Wireless Sensor Networks

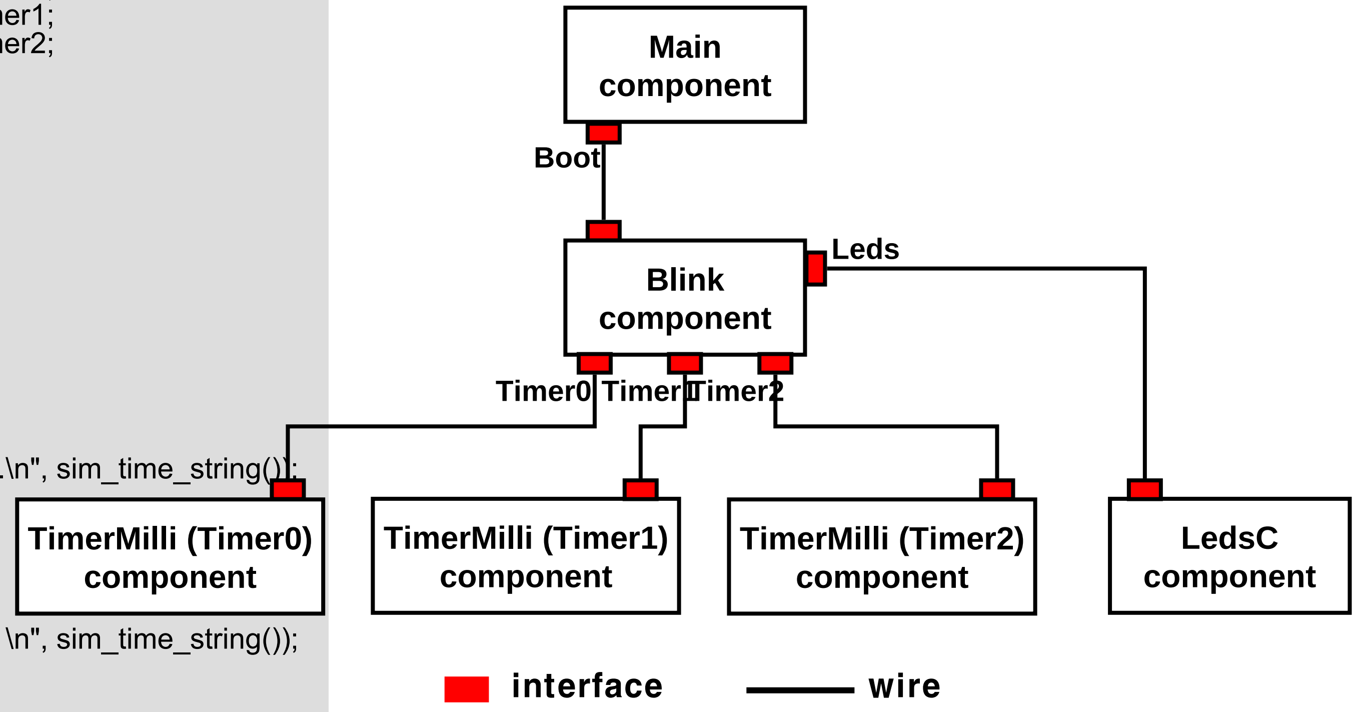
Interface

Lecturer: Junseok KIM
http://usn.konkuk.ac.kr/~jskim

Blink Example

- If a component uses a interface, it can call the interface's commands and must implement handlers for the interface's events.

```
-- Blink Component --
module BlinkC @safe()
{
  uses interface Timer<TMilli> as Timer0;
  uses interface Timer<TMilli> as Timer1;
  uses interface Timer<TMilli> as Timer2;
  uses interface Leds;
  uses interface Boot;
}
implementation
{
  event void Boot.booted()
  {
    call Timer0.startPeriodic( 250 );
    call Timer1.startPeriodic( 500 );
    call Timer2.startPeriodic( 1000 );
  }

  event void Timer0.fired()
  {
    dbg("BlinkC", "Timer 0 fired @ %s.\n", sim_time_string());
    call Leds.led0Toggle();
  }

  event void Timer1.fired()
  {
    dbg("BlinkC", "Timer 1 fired @ %s \n", sim_time_string());
    call Leds.led1Toggle();
  }

  event void Timer2.fired()
  {
    dbg("BlinkC", "Timer 2 fired @ %s.\n", sim_time_string());
    call Leds.led2Toggle();
  }
}
```

