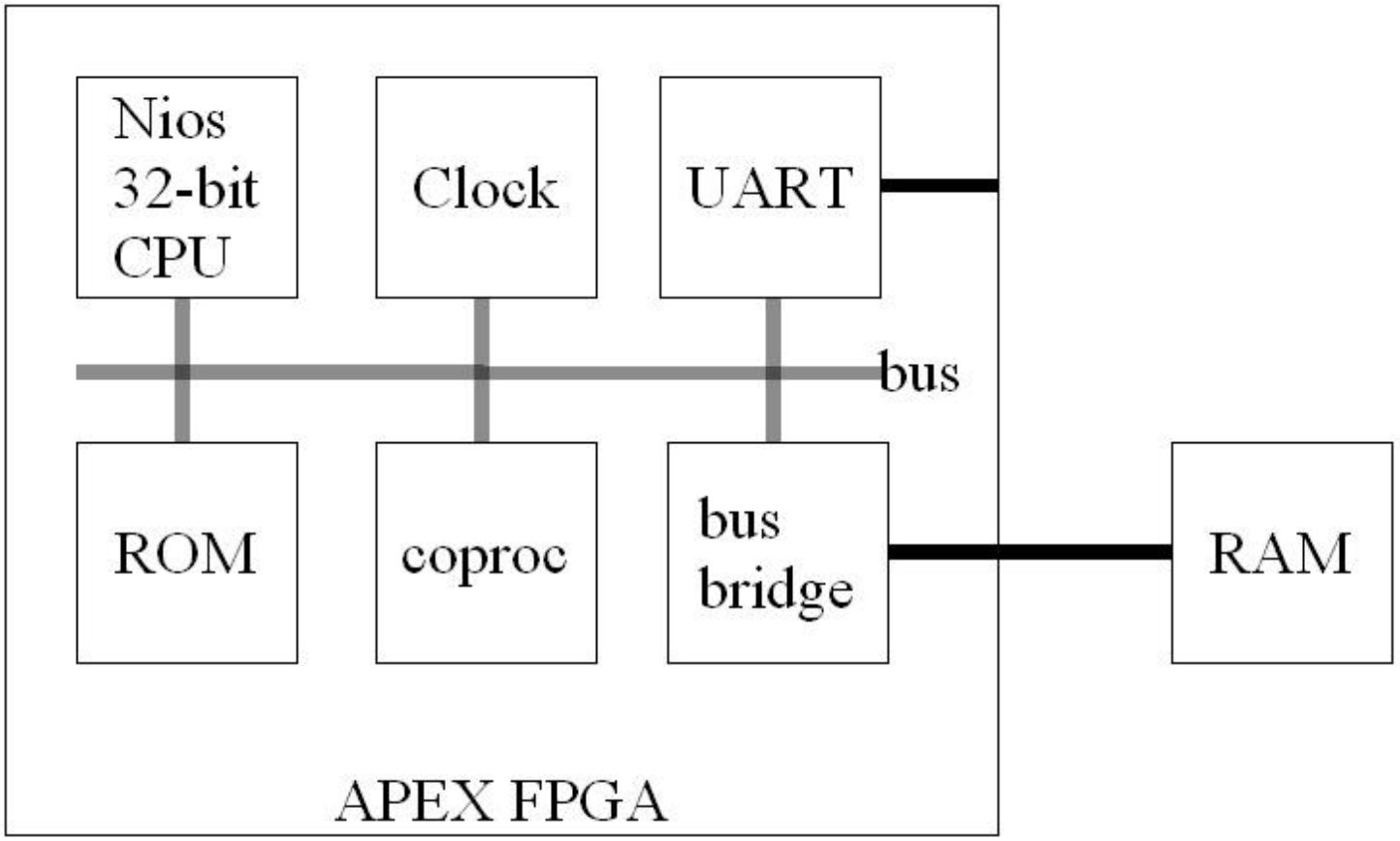
Real-time Kernel Support for Coprocessors: Empirical Study of an SoPC

Andrew Morton and Wayne M. Loucks
Electrical and Computer Engineering
University of Waterloo
Waterloo, Ontario, Canada

