Técnicas de Emulação de Operações de Ponto Flutuante em Sistemas Operacionais Modernos

Lucas C. Villa Real
Edson T. Midorikawa
Marcelo K. Zuffo

Universidade de São Paulo
Escola Politécnica

Organização

- Motivação e Foco
- Introdução ao ARM
- Emuladores de ponto flutuante
  - Orientados a exceções
  - Implementados em espaço de usuário
- Experimentos e resultados de desempenho
- Prototipações desenvolvidas
- Conclusões

Motivação

- Uso de processadores especializados em eletrônicos de consumo
- Recursos de hardware limitados, softwares dedicados
- Como otimizar seu uso em projetos que exijam recursos não implementados em hardware?

Foco

- Processamento de ponto flutuante em processadores ARM
- Emulação de unidade de ponto flutuante em software
  - Kernel Linux (2.6.12)
  - Compilador GCC (3.4.5)

Introdução ao ARM

- Processador RISC com baixo consumo de energia
- Conjunto de instruções dividido em duas partes:
  - Core: load/store, movimentação de dados, semáforos, etc
  - Instruções para conversação com co-processadores

Introdução ao ARM

- 7 modos de execução:
  - *User*: modo normal de execução
  - *System*: executa tarefas privilegiadas do sistema operacional
  - *Supervisor*: modo protegido
  - *FIQ/IRQ*: tratamento rápido e normal de interrupções
  - *Abort*: implementação de memória virtual
  - *Undefined*: emulação em software para co-processadores ausentes

Co-Processadores

- Protocolo baseado em ACK.
- Caso ACK não seja recebido pela CPU:
  - Processador entra no modo *Undefined*
  - Interrupções normais são desabilitadas
  - Execução é desviada para a função definida no vetor de interrupções
  - Instrução não suportada é emulada
  - Interrupções normais são habilitadas novamente