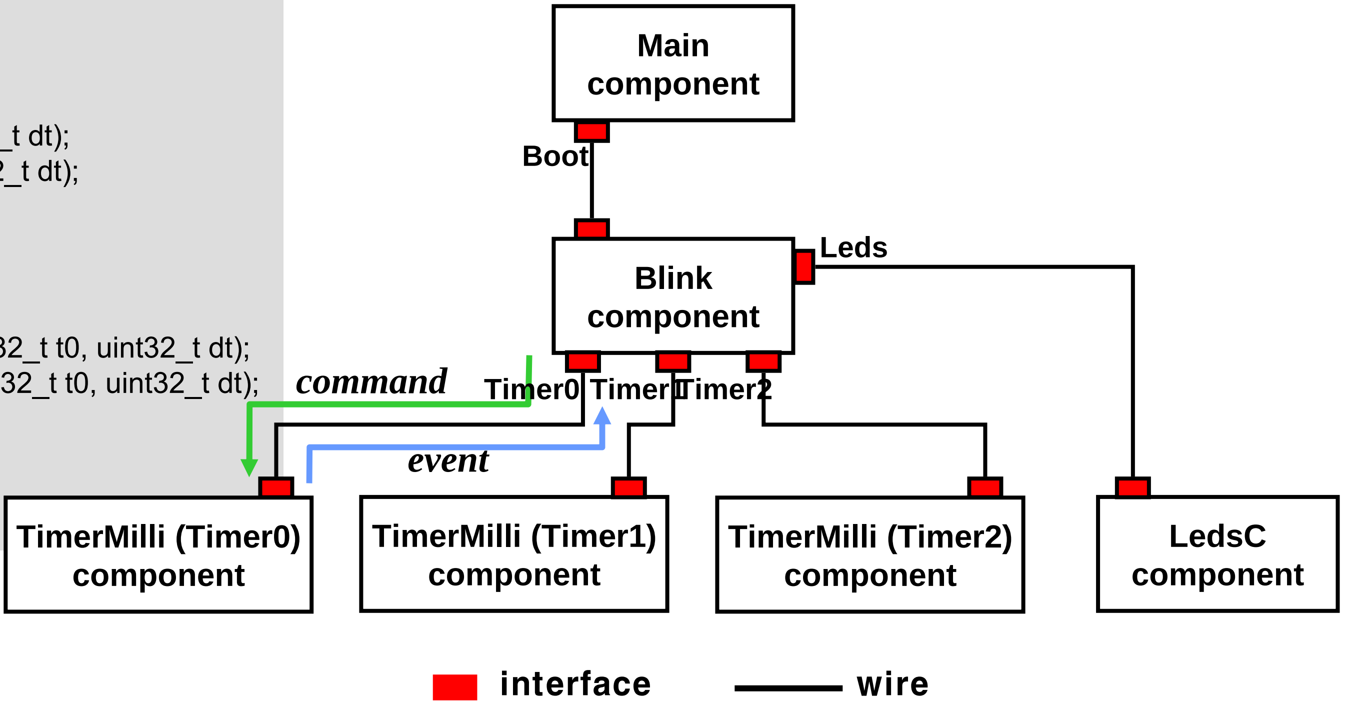


Main component
Boot
Blink component
Leds
Timer0
Timer1
Timer2
TimerMilli (Timer0) component
TimerMilli (Timer1) component
TimerMilli (Timer2) component
LedsC component
interface
wire

