ANRG

Implicit Network Time Synchronization

Pedro Henrique Gomes

What is it?

Objective:

- Synchronize the clocks of all nodes in a network

Approach:

- It was supposed to be **Implicit** – Need no explicit time synchronization messages
- Uses underlying **radio driver** to **timestamp** all messages, incoming and outgoing
- Currently only works on **Tmote Sky**
- Every node has an **authority level**
- Higher authorities synchronize the lower ones

How to enable it?

- Use definition TIMESYNCH_CONF_ENABLED
- How to enable it? And other definitions…

1. Create file project-conf.h

```
#ifndef __PROJECT_CONF_H__
#define __PROJECT_CONF_H__

#undef TIMESYNCH_CONF_ENABLED
#define TIMESYNCH_CONF_ENABLED 1

#endif /* __PROJECT_CONF_H__ */
```

2. Add a CFLAG to the Makefile

```
CFLAGS += -DPROJECT_CONF_H=\"project-conf.h\"
```

How it works

- Process “**Timesynch process**” is initialized in contiki-sky-main.c
- **Authority level** is set in contiki-sky-main.c

- Files core/net/rime/timesynch.{c,h}

Timesynch.h

- `void timesynch_init(void);`
  - Initialize the timesynch module
- `rtimer_clock_t timesynch_time(void);`
  - Get the current time-synchonized time
- `rtimer_clock_t timesynch_time_to_rtimer(rtimer_clock_t synched_time);`
  - Returns the local time corresponding to the synchronized time
- `rtimer_clock_t timesynch_rtimer_to_time(rtimer_clock_t rtimer_time);`
  - Returns the synchronized time corresponding to the local time
- `rtimer_clock_t timesynch_offset(void);`
  - Returns the current offset
- `int timesynch_authority_level(void);`
  - Returns the authority level
- `void timesynch_set_authority_level(int level);`
  - Set the authority level

Timesynch.c

- Timesynch Process
  - Opens a broadcast connection (channel 7)
  - Broadcasts timesynch_msg periodically

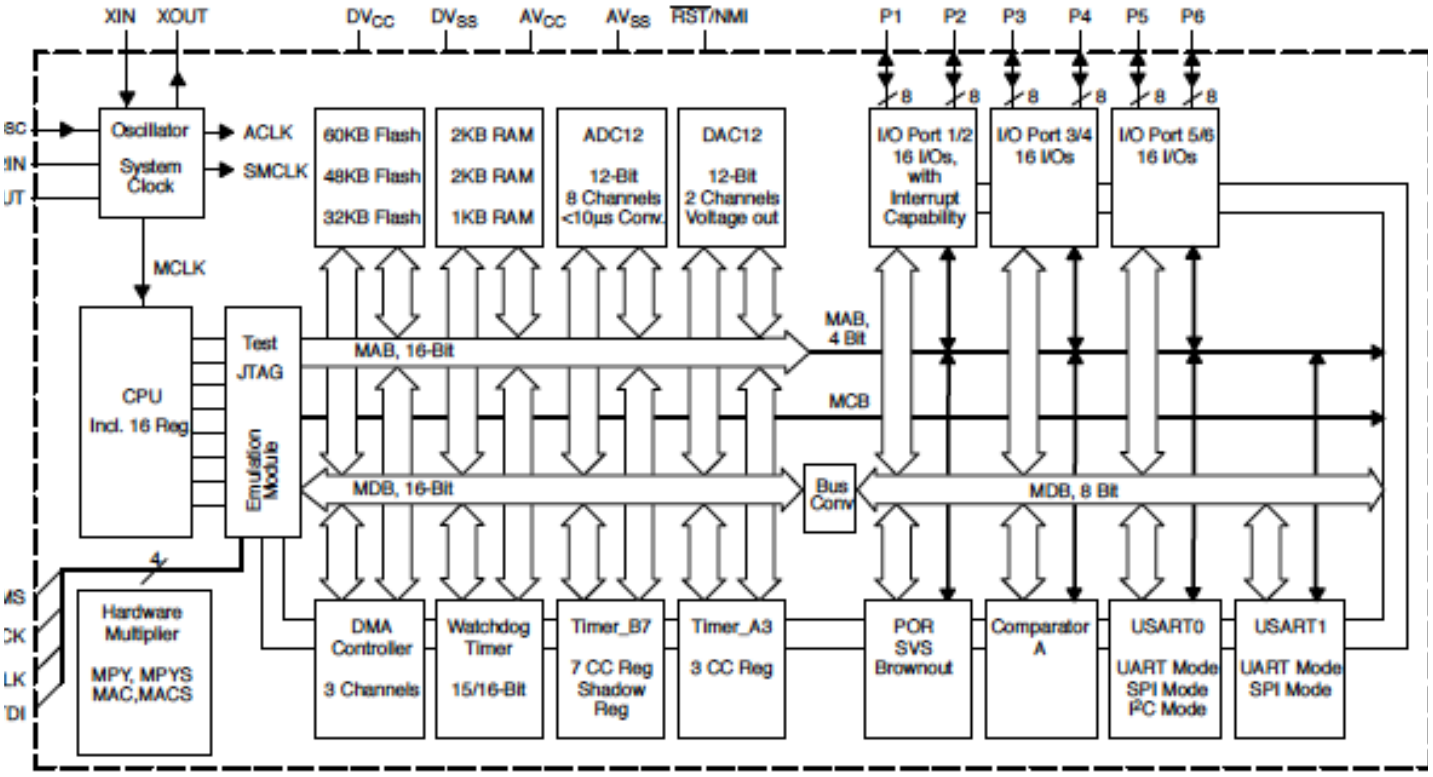
```
struct timesynch_msg {
  uint8_t authority_level;
  uint8_t dummy;
  uint16_t authority_offset;
  uint16_t clock_fine;
  clock_time_t clock_time;
  uint32_t seconds;
  /* We need some padding so that the radio has time to update the
     timestamp at the end of the packet, after the transmission has
     started. */
  uint8_t padding[16];

  /* The timestamp must be the last two bytes. */
  uint16_t timestamp;
};
```

