Programming the FlexRAM Parallel Intelligent Memory System

B. B. Fraguela*, J. Renau†, P. Feautrier‡
D. Padua† and J. Torrellas†

*Univ. da Coruña
†Univ. of Illinois
‡ENS de Lyon
Intelligent Memory Architectures

- Main memory enhanced with many simple processors

- Heterogeneous
- Highly parallel

Problem: Little research on how to program these architectures
Contributions

- Language support for intelligent memories
  - OpenMP-like directives (CFlex)
  - Library of Intelligent Memory Operations (IMOs)
- Runtime system and OS extensions
- Speedups of over one order of magnitude
Outline

- FlexRAM Architecture
- Software Support
- CFlex
- IMOs
- Evaluation
- Related Work
- Conclusions
FlexRAM Architecture

- 64 PArrays/chip
- 2D torus inside each chip
- PArrays are much simpler than the PHost(s)
- Controller: comm & synchr tasks
- The FlexRAM bus interconnects all the chips

B. B. Fraguela et al Programming FlexRAM, PPoPP 2003
FlexRAM Architectural Issues

- PArrays cannot interrupt/invoke the PHost(s)
  - PArray requests are sent to chip controller
  - PHost polls the controllers and services the requests
- Communication PHost - PArray: pass input and output arguments through memory
- No HW cache coherence
  - Compiler inserted cache flushes and invalidations
Outline

- FlexRAM Architecture
- **Software Support**
- CFlex
- IMOs
- Evaluation
- Related Work
- Conclusions
Operating System Extensions

- Common address space for PHost(s) and PArrays
- PArrays kernel
  - Manages the TLB
  - Manages spawn and termination of local tasks
- PHost OS
  - Updates the shared page table
  - First-touch placement of pages
  - Cooperates with PArray kernels to keep TLBs coherent
Programming FlexRAM

- OpenMP-like directives (CFlex)
- Library of Intelligent Memory Operations (IMOs)
CFlex

❖ CFlex: family of directives inspired by OpenMP
  ❖ Execution modifiers: build/sync tasks
  ❖ Data modifiers: properties of data structures
  ❖ Executable directives: barriers, prefetches,…

#pragma FlexRAM directive-type [clauses]
Execution Modifier: Spawn

*Directive-type*

- `[phost | parray]`: specifies kind of processor to use

*Clauses*

- `on_home(x)`: run the task on the PArray on whose bank `x` is located.
- `sync/async`: parent task must stop until child finishes (sync) or not (async)
- `pfor`: parallelize for loop
Execution Modifier: Spawn (II)

- Clauses (cont.):
  - if (cond) / else: conditional execution of the compiler directive
  - shared, private, firstprivate, lastprivate, reduction: scope clauses with the same meaning as in OpenMP
  - flush: specify which pieces of data to flush from PHost cache
Example: Parallelizing a Loop

```
for(p = head; p != NULL; p = p->next)
    process(p->data);
```
Example: Parallelizing a Loop

```c
for(p = head; p != NULL; p = p->next)
    process(p->data);
```
Example: Parallelizing a Loop

```c
#pragma FlexRAM phost sync
for(p = head; p != NULL; p = p->next)
#pragma FlexRAM parray async on_home(*(p->data)) \ firstprivate(p)
    process(p->data);
```

B. B. Fraguela et al Programming FlexRAM, PPoPP 2003
Example: Parallelizing a Loop

```c
#pragma FlexRAM parray pfor on_home(*(p->data)) \ 
    firstprivate(p)

    for(p = head; p != NULL; p = p->next)
    process(p->data);
```
int TreeAdd (register tree_t *t) {
    if (t == NULL) return 0;
    else {
        int leftval, rightval;

        leftval = TreeAdd(t->left);

        rightval = TreeAdd(t->right);

        return leftval + rightval + t->val;
    }
}
int TreeAdd (register tree_t *t) {
    if (t == NULL) return 0;
    else {
        int leftval, rightval;

        #pragma FlexRAM phost sync
        {
            #pragma FlexRAM pararray async on_home(*t->tleft)) if (lcl(t->left))
            #pragma FlexRAM phost async else
                leftval = TreeAdd(t->left);
            
            #pragma FlexRAM pararray async on_home(*t->tright)) if (lcl(t->right))
                rightval = TreeAdd(t->right);
        } 
        
        return leftval + rightval + t->val;
    }
}
Outline

- FlexRAM Architecture
- Software Support
- CFlex
- IMOs
- Evaluation
- Related Work
- Conclusions
Intelligent Memory Operations (IMOs)

- Libraries that hide FlexRAM while providing near-optimal performance
- Implement common operations on data structures often used in programs
- Highly optimized, both the sequential and the parallel versions
### Example IMOs: Vector Container

<table>
<thead>
<tr>
<th>IMO Description</th>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Apply func f with arg a</strong></td>
<td><code>Vector_apply(v,f,a)</code></td>
</tr>
<tr>
<td><strong>Search element that fulfills cond f with arg a</strong></td>
<td><code>Vector_search(v,f,a)</code></td>
</tr>
<tr>
<td><strong>Generate vector with the result of appl func f with arg a</strong></td>
<td><code>v2=Vector_map(v,f,a)</code></td>
</tr>
<tr>
<td><strong>Reduce vector applying func f, whose neutrum is a</strong></td>
<td><code>Vector_reduce(v,f,a)</code></td>
</tr>
<tr>
<td><strong>Process two vectors and an arg a, generating a new vector</strong></td>
<td><code>v3=Vector_map2(v,v2,f,a)</code></td>
</tr>
</tbody>
</table>
Outline