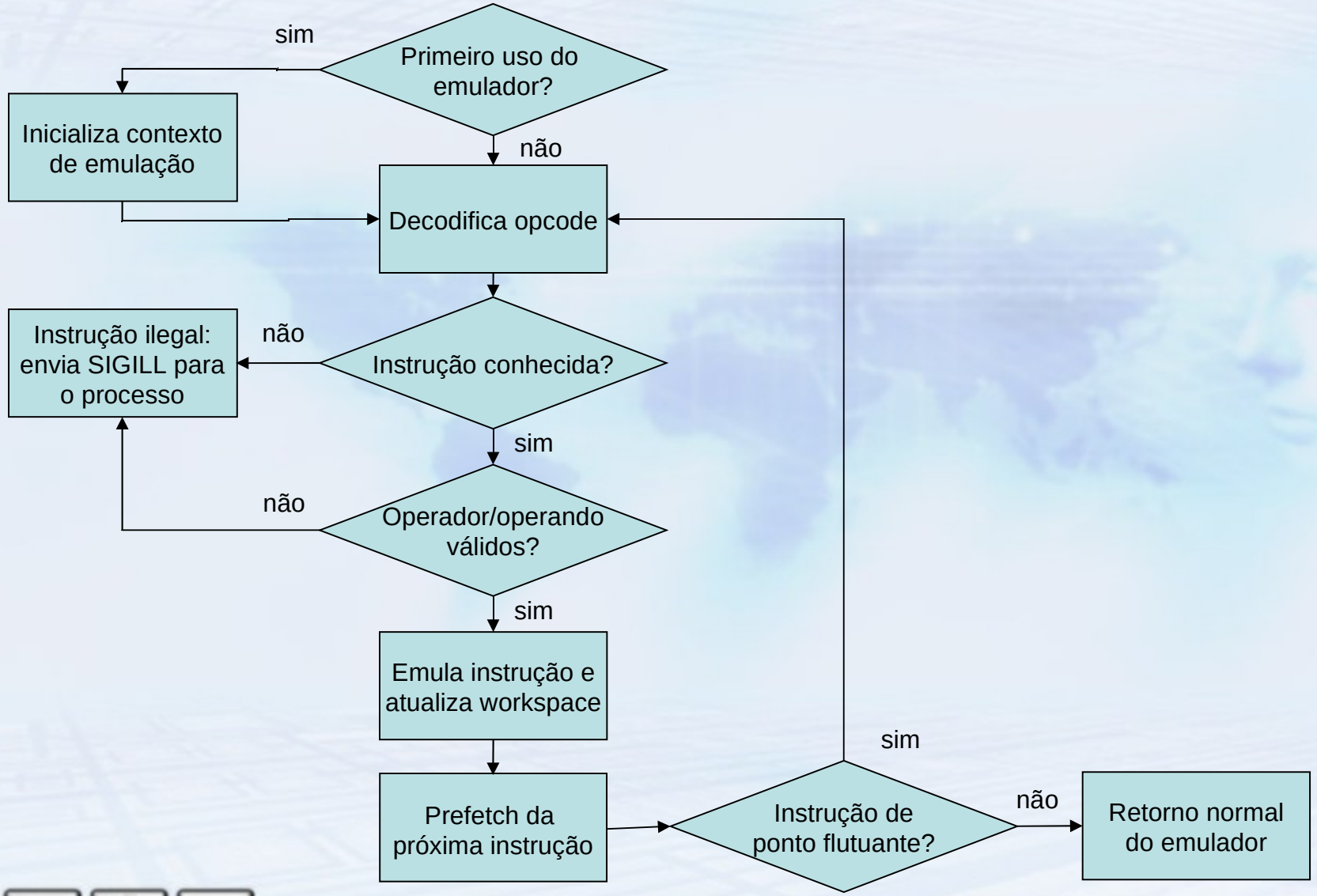
Emuladores de Ponto Flutuante

- Necessários quando um co-processador de FP não está presente na plataforma
- Podem ser implementados de duas maneiras:
  - Orientados a exceções
  - Implementados em espaço de usuário

Emuladores orientados a exceções

- Invocados no modo *Undefined*
- Implementados em kernel
- Linux: NWFPE (NetWinder Floating Point Emulator)
  - Precisões simples, dupla e dupla estendida
  - Utiliza técnicas para tentar reduzir a ocorrência de exceções

Emuladores orientados a exceções



Emuladores em espaço de usuário

- Implementados pelo compilador
- Convertem operações de FP em chamadas de função que as emulam
- Não geram exceções
- GCC para ARM: 2 implementações, em C e em assembly
  - Precisões simples e dupla

Emuladores em espaço de usuário

```
(...)
10 84ac: e59f3044      ldr r3, [pc, #68]
11 84b0: e50b3020      str r3, [fp, #-32]
12 84b4: e51b001c      ldr r0, [fp, #-28]
13 84b8: e51b1020      ldr r1, [fp, #-32]
14 84bc: eb0000e4 bl 8854 <__addsf3>
(...)
```

Experimentos e resultados de desempenho

- Medições simples, visando cobrir o uso de FP em operações básicas
  - Cálculo de médias, porcentagens, funções de hash, sementes aleatórias, etc

Experimentos e resultados de desempenho

- Software LMbench
- Operações de soma, multiplicação, divisão e combo
- 50 execuções, com 500 iterações cada em um sistema mono-usuário
- Hardware: Intel XScale, 400MHz
- Linux 2.6.12-mm2 / Glibc 2.3.6

Experimentos e resultados de desempenho

Tempos de execução no Xscale, representados em nano-segundos

	Soma	Multiplicação	Divisão	Combo
LibGCC/ASM	182,68	124,75	355,84	866,14
LibGCC/C	650,92	573,94	1013,45	3098,50
NWFPE(Emulação)	700,08	939,05	1382,33	6284,00
NWFPE(Total)	1090,08	1329,50	1772,33	6674,00

Ganhos de desempenho em relação ao tempo total do NWFPE, em %

	Soma	Multiplicação	Divisão	Combo
LibGCC/ASM	83,24	90,61	79,92	87,02
LibGCC/C	40,29	56,82	42,82	53,57
NWFPE(Emulação)	35,78	29,36	22,00	5,84

Prototipações desenvolvidas

- Instrumentação do NWFPE, via sysfs
  - /sys/nwfpe/counter
- Número de inicializações do estado da FPU
- Instruções simuladas:
  - CPDT: incrementado a cada load/store
  - CPRT: opcodes de conversão, comparação e transferência entre registradores
  - CPDO: opcodes aritméticos com 1 e 2 operandos

Prototipações desenvolvidas

- Constatações:
  - `longjmp()` e `setjmp()` salvam estado da FPU
  - `/bin/cat` invoca 7 vezes a emulação do NWFPE
  - `ssh –help` entra 38 vezes no NWFPE
  - Autenticação bem sucedida no `ssh` acarreta 435 traps
  - Inicialização do servidor X (Xorg 7.0) realiza 54391 acessos ao NWFPE
- Lembrando que a emulação ocorre com interrupções desabilitadas

Conclusões

- Análise de desempenho de 2 estratégias alternativas para a emulação de ponto flutuante
- Emulação em espaço de usuário propicia uma sensível melhora em relação ao tratamento via traps (de até 90%)
- Questão da emulação é fundamental para sistemas operacionais voltados a dispositivos embarcados
- Espaço para estudo de otimização do emulador em C e do *backend* de otimização do compilador.

Técnicas de Emulação de Operações de Ponto Flutuante em Sistemas Operacionais Modernos

Lucas C. Villa Real
Edson T. Midorikawa
Marcelo K. Zuffo

lucasvr@lsi.usp.br
edson.midorikawa@poli.usp.br
mkzuffo@lsi.usp.br