Outline

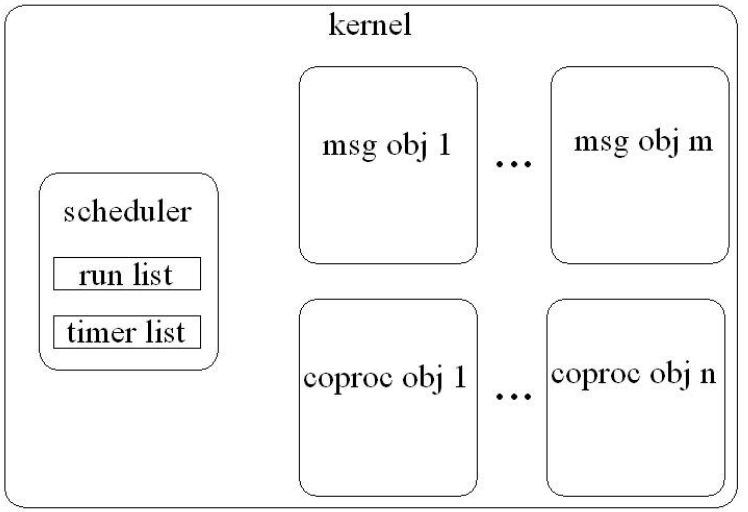
- Kernel
  - resources, tasks, scheduling
- Application
  - idle engine simulation and control
  - cordic coprocessor
- EDF Scheduling
  - with coprocessors
- Conclusions

System on Programmable Chip



Kernel

- real-time
- uni-processor
- scheduling policy
  - earliest deadline first
- resources
  - real-time clock
  - message queues
  - coprocessors



