Minimizing Bank Selection Instructions for Partitioned Memory Architectures

Bernhard Scholz$^1$, Bernd Burgstaller$^1$, and Jingling Xue$^2$

$^1$The University of Sydney

$^2$University of New South Wales
Motivation

Huge market for microcontrollers ($\mu$Cs):

- SIA forecast: 8-bit $\mu$C market reaches $5.32$ billion in 2006
- PIC Microchip sold 1 billion units between 2004 and 2005

$\mu$Cs neglected by research community as “assembly language playground”

Productivity considerations require move to high-level languages

Compiler optimizations for $\mu$Cs required
Partitioned what???

\[ 2^N - 1 \]

instruction encoding:

|_OPCODE | f |
\---|---|
 MOVWF | f |

Operand: \( 0 \leq f \leq 2^N - 1 \)
Operation: \((W) \rightarrow f\)

- effective address \( f \) encoded in \( N \) bits
- large instruction word sizes increase code size
Partitioned Address Space

\[ 2^N - 1 \]

\[ \begin{array}{c}
\vdots \\
0
\end{array} \]

\[ a \]

\[ 2^k \text{ banks:} \]

\[ \begin{array}{c}
0 \\
X \\
\vdots \\
X \\
0 \\
\vdots \\
0
\end{array} \]

\[ \begin{array}{c}
1 \\
X \\
\vdots \\
X \\
0 \\
\vdots \\
0
\end{array} \]

\[ 2^k - 1 \]

- requires *address disambiguation mechanism*
- reduces effective address to \( N - k \) bits
- increases memory space without increasing address bus
- reduces instruction word size and memory footprint
Address Disambiguation

- CPU can access one bank at a time
- active bank stored in CPU bank register
- bank selection instruction `BSL <nr>` switches banks
- access of entity `a`: `BSL 7
  MOVWF X`
Bank Selection Instruction Placement

bank 0: A, B
bank 1: X, Y, Z

CFG

speed

space
Bank Switching in Prevalent μC Architectures

Used for data, code, and CPU registers

- Motorola 68HC11
- PIC 12*, 14* and 16* families
- Intel 8051 family
- Ubicom SX, Toshiba T900

Significant potential for compiler improvement

- to reduce bank switching overhead
Problem Statement

Minimize number of bank selection instructions of a program for

- a static allocation of variables to banks
- a cost metric (speed, space, etc.)
- a fixed instruction schedule (no instruction re-ordering).
Outline of Solution

Optimization within basic blocks

Intra-procedural optimization

- modelled as a discrete optimization problem

Extension to the interprocedural case

Mapping of the discrete optimization problem to a solver
Basic Block Optimization

Performs linear scan of a basic block, memorizing the state of the bank selection register

Inserts bank selection statements except

- before the first bank-sensitive statement
- after the last bank-sensitive statement

Example:

Input:  
1  LD (bank 1) X
2  CALL foo
3  LD (bank 0) A
4  ST (bank 1) Y
5  ST (bank 1) Z

Output:  
1  LD X  
2  CALL foo  
3  BSL 0; LD A  
4  BSL 1; ST Y  
5  ST Z

Minimizing Bank Selection Instructions for Partitioned Memory Architectures – p.10
Intra-Procedural Optimization

Introduce controlling variables $P$ and $Q$ for each basic block

Domain: $\mathbb{D} = \{0, \ldots, 2^k - 1, b?\}$

- $P$ specifies bank register state upon entry (guarantee)
- $Q$ specifies bank register state upon exit (obligation)

Discrete optimization “attaches” to controlling variables

\[ P \\
\vdots \\
Q \]
Controlling Variables on Bank-Sensitive Basic Blocks

\[ P = 0 \]

LD (bank 1) Z

\[ P = 1 \]

LD (bank 1) Z

ST (bank 0) B

BSL 1

\[ Q = 1 \]

BSL 1

\[ Q = 1 \]

BSL 1

Cost function \( n\text{-cost}(P) \)

- accounts for the costs at the entry of the basic block

Cost function \( e\text{-cost}(Q) \)

- accounts for the costs at the exit of the basic block
Controlling Variables on Transparent Basic Blocks

Cost function $t\text{-}cost(P, Q)$
- accounts for the costs at a transparent basic block

Cost functions can model arbitrary costs (speed, space, power consumption, ...)