- FlexRAM Architecture
- Software Support
- CFlex
- IMOs
- Evaluation
- Related Work
- Conclusions
Architecture Parameters

- **PHost**
  - 1.6 GHz, 5 issue
  - L1 cache: 32 KB
  - L2 cache: 1 MB

- **PArray**
  - 1.2 GHz, 2 issue in order
  - L1 cache: 8 KB
  - No FP support

- **Mem latency 180 cycles**

- **Mem latency 14 cycles**
## Applications (I)

<table>
<thead>
<tr>
<th>Application</th>
<th>Suite</th>
<th>Access</th>
<th>Data</th>
<th>Task Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>TSP</td>
<td>Olden</td>
<td>Ptr</td>
<td>FP</td>
<td>Large</td>
</tr>
<tr>
<td>TreeAdd</td>
<td>Olden</td>
<td>Ptr</td>
<td>Int</td>
<td>Large</td>
</tr>
<tr>
<td>Swim</td>
<td>SPEC OMP 2001</td>
<td>Reg</td>
<td>FP</td>
<td>Med</td>
</tr>
<tr>
<td>Mgrid</td>
<td>SPEC OMP 2001</td>
<td>Reg</td>
<td>FP</td>
<td>Var</td>
</tr>
<tr>
<td>Dmxdm</td>
<td>Kernel</td>
<td>Reg</td>
<td>FP</td>
<td>Large</td>
</tr>
<tr>
<td>Spmxv</td>
<td>Kernel</td>
<td>Ind</td>
<td>FP</td>
<td>Med</td>
</tr>
<tr>
<td>Distance</td>
<td>CAM</td>
<td>Ptr</td>
<td>Int</td>
<td>Large</td>
</tr>
<tr>
<td>Path</td>
<td>CAM</td>
<td>Ptr</td>
<td>Int</td>
<td>Small</td>
</tr>
</tbody>
</table>
Applications (II)

Original code not changed

<table>
<thead>
<tr>
<th>Application</th>
<th>Code size (lines)</th>
<th>Directives</th>
<th>Additional lines</th>
</tr>
</thead>
<tbody>
<tr>
<td>TSP</td>
<td>485</td>
<td>12</td>
<td>5</td>
</tr>
<tr>
<td>TreeAdd</td>
<td>71</td>
<td>8</td>
<td>4</td>
</tr>
<tr>
<td>Swim</td>
<td>272</td>
<td>8</td>
<td>0</td>
</tr>
<tr>
<td>Mgrid</td>
<td>470</td>
<td>13</td>
<td>0</td>
</tr>
<tr>
<td>Dmxdm</td>
<td>81</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>Spmxv</td>
<td>47</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Distance</td>
<td>108</td>
<td>17</td>
<td>7</td>
</tr>
<tr>
<td>Path</td>
<td>165</td>
<td>17</td>
<td>9</td>
</tr>
</tbody>
</table>
Speedups

Speedups: Over one order of magnitude!
Further Optimizations

- Single task for consecutive pages affected by an `on_home` residing in the same bank
- Consecutive rather than cyclic spawn among the FlexRAM chips
- Limiting the usage of `on_home` to large loops
- Increase the average speedup by about 30%
Outline

- FlexRAM Architecture
- Software Support
- CFlex
- IMOs
- Evaluation
- Related Work
- Conclusions
Other Intelligent Mem Arch.

- Active Pages [Oskin et al], DIVA [Hall et al]
  - Program the machine by hand
  - Can also be programmed with CFlex/IMOs
  - Seem more sensitive to data placement than FlexRAM
  - Further extensions of CFlex to specify alignment and placement of data structures
  - Message passing is more natural for DIVA
Related Work

- **FlexRAM [Solihin et al]:** automatic partition and mapping by a compiler
  - Only feasible for simple codes
  - Performance is limited

- **Widespread compiler directives**
  - OpenMP: UMA model, no locality clauses
  - HPF: extensive alignment + replication
  - Both: Unadequate for irregular applications
Conclusions

- Effective programming support for Intelligent Mem
  - CFlex: family of pragmas inspired by OpenMP
  - IMOs: library of intelligent memory operations
- CFlex parallelizes more problems than OpenMP
- Complexity can be further hidden using IMOs
- Speedups over one order of magnitude
Programming the FlexRAM Parallel Intelligent Memory System

B. B. Fraguela*, J. Renau†, P. Feautrier‡
D. Padua† and J. Torrellas†

*Univ. da Coruña
†Univ. of Illinois
‡ENS de Lyon
Runtime System

- Task management
  - Creation of buffer with input args + message
  - PHost spins on a termination flag set by the PArray
- Interface to chip controller locks and constructions built upon them, like barriers
- Heap memory management
- Polling of the FlexRAM chips
PArray Structure
Further Optimizations

Initial optimizations:
- Alignments using the OS first-touch policy
- `on_home` clause to exploit locality

New optimizations:
- H: single task for consecutive pages affected by an `on_home` residing in the same bank
- C: consecutive rather than cyclic spawn among the FlexRAM chips
- L: limiting the usage of `on_home` to large loops
Software Optimization Results

![Graph showing speedup in various configurations of FlexRAM chips]

- NoOpt
- Opt
- Opt+H
- Opt+H+C
- Opt+H+C+L

**B. B. Fraguela et al**
Programming FlexRAM, PPoPP 2003
Final Results