Tasks

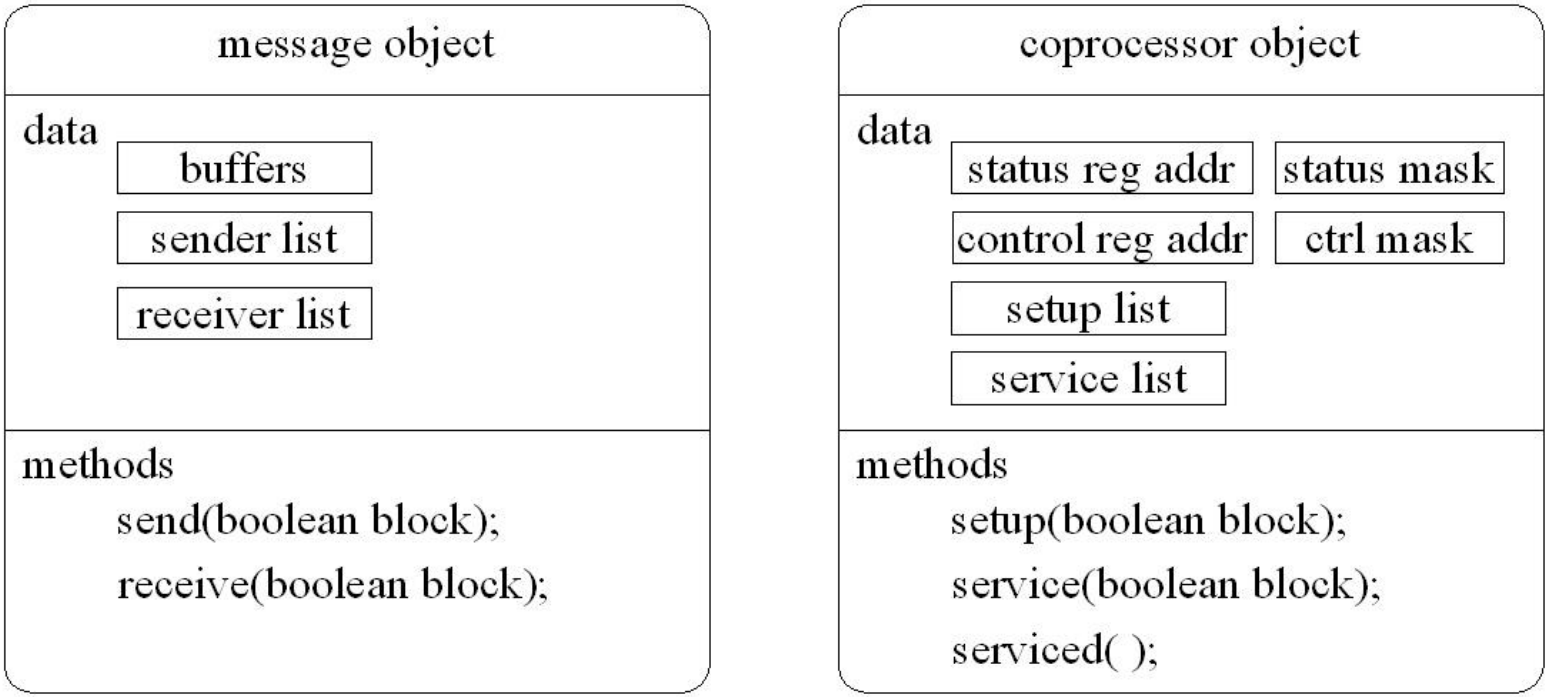
Periodic

- start time (s)
- period (T)
- relative deadline (D)
- worst-case execution time (C)

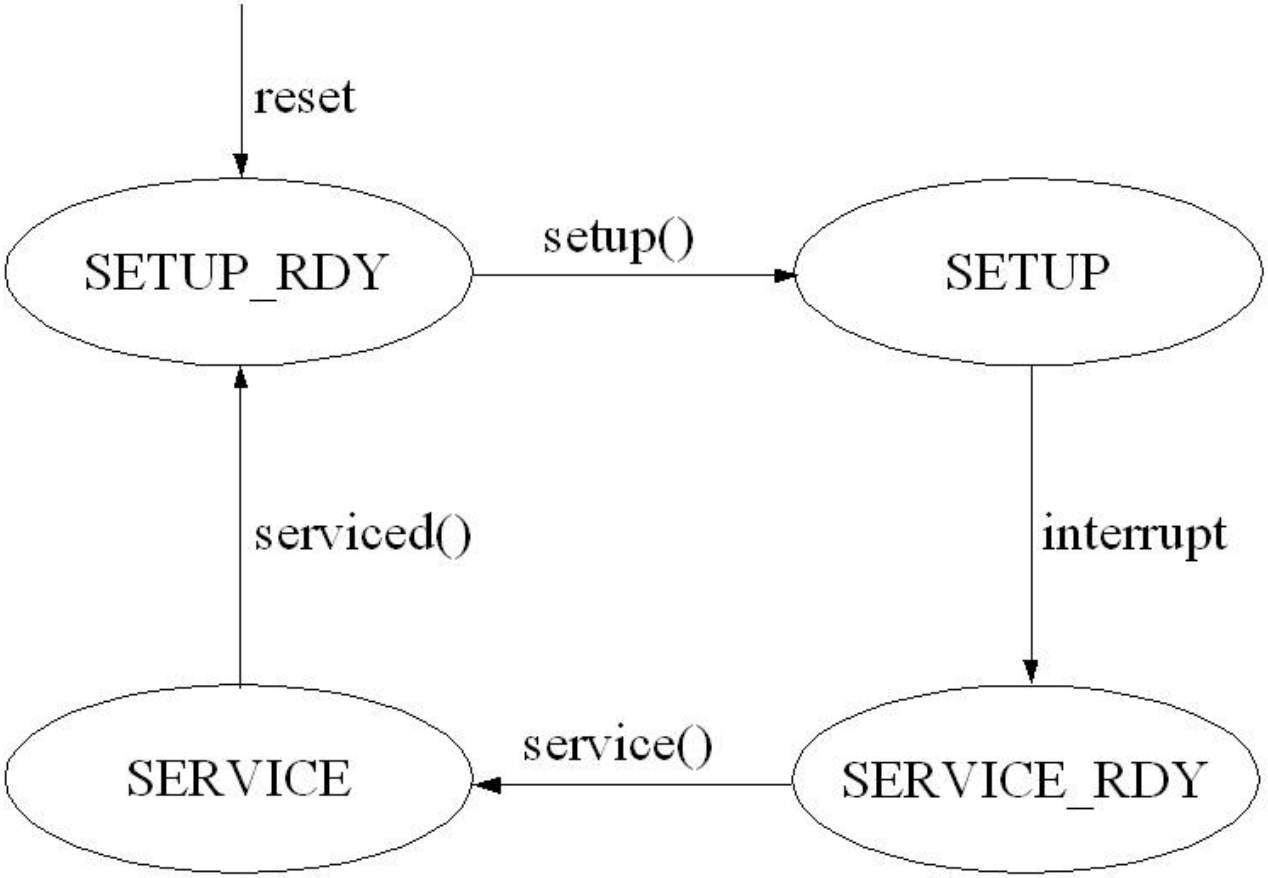
Aperiodic

- released by message arrival or coprocessor interrupt
- relative deadline (D)
- worst-case execution time (C)

