Compiler Optimizations on DSP Cores for High-Performance and Low-Power

Jenq-Kuen Lee

jklee@cs.nthu.edu.tw
Programming Language Research Lab.
Department of Computer Science
National Tsing-Hua Univ.
Compilers for PDA environments

SDK Kits and Fast prototyping for SoC/IP

Compiler issues for low-power systems
  1. Compilers for code size reduction
  2. Software for reducing heat dissipations
  3. Compilers for average power reductions

Compiler optimization issues for embedded java processors
Outline of Research Work

- System Software for Power Managements
  - Compilers
  - Operating Systems
- Portabilities with Efficient DSP Libraries
  - Pseudo Assembly (ISDL-like Descriptions)
  - Annotation Languages for Compiler Optimizations
Compilers for low-power

- Research Problems
  VLIW instruction schedulings aim at reducing power consumptions for instruction bus
  (jointly done by Ching-Ren Lee, Ting-Ting Hwang, Shi-Chun Tsai, and Jenq-Kuen Lee)
Solutions

- **Horizontal Scheduling**
  - Permute micro-instructions within a given VLIW instruction

- **Vertical Scheduling**
  - Reorder VLIW instructions’ sequence in a basic block

- **Possible Component Activity Solution**
  - Extension to Pipeline States
Our compiler phases on lower optimization
Experiment Results (Cont’d)

Our Test Suites Results on 4-Way Issues
Directions with Low Power Issues

- Relationships between CPU’s component activities & power consumptions.
- A strategy to integrate different compiler schemes for low-power.
- Instruction sets to control power constructions.
- Evaluate power consumption models and simulators.
Power Managements at OS layer

- Minimize power consumptions while meet the deadline of real-time tasks
- Joint work done by Yi-Ping You, Ching-Ren Lee,, Jenq-Kuen Lee, Wei-Kuan Shih.
- To be submitted to IEEE Workshop on Power Management for Real-Time and Embedded Systems.
Power Model

- Total energy consumption

\[ E_{total} = \sum_{j=1}^{n} \alpha_j C_L V_{ddj}^2 s_j \]

- Average power consumption

\[ P_{ave} = \frac{1}{n} \sum_{j=1}^{n} \alpha_j C_L V_{ddj}^2 f_j \]

**\( \alpha_j \): average switching activity of task j**
Scheduling Algorithm

1. Assume there are n periodic tasks to be scheduled.
2. Sort deadlines in ascending order, namely \( T_1, T_2, \ldots, T_n \). And put them in a list, called reservation list.

Repeat 3-6 when the reservation list is not empty

3. Remove the first task, \( T_i \), from the list.
4. Compute slack time of both low and high voltage schedule, i.e. \( STL \) and \( STH \).
5. Compute \( CTL(T_i) \) and \( CTH(T_i) \).
6. Schedule \( T_i \)
   \[
   \begin{cases} 
   CTL(T_i) \leq STL, & \text{schedule } T_i \text{ with low voltage if possible.} \\
   STL < CTL(T_i) \leq STH, & \text{call decision algorithm.} \\
   CTL(T_i) > STH, & \begin{cases} 
   CTH(T_i) \leq STH, & \text{schedule } T_i \text{ with high voltage if possible.} \\
   CTH(T_i) > STH, & \text{call exception (real-time failures).} 
   \end{cases}
   \end{cases}
   \]
Slack Computation

<table>
<thead>
<tr>
<th>STL</th>
<th>STH</th>
</tr>
</thead>
</table>

: high voltage scheduling

: low voltage scheduling

latest deadline

high voltage

low voltage

time
Decision Algorithm

- Comparison with energy (switching-activity sensitive)
- Slack time
Compiler Toolkits for DSP Cores

Fast System Software Prototyping

A heterogeneous system-on-a-chip

- Peripherals
- DSP Core or ASIP
- RAM
- ASIC

Program ROM

Simulator Environment

Retargetable Compilers and SDK Kits

Hardware description language
DSP  Compilers

DSP Compiler for TMS320C62x

- Optimizing Core-routine in Internet Telephony (G.723.1 …etc)
  - An initial testbed for experimenting retargetting of dsp libraries
  - On-Going work by Yung-Chia Lin & Jenq-Kuen Lee
DSP Compiler for TMS320C62x

- Compiler Prototype based on SUIF and Our Back-End Part
- Machine-dependent Code Generation for TMS320C62x
- SUIF IR Scalar Optimization
- Machine-dependent Code Code Optimization
Specifications for DSP Libraries
(Our Approach)

- ISDL-like (Pseudo) Description Languages
- Compound Instruction Descriptions
- Equation Descriptions
- Tensor-Product Descriptions
- Matrix Operation Descriptions
/*annotate( x= Sat(y+z), x,y,z:16) */

Word16 add(Word16 var1, Word16 var2)
{
    Word16 var_out;
    Word32 L_somme;

    L_somme = (Word32) var1 + (Word32) var2;
    var_out = sature(L_somme);
    return(var_out);
}

Replacement with intrinsic assembly routines
Examples of Equation Descriptions for Intrinsic Placements

- `Word16 add(Word16 var1, Word16 var2); /* Short add, 1 */`
- `Word16 sub(Word16 var1, Word16 var2); /* Short sub, 1 */`
- `Word16 abs_s(Word16 var1); /* Short abs, 1 */`
- `Word16 shl(Word16 var1, Word16 var2); /* Short shift left, 1 */`
- `Word16 shr(Word16 var1, Word16 var2); /* Short shift right, 1 */`
- `Word16 mult(Word16 var1, Word16 var2); /* Short mult, 1 */`
- `Word32 L_mult(Word16 var1, Word16 var2); /* Long mult, 1 */`
- `Word16 negate(Word16 var1); /* Short negate, 1 */`
- `Word16 extract_h(Word32 L_var1); /* Extract high, 1 */`
- `Word16 extract_l(Word32 L_var1); /* Extract low, 1 */`
- `Word16 round(Word32 L_var1); /* Round, 1 */`
- `Word32 L_mac(Word32 L_var3, Word16 var1, Word16 var2); /* Mac, 1 */`
- `Word32 L_msu(Word32 L_var3, Word16 var1, Word16 var2); /* Msu, 1 */`
- `Word32 L_macNs(Word32 L_var3, Word16 var1, Word16 var2);`
- ...
Performance Effects with Durbin routines in G.723.1
(Equation Descriptions)
Annotations for Array Computations

Example 1:
/*$Annotation void matrix(matrix_name;dimension) */
/*$Annotation void matrix(matrix_name;(a11 a12…)(a21 a22…)…) */
/*$Annotation void diagonal(matrix_name,d1 d2…) */
/*$Annotation void transpose(matrix_name;dimension) */
/*$Annotation void permutation(matrix_name,p1 p2…) */
/*$Annotation matrix kron(matrix…) */
/*$Annotation matrix compose(matrix…,matrix result) */

Example 2:
\[ F_8 = (F_2 \bullet I_4)(I_2 \bullet F_4) \] (where \( \bullet \) is Kronecker tensor product)

/* $Annotate compose( kron((F;2),(I;4)),(T;8;4),
   kron((I;2),(F;4)),(L;8;2),
   (F;8) ) */
Conclusions

- We experimented with a compiler solution to reduce power consumption on instruction bus.
- OS layer optimizations for reducing power consumptions are important.
- Portability for efficient DSP libraries presents challenging research issues.
- Simulators for both performance and power consumptions are important.