CS122A: Intermediate Embedded and Real Time Operating Systems

Jeffrey McDaniel

University of California, Riverside

Using sleep to reduce power consumption

- Microcontrollers consume power while running programs
- Instructions such as:

```
while (!timerFlag) {...}
```

- These loops are wasteful because nothing can happen until the *ISR* is called
- Microcontrollers have a **sleep** mode which consumes less power
- The execution of the program is stopped in **sleep** mode, but the hardware continues to function.

Using sleep to reduce power consumption

- In general it is challenging to know when to call sleep
- The disciplined SynchSM approach simplifies the issue.
- **Sleep** can be called while the user waits for *TimerISR* to be called

```
while(1) {
  Sleep(); // Put processor in low-power mode
  while(!TimerFlag); // Once ISR is called
  // processor wakes up and continues executing
}
```

Using sleep to reduce power consumption

- Assume a microcontroller consumes $1mW$ while running
- and $1\mu W$ when asleep

Using sleep to reduce power consumption

- Assume a microcontroller consumes $1mW$ while running
- and $1\mu W$ when asleep
- A small battery stores 1,000 Joules (J)

Using sleep to reduce power consumption

- Assume a microcontroller consumes $1mW$ while running
- and $1\mu W$ when asleep
- A small battery stores 1,000 Joules (J)
- If the microcontroller is always running:

$$Lifetime = 1,000J * (1s/0.001J)$$

Using sleep to reduce power consumption

- Assume a microcontroller consumes $1mW$ while running
- and $1\mu W$ when asleep
- A small battery stores 1,000 Joules (J)
- If the microcontroller is always running:

$$Lifetime = 1,000,000s$$

Using sleep to reduce power consumption

- Assume a microcontroller consumes $1mW$ while running
- and $1\mu W$ when asleep
- A small battery stores 1,000 Joules (J)
- If the microcontroller is always running:

$$Lifetime = 277 hours$$

Using sleep to reduce power consumption

- Assume a microcontroller consumes $1mW$ while running
- and $1\mu W$ when asleep
- A small battery stores 1,000 Joules (J)
- If the microcontroller is always running:

$$Lifetime = 11days$$

Using sleep to reduce power consumption

- Assume a microcontroller consumes $1mW$ while running
- and $1\mu W$ when asleep
- A small battery stores 1,000 Joules (J)
- If the microcontroller is always running:

$$Lifetime = 11days$$

- If the microcontroller is asleep 99% of the time:

Using sleep to reduce power consumption

- Assume a microcontroller consumes $1mW$ while running
- and $1\mu W$ when asleep
- A small battery stores 1,000 Joules (J)
- If the microcontroller is always running:

$$Lifetime = 11days$$

- If the microcontroller is asleep 99% of the time:

$$Lifetime = 1,000J * (1\% * 1/0.001) + (99\% * (1/0.000001))$$

Using sleep to reduce power consumption

- Assume a microcontroller consumes $1mW$ while running
- and $1\mu W$ when asleep
- A small battery stores 1,000 Joules (J)
- If the microcontroller is always running:

$$Lifetime = 11days$$

- If the microcontroller is asleep 99% of the time:

$$Lifetime = 990,010,000s$$

Using sleep to reduce power consumption

- Assume a microcontroller consumes $1mW$ while running
- and $1\mu W$ when asleep
- A small battery stores 1,000 Joules (J)
- If the microcontroller is always running:

$$Lifetime = 11 days$$

- If the microcontroller is asleep 99% of the time:

$$Lifetime = 275,002 hours$$

Using sleep to reduce power consumption

- Assume a microcontroller consumes $1mW$ while running
- and $1\mu W$ when asleep
- A small battery stores 1,000 Joules (J)
- If the microcontroller is always running:

$$Lifetime = 11 days$$

- If the microcontroller is asleep 99% of the time:

$$Lifetime = 11,458 days$$

Using sleep to reduce power consumption

- Assume a microcontroller consumes $1mW$ while running
- and $1\mu W$ when asleep
- A small battery stores 1,000 Joules (J)
- If the microcontroller is always running:

$$Lifetime = 11 days$$

- If the microcontroller is asleep 99% of the time:

$$Lifetime = 11,458 days$$

- 1,041.65x as long when asleep 99% of the time

Using sleep to reduce power consumption

- Aperiodic tasks allow for more optimization
- Dynamically change the Timer's tick rate to GCD of *active* tasks