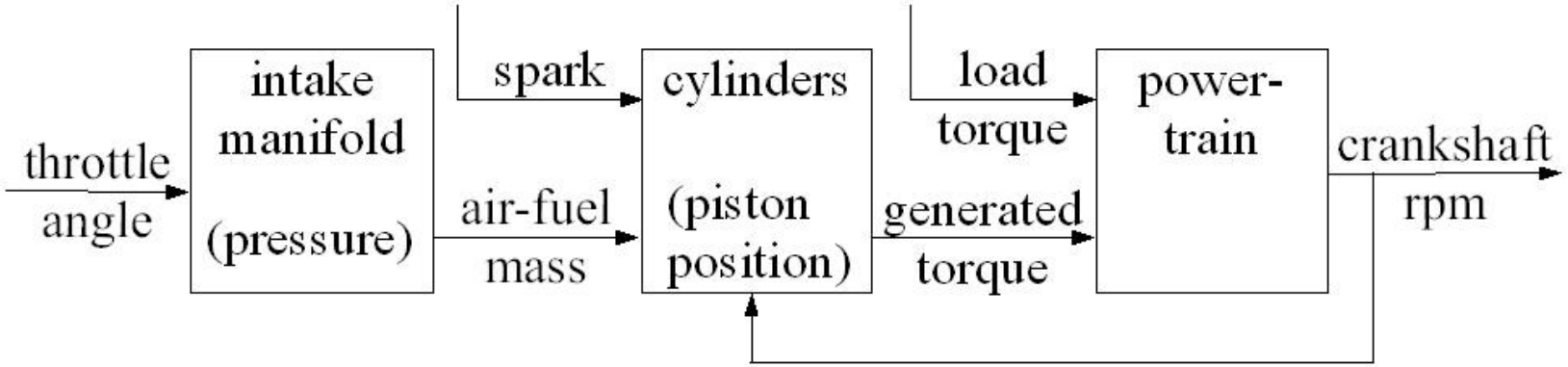
Kernel Objects



Coprocessor Object FSM



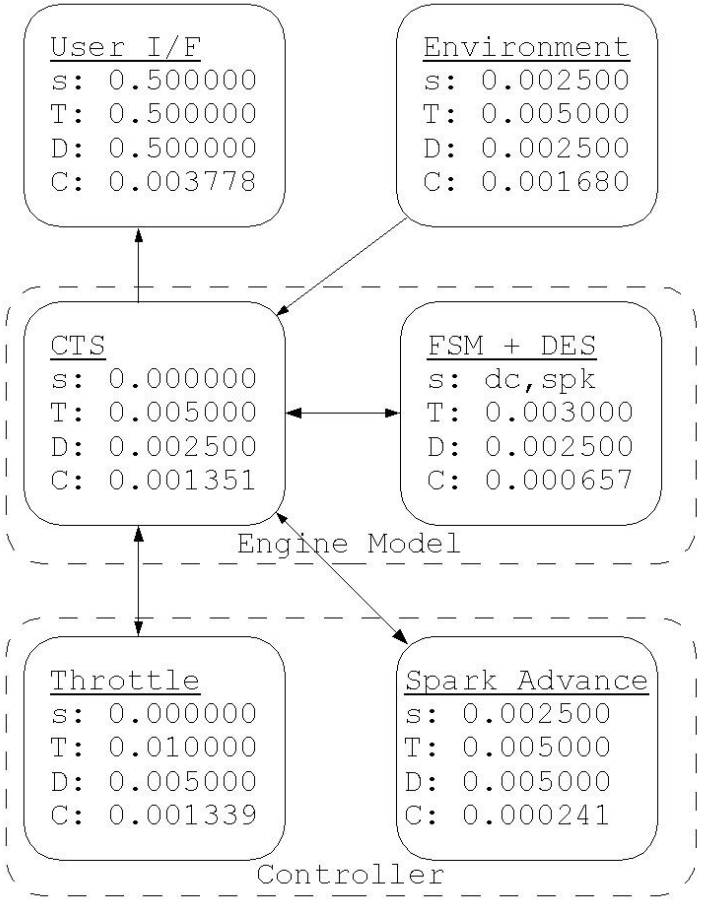
Idle Engine Model



- Continuous Time System (CTS):
  - manifold pressure, piston position, crankshaft rpm
- Discrete Event System (DES):
  - torque, air-fuel mass
- Finite State Machine (FSM):
  - piston phase (dead-centers and sparks)