- $P = 1$
- $P = 0$
- $Q = 1$
- BSL 1

$P$

$Q$

$Q = 1$

$Q = 1$
Correctness Criteria for Intra-Procedural Optimization

“adjacent controlling variables have to agree”:

forall edges (u,v): $(P_v \neq b?) \Rightarrow (Q_u = P_v)$
The Bank Selection Placement Optimization Problem

Objective function $f$ accounts for the costs of basic blocks

Controlling variables determine costs of transformations

\[
\begin{align*}
\text{s.t.} & \quad \forall u \in V : P_u, Q_u \in \mathbb{D} \\
& \quad P_s = b_? \\
& \quad \forall (u, v) \in E : (P_v \neq b_?) \Rightarrow (Q_u = P_v) \\
\min f &= \sum_{u \in V} \text{cost}_u(P_u, Q_u) \\
&\quad = \sum_{u \in S} \text{n-cost}_u(P_u) + \sum_{u \in S} \text{e-cost}_u(Q_u) + \sum_{u \in T} \text{t-cost}_u(P_u, Q_u)
\end{align*}
\]

bank-sensitive

transparent
Interprocedural Extension

Bank selection instructions can be hoisted across call sites

Additional placement of BSL instructions:
- at the entry/exit of a subroutine, and before a call

Caller/Callee correctness constraints:

\[(P_s \neq b_\gamma) \Rightarrow (P_s = P_i)\]

\[(Q_i \neq b_\gamma) \Rightarrow (Q_i = Q_e)\]

- one discrete optimization problem for the whole program
Experimental Results

Implementation of optimizations for a PIC16F877A μC

- RISC-based Harvard architecture
- 8-bit data bus
- 4 data banks, 368 bytes data memory

Evaluated optimizations for

- MiBench (as applicable),
- DSPStone (as applicable),
- a μC real-time kernel, and
- μC driver routines

Reference point: HI-TECH PICC high-performance C-compiler
Experimental Results (speedup)

![Speedup Bar Chart]

- Benchmark Programs: adpcm, basicmath, FFT, lcl, matrix, rikernel, sha, all
- Speedup (%): 0 to 32
- Categories: Speed, Space, Mixed
Experimental Results (code size reduction)

![Bar Chart]

<table>
<thead>
<tr>
<th>Benchmark Programs</th>
<th>Speed</th>
<th>Space</th>
<th>Mixed</th>
</tr>
</thead>
<tbody>
<tr>
<td>adpcm</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>basicmath</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FFT</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>lcd</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>matrix</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>rtkernel</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sha</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>all</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Size Reduction (%)
## Optimally solved procedures, all of MiBench

