

Time Synchronization in Wireless Sensor Networks

Su Ping



Intel Research
laboratory @ berkeley

The need for time synchronization in Wireless Sensor Networks

- Better localization performance
- Energy efficient scheduling
- Event time stamping
- Coordinate future event

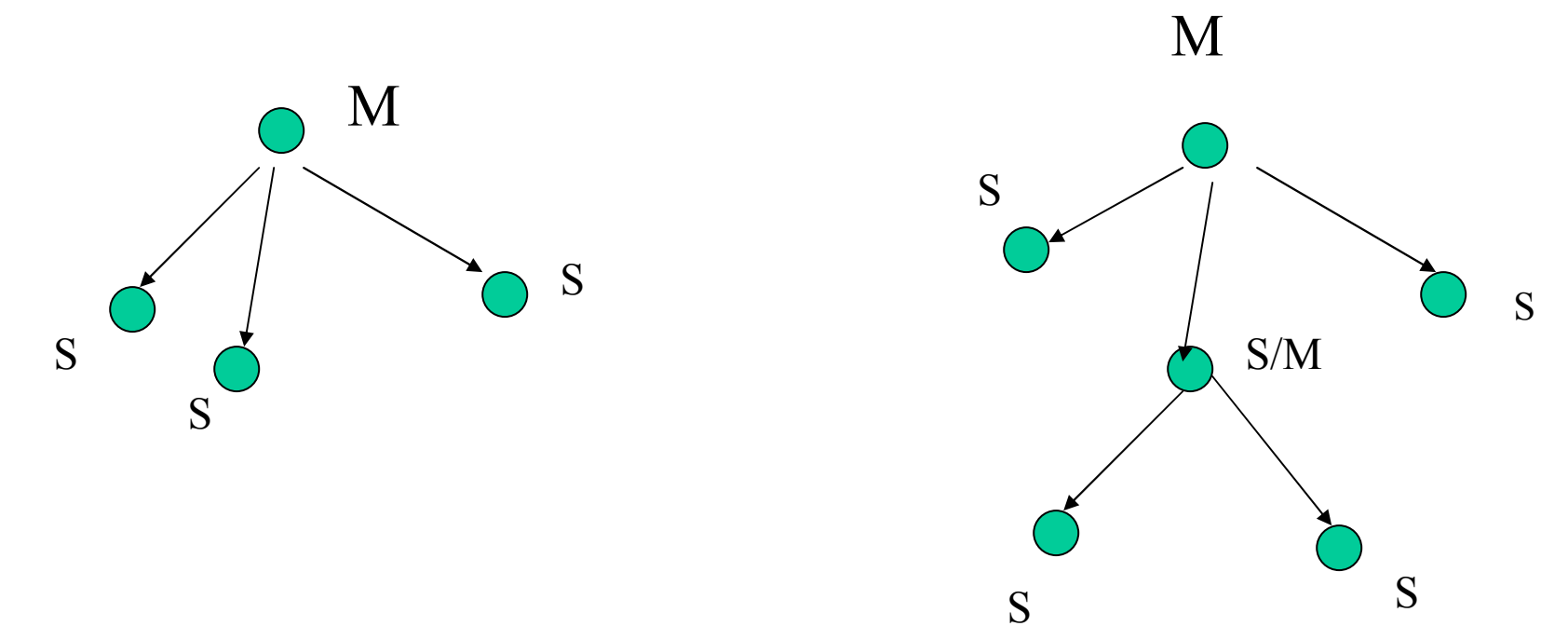
Mica HW clock

- Async. With system Clock
- 8 bits timer/counter
- External crystal
 - F = 32,768 Hz
 - Skew = 20 ppm
 - Scalable

SW clock: Logical Time

- Built on top of HW Clock
- 64 bits time counter
- Precision: 1 binary us
- Resolution: 1/32768, 1/4096, 1/1024, 1/512

Synchronize logical time



- Master- slave model
- Master broadcast its time periodically
- Slave adjust its logical time to Master's
- If a mote has dual personality, it broadcast its time after adjust its own time

Master Selection

Mote of smallest id is selected as master.
Mote 0 is master by default

- If multiple TS message is received, the one with smallest leader level wins.

TS message format:

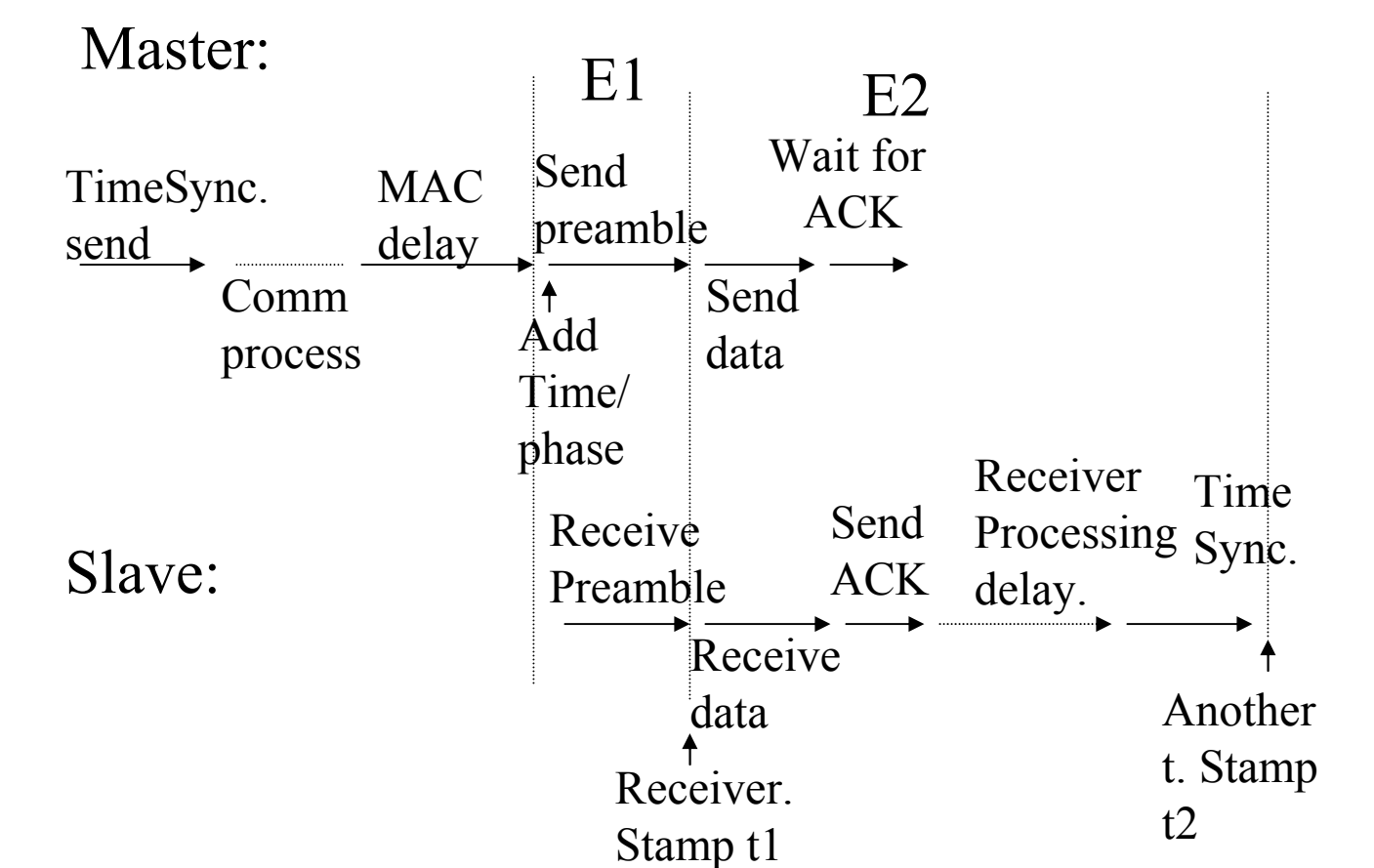
```
struct TimeSyncMsg {
    uint16_t source_addr;
    uint8_t sub_type;
    uint8_t source_status;
    uint32_t timeH;
    uint32_t timeL;
    uint8_t phase;
    uint8_t hop_count;
};
```

Source of time sync errors

- Media access time in sender side
- Sender's transmit time
- Receiver processing time
- Radio propagation time

Reduce Time sync errors

- Eliminate media access error by add logical time after MAC delay
 - Compensate delay for sending preamble/start symbols (E1)
 - Using 16 bit HW timer1 to calculate data transmit time & receiver processing time
- $E2 = t2 - t1$



Time sync accuracy

- 2 us RF sync error
- + Radio propagation time
- + E1 estimation error
- E2 estimation error

Event Sync using the synchronized logical time

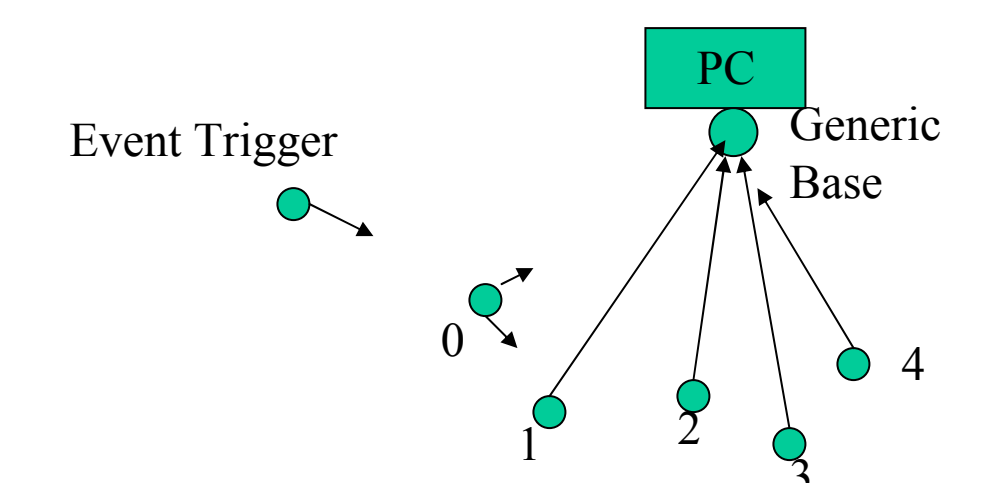
Schedule event using AbsoluteTimer

```
interface AbsoluteTimer {
    command result_t set(tos_time_t t);
    command result_t cancel();
    event result_t fired();
}
```

Accuracy = TS accuracy + Clock resolution + ATimer processing time

A time sync demo

- All mote sync with mote 0
- When a event is detected all mote record the time and send it out over RF
- Generic base collect the data and send to PC



LEDs:
Green: Event received
Red: send a time stamp
Yellow: receive a sync msg