:
    - Z80, 68HC11, C166, M7000, H8, ...
  - Medium-to-large European companies

Studied Programs (2)
- SpecInt95:
  - 7 programs, 97 kloc
  - Integer programs only
  - All but gcc
  - Assume generic 32-bit CPU
Methodology (2)

And then I use Excel to handle the data and draw graphs and conclusions.

Summary of Differences

- Differences:
  - Variables: sign, size, scope
  - Operations: logic vs arithmetic, sign
  - Functions: parameters, return types, style
  - Libraries: no dynamic memory, no OS
  - Complexity: loop depths, functions
  - Hardware interfacing

Variables: Types

<table>
<thead>
<tr>
<th>Code Pointer</th>
<th>Pointer</th>
<th>Array</th>
<th>Struct/Union</th>
<th>Flow</th>
<th>Integer</th>
</tr>
</thead>
<tbody>
<tr>
<td>52%</td>
<td>12%</td>
<td>10%</td>
<td>8%</td>
<td>8%</td>
<td>6%</td>
</tr>
<tr>
<td>10%</td>
<td>10%</td>
<td>10%</td>
<td>10%</td>
<td>10%</td>
<td>10%</td>
</tr>
<tr>
<td>8%</td>
<td>8%</td>
<td>8%</td>
<td>8%</td>
<td>8%</td>
<td>8%</td>
</tr>
<tr>
<td>6%</td>
<td>6%</td>
<td>6%</td>
<td>6%</td>
<td>6%</td>
<td>6%</td>
</tr>
<tr>
<td>4%</td>
<td>4%</td>
<td>4%</td>
<td>4%</td>
<td>4%</td>
<td>4%</td>
</tr>
<tr>
<td>2%</td>
<td>2%</td>
<td>2%</td>
<td>2%</td>
<td>2%</td>
<td>2%</td>
</tr>
<tr>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
</tr>
</tbody>
</table>

Variables: Across Scopes

- By count
- By bytes

Variables: Integers

- SpecInt95 integers
- Embedded integers

Operations: Categories

- Categories:
  - Pointer
  - Compares
  - Arithmetic
  - Logic
- Ignored:
  - Jumps
  - Loads/stores

Graph
### Operations: Categories (2)

<table>
<thead>
<tr>
<th>Category</th>
<th>SpecInt95</th>
<th>Embedded</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arithmetic</td>
<td>46%</td>
<td>27%</td>
</tr>
<tr>
<td>Logic</td>
<td>9%</td>
<td>24%</td>
</tr>
<tr>
<td>Compares</td>
<td>45%</td>
<td>49%</td>
</tr>
</tbody>
</table>

### Operations: Signs

- **Spec:**
  - Arithmetic
  - Signed
  - Many decisions
- **Embedded:**
  - Logic
  - Unsigned
  - Many decisions

### Other Observations

- **About Embedded Programs**
  - 0/13 use malloc & free
  - 5/13 use OS dynamic memory allocation
  - 9/13 use function pointers
  - 11/13 use intrinsic functions
  - 13/13 use the standard library
  - Some recursion (!)

### Functions: Return Types

- **void**
  - 79.72%
  - 69.94%
- **char**
  - 0%
  - 0%
- **uchar**
  - 10%
  - 10%
- **short**
  - 20%
  - 30%
- **ushort**
  - 30%
  - 40%
- **long**
  - 40%
  - 50%
- **ulong**
  - 0%
  - 0%
- **ptr**
  - 10%
  - 10%
- **void**
  - 0%
  - 0%

### Functions: Parameter Count

- **0%**
  - SpecInt95
  - Embedded
- **10%**
  - SpecInt95
  - Embedded
- **20%**
  - SpecInt95
  - Embedded
- **30%**
  - SpecInt95
  - Embedded
- **40%**
  - SpecInt95
  - Embedded
- **50%**
  - SpecInt95
  - Embedded
- **60%**
  - SpecInt95
  - Embedded
- **70%**
  - SpecInt95
  - Embedded

### Functions: void-void???

- **void-void**
  - 4%
  - 35%
  - 35%
  - return value
  - 4%
  - 15%
  - parameters
  - 28%
  - 25%
  - return value
  - 35%
  - void-void
  - 4%
  - 15%
**Functions: Complexity**

- Trivial: single BB
- Non-loop: decisions
- Complex: loops
  - SpecInt95 deeper than embedded
  - No embedded loop deeper than 4

**Conclusions**

- Embedded and SpecInt95 programs are very different

Need special benchmarks for embedded systems and tools

**Future Work**

- Examine more programs
- Examine candidates:
  - EEMBC (EDN Embedded Microprocessor Benchmark Consortium)
    - CPU Performance comparisons
    - If we can get funding to get the code
  - Any other benchmarks?

**More Information**

- Tech Report on MARE
- Paper accepted for RTAS ’99
  - See you in Vancouver!
- My homepage:
  - www.docs.uu.se/~jakob
  - Will have those slides online!