<table>
<thead>
<tr>
<th>Benchmark</th>
<th>proc</th>
<th>node</th>
<th>edge</th>
<th>proc</th>
<th>node</th>
<th>edge</th>
<th>P,Q</th>
<th>optproc</th>
</tr>
</thead>
<tbody>
<tr>
<td>basicmath</td>
<td>6</td>
<td>97</td>
<td>143</td>
<td>174</td>
<td>220</td>
<td>348</td>
<td>6</td>
<td>100.0%</td>
</tr>
<tr>
<td>bitcount</td>
<td>15</td>
<td>88</td>
<td>102</td>
<td>124</td>
<td>138</td>
<td>248</td>
<td>13</td>
<td>86.7%</td>
</tr>
<tr>
<td>qsort</td>
<td>4</td>
<td>40</td>
<td>55</td>
<td>59</td>
<td>74</td>
<td>118</td>
<td>3</td>
<td>75.0%</td>
</tr>
<tr>
<td>susan</td>
<td>19</td>
<td>698</td>
<td>1102</td>
<td>1104</td>
<td>1509</td>
<td>2208</td>
<td>11</td>
<td>57.9%</td>
</tr>
<tr>
<td>jpeg</td>
<td>375</td>
<td>6998</td>
<td>10255</td>
<td>10948</td>
<td>14211</td>
<td>21896</td>
<td>262</td>
<td>69.9%</td>
</tr>
<tr>
<td>lame</td>
<td>213</td>
<td>5720</td>
<td>8510</td>
<td>8735</td>
<td>11528</td>
<td>17470</td>
<td>148</td>
<td>69.5%</td>
</tr>
<tr>
<td>mad</td>
<td>1</td>
<td>3</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>6</td>
<td>1</td>
<td>100.0%</td>
</tr>
<tr>
<td>tiff</td>
<td>510</td>
<td>10735</td>
<td>15668</td>
<td>15678</td>
<td>20628</td>
<td>31356</td>
<td>320</td>
<td>62.7%</td>
</tr>
<tr>
<td>typeset</td>
<td>399</td>
<td>20650</td>
<td>31675</td>
<td>31322</td>
<td>42348</td>
<td>62644</td>
<td>251</td>
<td>62.9%</td>
</tr>
<tr>
<td>dijkstra</td>
<td>12</td>
<td>138</td>
<td>184</td>
<td>198</td>
<td>246</td>
<td>396</td>
<td>10</td>
<td>83.3%</td>
</tr>
<tr>
<td>patricia</td>
<td>5</td>
<td>160</td>
<td>227</td>
<td>217</td>
<td>285</td>
<td>434</td>
<td>2</td>
<td>40.0%</td>
</tr>
<tr>
<td>ghostscript</td>
<td>3551</td>
<td>47967</td>
<td>65967</td>
<td>67117</td>
<td>85120</td>
<td>134234</td>
<td>2734</td>
<td>77.0%</td>
</tr>
<tr>
<td>ispell</td>
<td>107</td>
<td>2368</td>
<td>3611</td>
<td>3757</td>
<td>5001</td>
<td>7514</td>
<td>51</td>
<td>47.7%</td>
</tr>
<tr>
<td>rsynth</td>
<td>53</td>
<td>1326</td>
<td>2012</td>
<td>2074</td>
<td>2760</td>
<td>4148</td>
<td>32</td>
<td>60.4%</td>
</tr>
<tr>
<td>sphinx</td>
<td>684</td>
<td>10597</td>
<td>14895</td>
<td>15732</td>
<td>20043</td>
<td>31464</td>
<td>522</td>
<td>76.3%</td>
</tr>
<tr>
<td>stringsearch</td>
<td>13</td>
<td>176</td>
<td>240</td>
<td>262</td>
<td>326</td>
<td>524</td>
<td>7</td>
<td>53.8%</td>
</tr>
<tr>
<td>blowfish</td>
<td>14</td>
<td>237</td>
<td>344</td>
<td>373</td>
<td>483</td>
<td>746</td>
<td>13</td>
<td>92.9%</td>
</tr>
<tr>
<td>pgp</td>
<td>320</td>
<td>7381</td>
<td>10959</td>
<td>11207</td>
<td>14788</td>
<td>22414</td>
<td>195</td>
<td>60.9%</td>
</tr>
<tr>
<td>rijndael</td>
<td>7</td>
<td>157</td>
<td>223</td>
<td>220</td>
<td>286</td>
<td>440</td>
<td>2</td>
<td>28.6%</td>
</tr>
<tr>
<td>sha</td>
<td>7</td>
<td>56</td>
<td>70</td>
<td>78</td>
<td>92</td>
<td>156</td>
<td>7</td>
<td>100.0%</td>
</tr>
<tr>
<td>adpcm</td>
<td>5</td>
<td>79</td>
<td>105</td>
<td>103</td>
<td>132</td>
<td>206</td>
<td>3</td>
<td>60.0%</td>
</tr>
<tr>
<td>CRC32</td>
<td>4</td>
<td>32</td>
<td>41</td>
<td>47</td>
<td>56</td>
<td>94</td>
<td>3</td>
<td>75.0%</td>
</tr>
<tr>
<td>FFT</td>
<td>6</td>
<td>83</td>
<td>114</td>
<td>124</td>
<td>156</td>
<td>248</td>
<td>6</td>
<td>100.0%</td>
</tr>
<tr>
<td>gsm</td>
<td>65</td>
<td>1589</td>
<td>2244</td>
<td>2097</td>
<td>2754</td>
<td>4194</td>
<td>44</td>
<td>67.7%</td>
</tr>
<tr>
<td>all</td>
<td>6395</td>
<td>117375</td>
<td>168748</td>
<td>171753</td>
<td>223186</td>
<td>343506</td>
<td>4646</td>
<td>72.7%</td>
</tr>
</tbody>
</table>
Solve times wrt. the number of banks, all of MiBench
Solve times wrt. the number of CFG edges, all of MiBench

![Graph showing the relationship between the number of CFG edges and execution time in milliseconds.](image-url)
Summary

Devised algorithm to minimize bank selection instructions for
  ● a given instruction order
  ● and a given data partitioning

Formulated bank selection instruction placement as a discrete optimization problem

Optimization objectives are formulated as cost metrics

Experimental results for PIC16F877A $\mu$-controller:
  ● code size reduction between 2.7% and 18.2%
  ● speedup between 5.1% and 28.8%
  ● 100% of PIC benchmarks solved optimally, 72% of MiBench