Interface Example: Timer

- There are two types of function: command and event function
  - command: give command from upper to lower layer to do something
  - event: give event from lower to upper layer with results of previous command

```
#include "Timer.h"

interface Timer<precision_tag>
{
   command void startPeriodic(uint32_t dt);
   command void startOneShot(uint32_t dt);
   command void stop();
   event void fired();
   command bool isRunning();
   command bool isOneShot();
   command void startPeriodicAt(uint32_t t0, uint32_t dt);
   command void startOneShotAt(uint32_t t0, uint32_t dt);
   command uint32_t getNow();
   command uint32_t gett0();
   command uint32_t getdt();
}
```



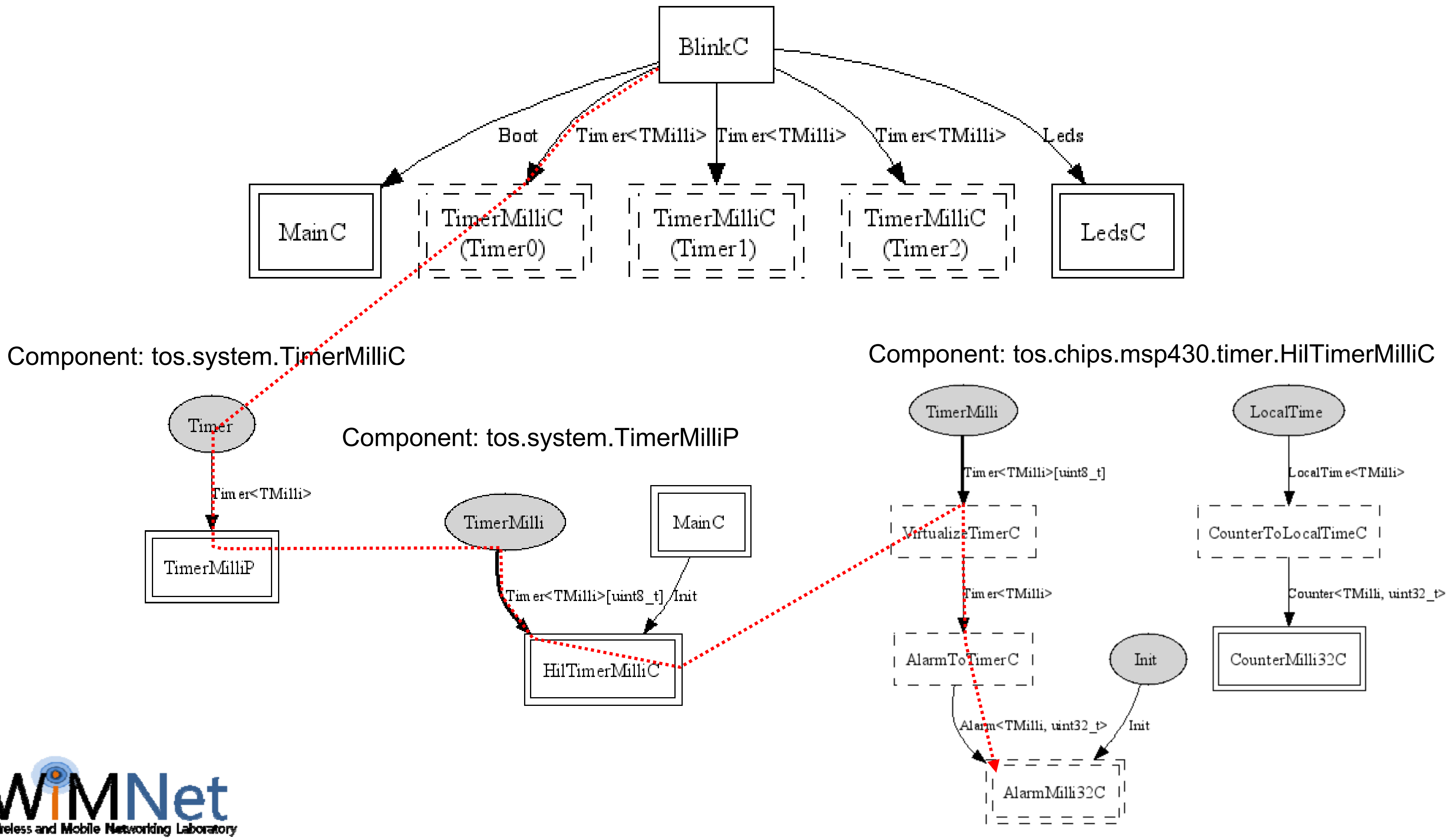
Blink Example

- If a component uses a interface, it can call the interface's commands and must implement handlers for the interface's events.

```
-- Blink Component --
module BlinkC @safe()
{
  uses interface Timer<TMilli> as Timer0;
  uses interface Timer<TMilli> as Timer1;
  uses interface Timer<TMilli> as Timer2;
  uses interface Leds;
  uses interface Boot;
}
implementation
{
  event void Boot.booted()
  {
    call Timer0.startPeriodic( 250 );
    call Timer1.startPeriodic( 500 );
    call Timer2.startPeriodic( 1000 );
  }

  event void Timer0.fired()
  {
    dbg("BlinkC", "Timer 0 fired @ %s.\n", sim_time_string());
    call Leds.led0Toggle();
  }

  event void Timer1.fired()
  {
    dbg("BlinkC", "Timer 1 fired @ %s \n", sim_time_string());
    call Leds.led1Toggle();
  }

  event void Timer2.fired()
  {
    dbg("BlinkC", "Timer 2 fired @ %s.\n", sim_time_string());
    call Leds.led2Toggle();
  }
}
```

Make Document for Analyzing

- make hybus docs
  - Graphviz program should be installed ahead



Base Components for Timer: AlarmToTimerC & Msp430AlarmC

```