Local Time (RTIMER)

- The RTIMER works based on a **32768 Hz** crystal
- **Why 32768 Hz ???**

- How is it implemented in hardware?
  - Tmote sky uses a MSP430F1611
  - It has two 16-bit Timers (TimerA and TimerB)
  - RTIMER uses TimerA
  - Architecture-specific files cpu/msp430/f1xxx/rtimer-arch.c
  - core/sys/rtimer.{c,h}


XIN
XOUT
DVCC
DVSS
AVCC
AVSS
RST/NMI
P1
P2
P3
P4
P5
P6
Oscillator
System Clock
ACLK
SMCLK
MCLK
60KB Flash
48KB Flash
32KB Flash
2KB RAM
2KB RAM
1KB RAM
ADC12
12-Bit
8 Channels
<10µs Conv.
DAC12
12-Bit
2 Channels
Voltage out
I/O Port 1/2
16 I/Os,
with
Interrupt
Capability
I/O Port 3/4
16 I/Os
I/O Port 5/6
16 I/Os
CPU
Incl. 16 Reg
Test
JTAG
Emulation
Module
MAB, 16-Bit
MAB,
4 Bit
MCB
MDB, 16-Bit
MDB, 8 Bit
Bus
Conv
Hardware
Multiplier
MPY, MPYS
MAC,MACS
DMA
Controller
3 Channels
Watchdog
Timer
15/16-Bit
Timer_B7
7 CC Reg
Shadow
Reg
Timer_A3
3 CC Reg
POR
SVS
Brownout
Comparator
A
USART0
UART Mode
SPI Mode
I²C Mode
USART1
UART Mode
SPI Mode

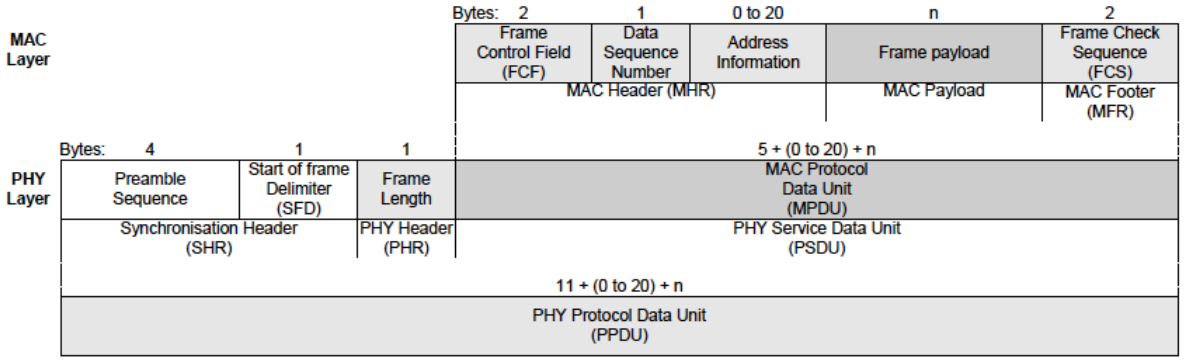
Local Time (RTIMER)

- So, local time runs forever, and every 2 seconds it rounds up to 0
- If you want to know the current time:
  - **RTIMER_NOW()**
  - It basically returns register **TAR** (Timer_A)

- The synchronization protocol keeps updating the offset of your local TIMER_A with the TIMER_A of the other nodes

How are packets time-stamped?

