Watchdog Processor for the MEMSY Multiprocessor

Istvan Majzik

Main features:

1. A new signature assignment method:

   SEIS: Signature Encoded Instruction Stream

2. Hierarchical checking of the application

3. Integration into the system error recovery

4. Shared use of a single hardware
System architecture:

Hardware:

B level: supervisor node

VME bus

W level: watchdog (WP)

COM bus

A level: working nodes

Software:

Assigned signatures:

(application programs modified by the SEIS preprocessor)

Support of error recovery:

(checkpoint generation, rollback recovery) in the WP

Support of multiprocess systems:

compile time: unified process ID
run time: unique process ID
(translation: MMU)

Support of diagnosis:

error log for the supervisor node
Signature structure:

<table>
<thead>
<tr>
<th>Processor</th>
<th>Process</th>
<th>Procedure</th>
<th>Type</th>
<th>Label1</th>
<th>Label2</th>
<th>Label3</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Hierarchical checking:

Statement level checking:
- Encoded program control flow graph (CFG)
- Signatures identify:
  - program location +
  - valid successors
- A single reference signature for a CFG

Procedure level checking:
- Procedure return:
  - signature stack in the WP

Process level checking:
- Scheduler monitoring
- Time-out of the signature transfer
SEIS signature assignment overview:

```plaintext
procedure() {
    for (i=0; i<MAX; i++) {
        if (a>b) {
            stat1;
        } else {
            stat2;
        }
    }
}
```

```plaintext
procedure() {
    SEND(1,10,1);
    for (i=0; i<MAX; i++) {
        SEND(2,5,2);
        if (a>b) {
            SEND(3,3,3);
            stat1;
        } else {
            SEND(6,6,6);
            stat2;
        }
        SEND(4,7,4);
    } SEND(8,11,8);
}
```

Control graph extraction:

Control graph encoding:
SEIS statement label assignment: CFG processing

- Insert additional edges: \( \rightarrow \) directed Euler graph

- Compose an Eulerian circuit:

- Encode the edge trails:
  - \( c_0 = 1 \) initial
  - \( c_{i+1} = c_i + 1 \) in the trails
  - \( c_{i+1} = c_i + 2 \) start of trails

- Compose statement labels:
Statement level checker module:

*combinational signature evaluation*

Evaluation:
- compare actual statement label
- update reference label

Valid statement label:

*one of its sublabels is successor of one of the sublabels of the reference*

*here presented for 2 sublabels (A1, A2; R1, R2)*
Procedure level checking:

Signature type:
- start of procedure (SOP)
- end of procedure (EOP)
- normal signature (NRM)

Signature stack operation in the WP:

NRM: check actual signature
   actual signature → new reference

SOP: push reference
   actual signature → new reference

EOP: check actual signature
     pop reference

Summary of the checked hierarchy:

<table>
<thead>
<tr>
<th>Level</th>
<th>Checked operation</th>
<th>Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Statement</td>
<td>Statement sequence</td>
<td>Statement label</td>
</tr>
<tr>
<td>Procedure</td>
<td>Procedure return</td>
<td>Signature stack</td>
</tr>
<tr>
<td>Process</td>
<td>Scheduling</td>
<td>Process ID</td>
</tr>
<tr>
<td></td>
<td>Hung process</td>
<td>Signature timer</td>
</tr>
</tbody>
</table>
Shared WP hardware:

Multiple logical WP

- Process selector
- Stack pointer
- Signature stack
- Signature comparator
- Signature timer
- Process time limit
- RS
- RF

Procedure, process ID
Statement label
Error status

5 x 256 possible processes

RS: reference label
RF: reference proc. ID
Internal hardware architecture:

Shared resources:

*Dynamic splitting of the global stack* (linked list)
Actual checkpoint area (A, B)
Tentative checkpoint area (A, C, D)
Operational stack area (A, C, E)

Operations:
(in limited time)

Tentative checkpoint generation
Actual checkpoint generation
Recovery

Linked to the free list
Top of stack
Measurement results:

Fault coverage:

- Fault coverage (% of the previously undetected faults)

- Run time %

Error latency:

- Signatures (1000)
  - Original
  - With reduction

Instructions
Conclusions and future work:

**Advantages:**

- high speed
  
  *combinational signature evaluation*

- higher-level checks
  
  *hierarchical checker modules*

- easy integration
  
  *hw: uniform signature interface*
  
  *sw: preprocessors for different languages*
  
  *support of error recovery*

**To be improved:**

- tuning the signature transfer rate
  
  *ideal case: uniform time periods*
  
  *real case: rough granularity*

  Solution: weighting CFG with *execution times*
  
  (intermediate compiler level)

- sophisticated test of synchronization
  
  (process algebra modelling)

**To be measured:**

- performance in the multiprocessor