-- AlarmToTimerC --
#include "Timer.h"

generic module AlarmToTimerC(typedef precision_tag) @safe()
{
  provides interface Timer<precision_tag>;
  uses interface Alarm<precision_tag,uint32_t>;
}
implementation
{
  uint32_t m_dt;
  bool m_oneshot;

  void start(uint32_t t0, uint32_t dt, bool oneshot)
  {
    m_dt = dt;
    m_oneshot = oneshot;
    call Alarm.startAt(t0, dt);
  }

  command void Timer.startPeriodic(uint32_t dt)
  { start(call Alarm.getNow(), dt, FALSE); }

  command void Timer.startOneShot(uint32_t dt)
  { start(call Alarm.getNow(), dt, TRUE); }

  command void Timer.stop()
  { call Alarm.stop(); }

  task void fired()
  {
    if(m_oneshot == FALSE)
      start(call Alarm.getAlarm(), m_dt, FALSE);
    signal Timer.fired();
  }

  async event void Alarm.fired()
  { post fired(); }
...
}
```

```
-- Msp430AlarmC --
generic module Msp430AlarmC(typedef frequency_tag) @safe()
{
  provides interface Init;
  provides interface Alarm<frequency_tag,uint16_t> as Alarm;
  uses interface Msp430Timer;
  uses interface Msp430TimerControl;
  uses interface Msp430Compare;
}
implementation
{
...
async command void Alarm.startAt( uint16_t t0, uint16_t dt )
  {
    atomic
    {
      uint16_t now = call Msp430Timer.get();
      uint16_t elapsed = now - t0;
      if( elapsed >= dt )
      {
        call Msp430Compare.setEventFromNow(2);
      }
      else
      {
        uint16_t remaining = dt - elapsed;
        if( remaining <= 2 )
          call Msp430Compare.setEventFromNow(2);
        else
          call Msp430Compare.setEvent( now+remaining );
      }
      call Msp430TimerControl.clearPendingInterrupt();
      call Msp430TimerControl.enableEvents();
    }
  }
 async event void Msp430Compare.fired()
  {
    call Msp430TimerControl.disableEvents();
    signal Alarm.fired();
  }
  ...
```

Q and A