Idle Engine Application

- Model
- Load Torque
- Controller
  - maintain 800±30 rpm
- User Interface



The Problem

- Processor utilization

$$U = \sum_{i=1}^{n} \frac{C_i}{T_i}$$

  - feasible if $U \leq 1$
- Idle Engine App
  - U = 1.01486
  - infeasible
  - Task Throttle late 22 times in 10 second interval

The Solution

- Task Environment
  - invokes cos( )
  - C_{cos} = 0.001435 s
  - replace with coprocessor
- Cordic
  - hardware algorithm for sine, cosine, etc.
  - C_{cordic} = 0.000081 s

Software Modification

Before

```
cosine = cos(angle);
```

After

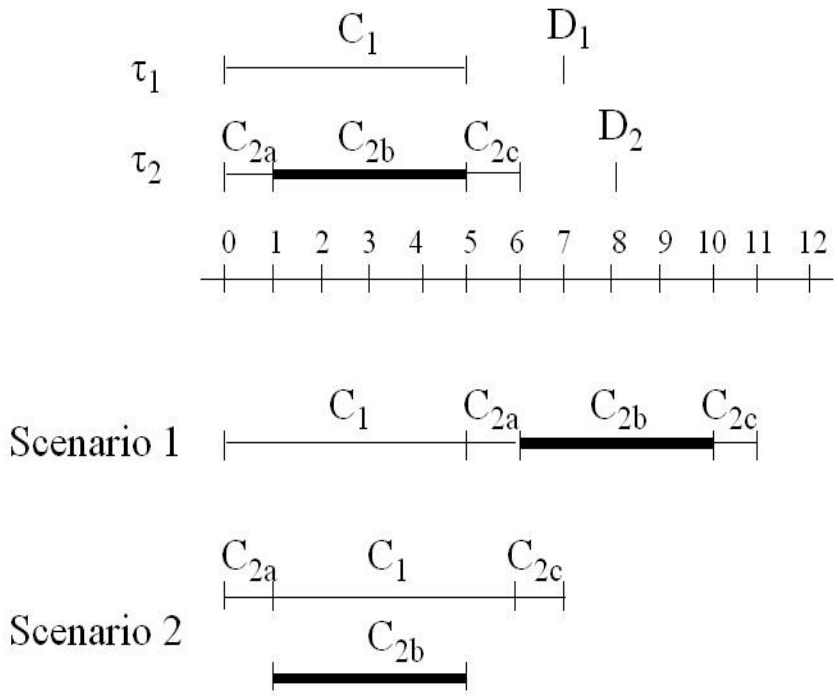
```
system.coproc[0].setup();
cordic->data = angle;
cordic->go = 0;
cordic->statCtrl =
  Cordic::DoneMask;
system.coproc[0].service(status);
cosine = cordic.data;
system.coproc[0].serviced();
```

Execution Times

cos( )	0.001435 s
cordic	0.000081 s
Env with cos( )	0.001680 s
Env with cordic	0.000343 s

- Idle Engine App
  - U = 0.7474576
  - no late tasks

Issue #1: EDF with Coprocessor



- Scenario 1
  - task 2 late
- Scenario 2
  - modify deadline
  - $d_{2a} = d_2 - (C_{2b}+C_{2c})$
  - $d_{2c} = d_2$
  - both finish on time

Issue #2: Shared Coprocessor

- Task FSM+DES
  - invokes sqrt( )
  - replace with cordic
  - execution time decreases
    - with sqrt( ) 0.000657 s
    - with cordic 0.000533 s
  - however, no application speedup
- Factors
  - setup and service time, extra context switches, contention for coprocessor

Conclusions

- Kernel supports hardware coprocessors via coprocessor objects
  - little custom synthesis required to integrate hardware
- Application demonstrates coprocessor issues
  - modified scheduling requirement
  - limited benefit
- Future Work
  - EDF analysis for tasks using coprocessors