Task 1: SynchSM

- WCET 10 ms
- Period 500 ms

Using sleep to reduce power consumption

- Aperiodic tasks allow for more optimization
- Dynamically change the Timer's tick rate to GCD of *active* tasks

Task 1: SynchSM

- WCET 10 ms
- Period 500 ms

Task 2: SynchSM

- WCET 50 ms
- Period 1000 ms

Using sleep to reduce power consumption

- Aperiodic tasks allow for more optimization
- Dynamically change the Timer's tick rate to GCD of *active* tasks

Task 1: SynchSM
- WCET 10 ms
- Period 500 ms

Task 2: SynchSM
- WCET 50 ms
- Period 1000 ms

Task 3: SynchSM
- WCET 5 ms
- Period 100 ms

Using sleep to reduce power consumption

- Aperiodic tasks allow for more optimization
- Dynamically change the Timer's tick rate to GCD of *active* tasks

Task 1: SynchSM
- WCET 10 ms
- Period 500 ms

Task 2: SynchSM
- WCET 50 ms
- Period 1000 ms

Task 3: SynchSM
- WCET 5 ms
- Period 100 ms

System Period: 100ms

Using sleep to reduce power consumption

- Aperiodic tasks allow for more optimization
- Dynamically change the Timer's tick rate to GCD of *active* tasks

Task 1: SynchSM
- WCET 10 ms
- Period 500 ms

Task 2: SynchSM
- WCET 50 ms
- Period 1000 ms

Task 3: SynchSM
- WCET 5 ms
- Period 100 ms

System Period: 100ms
$Utilization = ((10 * 2) + (50 * 1) + (5 * 10))/1000$

Using sleep to reduce power consumption

- Aperiodic tasks allow for more optimization
- Dynamically change the Timer's tick rate to GCD of *active* tasks

Task 1: SynchSM
- WCET 10 ms
- Period 500 ms

Task 2: SynchSM
- WCET 50 ms
- Period 1000 ms

Task 3: SynchSM
- WCET 5 ms
- Period 100 ms

System Period: 100ms
$Utilization = 120/1000$

Using sleep to reduce power consumption

- Aperiodic tasks allow for more optimization
- Dynamically change the Timer's tick rate to GCD of *active* tasks

Task 1: SynchSM
- WCET 10 ms
- Period 500 ms

Task 2: SynchSM
- WCET 50 ms
- Period 1000 ms

Task 3: SynchSM
- WCET 5 ms
- Period 100 ms

System Period: 100ms
Utilization = 12%

Using sleep to reduce power consumption

- Aperiodic tasks allow for more optimization
- Dynamically change the Timer's tick rate to GCD of *active* tasks

Task 1: SynchSM
- WCET 10 ms
- Period 500 ms

Task 2: SynchSM
- WCET 50 ms
- Period 1000 ms

Task 3: SynchSM
- WCET 5 ms
- Period 100 ms

System Period: 100ms
$Utilization = 12\%$
$Lifetime = 1,000J * (12\% * 1/0.001) + (88\% * (1/0.000001))$

Using sleep to reduce power consumption

- Aperiodic tasks allow for more optimization
- Dynamically change the Timer's tick rate to GCD of *active* tasks

Task 1: SynchSM
- WCET 10 ms
- Period 500 ms

Task 2: SynchSM
- WCET 50 ms
- Period 1000 ms

Task 3: SynchSM
- WCET 5 ms
- Period 100 ms

System Period: 100ms
$Utilization = 12\%$
$Lifetime = 880,120,000s$

Using sleep to reduce power consumption

- Aperiodic tasks allow for more optimization
- Dynamically change the Timer's tick rate to GCD of *active* tasks

Task 1: SynchSM
- WCET 10 ms
- Period 500 ms

Task 2: SynchSM
- WCET 50 ms
- Period 1000 ms

Task 3: SynchSM
- WCET 5 ms
- Period 100 ms

System Period: 100ms
$Utilization = 12\%$
$Lifetime = 244,478 hours$

Using sleep to reduce power consumption

- Aperiodic tasks allow for more optimization
- Dynamically change the Timer's tick rate to GCD of *active* tasks