- We use the other timer (**TIMER_B**) to timestamp the packets
- Timer_B is set with the same clock (32K) and it initialized equal to Timer_A
- File cpu/msp430/cc2420-arch-sfd.c
- Timer_B executes interruption as CC2420 changes its **SFD (Start of Frame Delimiter)** pin
- Update variables **cc2420_sfd_start_time** and **cc2420_sfd_end_time**



"When the SFD pin goes high, this indicates that a start of frame delimiter has been detected."

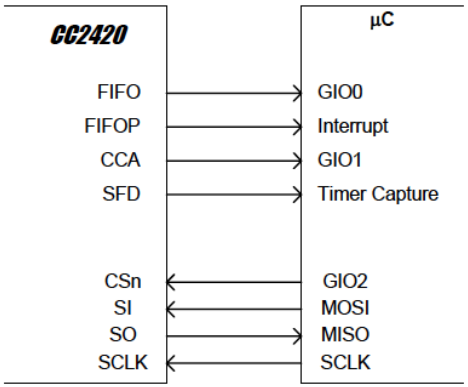
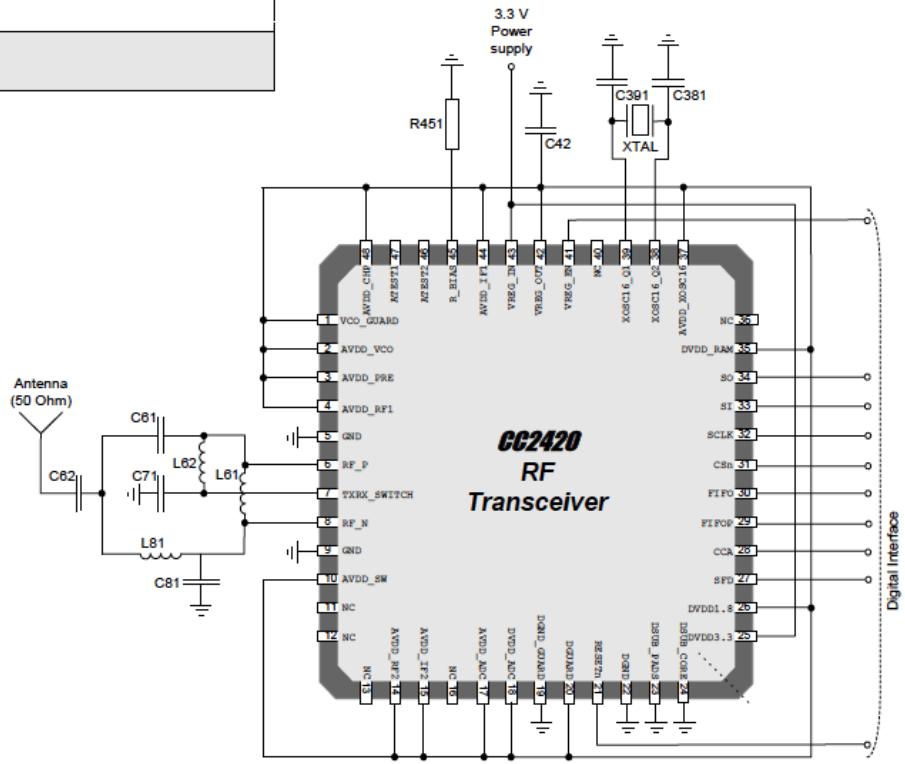


Figure 12. Microcontroller interface example



How are packets time-stamped?

- Time values are actually stamped into the packets at transmit/receive functions
- core/dev/cc2420.c
- Transmit: line 387
- Receive: line 623 and 645

Testing the synchronization

11.3 Timer_A Registers

The Timer_A registers are listed in Table 11–3:

Table 11–3. Timer_A Registers

Register	Short Form	Register Type	Address	Initial State
Timer_A control	TACTL	Read/write	0160h	Reset with POR
Timer_A counter	TAR	Read/write	0170h	Reset with POR
Timer_A capture/compare control 0	TACCTL0	Read/write	0162h	Reset with POR
Timer_A capture/compare 0	TACCR0	Read/write	0172h	Reset with POR
Timer_A capture/compare control 1	TACCTL1	Read/write	0164h	Reset with POR
Timer_A capture/compare 1	TACCR1	Read/write	0174h	Reset with POR
Timer_A capture/compare control 2	TACCTL2	Read/write	0166h	Reset with POR
Timer_A capture/compare 2	TACCR2	Read/write	0176h	Reset with POR
Timer_A interrupt vector	TAIV	Read only	012Eh	Reset with POR