Task 1: SynchSM
- WCET 10 ms
- Period 500 ms

Task 2: SynchSM
- WCET 50 ms
- Period 1000 ms

Task 3: SynchSM
- WCET 5 ms
- Period 100 ms

System Period: 100ms
$Utilization = 12\%$
$Lifetime = 10,187 days$

Using sleep to reduce power consumption

- Aperiodic tasks allow for more optimization
- Dynamically change the Timer's tick rate to GCD of *active* tasks

Task 1: SynchSM

- WCET 10 ms
- Period 500 ms

Task 2: SynchSM

- WCET 50 ms
- Period 1000 ms

Task 3: Triggered SM

- WCET 5 ms
- Period 100 ms

System Period: 500ms

Using sleep to reduce power consumption

- Aperiodic tasks allow for more optimization
- Dynamically change the Timer's tick rate to GCD of *active* tasks

Task 1: SynchSM

- WCET 10 ms
- Period 500 ms

Task 2: SynchSM

- WCET 50 ms
- Period 1000 ms

Task 3: Triggered SM

- WCET 5 ms
- Period 100 ms

System Period: 500ms
$Utilization = ((10 * 2) + (50 * 1))/1000$

Using sleep to reduce power consumption

- Aperiodic tasks allow for more optimization
- Dynamically change the Timer's tick rate to GCD of *active* tasks

Task 1: SynchSM

- WCET 10 ms
- Period 500 ms

Task 2: SynchSM

- WCET 50 ms
- Period 1000 ms

Task 3: Triggered SM

- WCET 5 ms
- Period 100 ms

System Period: 500ms
Utilization $= 70/1000$

Using sleep to reduce power consumption

- Aperiodic tasks allow for more optimization
- Dynamically change the Timer's tick rate to GCD of *active* tasks

Task 1: SynchSM

- WCET 10 ms
- Period 500 ms

Task 2: SynchSM

- WCET 50 ms
- Period 1000 ms

Task 3: Triggered SM

- WCET 5 ms
- Period 100 ms

System Period: 500ms
Utilization $= 7\%$

Using sleep to reduce power consumption

- Aperiodic tasks allow for more optimization
- Dynamically change the Timer's tick rate to GCD of *active* tasks

Task 1: SynchSM
- WCET 10 ms
- Period 500 ms

Task 2: SynchSM
- WCET 50 ms
- Period 1000 ms

Task 3: Triggered SM
- WCET 5 ms
- Period 100 ms

System Period: 500ms
$Utilization = 7\%$
$Lifetime = 1,000J * (7\% * 1/0.001) + (93\% * (1/0.000001))$

Using sleep to reduce power consumption

- Aperiodic tasks allow for more optimization
- Dynamically change the Timer's tick rate to GCD of *active* tasks

Task 1: SynchSM
- WCET 10 ms
- Period 500 ms

Task 2: SynchSM
- WCET 50 ms
- Period 1000 ms

Task 3: Triggered SM
- WCET 5 ms
- Period 100 ms

System Period: 500ms
$Utilization = 7\%$
$Lifetime = 930,070,000s$

Using sleep to reduce power consumption

- Aperiodic tasks allow for more optimization
- Dynamically change the Timer's tick rate to GCD of *active* tasks

Task 1: SynchSM

- WCET 10 ms
- Period 500 ms

Task 2: SynchSM

- WCET 50 ms
- Period 1000 ms

Task 3: Triggered SM

- WCET 5 ms
- Period 100 ms

System Period: 500ms
$Utilization = 7\%$
$Lifetime = 258,353 hours$

Using sleep to reduce power consumption

- Aperiodic tasks allow for more optimization
- Dynamically change the Timer's tick rate to GCD of *active* tasks

Task 1: SynchSM
- WCET 10 ms
- Period 500 ms

Task 2: SynchSM
- WCET 50 ms
- Period 1000 ms

Task 3: Triggered SM
- WCET 5 ms
- Period 100 ms

System Period: 500ms
$Utilization = 7\%$
$Lifetime = 10,764 days$

Using sleep to reduce power consumption

- Aperiodic tasks allow for more optimization
- Dynamically change the Timer's tick rate to GCD of *active* tasks

Task 1: SynchSM
- WCET 10 ms
- Period 500 ms

Task 2: SynchSM
- WCET 50 ms
- Period 1000 ms

Task 3: Triggered SM
- WCET 5 ms
- Period 100 ms

System Period: 500ms
$Utilization = 7\%$
$Lifetime = 10,764 days$
1.06x as long